**AZURE DEVOPS ENVIRONMENT SETUP**

**EXP NO: 1**

**Aim:**

Create Epic, Features, User Stories, Task

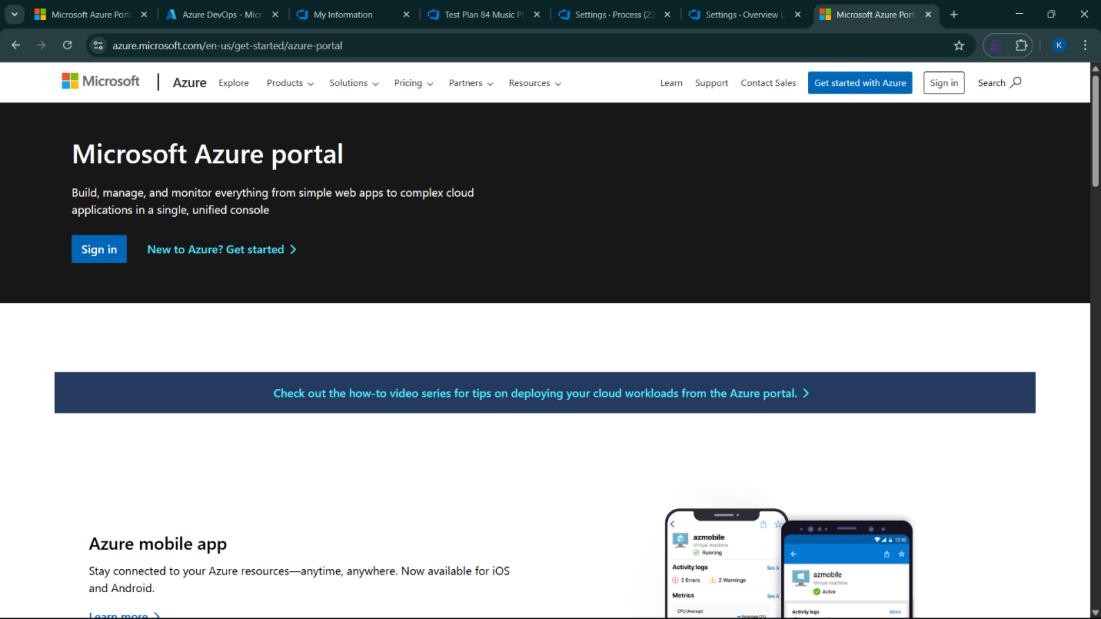
To set up and access the Azure DevOps environment by creating an organization through the Azure portal.

**INSTALLATION**

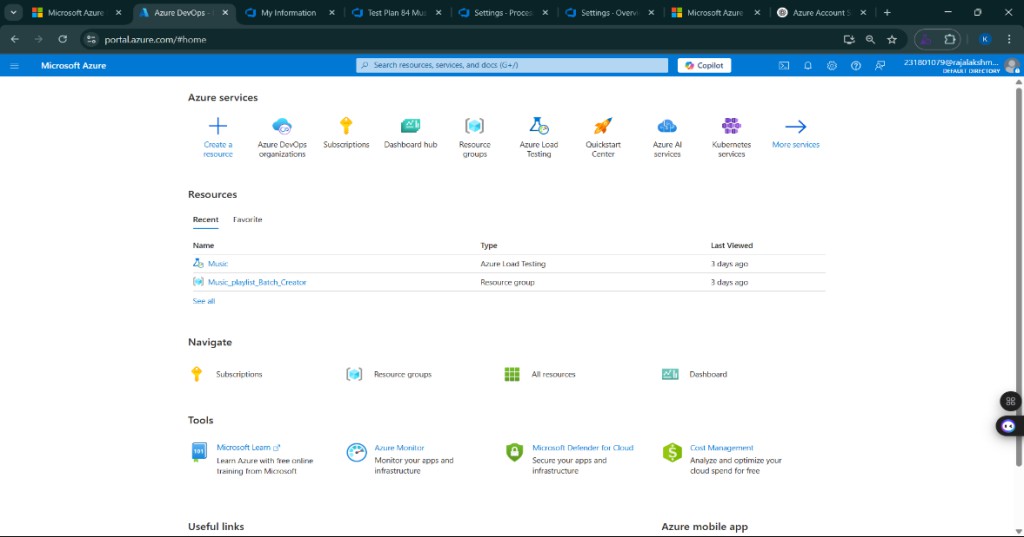
1. Open your web browser and go to the Azure website: [https://azure.microsoft.com/en-us/get-](https://azure.microsoft.com/en-us/get-started/azure-portal) [started/azure-portal](https://azure.microsoft.com/en-us/get-started/azure-portal).

Sign in using your Microsoft account credentials.

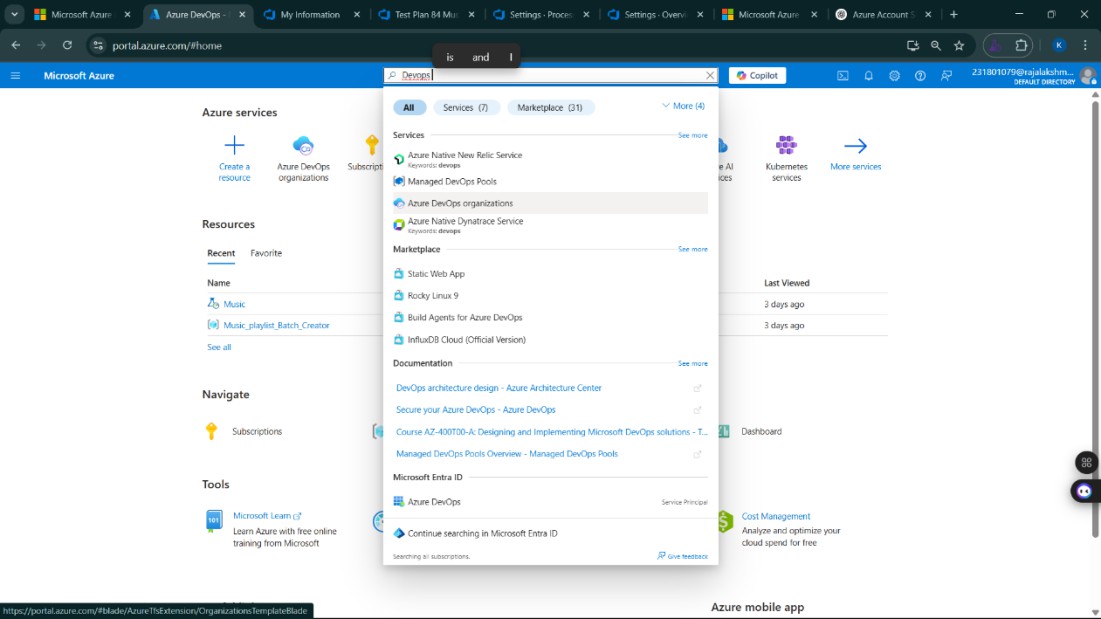
If you don't have a Microsoft account, you can create one here: <https://signup.live.com/?lic=1>



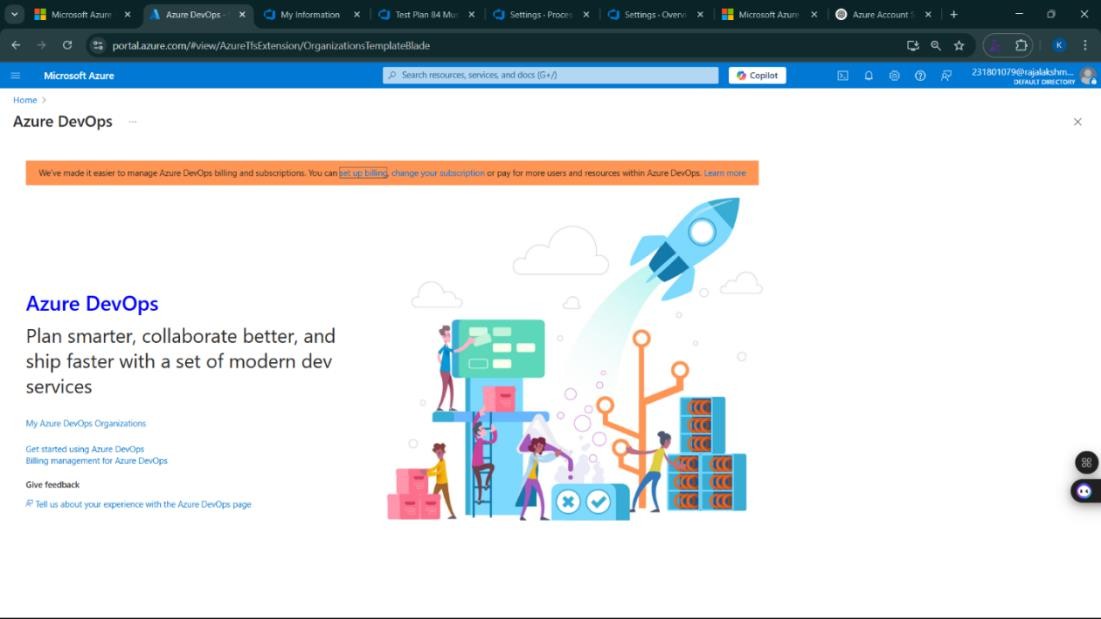
1. Azure home page



1. Open DevOps environment in the Azure platform by typing ***Azure DevOps Organizations*** in the search bar.



1. Click on the ***My Azure DevOps Organization*** link and create an organization and you should be taken to the Azure DevOps Organization Home page.



**Result:**

Successfully accessed the Azure DevOps environment and created a new organization through the Azure portal.

**AZURE DEVOPS PROJECT SETUP AND USER STORY MANAGEMENT**

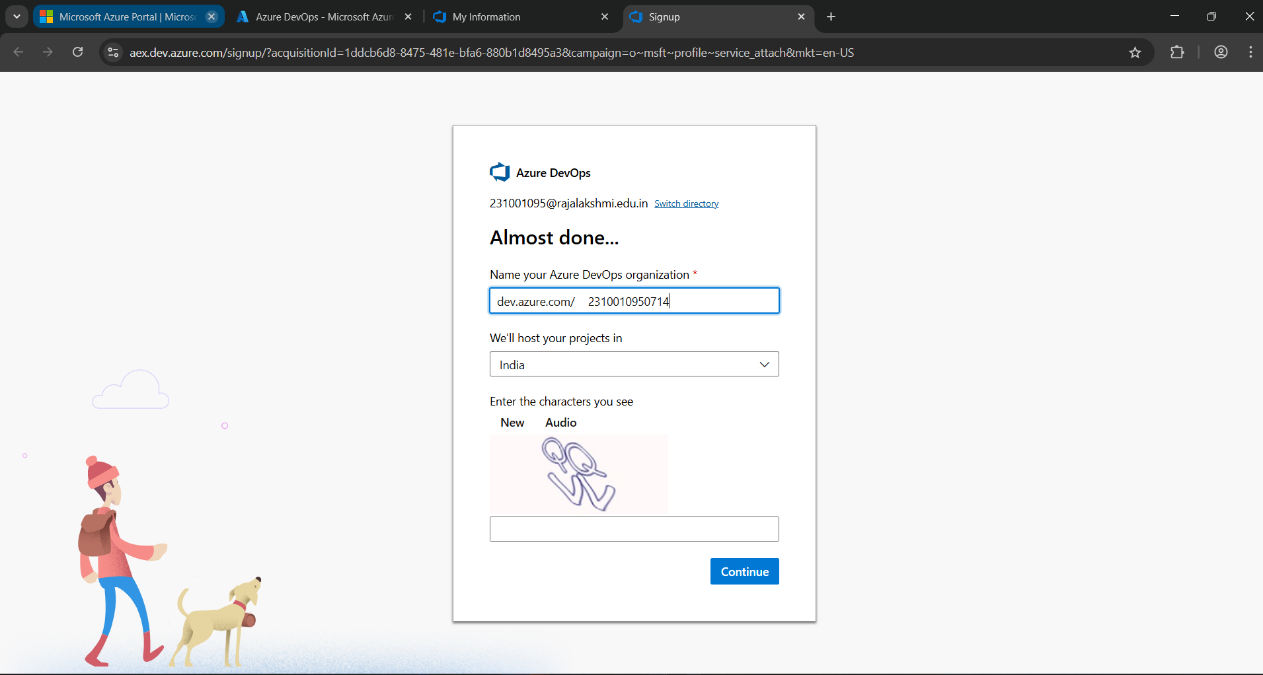
**EXP NO: 2**

**Aim:**

Create Epic, Features, User Stories, Task

To set up an Azure DevOps project for efficient collaboration and agile work management.

1. Create An Azure Account

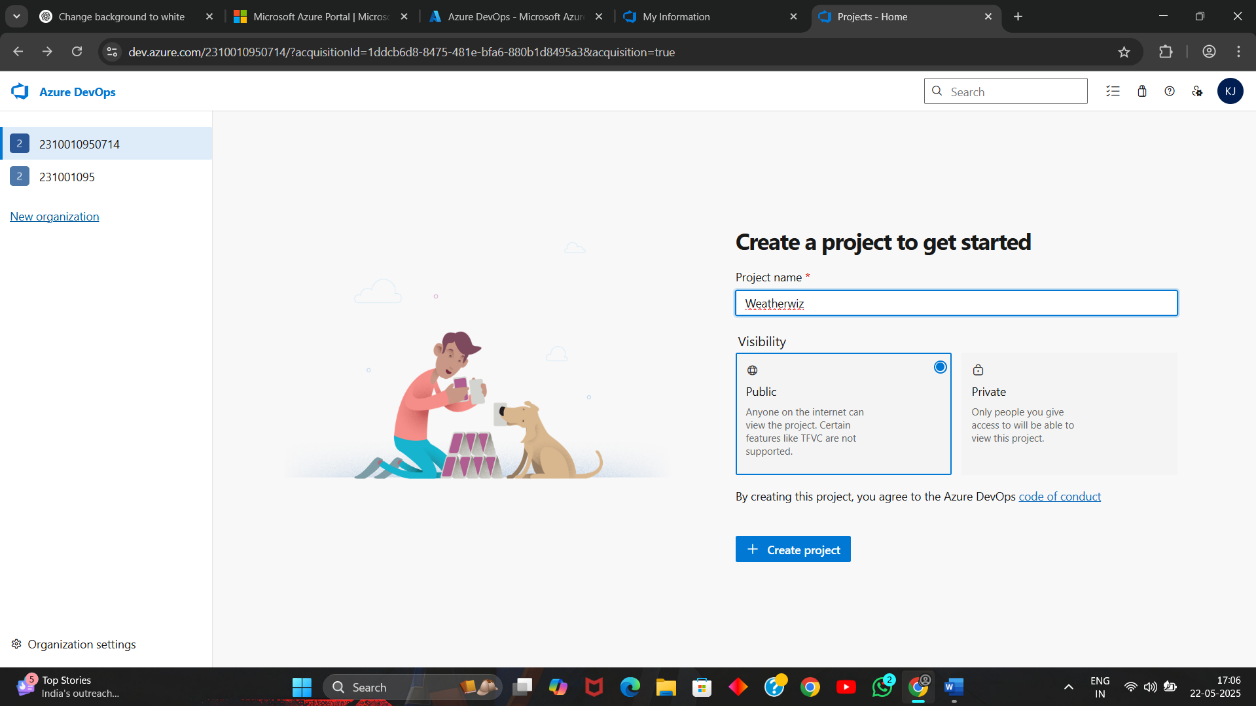


1. Create the First Project in Your Organization
   1. After the organization is set up, you’ll need to create your first **project**. This is where you'll begin to manage code, pipelines, work items, and more.
   2. On the organization’s **Home page**, click on the **New Project** button.
   3. Enter the project name, description, and visibility options:

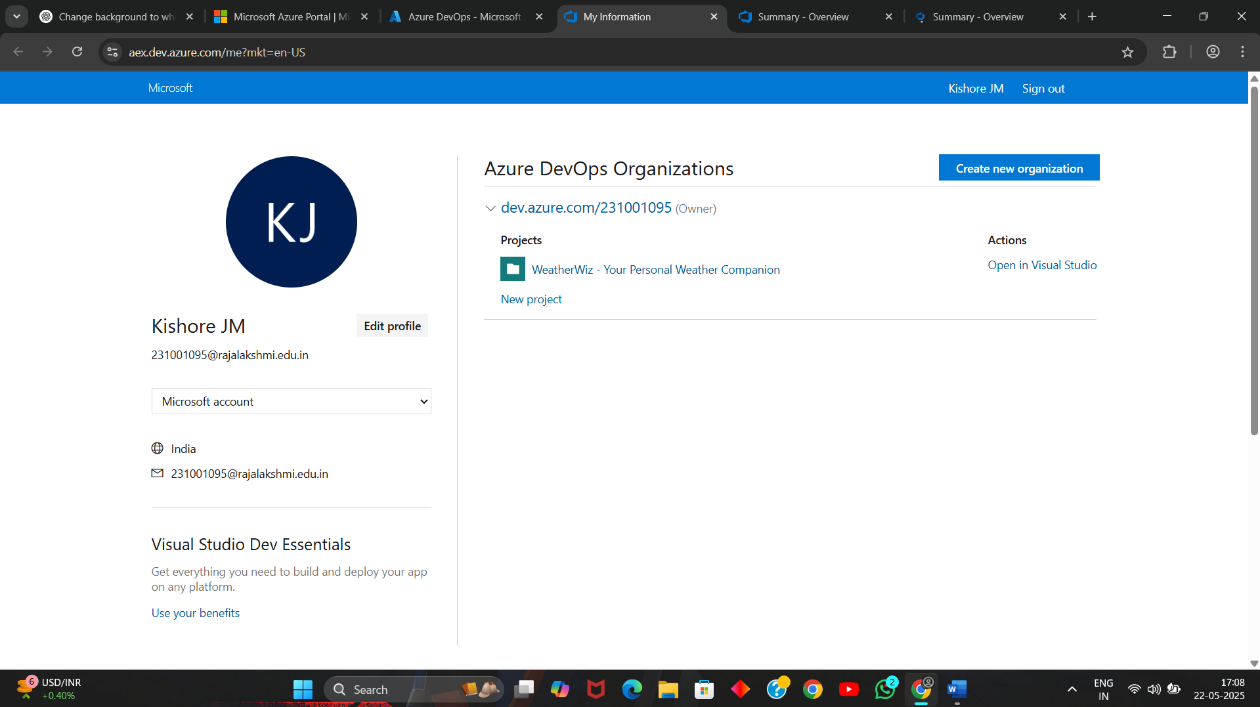
***Name****:* Choose a name for the project (e.g., WeatherWiz).

***Description****:* Optionally, add a description to provide more context about the project. ***Visibility****:* Choose whether you want the project to be **Private** (accessible only to those invited) or **Public** (accessible to anyone).

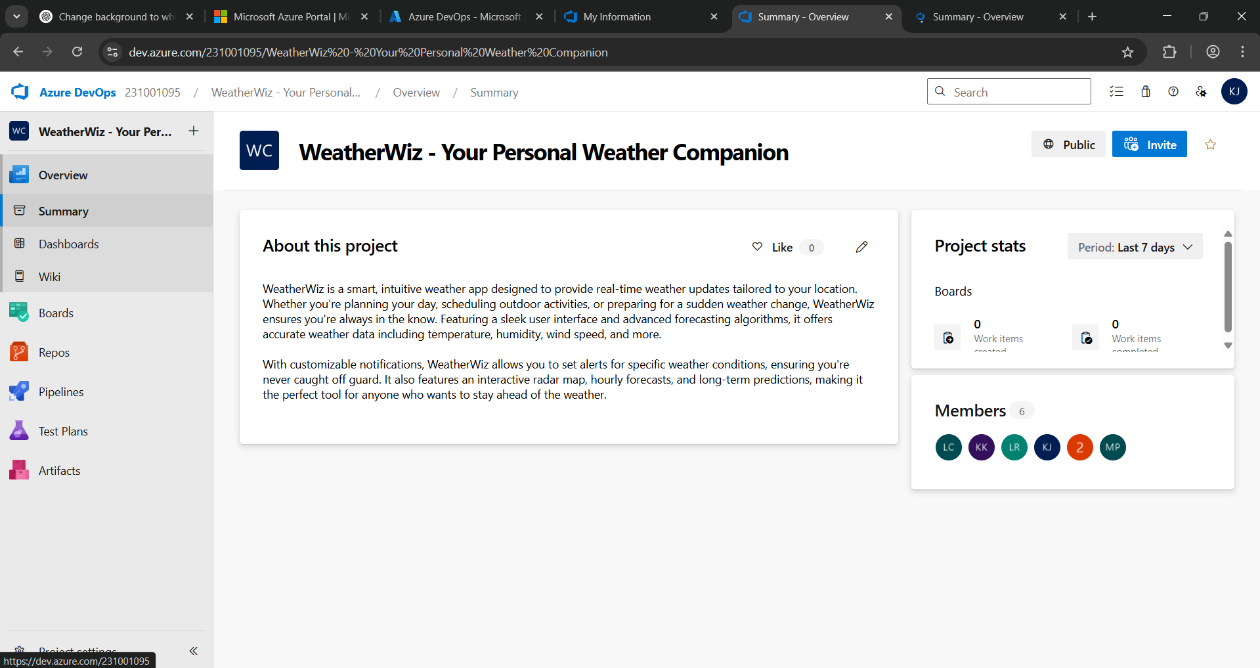
* 1. Once you’ve filled out the details, click **Create** to set up your first project.



1. Once logged in, ensure you are in the correct organization. If you're part of multiple organizations, you can switch between them from the top left corner (next to your user profile). Click on the Organization name, and you should be taken to the Azure DevOps Organization Home page.



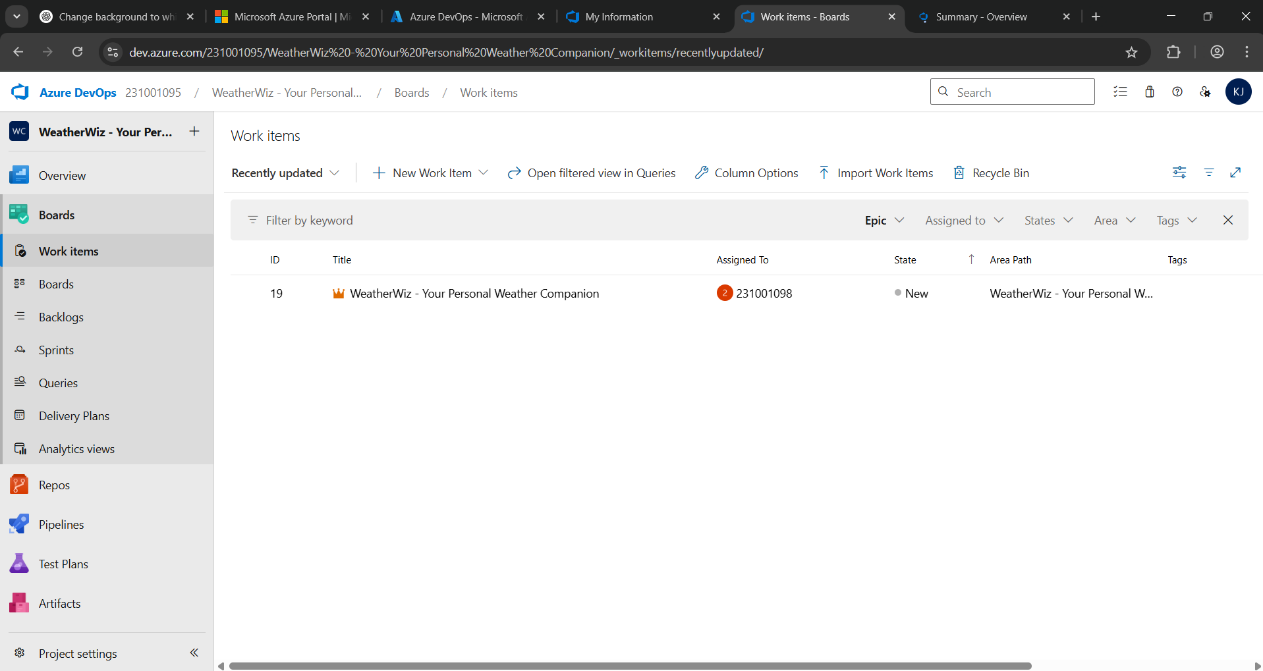
1. Project dashboard

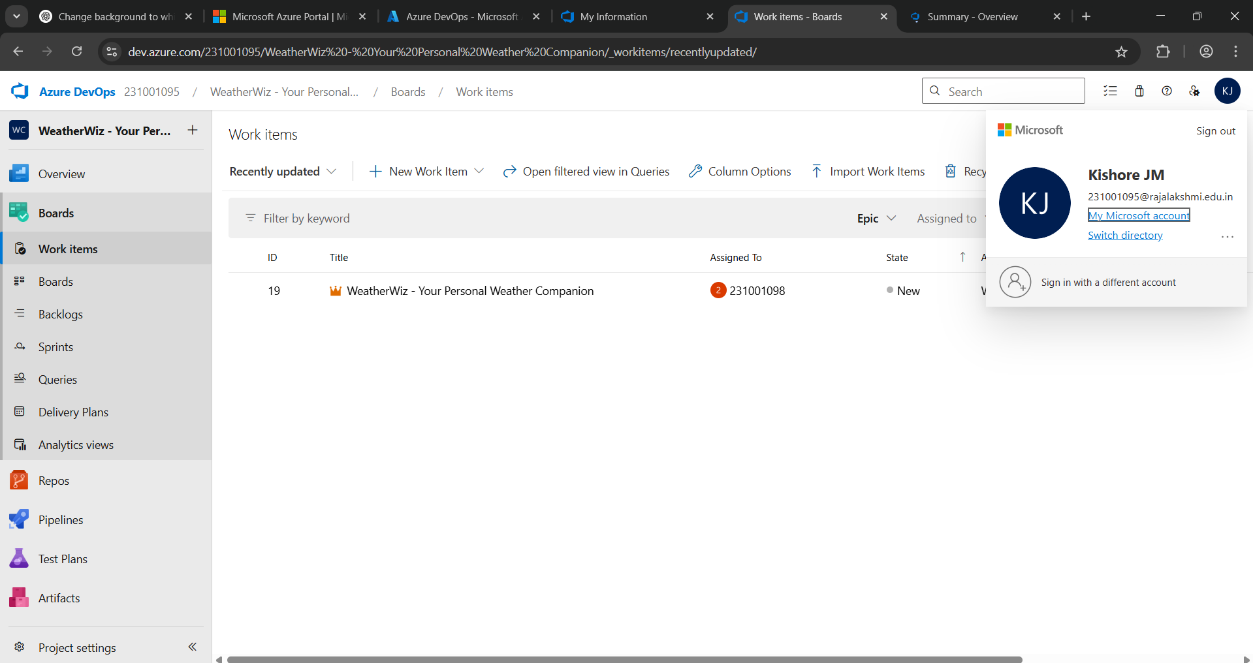


1. To manage user stories:
   1. From the **left-hand navigation menu**, click on **Boards**. This will take you to the main **Boards**

page, where you can manage work items, backlogs, and sprints.

* 1. On the **work items** page, you'll see the option to **Add a work item** at the top. Alternatively, you can find a **+** button or **Add New Work Item** depending on the view you're in. From the **Add a work item** dropdown, select **User Story**. This will open a form to enter details for the new User Story.





**Result:**

setup.

Successfully created an Azure DevOps project with user story management and agile workflow

**SETTING UP EPICS, FEATURES, AND USER STORIES FOR PROJECT PLANNING**

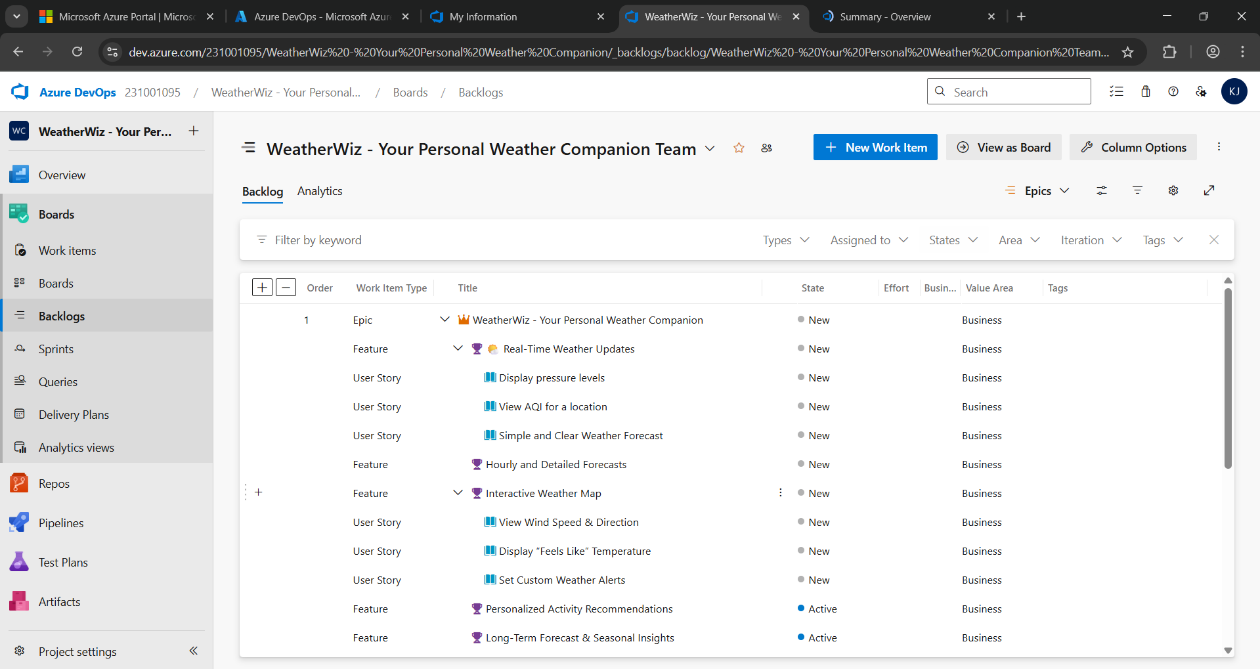
**EXP NO: 3**

**Aim:**

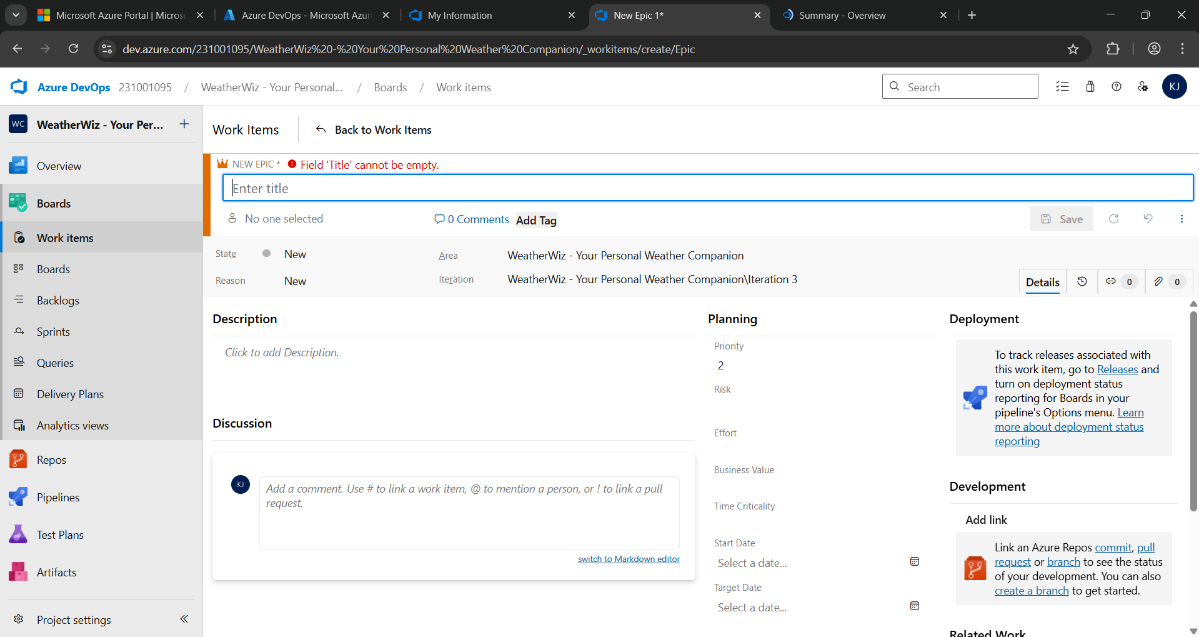
Create Epic, Features, User Stories, Task

To learn about how to create epics, user story, features, backlogs for your assigned project.

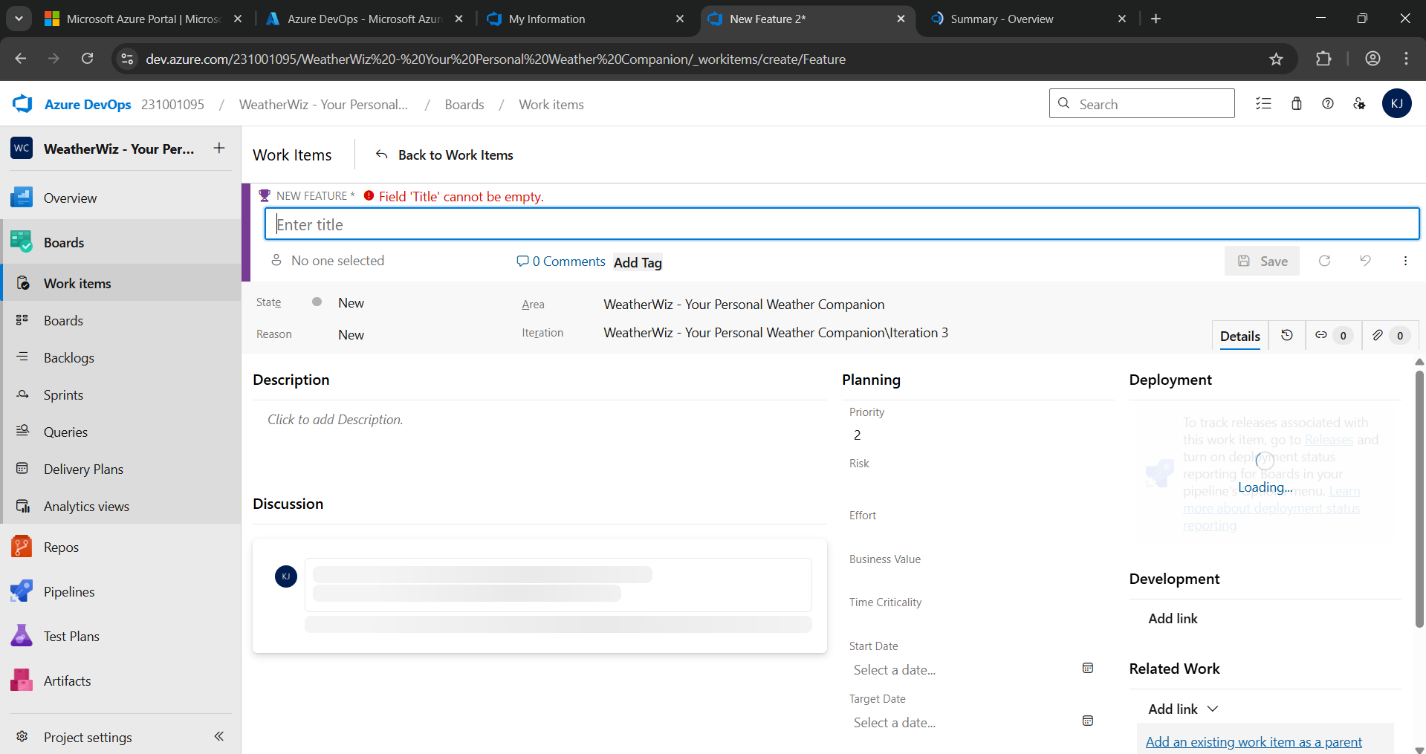
**Create Epic, Features, User Stories, Task**

****

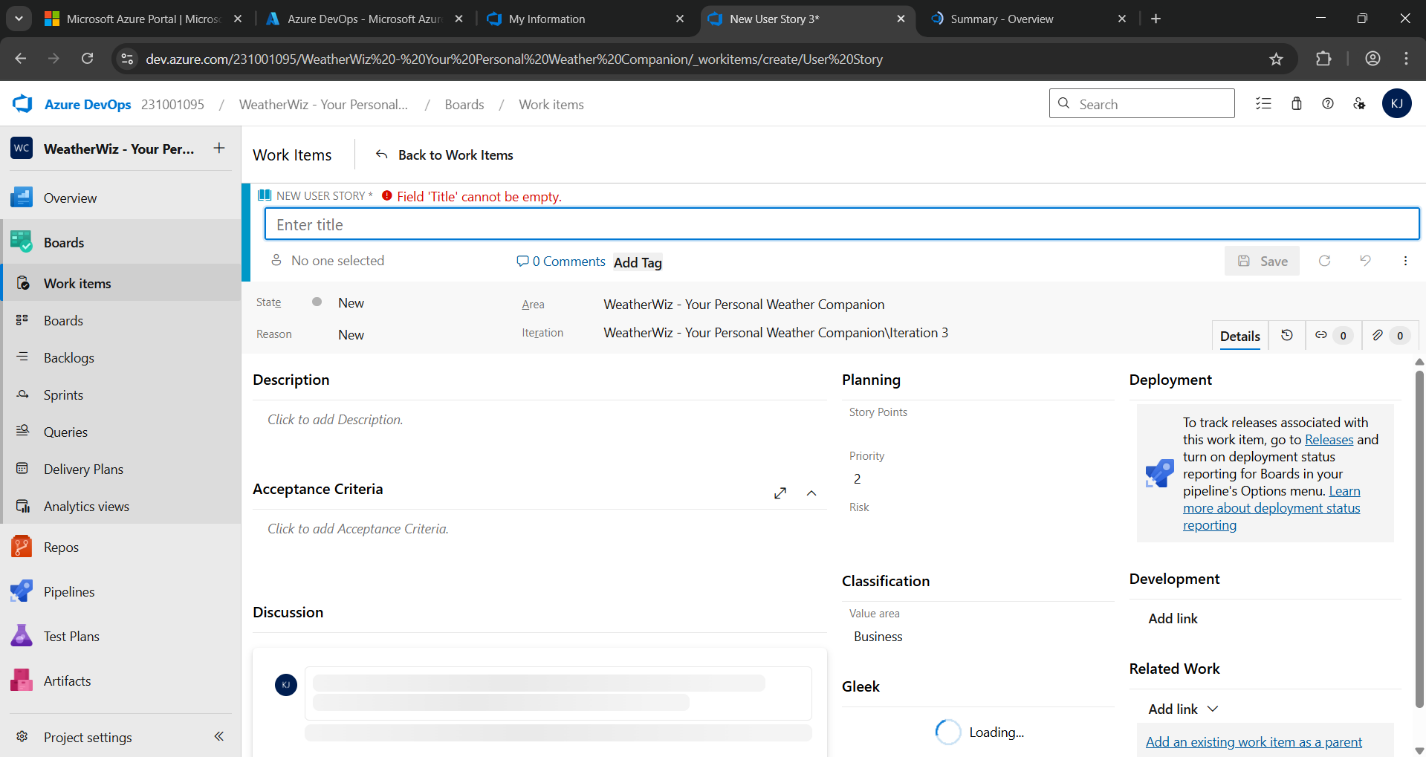
1. **Fill in Epics**

****

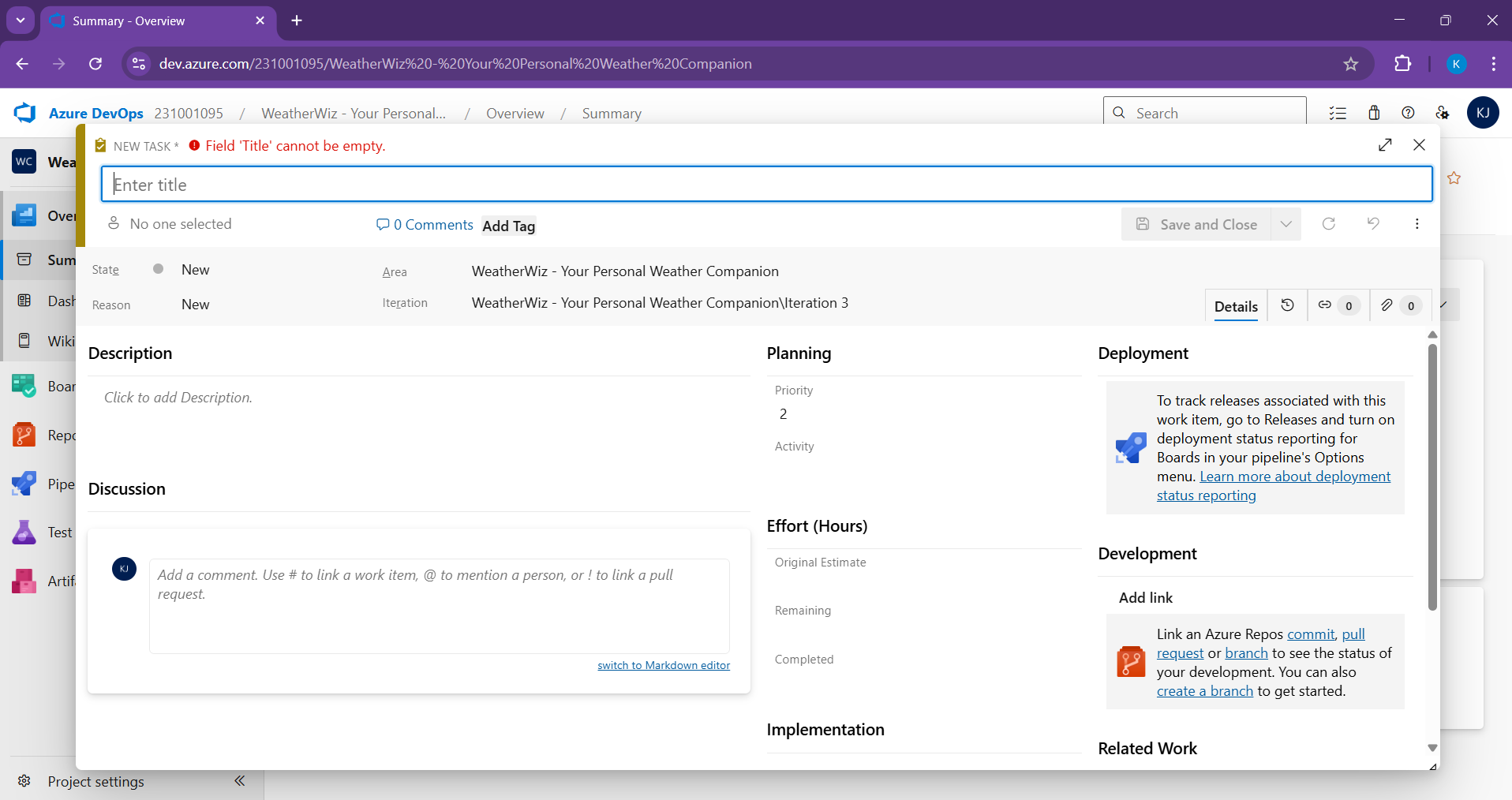
1. **Fill in Features**

****

1. **Fill in User Story Details**

****

**4. Fill in Task Details**

****

**Result:**

Thus, the creation of epics, features, user story and task has been created successfully.

**SPRINT PLANNING**

**EXP NO: 4**

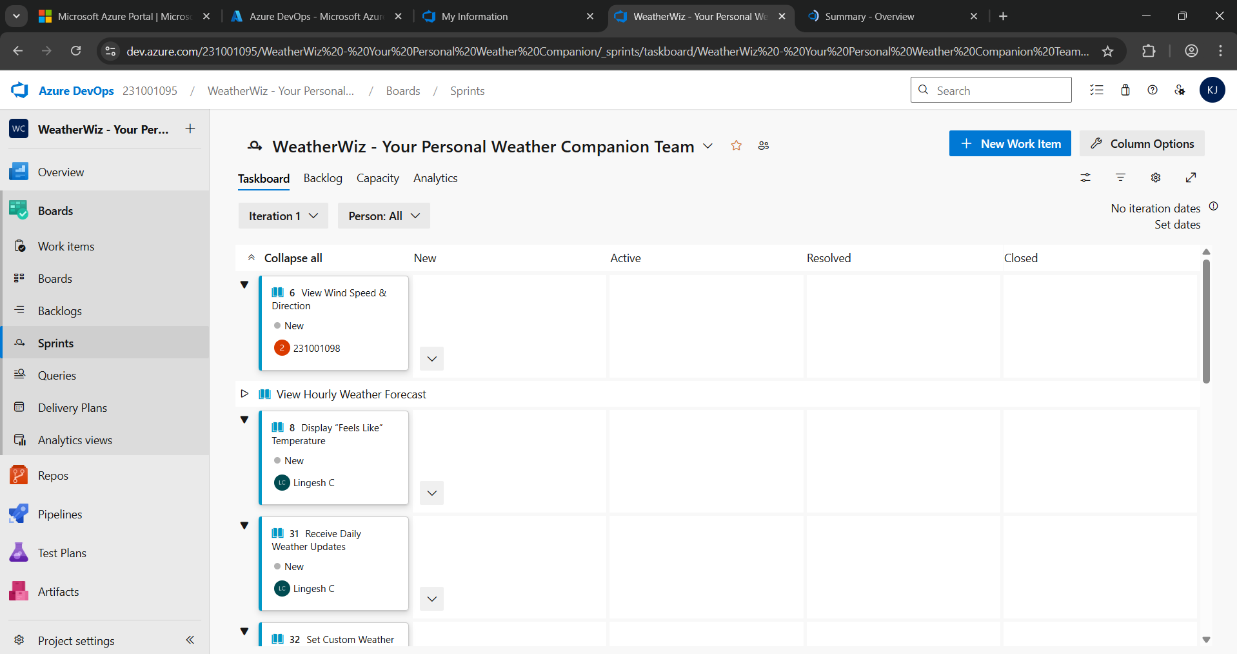
**Aim:**

Create Epic, Features, User Stories, Task

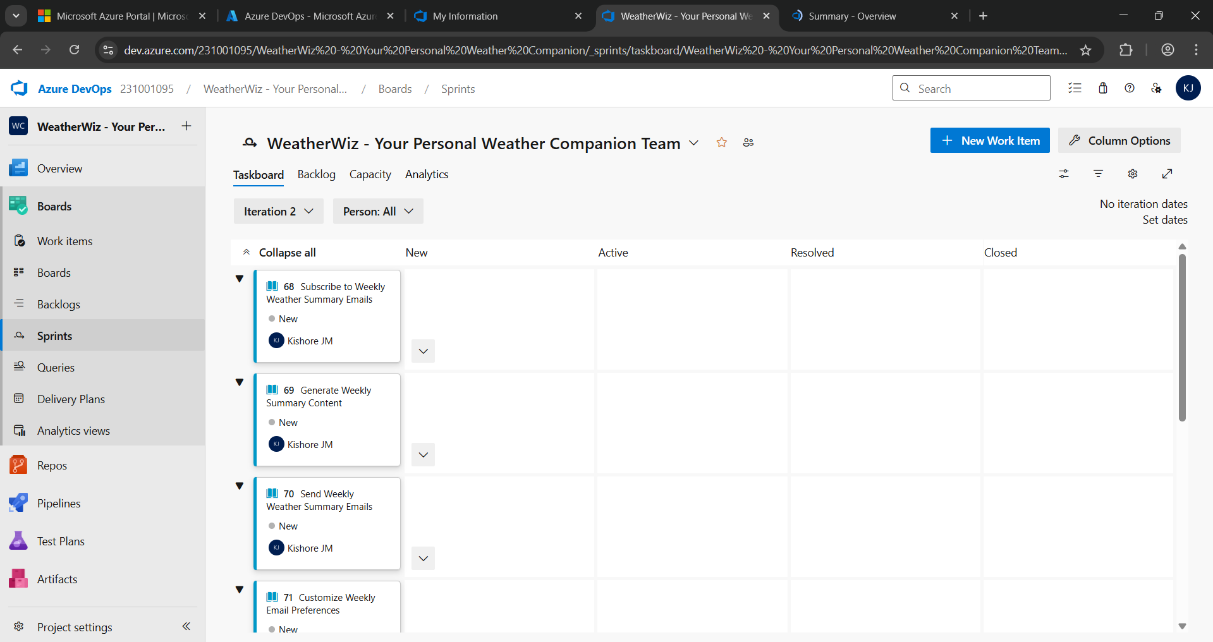
To assign user story to specific sprint for the WeatherWiz - Weather Companion Project.

**Sprint Planning :**

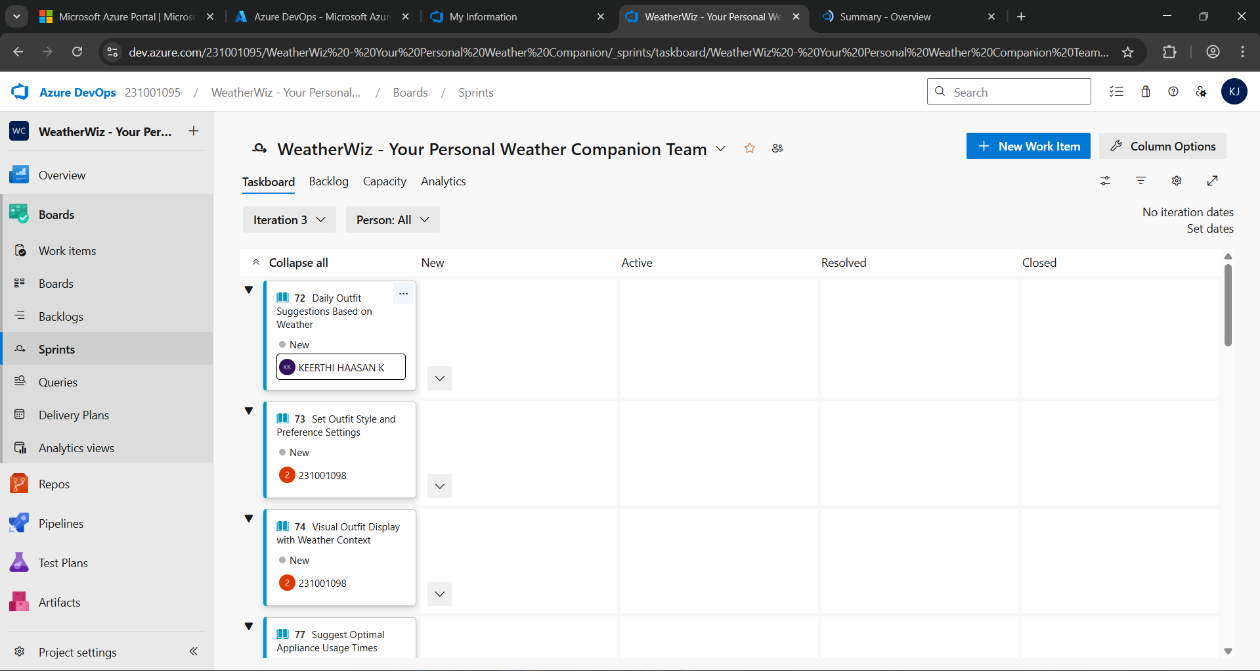
**Sprint 1**

****

**Sprint 2**

****

**Sprint 3**

****

**Result:**

The Sprints are created for the WeatherWiz Project.

**POKER ESTIMATION**

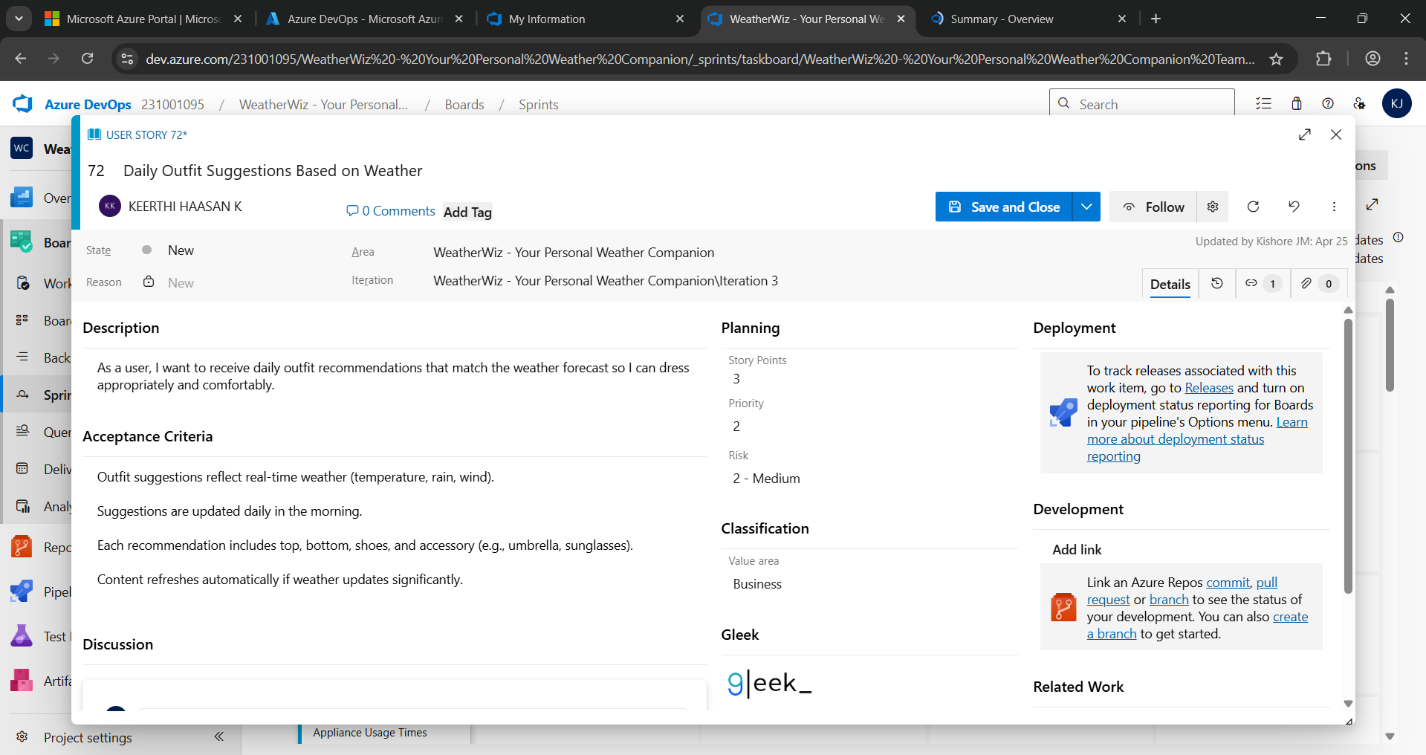
**EXP NO: 5**

**Aim:**

Create Epic, Features, User Stories, Task

Create Poker Estimation for the user stories - WeatherWiz - Weather Companion Project.

**Poker Estimation**

****

**Result:**

The Estimation/Story Points is created for the project using Poker Estimation.

**DESIGNING CLASS AND SEQUENCE DIAGRAMS FOR PROJECT ARCHITECTURE**

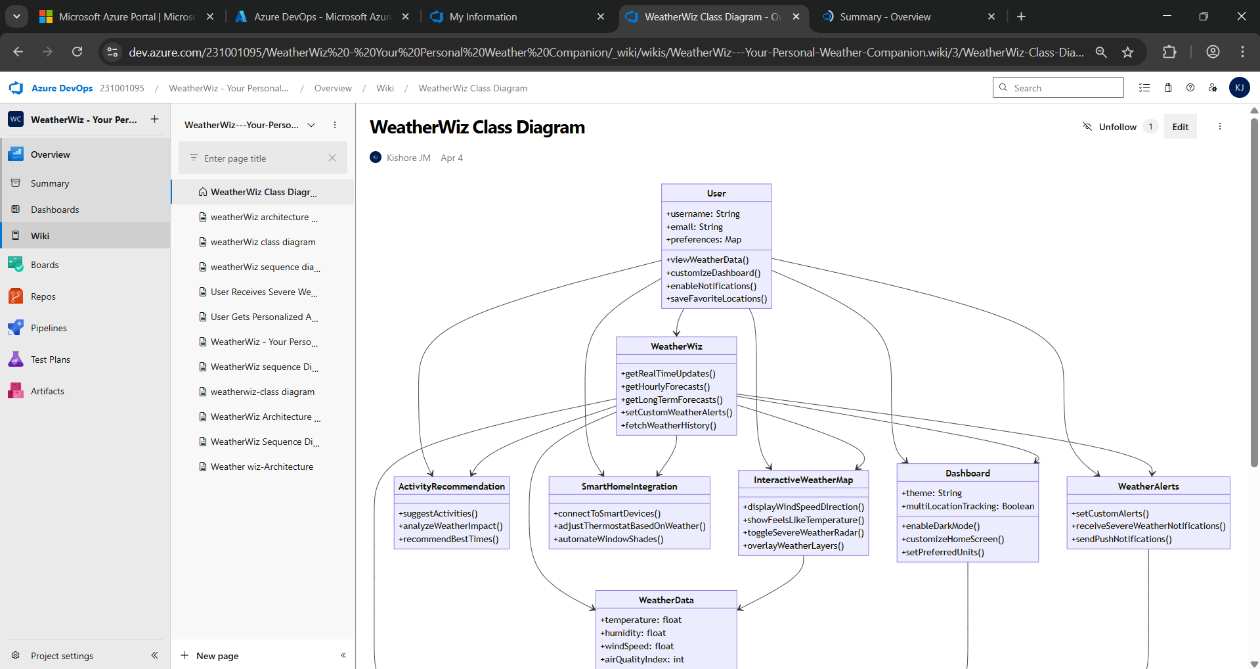
**EXP NO: 6**

**Aim:**

Create Epic, Features, User Stories, Task

To Design a Class Diagram and Sequence Diagram for the given Project.

**6A. Class Diagram**

****

**6B. Sequence Diagram**

****

**Result:**

The Class Diagram and Sequence Diagram is designed Successfully for the Music Playlist Batch

Creator.

**DESIGNING ARCHITECTURAL AND ER DIAGRAMS FOR PROJECT STRUCTURE**

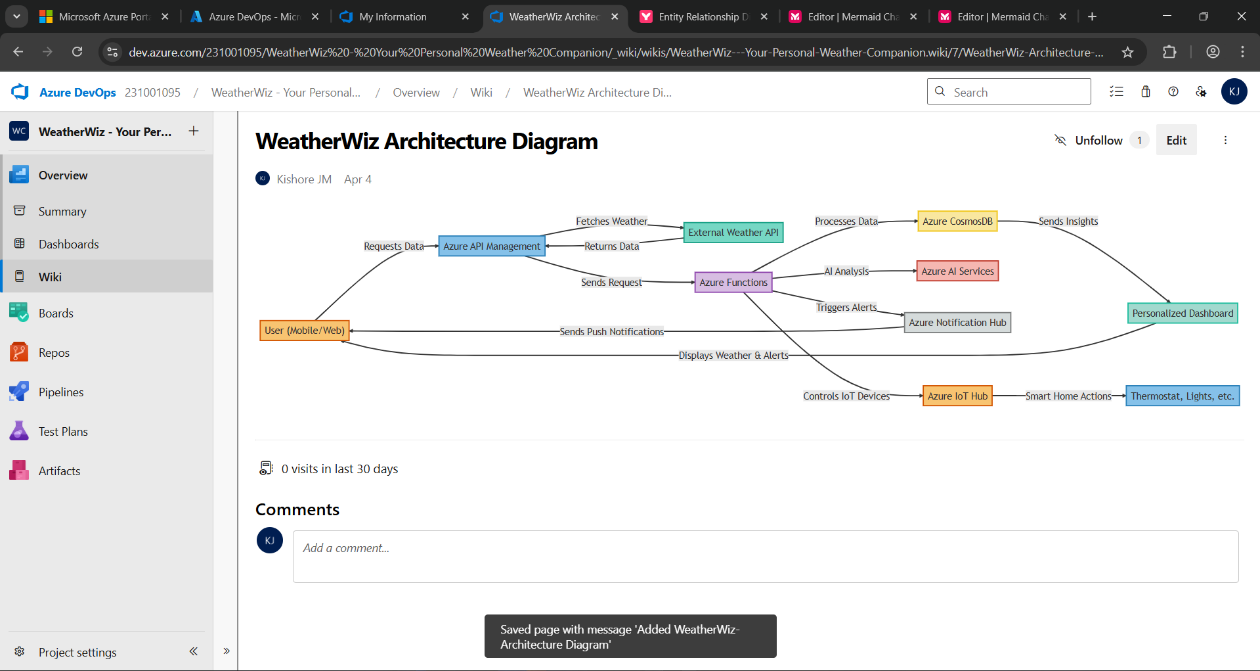
**EXP NO: 7**

**Aim:**

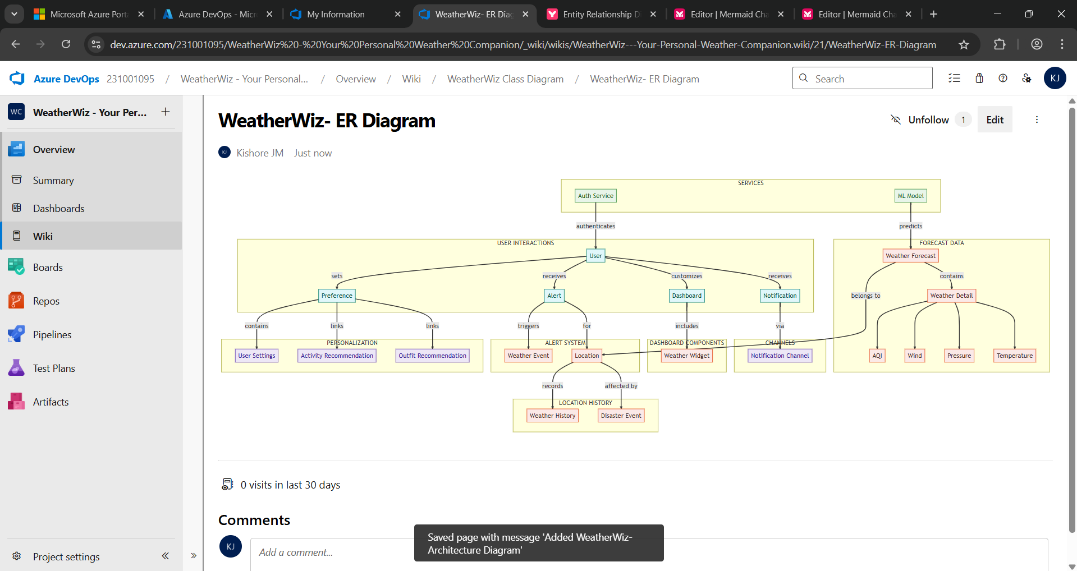
Create Epic, Features, User Stories, Task

To Design an Architectural Diagram and ER Diagram for the given Project.

**7A. Architectural Diagram**

****

**7B.ER Diagram**

****

**Result:**

The Architecture Diagram and ER Diagram is designed Successfully for the WeatherWiz - Weather Companion.

**TESTING – TEST PLANS AND TEST CASES**

**EXP NO: 8**

**Aim:**

Create Epic, Features, User Stories, Task

Test Plans and Test Case and write two test cases for at least five user stories showcasing the

happy path and error scenarios in azure DevOps platform.

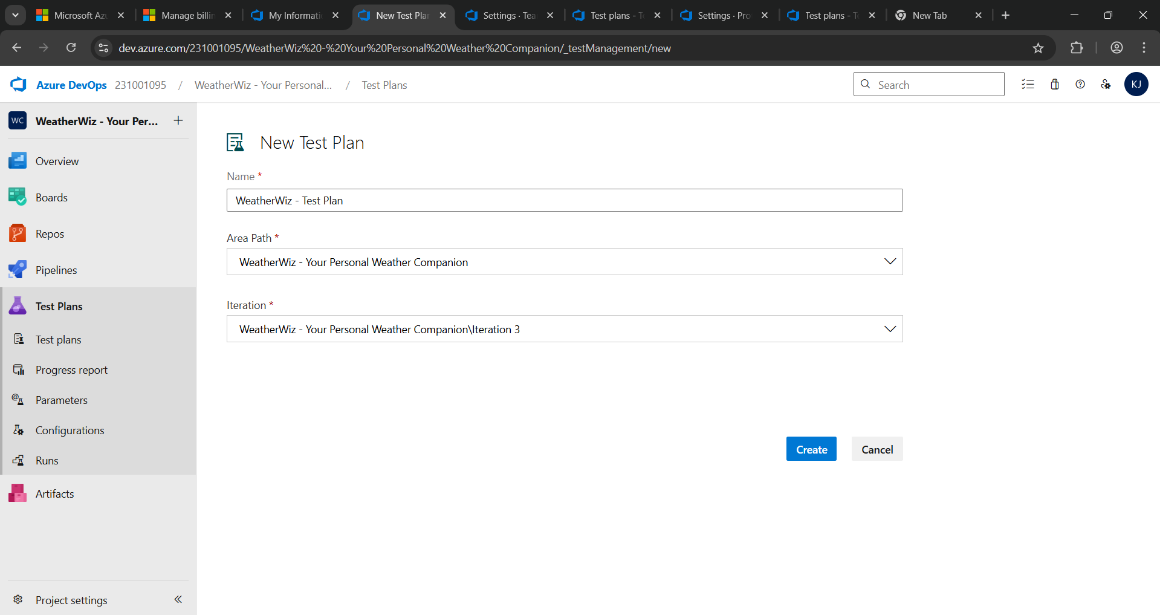
**Test Planning and Test Case Test Case Design Procedure**

1. **Understand Core Features of the Application**
   * View Current Weather by Location
   * Hourly and Weekly Forecasts
   * Severe Weather Alerts
   * Add and Manage Favorite Locations
   * View Weather by Categories (e.g., temperature, wind, precipitation)
   * Weather-Based Recommendations (e.g., clothing, travel tips)
2. **Define User Interactions**

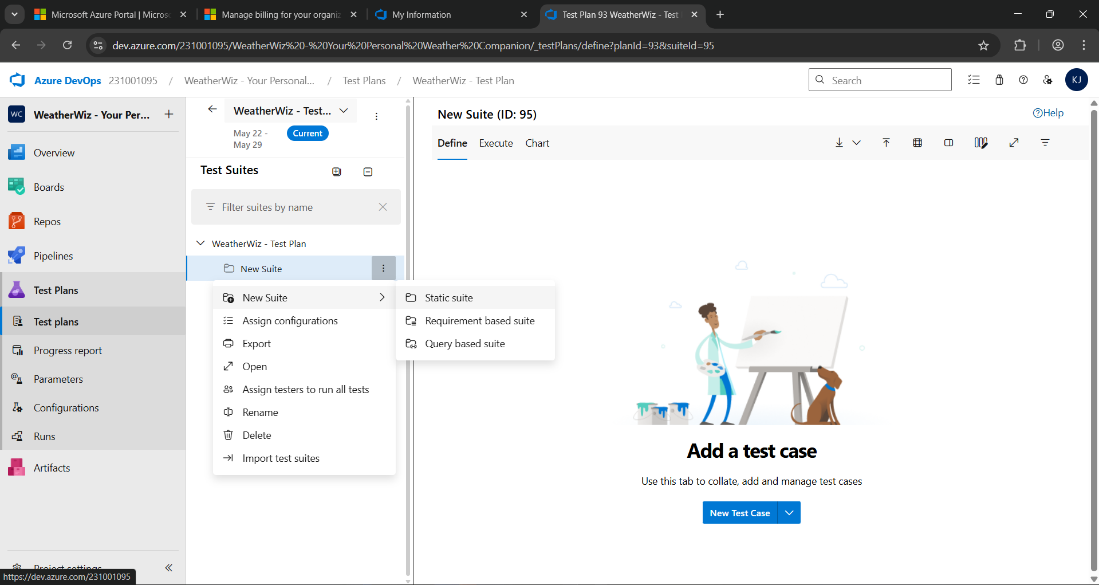
Each test case simulates a real user behavior such as :

* + - * Logging in
      * Searching for a city's weather
      * Adding a location to favorites
      * Viewing extended forecast
      * Receiving alerts or recommendations

1. **Design Happy Path Test Cases**
   * Focused on validating that all features function as expected under normal conditions.
   * Example: User logs in successfully, adds a city to favorites, or Receives accurate 7-day forecast.
2. **Design Error Path Test Cases**
   * Simulate negative or unexpected scenarios to test robustness and error handling.
   * Example: Login fails with invalid credentials, Weather data fetch fails due to no internet, Invalid city name entered in search.
3. **Break Down Steps and Expected Results**
   * Each test case contains step-by-step actions and a corresponding expected outcome.
   * Ensures clarity for both testers and automation scripts.
4. **Use Clear Naming and IDs**
   * Test cases are named clearly (e.g., TC01 – Successful Login, TC08 – Add Favorite Location).
   * Helps in quick identification and linking to user stories or features.
5. **Separate Test Suites**
   * Grouped test cases based on functionality (e.g., Login, Weather Data Retrieval, Forecast and Alerts, Recommendations and Tips).
6. **Prioritize and Review**
   * Critical user actions are marked high-priority.
   * Reviewed for completeness and traceability against feature requirements.
7. **New test plan**

****

1. **Test suite**

****

1. **Test case**

Give two test cases for at least five user stories showcasing the happy path and error scenarios in azure DevOps platform.

WeatherWiz - Weather Companion– Test Plans

**USER STORIES**

* + As a user, I want to get real-time weather updates (ID: 201).
  + As a user, I want to see weather by location (ID: 202).
  + As a user, I want to save favorite cities for quick access (ID: 203).
  + As a user, I want weather alerts (e.g., rain, storm) (ID: 204).
  + As a user, I want to customize temperature units (Celsius/Fahrenheit) (ID: 205).

**Test Suites**

**Test Suit: TS01 - Real-Time Weather Updates (ID: 301)**

**1. TC01 – Real-Time Weather Display**

* + **Action:**
    - Open app.
    - Enable device location
    - Wait for weather to load
  + **Expected Results:**
    - Current temperature.
    - Weather condition.
    - Icon display correctly.
  + **Type**: Happy Path

1. **TC02 – No Internet Weather Error**
   * **Action:**
     + Turn off Wi-Fi/Mobile Data.
     + Open the app.
   * **Expected Results:**
     + Message: “Cannot fetch weather. Please check your internet connection.”
   * **Type:** Error Path

**Test Suit: TS02 - Location-Based Weather Search (ID: 302)**

1. **TC03 – Search Weather for a City**
   * **Action:**
     + Tap “Search”.
     + Enter “Delhi”.
   * **Expected Results:**
     + Weather info for Delhi is displayed.
   * **Type:** Happy Path
2. **TC04 – Invalid City Search**
   * **Action:**
     + Tap “Search”.
     + Enter "%%%$$$".
   * **Expected Results:**
     + Error: “City not found. Please enter a valid city.”
   * **Type:** Error Path

**Test Suit: TS03 - Manage Favorite Cities (ID: 303)**

1. **TC05 – Add City to Favorites**
   * **Action:**
     + Search for a city.
     + Tap “★” to add it to favorites.
   * **Expected Results:**
     + City added to “Favorites” list.
   * **Type:** Happy Path
2. **TC06 – Add Duplicate Favorite**
   * **Action:**
     + Try adding same city again.
   * **Expected Results:**
     + Message: “City already exists in favorites”.
   * **Type:** Error Path

**Test Suit: TS04 - Weather Alerts (ID: 304)**

1. **TC07 – Show Rain Alert**
   * **Action:**
     + App detects rain in forecast.
   * **Expected Results:**
     + User receives in-app or push alert: “Rain expected today”.
   * **Type:** Happy Path
2. **TC08 – Alert Fails to Trigger**
   * **Action:**
     + Alert is enabled.
     + Forecast has storm.
     + No alert shown.
   * **Expected Results:**
     + Fallback message: “Alert system unavailable. Please check manually.”
   * **Type:** Error Path

**Test Suit: TS05 - Unit Preferences (ID: 305)**

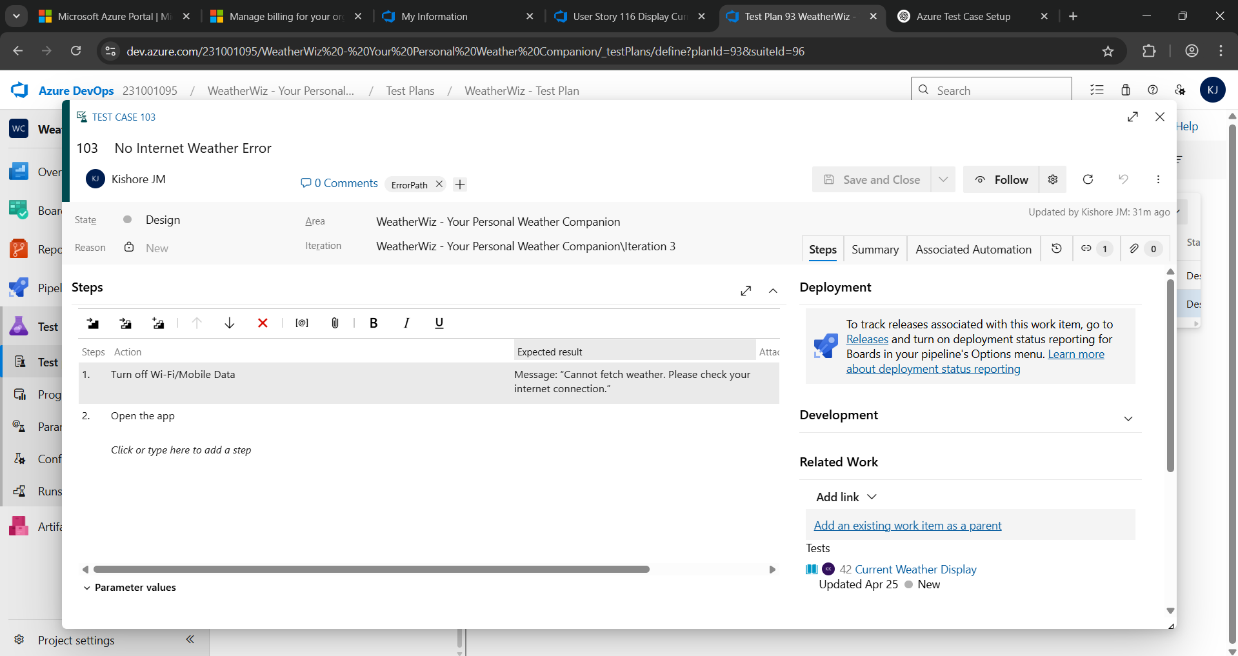
1. **TC09 – Switch to Fahrenheit**
   * **Action:**
     + Go to Settings.
     + Select “Fahrenheit”.
   * **Expected Results:**
     + Weather values are updated to °F.
   * **Type:** Happy Path
2. **TC10 – Unit Switch Fails**
   * **Action:**
     + Go to Settings.
     + Tap on unit toggle.
     + No change occurs.
   * **Expected Results:**
     + Error: “Unable to update units. Please try again.”.
   * **Type:** Error Path

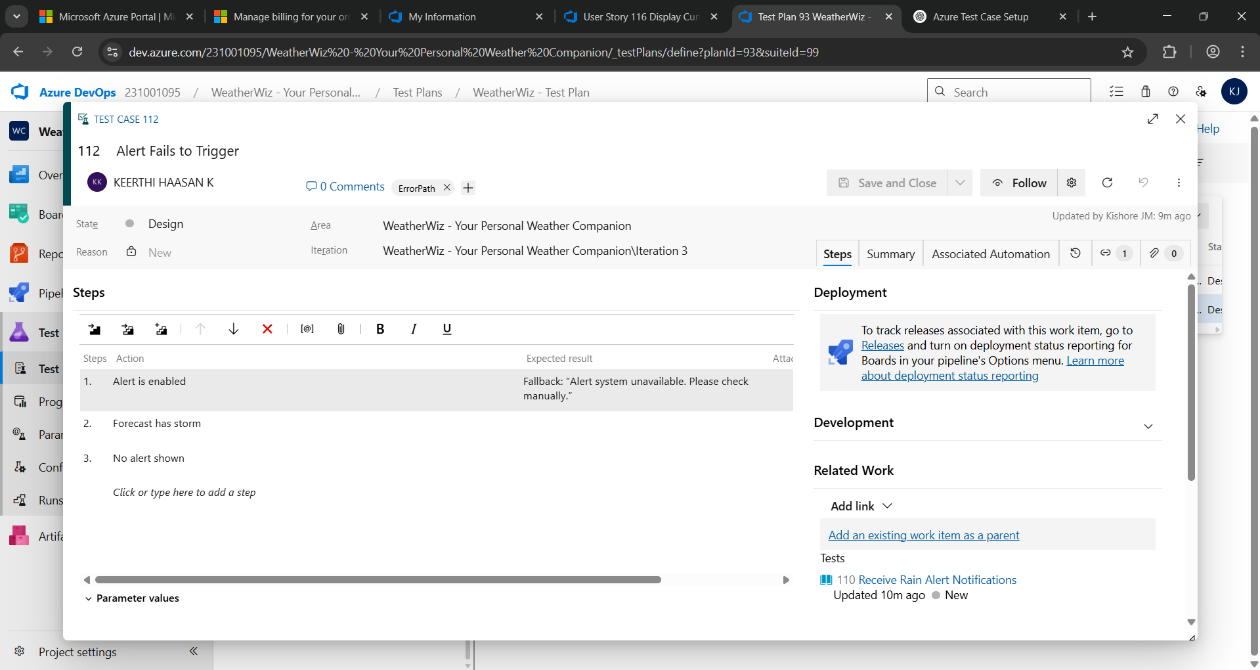
**Test Suit: TS06 - Weather Widget Integration (ID: 306)**

1. **TC11 – Display Weather on Home Widget**
   * **Action:**
     + Add WeatherWiz widget to the home screen.
     + Allow location access.
   * **Expected Results:**
     + Widget displays current temperature and condition (e.g., "26°C – Cloudy").
   * **Type:** Happy Path
2. **TC12 – Widget Fails Without Permission**
   * **Action:**
     + Add widget.
     + Deny location permission.
   * **Expected Results:**
     + Message on widget: “Enable location to display weather”.

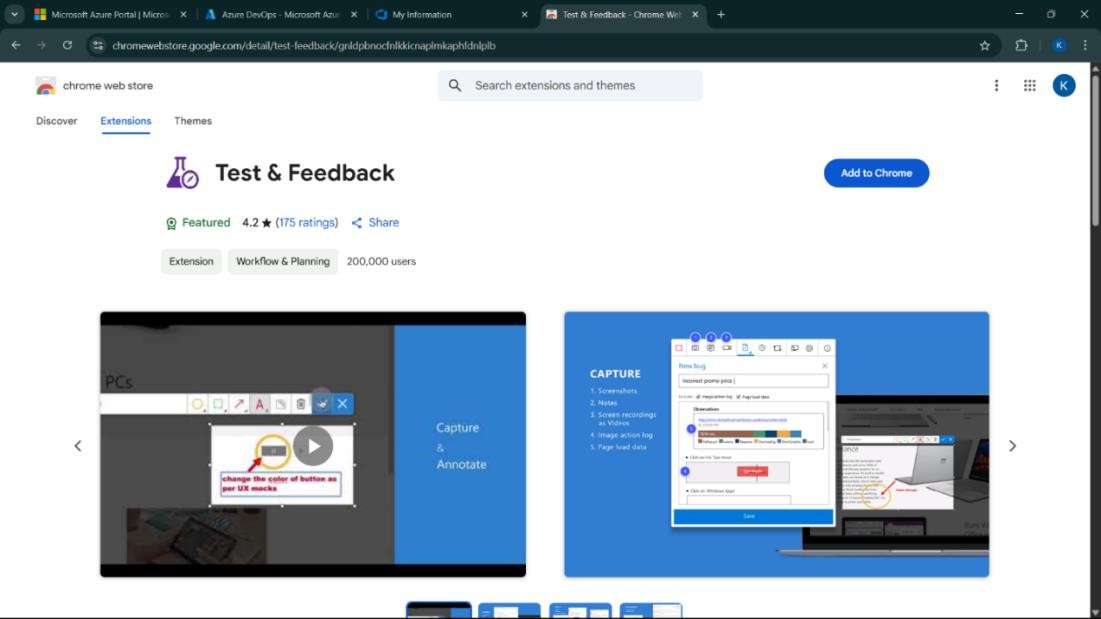
* **Type:** Error Path

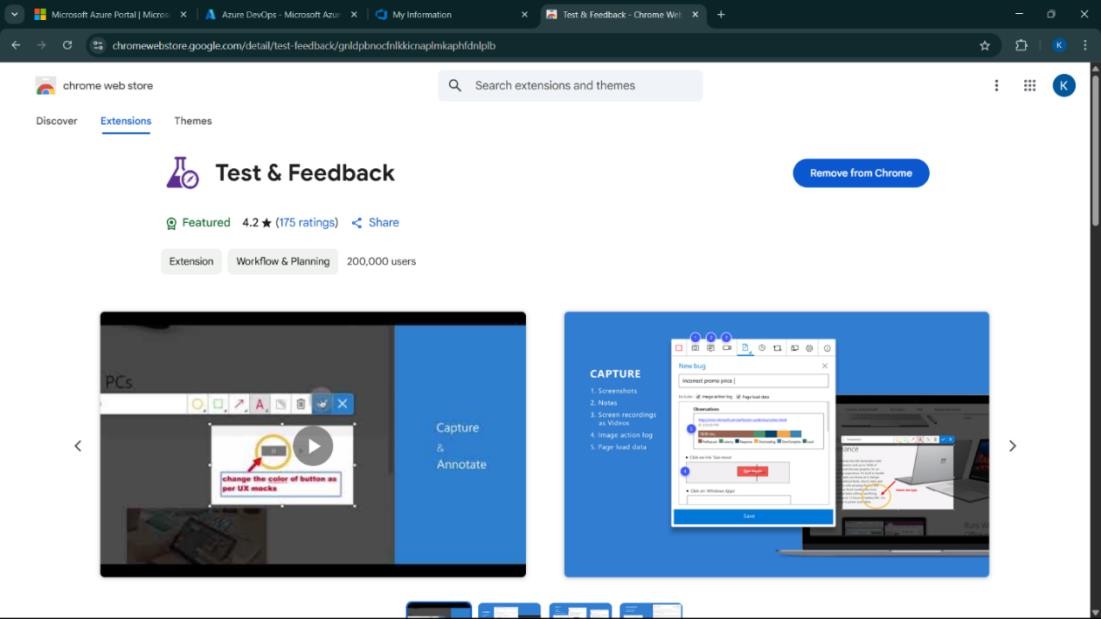
**Test Cases**

****

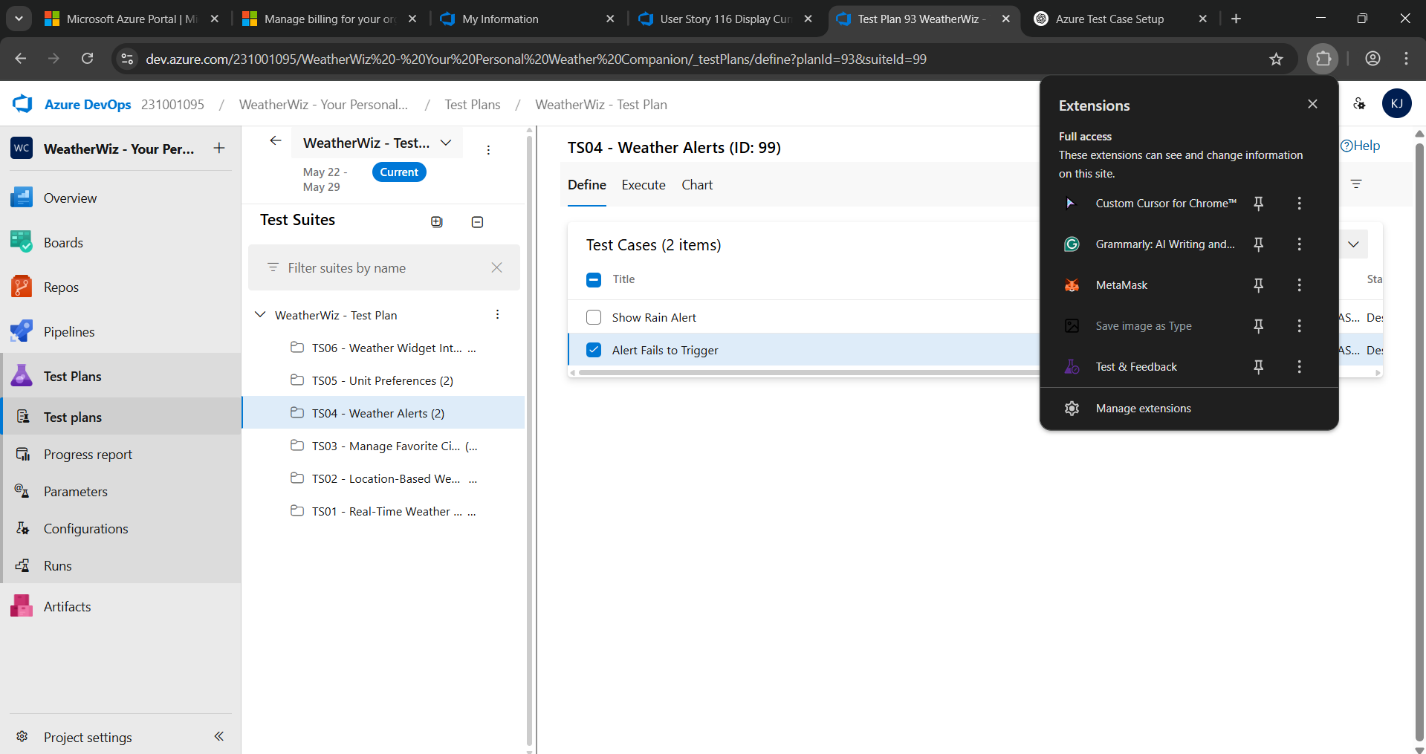


1. **Installation of test**

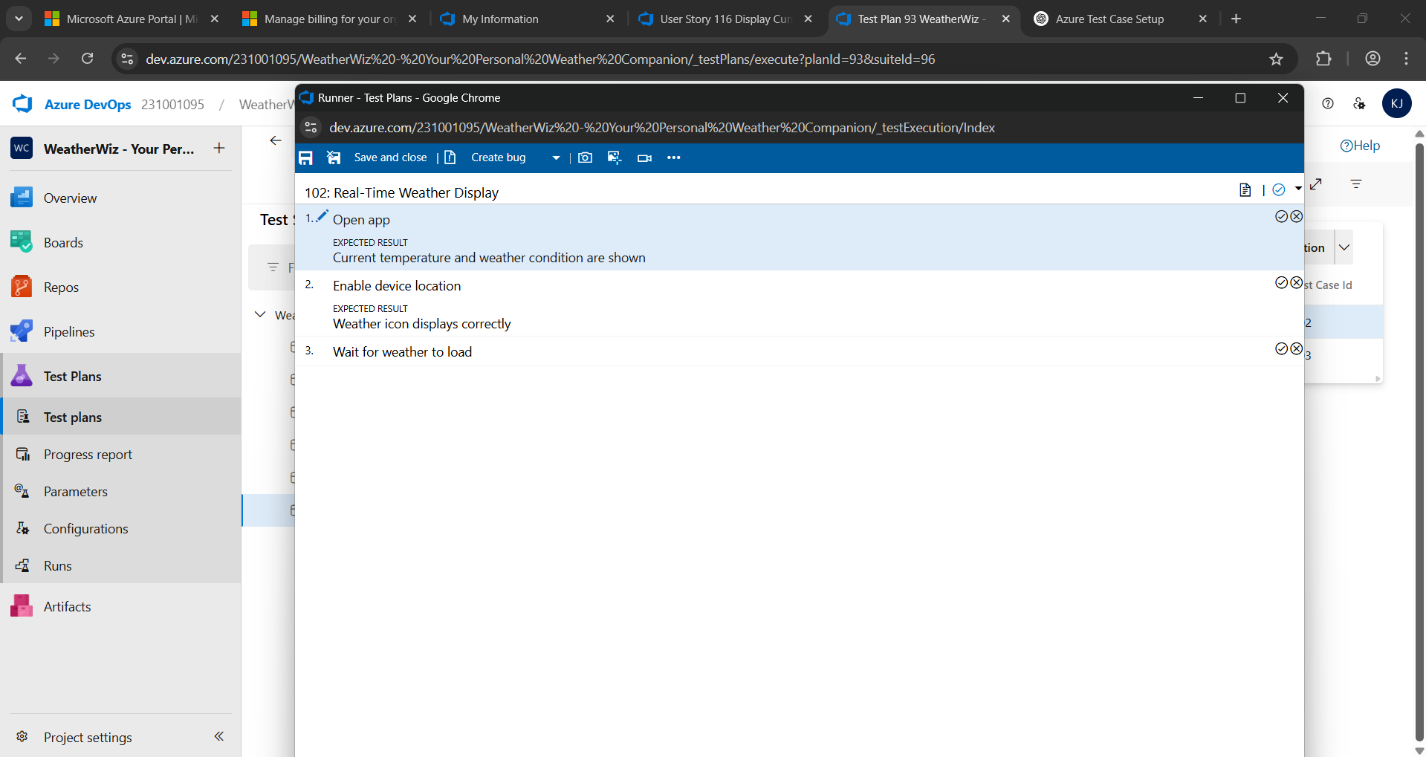
****



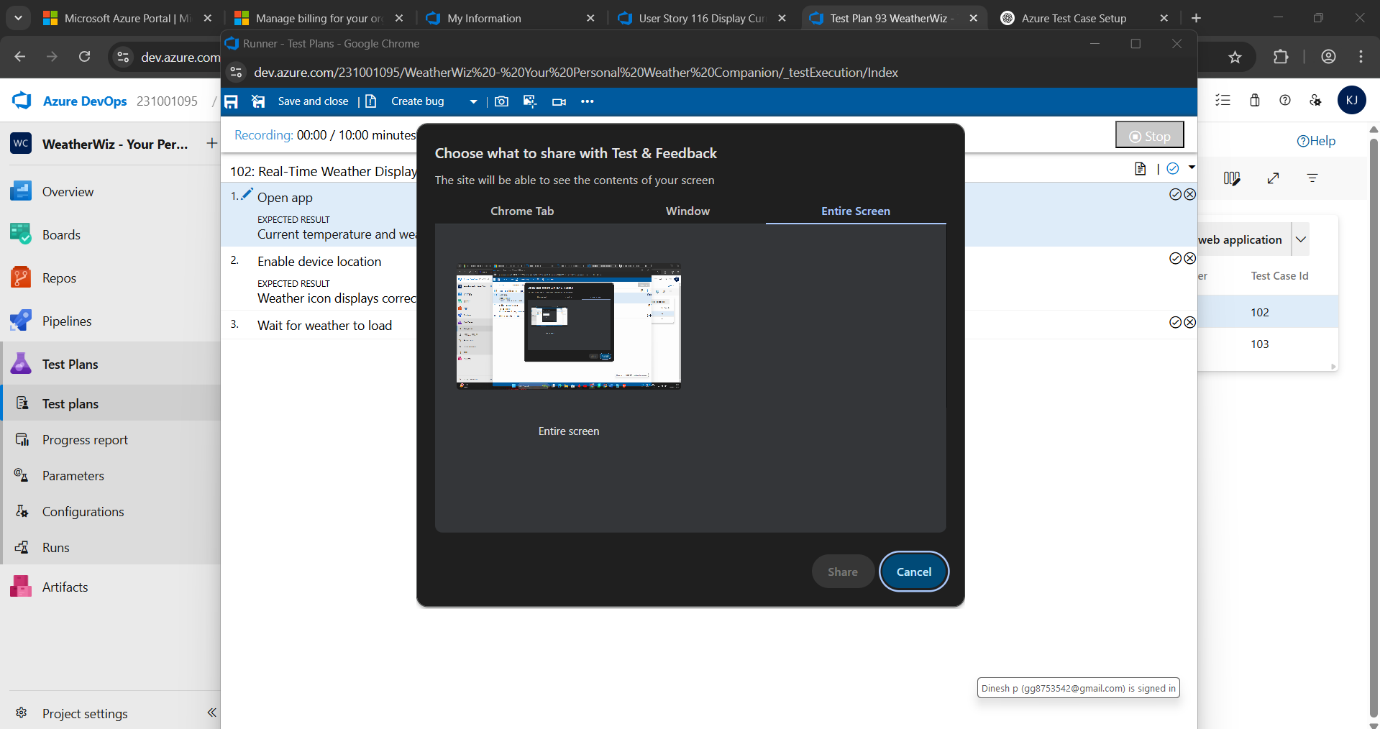
Test and feedback Showing it as an extension



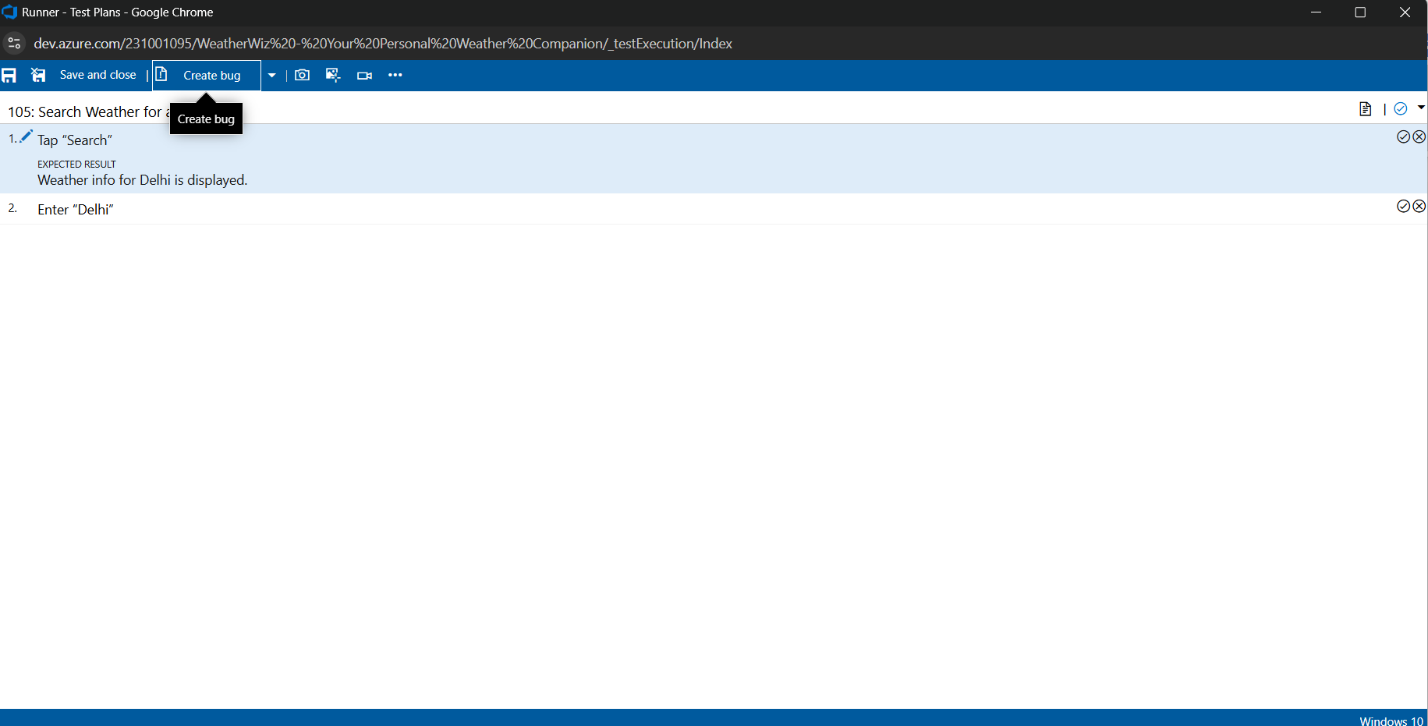
1. **Running the test cases**

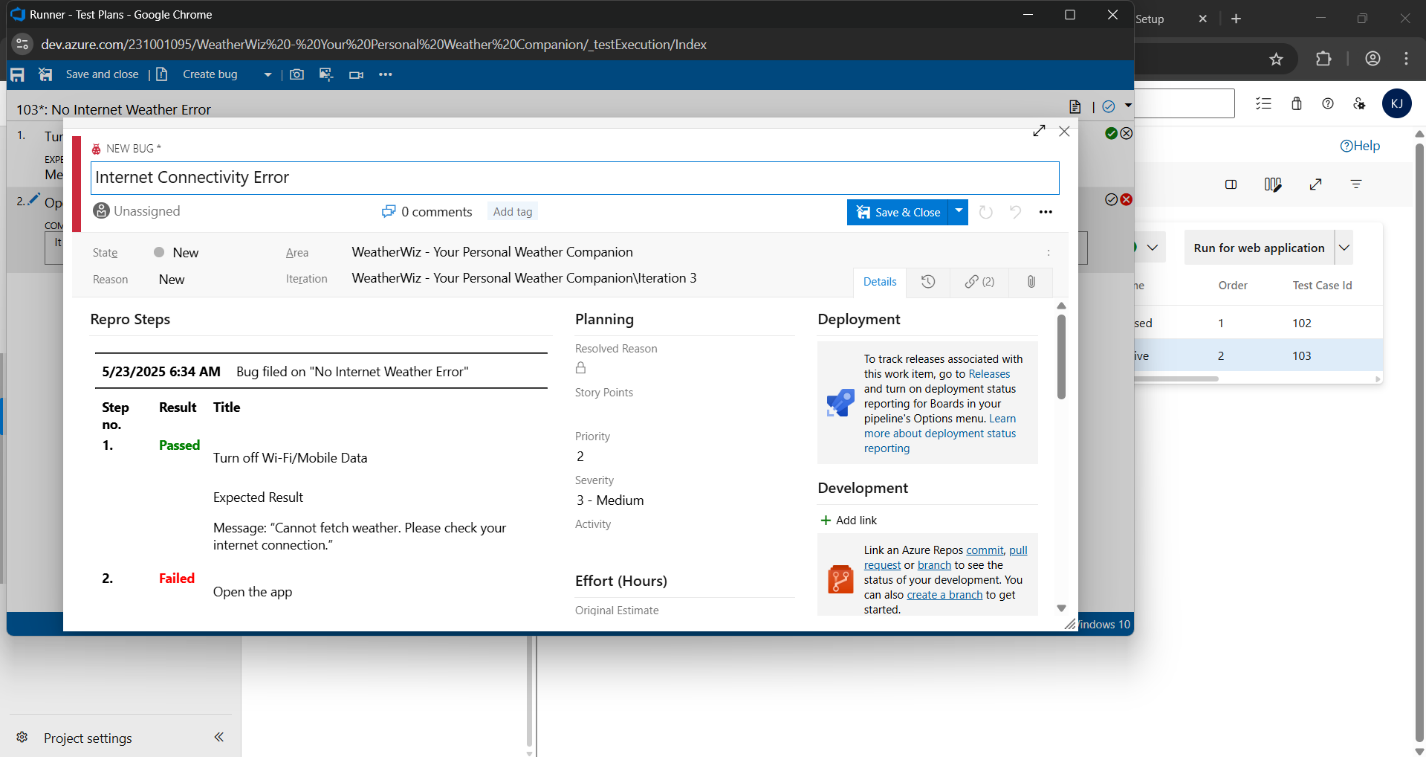
****

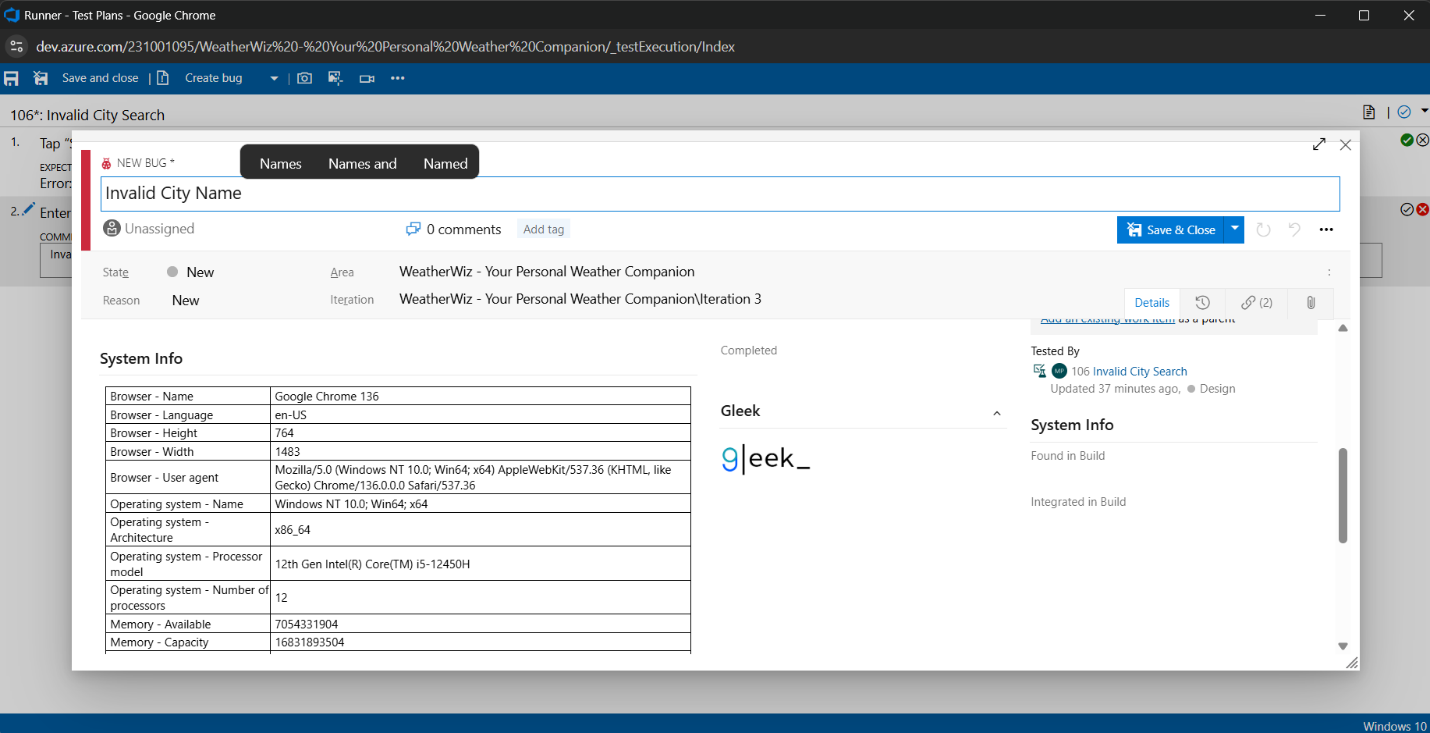
1. **Recording the test case**

****

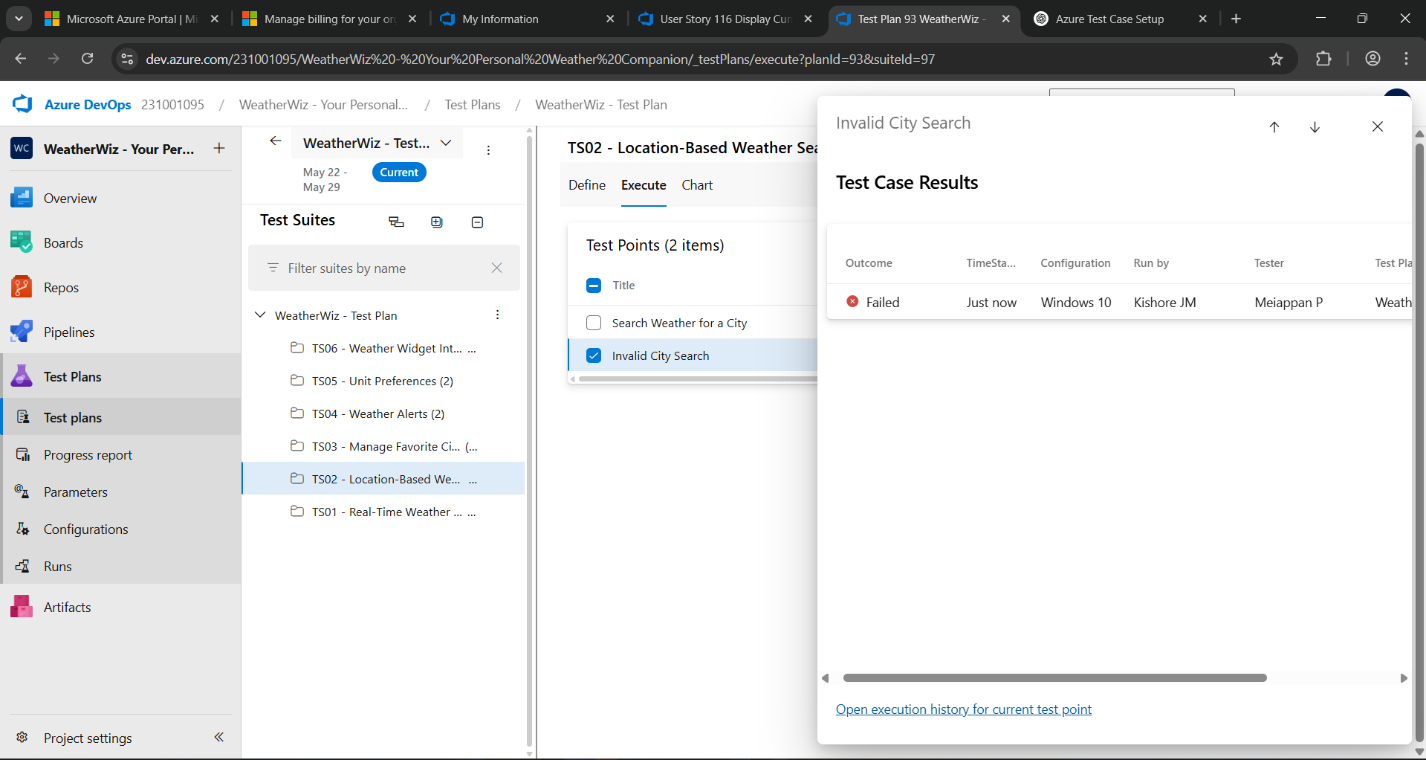
1. **Creating the bug**

****

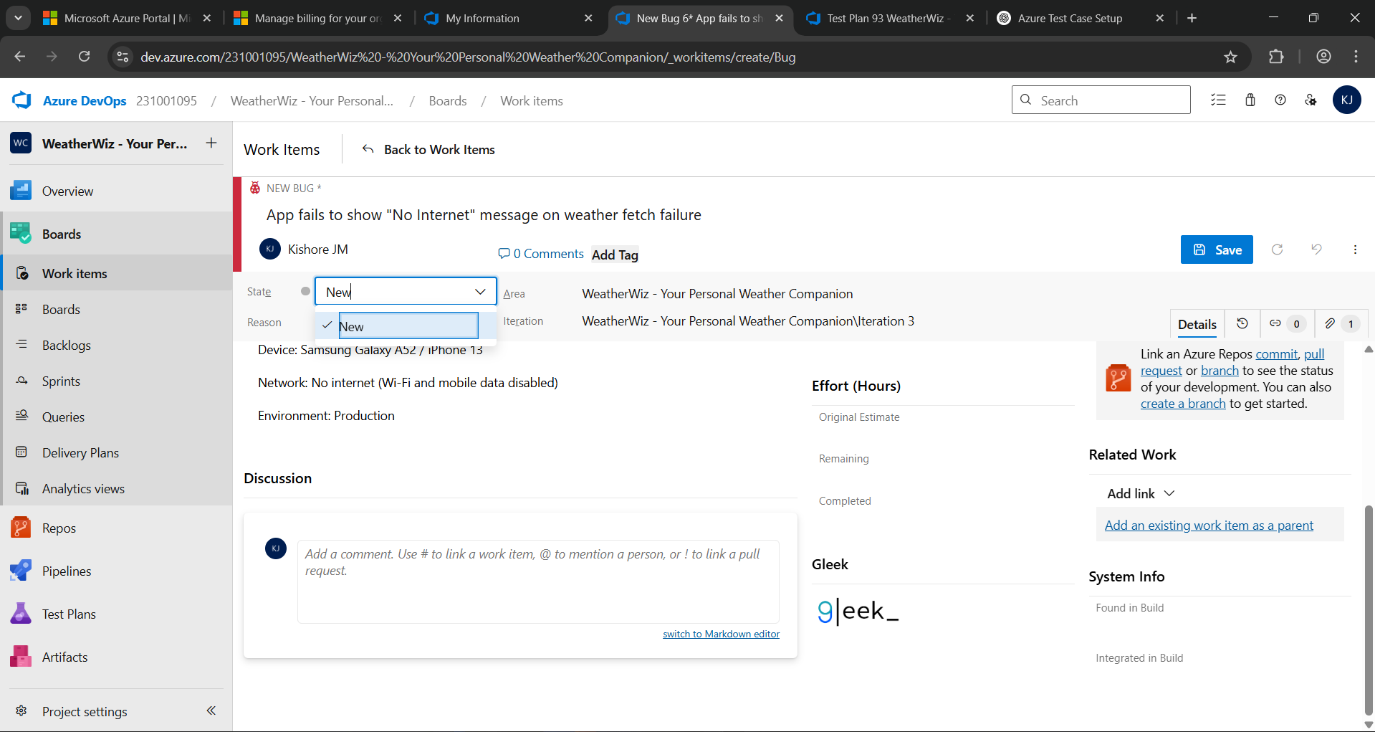


****

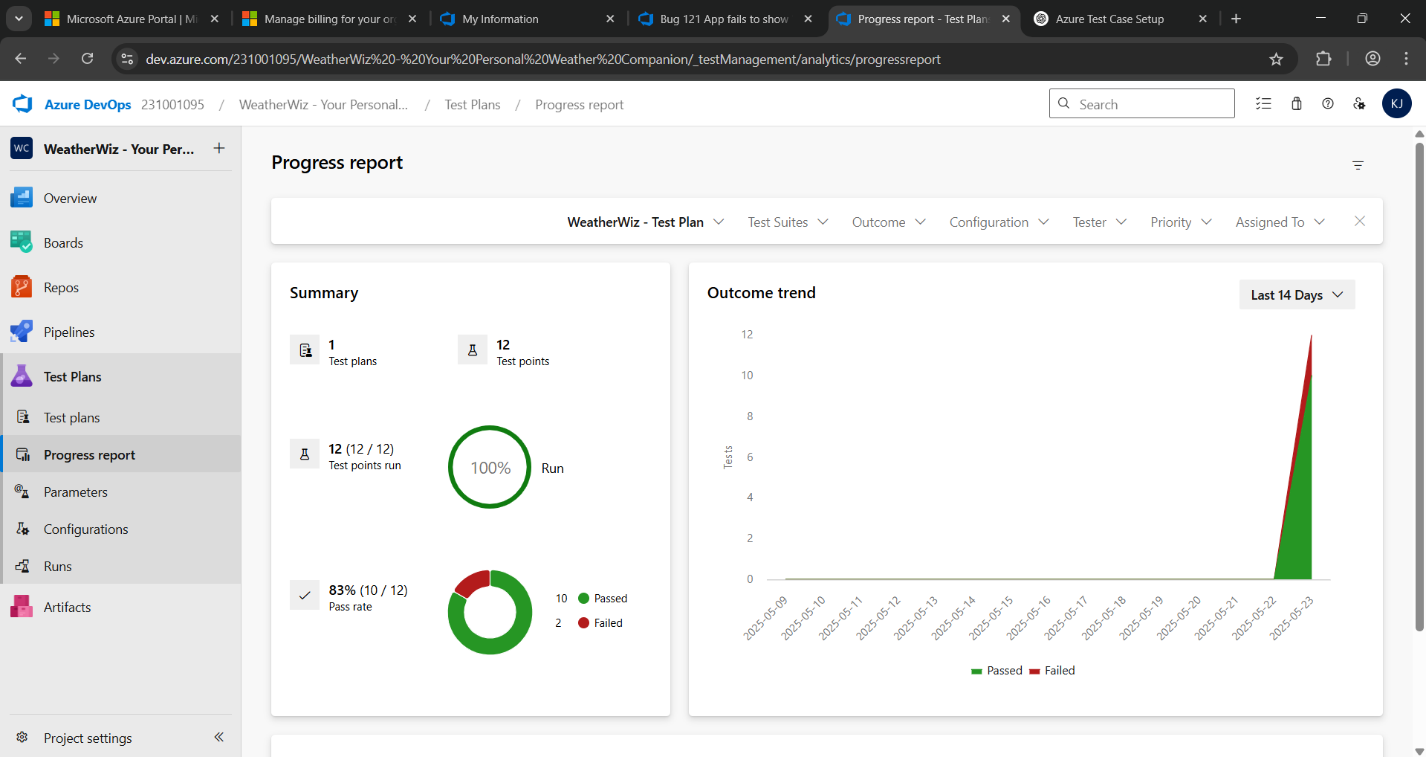
1. **Test case results**

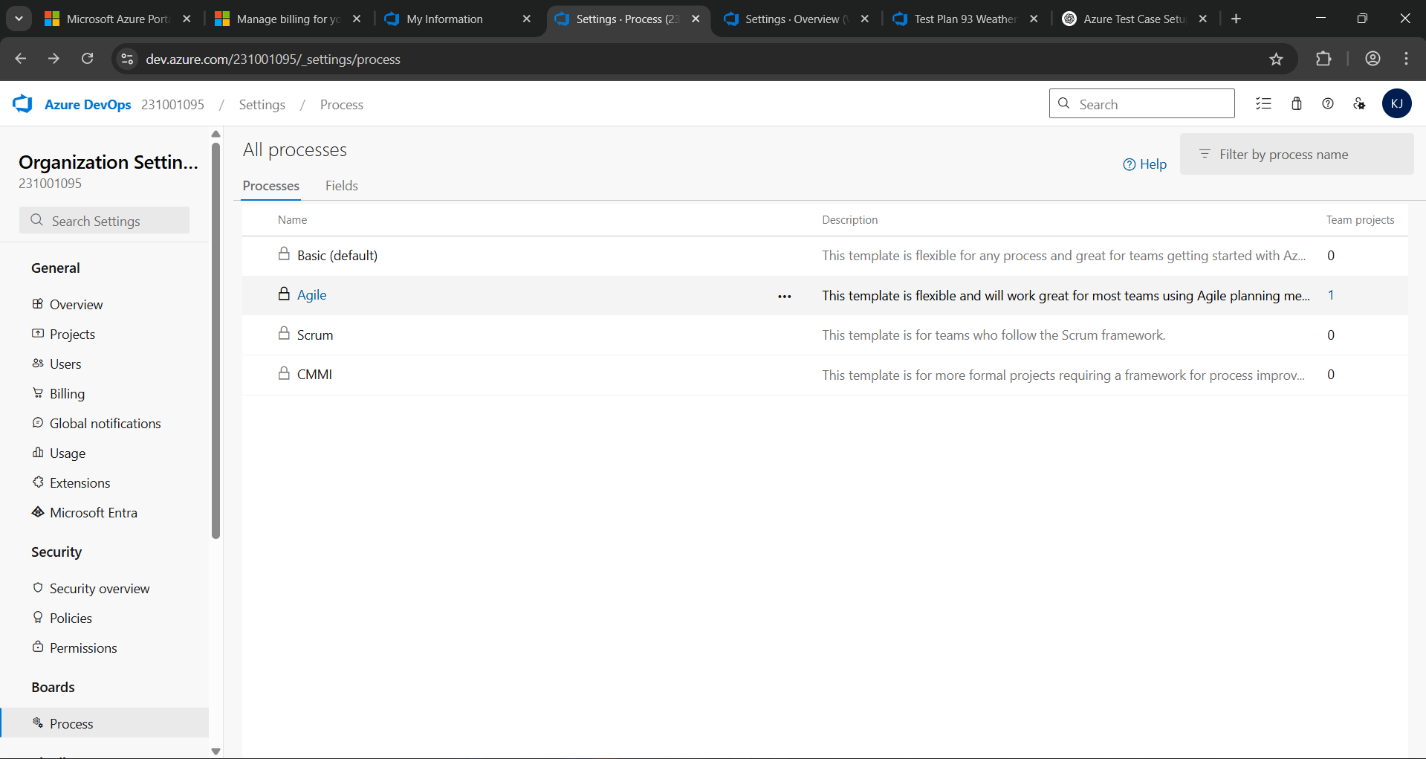
****

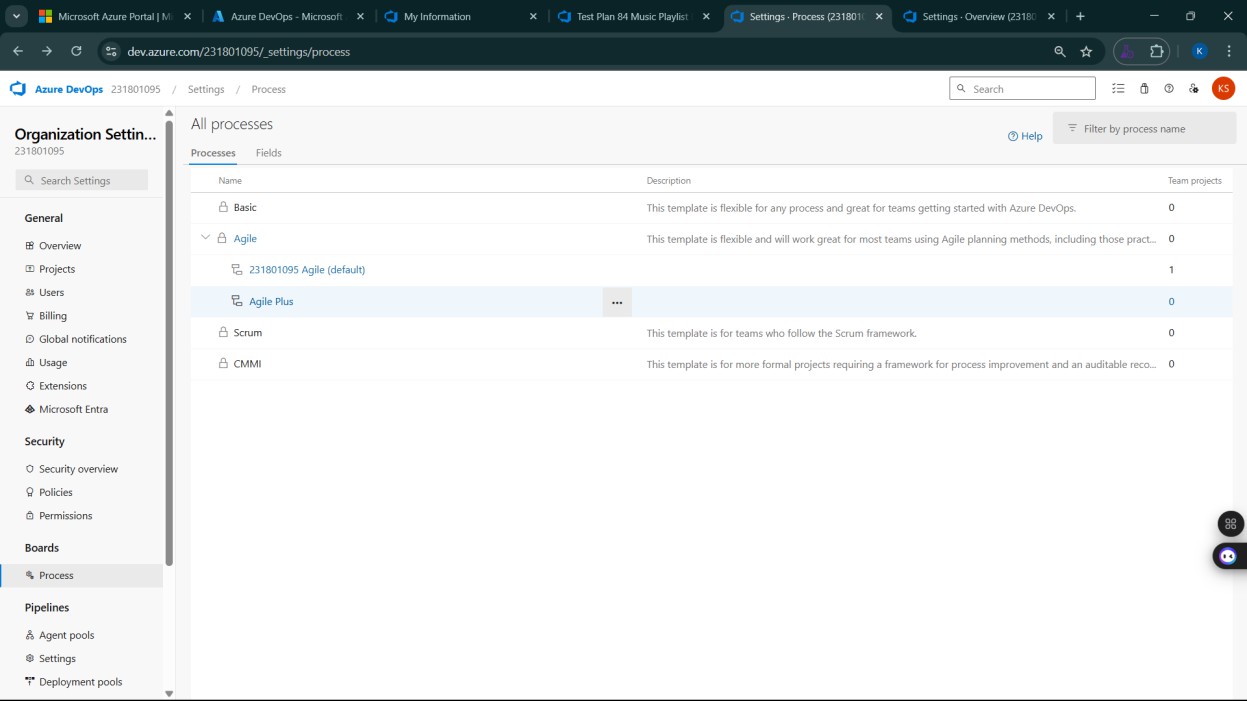
1. **Test report summary**

****

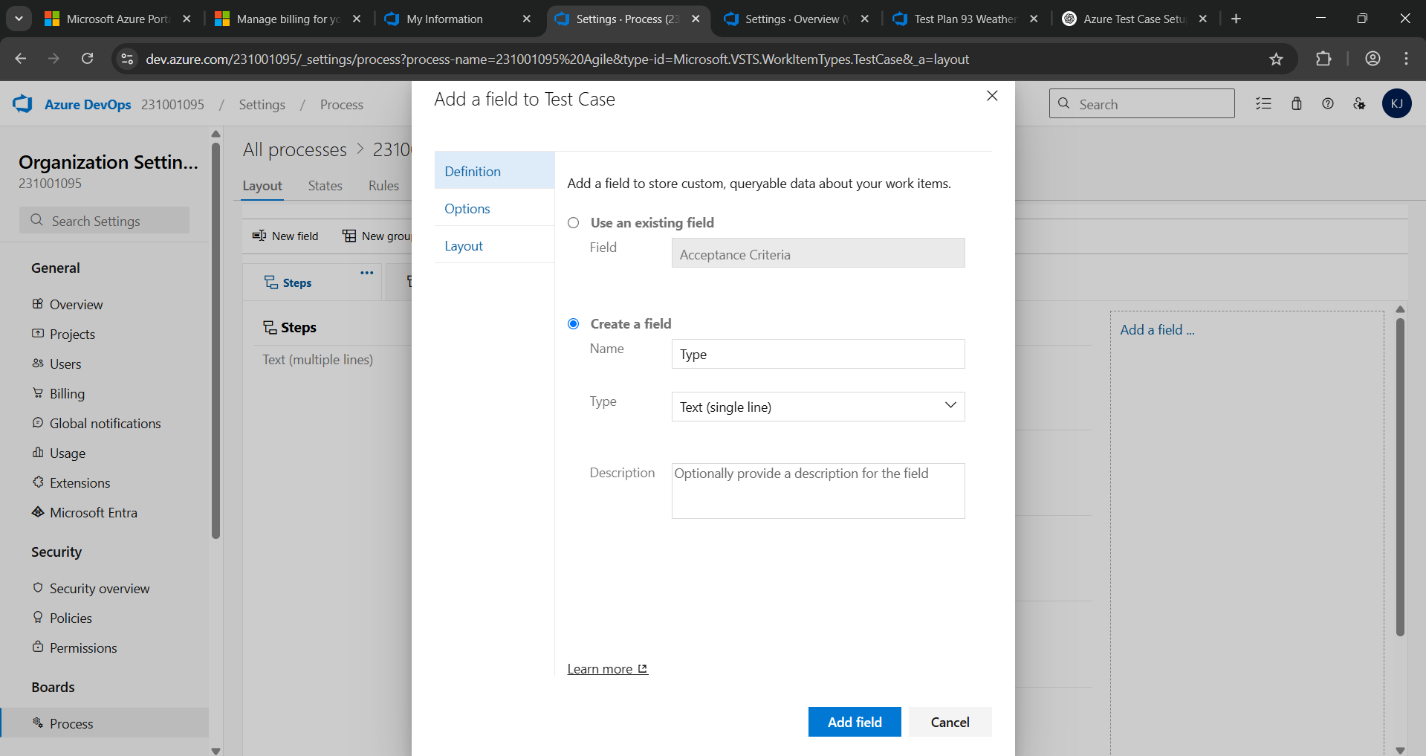
* + Assigning bug to the developer and changing state

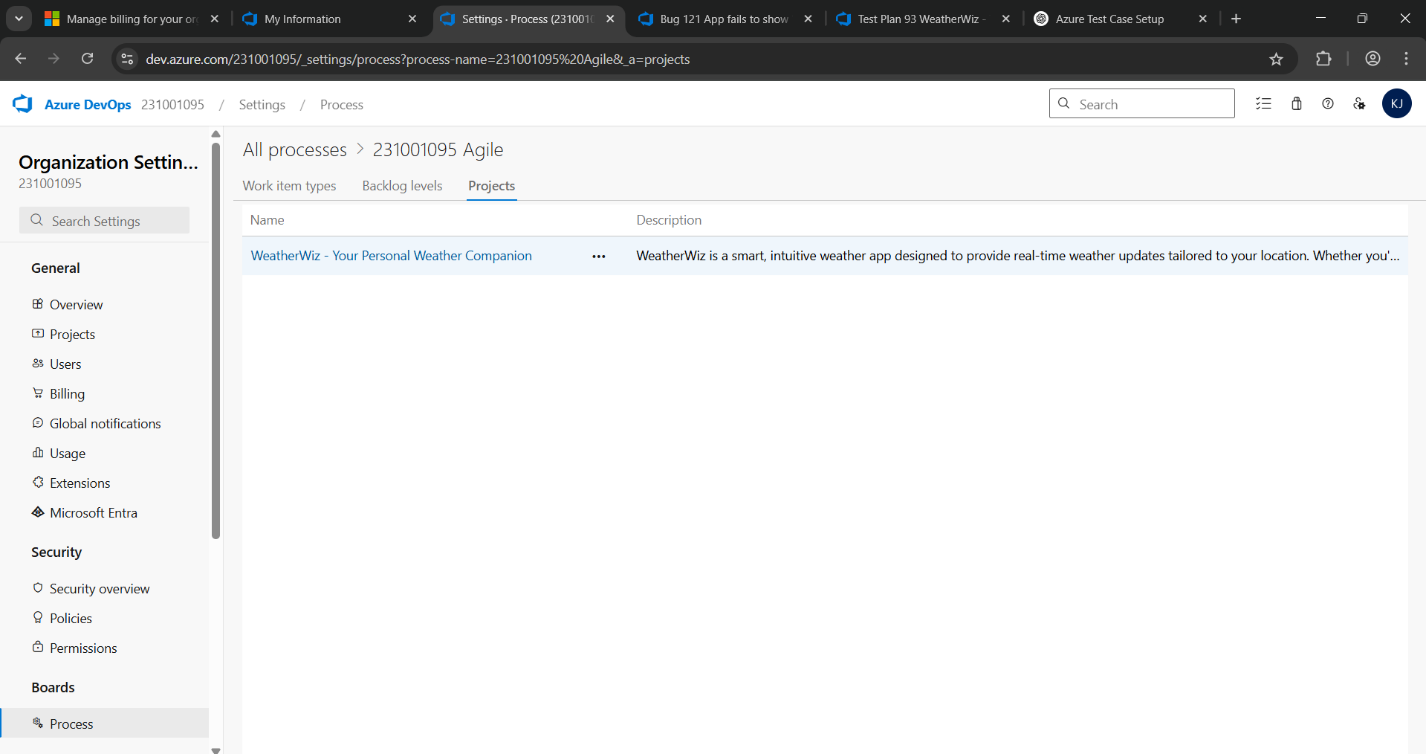
1. **Progress report**
2. **Changing the test template**

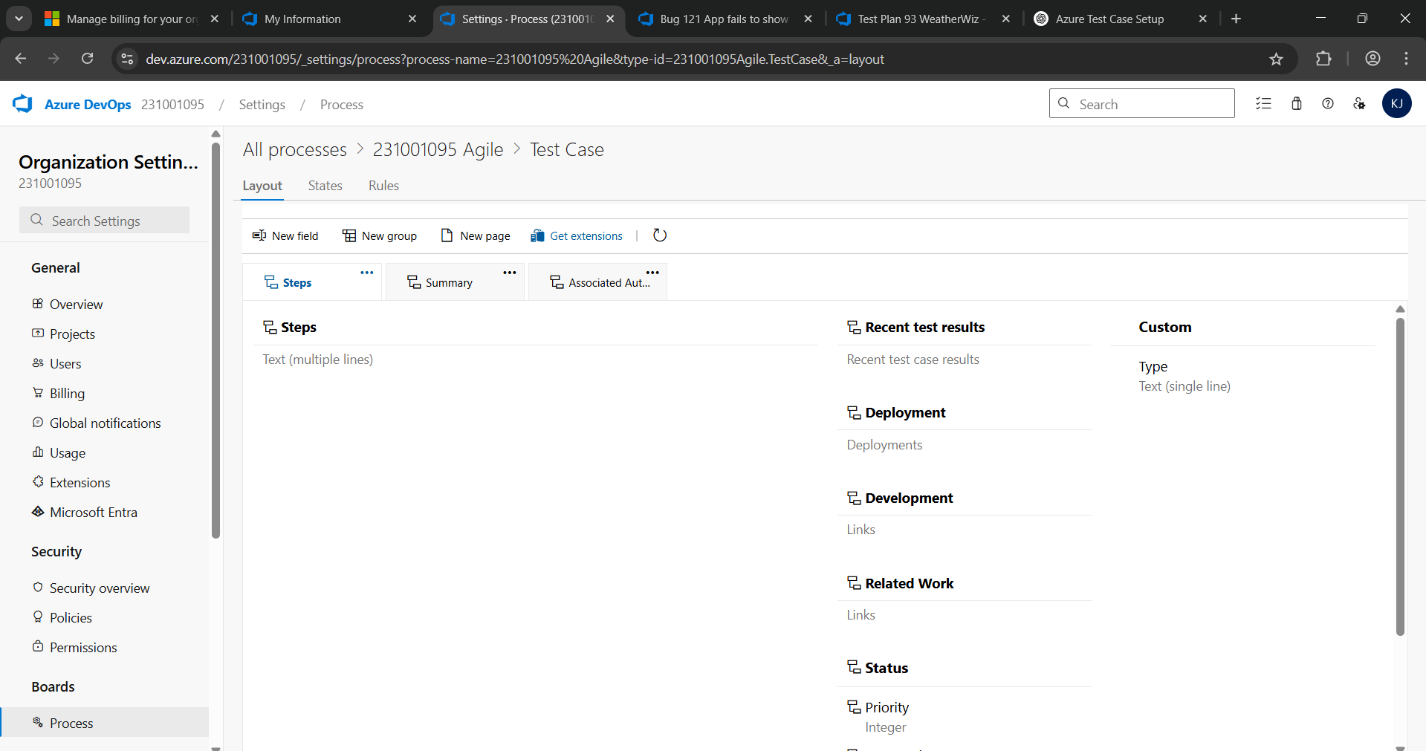
****

****

1. **View the new test case template**

****

****



**Result:**

The test plans and test cases for the user stories is created in Azure DevOps with Happy Path and

Error Path

**LOAD TESTING AND PERFORMANCE TESTING**

**EXP NO: 9**

**Aim:**

Create Epic, Features, User Stories, Task

To create an Azure Load Testing resource and run a load test to evaluate the performance of a target endpoint.

**Load Testing**

**Steps to Create an Azure Load Testing Resource:**

Before you run your first test, you need to create the Azure Load Testing resource:

1. Sign in to Azure Portal

Go to [https://portal.azure.com](https://portal.azure.com/) and log in.

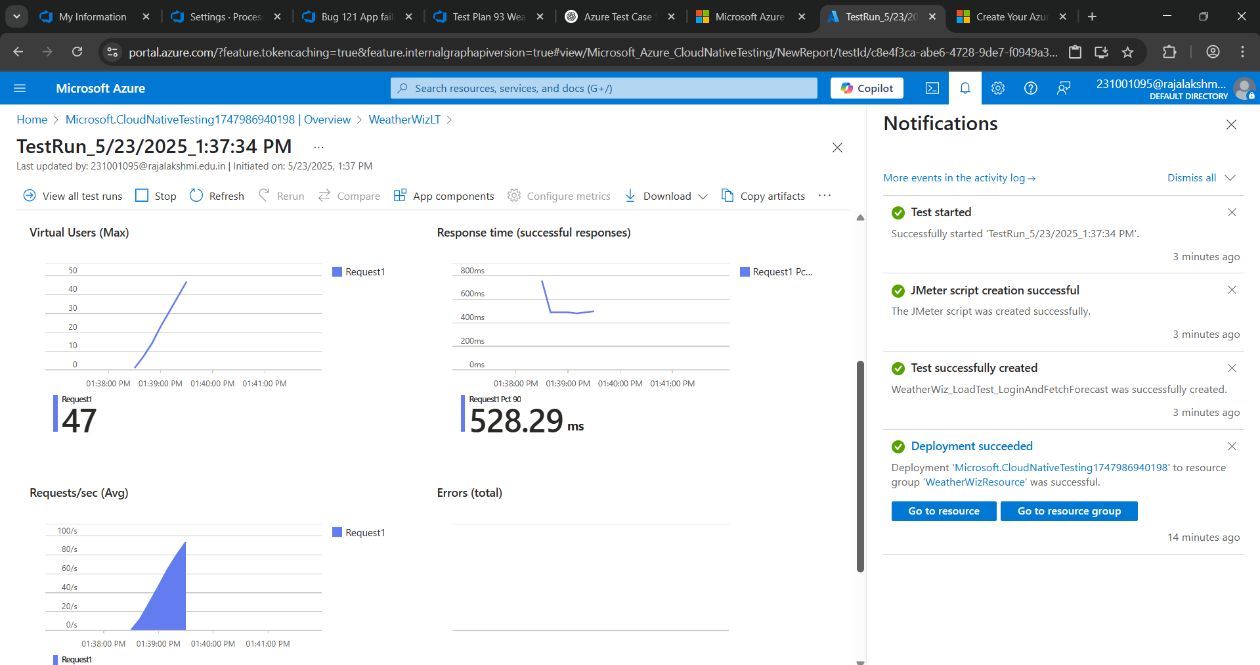
1. Create the Resource
   * Go to *Create a resource* → Search for “Azure Load Testing”.
   * Select Azure Load Testing and click Create.
2. Fill in the Configuration Details
   * Subscriptio*n:* Choose your Azure subscription.
   * *Resource Group:* Create new or select an existing one.
   * *Name:* Provide a unique name (no special characters).
   * *Location:* Choose the region for hosting the resource.
3. (Optional) Configure tags for categorization and billing.
4. Click Review + Create, then Create.
5. Once deployment is complete, click Go to resource.

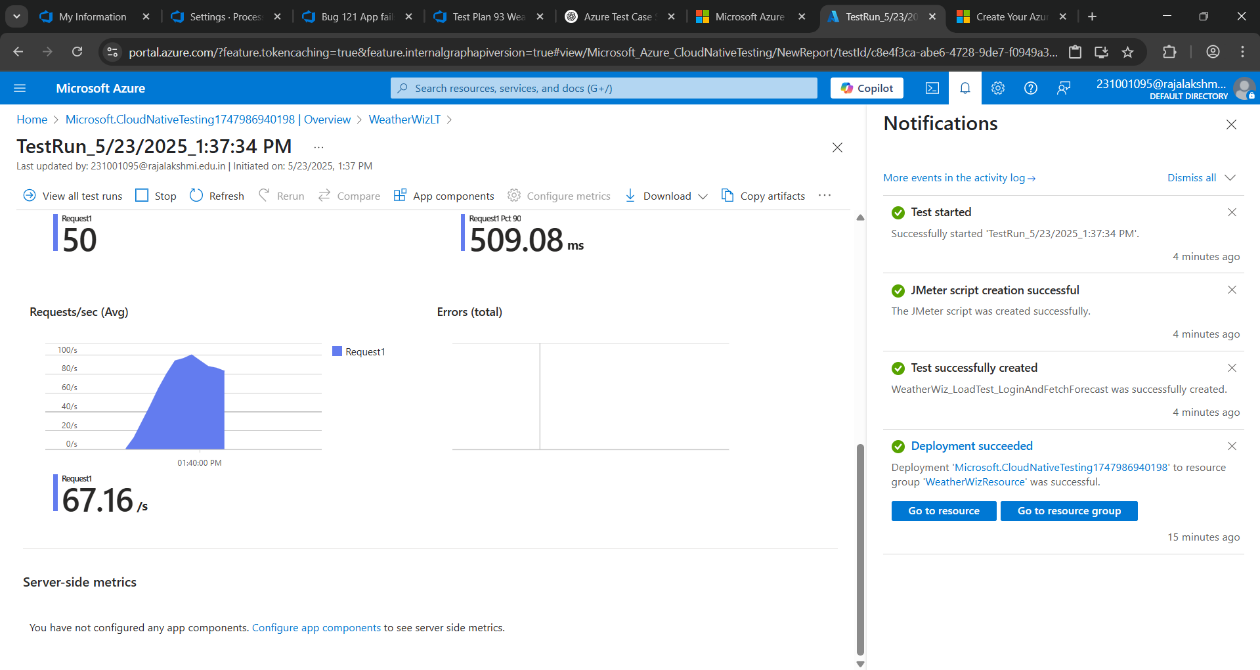
**Steps to Create and Run a Load Test:**

Once your resource is ready:

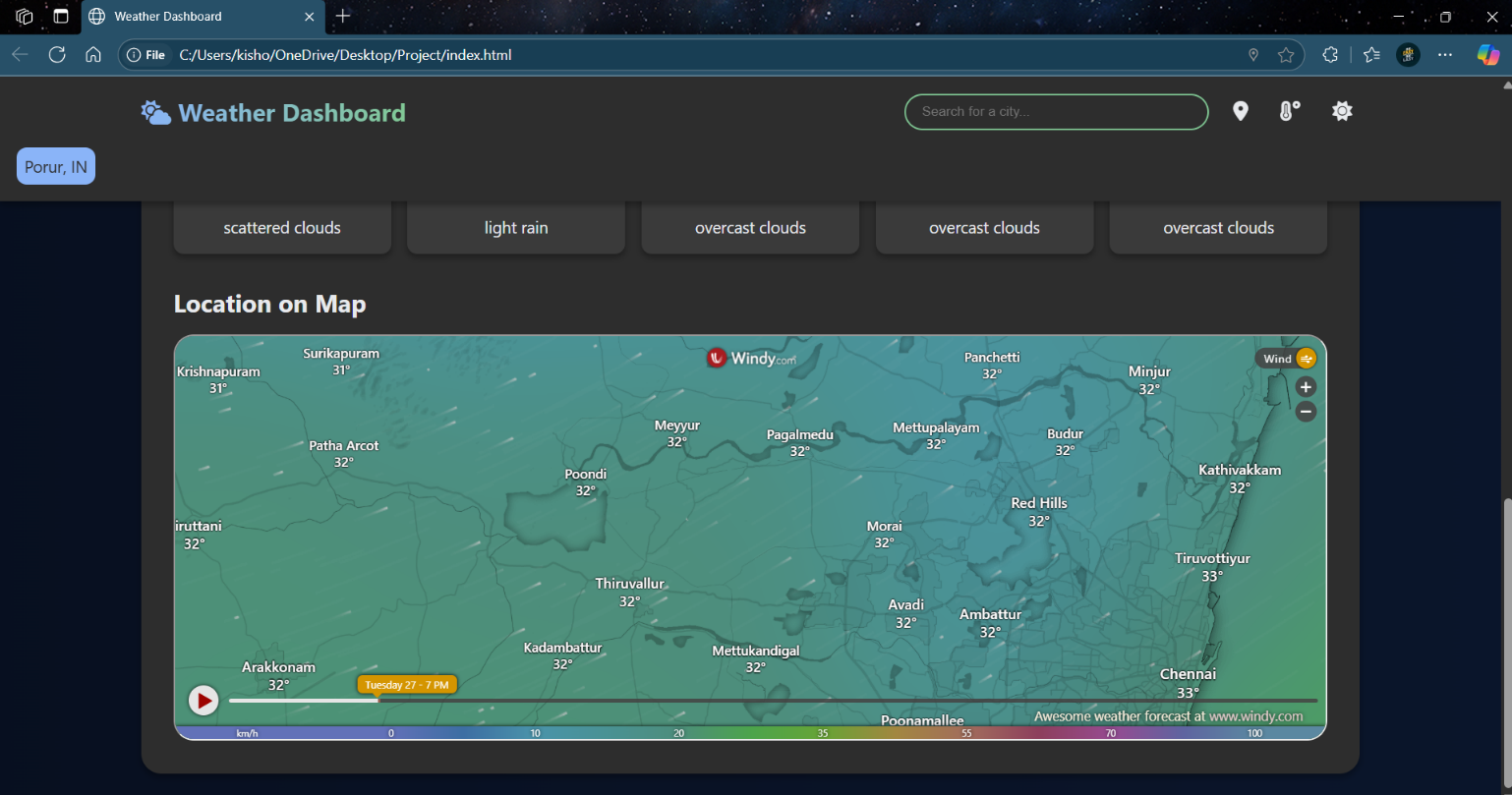
1. Go to your Azure Load Testing resource and click Add HTTP requests > Create.
2. Basics Tab
   * *Test Name:* Provide a unique name.
   * *Description:* (Optional) Add test purpose.
   * *Run After Creation:* Keep checked.
3. Load Settings
   * *Test URL:* Enter the target endpoint (e.g., https://yourapi.com/products).
4. Click Review + Create → Create to start the test.

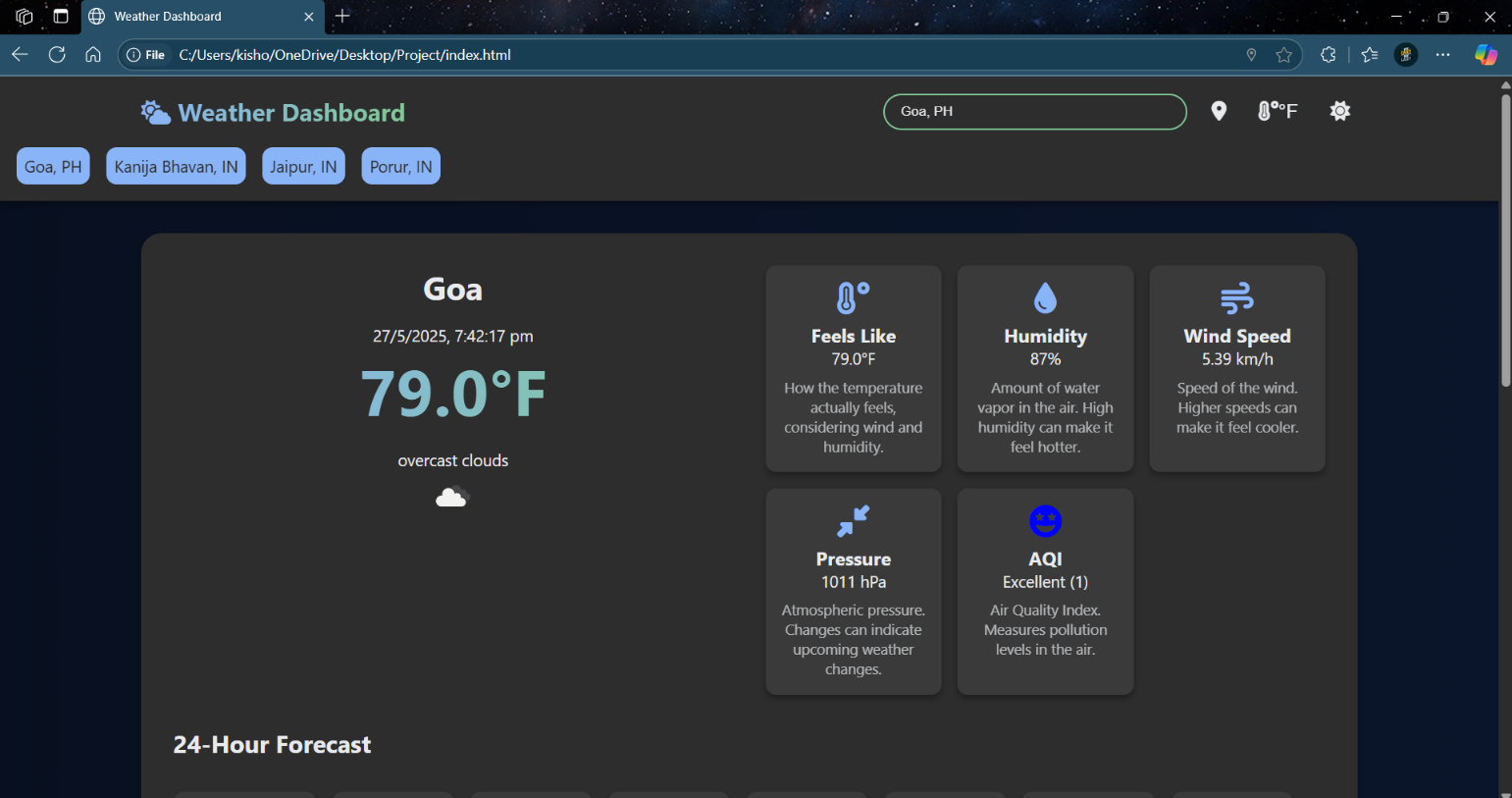
**Load Testing**

****

****



****

****

**Result:**

Successfully created the Azure Load Testing resource and executed a load test to assess the

performance of the specified endpoint.

**GITHUB: PROJECT STRUCTURE & NAMING CONVENTIONS**

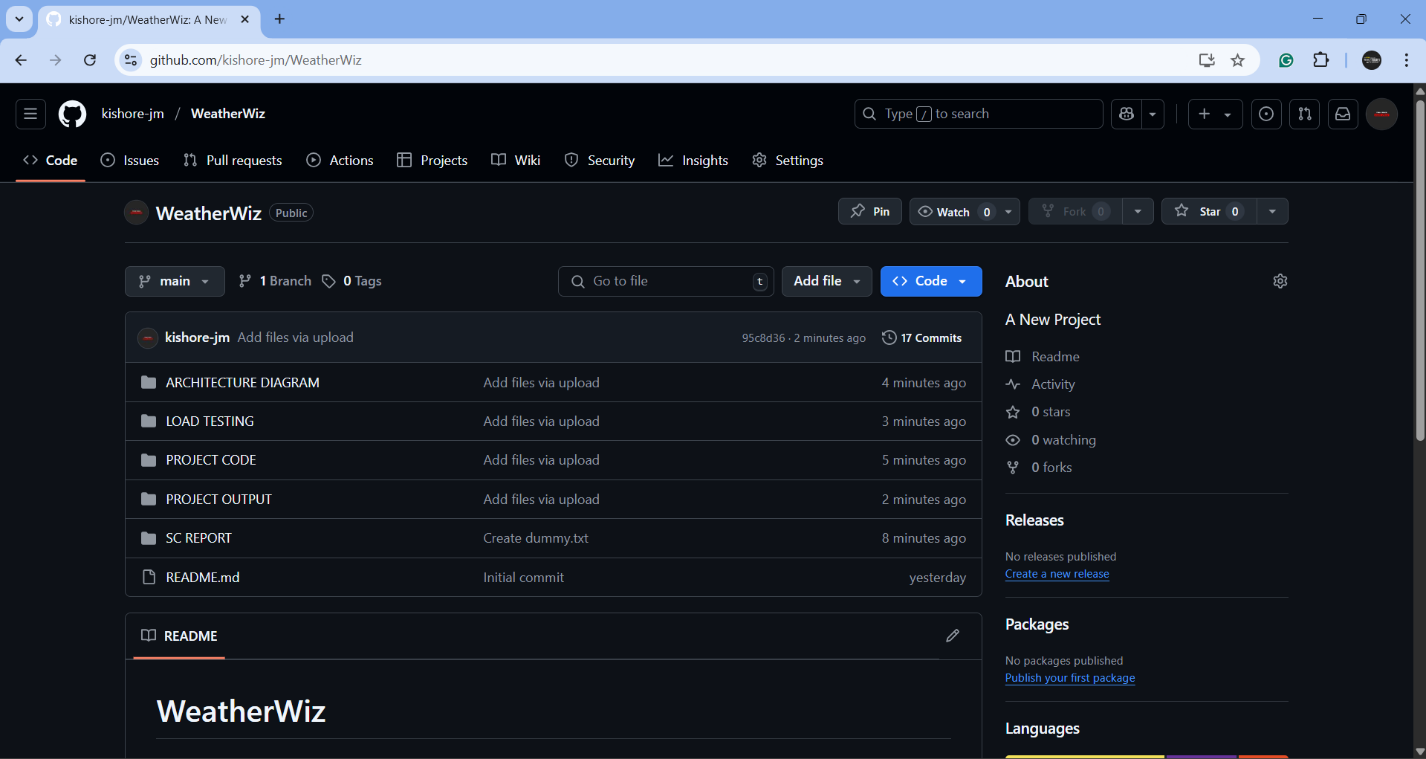
**EXP NO: 10**

**Aim:**

Create Epic, Features, User Stories, Task

To provide a clear and organized view of the project's folder structure and file naming conventions, helping contributors and users easily understand, navigate, and extend the WeatherWiz - Weather Companion project.

**GitHub Project Structure**

****

**Result:**

The GitHub repository clearly displays the organized project structure and consistent naming

conventions, making it easy for users and contributors to understand and navigate the codebase.