WeatherCoach

Sarah Sollauer, Eric Banach, George Powell
School of Computing & Data Science
Wentworth Institute of Technology

Abstract

WeatherCoach is a mobile app that provides users with accurate weather information and clothing recommendations based on their location. The app features three displays: a weather overview, a detailed forecast, and a clothing predictions display. Users will get all of the weather information that they need and will also be given outfit recommendations that are tailored to the user. WeatherCoach is designed to help users make informed decisions on what to wear, no matter what the weather brings.

Main Features

Weather Overview Display (WOD)
The WOD is the default screen that clearly displays the weather in the selected location. The display only shows the basic information for the day about weather conditions such as hourly temperatures, current conditions, etc.

Forecast Detail Display (FDD)
The FDD is a more in depth version of the WOD, which includes more data about the weather for the week including wind chill, UV index, etc. The user will be able to select which day in the next week they wish to view.

Clothing Preparation Display (CPD)
The CPD will recommend clothing choices based on the current weather at a user-specified location to optimize user comfort when going out. The display will show a printed list of recommended articles of clothing which will aid people who struggle with decision making early in the morning.

Design & Patterns

The architecture follows the Model View Controller (MVC) pattern. The focus of MVC architecture is the separation of a view that the user will see, a controller that takes user input and passes data on to the model, and the model which does data processing and storage.

Views:
The project consists of four views: WOD, FDD, CPD, and main. Each of the displays contains their respective data as discussed in the introduction, while the main acts as a container to hold the three and provide tabbar and swipe view functionality.

Controller:
The controller acts as a middle man between the views and the models. It both requests weather data from the model and updates all of the views respectively. The main engine for the application also resides here.

Model:
The Weather Model interacts with both the API and the database to gather information, parse out the requested data, and passes the data back to the controller. All bulk data processing occurs here.

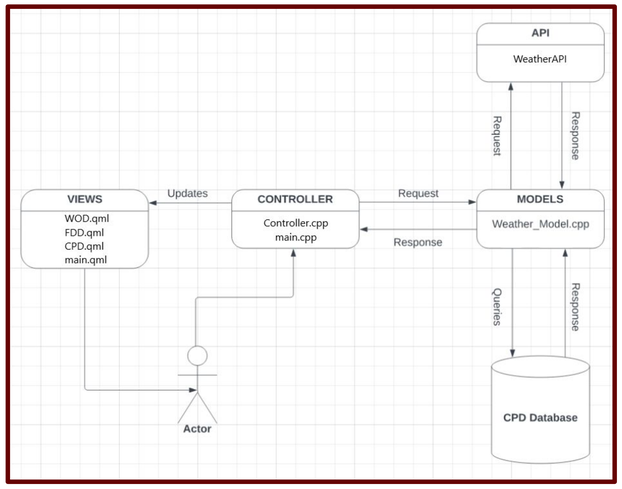


Figure 1: System Architecture

Implementation

The WeatherCoach app was developed using Qt Design Studio & Creator, design tools that allows developers to create high-quality GUIs with backend functionality.

Front-end Design:

The views and engine to run the views were developed in QT QML. The Qt Design Studio allowed the team to quickly create a prototype UI for the system, which was then finalized and connected to the backend in Qt Creator. QML allowed the team to create beautiful graphics while simultaneously tweaking properties to send and receive data to and from the controller.

Back-end Design:

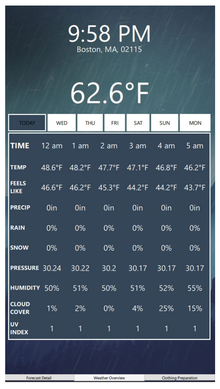
Both the controller and the model were created in Qt C++ through Qt Creator. Both were standard C++ classes that inherited the QT QObject class for integration with the front-end. While the controller primarily consisted of getter and setter functions, the Weather Model made calls to Weather API for a 7 day forecast with the required weather.

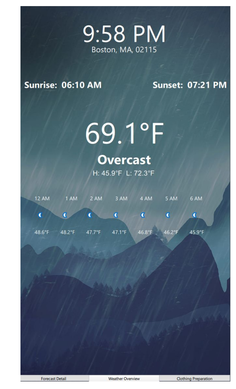
Data Analysis:

To store and manage the app's data, the team used SQLite, a relational database management system. SQLite was chosen because of its efficiency and ease of use, making it well-suited for the needs of the WeatherCoach app.

Results

Through our development process, we were able to successfully implement the core features of our app on the WOD and the FDD, including retrieving weather data, updating displays, and displaying the current time. Although there are features that still need to be implemented, including the hamburger menu, the scrollable timebar, an the reactive background photo, we are proud of the progress we have made so far. Moving forward, we will focus on implementing the remaining features including those on the CPD.





Conclusion

During the development process, we faced challenges familiarizing ourselves with the software used in the time given. Moving forward, we have identified the need to allocate time for this earlier in the project timeline. Our future plans for the app are to have all features fully functional and deploy it in the app store!

Sources

WeatherAPI. https://www.weatherapi.com/. Accessed on 11 Apri. 2023.