**Module 8 Capstone Project**

Lucy Mills

Colorado State University, Global

CSC480 - Capstone Computer Science

Dr. Lori Farr

June 11, 2023

**Topic Approval**

To make my capstone software project applicable to my life, I want to spend this quarter developing a mobile application for managing a small business. Working with Android apps last quarter was both engaging and rewarding for me, so I would like to continue learning about mobile app development and study more advanced concepts. The current business problem at my job is that we are lacking a way to properly document hours, locate pay stubs, and see updates about changes in the schedule. An organized and well-developed app could be a great solution to these current issues.

The topic I am studying for my capstone project is ‘an android application for work hours and scheduling’. Of course, I plan on adding more features to the app beyond these two elements, but these were a solid starting point for the overall structure and functionality of the app. While there are many scheduling apps currently available, my boss would prefer to have something more private and secure for handling this information. Also, the app will need to be able to add this information to an existing Google sheet for tax purposes. As the structure is simple, overall, it should be achievable in the time frame of 8 weeks.

The planning phase of designing an hours/ scheduling app includes accessing the project’s goal and defining features, then determining the best practices and processes for creating the application. Some features to consider for a scheduling app for ease of use are making scheduling simple, having an editable calendar, sending reminder texts automatically, and including roster management (Kvartalnyi, 2022). Additionally, helpful practices for the development of the app involve building a minimum value product, increasing adaptability, and monitoring performance (Kvartalnyi, 2022). With these ideas in mind, I now have more direction for developing my capstone software project.

**Proposal**

Utilizing the Android Studio program, I will be able to develop a functional, multi-activity based mobile application for EB Outdoors employees. Because the Google Play store is more accessible in terms of uploading a custom app to the public, the application will be Android based instead of IOS. Additionally, IOS app development follows a different structure and uses alternative software which I am unfamiliar with making it a bad choice for my capstone project as I want to set myself up for success.

Many classes are involved in creating an application with multiple activities in Android Studio. Currently, the app has four activity pages each with Java file for handling the page’s logic and an XML layout file for creating the user interface for the activity. There are additional XML files for saving values being used by multiple files such as colors, dimensions, strings, and styles. Three more activities have to be added to the app for 1) the login page, 2) the hours input page, and 3) the payments page; meaning that at least six more files will need to be added to the project to make it functional.

Two of the biggest challenges I will have to face when building this app include properly setting up the login accounts with an SQLite database and accurately adding data from the app into an existing Google sheet. An SQLite database will store the employee’s login data to ensure that only employees are able to access the information on the Google sheet and document (JackRutorial, 2018). Next, the hours activity will have an easy-to-follow interface for inserting data into the Google sheet. To do this, an app script for inserting the data online is made online in the Google script console and an Http Post request is setup in the Android app, which is written in the activity’s Java file (Crazy, 2017). Following the steps outlined on these websites, I should be able to properly setup the login and hours activities in my own application easily.

**Software Design**

**Figure 1**

*EB Outdoors Class Diagram*

*A picture containing text, receipt, screenshot, diagram

Description automatically generated*

Note. *The class diagram for my capstone project illustrates the different activities available to a user in the app depending on if they are a logged in employee or not (Balafkan, 2018).*

**Figure 2**

*Employee Account Interactions State Machine Diagram.*

*A picture containing text, screenshot, diagram, font

Description automatically generated*

Note. *The state machine diagram created for the EB Outdoor app analyzes the two possible account interactions for a user to make: logging into an existing account or creating a new account (Lucidchart, 2023).*

**Testing Methodologies**

As there are many kinds of software developing methodologies available, there are associated testing methods that vary from each other. To begin, in the waterfall model testing starts after the development of the code is completed, which works well for small companies, but this model makes correcting bugs more difficult making it an unsuitable choice for bigger projects (Camacho, 2023). While the agile model can be used for the development of larger products because it is an incremental model that tests after every iteration and at the final completion of the project. The iterative testing model involves using basic pieces of software to review and amend in small sections at time, which can work well for very large and complex applications as bugs are detected earlier. Finally, the DevOps approach implements continuous testing, where a testing team works in collaboration with developers to complete testing as the software is being built (Camacho, 2023). Since I am developing the EB Outdoors application myself, I have been using the DevOps testing approach and double as programmer and tester to test each section of my code after its written to find issues faster. It can be quite frustrating to locate an issue in the code after editing several files, so testing the application with the emulator after altering each file makes patching bugs much easier.

**Use Cases**

In short words, use cases are processes that are used to review the software requirements and features in the system. Some of the factors that use cases outline are the intended users, the intentions of the user, the steps a user has to take to achieve a certain task, and the reaction of the application to the user’s input (Indeed, 2023). The formatting of use cases involves naming and number each case with appropriate actors and descriptions. The five use cases that the EB Outdoors system has to respond to are listed below.

**Use Case #1:** an average user opens the EB Outdoors app.

Description: Someone launches the application and is presented with the main activity that presents the available options in the form of four buttons: Mission, Programming, Sponsorship, and Employee Portal.

Actors: The potential actors involved are future clients who are interested in learning about the EB Outdoors program or employees who are about to log into the portal.

Stakeholder (for all use cases): The owner of EB Outdoors, Emma Brown, is the stakeholder who is responsible for uploading payment information for employees and verifying employee hours in Google.

Precondition: The user clicks on the EB Outdoors icon to launch the application on their personal device.

Basic Flow: The user is redirected to the appropriate activity upon pressing a button on the home screen.

Alternative Path: The system fails, or the user is redirected to the incorrect page after pressing a button.

**Use Case #2:** An employee uses the EB Outdoors app.

Description: An employee launches the application and presses the button labeled Employee Portal.

Actors: The actors involved are employees who are about to log into the portal or create an account.

Precondition: The user intends on logging into the employee portal.

Basic Flow: The employee is redirected to the employee login page upon pressing the button on the home screen.

Alternative Path: The system fails, or the employee is redirected to the incorrect page after pressing the button.

**Use Case #3:** An employee wants to create a new account.

Description: An employee plans on creating an account for the SQLite database.

Actors: The new employee.

Precondition: The employee opens the app, clicks on the employee portal on the home screen, and then chooses ‘create account’ option underneath the login button.

Basic Flow: The employee is redirected to the account creation page upon pressing the button on the employee login page where their credentials are added to the SQLite database.

Alternative Path: The employee is redirected to the incorrect page after pressing the button, or the app fails to add the information to the database.

**Use Case #4:** An employee logs into the EB Outdoors employee portal.

Description: An employee uses their credentials to log into the employee portal to edit their hours or view pay stubs.

Actors: The employee.

Precondition: The employee has an account, fills out their credentials, and presses the login button.

Basic Flow: The employee is redirected to the employee navigation page upon pressing the button on the login.

Alternative Path: The login is not recognized, the system crashes, or the employee is redirected to the incorrect page after pressing the button.

**Use Case #5:** An employee wants to view their pay stubs from the EB Outdoors app.

Description: An employee logs in to see their payment history.

Actors: The paid employee.

Precondition: The employee logs into the portal, navigates to the payment activity, and clicks on the ‘payments link’ button.

Basic Flow: The app launches the link in the system’s browser to show employees their pay stubs.

Alternative Path: The button does not respond, the link fails, or the application crashes.

**Inputs and Results**

Because my capstone project is an Android application, the inputs are interactions from the user in the form of touches or input text, and the results are the responses of the application. My application appropriately responds when buttons are touched, in terms of not terminating the program or sending users to the incorrect location. Additionally, all buttons that use a link to redirect the user to a website appropriately launch webpages for the user. The login activity takes input from the user in terms of a username and password, which will result in unlocking the rest of the employee activities. The edit hours page takes in the user’s name, number of hours, date, and tasks performed to be added to a Google sheet but needs to be debugged as it currently throws an ‘error 401’.

**Performance Expectations and Failures**

In terms of a performance review, the expectations include the job description for the tasks being completed (Dziuba, 2021). The intended performance expectations that I set at the beginning of this term were for the application to run with a new theme, added activities for employees, and links to outside sources. Of course, working on a team of one, there were going to be some shortcomings to own up to. While I was able to get the application running with all new activities, apply two different themes to various activities, and create buttons with working links, the login aspect of the app is lacking logic and the editable hours page to the Google sheet does not work even after hours of debugging. Even though there were some failures I am proud of the EB Outdoors App I was able to create, despite the many issues that occurred along the way, and I can say that I learned very much in the process.

**Capstone Project Recap**

At the beginning of the term, decided to make completing the EB Outdoors Android application my capstone project. I decided to not only upgrade the user experience of the app by creating themes, but to include five new activities to enhance the app with new functionalities. To help me understand the activities needed for the project, their relationships to each other, and the components required, I created a visual diagram of the five new activities that were added to the app included below as figure 3 below. To review the work that I completed this term, I have broken down my findings into completed objectives, works in progress, and system bugs.

**Figure 3**

*EB Outdoors Activity Diagram*

Diagrama

Descripción generada automáticamente

*Note.* I used this document throughout the entire development process as a cheat sheet to help remind me of the different activity functions and their relations.

**Completed Objectives**

Most of the requirements that I set for myself were complete including, building the structure for all five, new activities, creating different themes to be applied to various activities in the app, and including a link to a Google doc for employees to easily check their hours. The two themes created are the ‘basic’ theme for the old pages of the app (including the home screen) and the ‘employee’ theme for all the new activities (Tufan, 2020). Some issues arose with the page backgrounds when trying to apply the themes to the pages of the app, which were resolved by creating drawable files with color blocks inside to fill the background of the activities. The employee payments page is successfully able to launch the Google doc link in the system’s browser to show employees copies of their pay stubs. Screenshots of the EB Outdoors app in execution are shown below in Figure 4.

**Figure 4**

*Execution of App Activities: Main, Employee Portal Navigation, and Employee Payments.*

Captura de pantalla de computadora

Descripción generada automáticamente Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

*Note.* On the far left, the main activity is shown in the basic theme, and the other two employee pages are in the alternative theme. The screenshot on the far left shows the launched link from the app.

**Work in Progress**

Being a singular developer, I had to make a choice on where to spend my time in developing this project. The most complex element of the app was going to be a login system that uses an SQLite database to store users had to become the minimum value product as I was not going to be able to finish it on my own (Massachusetts, 2021). That being said, I still put time in where I could to create the structure for the system in the project’s XML files. The two screenshots of the application in Figure 5 show the activities created for logging into the portal or creating a new account.

**Figure 5**

*Execution of App Activities: Employee Portal Login and Account Creation.*

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

*Note.* The buttons on these pages are functional and redirect the user to the appropriate activities.

**System Bug**

The most frustrating bug that I encountered when working on this project relates to the edit hours page in the employee portal. Many files went into creating an Android function that can interact with a written Google script to edit an online Google sheet, and even after spending hours debugging each file, the page still throws an error ‘false:401’ message when the button is pressed to send to information to the database (Crazy, 2017). Figure 6, below, shows the toast output that the app sends when the user attempts to send their hour information online.

**Figure 6**

*Execution of Employee Hours Activity.*

Captura de pantalla de un teléfono celular

Descripción generada automáticamente

*Note.* After the user presses the button again after receiving the message, the program clears the test fields and prompts the user to once again provide information.

**Conclusion**

Because there were so many different elements involved in developing the application in terms of the user interface and logic, as well as documentation to keep up with, there was a lot I was able to take away from this final term. While the EB Outdoors app needs a bit of tweaking before being put to daily use, I believe that the application could be a viable option for the company to use for employees to keep with their hours and pay stubs in a convenient location. Coding the app came with many challenges that sometimes involved starting from scratch and rebuilding the project to ensure that it had functional features and was up to date with the latest software available.

**References**

Balafkan, N. (2018). UML Diagram. *The Story of Us*. <https://generic.wordpress.soton.ac.uk/thestoryofus/2018/05/02/uml-diagram/>

Camacho, R. (2023, April 18). *Software Testing Methodologies Guide: A High-Level Overview - Parasoft*. Parasoft. <https://www.parasoft.com/blog/software-testing-methodologies-guide-a-high-level-overview/>

Crazy Coders Club. (2017). *Using Google Spread sheet as DataBase Part -2 | Posting ( Insert ) data into sheets from Android App*. <https://www.crazycodersclub.com/android/using-google-spread-sheet-as-database-part-2-posting-insert-data-into-sheets-from-android-app/>

Dziuba, A. (2021). *How to Run a Successful Software Engineer Performance Review [Template Included].* Relevant Software. <https://relevant.software/blog/run-successful-software-engineer-performance-review/>

Indeed Editorial Team. (2023). *Use Cases: What They Are and a List of Examples.* Indeed.com. <https://www.indeed.com/career-advice/career-development/list-of-use-cases-examples>

JackRutorial. (2018). *Android Login with SQLite Database Tutorial*. Learning to Write Code for Beginners With Tutorials. <https://www.jackrutorial.com/2018/03/android-login-with-sqlite-database-tutorial.html>

Kvartalnyi, N. (2022, November 15). How to Make a Scheduling App [Step-by-Step Guide]. *Inoxoft |*. <https://inoxoft.com/blog/how-to-make-a-scheduling-app-complete-step-by-step-guide/>

Lucidchart. (2023). *State Machine Diagram Tutorial*. <https://www.lucidchart.com/pages/uml-state-machine-diagram>

The Massachusetts Institute of Technology. (2021). *Saving data in SQL databases*. Android Developers. <https://stuff.mit.edu/afs/sipb/project/android/docs/training/basics/data-storage/databases.html>

Tufan, B. (2020, April 25). *How to build and apply own custom theme in Android.* Thecodeprogram. <https://thecodeprogram.com/how-to-build-and-apply-own-custom-theme-in-android>