MAPPING NATIVE PLANTS:

A MOBILE GIS APPLICATION FOR SHARING INDIGENOUS KNOWLEDGE IN SOUTHERN CALIFORNIA

by

Michael Alan Wahl

A Thesis Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
MASTER OF SCIENCE
(GEOGRAPHIC INFORMATION SCIENCE AND TECHNOLOGY)

December 2013

DEDICATION

This thesis is dedicated to Natalie, for inspiring me and supporting all that I do. Also, for my parents and grandparents, especially my grandmothers, Grace and Edith, for showing me how to love others and live a good life.

ACKNOWLEDGMENTS

I would like to thank Dr. Jennifer Swift for her immense help and guidance throughout this project. Major parts of this thesis would not have been possible without her. Also, thanks are in order to Dr. Jordan Hastings for his help with forming the ideas for this thesis in the beginning, and Dr. Flora Paganelli for her valuable advice in editing this paper. I also owe my gratitude to all of my friends and family for their endless support and love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	Application Programming Interface
ARC	Automatic Reference Counting
CPU	Central Processing Unit
GI	Geographic Information
GIS	Geographic Information System/Science
GIST	Geographic Information Science and Technology
GLONASS	Global Navigation Satellite System
GPS	Global Positioning System
GUI	Graphical User Interface
IDE	Integrated Development Environment
iOS	iPhone Operating System
IT	Information Technology
NIB	NeXT Interface Builder
OS	Operating System
PDF	Portable Document Format
PPGIS	Public Participatory Geographic Information System
REST	Representational State Transfer
SDE	Spatial Database Engine
SDK	Software Development Kit
SDM	Species Distribution Modeling
SMS	Self Management Support
SQL	Structured Query Language

SSI	Spatial Sciences Institute
UCGIS	University Consortium of Geographic Information Science
URL	Uniform Resource Locator
USC	University of Southern California
VGI	Volunteered Geographic Information
VM	Virtual Machine
VPN	Virtual Private Network
WGS	World Geodetic System
Wi-Fi	Wireless Fidelity

ABSTRACT

Native plants have been cultivated and utilized for thousands of years. From medicine and food to shelter and clothing, native plants have played an integral role in forming the indigenous religions, languages and cultures of Southern California. In recent years there has been a revival of indigenous culture with a focus on using native plants to teach about the languages and traditions of native people.

The “Mapping Native Plants” application developed as part of this thesis work is a novel iPhone application that puts self-learning tools into the hands of the general public. Using the theory of Volunteered Geographic Information (VGI) this application will give the power of the map to the people and not just to scientists and other specialists. Users can geotag the location of native plants, learn the plants’ native names, and read about how native plants have been used for thousands of years. Geotagging involves the user creating a geographic point in the map interface of the application that will represent the native plant of their choice.

An enterprise geodatabase was created using Esri ArcGIS 10.1 for Server to store the native plant data and allow multiple users to create and edit their own geotags of native plants. The geographic data embedded in the application is a product of Esri ArcGIS Online’s web map and feature services so that anyone can view the data and reference it in their own ArcMap projects.

Results from usability, performance and laboratory testing show that the application is understandable, easy to use in the field, effectively developed to run at optimal speeds on the iPhone, and that all functions and tools work without error. The application is more than just a map that will show the location of native plants, it is a tool

of self-education that will open a new perspective on the environment in the eyes of users and allow them to access a wealth of indigenous plant knowledge that has evolved and persisted for millennia.

CHAPTER ONE: INTRODUCTION

The term ‘plant blindness’ is used to describe the zoocentric and anthropocentric perceptions that the average person has when viewing their landscapes (Clary & Wandersee 2011). Because there are approximately 2,200 different endemic plant varieties in Southern California it would make sense that we take off these blinders and pay more attention to the plants in our environments (Rundel & Gustafson 2005). The focus of this thesis is on native plants for many reasons but particularly for the reason of sharing local indigenous knowledge about native plants and their many uses.

Endemic plants have been utilized by Native Americans in Southern California for thousands of years providing a cultural connection and knowledge of how local vegetation can be used in treating disease, improving diets, and, more recently, creating a legal defense for land rights issues (Bean & Saubel 1972; Timbrook 2007). In many Native American communities, plants are not just seen as a resource to use, but as an integral part of the native culture and worldview, something to be respected and appreciated rather than controlled (Bean 1972; Timbrook 2007). Many Native Americans use native plant foods as a type of homeopathy for treating diseases, particularly diabetes. Others use these plants to revive their culture and teach their children how their ancestors once lived and interacted in their environment (Roberts 2004; Hernandez 2005; Service 2011). The fact that Native Americans are more susceptible to diabetes and have the highest rates of this disease throughout the world makes the knowledge of how to locate and use native plants even more imperative (Elliot 1997; Rhoades 2000; Hernandez 2005).

Native plants also provide essential ecological services and function as indicators of ecological health, absent humans. In California, native grasses are used to study the environmental changes caused by global warming as well as regenerate the nitrogen in soils that have been depleted due to agricultural practices (Zavaleta, Shaw, Chiariello, Mooney, & Field 2003; Mahaney 2010). Koester (2008) argues that native plants are good for urban sustainability and cutting CO2 emissions through their use in living green roofs and mulch farms as well as for conserving water because they are more drought resistant than exotic plants. Native plants are also used to treat erosion in Big Sur (Parker 1999) and re-vegetate mountain roadsides throughout the United States (Landis 2005). Projects such as these have the significant potential to benefit from the shared knowledge of people using a smart phone application to map native plants.

Dunn (2007) claims that a participatory GIS gives political backing to local issues. It legitimizes the indigenous geographical knowledge and democratizes the citizen-users by giving them control and ownership of the data. The mapping of native plants should be open to everyone because a collective understanding of the plants will empower people to make better decisions when it comes to land use and the conservation of natural areas. Native American tribes can use this application to fight for land because it provides them with hard evidence of plant existence and creates a pathway for information exchange and education. And while some may argue that Native Americans might not use smart phone technology, Vos (2009) states that this is only an assumption because many reservations are not equipped with broadband Internet. Also, past studies show that as soon as the technology becomes available to reservations there has been a remarkable uptake, higher than the national norm, of smart phones and computers (Vos 2009).

1.1 Objective of the Mapping Native Plants App

The main purpose of the Mapping Native Plants application, hereafter termed “app”, is to create an app that has the potential to be used to help educate all people about native plants, their uses in the past, and their roles in human culture and the environment. Since the app has been developed using iPhone Operating System (iOS) programming the user group currently consists of those who own iPhones. The project area is all of Southern California so the list of native plants and the user group is targeted to people and plants that are within that geography. Southern California was chosen as the project area because it is home to many different biotic areas where thousands of native plants grow. Along with plant and ecological diversity there is also a spatially stable yet ethnically diverse population of native cultures in Southern California whose traditions, stories, and worldviews are deeply connected to native plants.

The user interface for the Mapping Native Plants app was tested on both people who have never used an iPhone application and people who have had experience with iPhones to see if they have any suggestions for ease of use. As a way to give credibility to the data that is collected the users can leave their name or initials, giving them the option to remain anonymous to other users or to take credit for their work. Users will also be able to rate and comment on each other’s contributions so that corrections can be made if the wrong plant or location is tagged.

The research objectives in the development of the Mapping Native Plants app were the following:

1) Design a mobile GIS application that allows users to select, add, modify and delete geotags representing native plants in the Southern California area.
2) Code and implement this mobile app in Objective C for the iPhone, using ArcGIS Server, Xcode, and Esri’s ArcGIS SDK for iOS.
3) A methodology for allowing mobile device and web browser users to share the data they have collected.

The research questions to be addressed were mainly related to the use of the app:

1) use by general public and mainly related to the friendliness and effectiveness of the developed Mapping Native Plants app’s user interface and functions;
2) use by researchers to access the data through the app’s web map source, shared with the public through ArcGIS Online, and integrate this spatial data into their own research.

CHAPTER TWO: LITERATURE REVIEW

The Mapping Native Plants app is first, and foremost, a product of volunteered geographic information. Although the technology and practice of VGI may be new, the concept of user-generated geographic data being volunteered through the Internet has been discussed for many years (Goodchild 2007). New VGI mobile applications being created and used by what are known as ‘neogeographers’ are changing the way GIS professionals view their profession and their conceptualizations of accuracy, scale and credibility (Goodchild 2008; Liu & Palen 2010; Kraak 2011). Whether these new advances in GIS are seen as good or bad, the citizen as a form of remote sensor can be highly motivated to contribute their local knowledge to large databases to help construct what former Vice President Al Gore called the “Digital Earth” in 1998 (Wilson & Fotheringham 2008).

In this chapter, there is a discussion on the development of VGI and its perception in the professional GIS community, the scientific studies that use, or could use, location-based data of native plants to fulfill research goals, and the Native American tribes that have used native plants as a part of their cultural revitalization.

2.1 Volunteered Geographic Information

The term Public Participatory GIS (PPGIS) was created to describe a new relationship of the user to computerized geographic information (GI) (Wilson & Fotheringham 2008). Before PPGIS, data was distributed in a top-down arrangement where experts created and disseminated data through servers and spatial data infrastructures and end users would download the data for a specific task (Budhathoki, Bruce & Nedovic-Budic 2008; Song & Sun 2009). Through PPGIS, this top-down framework was flipped around and the user became the creator and disseminator of digital mashups where two or more sources of GI are combined and shared with the general public (Liu & Palen 2010; Kraak 2011).

Examples of PPGIS are Wikimapia, OpenStreetMap, and Google Earth, where users can geotag placenames or pictures and make comments to create a new map that is enhanced by their local knowledge (Goodchild 2008). In this digital environment the cartographic expert’s knowledge is only needed in developing the software applications while the neogeographers take control of the decision on what they want to map, at what scale the geotag is placed, and who they will share it with, regardless whether the experts thought it was inaccurate, deceptive, or credible (Goodchild 2008; Liu & Palen 2010; Kraak 2011). While some may see this as the end of traditional GIS, Dunn (2007) has a more positive outlook in stating that PPGIS is an integration of local and expert knowledge and it opens the “narrowness” of GIS to “alternative visualizations of space, place and reality”.

In 2007, Goodchild coined the term “volunteered geographic information” as a branch of PPGIS where the users are doing more than just participating in geographic studies by clicking on a web map, unaware of how the data might be used. They are volunteering GI that they have collected on their own. This is made possible with the recent advancements in broadband Internet, grid computing, GPS technology, smart phones equipped with GPS, and graphics software that make it quicker and easier to view maps online (Goodchild 2007; Goodchild 2008; Gouveia & Fonseca 2008; Sui 2008).

In the past, a cartographer would depend on remote sensing from satellite imagery, aerial photos or hire a group of trained individuals to GPS a certain area in order to learn about the field environment. With VGI, in the chosen study area there are over 6 billion potential human sensors each with their own unique, local knowledge that can help researchers answer questions about the environment. What was once an expensive and timely endeavor tightly controlled by expert cartographers is now becoming cheap, quick and open to all who have access to the technology (Seeger 2008; Goodchild 2008).

2.1.1 Advantages and Disadvantages of VGI

To understand the impact of VGI it is necessary to understand the concepts behind the application. As ‘volunteers’ the users are representing a diverse public that is treated as a collection of individuals rather than as a single entity. ‘Empowerment’ means they are motivated to volunteer, disregarding whether they feel empowered or are actually empowered. With this, the concept of power is seen as the way the user controls their data and expresses their personal perceptions through it, not as a goal or as something that can be tangibly quantified. And ‘local knowledge’ is defined as the exposure of a wider audience to personal knowledge, although it can be difficult to assess its reliability. Overall, the concept of VGI is to allow people to investigate the places that are important to the individual. As a result, this can help people form policy decisions or inform others to influence their own policy decisions (Tulloch 2008).

Gouveia (2008) lists the advantages and disadvantages of VGI to weigh the ultimate benefits of using it to create a framework for environmental modeling. On the data side, VGI is beneficial to the support of early warning systems when catastrophe or emergencies occur. The downside is a question of data credibility, nonexistent metadata, lack of trained professionals collecting the data, and the difficulty in motivating unpaid volunteers to commit fully to a project. Logistically, VGI reduces the cost of data collection and maintenance and allows for more time and a wider area to be covered. The reverse is that it can be difficult to train the public on the proper data collection procedures or provide them with the appropriate training materials. Socially, VGI has allowed the public to become more informed and educated concerning environmental issues by promoting a consensus among the volunteers instead of the “traditional ‘us v.

them' approach" (p.187). On the other hand, VGI may be affected by citizen participation because people could harbor ill will toward scientific research and use what is known as the NIMBY approach: Not In My Back Yard. The challenge for VGI is to motivate people into volunteering and convince the experts that VGI is credible, valid, and potentially more accurate than the data collected by experts.

2.1.2 VGI Credibility and Local, Indigenous Knowledge vs. Expert Knowledge

How do we know that the data volunteers are sharing is valid? Will using such data destroy the credibility of the researcher or their project? And, should we assume the altruistic notion that volunteers will not corrupt any of the data they are collecting? Goodchild (2008) claims that VGI is not inaccurate, rather he thinks that people with local knowledge will be quicker to make corrections to any errors they see because they are the experts of their locality. In fact, locals often carry the perception that so-called "expert" knowledge is incomplete or inaccurate (Dunn 2007). And research has shown that local knowledge oftentimes exceeds expert knowledge at the local level (Flanagin & Metzger 2008). VGI allows us to alter our perception of the expert being the authority on data credibility and realize that the local can, and should, be the authority on local subjects.

While some may fear that VGI will lead to a combination of "sloppy GIS practices with sloppy social science" there are others who believe it will create an "engagement between two geographies which currently remain in two separate worlds" (Dunn 2007, p.627 & 630). The expert and the local are two different examples of how geographic knowledge is processed in the human brain and this dichotomy should be viewed as an exploration of the social side of GIS where greater communication is facilitating the maturation of GIS as a true science.

Goodchild (2007, 2008) claims that data accuracy is determined by the local need in VGI applications while Seeger (2008) takes this concept further and states that GI is valid as long as the collector of the data is a member of the community in which it was collected. Flanagin & Metzger (2008) propose that VGI applications utilize user ratings so the volunteer community can correct itself when invalid data is collected. By employing user ratings he claims that eventually, as the application is used more often, the more credible the data will become.

2.1.3 Citizens as Sensors: The Motivations and Consequences of Volunteering

Another question is how and why are people motivated to volunteer geographic information? Some think VGI may be a ‘flash in the pan’ or a GIS fad until the allure of this new technology dies out and people are less intrigued by creating their own maps (Goodchild 2008). Are people only interested in self-promotion, just wanting to make their own mark on a map, even if it is anonymous? Does it make them feel more connected to society by partaking in VGI? And are they sacrificing their personal privacy by volunteering? These are questions that researchers have considered since the creation of VGI and there does not seem to be any one answer.

More importantly, will VGI democratize those who are marginalized or excluded from decision-making processes? Or will it subject them to further domination by the majority? This brings up the aspect of the digital divide. How will people with no access to broadband Internet or smart phone technology be able to share in the democratization that VGI can provide? While there are still many people who have no Internet access, it can be said that VGI is helpful to building "geo-information into the local knowledge process" (Dunn 2007, p.624), it provides excellent data in emergency response, is a source for viewing people's understanding of their environment telling about 'life at the local level', and is widely available to anyone with the means to access it (Goodchild 2007).

2.1.4 The Digital Earth of Al Gore

It is clear that the goal of GIS and VGI is to connect people, data, software and hardware into making models of our environment that help us make better decisions and educate us about the world we live in. In 1998, Vice-President Al Gore presented the concept of a Digital Earth to motivate scientists into creating a digital, full-scale model of our planet that has connections to every data network known to man (Wilson & Fotheringham 2008). While this goal may be far from reality at the moment, we have stepped up the pace with the advent of VGI. The role of data collector is no longer in the hands of a few experts; rather it is under the authority of billions of potential human sensors all with unique perceptions and worldviews. The future of GIS is dependent on the study of VGI applications and how we can make better connections between each human sensor and use their local knowledge for the benefit of all people and the planet. The Mapping Native Plants application is just the microcosm of this larger network and has the ability to connect people at the local level, educate them about their environment, and influence the way people make decisions about personal health, land use, development, and conservation.

2.2 Plant Geography

In the McGraw-Hill Concise Encyclopedia of Science and Technology (Box 2006) the term ‘plant geography’ is defined as “the study of the spatial distributions of plants and vegetation and of the environmental relationships which may influence these distributions.” The Earth is divided into specific floristic regions and this study takes place within the ‘California Floristic Province’ where there are approximately 2,200 endemic plants out of 5,300 vascular plants total (Rundel & Gustafson 2005). In Southern California there are many different microclimates and bioregions ranging from the alpine tundra at the top of Mt. San Jacinto (10,834 ft. elevation) to the arid deserts that go below sea level. Many plant geography studies are concerned with invasive species since they are more widespread and often damage the ecosystems they invade. Nevertheless researchers are beginning to acknowledge that more study needs to be done on endemic populations so they can serve as donor specimens for replacing unwanted invasive plants (Stohlgren, et al. 2011).

2.2.1 Plant Geography Studies That Would Benefit From Mapping Native Plants

Species distribution modeling (SDM) is important to predicting the occurrence of certain plant populations and assessing the impact of plants (invasive or endemic) in a given area. This information can be used to design plant reserves and plan for conservation, manage plant resources, restore ecological zones, and determine the effects of global warming on ecosystems and their biodiversity (Franklin 2009). By using VGI to map native plants the data becomes cheaper to collect and more accurate because local volunteers know the terrain better and will be motivated to provide accurate data if they know it will help their own environments.

There are many scientific studies that could benefit from knowing the location of endemic plants. Teachers can teach students about their environment (Batcheler), researchers can track the distribution of native plants that have been recently restored in places where they once disappeared (Serrill 2006), and medical professionals can study and locate various native plants that may help them create new medicines to treat diseases (Dutt, Kumar, & Garg 2009). Global climate change studies are also increasing and the knowledge of native plant locations is beneficial to marking the health of an ecosystem (Zavaleta, Shaw, Chiariello, Mooney, & Field 2003; Seo, Thorne, Hannah, & Thuiller 2009).

Some researchers need native plant data to decide which plants should be considered rare or endangered (Crain & White 2011), while others can be assisted in mapping native plants that have traditionally been plotted using remote sensing (Huang & Geiger 2008). Gelbard (2003) studied how roads affect the transport of invasive species in an area. With the knowledge of native plant locations the case could be strengthened to claim that roads are damaging to the ecosystem and that some areas need to be protected from development based on the occurrence of rare, endemic species that have been plotted there. While the Mapping Native Plant app has a focus on educating people about their native plant uses in a cultural context it also has great potential for assisting many different studies that are concerned with plant geography.

2.2.2 WHO Traditional Medicine Strategy, 2002-2005

The World Health Organization developed a strategy in 2002 with a goal of providing traditional medicine to people from different cultures all over the world (Organization 2002). The issue is that many people do not trust modern ‘Western’ medicine so it is important for health care providers to deliver traditional medicines, also known as ‘alternative medicine.’ One major concern of health care providers is the accessibility and affordability of the ingredients, namely native plants, used in traditional medicine. It is envisioned that an app like Mapping Native Plants could encourage people who own or have legal access to the land they are mapping to go out and map the plants they need. This will cut down on the cost of making the medicine, because it will be relatively free unless they need a permit or special permission to harvest the plant, as well as the cost of locating the plants. Not all native plants are easily attainable, but in many cases the Mapping Native Plants app would be beneficial simply in recording known locations.

2.3 Southern California Tribes: Plants, Language and Revitalization

There are many different Native American tribes in the Southern California region that have survived for millennia because of their relationship with native plants. Ethnobotanical studies have been conducted for over 100 years and the tribal elders have worked with anthropologists, linguists and archaeologists to try and classify the native plants that have been used over the centuries. Native plants are used for food, clothing, housing, decoration, medicine and ritual and are an integral connection to the culture and language of the tribes (Barrows 1900; Bean 1972; Bean & Saubel 1972; Timbrook 2007; Largo, McCarthy, & Roper 2009).

Unfortunately, as contact with Europeans became more consistent and aggressive the culture began to change. The people were forced onto reservations and given food high in processed flour, sugar, lard and other unhealthy additives. When their connection with native foodstuffs was lost, the connection with their rituals, medicine, language, and culture was lost, as well. Thankfully there were a few in many of these tribes who remembered the traditional ways, or remembered what their ancestors told them, and were able to bring back some of the traditional knowledge and revive their culture through education and native plant food (Bean & Saubel 1972). The traditional names and details for native plants in the app are a product of those memories and will act as a tool to perpetuate Native American cultural learning.

2.3.1 Diabetes in Native American Communities

Type-two diabetes is steadily rising in Native American communities because of the Western diet (Jernigan & Lorig 2010). Doctors have been studying the genetics, the culture, the history and environments of Native Americans to understand why they are more susceptible to diabetes than any other population on the planet (Spero 2006). Recent studies have shown that diabetes is not a purely genetic, cultural or environmental disease, but is rather a social disease brought on by unhealthy diets (Spero 2006). Many tribes are seeking education about traditional food sources to strengthen their diets and become healthier (Hernandez 2005). In fact, the Tohono O’odham in Arizona, who have the highest prevalence of diabetes in the world, have turned to farming traditional foods (teppary beans, cholla, corn, squash, etc.) to fight against diabetes (Bazell & Carroll 2011). Native Americans throughout the country realize that in order to be healthier they need to eat traditional foods again.

“Social diseases need social approaches” is a quote by Spero (2006, p.22) stating the fact that social support is needed to combat diabetes. Self Management Support (SMS) is used to describe how ninety percent of diabetes care is self-care. People can go to doctors for medicine, however they will only defeat the disease once they have changed their own lifestyle and diets. But there are two major barriers in SMS: getting people motivated to care for themselves, and finding access to the foods that will make them healthy (Spero 2006).

This is where the Mapping Native Plants app would be helpful. Many people find great pride in their own culture and being a part of their community. Unfortunately, their pride also keeps them from doing what others, such as healthcare professionals, tell them to do (Griffin, Gihiland, Perez, Upson, & Carter 2000). By using the app they can forego having someone else tell them what to do and educate themselves about native plants and find strength in numbers by participating with others in locating the plants. This app is not a cure for diabetes, but it would serve as a great incentive to people who want to become more active and change their dietary choices through self-education.

2.3.2 Native Americans and Geotechnology

About sixty-nine percent of Native Americans have access to the Internet, compared to seventy-three percent for Whites, sixty-one percent for Blacks, and seventy-six percent for non-Spanish speaking Hispanics (Jernigan & Lorig 2010). Along with this statistic it is worth noting that internet-based SMS's have been "accepted and effective approaches for changing behaviors and improving health outcomes…" in Native American communities (Jernigan & Lorig 2010, p. 262). According to a doctoral dissertation by Eades (2010) geo-technology is quickly being adapted by Native Americans, especially iPhones, in the younger generation. Sharing information is dire to the survival of Native American culture and thus there is a potential for VGI and geotechnology to enhance communication between the elders and youth so they can pass on knowledge in novel and different ways.

2.4 Similar Smartphone Apps

At the time this app was created, there were only a few smart phone applications that serve a similar purpose as the Mapping Native Plants app, yet there are major differences that set this application apart from the others. Some apps focus solely on invasive species, some focus on just the educational aspect and do not allow users to collect geographic information, and most are set in an entirely different geographic area. The main thing that sets the Mapping Native Plants app apart from the others is its cultural component. The Mapping Native Plants app has a primary focus on native plants that have been and are used by indigenous Southern California cultures.

The Calflora "Observer" app is probably the most popular mapping app for plants in California (Calflora, http://www.calflora.org/phone/ (last accessed 30 October 2013)). It allows users to collect geographic data on both native and non-native plant species in all of California. The downside is that it is limited to users of the Android platform although it claims that it can be used on other platforms. The Observer is an efficient app and is very useful for tracking native and non-native species, but it does not educate the people about the cultural uses of certain native plants and does not allow users to edit or update others' plant features and rate them based on their accuracy as the Mapping Native Plants app does.

"Leafsnap" is another highly innovative app that lets the user take a picture of a tree leaf and identify what type of tree it is (Smithsonian Institute, http://www.si.edu/apps/leafsnap (last accessed 30 October 2013)). The main difference from the Mapping Native Plants app is that its intended use is limited to identifying tree species, and no other type of plants may be mapped. The second is that it is geographically limited to the Northeastern region of the United States. It runs on both iPhone and iPad but it does not allow them to collect geographic data in a database. Thus it is closer to being a tree guide as opposed to a mapping application.

"What's Invasive!" is a popular app in California that can be used on both iPhone and Android platforms but is focused on invasive species of plants and animals (The Center for Embedded Network Sensing, http://www.whatsinvasive.com (last accessed 30 October 2013)). The allowable study areas are limited to park boundaries and cannot be used outside an area that does not qualify as a park. This app also does not allow cultural element information gathering, though it does allow users to collect geographic information on the invasive plants/animals.

The "Audubon Wildflowers California" app is a field guide for its users. However, this app does not let users collect or share geographic data about wildflowers (National Audubon Society, http://www.audubonguides.com/field-guides/california.html (last accessed 30 October 2013)). This app has a focus solely on native wildflowers and includes no other types of plants. There is no cultural element functionality in this app, yet it does run on multiple smart phone platforms.

The Mapping Native Plants app is unique because it includes the vast study area of Southern California, provides information on native plants that have been used within a cultural context for thousands of years, and allows the user to collect and share spatial data with all other users. At the time this study was conducted, there was no other smart phone app (iPhone or Android) that has the same capabilities as the Mapping Native Plants app. Users of the app will be able to share information with each other and help each other learn about native plants endemic to Southern California and help carry on the ancient knowledge of native plants for generations to come. Within the app there is a window where users can find more information about the books and articles used to supply the data for the native plant list, and about the museums and botanical gardens they can visit to further their education. Other apps promote only the institutions that work with the app authors or creators, but the Mapping Native Plants app promotes all institutions in the Southern California region that offer insight and education on native cultures in this region and the native plants they use.

CHAPTER THREE: DATA AND DATABASE STRUCTURE

To have multiple users create and edit native plant features in the Mapping Native Plants application it is necessary to implement an enterprise geodatabase, using ArcCatalog and SQL Server, to publish a feature service. Most importantly, the database needs to be designed in a way that will assist the workflow of field data collection .The relationships of entities and their attributes in the geodatabase also need to be carefully considered so people can use the app without any confusion or misunderstandings. Since there is only one feature class in the geodatabase a simplistic data model, without any complex relationships or topological rules, was used. In this chapter the geodatabase structure is described in detail along with the native plant feature class maintained within.

3.1 Enterprise Geodatabase Structure

In designing an enterprise geodatabase for a mobile application there are several questions that need to be considered. How will the geodatabase best support the workflow of collecting data in the field? How will users describe the data that is being collected? Which layers will be edited and which will not? How will updates be synchronized from the field to the database? How will those updates be managed? And will versioning be necessary? Once these questions are answered a data model for the geodatabase can be produced to develop the various entities, attributes and relationships that will aid in real world data collection.

The Mapping Native Plants geodatabase is populated with only one feature class that has the sole purpose of collecting data about native plants. There are no other data layers needed since ArcGIS Online will provide the base maps used in the app. The geodatabase is designed to instill data integrity by using domains and feature templates linked to the attribute fields that will be used by the editing window in the app. The users will have different levels of understanding of native plants so the domains need to use simple terms and withhold any complex language. This will enable users to collect data more speedily and give them a full appreciation for the types of attributes they are collecting.

There are three domains in the geodatabase that will ensure the data integrity for certain attributes when the user is creating or editing a feature. Each domain uses coded values and the long integer data type in their fields. The first domain is called “Plant Name” and has coded values for all two hundred and sixty-seven plant types in the native plant list. In the map document the layer rendering symbology for the feature class is sorted by the names listed in the plant name domain. This then creates a feature template that is used in the app code’s template picker method. The method creates a pop-up that allows the user to choose what type of plant they want to geocode based on the plant names listed in the plant name domain.

The second domain is called “Rating” and contains four different coded values: Excellent, Good, Fair and Poor. This domain is linked to a field that will allow the user to rate the quality of other users contributions to the app. This will bring integrity to the native plant data by instilling a uniform rating system that doesn’t have a million different descriptions all with similar meanings. If the user really likes the geotag, they can rate it excellent, if not, they can rate it poor.

The third domain is called “Distribution.” It has five different coded values used to describe how many plants the geotag represents: one plant, two to ten plants, eleven to twenty-five plants, twenty-six to fifty plants, or more than fifty plants. This eliminates the need for users to create a geotag for each plant in that area. It would take a much longer time to geotag each and every plant than it would to make one geotag that represents them all. The user can also take a picture and add it to the attachments of the feature as evidence for how many plants are in that area. This domain is important because different people might have different ways of describing the plant distribution in an area. So, to stray from having people record “a lot” or “a few” or “some scattered about” users can choose the exact range they see and the data will maintain its integrity.

The geodatabase is registered as versioned so users can post edits to the default version immediately when they hit the “done” button in the editing window. These updates will be automatically seen in the map window of the app. A database administrator can manage the updates and revert the database to an older version if any large editing mistakes happen, if desired.

The geodatabase user role is created using an authenticated database login and is accessed by the users of the app to give them the appropriate permissions to create, edit and delete feature data residing within the geodatabase. It also provides users with an application server connection to the database to quicken the editing process. This means that when a user turns on the application they will automatically be recognized as a database user, granted all editing permissions, and allowed to access the database whenever they need to sync their edits to the server, as long as they have network connectivity.

All of these features of the geodatabase work together to ensure the ease of use for the app user, maintain the data integrity, and quicken the editing process for collecting data in the field. The user does not have to log in to the geodatabase or do any special posting and reconciling procedures to sync the data they collected to the server. The users are utilizing GIS, performing complex geodatabase tasks, and contributing to the local knowledge of native plants without being aware of the major processes it takes to execute these functions. In Figure 1, an entity-relationship diagram shows the various database properties and how these all interact with each other in the field data collection workflow.

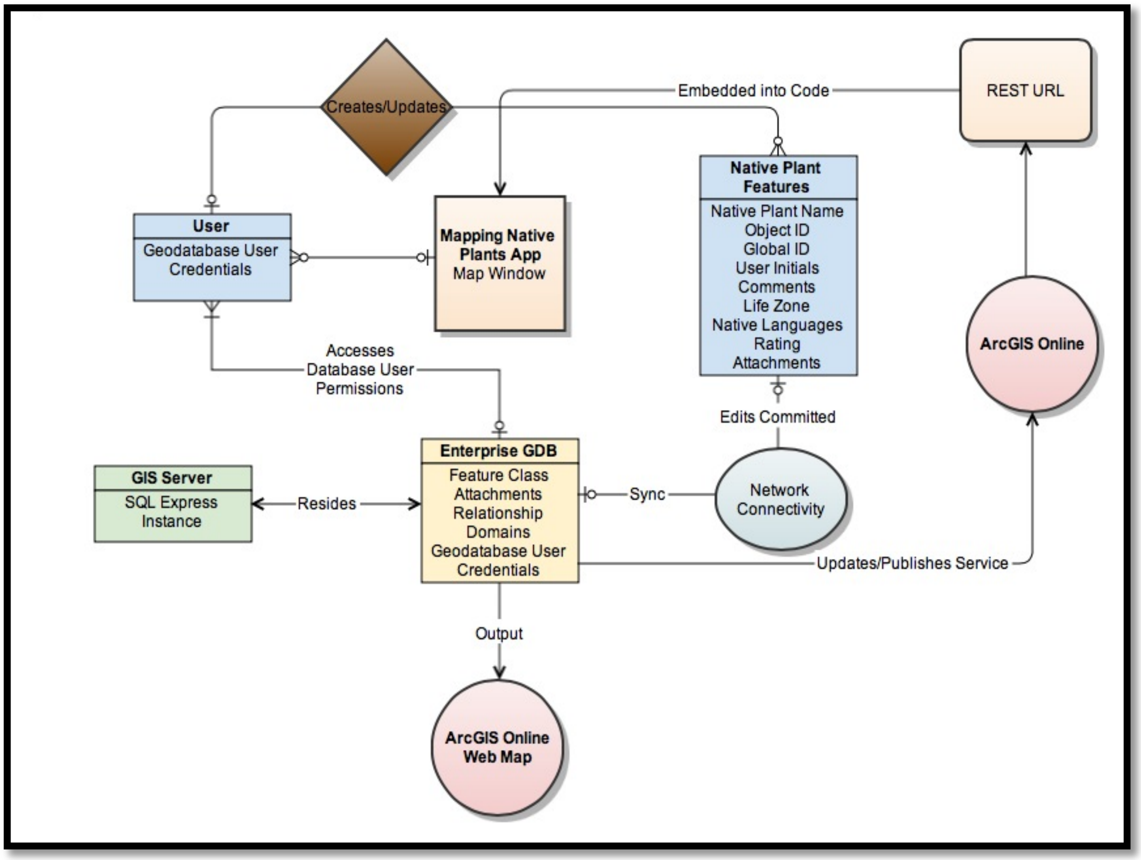


Figure 1: Native plant geodatabase and field data collection workflow

The workflow begins with the enterprise geodatabase residing within the GIS Server and referencing the SQL Express instance for online web editing. Within the geodatabase are the native plants feature class, the attachments, the relationship class, the domains and the geodatabase user credentials. The feature service is published to ArcGIS Online and a REST URL is created for that service. The REST URL is embedded in the Mapping Native Plants app code so the feature can be accessed by the user through the map window in the app. The users have a many-to-one relationship with the app so that multiple users can access the app at the same time. The user is automatically given the geodatabase user credentials when they open the app and therefore they have a many-to-one relationship with the geodatabase as well because more than one user can access the database at the same time.

The user also has a one-to-many relationship with the native plant features because they can edit as many features as they want. When users create or update a native plant feature they can edit that feature's attributes (Native Plant Name, User Initials, Comments, Rating) and add Attachments (photos or video). Once users commit their edits, based on whether they have network connectivity or not, they will sync their newly created or edited data with the geodatabase server. This then updates the feature service published on ArcGIS Online and the new feature or new edits are displayed in the app's map window. The output of this workflow is the web map that references the feature service and is also published and shared in ArcGIS Online.

This workflow shows that the geodatabase is designed so that the only responsibilities the user has is updating or creating features without having to perform any type of data validation or geodatabase maintenance. The geodatabase is configured to let the user geotag as many features as they can and sync them to the server as quickly as possible without hassle as long as they have network connectivity.

3.2 Data Preparation for Publishing the Feature Service

For the app to provide a map window containing an editable data layer that users can create and edit the features of there needs to be a feature service published from the feature class created in the enterprise geodatabase. The attribute fields, scale, projection, geometry, data types and extents all need to be considered before publishing the layer as a feature service. And after it is published there needs to be consideration of how it will be configured to work operationally within the app. For this project there were several different feature services published and tested before the right combination of feature class attributes, database settings, and feature service configurations worked.

The first task is to consider what the data is supposed to represent in the real world and what kind of attributes are important for describing the feature, for the purpose of the project, for the understanding of the users, and for the use of future studies. The native plants feature class is a point geometry layer representing native plant objects that exist in the real world. Since the app will be used by many different kinds of people that have varying experiences with mobile apps it is necessary to develop attributes that are straightforward and use language that is easy to understand while also fulfilling the purpose of the project. The types of attributes that would be the most important and easily understood are: the plant names, the location of the features, how many plants are represented by those features, and the day and time the features were geotagged. Other attributes that contribute to verifying the validity of the geotags are the user initials, user comments, and user ratings.

To keep each record unique in an enterprise geodatabase that can be replicated and versioned, a global id field was created. But the only attribute fields the user will interact with in the app are the plant name, user initials, comments, and the date, ratings and distribution fields. All other attributes can be viewed in the attribute tables of the feature service shared on ArcGIS Online.

Another requirement for publishing the feature service is that an attachment table is made to hold any photos or videos that are associated with the feature. An attachment table is created with the same name as the feature class and related to the features through a primary key: the ObjectID field. This relationship has a one-to-many relationship where many attachments can be created for one record. The other attributes of the attachment table are the attachment ID, the related object id (foreign key), content type, attachment name, data size, data, and Global IDs. All of these fields are automatically updated in the attribute table once the attachment is synced to the server.

The entity-relationship diagram shown in Figure 2 outlines the attributes in the feature class and the attachment tables that are related within the geodatabase.

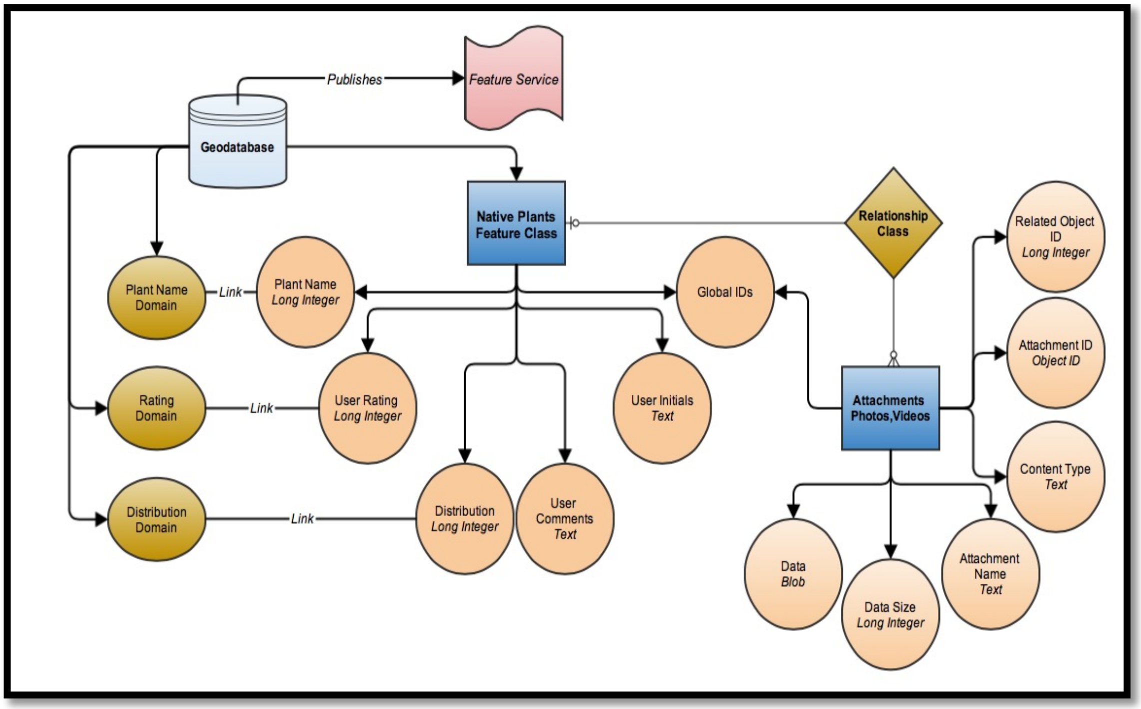


Figure 2: Detailed native plant geodatabase diagram

The geodatabase contains the three domains that are linked to their associated attribute fields in the feature class. The native plants feature class contains the five fields that the user can update and one field that is used to relate it to the attachments table (global ids). The geodatabase also has a relationship class that is used to define the rules of the relationship between the attachments and the feature class. The attachments table has six attribute fields that are unseen to the user and are automatically updated whenever an attachment is added to a feature. All of these attributes, tables, domains and relationship classes are published as part of the feature service that is embedded in the app's map window.

Once these characteristics of the feature class have been created there are several other considerations that need to be addressed before it can be published as a feature service. The scale of the feature class is set to the extents of the Southern California area ranging from the Mexican border in the south to Santa Barbara in the north and as far east as the border with Arizona. For the feature service to behave well with the ArcGIS Online basemaps in the map window the coordinate system is set to WGS 1984 Web Mercator (auxiliary sphere). Once the layer extents and coordinate system have been updated there needs to be several test features created in the map document to check the performance of the domains and make sure the data types of the fields are all set properly.

When the feature service is being published, ArcGIS allows the administrator to validate the settings and configure the service for whatever functions it will perform on the web. Validation will ensure that the extents are set properly, that the data is being referenced in the map correctly and that all geodatabase settings are appropriate for creating the feature service (versioning, global ids, etc.). The configuration of the feature service is very important because it is where crucial metadata is entered about the service and where the feature’s capabilities and settings can be modified.

Since the feature service is being used in a mobile app as well as a web map the Feature Access capabilities should be enabled. Feature access capabilities allow the users to update the geometry of native plant features and perform create, delete, query and update operations in the map window of the app. All other settings and configurations are set to default but it is important that they are understood.

A major concern for future studies is metadata and how it can be maintained and accessed on a regular basis. Currently, metadata can be viewed in the ArcGIS Online feature service and web map links where there is a full description of all the data, templates, features, and attributes. Metadata is also stored within the geodatabase and can be updated by the administrator on a frequent basis where it can be exported to researchers upon request.

CHAPTER FOUR: TECHNOLOGY AND PROGRAMMING

In the beginning of this project a major decision was made on what type of mobile platform the app should be developed for. The top two choices were either Apple’s iPhone or smartphones using the Android operating system (OS). There are several reasons why it was decided to develop with Apple’s iPhone. An advantage of the iOS platform is that it is a closed-system model where developers need only create apps for one type of smartphone. Whereas the Android OS uses the Java computing language that is an open-system model that can be more challenging to develop if one wants to support all of the various, multiple smartphones it is used with. The open-system platform is a plus for the user because the phones that use it are less expensive and sometimes more user friendly, but open-systems can be a hardship for the novice developer in terms of availability of technical support. Although there are pluses and minuses for each operating system, when the project was started in late 2011 the current research had shown a trend towards users preferring the iOS platform (Ryan 2011). Figures 3 and 4 show that users not only preferred Apple's iPhone to other smartphones in 2011, but they were also more satisfied.

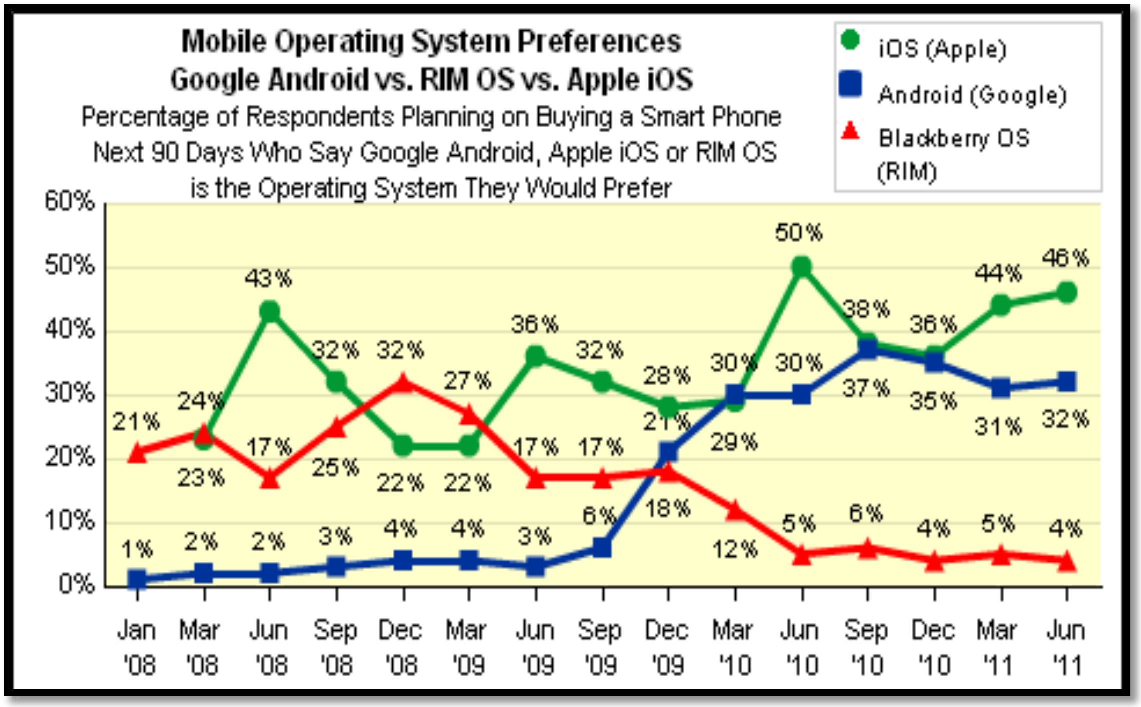


Figure 3: Operating system preferences (Ryan 2011)

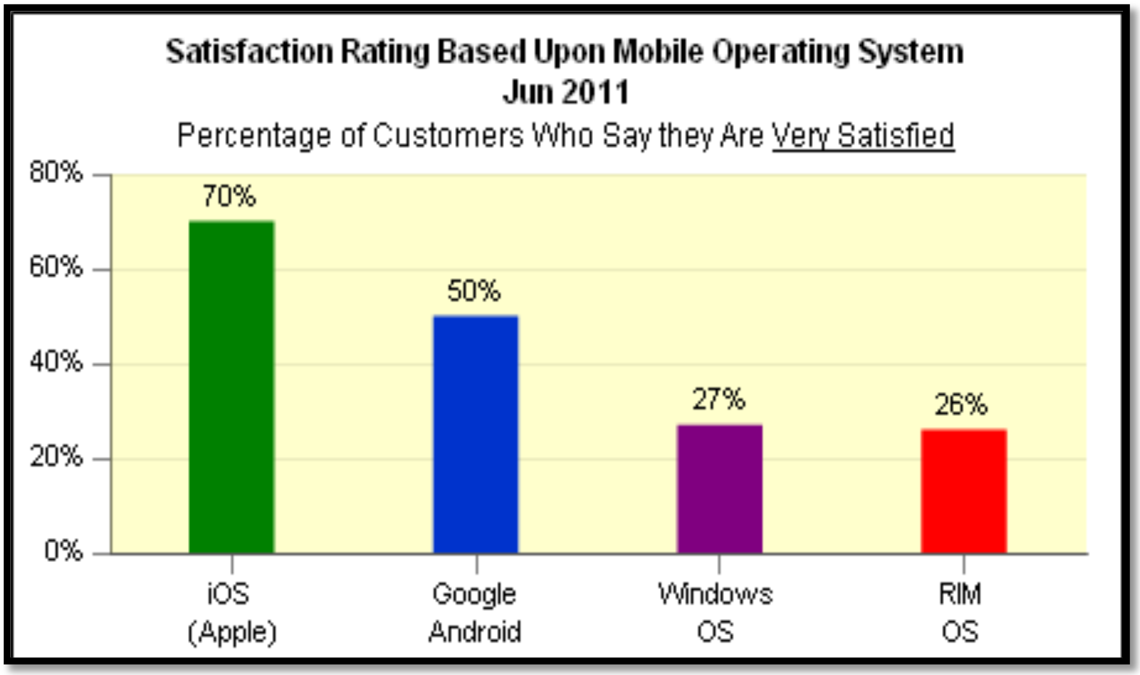


Figure 4: User satisfaction for operating systems (Ryan 2011)

Recent statistics in 2013 show that Android has gained more popularity in the market than the iPhone because of the many different types of phones it can be used with (Manjoo 2013). But some research indicates that Android users are slowly turning back to the iPhone claiming that the phones the Android OS is being used on are lacking the quality that Apple provides (Manjoo 2013; Woollaston 2013). And while smartphones that use the Android OS are outcompeting the rest of the market in sales there is evidence that the iOS apps are still being used more than Android apps (Lunden 2013). Android OS had made it possible to create less expensive phones but for the time being may have sacrificed the quality that Apple's iPhone has to offer. There are many ongoing debates on this subject but for this thesis the iPhone was chosen primarily because of its substantial support for developers, its popularity among younger users, its dominance in the market, and its user satisfaction (Hassell 2010; Patterson 2013; Reed 2013).

4.1 iOS Xcode (Version 4.6.3)

Apple's development software, Xcode, was used to develop the application. The programming language is Objective-C, which is structured similarly to the object-oriented themes in GIS. To install Xcode one also needs to install the iOS Software Development Kit (SDK). Together, Xcode and the iOS SDK allow developers to use the Xcode Integrated Development Environment (IDE), the iOS Simulator, and access the application testing software called Instruments. These products provide the interface to develop the app for the iPhone, simulate how it will run on an actual smartphone, and test the code for memory leaks, performance issues, and how much energy it may consume which is detailed in Chapter 5. Xcode also includes most of the developer documentation in its help files and provides developers with an Interface Builder to construct the graphical user interface (GUI) of the application. Apple is consistently improving and updating Xcode and has plans to release Xcode version 5 very soon (Apple Insider 2013). The most recent version of Xcode has been updated to develop apps for the iPhone 5 and earlier versions.

4.2 Methodology of Application Development

Before any coding or user interface construction can begin it is important to visualize what the app will look like and how it will function in the hands of users. For the Mapping Native Plants app three main views were considered: one, a list of all the native plants that have importance to the indigenous cultures of Southern California, two, the details of those plants describing their Linnaean names, Native American names and uses, their natural habitats or life zones, and any other information about them, and three, a map window where the geographic data can be viewed and features can be added, deleted, and modified by users (Barrows 1900; Bean 1972; Bean & Saubel 1972; Timbrook 2007; Largo, McCarthy, & Roper 2009). Beginning with those three elements a conceptual user interface was designed and a workflow diagram was created to describe the various functions that would be developed for the app.

To enhance the experience of users as they learn about native plants and geotag native plant locations, there needed to be more responsibility taken by the user for each plant located. One idea was to have the user login so they can take pride in the types of plants they create and allow other users to see what kind of plants they have geotagged. Another idea was to let the users create a favorites list of all the plants they like best to geotag. Users can refer to the list whenever they want to review the native plant name is once more or learn additional information without having to scroll through the plant list. It would also be beneficial to the user if the plant detail window had a connection to the plant's Wikipedia page so users can scroll through even more information than could be fit into the app window programmatically. Lastly, including an "About" window designed so the user could find specific information about the app and view the disclaimer was considered.

There were three phases in developing the functions of the app. In the first phase, a proposed workflow was created to layout the basic functional needs of the app and visualize how it will work. The second phase was to develop the app and write the code to execute the functions based on the proposed workflow. Once the app was developed, the third phase was to revise the workflow to reflect the actual functions of the app. These three phases are described in detail in the following sections.

4.3 Mapping Native Plants Proposed Workflow Diagram

The first workflow diagram, seen in Figure 3, shows the initial design on which the app's development was based on. There are four main windows with multiple functions and pop-ups that allow users to learn about native plants, geotag their locations, edit other users' contributions, and upload the features they have collected to the enterprise geodatabase. This section describes how the proposed app should have conceptually worked in the hands of the users.

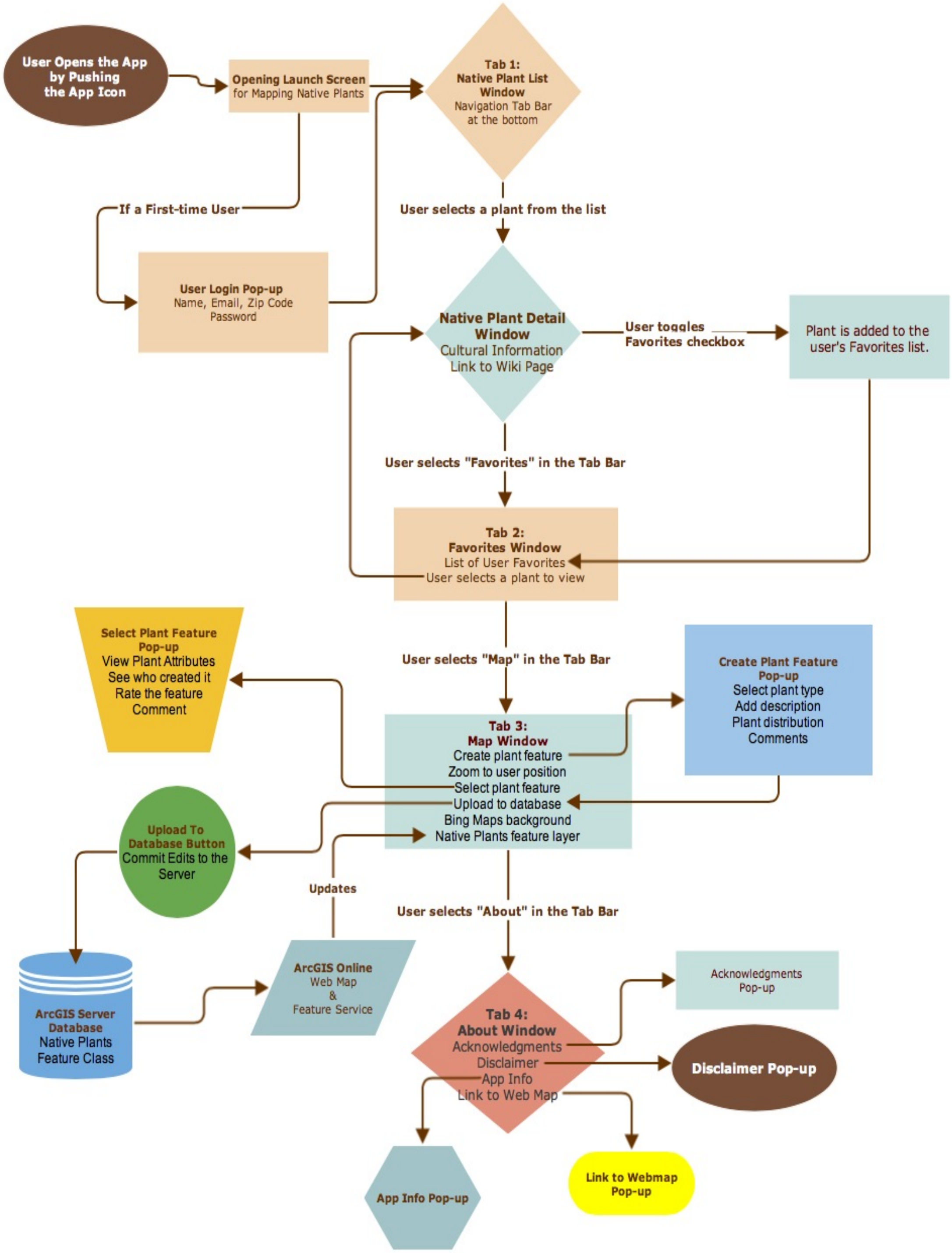

User Opens the App by Pushing the App Icon
Opening Launch Screen for Mapping Native Plants
Tab 1: Native Plant List Window Navigation Tab Bar at the bottom
If a First-time User
User selects a plant from the list
User Login Pop-up Name, Email, Zip Code Password
Native Plant Detail Window Cultural Information Link to Wiki Page
User toggles Favorites checkbox
Plant is added to the user's Favorites list.
User selects "Favorites" in the Tab Bar
Tab 2: Favorites Window List of User Favorites User selects a plant to view
Select Plant Feature Pop-up View Plant Attributes See who created it Rate the feature Comment
User selects "Map" in the Tab Bar
Create Plant Feature Pop-up Select plant type Add description Plant distribution Comments
Tab 3: Map Window Create plant feature Zoom to user position Select plant feature Upload to database Bing Maps background Native Plants feature layer
Upload To Database Button Commit Edits to the Server
Updates
User selects "About" in the Tab Bar
ArcGIS Server Database Native Plants Feature Class
ArcGIS Online Web Map & Feature Service
Acknowledgments Pop-up
Tab 4: About Window Acknowledgments Disclaimer App Info Link to Web Map
Disclaimer Pop-up
App Info Pop-up
Link to Webmap Pop-up

Figure 5: Proposed workflow diagram

To start the Mapping Native Plants app the user taps the app icon on their iPhone. This action opens the launch screen while the app data is being loaded. If it is the first time the app is being used, a pop-up appears asking that the user login with their name, email, zip code and password. Subsequent to the first use the app automatically recognizes the user each time it is opened on their iPhone and any editing or manipulating of the data that they do by identifying their username as the editor. The user's email is used to contact them about updates to the app and their zip code is used as a geographic identifier so that other users and researchers know that this user lives in a certain area and may have more or less knowledge about the area they are geotagging plants in. For example, if a user that resides in the City of Los Angeles is geotagging plants in San Diego then the quality of their data can be accessed independently by other users based on where that particular user (data provider) lives.

Once the app is loaded, the first window the user will see is the Native Plant List window with a navigation tab bar on the bottom giving them a choice to open any of the other main windows. The Native Plant List shows all the native plants that have cultural importance in the Southern California region in alphabetical order (Barrows 1900; Bean 1972; Bean & Saubel 1972; Timbrook 2007; Largo, McCarthy, & Roper 2009). Each row has the native plant's common name and Linnaean name with a thumbnail stock photo of the plant. A user can scroll through this list and select which plant they want to learn more about.

When a user selects a native plant from the list, the Native Plant Detail window appears showing the details of that plant. Users can view the Native American names and uses for the plant as well as what life zone it is normally found in (high desert, coastal chaparral, grassland, forest woodland, etc.). There is also a web view to show the Wikipedia page of that specific plant to give the user even more information about the plant.

Within the plant detail window there is a back button for the user to go back to the plant list window and a favorites checkbox that, if toggled, will allow users to add that plant to their favorites list and simultaneously put a star next to that plant’s name in the native plant list. The Favorites window has the same format as the native plants list, where the favorite plants are identified by their common name, Linnaean name and thumbnail picture. And, like the native plant list, if the user selects a plant from their favorites list the app will automatically bring up the native plant detail window for that plant. If the user wants to take a plant off of their favorites list the user just needs to toggle off the favorites checkbox in that plant’s detail window.

The third main window in the navigation tab bar is the map window. This is where the user can see all of the native plants that have been geotagged by themselves and other users. The map opens with Bing Maps world satellite imagery and the native plants feature service layer. Users can select a plant (feature) to view its attributes, which includes who created and last edited the feature, the date and time it was created, the type of plant it is, the plant’s distribution in that area, the comments added by the user, and the plant’s rating by other users.

When the user selects a plant in the map window a tiny pop-up box appears with the plant’s name and the date and time the feature was created. There is an accessory button in the pop-up box that, if tapped, opens the Select Plant Feature pop-up containing all the attributes of that feature. Within that pop-up there is a button that provides users three options: delete, edit, or cancel. If users delete a feature, an email is automatically sent to the user who created the feature letting them know it was deleted. If the creator contests this deleting action then they can email the geodatabase administrator who can undo the delete action through the geodatabase. The edit option also allows users to rate the feature and make changes to any of the attribute values that they think need to be changed. These actions trigger an email sent to the user who created the feature and that user can contest those changes as well. The cancel option lowers the option window and the user can continue viewing the feature's attributes. When users are done viewing the attributes they can hit the done button and the map window returns.

In the map window there are three buttons: Zoom, Add, and Upload. When the zoom button is pressed the map zooms to the user's location using the GPS device within the smart phone. The user can then see all the native plant features that are near them and geotag new plant features that they see in the field. The user can also pan by dragging their finger across the screen and zoom in or out by pinching or un-pinching the screen.

When the add button is tapped the Create Plant Feature pop-up appears and allows the user enter attribute values for the feature the user is creating. The feature's location is created through the user's position on the map. The user selects the plant type from a drop-down list of native plants, adds descriptive text and comments into the text window, and selects the plant distribution from another drop-down list. The date, time and username are automatically attributed to the feature. When the user presses the done button the map window reappears.

Even though the user has created a feature, the database has not been updated with that feature. This is the job of the upload button. With the add button the user is creating a copy of a feature and saving it locally to the smart phone. If there is no network connectivity the user can continue to collect features offline without an error message telling them they cannot sync to the database. When they come online again the user needs to hit the upload button to commit all the features they had collected offline to the enterprise geodatabase hosted on ArcGIS Server, assuming the user wants to contribute the data to the database. The web map will then be automatically updated through the feature service that is shared within ArcGIS Online. The service is embedded into the app and the changes are seen immediately in the map window.

If the user is looking for more information about the app the fourth tab which is the about window, can be chosen. In this window there are four basic pop-ups: the acknowledgments, the disclaimer, a link to the webmap, and app info. The acknowledgments pop-up is purely a text view that has a list of people who are thanked for their supporting this project. The disclaimer pop-up is also a text view with the warnings and recommendations about how to use the app. The link to the webmap pop-up is where researchers and users interested in accessing the data sources outside of the app can find the URL for the webmap that is shared with users logged into ArcGIS Online. This pop-up has a text view for the URL and a live web view showing how the map looks in ArcGIS Online. The last pop-up, app info, is a table view list of all the books and journal references that were used to generate the native plant list, as of October 2013. If users are interested in furthering their knowledge about native plants and ethnobotany of the study area they can research these resources and learn even more.

4.4 Esri’s ArcGIS Server, Microsoft’s SQL Server and Bing Map Backgrounds

To host the data to be accessed by the Mapping Native Plants app, Esri’s ArcGIS Server 10.1 and its backend connection to Microsoft’s SQL Server 2008 R2 Management Studio were utilized. An enterprise geodatabase was created using ArcGIS Server and ArcCatalog, supported by SQL Server, to manage and create the feature class that the app references and commits edits to.

Aerial imagery from Bing Maps was used in the application to give the user a frame of reference for where they are in the world. Esri has had a partnership with Bing Maps where anyone can use Bing Maps as a background layer in ArcMap for free. This partnership will be changing in December of 2013, when developers will need to obtain a Bing Maps key from Microsoft to use Bing Maps in ArcMap (K, Dana 2013). Esri has other background images that are free to use but they tend to lack the resolution and quality of Bing Maps. Other imagery can be used if necessary, but it was preferred to use the higher quality Bing Map imagery for this thesis study.

4.5 ArcGIS Online

ArcGIS Online, or ArcGIS.com, is a website hosted by Esri that allows users of their products to find help, search for and share data, create groups, and publish web maps for web and mobile apps (http://www.arcgis.com/features/ (last accessed 30 October 2013)). A user can login to ArcGIS Online through ArcMap and connect to and share data that resides online at ArcGIS.com. ArcMap and ArcGIS Server-based feature service data can be shared via the Esri cloud (i.e. ArcGIS.com services) or through the developer’s own server. Other users can find the data that is shared in ArcGIS.com and make their own maps with it through their version of ArcMap or ArcGIS Online’s web map services. Once the feature service is published and shared any researcher or user can access the data and freely use it in their own study if the data owner (creator) grants access to the data to the general public. Feature services may also be kept private by sharing with select users of “Groups” of users who are also registered with ArcGIS.com, for example within companies, agencies or schools.

ArcGIS Online allows developers to share their data and create web maps that can be embedded into mobile apps without having to create a web site of their own to host an online version of the data. It also allows researchers interested in using the data for their own studies appropriate access to the data. If the data is being used for any unethical purpose those users can be restricted from accessing it through access and use constraints currently available in ArcGIS Online.

4.6 Anticipated output

The ArcGIS Online web map provides an immediate output for users to view the app’s data. When the geodatabase is updated, the web map and feature service it references are also updated. A user can find the URL for the web map and feature service in the app info pop-up of the about window. Once users login to ArcGIS Online they can use the feature service in their own maps, and if desired, users can perform spatial or other data analyses for their own research.

Users also have the option to open the web map and add their own layers to it (other feature services, layers, backgrounds) and save it as their own content in ArcGIS Online. Users can either open the webmap in the ArcGIS.com main (default) Map Viewer, in ArcGIS Explorer Online, or on their own computers using ArcGIS 10.1 or higher for Desktop. In addition, from within ArcGIS Online, a user can print out a map and export the native plant data to their own web maps and ArcGIS projects. It is important to note that users are not permitted to change the web map or feature service properties of the Mapping Native Plants app, which preserves the embedded link within the original app.

4.7 Development

To start this project and address the second principle research objective, to code and implement this mobile app for the iPhone, there were several tasks that needed to be accomplished: learn objective-C computing language, register as an Apple developer, download the Xcode development software and the appropriate SDKs and APIs. Simultaneously, the third principle research objective, to create and implement a methodology for allowing mobile device and web browser users to share the data they have collected, was achieved by figuring out a way to create a user-friendly application that allows users to geotag native plants and act as a tool of self-education at the same time. Along the way there were many obstacles and some frustration with learning all of these new things. There were many functions to code and a lot of background database construction that had to be completed to enable those functions. As development progressed there were additional app functionalities originally planned for that turned out not to be possible to develop using the software utilized, and there were other functions omitted because they became unnecessary or beyond the scope of this project given the allotted time frame for this thesis work. Ultimately, a unique, highly useful, and robust application was completed and ready for testing.

4.7.1 Apple Developer and Objective-C Coding

The first task accomplished was to read almost every document in the Apple Developer Library. Within these documents there is information on what objective-C programming is, how to use the Xcode developer software, how to design a new application, how to work with different frameworks, how to build a human interface for the app, how to test the app using Xcode Instruments, guidelines for SDK compatibility, and the different ways to create an app. There were also several books on Xcode programming and how to create iPhone applications that were read to prepare for app development. Basically, the first two months of the project were spent learning how to develop an iPhone application.

Next, several tutorials on how to create simple iPhone apps were followed to learn this type of workflow and to understand what was needed to get an app up and running. During and subsequent to these steps, the app design was visualized and simple hand-drawings of how the app was to look and behave were made, as shown in Figure 6.

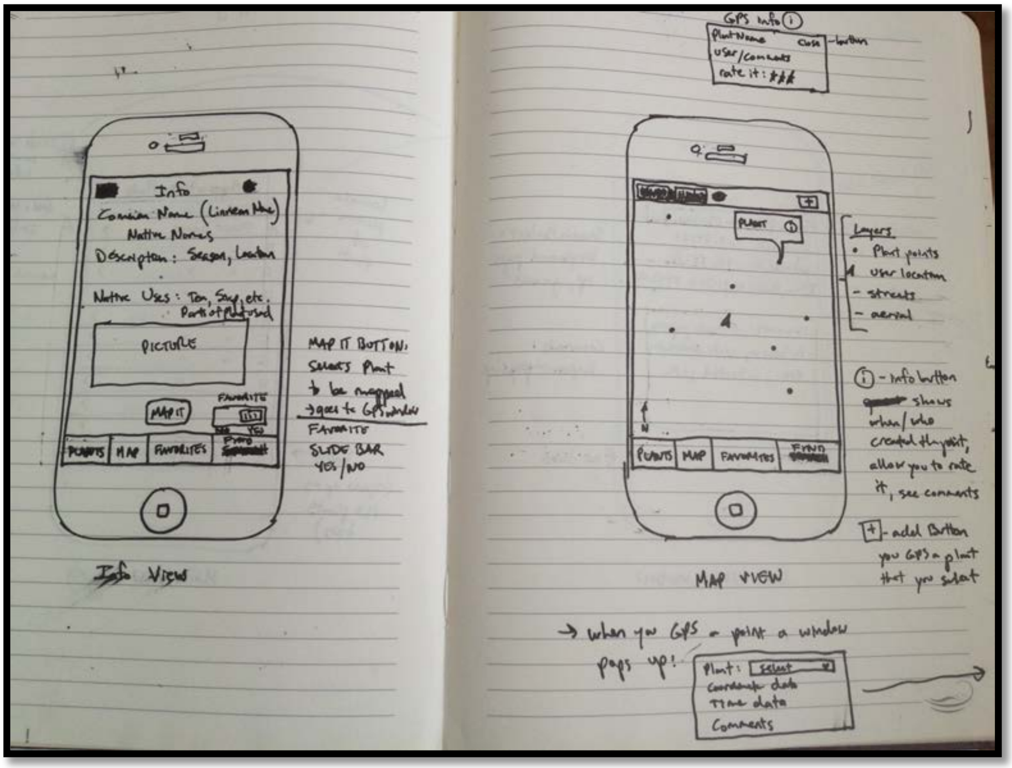


Figure 6: Drawing the proposed user interface

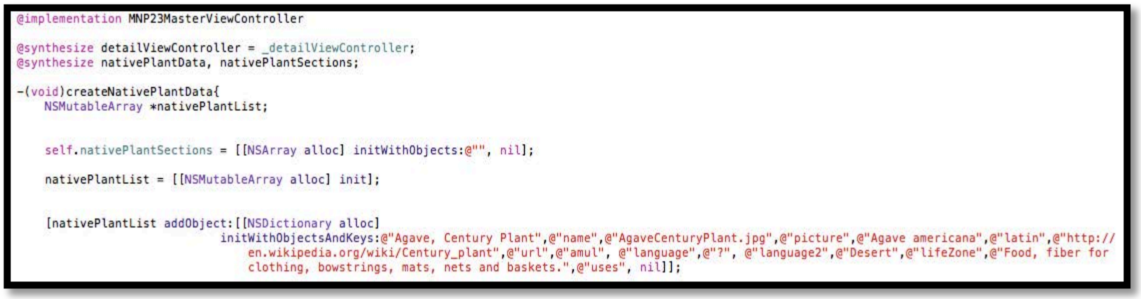
A few test apps were created trying out different app layout styles and to sample a few interesting ideas on how to best develop the app. The main goal was to create what is called a tab bar application where the user can navigate between several main windows using a bar at the bottom of the screen with tabs for each window. It was envisioned that there should be a tab for the Native Plant List window, the map window, the Favorites window and the About window. Because the first window in the app was to be a list of native plants, the main focus of the app development then concentrated on how to develop that window with all the appropriate functions.

4.7.2 The Native Plant List Tab

The first step is to create a list of all native plants to be included in the app. Research was conducted to compile as comprehensive a dataset of native plants in Southern California and their uses in native cultures as possible in the prescribed time frame of this thesis project (Barrows 1900; Bean 1972; Bean & Saubel 1972; Timbrook 2007; Largo, McCarthy, & Roper 2009). Two hundred and sixty-seven plants were chosen, partly because detailed descriptions of their cultural uses and native names were discovered through these research efforts (Barrows 1900; Bean 1972; Bean & Saubel 1972; Timbrook 2007; Largo, McCarthy, & Roper 2009). This list was input into an excel spreadsheet and attributes deemed most critical to native plant studies and their cultural context were recorded: Linnaean names, common names, life zone, cultural uses, cultural names and the URL to each plant's current Wikipedia page.

Southern California tribes are revitalizing many of the languages that were spoken in the past and only a few have many of the native plant names documented (Barrows 1900; Bean 1972; Bean & Saubel 1972; Timbrook 2007; Largo, McCarthy, & Roper 2009). There were many native names documented from the Cahuilla and Chumash tribes for native plants, who both lived in extensive areas of Southern California in the past, and so all of the native plant names used in the app came from these two tribe's languages.

In developing any sort of mobile app, especially iPhone apps, there seem to be ten different ways to do program any one task. In this project an SQLite database was created, a properties list (plist) was written and multiple arrays were coded. Each of these tasks were intended to create a table view in Xcode that would display the list of native plants within the app. The SQLite database turned out not to be compatible with loading data into the table view, and although the plist would load it was very complicated to add attributes into the list and then have the plants in the list reference those attributes. After a great deal of testing, one coding method worked satisfactorily: a mutable (editable) array called “nativePlantList” was hard-coded to hold each plant object and their attributes, as can be seen in Figure 7.



```
@implementation MNP23MasterViewController

@synthesize detailViewController = _detailViewController;
@synthesize nativePlantData, nativePlantSections;

-(void)createNativePlantData{
    NSMutableArray *nativePlantList;


    self.nativePlantSections = [[NSArray alloc] initWithObjects:@"", nil];

    nativePlantList = [[NSMutableArray alloc] init];


    [nativePlantList addObject:[[NSDictionary alloc]
                                initWithObjectsAndKeys:@"Agave, Century Plant",@"name",@"AgaveCenturyPlant.jpg",@"picture",@"Agave americana",@"latin",@"http://
                                    en.wikipedia.org/wiki/Century_plant",@"url",@"amul", @"language",@"?", @"language2",@"Desert",@"lifeZone",@"Food, fiber for
                                    clothing, bowstrings, mats, nets and baskets.",@"uses", nil]];
```

Figure 7: Coding the Native Plant array in Xcode

Using the nativePlantList array the native plant objects can be referenced in the table view as well as in a detail window that pops up when a certain plant is selected. Although it took a considerable amount of time to manually enter all two hundred and sixty-seven plant objects into the array, this functionality is important to have in the app. The user can thus scroll through the native plant list, select a plant, and view the details of each plant.

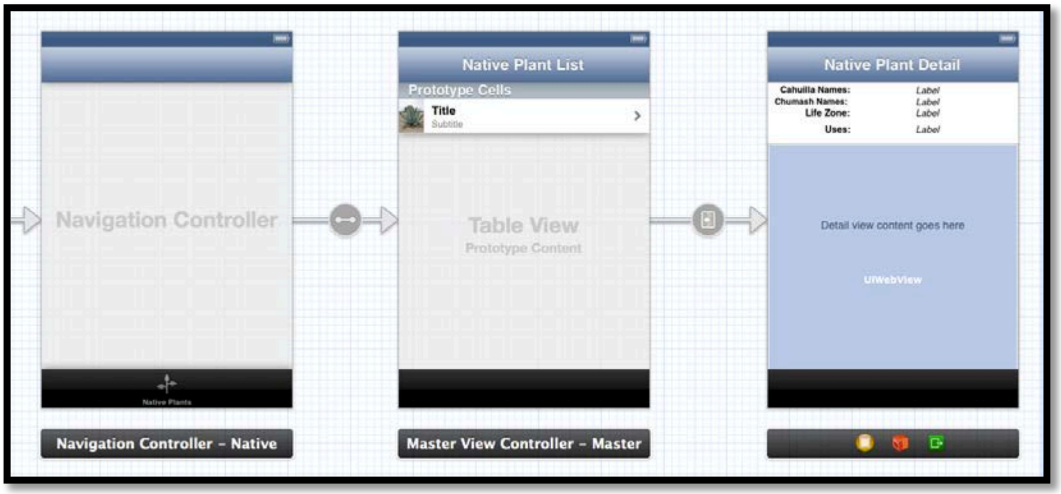


Figure 8: The Native Plant List user interface as depicted in the Xcode Simulator

Figure 8 shows how the user interface is designed for the Native Plant List and Native Plant Detail view controllers within Xcode. The titles and subtitles in the cells of the table view and the labels in the detail view are designed to reference the object attributes that are written within the native plant coded array. A user interface web view (UIWebView) was also created to reference the current Wikipedia page for each plant. Photos of each plant were then found on Wikipedia, saved within the app as project files and manually resized to fit inside the cells of the table view. Once the native plant window was completed the next task was to develop a map window that allows users to geotag native plants, share their data, and view the data that others have shared.

4.7.3 The Map Tab

During research on how to create a map window, several iPhone tutorials were found that used the iOS MapKit framework to develop map windows and host data from the cloud services. With these tutorials a user interface is developed to show a basic map background layer, but there was no straightforward way to go about hosting data from a private web site or from a commercial cloud-based hosting service without spending a lot of money (for the hosting privileges), and additional time and effort in coding. The reality is that a server is still required to host the data. And to share the data output from the app with other users and researchers there needs to be a website in which the feature service could be embedded. In addition, to address the principle research objectives the ability for users to be able to add, delete, edit, and query this data in the map window was required.

Ultimately it was determined that the app could be designed to rely on the resources of the Spatial Sciences Institute (SSI) at the University of Southern California (USC) and their site licensing with Esri. The solution was to have the Information Technology (IT) administrator at SSI create a Virtual Machine (VM), with all the Esri software, licenses and administrator privileges installed, to handle database development and feature service publishing. Through a remote connection to this VM, the enterprise geodatabase was created and the native plants feature class was published as a feature service in ArcGIS Online.

First, in version 10.1 a connection to ArcGIS Server was made within ArcCatalog. This published an ArcGIS Server connection with a server URL and grants the developer administrative permissions for that server with a user-authenticated login. Then, the folders that the feature service data were saved to were registered in the GIS Server Properties Data Store. This ensured that the data (maps, feature classes, etc.) saved into those folders was registered and associated with the GIS Server being used.

Second, an enterprise geodatabase was created in ArcCatalog using a SQL Server instance. A connection was established to the geodatabase in ArcCatalog using an administrator login and a database user named “plantWatcher” was created to allow the app user access to the geodatabase by using the plantWatcher’s user name and password. A second geodatabase connection was created using the plantWatcher user login. This second connection was then registered with ArcGIS Server just like the data folder registered previously. Once the registered data folder and geodatabase connection were validated with the ArcGIS Server connection the native plants feature class was created and published.

The connection to ArcGIS Server was then published as a feature service and simply named “plants”. In order to use the feature service in the app, Esri’s ArcGIS Runtime SDK for iOS was downloaded and the Xcode project was configured to work with the SDK. This includes adding the appropriate framework search paths, links to the ArcGIS libraries, adding the ArcGIS resource bundle and modifying the build flags. Esri has also created a resource website called “ArcGIS for Developers” that contain the files, tutorials, and samples to help developers use the SDK within the applications they are developing (Esri 2013). To complete the map window several different Esri samples and tutorials were used and credit was given to Esri by placing their logo within the map view (Esri 2013).

The code for many of the functions in the map window were created using the Esri samples, including: zooming into the user location, switching the backgrounds, manipulating the settings of the GPS in the iPhone, selecting a feature and showing it’s attributes, editing the feature attributes, and adding a new feature to the map (Esri 2013).

The sample codes were modified to use the plants feature service and perform the functions designed for the app.

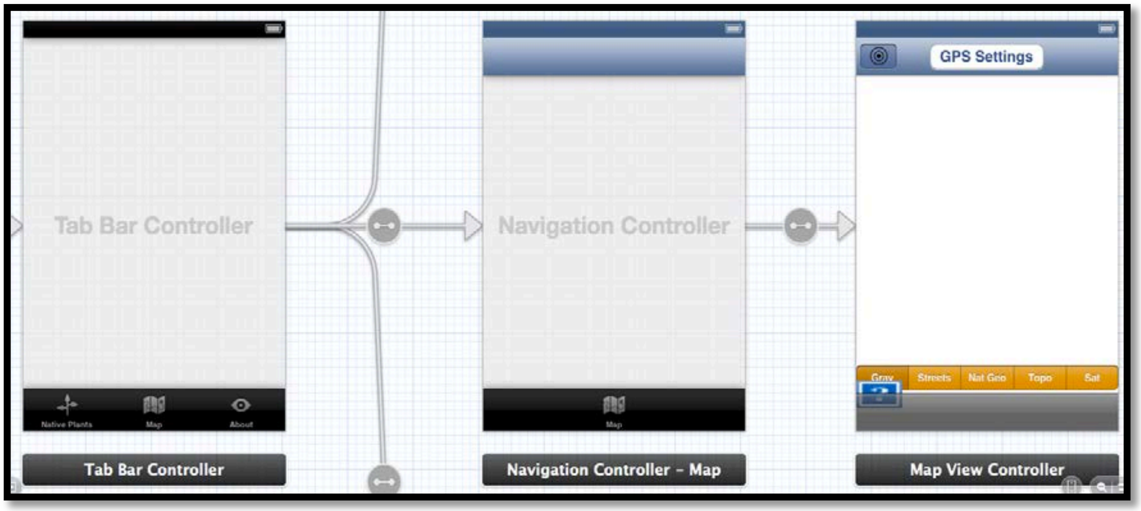


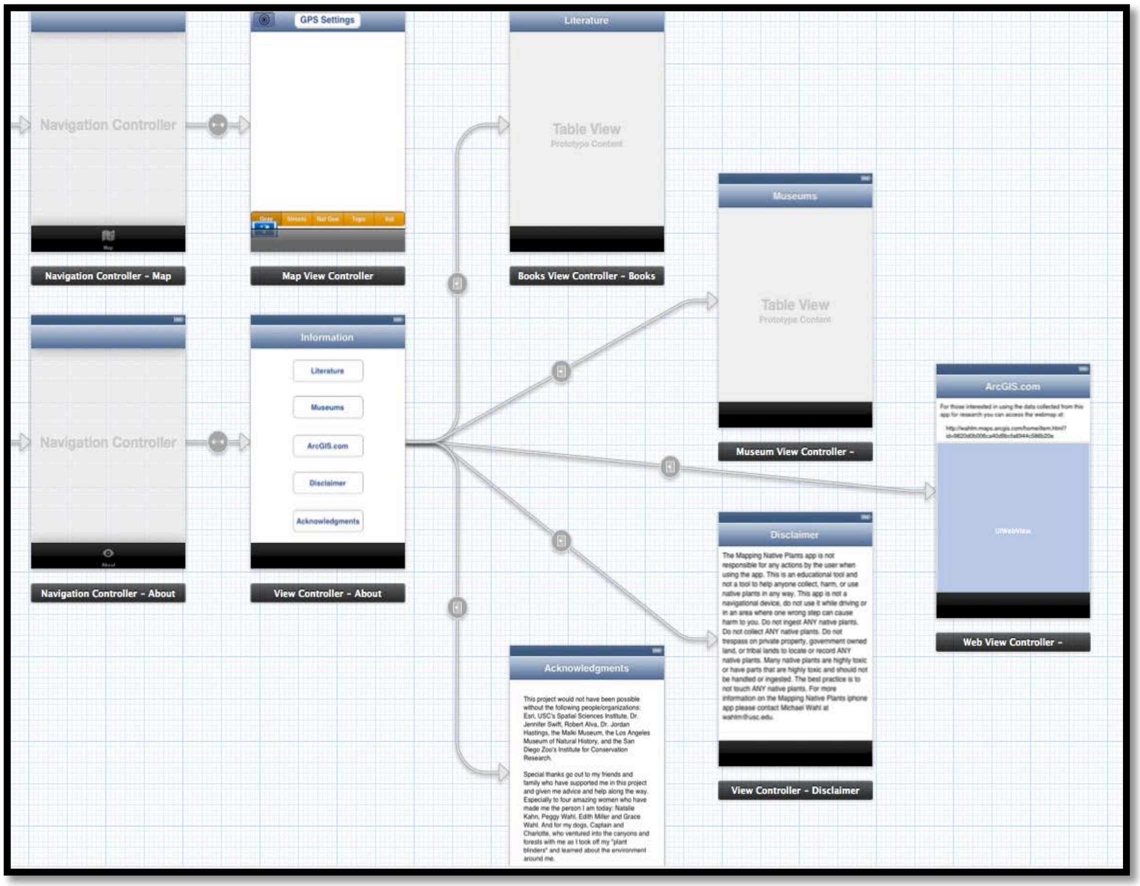
Figure 9: Map user interface, as depicted in the Xcode simulator

Figure 9 depicts how the user interface for the Map window was developed in the Xcode Interface Builder. Each function has its own method in the code and some methods had pop-up windows coded for them manually so they do not appear in the user interface. You can see the zoom in button and the GPS Settings button at the top, and the Esri logo placeholder and background maps switch bar on the bottom. The add button was manually coded as was the pop-up showing the native plant attribute and template picker windows.

While the map window looks fairly simple it contains multiple pages of coding and was very complex to organize and design. It was challenging to blend all the samples into one app and have all the various objects and methods work together. Through multiple cases of trial and error a map window was developed to provide most of the functions that were in the original app design and thus address this primary thesis objective.

4.7.4 The About Tab

The About tab was the final part of the app to be developed. This was relatively simple to code because it is basically a collection of view windows displaying information about the app. It consists of a view controller with five buttons connected to five pop-ups containing information about the app. The five pop-ups include: Literature, Museums, ArcGIS.com, Disclaimer and Acknowledgments. The Literature pop-up is a list of all the books and journal references that were used to construct the native plant list for the thesis study area. It consists of a table view that the user can scroll down in order to see all the different resources that are referenced. The Museums pop-up is similar to the Literature pop-up, a table view, but instead has a list of all the museums and botanical gardens that have information about native plants and native cultures in the Southern California area. The ArcGIS.com pop-up has a text holder and a UIWebView that shows the link to the output web map and rest URL for the feature service. The Disclaimer pop-up also has a text holder where users can view legal warnings and advice for using the app. The last pop-up, the Acknowledgments, is a text holder where the people who helped with this project are thanked. See Figure 10 to view the app interface developed in the About Window.



For those interested in using the data collected from this app for research you can access the webmap at:

http://wahlm.maps.arcgis.com/home/item.html?id=9820d0b006ca40d9bcfa0944c586b20e

The Mapping Native Plants app is not responsible for any actions by the user when using the app. This is an educational tool and not a tool to help anyone collect, harm, or use native plants in any way. This app is not a navigational device, do not use it while driving or in an area where one wrong step can cause harm to you. Do not ingest ANY native plants. Do not collect ANY native plants. Do not trespass on private property, government owned land, or tribal lands to locate or record ANY native plants. Many native plants are highly toxic or have parts that are highly toxic and should not be handled or ingested. The best practice is to not touch ANY native plants. For more information on the Mapping Native Plants iphone app please contact Michael Wahl at wahlm@usc.edu.

This project would not have been possible without the following people/organizations: Esri, USC's Spatial Sciences Institute, Dr. Jennifer Swift, Robert Alva, Dr. Jordan Hastings, the Malki Museum, the Los Angeles Museum of Natural History, and the San Diego Zoo's Institute for Conservation Research.

Special thanks go out to my friends and family who have supported me in this project and given me advice and help along the way. Especially to four amazing women who have made me the person I am today: Natalie Kahn, Peggy Wahl, Edith Miller and Grace Wahl. And for my dogs, Captain and Charlotte, who ventured into the canyons and forests with me as I took off my "plant blinders" and learned about the environment around me.

Figure 10: About window interface, as depicted in the Xcode Simulator

4.8 Challenges: Cultural and Technological

Developing the app took many hours and days of hard work to complete and there were many types of obstacles and challenges to overcome. This section will describe the various challenges that were met in this project and the strategies developed to work through them.

The first challenge was not a coding or software and hardware challenge, but a cultural challenge. Before the app was developed it was important to decide whom the client, or main users or audience, of the app would be. Several Southern California tribes were contacted through their museums and tribal liaisons to ask for their support and interest in a project such as this so the app could be developed for and tested by Native American users. Instead of receiving the interest and acceptance that was anticipated, there appeared to be hesitation on the part of the intended client. The main concern of the tribes was interpreted to be that this app would entice outsiders to venture onto private tribal lands and disrespect or misuse the plants. It was reasoned that a legal disclaimer would be included in the app to discourage anyone from collecting or using native plants in any way and deter users from trespassing on any kind of private property. Ultimately, no response whatsoever was ever received from the tribes approached about the project about whether or not they wanted to participate in it's development process, so the idea of creating an app specifically for the tribes and in cooperation with them was then no longer considered.

To overcome this challenge the client/user group was expanded to include all people living in the Southern California area. The scope of the project was then broadened in focus to include more cultures and more languages with the potential to involve many more people to contribute their local knowledge of native plants. It was disheartening not to have the participation of the tribes that had originally inspired this thesis, but as a result the app developed is more robust because the scope was expanded to include the potential usage of all people in Southern California.

All other challenges were technical in nature with one of the more frustrating challenges being the amount of software updates that occurred over the one and a half years it took to program the app. Although updates are a good way to fix bugs and keep the software relevant with new technology, it was troubling to build code that had been working for several months then have i.e. ninety-seven error messages suddenly appear.

The error messages were sometimes generic in nature, which did not help to pinpoint the specific technical problem. Here is a sample of such an error message:

"No visible @ interface for 'AGSMapView' declares the selector 'mapLayerWithName:"

After several days of searching the Internet to find out what this error message meant and how to fix it, it turned out that a software update was required. When Xcode was updated to its newer version all of the error messages disappeared and the app was up and running once again.

When the project first started, Apple had updated the Xcode software to version 4.1. There were major differences in this version from the previous versions. Xcode now allows the developer to use Automatic Reference Counting (ARC) and implement storyboards as the user interface instead of using NIB (NeXT Interface Builder) files as were previously used. It made the developing environment easier to use but all of the books, tutorials, and sample code had not been updated along with this change. None of the examples worked with Esri software, they had special code for memory releasing that was no longer needed in the new version of Xcode, and they all used NIB files as a user interface. The only book that dealt with version 4.1 of Xcode was purchased online but the author kept making changes and the book wasn't released for six months after it was ordered. Fortunately, over time more and more tutorials and samples were made for the new version and these explained how to get around using NIB files and code to deal with issues, such as memory leaks, appropriately.

Through the course of developing the code for the Mapping Native Plants app there were many stages of trial and error. Other functionality originally intended for the app included a search bar for the native plant list, a user authenticated login that would update the user field in the native plants feature service, a user favorites window attached to the native plant list, and an offline capability for when the user was out of network range and couldn’t connect to the server to sync their geotags. Although the code was written to support all of these functions, these were not successfully implemented.

The search bar would not display the names in the native plants array and in order to get it to work the entire native plant list would need to have been rebuilt using a different method in the code. The user-authenticated login required the creation of a separate database for user names and passwords that did not work well with the rest of the app. There were also ethical concerns about user privacy that fell beyond the scope of this project so it was decided to add the “user initials” field instead and give the user the option to enter their initials, or not, to identify themselves. The user favorites tab was another function that went beyond the scope of the project. Originally, it was planned to be a main window in the app as a convenience, but it really had nothing to do with learning about native plants or geotagging them so it was decided to omit this function.

The offline editing capability, on the other hand, is something that was not successfully incorporated into the app. When a user goes offline they should be able to add a feature and a local copy of the feature is created on the iPhone. When the user comes online again they can push the upload button to sync the copied feature to the server, if desired. If the user does not press the upload button then their geotags will not be added to the feature service. The code was written to implement this function but there were problems with connecting it to the geodatabase. The original sample code from Esri shows how to code for offline editing using a feature service that was published from the Esri cloud but the code was outdated and not ARC-enabled.

Another challenge was implementing NIB files from older sample code in the app, which now uses a storyboard user interface. The NIB files and their code needed to be translated to the storyboard format to build a user interface and attach all the action and outlet delegates to their proper buttons, labels and windows. This was doable for most of the UI windows, but there were some functions that still relied on the NIB files in the older projects they referenced. Copies needed to be made of those NIB files then imported into the mapping native plants Xcode project and the code rewritten to work with the new NIB files. This was accomplished by deleting all of the old code builds of the project that still referenced the older projects. This was learned when after adding the NIB files a lot of the manually programmed pop-ups were appearing black when the project was built. After deleting the old references and attaching the newly copied references the black screen problem was solved.

One of the more major technical challenges was using the VM and creating the enterprise geodatabase properly. The first issue was with using the "localhost:6080" instance of SQL Server to host the geodatabase. There was no problem with creating the geodatabase, adding a database user, registering the folders and database, and publishing the feature service but when it came time to access the rest services URL for the feature service outside of the VM there was an Internet error message saying it could not connect. What was needed was a unique host name for the server that could be accessed outside of the VM. The instance had to be named after the VM and use the sqlexpress database engine called "gist-wahlm/sqlexpress". There was another problem that appeared immediately after, where the login authentication for the database user was set to Operating System (OS) authentication but actually needed to be Database Authentication. When the database user was created this error message was received: “The connection property set was missing a required property or the property value was unrecognized. Bad login user.”

There was an administrative permission issue in the VM that was not allowing Database Authentication. Once the IT administrator at the SSI updated the admin settings to allow the author privileges to create new databases, a new database using Database Authentication was generated and a database user with the proper credentials was created.

Although these challenges may sound like they should have been easy issues to overcome, many took days, weeks and sometimes months to figure out and fix. The ArcGIS Server 10.1 and SQL Server 2008 database issues alone took months to figure out and fix. There were a couple of strategies developed to overcome these challenges. The first strategy was to email the advisor of this thesis with all of the problems and let her decide if it needed the IT administrator’s attention or not. They both helped to understand the problems and figure out ways to fix them. They would meet to discuss each issue and spent many hours replicating these procedures to figure out what was wrong. Without their help this project would have never gotten through to the development stage.

The strategy used to battle the coding challenges was to query the Internet for help. There are countless forums and chat rooms for developers to discuss their issues and help each other solve problems. Many times a Google search of the error message would lead to finding an answer. One free website in particular was extremely helpful, www.stackoverflow.com (Stack Exchange Inc 2013 (last accessed 30 October 2013)). Through this site a developer can ask a question about a problem, provide the code and error message they are getting, and professional developers from around the world will give their best answers. For example, Stackoverflow.com reported that iOS gives a generic error message for a thousand different kinds of errors. Countless hours were spent searching the Internet looking for solutions to error messages that failed to explain what the actual problem was.

Another strategy was to refer to all the documentation that Apple provides for their developers. Apple has a special series of documents just for troubleshooting developer problems (Apple Developer 2013). For instance, there was a problem with uploading the Mapping Native Plants app to an iPhone because there was an issue with the developer's provisioning certificate. The troubleshooting documents showed how to delete the old certificates and create new ones through the Xcode Organizer window. Simply following the steps in the troubleshooting document solved the problem.

There were numerous smaller problems that appeared and solutions discovered throughout the development of the app, but there were simply too many to document in this thesis. They ranged from issues with sourcing the correct files and connecting objects to their proper methods and delegates, to missing a comma or capitalizing a letter that should have been lowercased. Overall, every problem was fixed and an app was developed that executes every function needed to let the users learn more about native plants in Southern California, geotag the plants they find in the field, and view, edit, and rate the contributions made by other users.

4.9 Mapping Native Plants Revised Workflow Diagram

After the code for the Mapping Native Plants app was completed there were differences, between the original proposed workflow and the successful workflow of the app. As mentioned in the challenges section there were a few functions that were not feasible, beyond the scope of the project, and others that were deemed not possible because they required the entire project to be reformatted. The revised, final workflow shown in Figure 11, still contains the main elements of the app that are needed to inform people of native plants and allow them to geotag them using their smart phones, thus addressing the principle research objectives of this thesis project. In this section each element of the new workflow is described while pointing out the differences between the initially proposed and revised final workflows.

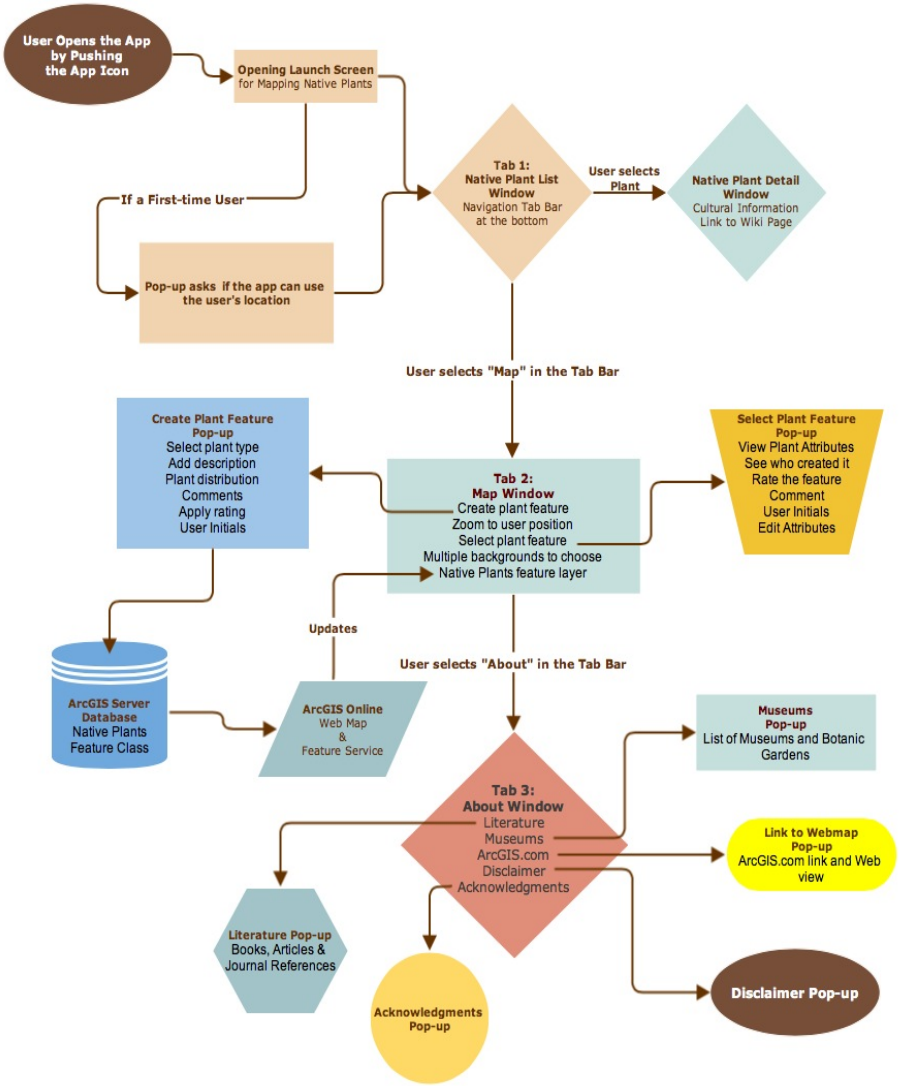

User Opens the App by Pushing the App Icon
Opening Launch Screen for Mapping Native Plants
If a First-time User
Pop-up asks if the app can use the user's location
Tab 1: Native Plant List Window Navigation Tab Bar at the bottom
User selects Plant
Native Plant Detail Window Cultural Information Link to Wiki Page
User selects "Map" in the Tab Bar
Create Plant Feature Pop-up Select plant type Add description Plant distribution Comments Apply rating User Initials
Tab 2: Map Window Create plant feature Zoom to user position Select plant feature Multiple backgrounds to choose Native Plants feature layer
Select Plant Feature Pop-up View Plant Attributes See who created it Rate the feature Comment User Initials Edit Attributes
Updates
ArcGIS Server Database Native Plants Feature Class
ArcGIS Online Web Map & Feature Service
User selects "About" in the Tab Bar
Museums Pop-up List of Museums and Botanic Gardens
Tab 3: About Window Literature Museums ArcGIS.com Disclaimer Acknowledgments
Link to Webmap Pop-up ArcGIS.com link and Web view
Literature Pop-up Books, Articles & Journal References
Acknowledgments Pop-up
Disclaimer Pop-up

Figure 11: Revised workflow diagram

In the original workflow the user starts the app by tapping the app icon on their phone and the launch screen opens. If it were the first time opening the app the user enters their login information (username, password, zip code) into a login pop-up, and then be taken to the Native Plant List window. In the revised workflow, instead of a login pop-up they are greeted with a pop-up asking if the app can access the user's current location data through the GPS in the iPhone. If the user chooses yes, the app will then center on this location when the zoom button is pressed in the map window. If a user chooses “no” the location method in the code will not be called when they use the zoom button and the map will not center on their location, and the user may zoom and pan as desired.

In the Native Plants List window, next the users can scroll through all two hundred and sixty-seven native plants and view the common name, Linnaean name, and a thumbnail stock photo of the plant. When the user selects a plant they want to know more about they can open the Native Plant Detail window and see the cultural information of Chumash and Cahuilla names and their native uses, the life zone of the plant’s natural habitat, and the plant’s current Wikipedia page.

When the user taps the “Map” tab this action opens the map window. The map automatically displays the Light Gray Canvas background from ArcGIS Online and the native plants feature service overlaid on top of it. There is a switch bar at the bottom that lets them choose what kind of app background they would like: Light Gray Canvas (default), Street Map, National Geographic, Topographic, or Satellite imagery. In the proposed workflow the map was coded to default to Bing Maps imagery, but because Bing Maps was going to change proprietary hands in several months it was decided to use the free backgrounds available on ArcGIS Online is used in the final version.

When the user taps the zoom button the map zooms to their position and a blue bulls-eye appears in their location. If the user is moving around they can change the settings of the GPS by hitting the GPS Settings button. Users can control the accuracy of the GPS and frequency of when it updates based on how far they have to move to get an update. The greatest accuracy an iPhone can get is approximately 10m. From testing the iPhone against other GPS units in a field project on Catalina Island several GIST students, including the author, were able to produce an accuracy of 3-5m for the iPhone. The accuracy largely depends on three elements: the GPS unit inside the phone, the network connectivity, and the cellular provider’s signal. The default accuracy for the Mapping Native Plants app is set at “best” but the user can change that to 10m, 100m, or 1km depending on whether they want to conserve energy in the phone being used by the GPS.

When the user selects a native plant feature in the map window a small pop-up appears and displays either the comment or the user initials of the user who created the feature and the date it was created. An accessory button appears in the pop-up and if it is tapped the user can view all the feature’s attributes and edit them or delete the feature. Once users are done editing, the update is synced to the server or they can cancel out and return to the map window.

To create a new feature in the map the user taps the add button and a template picker window appears for them to choose what kind of plant they want to geotag. When they have picked the plant they want to geocode a pop-up appears with all the empty attribute values of the feature. The user fills in as much information as desired and optionally can add a photo or video attachment to the feature. When the user is finished creating the feature they tap the done button and the feature is automatically synced to the server and appears in the web map in ArcGIS Online and in the app’s map window, usually within a few seconds.

In the final workflow, if the user wants to learn more about native plants in Southern California and the native cultures who used them they can navigate to the

“About” tab and see the books and articles that were used to create the Native Plant List in a pop-up called “Literature”. There is also a pop-up called “Museums” where they can find a list of all the museums and botanical gardens in the Southern California region. In the “Disclaimer” pop-up they can view the legal warnings and recommendations for using the Mapping Native Plants app. For those interested in how they can use the data in their own research there is the “ArcGIS.com” pop-up that has a link to the ArcGIS Online web map, the rest services URL for the feature service, and a web view of the web map itself. The “Acknowledgments” pop-up is where credit is given and thanks is made to those who helped with this project.

Although the revised workflow diagram is a condensed version of the proposed workflow, it remains true to the original objectives of this study by adequately addressing all of the principle research objectives. Although the app functions, including the login window, the favorites tab and the offline editing function, were all scratched from the proposed workflow, the final output (the web map) is the same and still aims to educate people about native plants and their ethnobotanical uses by letting them geotag and share their own knowledge of native plants with other users.

4.10 Application Evaluation Method

The testing cycle for the Mapping Native Plants app was broken up into three parts. The first part was to assess the performance of the app on an iPhone being used in the field to make sure it functions in different types of environmental scenarios. The second is a type of laboratory testing to evaluate if the code was written properly and not leaking memory or using too much of the iPhone’s battery (see Chapter 5). The third part was a usability test where five volunteers are asked to use, or test, the app and fill out a questionnaire answering questions about how they liked or disliked the app and whether they had any recommendations for improvements.

Depending on the results of each test, improvements were made to the app. Participants in the usability testing are archaeologists both with and without experience using apps on an iPhone. The testing framework to evaluate if the app is a success is based around the criteria that the app's functions all worked, that there were no problems with the code, that the user's found the app understandable and easy to use, and that it fulfills the purpose of this thesis for geotagging native plants and sharing the knowledge of their location and cultural uses. The testing period took two weeks to complete and the results were collected in the form of screenshots of computerized tests through Xcode's testing software called Instruments, the questionnaire passed out to all test subjects then collected, and a spreadsheet used to record the app functionality results.

The first test was a functionality and performance test. After installing the app on an iPhone 4S, it was taken into the field and used in different environmental conditions under different states of network connectivity and battery life. In this test the length of time it took for windows to load and the functions to perform in different parts of Oahu (downtown Honolulu with tall buildings, dense forest canopy, canyons with steep walls, open fields, streets with power lines overhead, in a moving car, in my apartment, and at the beach) were recorded. In each area the app was tested, the network connection (Wi-Fi, 4G, or intermittent connectivity) and the battery life of the phone (100%, 50%, and <20%) were noted. A spreadsheet was created to calculate the average times it took for the app's functions to perform in each of these environmental conditions and states of connectivity and battery life as well as answer some basic questions about app performance (did it crash?, do any functions use more or less battery life to perform?, do any error messages appear?, is there lag time during loading?, etc.).

The second test was a laboratory test to assess the quality of the coding as well as an interrupt test to evaluate how the app behaves on the iPhone when other apps are running, when phone calls and text messages are received, and when app notifications appear while the app is running. To run this test the iPhone needed to be connected to a laptop running Xcode and using the Instruments testing software. Since this testing method required the iPhone to be plugged into a laptop the test was run when the app was connected to a Wi-Fi network and in a charging state with 100% battery life. The results will be shown through a screenshot of the Instruments window after each test is completed.

The third test is to have users use it and determine its usefulness, aesthetic value, and usability. Five Oahu archaeologists were used as test subjects and they all used the app loaded onto an iPhone 4S for however long it took them to use every app function. They were asked to do anything they could to make the app crash. If they could break the app then there was a major problem with it that needed fixing. After they were done testing the app they were handed a questionnaire and asked to be as truthful as possible because their answers will help to make the app even stronger.

The testing schedule took two weeks to complete with the brunt of the work being the fieldwork and travelling around Oahu to find the appropriate places to test the app. The preferred place to test the app would have been in Southern California, employing local volunteers who grew up there as test subjects, but the author had moved to Oahu in the middle of the project and flying back to California to test the app was not within the project's budget. Testing the app in Oahu was sufficient for the testing phase and turned out to be a better place to test due to fewer choices related to testing where there was good network connectivity, and it required less travel time between the different types of environments in which to test the app. In the future the app could also be tested in Southern California to determine if the results of those tests would differ, but no differences are anticipated.

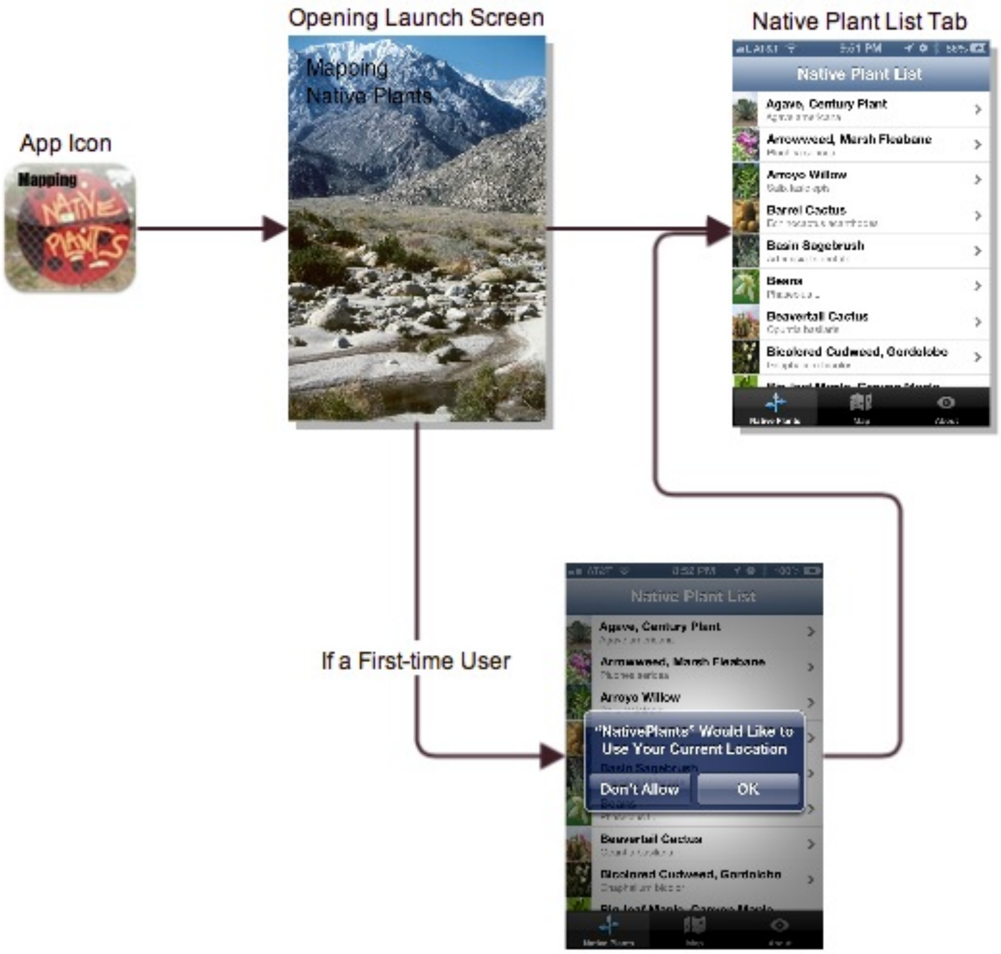
Overall, the coding effort was aimed at satisfying all three of the principle research objectives, as stated in Chapter One. The final results of this programming effort are reported in detail in the next chapter of this thesis.

CHAPTER FIVE: RESULTS

The Mapping Native Plants app took over a year and a half to design and develop. Once the major issues with the geodatabase were solved, the coding for the map window took very little time in comparison to complete, although not without the challenges described in detail in previous chapters of this thesis. This chapter includes an explanation about how the app functions work in the real world and the results from testing that have been evaluated. This discussion covers in detail how the first and third principle research objectives of this work were successfully addressed: the design of a mobile GIS application that allows users to select, add, modify, and delete geotags representing native plants in the Southern California area, and how the methodology for allowing mobile device and web browser users to share the data they have collected is implemented in the real world.

5.1 Mapping Native Plants App Functions

When a user first opens the Mapping Native Plants app on their iPhone they will see the opening launch screen showing a picture of Mt. San Jacinto near Palm Springs, California (Figure 12). If it is the first time the user is using the app, a notification will appear asking the user if the app could utilize their location. Once a user chooses “Yes” the Native Plant List tab will appear and they can start using the app. If the user clicks “No” then the map window will not show or zoom into the user’s location. For a returning user the app will automatically open to the Native Plant List tab. The lag time for initial app loading is set to three seconds so every time the app is opened the launch screen will appear for three seconds before the Native Plant List tab appears.



Agave, Century Plant
Arrowweed, Marsh Fleabane
Arroyo Willow
Barrel Cactus
Basin Sagebrush
Beans
Beavertail Cactus
Bicolored Cudweed, Gordolobo

"NativePlants" Would Like to Use Your Current Location

Figure 12: Opening launch sequence, as depicted in the Xcode Simulator

If the user rotates the screen from portrait to landscape the orientation of the app will automatically change as well (see Figure 13). When the orientation is changed it calls upon the “shouldAutorotateToInterfaceOrientation” method as depicted in Figure14.

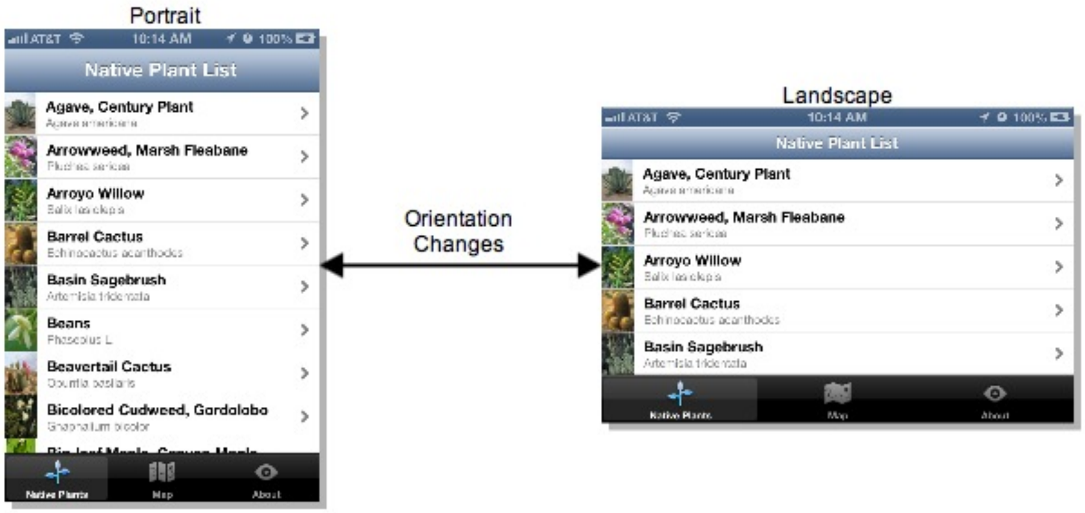
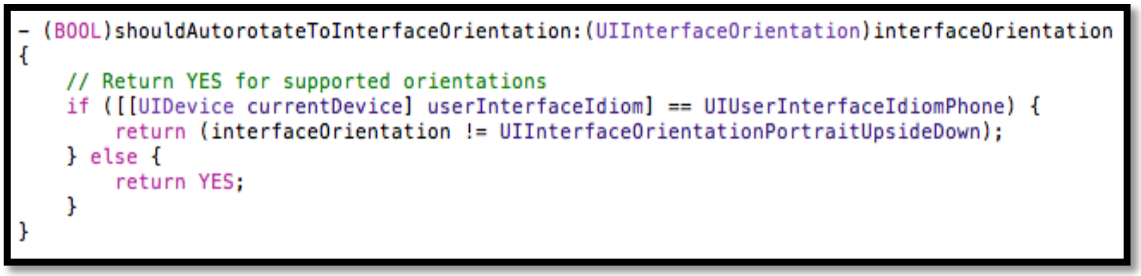


Figure 13: Rotating the user interface orientation, as depicted in the Xcode Simulator



```
- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
{
    // Return YES for supported orientations
    if ([[UIDevice currentDevice] userInterfaceIdiom] == UIUserInterfaceIdiomPhone) {
        return (interfaceOrientation != UIInterfaceOrientationPortraitUpsideDown);
    } else {
        return YES;
    }
}
```

Figure 14: Code to rotate the interface orientation, as depicted in Xcode

The user can navigate through the app by using a tab bar at the bottom of the app screen containing the three tabs for each window: Native Plants, Map, and About. They simply touch the tab for whichever window they want to navigate to and that window will appear (see Figure 15). The user can navigate to any of these windows at any time during the operation of the app, except when a feature is being added, edited or synced to the server.

The Mapping Native Plants project in Xcode was created as a Tab Bar Application from the start. This means that the resulting tab bar is automatically created in the user interface. It is up to the developer to create the view windows and link them to their spaces on the tab bar, as described in Chapter 4.

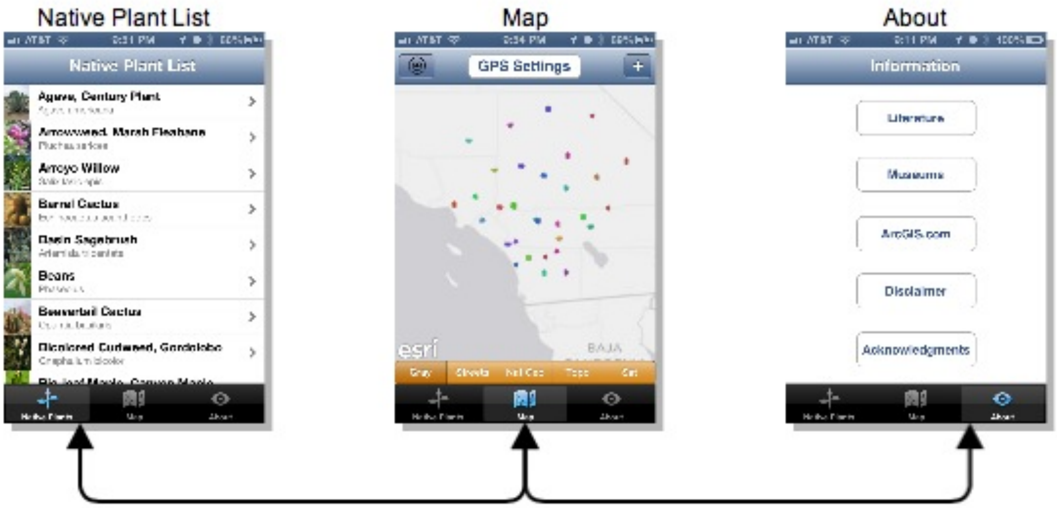


Figure 15: The navigation tab bar, as depicted in the Xcode Simulator

The Native Plant list is the first thing the user can interact with in the app. It consists of two hundred and sixty-seven different types of native plants that have been used by native cultures in Southern California (Barrows 1900; Bean 1972; Bean & Saubel 1972; Timbrook 2007; Largo, McCarthy, & Roper 2009). Users can scroll through the list (which is in alphabetical order) and view the common name, Linnaean name, and a thumbnail picture for each plant. When users find the plant they are looking for they can select that plant by touching its row in the list. This will highlight that row and open the Native Plant Detail window for that plant as shown in Figure 16.

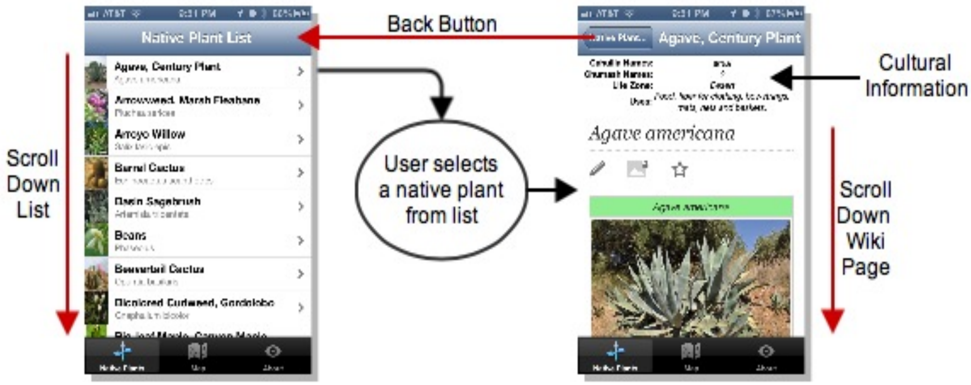


Figure 16: The native plant list and native plant detail windows, as depicted in the Xcode Simulator

In the native plant detail window users can view the cultural information for each plant (Cahuilla and Chumash native names and uses), the life zone of the plant’s habitat (where it can be found in Southern California), and scroll through the plant’s Wikipedia page to learn additional current information about each plant queried. Not every plant has a native name that is provided in the literature of both tribes researched in this project. A plant may have a Cahuilla name and not a Chumash name, or vice versa. When a user creates a plant feature they can manually write in comments about what they think the native name is for a specific plant. Although many different coding methods were created and tested to implement the display of the table views and configure the appropriate detail window to appear for the selected native plants, the most appropriate methods were finally chosen and implemented in this study to address the original thesis objectives as well as follow the time constraints of completing this work in a timely manner.

The map window is where most of the app functionality resides. When it opens, the screen is loaded with a Light Gray Canvas background and the native plants feature class layers. The map extents are programmed to automatically center on the Southern California area. The user can double tap the screen to zoom in and pinch outward to zoom out of the map. Users can also pan the map by dragging their finger across the screen.

The zoom button in the upper left corner of the map window (looks like a bulls eye) can be touched to zoom into the user's location. This is a two-step process that can be seen in Figure 17. The map will pan over to the user's area and then zoom into their location as far as the map can be zoomed into (1:5,000 scale). The user's location is highlighted by a blue dot with a blue circle radiating around it to tell the user approximately where they are standing in the context of the map. The bigger the blue circle the less accurate the position is, and vice versa. From there the user can see what native plants have been geotagged near them. When the user hits the bull's eye button the "showCurrentLocation" method is called to automatically pan to the iPhone's location based on the internal GPS, the network connection, and the cellular connection (see Figure 18).

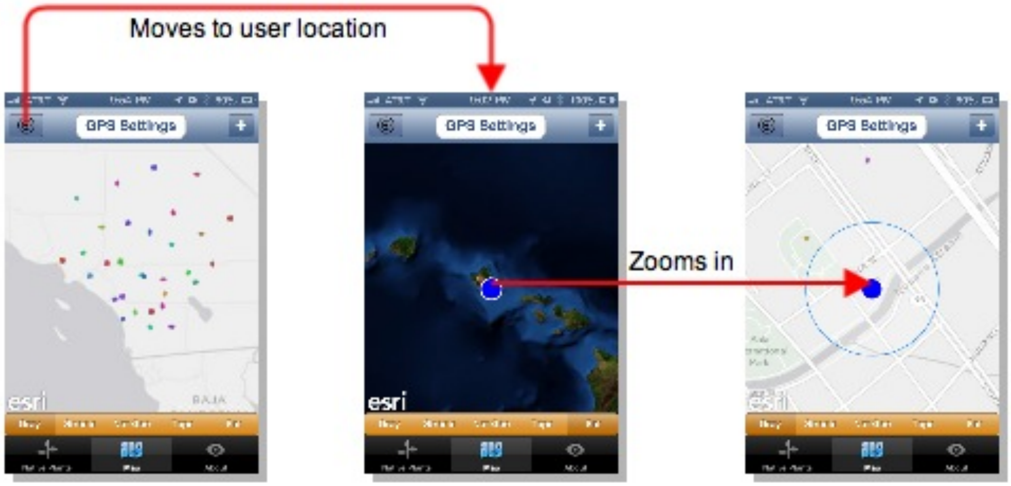
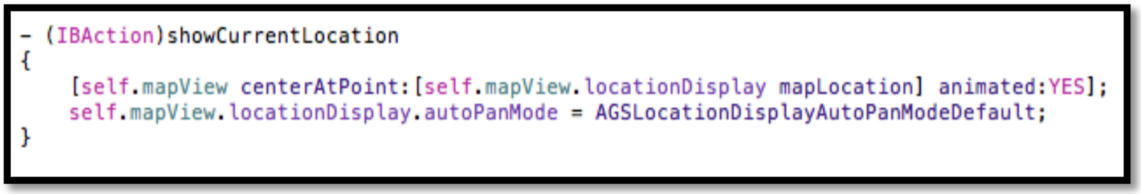


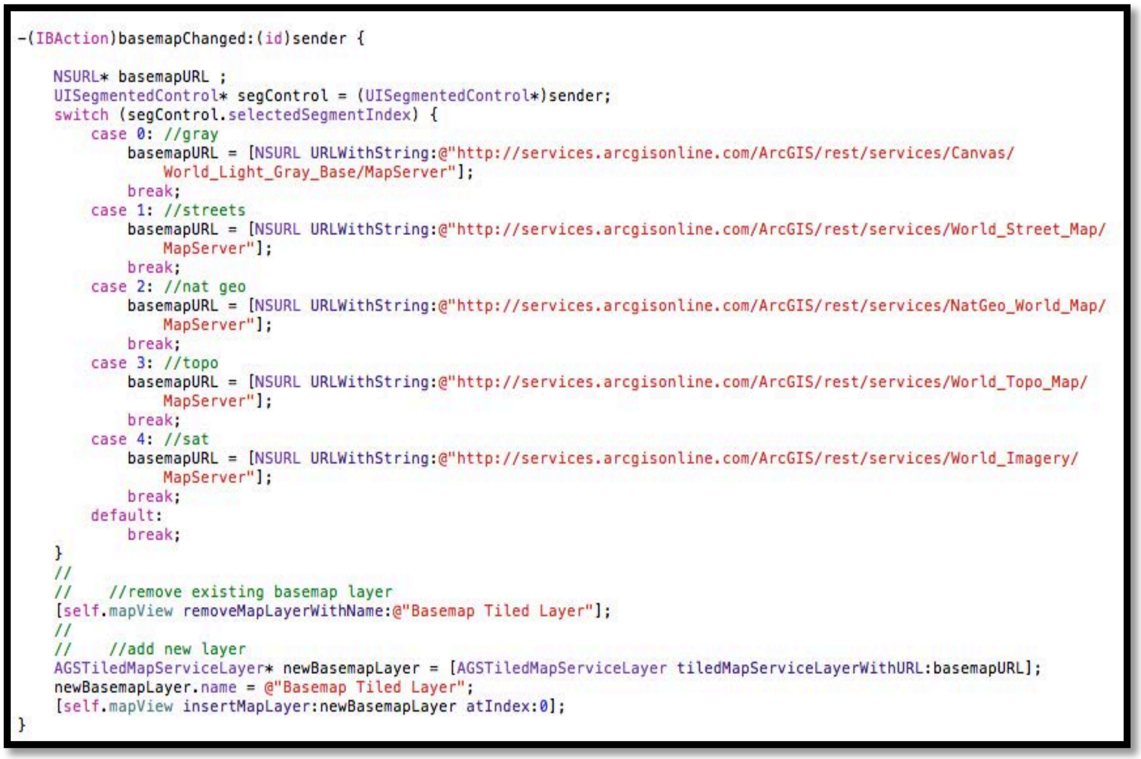
Figure 17: Zooming in to the User's Location



```
- (IBAction)showCurrentLocation
{
    [self.mapView centerAtPoint:[self.mapView.locationDisplay mapLocation] animated:YES];
    self.mapView.locationDisplay.autoPanMode = AGSLocationDisplayAutoPanModeDefault;
}
```

Figure 18: Code to Show the User's Location

At the bottom of the map window there is a switch bar that allows the user to switch from one type of background to the next. This action is performed by calling the “basemapChanged” method when the user chooses a different background in the switch bar (see Figure 19). Each button is given a number from 0 to 4 and when that button is pushed the basemap that represents that number in the code will appear and the previous basemap will be removed. Figure 20 shows the choices for basemaps that can be selected by the user, using the switch bar.



```
-(IBAction)basemapChanged:(id)sender {

    NSURL* basemapURL ;
    UISegmentedControl* segControl = (UISegmentedControl*)sender;
    switch (segControl.selectedSegmentIndex) {
        case 0: //gray
            basemapURL = [NSURL URLWithString:@"http://services.arcgisonline.com/ArcGIS/rest/services/Canvas/
                World_Light_Gray_Base/MapServer"];
            break;
        case 1: //streets
            basemapURL = [NSURL URLWithString:@"http://services.arcgisonline.com/ArcGIS/rest/services/World_Street_Map/
                MapServer"];
            break;
        case 2: //nat geo
            basemapURL = [NSURL URLWithString:@"http://services.arcgisonline.com/ArcGIS/rest/services/NatGeo_World_Map/
                MapServer"];
            break;
        case 3: //topo
            basemapURL = [NSURL URLWithString:@"http://services.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/
                MapServer"];
            break;
        case 4: //sat
            basemapURL = [NSURL URLWithString:@"http://services.arcgisonline.com/ArcGIS/rest/services/World_Imagery/
                MapServer"];
            break;
        default:
            break;
    }
    //
    //    //remove existing basemap layer
    [self.mapView removeMapLayerWithName:@"Basemap Tiled Layer"];
    //
    //    //add new layer
    AGSTiledMapServiceLayer* newBasemapLayer = [AGSTiledMapServiceLayer tiledMapServiceLayerWithURL:basemapURL];
    newBasemapLayer.name = @"Basemap Tiled Layer";
    [self.mapView insertMapLayer:newBasemapLayer atIndex:0];
}
```

Figure 19: Code to Switch the Basemaps

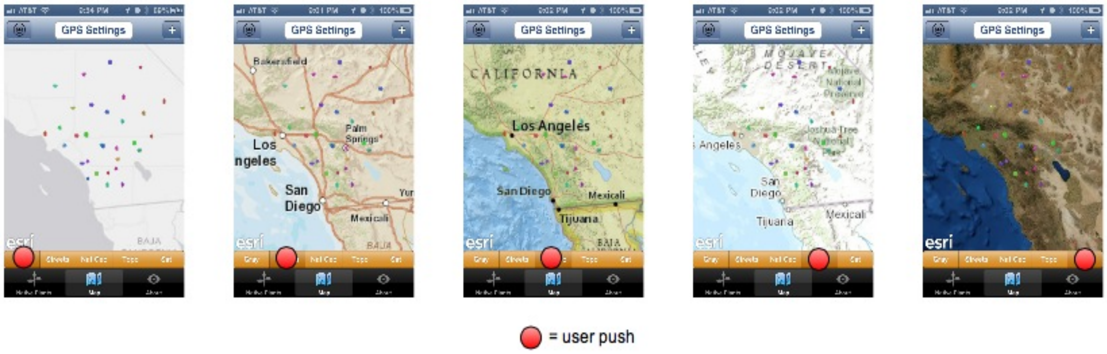
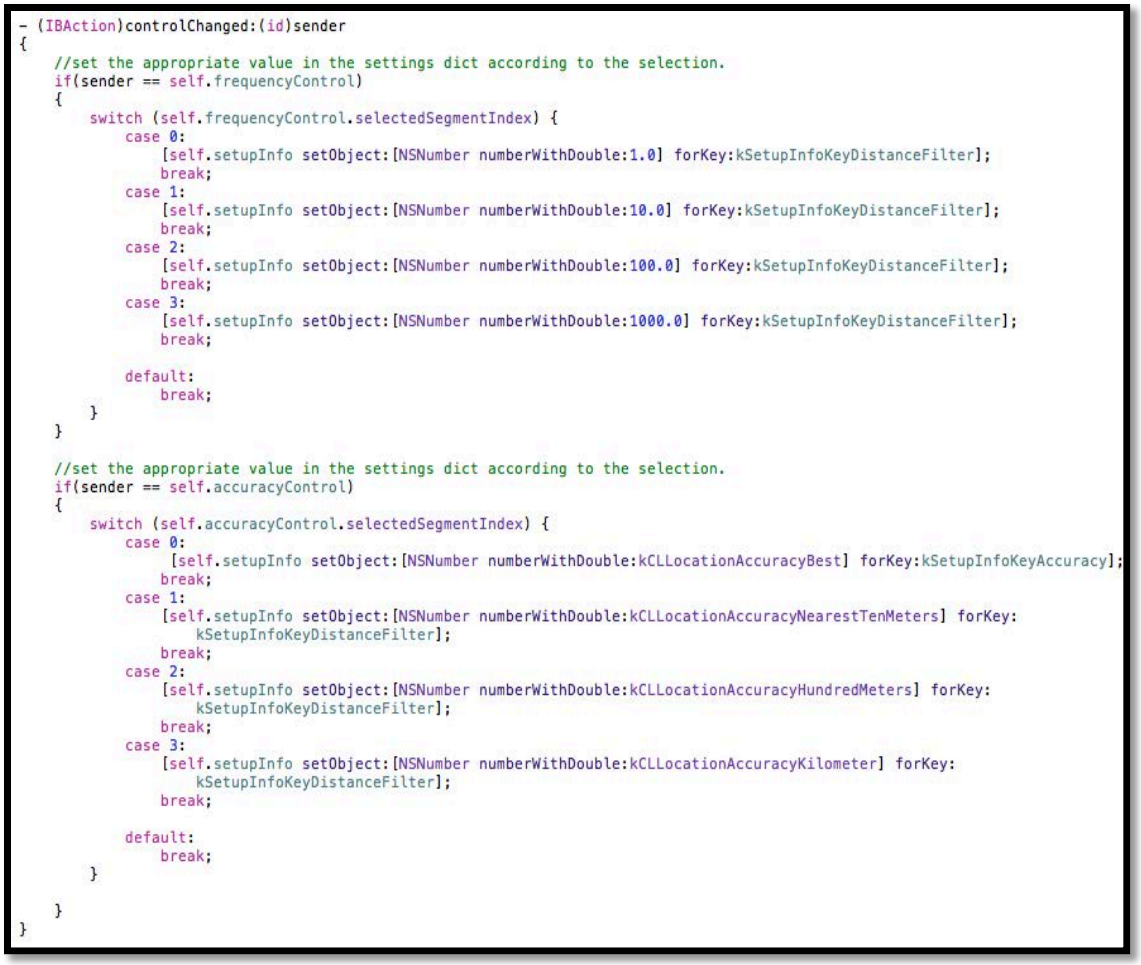


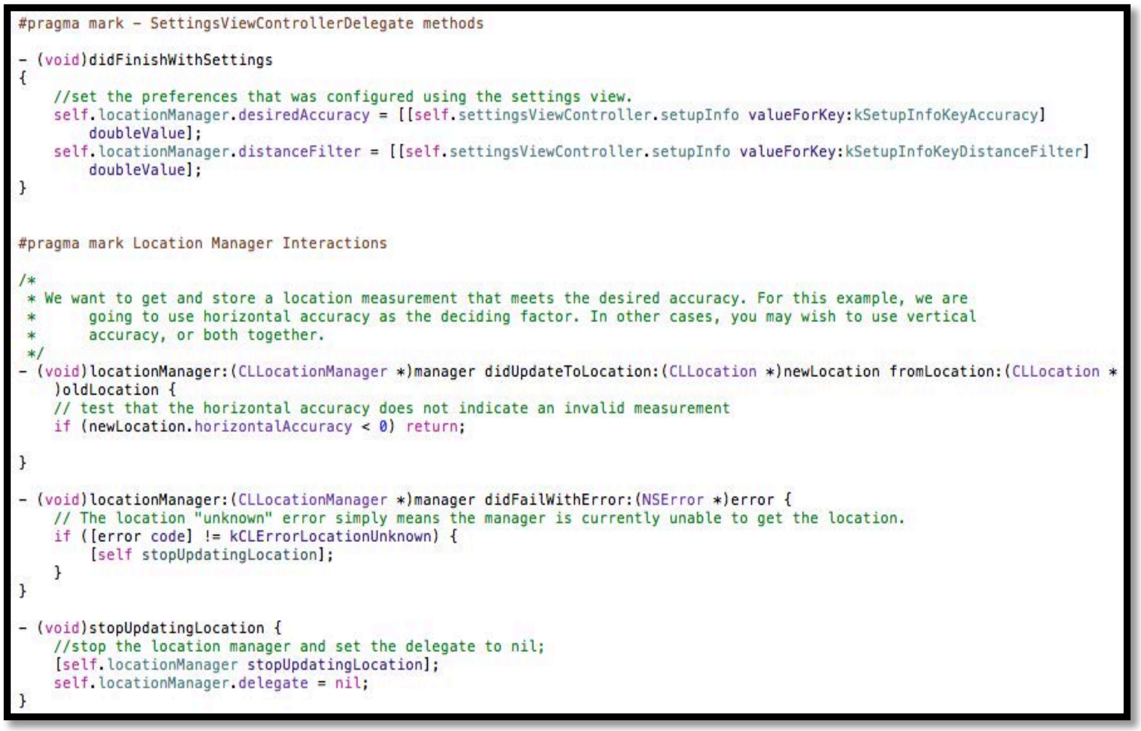
Figure 20: Switching the Basemaps

If the user is walking along a trail while using the app to try and find a certain plant that has already been geocoded, the user can change the settings of the GPS to update their location more or less frequently based on how far they walk. For example, by updating less frequently a user utilizes less power in the app and saves the battery life of the iPhone. They can also change the accuracy settings for the same reason, to use less battery. Both settings are set to default on the best accuracy and the shortest frequency, or highest energy consumption.



```
- (IBAction)controlChanged:(id)sender
{
    //set the appropriate value in the settings dict according to the selection.
    if(sender == self.frequencyControl)
    {
        switch (self.frequencyControl.selectedSegmentIndex) {
            case 0:
                [self.setupInfo setObject:[NSNumber numberWithDouble:1.0] forKey:kSetupInfoKeyDistanceFilter];
                break;
            case 1:
                [self.setupInfo setObject:[NSNumber numberWithDouble:10.0] forKey:kSetupInfoKeyDistanceFilter];
                break;
            case 2:
                [self.setupInfo setObject:[NSNumber numberWithDouble:100.0] forKey:kSetupInfoKeyDistanceFilter];
                break;
            case 3:
                [self.setupInfo setObject:[NSNumber numberWithDouble:1000.0] forKey:kSetupInfoKeyDistanceFilter];
                break;

            default:
                break;
        }
    }

    //set the appropriate value in the settings dict according to the selection.
    if(sender == self.accuracyControl)
    {
        switch (self.accuracyControl.selectedSegmentIndex) {
            case 0:
                 [self.setupInfo setObject:[NSNumber numberWithDouble:kCLLocationAccuracyBest] forKey:kSetupInfoKeyAccuracy];
                break;
            case 1:
                [self.setupInfo setObject:[NSNumber numberWithDouble:kCLLocationAccuracyNearestTenMeters] forKey:
                    kSetupInfoKeyDistanceFilter];
                break;
            case 2:
                [self.setupInfo setObject:[NSNumber numberWithDouble:kCLLocationAccuracyHundredMeters] forKey:
                    kSetupInfoKeyDistanceFilter];
                break;
            case 3:
                [self.setupInfo setObject:[NSNumber numberWithDouble:kCLLocationAccuracyKilometer] forKey:
                    kSetupInfoKeyDistanceFilter];
                break;

            default:
                break;
        }

    }
}
```

Figure 21: Code to implement the GPS settings pop-up, part I



```
#pragma mark - SettingsViewControllerDelegate methods

- (void)didFinishWithSettings
{
    //set the preferences that was configured using the settings view.
    self.locationManager.desiredAccuracy = [[self.settingsViewController.setupInfo valueForKey:kSetupInfoKeyAccuracy]
        doubleValue];
    self.locationManager.distanceFilter = [[self.settingsViewController.setupInfo valueForKey:kSetupInfoKeyDistanceFilter]
        doubleValue];
}


#pragma mark Location Manager Interactions

/*
 * We want to get and store a location measurement that meets the desired accuracy. For this example, we are
 *      going to use horizontal accuracy as the deciding factor. In other cases, you may wish to use vertical
 *      accuracy, or both together.
 */
- (void)locationManager:(CLLocationManager *)manager didUpdateToLocation:(CLLocation *)newLocation fromLocation:(CLLocation *
    )oldLocation {
    // test that the horizontal accuracy does not indicate an invalid measurement
    if (newLocation.horizontalAccuracy < 0) return;

}

- (void)locationManager:(CLLocationManager *)manager didFailWithError:(NSError *)error {
    // The location "unknown" error simply means the manager is currently unable to get the location.
    if ([error code] != kCLErrorLocationUnknown) {
        [self stopUpdatingLocation];
    }
}

- (void)stopUpdatingLocation {
    //stop the location manager and set the delegate to nil;
    [self.locationManager stopUpdatingLocation];
    self.locationManager.delegate = nil;
}
```

Figure 22: Code to implement the GPS settings pop-up, part II

In Figure 21 the accuracy and frequency buttons are assigned numbers associated with a distance filter (<10 meters (or Best), 10 meters, 100 meters, or 1 kilometer) through the “controlChanged” method. In the “didFinishWithSettings” method in Figure 22 these objects are delegated to the Settings View Controller that pops up when the GPS Settings button is pushed. Figure 23 provides the GPS Settings button and window.

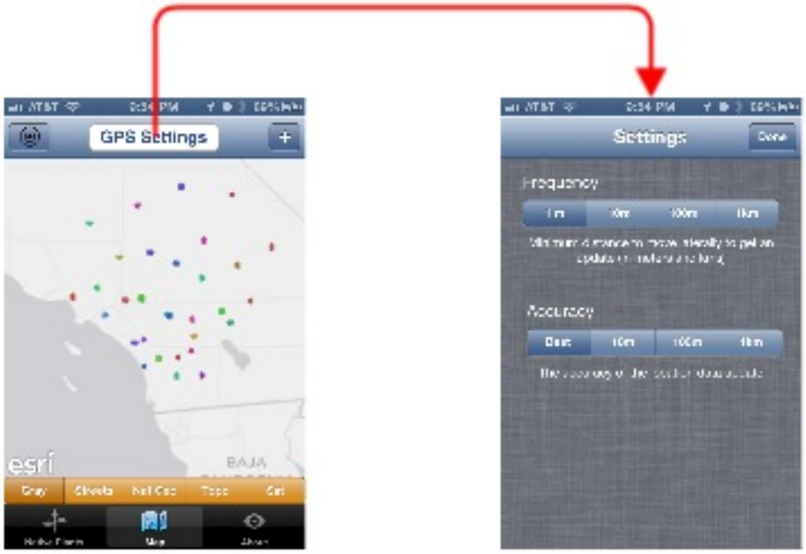
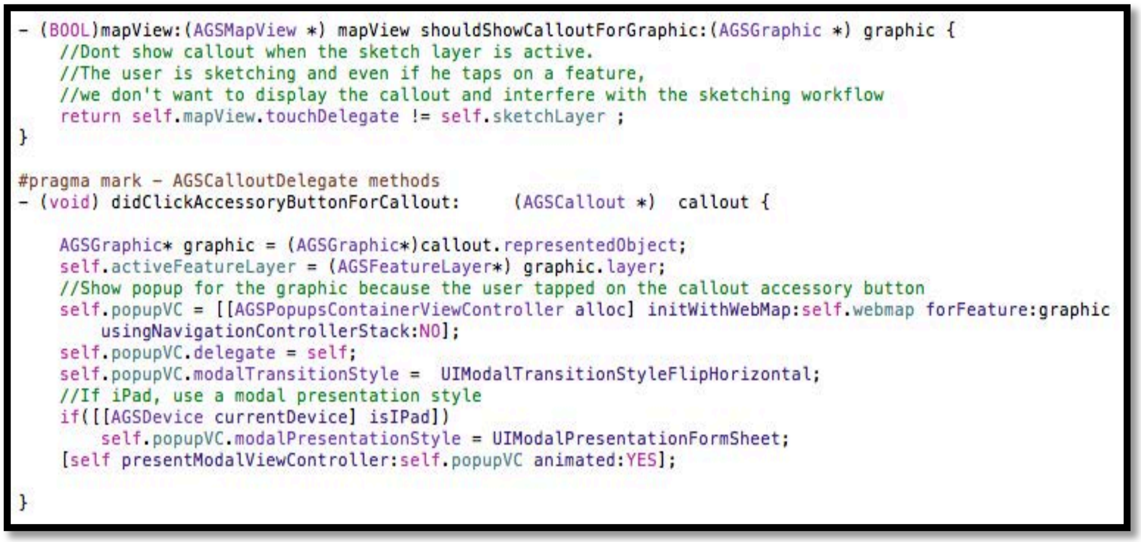


Figure 23: The GPS settings button and pop-up

Once the map appears and the native plants feature service is loaded the user can select one of the native plant features nearby and view their attributes. When a user touches the feature a small pop-up appears with a label showing either the comment or the creator's initials and the date it was last updated. If there is no comment it will show the initials for the user who created the feature, if there are no initials it will use the name of the plant. If the user wants to view the attributes of this existing native plant feature they can hit the accessory button in the label and the attribute table view for that feature will appear. In Figure 24 the "shouldShowCalloutForGraphic" method is called when the user touches an existing native plant feature on the map and the "didClickAccessoryButtonForCallout" method is called when they hit the accessory button inside the callout.

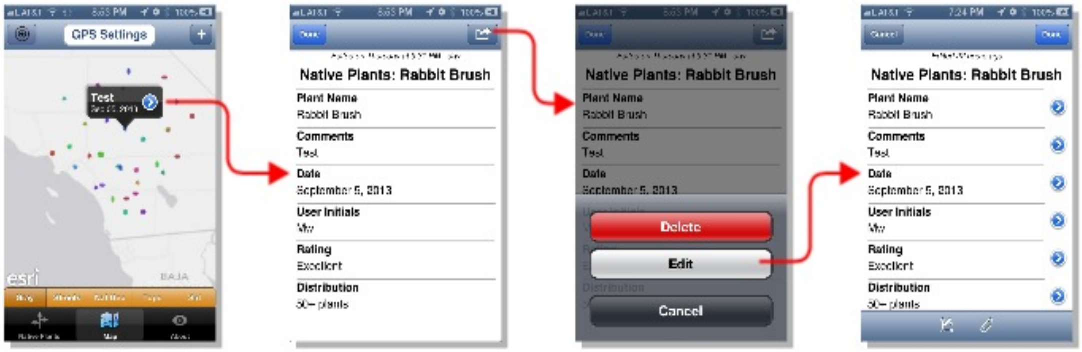


```
- (BOOL)mapView:(AGSMapView *) mapView shouldShowCalloutForGraphic:(AGSGraphic *) graphic {
    //Dont show callout when the sketch layer is active.
    //The user is sketching and even if he taps on a feature,
    //we don't want to display the callout and interfere with the sketching workflow
    return self.mapView.touchDelegate != self.sketchLayer ;
}

#pragma mark - AGSCalloutDelegate methods
- (void) didClickAccessoryButtonForCallout:     (AGSCallout *)  callout {

    AGSGraphic* graphic = (AGSGraphic*)callout.representedObject;
    self.activeFeatureLayer = (AGSFeatureLayer*) graphic.layer;
    //Show popup for the graphic because the user tapped on the callout accessory button
    self.popupVC = [[AGSPopupsContainerViewController alloc] initWithWebMap:self.webmap forFeature:graphic
        usingNavigationControllerStack:NO];
    self.popupVC.delegate = self;
    self.popupVC.modalTransitionStyle =  UIModalTransitionStyleFlipHorizontal;
    //If iPad, use a modal presentation style
    if([[AGSDevice currentDevice] isIPad])
        self.popupVC.modalPresentationStyle = UIModalPresentationFormSheet;
    [self presentModalViewController:self.popupVC animated:YES];

}
```

Figure 24: Code to select a feature in the map window

After the attribute pop-up has loaded the user can view the attribute data for the existing feature they selected and can press the option button if they want to delete or edit the feature, or cancel out of the option window. If the current user deletes the native plant feature it will no longer exist on the server. If the current user chooses to edit the feature then accessory buttons will appear next to all the plant attributes and the user can go through each one of them and change the feature’s attributes. The current user can also change the feature’s position on the map, view the photo attachments, and add new photos for the same feature. Figure 25 shows the process a user would follow for editing a feature in the map.

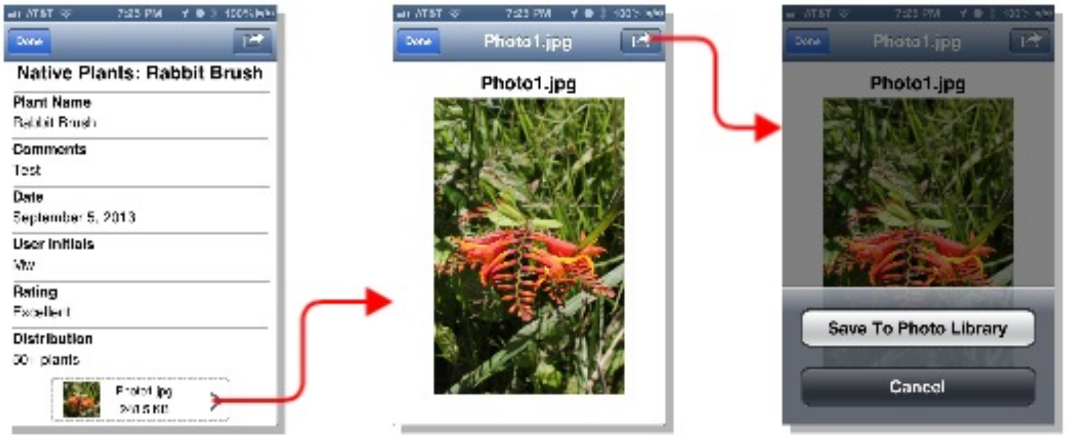
There are many methods coded into the map window that are shared between the “adding a native plant to the map” function and the “editing during selection” function because both functions are essentially doing the same thing: updating a feature in the feature service and syncing it to the server. Once edits are committed to the server there is no undo function. At this time, the only way to undo a user’s edits is for the database administrator to revert to an older version of the geodatabase that existed before the new edits were made. Possible improvements to this process, such as email notifications for the original native plant feature creator concerning edits or deletions or their data, are discussed in Chapter 6.



Plant Name
Rabbit Brush
Comments
Test
Date
September 5, 2013
User Initials
Mw
Rating
Excellent
Distribution
50+ plants

Figure 25: Editing an existing feature

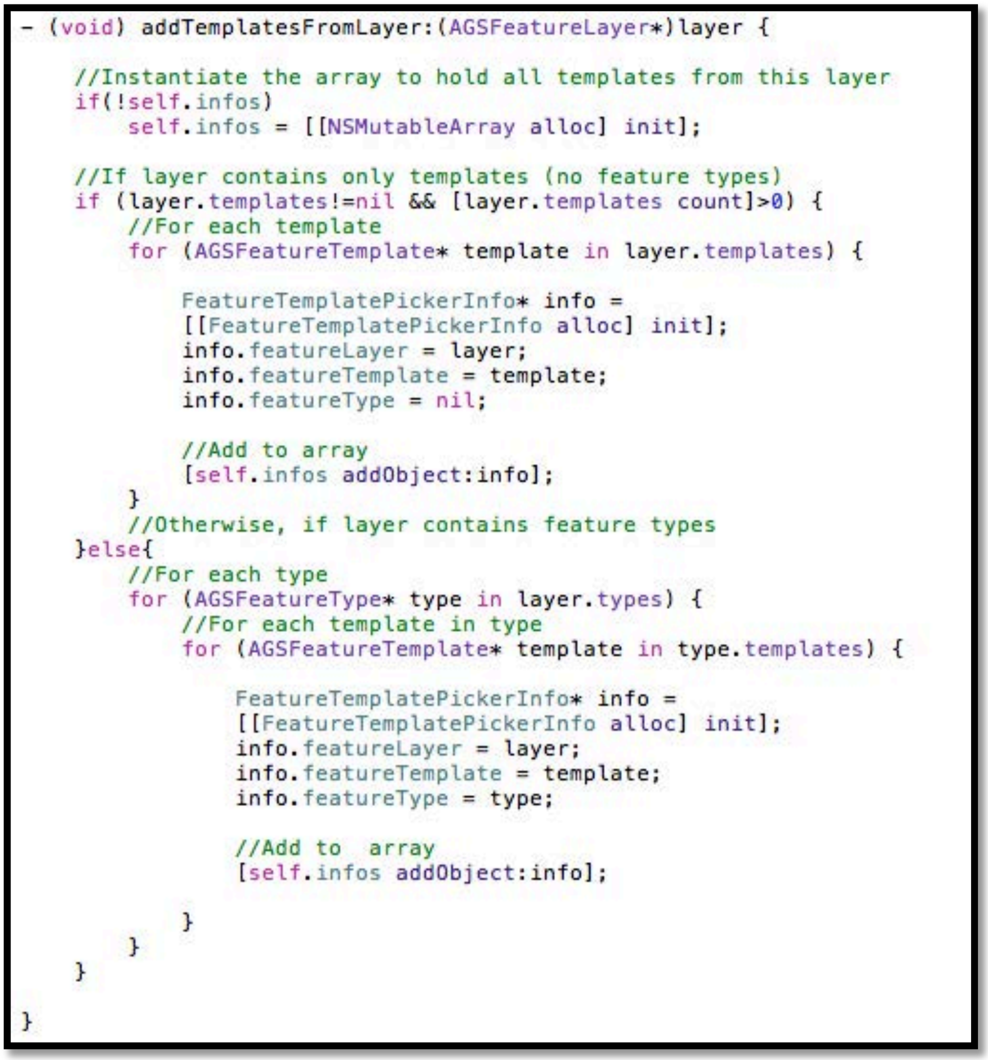
The user can also view the photo attachments of a feature and save those photos to their own photo library in the iPhone as shown in Figure 26.



Plant Name
Rabbit Brush
Comments
Test
Date
September 5, 2013
User Initials
Mw
Rating
Excellent
Distribution
50+ plants

Figure 26: Saving a photo attachment to the photo library

When users in the field come across a native plant they would like to geotag, they can first identify the plant using the native plant list to make sure they are geotagging the correct type of plant. Then to create a new native plant feature on the map, they can geotag the native plant in the map window by pressing the add button in the upper right corner. This will open up a picker window with a list of the native plants for them to choose the type of plant they have found. The code shown in Figure 27 depicts how the templates from the feature class are selected and imported into the template picker window:



```
- (void) addTemplatesFromLayer:(AGSFeatureLayer*)layer {

    //Instantiate the array to hold all templates from this layer
    if(!self.infos)
        self.infos = [[NSMutableArray alloc] init];

    //If layer contains only templates (no feature types)
    if (layer.templates!=nil && [layer.templates count]>0) {
        //For each template
        for (AGSFeatureTemplate* template in layer.templates) {

            FeatureTemplatePickerInfo* info =
            [[FeatureTemplatePickerInfo alloc] init];
            info.featureLayer = layer;
            info.featureTemplate = template;
            info.featureType = nil;

            //Add to array
            [self.infos addObject:info];
        }
        //Otherwise, if layer contains feature types
    }else{
        //For each type
        for (AGSFeatureType* type in layer.types) {
            //For each template in type
            for (AGSFeatureTemplate* template in type.templates) {

                FeatureTemplatePickerInfo* info =
                [[FeatureTemplatePickerInfo alloc] init];
                info.featureLayer = layer;
                info.featureTemplate = template;
                info.featureType = type;

                //Add to  array
                [self.infos addObject:info];

            }
        }
    }

}
```

Figure 27: Code to create the template picker

After a template is selected an attribute window with the native plant name filled in and the rest of the attribute fields that the user may fill in will appear. If the user had chosen the wrong plant name they can still change it in the attribute window by pressing the accessory button next to the plant name. The user can leave a comment, select what date it is from the date picker, enter their initials, rate the quality of the geotag, and select how widely the plant is distributed in that area.

Once the user is finished updating the attributes they can select the feature attachments button at the bottom and add one or more photos of the plant they have geotagged. Then the user can tap the geometry button to select the place on the map where the plant is located. The geometry button switches back to the map view and the user touches or taps the place where they want the point to be on the map. When this user has selected or tapped the spot, a red dot will appear in that location. The user then must tap the “Sketch Done” button to return to the attribute window. If the user is satisfied with the entry they can then tap the “Done” button in the attribute window and a couple of transparent windows will appear, one after the other, telling them the feature details and feature attachments have been synced with the server. If there are any problems saving the feature data to the server an error message will pop up telling them to try again. See Figure 28 for a diagram of the entire editing workflow illustrating the process of a user adding a new native plant feature to the map and thus to the database on the server.

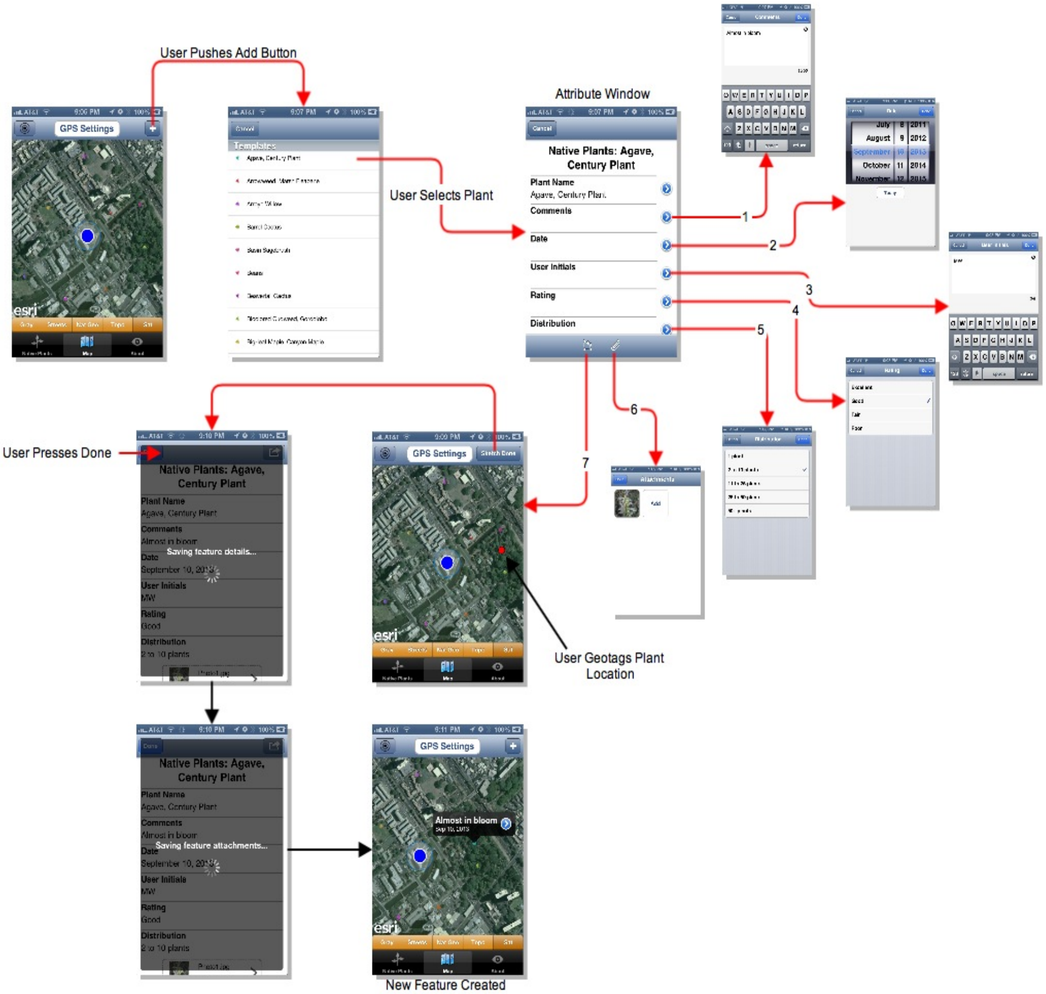
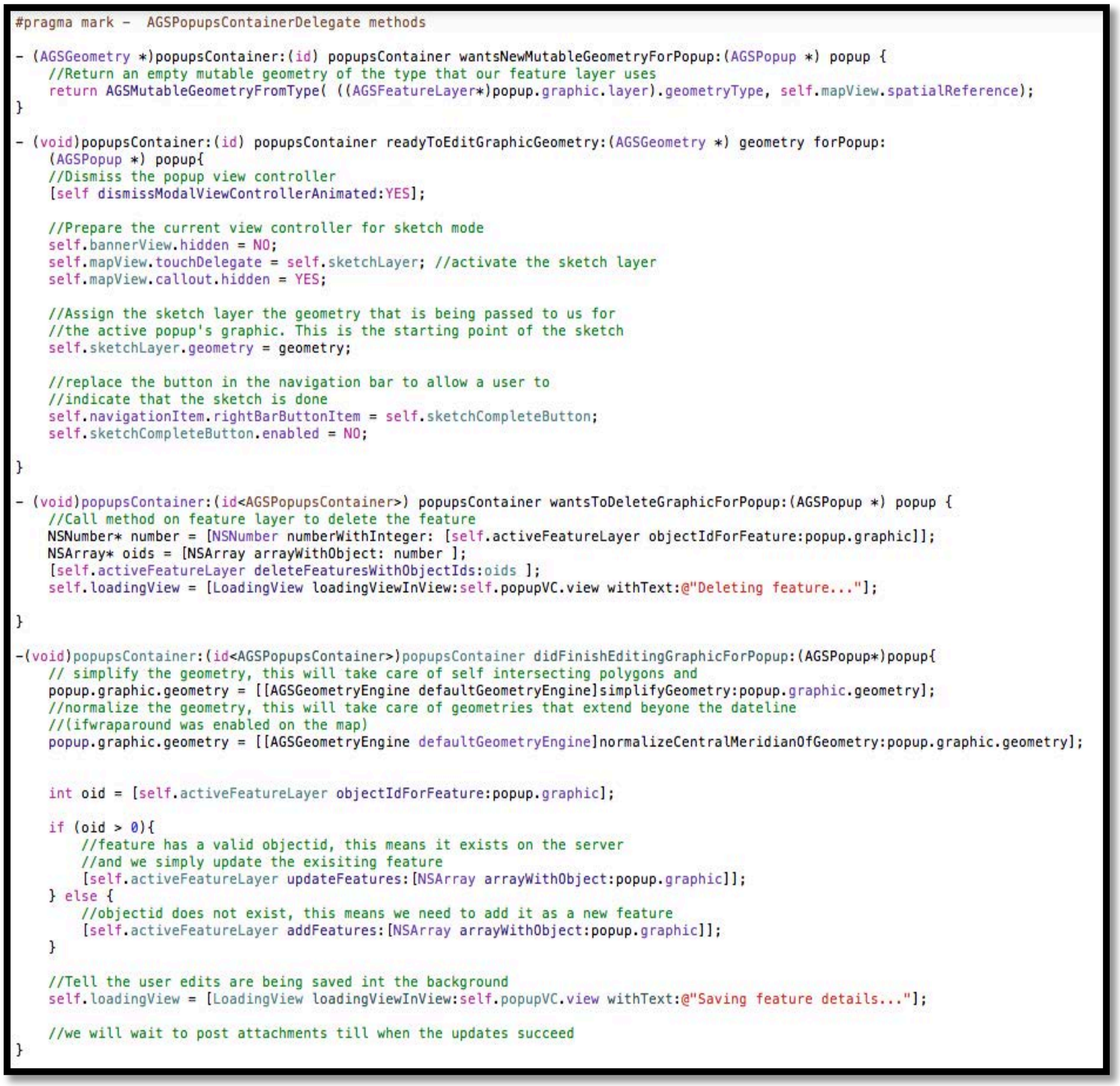


Figure 28: Creating a new feature

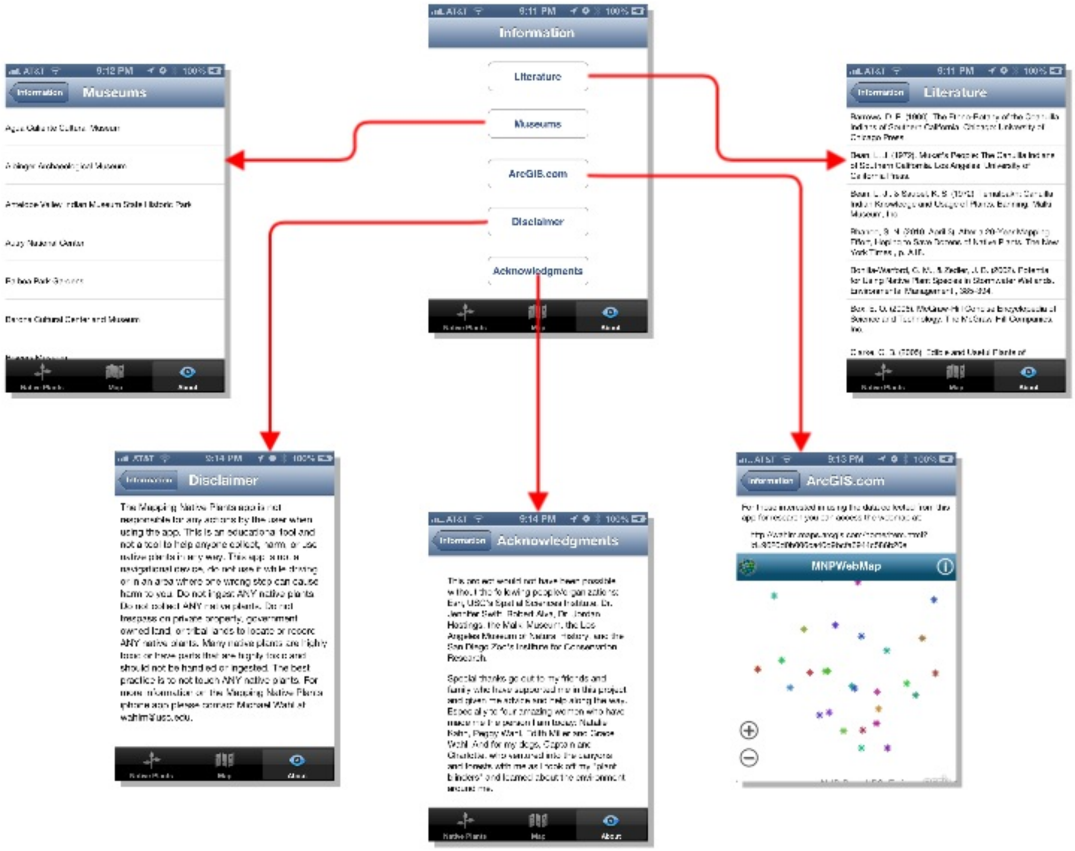
The map tab contains the majority of the code written for the Mapping Native Plants app. Each function is interconnected and there are multiple objects and methods to be called for the app to run properly. For example, the code in Figure 29 was written to tell the app how to behave when feature editing has commenced. A “mutable geometry” is an editable feature point and is part of the overlaying sketch layer that has not been copied to the feature service yet (the red dot). The feature callouts have been disabled so the user doesn’t accidentally open up a callout (a selection function) instead of geotagging a plant. And once a point has been made the sketch complete button will appear so the user can go back to the attribute window to make any further attribute changes. When the done button is tapped in the attribute window the sketch layer copy is added to the native plants feature service and the web map is automatically updated. The second method allows the user to delete a feature while the third method tells the app to show the transparent pop-up windows notifying the user that it is saving the features details and to wait until the native plant feature and its attachments have been saved, or synced, to the database on the server.



```
#pragma mark -  AGSPopupsContainerDelegate methods

- (AGSGeometry *)popupsContainer:(id) popupsContainer wantsNewMutableGeometryForPopup:(AGSPopup *) popup {
    //Return an empty mutable geometry of the type that our feature layer uses
    return AGSMutableGeometryFromType( ((AGSFeatureLayer*)popup.graphic.layer).geometryType, self.mapView.spatialReference);
}

- (void)popupsContainer:(id) popupsContainer readyToEditGraphicGeometry:(AGSGeometry *) geometry forPopup:
    (AGSPopup *) popup{
    //Dismiss the popup view controller
    [self dismissModalViewControllerAnimated:YES];

    //Prepare the current view controller for sketch mode
    self.bannerView.hidden = NO;
    self.mapView.touchDelegate = self.sketchLayer; //activate the sketch layer
    self.mapView.callout.hidden = YES;

    //Assign the sketch layer the geometry that is being passed to us for
    //the active popup's graphic. This is the starting point of the sketch
    self.sketchLayer.geometry = geometry;

    //replace the button in the navigation bar to allow a user to
    //indicate that the sketch is done
    self.navigationItem.rightBarButtonItem = self.sketchCompleteButton;
    self.sketchCompleteButton.enabled = NO;

}

- (void)popupsContainer:(id<AGSPopupsContainer>) popupsContainer wantsToDeleteGraphicForPopup:(AGSPopup *) popup {
    //Call method on feature layer to delete the feature
    NSNumber* number = [NSNumber numberWithInteger: [self.activeFeatureLayer objectIdForFeature:popup.graphic]];
    NSArray* oids = [NSArray arrayWithObject: number ];
    [self.activeFeatureLayer deleteFeaturesWithObjectIds:oids ];
    self.loadingView = [LoadingView loadingViewInView:self.popupVC.view withText:@"Deleting feature..."];

}

-(void)popupsContainer:(id<AGSPopupsContainer>)popupsContainer didFinishEditingGraphicForPopup:(AGSPopup*)popup{
    // simplify the geometry, this will take care of self intersecting polygons and
    popup.graphic.geometry = [[AGSGeometryEngine defaultGeometryEngine]simplifyGeometry:popup.graphic.geometry];
    //normalize the geometry, this will take care of geometries that extend beyone the dateline
    //(ifwraparound was enabled on the map)
    popup.graphic.geometry = [[AGSGeometryEngine defaultGeometryEngine]normalizeCentralMeridianOfGeometry:popup.graphic.geometry];


    int oid = [self.activeFeatureLayer objectIdForFeature:popup.graphic];

    if (oid > 0){
        //feature has a valid objectid, this means it exists on the server
        //and we simply update the exisiting feature
        [self.activeFeatureLayer updateFeatures:[NSArray arrayWithObject:popup.graphic]];
    } else {
        //objectid does not exist, this means we need to add it as a new feature
        [self.activeFeatureLayer addFeatures:[NSArray arrayWithObject:popup.graphic]];
    }

    //Tell the user edits are being saved int the background
    self.loadingView = [LoadingView loadingViewInView:self.popupVC.view withText:@"Saving feature details..."];

    //we will wait to post attachments till when the updates succeed
}
```

Figure 29: Code to sync a feature to the server

The About tab is the simplest part of the Mapping Native Plants app. Figure 30 shows that it consists of a view controller with several buttons connected to pop-ups. The pop-ups contain information about the native plants and native cultures in Southern California, a legal disclaimer for the app, a link to the app web map on ArcGIS Online, and the acknowledgements of people who have participated in and supported this project.



Agua Caliente Cultural Museum

A.[illegible] Archaeological Museum

Antelope Valley Indian Museum State Historic Park

[illegible] Nature Center

Balboa Park Gardens

Barona Cultural Center and Museum

Barrows, D. P. (1900). The Ethno-Botany of the Coahuilla Indians of Southern California. Chicago: University of Chicago Press.

Bean, L. J. (1972). Mukat's People: The Cahuilla Indians of Southern California. Los Angeles: University of California Press.

Bean, L. J., & Saubel, K. S. (1972). Temalpakh: Cahuilla Indian Knowledge and Usage of Plants. Banning: Malki Museum, Inc.

[illegible], S. N. (2010, April 5). After a 20-Year Mapping Effort, Hope to Save Dozens of Native Plants. The New York Times, p. A16.

Bonilla-Warford, C. M., & Zedler, J. B. (2002). Potential for Using Native Plant Species in Stormwater Wetlands. Environmental Management, 385-394.

[illegible] (2005). McGraw-Hill Concise Encyclopedia of Science and Technology. The McGraw-Hill Companies, Inc.

Clarke, C. B. (2006). Edible and Useful Plants of

The Mapping Native Plants app is not responsible for any actions by the user when using the app. This is an educational tool and not a tool to help anyone collect, harm, or use native plants in any way. This app is not a navigational device, do not use it while driving or in an area where one wrong step can cause harm to you. Do not ingest ANY native plants. Do not collect ANY native plants. Do not trespass on private property, government owned land, or tribal lands to locate or record ANY native plants. Many native plants are highly toxic or have parts that are highly toxic and should not be handled or ingested. The best practice is to not touch ANY native plants. For more information on the Mapping Native Plants iphone app please contact Michael Wahl at wahlm@usc.edu.

This project would not have been possible without the following people/organizations: Esri, USC's Spatial Sciences Institute, Dr. Jennifer Swift, Robert Alva, Dr. Jordan Hastings, the Malki Museum, the Los Angeles Museum of Natural History, and the San Diego Zoo's Institute for Conservation Research.

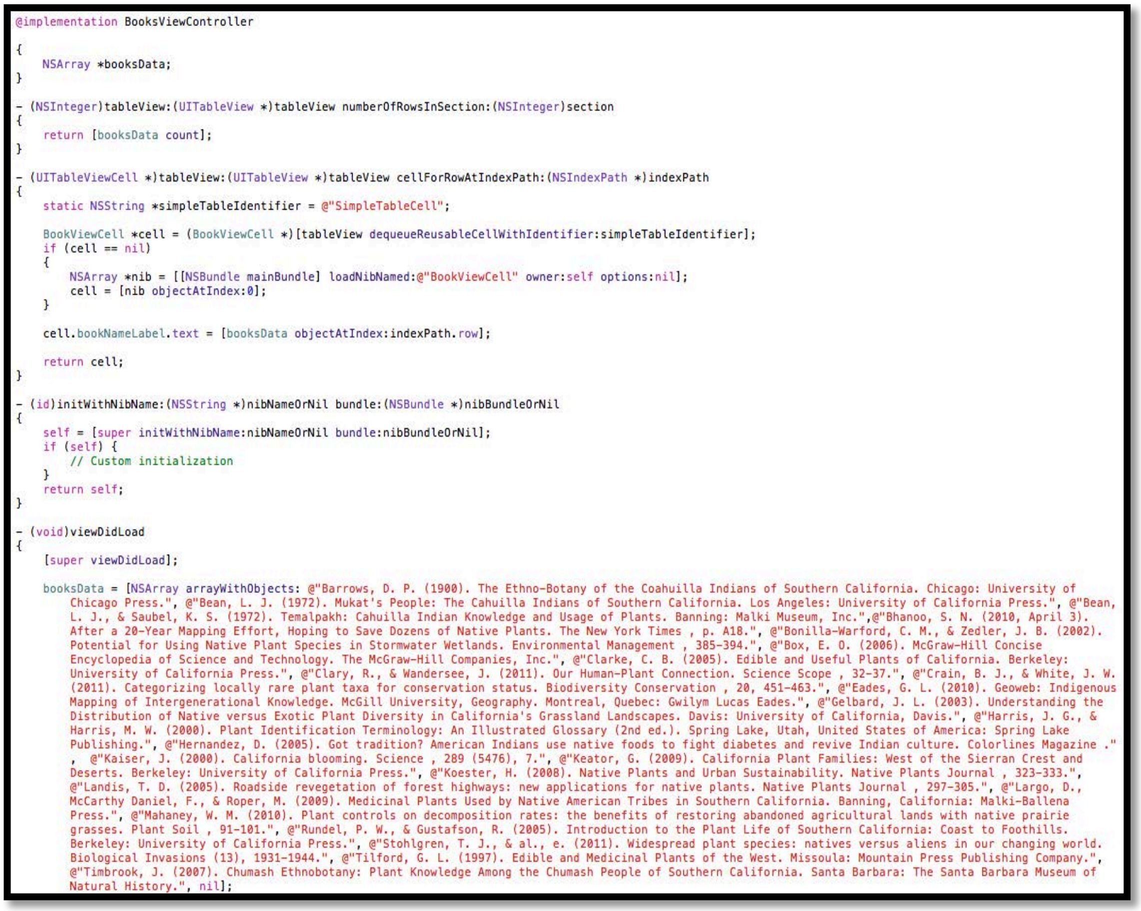
Special thanks go out to my friends and family who have supported me in this project and given me advice and help along the way. Especially to four amazing women who have made me the person I am today: Natalie Kahn, Peggy Wahl, Edith Miller and Grace Wahl. And for my dogs, Captain and Charlotte, who ventured into the canyons and forests with me as I took off my 'plant blinders' and learned about the environment around me.

For those interested in using the data collected from this app for research you can access the webmap at:

http://[illegible].maps.arcgis.com/home/item.html?id=[illegible]

Figure 30: The About Tab user interface

The Literature pop-up is a table view with a list of all the references used to create the native plant list. In the code in Figure 31 an array is created of all the references used in the app and a table is constructed to hold them. In the future there may be a detail window for each reference to guide the user to a web page where they can purchase or print such reference resources.

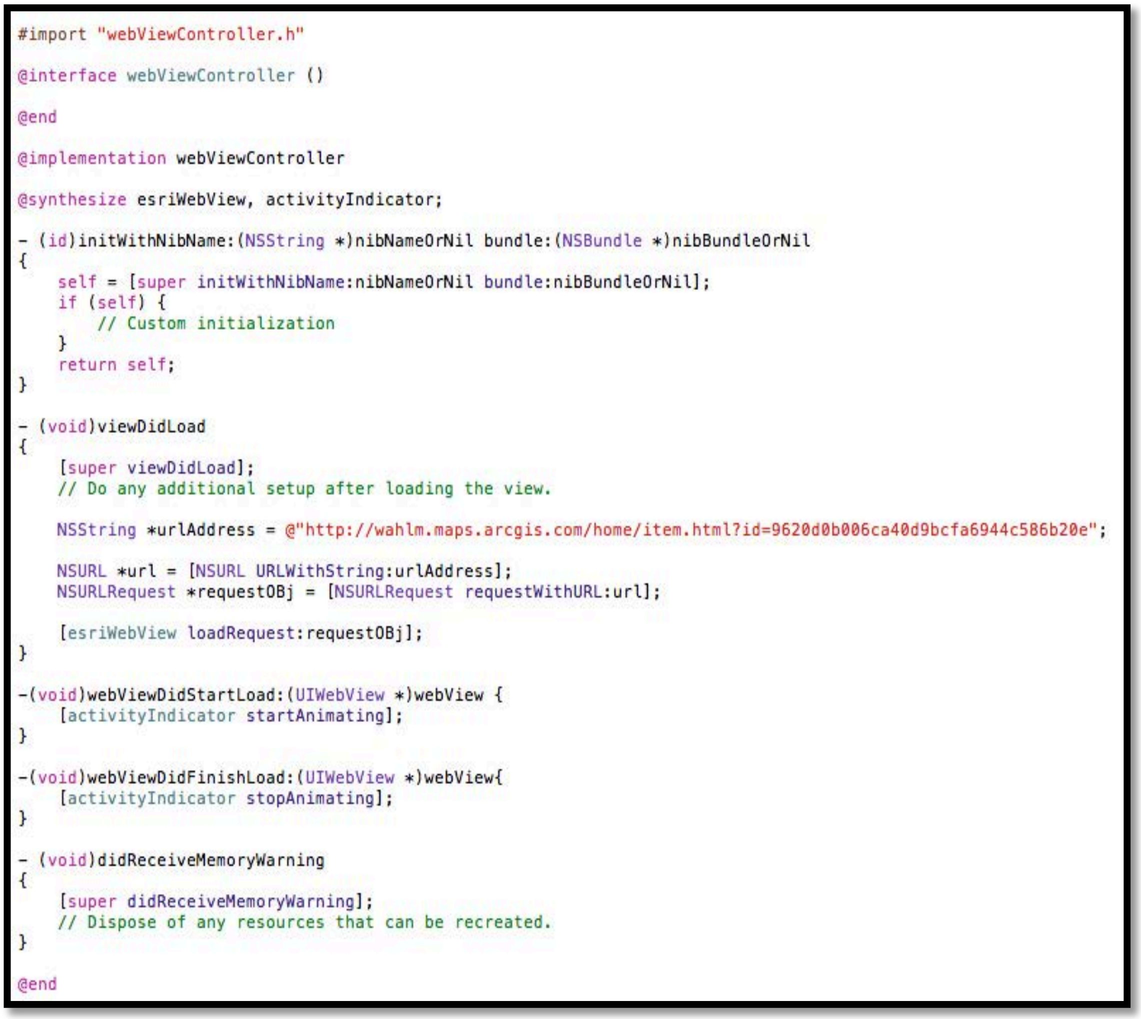


```
@implementation BooksViewController

{
    NSArray *booksData;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
    return [booksData count];
}

- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    static NSString *simpleTableIdentifier = @"SimpleTableCell";

    BookViewCell *cell = (BookViewCell *)[tableView dequeueReusableCellWithIdentifier:simpleTableIdentifier];
    if (cell == nil)
    {
        NSArray *nib = [[NSBundle mainBundle] loadNibNamed:@"BookViewCell" owner:self options:nil];
        cell = [nib objectAtIndex:0];
    }

    cell.bookNameLabel.text = [booksData objectAtIndex:indexPath.row];

    return cell;
}

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
        // Custom initialization
    }
    return self;
}

- (void)viewDidLoad
{
    [super viewDidLoad];

    booksData = [NSArray arrayWithObjects: @"Barrows, D. P. (1900). The Ethno-Botany of the Coahuilla Indians of Southern California. Chicago: University of
        Chicago Press.", @"Bean, L. J. (1972). Mukat's People: The Cahuilla Indians of Southern California. Los Angeles: University of California Press.", @"Bean,
        L. J., & Saubel, K. S. (1972). Temalpakh: Cahuilla Indian Knowledge and Usage of Plants. Banning: Malki Museum, Inc.",@"Bhanoo, S. N. (2010, April 3).
        After a 20-Year Mapping Effort, Hoping to Save Dozens of Native Plants. The New York Times , p. A18.", @"Bonilla-Warford, C. M., & Zedler, J. B. (2002).
        Potential for Using Native Plant Species in Stormwater Wetlands. Environmental Management , 385-394.", @"Box, E. O. (2006). McGraw-Hill Concise
        Encyclopedia of Science and Technology. The McGraw-Hill Companies, Inc.", @"Clarke, C. B. (2005). Edible and Useful Plants of California. Berkeley:
        University of California Press.", @"Clary, R., & Wandersee, J. (2011). Our Human-Plant Connection. Science Scope , 32-37.", @"Crain, B. J., & White, J. W.
        (2011). Categorizing locally rare plant taxa for conservation status. Biodiversity Conservation , 20, 451-463.", @"Eades, G. L. (2010). Geoweb: Indigenous
        Mapping of Intergenerational Knowledge. McGill University, Geography. Montreal, Quebec: Gwilym Lucas Eades.", @"Gelbard, J. L. (2003). Understanding the
        Distribution of Native versus Exotic Plant Diversity in California's Grassland Landscapes. Davis: University of California, Davis.", @"Harris, J. G., &
        Harris, M. W. (2000). Plant Identification Terminology: An Illustrated Glossary (2nd ed.). Spring Lake, Utah, United States of America: Spring Lake
        Publishing.", @"Hernandez, D. (2005). Got tradition? American Indians use native foods to fight diabetes and revive Indian culture. Colorlines Magazine ."
        ,  @"Kaiser, J. (2000). California blooming. Science , 289 (5476), 7.", @"Keator, G. (2009). California Plant Families: West of the Sierran Crest and
        Deserts. Berkeley: University of California Press.", @"Koester, H. (2008). Native Plants and Urban Sustainability. Native Plants Journal , 323-333.",
        @"Landis, T. D. (2005). Roadside revegetation of forest highways: new applications for native plants. Native Plants Journal , 297-305.", @"Largo, D.,
        McCarthy Daniel, F., & Roper, M. (2009). Medicinal Plants Used by Native American Tribes in Southern California. Banning, California: Malki-Ballena
        Press.", @"Mahaney, W. M. (2010). Plant controls on decomposition rates: the benefits of restoring abandoned agricultural lands with native prairie
        grasses. Plant Soil , 91-101.", @"Rundel, P. W., & Gustafson, R. (2005). Introduction to the Plant Life of Southern California: Coast to Foothills.
        Berkeley: University of California Press.", @"Stohlgren, T. J., & al., e. (2011). Widespread plant species: natives versus aliens in our changing world.
        Biological Invasions (13), 1931-1944.", @"Tilford, G. L. (1997). Edible and Medicinal Plants of the West. Missoula: Mountain Press Publishing Company.",
        @"Timbrook, J. (2007). Chumash Ethnobotany: Plant Knowledge Among the Chumash People of Southern California. Santa Barbara: The Santa Barbara Museum of
        Natural History.", nil];
```

Figure 31: Code to create the references data array

The same procedure as described above for the About tab is used for the Museums pop-up, but instead an array was created with the names of museums and botanical gardens in the Southern California area. In the future there may also be a detail window for each museum or botanical garden to show the user the museum web page, or the locations on the same or different map.

The ArcGIS.com pop-up is a text box that contains the URL for the web map and the rest URL for the native plants feature service. A web view was also added and the Mapping Native Plants webmap was embedded into it using the code provided in Figure 32.



```
#import "webViewController.h"

@interface webViewController ()

@end

@implementation webViewController

@synthesize esriWebView, activityIndicator;

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
        // Custom initialization
    }
    return self;
}

- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view.

    NSString *urlAddress = @"http://wahlm.maps.arcgis.com/home/item.html?id=9620d0b006ca40d9bcfa6944c586b20e";

    NSURL *url = [NSURL URLWithString:urlAddress];
    NSURLRequest *requestOBj = [NSURLRequest requestWithURL:url];

    [esriWebView loadRequest:requestOBj];
}

-(void)webViewDidStartLoad:(UIWebView *)webView {
    [activityIndicator startAnimating];
}

-(void)webViewDidFinishLoad:(UIWebView *)webView{
    [activityIndicator stopAnimating];
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end
```

Figure 32: Code to embed the web map in the web viewer

The web map is interactive inside the app’s web view. Features can be selected, their attributes viewed and the map can be panned and zoomed in and out. The background map in the web map has been made transparent because when it is added to the map window the background will overlay all the other backgrounds that could be chosen by the user through the switch bar control.

The Disclaimer and Acknowledgments pop-ups are both text views. The disclaimer text is as follows:

“The Mapping Native Plants app is not responsible for any actions by the user when using the app. This is an educational tool and not a tool to help collect, harm, or use native plants in any way. This app is not a navigational device, do not use it while driving or in an area where one wrong step can cause harm to you. Do not ingest ANY native plants. Do not collect ANY native plants. Do not trespass on private property, government owned land, or tribal lands to locate or record ANY native plants. Many native plants are highly toxic or have parts that are highly toxic and should not be handled or ingested. The best practice is to not touch ANY native plants. For more information on the Mapping Native Plants iPhone app please contact Michael Wahl at wahlm@usc.edu.”

And this is the text for the Acknowledgments:

“This project would not have been possible without the following people/organizations: Esri, USC's Spatial Sciences Institute, Dr. Jennifer Swift, Robert Alvarez, Dr. Jordan Hastings, the Malki Museum, the Los Angeles Museum of Natural History, and the San Diego Zoo's Institute for Conservation Research. Special thanks go out to my friends and family who have supported me in this project and given me advice and help along the way. Especially to four amazing women who have made me the person I am today: Natalie Kahn, Peggy Wahl, Edith Miller and Grace Wahl. And for my dogs, Captain and Charlotte, who ventured into the canyons and forests with me as I took off my "plant blinders" and learned about the environment around me.”

Both the Disclaimer and Acknowledgments pop-ups required no special coding and are purely user interface elements. Each pop-up contains a back button that allows the user to go back once they have viewed the pop-up.

The app functions described in this section have been run and built in the Xcode simulator without any errors or problems. The next step was to test the app functions in the real world and run through Xcode’s Instruments to determine if the app code is structured well enough to handle any exceptions and errors that might be generated when using in an iPhone.

5.2 Test Results: Functionality and Performance

After a week of travelling around Oahu and testing the app in different environments the field notes were compiled and calculations were made for the average times it took for most of the app functions to be executed. Some basic questions were also answered about its performance under various topographic and weather conditions.

The app was tested on an iPhone 4s using iOS 6.1.3 under these circumstances: connected to Wi-Fi, connected to the cellular network, and intermittent connectivity. All of these circumstances were tested using three different variables: 100%, 50%, and <20% battery life. The testing took place in several different locations: downtown Honolulu with tall buildings, a jungle forest with dense canopy, a canyon with steep walls, an open field, on the streets under power lines, in a moving car, and at the beach. The following is a list of basic questions that were answered by persons testing the app under each of these circumstances:

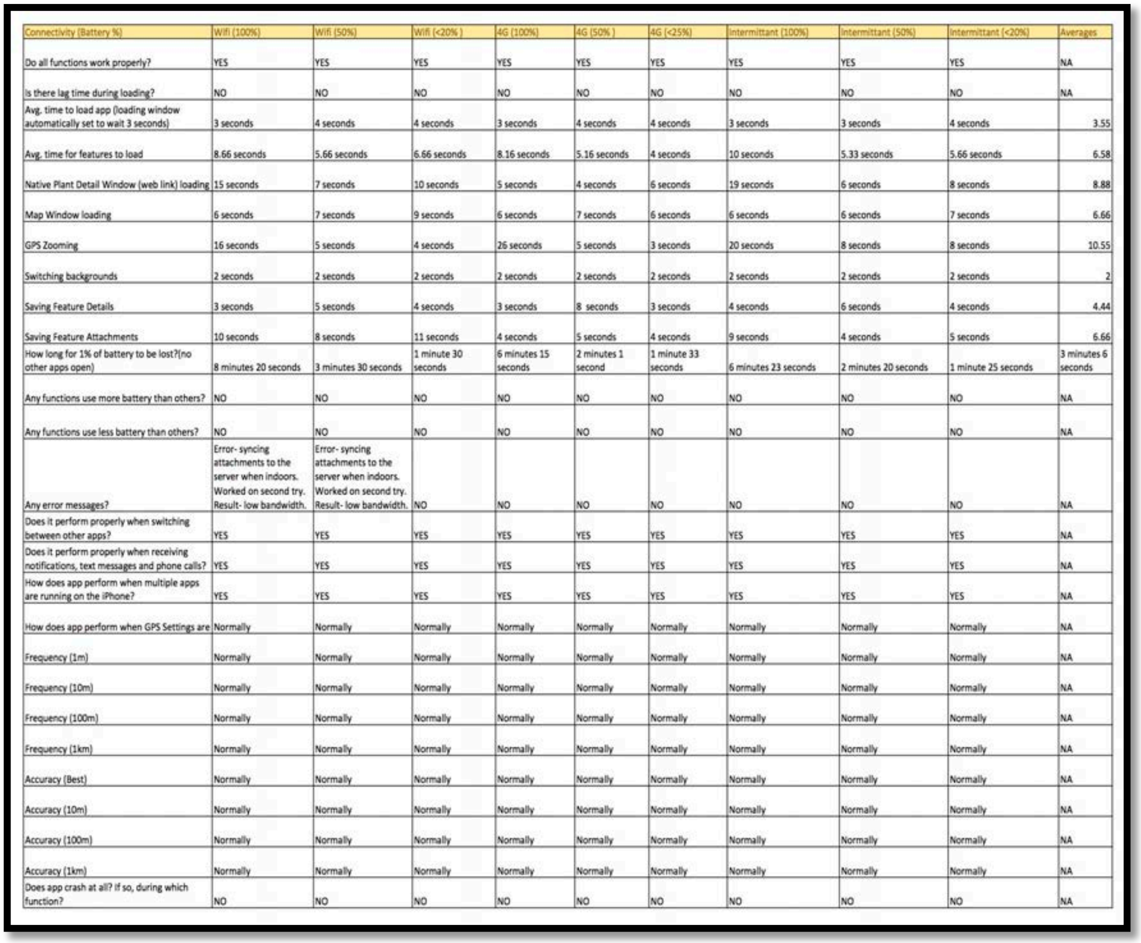
1. Do all functions work properly?

2. Is there any lag time during loading of the app?
3. What is the time it takes for the app to load?
4. What is the time it takes for these features to load:
    a. Native Plant Detail window?
    b. Map window?
    c. GPS zooming?
    d. Switching backgrounds?
    e. Saving feature details?
    f. Saving feature attachments?
5. How long does it take for 1% of battery life to expire when the app is being used?
6. Do any functions use more battery than others?
7. Do any functions use less battery than others?
8. Are there any error messages during use?
9. Does the app perform properly when switching between other apps?
10. Does the app perform properly when receiving notifications, text messages and phone calls?
11. Does the app perform properly when multiple apps are running on the iPhone?
12. Does the app perform properly when GPS settings are changed?
    a. Frequency (1m, 10m, 100m, 1km)?
    b. Accuracy (Best, 10m, 100m, 1km)?
13. Does the app crash at all, if so during which function?

The testing results were then compiled from the field test notes into the spreadsheet as shown in Table 1. The average time it took to perform each function was then calculated.

The table entries include these averages of the times it took to perform each function as recorded in each of the different locations the app was tested in.

Table 1: Functionality and Performance Test Results



Connectivity (Battery %)	Wifi (100%)	Wifi (50%)	Wifi (<20%)	4G (100%)	4G (50%)	4G (<25%)	Intermittant (100%)	Intermittant (50%)	Intermittant (<20%)	Averages
Do all functions work properly?	YES	YES	YES	YES	YES	YES	YES	YES	YES	NA
Is there lag time during loading?	NO	NO	NO	NO	NO	NO	NO	NO	NO	NA
Avg. time to load app (loading window automatically set to wait 3 seconds)	3 seconds	4 seconds	4 seconds	3 seconds	4 seconds	4 seconds	3 seconds	3 seconds	4 seconds	3.55
Avg. time for features to load	8.66 seconds	5.66 seconds	6.66 seconds	8.16 seconds	5.16 seconds	4 seconds	10 seconds	5.33 seconds	5.66 seconds	6.58
Native Plant Detail Window (web link) loading	15 seconds	7 seconds	10 seconds	5 seconds	4 seconds	6 seconds	19 seconds	6 seconds	8 seconds	8.88
Map Window loading	6 seconds	7 seconds	9 seconds	6 seconds	7 seconds	6 seconds	6 seconds	6 seconds	7 seconds	6.66
GPS Zooming	16 seconds	5 seconds	4 seconds	26 seconds	5 seconds	3 seconds	20 seconds	8 seconds	8 seconds	10.55
Switching backgrounds	2 seconds	2 seconds	2 seconds	2 seconds	2 seconds	2 seconds	2 seconds	2 seconds	2 seconds	2
Saving Feature Details	3 seconds	5 seconds	4 seconds	3 seconds	8 seconds	3 seconds	4 seconds	6 seconds	4 seconds	4.44
Saving Feature Attachments	10 seconds	8 seconds	11 seconds	4 seconds	5 seconds	4 seconds	9 seconds	4 seconds	5 seconds	6.66
How long for 1% of battery to be lost?(no other apps open)	8 minutes 20 seconds	3 minutes 30 seconds	1 minute 30 seconds	6 minutes 15 seconds	2 minutes 1 second	1 minute 33 seconds	6 minutes 23 seconds	2 minutes 20 seconds	1 minute 25 seconds	3 minutes 6 seconds
Any functions use more battery than others?	NO	NO	NO	NO	NO	NO	NO	NO	NO	NA
Any functions use less battery than others?	NO	NO	NO	NO	NO	NO	NO	NO	NO	NA
Any error messages?	Error- syncing attachments to the server when indoors. Worked on second try. Result- low bandwidth.	Error- syncing attachments to the server when indoors. Worked on second try. Result- low bandwidth.	NO	NO	NO	NO	NO	NO	NO	NA
Does it perform properly when switching between other apps?	YES	YES	YES	YES	YES	YES	YES	YES	YES	NA
Does it perform properly when receiving notifications, text messages and phone calls?	YES	YES	YES	YES	YES	YES	YES	YES	YES	NA
How does app perform when multiple apps are running on the iPhone?	YES	YES	YES	YES	YES	YES	YES	YES	YES	NA
How does app perform when GPS Settings are	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Frequency (1m)	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Frequency (10m)	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Frequency (100m)	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Frequency (1km)	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Accuracy (Best)	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Accuracy (10m)	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Accuracy (100m)	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Accuracy (1km)	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	Normally	NA
Does app crash at all? If so, during which function?	NO	NO	NO	NO	NO	NO	NO	NO	NO	NA

Previous studies have researched the performance and usability of iPhone vs. Android apps the concerning the average times for certain functions to work, comparing several popular apps such as Twitter, Facebook, Pandora, Soundhound, Google Translator, and a mapping app called Geocaching were found (Spyre Studios 2011). The usability test results of past studies show the average times it took for most functions to work in these apps was between 4 and 18 seconds (Spyre Studios 2011). Based on such evidence it is assumed that the performance and functionality of the Mapping Native Plants app produces more than acceptable performance in terms of speed, since the functions take only from 2 to 19 seconds on average to execute with most functions performing in under 8 seconds. Functions in this context refers to every button, window, and action that the user can execute in the app. For example, zooming into the user's location in the map window, selecting a native plant feature, choosing a different background, viewing the references and library information, syncing a feature to the database on the server, etc., are all types of functions that were tested.

In general, the results from testing the Mapping Native Plants app show that all functions work appropriately in the field, taking an average time of six seconds for any one function to execute. Overall there was little lag time for loading the app and testers did not report the app crashing even once during the testing phase. The fastest performing function was the switching backgrounds function which took an average of 2 seconds, while the slowest performing function was the GPS Zoom function which needed an average of 8 seconds to run, the longest individual run times being 16 and 20 seconds. The app performed consistently across the board in each circumstance and location. There were two errors when the app was tested using an indoor Wi-Fi connection during syncing of feature attachments to the server. The sync was successful on the second try and the errors were most likely due to poor connectivity at the time (only two Wi-Fi bars whereas the other successful test segments of indoor Wi-Fi had a full four bars of connectivity).

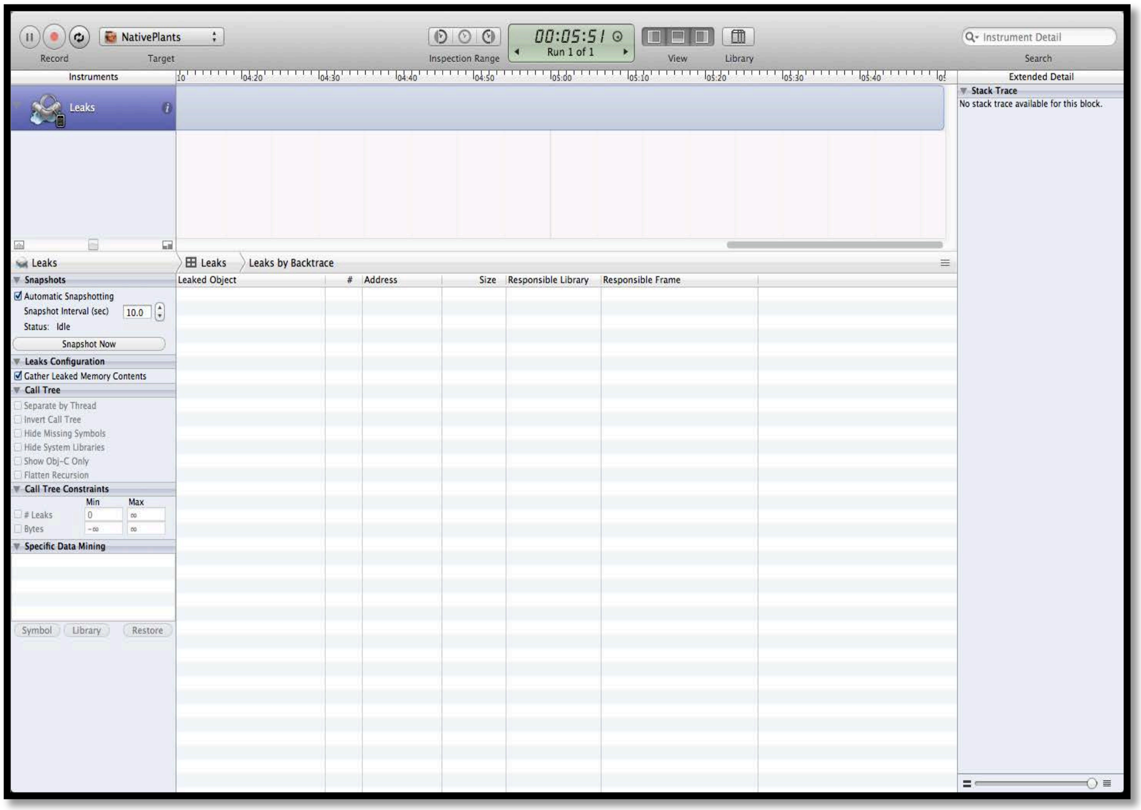
There were no obvious significant changes in battery charge level when the Mapping Native Plants app was being used, nor did any one function use more or less battery power while or immediately after the functions were executed. The GPS settings were changed when walking in different field test areas and the app behaved as expected under each GPS setting. Several people called and texted the iPhone while testing the app to determine if the app had any problems when incoming calls and texts were received. It was able to receive the texts and calls with no problem and the app worked normally during the calls. There even were several times when random notifications from other apps appeared during testing, and each time the app responded normally. According to these results the app passed the functionality and performance tests with no apparent flaws or requirements for improvement.

5.3 Test Results: Laboratory

For this segment of the Mapping Native Plants app testing the analyzer function in Xcode was used while the app was running in the Xcode simulator, and then four other tests were run in Xcode’s Instruments testing software while running the app on the iPhone as it was connected to the laptop. Each test measured a different aspect of the app as it was running. The four different tests in Instruments performed included: Leaks, Energy Diagnostics, Allocations, and Time Profiler. The results of each test were recorded and the output is interpreted in this section.

There are two types of memory leaks that can happen in a smart phone application. One is a “true memory leak” where an object has been allocated memory on the device, hasn’t been deallocated in the code, and is no longer referenced by any method. In this case, the memory it has been allocated can never be used and is therefore considered a “leak”. The second type of memory leak is called “unbounded memory growth” where memory is continually being allocated to an object but is never deallocated. Eventually the memory usage can build up until the application is forced to crash by the device’s OS.

The leaks Instrument test is concerned with detecting true memory leaks. The leaks test was run on the Mapping Native Plants app for 5 minutes and 51 seconds with each function being executed. The leaks instrument automatically takes a snapshot of the process every 10 seconds and analyzes the output for leaks. The results of this test as shown in Figure 33 indicate that there are no true memory leaks in the app. There were no leaks so there was no data collected by the leaks Instrument, thus there is nothing listed in the test output shown in Figure 33.



Stack Trace
No stack trace available for this block.

Leaked Object	#	Address	Size	Responsible Library	Responsible Frame

Snapshots
☑ Automatic Snapshotting
Snapshot Interval (sec) 10.0
Status: Idle
Snapshot Now

Leaks Configuration
☑ Gather Leaked Memory Contents

Call Tree
□ Separate by Thread
□ Invert Call Tree
□ Hide Missing Symbols
□ Hide System Libraries
□ Show Obj-C Only
□ Flatten Recursion

Call Tree Constraints

	Min	Max
□ # Leaks	0	∞
□ Bytes	-∞	∞

Specific Data Mining

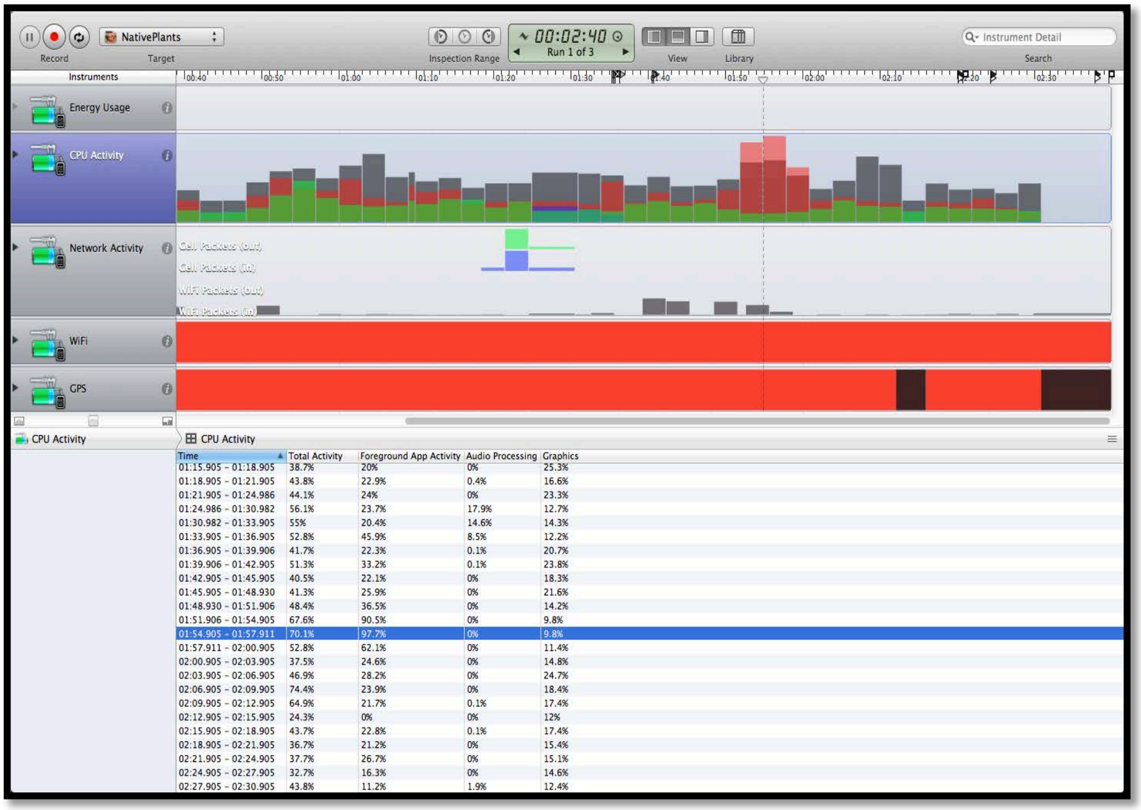
Figure 33: Leaks Instrument output

Nevertheless, just because the leaks Instrument does not find a leak does not necessarily mean there is not a leak that the Instrument could not detect. Although it is easy to rely on automatic reference counting for memory management when developing an app, it is up to the developer to make sure all of the objects in the code are being retained and released in the code. For this reason the code was double and triple-checked to make sure all objects within the Mapping Native Plants app code were retained and released during development.

The Energy Diagnostics instrument tracks all energy usage, CPU activity, network activity, Wi-Fi, and GPS activity being used in the app. This test was run for 2 minutes and 40 seconds, and the app was turned off and on at the end of the test to see how much energy it used when it was turned off. Again, every function was executed and notes were taken on any differences observed in the test output.

As can be seen in Figure 34, there was little energy usage to be accounted for. This is reported as a very thin line at the bottom of the Energy Usage graph, and when testing the Mapping Native Plants app this line never spikes or rises. The CPU Activity monitors the foreground app activity, the audio processing and the graphics. The foreground app activity peaked when the map was being zoomed into the user location while the other variables maintained a consistent status throughout the test. When a native plant feature and its attachments were synced to the server there was a spike in network activity, illustrated by the blue and green rises of Cell Packets going in and out. The Wi-Fi activity was turned on during the entire test, even when the app was turned off, because the iPhone was connected to the Wi-Fi throughout the duration of the test.

Testing the GPS activity was interesting because it was turned on the entire time the Mapping Native Plants app was on and it only turned off when the app was turned off. The GPS in the iPhone is supposed to be a significant energy (battery) consumer, and most developers will write code to turn the GPS off whenever it isn’t being used. Thus the fact that the Energy Usage graph isn’t showing an abundance of energy being used when the GPS is enabled in the app is an unexpected result of this test. At the end of the test the dark red blotches in the GPS graph indicating high energy consumption appear when the app was turned off, then on, then off again. Based on these results, turning the GPS off whenever the map window is not in the foreground should be considered for future development, since leaving it on does not consume enough battery to make coding this extra function a priority at this time.

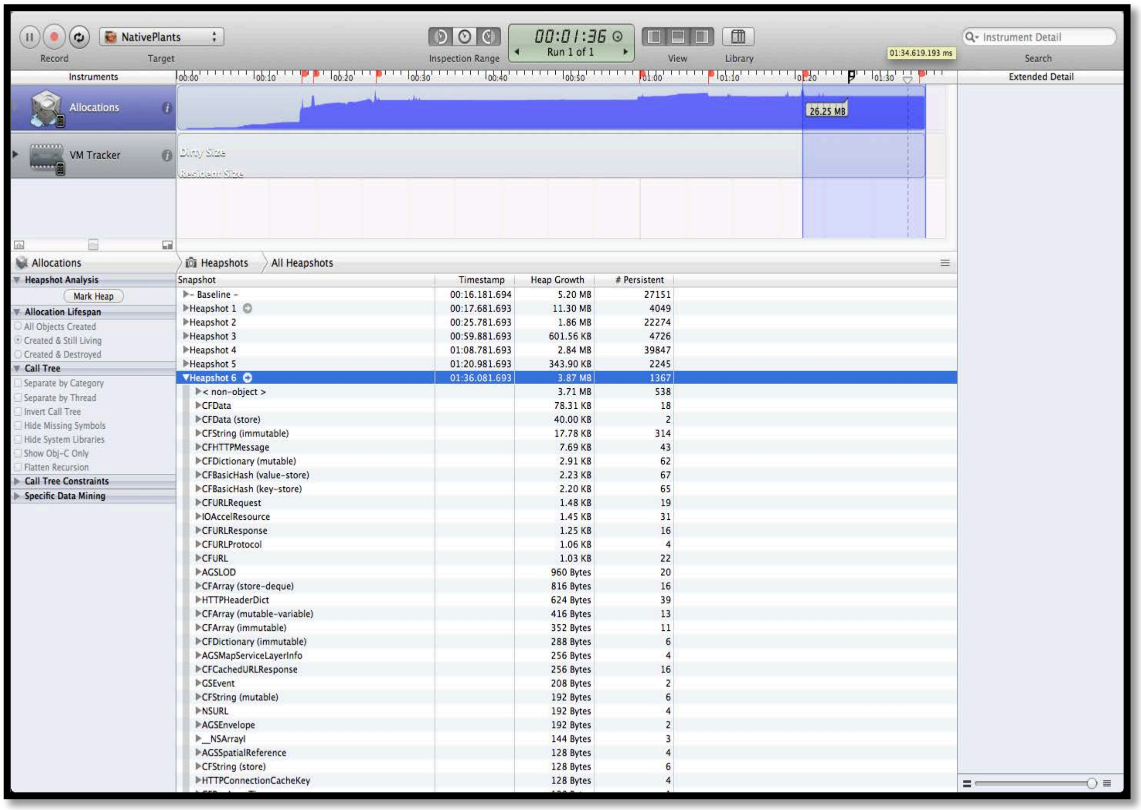


CPU Activity

Time	Total Activity	Foreground App Activity	Audio Processing	Graphics
01:15.905 - 01:18.905	38.7%	20%	0%	25.3%
01:18.905 - 01:21.905	43.8%	22.9%	0.4%	16.6%
01:21.905 - 01:24.986	44.1%	24%	0%	23.3%
01:24.986 - 01:30.982	56.1%	23.7%	17.9%	12.7%
01:30.982 - 01:33.905	55%	20.4%	14.6%	14.3%
01:33.905 - 01:36.905	52.8%	45.9%	8.5%	12.2%
01:36.905 - 01:39.906	41.7%	22.3%	0.1%	20.7%
01:39.906 - 01:42.905	51.3%	33.2%	0.1%	23.8%
01:42.905 - 01:45.905	40.5%	22.1%	0%	18.3%
01:45.905 - 01:48.930	41.3%	25.9%	0%	21.6%
01:48.930 - 01:51.906	48.4%	36.5%	0%	14.2%
01:51.906 - 01:54.905	67.6%	90.5%	0%	9.8%
01:54.905 - 01:57.911	70.1%	97.7%	0%	9.8%
01:57.911 - 02:00.905	52.8%	62.1%	0%	11.4%
02:00.905 - 02:03.905	37.5%	24.6%	0%	14.8%
02:03.905 - 02:06.905	46.9%	28.2%	0%	24.7%
02:06.905 - 02:09.905	74.4%	23.9%	0%	18.4%
02:09.905 - 02:12.905	64.9%	21.7%	0.1%	17.4%
02:12.905 - 02:15.905	24.3%	0%	0%	12%
02:15.905 - 02:18.905	43.7%	22.8%	0.1%	17.4%
02:18.905 - 02:21.905	36.7%	21.2%	0%	15.4%
02:21.905 - 02:24.905	37.7%	26.7%	0%	15.1%
02:24.905 - 02:27.905	32.7%	16.3%	0%	14.6%
02:27.905 - 02:30.905	43.8%	11.2%	1.9%	12.4%

Figure 34: Energy diagnostics Instrument output

The Allocations instrument gives detailed information about the objects being created and the memory that they use along with their retained counts. The initial test was run for 1 minute and 36 seconds, each function in the app was used, and a “heap shot” was created each time a function was being used. This test was run multiple times to create heap shots for several of the main functions in the app. A heap shot will track how much memory is being allocated when a function is being used. If more and more memory is being “heaped” every time the function is used then there is a memory leak somewhere. The allocations instrument tracks the unbounded memory growth for each tested function.



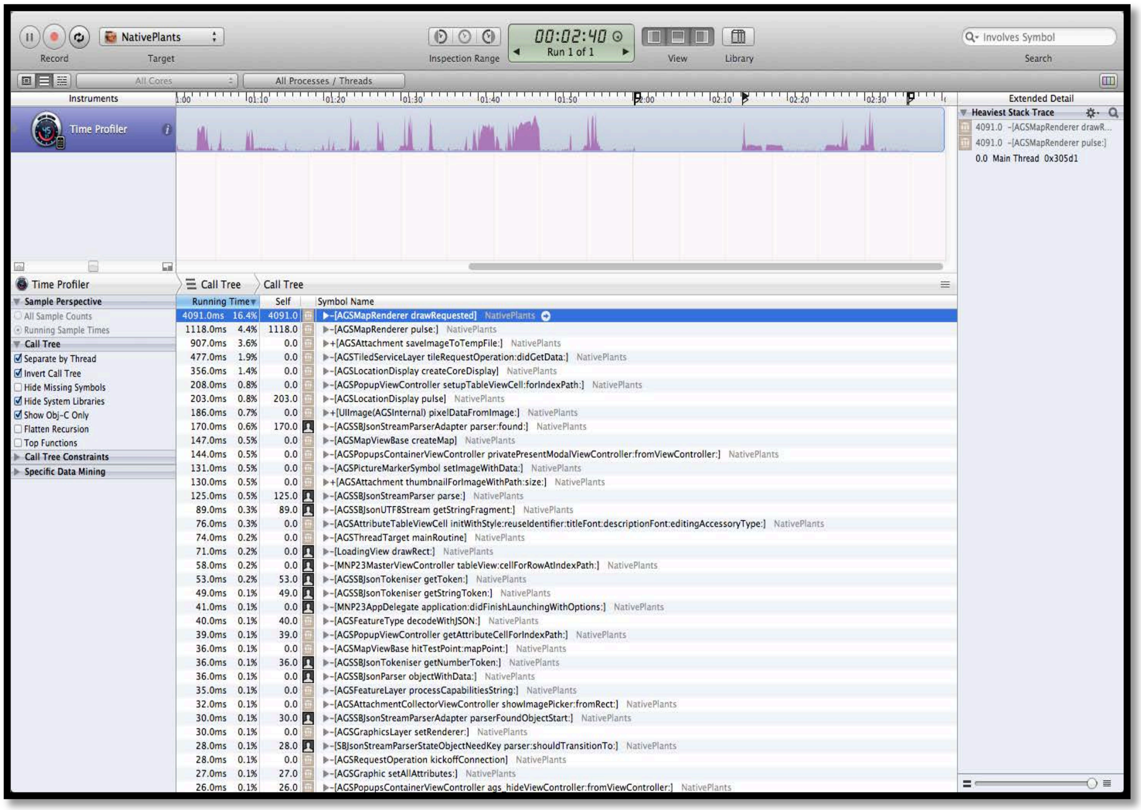
Allocations › Heapshots › All Heapshots

Snapshot	Timestamp	Heap Growth	# Persistent
▶- Baseline -	00:16.181.694	5.20 MB	27151
▶Heapshot 1	00:17.681.693	11.30 MB	4049
▶Heapshot 2	00:25.781.693	1.86 MB	22274
▶Heapshot 3	00:59.881.693	601.56 KB	4726
▶Heapshot 4	01:08.781.693	2.84 MB	39847
▶Heapshot 5	01:20.981.693	343.90 KB	2245
▼Heapshot 6	01:36.081.693	3.87 MB	1367
▶< non-object >		3.71 MB	538
▶CFData		78.31 KB	18
▶CFData (store)		40.00 KB	2
▶CFString (immutable)		17.78 KB	314
▶CFHTTPMessage		7.69 KB	43
▶CFDictionary (mutable)		2.91 KB	62
▶CFBasicHash (value-store)		2.23 KB	67
▶CFBasicHash (key-store)		2.20 KB	65
▶CFURLRequest		1.48 KB	19
▶IOAccelResource		1.45 KB	31
▶CFURLResponse		1.25 KB	16
▶CFURLProtocol		1.06 KB	4
▶CFURL		1.03 KB	22
▶AGSLOD		960 Bytes	20
▶CFArray (store-deque)		816 Bytes	16
▶HTTPHeaderDict		624 Bytes	39
▶CFArray (mutable-variable)		416 Bytes	13
▶CFArray (immutable)		352 Bytes	11
▶CFDictionary (immutable)		288 Bytes	6
▶AGSMapServiceLayerInfo		256 Bytes	4
▶CFCachedURLResponse		256 Bytes	16
▶GSEvent		208 Bytes	2
▶CFString (mutable)		192 Bytes	6
▶NSURL		192 Bytes	4
▶AGSEnvelope		192 Bytes	2
▶__NSArrayI		144 Bytes	3
▶AGSSpatialReference		128 Bytes	4
▶CFString (store)		128 Bytes	6
▶HTTPConnectionCacheKey		128 Bytes	4

Figure 35: Allocation Instrument output

The results of the allocations test in Figure 33 show that memory usage is consistent and every object that is allocated memory is having that memory deallocated after it is being used. You can see in the image above that the memory spikes when the native plants list is opened and each time a plant is selected a small spike in memory occurs but then goes back down to the optimal level once the user goes back to the list. The red flags are the heap shots made every time a function is performed. The two heap shots in the middle show a plant being selected from the list, there’s a rise in memory allocation, and then a drop after the user goes back to the list. These results show that the memory being used in the app is properly deallocated after the functions have been executed and no modifications need to be made to any code concerning memory allocation.

The Time Profiler instrument halts the execution of the program and performs a stack trace on each running thread. This is similar to pressing pause during a debugging session and analyzing the runtime for each method. A test was run with the time profiler instrument for 2 minutes and 40 seconds as each function was cycled through. The test was configured to hide the system libraries and show only the Objective-C methods in order to preview which methods are taking the most time (in milliseconds) to perform and determine if any of those methods need to be modified to make them run quicker.



Heaviest Stack Trace
4091.0 -[AGSMapRenderer drawR...
4091.0 -[AGSMapRenderer pulse:]
0.0 Main Thread 0x305d1

Time Profiler

Sample Perspective
○ All Sample Counts
● Running Sample Times

Call Tree
☑ Separate by Thread
☑ Invert Call Tree
☐ Hide Missing Symbols
☑ Hide System Libraries
☑ Show Obj-C Only
☐ Flatten Recursion
☐ Top Functions

Call Tree Constraints

Specific Data Mining

Running Time		Self	Symbol Name	
4091.0ms	16.4%	4091.0	-[AGSMapRenderer drawRequested]	NativePlants
1118.0ms	4.4%	1118.0	-[AGSMapRenderer pulse:]	NativePlants
907.0ms	3.6%	0.0	+[AGSAttachment saveImageToTempFile:]	NativePlants
477.0ms	1.9%	0.0	-[AGSTiledServiceLayer tileRequestOperation:didGetData:]	NativePlants
356.0ms	1.4%	0.0	-[AGSLocationDisplay createCoreDisplay]	NativePlants
208.0ms	0.8%	0.0	-[AGSPopupViewController setupTableViewCell:forIndexPath:]	NativePlants
203.0ms	0.8%	203.0	-[AGSLocationDisplay pulse]	NativePlants
186.0ms	0.7%	0.0	+[UIImage(AGSInternal) pixelDataFromImage:]	NativePlants
170.0ms	0.6%	170.0	-[AGSSBJsonStreamParserAdapter parser:found:]	NativePlants
147.0ms	0.5%	0.0	-[AGSMapViewBase createMap]	NativePlants
144.0ms	0.5%	0.0	-[AGSPopupsContainerViewController privatePresentModalViewController:fromViewController:]	NativePlants
131.0ms	0.5%	0.0	-[AGSPictureMarkerSymbol setImageWithData:]	NativePlants
130.0ms	0.5%	0.0	+[AGSAttachment thumbnailForImageWithPath:size:]	NativePlants
125.0ms	0.5%	125.0	-[AGSSBJsonStreamParser parse:]	NativePlants
89.0ms	0.3%	89.0	-[AGSSBJsonUTF8Stream getStringFragment:]	NativePlants
76.0ms	0.3%	0.0	-[AGSAttributeTableViewCell initWithStyle:reuseIdentifier:titleFont:descriptionFont:editingAccessoryType:]	NativePlants
74.0ms	0.2%	0.0	-[AGSThreadTarget mainRoutine]	NativePlants
71.0ms	0.2%	0.0	-[LoadingView drawRect:]	NativePlants
58.0ms	0.2%	0.0	-[MNP23MasterViewController tableView:cellForRowAtIndexPath:]	NativePlants
53.0ms	0.2%	53.0	-[AGSSBJsonTokeniser getToken:]	NativePlants
49.0ms	0.1%	49.0	-[AGSSBJsonTokeniser getStringToken:]	NativePlants
41.0ms	0.1%	0.0	-[MNP23AppDelegate application:didFinishLaunchingWithOptions:]	NativePlants
40.0ms	0.1%	40.0	-[AGSFeatureType decodeWithJSON:]	NativePlants
39.0ms	0.1%	39.0	-[AGSPopupViewController getAttributeCellForIndexPath:]	NativePlants
36.0ms	0.1%	0.0	-[AGSMapViewBase hitTestPoint:mapPoint:]	NativePlants
36.0ms	0.1%	36.0	-[AGSSBJsonTokeniser getNumberToken:]	NativePlants
36.0ms	0.1%	0.0	-[AGSSBJsonParser objectWithData:]	NativePlants
35.0ms	0.1%	0.0	-[AGSFeatureLayer processCapabilitiesString:]	NativePlants
32.0ms	0.1%	0.0	-[AGSAttachmentCollectorViewController showImagePicker:fromRect:]	NativePlants
30.0ms	0.1%	30.0	-[AGSSBJsonStreamParserAdapter parserFoundObjectStart:]	NativePlants
30.0ms	0.1%	0.0	-[AGSGraphicsLayer setRenderer:]	NativePlants
28.0ms	0.1%	28.0	-[SBJsonStreamParserStateObjectNeedKey parser:shouldTransitionTo:]	NativePlants
28.0ms	0.1%	0.0	-[AGSRequestOperation kickoffConnection]	NativePlants
27.0ms	0.1%	27.0	-[AGSGraphic setAllAttributes:]	NativePlants
26.0ms	0.1%	26.0	-[AGSPopupsContainerViewController ags_hideViewController:fromViewController:]	NativePlants

Figure 36: Time Profiler Instrument output

The results of the time profiler test in Figure 34 show that 16.4 percent (4091ms) of the app runtime is spent drawing the map when it opens. 4.4 percent (1148ms) is spent rendering the native plants layer, 3.6 percent (907ms) is spent saving the feature attachments (photos) to the server, 1.9 percent (477ms) is spent switching the backgrounds in the map window, and 1.4 percent (356ms) is spent zooming to the user's location. All other methods spend under 0.8 percent of the app's runtime with the majority being 0.0 percent (1ms). Based on these results it can be assured that the app runs at an optimal speed and there is not any further development needed to quicken any of the functional methods.

The laboratory test results overall have shown that the app was designed well and runs at an optimal speed. The app has no memory leaks, uses a minimal amount of energy, has all the object memory allocated for and released properly, and the functions all execute at the ideal speed for a mapping app of this caliber. Due to these positive results there are not any adjustments that need to be made to the app at this time.

5.4 Test Results: Usability

For the last test five archaeologists volunteered to use the app loaded onto an iPhone 4S and run through every function. Each user was told what the purpose of the app was and how it was supposed to be used. They were allowed as much time as they wanted to use the app and test all of its functions. After each one tested the app they were handed a questionnaire that asked them about their experience with the app. Some of the users own an iPhone and have lots of experience using iOS apps while others had minimal experience with the iPhone and smart phone apps in general. This is what the questionnaire looked like:

1. How easy was it to use the app on a scale of 1 to 10 (10 being super easy, and 1 being super hard)? Explain if you want to.
2. Did the app function correctly? If not, what was the problem?
3. Was there anything in the app that was difficult to understand? If so, what?
4. Was there a part of the app you really liked? What and why?
5. Is there anything in the app that you would change or think needs improvement?
6. Overall, how would you rate the app on a scale of 1 to 10 (10 being a good app and 1 being a poor app)? Explain if you want to.
7. Any additional comments, suggestions or questions?
8. How would you like to be identified in this study once it is published by USC? (Anonymous, use your initials, use your name) Please enter your initials or name at the top of the page, or circle anonymous in the choice above.

Thank you for your participation!

The results in Appendix B show that the participants found the app fairly easy to use and gave it an average of 8.4 out of 10 for ease of use. They reported that all of the functions worked properly except for complaints about the size of the feature symbol being too small for them to select. In the simulator the symbols are easy to select because the mouse pointer is small enough to select them, but for people with fingers bigger than a mouse pointer it wasn’t as easy.

The majority of users found the app easy to understand and they really liked the native plant list and the cultural information it gave for each native plant. Some suggested that more information should be given to the plant details like their status as an endangered species, while others wanted to see information like what the nearest plant was to them, a hyperlink to the museum websites in the About tab, and more information in the app concerning its purpose. Testers gave the app an average score of 8.2 out of 10 for the overall quality.

Considering these results changes were made to the size of the native plant symbol in the map window to make it easier for users to select the features in the map. In the future it would be a great idea to include the endangered species status for each plant in the native plant detail window as well as hyperlinks to the websites for the museums and botanical gardens listed in the museum pop-up.

Overall, the results from these five tests have shown that the app is well designed, works properly in the field and in the hands of users, and needs no further development to fulfill the thesis' purpose.

CHAPTER SIX: DISCUSSION AND CONCLUSION

Throughout the course of this project it became apparent that a mobile application such as the Mapping Native Plants app has the potential to become a great resource for local communities worldwide. In Southern California the indigenous languages are struggling to survive. Many of the native speakers have passed on and along with them a whole way of life. Their worldview, their history and culture, which have thrived for millennia, are slowly being muted and forgotten. There are those who aim to revitalize the languages and continue the traditions taught to them by their elders and they need help to teach the younger generations. The Mapping Native Plants app can be used as a tool to bridge the gap between the young and old, the uninformed and the knowledgeable, the amateur and the professional. The app is intended to be a tool of learning and when applied in the right setting it has the potential to inspire and create the desire to learn even more about native plants and indigenous cultures.

The key to learning about the past can be embedded in the technology we have today. As people who have inherited this land, in one way or another, it is our duty to do what we can to preserve the past and teach our children and neighbors the importance of native plants and the cultures that have tended to them for thousands of years. Native plants are vital organisms not only to the ecosystem but also to the survival of native cultures and humanity, such that when a plant goes extinct, a part of human history goes extinct.

The three research objectives of this thesis were achieved and successfully implemented. For the first objective, a mobile GIS application was developed to allow users to select, add, modify and delete native plant geotags in the Southern California study area. For the second objective, the app was coded and implemented using the Objective C programming language in Xcode, a geodatabase was constructed using ArcGIS for Server 10.1, and Esri’s ArcGIS SDK for iOS was utilized to incorporate ArcGIS tools into the app’s mapping functions. And for the third objective, a methodology was created to allow users of the app to access and share native plant geographic information through their iPhones and ArcGIS Online.

The two main research questions were answered through user and laboratory testing. The Mapping Native Plants app was determined to be user-friendly and easy to use by both experienced and inexperienced smartphone users. And the user interface is developed in a way that the main functions of the app are self-explanatory and no prior training is needed to use the app effectively. Also, researchers are able to access the app data through an ArcGIS Online subscription to allow them to integrate the spatial and attribute data into their own research projects.

This chapter highlights the main advances brought on by the Mapping Native Plants app, discusses what the next steps are for further developing the app, and considers the future trends for this type of mobile GIS application.

6.1 Main Advances Brought by the Mapping Native Plants Application

What this app brings to geographic information science, and to smart-phone applications in general, is a cultural connection. Throughout human history communication has evolved from the spoken word to images painted on rock walls and symbols arranged in a logical order and written down on paper. The technology and format has also evolved from paint and paper to radio, television, personal computers, and handheld devices. Smart phone applications are the next step in this evolutionary path that extend our reach to people in other communities who we may never meet but still have an influence on and be influenced by. Millions of people could use this app and never have the chance to meet each other but they can still share their perspective of the environment with each other and contribute to a collective knowledge of native plants.

The cultural component within the Mapping Native Plants app sets it apart from any other mapping app in existence. It can bring together people from all different backgrounds, industries, sciences, communities and cultures to share in the adventure of geotagging native plants and learning about how indigenous cultures use them in the past and present. This app starts a conversation between the person who is interested in nature and the senior ecologist, between the person interested in anthropology and the cultures they are studying and the descendants of those cultures, between the researcher looking for data and the avid conservationist aiming to protect natural resources. It can bring to light the concerns of the public to the private developers and lawmakers by acting as a geographical place marker for the status of endangered species. It can influence the decision making of tribal entities and federal parks managers by showing them which areas need to be protected the most. It can teach people that native plants are important to our survival as well as the survival of the ecosystem around us.

The Mapping Native Plants app has positively demonstrated that it can bridge the gap between the amateur and professional geographic information collector. In the subfields of volunteered geographic information and public participatory GIS this app is a tool that can enhance scientific study, lower the expense for collecting geographic data, and verify the quality of data because the public is more capable of doing fieldwork since they have less time to travel if native plants are in their backyard, they don't have any deadlines on when data needs to be collected, and they have a knowledge of the areas they live in much greater than any professional scientist who has only read about the area in a book. This app puts power into the hands of the individual and the community by allowing them to collect data in places that the normal researcher might not have access to and thereby qualifies their local knowledge as being equal to, if not greater than, the professional. This app is more than crowdsourcing, it is crowd empowering. Anyone can use the data output in the web map. Anyone can see what other people have geotagged and contributed. Anyone can disagree with anyone else's geotags and correct them. This fosters a greater communal knowledge of the environment and lets people freely participate in scientific studies as well as aid them in self-education and teaching others about native plants.

By taking a tangible product that can be located all over Southern California, native plants, and mixing them with an intangible value, cultural linguistics and traditions, the Mapping Native Plants app promotes cultural learning while being primarily a tool of GIS. GIS has gained more popularity over the years but there are still many people who do not understand what it is and how it can be used for the good of people and the environment. This app allows anyone with an iPhone to use GIS and see how geographic information can be used to foster communication, create understanding, and spread local knowledge to a greater audience. The app sets a good example for how mobile GIS can be used in a way that is sensitive to cultural revitalization while also promoting environmental responsibility.

6.2 The Future of the Mapping Native Plants Application

There are many possibilities for the development of this application in the future. However, before any type of development could take place, upon completion of this thesis the data collection functionality would need to be either hosted on the cloud or on a server with ArcGIS licensing, because the app will no longer have access to the Spatial

Sciences Institute’s VM or licensing. When either of those options becomes possible the app may eventually grow out of its Southern California roots and branch out to other parts of the world. There could be a version of the app for each state and, possibly, for other countries. Although it would take an immense amount of work to research all the native plants in each area and find out which indigenous cultures use them and what they call them in their native languages, all of the project structure, methodology and example app code for iOS devices is now available in the app to be customized for other locations. It would take a massive effort and collaboration from many agencies and tribal entities to compile such a database of information about native plants for the entire United States. Despite the work it would take to expand the reach of the app it would be beneficial because of the potential of the app to be extremely useful in the United States and perhaps all areas of the world, wherever there are native plants and indigenous cultures with data collection needs.

There are already plans to create a version of the app for the state of Hawaii, which has an indigenous culture that is very much alive with their language and traditions being taught in many public and private schools. There is even the idea of adding some native Hawaiian faunal species to the list to help track the locations of endangered animals and teach people what to do when they recognize them. There are many projects in Hawaii, and across the globe, with the goal of tracking and protecting native plants and animals, this app could be used as a tool to engage the public in those ventures and help researchers and conservationists reach those goals.

The first idea is to expand the app to as many regions as possible but there are several features that could be added to the Mapping Native Plants app that would also be beneficial, in a research perspective, to make it more interesting and user friendly for the average user. Adding the element of time to the app is a major goal. Once the app has been in use for several years it would be nice to see how the distribution of native plants has changed over time. There would be a time slider that allows people to see how many native plants have come and gone over the years. A query function would also be added so the user could select a type of plant and watch its distribution change over time. This feature would be useful to long-term environmental studies concerned with climate change and also allow researchers to track certain species' populations and determine whether the distribution changes are because of climate change or any other type of ecological disturbance such as a wild fire, hurricane, drought, etc.

The next goal is to add user logins to the app. People can enter a user name and password and, as an option, their email address and zip code. With user logins the app can be made more secure by ensuring that people who abuse the app are restricted from using it in the future. User logins can also be used to enhance the app by adding user names to geotags so people can take pride in their contributions. Also, the fun of geotagging native plants can be made into a competition where a list in the app will show the top ten contributors for certain areas and plants. And if the native plant list is connected to the geodatabase the native plant detail window can show the top contributors for the plant that is selected. This will give an incentive to users to go out and geotag more plants and try to find some of the more rare native plants that are out there.

It would also be a good idea to set a geographic threshold, or boundary area, where users are allowed to geotag native plants. This threshold would be something like a

100-meter buffer surrounding the user's location. It would make sure that users are not creating false native plant geotags in areas that they have never been. If they want to contribute a geotag to the app then they need to be at least within 100 meters of the plant. If they try to geotag beyond this buffer zone an error message will appear telling them they need to move closer to the plant they are trying to geotag.

Another feature would be a red flag system for any endangered species or native plants that are toxic. It would serve as a warning to users on how to interact with these plants while also letting researchers know where they are being geotagged. The flags would have different colors depending on the status of the plant as an endangered species or its level of toxicity. When a user selects a plant to geotag from the template picker a warning flag would pop-up telling them to not touch it and what to look out for.

It would also be highly valuable if all native plants could be added to the list, not just two hundred and sixty-seven. And with more engagement and collaboration with the local tribes there would be more information on the native names and uses for each plant. There are many different native cultures and languages in Southern California and it is hopeful that in the future there will be more information on the names of all native plants for each language. The Cahuilla and Chumash languages were chosen not only because their territories covered such wide areas of Southern California but also because they currently have the most information on indigenous names for native plants.

Another goal is a function that allows the user to tap on the native name of a plant and a sound bite would play so they could hear what the words sound like coming from a native speaker. It would be beneficial to the overall outreach of the app to include all

Southern California native languages, if possible, and to have native speakers contribute their voices so that everyone can hear how the words are supposed to be pronounced.

Basic modifications of several app functions will also be implemented to enhance the user experience. The rating for each user’s contribution needs to be defined within the app. The user should know what it means to rate a geotag as “excellent” or “poor” so these terms need to be better associated with the credibility of the data (location and information given). There should be a pop-up in the About window that defines the rating terms and gives examples of what kind of geotag would deserve a certain type of rating. Also, a user’s rating should not be editable, either. Rather, the ratings of all users for a particular geotag should be compiled and averaged. This average rating will be what is shown in the feature’s details.

Feedback within the app would also be a valuable resource to make the app stronger and more relatable to the users. The users could rate the app, leave comments and share their frustrations or ideas they have for the app. This information can be used to further develop it and create a product that is more user-friendly and functional.

Another feature that is needed in the future is a more pronounced disclaimer. In the current version of the app the disclaimer is hidden within the About window and is very easy for the user to pass it without reading it. The disclaimer needs to be at the forefront of the app by opening in a window right after the launch screen disappears. The user will have to acknowledge they have read the disclaimer by pressing a button at the bottom before they can begin using the app. This will ensure that the users are more aware of the disclaimer and will not use the app in a way that will put them in any physical or legal danger.

The app users would also benefit from having better imagery and some extra background layers to choose from. The app is currently using background layers provided by Esri and they work well with the map window but the quality and resolution of the imagery is not high. With high resolution imagery the user can see some of the larger plants and their geotags can become more precise. With the Esri imagery the image can only be zoomed in so far (about 1:2500 scale). With higher resolution imagery the user can zoom in closer to their position and see more of what is around them. This would help the user locate certain plants if there was a background layer that had the native plant life zone polygons on it. They could learn about a plant’s life zone in the native plant details and then in the map window search for where that life zone is relative to where they are. Other useful background layers would be a soils layer, a layer showing where botanical gardens and parks are, and a layer showing the prehistoric tribal boundaries.

The last goal is to have the app run on multiple smartphone platforms. This would increase the reach of the app to more people and further add to the local knowledge that the app is collecting. It would be challenging to maintain the app as each operating system gets upgraded but the native plant feature service data would be consistent across platforms because it would be hosted in the same enterprise geodatabase on ArcGIS Server and embedded in the same web map on ArcGIS Online.

It is also worthy to note that Esri has just announced plans for a new site in September 2013 called ArcGIS Marketplace for Apps where app developers can promote and distribute their mapping apps and users can discover and access mobile apps that interest them. This would be a great way to market the Mapping Native Plants app and get it in the hands of more users, especially experienced GIS users, who can provide insight and advice for developing the app even further.

The overall goal for the future is to open the app up to more plants, more users, more cultures, more languages and in as many places as possible. It would be remarkable to see a Mapping Native Plants app for every possible region, in every possible language, and mapping every possible native plant, being used by anyone with access to a smartphone. This dream may take many years to become reality but with more and more advancements in GIS and smart phone technology it is not at all unlikely.

6.3 Future Trends for Mobile GIS Applications

The future of mobile GIS applications will become focused more on the public user rather than the professional cartographer. User interfaces will change to help people understand what they are doing with the map rather than treating them like blindfolded information donors who have no idea what the purpose of the project may be. The Mapping Native Plants app is willing to give information to get information instead of being solely an information collector. Mobile GIS apps in the future need to follow this model of information exchange to provide an incentive for users to share their local knowledge on whatever the subject may be.

The hope is that the Mapping Native Plants app will inspire others to create apps that are centered on helping the environment and the people who live in it. Mobile GIS applications will become learning tools in elementary schools, intermediate schools and colleges to help instructors teach a plethora of subjects. It can provide an interactive learning environment that aids in self-education and allows the students to learn at their own pace.

Mobile GIS applications are tools that can help people visualize the world around them and influence current policies and legislation. If a large developer wants to build a mini mall in a certain location the public could use a mobile GIS app to see what kind of endangered species might be threatened by that development, or vice versa, the developer could use it to prove there is nothing being endangered in that area. The idea is that mobile GIS apps are able to help us make more informed decisions concerning land use and our natural and cultural resources.

Mobile GIS apps have expanded the reach of GIS and opened its application to more than just road maps and travel guides in the minds of the general public. As GIS professionals it is our duty to promote the use of GIS to the public in an ethical way that protects the privacy and rights of the individual and community as a whole. And mobile GIS applications are capable of doing just that. They can protect the anonymity of the user and give them the choice on how their data is being used and how their privacy will be affected. The Mapping Native Plants app allows the user to remain anonymous while giving them the freedom to geotag native plants without revealing their exact location.

The trend of mobile GIS will hopefully veer away from unethical crowd sourcing and aim toward empowering the public with valuable information and the tools to decide what should be recorded or not. We need to trust the public and give them the ability to create data without the fear of its accuracy or reliability being poor. With the affordability that mobile GIS apps lend to scientific research this area of GIS will increase and will only be accepted by the profession once we create better ways to interpret volunteered geographic information. VGI is a valuable data source and we need further conceptual studies to evaluate and understand how it can be used appropriately and in ways that give it the authority it deserves.

6.4 Final Thoughts

The idea for this thesis began at a seminar on how to teach Native American school children in Southern California about the indigenous uses of native plants. At first, the idea was to build a geodatabase for all native plants in coordination with a local seed bank to help them locate where native plants are and where they should be growing. The idea then grew to be more about mapping the populations of native plants and what would be the best way to go about doing that. Already, this project was developing at a large scale and it needed to be narrowed down to a specific plant or region. But GIS is capable of focusing on more than one plant or one region. It can be inclusive of all native plants and all regions in Southern California and enable people to learn more about native plants and their indigenous uses.

With that thought in mind, the thesis needed to consider what was new in GIS, what had not been done before, and what needed to be done. It was eventually decided that the best way to engage the public to learn about native plants with the use of GIS technology was to create a smart phone app. It was later realized that this kind of app has the potential to provide important data to many different scientific studies as well as to native peoples through the use of VGI. There have been many trials and tribulations in forming this project and developing the Mapping Native Plants app but in the end it has met its original research goals by being able to aid in educating people about native plants and indigenous cultures in Southern California, provide valuable data for researchers in many different fields of study, and inspire others to create similar mobile GIS apps that aim to empower people with the knowledge about their environment.

References

Apple. 2011. *Apple - iPhone 4S - Technical Specifications*. http://www.apple.com/iphone/specs.html (accessed 30 October 2011).

Apple Developer. 2013. https://developer.apple.com. (accessed 30 October 2013)

Apple Insider. 10 June 2013. Apple seeds first iOS 7 and OS X Mavericks betas to developers. http://appleinsider.com/articles/13/06/10/apple-seeds-first-ios-7-and-os-x-mavericks-betas-to-developers. (accessed 30 October 2013).

ArcCatalog version 10.1, Esri. Redlands, California, United States of America.

ArcMap version 10.1, Esri. Redlands, California, United States of America.

ArcGIS for Server version 10.1, Esri. Redlands, California, United States of America.

Barrows, D. P. 1900. *The Ethno-Botany of the Coahuilla Indians of Southern California.* Chicago: University of Chicago Press.

Batcheler, M. no date. Nature Mapping. *Green Teacher 84* , 39-41.

Bazell, R., & Carroll, L. 9 December 2011. *Indian tribe turns to tradition to fight diabetes.* http://www.msnbc.msn.com/id/43257536/ns/health-diabetes/# (accessed 30 October 2013).

Bean, L. J. 1972. *Mukat's People: The Cahuilla Indians of Southern California.* Los Angeles: University of California Press.

Bean, L. J., & Saubel, K. S. 1972. *Temalpakh: Cahuilla Indian Knowledge and Usage of Plants.* Banning: Malki Museum, Inc.

Bing Maps version 7, Microsoft. Redmond, Washington, United States of America.

Box, E. O. 2006. *McGraw-Hill Concise Encyclopedia of Science and Technology.* The McGraw-Hill Companies, Inc.

Budhathoki, N. R., Bruce, B. C., & Nedovic-Budic, Z. 2008. Reconceptualizing the role of the user of spatial data infrastructure. *GeoJournal , 72*, 149-160.

Calflora 2013. Calflora application. http://www.calflora.org/phone/ (accessed 30 October 2013).

Case, P. 25 February 2011. Measure green area index on your iPhone. *Farmers Weekly ,* p. 64.

Center for Embedded Network Sensing (CENS). 2013. What's Invasive! Community Data Collection. http://www.whatsinvasive.org/about.html (accessed 30 October 2013).

Clary, R., & Wandersee, J. 2011. Our Human-Plant Connection. *Science Scope* , 32-37.

Crain, B. J., & White, J. W. (2011). Categorizing locally rare plant taxa for conservation status. *Biodiversity Conservation , 20*, 451-463.

Dunn, C. E. 2007. Participatory GIS - a people's GIS? *Progress in Human Geography* , 616-637.

Dutt, B., Kumar, S., & Garg, K. C. 2009. Mapping of plant-based medicine research in China and India. *Research Evaluation , 18* (1), 51-59.

Eades, G. L. 2010. *Geoweb: Indigenous Mapping of Intergenerational Knowledge.* McGill University, Geography. Montreal, Quebec: Gwilym Lucas Eades.

Elliot, L. 1997. Cases of diabetes expected to triple (among Native population). *Wind Speaker* , 1,19.

Esri 2013. ArcGIS for iOS Developer Samples. http://www.arcgis.com/home/group.html?owner=iOSDevelopmentTeam&title=ArcGIS%20for%20iOS%20Developer%20Samples (accessed 30 October 2013).

Flanagin, A. J., & Metzger, M. J. 2008. The credibility of volunteered geographic information. *GeoJournal , 72*, 137-148.

Franklin, J. 2009. *Mapping Species Distribution: Spatial Inference and Prediction.* Cambridge: Cambridge University Press.

Gelbard, J. L. 2003. *Understanding the Distribution of Native versus Exotic Plant Diversity in California's Grassland Landscapes.* Davis: University of California, Davis.

Goodchild, M. F. 2007. Citizens as sensors: the world of volunteered geography. *GeoJournal , 69*, 211-221.

Goodchild, M. F. 2008. Commentary: whither VGI? *GeoJournal , 72*, 239-244.

Gouveia, C., & Fonseca, A. 2008. New approaches to environmental monitoring: the use of ICT to explore volunteered geographic information. *GeoJournal , 72*, 185-197.

Griffin, J. A.2000. Challenges to Participating in a Lifestyle Intervention Program: The Native American Diabetes Project. *The Diabetes Educator* , 681-689.

Hassell, Jonathan. 6 July 2010. Developing for the iPhone and Android: The pros and cons. http://www.computerworld.com/s/article/9178684/Developing_for_the_iPhone_and_Android_The_pros_and_cons?pageNumber=1 (accessed 30 October 2013).

Hernandez, D. 2005. Got tradition? American Indians use native foods to fight diabetes and revive Indian culture. *Colorlines Magazine* .

Huang, C.-y., & Geiger, E. L. 2008. Climate anomalies provide opportunities for large-scale mapping of non-native plant abundance in desert grasslands. *Diversity and Distributions , 14*, 875-884.

IntuitionHQ. 2011. Android and iOS Video Example: Results. http://video.intuitionhq.com/pub/624/1. (accessed 30 October 2013).

Instruments version 2.7, Apple Inc. Cupertino, California, United States of America.

iOS version 6.1.4, Apple Inc. Cupertino, California, United States of America.

iPhone 4S. Apple Inc. Cupertino, California, United States of America.

Jernigan, V. B., & Lorig, K. 2010. The Internet Diabetes Self-Management Workshop for American Indians and Alaska Natives. *Health Promotion Practice , 12*, 261-270.

K, Donna. 12 April 2013. Esri Support Services Blog: Bing Maps Use in ArcMap Has Changed. http://blogs.esri.com/esri/supportcenter/2013/04/12/bing-maps-use-in-arcmap-has-changed/. (accessed 30 October 2013).

Koester, H. 2008. Native Plants and Urban Sustainability. *Native Plants Journal* , 323-333.

Kraak, M.-J. 2011. Is There a Need for Neo-Cartography? *Cartography and Geographic Information Science , 38* (2), 73-78.

Landis, T. D. 2005. Roadside revegetation of forest highways: new applications for native plants. *Native Plants Journal* , 297-305.

Largo, D., McCarthy Daniel, F., & Roper, M. 2009. *Medicinal Plants Used by Native American Tribes in Southern California.* Banning, California: Malki-Ballena Press.

Liu, S. B., & Palen, L. 2010. The New Cartographers: Crisis Map Mashups and the Emergence of Neogeographic Practice. *Cartography and Geographic Information Science , 37* (1), 69-90.

Lunden, Ingrid. 1 August 2013. Despite Google's Gains, iPhone Still Edges Out Android Devices In App And Overall Smartphone Usage, Says Forrester. http://techcrunch.com/2013/08/01/despite-googles-gains-iphone-still-edges-out-android-devices-in-app-and-overall-smartphone-usage-says-forrester/ (accessed 30 October 2013).

Mahaney, W. M. 2010. Plant controls on decomposition rates: the benefits of restoring abandoned agricultural lands with native prairie grasses. *Plant Soil* , 91-101.

Manjoo, Farhad. 17 July 2013. Give Me Back My iPhone! http://www.slate.com/articles/technology/technology/2013/07/android_vs_iphone_why_apple_still_has_the_edge_over_google_s_operating_system.html. (accessed 30 October 2013).

Microsoft SQL Server Management Studio 2008 R2, Microsoft. Redmond, Washington, United States of America.

National Audubon Society. 2010. Audubon Guides. http://www.audubonguides.com/field-guides/california.html. (accessed 30 October 2013).

Organization, W. H. 2002. *WHO Traditional Medicine Strategy 2002-2005.* Geneva: World Health Organization.

Parker, B. D. November 1999. Erosion Control in the Big Sur. *Public Works* , pp. 26-29.

Patterson, Sean. 6 August 2013. Apple, Samsung Set to Dominate Sales. http://www.webpronews.com/apple-samsung-set-to-dominate-smartphone-sales-2013-08. (accessed 30 October 2013).

Reed, Brad. 21 March 2013. iPhone once again trounces rivals in J.D. Power customer satisfaction survey. http://bgr.com/2013/03/21/iphone-customer-satisfaction-best-389139/. (accessed 30 October 2013).

Rhoades, M. E. R. 2000. *American Indian Health.* Baltimore: The Johns Hopkins University Press.

Roberts, S. S. 2004. Talking Circles Ease Burden of Diabetes for Native Americans. *The Diabetes Advisor* , 23-24.

Rundel, P. W., & Gustafson, R. 2005. *Introduction to the Plant Life of Southern California: Coast to Foothills.* Berkeley: University of California Press.

Ryan, Kim. 18 July 2011. IPhone rides Verizon to gain preference edge over Android. http://gigaom.com/2011/07/18/iphone-rides-verizon-to-gain-preference-edge-over-android/. (accessed 30 October 2013).

Seeger, C. J. 2008. The role of facilitated volunteered geographic information in the landscape planning and site design process. *GeoJournal* , *72*, 199-213.

Serrill, W. D. 2006. Restoration of Native Plants on Catalina Island, California. *Native Plants Journal* , *7* (1), 4-14.

Service, I. H. 1 January 2011. *IHS Fact Sheets*. http://www.ihs.gov/PublicAffairs/IHSBrochure/Diabetes.asp (accessed 30 October 2013).

Smithsonian Institute, 2013. Leafsnap application. http://www.si.edu/apps/leafsnap (accessed 30 October 2013).

Song, W., & Sun, G. 2009. *The Role of Mobile Volunteered Geographic Information in Urban Management.* Fuxin, China: Liaoning Technical University, School of Geomatics.

Song, W., & Sun, G. 2009. Using Mobile GIS as Volunteered GI Provider. *The 1st International Conference on Information Science and Engineering* (pp. 2229-2232). Fuxin, China: IEEE Computer Society.

Spero, D. 2006. *Diabetes: Sugar-Coated Crisis.* Gabriola Island, British Columbia, Canada: New Society Publishers.

Spyre Studios. 2011. Android vs iOS; A Usability Battle. http://spyrestudios.com/android-vs-ios-a-usability-battle/. (accessed 130 October 2013).

Stack Exchange Inc. 2013. http://stackoverflow.com. (accessed 30 October 2013).

Stohlgren, T. J., Pyšek, P., Kartesz, J., Nishino, M., Pauchard, A., Winter, M., Pino, J., Richardson, D. M., Wilson, J. R. U., Murray, B. R., Phillips, M. L., Ming-yang, L., Celesti-Grapow, L., & Font, X. 18 January 2011. Widespread plant species: natives versus aliens in our changing world. *Biological Invasions* (13), 1931-1944.

Sui, D. Z. 2008. The wikification of GIS and its consequences: Or Angelina Jolie's new tattoo and the future of GIS. *Computers, Environment and Urban Systems , 32,* 1-5.

Timbrook, J. 2007. *Chumash Ethnobotany: Plant Knowledge Among the Chumash People of Southern California.* Santa Barbara: The Santa Barbara Museum of Natural History.

Tulloch, D. L. 2008. Is VGI participation? From vernal pools to video games. *GeoJournal* , *72*, 161-171.

Vos, E. 19 November 2009. *Report on Internet use in Native American communities shows huge broadband gap*. http://www.muniwireless.com/2009/11/19/report-on-internet-use-in-native-american-communities-shows-huge-broadband-gap/ (accessed 30 October 2013).

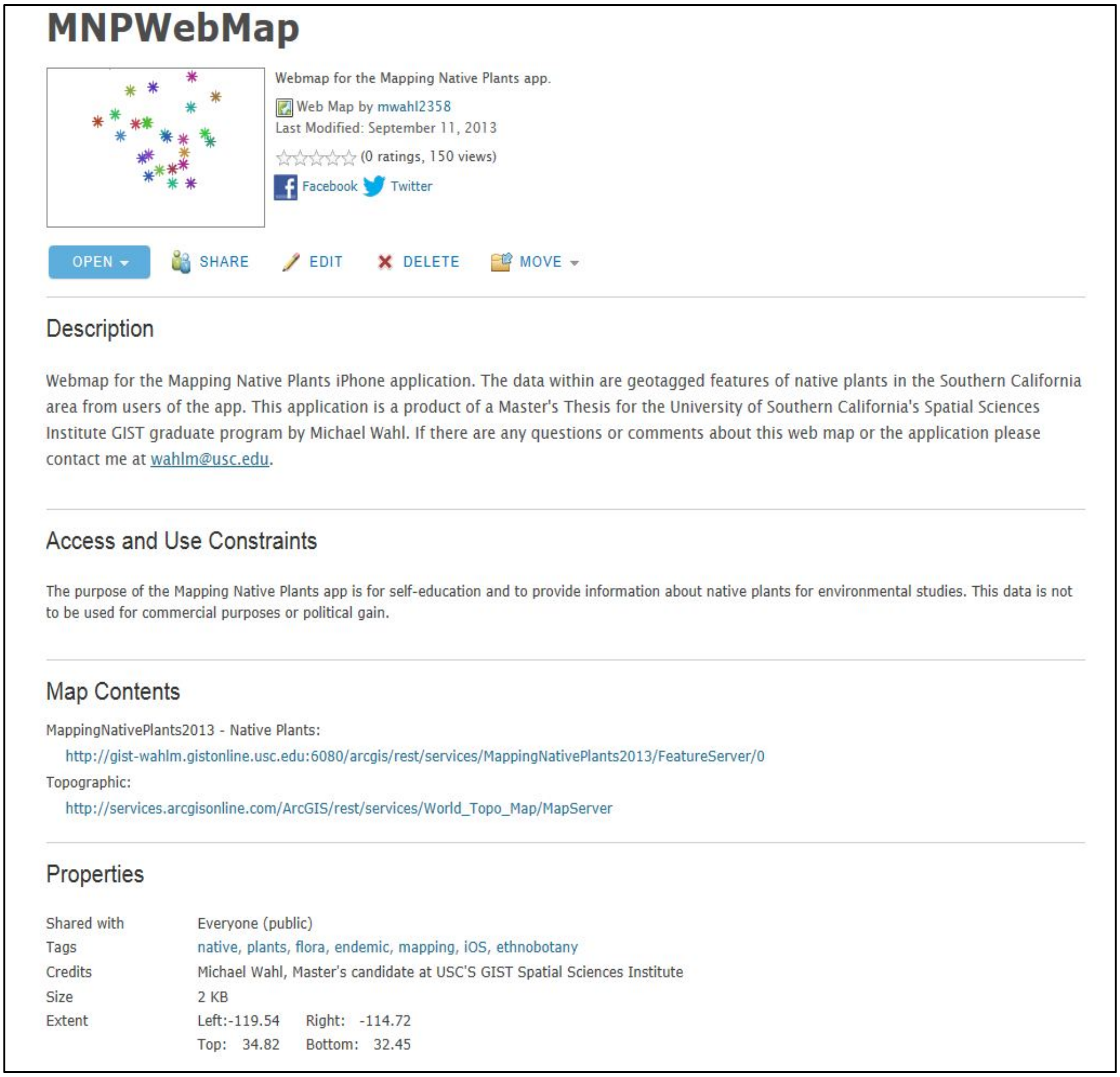
Wilson, J. P., & Fotheringham, A. S. 2008. *The Handbook of Geographic Information Science.* Malden, MA, USA: Blackwell Publishing.

Woollaston, Victoria. 20 August 2013. Most first-time smartphone buyers for a Samsung but one fifth of Android users are switching to Apple. http://www.dailymail.co.uk/sciencetech/article-2397833/Android-users-likely-switch-Apples-iPhone-time-smartphone-buyers-Samsung.html. (accessed 30 October 2013).

Xcode version 4.6.3, Apple Inc. Cupertino, California, United States of America.

Zavaleta, E. S., Shaw, M. R., Chiariello, N. R., Mooney, H. A., & Field, C. B. 2003. Additive effects of simulated climate changes, elevated CO2, and nitrogen deposition on grassland diversity. *PNAS* , 7650-7654.

Appendix A: Web Map Output



MNPWebMap

Webmap for the Mapping Native Plants app.

Web Map by mwahl2358

Last Modified: September 11, 2013

☆☆☆☆☆ (0 ratings, 150 views)

Description

Webmap for the Mapping Native Plants iPhone application. The data within are geotagged features of native plants in the Southern California area from users of the app. This application is a product of a Master's Thesis for the University of Southern California's Spatial Sciences Institute GIST graduate program by Michael Wahl. If there are any questions or comments about this web map or the application please contact me at wahlm@usc.edu.

Access and Use Constraints

The purpose of the Mapping Native Plants app is for self-education and to provide information about native plants for environmental studies. This data is not to be used for commercial purposes or political gain.

Map Contents

MappingNativePlants2013 - Native Plants:

http://gist-wahlm.gistonline.usc.edu:6080/arcgis/rest/services/MappingNativePlants2013/FeatureServer/0

Topographic:

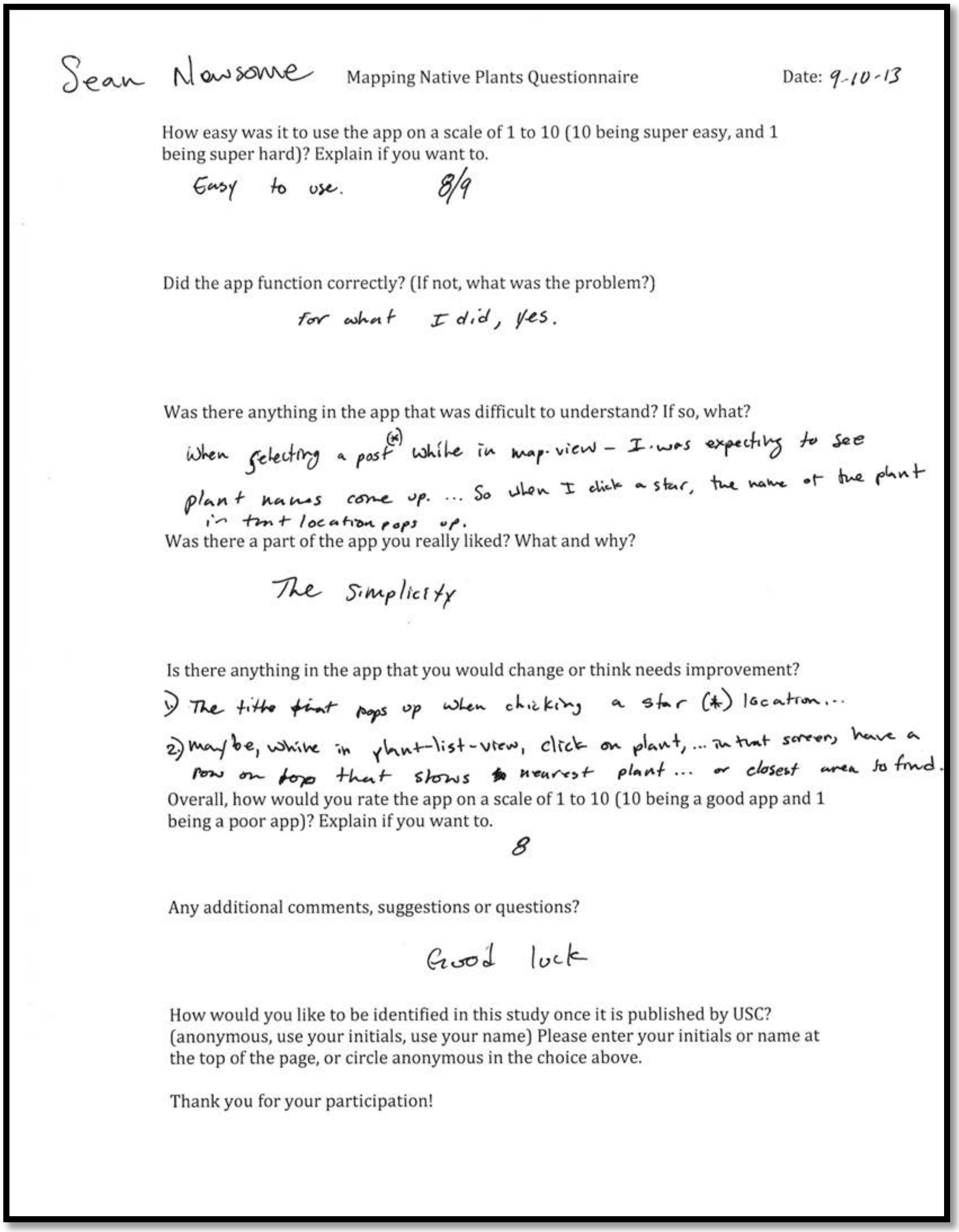
http://services.arcgisonline.com/ArcGIS/rest/services/World_Topo_Map/MapServer

Properties

Shared with	Everyone (public)	
Tags	native, plants, flora, endemic, mapping, iOS, ethnobotany	
Credits	Michael Wahl, Master's candidate at USC'S GIST Spatial Sciences Institute	
Size	2 KB	
Extent	Left:-119.54	Right: -114.72
	Top: 34.82	Bottom: 32.45

Appendix A: Web Map Output

Appendix B: Usability Test Questionnaires



Sean Nowsome Mapping Native Plants Questionnaire Date: 9-10-13

How easy was it to use the app on a scale of 1 to 10 (10 being super easy, and 1 being super hard)? Explain if you want to.

Easy to use. 8/9

Did the app function correctly? (If not, what was the problem?)

for what I did, yes.

Was there anything in the app that was difficult to understand? If so, what?

When selecting a post(*) while in map-view – I was expecting to see plant names come up. ... So when I click a star, the name of the plant in that location pops up.

Was there a part of the app you really liked? What and why?

The Simplicity

Is there anything in the app that you would change or think needs improvement?

1.) The titte ~~that~~ pops up when clicking a star (*) location...

2.) maybe, while in plant-list-view, click on plant, ... in that screen, have a row on top that shows ~~to~~ nearest plant ... or closest area to find.

Overall, how would you rate the app on a scale of 1 to 10 (10 being a good app and 1 being a poor app)? Explain if you want to.

8

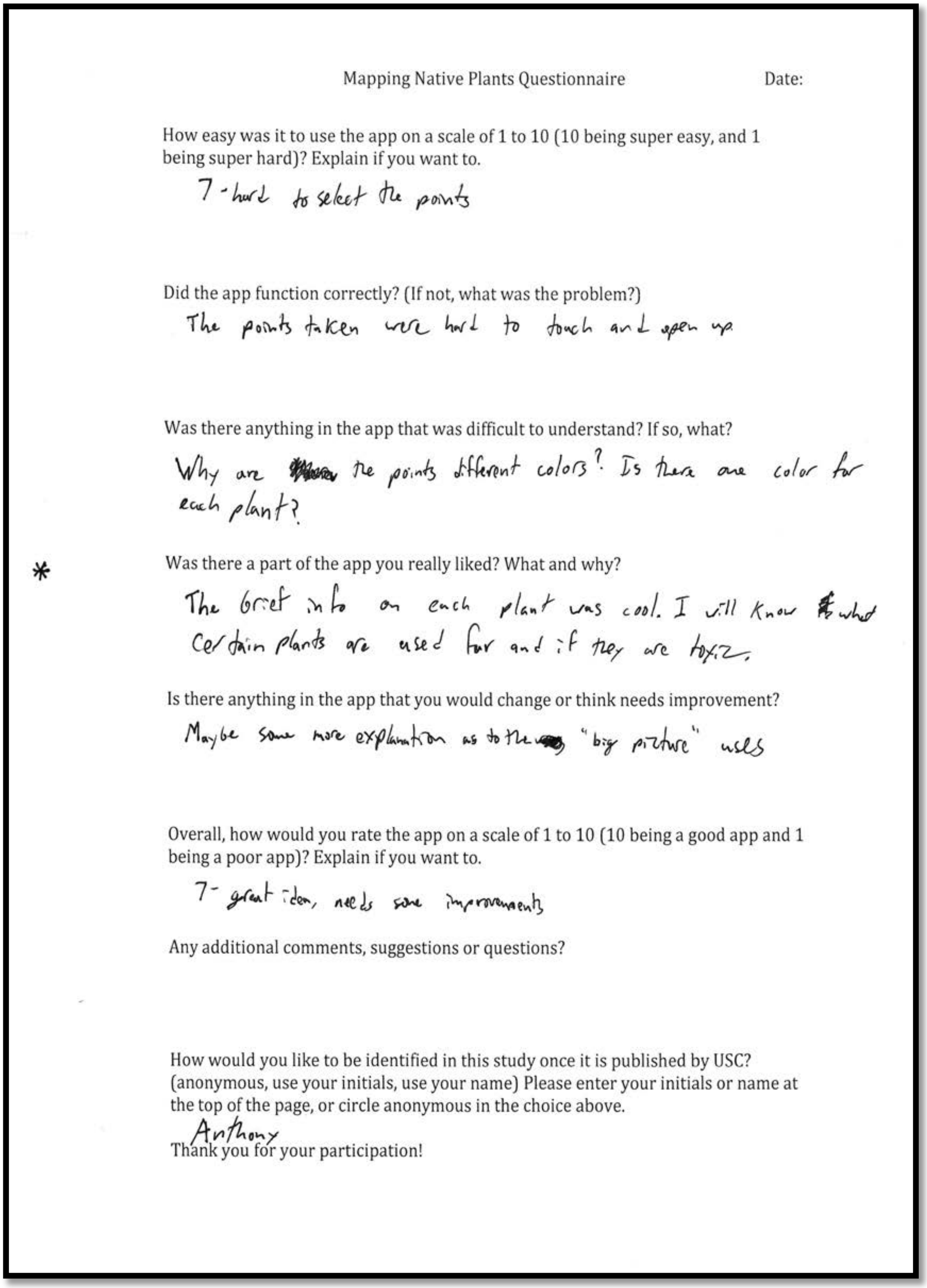
Any additional comments, suggestions or questions?

Good luck

How would you like to be identified in this study once it is published by USC? (anonymous, use your initials, use your name) Please enter your initials or name at the top of the page, or circle anonymous in the choice above.

Thank you for your participation!

Appendix B: User Questionnaire 1



Mapping Native Plants Questionnaire Date:

How easy was it to use the app on a scale of 1 to 10 (10 being super easy, and 1 being super hard)? Explain if you want to.

7 - hard to select the points

Did the app function correctly? (If not, what was the problem?)

The points taken were hard to touch and open up.

Was there anything in the app that was difficult to understand? If so, what?

Why are the points different colors? Is there one color for each plant?

* Was there a part of the app you really liked? What and why?

The brief info on each plant was cool. I will know what certain plants are used for and if they are toxic.

Is there anything in the app that you would change or think needs improvement?

Maybe some more explanation as to the "big picture" uses

Overall, how would you rate the app on a scale of 1 to 10 (10 being a good app and 1 being a poor app)? Explain if you want to.

7- great idea, needs some improvements

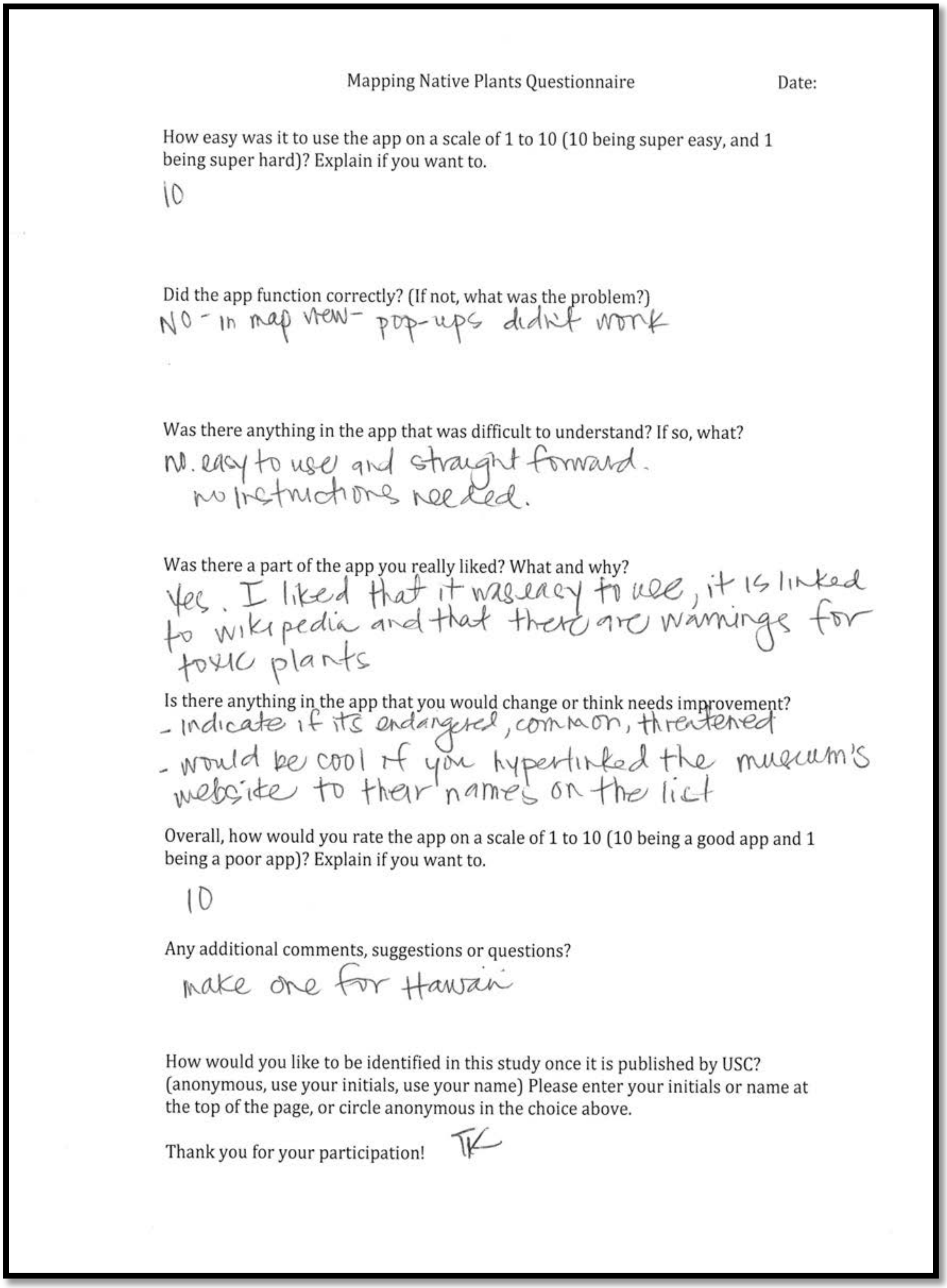
Any additional comments, suggestions or questions?

How would you like to be identified in this study once it is published by USC? (anonymous, use your initials, use your name) Please enter your initials or name at the top of the page, or circle anonymous in the choice above.

Anthony

Thank you for your participation!

Appendix B: User Questionnaire 2



Mapping Native Plants Questionnaire Date:

How easy was it to use the app on a scale of 1 to 10 (10 being super easy, and 1 being super hard)? Explain if you want to.

10

Did the app function correctly? (If not, what was the problem?)

NO - in map view - pop-ups didn't work

Was there anything in the app that was difficult to understand? If so, what?

no. easy to use and straight forward.
no instructions needed.

Was there a part of the app you really liked? What and why?

Yes. I liked that it was eacy to use, it is linked to wikipedia and that there are warnings for toxic plants

Is there anything in the app that you would change or think needs improvement?

- indicate if its endangered, common, threatened
- would be cool if you hyperlinked the museum's website to their names on the list

Overall, how would you rate the app on a scale of 1 to 10 (10 being a good app and 1 being a poor app)? Explain if you want to.

10

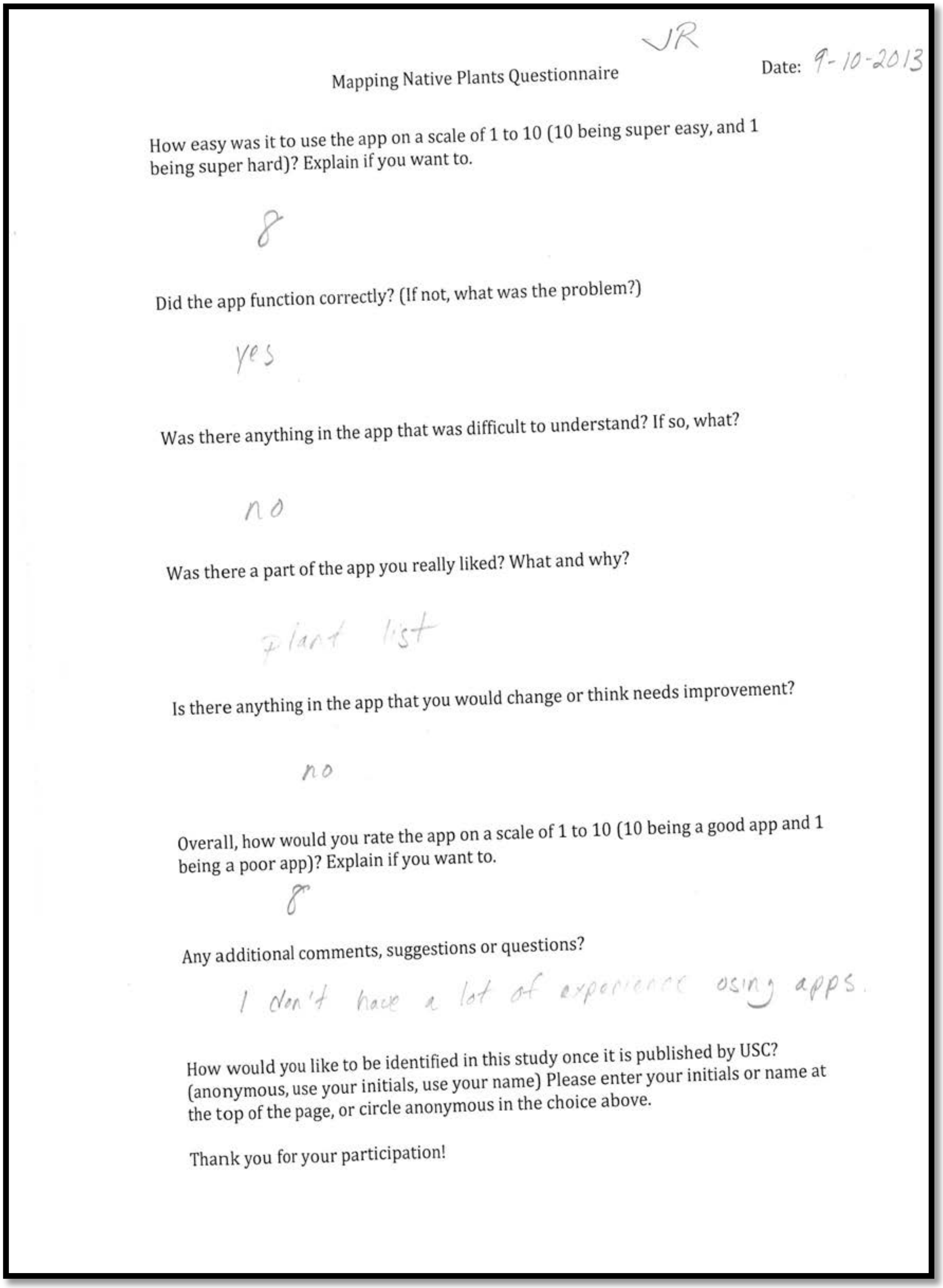
Any additional comments, suggestions or questions?

make one for Hawaii

How would you like to be identified in this study once it is published by USC? (anonymous, use your initials, use your name) Please enter your initials or name at the top of the page, or circle anonymous in the choice above.

Thank you for your participation! TK

Appendix B: User Questionnaire 3



JR

Mapping Native Plants Questionnaire

Date: 9-10-2013

How easy was it to use the app on a scale of 1 to 10 (10 being super easy, and 1 being super hard)? Explain if you want to.

8

Did the app function correctly? (If not, what was the problem?)

yes

Was there anything in the app that was difficult to understand? If so, what?

no

Was there a part of the app you really liked? What and why?

plant list

Is there anything in the app that you would change or think needs improvement?

no

Overall, how would you rate the app on a scale of 1 to 10 (10 being a good app and 1 being a poor app)? Explain if you want to.

8

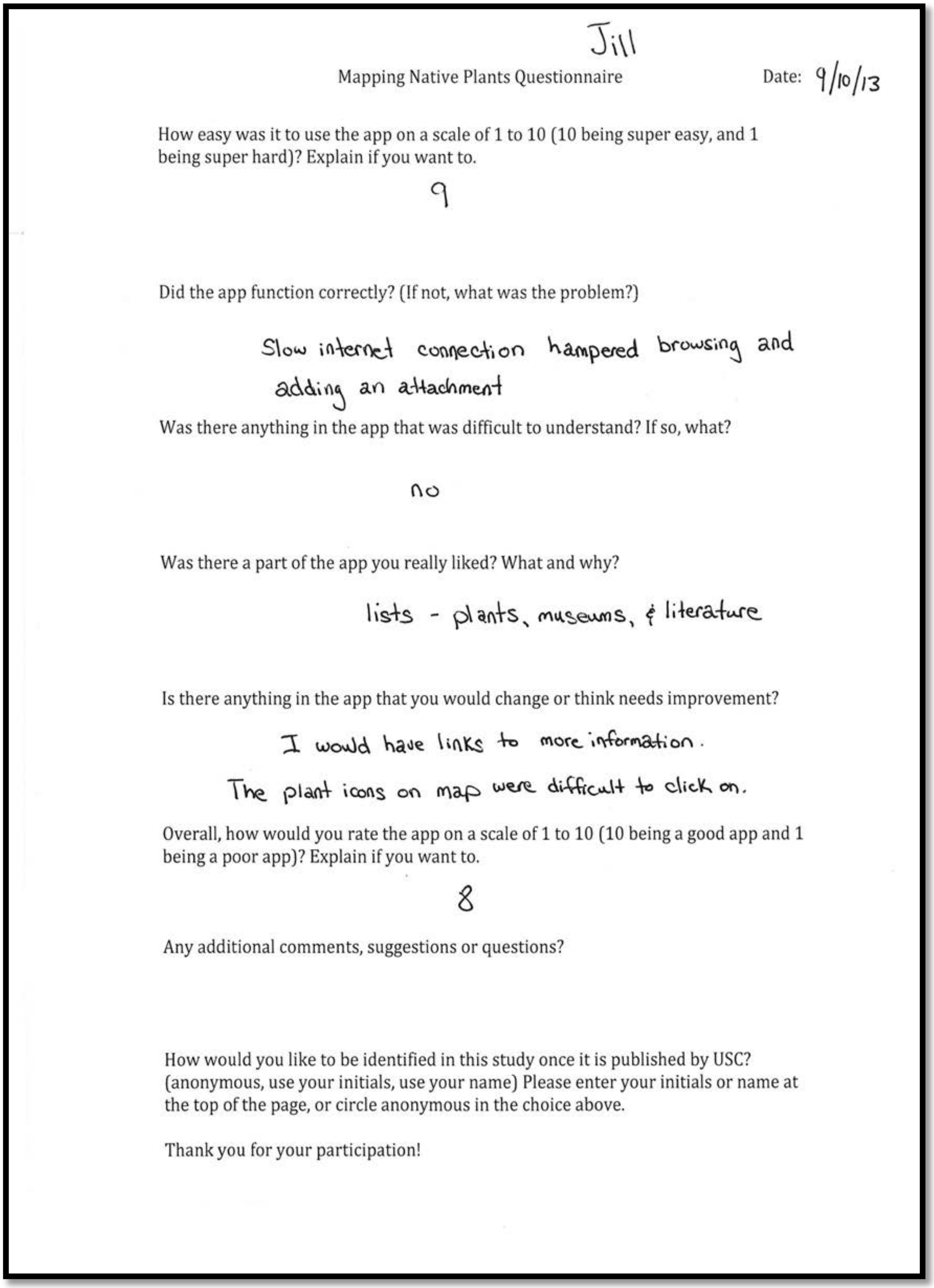
Any additional comments, suggestions or questions?

I don't have a lot of experience using apps.

How would you like to be identified in this study once it is published by USC? (anonymous, use your initials, use your name) Please enter your initials or name at the top of the page, or circle anonymous in the choice above.

Thank you for your participation!

Appendix B: User Questionnaire 4



Jill

Mapping Native Plants Questionnaire

Date: 9/10/13

How easy was it to use the app on a scale of 1 to 10 (10 being super easy, and 1 being super hard)? Explain if you want to.

9

Did the app function correctly? (If not, what was the problem?)

Slow internet connection hampered browsing and adding an attachment

Was there anything in the app that was difficult to understand? If so, what?

no

Was there a part of the app you really liked? What and why?

lists - plants, museums, & literature

Is there anything in the app that you would change or think needs improvement?

I would have links to more information.

The plant icons on map were difficult to click on.

Overall, how would you rate the app on a scale of 1 to 10 (10 being a good app and 1 being a poor app)? Explain if you want to.

8

Any additional comments, suggestions or questions?

How would you like to be identified in this study once it is published by USC? (anonymous, use your initials, use your name) Please enter your initials or name at the top of the page, or circle anonymous in the choice above.

Thank you for your participation!

Appendix B: User Questionnaire 5