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

# AIM:-

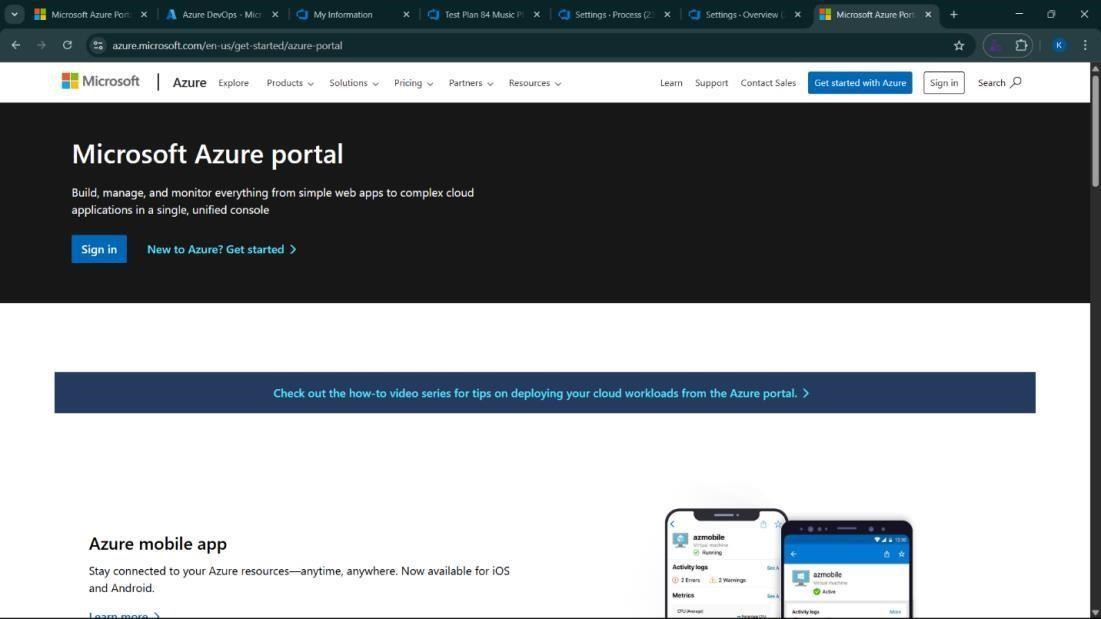
To set up and access the Azure DevOps environment by creating an organization through the Azure portal.

# INSTALLATION:-

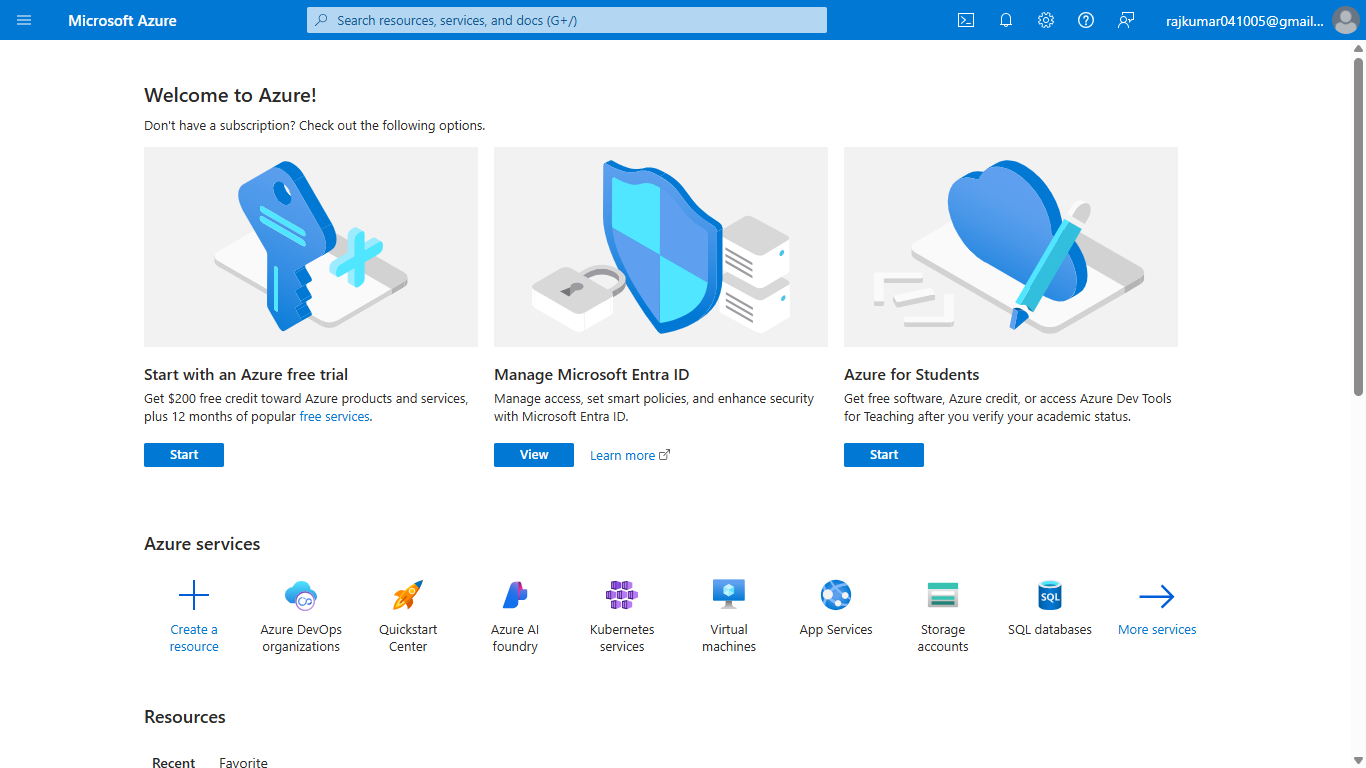
1. Open your web browser and go to the Azure website: [https://azure.microsoft.com/en-us/get- started/azure-portal.](https://azure.microsoft.com/en-us/get-started/azure-portal)

Sign in using your Microsoft account credentials.

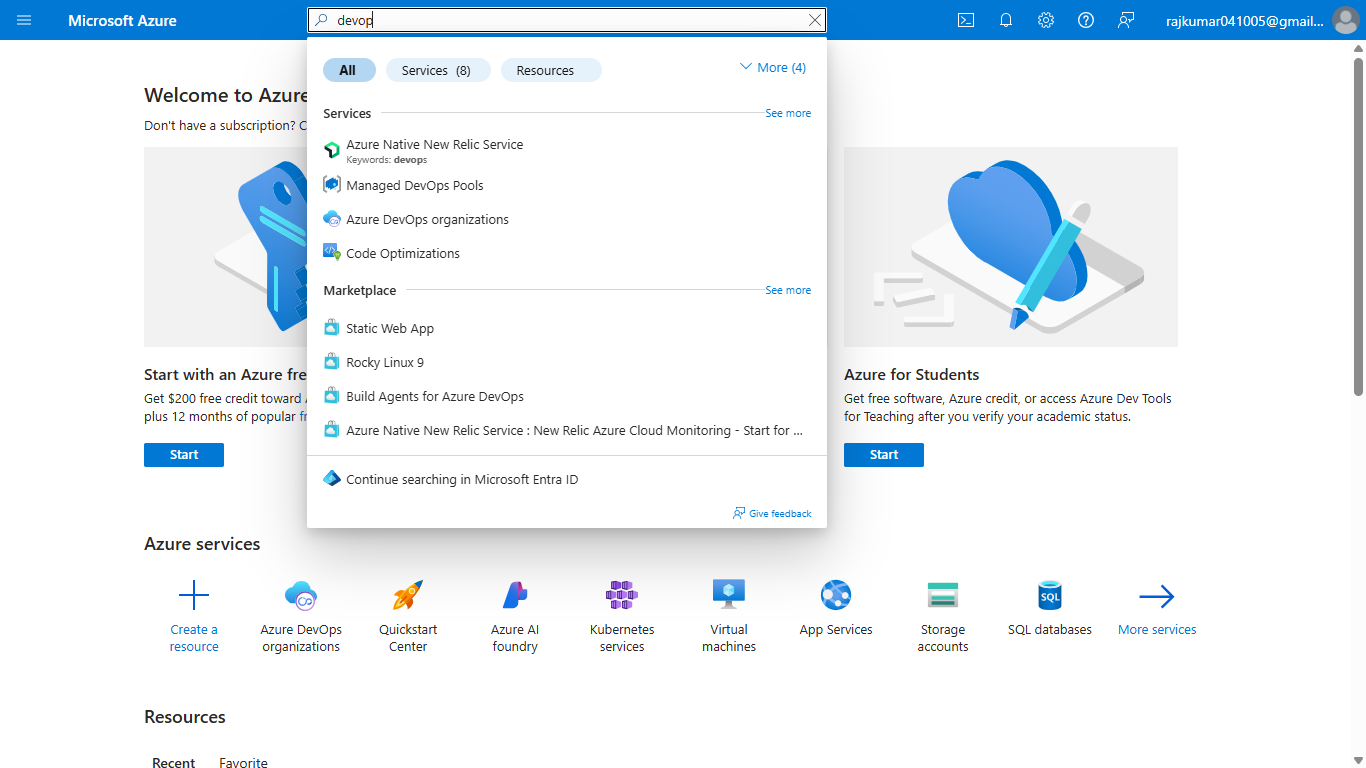
If you don't have a Microsoft account, you can create one here:<https://signup.live.com/?lic=1>



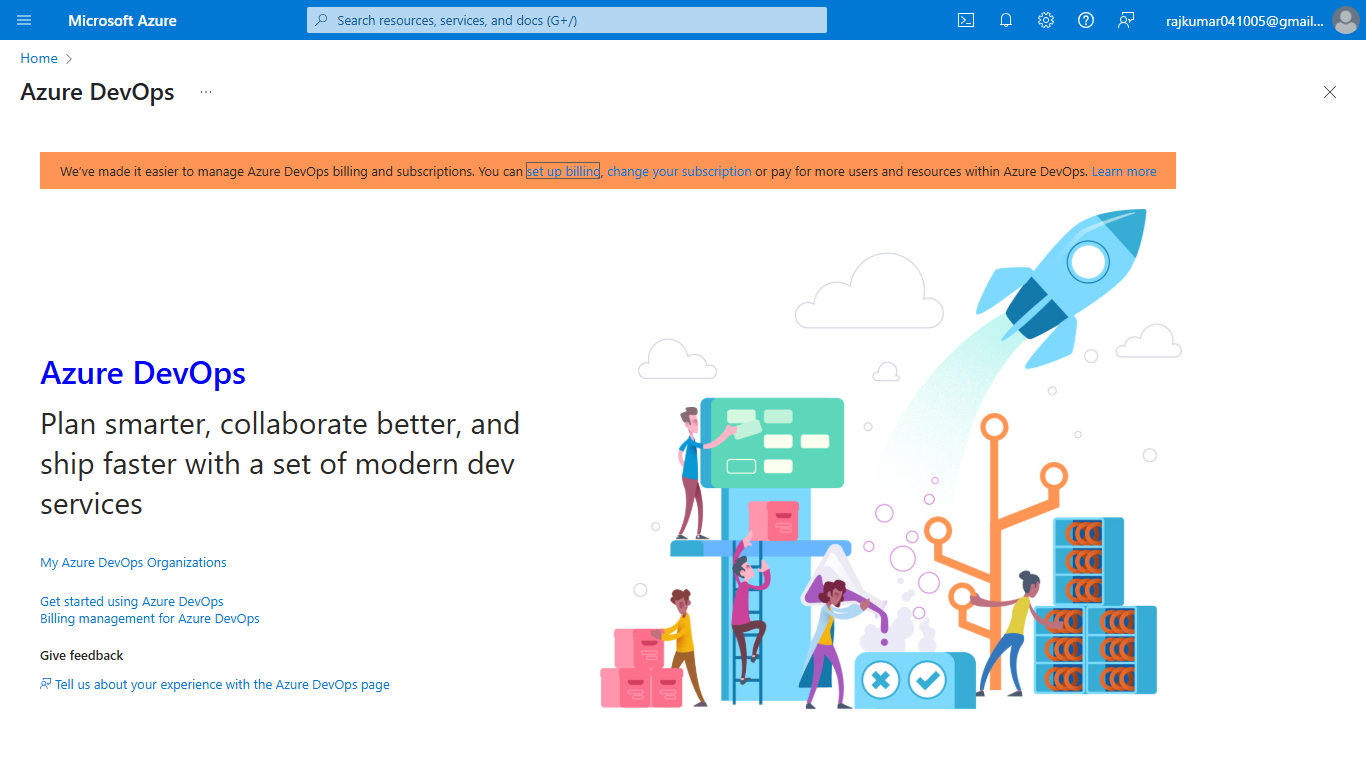
1. Azure home page



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:**

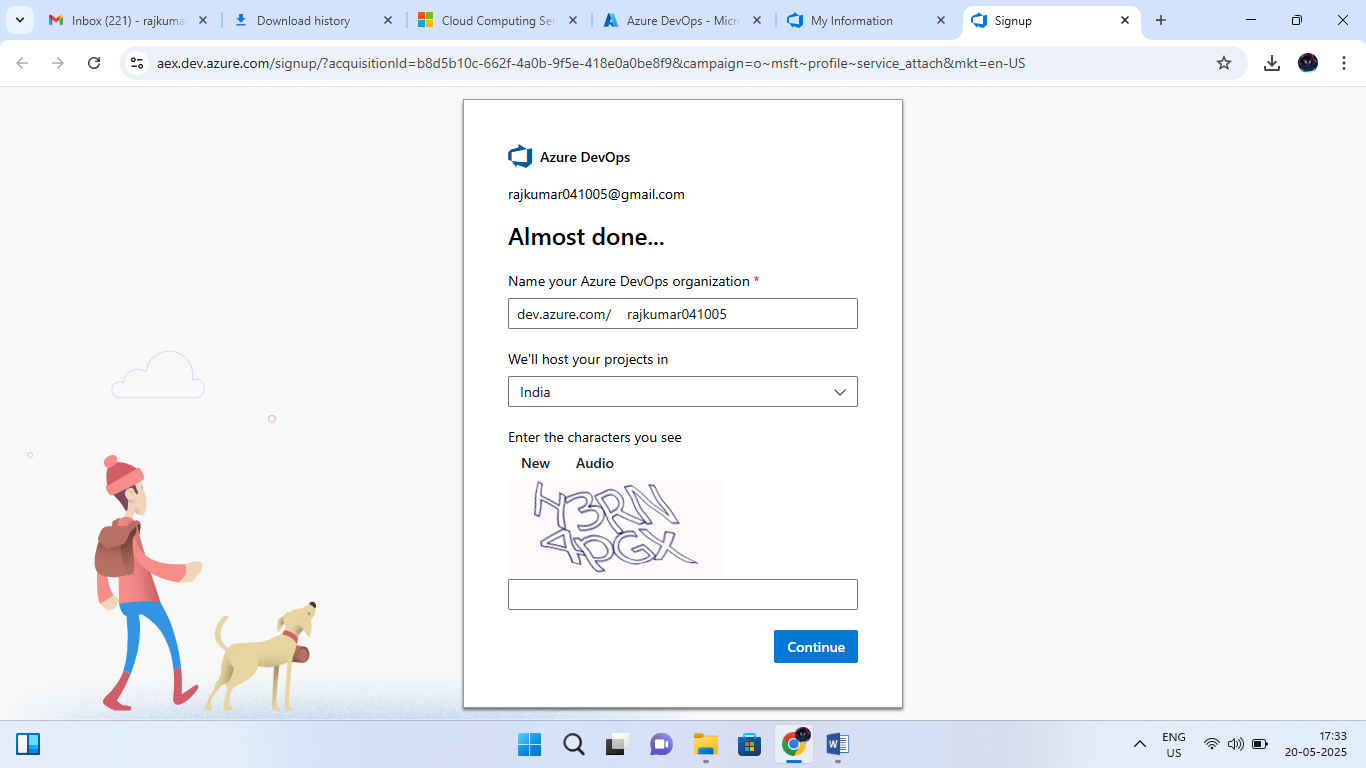
Successfully accessed the Azure DevOps environment and created a new organization through the Azure portal.

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

# AIM:-

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

1. Create An Azure Account

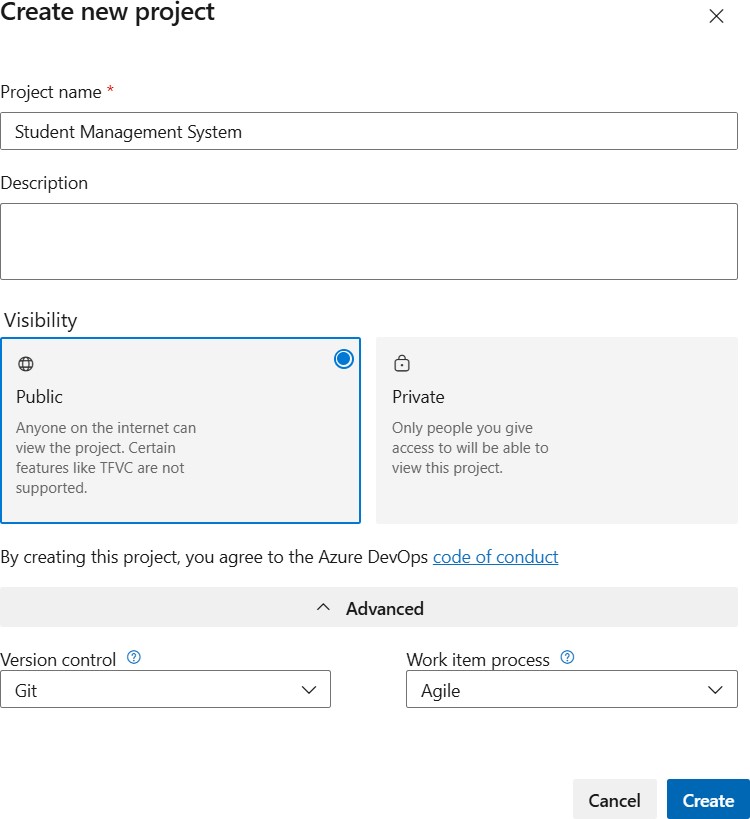


1. Create the First Project in Your Organization
   1. After the organization is set up, you’ll need to create your first **project**. This is where you'll begin to manage code, pipelines, work items, and more.
   2. On the organization’s **Home page**, click on the **New Project** button.
   3. Enter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

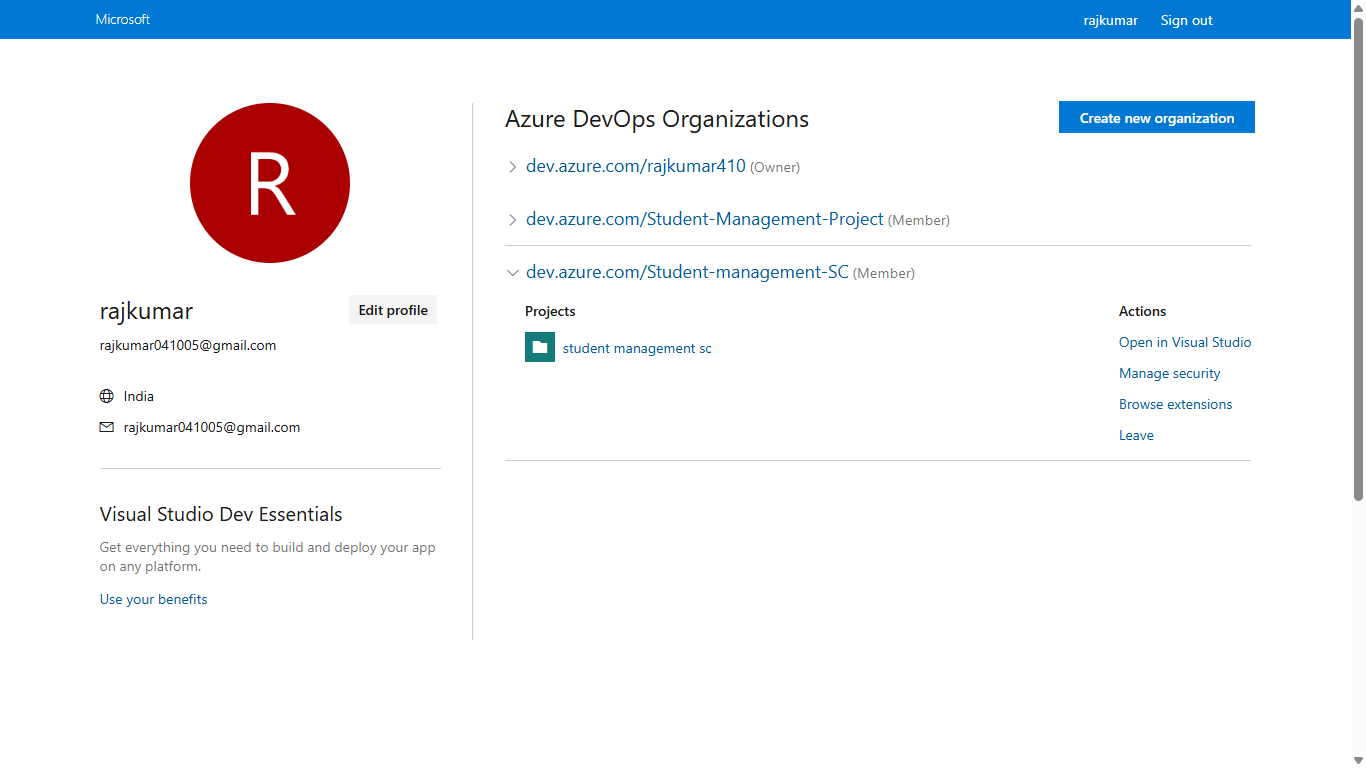
***Name****:* Choose a name for the project (e.g., LMS).

***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).

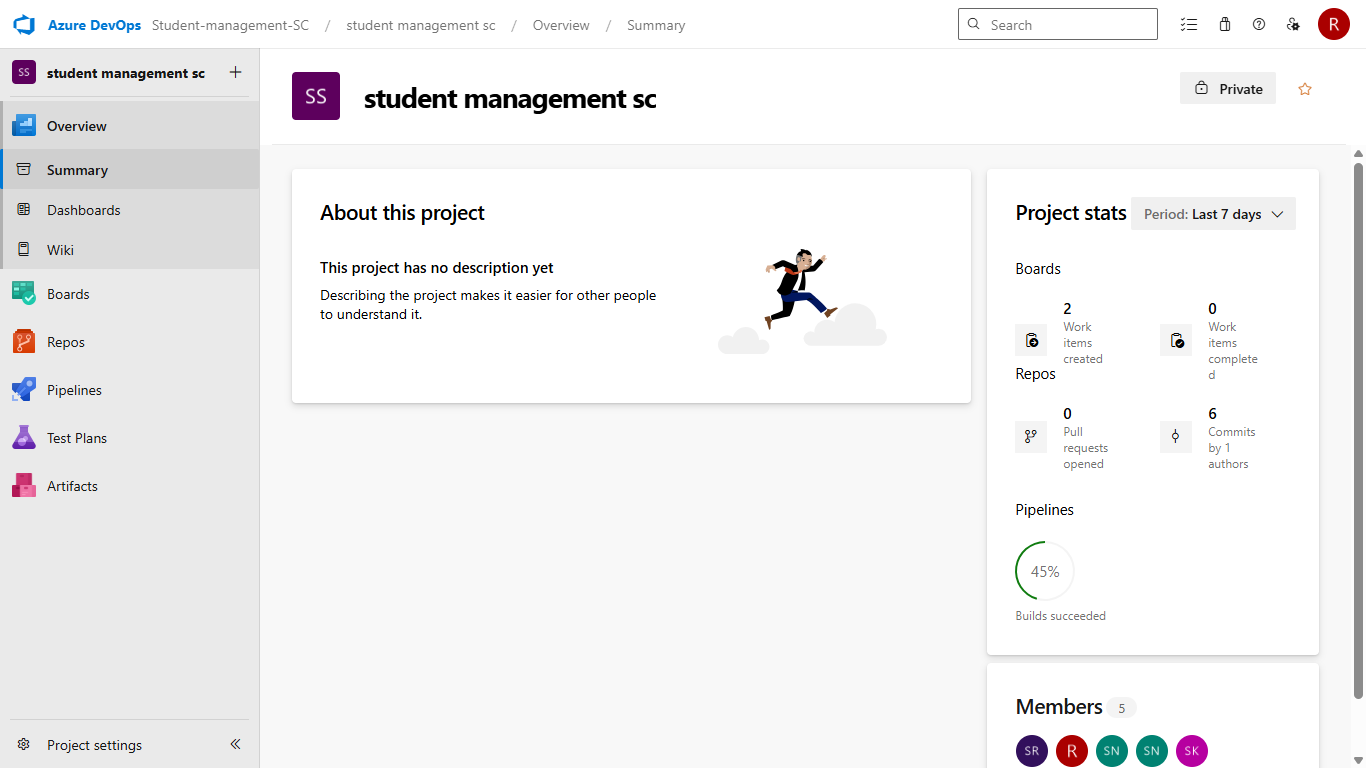
* 1. Once you’ve filled out the details, click **Create** to set up your first project.



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

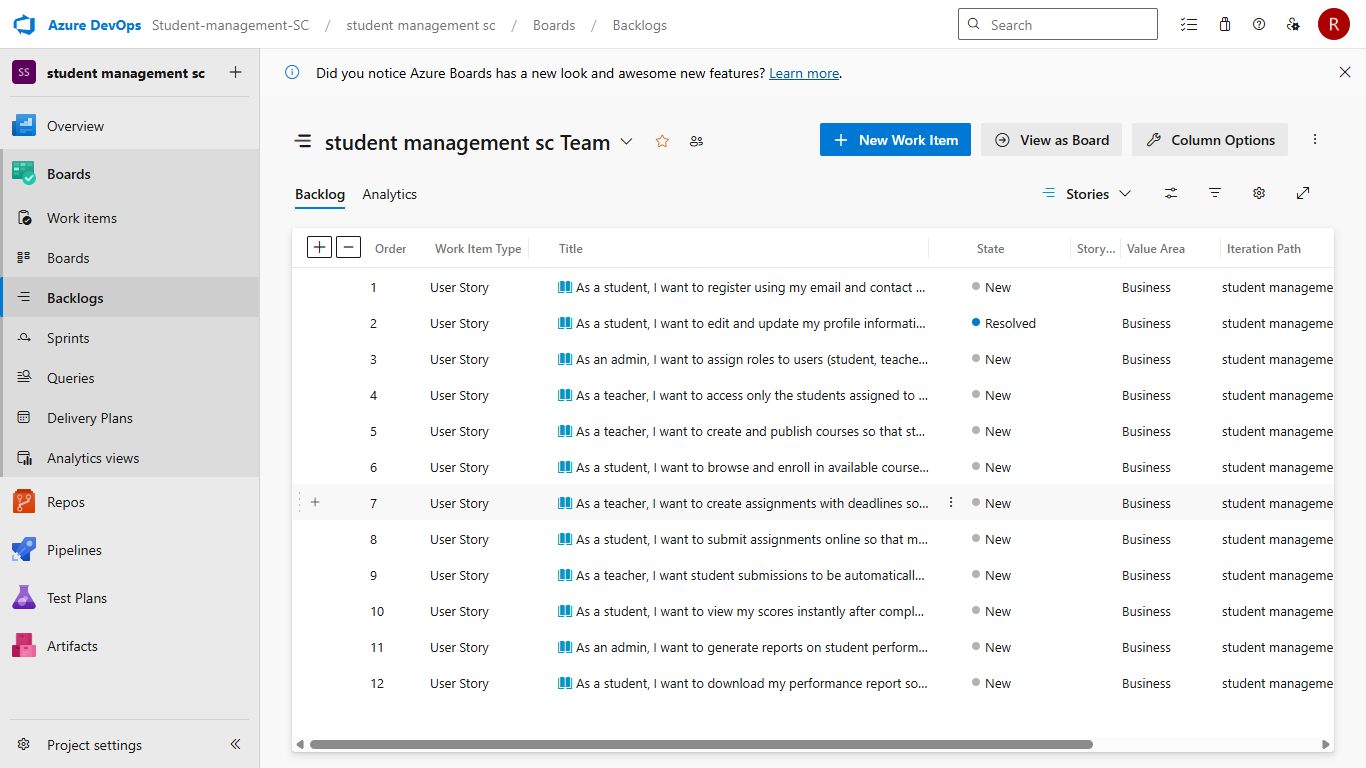


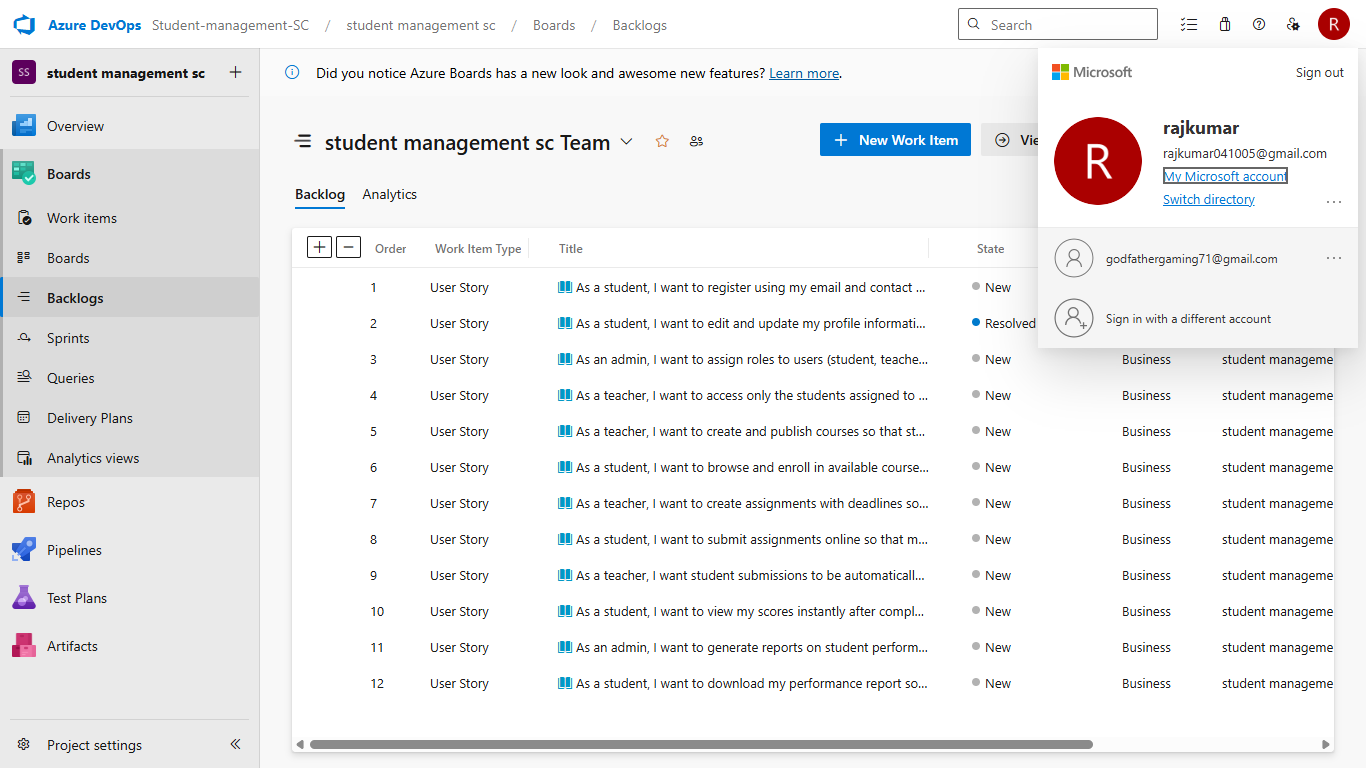
1. To manage user stories:
   1. From the **left-hand navigation menu**, click on **Boards**. This will take you to the main

**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:**

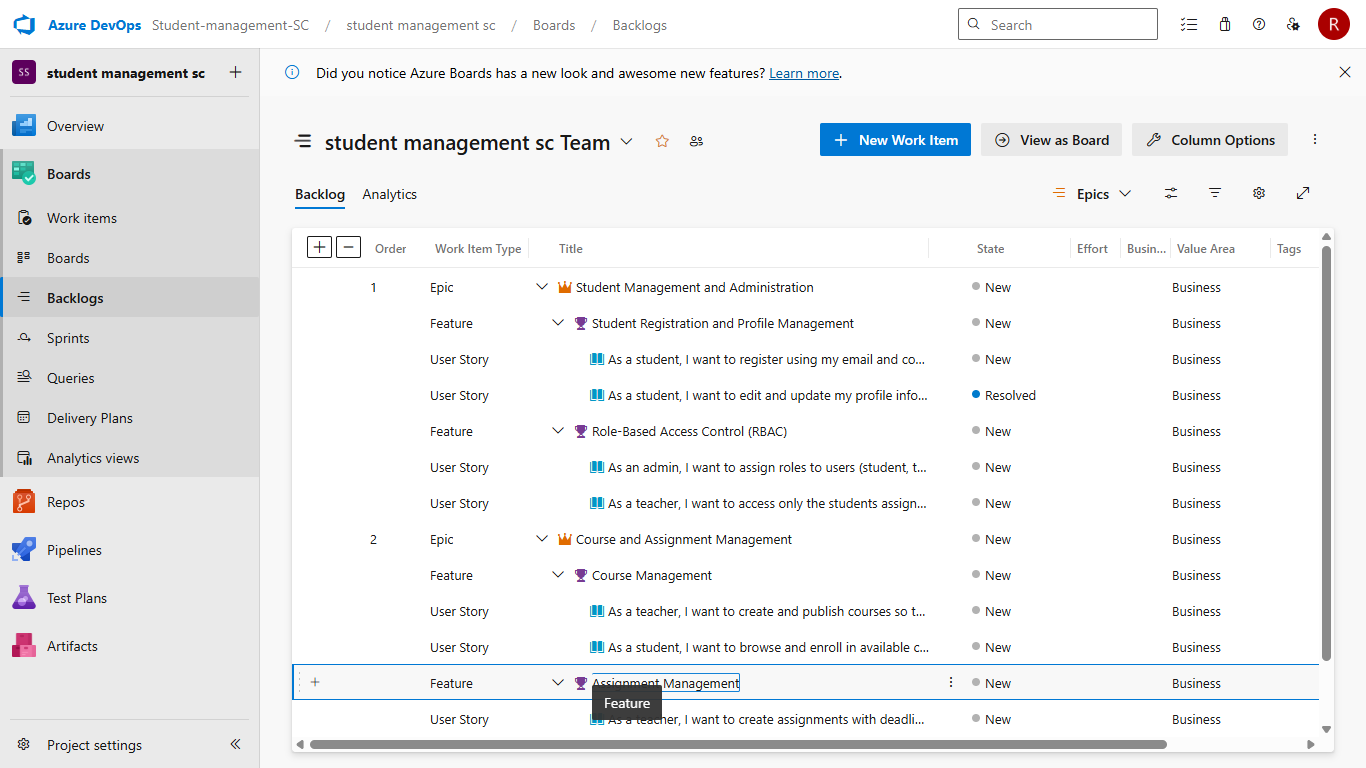
Successfully created an Azure DevOps project with user story management and agile workflow setup.

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

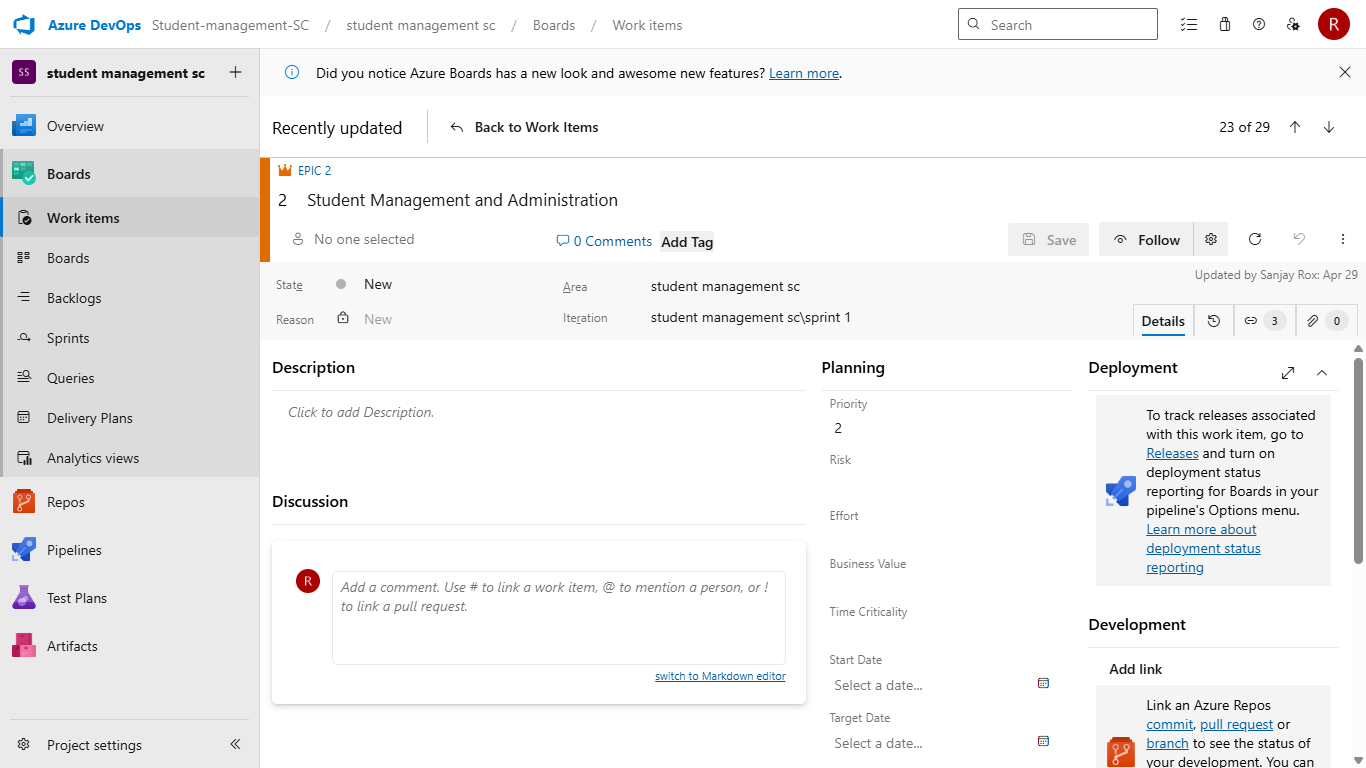
**AIM:-**

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

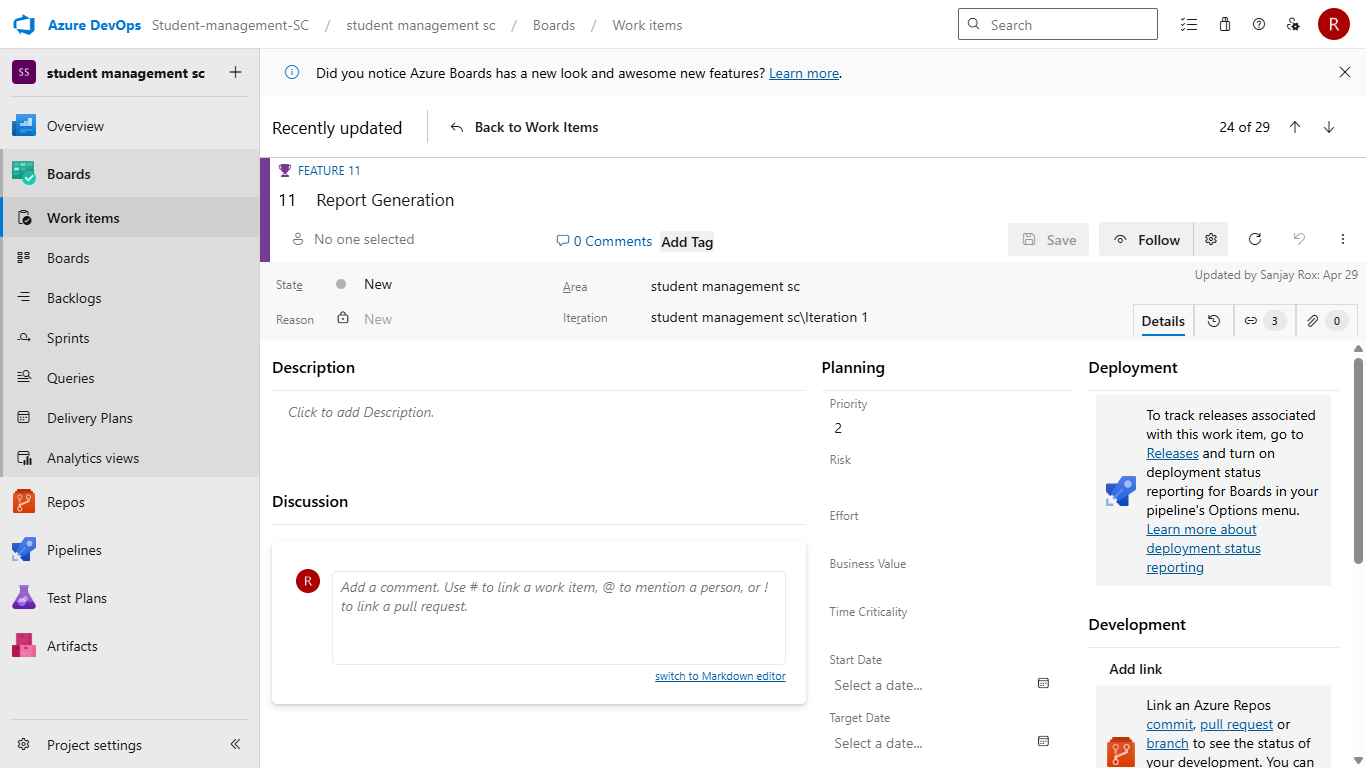
# Create Epic, Features, User Stories, Task



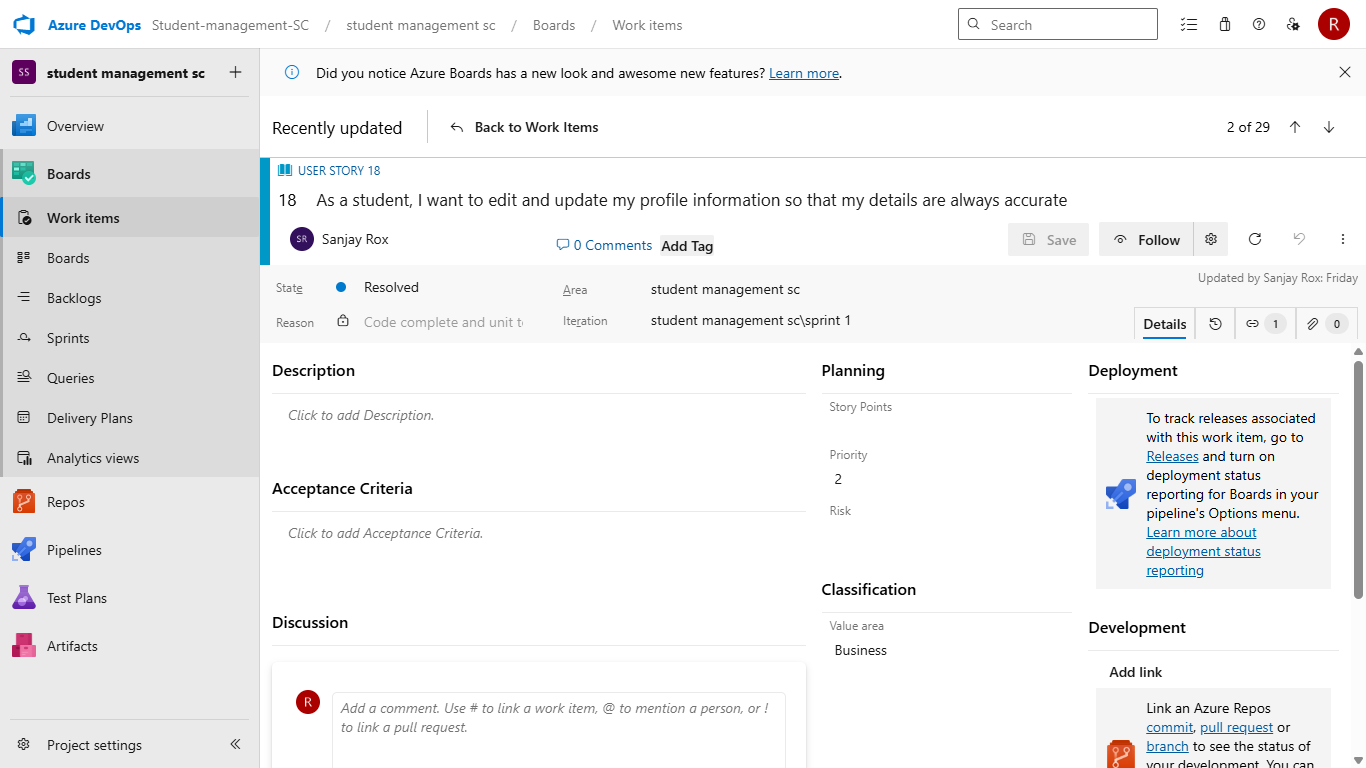
## 1. Fill in Epics



## 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.

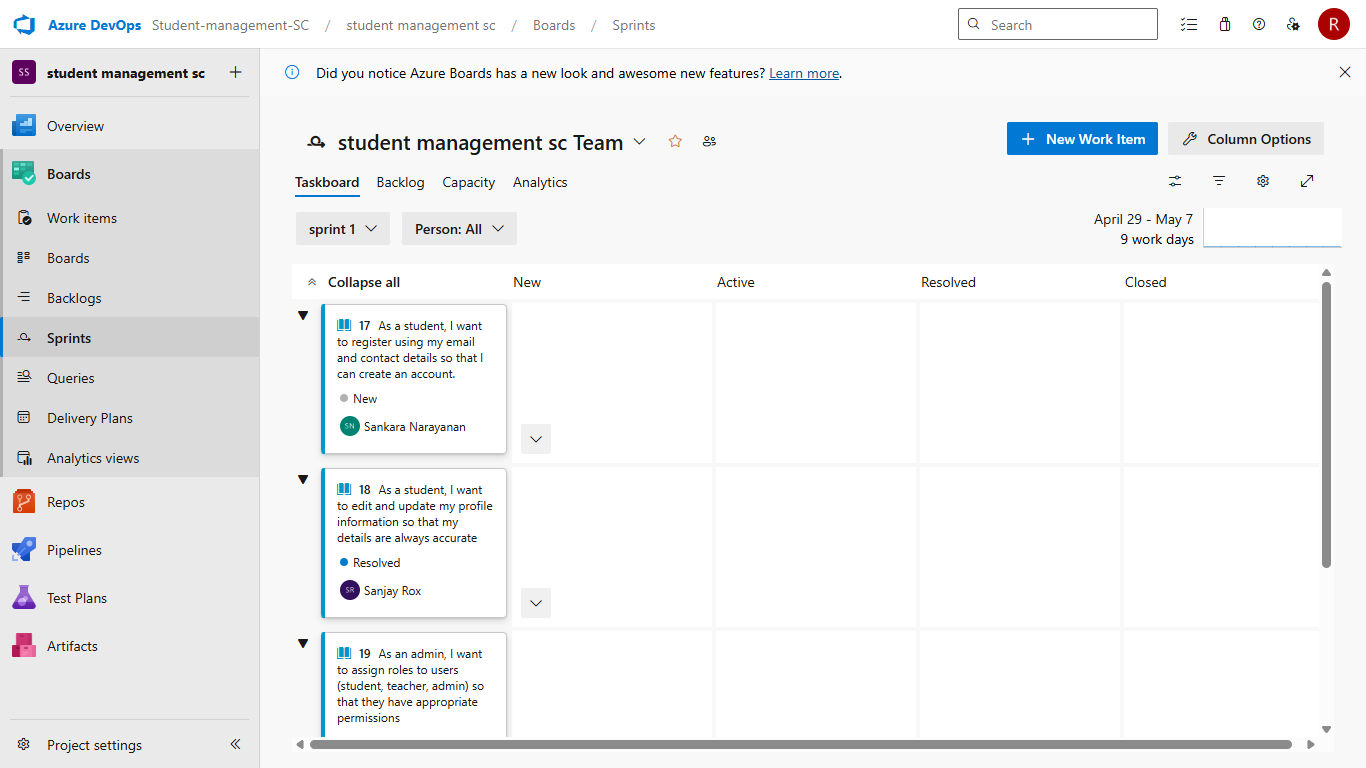
|  |  |
| --- | --- |
| **EXP NO: 4 DATE:** | **SPRINT PLANNING** |

**AIM:-**

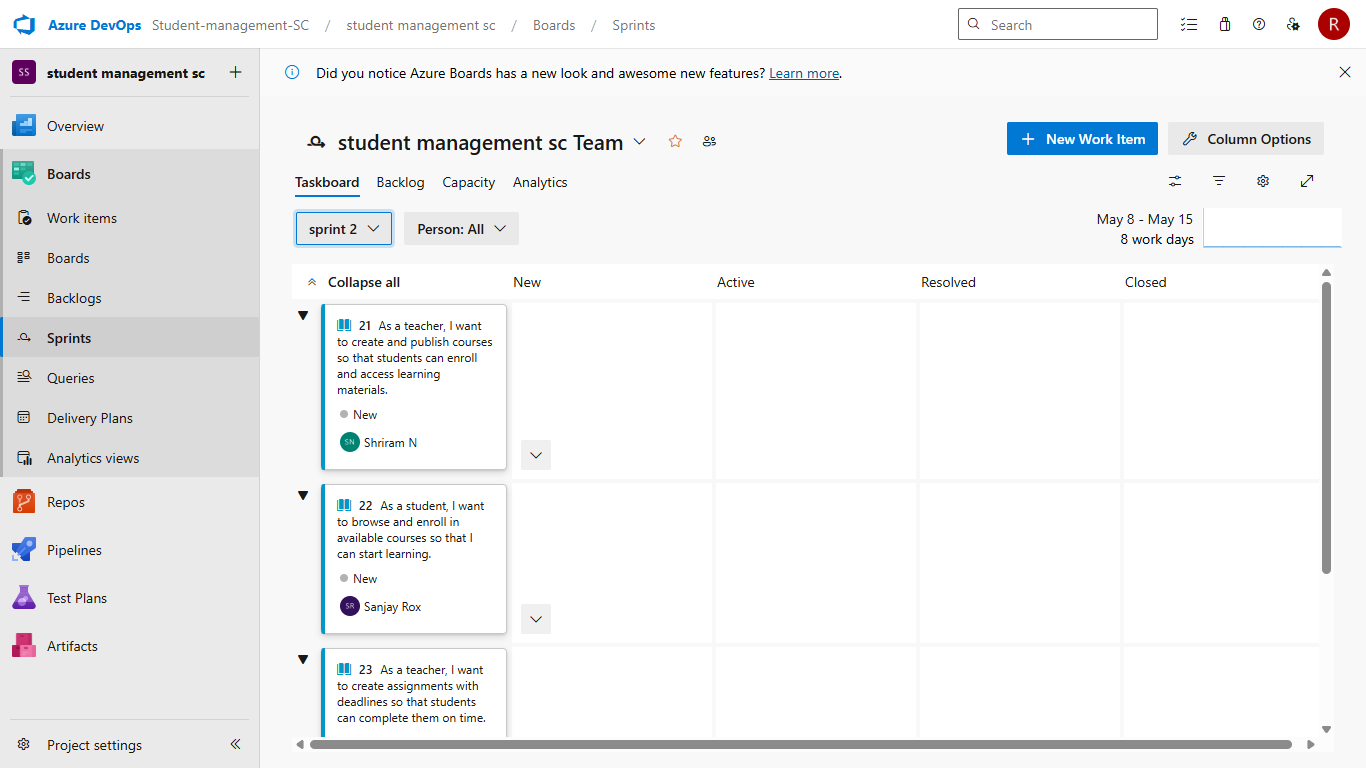
To assign user story to specific sprint for the Student Management System Project.

# Sprint Planning:-

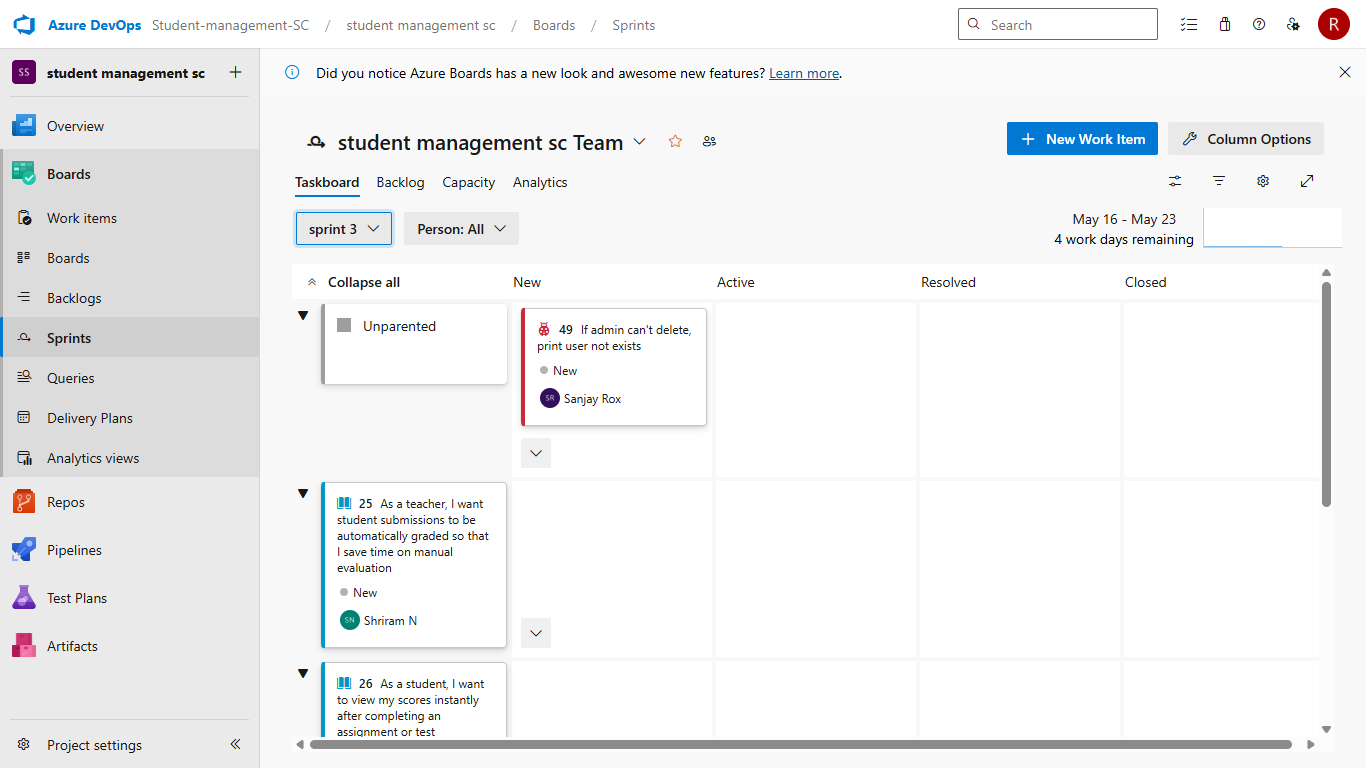
**Sprint 1**



**Sprint 2**



**Sprint 3**



# RESULT:-

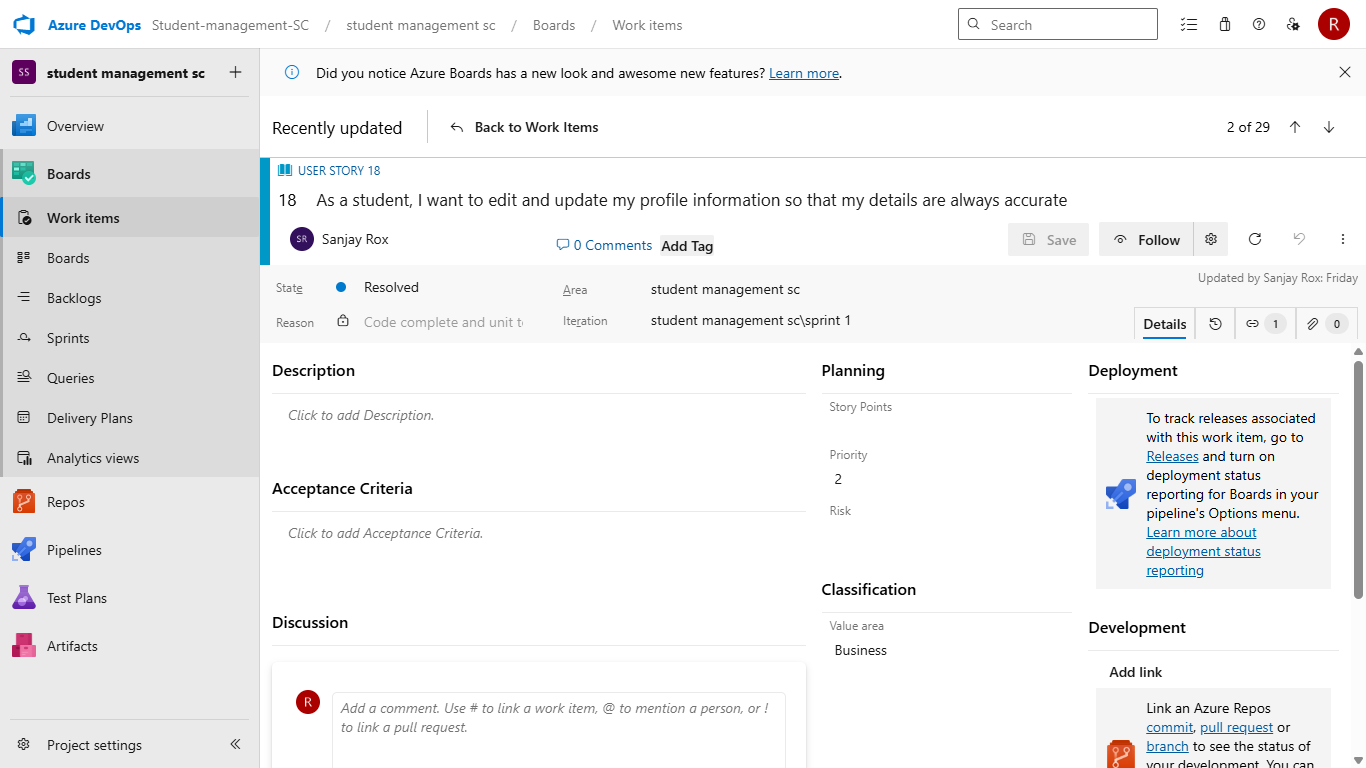
The Sprints are created for the Student Management System.

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

**AIM:-**

Create Poker Estimation for the user stories -Student management System.

# Poker Estimation:-



# RESULT:-

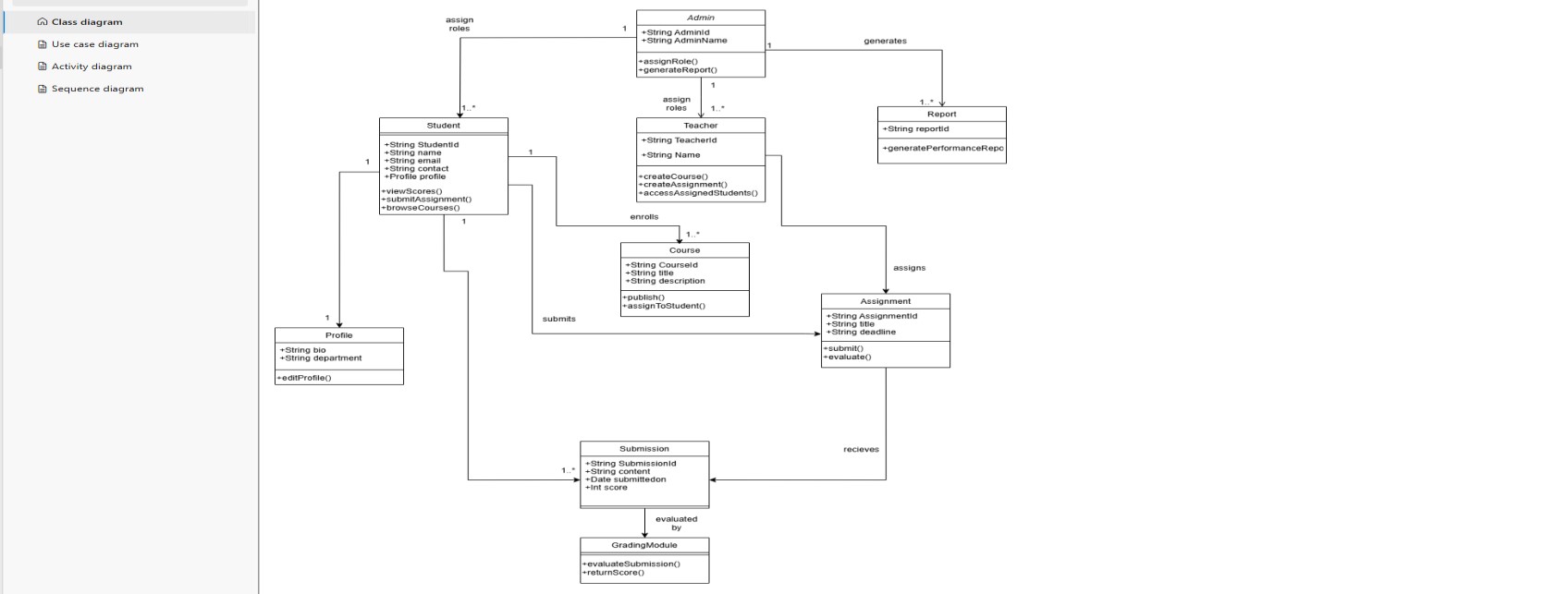
The Estimation/Story Points is created for the project using Poker Estimation.

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

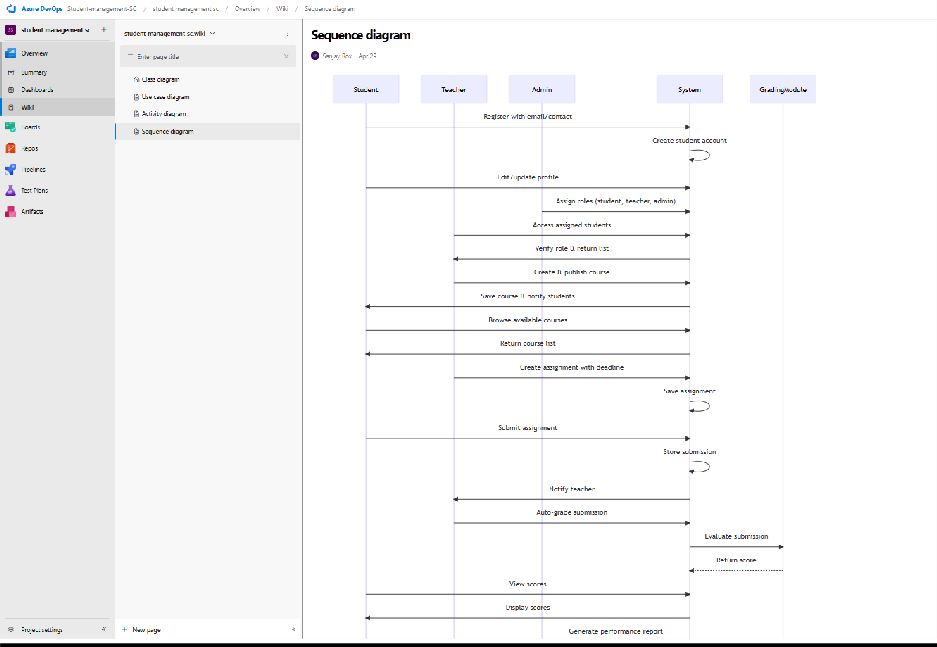
# AIM:-

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

## 6A. CLASS DIAGRAM:-



# 6B. SEQUENCE DIAGRAM:-



**Result:**

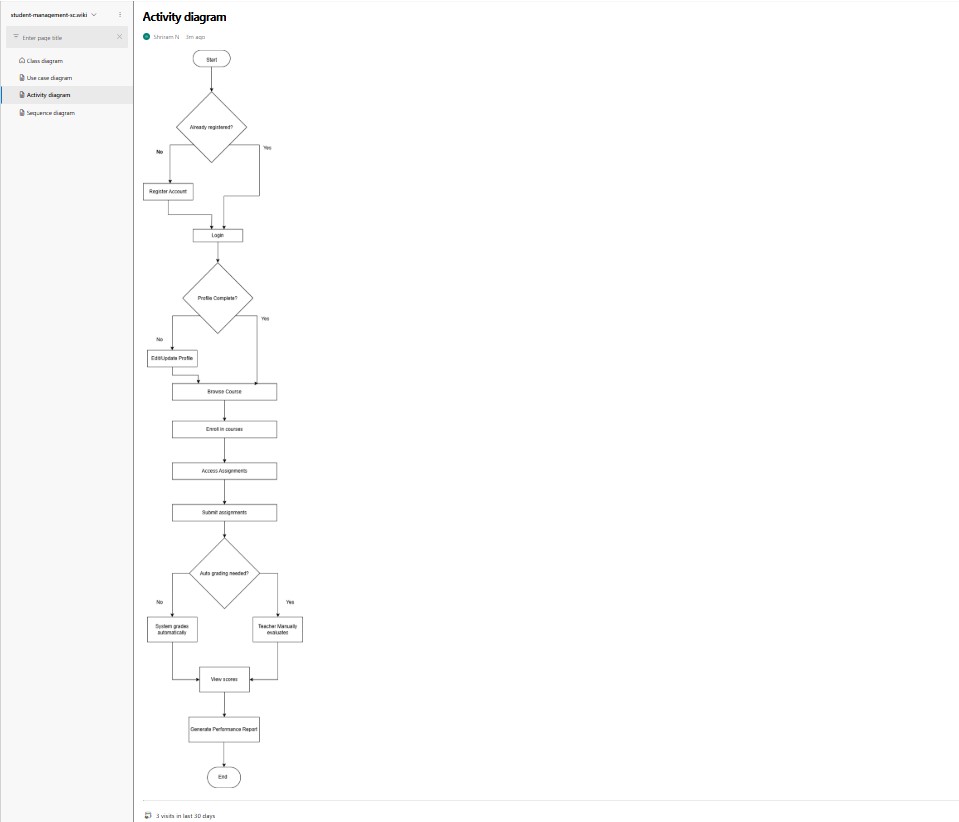
The Class Diagram and Sequence Diagram is designed Successfully for given project.

|  |  |
| --- | --- |
| **EXP NO: 7 DATE:** | **DESIGNING USE CASE DIAGRAMS AND ACTIVITY DIAGRAM FOR PROJECT ARCHITECTURE** |

**AIM:-**

To Design an activity Diagram and use case Diagram for the given Project.

# 7A. Activity Diagram:-



**7B. USE CASE DIAGRAM:-**



# RESULT:-

Thus activity and use case diagram has been designed successfully for Student Management System

|  |  |
| --- | --- |
| **EXP NO: 8 DATE:** | **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

* User Signup & Login
* Student Enrollment in Courses
* Assignment Submission & Evaluation
* Viewing Grades and Attendance
* Admin Role Management and Report Generation

## 2. Define User Interactions

 Each test case simulates a real user behavior (e.g., logging in, submitting an assignment, viewing results).

## 3. Design Happy Path Test Cases

* Focused on validating that all core functionalities work correctly under normal conditions.
* **Example**: Student registers and logs in, enrolls in a course, submits assignment, and views grades.

## 4. Design Error Path Test Cases

* Simulate invalid inputs, system issues, or failed actions to ensure proper error handling.
* **Example**: Login with wrong credentials, submission without attachment, unauthorized access to admin panel.

## 5. Break Down Steps and Expected Results

* Each test case includes a clear sequence of actions and expected results.
* Ensures both manual testers and automation tools can follow the process easily.

## 6. Use Clear Naming and IDs

* Test cases are uniquely identified (e.g., TC01 – Student Login Success, TC12 – Invalid Assignment Submission).
* Facilitates easy mapping to features and tracking in Azure DevOps.

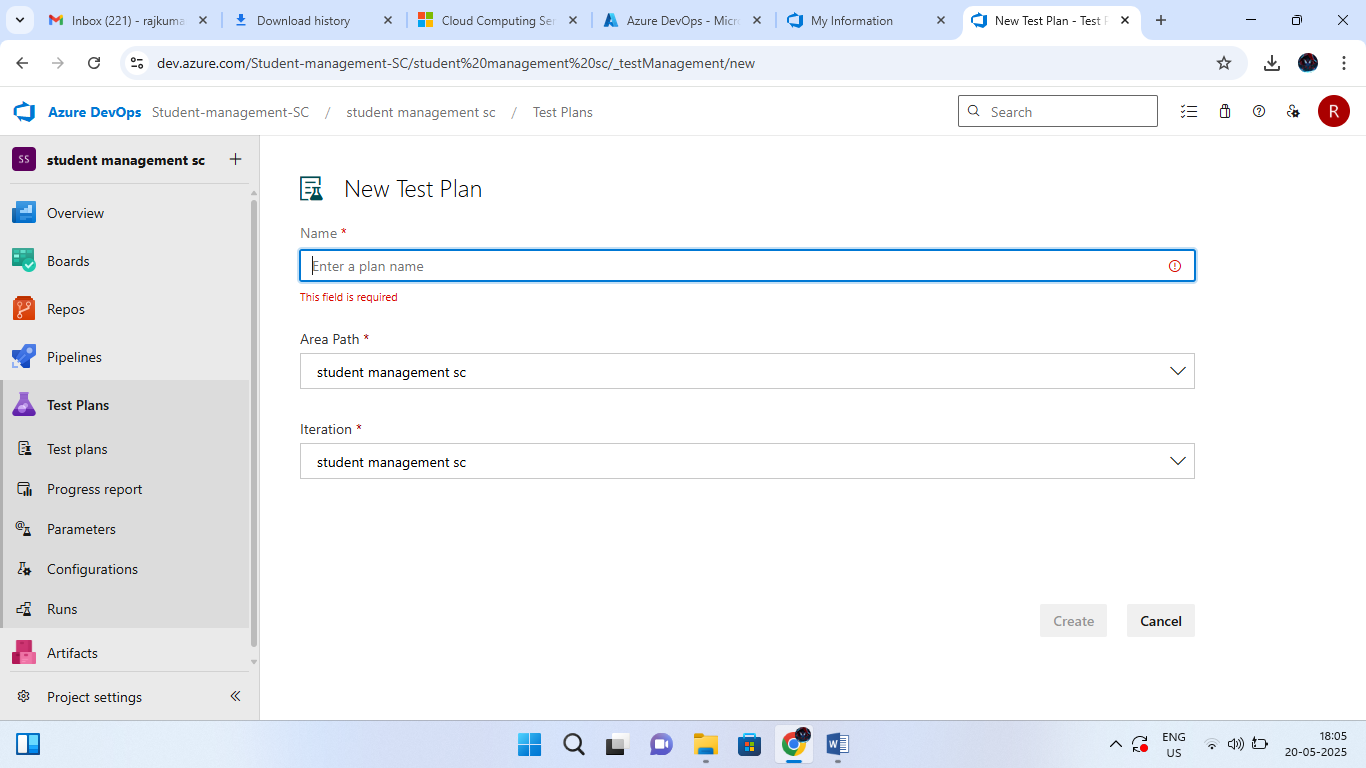
## 7. Separate Test Suites

* Grouped by functionality such as:
  + Login and Registration o Course Enrollment
  + Assignment Submission o Report Generation o Admin Functions
* Improves organization and enables focused execution in Azure DevOps.

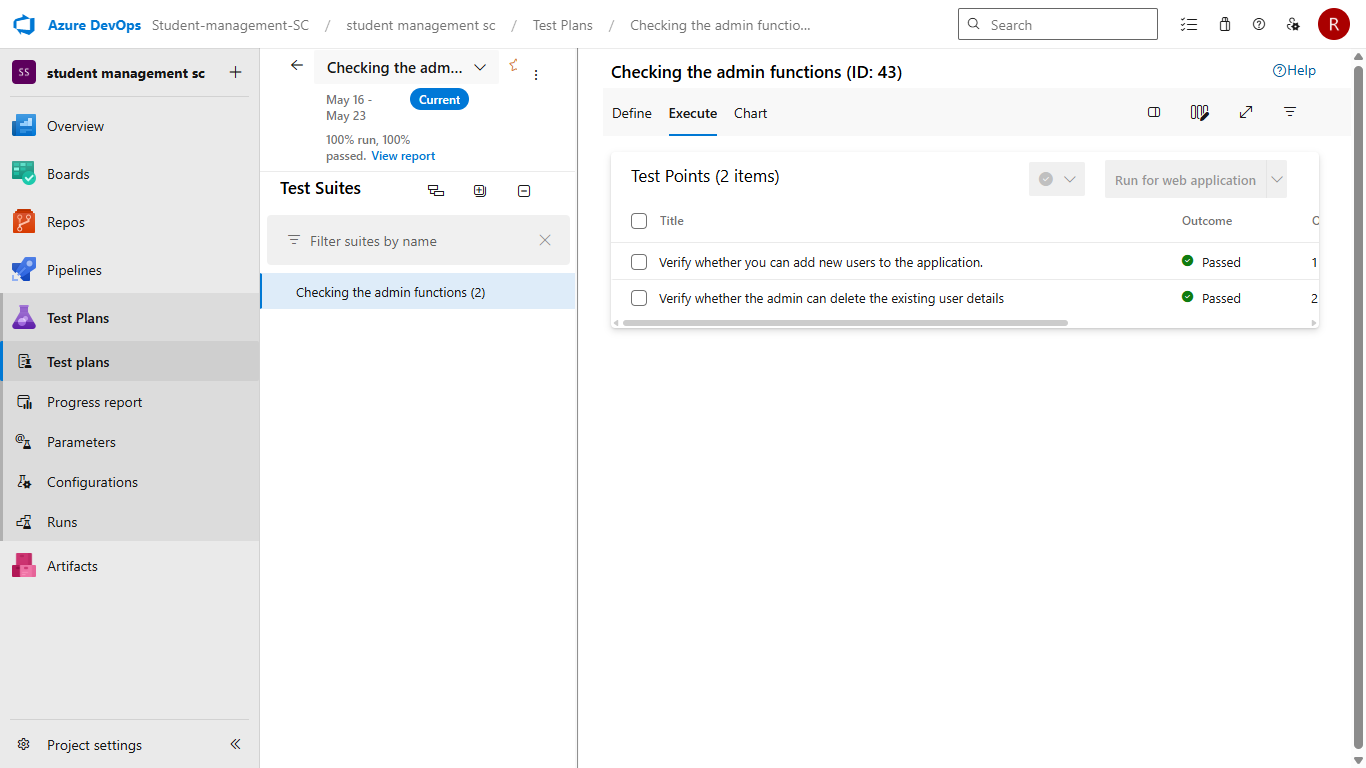
## 8. Prioritize and Review

* High-priority assigned to critical workflows like login, course access, and grading.
* Reviewed for completeness, accuracy, and alignment with user stories and feature definitions.

## 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.

Student Management System– Test Plans

### USER STORIES:-

* As a student, I want to sign up and log in securely so I can access academic services. *(ID: 101)*
* As a student, I need to enroll in courses to participate in classes. *(ID: 102)*
* As a student, I should be able to submit assignments online for evaluation. *(ID: 103)*
* As a student, I want to view grades and attendance in one place. *(ID: 104)*
* As an admin, I want to manage users and generate performance reports. *(ID: 105)* **Test Suites:-**

**Test suit : TS01 – User Login & Registration (ID: 201)**

1. **TC01 – Successful Student Sign-Up**  **Action:**

o Go to the Sign-Up page o Enter valid name, email, and password o Click "Sign Up"

* + **Expected Result:** o Student account is created and redirected to dashboard
  + **Type:** Happy Path

1. **TC02 – Login with Valid Credentials**  **Action:** o Go to Login page

o Enter valid student email & password o Click "Login"

 **Expected Result:**

o Logged in and taken to student dashboard

 **Type:** Happy Path

1. **TC03 – Sign-Up with Existing Email**  **Action:**

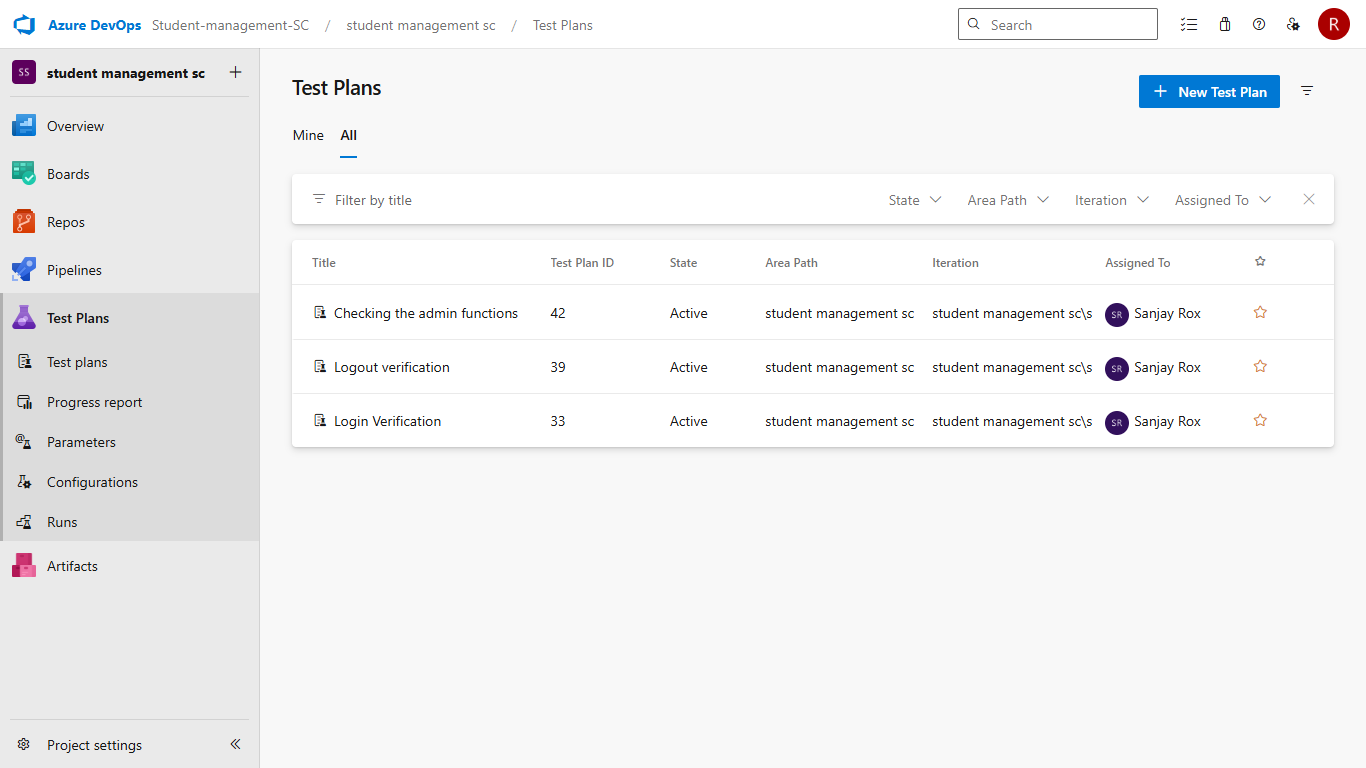
o Go to Sign-Up o Enter already registered email o Click "Sign Up"

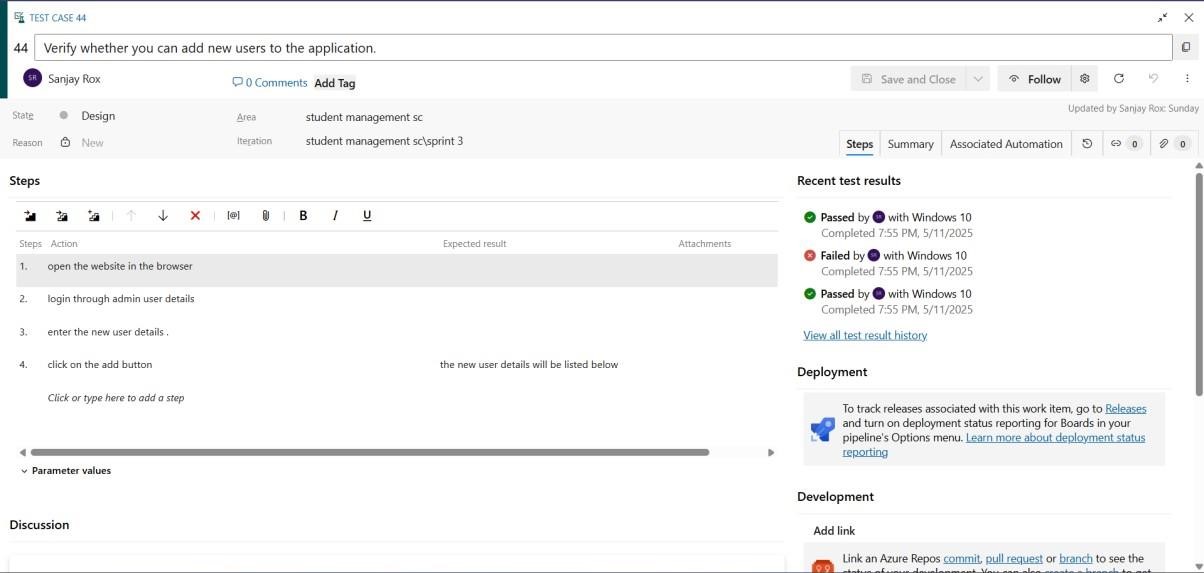
* + **Expected Result:** o Error message: "Email already registered"
  + **Type:** Error Path

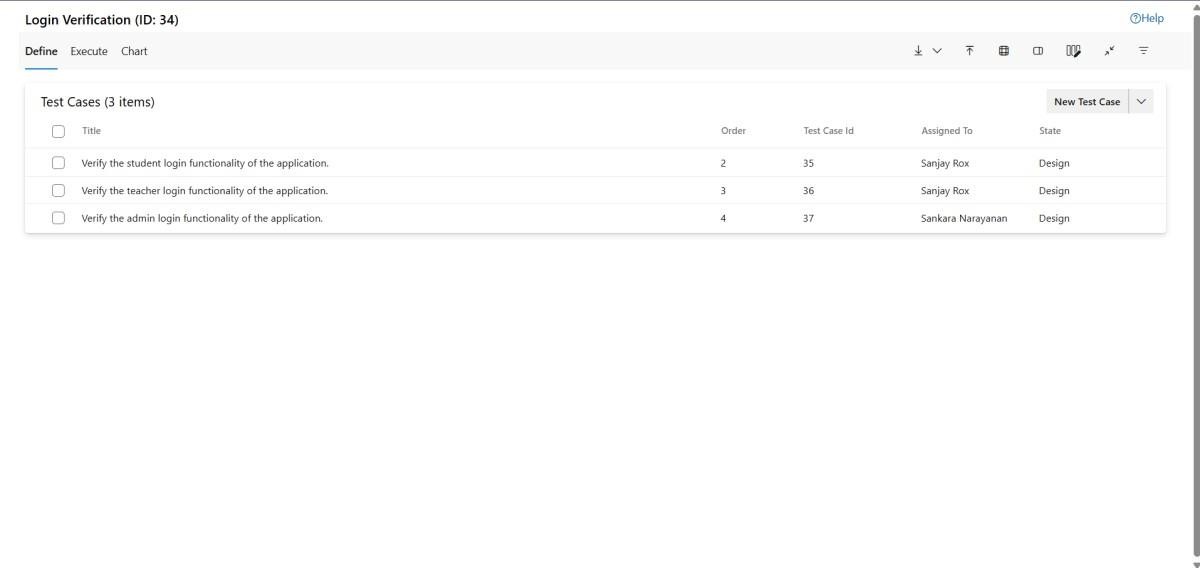
|  |  |
| --- | --- |
| 4. **TC04 – Login with Incorrect Password**  **Action:**  o Enter wrong password o Click "Login"   * **Expected Result:** o Error: "Invalid username or password" * **Type:** Error Path   **Test Suite: TS02 – Course Enrollment (ID: 202)**   1. **TC05 – Enroll in Available Course**  **Action:**   o Login → Navigate to Course List o Select a course → Click "Enroll"   * + **Expected Result:** o Enrollment confirmation shown   + **Type:** Happy Path  1. **TC06 – Try Enrolling in a Full Course**  **Action:**   o Select a course with no seats available o Click "Enroll"  **Expected Result:** o Error: "Enrollment failed – course full   * + **Type:** Error Path   **Test Suite: TS03 – Assignment Submission (ID: 203)**   1. **TC07 – Submit Assignment Successfully**  **Action:**   o Login → Go to Assignments o Upload file and click "Submit"   * + **Expected Result:** o Submission confirmation message shown   + **Type:** Happy Path  1. **TC08 – Submit Without Uploading File**  **Action:**   o Open assignment submission o Click "Submit" without file   * + **Expected Result:** o Error: "Please upload a file before submitting"   + **Type:** Error Path | CS23432 |

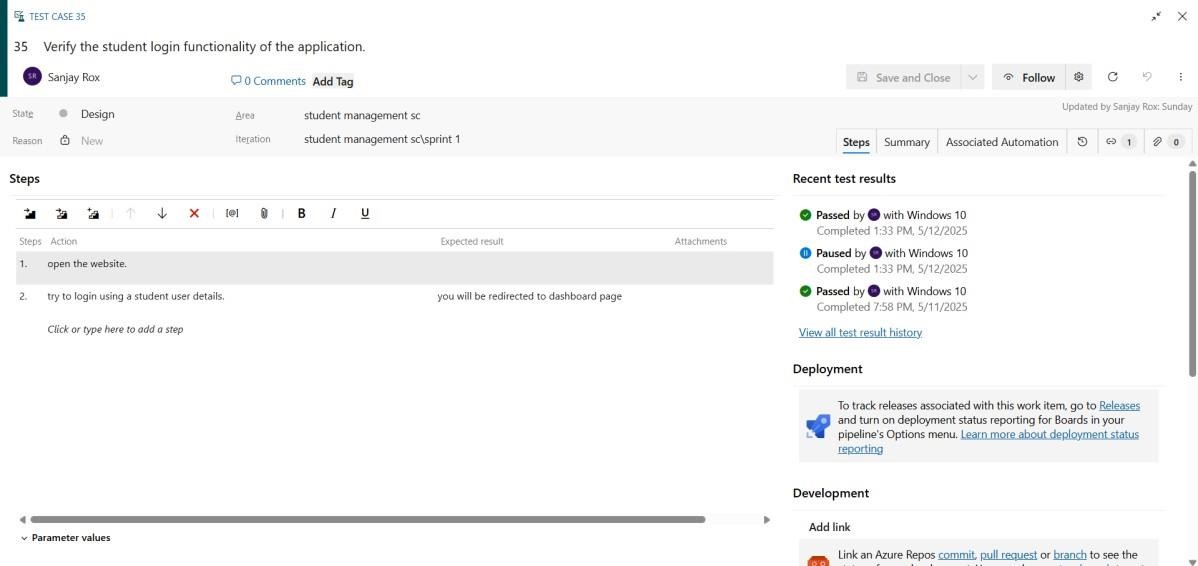
|  |  |
| --- | --- |
| **Test Suite: TS04 – View Grades & Attendance (ID: 204)**   1. **TC09 – View Grades Dashboard**  **Action:**   o Login → Go to "My Grades"   * + **Expected Result:** o All grades are displayed by subject   + **Type:** Happy Path  1. **TC10 – Fail to Fetch Grades (Offline Scenario)**  **Action:**   o Disconnect network → Go to "My Grades"   * + **Expected Result:** o Error: "Unable to fetch data – check connection"   + **Type:** Error Path   **Test Suite: TS05 – Admin Management (ID: 205)**   1. **TC11 – Assign Role to New User**  **Action:**   o Login as Admin → Navigate to User Management o Select user and assign role   * + **Expected Result:** o Confirmation: "Role assigned successfully"   + **Type:** Happy Path  1. **TC12 – Generate Report with No Data**  **Action:**   o Navigate to Reports → Choose empty semester o Click "Generate"   * + **Expected Result:** o Message: "No data available to generate report"   + **Type:** Error Path   2116231501154 | CS23432 |

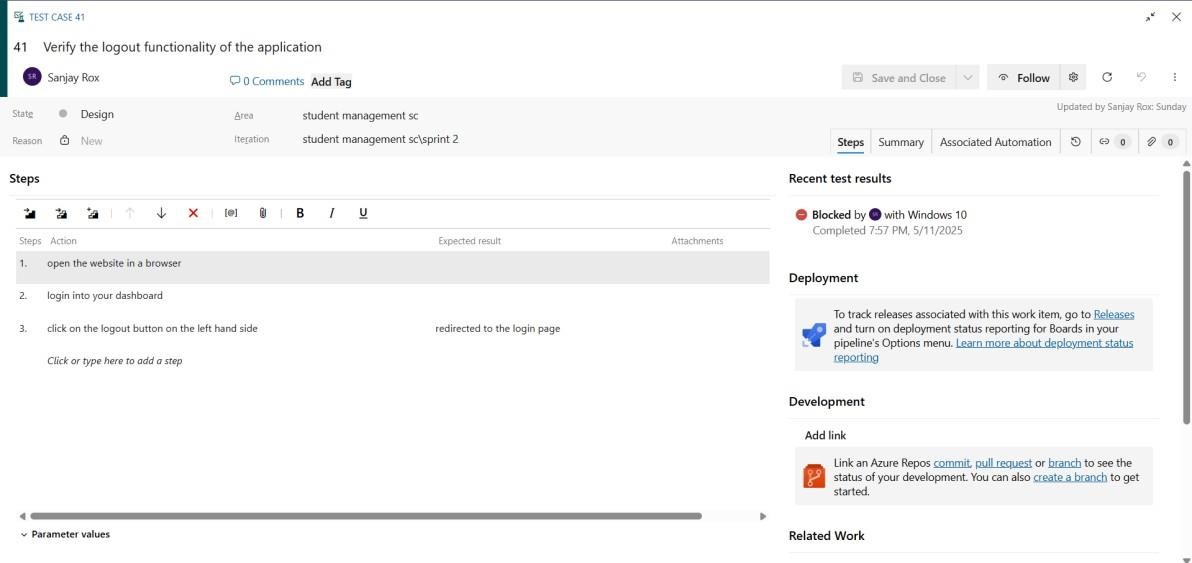
**3. Test Cases**

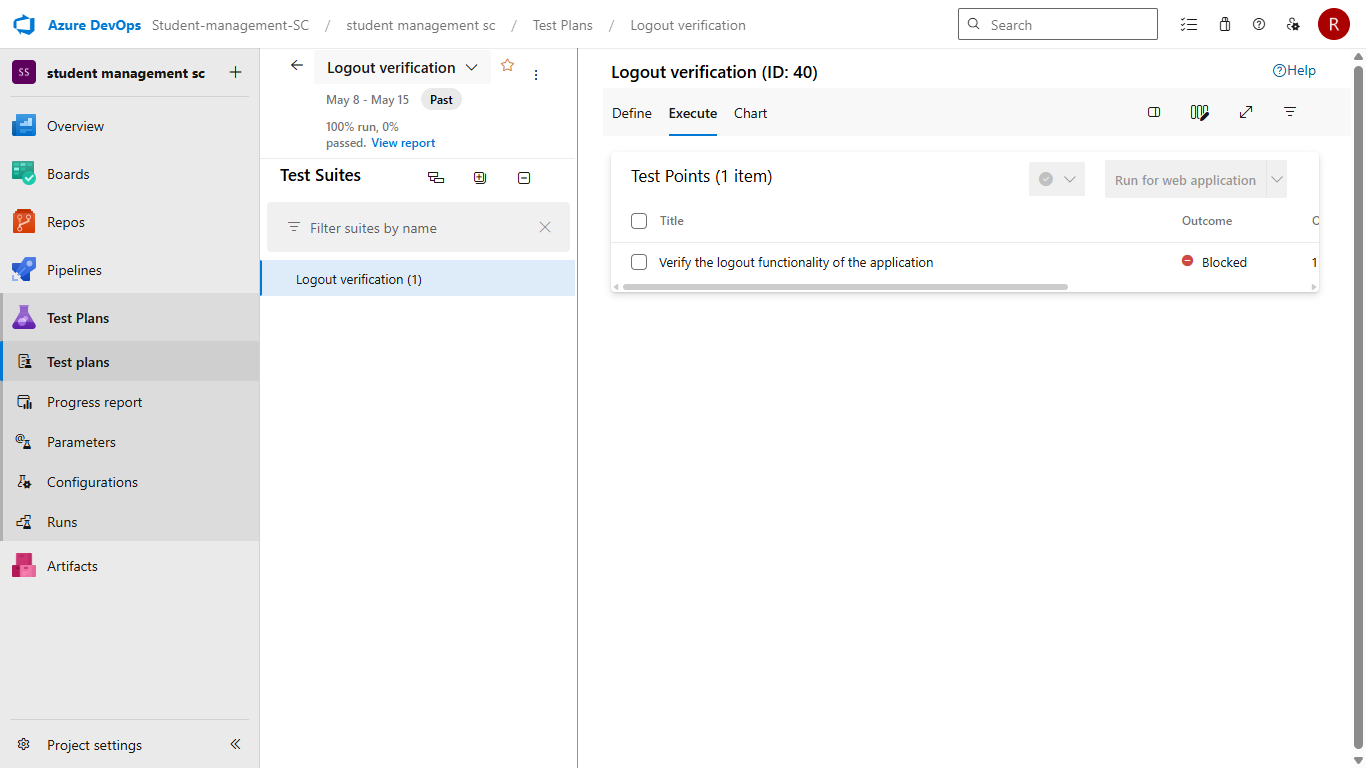




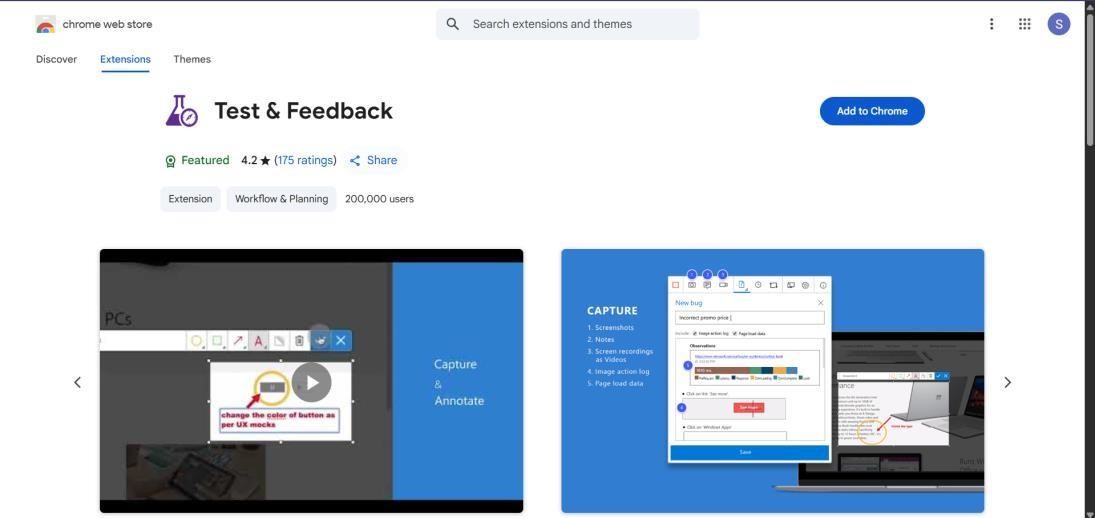


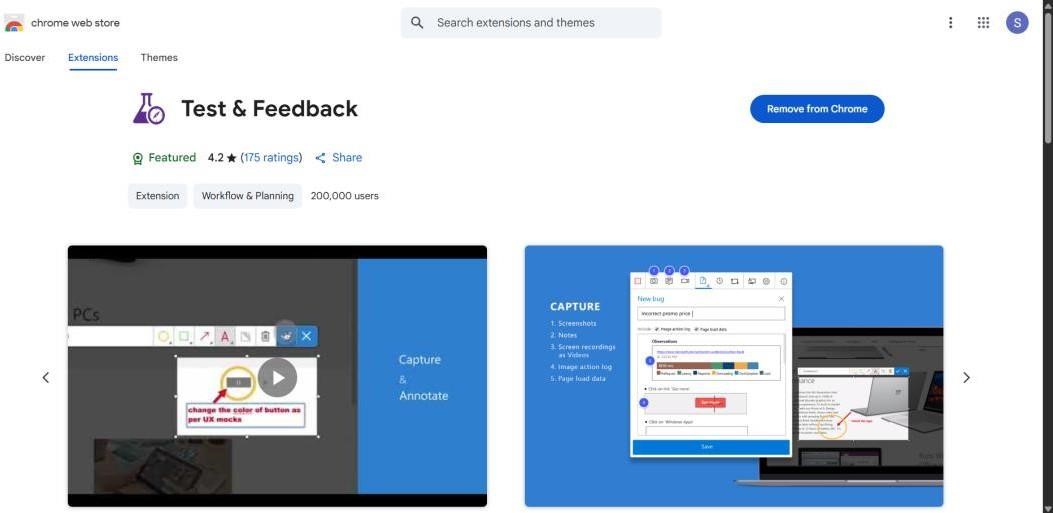




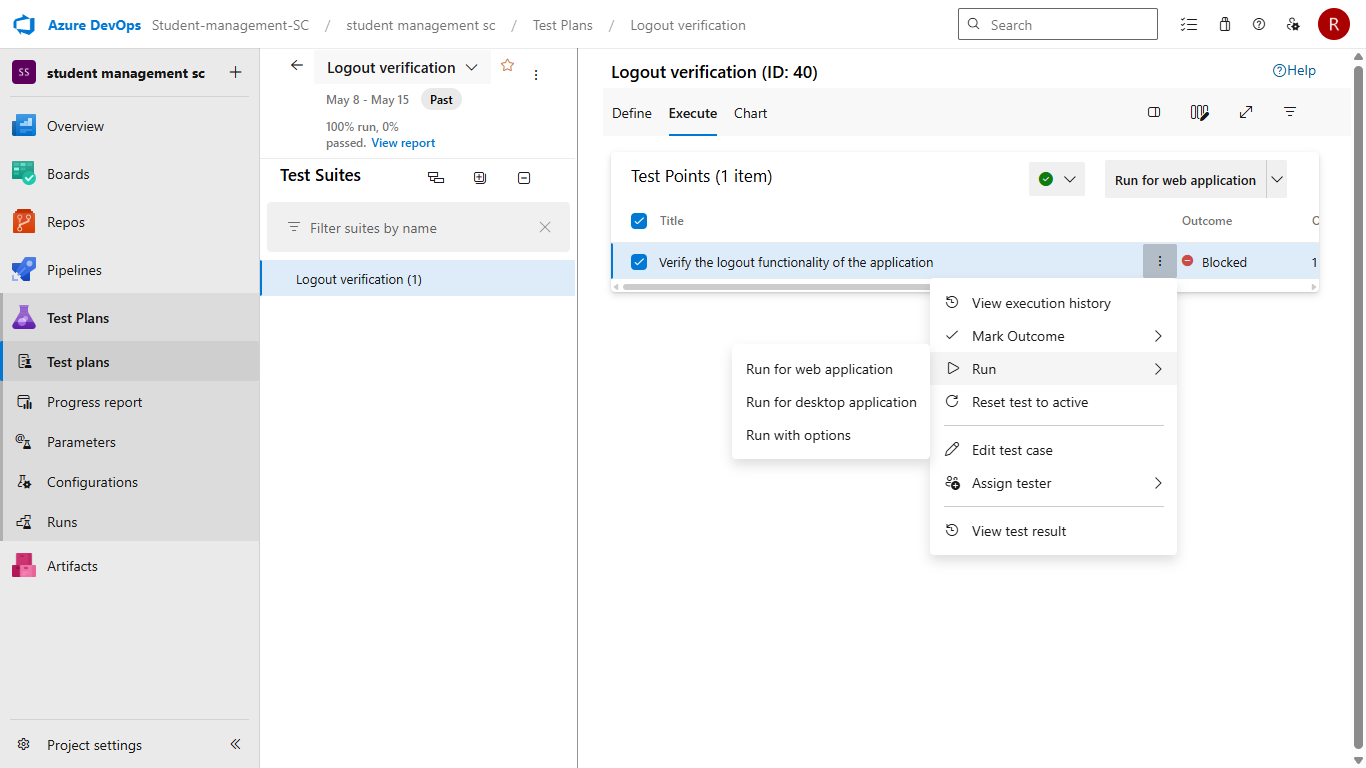


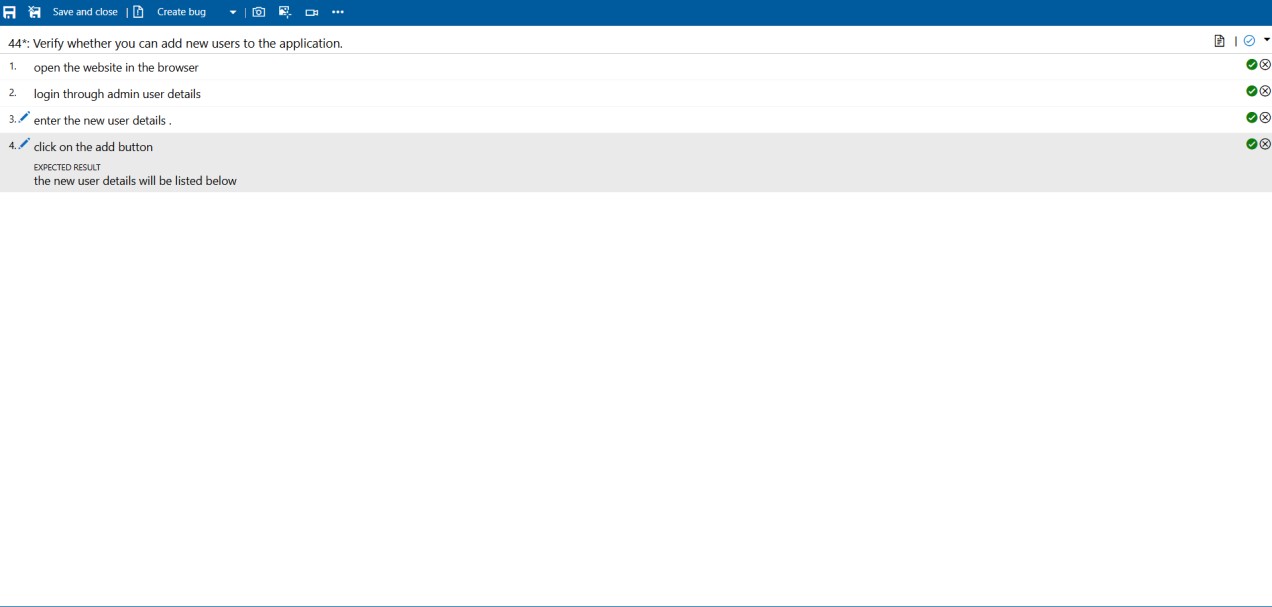
**Installation of test**



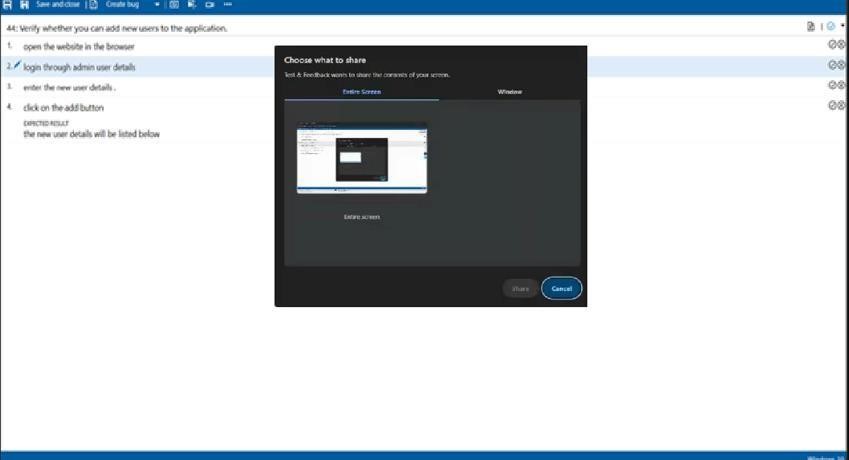


**Running the test cases**

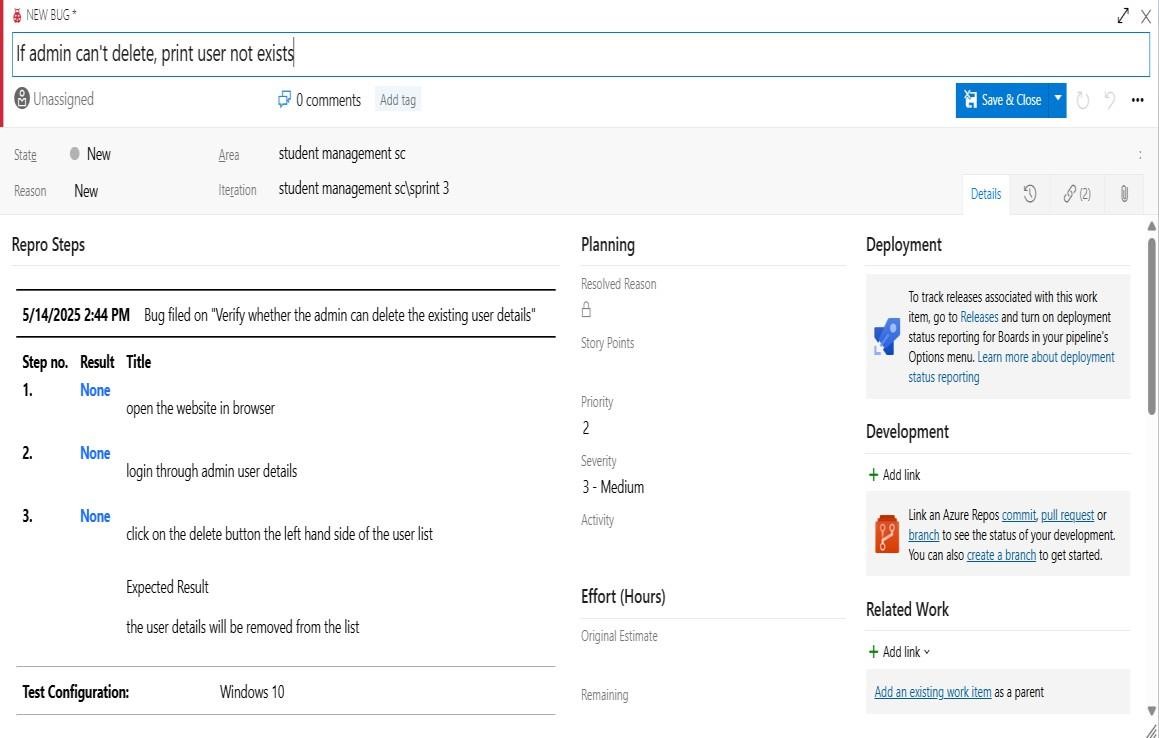


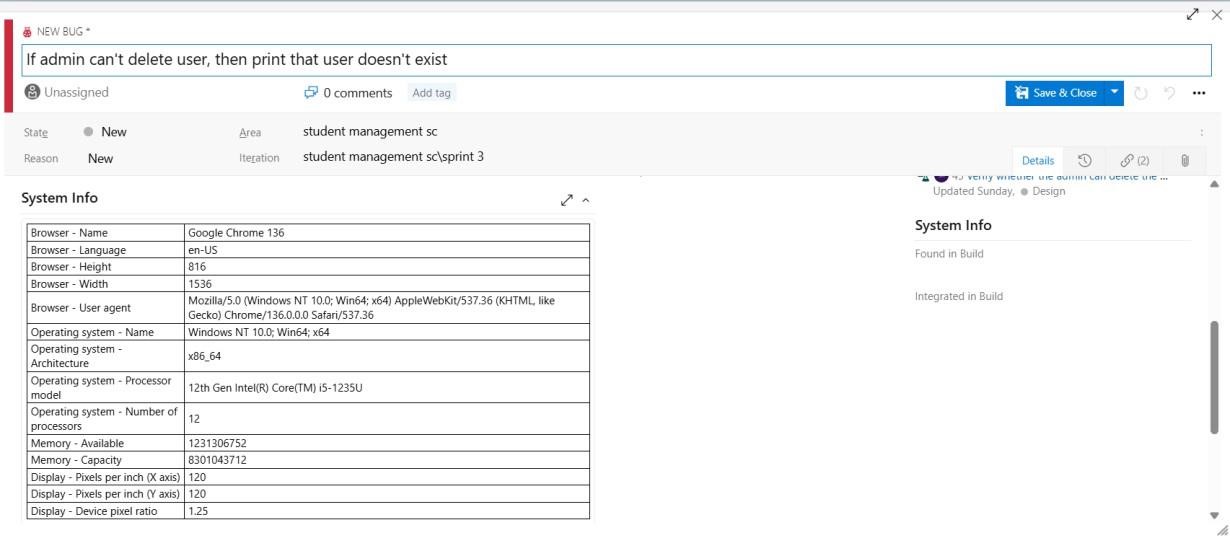


**Recording the test case**

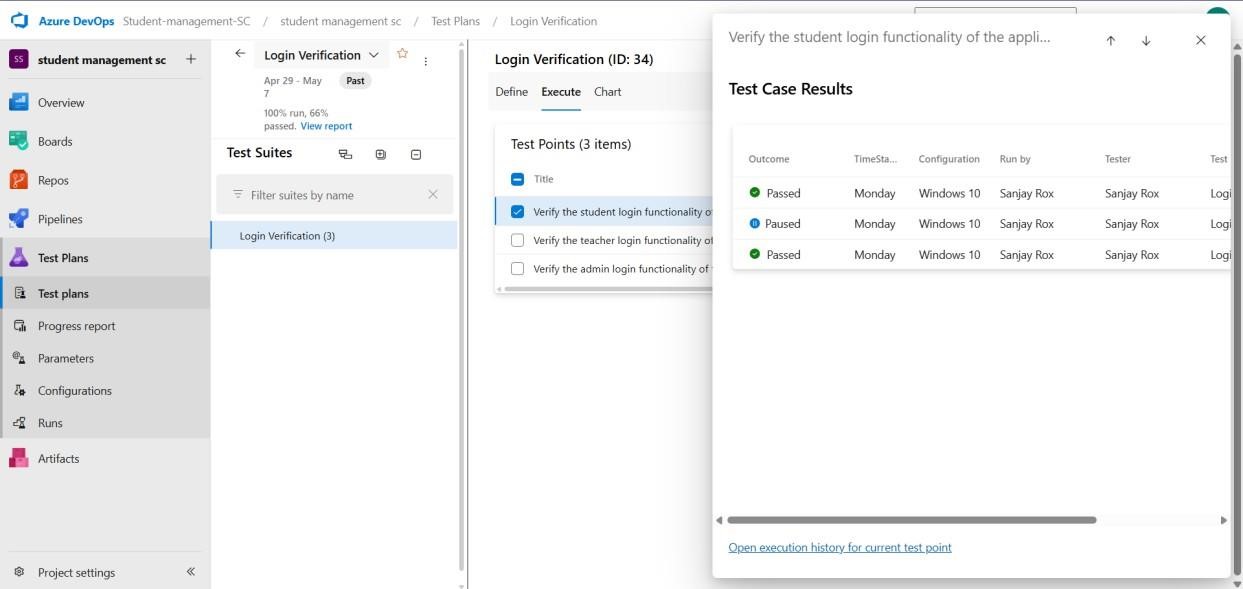


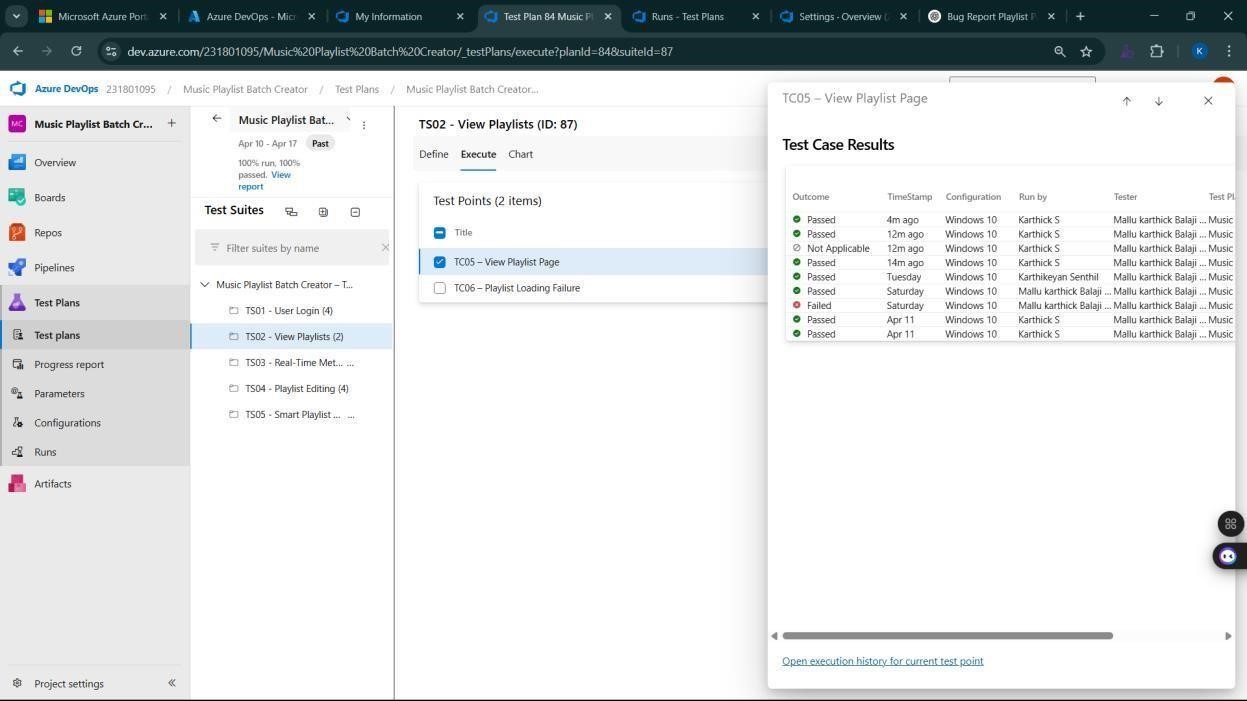
* 1. **Creating the bug**



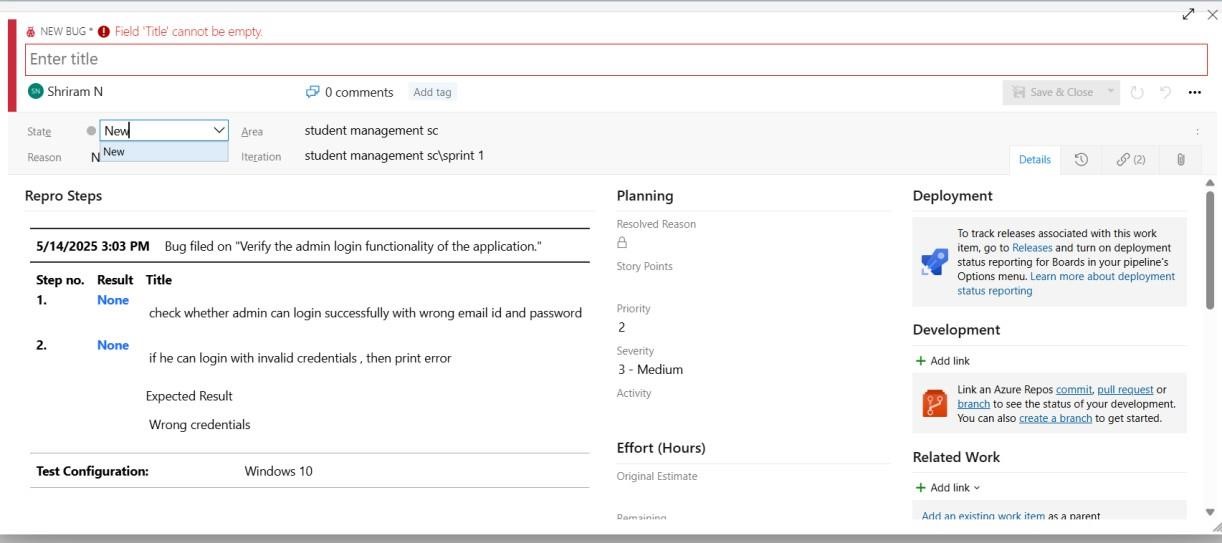


**Test Results**



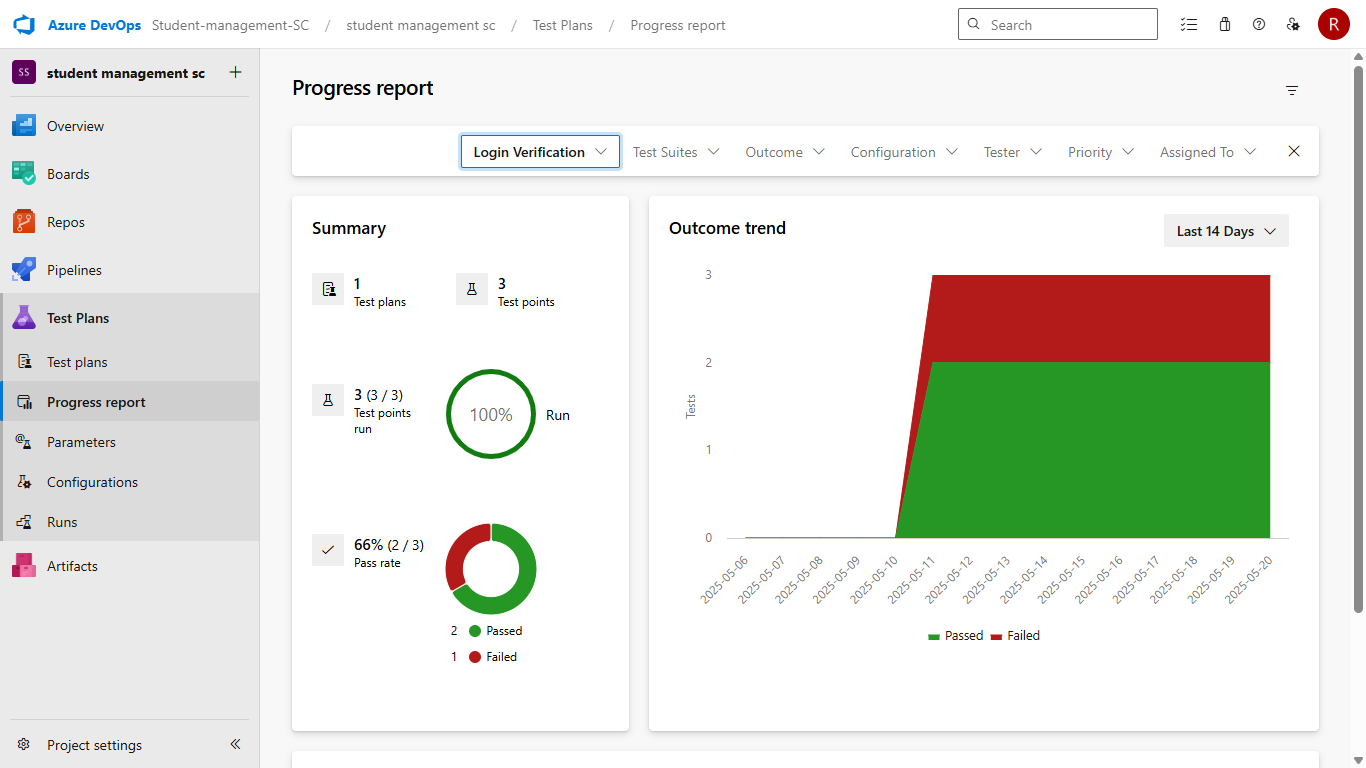


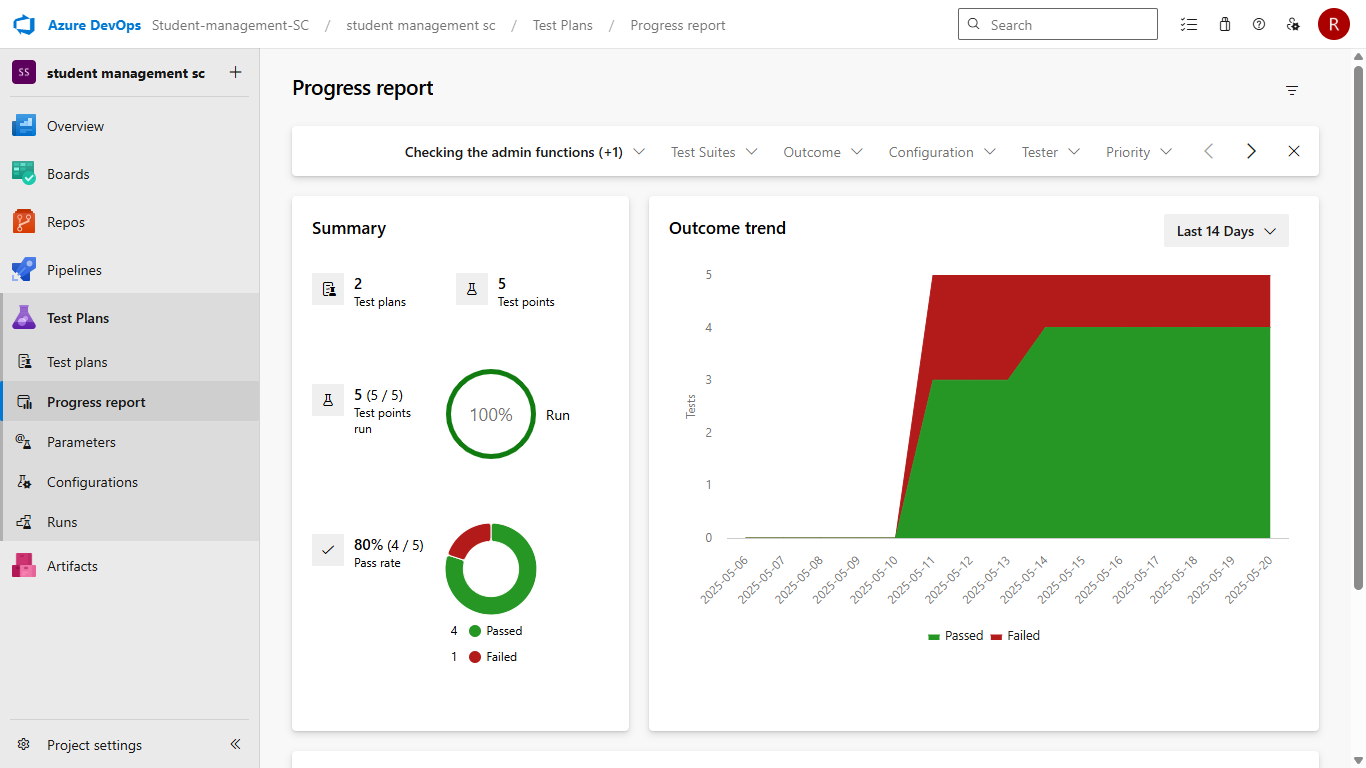
* 1. **Test report summary**

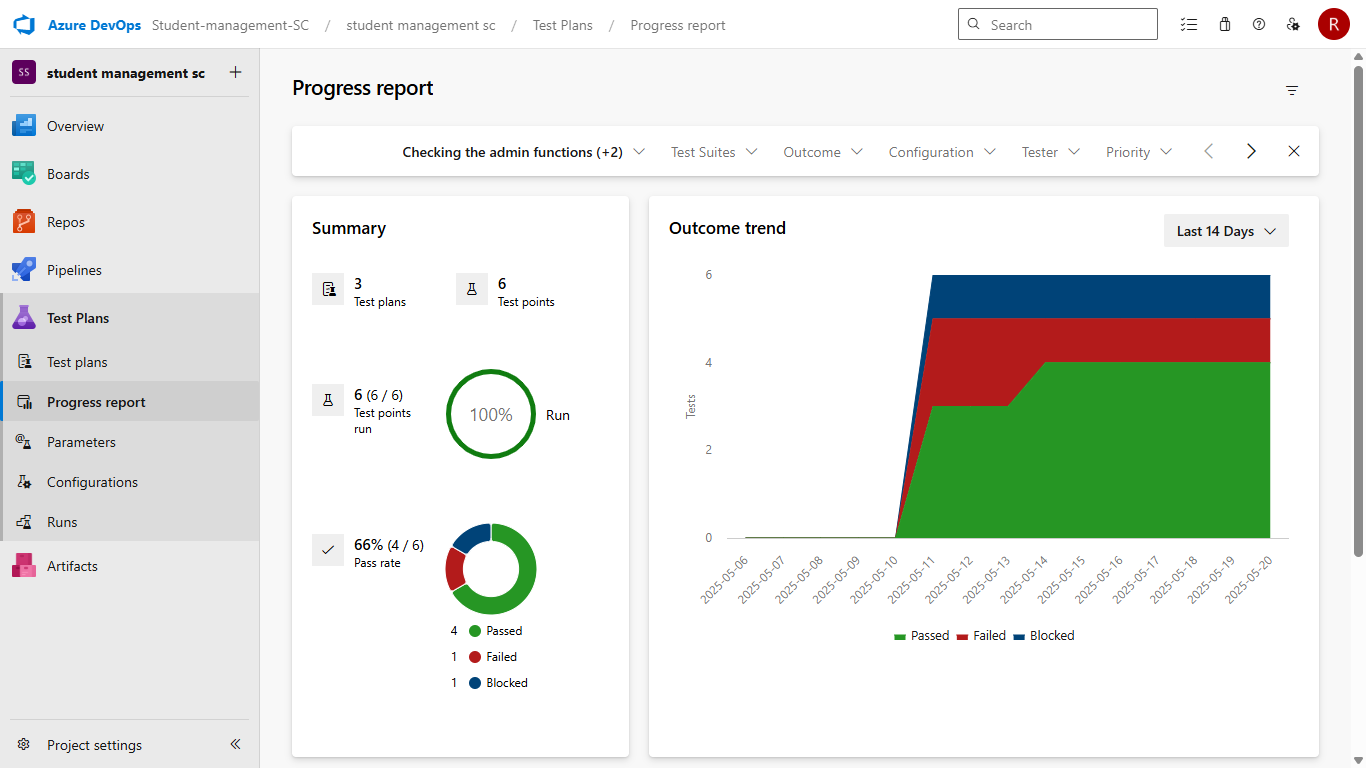


 Assigning bug to the developer and changing state

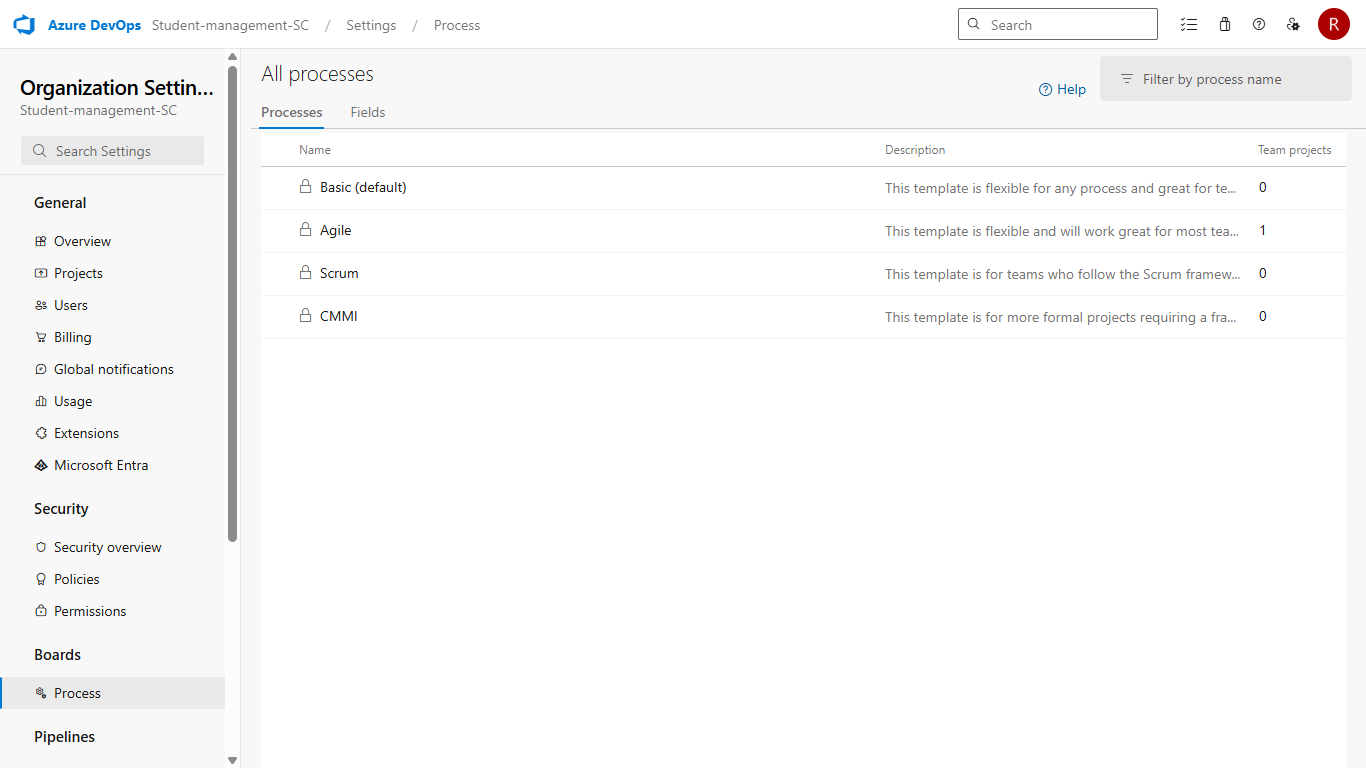
* 1. **Progress report**







* 1. **Changing the test template**



**Result:**

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

Error Path

|  |  |
| --- | --- |
| **EXP NO: 9 DATE:** | **Ci/CD PIPELINES IN AZURE** |

**AIM:**

To implement a Continuous Integration and Continuous Deployment (CI/CD) pipeline in Azure DevOps for automating the build, testing, and deployment process of the Student Management System, ensuring faster delivery and improved software quality.

**PROCEDURE:**

**Steps to Create and implement pipelines in Azure:**

* + 1. Sign in to Azure DevOps and Navigate to Your Project

Log in to [dev.azure.com,](https://dev.azure.com/) select your organization, and open the project where your Student Management System code resides.

* + 1. Connect a Code Repository (Azure Repos or GitHub)

Ensure your application code is stored in a Git-based repository such as Azure Repos or GitHub. This will be the source for triggering builds and deployments in your pipeline.

* + 1. Create a New Pipeline

Go to the Pipelines section on the left panel and click “Create Pipeline”.

Choose your source (e.g., Azure Repos Git or GitHub), and then select the repository containing your project code.

* + 1. Choose the Pipeline Configuration

You can select either the YAML-based pipeline (recommended for version control and automation) or the Classic Editor for a GUI-based setup.

If using YAML, Azure DevOps will suggest a template or allow you to define your own.

* + 1. Define Build Stage (CI - Continuous Integration) from YAML file

* + 1. Install dependencies (e.g., npm install, dotnet restore)

* + 1. Build the application (dotnet build, npm run build)

* + 1. Run unit tests (dotnet test, npm test)

* + 1. Publish build artifacts to be used in the release stage
    2. Save and Run the Pipeline for the First Time Save the YAML or build definition and click “Run”.

Azure will fetch the latest code and execute the defined build and test stages.

* + 1. Configure Continuous Deployment (CD)

Navigate to the Releases tab under Pipelines and click “New Release Pipeline”. Add an Artifact (from the build stage) and create a new Stage (e.g., Development, Production).

* + 1. Configure the CD stage with deployment tasks such as deploying to Azure App Service, running database migrations or scripts, and restarting services using the Azure App Service Deploy task linked to your subscription and app details.

* + 1. Set Triggers and Approvals

Enable continuous deployment trigger so the release pipeline runs automatically after a successful build.

For production environments, configure pre-deployment approvals to ensure manual verification before release.

* + 1. Monitor Pipelines and Manage Logs

View all pipeline runs under the Runs section.

Check logs for build/test/deploy stages to debug any errors.

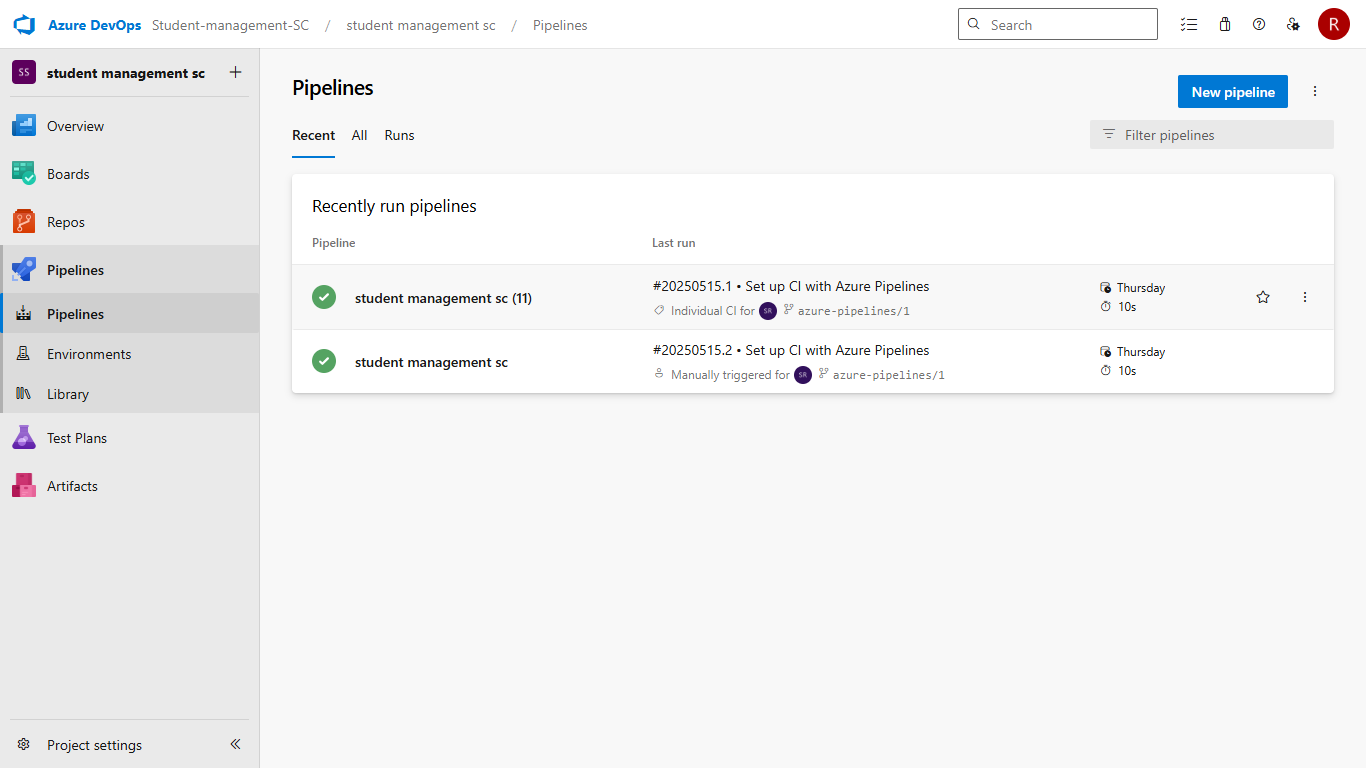
You can also integrate email alerts or Microsoft Teams notifications for build failures.

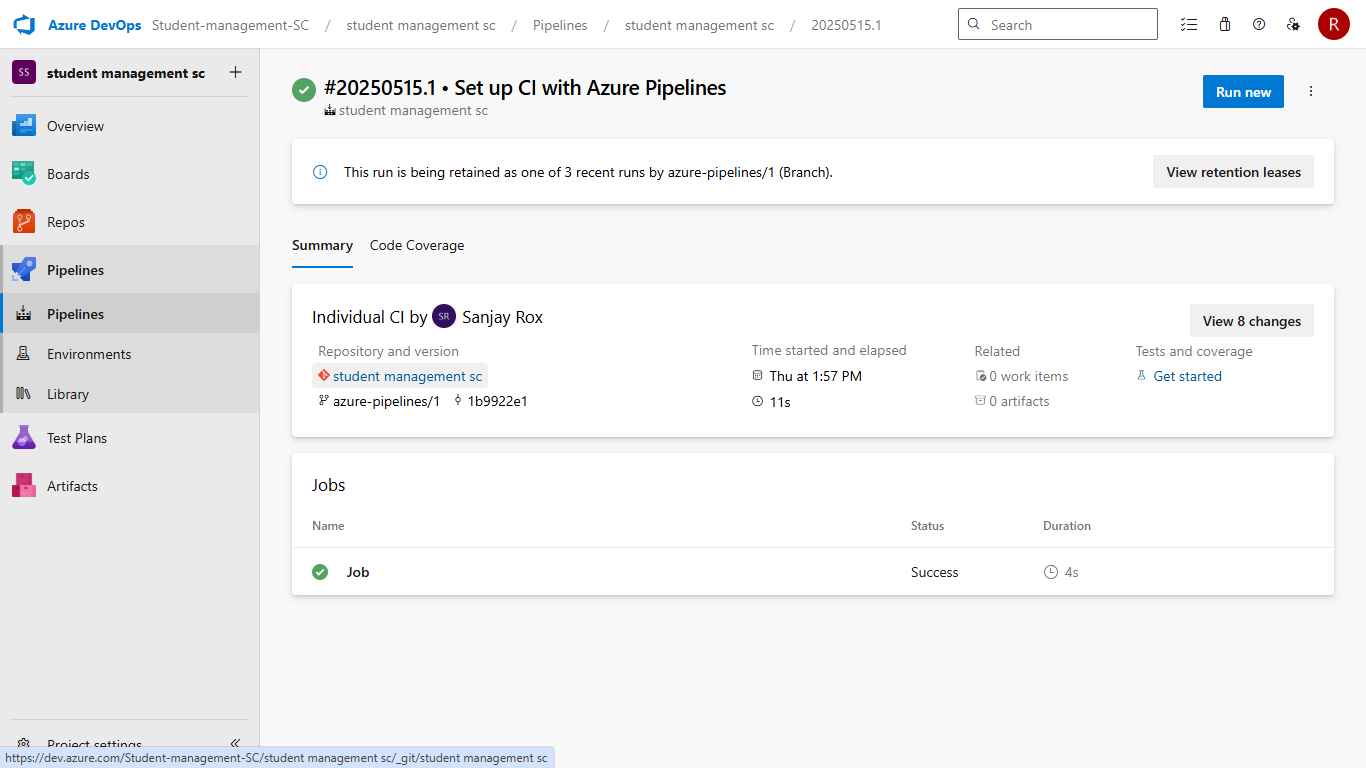
* + 1. Review and Maintain Pipelines

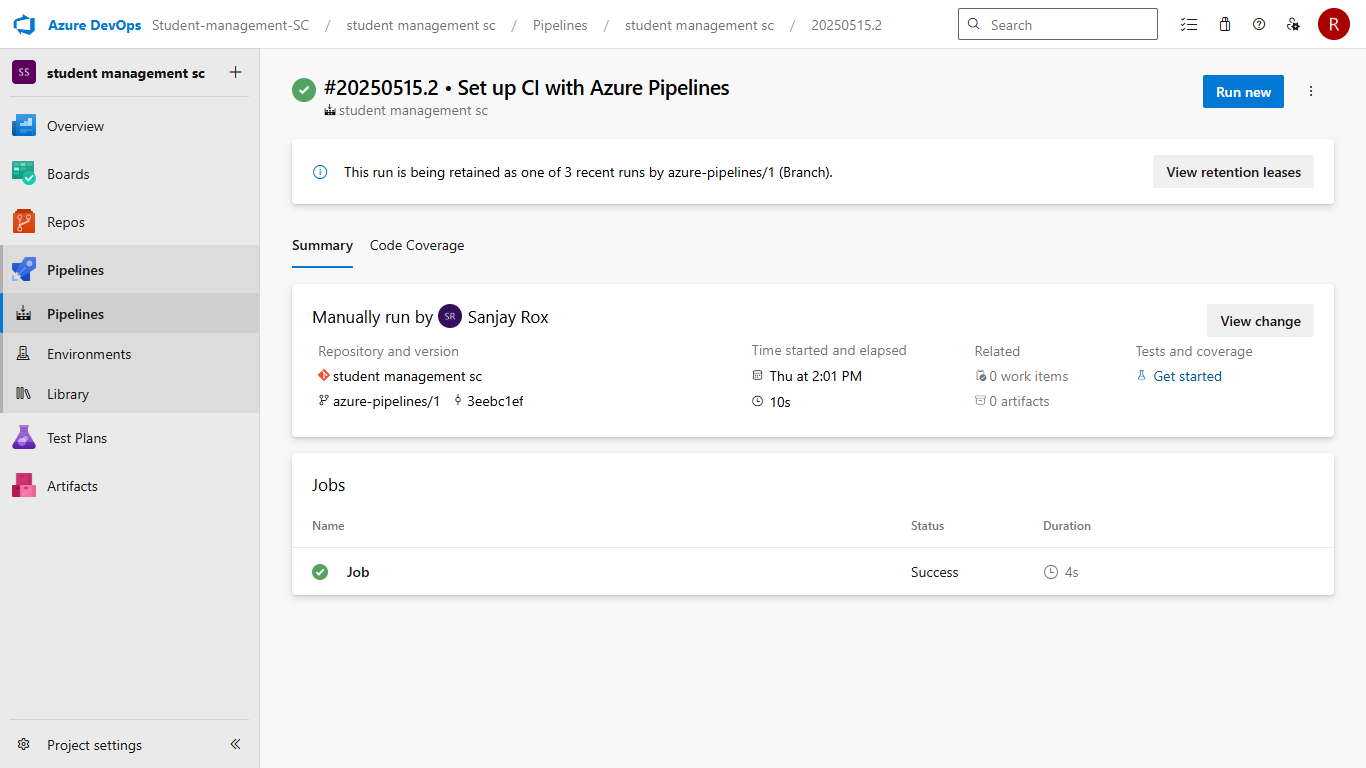
Regularly update your pipeline tasks or YAML configurations as your application grows.

Ensure pipeline runs are clean and artifacts are stored securely.

Integrate quality gates and code coverage policies to maintain code quality.







**RESULT:**

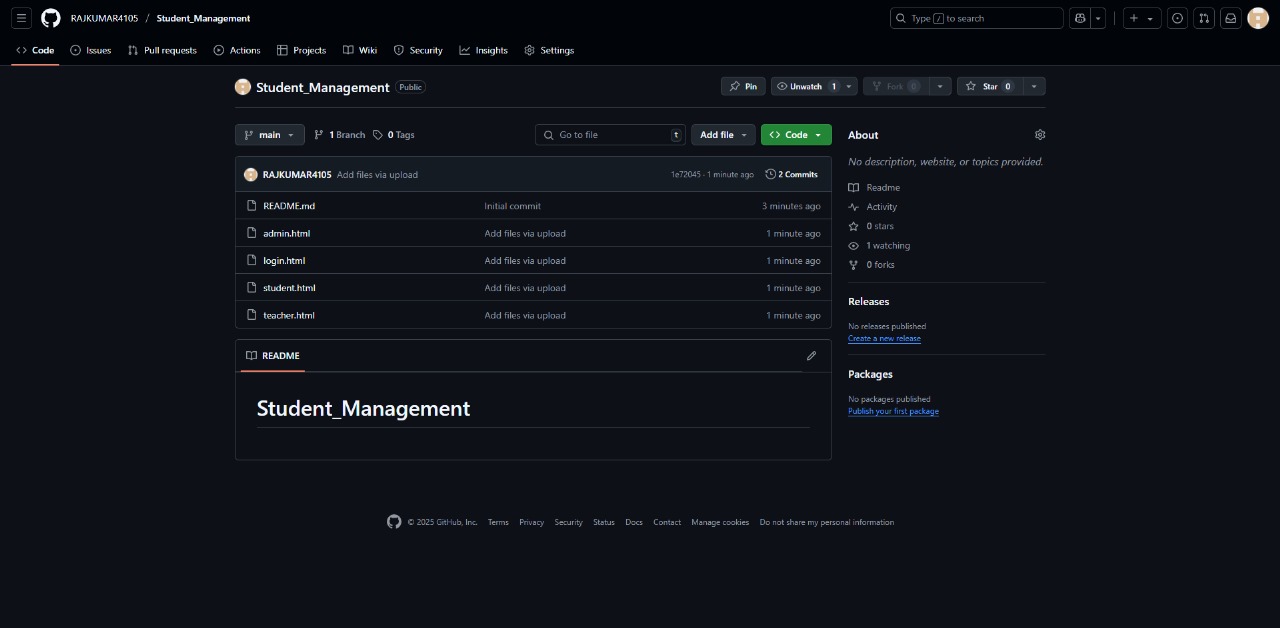
Thus the pipelines for the given project “Student Management System has been executed successfully

|  |  |
| --- | --- |
| **EXP NO: 10 DATE:** | **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 Student Management System 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.