Software Architecture Specification

Overview

The Flora of Indiana application is a mobile application which will reside on users’ physical mobile devices and will allow the user to access information stored on a variety of interconnected databases. The application software will be compiled to both Android- and iOS-comprehensible code, will be distributed via each operating system’s primary application store, and will utilize a third-party tool called IDigBio to interact with a variety of databases.

Subsystem Decomposition

**Database** – The data storage mechanism in our case will be a database or—more likely—a series of geographically disparate databases which will be maintained by various third party institutions (colleges, herbaria, and scientific organizations). We will interact with the databases through IDigBio.

**IDigBio** – This system is a third-party application which communicates with diverse databases and coordinates the retrieval and updating of information from those databases. IDigBio sits “on top of” the databases and is responsible for providing a uniform access point for the data via a defined API.

**Application** – The application will be developed and maintained by our team and will provide the logic which will govern the organization and presentation of data as well as the options for data interaction which will be available to the user.

• **Data Access Layer** – This system will be responsible for retrieving data from sources external to the application (the database via IDigBio) and will also be responsible for handing information to the rest of the application.

• **Plant Data Model** – This system will function as a data structure and a packaging mechanism. It will be the organizing container for data retrieved by the data access point. The model will define a valid format to which retrieved data will be required to conform. This will be the “object-maker,” and individual plant data objects will be able to be interacted with by other application subsystems.

• **Controllers** – This system will be responsible for accepting user input from mobile devices and converting that input into commands that are capable of acting upon the data model and views.

• **Views** – The views will provide visual interfaces and affordances to the user, communicating to the user which actions are possible within the system. The views will be the only means through with the user will access or manipulate the model and the data which it represents.

**User Interface** – Users will use touch screens to interact with our system on physical mobile devices. The user interface will be rendered via an Android or iOS operating system and API/libraries, depending upon the device. This system will be designed and maintained by a third party.

Insert some picture here

Hardware/Software Mapping

**Database** – The “database” in our case will actually consist of multiple physical and/or virtual databases which will be maintained by multiple third-party institutions. Interaction with these databases will be mediated through IDigBio, so access with the databases will be abstracted (from the developer and the application perspective), and we will not be responsible for maintaining the databases.

**IDigBio** – This application will probably be hosted on a third party machine or a series of machines and will be maintained by a third party.

**Application** – The application code will be deployed to the Android and iOS application stores, and users will be able to access the code from there and “deploy” the software to their own physical devices by downloading it. Once downloaded, the application will reside on the user’s physical device.

**User Interface** – The user interface will reside entirely on the user’s device, which will be maintained by the user and the third parties that manufactured it and provided the operation system.

Persistent Data Management

Data will be stored in two locations:

• in physical (or virtual) databases, which will be maintained by third parties, and

• in data objects which will be structured and validated by the application logic and will reside on the user’s device.

The database will contain a large number of details about individual plant specimens and species. The majority of this information is unknown to us currently, as we are still in the process of gaining access to the databases.

The application will maintain a model of individual plant species based on a subset of information which it will retrieve from the external databases.

The Plant Model will require the following information:

• **ID** – The external database appears to use a numeric primary key. If this is the case, we will use the same numeric ID within our system to facilitate simpler communication

• **Genus** and **Species** – The scientific genus and species name, two data points which, together, will provide a functional (and traditional) primary key.

• **Thumbnail or identifying photograph** – A small, representative photograph of the plant species, to aid in identification of the plant.

• **Description** – A description of the plant, to aid in identification.

• **Qualities** – A list of distinct plant qualities upon which the user will be able to filter species.

Access Control and Security

All users of this application will have the same level of access. The application will be downloaded from an application store, and access to that store will be managed by Google or Apple, depending upon the device. No authentication will be performed. Furthermore, our application will only provide read-only access to the user, so data will not be able to be updated from our application. To prevent malicious use of the read-only access, we will limit the number of requests which an individual device will be able to make to the database.

Global Software Control

The system will be event-driven, with the user submitting a series of commands to the system, and the system responding by providing modeled data through a series of views.

The views will be as follows:

• **Landing** – This page will display the title of the application and will function as the user’s first point of contact with the application. The user’s only option for interaction will be to tap the screen, which will allow the user to progress to the **PickFromThree** view.

• **PickFromThree** – This view will allow the user to select between three options: the ability to search based on GPS coordinates, the ability to search through navigation of a plant identification key, or the ability to navigate a list of alphabetically- and hierarchically- organized plant species. Selection of the GPS-based search option will take the user to the **GPS** view. Selection of the key-based search option will take the user to the **Key** view. Selection of the direction navigation option will take the user to the **Known** view.

• **GPS** – This view will display a list of nearby plant species. If the user clicks on an individual plant species from this view, the user will be taken to the **Profile** view for that species. The user will also be presented with a button which will allow the user to start over, returning to the **PickFromThree** view. The user will also have the option to select a “refine” option, which will take the user to a view by the same name (**Refine**).

• **Key** – This view will display a brief list of distinguishing plant characteristics. The list which is displayed at any given point will be determined by application logic which will be informed

by knowledge of plant taxonomic keys. Each set of choices will be grouped based on mutual exclusivity, and selection of an option from each set of choices will provide filtering data for the application so that individual plant species can be identified. After one option has been selected, the Key view will transition into the **Refine** view, which will allow the user to continue selected options while also viewing a list of filtered results. The Key view will also contain a “back” button, which will return the user to the **PickThree** view.

• **Known** – This view will provide the user with a text-based, treelike list of all plant species within the system. The list will be organized hierarchically, by family, then genus, then species, with each subgroup organized alphabetically. The user will be able to navigate the list by swiping up or down. If the user clicks on an individual plant species from this view, the user will be taken to the **Profile** view for that species. This screen will also provide a “back” button, which will return the user to the **PickThree** view.

• **Refine** – This view will divide the screen into a top and bottom half. The top half will display the same types of choices which will be accessible from the **Key** view. The bottom half will display a user-navigable list of plants which contain all of the qualities upon which the plants have been filtered. If the user clicks on an individual plant species from this view, the user will be taken to the **Profile** view for that species. This view will also contain a “back” button, which will take the user “back” one option, removing that option from the set of characteristics which will be used for filtering, and a “start over” button, which will return the user to the **PickFromThree** view.

• **Profile** – This view will display a full-screen image of the plant selected, along with the plant’s genus and species, a brief description of the plant, and the filterable qualities which apply to the plant. A small “x” in the top corner of this window will allow the user to exit the Profile view and return to the **GPS**, **Known**, or **Refine** view, depending upon how the user arrived at the Profile view. The user will also be able to swipe from left to right in order to display alphabetically adjacent species.

Boundary Conditions

The databases will be maintained by third parties. The raw source code will be uploaded to GitHub and will be accessible and compilable by anyone who wishes to access it. The compiled code will be deployed by the developers to the application stores on which our application will initially be stored. This will likely be a variety of cloud servers maintained by the third parties which offer these application stores. The compiled code will then be downloadable by the users and will be deployed to their physical devices, where the application will be bootable by clicking on the application’s icon on their device. The application will be shut down by whichever operating-system-specific method the user chooses to utilize.

The underlying data will be managed by third party organizations, and no maintenance of the data will be required outside of any maintenance that those third party organizations would have already performed. The same will be true of the application hosting servers and the users’ devices. Any maintenance will be performed by third parties.

If changes are made to IDigBio, it may be necessary for an application developer to periodically update the application and re-deploy it to the third party application stores. This is the only foreseeable maintenance.

Bulk downloads of the database’s data is already a service provided by the maintainers of the IDigBio software.

Application errors and exceptions will be handled programmatically through code. Physical hardware errors will be handled by the users’ device manufacturers and OS developers.