Team: Floradex

Preliminary Architecture

**Summary**: Our goal is to develop an iPhone app that can help a user identify plants through a dichotomous key or keyword search.

Components:

**Homepage**: This gives the user the ability to choose what type of search they would like to perform. Using either the dichotomous key or the keyword search, the user will be able to look directly for a specific plant. Depending on their selection, they will be directed to whichever of these pages is appropriate. They will also have the option of looking at their MyPlants list rather than executing a search from this page as well. The main requirement of the home page is to allow the user to access the actual functionality of our application.

**Dichotomous Key:** Starts with a broad question (such as “Is it a tree, shrub, or flower?”) and narrows down to more specific as more questions are answered. Each question will give only a few options, with accompanying pictures to clarify, to make this search easier for the user. As each question is answered, the answer to the question is stored in a list of keywords to plug into the keyword of the database. We are planning to store the question hierarchy as a tree, with pictures and keywords stored in the nodes that match each question.

**Keyword Search**: The user is given options of keyword fields they can search by, such as location, color, or common name. Once the known fields are filled in, the list of words will be passed to the database for a search to be executed.

**Plant Database**: The database will be implemented using SQLite. It will contain the keyword information about the plants, as well as a link to the fact sheets. Each plant’s entry will contain information regarding habitat, scientific name, leaf type, etc. that match the keyword search fields. Once a list of probable plants based on the input from the search has been found the information will be passed on to the potential results page. The database works completely in the background of the app, so the user will never be interacting with it directly.

**Potential Results**: Based on the results of the database search, the closest matching plants will be shown in a list to the user for browsing. From there they can choose which one is the plant they were looking for and be directed to the fact sheet for that plant.

**Fact Sheets**: Each plant will have its own fact sheet stored as a wiki page, which will include a picture, growing conditions, interesting facts, and other relevant information. Each wiki page will give the user the opportunity to store the plant in their MyPlants list if they want to.

**MyPlants**: The user’s personal list of plants that they have saved from their searches. This will be implemented using SQLite, making it simple to pass plants from the plant database to the user’s favorites list. Using the iPhone’s SQL capabilities, we will be able to make this list interactive for the user, allowing them to browse, look at the fact sheets for their plants, rearrange, or remove plants at any time they wish.

From the developer’s perspective:

Home Page

Keyword Search

Dichotomous Key

MyPlants

Plant Database

Potential Results

Fact Pages

SQL query

From the user’s perspective:

Home Page

Keyword Search

Dichotomous Key

Potential Results

MyPlants

Fact Pages

**Interfaces**

**External interface**: Because the iPhone screen is relatively small, we want to make sure that each page has only the necessary features to keep it from becoming cluttered. We are looking at using large buttons and text input boxes to allow for easy navigation throughout the system. The prototype of our interface is in a separate document, and provides a visual display of different pages within the application.

**Internal interface**: The primary interface that we are looking at is passing SQL information between internal components. Both the plant database and MyPlants rely on SQL databases, which means that, given that they have matching fields for the plants, they will be able to pass data back and forth without trouble. SQLite, the implementation we are using, has good tools to interact directly with iPhone development software – we will be able to pull data out of the databases and format it within our pages without too much trouble as well. As users input text into the keyword search, it can easily be converted to an SQL query using regular database searching tools. The fact pages are currently being stored within the database, but in the future (after version 1.0) we hope to use the features of wiki pages to allow our users to contribute information and pictures to these pages as well.

**Key issues**: Initially, we had trouble narrowing down our list of desired features to provide in version 1.0 of our application – we had come up with a very long list of requirements in the planning stage of the project. The features we have decided upon to provide in this version came out of the following rationale: the main functionality of our system was always intended to be plant identification; other features were fun and interesting but weren’t necessary to fulfill this basic task. Using a database to implement keyword search and a dichotomous key are new territories for us, but we think that they are the key features of our product and should be focused on first. All other features can build on the functionality of our identification system after that is implemented.

**Major issues still outstanding:** There are a few different areas that we are still unsure of. Since none of us have experience with large databases, more in-depth research will be required to figure out the specifics of our implementation, especially the interactions between our different SQL local databases. We also need to look into how to sync a local database with our main database for updates. Because this initial version has only a few features included, we are unsure of how our product will be better than current apps that are available – many of the features that made ours stand out were removed for the initial version. Finally, the logistics of a user-input based search bring up a few issues. We have learned that drop-down menus to fill in fields are extremely difficult to use on touchscreens; at the moment we assume that users input “correct” keywords into the search; how do we deal with situations such as “yellow” vs. “gold” as an input, and how would that be handled in the database information; for inputting colors, would providing a gradient slider or simply giving swatches to choose from be the most effective? We will need to start reaching out to users to see what would be most usable in our app.