Lessons Learned:

MOBILE APPLICATION AS A TOOL FOR THE CAPTURE OF DATA FROM PLANT SPECIMENS, IN THE PROCESS OF COLLECTING BOTANICAL SAMPLES IN COLOMBIA

Upon reviewing the methodology used, it's evident that there are several alternatives to solve the problem. Initially, a form was considered that would use all the fields from the Excel files typically used when collecting botanical samples. This process involved writing the information for each sample in a field notebook, and upon returning to the city, this information would be recorded in an Excel file. The application aimed to create a form that would export the botanical information directly to an Excel file, thus simplifying data entry and processing.

In summary, a dialogue was first initiated with various environmental professionals to determine which needs could be addressed by an application. This turned into a list of requirements that was addressed by adding functionalities that would solve them:

A. The application will operate entirely offline in all its functionalities.

B. Georeferencing will be allowed through the mobile device's GPS and will automatically identify the corresponding department, municipality, national natural park, and indigenous reserve for the sampled point.

C. The precision of georeferencing will be displayed to determine the reliability of the coordinate provided by the phone.

D. The information for each sample must be saved in a personal database within the application, and all this information should be viewable in a section of the app.

E. The main collector's collection number will increase automatically as botanical samples are collected.

F. Sample information must be exportable to an Excel file in the Herbarium label format, compatible with the Darwin Core system.

G. The user will have the option to select the botanical family and genus of the sample using dropdowns and an autocomplete system.

H. The form will allow the addition of a sample to the database with the minimum required information, which is:

- Department.
- Municipality.
- Locality.

- Latitude and Longitude coordinates.
- Individual description.
- Main collector's name.
- Main collector's acronym.
- Collection number.

I. The collection date will be automatically set to the sampling day.

J. The collector's name, their acronym, and the collection number should be automatically filled for all samples after being configured.

However, not much attention was paid to UX/UI design. For this, it would be necessary to create sketches and mockups tailored to its use in forests, which would entail research and on-site testing costs. This is an issue that is often overlooked in software development, and nowadays, it's even more significant than the technical aspects of programming. An application that isn't ergonomically and visually comfortable is doomed to fail, regardless of the technical prowess with which it was made. Therefore, the primary recommendation is to deeply emphasize the UX/UI area.

From a technical standpoint, it was noted that not separating the code's logic from the interface logic caused many difficulties, perhaps resolved not with the most optimal or ideal code. This also made testing challenging as it wasn't possible to separate the code to test it in isolation.

One consideration would be to use the MVC architecture for a future version of the application, which would streamline the development and testing process. On the other hand, it was decided to use XML files for the application's views and Kotlin (kt) files that connected and controlled these views.

The modern way to do this is with JetPack on Android, where the visual aspect of the application is controlled via code, making it easier to implement an architecture like MVC. It was decided not to proceed this way because it's a very recent technology, and the application presented challenges that were uncertain if they could be resolved using JetPack. The older method has been around for many years, implying that there's a larger community knowledgeable in that area.

Feedback from Phase 1 and 2:

The APP project focused on developing a mobile app for capturing and processing plant specimen data. Key functional and non-functional requirements identified include:

Functional Requirements

1. **Offline functionality**: The app operates fully offline.
2. **Georeferencing and automatic location detection** for departments, municipalities, and protected areas via GPS.
3. **Data export** to an Excel file compatible with the Darwin Core standard.
4. **Automation** in assigning collection numbers to botanical samples.
5. **Data capture form** with specific fields for sample description.
6. **Selection of botanical family and genus** through dropdown menus and autocomplete.

Non-Functional Requirements

1. **Usability**: User-friendly interface for field data capture.
2. **Performance**: Efficient operation on mobile devices without excessive resource consumption.
3. **Security**: Protection of stored data against unauthorized access.
4. **Compatibility**: Functions across a variety of Android devices, ensuring broad accessibility.

System Design

The system design of the APP project is based on a structured methodology across several phases, utilizing specific tools and technologies for the mobile app development. Here's a detailed overview of the system design based on the provided report:

Development Environment and Programming Language

- **Android Studio**: Chosen as the official Integrated Development Environment (IDE) for Android, designed to facilitate developers in creating high-quality Android apps. Android Studio offers a broad range of tools that simplify app development, including a code editor with syntax highlighting, user interface design, an integrated app debugger, support for multiple devices, and an Android emulator for testing.
- **Kotlin**: Selected as the primary programming language due to its modern features, expressiveness, and a clean, concise syntax that enhances type safety, reduces errors, and increases code efficiency. Kotlin is interoperable with Java, allowing easy integration with existing Java code, thus facilitating the use of Java libraries and tools alongside Kotlin.

Design and Functionalities

- **Interface Design**: Implemented each design and requirement using Android Studio's design modality for visual appearance, while programming modality was used to specify each interaction with the app.

- **SQLite Database**: A SQLite database was configured to store and read the geographical information and data of collected botanical samples. ShapeFile files of geographical entities were converted into geographical tables in the database to perform intersection operations between geolocation coordinates and polygons in the geographical tables.
- **JSON Files**: The app uses two JSON files to read information of each department with its municipalities and each arboreal botanical family with its genera, facilitating selection in the form fields.
- **Plugins (Libraries)**: Various libraries were integrated into the app development, including:
    - **Spatial-Room**: To convert the database into one with Geographic Information System (GIS) capabilities.
    - **ExpandableFab**: For a floating button with expandable secondary buttons, used for the exportation of samples.
    - **SDP**: A scalable size unit to adapt each visual element to different mobile devices.
    - **UCE Handler**: A window to capture and allow sending information about unexpected errors in the app.

This system design reflects a comprehensive and detailed approach to developing a robust mobile application, focusing on usability, performance, security, and compatibility, aligned with the identified functional and non-functional requirements for the project.

Risks and Challenges

Potential risks that this project might face:

1. **GPS Accuracy Dependence**: The application relies on GPS for georeferencing botanical samples. GPS accuracy can be affected by environmental or technological factors, which could compromise the precision of location data.
2. **Data Security**: Although the application does not collect personal data, the security of the collected botanical sample data is crucial. There's a risk of data loss or unauthorized access to data stored locally on the device.
3. **Maintenance and Updates**: The need to keep the app updated with the latest security and functionality improvements, as well as ensuring compatibility with new Android versions, represents an ongoing challenge.
4. **User Interface and Experience**: Designing an intuitive and user-friendly interface is critical for user adoption and satisfaction. Poor interface design decisions could limit the app's utility for end-users.

5. **Adaptability to Changes in Data Standards**: The app must be able to adapt to changes in Darwin Core standards or any other relevant data export standards, which could require significant system updates.

Component diagram

Below is a diagram representing each section and part of the application design and development.