ICS 372-01 Fall 2023

Mohamed Yusuf, Steve Hartmann, Jordan Mielke

Project Milestone #1 Report

Project Name:

Climate Watch

Problem Statement:

Have you ever gone to your weather app on your phone, and it shows that it's bright skies and no rain in sight, or maybe it's supposed to be a high of 98 degrees, or maybe you want to see what the weather is like somewhere else around the world. Look no further, we're here to provide you with the utmost accuracy in weather information, allowing you to see not only your weather information, but the weather information for when you're on vacation next week.

Product Objective:

Our application will give you the option to see what the temperature will be like, not only where you are, but you can also select the location to check out what the highs and lows will be for the family vacation next week. You also will have detailed information, allowing you to see the wind and humidity predictions as well as sunset and sunrise.

Functional Requirements:

- The system should list weather for the 5 most popular US cities and the 5 most popular non-US cities when the user first accesses the application. The application will hardcode the list of cities and use the OpenWeather API to fetch the weather data for each city. It will then display the weather for each city in a list view with city name, temperature, a graphic, and button to show more info.
- The system should allow the user to search for a ZIP code and then display the weather for that ZIP code on the list view and the expanded detail's view. The application will use a text field to capture the user input and then use the OpenWeather API to fetch the weather data for that city. If the user enters an invalid ZIP code that is a number but doesn't exist or that is not a number -- i.e., "00000" and "*Scooby Dooby Doo*", respectively -- then an error message popup will be displayed saying the ZIP code is invalid.
- The system should allow the user to view more weather information about a specific city. While the user is on the list view, they can click on the Show More button in the accordion and it will

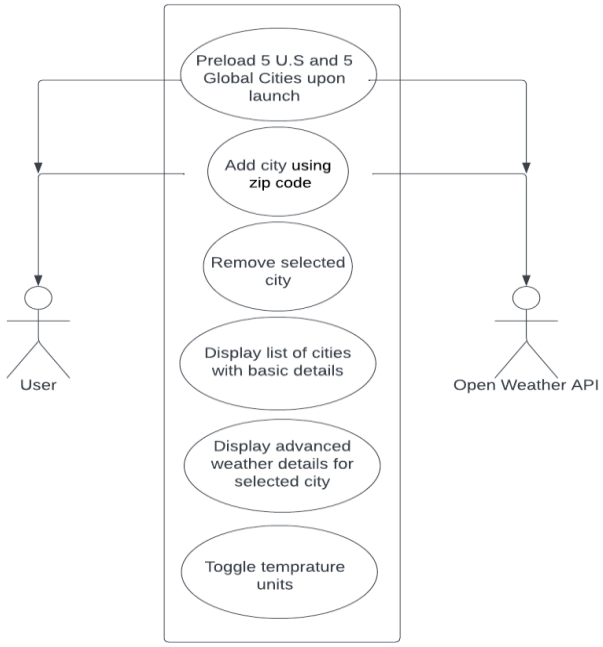
display all the data for the city including temperature, time of day, location, high/low, sunset/sunrise, wind/humidity, and a 10-day forecast depicting the upcoming weather.

- The system should allow the user to choose between displaying the temperature in Celsius or Fahrenheit and by default it will be displayed in Fahrenheit. The application will present a temperature units toggle button on the User Preferences tab that toggles between Celsius and Fahrenheit. If the user chooses a different unit than the current one selected, the application display will be refreshed, and the temperature will be displayed in that unit.
- The system should prompt the user to access their location and then read out their IP address. It will use the Ipstack Geolocation API to convert their IP address to their actual location. It will then use the OpenWeather API to look up the weather for their location and display it in the application. If the user denies the location prompt, there will be a message popup displayed saying the application can't display the weather for their location.

Non-Functional Requirements:

- Desktop application so no mobile support
- JavaFX for GUI
- Include design artifacts and documentation (readme, post-mortem report, etc.)
- Use free Weather API for weather data
    - https://openweathermap.org
- Use free IP geolocation API for converting user's IP address to location
    - https://ipstack.com
- Accept user input and do error handling
- Cache weather data and refresh every 15 minutes
- Hardcoded list of 5 most popular US cities and 5 most popular non-US cities

Use Cases / Case Descriptions:



Use case name	Add City
Use case number	001
System	JavaFX Application
Stakeholders/actors	1. User 2. Open Weather API 3.
Use case goal	To be able to add a city
Primary actor	User
Preconditions	1. User must press a button to execute. 2. City must exist to be able to add to list. 3.
Basic flow	User presses the add city button, displaying an option for the user to type in a zip code or city name. Adding it to the list.
Alternative flows	1. If the zip code doesn't exist, we'll have an error exception, "City doesn't exist" 2. If the city already exists in the list, we'll have an error exception, "City has already been added" 3.If the user inputs special characters, "City doesn't exist"

Use case name	Remove City
Use case number	002
System	Java FX Application
Stakeholders/actors	1. User 2. 3.
Use case goal	Remove Selected City from View
Primary actor	User
Preconditions	1. City must be present in list 2. User must open expanded detail view of city 3. Must have button component to execute deletion

Basic flow	User presses the delete button component displayed in expanded view with cursor. The city is then removed from the list.
Alternative flows	1. If the user is in basic list view, the user will select city to reveal the delete button. 2. 3.

Use case name	Toggle Temperature Unit
Use case number	003
System	Java FX Component
Stakeholders/actors	1. User 2. 3.
Use case goal	Switch between Fahrenheit and Celsius
Primary actor	User
Preconditions	1. A tab component to toggle temperature must be present 2. A default temperature preference must be pre-selected 3. A button component to switch temperature unit must be present
Basic flow	User hovers cursor and selects preferences tab then with selects a button component to toggle temperature unit
Alternative flows	1. If user prefers Celsius instead of Fahrenheit, the user enters the settings tab to switch unit with a button 2. 3.

JavaFX UI Components:

1. Label -- https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/Label.html
2. Text Field -- https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/TextField.html
3. Check Box -- https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/CheckBox.html

4. Toggle Button -- https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/ToggleButton.html
5. Table -- https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/TableView.html
6. Accordion -- https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/Accordion.html