

**DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**

**LAB MANUAL**

**CS23432 – Software Construction**

**(REGULATION 2023)**

**RAJALAKSHMI ENGINEERING COLLEGE**

**Thandalam, Chennai-602015**

Name: Nithesha.J.S

Register No: 231801120

Year / Branch / Section: 2nd / AI&DS /AC

Semester: IV

Academic Year: 2024 - 2025

**INDEX**

|  |  |  |
| --- | --- | --- |
| S.No. | Date | Title |
| 1. | 03/2/25 | Azure Devops Environment Setup. |
| 2. | 04/2/25 | Azure Devops Project Setup and User Story Management. |
| 3. | 18/2/25 | Setting Up Epics, Features, And User Stories for Project Planning. |
| 4. | 06/2/25 | Sprint Planning. |
| 5. | 20/3/25 | Poker Estimation. |
| 6. | 27/3/25 | Designing Class and Sequence Diagrams for Project Architecture. |
| 7. | 03/4/25 | Designing Architectural and ER Diagrams for Project Structure. |
| 8. | 10/4/25 | Testing – Test Plans and Test Cases. |
| 9. | 17/4/25 | Load Testing and Pipelines. |
| 10. | 24/4/25 | GitHub: Project Structure & Naming Conventions. |

|  |  |
| --- | --- |
| **EXP NO:** **1** | **AZURE DEVOPS ENVIRONMENT SETUP** |

**Aim:**

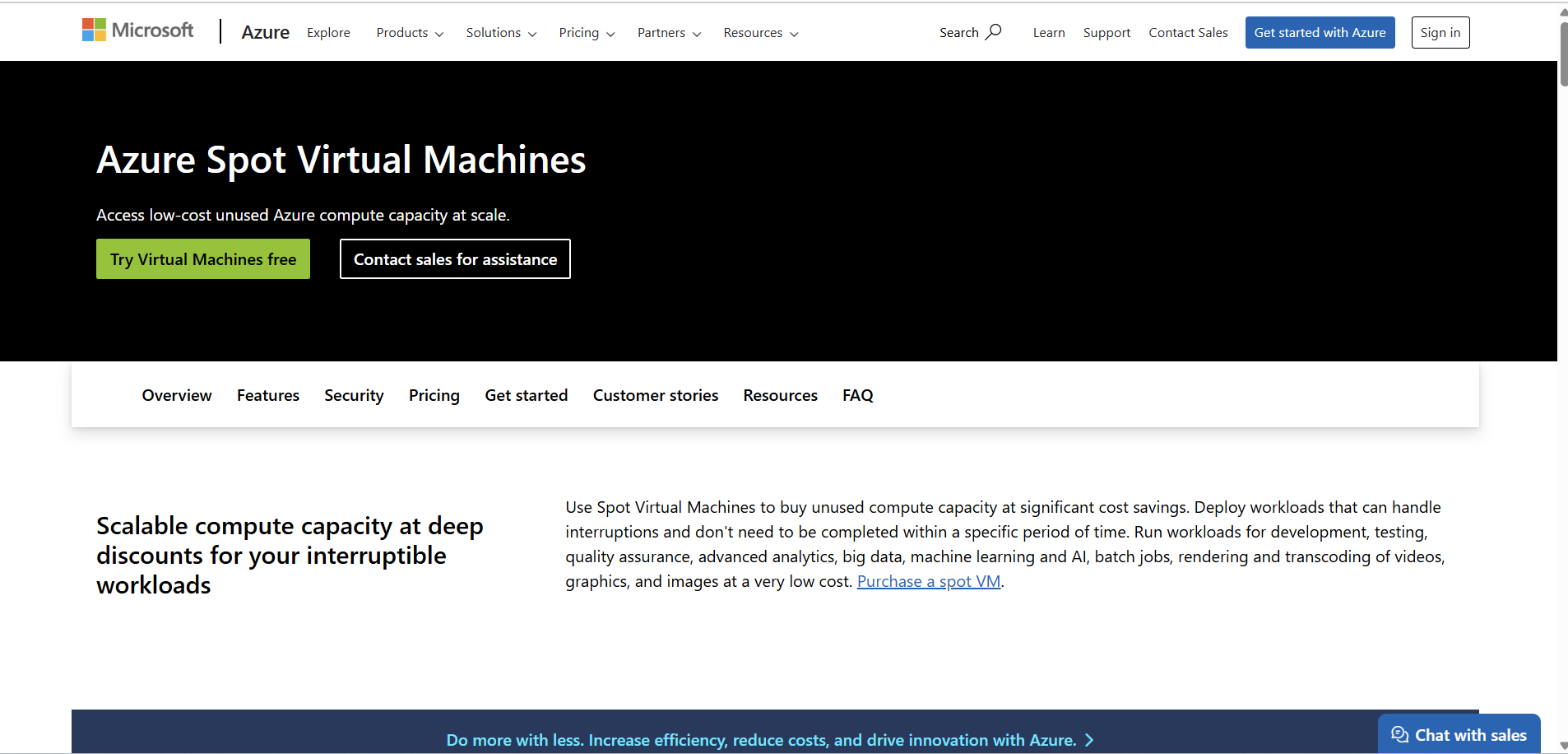
To set up and access the Azure DevOps environment by creating an organization through the Azure portal.

**INSTALLATION**

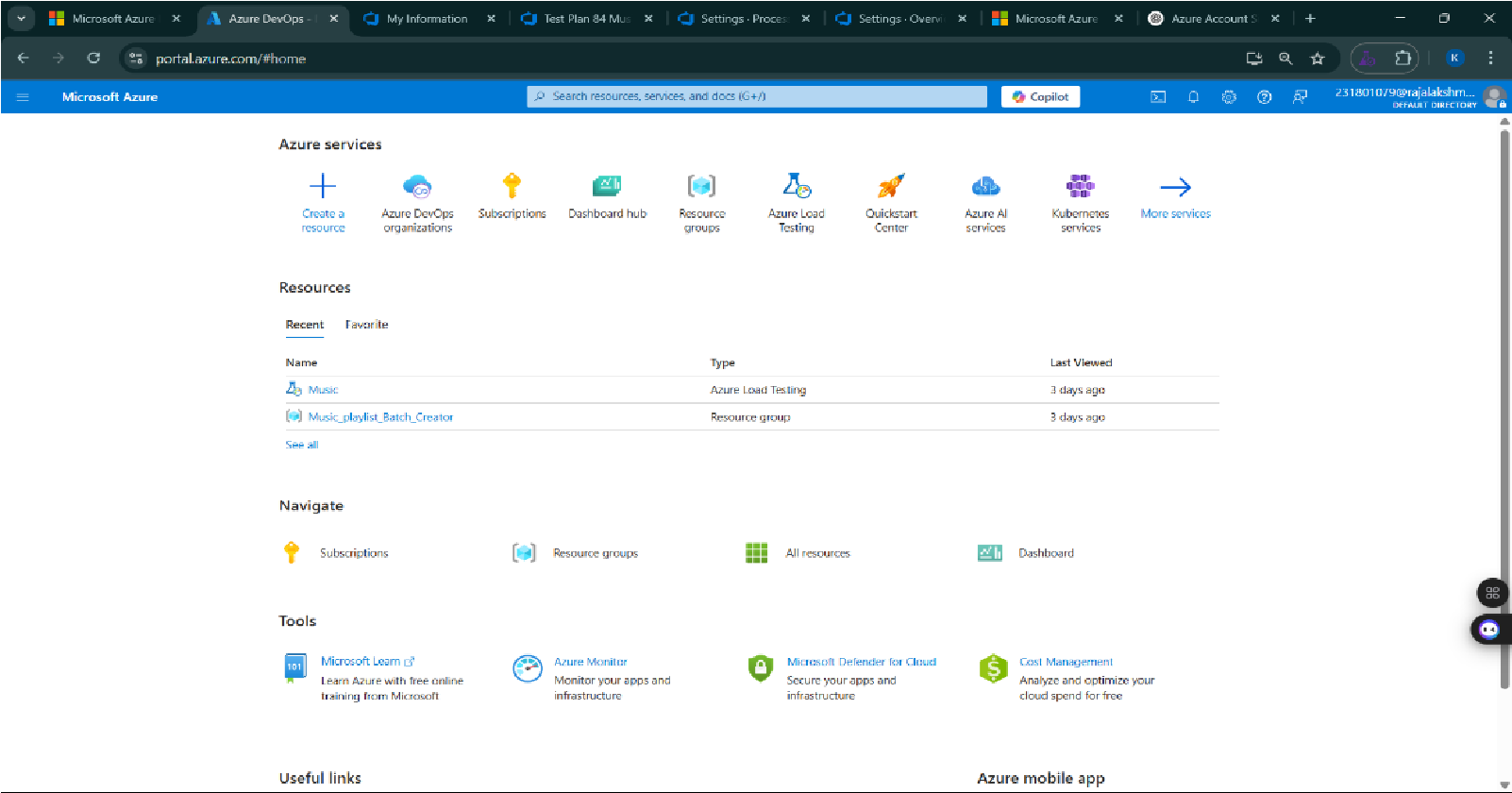
1.Open your web browser and go to the Azure website[: https://azure.microsoft.com/en-us/getstarted/azure-portal.](https://azure.microsoft.com/en-us/get-started/azure-portal)

Sign in using your Microsoft account credentials.

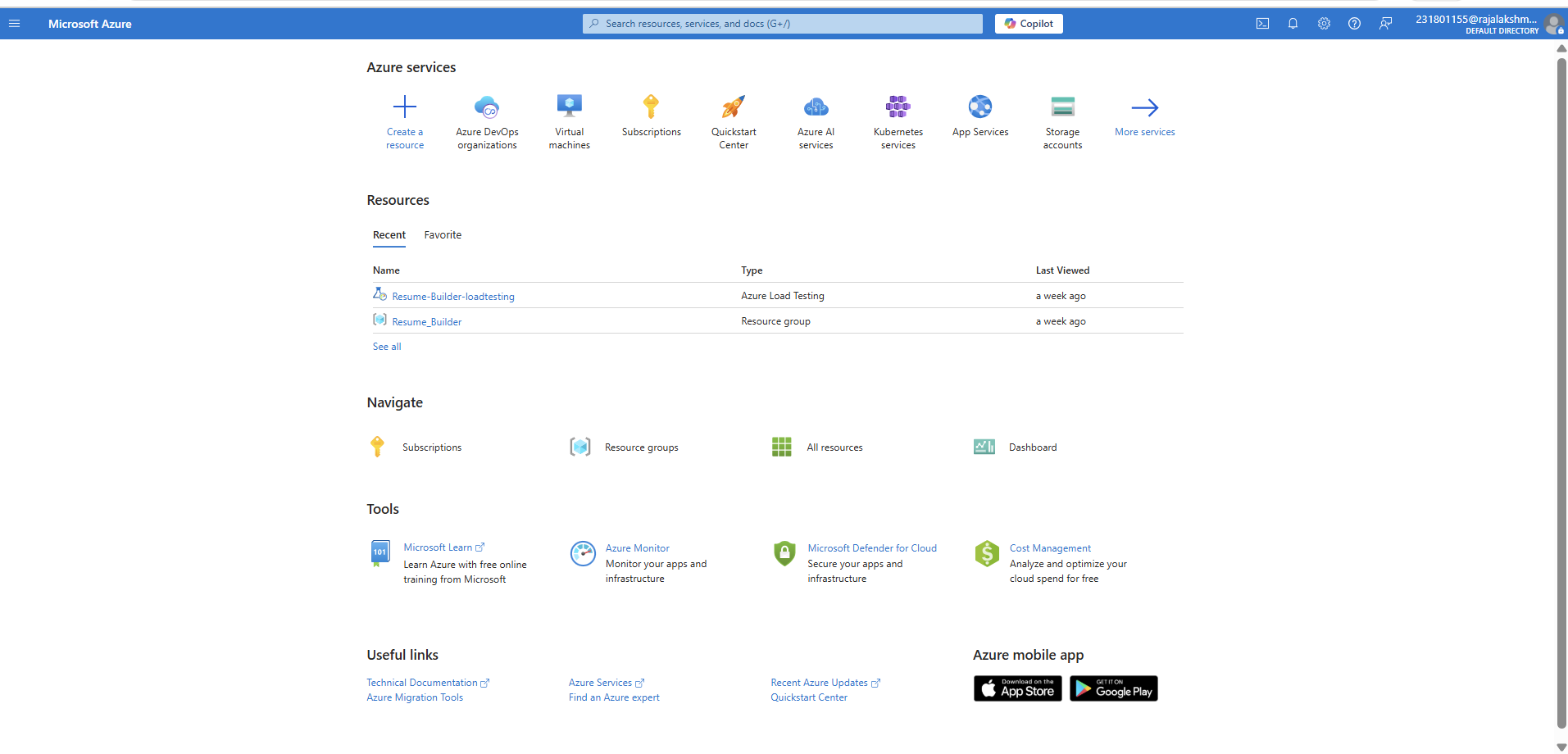
If you don't have a Microsoft account, you can create one her[e: https://signup.live.com/?lic=1](https://signup.live.com/?lic=1)



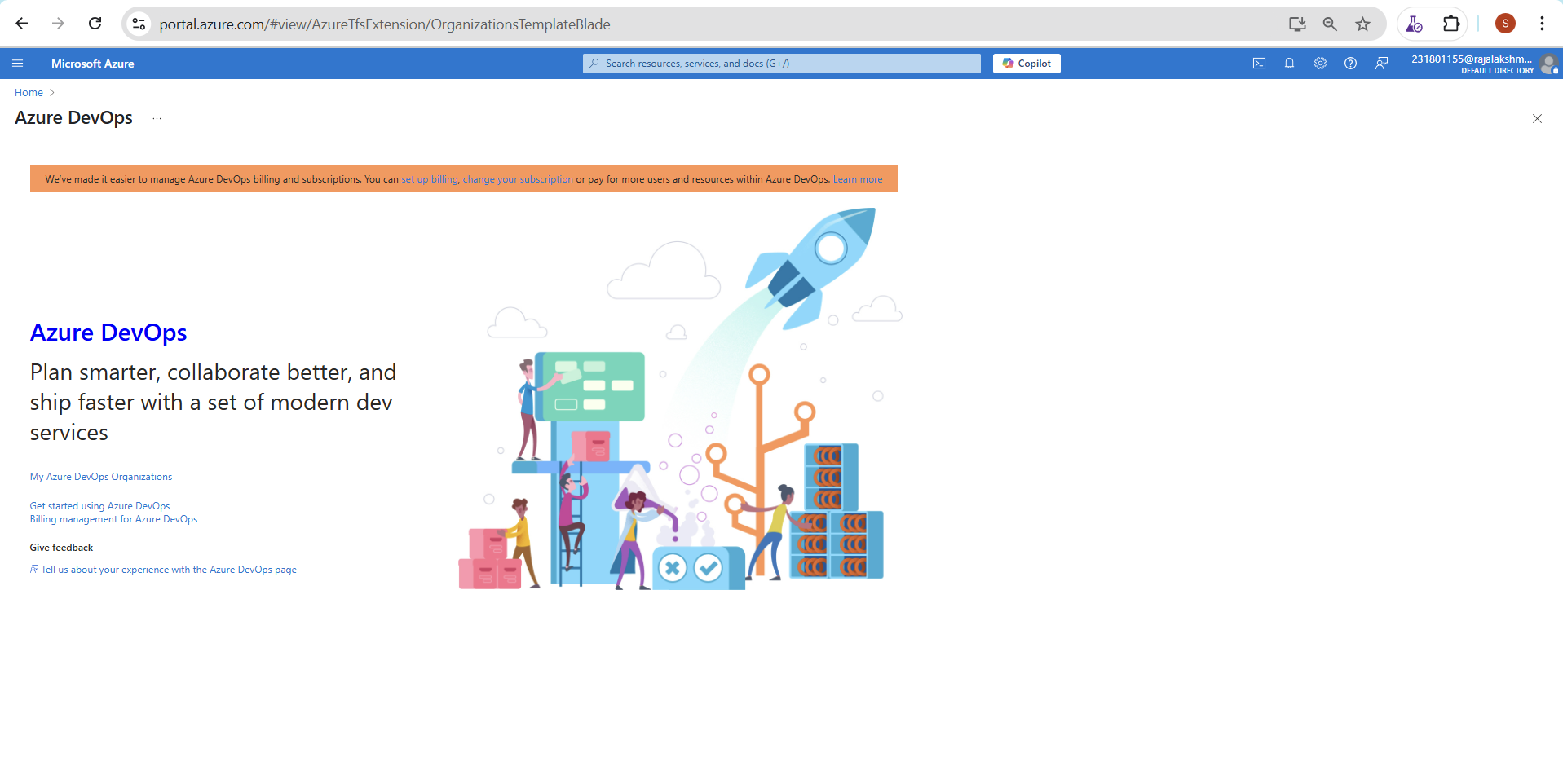
2.Azure home page



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



4.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:**

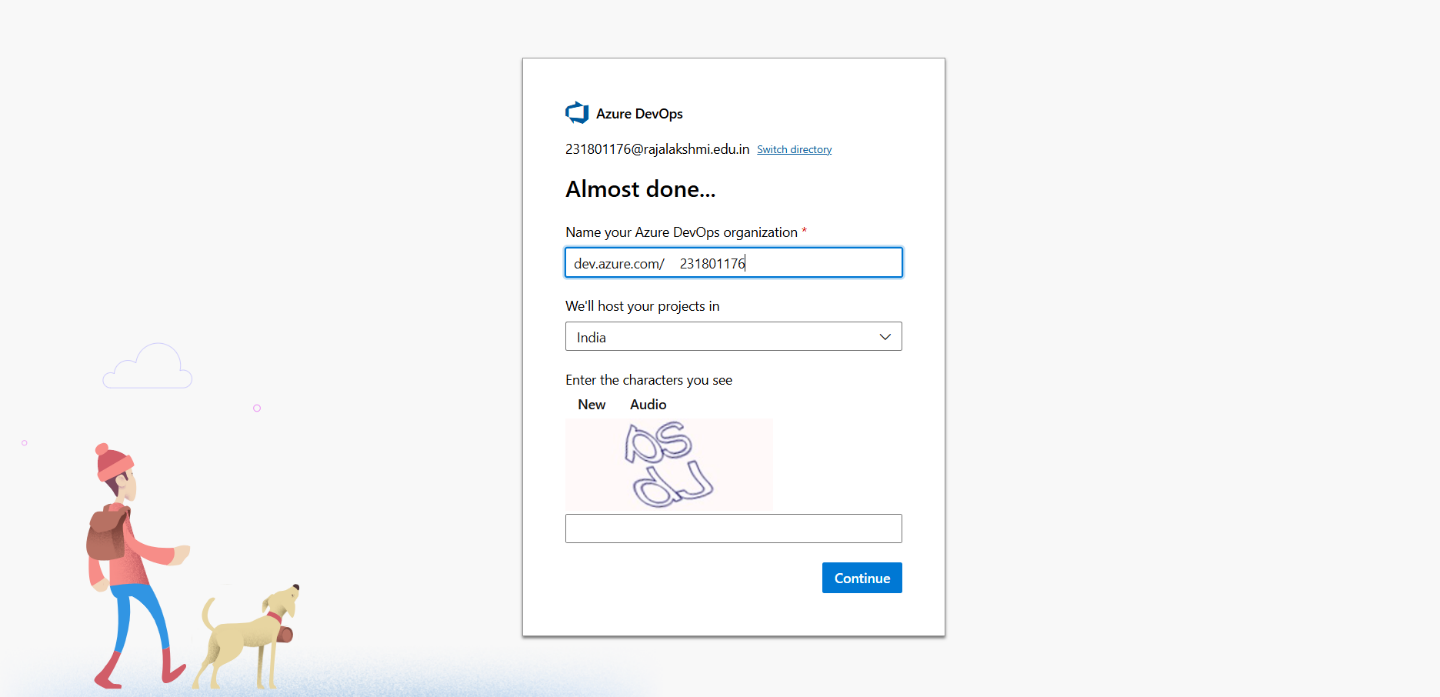
Successfully accessed the Azure DevOps environment and created a new organization through the Azure portal.

|  |  |
| --- | --- |
| **EXP NO:** **2** | **AZURE DEVOPS PROJECT SETUP AND USER STORY**  F  **MANAGEMENT** |

**Aim:**

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

1.Create An Azure Account



2.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

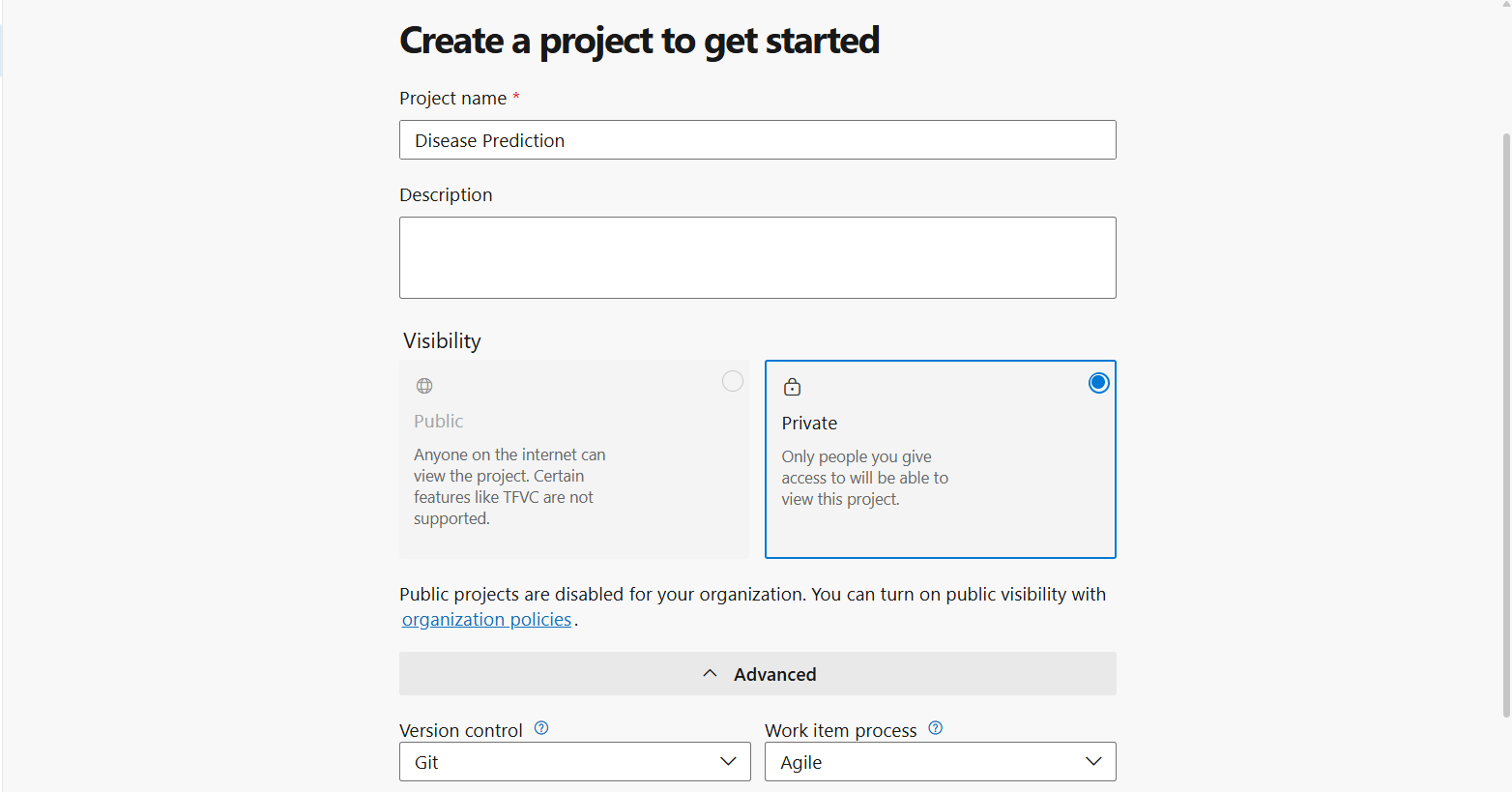
begin to manage code, pipelines, work items, and more.

1. On the organization’s **Home page**, click on the **New Project** button.
2. Enter the project name, description, and visibility options:

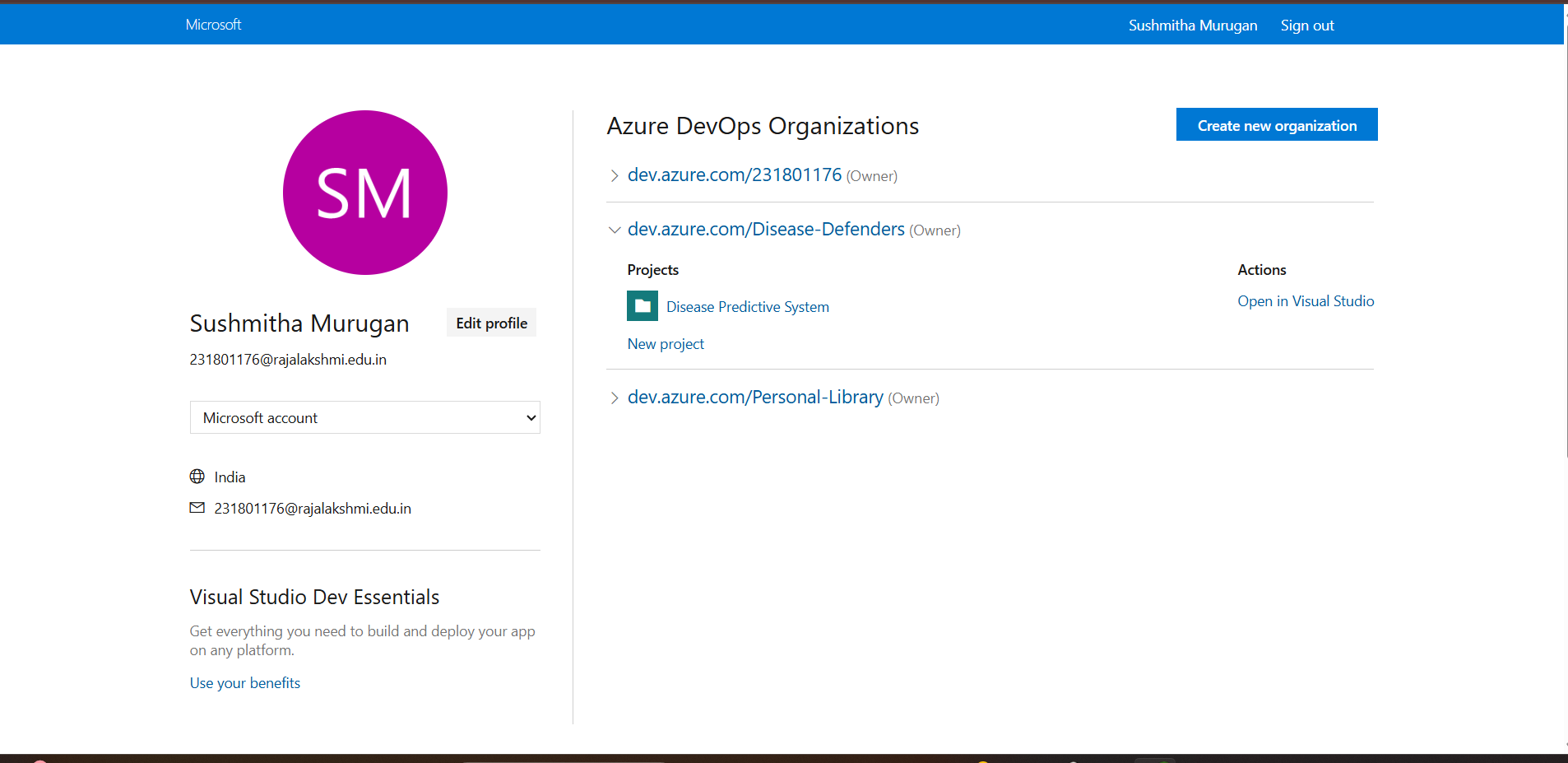
***Name****:* Choose a name for the project (e.g., LMS).

***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).

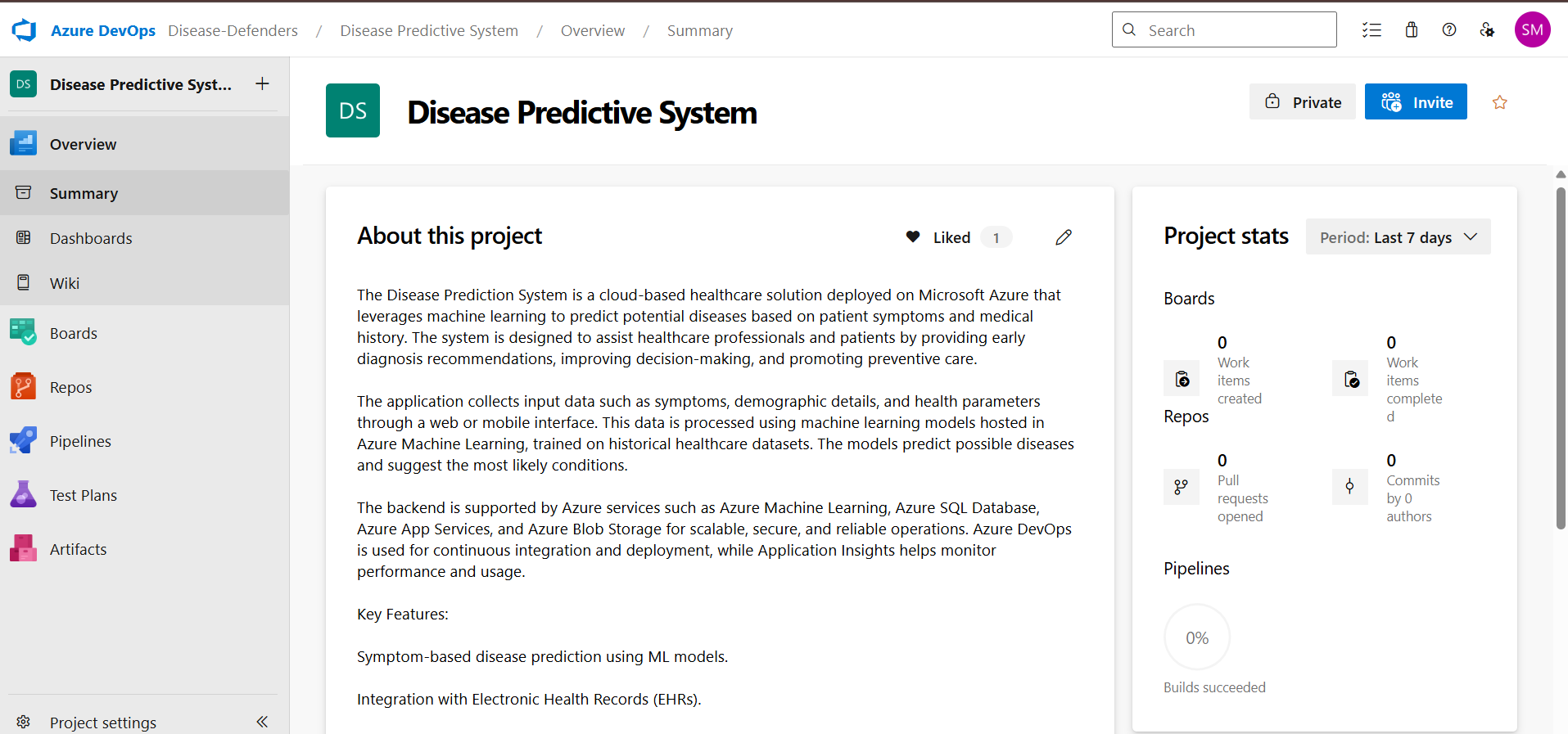
1. Once you’ve filled out the details, click **Create** to set up your first project.



3.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.



4.Project dashboard



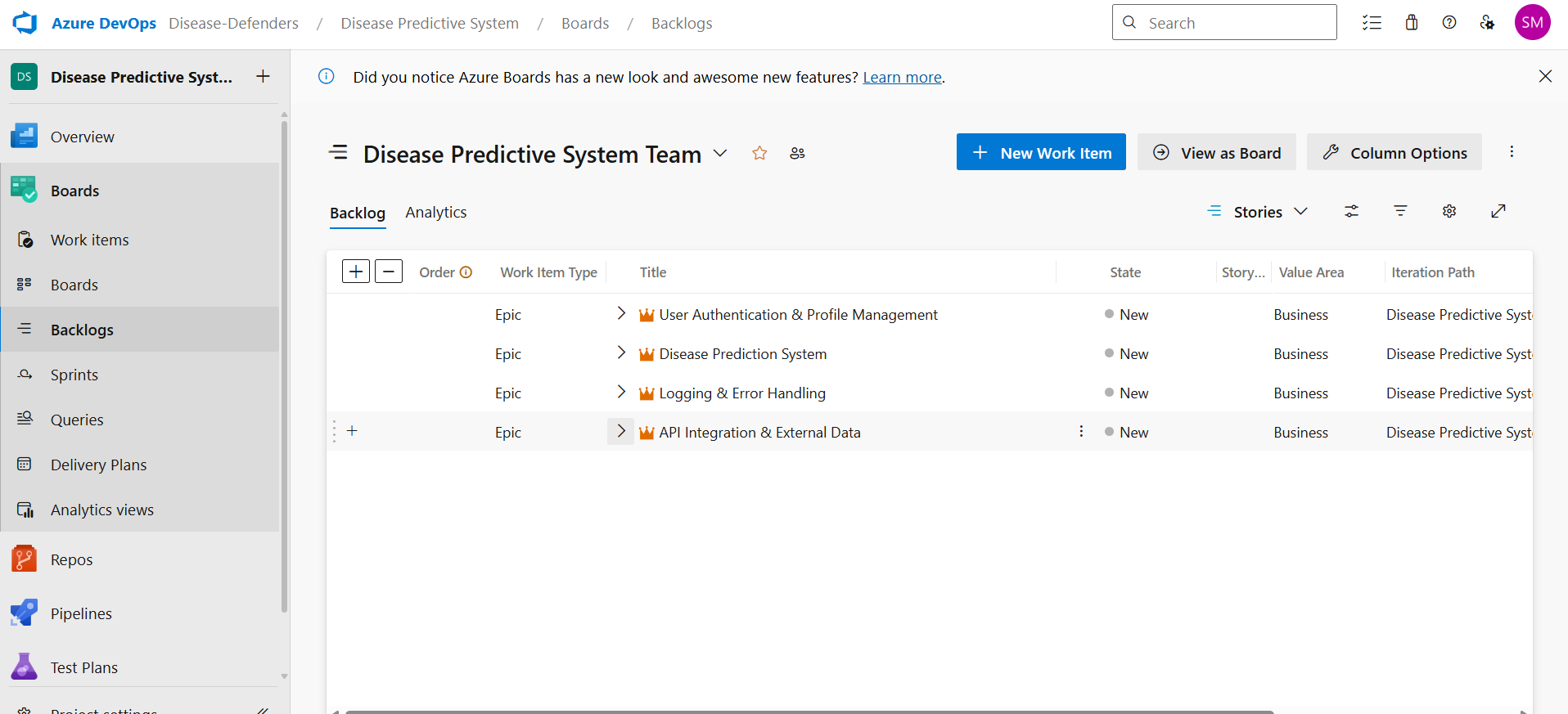
5.To manage user stories:

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:**

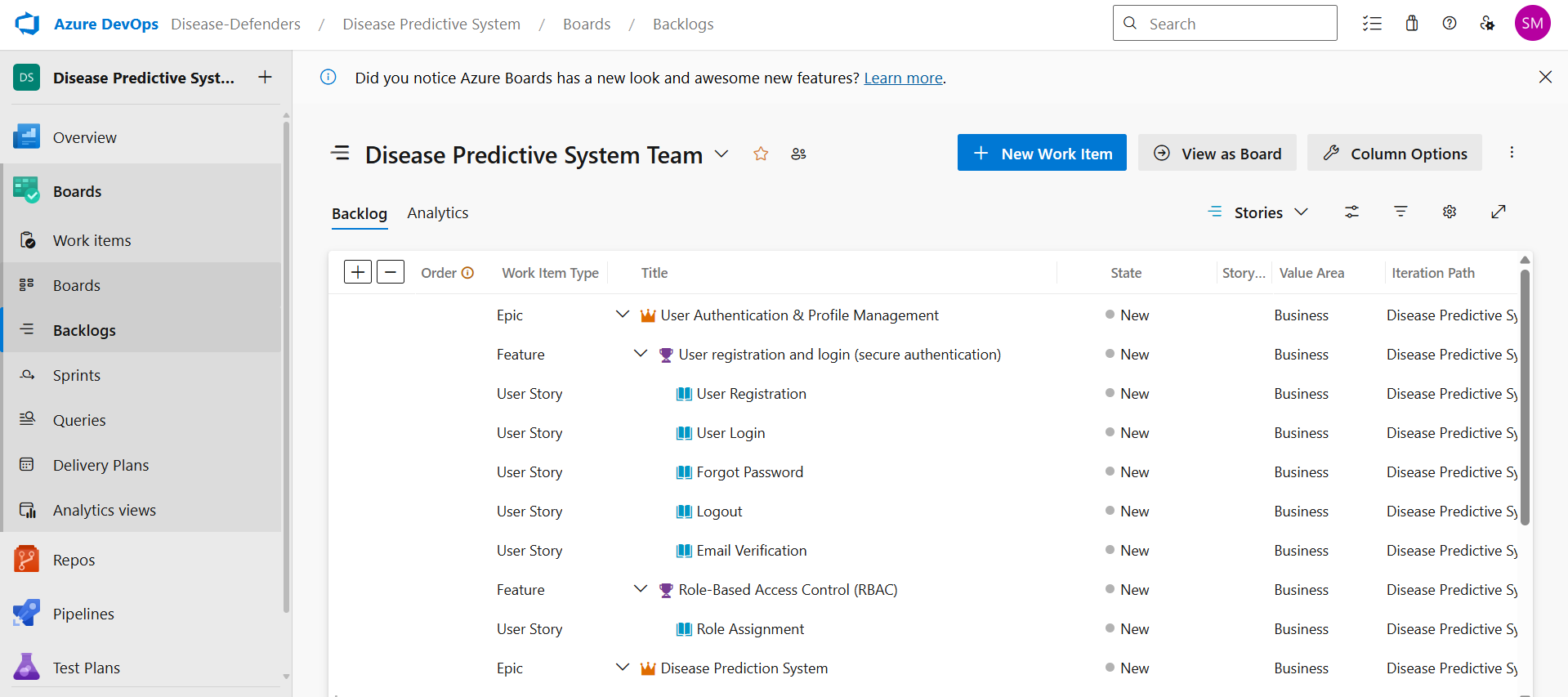
Successfully created an Azure DevOps project with user story management and agile workflow setup.

|  |  |
| --- | --- |
| **EXP NO:** **3** | **SETTING UP EPICS, FEATURES, AND USER STORIES**  **FOR PROJECT PLANNING** |

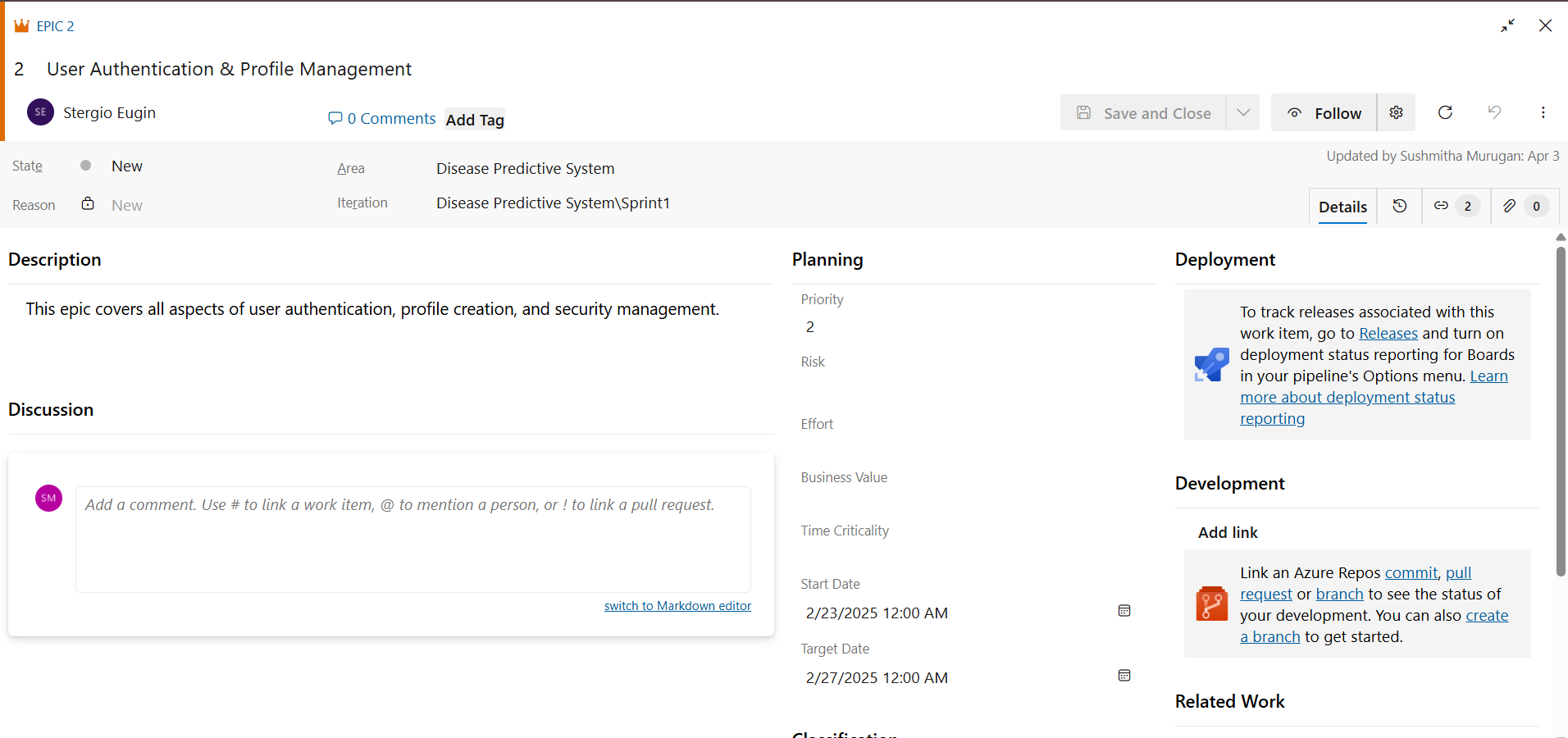
**Aim:**

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

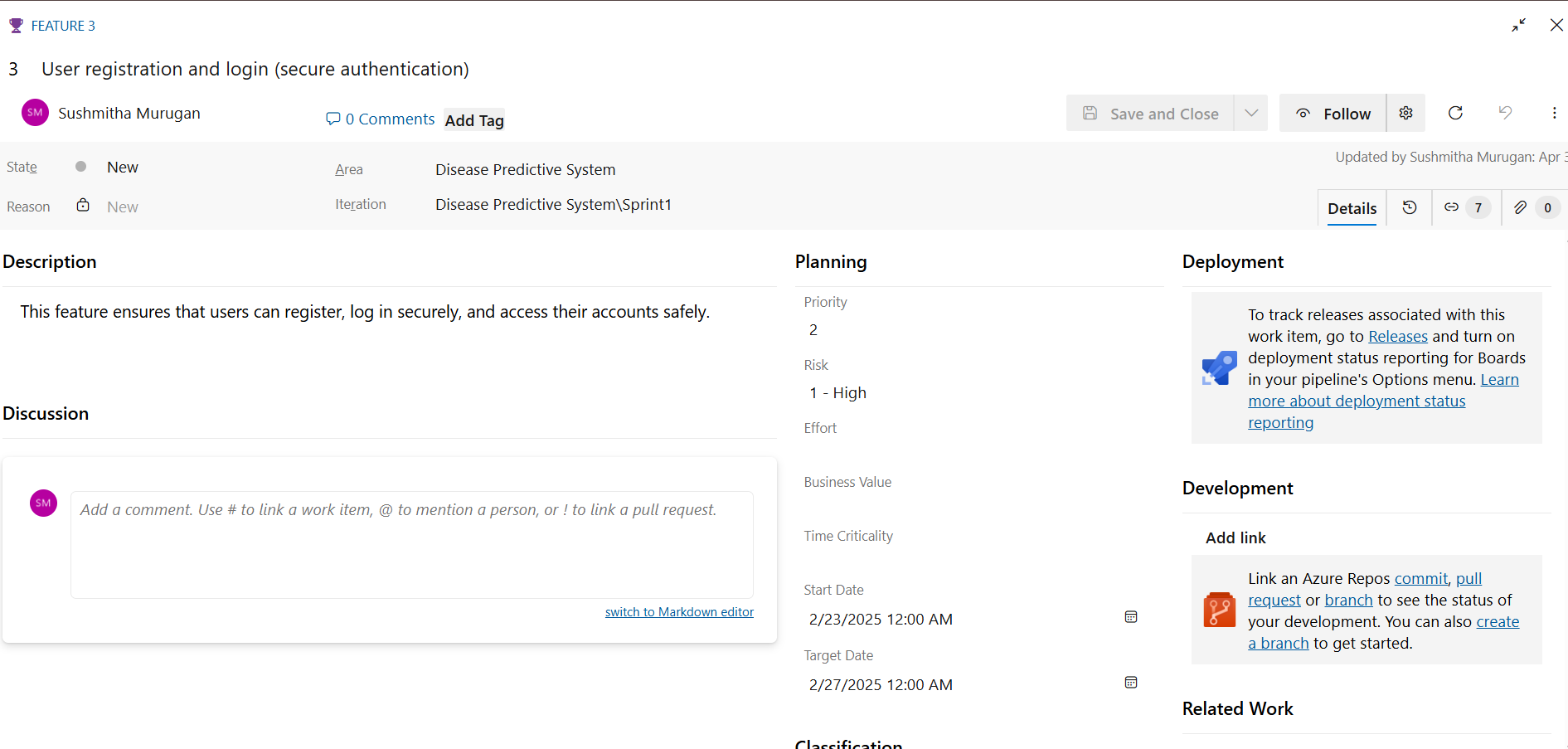
**Create Epic, Features, User Stories, Task**



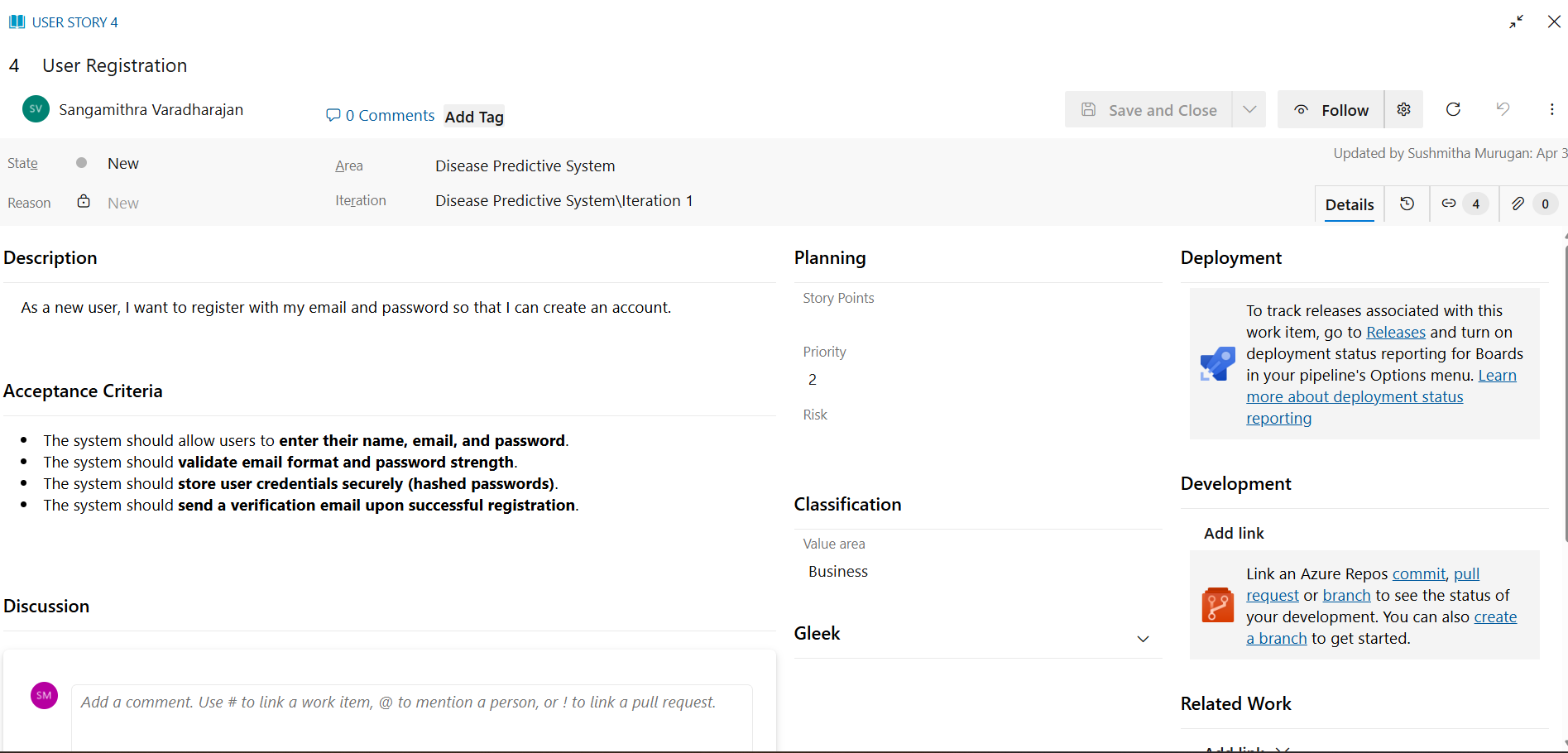
# 1.Fill in Epics



# 2.Fill in Features



# 3.Fill in User Story Details



**Result:**

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

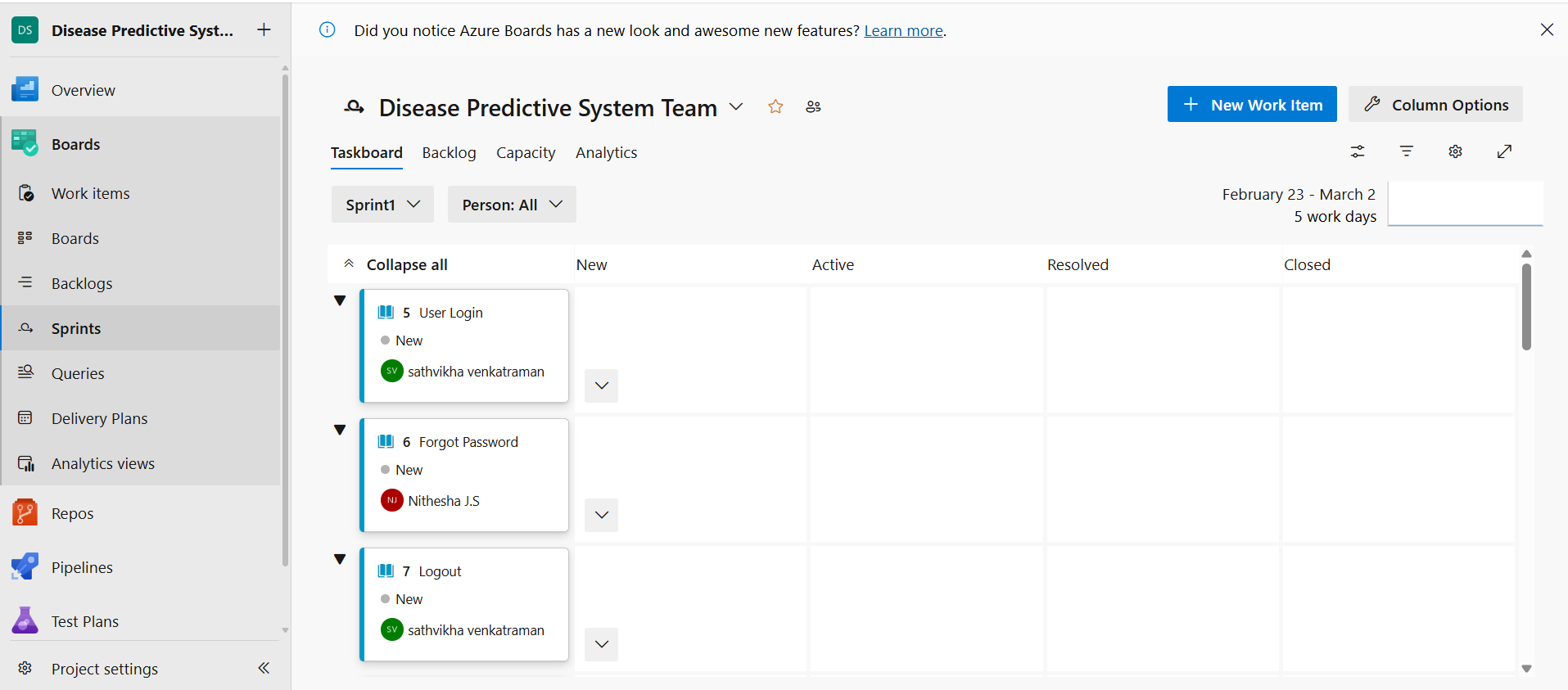
|  |  |
| --- | --- |
| **EXP NO:****4** | **SPRINT PLANNING** |

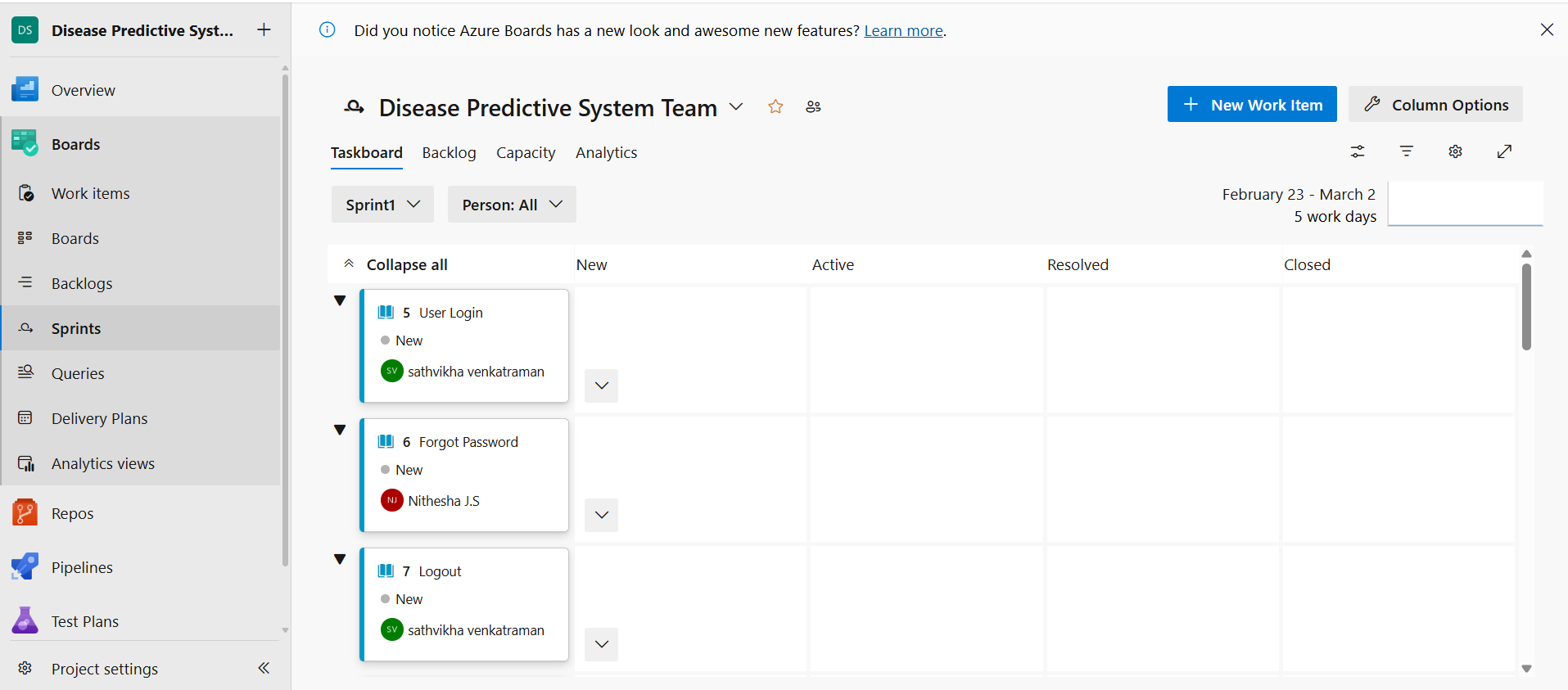
**Aim:**

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

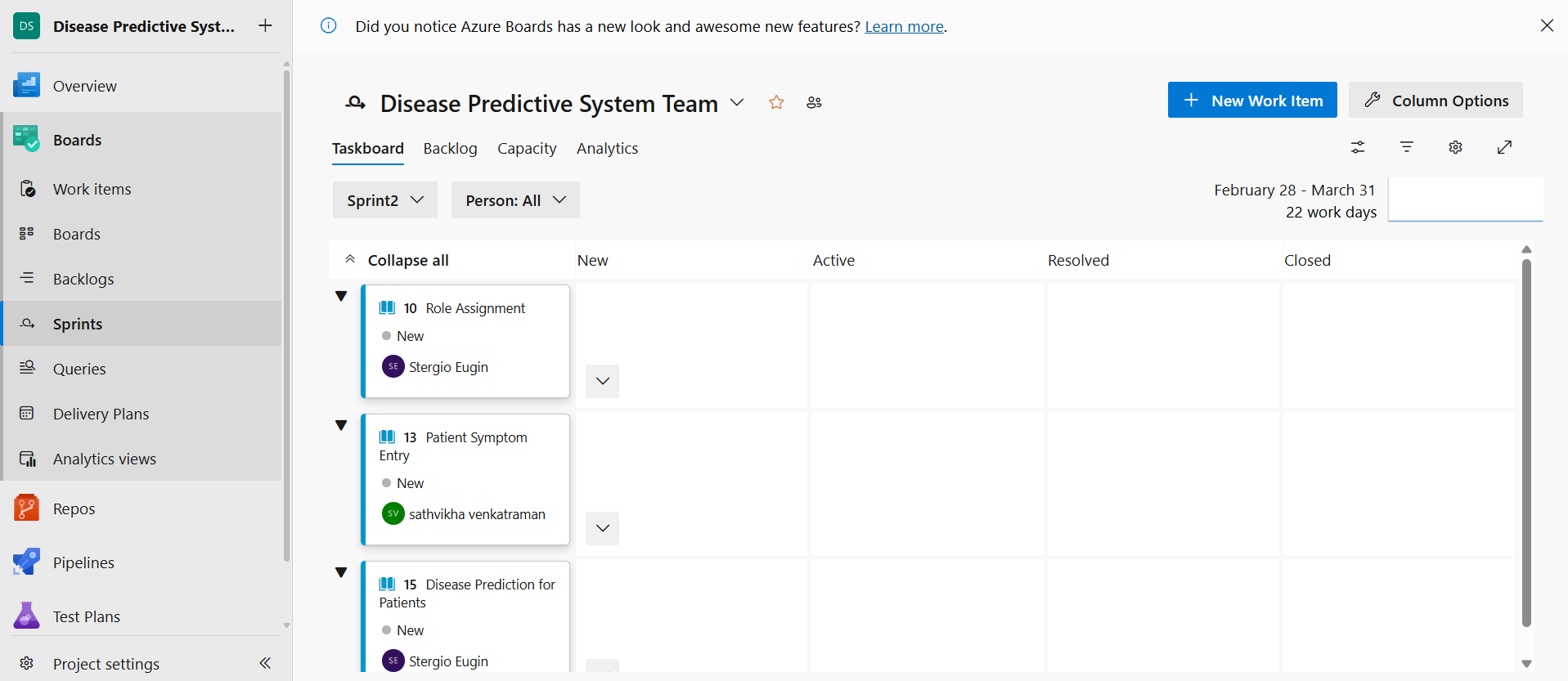
**Sprint Planning**

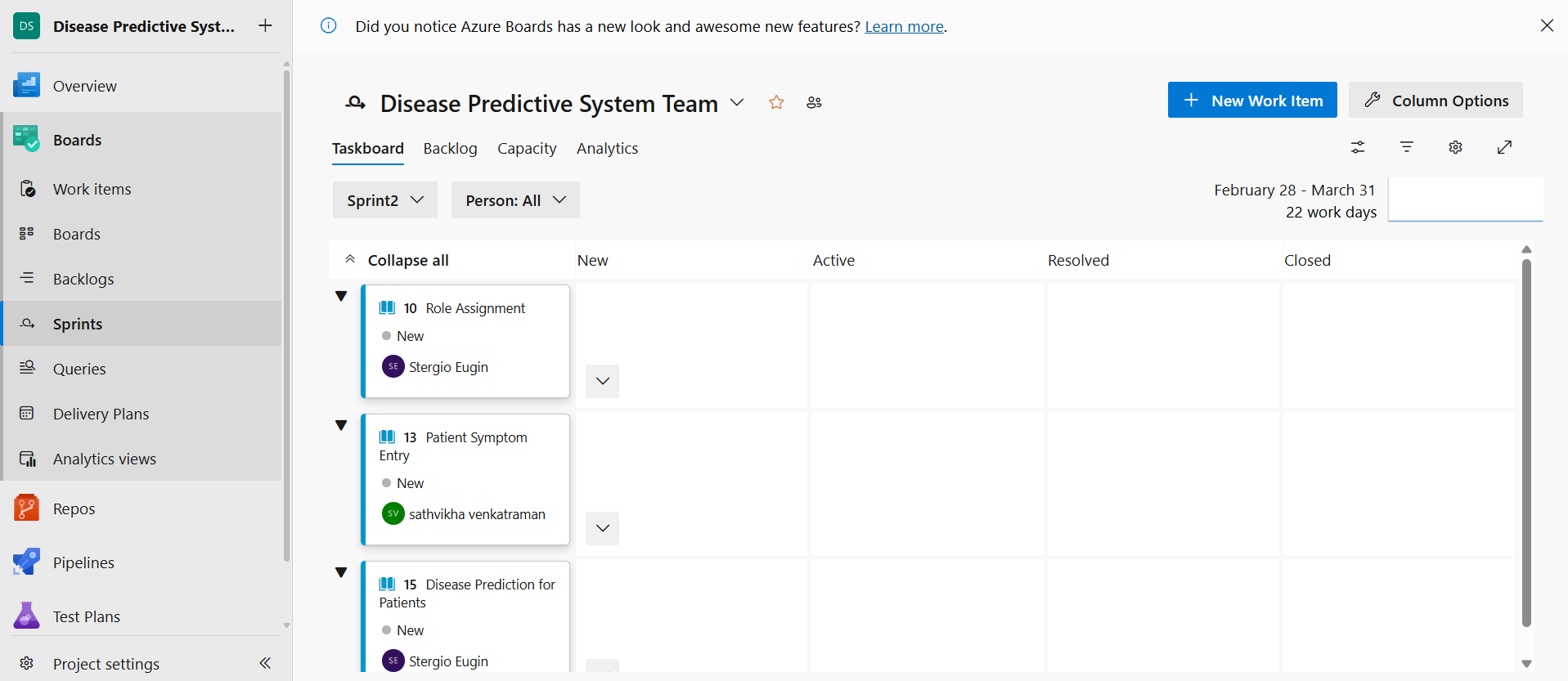
**Sprint 1 (iteration 1)**



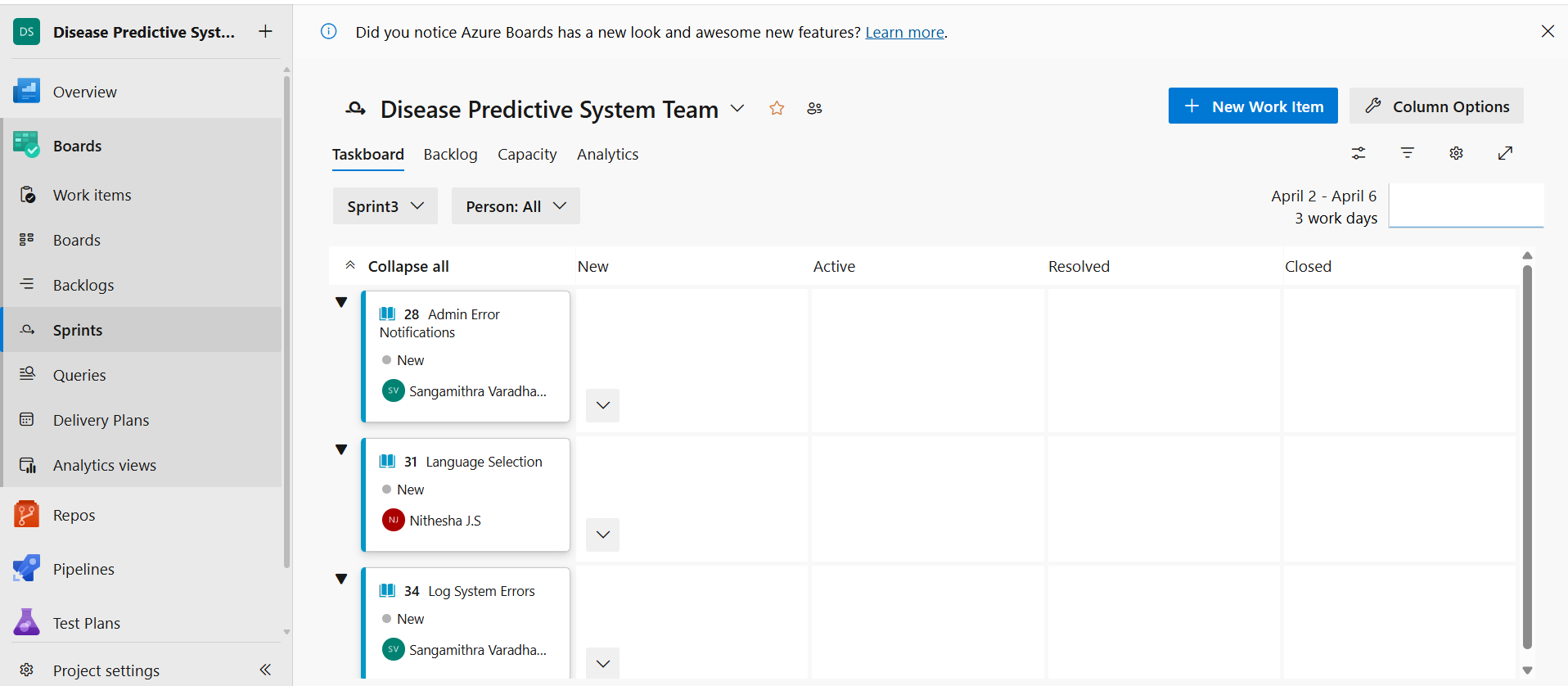


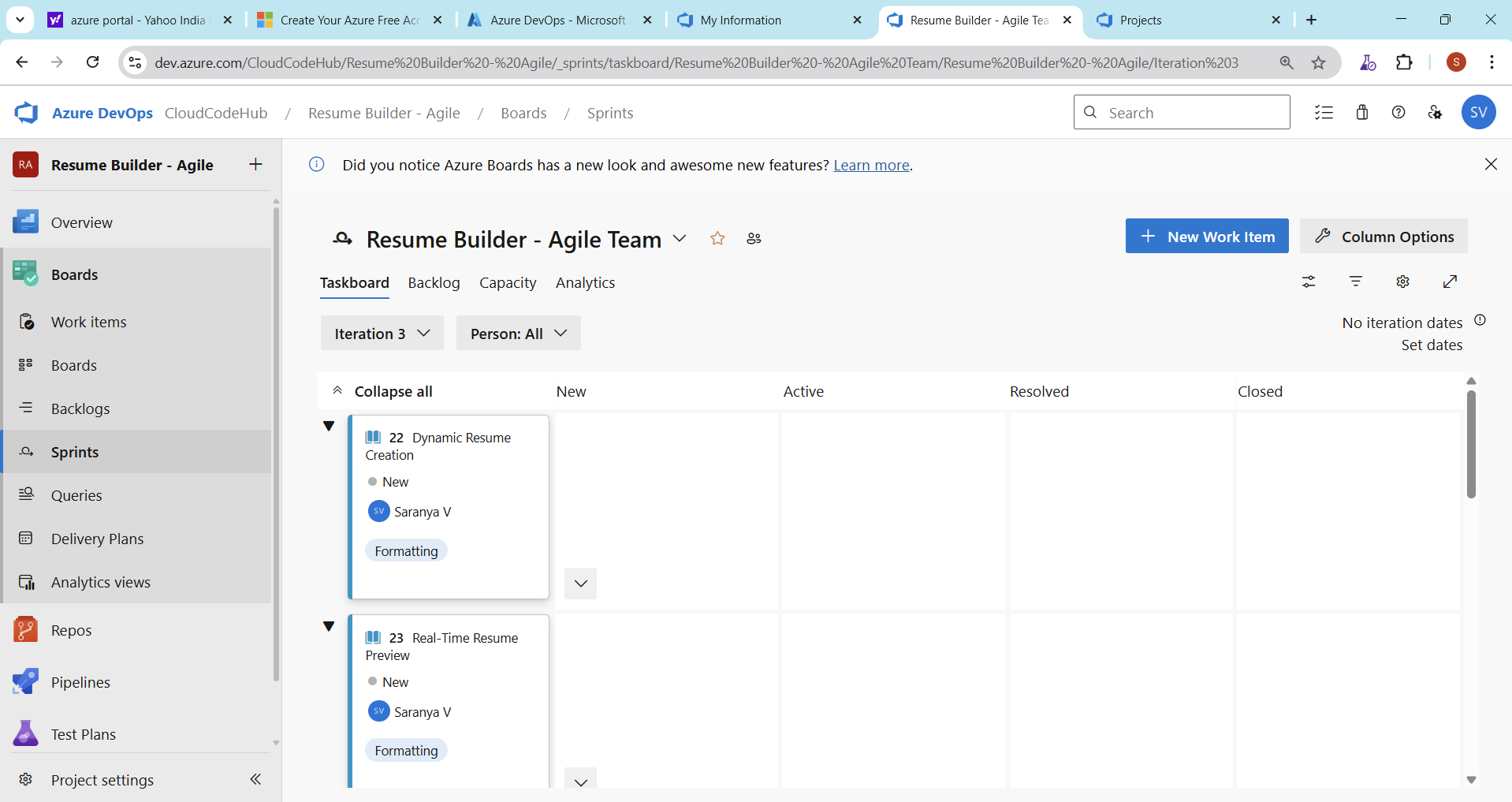
**Sprint 2 (iteration 2)**





**Sprint 3 (iteration 3)**





**Result:**

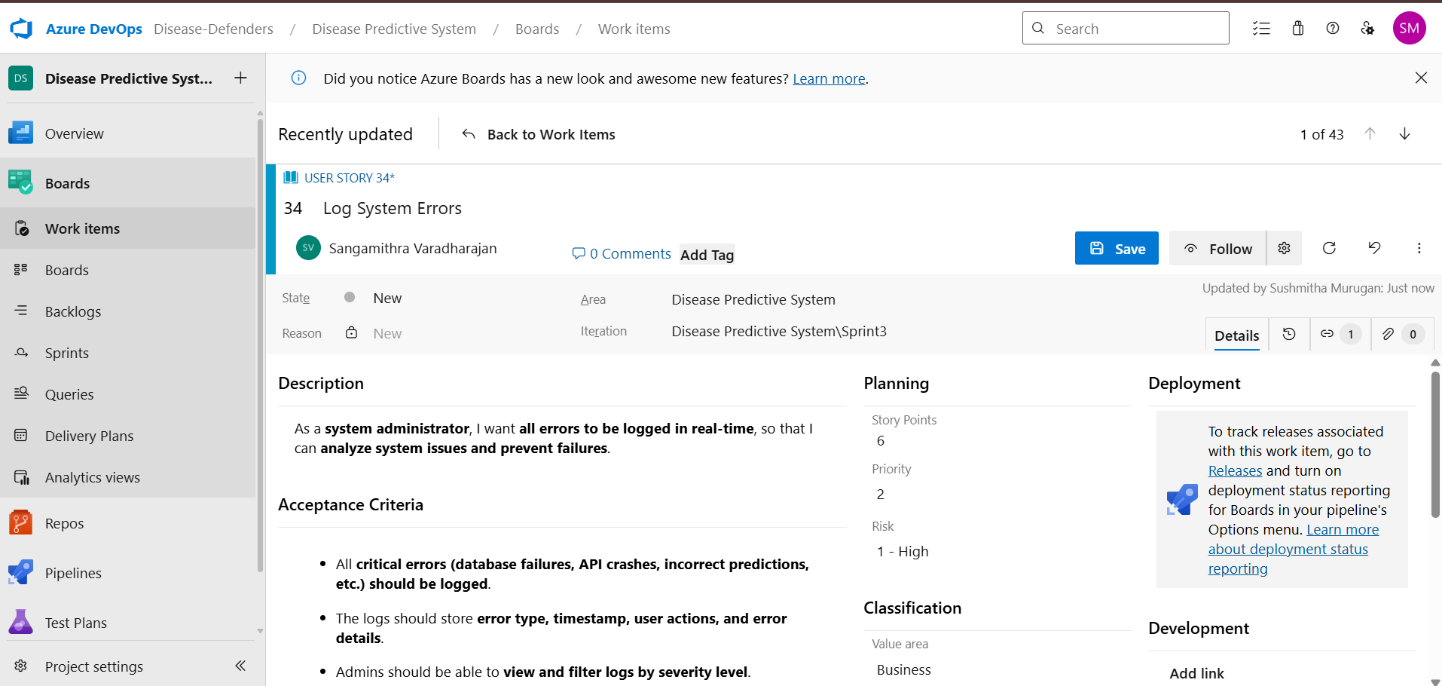
The Sprints are created for Resume Builder the Creator Project.

|  |  |
| --- | --- |
| **EXP NO:** **5** | **POKER ESTIMATION** |

**Aim:**

Create Poker Estimation for the user stories - Resume Builder Creator Project.

**Poker Estimation**



**Result:**

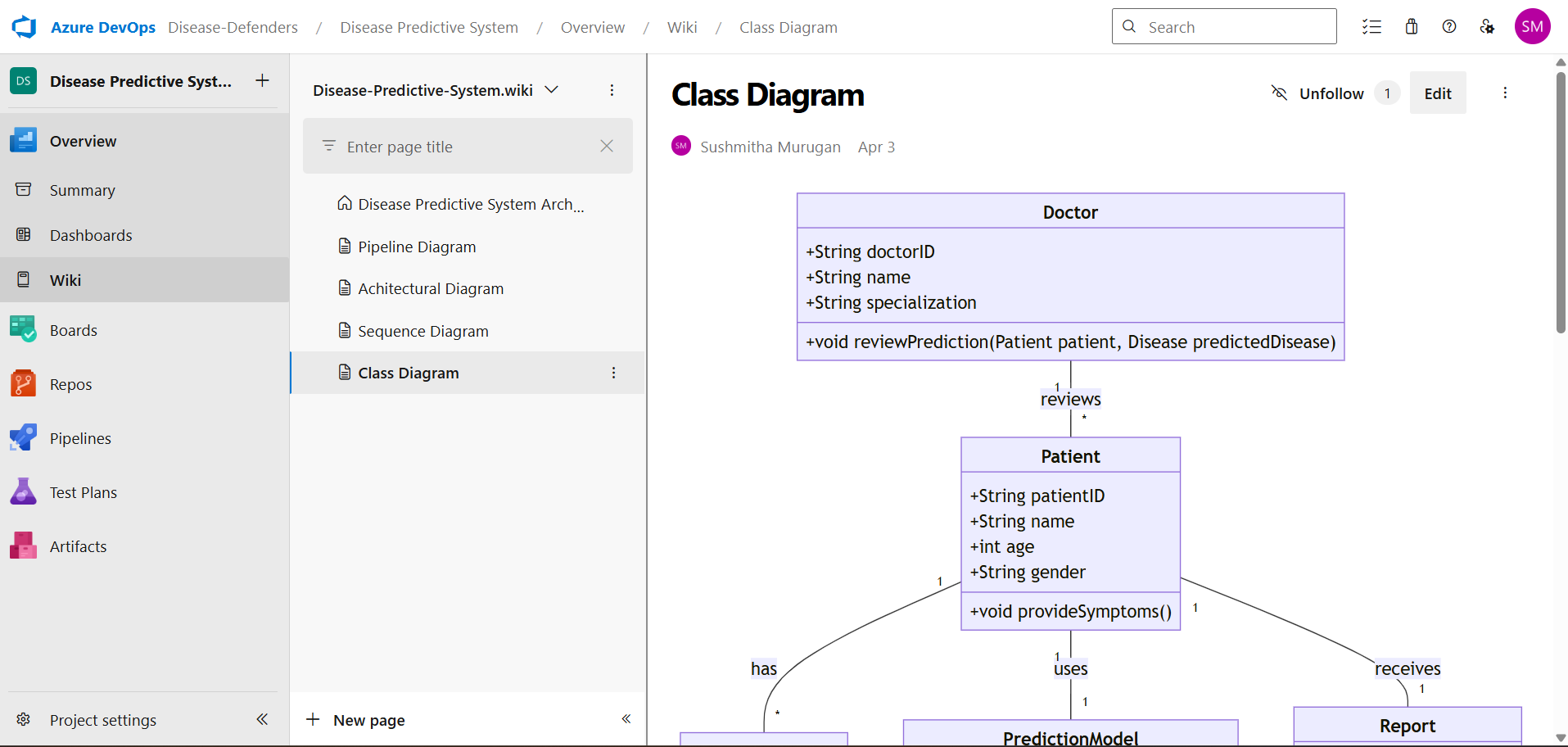
The Estimation/Story Points is created for the project using Poker Estimation.

|  |  |
| --- | --- |
| **EXP NO:** **6** | **DESIGNING CLASS AND SEQUENCE DIAGRAMS FOR**  F  **PROJECT ARCHITECTURE** |

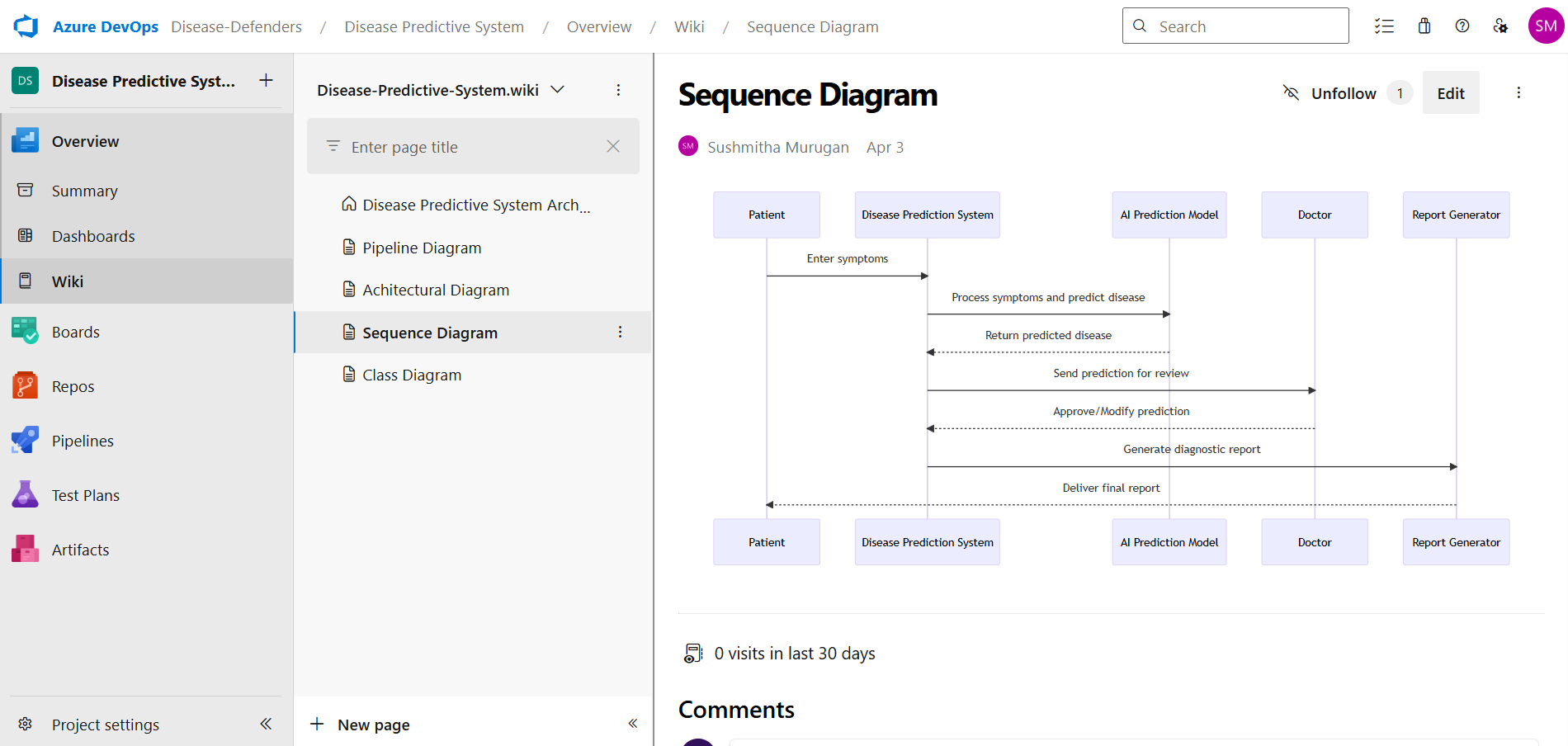
**Aim:**

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

**1.Class Diagram**

****

**2.Sequence Diagram**



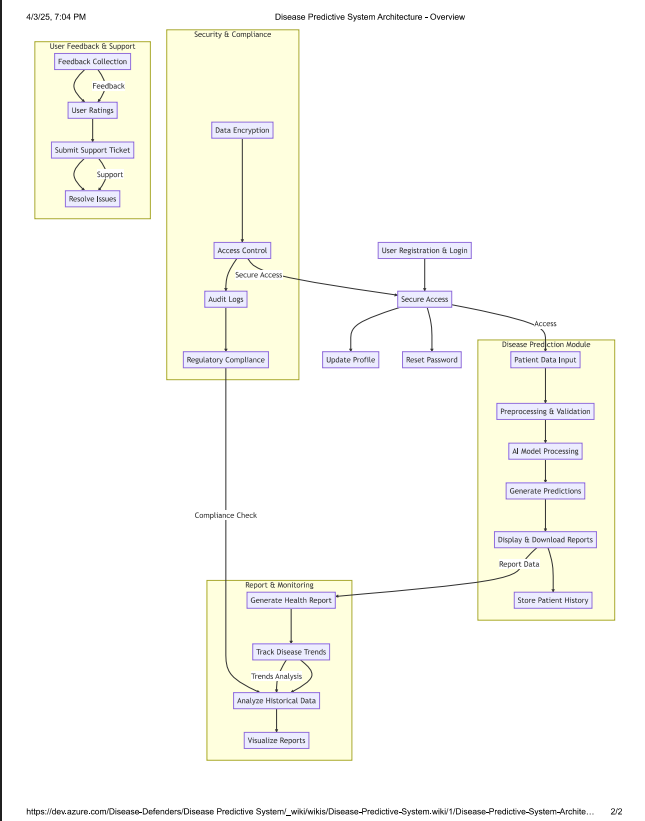
**Result:**

The Class Diagram and Sequence Diagram is designed Successfully for the Resume Builder Creator.

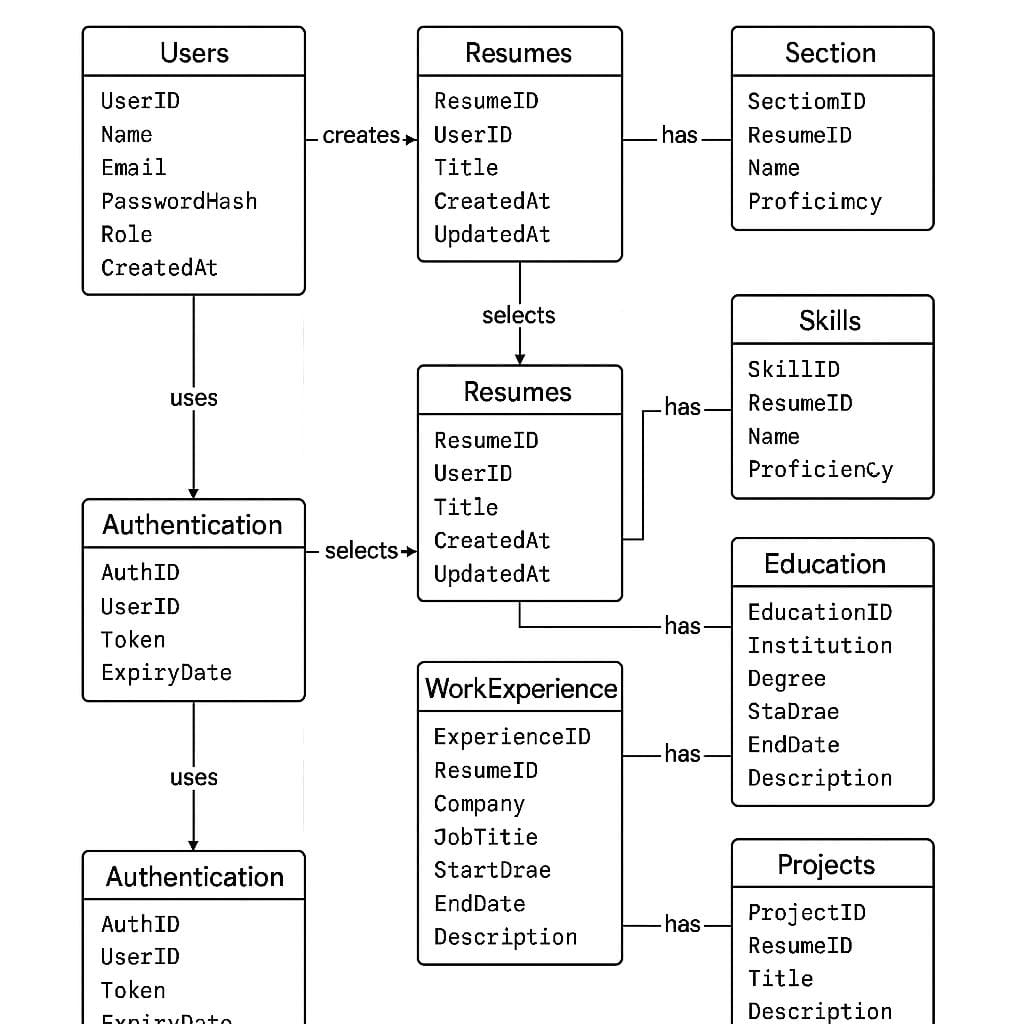
|  |  |
| --- | --- |
| **EXP NO:** **7** | **DESIGNING ARCHITECTURAL AND ER DIAGRAMS**  F  **FOR PROJECT STRUCTURE** |

**Aim:**

To Design an Architectural Diagram and ER Diagram for the given Project. **7A. 1.Architectural Diagram**



**2.ER Diagram**



**Result:**

The Architecture Diagram and ER Diagram is designed Successfully for the Resume Builder

Creator.

|  |  |
| --- | --- |
| **EXP NO:** **8** | **TESTING – TEST PLANS AND TEST CASES** |

**Aim:**

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:**

o User Signup & Login o Viewing and Managing Resume

o Fetching Real-time Metadata

o Editing Resume (rename, reorder, record)

o Creating Curriculum Vitae Resume based on categories (Dynamic Resume Personalization)

1. **Define User Interactions:**

o Each test case simulates a real user behaviour (e.g., logging in, renaming a resume, adding a resume).

**Design Happy Path Test Cases:**

o Focused on validating that all features function as expected under normal conditions.

o Example: User logs in successfully, adds item to resume, or creates a category-based resume.

1. **Design Happy PathTest Cases :**

o Simulate negative or unexpected scenarios to test robustness and error handling.

o Example: Login fails with invalid credentials, save fails when offline, no recommendations

found.

1. **Break Down Steps and Expected Results :**

o Each test case contains step-by-step actions and a corresponding expected outcome.

o Ensures clarity for both testers and automation scripts.

1. **Use Clear Naming and IDs:**

o Test cases are named clearly (e.g., TC01 – Successful Login, TC10 – Save Resume Fails).

o Helps in quick identification and linking to user stories or features.

1. **Separate Test Suites:**

o Grouped test cases based on functionality (e.g., Login, Resume Editing, Recommendation

System).

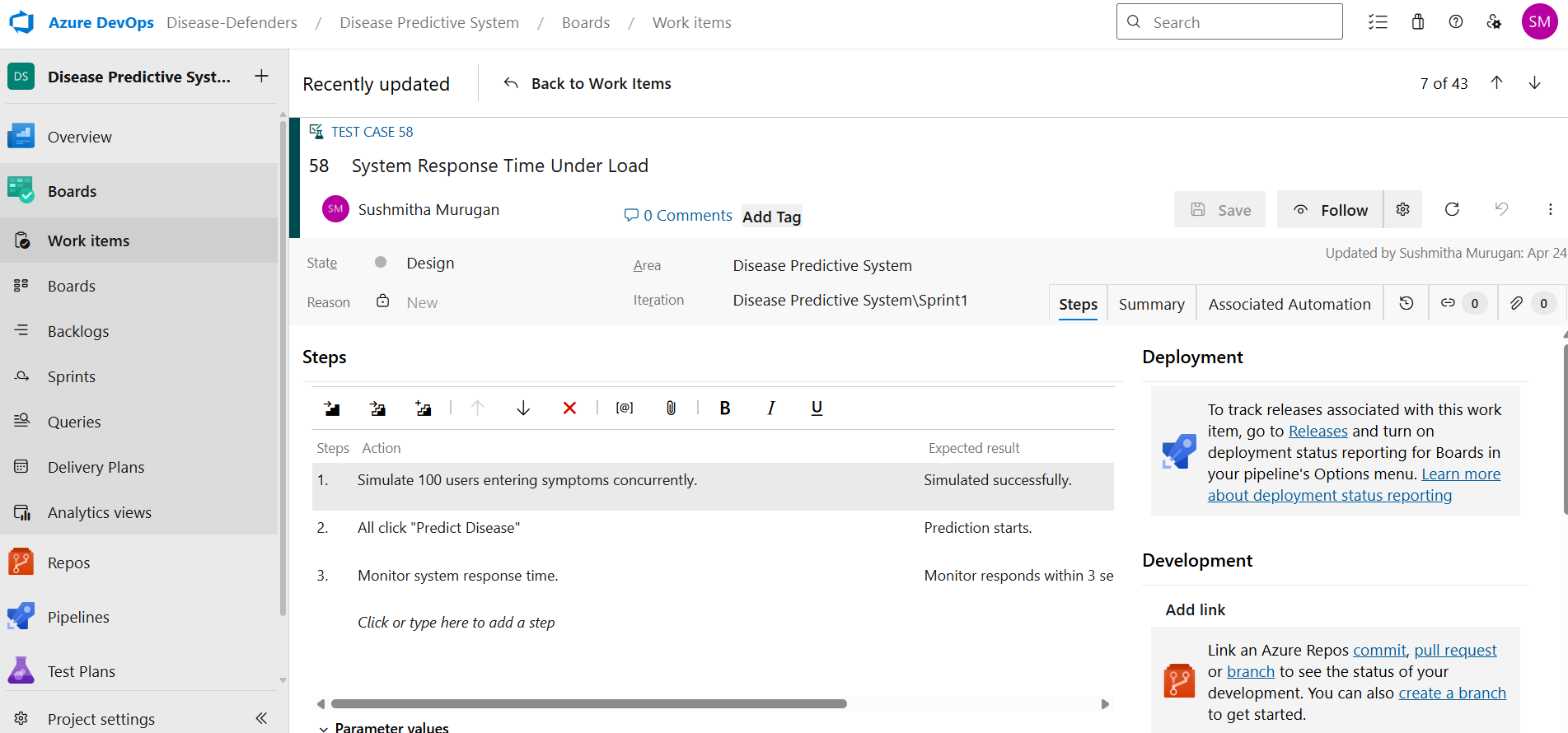
o Improves organization and test execution flow in Azure DevOps.

1. **Prioritize and Review:**

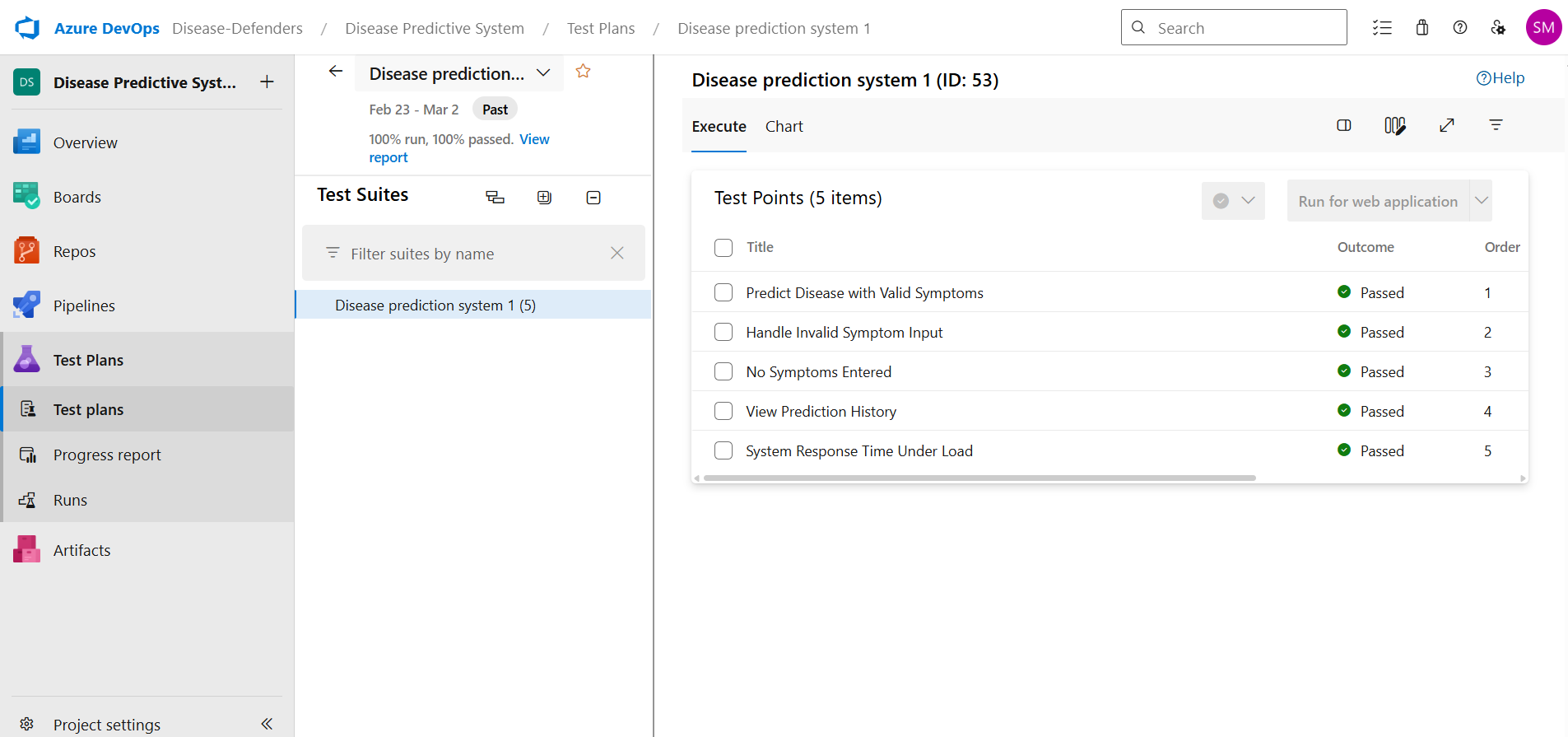
o Critical user actions are marked high-priority.

o Reviewed for completeness and traceability against feature requirements.

**1.New test plan**



**2.Test suite**

****

**3.Test case**

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

Resume Builder Creator – Test Plans

**USER STORIES**

* + As a user I want to register and log in securely so that I can access my account and manage my resumes.
  + As a user, I want to update my profile details so that my account information remains accurate.
  + As an admin, I want to manage user roles so that only authorized users can access admin functionalities.
  + As a user, I want to reset my password if I forget it so that I can regain access to my account.
  + Enter Resume Details via a User-Friendly Form

**Test Suites**

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

* 1. **TC01 – Successful Sign Up** o **Action:** 
     + - * Go to the Sign-Up page.
         * Enter valid name, email, and password.
         * Click "Sign Up".

o **Expected Results:**

* + - * + Sign-Up form is displayed.
        + Fields accept values without error.
        + Account is created, and the user is redirected to the dashboard.

o **Type**: Happy Path

* 1. **TC02 – Secure Login**

o **Action:**

* + - * + Go to the Login page.
        + Enter valid email and password.
        + Click on "Login".

o **Expected Results:**

* + - * + Login form is displayed.
        + Fields accept data without error.
        + User is logged in and redirected to the dashboard.

o **Type:** Happy Path

* 1. **TC03 – Sign Up with Existing Email**

o **Action:**

* + - * + Go to the Sign-Up page.
        + Enter a name and an already registered email.
        + Click on "Sign Up".

o **Expected Results:**

* + - * + Fields accept data.
        + Error message "Email already registered" is displayed.

o **Type:** Happy Path

* 1. **TC04 – Login with Wrong Password**

o **Action:**

* + - * + Go to the Login page.
        + Enter valid email and incorrect password.
        + Click on "Login".

o **Expected Results:**

* + - * + Input is accepted.
        + Error message "Invalid username or password" is shown.

o **Type:** Happy Path

**Test Suit: TS02 - View Resume (ID: 87)**

**1. TC05 – View Resume Page**

o **Action:**

* + - * + Log in successfully.
        + Navigate to "My Resume " section.

o **Expected Results:**

▪ All created Resume are displayed clearly.

o **Type:** Happy Path

2. **TC06 – Resume Loading Failure**

o **Action:**

* + - * + Disconnect from the internet.
        + Navigate to "My Resume ".

o **Expected Results:**

* + - * + Network is offline.
        + Error message "Unable to load Resume " is shown.

O **Type:** Happy Path

**Test Suit: TS03 - Real-Time Metadata (ID: 88)**

1. **TC07 – Real-Time Metadata Display**

o **Action:**

* + - * + Play a resume.
        + Observe the metadata panel.

o **Expected Results:**

▪ Metadata (title, artist, album, duration) is displayed and updates in real time.

o **Type:** Happy Path

1. **TC08 – Metadata Not Updating**

o **Action:**

* + - * + Play a different resume.
        + Observe the metadata panel.

o **Expected Results:**

▪ Metadata remains static or shows default/fallback message.

o **Type:** Happy Path

**Test Suit: TS04 - Resume Editing (ID: 89)**

**1. TC09 – Rename Resume Successfully**

o **Action:**

* + - * + Navigate to "My Resume ".
        + Click "Rename" next to a resume.
        + Enter a new name and click "Save".

o **Expected Results:**

▪ Resume name updates successfully.

o **Type:** Happy Path

1. **TC10 – Rename with Blank Name**

o **Action:**

* + - * + Click "Rename" on a resume.
        + Leave the field blank.
        + Click "Save".

o **Expected Results:**

▪ Error message "Resume name cannot be empty" is shown.

o **Type:** Happy Path

**3. TC11 – Change Resume Order**

o **Action:**

* + - * + Open a resume.
        + Drag and drop resumes to reorder.
        + Click "Save".

o **Expected Results:**

▪ Resume order is updated and saved.

* **Type:** Happy Path

**4. TC12 – Change Resume Order Fails**  **Action:**

* + - * + Login and go to “My Resume ”.
        + Select a resume.
        + Go offline or simulate server error.
        + Reorder resumes and click “Save Order”.

o **Expected Results:**

▪ Error message: "Failed to update order. Please check your connection".

o **Type:** Happy Path

**Test Suit: TS05 - Smart Resume Creation (ID: 90)**

**1. TC13 – Generate Resume Based on Various Categories** o **Action:**

* + - * + Login with valid credentials.
        + Click on "Generate Resume".
        + Select categories.
        + Click "Generate Resume".

o **Expected Results:**

▪ Resume is generated based on selected profession and categories.

o **Type:** Happy Path

**2. TC14 – Fail to Generate Resume Due to Missing Category Selection or Invalid Input** o **Action:**

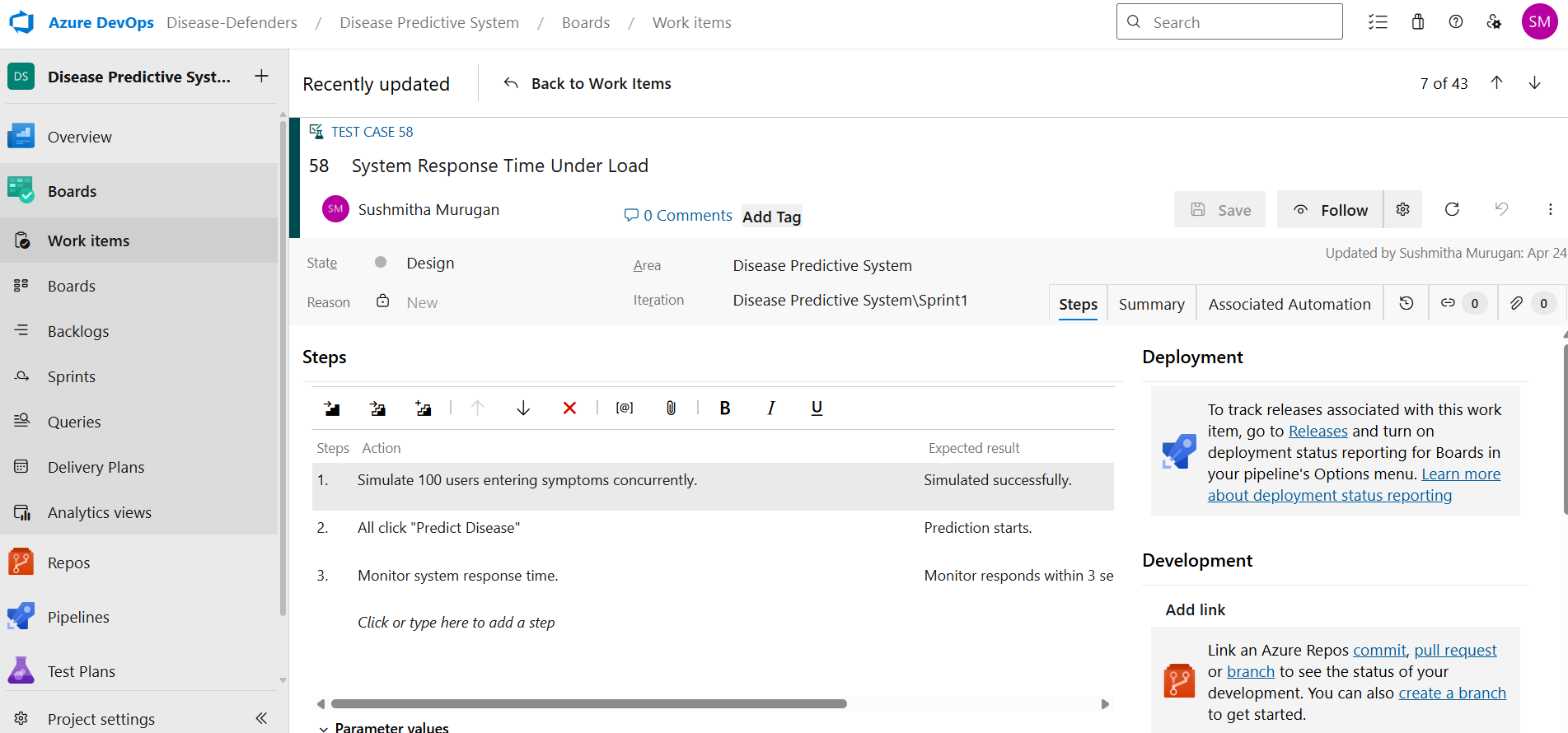
* + - * + Login with valid credentials.
        + Click on "Generate Resume".
        + Select categories.
        + Click "Generate Resume".

o **Expected Results:**

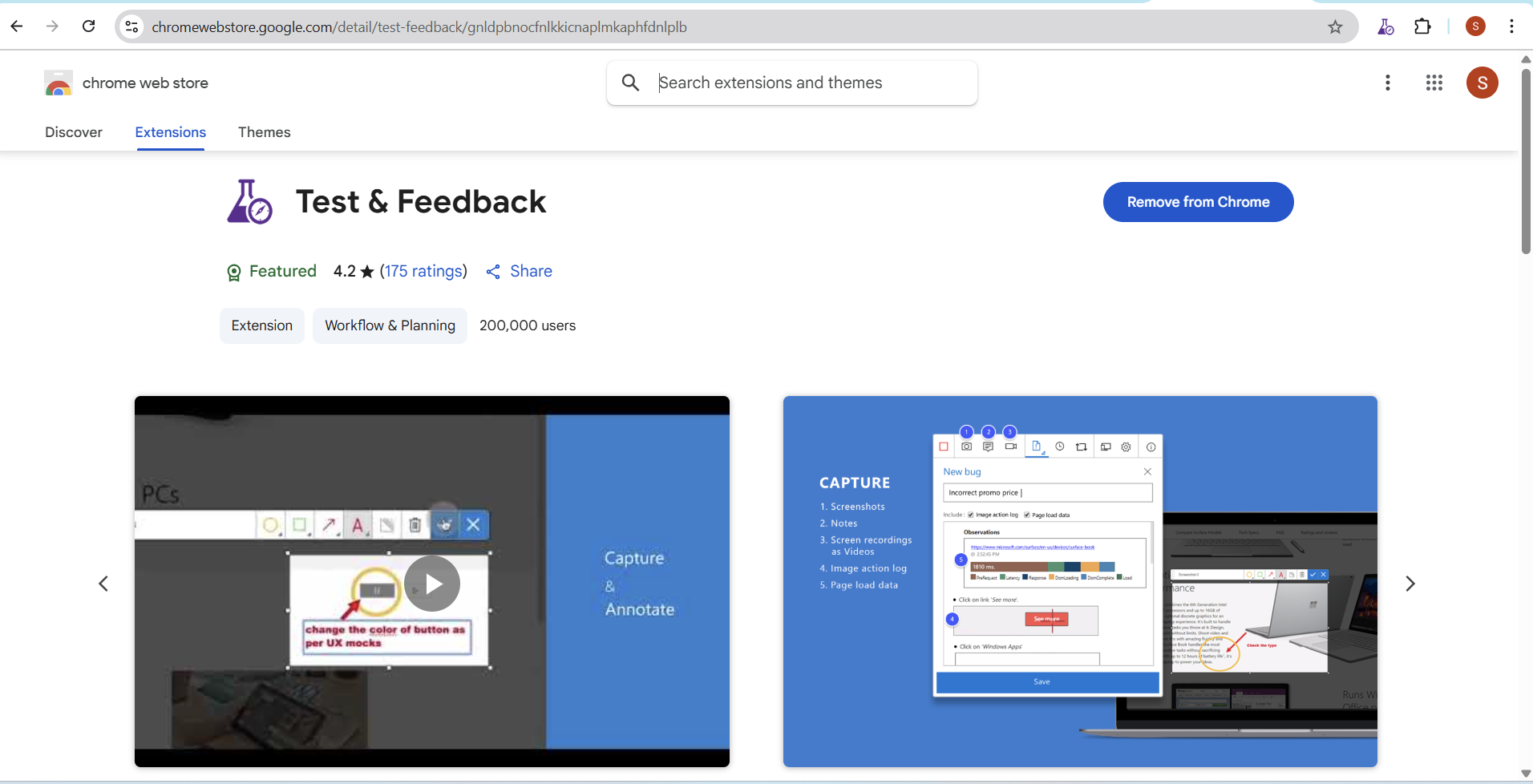
▪ Error message: "Please select at least one valid category" or "No recommendations found for the selected filters".

o **Type:** Happy Path

**Test Cases**

****

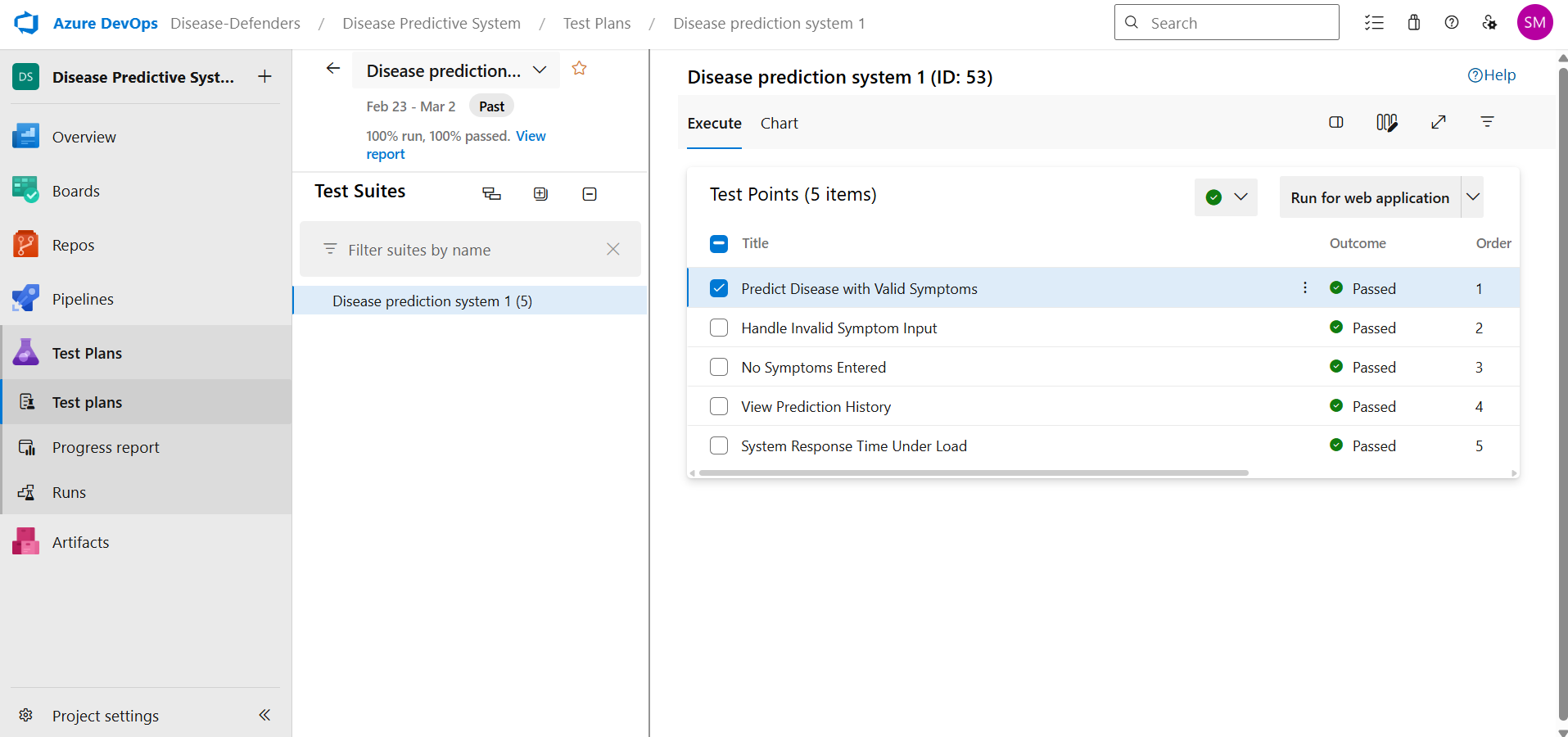
**4.Installation of test**



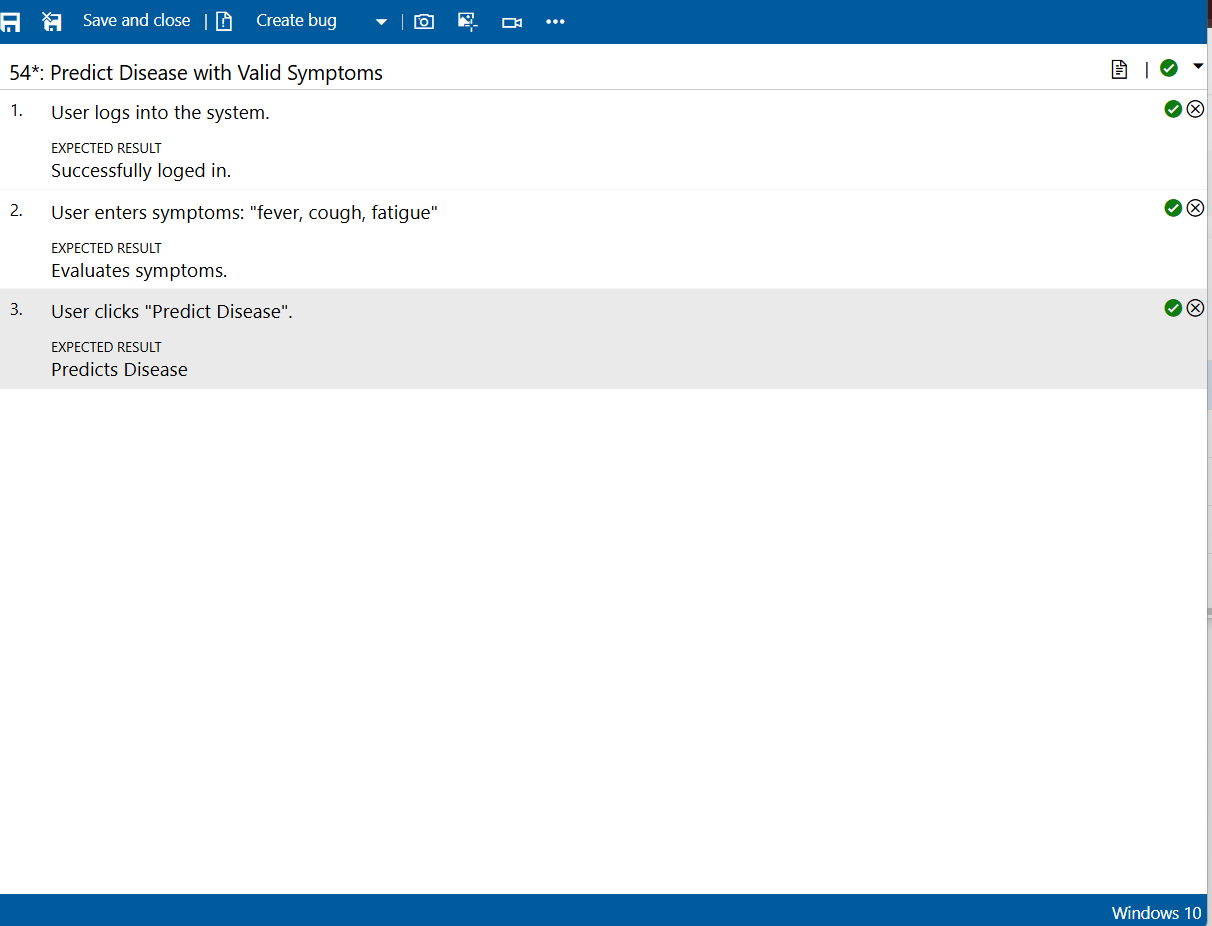
Test and feedback

Showing it as an extension

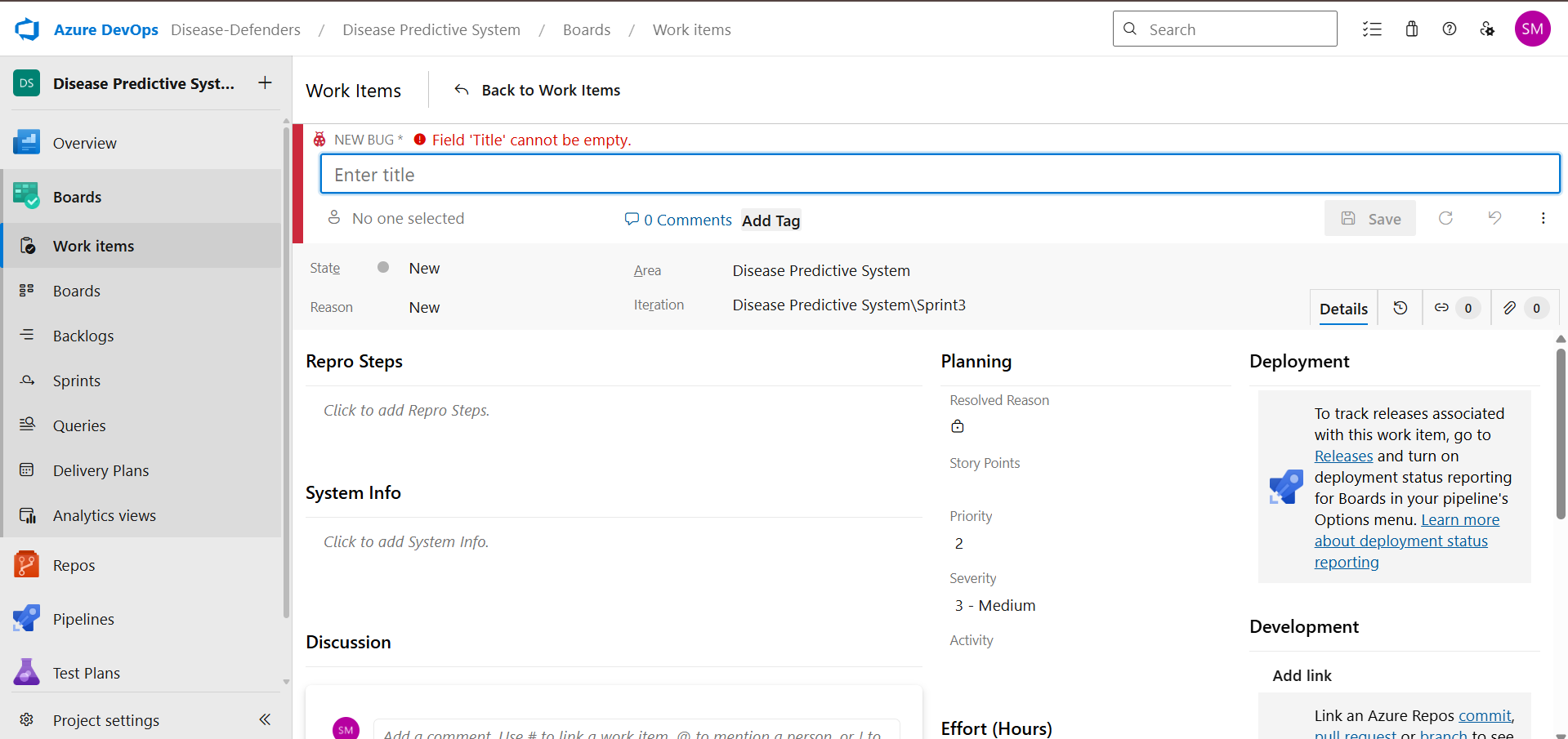
**5.Running the test cases**

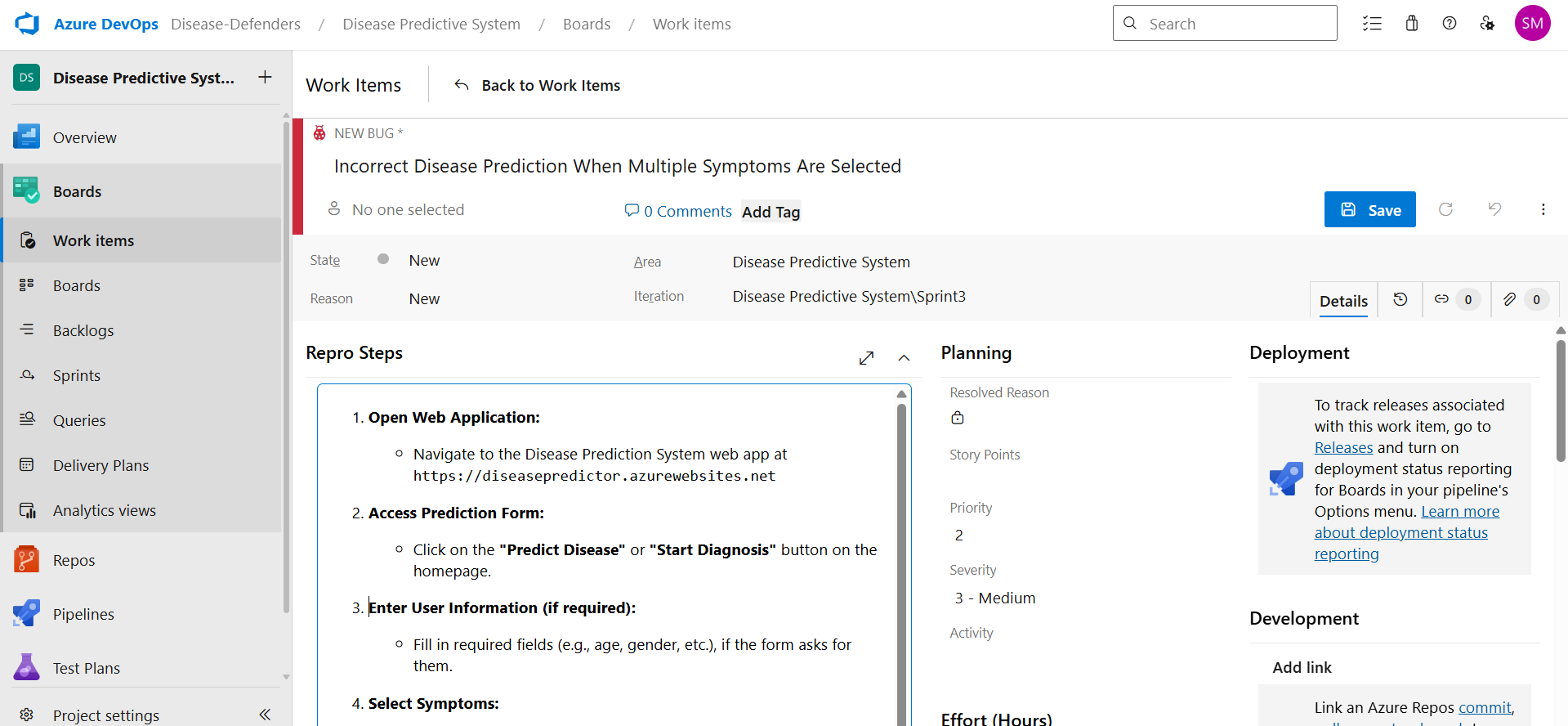
****

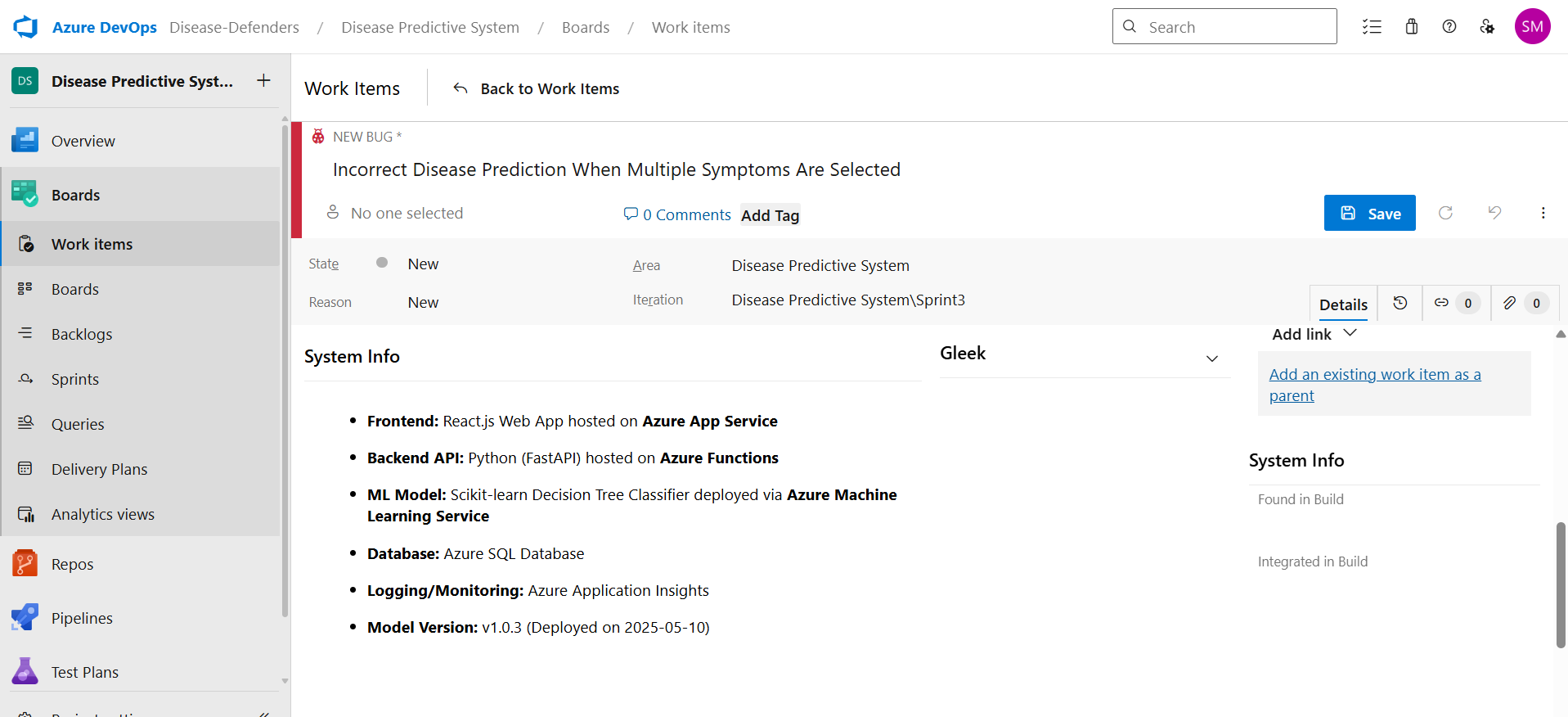
**6.Recording the test case**



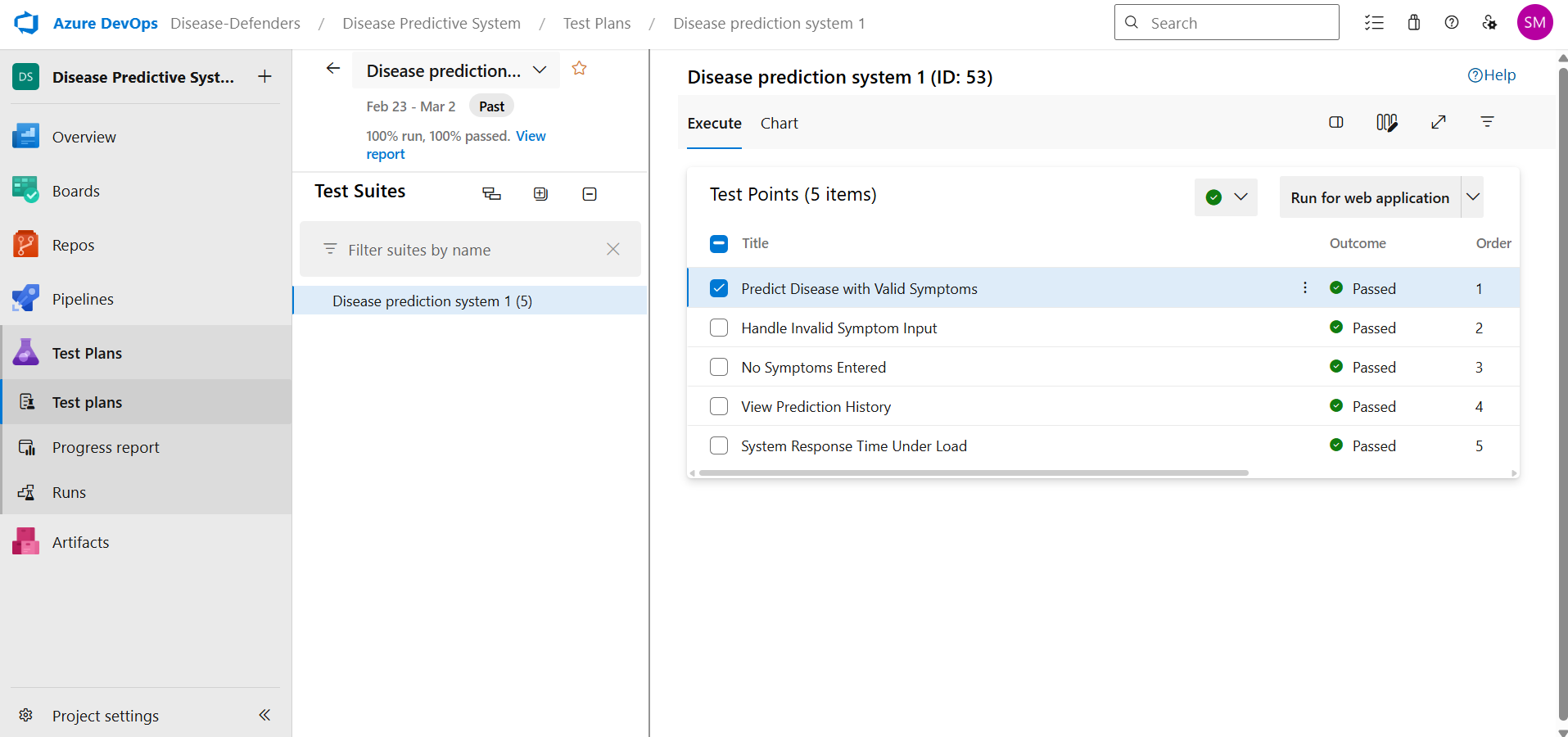
**7.Creating the bug**

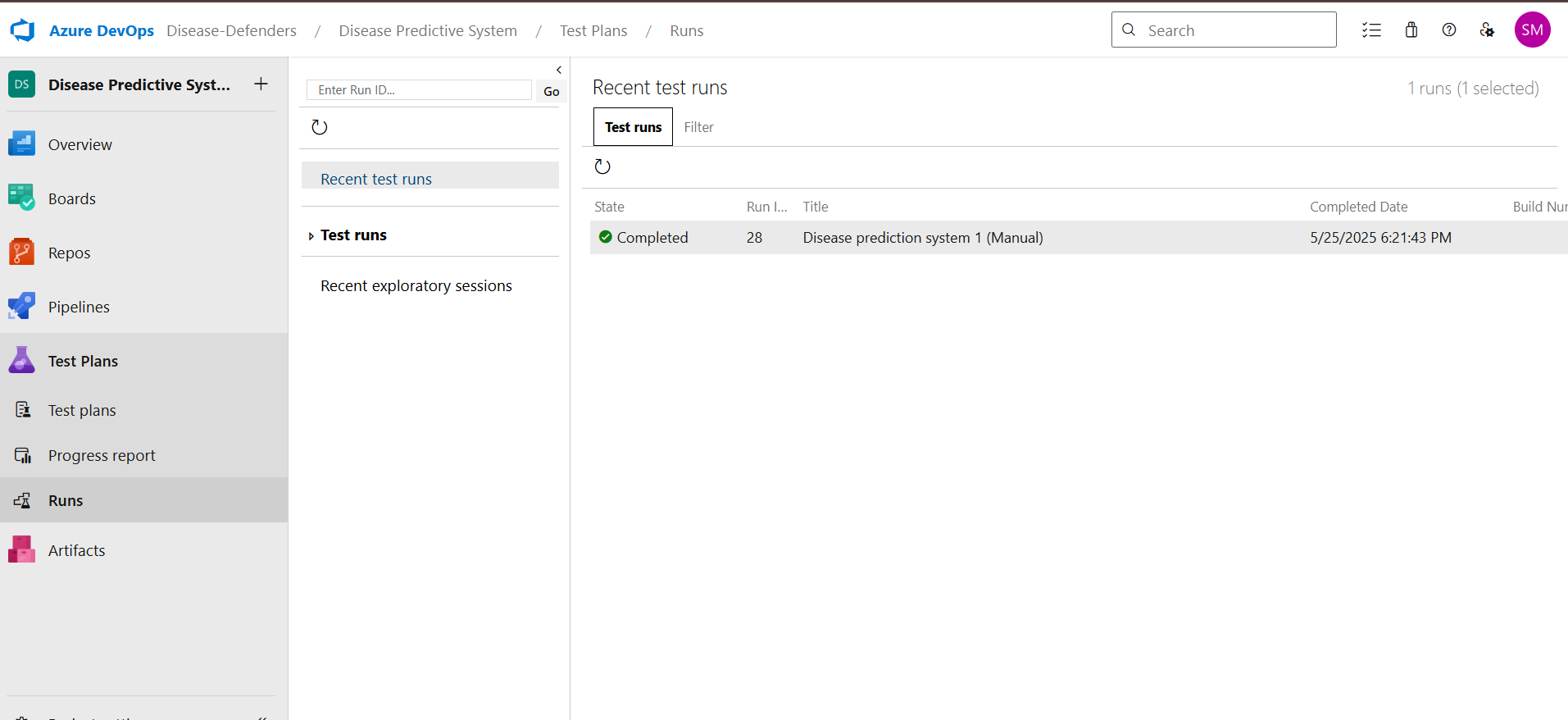




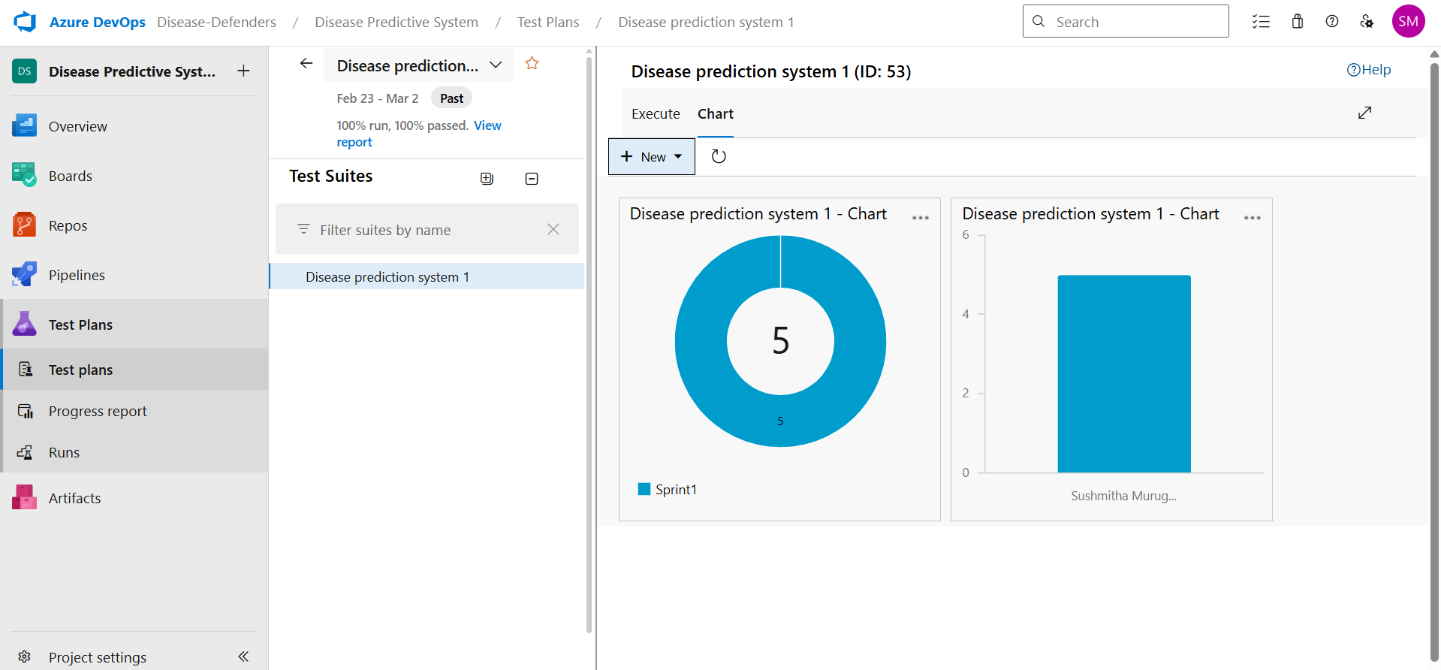


**8.Test case results**

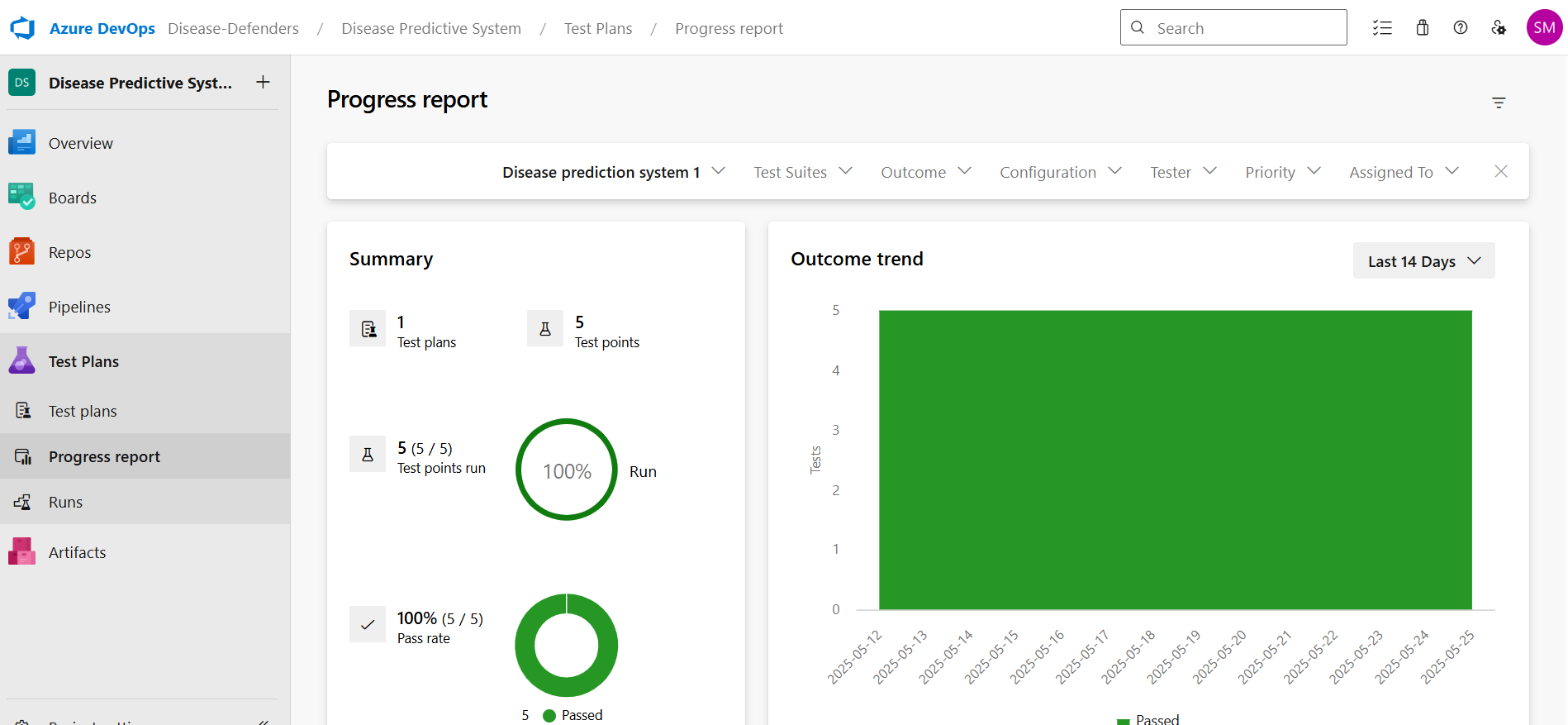
****

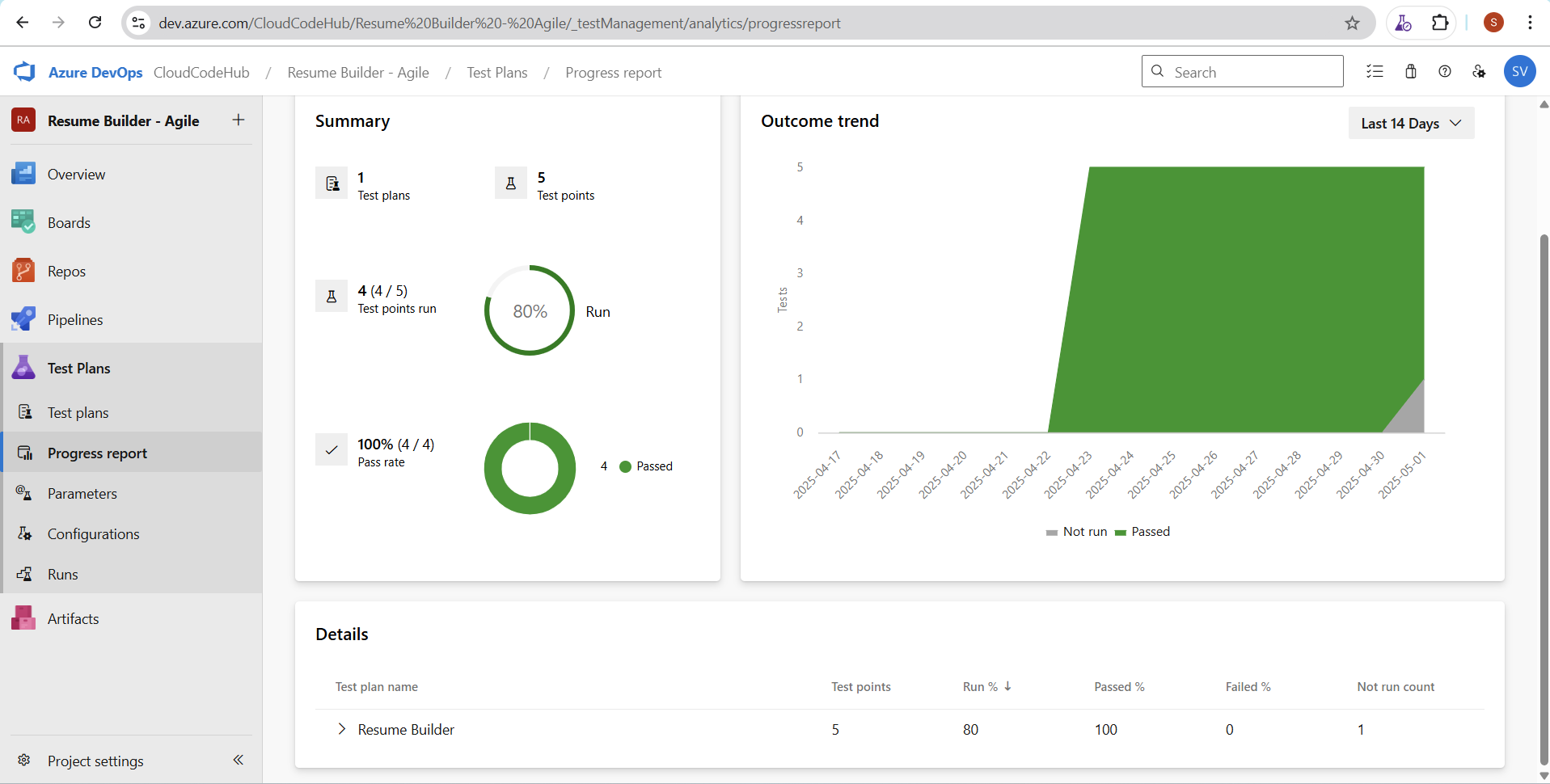


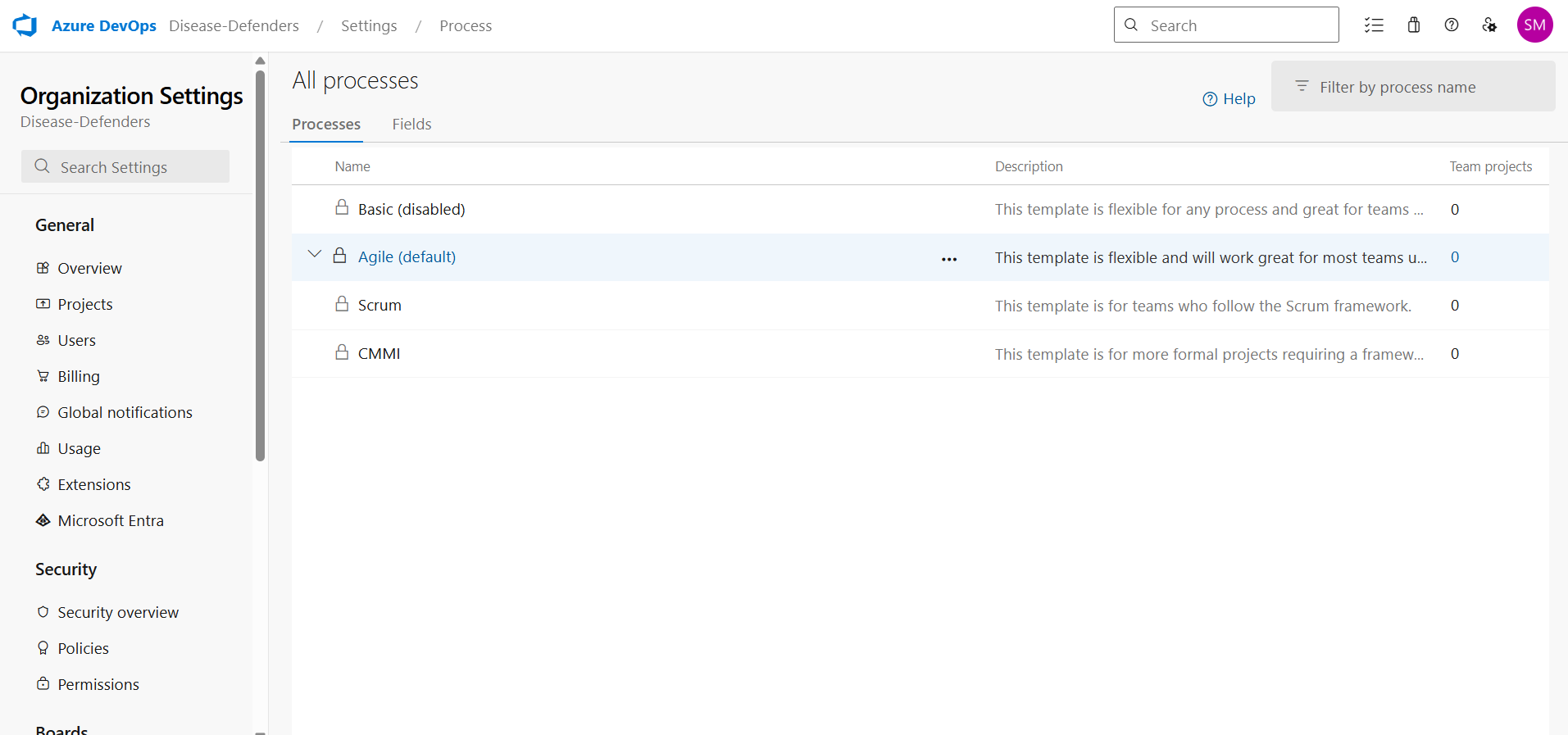
**9.Test report summary**

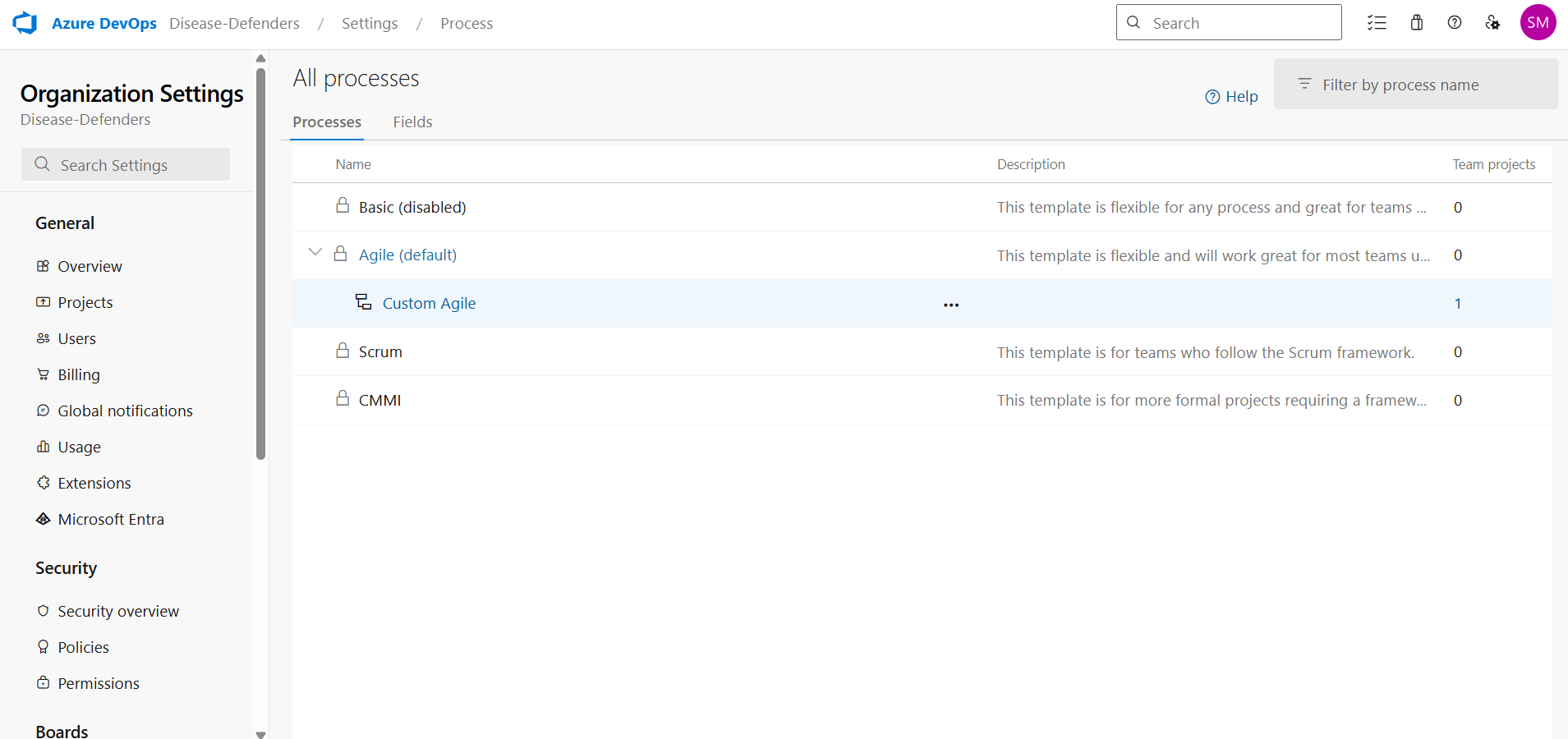


**10.Progress report**

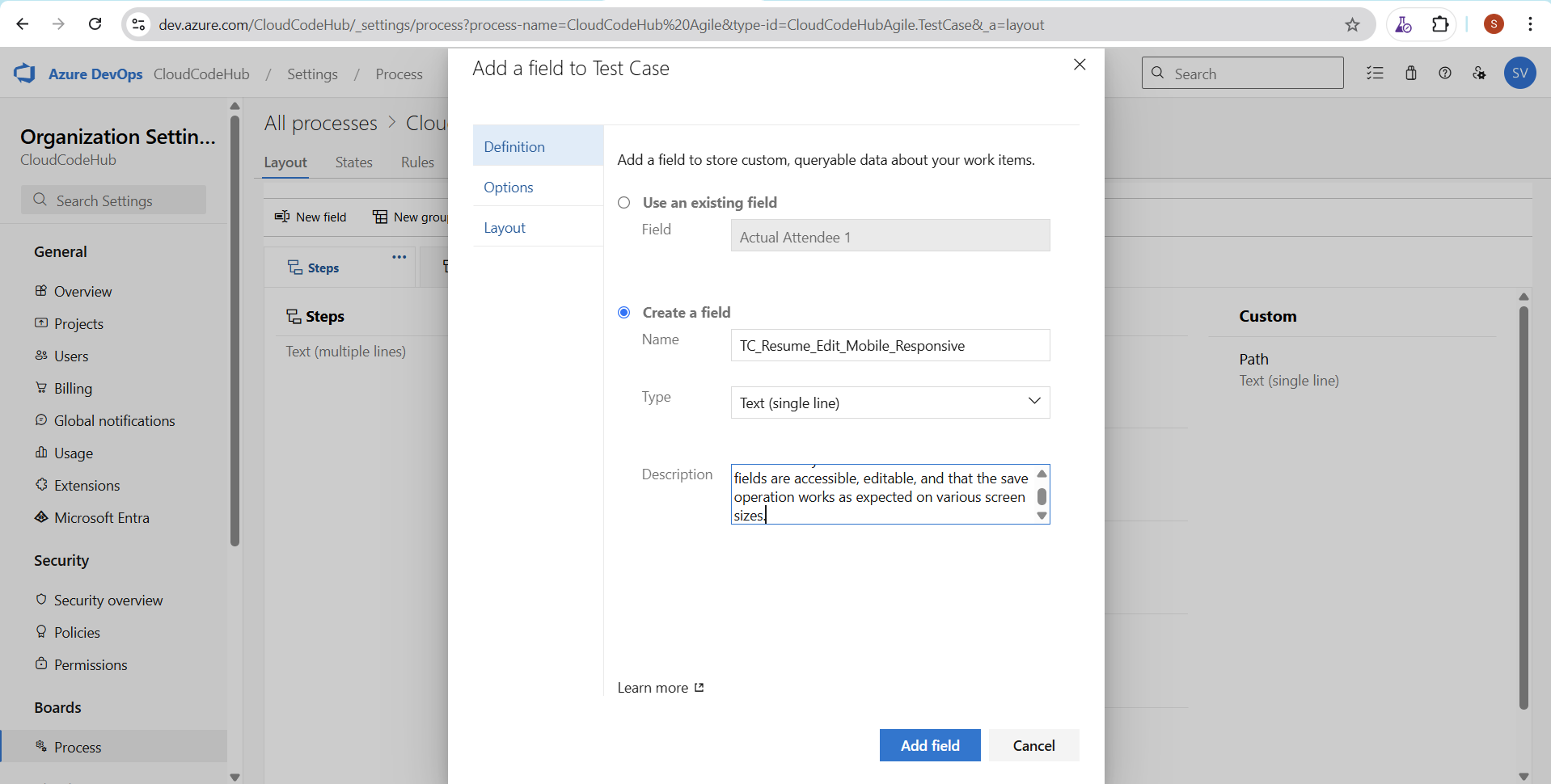


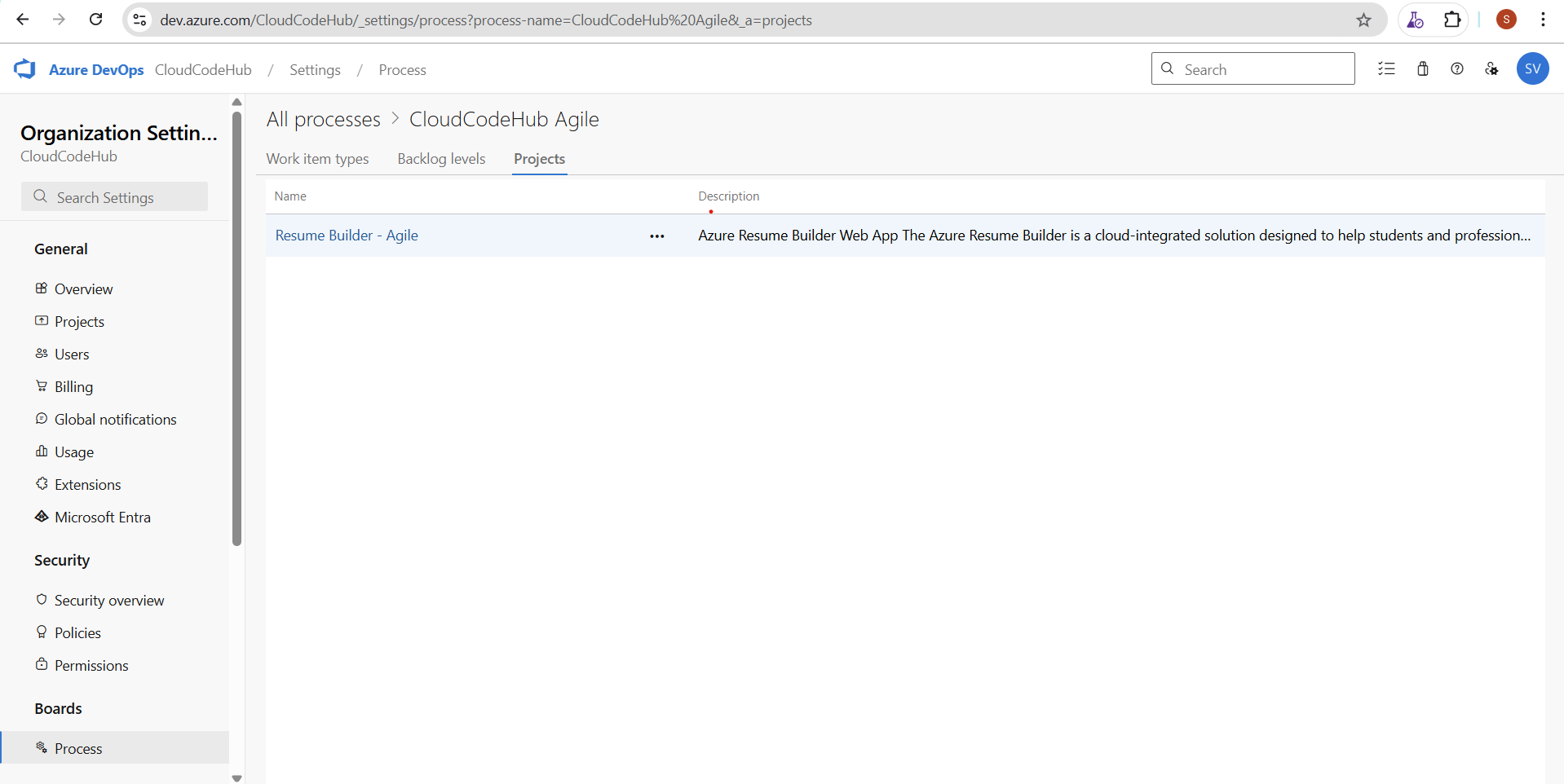


**11.Changing the test template** 



**12.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

Happy Path

|  |  |
| --- | --- |
| **EXP NO:** **9** | **LOAD TESTING AND PIPELINES** |

**Aim:**

To create an Azure Load Testing resource and run a load test to evaluate the performance of a target endpoint and to create and demonstrate an Azure DevOps pipeline for automating application builds, tests, and deployment.

**Load Testing**

Azure Load Testing:

Azure Load Testing allows you to simulate high traffic and stress tests for your web applications and APIs to understand how they perform under load. It helps identify performance bottlenecks, scalability issues, and optimize resource usage before deployment.

**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 t[o https://portal.azure.com](https://portal.azure.com/) and log in.

1. Create the Resource o Go to *Create a resource* → Search for “Azure Load Testing”.

o Select Azure Load Testing and click Create.

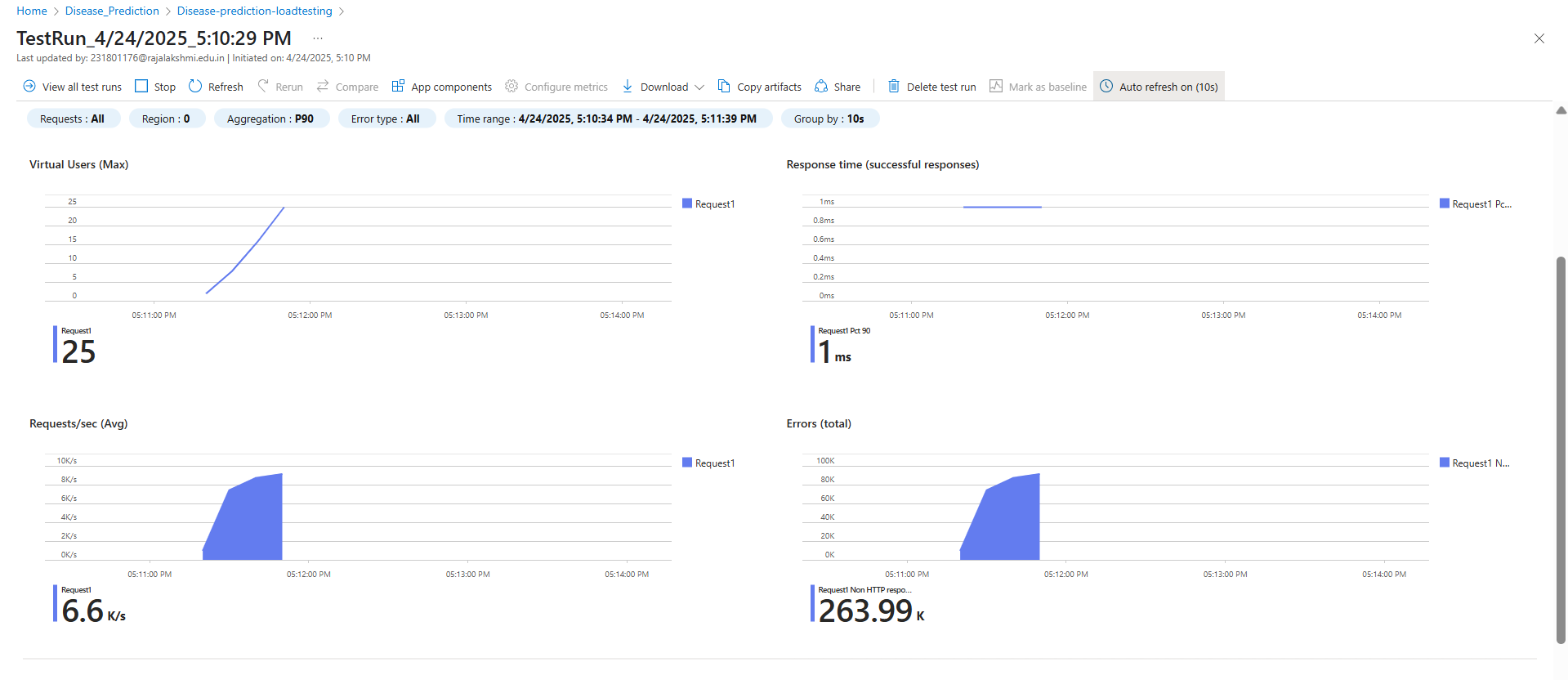
1. Fill in the Configuration Details o Subscriptio*n:* Choose your Azure subscription. o *Resource* *Group:* Create new or select an existing one. o *Name:* Provide a unique name (no special characters). o *Location:* Choose the region for hosting the resource.
2. (Optional) Configure tags for categorization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

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

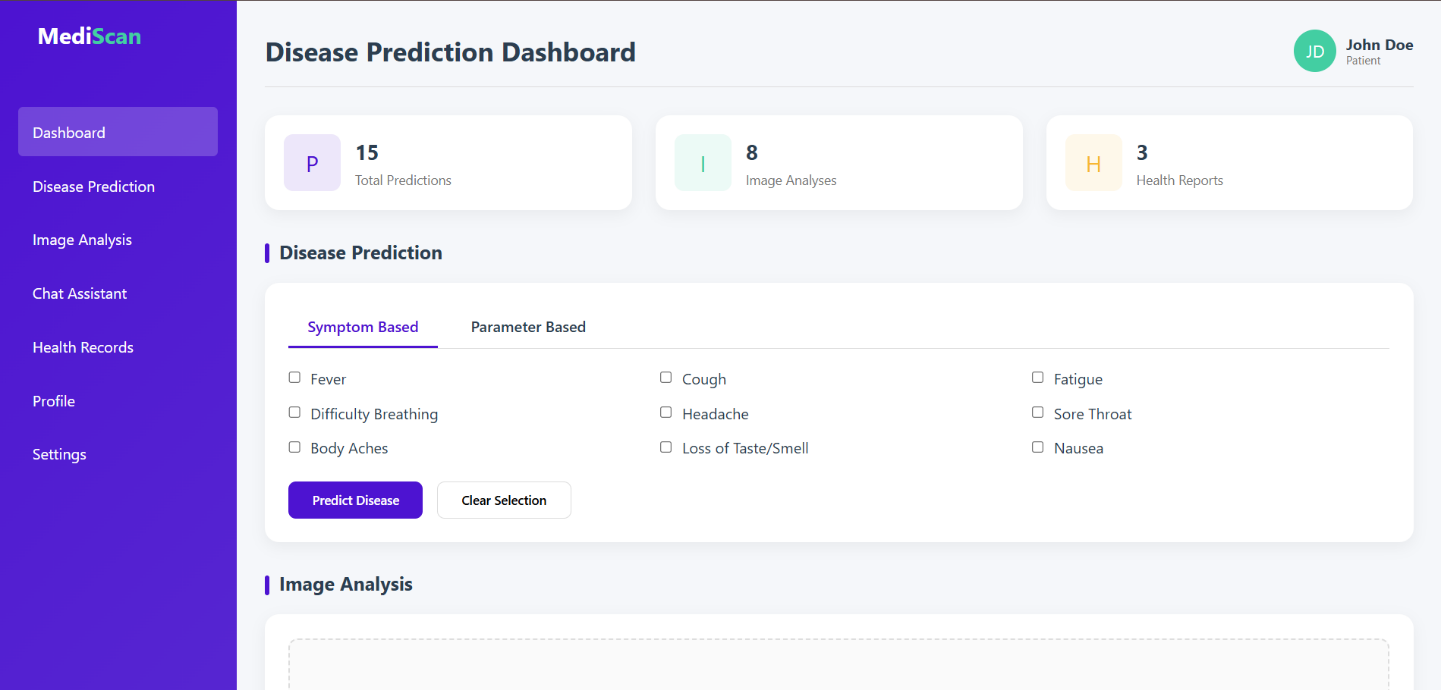
Once your resource is ready:

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

**Load Testing**

****





**Pipelines**

**Description:**

This experiment demonstrates how to connect a GitHub-hosted Flask-based music recommendation project with Azure DevOps. The pipeline will automatically install dependencies, run basic tests, and publish artifacts. This ensures that every commit triggers checks for reliability and smooth deployment.

**Steps:**

1. Connect GitHub to Azure DevOps:
   * In Azure DevOps, create a new project. o Create a pipeline and select GitHub as the source.
   * Authorize access to your GitHub repository, ensuring that Azure DevOps can pull the repository for your pipeline.
2. Create azure-pipelines.yml in Your Repo Root:
   * In your GitHub repository, create a new file called azure-pipelines.yml in the root directory.
   * Add the following basic pipeline configuration for Python and Flask:

YAML CODE:-

trigger:

- main # Trigger pipeline when changes are pushed to the main branch

pool:

vmImage: ubuntu-latest # Use a hosted Ubuntu agent

steps:

# Step 1: Checkout code

- checkout: self

# Step 2: Set up Python environment (if your project uses Python backend tools)

- task: UsePythonVersion@0

inputs:

versionSpec: '3.x' # Use the latest Python 3.x version

displayName: "Set up Python"

# Step 3: Install dependencies

- script: |

python -m pip install --upgrade pip

pip install -r resume-builder/requirements.txt # Adjust this path as per your repo

displayName: "Install dependencies"

# Step 4: Run a basic script check

- script: |

python -c "print('✅ Resume Builder Pipeline Executed Successfully!')"

displayName: "Run a test script"

# Optional Step 5: Run tests

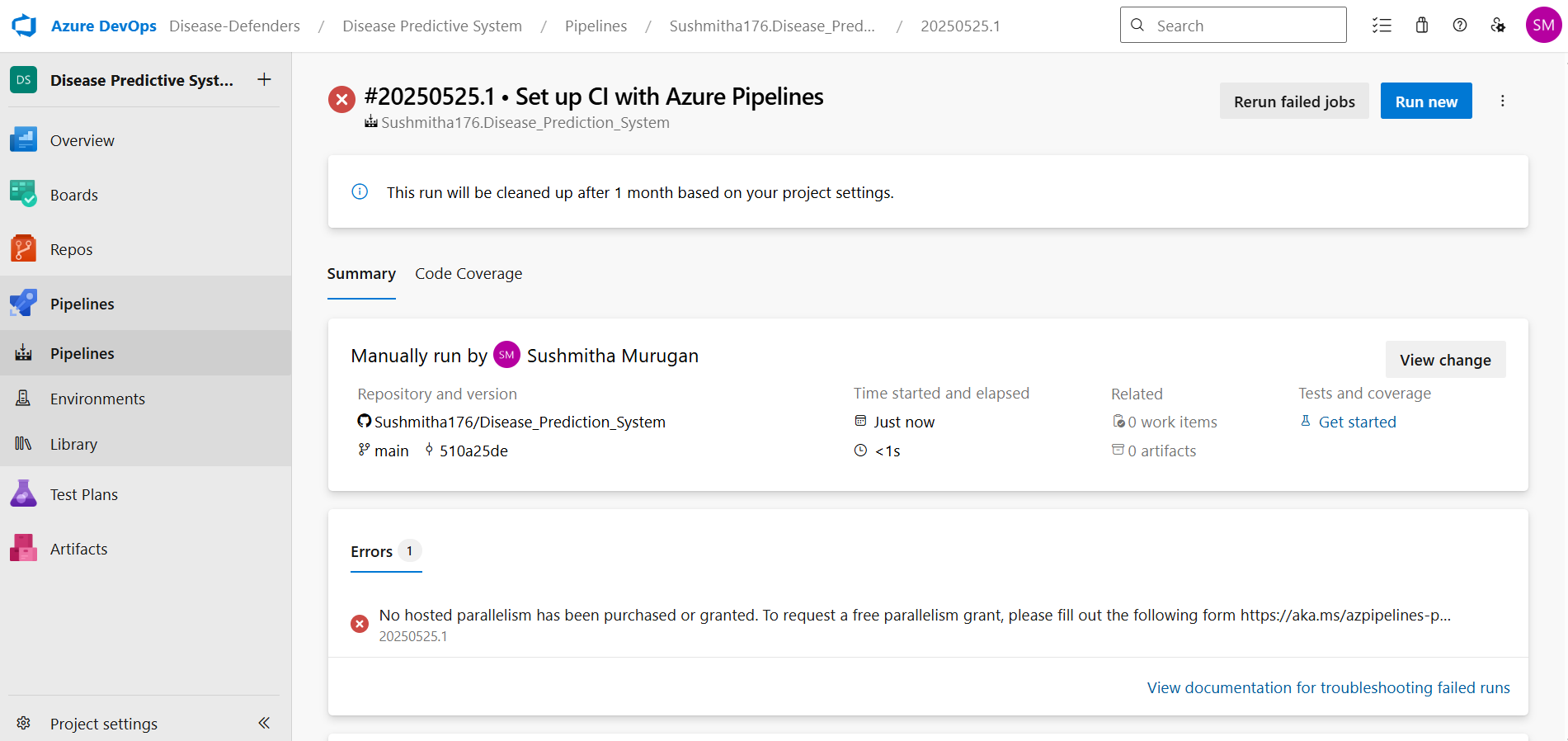
- script: |

pytest tests/ # Adjust this path to your test folder

displayName: "Run unit tests"

* 1. Pipeline Tasks Include:
     + Setting up the Python environment using the UsePythonVersion task.
     + Installing project dependencies from project/requirements.txt. Make sure the path to requirements.txt is correct (it is located under the project folder).
     + Running a simple Python script to verify that Python is set up correctly and the pipeline works.
  2. Run and Monitor Pipeline:
     + Commit changes to the main branch of your repository to trigger the pipeline in Azure DevOps.
     + Monitor the logs in the Azure DevOps portal to view logs, errors, or success messages and ensure everything runs smoothly.

**Pipeline**

****

**Result:**

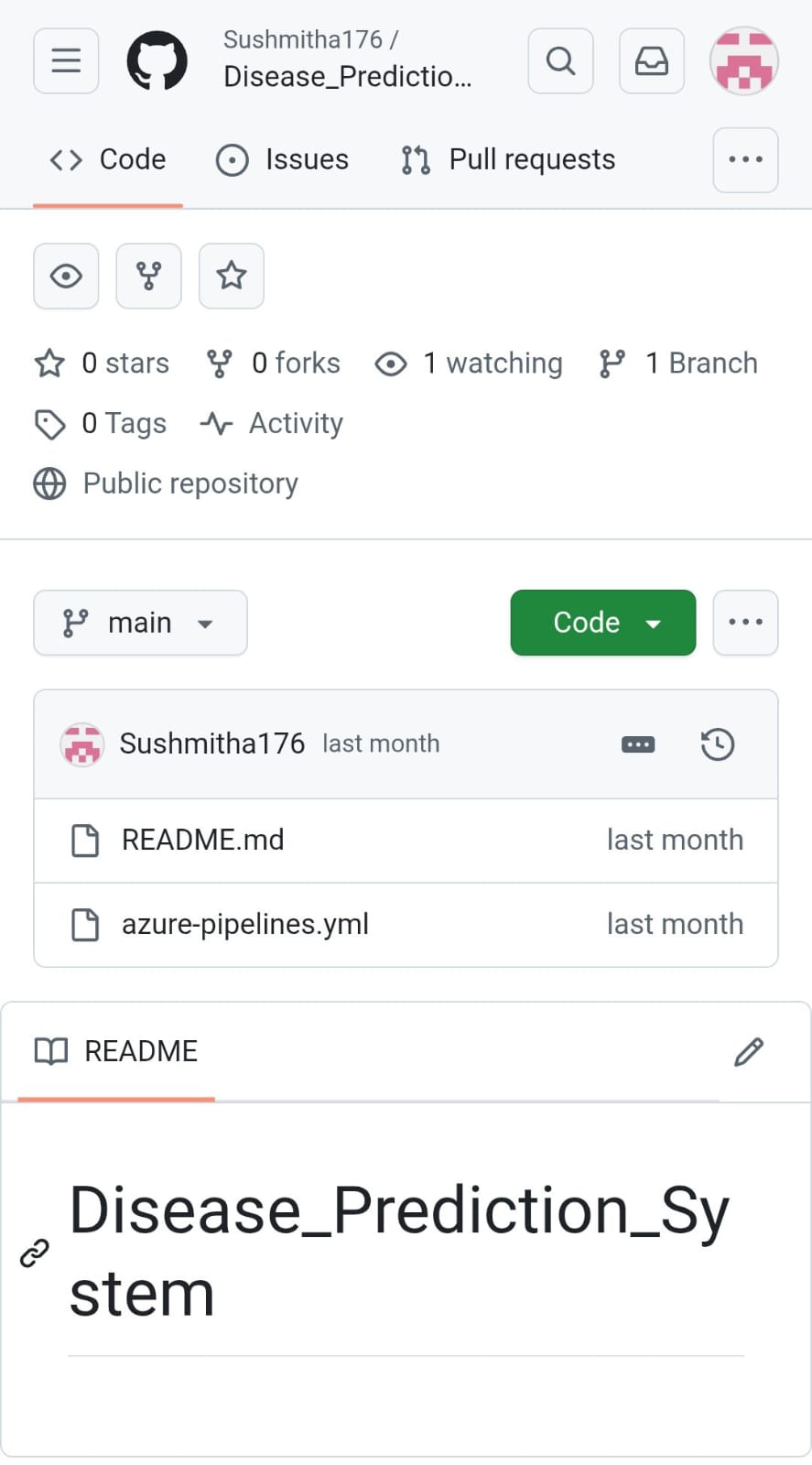
Successfully created the Azure Load Testing resource and executed a load test to assess the performance of the specified endpoint and also demonstrated pipelines in azure devops.

|  |  |
| --- | --- |
| **EXP NO:** **10** | **GITHUB: PROJECT STRUCTURE & NAMING**  **CONVENTIONS** |

**Aim:**

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 Resume Builder 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.