* **Cover Page**

1.1 Project Title, Author, Date

1.2 Course, Instituition, Instructor

1.3 Project Overview

* **Class-Responsibility-Collaborator (CRC) Cards**

2.1 CRC Card for WeatherRVModal.java

2.2 CRC Card for WeatherRVAdapter.java

2.3 CRC Card for MainActivity.java

2.4 CRC Card for LoginActivity.java

2.5 CRC Card for SignUpActivity.java

* **System Architecture**
  1. Architecture Diagram
  2. Description of Architecture and Design Patterns

COVER PAGE

**Project Title:** METEORMAP  
**Author:** Ansh Puri, Yashavi Mahajan, Rohan Patel, Robin Masson  
**Date:** 17th November, 2024.

**Course:** EECS 3311B  
**Institution:** YORK UNIVERSITY  
**Instructor:** Ilir Dema

**Project Overview**:  
This project involves the design and development of a weather forecasting application with the following functionalities:

* **WeatherRVModal.java**: Acts as the data model for storing hourly weather forecast information, including attributes like time, temperature, weather icons, and wind speed.
* **WeatherRVAdapter.java**: Serves as the adapter for binding WeatherRVModal data to a RecyclerView, displaying hourly forecasts in a user-friendly format.
* **MainActivity.java**: Manages the app's primary functionality, including:

1. Retrieving weather data via APIs.
2. Handling user interactions, such as searching for weather by city or fetching the current location's weather.
3. Displaying weather data on the main screen with a dynamic, real-time UI.

CRC CARDS

This project includes 3 java classes named as WeatherRVModal.java, WeatherRVAdapter.java and MainActivity.java

CRC card for **WeatherRVModal.java**

|  |
| --- |
| **Class Name:** WeatherRVModal.java |
| **Parent Class:** WeatherRVModal  **SubClasses:** getTemperature, getTime, getIcon, getWindSpeed, WeatherRVModal.   |  |  | | --- | --- | | **Responsibilities:**   **Store weather data attributes**:   1. Time: The time of the forecasted weather data. 2. Temperature: The temperature at the specified time. 3. Icon: The URL for the weather condition icon at the specified time. 4. windspeed: The wind speed at the specified time.  * Getter and setter methods for each attribute (time, temperature, icon, windSpeed) to allow easy access and modification of weather data. | **Collaborations**:   * **WeatherRVAdapter**: Uses WeatherRVModal instances to bind data to the RecyclerView, displaying each hour’s forecast (time, temperature, icon, wind speed). * **MainActivity**: Creates instances of WeatherRVModal to represent each hour’s weather forecast data retrieved from the weather API and passes them to WeatherRVAdapter for display. | |

CRC card for **MainActivity.java**

|  |
| --- |
| **Class Name:** MainActivity.java |
| **Parent Class:** AppCompatActivity  **SubClasses:** onCreate(Bundle savedInstanceState), onClick(View view), onClick(View v), onRequestPermissionsResult(…), getCityName(…), getWeatherInfo(…), onErrorResponse(…)   |  |  | | --- | --- | | **Responsibilities:**   * Initialize and manage UI components for displaying weather data. * Request location permissions and retrieve the user's location. * Fetch weather data from a remote API and update the UI with weather information. * Handle city-based weather lookup and display data. | **Collaborations:**   * WeatherRVAdapter: Used to display a list of hourly weather forecasts in a RecyclerView. * WeatherRVModal: Represents each hourly forecast data item shown in the RecyclerView. * LocationManager: Manages location services to get the user's current city. * Volley and JsonObjectRequest: Used for making network requests to retrieve weather data. * Picasso: Loads and displays images from URLs, such as weather icons and background images. * Geocoder: Converts latitude and longitude coordinates into a city name. | |

CRC card for **WeatherRVAdapter.java**

|  |
| --- |
| **Class Name:** WeatherRVAdapter.java |
| **Parent Class:** RecyclerView.Adapter  **SubClasses:** WeatherRCAdapter, onCreateViewHolder, onBindViewHolder, getItemCount(),ViewHolder   |  |  | | --- | --- | | **Responsibilities:**   * **Display hourly weather data**: Bind data from WeatherRVModal to individual items in the RecyclerView, showing attributes like time, temperature, weather icon, and wind speed for each hour. * **Inflate item layout**: Inflate the layout resource file (weather\_rv\_item.xml) for each item in the RecyclerView. * **Load images**: Use Picasso to load weather condition icons from URLs and display them in the item’s ImageView. * **Format time data**: Parse and format the time data from a String format to display in a user-friendly format (e.g., 12:00 AM). | **Collaborations:**   * **WeatherRVModal**: Provides weather data for each item in the RecyclerView, including time, temperature, icon URL, and wind speed. * **MainActivity**: Initializes and sets up the WeatherRVAdapter with an ArrayList of WeatherRVModal instances representing the hourly weather forecast. * **Picasso**: Loads and displays the weather condition icon for each hour’s forecast item in the RecyclerView. | |

CRC card for **LoginActivity.java**

|  |
| --- |
| **Class Name:** LoginActivity.java |
| **Parent Class:** onCreate(…), onResponse(…), onFailure(…), isUserLoggedIn(), storeJwtToken().   |  |  | | --- | --- | | **Responsibilities:**   Display and manage the login screen UI components.   Validate and send login credentials (email and password) to the server.   Handle server responses to log in the user or show error messages.   Manage session persistence by storing and checking the JWT token.   Redirect authenticated users to the main activity. | **Collaborations:**   Sends login requests to the server using the login() method.   Redirects the user upon successful login.   Redirects users to the sign-up screen when the "Sign Up" link is clicked.   Stores the JWT token for session management. | |

CRC card for **SignUpActivity.java**

|  |
| --- |
| **Class Name:** SignUpActivity.java |
| **Parent Class:** onCreate(…), onResponse(…), onFailure(…)   |  |  | | --- | --- | | **Responsibilities:**   Display and manage the sign-up screen UI components.   Collect and validate user information (name, email, password).   Send the sign-up data to the server.   Handle server responses to show success or error messages.   Redirect successful sign-ups to the login screen. | **Collaborations:**   Sends sign-up requests to the server using the signup() method.   Redirects users to the login screen upon clicking the "Login" link or after successful sign-up.   Encapsulates user input (name, email, password) to send in the API call.   Parses and handles the server response | |

SOFTWARE ARCHITECTURAL DESIGN

+----------------------------------+

| LoginActivity |

|----------------------------- |

| - Handles user login |

| - Validates credentials |

| - Stores JWT for sessions. |

| - Navigates to MainActivity. |

+-------------+---------------+

|

V

+----------------------------------+ +-------------------------------+

| SignupActivity | | MainActivity |

|-----------------------------------| |-------------------------------|

| - Registers new users | | - Manages main UI for weather |

| - Sends data to API | | - Fetches weather info via API|

| - Navigates to LoginActivity| | - Updates weather views |

+-------------+-------------------+ +-------------------------------+

| |

V V

+---------------------------------+ +-------------------------------+

| WeatherRVAdapter | | WeatherRVModal |

|-----------------------------------| |------------------------------------|

| - Binds weather data to UI | | - Stores weather data |

| - Populates RecyclerView | | - Attributes: time, temp, icon|

| - Loads weather icons | | and wind speed |

+----------------------------------+ +------------------------------------+

|

V

+---------------------------------+

| API Service |

|---------------------------------- |

| - Handles API calls for |

| login, signup, and weather. |

| - Uses Retrofit for RESTful |

| service integration |

+----------------------------------+

**Explanation**

1. **LoginActivity (Controller)**:

* Responsible for managing the login process.
* Interacts with the ApiService to validate credentials.
* Stores JWT tokens for session management.
* Navigates to MainActivity upon successful login or SignupActivity for registration.

1. **SignupActivity (Controller)**:

* Handles user registration by sending data to the API.
* Navigates back to LoginActivity upon successful signup.

1. **MainActivity (Controller)**:

* Manages the main weather application UI.
* Interacts with the ApiService to fetch weather data.
* Populates the WeatherRVAdapter with data from WeatherRVModal.

1. **WeatherRVAdapter (View)**:

* Acts as the View in the MVC architecture for the weather forecast.

Binds data from WeatherRVModal to the UI components in a RecyclerView.

1. **WeatherRVModal (Model)**:

* Stores weather-related data like time, temperature, icons, and wind speed.
* Provides getter and setter methods for data encapsulation.

1. **ApiService**:

* A shared component that interacts with external APIs for login, signup, and weather data.
* Uses Retrofit for HTTP calls and JSON parsing.

**Key Characteristics**

* **MVC Adherence**: The application strictly follows the Model-View-Controller architecture.
* **Separation of Concerns**: Each component has clear and distinct responsibilities, ensuring maintainability and scalability.
* **Integration**: Uses Retrofit for API communication, making the system extensible for additional features or data sources.