Name: Dev Arora

Register Number: 21BDS0130

Winter Semester 2023-24

Software Engineering Lab - BCSE301P

Phase – III

Report Requirements

Name: Dev Arora

Registration Number: 21BDS0130

Slot: L15+L16

- Consolidated report (SRS and SDS)

Project Name: Weather Application

Rev. No.	Revision Details	Date	Prepared By	Reviewed By Functional	Reviewed By Peer	Approved By
V01	Frist Release	14-1-24	Dev Arora			

Name: Dev Arora

Register Number: 21BDS0130

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to outline the functional and non-functional requirements for the development of a Weather Application. The application aims to provide users with accurate and up-to-date weather information, catering to a wide range of users for various purposes.

1.2 Scope

The Weather Application will offer features such as real-time weather updates, forecasts, radar images, and user customization options. It will be accessible on multiple platforms, including web browsers and mobile devices, ensuring a seamless experience for users.

1.3 Definitions, Acronyms, and Abbreviations

- **SRS**: Software Requirements Specification
- **API**: Application Programming Interface
- **GPS**: Global Positioning System
- **Alerts**: Notifications for severe weather conditions

1.4 References

- Business Requirements Document (BRD)
- User Stories

2. Overall Description

2.1 Product Perspective

The Weather Application will operate as a standalone system, interacting with external APIs to gather weather data. It will provide a user-friendly interface to access and visualize weather information.

2.2 Product Features

2.2.1 Core Features

1. **Real-time Weather Updates:**
    - Display current temperature, humidity, wind speed, and other relevant data.
    - Fetch and update data at regular intervals from a reliable weather data source.
2. **Weather Forecasts:**

Name: Dev Arora

Register Number: 21BDS0130

- Provide short-term and long-term weather forecasts.
- Allow users to view forecasts for specific dates and locations.

3. **Geolocation:**
    - Automatically detect the user's location for personalized weather information.
    - Allow users to search for weather information based on a specific location.
4. **Radar and Satellite Imagery:**
    - Integrate radar and satellite images for visualizing current weather conditions.
    - Allow users to zoom in and out for a more detailed view.

2.2.2 User Experience Features

1. **User Authentication:**
    - Allow users to create accounts and log in to personalize their experience.
    - Store user preferences for locations and units.
2. **Customizable Dashboard:**
    - Enable users to customize the dashboard with their preferred weather widgets.
    - Support different themes for user interface personalization.
3. **Weather Alerts:**
    - Provide customizable weather alerts for specific locations and conditions.
    - Send push notifications or emails for critical weather updates.

2.3 User Classes and Characteristics

1. **General Users:**
    - Seek current weather updates and forecasts.
    - May not require advanced features but appreciate a user-friendly interface.
2. **Weather Enthusiasts:**
    - Interested in detailed radar and satellite imagery.
    - Appreciate customizable dashboards and alerts.

2.4 Operating Environment and Constraints:

The application will be available on Android and iOS platforms. Constraints include the requirement for a reliable internet connection to fetch real-time weather data.

2.5 Major System Components and Interfaces:

- **GPS Module**: Interfaces with the device's location services.
- **Weather Data API**: Fetches real-time and forecast data.

Name: Dev Arora

Register Number: 21BDS0130

3. System Features

3.1 Real-time Weather Updates

3.1.1 Description

The application shall display real-time weather information, including temperature, humidity, wind speed, and other relevant data.

3.1.2 Functional Requirements

1. The application shall fetch weather data from a reliable API.
2. Weather updates shall be displayed in a visually appealing and easy-to-read format.
3. Updates shall occur at regular intervals, providing accurate and up-to-date information.

3.1.3 Non-functional Requirements

1. The application shall have a response time of less than 3 seconds for fetching and displaying real-time updates.

3.2 Weather Forecasts

3.2.1 Description

The application shall provide short-term and long-term weather forecasts for users' selected locations.

3.2.2 Functional Requirements

1. Short-term forecasts (next 24 hours) shall include hourly breakdowns.
2. Long-term forecasts (next 7 days) shall provide a daily overview.
3. Users can select specific dates for extended forecasts.

3.2.3 Non-functional Requirements

1. Forecast data shall be updated at least once every 6 hours.

3.3 Geolocation

3.3.1 Description

The application shall support automatic detection of the user's location for personalized weather information.

3.3.2 Functional Requirements

1. The application shall request permission to access the device's location.
2. Users shall have the option to manually enter a location for weather information.

3.3.3 Non-functional Requirements

1. Geolocation accuracy shall be within 1 kilometer.

3.4 Radar and Satellite Imagery

Name: Dev Arora

Register Number: 21BDS0130

3.4.1 Description

The application shall integrate radar and satellite images to visualize current weather conditions.

3.4.2 Functional Requirements

1. Users shall be able to toggle between radar and satellite views.
2. The imagery shall support zooming and panning for detailed exploration.

3.4.3 Non-functional Requirements

1. Image loading time shall be optimized for smooth user experience.

4. External Interface Requirements

4.1 User Interfaces

1. **Homepage:**
    - Display current weather conditions.
    - Provide quick access to forecasts, radar, and user settings.
2. **Weather Details Page:**
    - Present detailed weather information.
    - Include interactive graphs for temperature, wind speed, etc.
3. **Settings Page:**
    - Allow users to customize units, theme, and notification preferences.

4.2 Hardware Interfaces

The application shall be compatible with devices running on iOS, Android, and major web browsers.

4.3 Software Interfaces

1. The application shall integrate with a reliable weather data API.
2. Push notification services (e.g., Firebase Cloud Messaging) for weather alerts.

5. Other Non-functional Requirements

5.1 Performance Requirements

1. The application shall load within 5 seconds on 3G and 4G networks.
2. Response time for user interactions shall be less than 1 second.

5.2 Security Requirements

1. User data, including location information, shall be encrypted during transmission.
2. Secure user authentication mechanisms shall be implemented.

Name: Dev Arora

Register Number: 21BDS0130

5.3 Reliability and Availability

1. The application shall be available 99.9% of the time.
2. Data backup mechanisms shall be in place to prevent data loss.

5.4 Scalability

The application architecture shall support scalability to handle increased user demand.

6. Appendices:

6.1 Glossary:

- **Location Services**: Technology that determines a device's geographical position.
- **Severe Weather**: Extreme weather conditions requiring user attention.

6.2 Analysis Models:

- Use Cases: Illustrate scenarios like accessing current weather and receiving alerts.

6.3 Prototypes:

- Include wireframes demonstrating the app's user interface.

7. Conclusion

This Software Requirements Specification outlines the essential features, user classes, and system requirements for the development of a comprehensive Weather Application. It serves as a foundation for the design and implementation phases, ensuring the delivery of a robust and user-friendly application that meets the needs of a diverse user base.

8. Additional Considerations:

- **Clarity and Conciseness**: Use plain language, avoid jargon, and structure content logically.
- **Completeness**: Ensure all functionalities and constraints are addressed.
- **Verifiability**: Specify how each requirement will be tested.
- **Traceability**: Link each requirement to relevant design elements and test cases.
- **Maintainability**: Regularly update the SRS to reflect evolving requirements.

9. Visual Representations:

- **Diagrams**: Utilize flowcharts for user interactions and data flow.
- **Wireframes**: Include visuals of key screens and interactions.
- **Decision Tables**: Represent complex logic for weather alerts.

Process Model Selection for Weather Application: Incremental Model Justification:

1. **Changing Requirements**:
   Weather applications often face evolving data sources and user expectations. The Incremental model accommodates changes well, with each increment capable of addressing new requirements or modifications.
2. **Quick Releases**:

Name: Dev Arora

Register Number: 21BDS0130

Incremental development allows for the release of functional subsets rapidly. This is vital for a Weather Application where timely updates, new features, or bug fixes are crucial to providing accurate and up-to-date information to users.

3. **User Feedback**:
   Frequent releases enable gathering user feedback early in the development process. This feedback is crucial for refining features, improving user experience, and ensuring the application aligns with user expectations.
4. **Risk Management**:
   Breaking the project into increments allows for better risk management. If an issue arises in one increment, it can be addressed without affecting the entire application, reducing the overall project risk.
5. **Parallel Development**:
   Different increments can be developed simultaneously by different teams or individuals, facilitating parallel development and potentially speeding up the overall project timeline.
6. **Quality Assurance**:
   Incremental development allows for continuous testing throughout the project. This ensures that each increment meets quality standards, reducing the likelihood of major defects in the final product.
7. **Client Involvement**:
   Incremental development encourages client or stakeholder involvement at various stages. This ensures that the end product aligns closely with client expectations, reducing the chances of misunderstandings.

Why Other Models Are Not Suitable:

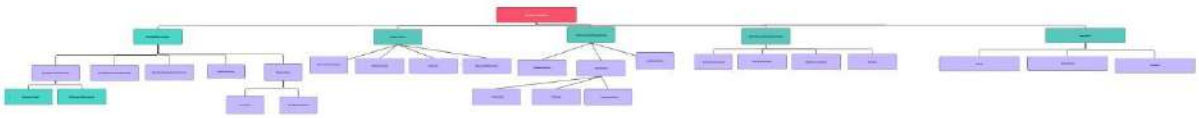
1. **Waterfall Model**:
   Inflexible and sequential, the Waterfall model doesn't adapt well to changing requirements. Weather applications, with dynamic data and evolving user needs, benefit more from an iterative approach.
2. **V-Model**:
   Similar to Waterfall, the V-Model follows a linear path, making it less suitable for the iterative and flexible nature of a Weather Application's development.
3. **Spiral Model**:
   While the Spiral model incorporates risk management, it may not be as efficient for rapid and incremental releases as required in a Weather Application.
4. **Agile Model**:
   Although Agile shares some similarities with Incremental, its focus on delivering a Minimum Viable Product (MVP) might not align perfectly with the continuous enhancements often needed in a Weather Application.

In conclusion, the Incremental model is preferred for a Weather Application due to its adaptability to changing requirements, ability to facilitate quick releases, efficient risk management, and provision for continuous user feedback and improvement. Other models are less suitable due to their rigid structures or different emphases, not aligning as well with the dynamic nature of weather-related software development.

Work Break Down Function

Name: Dev Arora

Register Number: 21BDS0130



https://www.canva.com/design/DAF8CEYeN8o/Okt87pWR-WcpaNYdunUg5A/view?utm_content=DAF8CEYeN8o&utm_campaign=celebratory_first_publish&utm_medium=link&utm_source=editor_celebratory_first_publish

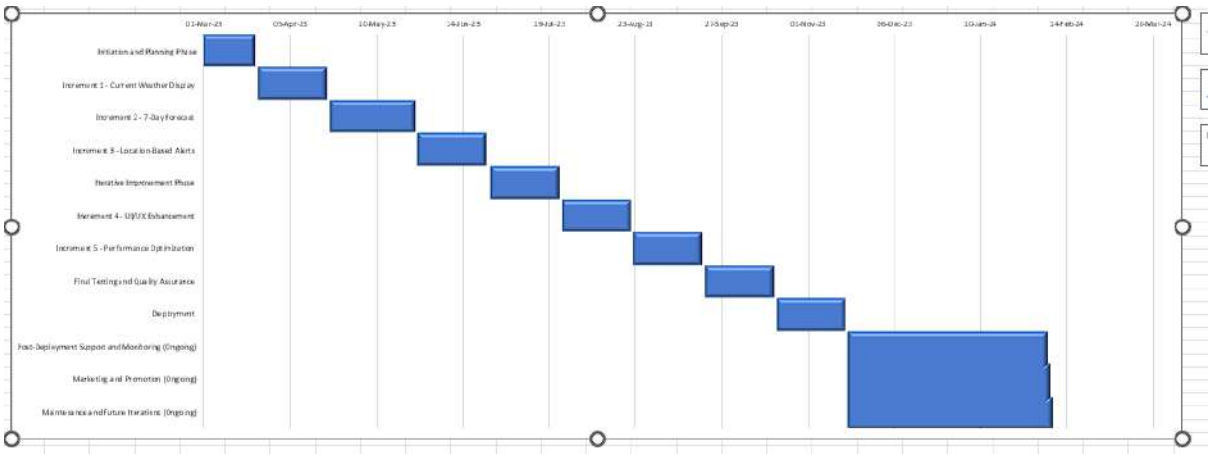
Project scheduling – Timeline charts

Task	Start Date	End Date	Duration
Initiation and Planning Phase	01-Mar-23	22-Mar-23	21
Increment 1 - Current Weather Display	23-Mar	20-Apr-23	28
Increment 2 - 7-Day Forecast	21-Apr	26-May-23	35
Increment 3 - Location-Based Alerts	27-May	24-Jun-23	28
Iterative Improvement Phase	25-Jun	23-Jul-23	28

Name: Dev Arora

Register Number: 21BDS0130

Increment 4 - UI/UX Enhancement	24-Jul	21-Aug-23	28
Increment 5 - Performance Optimization	22-Aug	19-Sep-23	28
Final Testing and Quality Assurance	20-Sep	18-Oct-23	28
Deployment	19-Oct	16-Nov-23	28
Post-Deployment Support and Monitoring (Ongoing)	17-Nov	6-Feb-24	81
Marketing and Promotion (Ongoing)	17-Nov	7-Feb-24	82
Maintenance and Future Iterations (Ongoing)	17-Nov	8-Feb-24	83



Winter Semester 2023-24

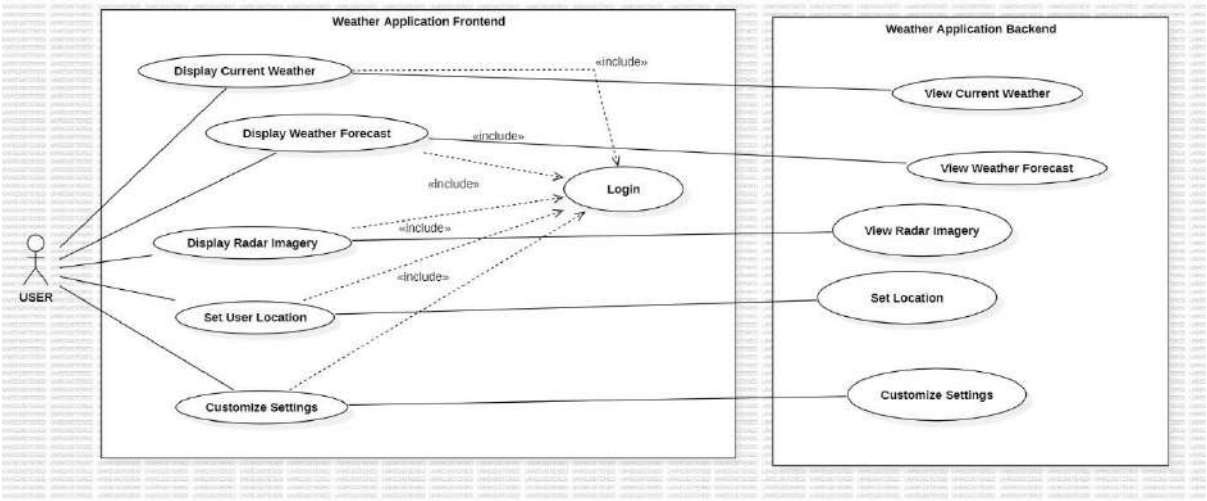
Software Engineering Lab

Phase - 2 (UML Diagrams)

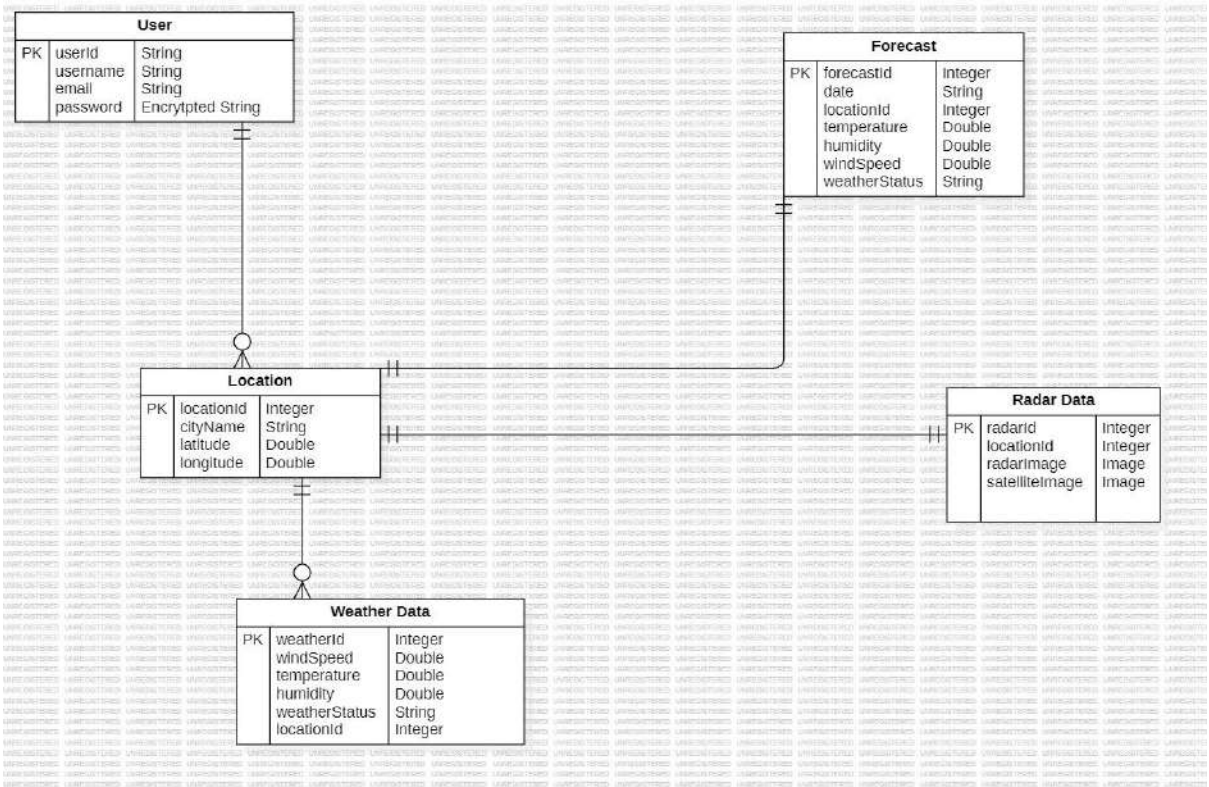
1. User Diagram

Name: Dev Arora

Register Number: 21BDS0130



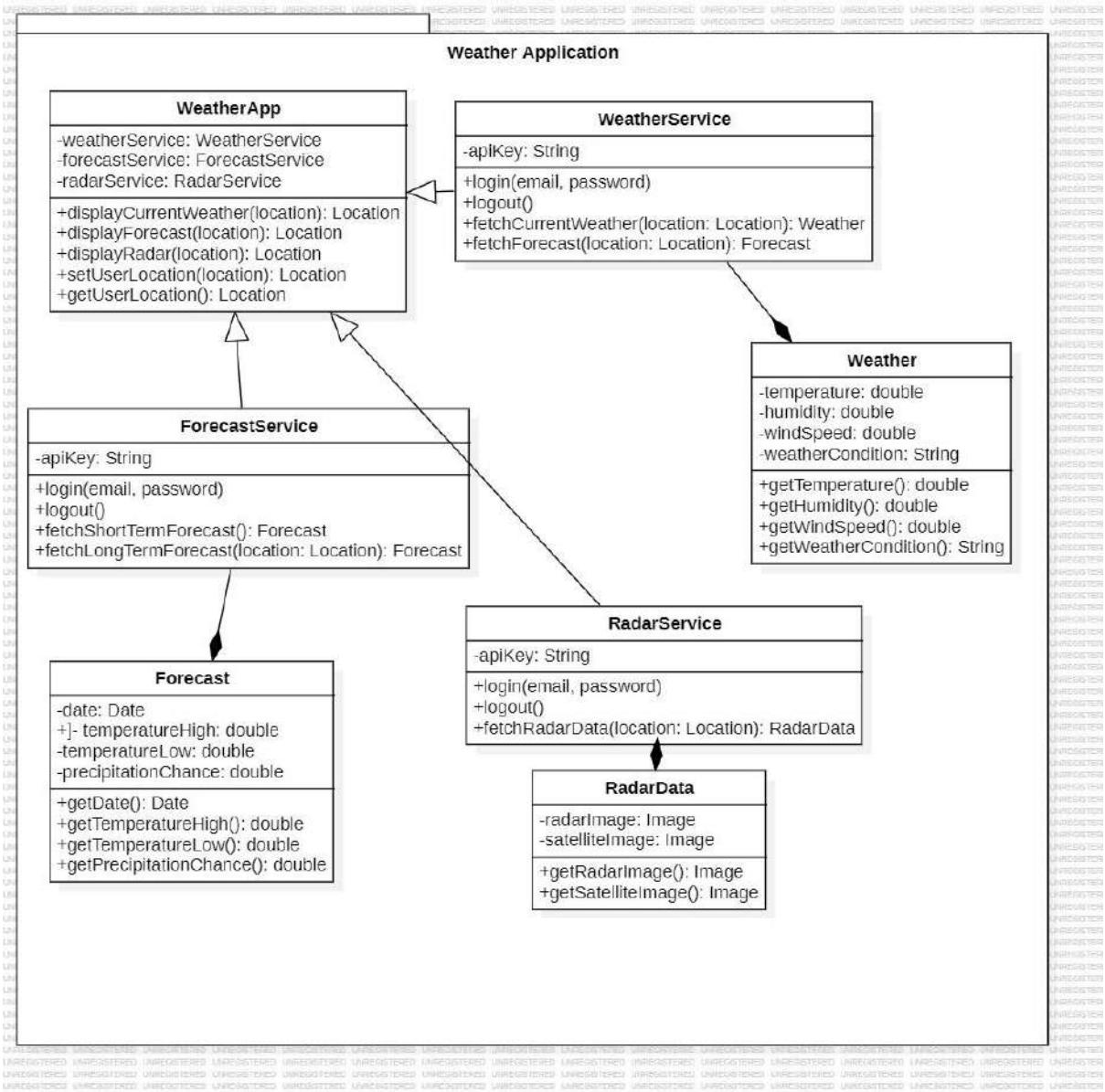
2. ER Diagram



3. Class Diagram

Name: Dev Arora

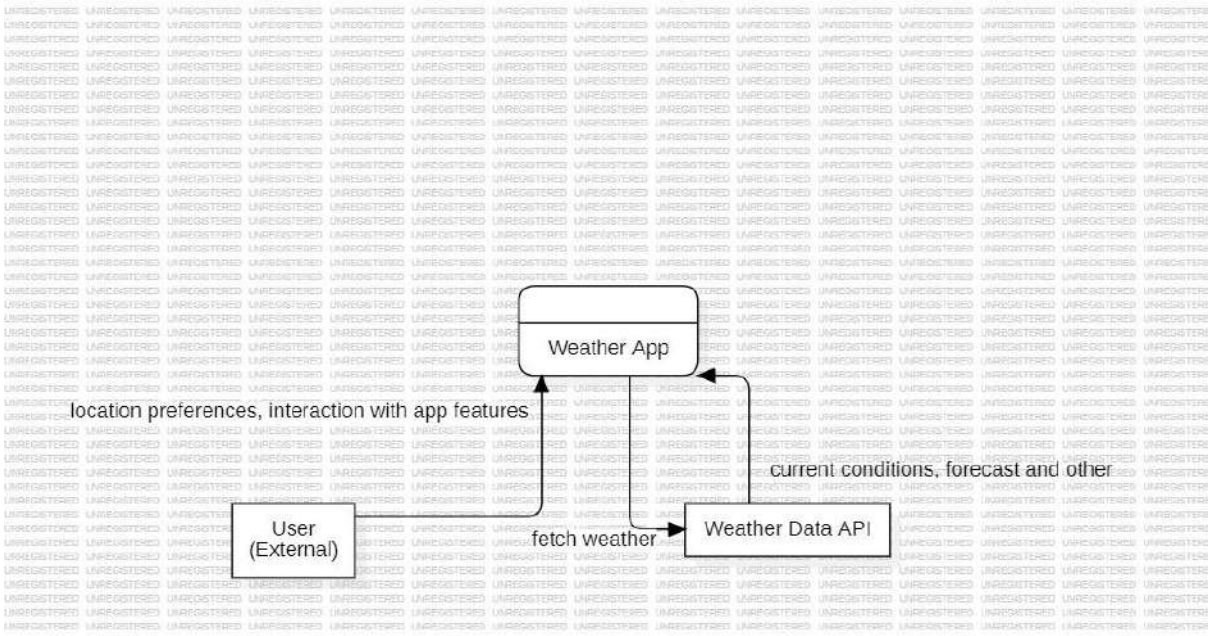
Register Number: 21BDS0130



Name: Dev Arora

Register Number: 21BDS0130

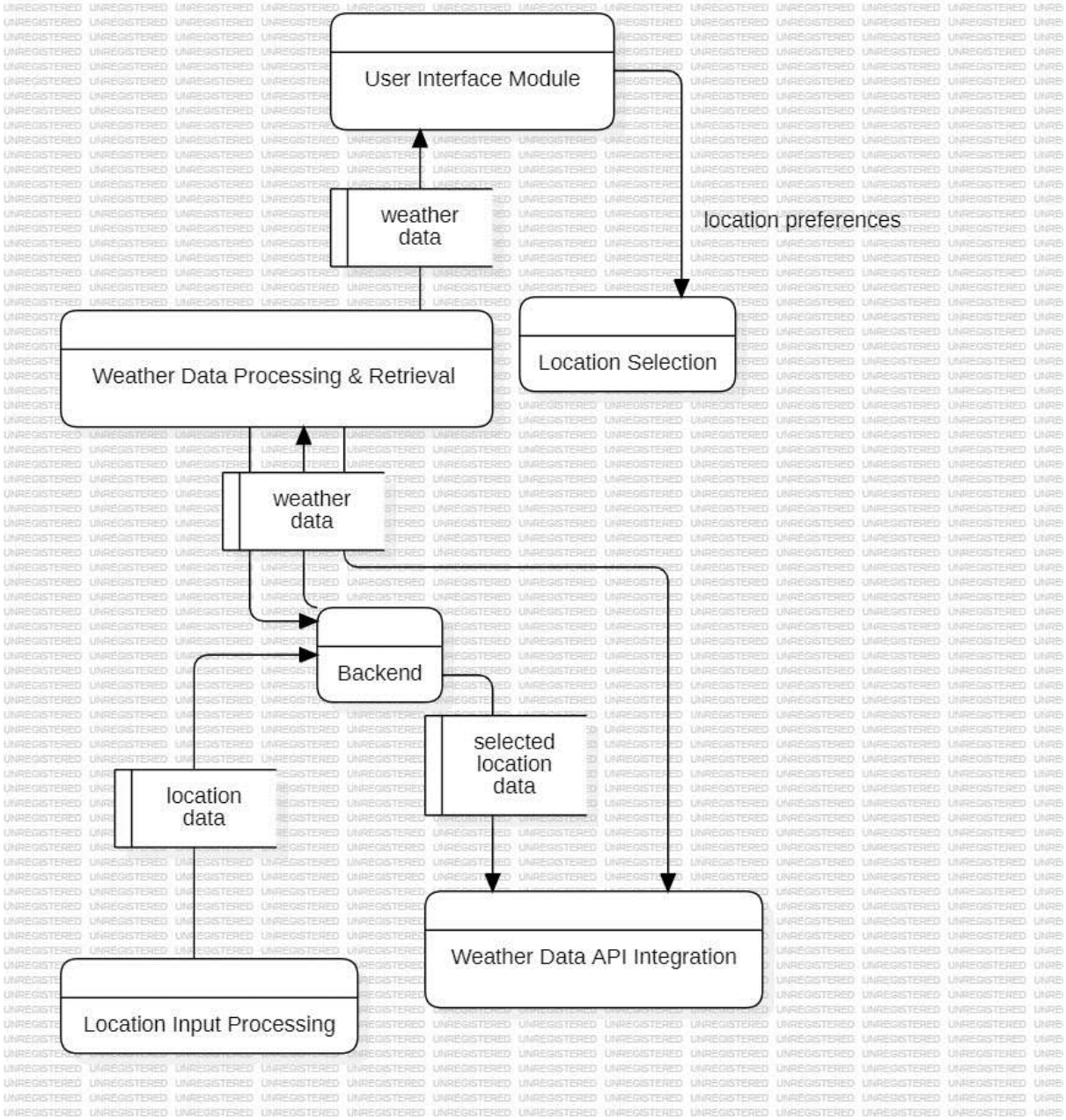
DFD Level 0



DFD Level 1

Name: Dev Arora

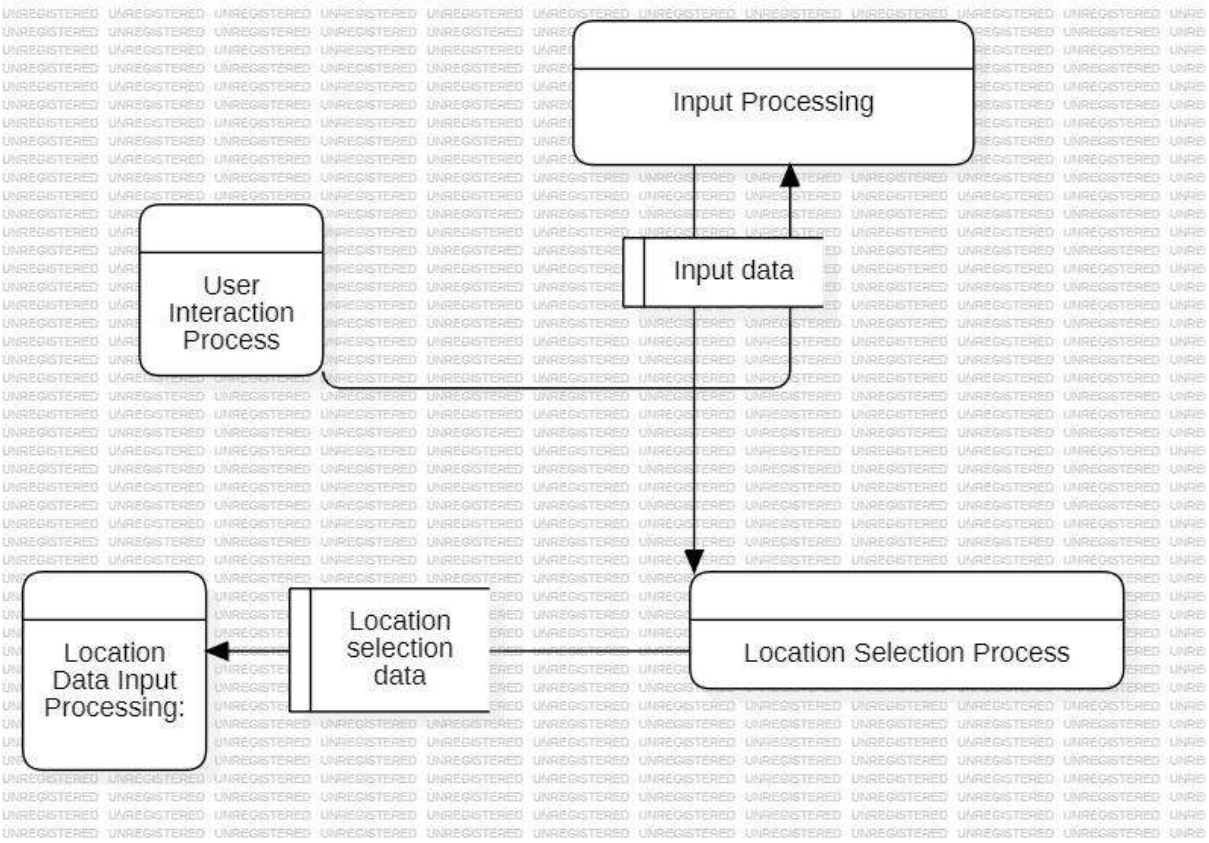
Register Number: 21BDS0130



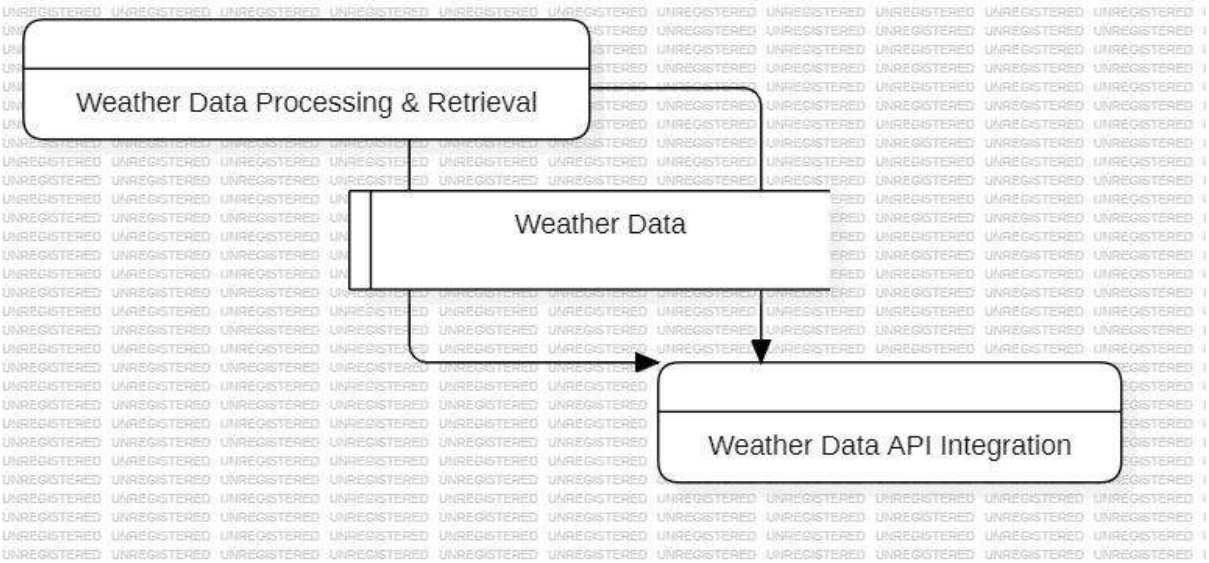
Name: Dev Arora

Register Number: 21BDS0130

Level 2 DFD for User Interface Module



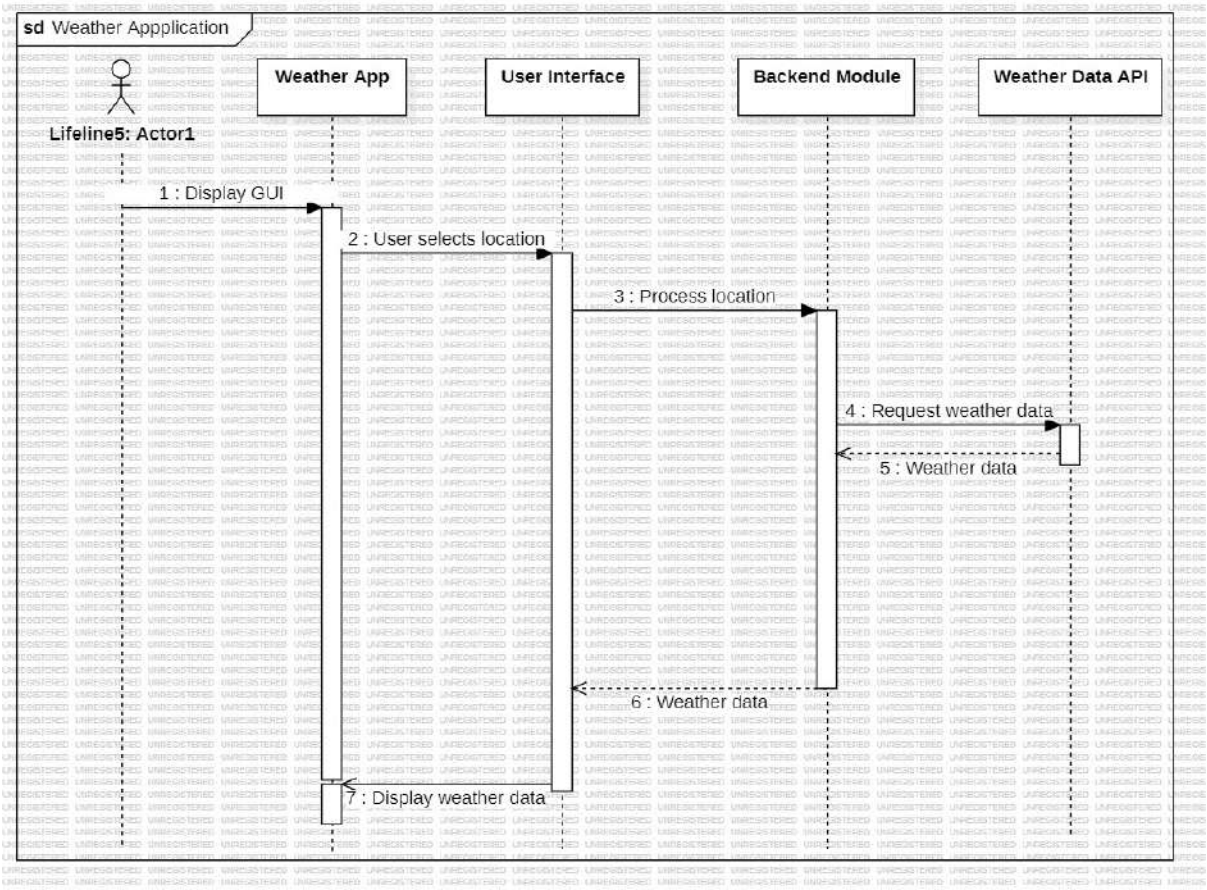
Level 2 DFD for Backend Module



Name: Dev Arora

Register Number: 21BDS0130

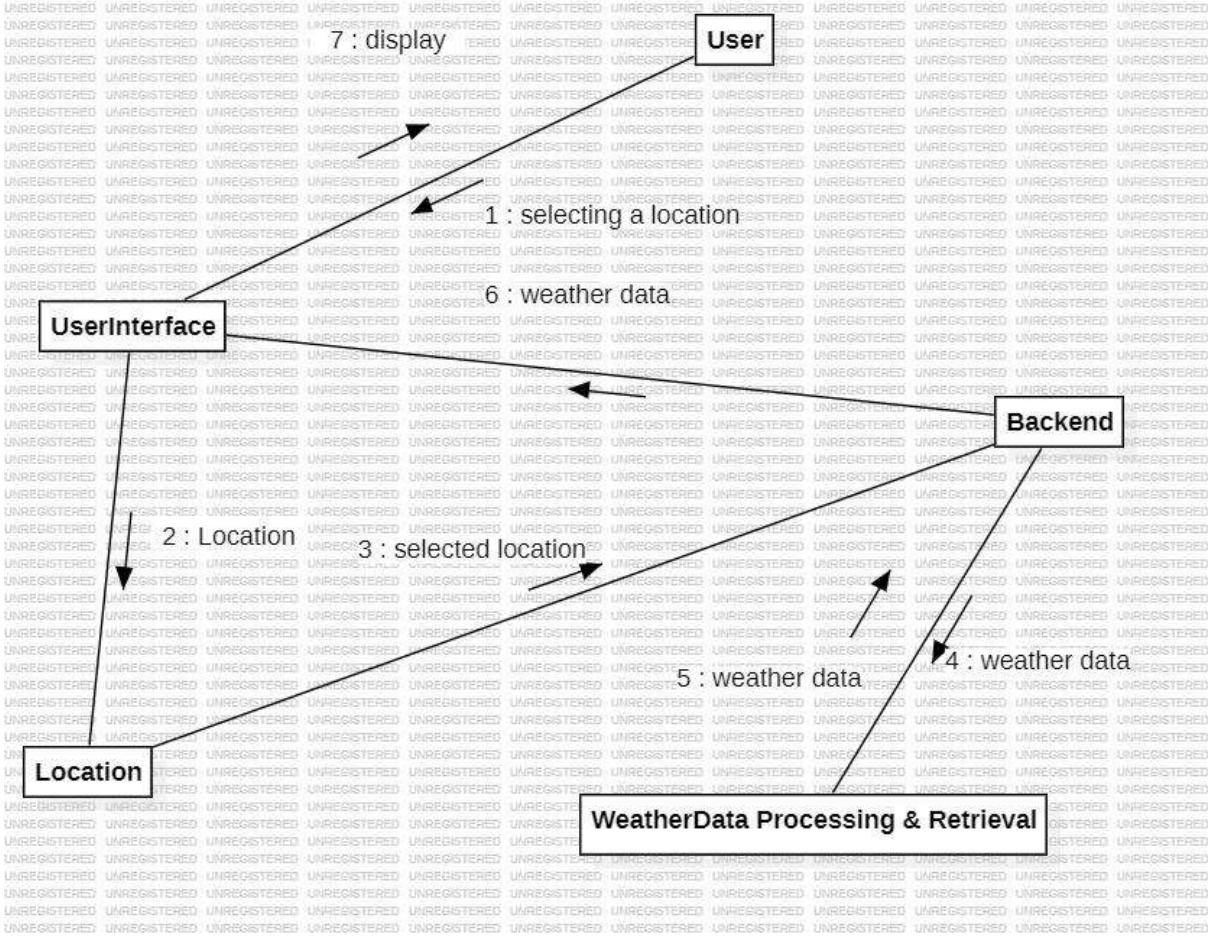
Sequence Diagram



Name: Dev Arora

Register Number: 21BDS0130

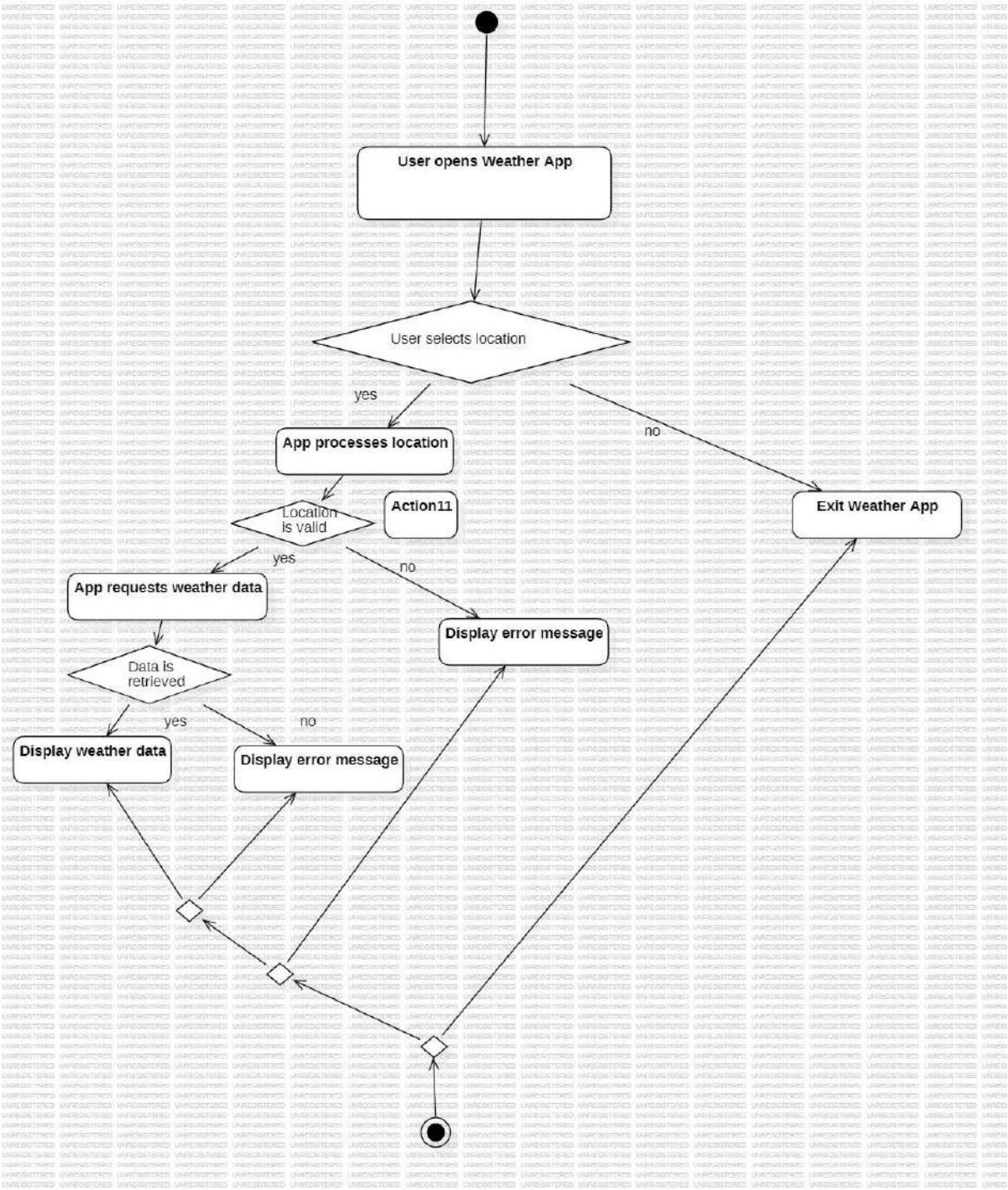
Collaboration diagram



Name: Dev Arora

Register Number: 21BDS0130

Activity Diagram



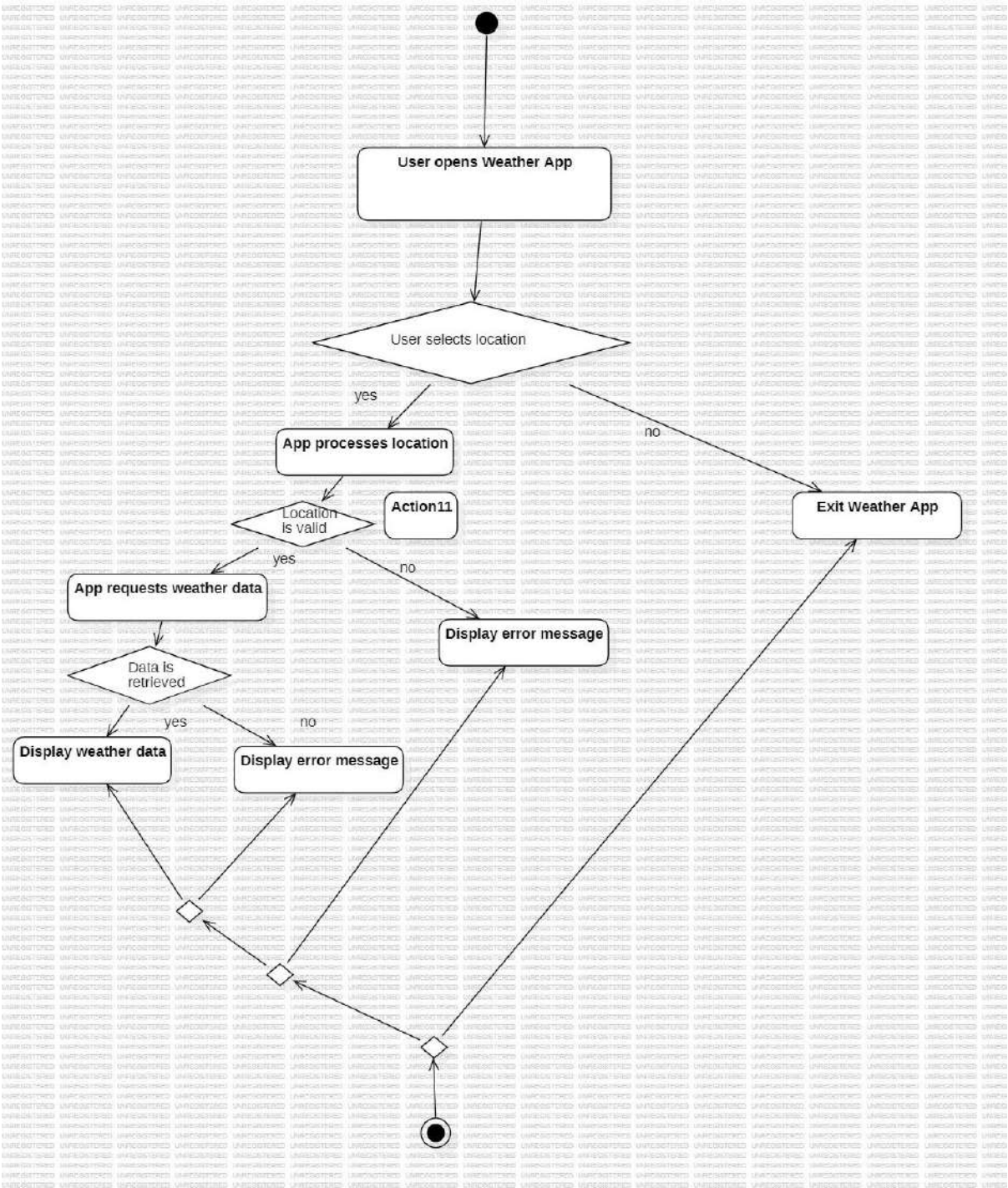
Name: Dev Arora

Register Number: 21BDS0130

State chart Diagram

Name: Dev Arora

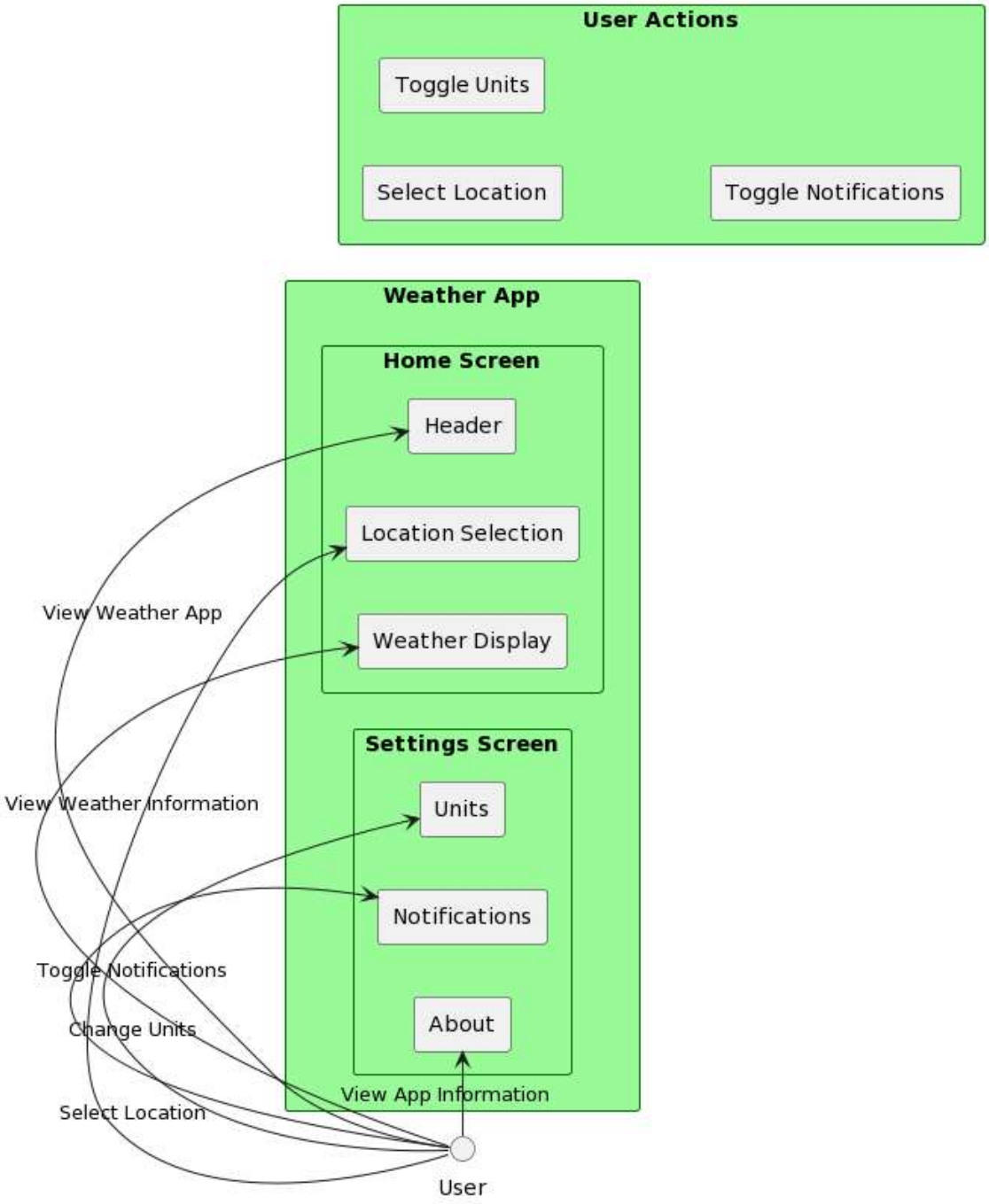
Register Number: 21BDS0130



Name: Dev Arora

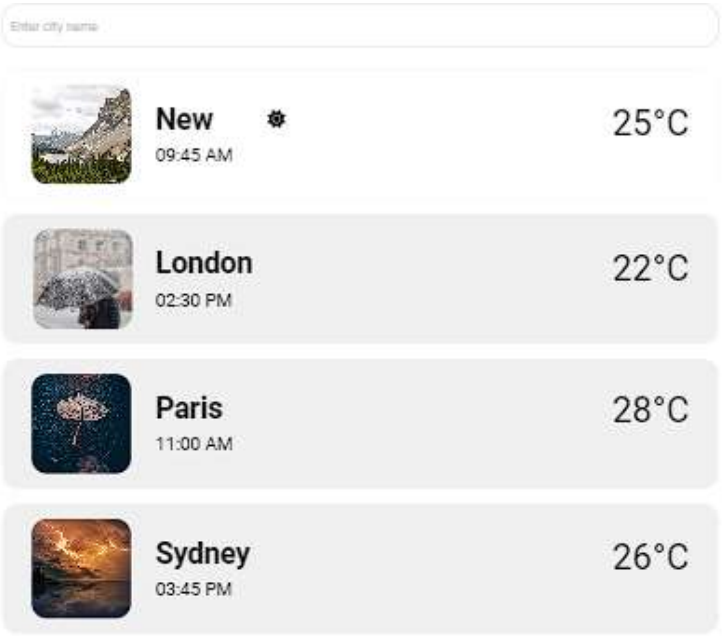
Register Number: 21BDS0130

User Interface diagram



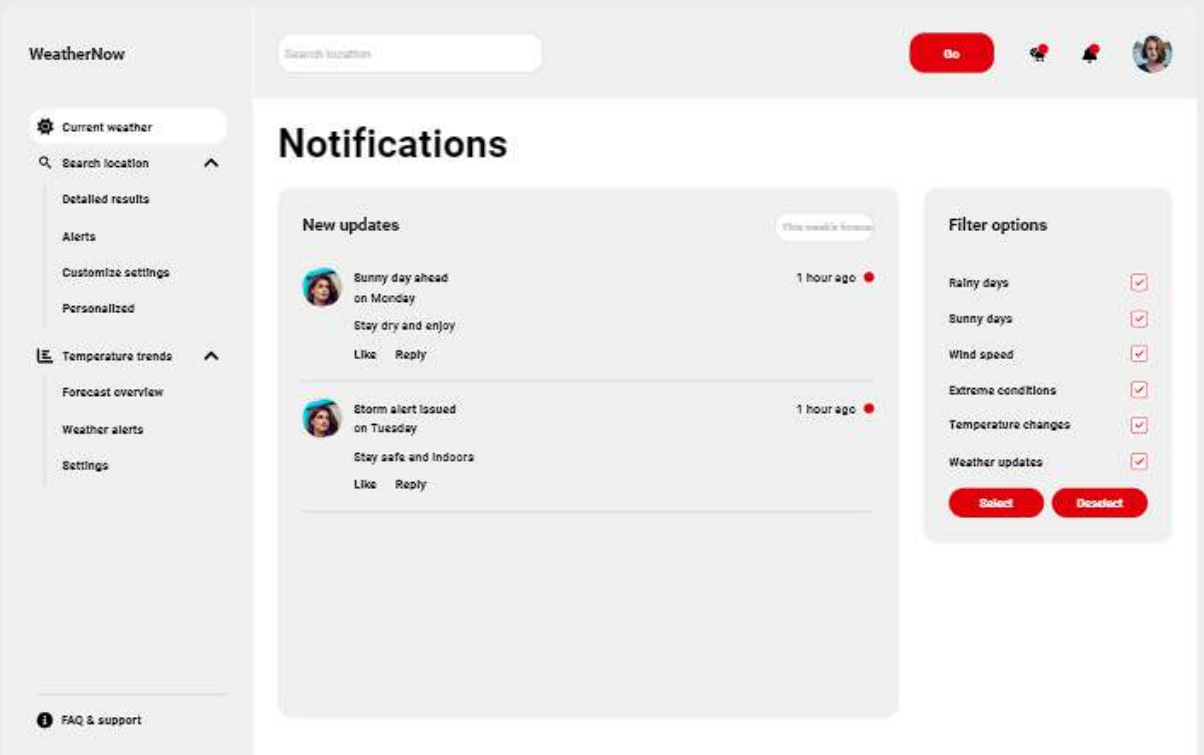
Name: Dev Arora

Register Number: 21BDS0130



Name: Dev Arora

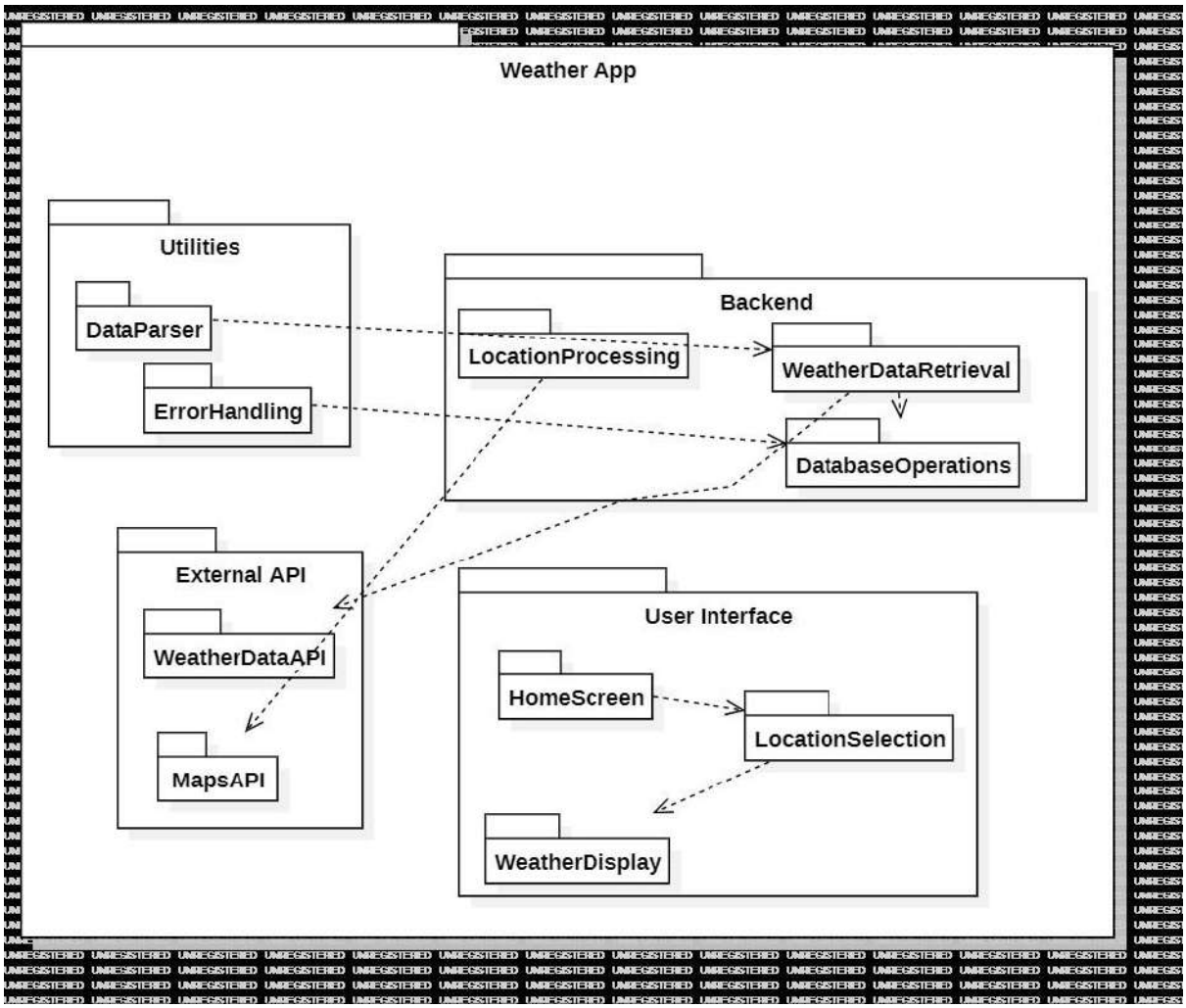
Register Number: 21BDS0130



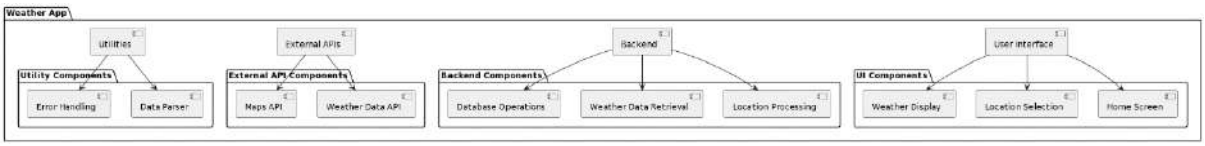
Package diagram

Name: Dev Arora

Register Number: 21BDS0130



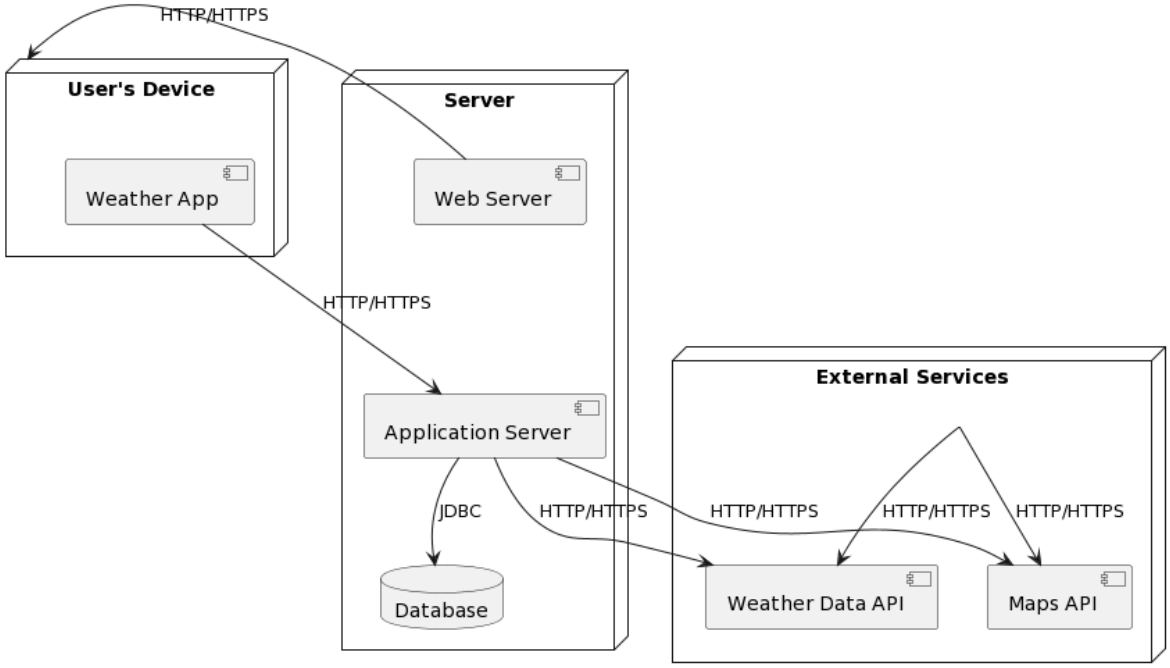
Component diagram



Name: Dev Arora

Register Number: 21BDS0130

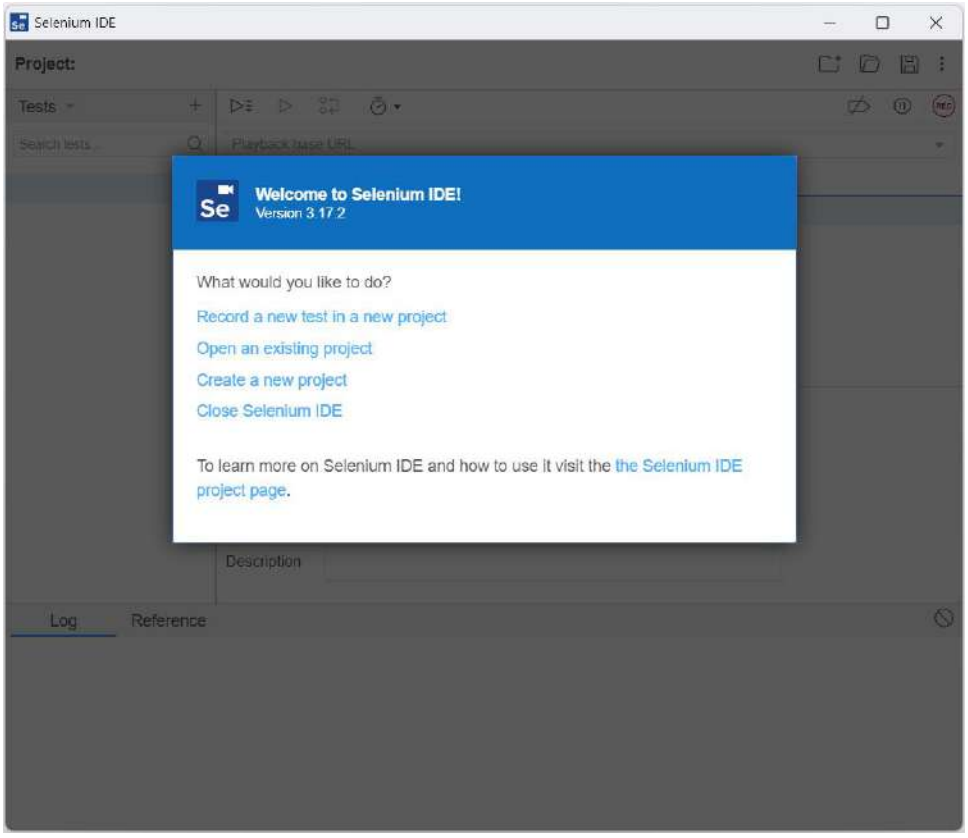
Deployment diagram



- Test cases report
- First, we'll start by adding the Selenium extension to our Chrome webpage, and then we'll click it to test our weather app. Select "Record a new test in a new project"

Name: Dev Arora

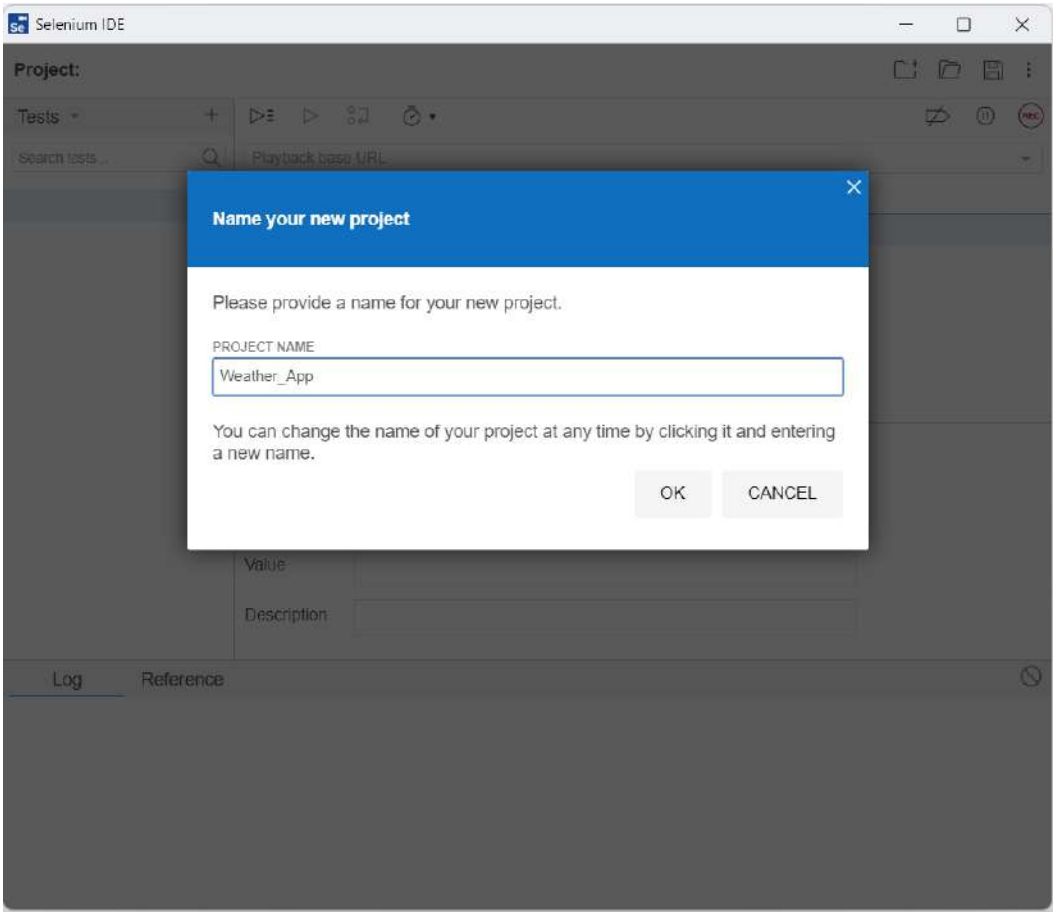
Register Number: 21BDS0130

- 

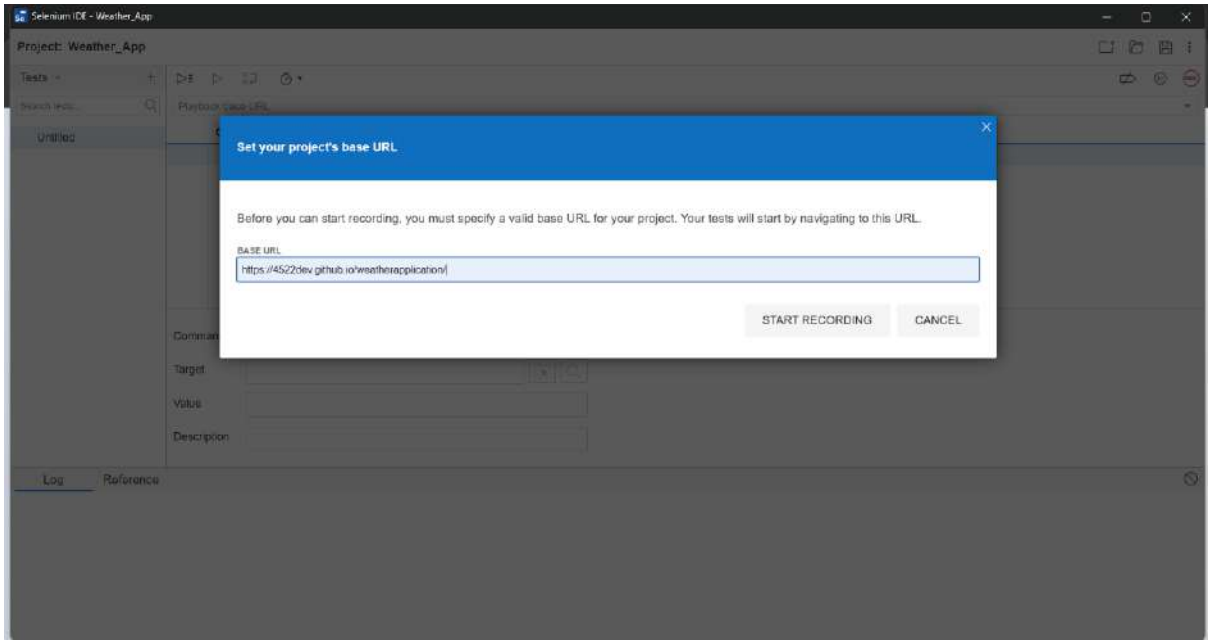
- We'll name our Project Weather_App

Name: Dev Arora

Register Number: 21BDS0130



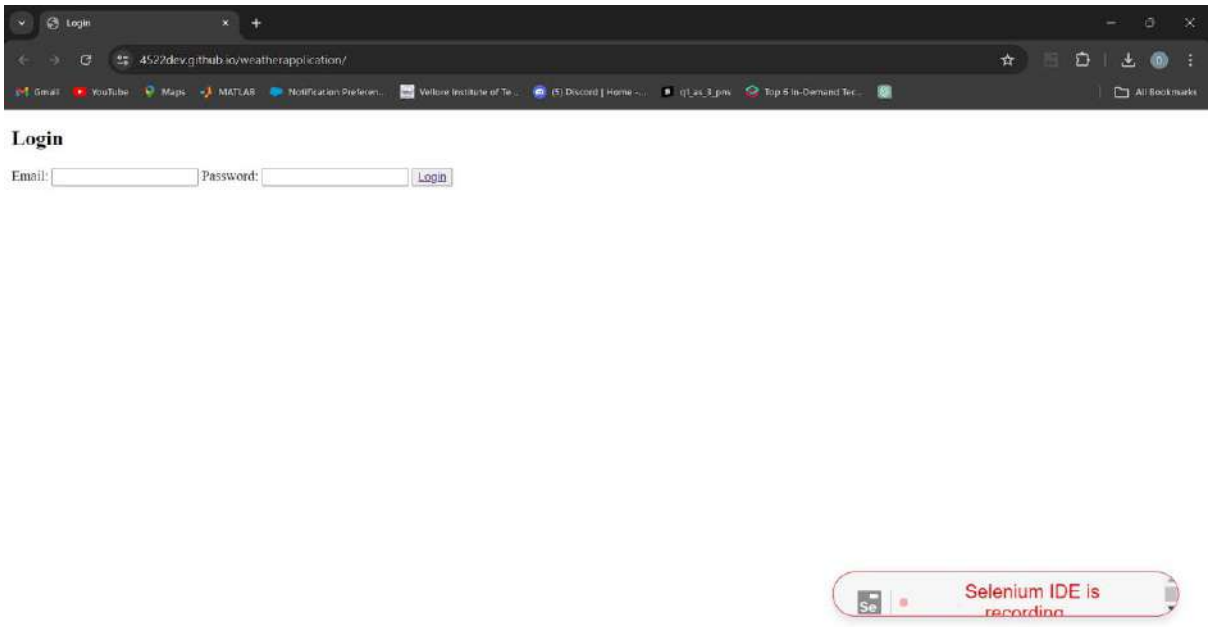
- We will select our react-frontend web application that we've made for our testing, and thus provide the URL of the hosting page as follows:



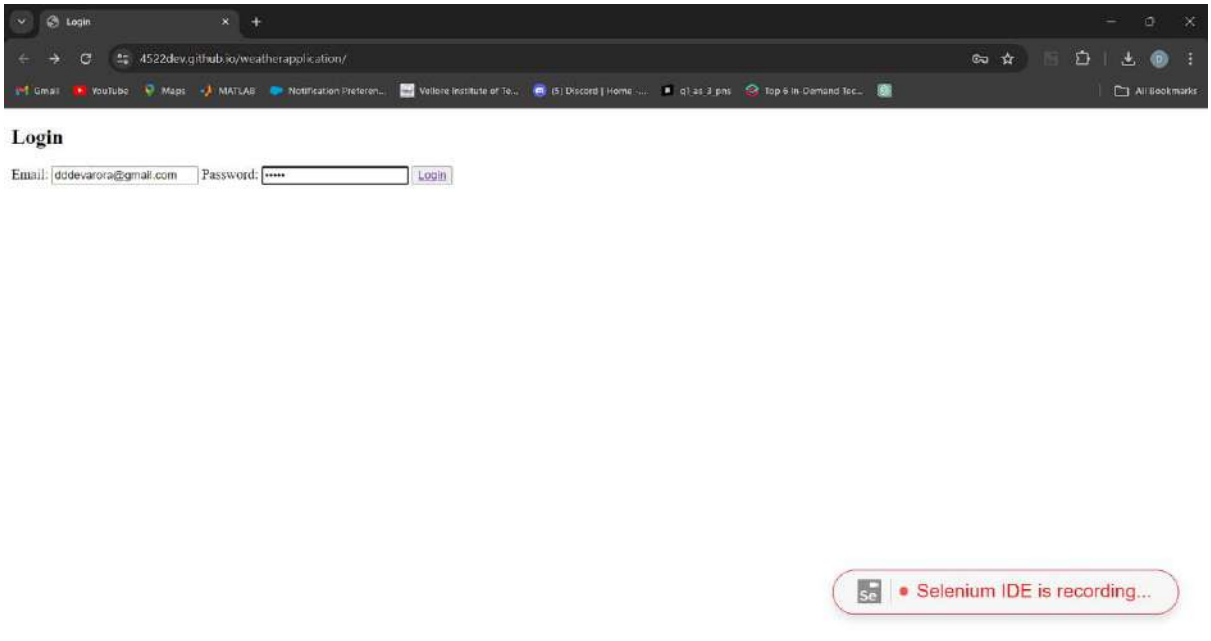
Name: Dev Arora

Register Number: 21BDS0130

After we enter the URL, and select Start Recording the Selenium IDE redirects to the selected webpage and starts recording



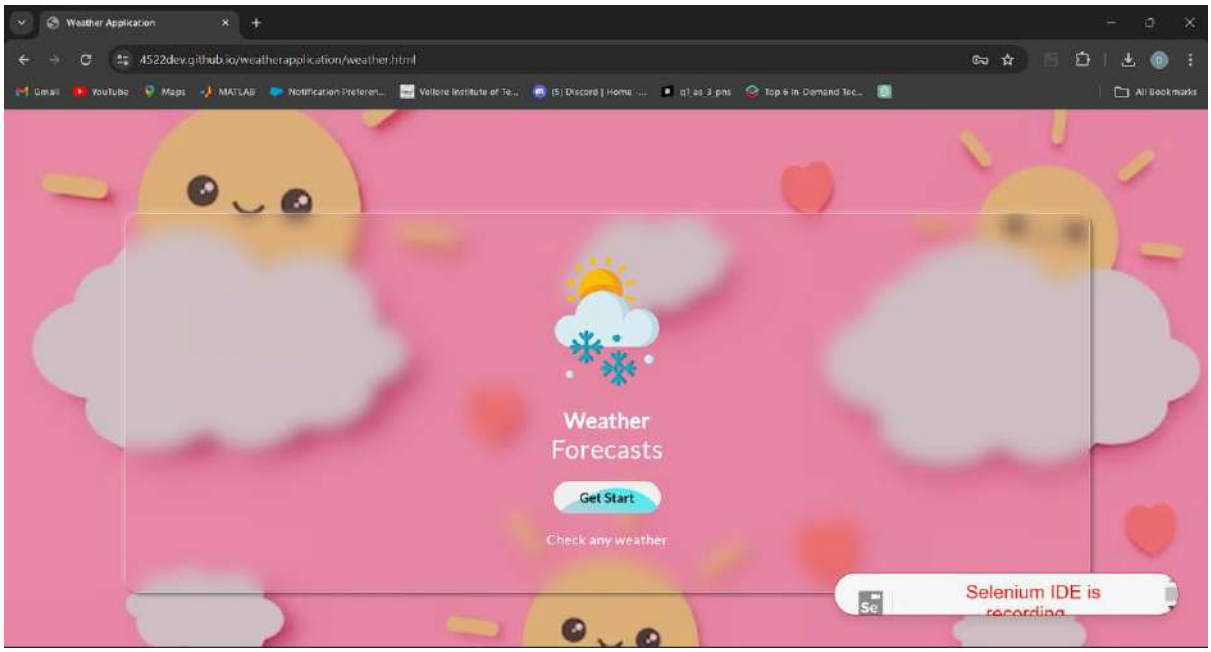
We'll enter the required details required to proceed to the next page



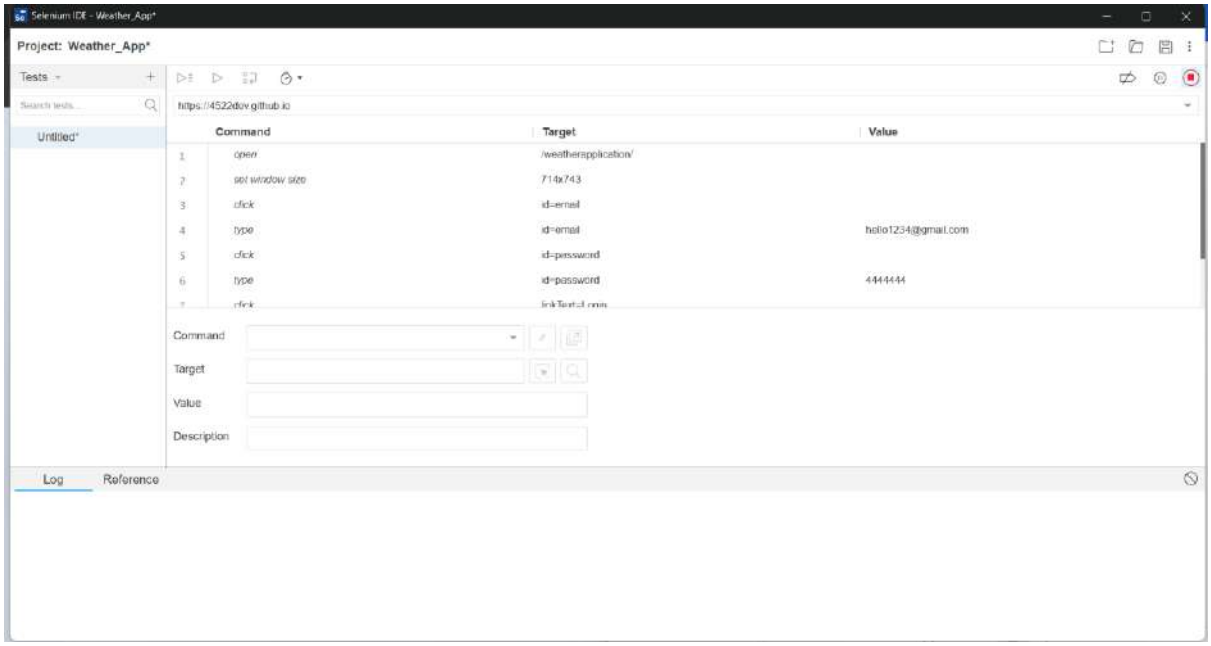
The UI interface is opened for the Weather_App and we still can see selenium records the testing below

Name: Dev Arora

Register Number: 21BDS0130



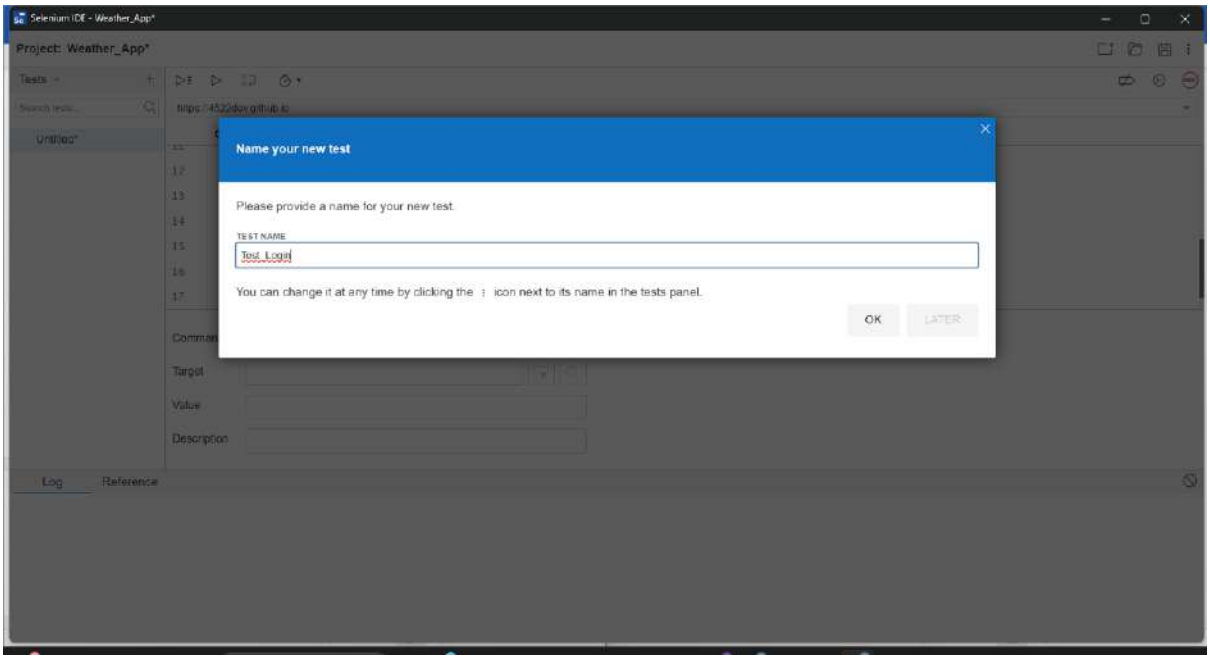
Now, we complete with the 1st phase of i.e Login Testing, we click stop recording



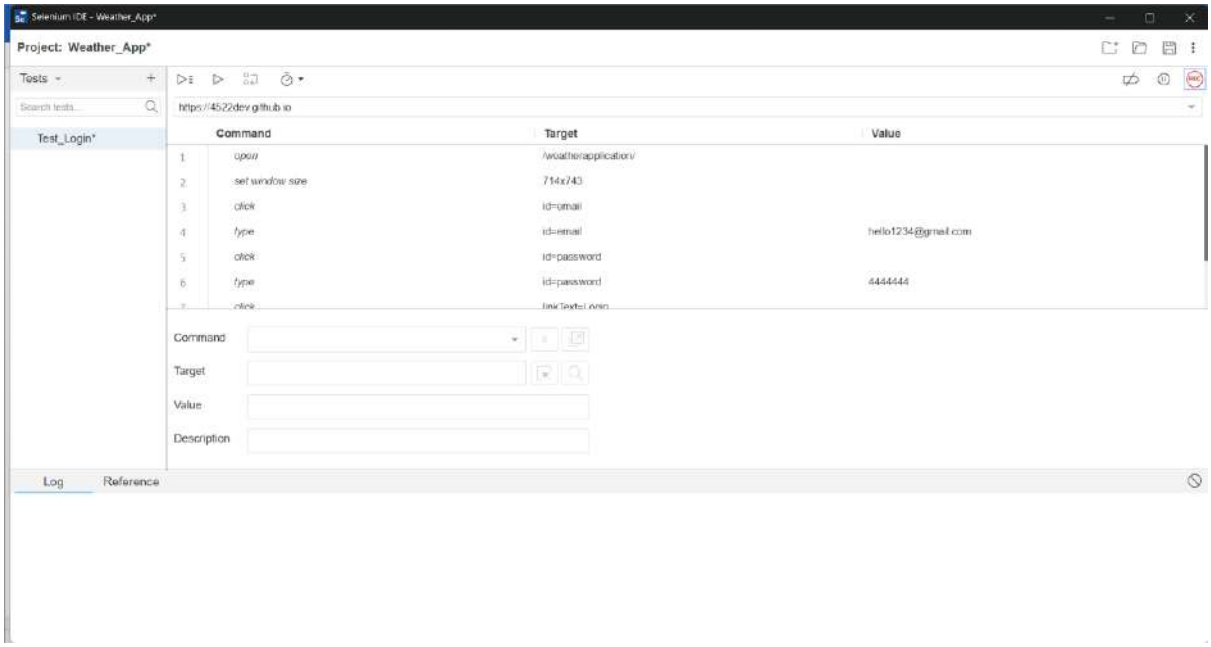
It'll ask us to provide the name of the test we've conducted

Name: Dev Arora

Register Number: 21BDS0130



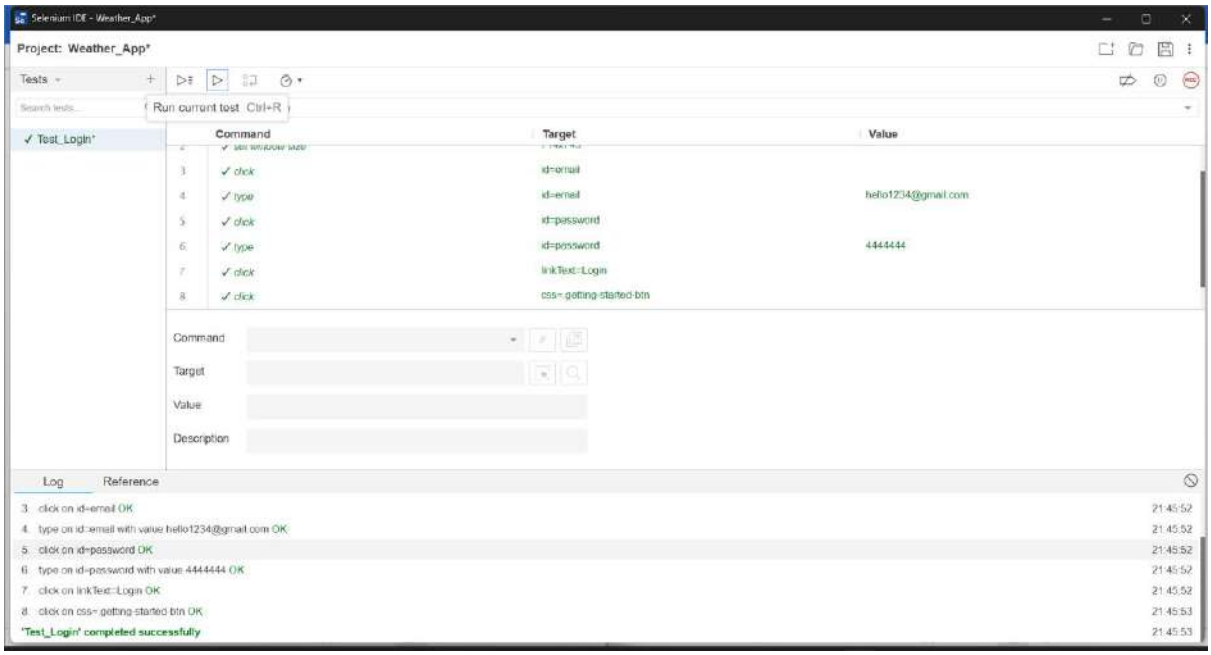
Now in order to check whether or not we've successfully tested the app, we'll click on run current test



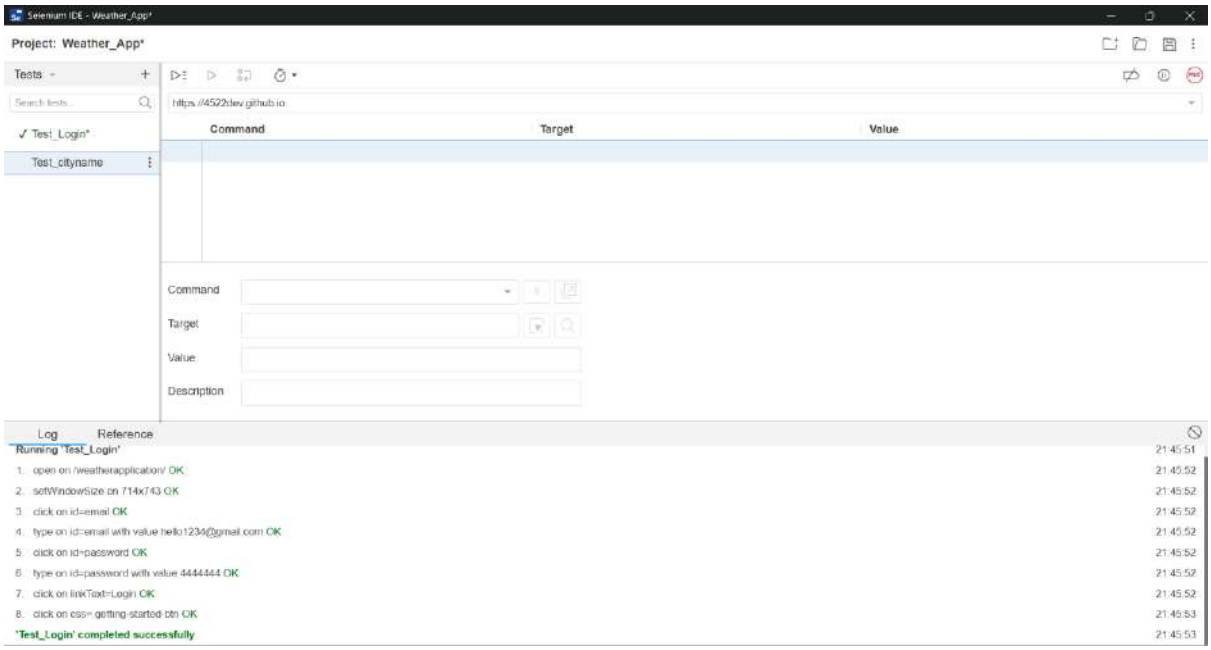
It'll retrace our steps by recording the clicks with respect to the provided URL

Name: Dev Arora

Register Number: 21BDS0130



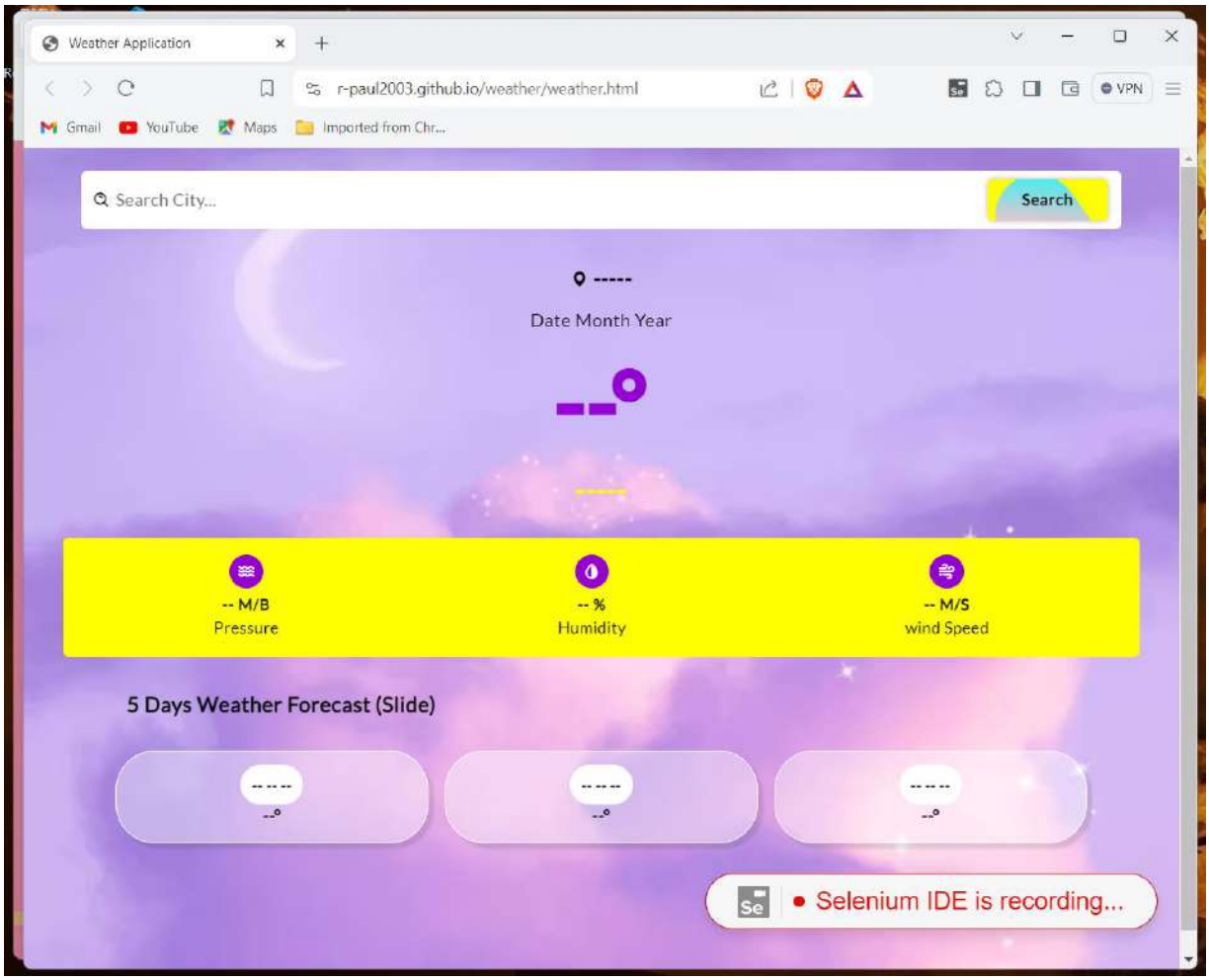
Thus, we see that our Login_Test case passed successfully, now we add another test case



Test Case_2: Checks the application part of our program

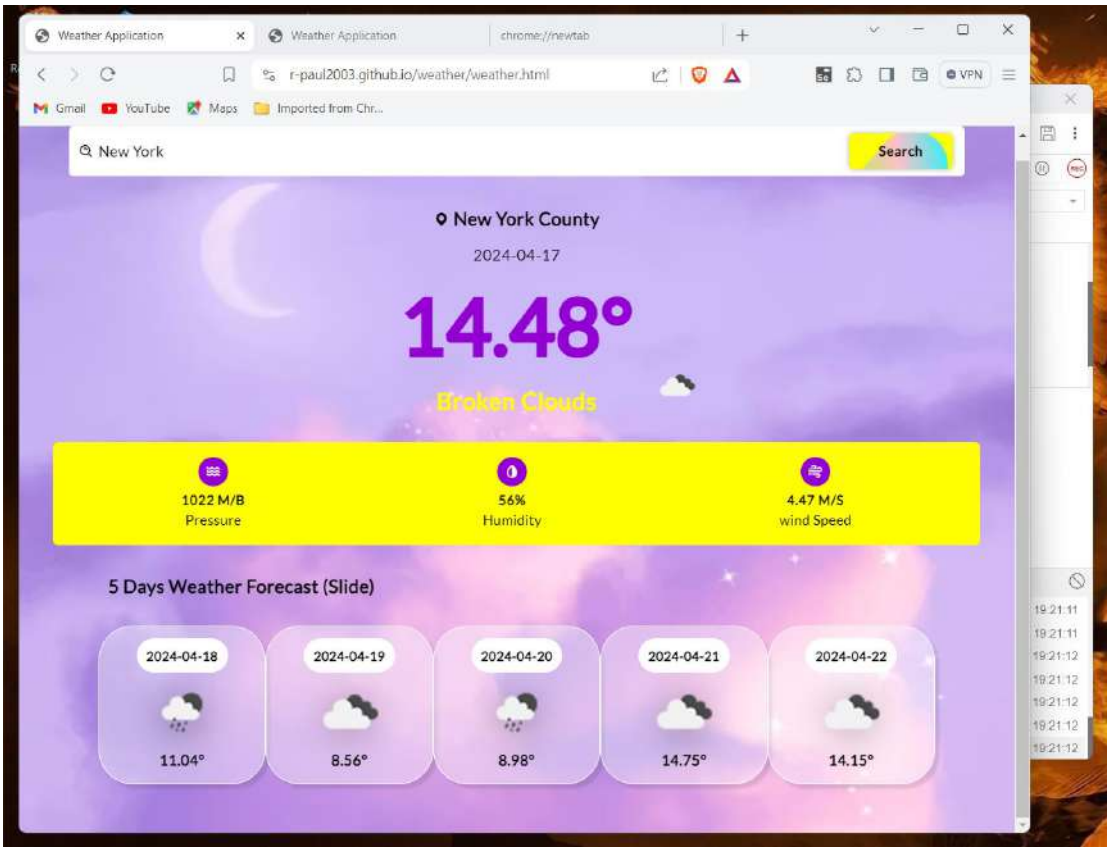
Name: Dev Arora

Register Number: 21BDS0130

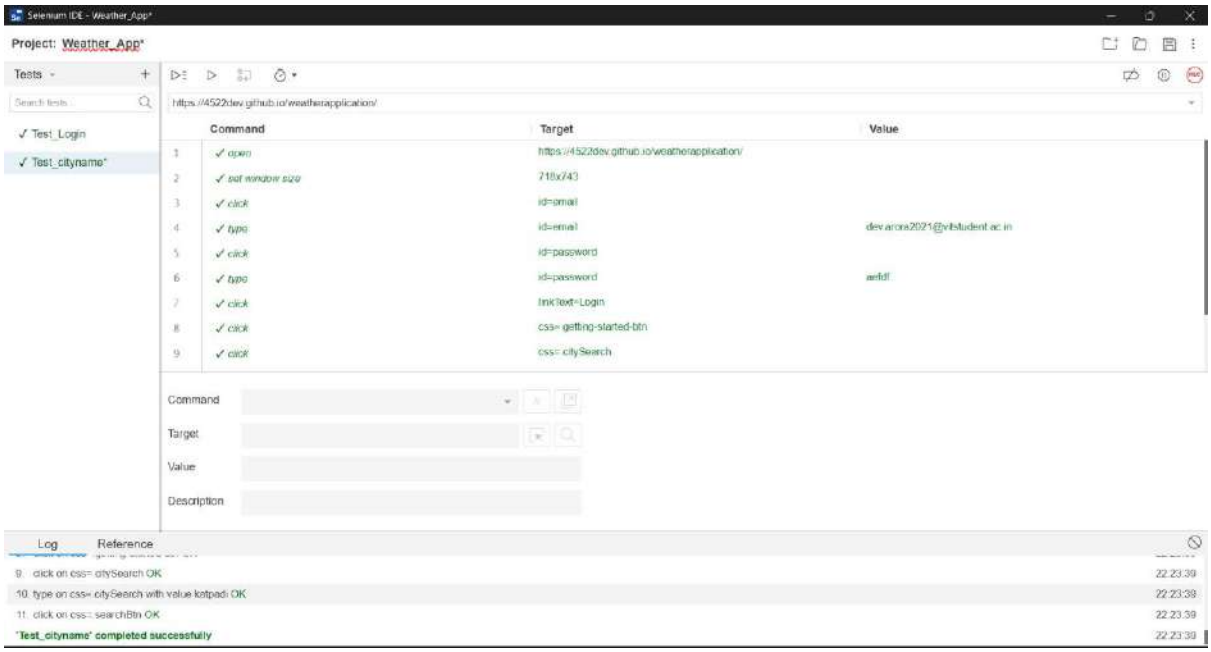


Name: Dev Arora

Register Number: 21BDS0130

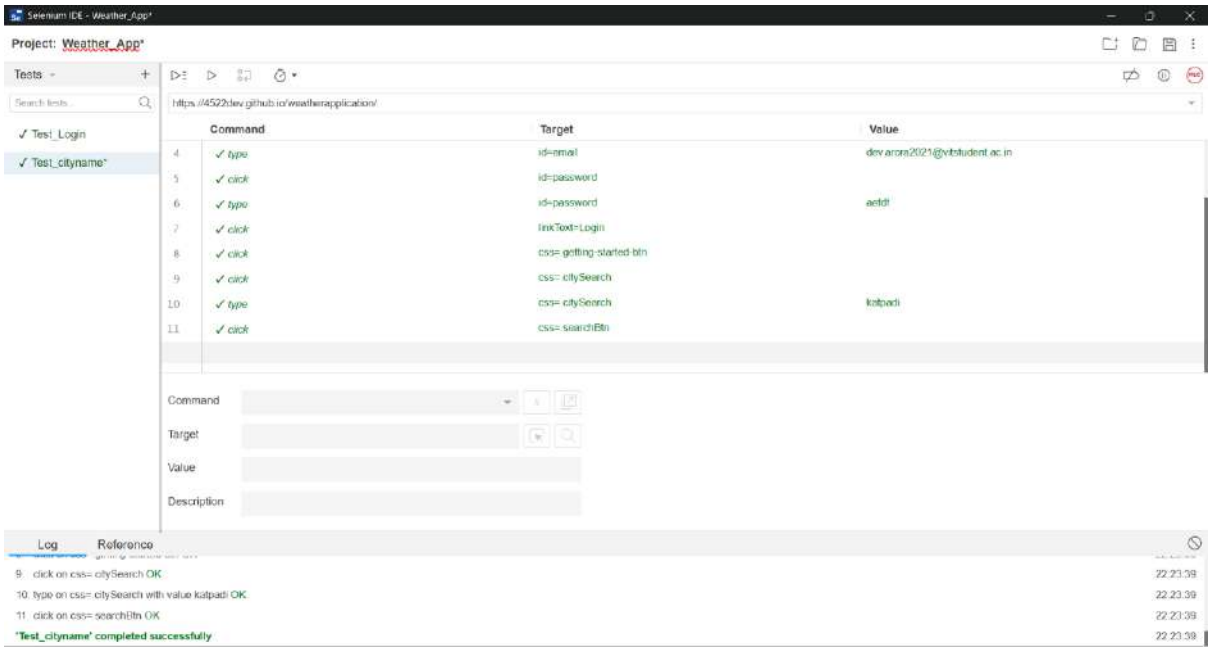


Since we Entered New York, it successfully provides the details required, now we stop the recording and run our test case:

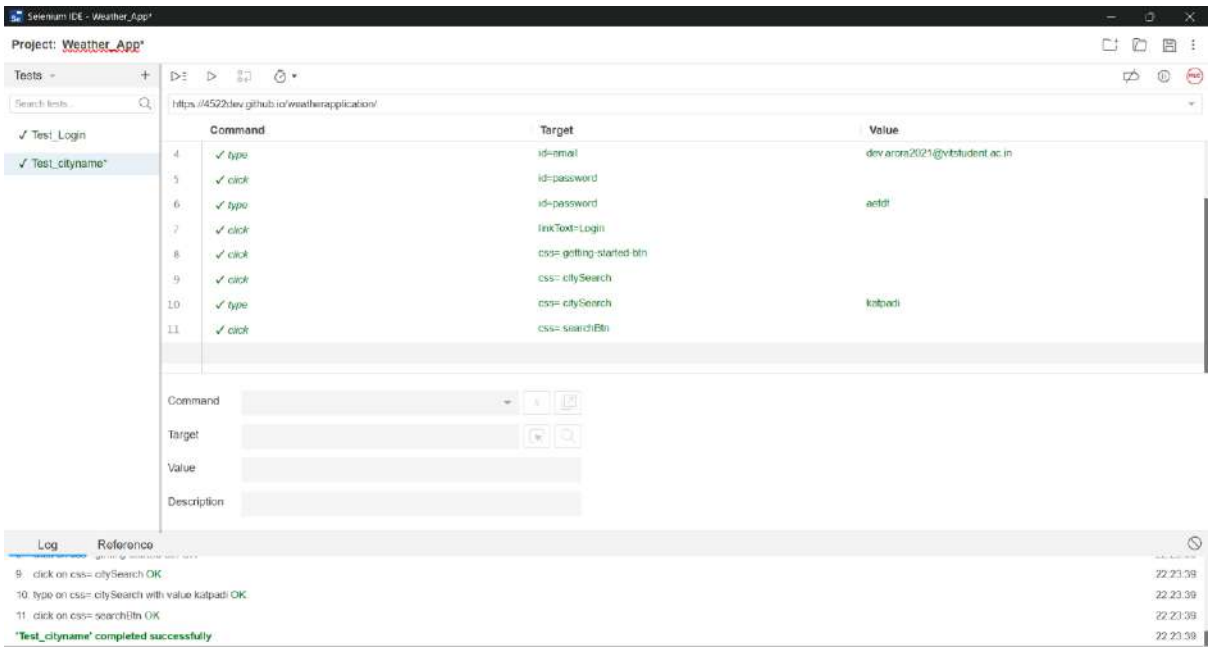


Name: Dev Arora

Register Number: 21BDS0130



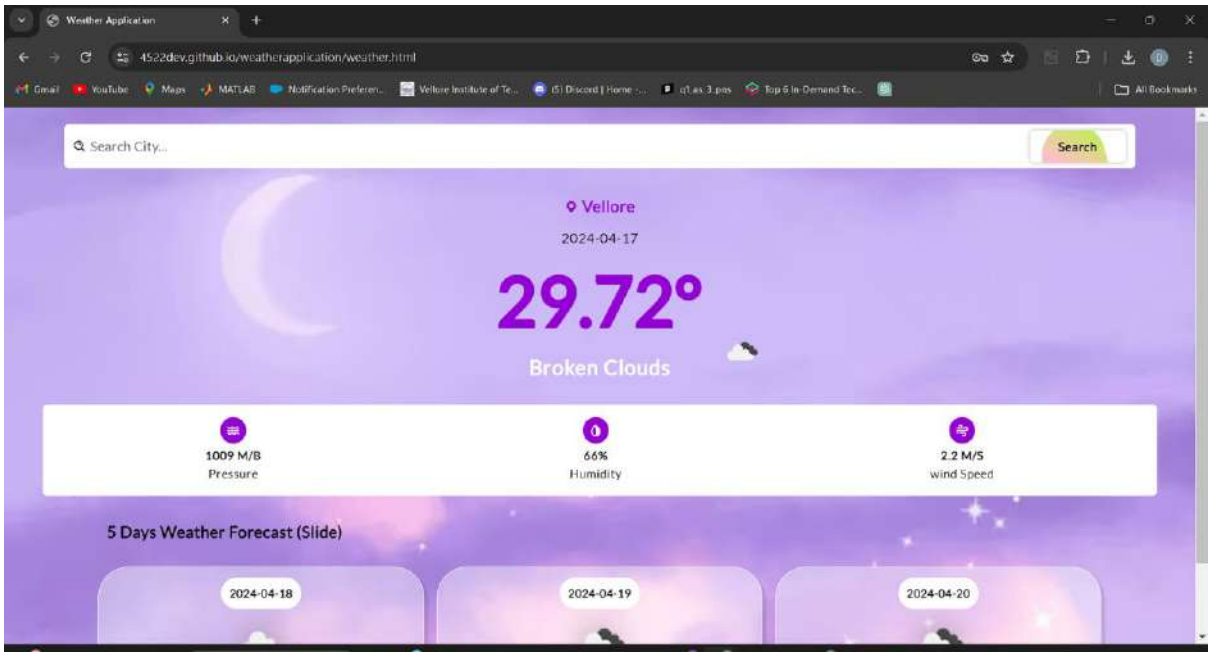
Thus, we see 2nd Test Case is also executed successfully



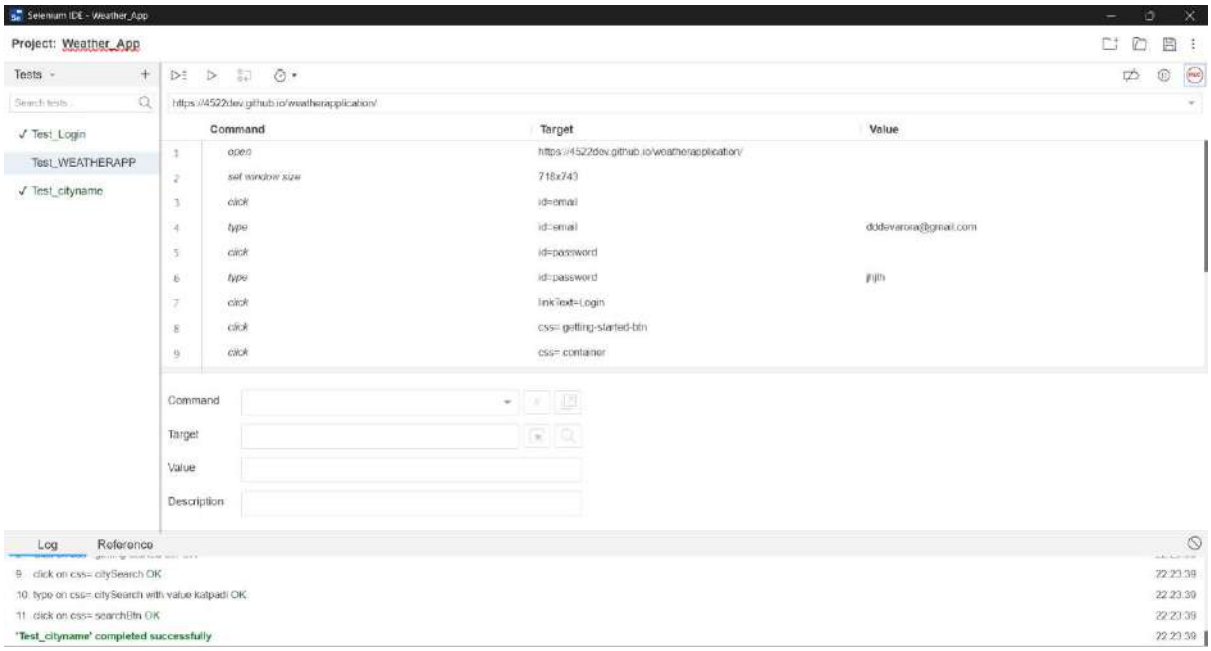
We add a new test Case to check the details of our weather app

Name: Dev Arora

Register Number: 21BDS0130

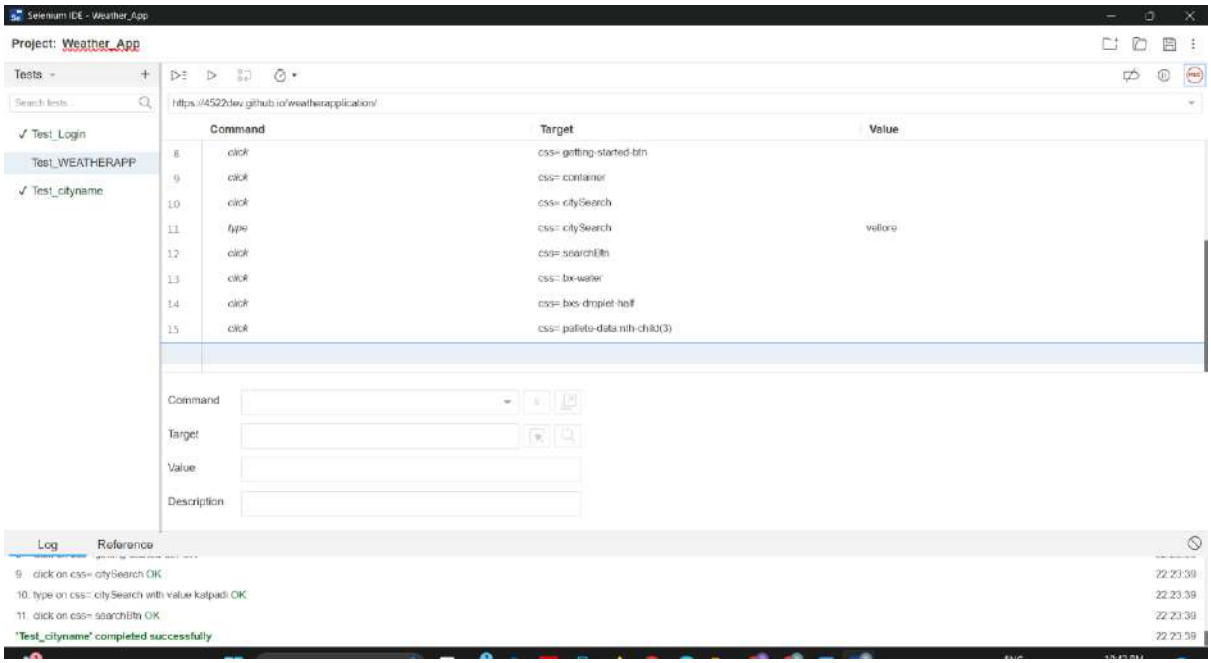


Thus, the details are recorded by our Selenium now we proceed to run our test case3

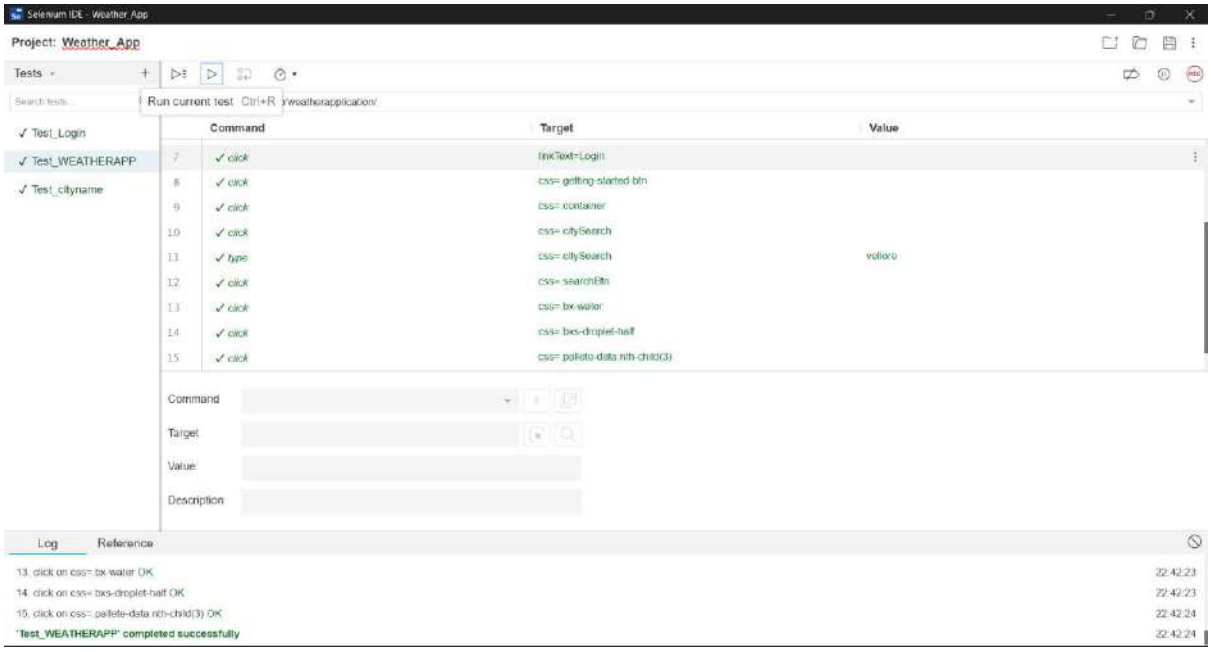


Name: Dev Arora

Register Number: 21BDS0130



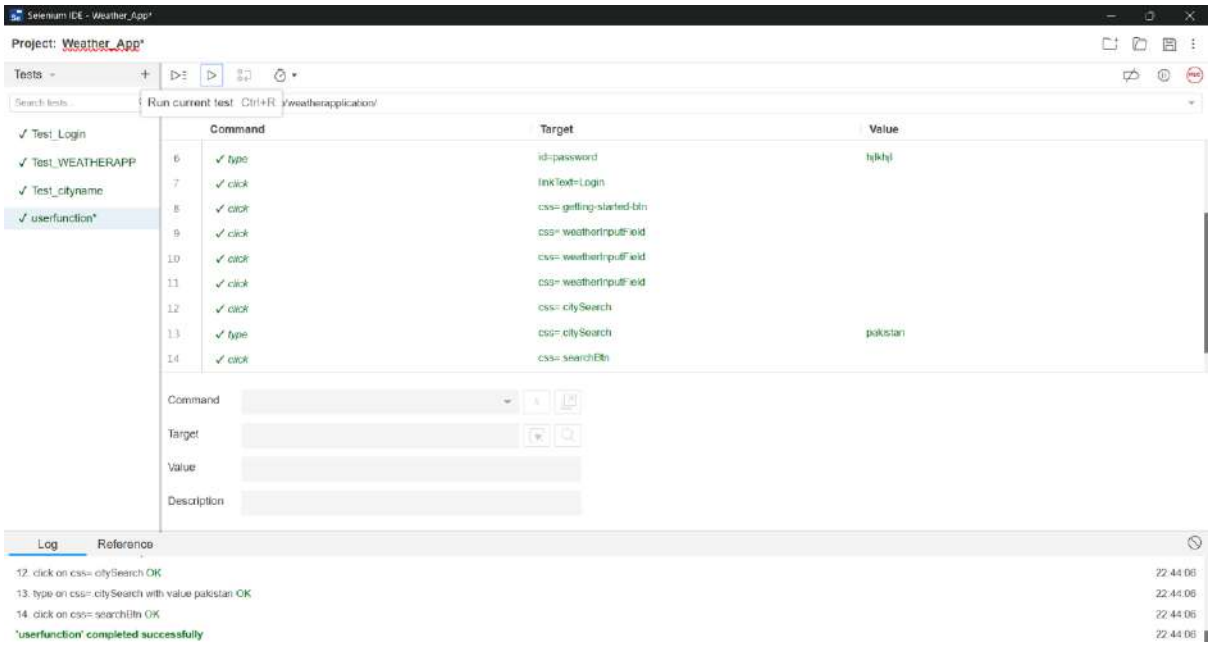
We've successfully tested the details represented by our weather app



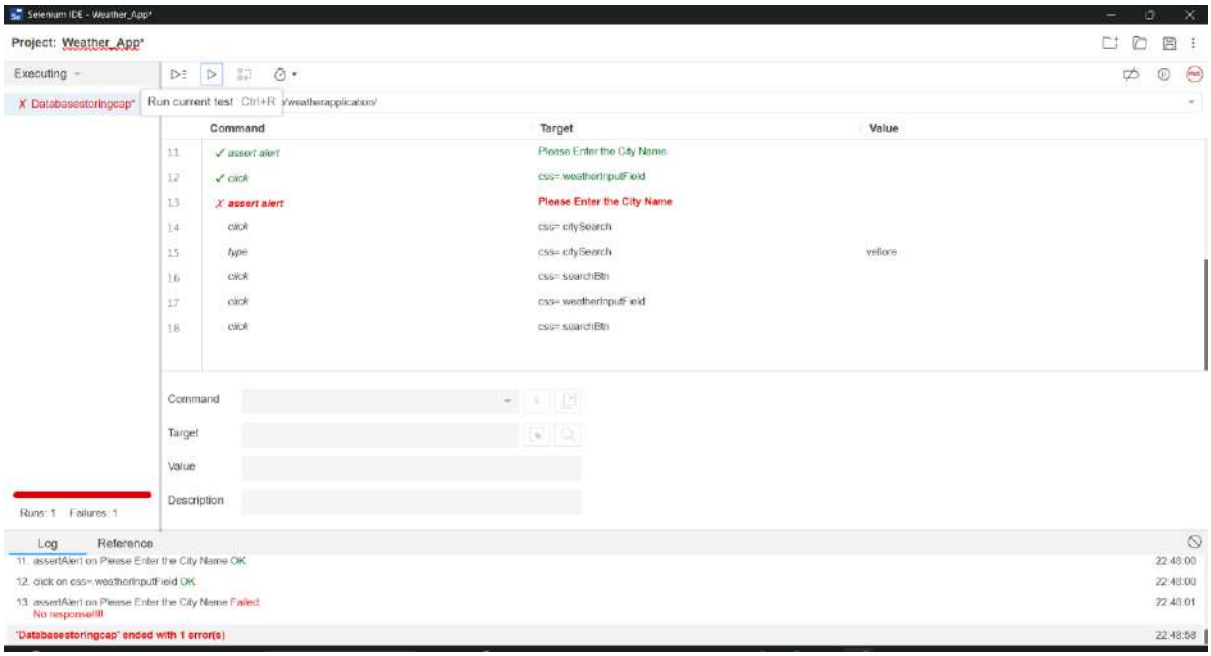
Thus, we proceed on with 2 more Test Cases: 1 to check User functionality, other to test database Storing functionality

Name: Dev Arora

Register Number: 21BDS0130



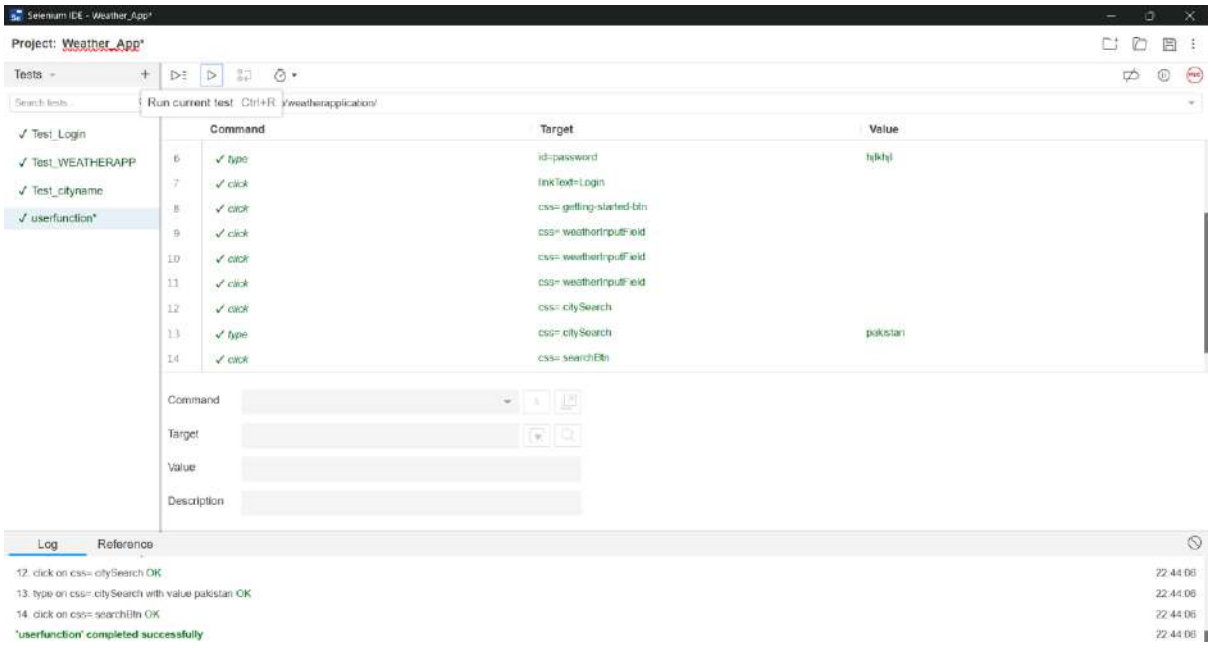
Now, since our project is a small react JS weather App, it isn't able to have an extensive database to store info about all the user details which have been entered, hence it fails this testcase

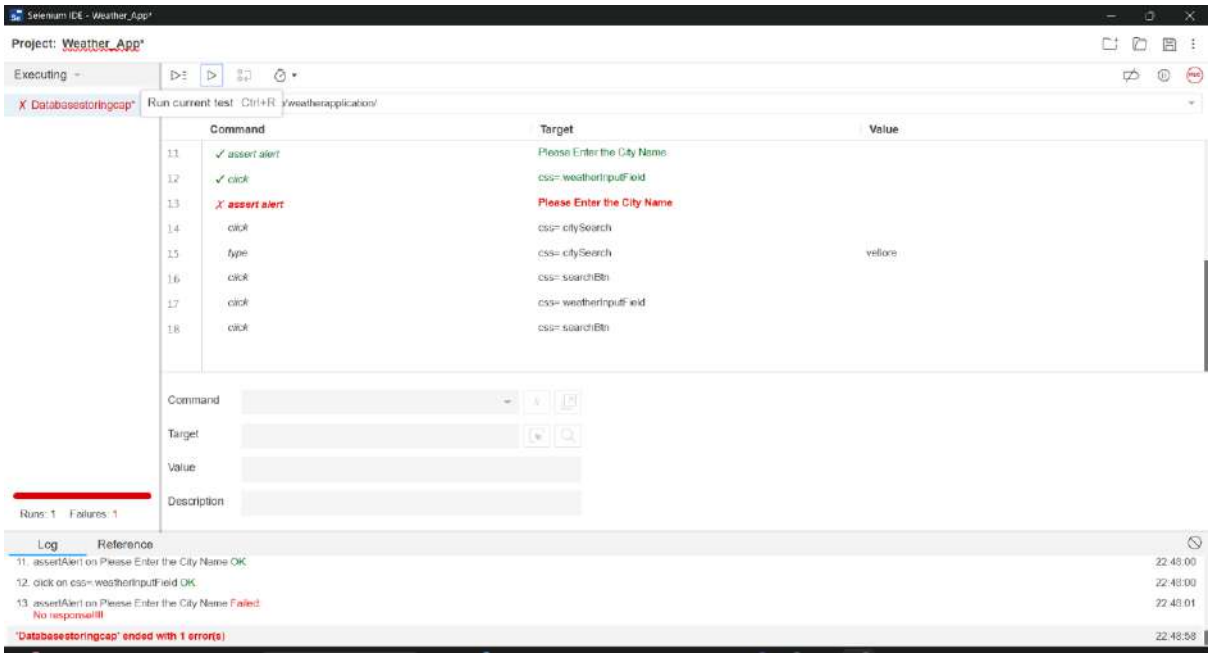


CONCLUSION

Name: Dev Arora

Register Number: 21BDS0130





Test Case{S.No.}	Name of Test Case	Pass/Fail
Test_Case1	Test_Login	Pass
Test_Case2	Test_cityname	Pass
Test_Case3	Test_WEATHERAPP	Pass
Test_Case4	userfunction	Pass
Test_Case5	Databasestoringcap	Fail

- The testing process for the small React JS weather application involved executing five test cases, and the results are summarized

in the table. The first four test cases, Test_Login, Test_cityname, Test_WEATHERAPP, and userfunction, passed successfully, indicating that the application's login functionality, city name input, weather details display, and user-related features are working as expected.

- However, the Databasestoringcap test case failed, highlighting a limitation of the application. As mentioned in the description, the project is a small React JS weather app and is not designed to have an extensive database for storing user details. This constraint is intentional, as the application's primary purpose is to provide weather information, and storing user data is not a core requirement.

- The failure of the Databasestoringcap test case does not necessarily imply a critical issue, as it aligns with the project's scope and limitations. The developers consciously decided not to implement an extensive database for user data storage, focusing instead on delivering the weather information functionality efficiently.

- While the lack of user data storage may be a potential enhancement for future iterations, the current implementation meets the primary objectives of the project. The successful execution of the other test cases demonstrates that the application's core features are functioning correctly, ensuring a satisfactory user experience for obtaining weather information.

FUTURE ENHANCEMENTS

Enhancements:

1. *Location Services Integration*: Incorporate location services to automatically detect the user's current location and provide weather information without requiring manual input of the city name. This

feature can enhance the user experience and make the application more convenient.

2. *Responsive Design:* Optimize the application's user interface for various screen sizes and devices, ensuring a seamless experience across desktops, tablets, and mobile phones.

3. *Multilingual Support*: Implement multilingual support to cater to a broader user base by providing weather information in different languages based on the user's preferences or device settings.

4. *Additional Weather Data:* Expand the weather data displayed by including more detailed information such as air quality index, UV index, and pollen count. These additional data points can be valuable for users with specific concerns or preferences.

5. *Weather Alerts and Notifications*: Implement a system to provide real-time weather alerts and notifications for severe weather conditions, such as thunderstorms, hurricanes, or heat waves, in the user's location.

6. *Historical Weather Data*: Incorporate historical weather data to allow users to view past weather patterns and trends for a specific location.

7. *User Preferences and Customization:* Allow users to customize the application's appearance, units of measurement, and preferred data display formats based on their preferences.

8. *Social Integration*: Integrate with social media platforms to enable users to share weather updates, forecasts, or interesting weather-related content with their friends and followers.

Name: Dev Arora

Register Number: 21BDS0130

Achievements:

1. Functional Weather Application: The application appears to be functional, displaying the current weather conditions, including temperature, pressure, humidity, and wind speed for the searched city (Vellore).

2. 5-Day Weather Forecast: The application provides a 5-day weather forecast, allowing users to plan their activities or make informed decisions based on the upcoming weather conditions.

3. User-Friendly Interface: The interface seems clean and user-friendly, with a prominent search bar for entering the city name and a visually appealing layout for displaying weather information.

4. Responsive Behavior: The "Selenium IDE is recording..." message in the image suggests that the application is being tested for responsiveness and functionality using automation tools like Selenium.

5. Successful Integration with Weather API: The application appears to be successfully integrating with a weather API to fetch and display accurate weather data for the requested location.

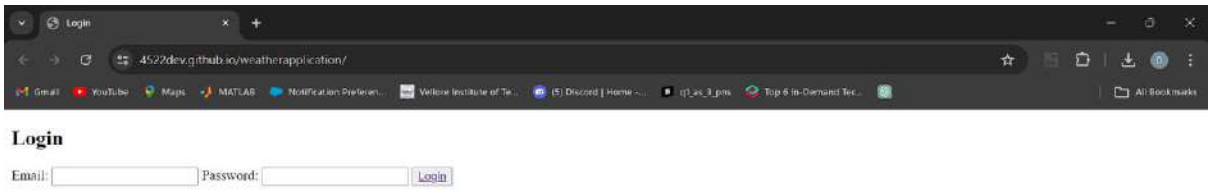
Overall, the weather application appears to be a functional and visually appealing solution for obtaining weather information. However, there is always room for improvement and additional features to enhance the user experience and provide more comprehensive weather-related information.

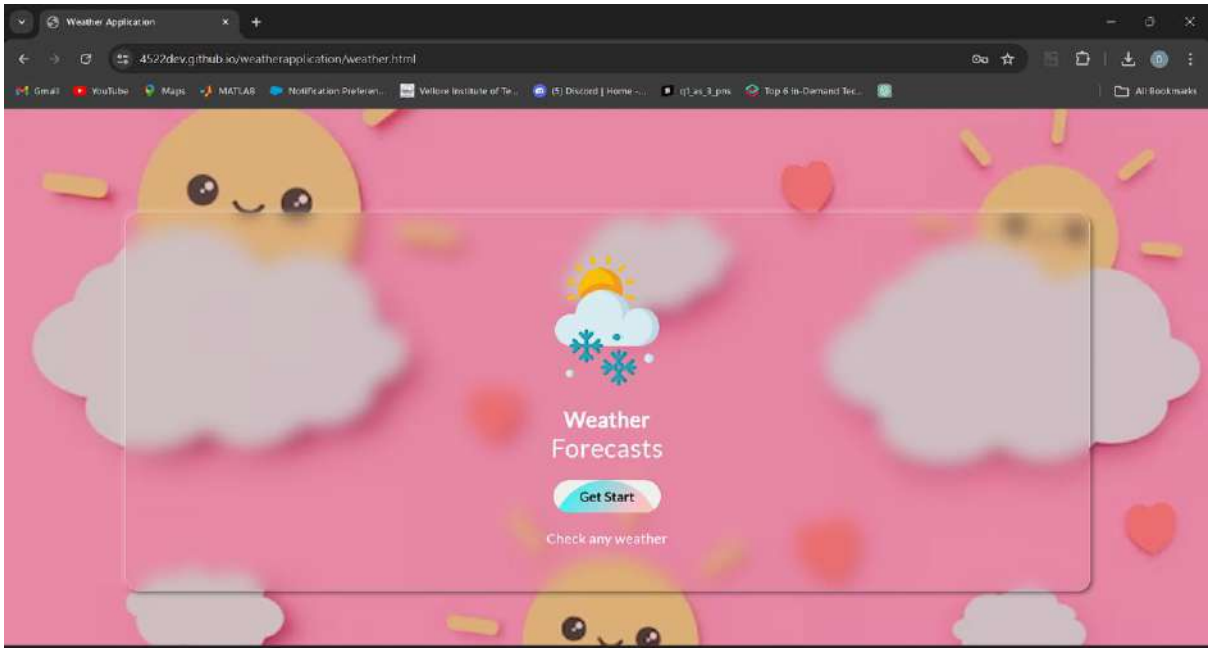
OR

Website:(https://4522dev.github.io/weatherapplication/)

Name: Dev Arora

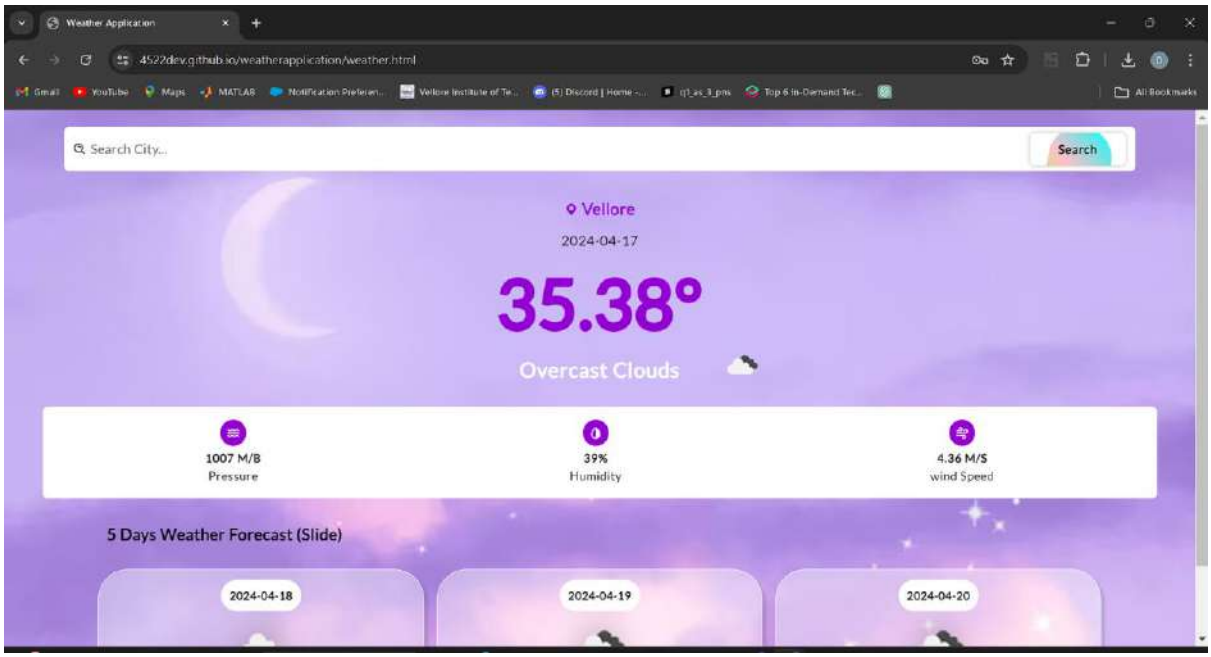
Register Number: 21BDS0130

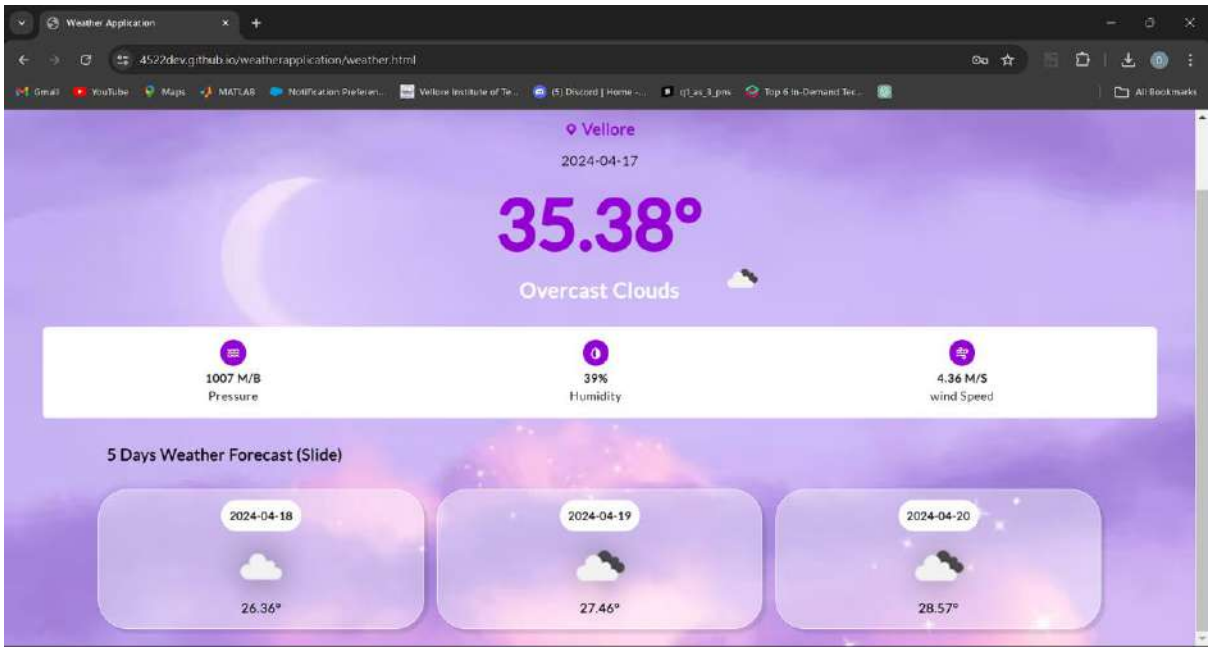




Name: Dev Arora

Register Number: 21BDS0130



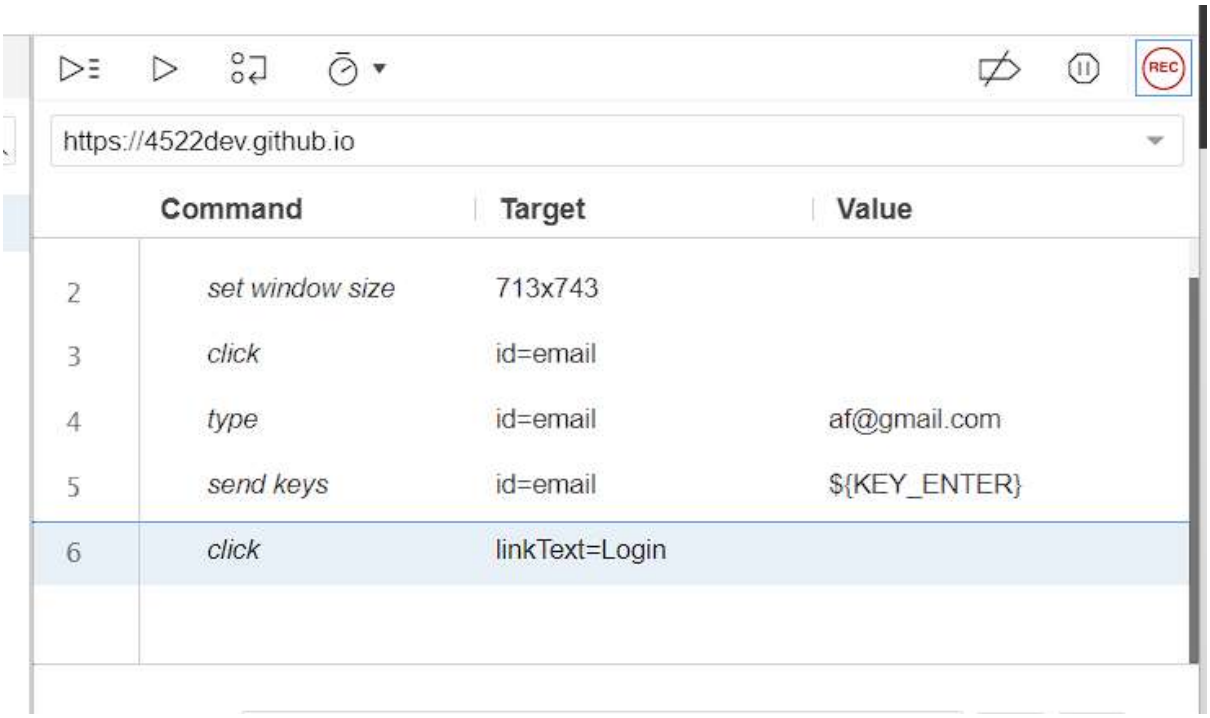


Testcase:

1. Failed Testcase

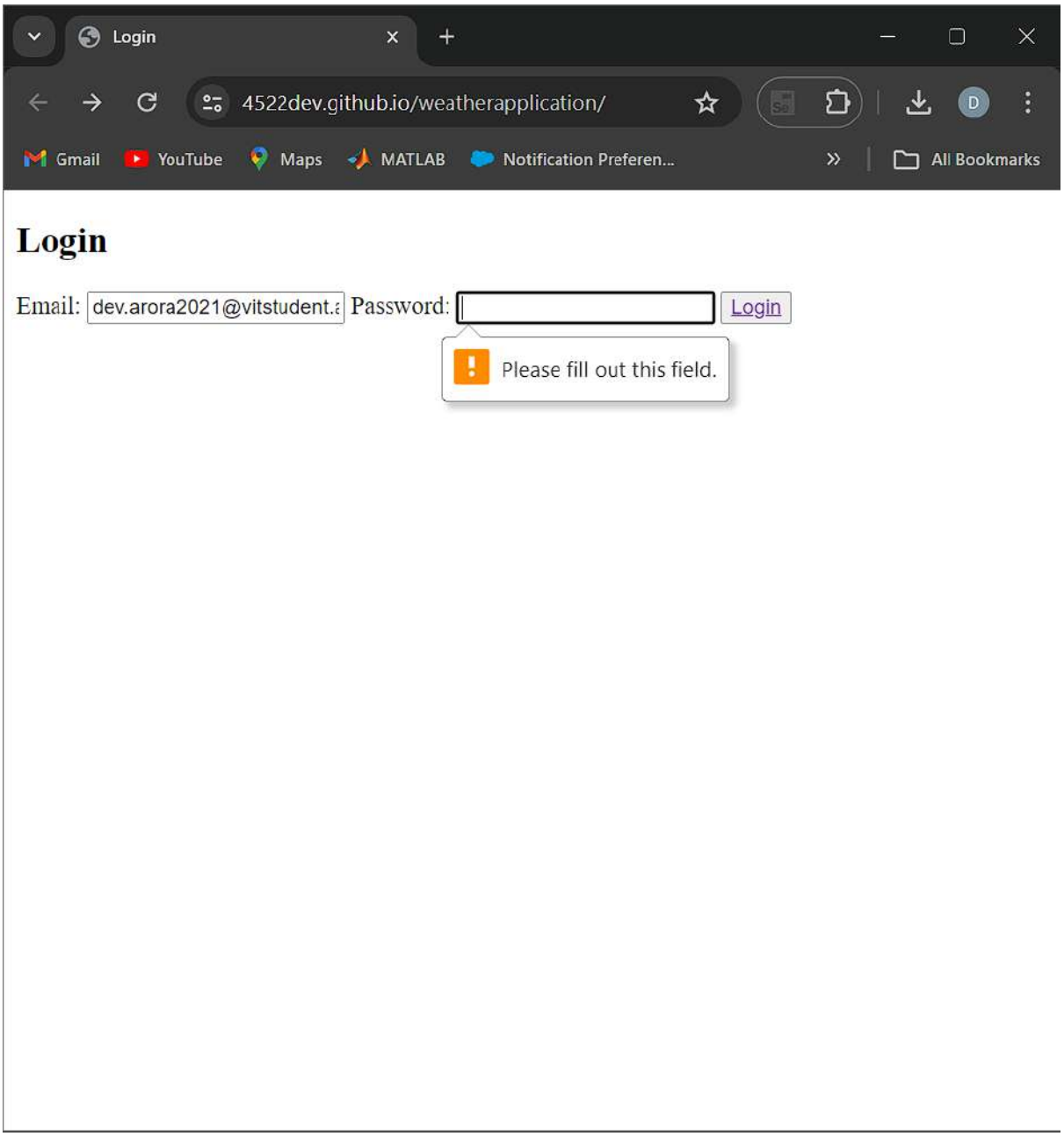
Name: Dev Arora

Register Number: 21BDS0130



Name: Dev Arora

Register Number: 21BDS0130



2. Passed Testcase

Name: Dev Arora

Register Number: 21BDS0130