**AZURE DEVOPS ENVIRONMENT SETUP**

**EXP NO: 1**

**Aim:**

Create Epic, Features, User Stories, Task

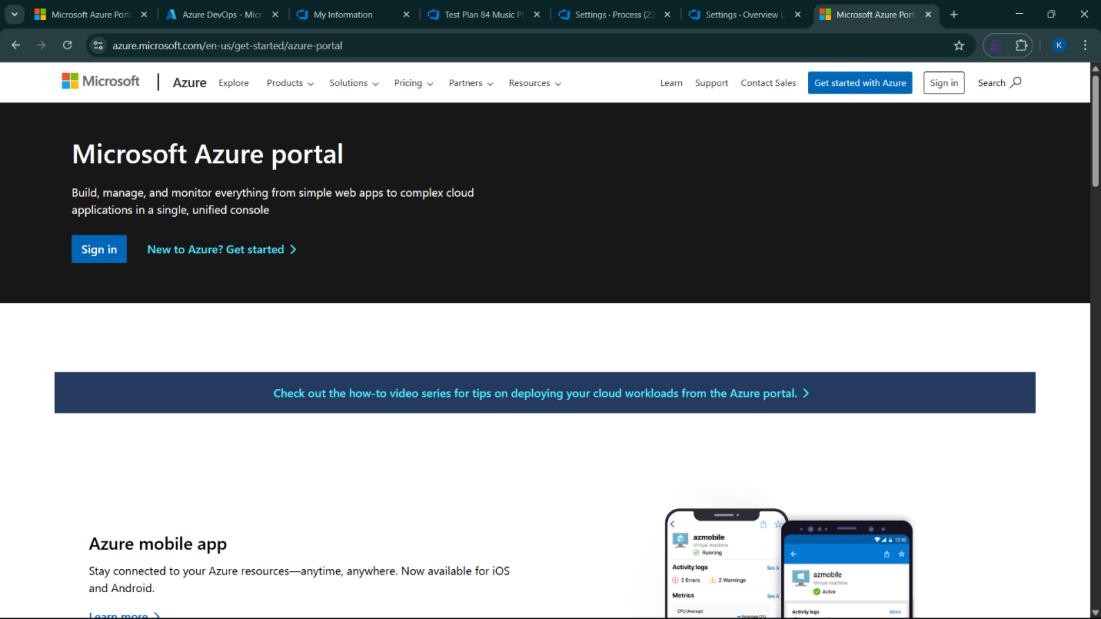
To set up and access the Azure DevOps environment by creating an organization through the Azure portal.

**INSTALLATION**

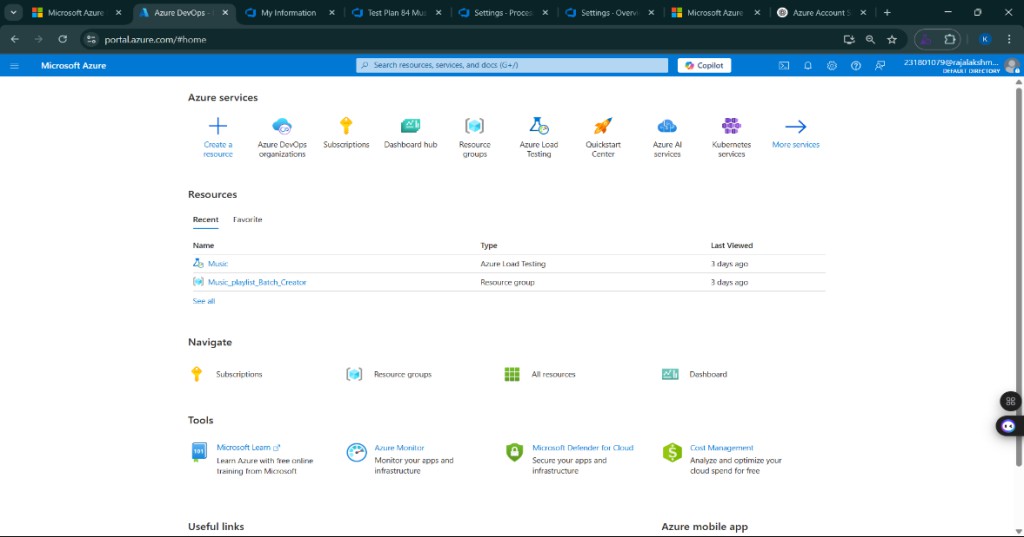
1. Open your web browser and go to the Azure website: [https://azure.microsoft.com/en-us/get-](https://azure.microsoft.com/en-us/get-started/azure-portal) [started/azure-portal](https://azure.microsoft.com/en-us/get-started/azure-portal).

Sign in using your Microsoft account credentials.

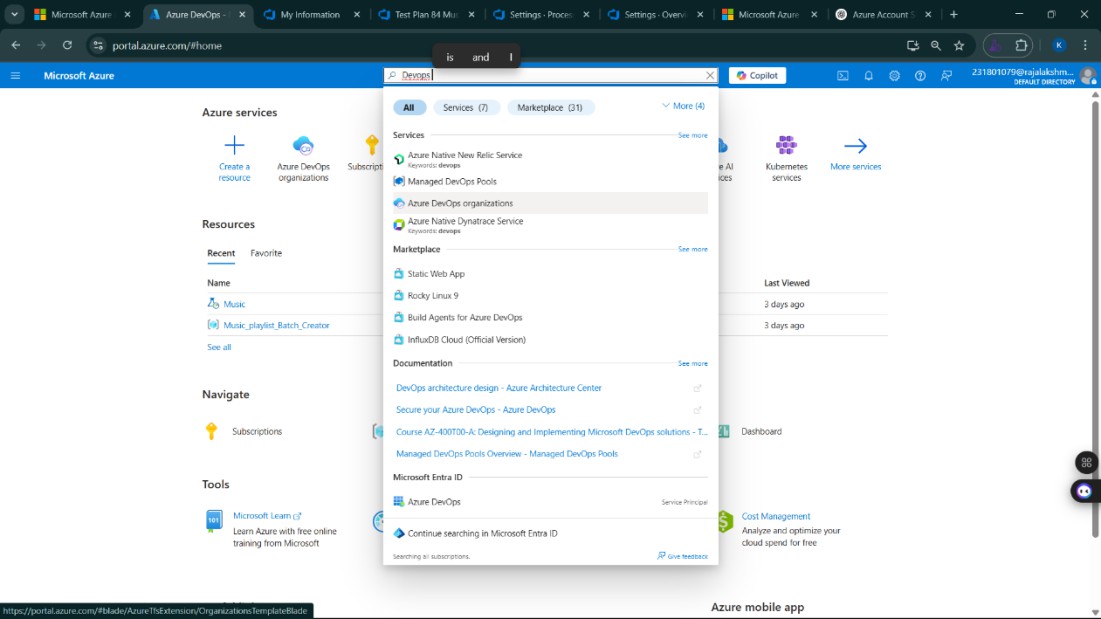
If you don't have a Microsoft account, you can create one here: <https://signup.live.com/?lic=1>



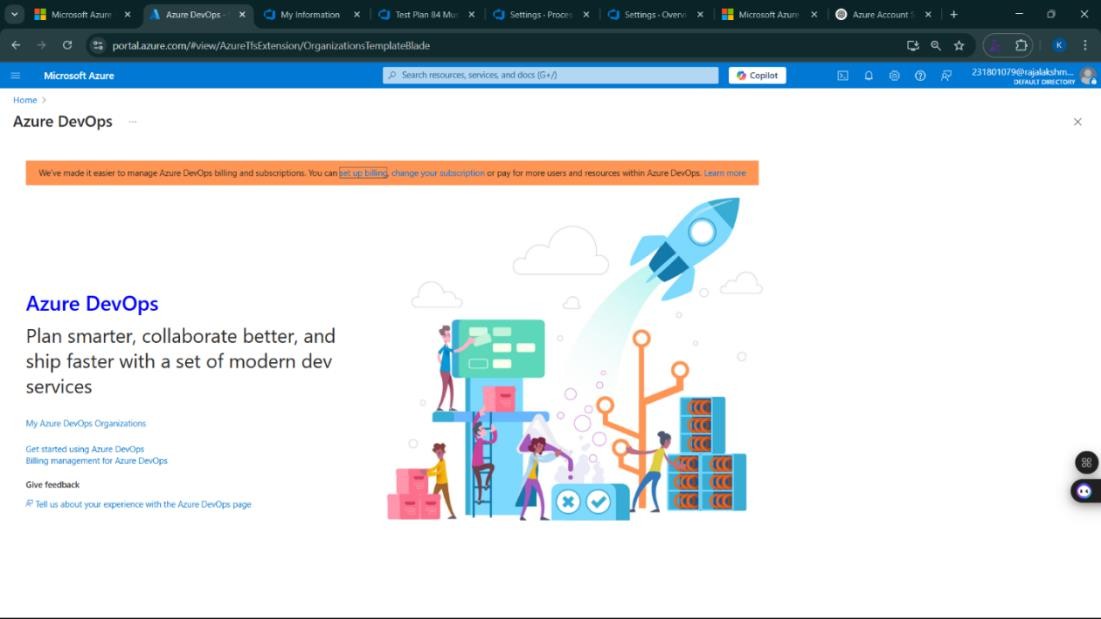
1. Azure home page



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**

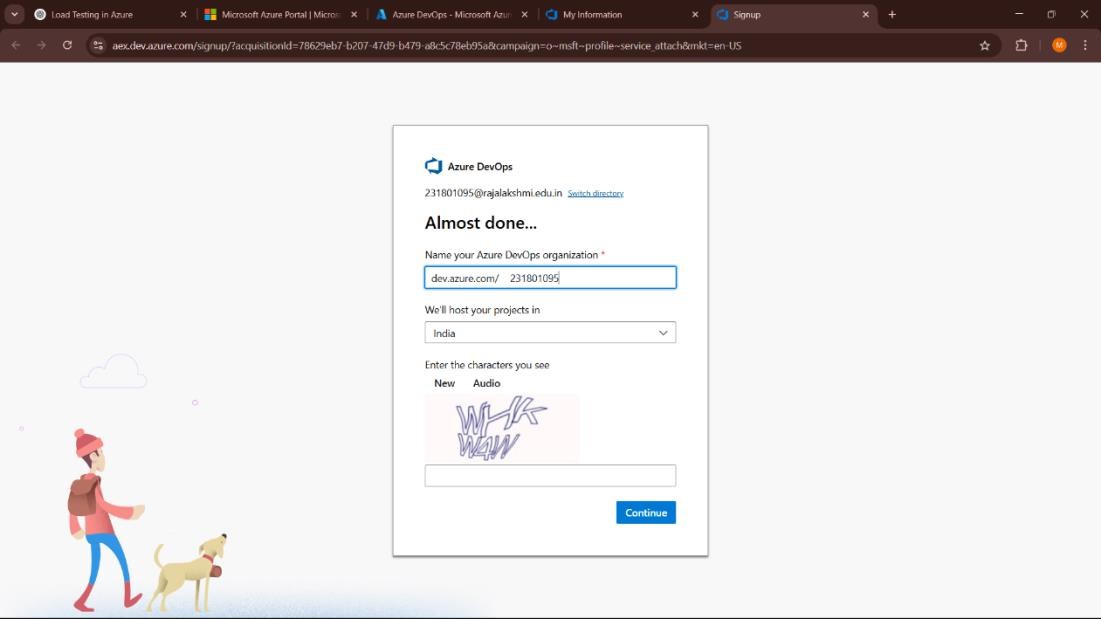
**EXP NO: 2**

**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

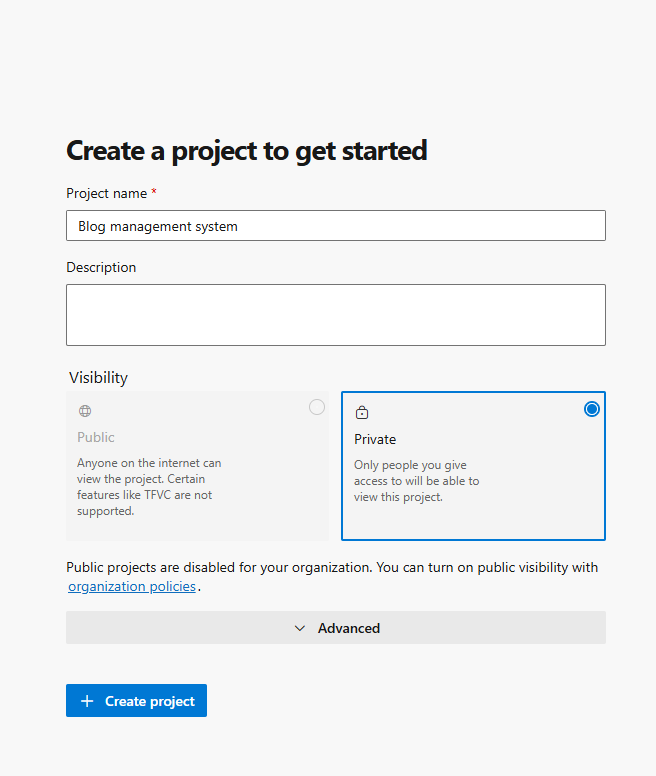


1. Create the First Project in Your Organization
   1. After the organization is set up, you’ll need to create your first **project**. This is where you'll begin to manage code, pipelines, work items, and more.
   2. On the organization’s **Home page**, click on the **New Project** button.
   3. Enter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

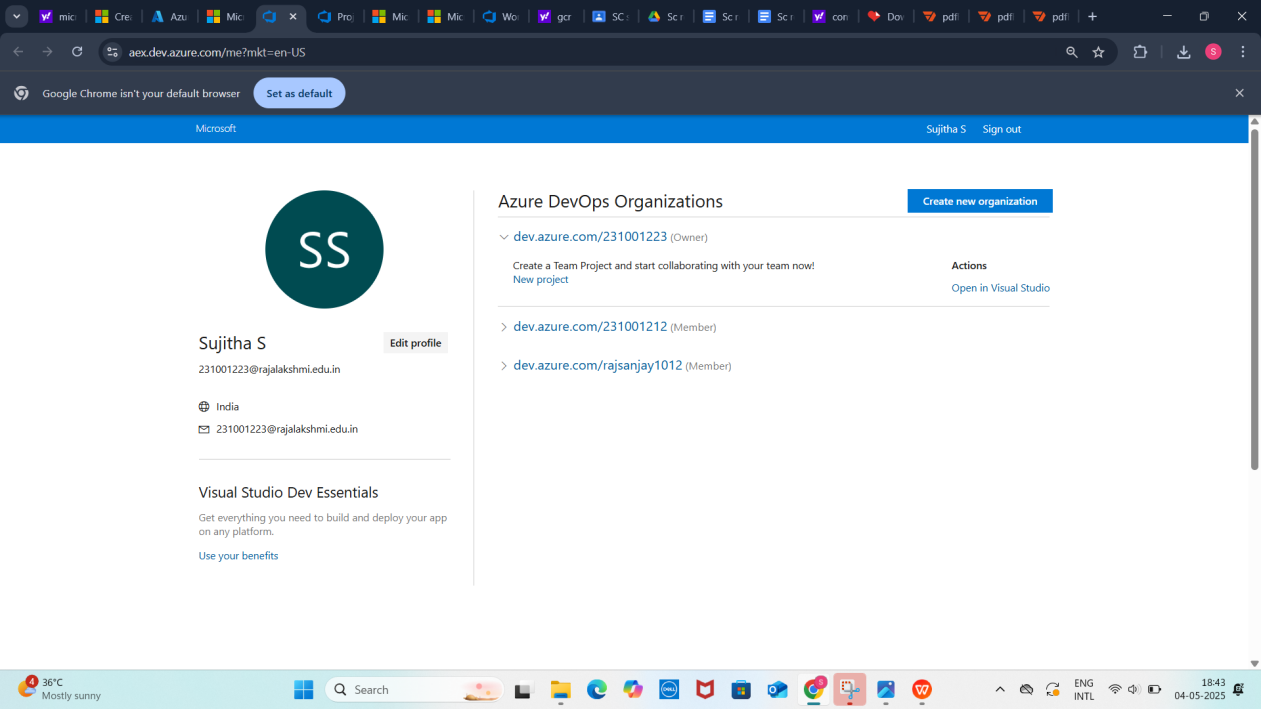
***Name****:* Choose a name for the project (e.g., LMS).

***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).

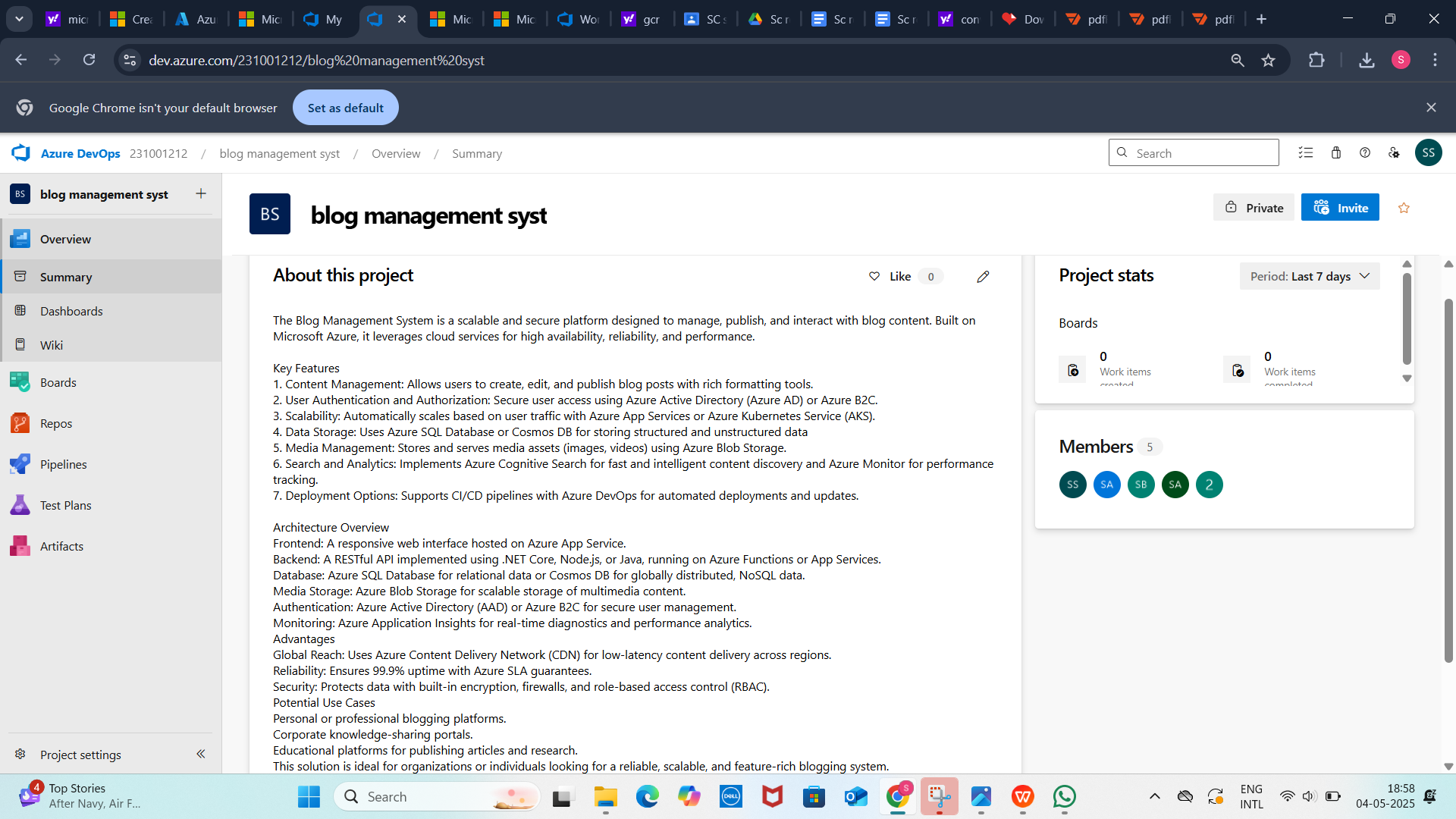
* 1. Once you’ve filled out the details, click **Create** to set up your first project.



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.



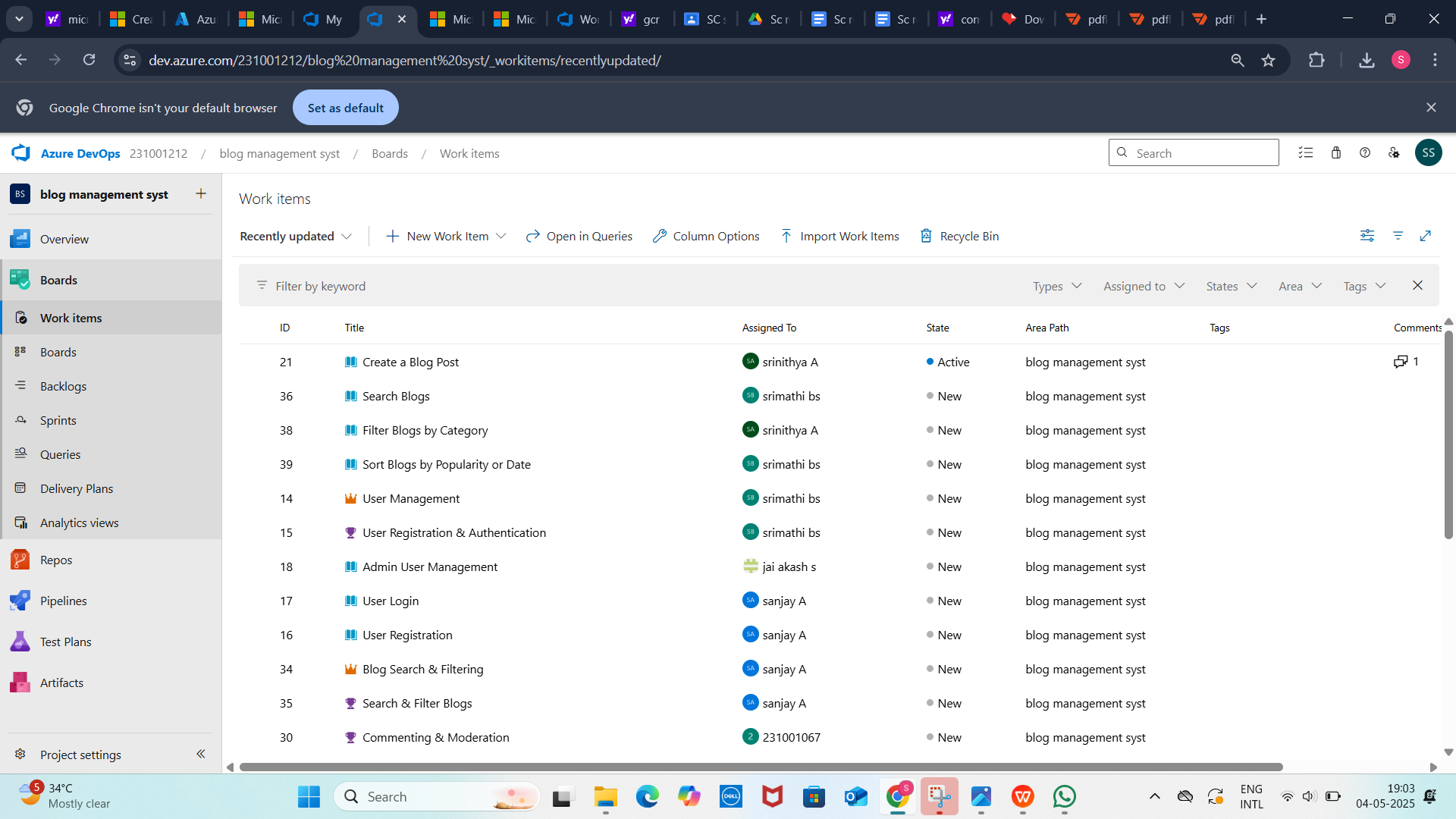
1. Project dashboard

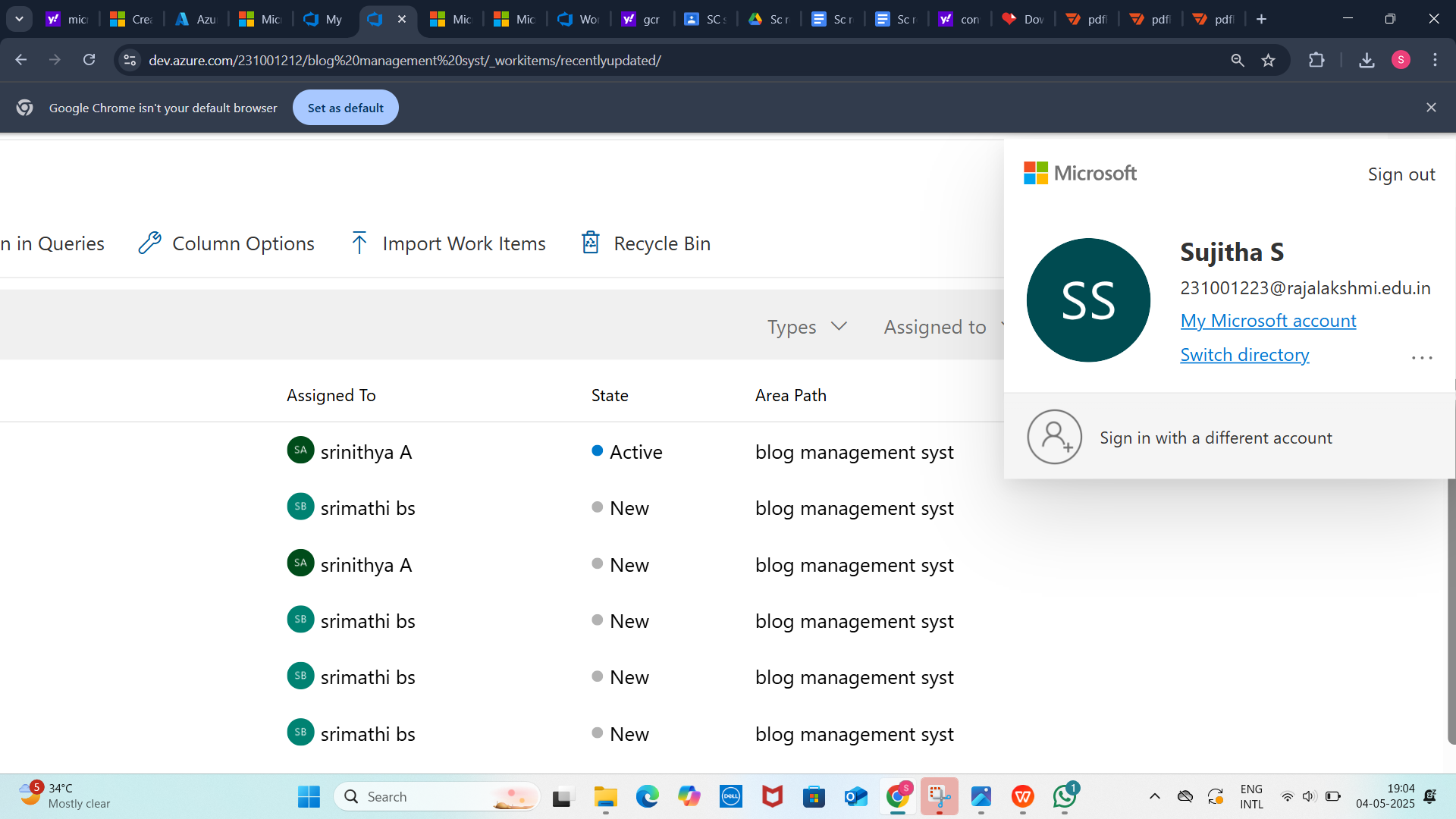


1. To manage user stories:
   1. From the **left-hand navigation menu**, click on **Boards**. This will take you to the main **Boards**

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:**

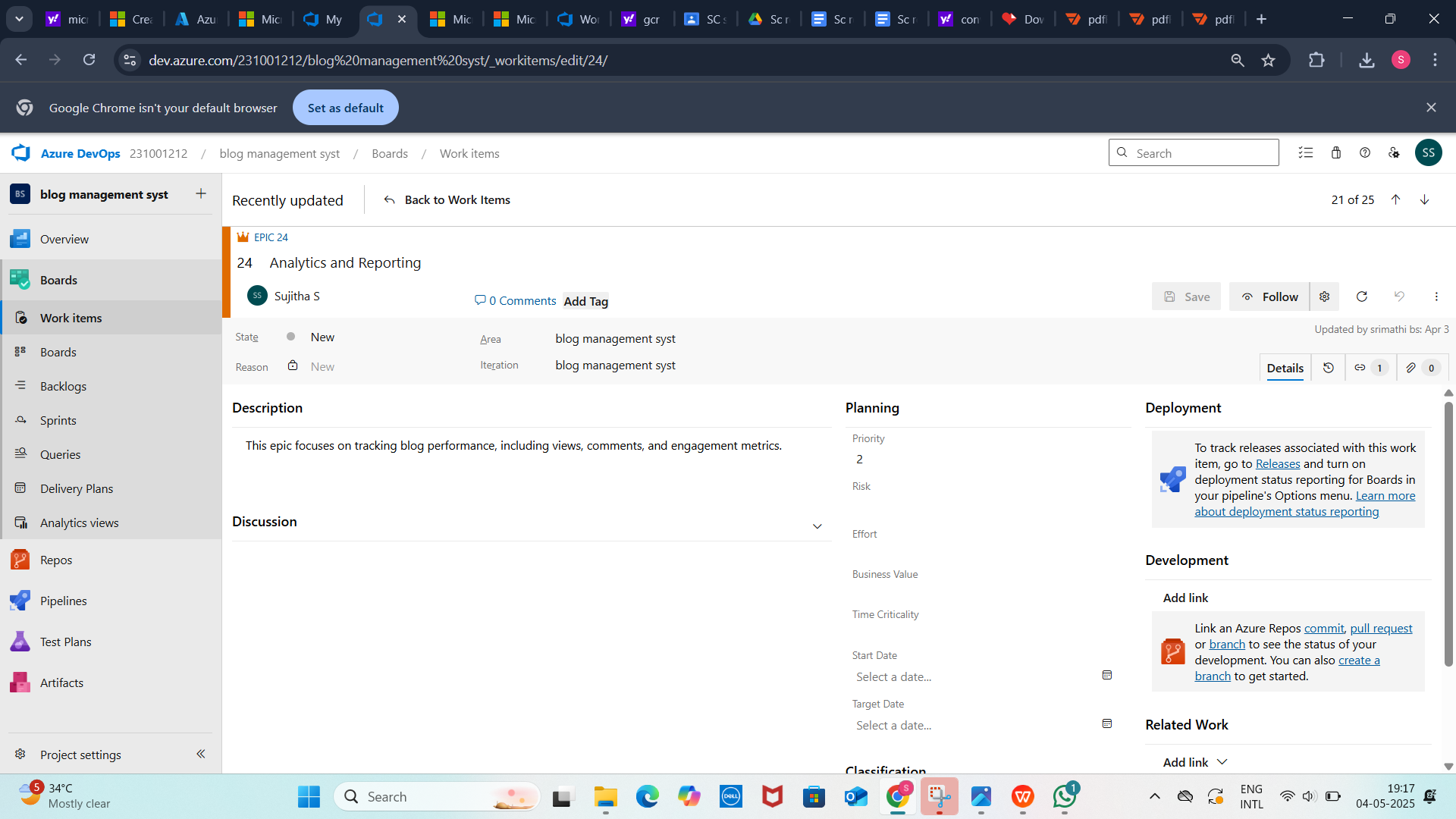
Create Epic, Features, User Stories, Task

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

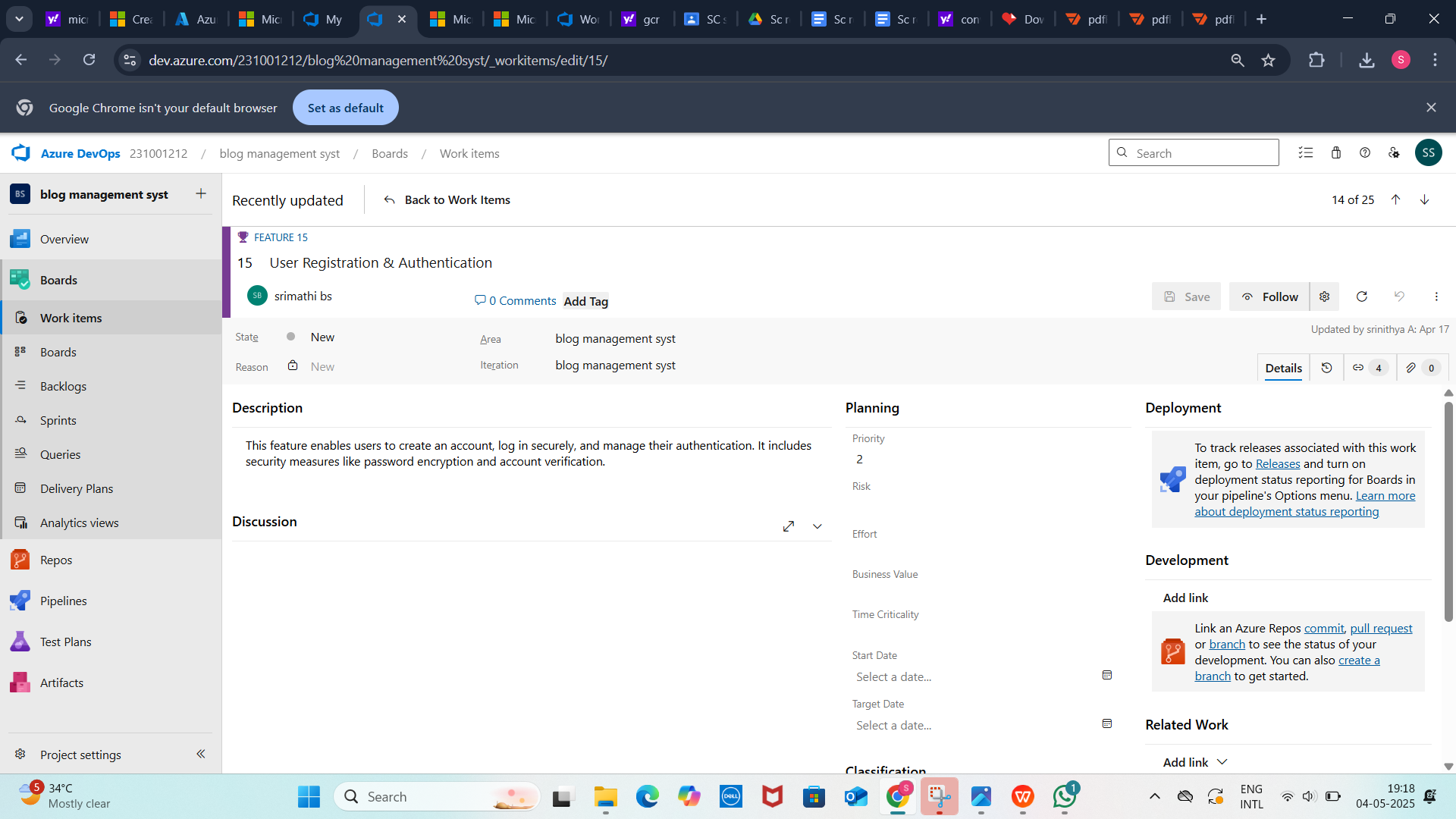
**Create Epic, Features, User Stories, Task**

****

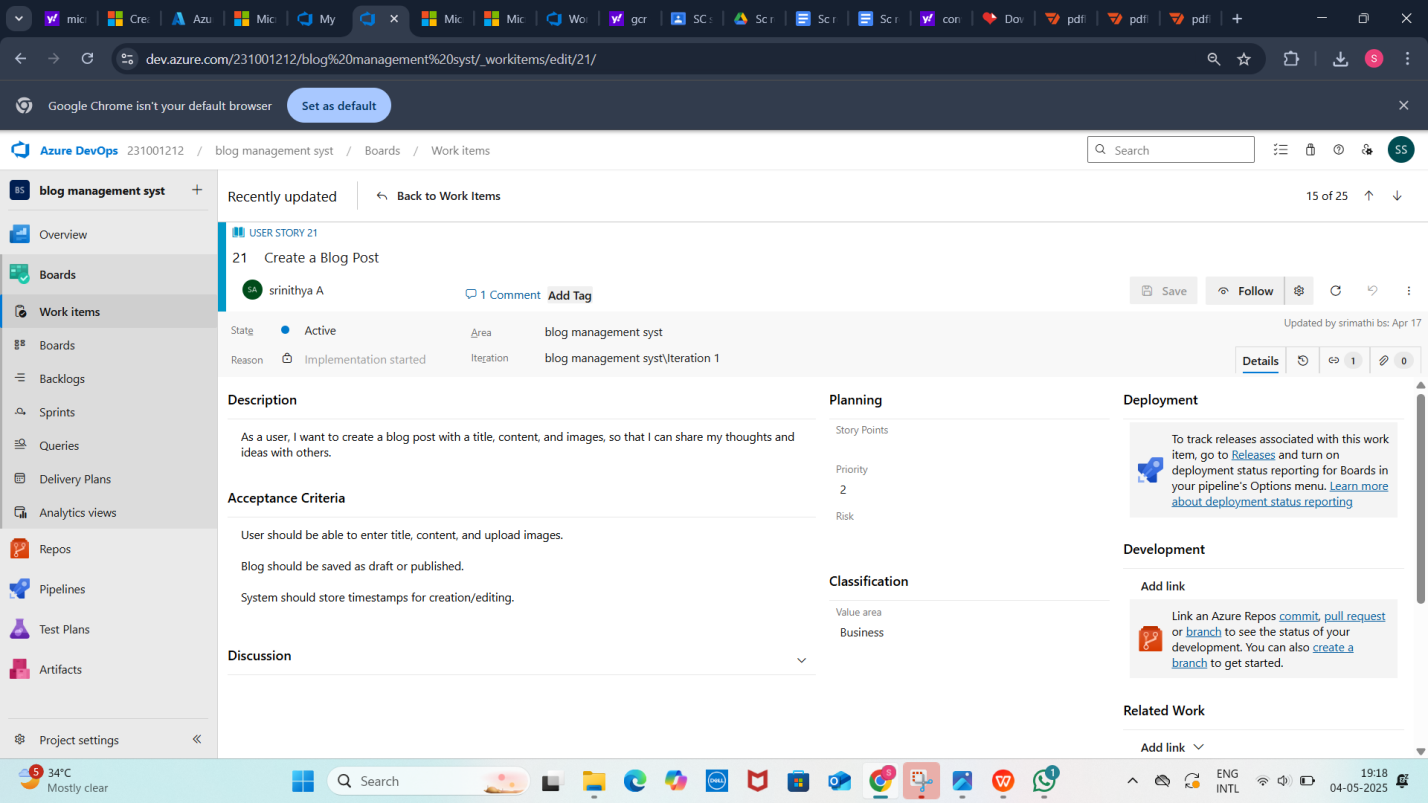
1. **Fill in Epics**

****

1. **Fill in Features**

****

1. **Fill in User Story Details**

****

**Result:**

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

**SPRINT PLANNING**

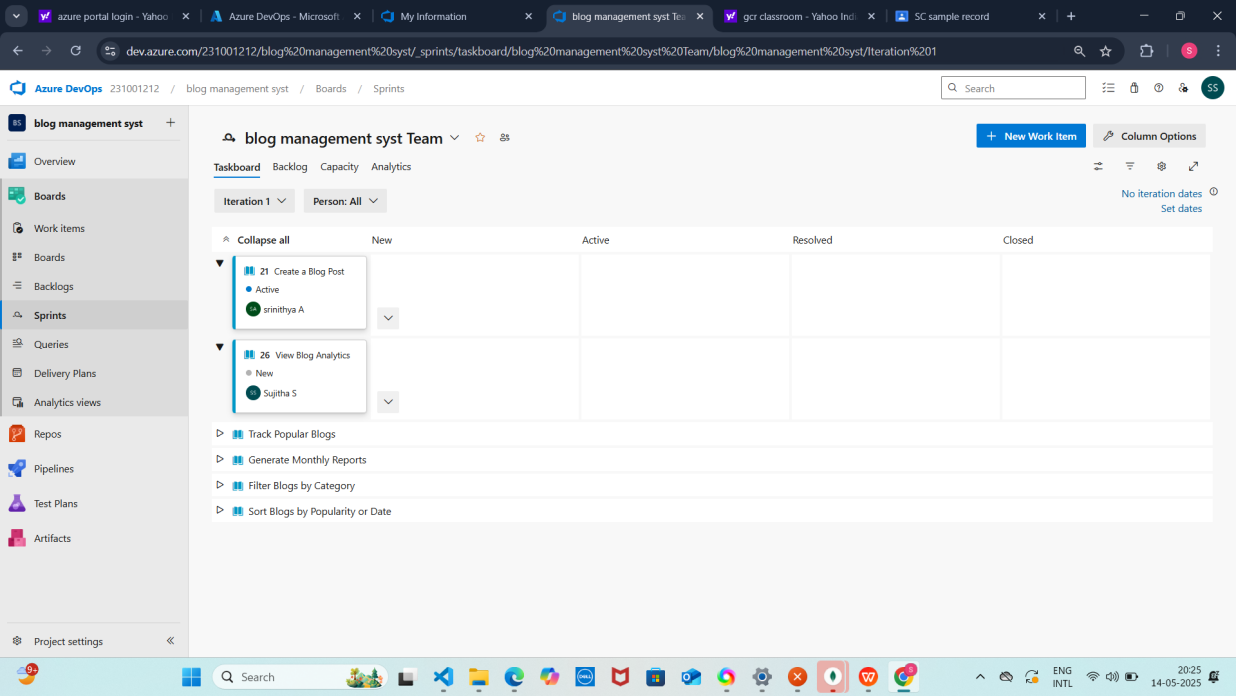
**EXP NO: 4**

**Aim:**

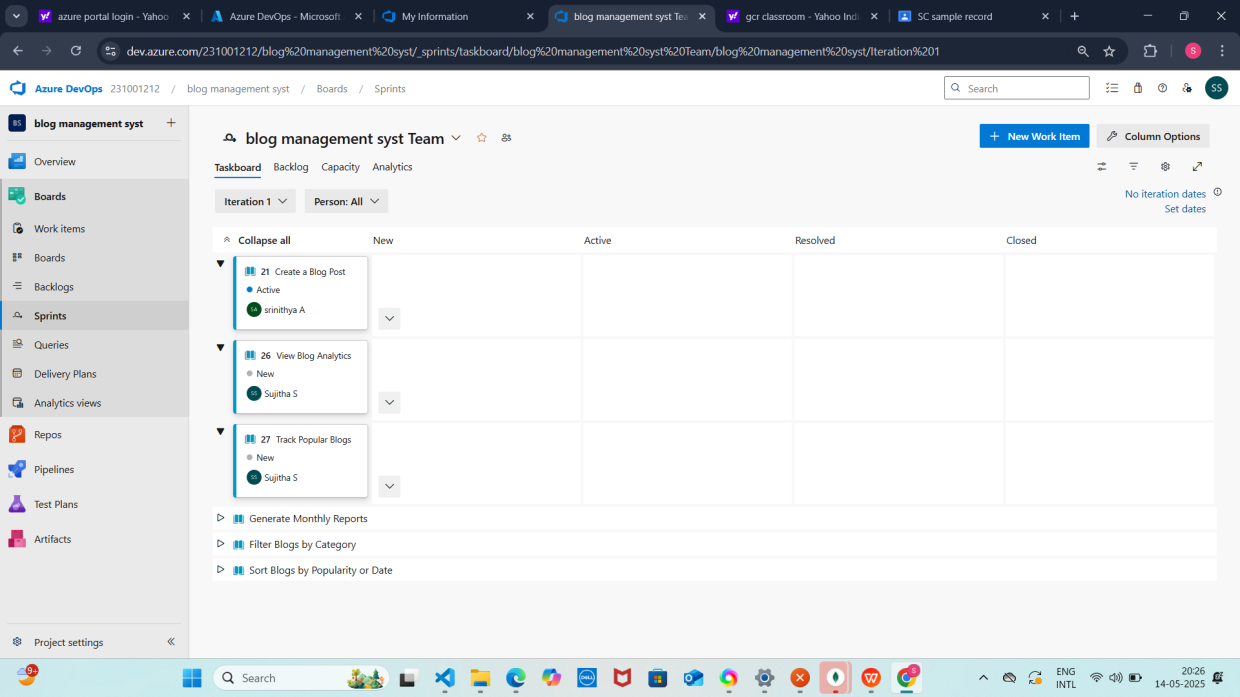
Create Epic, Features, User Stories, Task

To assign user story to specific sprint for the Blog Creator Project.

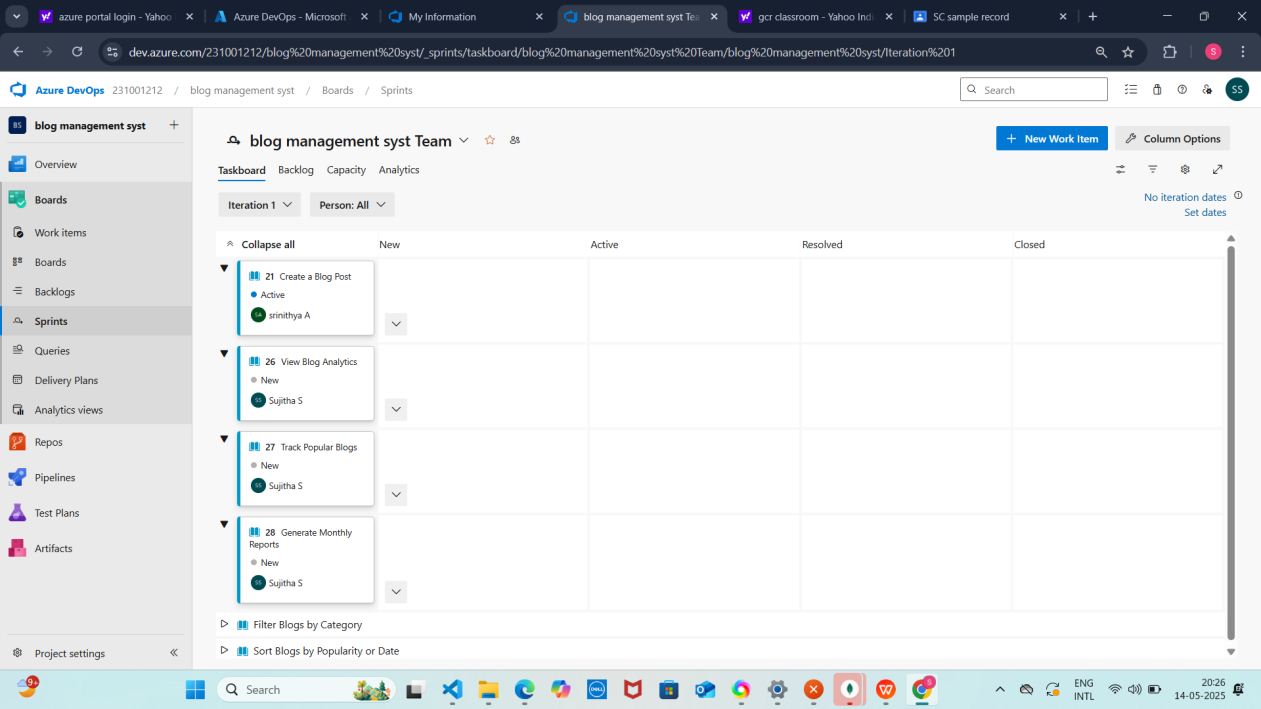
**Sprint Planning Sprint 1**

****

**Sprint 2**

****

**Sprint 3**

****

**Sprint 4**

****

**Result:**

The Sprints are created for the Blog Creator Project.

**POKER ESTIMATION**

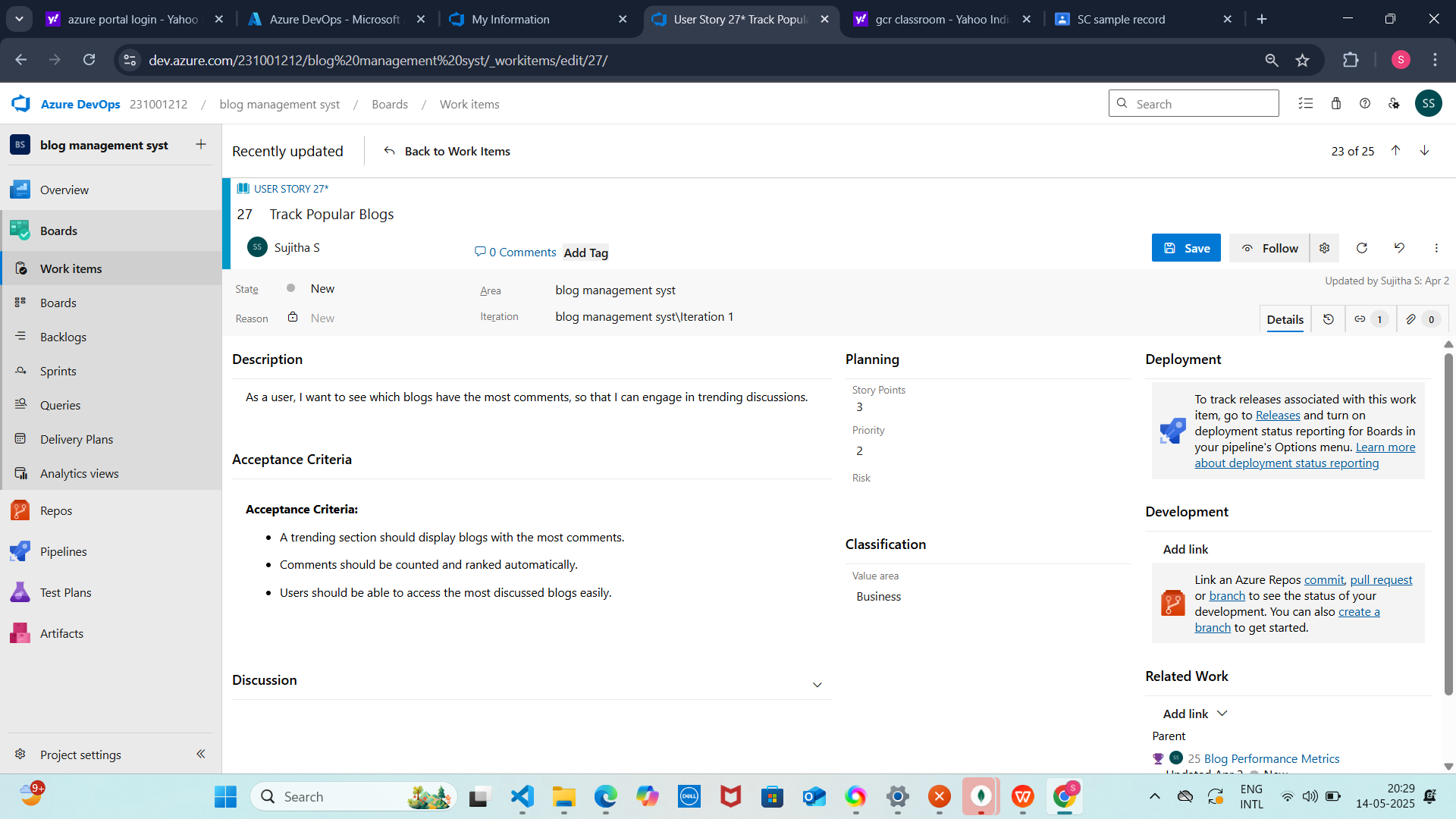
**EXP NO: 5**

**Aim:**

Create Epic, Features, User Stories, Task

Create Poker Estimation for the user stories - Blog Creator Project.

**Poker Estimation**

****

**Result:**

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

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

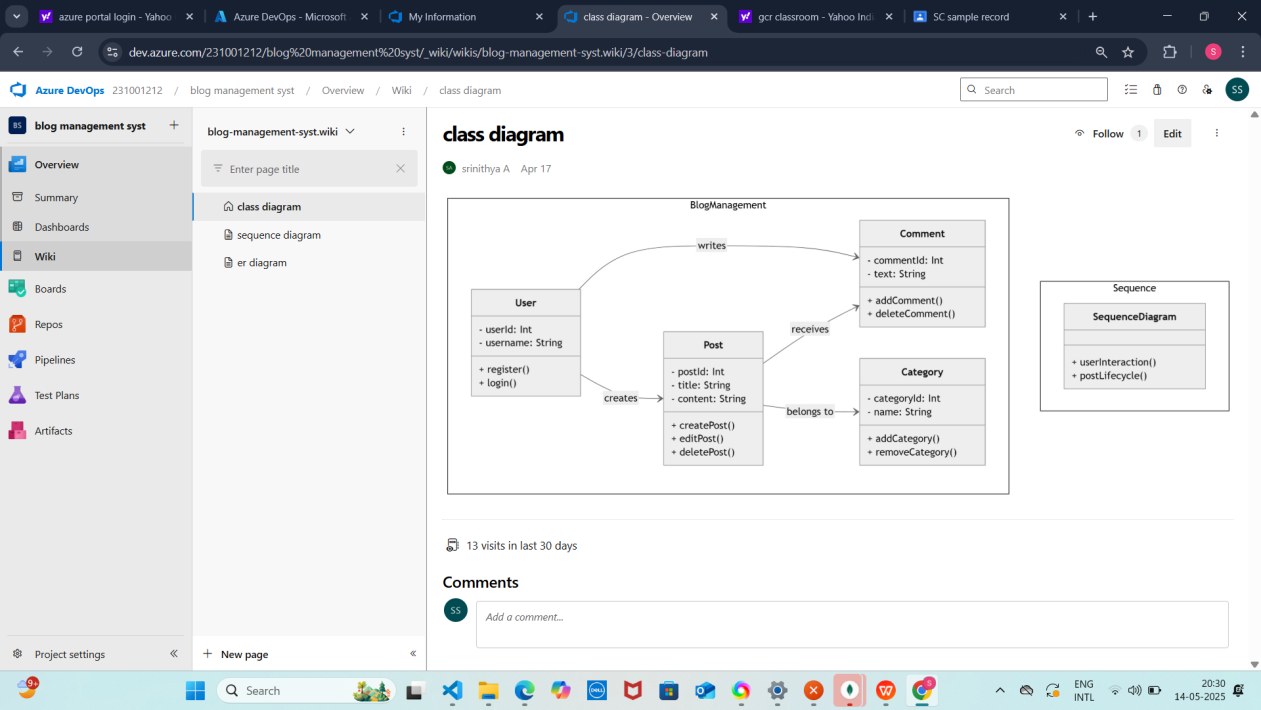
**EXP NO: 6**

**Aim:**

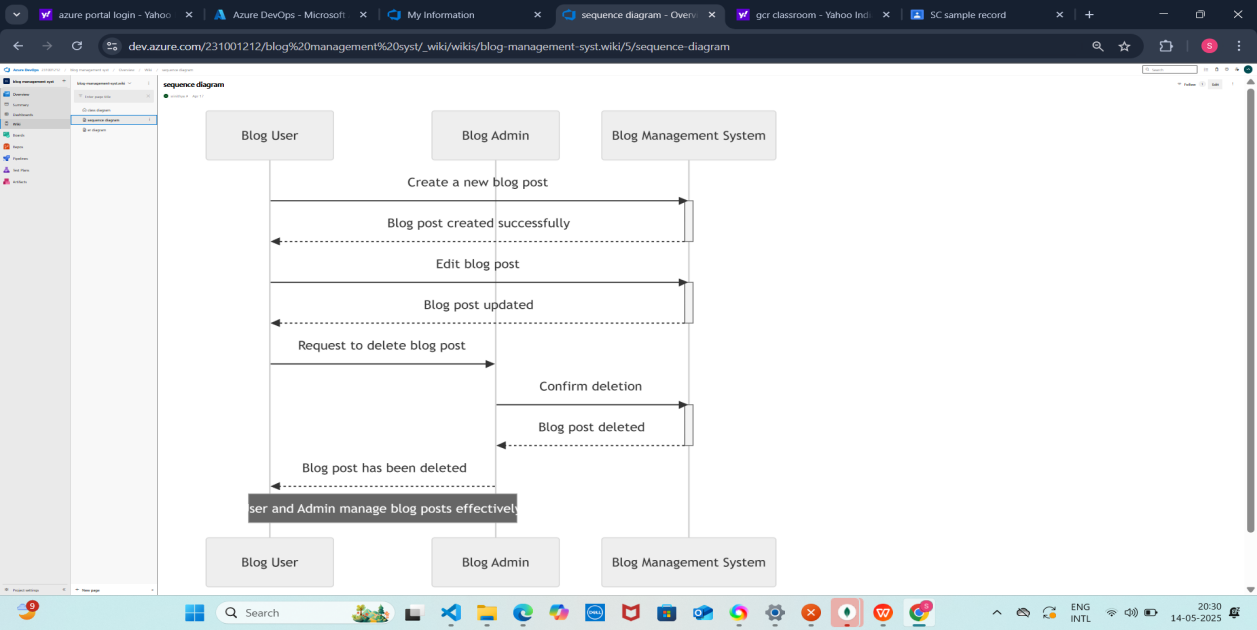
Create Epic, Features, User Stories, Task

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 Blog management system

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

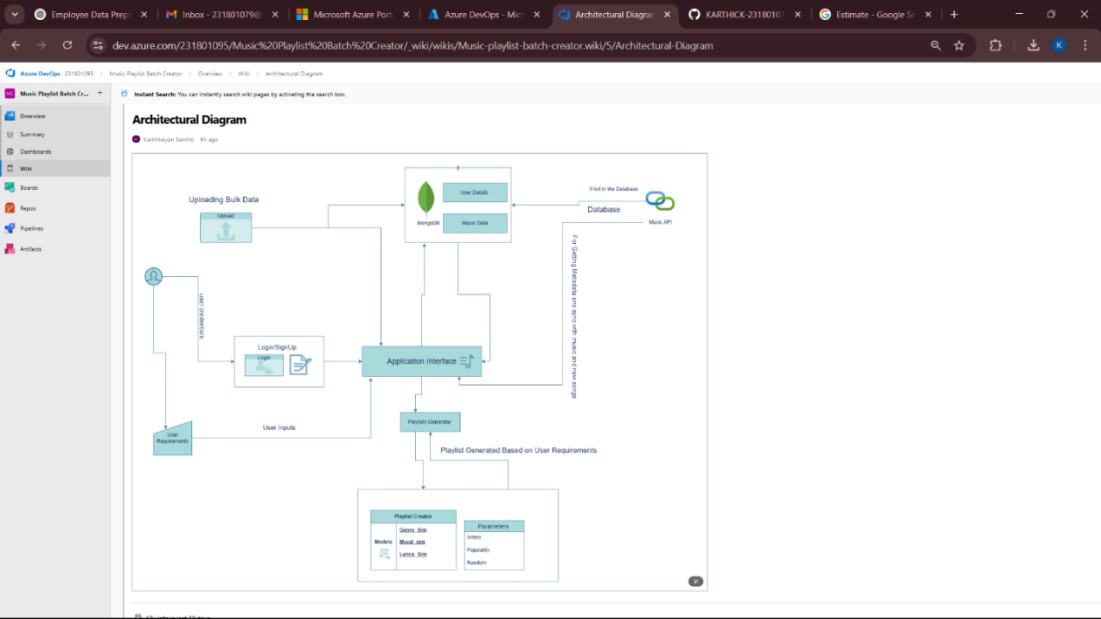
**EXP NO: 7**

**Aim:**

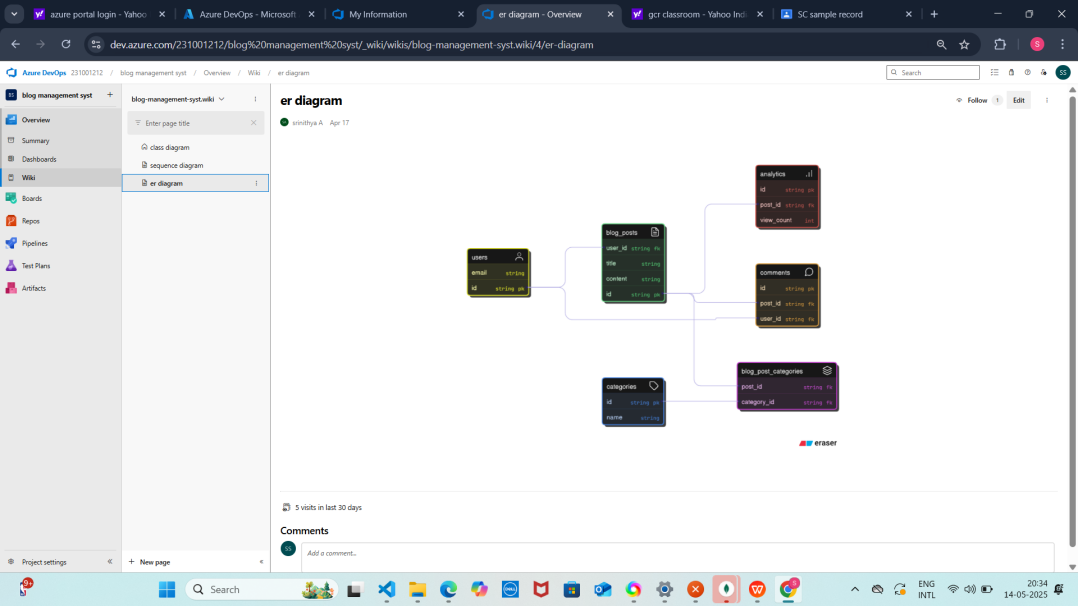
Create Epic, Features, User Stories, Task

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 Blog management sytem

-

**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

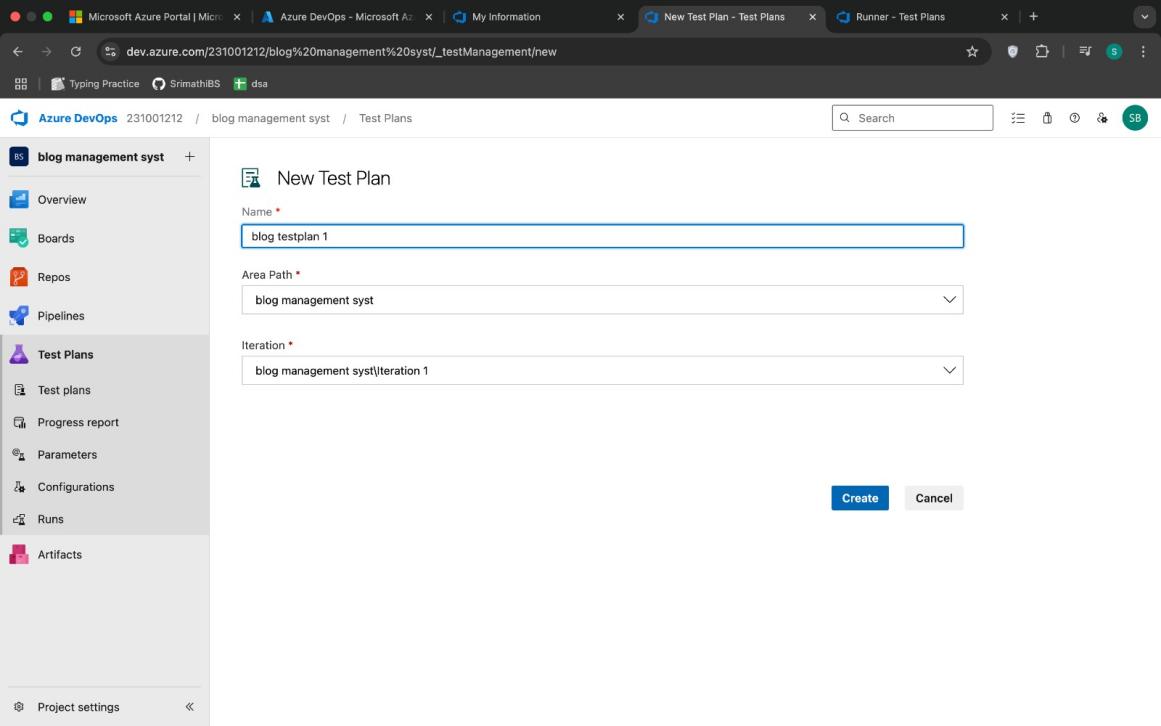
happy path and error scenarios in azure DevOps platform.

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

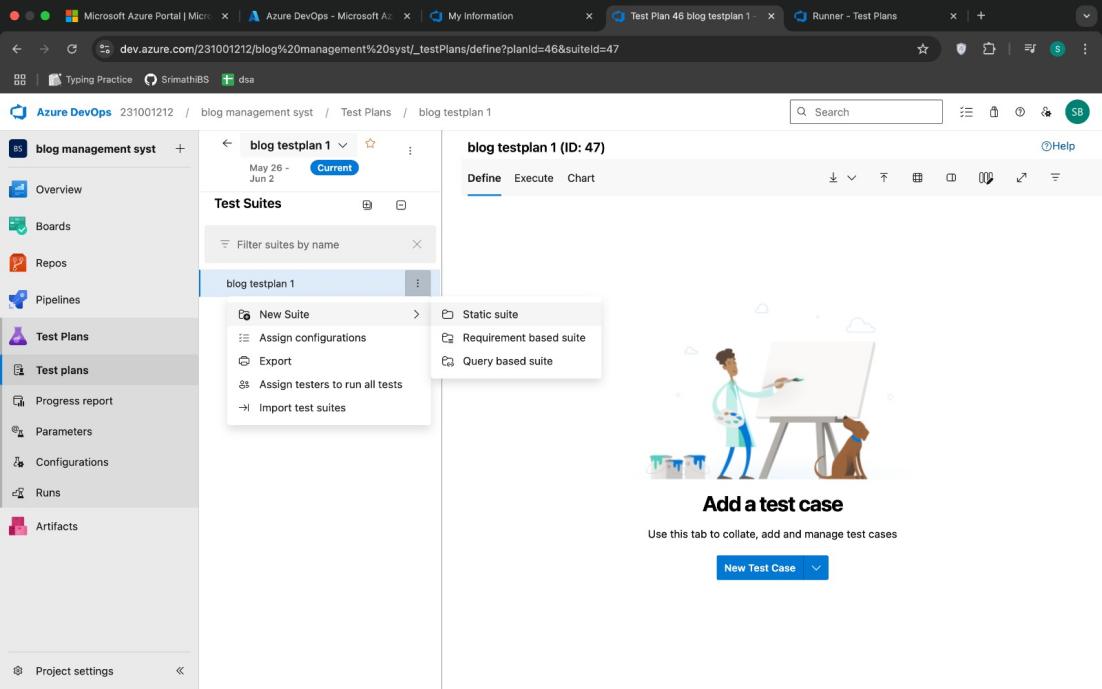
1. **Understand Core Features of the Application**
   * User Signup & Login
   * Viewing and Managing blog
   * Fetching Real-time Metadata
   * Editing blogs (rename, reorder, record)
   * Categorizing Blogs (tags, topics, categories)
2. **Define User Interaction**

Each test case simulates user behavior, such as signing in, creating a blog post, adding tags, and analyzing performance metrics.

1. **Design Happy Path Test Cases**
   * Focused on validating that all features function as expected under normal conditions.
   * Example: User logs in successfully, adds item , or creates a category-based
2. **Design Error Path Test Cases**
   * Simulate negative or unexpected scenarios to test robustness and error handling.
   * Example: Login fails with invalid credentials, save fails when offline, no recommendations found.
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, TC10 – Blog creation).
   * Helps in quick identification and linking to user stories or features.
5. **Separate Test Suites**
   * Grouped test cases based on functionality (e.g., Login, Blog Editing, Recommendation System).
   * Improves organization and test execution flow in Azure DevOps.
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.

Blog Batch Creator – Test Plans

**USER STORIES**

* + As a user, I want to sign up and log in securely so that I can access my blog (ID: 79).
  + As a user, I need to see my blog in one place (ID: 76).
  + As a user, I should be able to create an blog as needed (ID: 73).
  + As a user, I should be able to rename, record, and change the blog (ID: 68).
  + As a user, I need to have real-time metadata (ID: 65).

**Test Suites**

**Test Suit: TS01 - User Login (ID: 86)**

1. **TC01 – Successful Sign Up**
   * **Action:**
     + Go to the Sign-Up page.
     + Enter valid name, email, and password.
     + Click "Sign Up".
   * **Expected Results:**
     + Sign-Up form is displayed.
     + Fields accept values without error.
     + Account is created, and the user is redirected to the dashboard.
   * **Type**: Happy Path
2. **TC02 – Secure Login**
   * **Action:**
     + Go to the Login page.
     + Enter valid email and password.
     + Click on "Login".
   * **Expected Results:**
     + Login form is displayed.
     + Fields accept data without error.
     + User is logged in and redirected to the dashboard.
   * **Type:** Happy Path
3. **TC03 – Sign Up with Existing Email**
   * **Action:**
     + Go to the Sign-Up page.
     + Enter a name and an already registered email.
     + Click on "Sign Up".
   * **Expected Results:**
     + Fields accept data.
     + Error message "Email already registered" is displayed.
   * **Type:** Error Path
4. **TC04 – Login with Wrong Password**
   * **Action:**
     + Go to the Login page.
     + Enter valid email and incorrect password.
     + Click on "Login".
   * **Expected Results:**
     + Input is accepted.
     + Error message "Invalid username or password" is shown.
   * **Type:** Error Path

#### ****Test Suite: TS02 - Blog Viewing (ID: 76)****

**TC03 – View Blogs Successfully**

**Action**:

Log in with valid credentials.

Navigate to the "My Blogs" section.

**Expected Results**

All created blogs are displayed with titles, dates, and statuses.

**Type**: Happy Path

**TC04 – Blog Loading Failure**

**Action**:

Log in.

Disconnect from the internet.

Navigate to "My Blogs"

**Expected Results**:

Error message: "Unable to load blogs".

**Type**: Error Path

#### ****Test Suite: TS03 - Blog Creation (ID: 73)****

**TC05 – Create Blog Successfully**

**Action**:

Log in.

Navigate to "Create Blog".

Fill in all mandatory fields and click "Publish".

**Expected Results**:

Blog is created and displayed in the "My Blogs" section.

**Type**: Happy Path

**TC06 – Create Blog with Missing Fields**

**Action**:

Log in.

Navigate to "Create Blog".

Leave one or more mandatory fields blank and click "Publish".

**Expected Results**

Error message: "All fields are required to publish a blog".

**Type**: Error Path

#### ****Test Suite: TS04 - Blog Editing (ID: 68)****

**TC07 – Rename Blog Successfully**

**Action**:

Navigate to "My Blogs".

Click "Edit" next to a blog.

Change the title and click "Save"

**Expected Results**:

Blog title is updated successfully.

**Type**: Happy Path

**TC08 – Rename Blog with Empty Title**

**Action**:

Navigate to "My Blogs".

Click "Edit" next to a blog.

Leave the title field blank and click "Save".

**Expected Results**:

Error message: "Blog title cannot be empty".

**Type**: Error Path

#### ****Test Suite: TS05 - Real-Time Metadata (ID: 65)****

**TC09 – Metadata Updates in Real-Time**

**Action**:

Open a blog post.

Observe metadata like word count, reading time, and auto-save status while editing.

**Expected Results**:

Metadata updates dynamically as content changes.

**Type**: Happy Path

**TC10 – Metadata Not Updating**

**Action**:

Open a blog post.

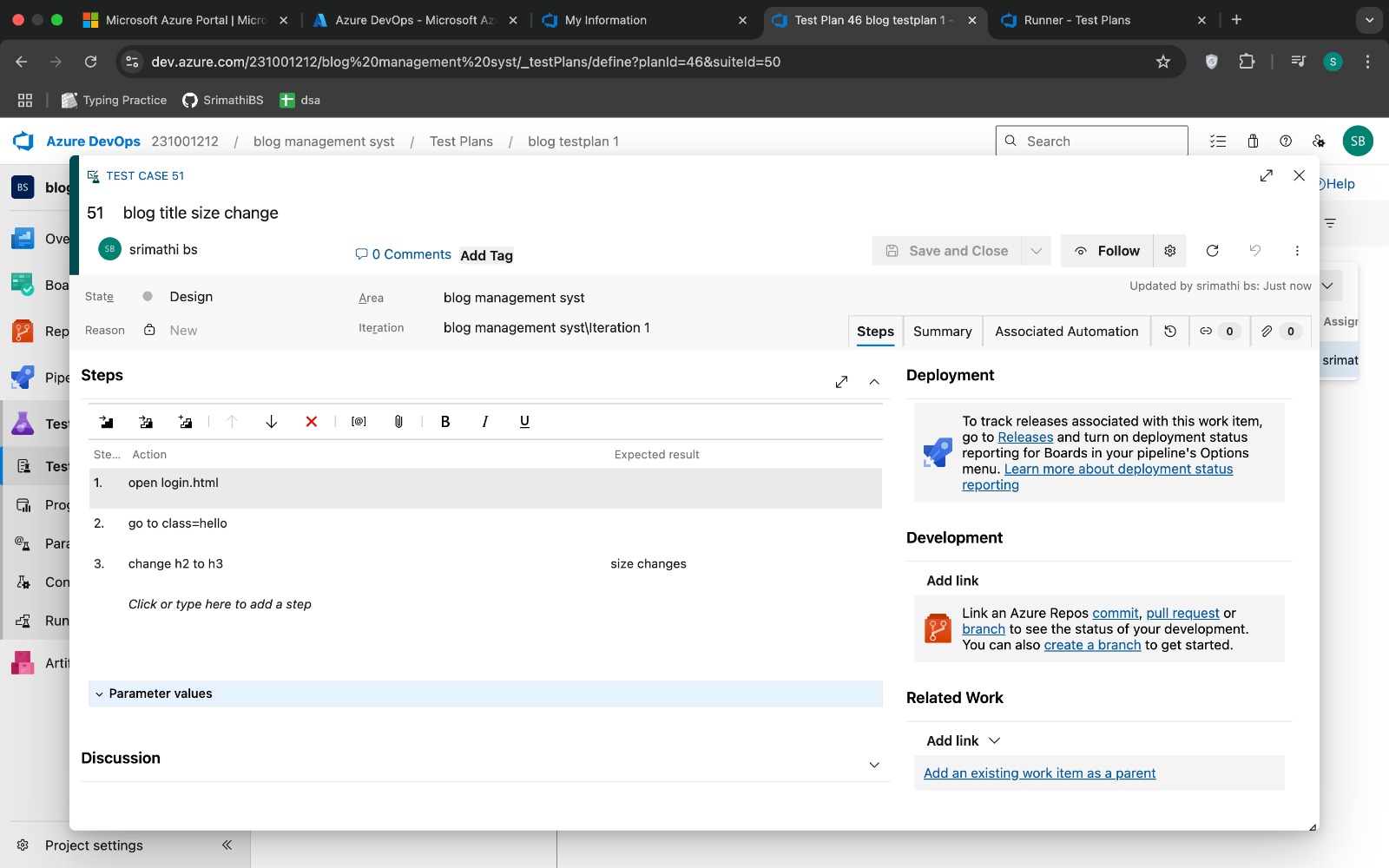
Make changes to the content.

Observe metadata updates.

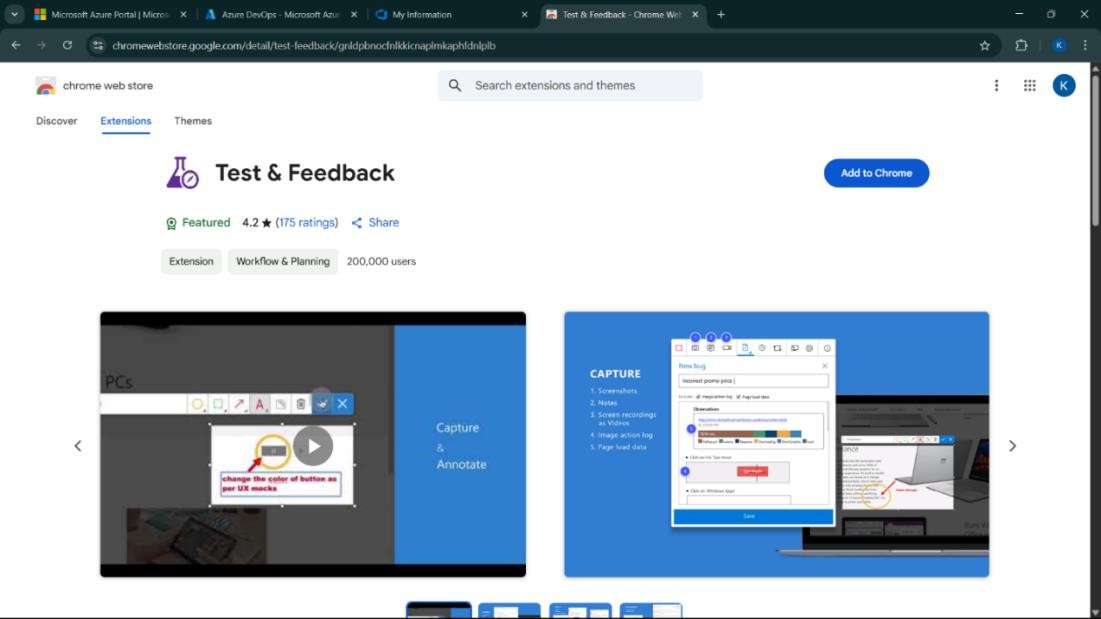
**Expected Results**:

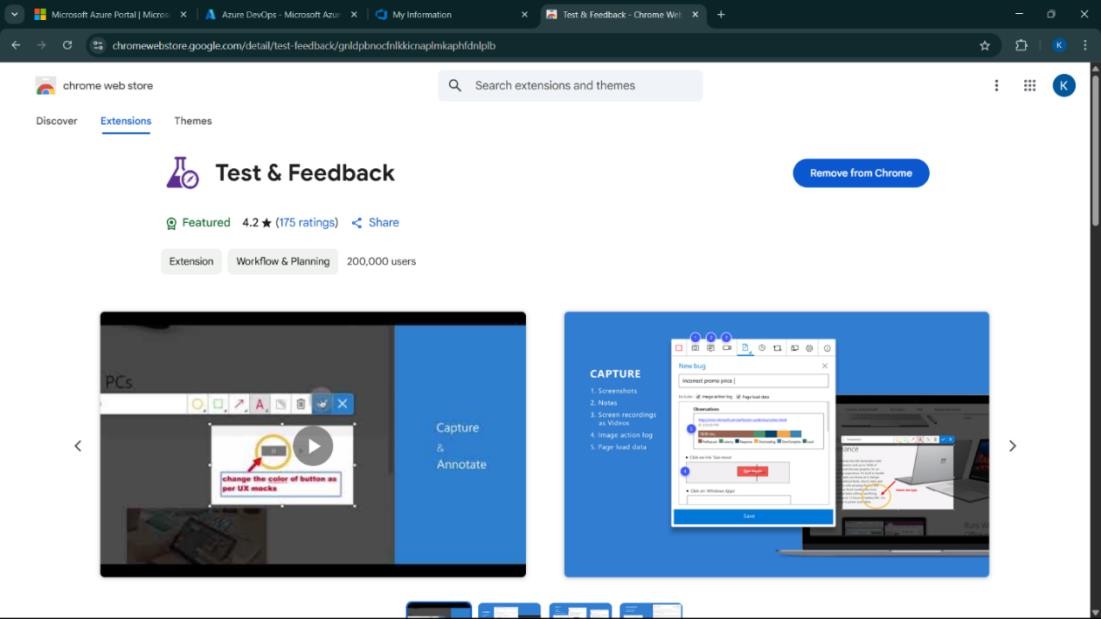
Metadata remains static or shows outdated information.

**Type**: Error Path

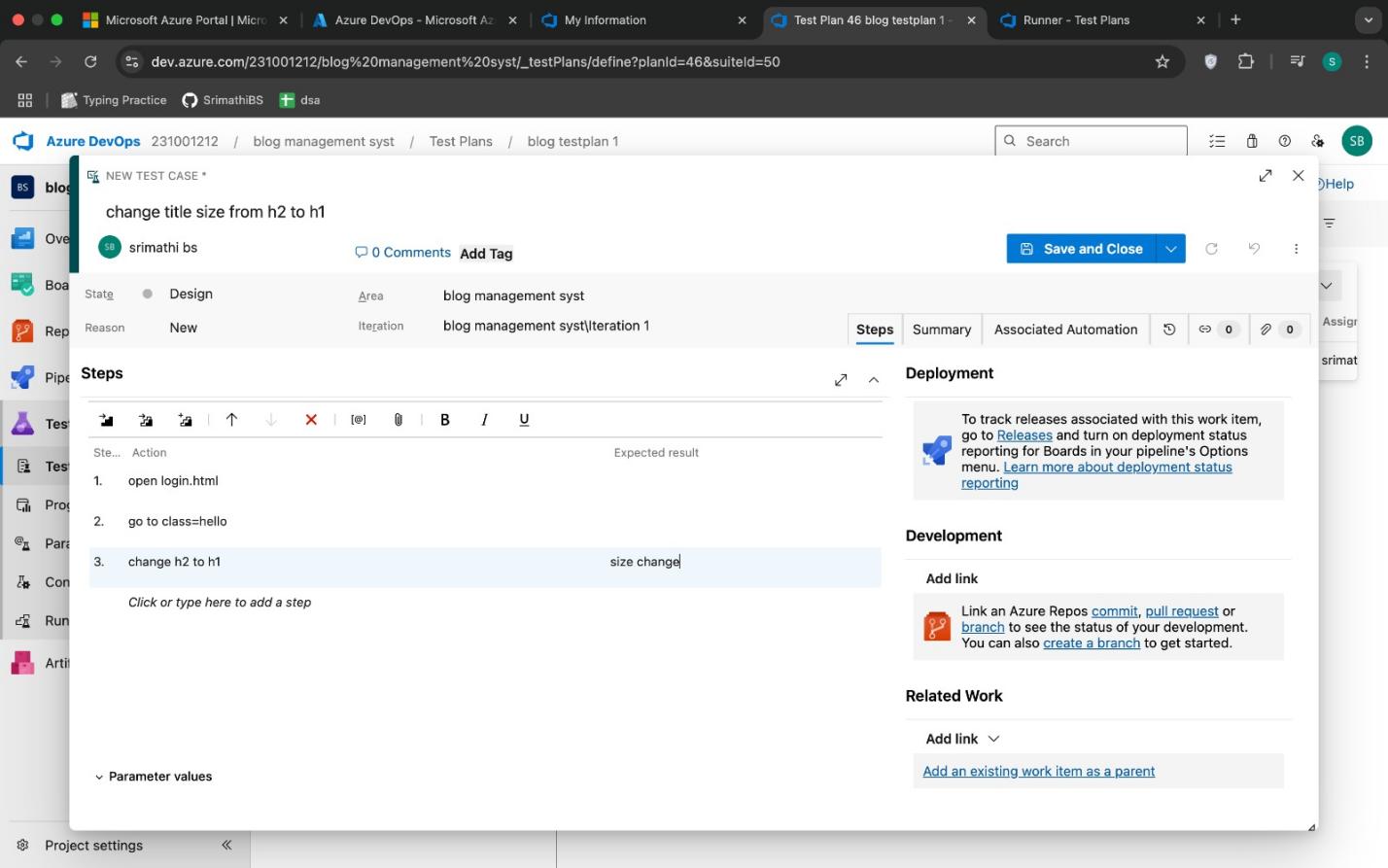


1. **Installation of test**

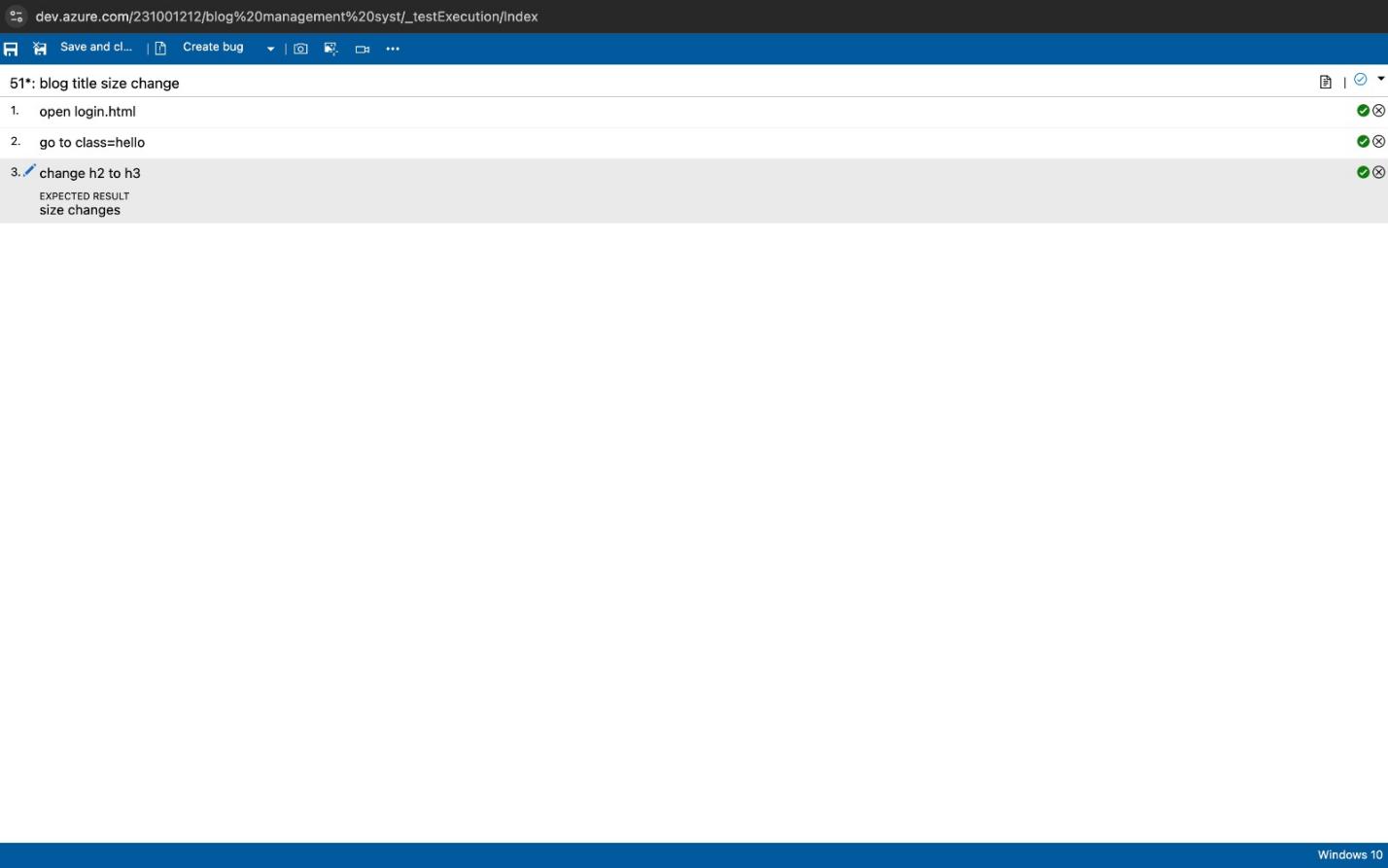
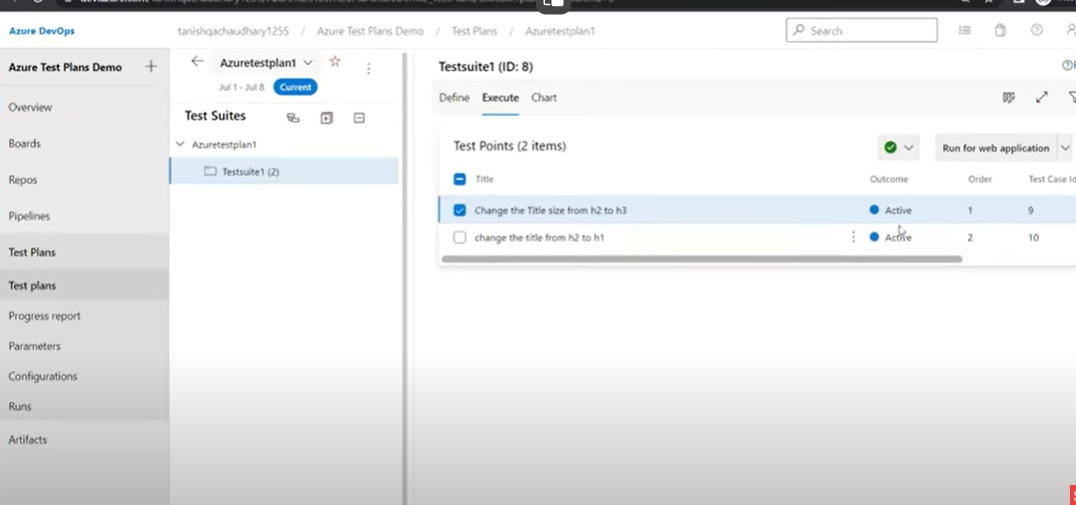
****



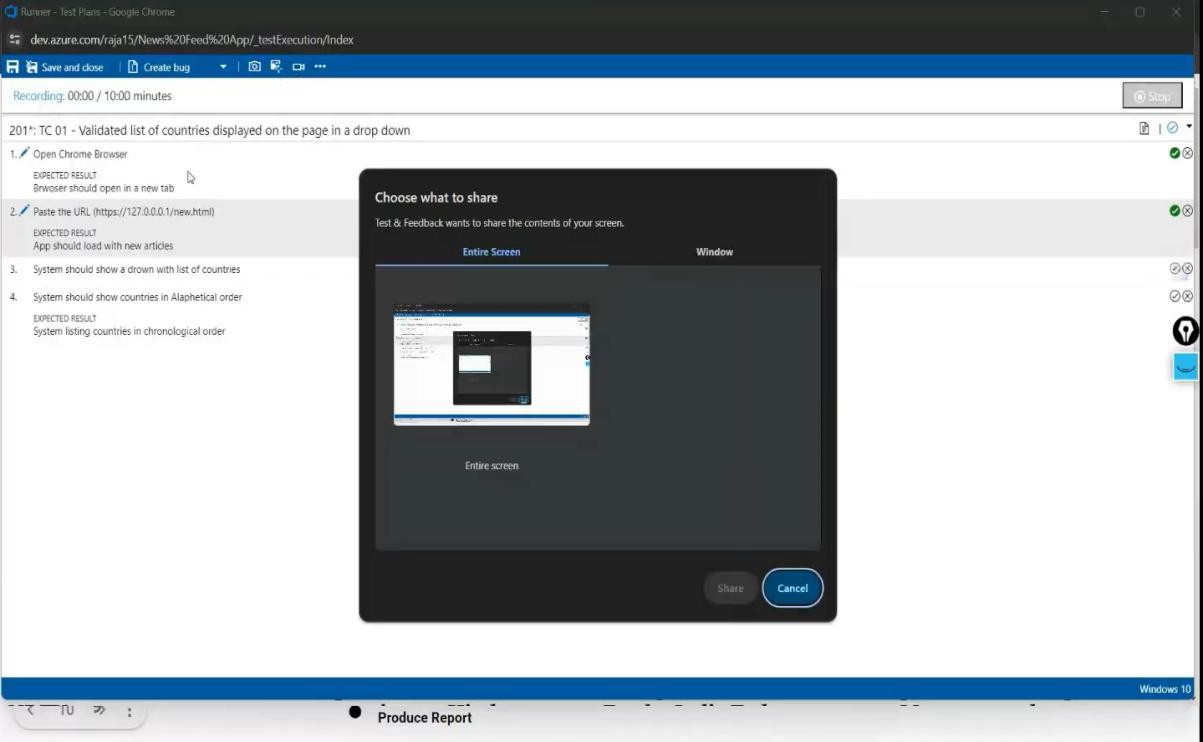
Test and feedback Showing it as an extension



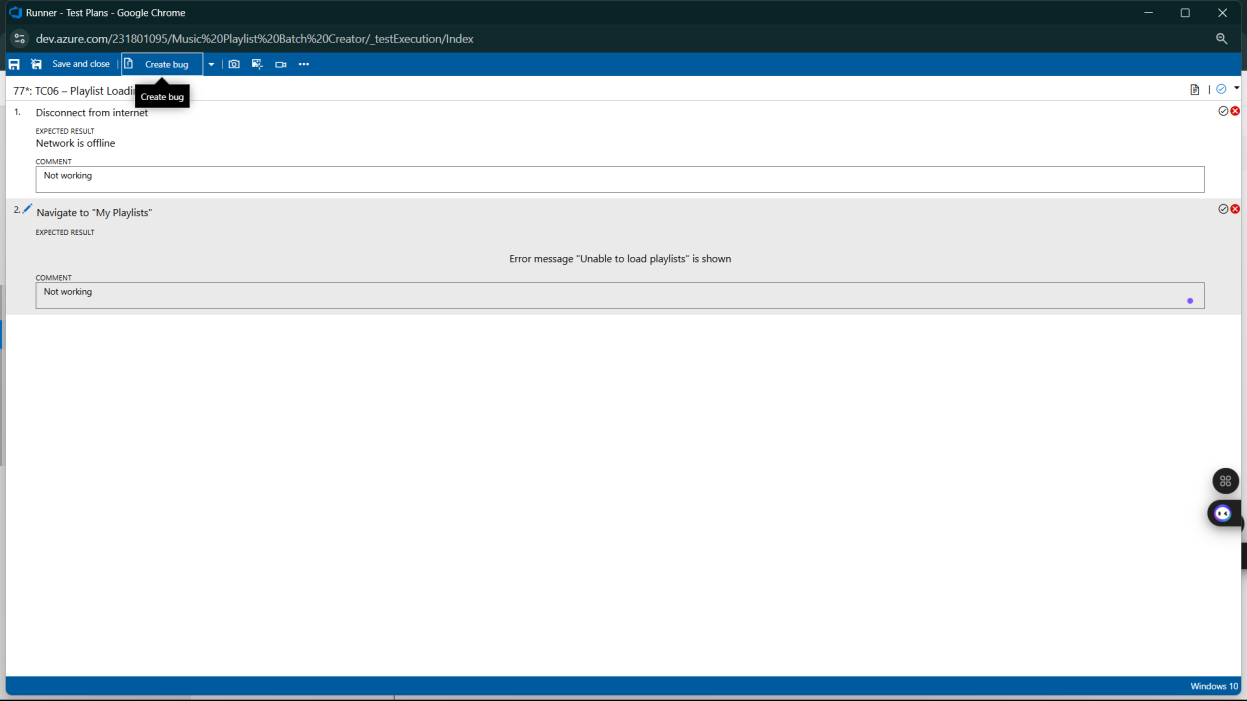
1. **Running the test cases**

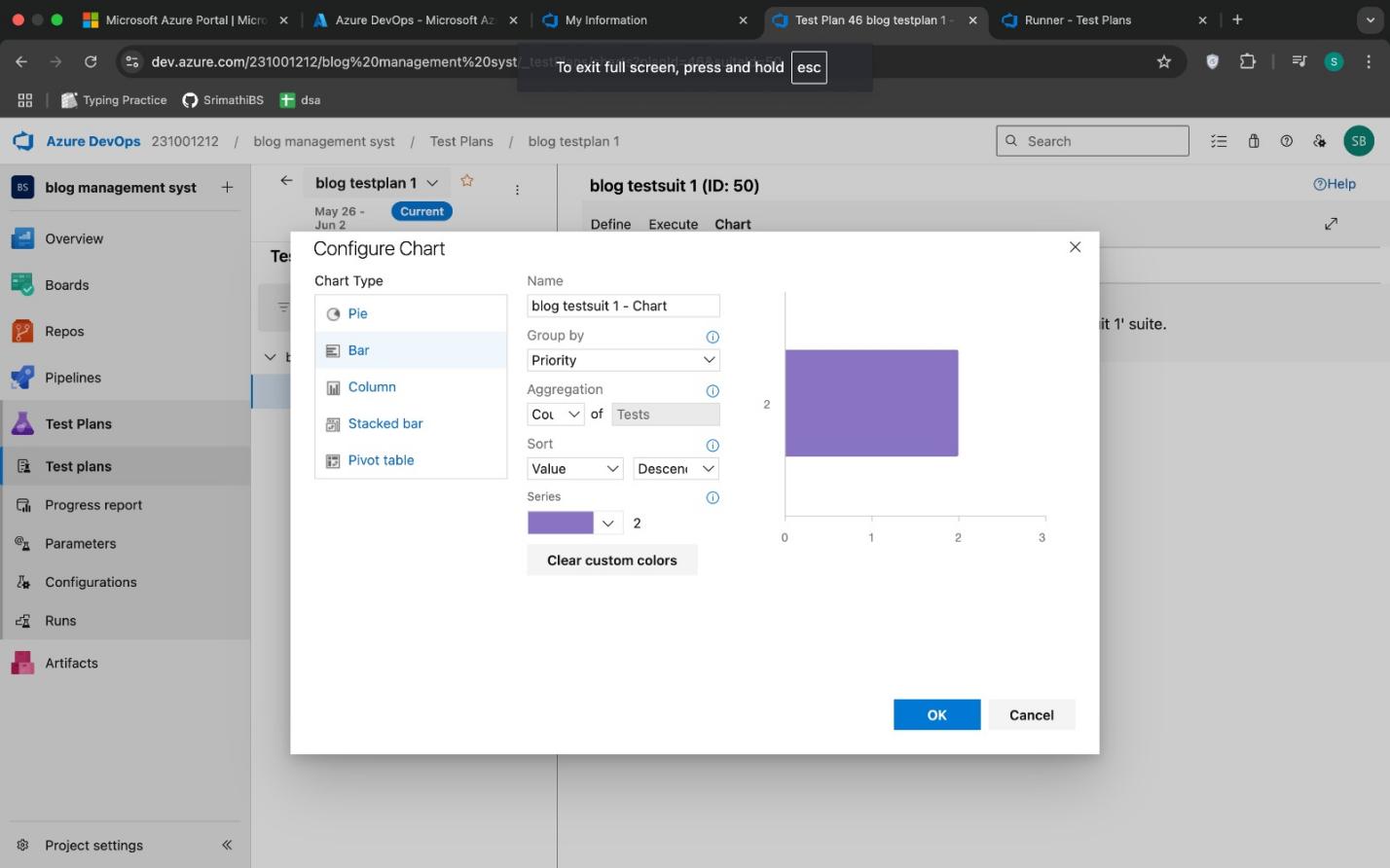
****

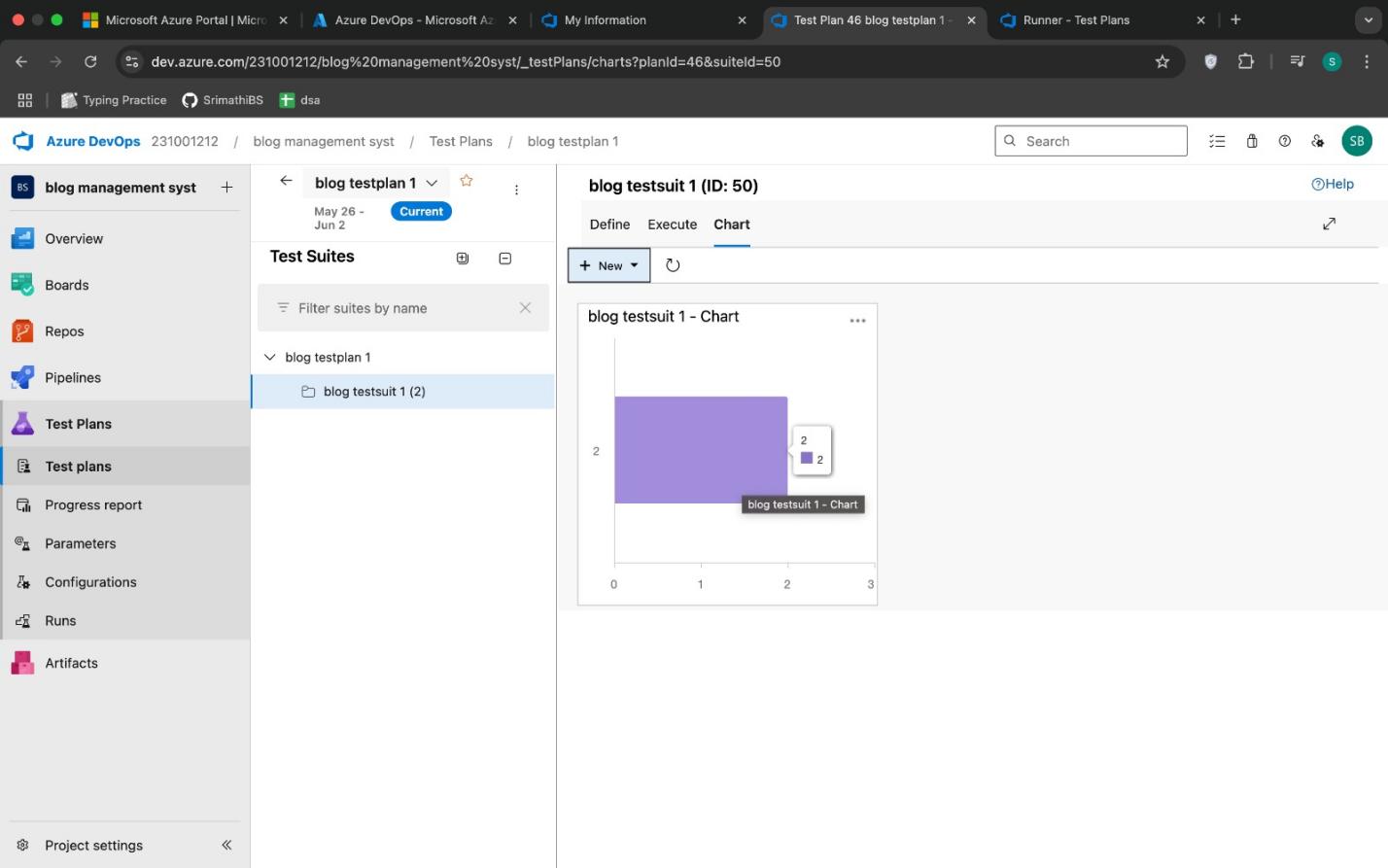
1. **Recording the test case**

****

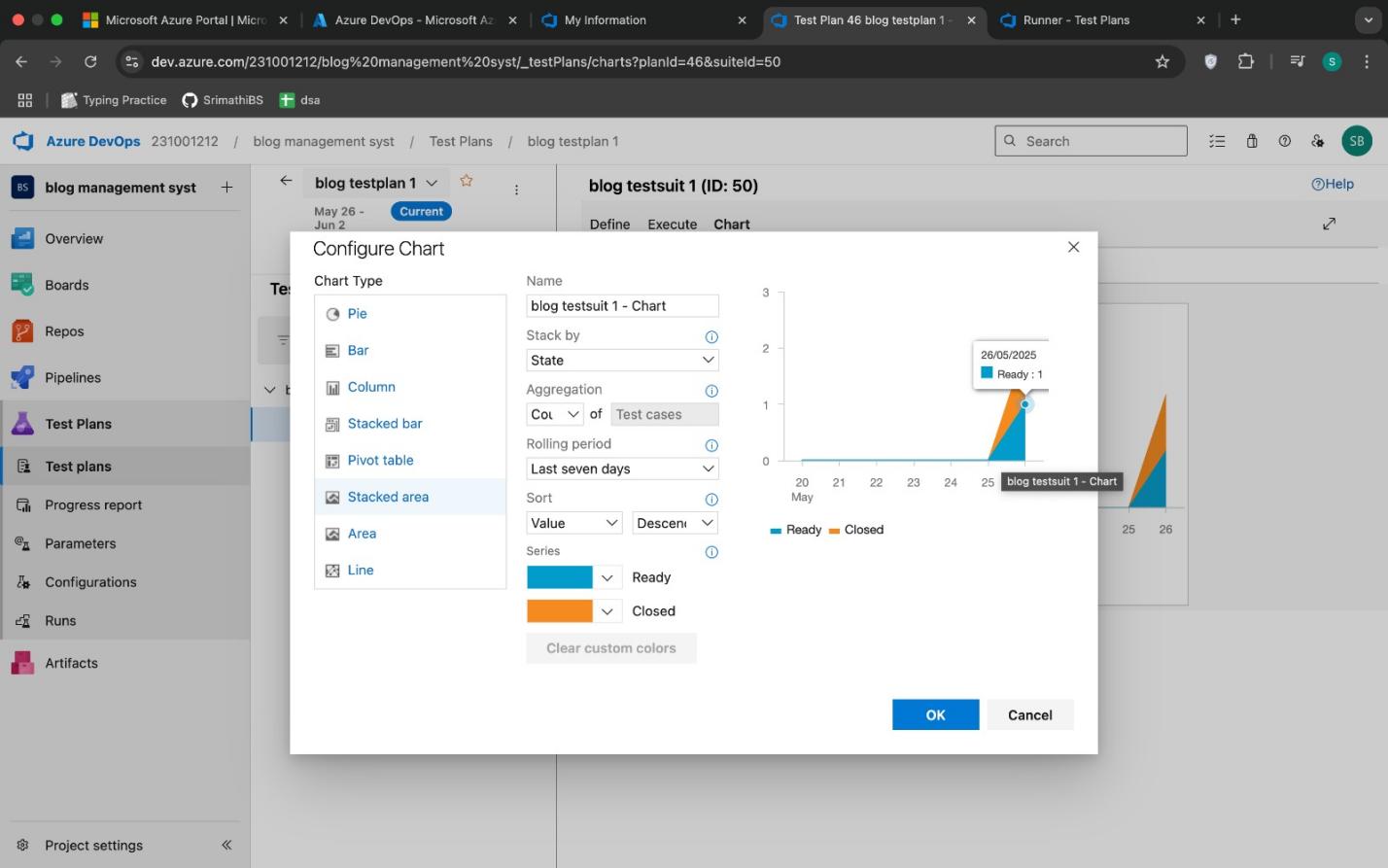
1. **Creating the bug**

****

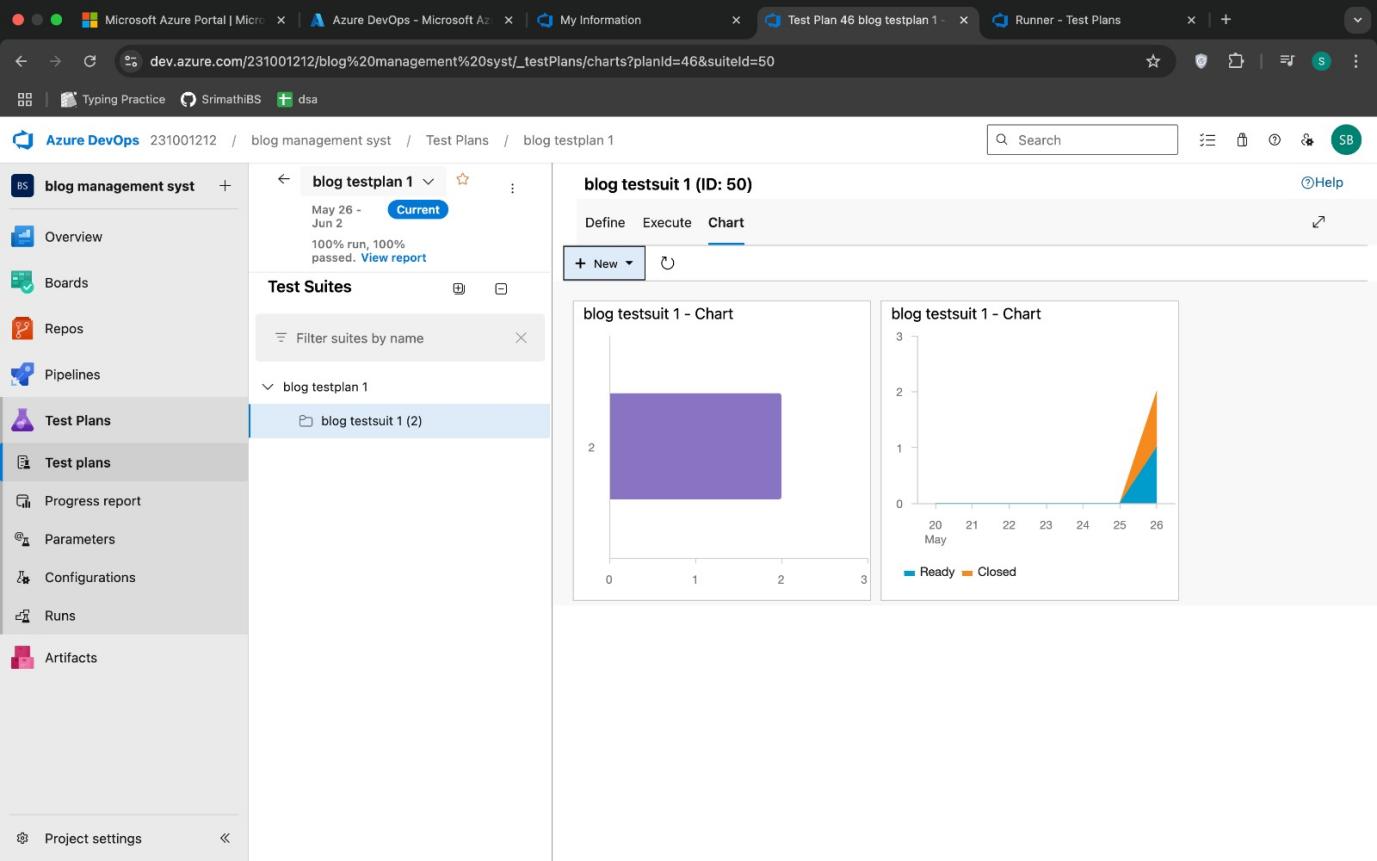


****

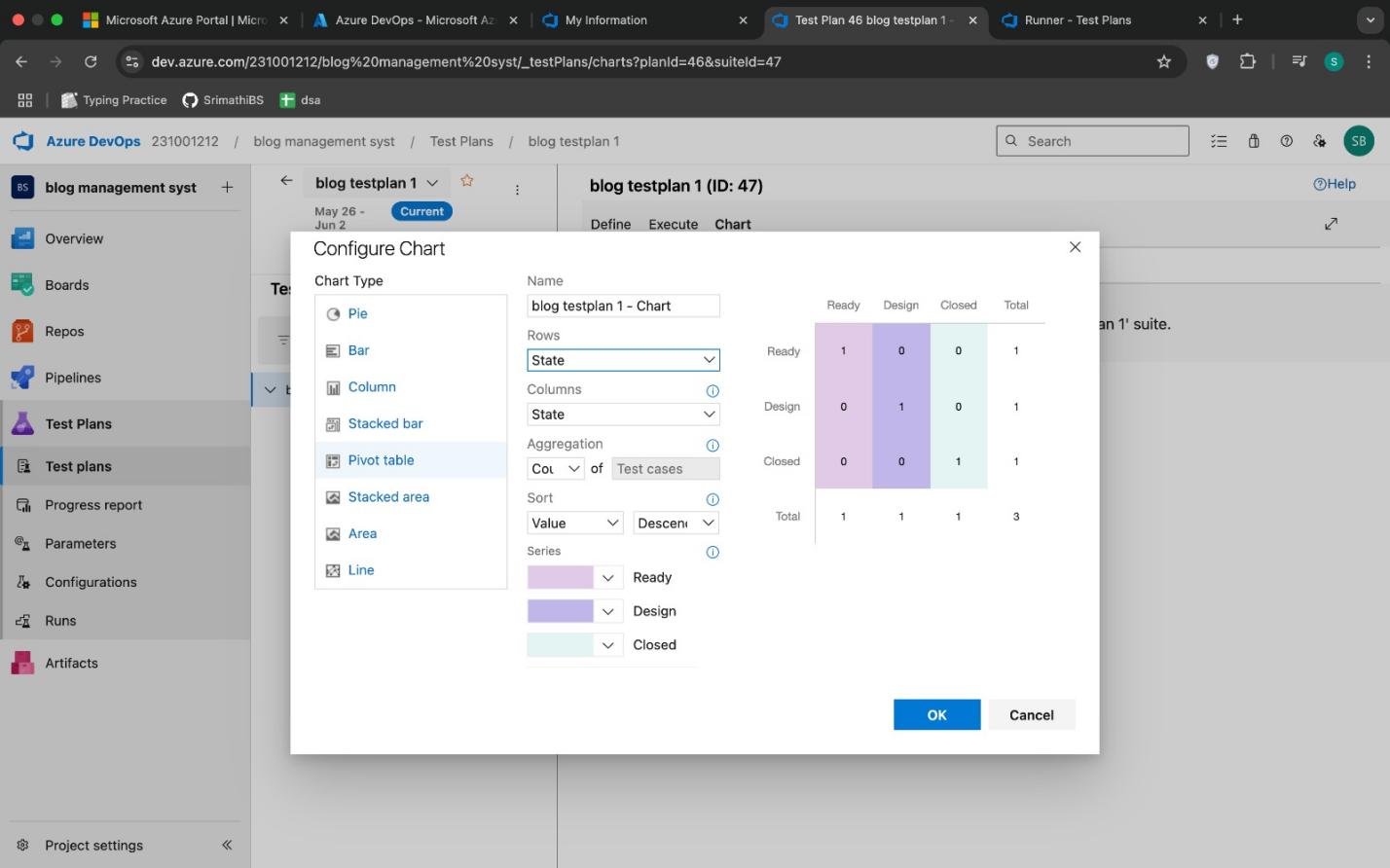
1. **Test case results**

****

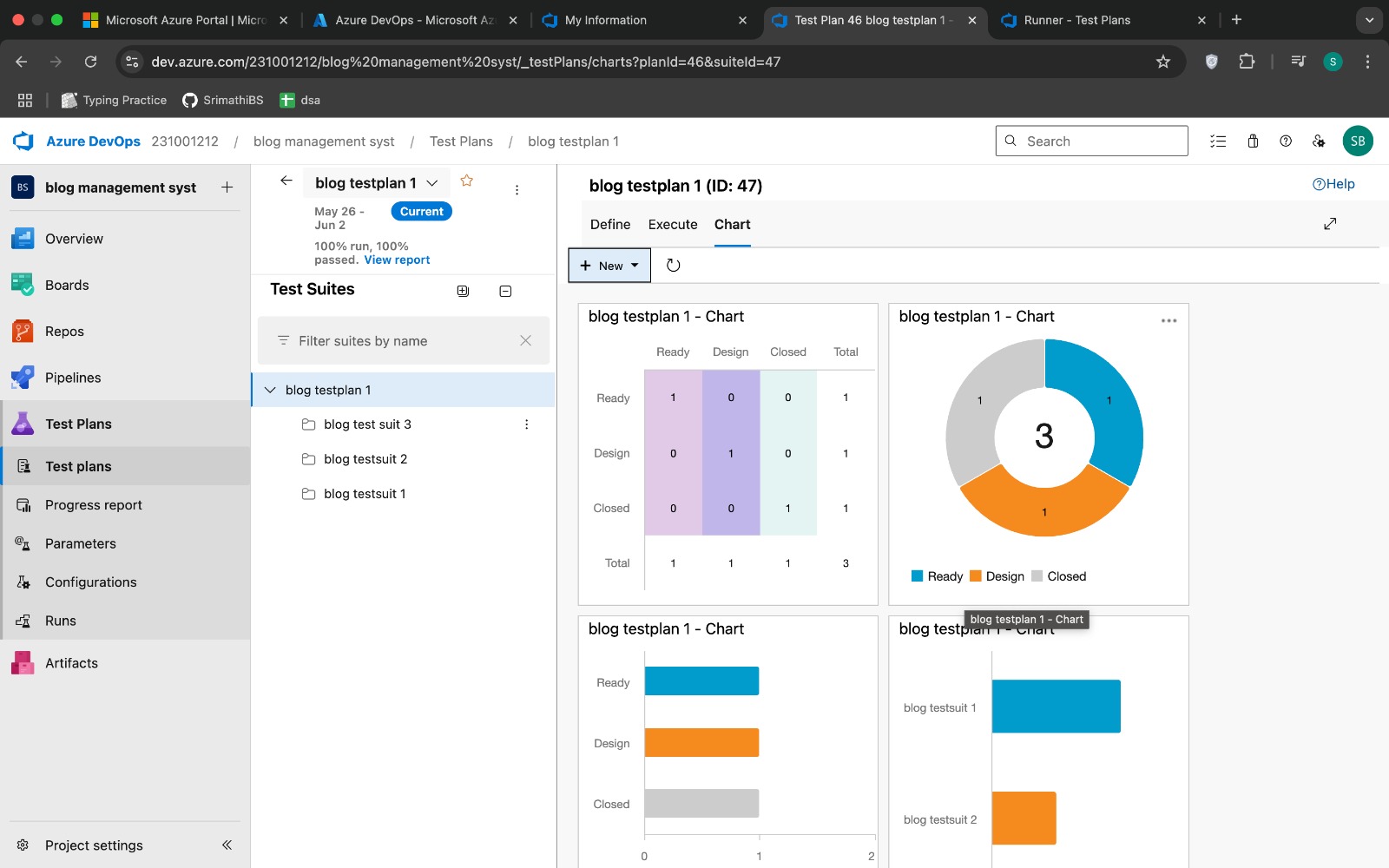
1. **Test report summary**

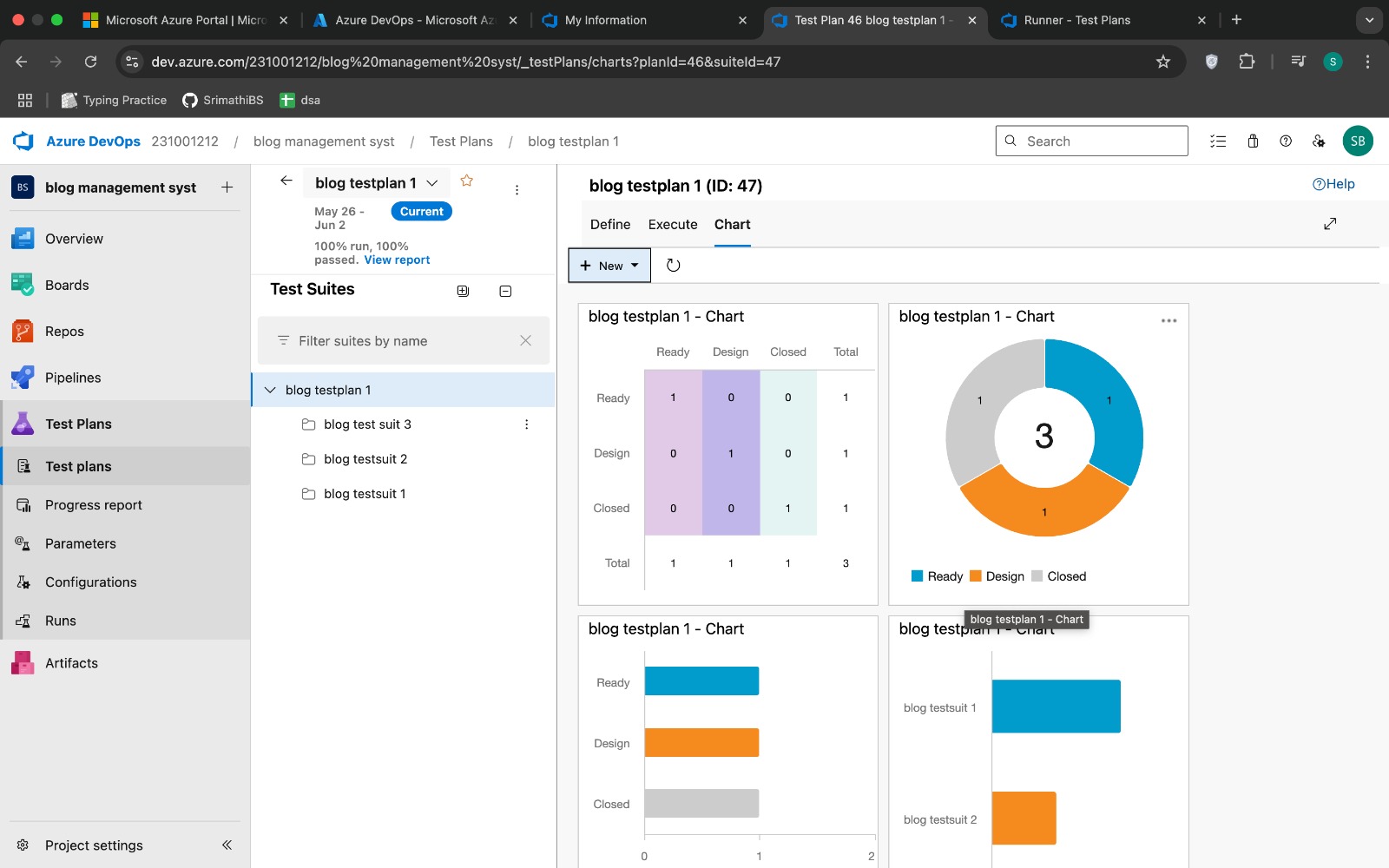
****

* + Assigning bug to the developer and changing state

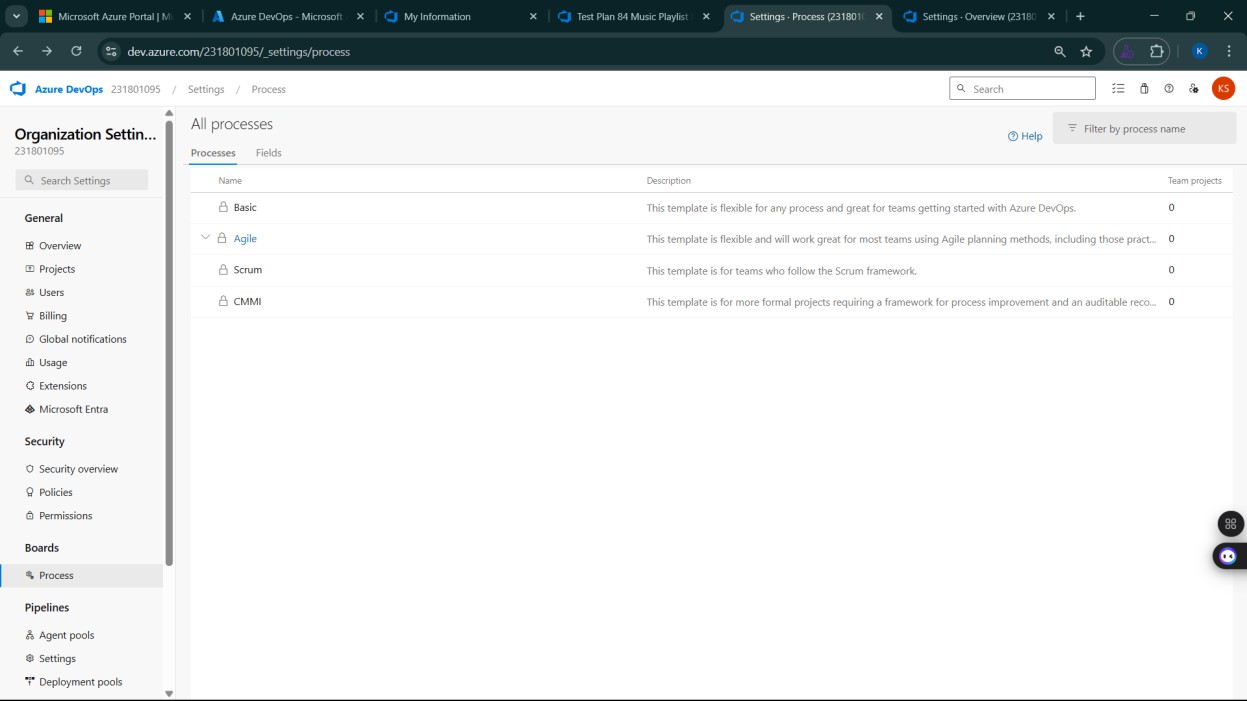


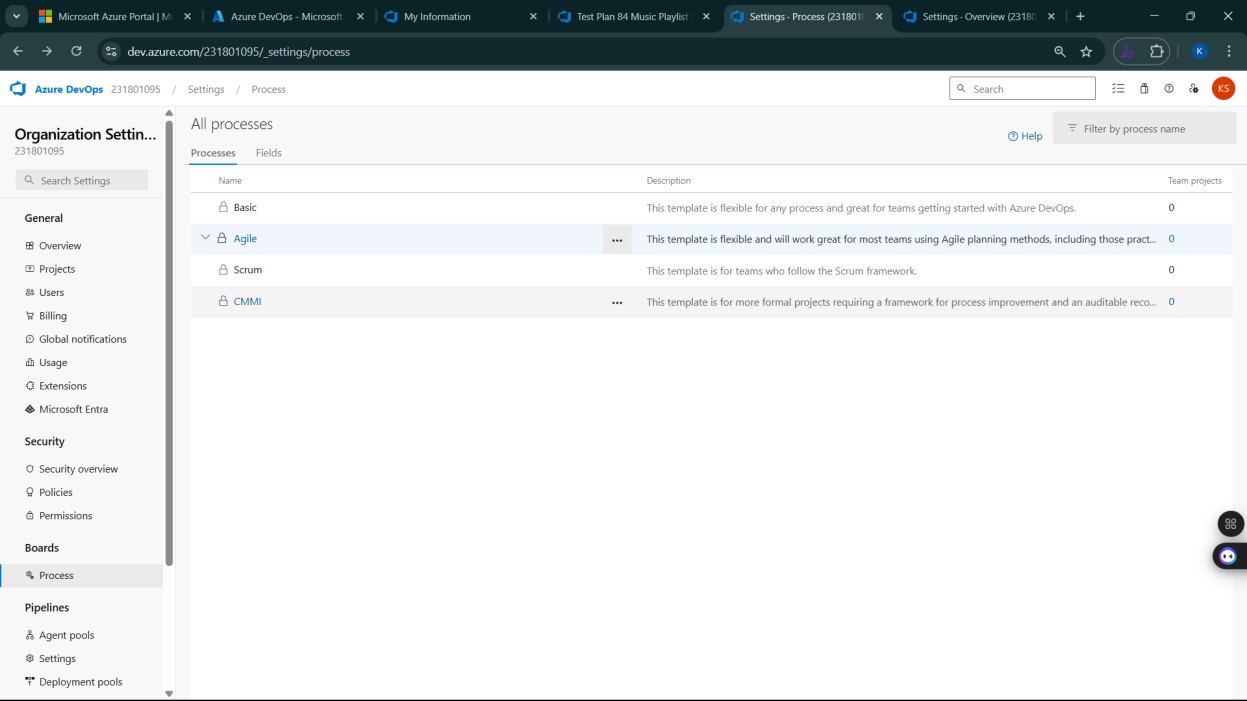
1. **Progress report**

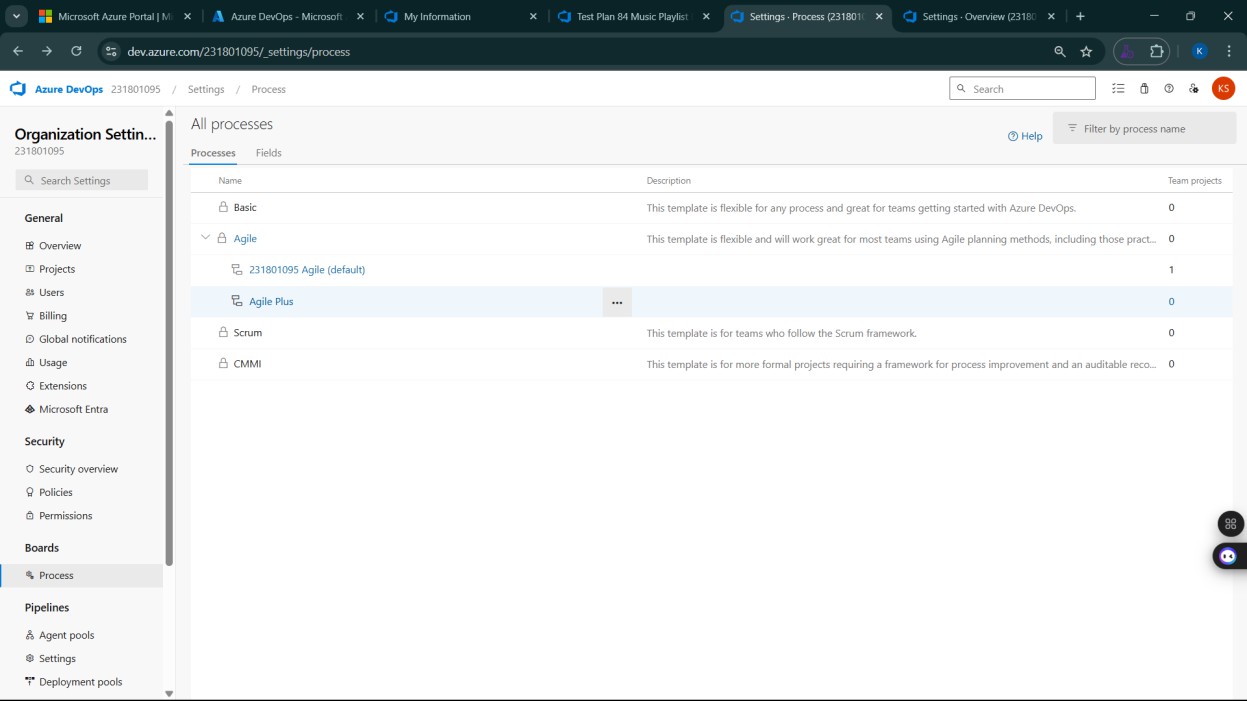
****



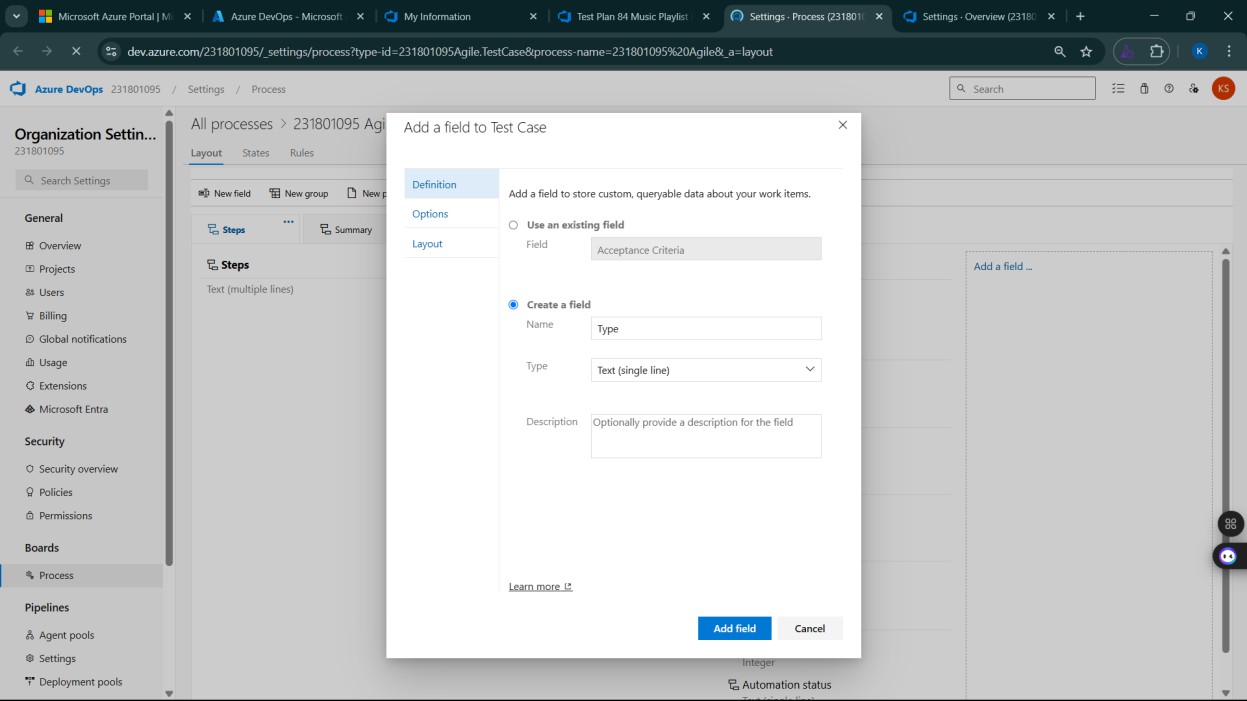
1. **Changing the test template**

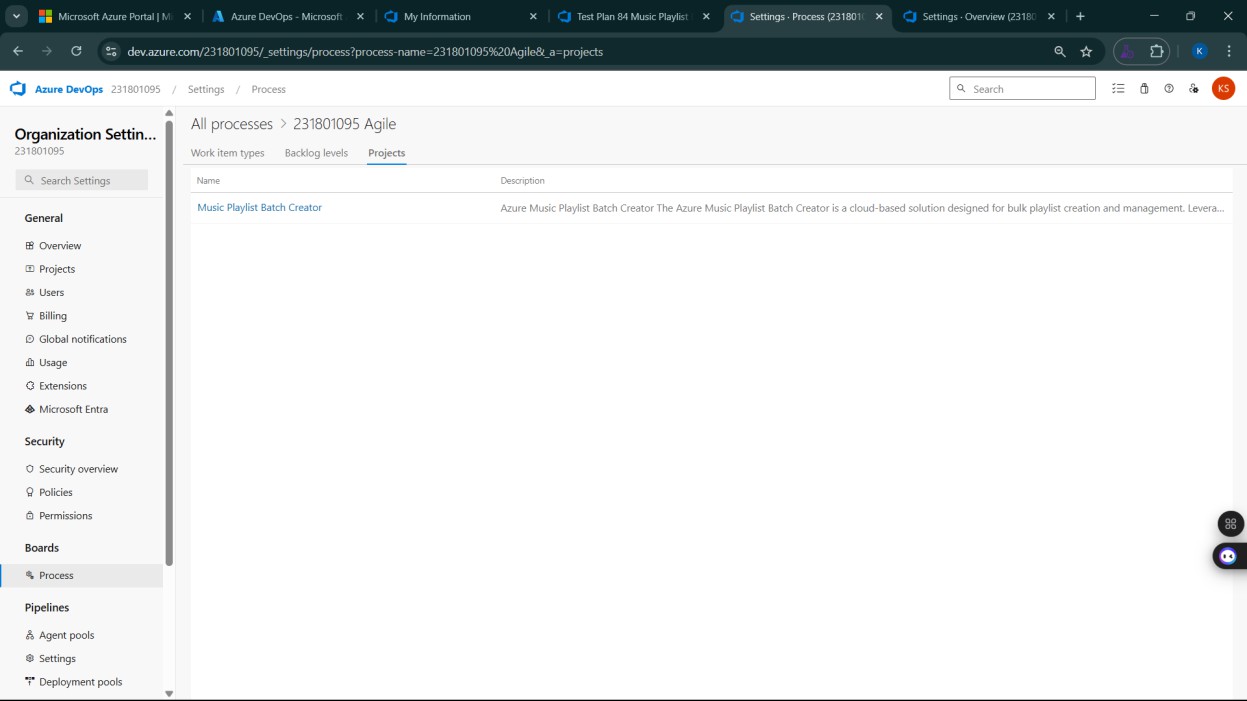
****

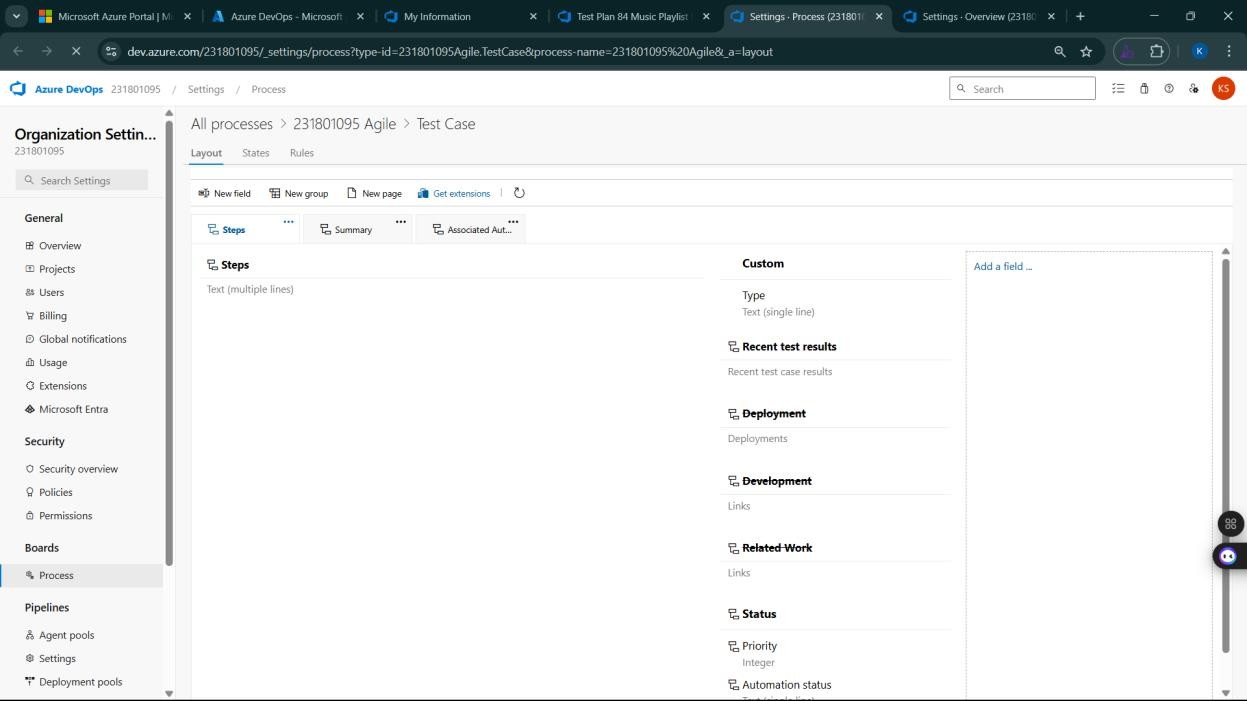


****

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.

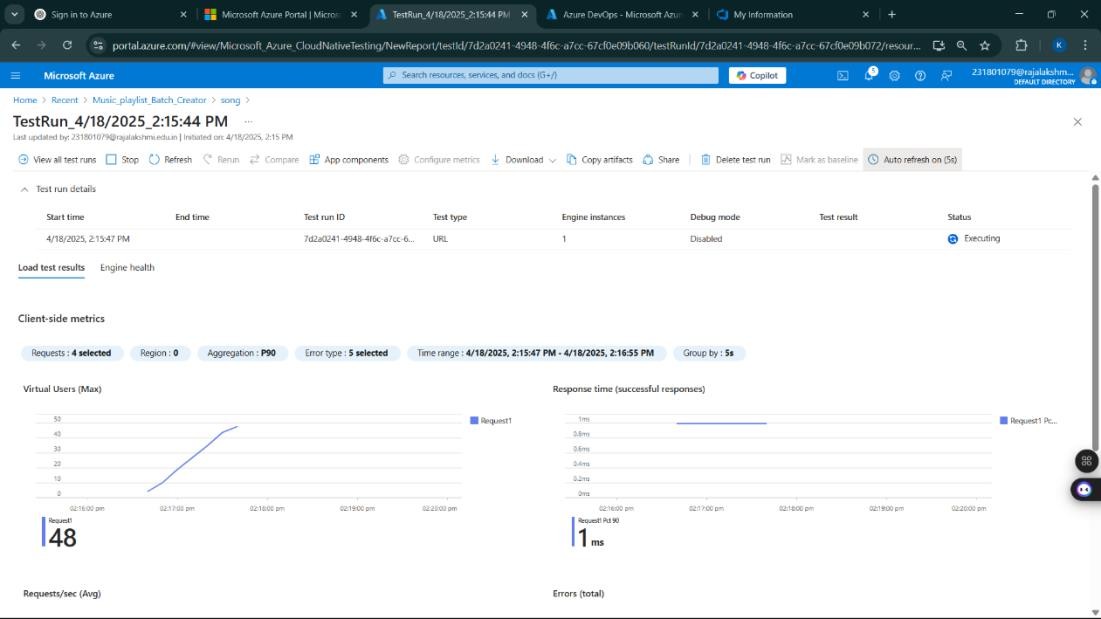
1. Create the Resource
   * Go to *Create a resource* → Search for “Azure Load Testing”.
   * Select Azure Load Testing and click Create.
2. Fill in the Configuration Details
   * Subscriptio*n:* Choose your Azure subscription.
   * *Resource Group:* Create new or select an existing one.
   * *Name:* 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.

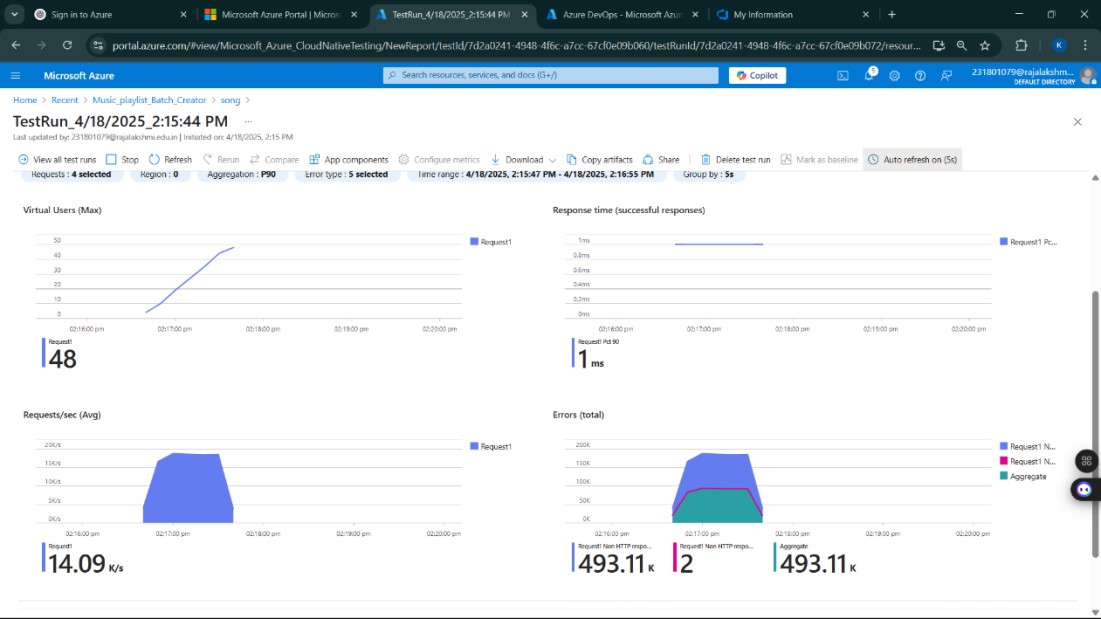
**Steps to Create and Run a Load Test:**

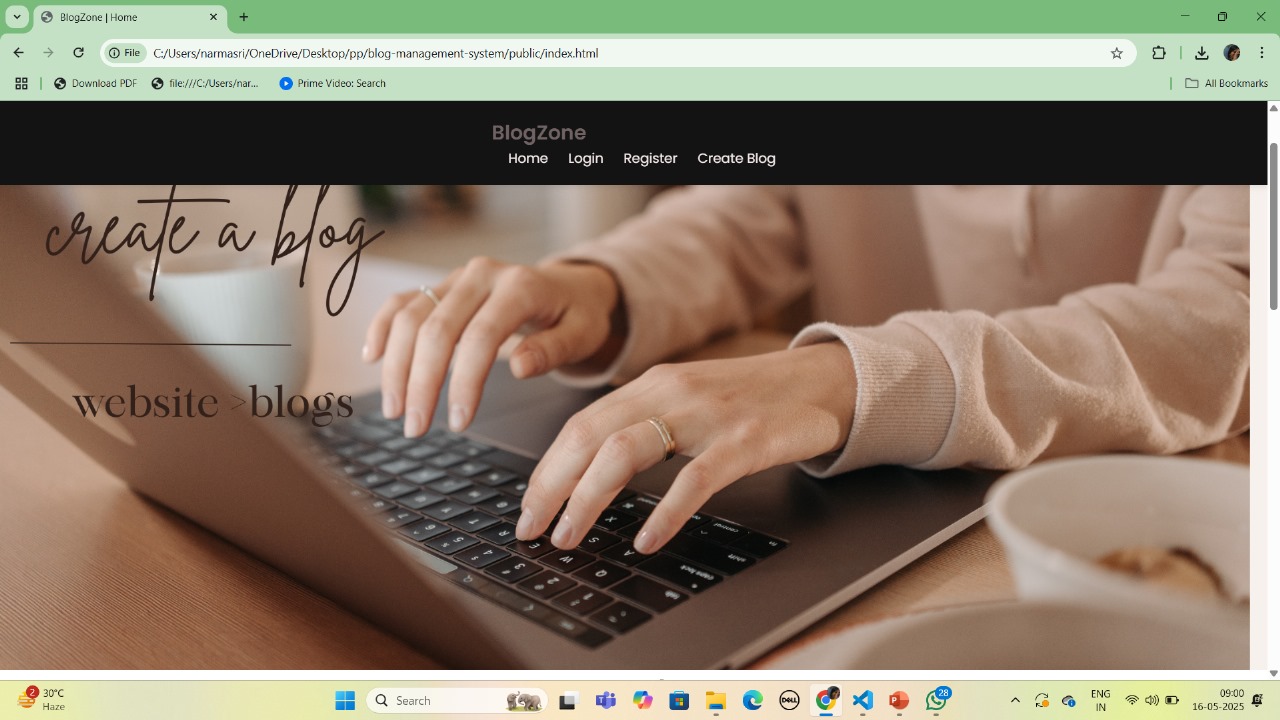
Once your resource is ready:

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**

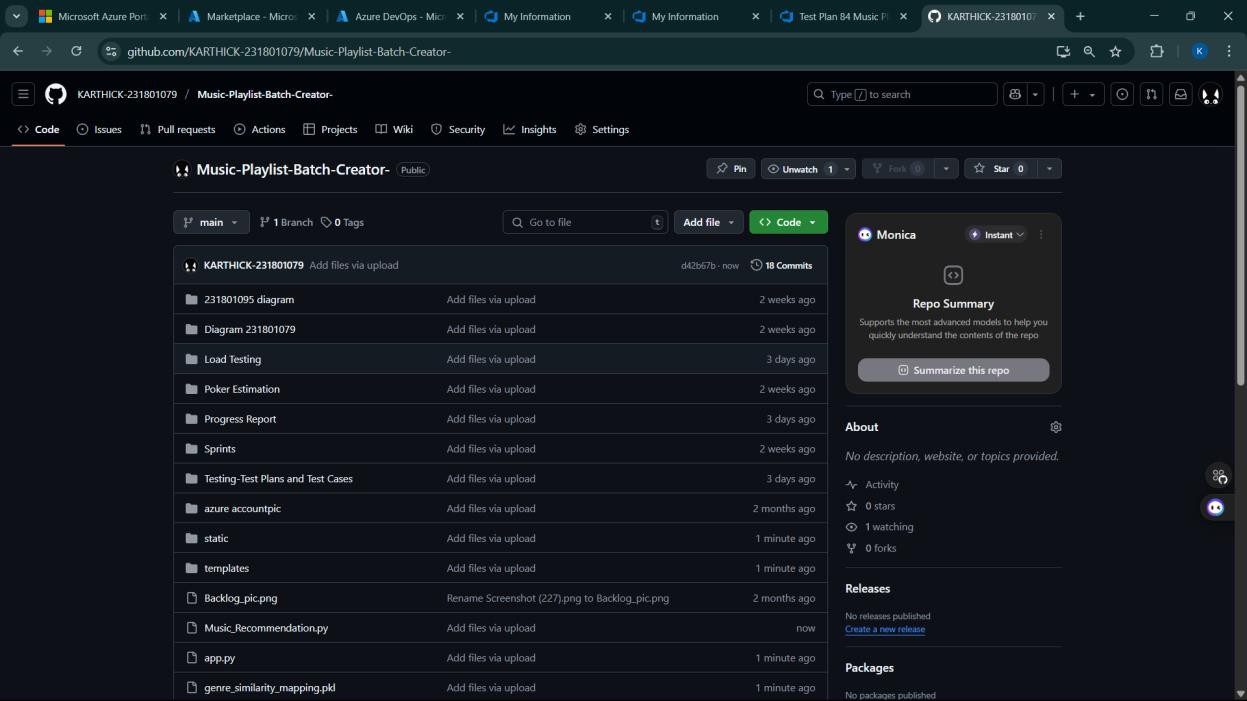
**EXP NO: 10**

**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 Music Playlist Batch Creator 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.