|  |  |  |
| --- | --- | --- |
| S. No. | Date | Title |
| 1. | 21/1/25 | Azure DevOps Environment Setup. |
| 2. | 21/1/25 | Azure DevOps Project Setup and User Story Management. |
| 3. | 28/1/25 | Setting Up Epic, Features, And User Stories for Project Planning. |
| 4. | 11/2/25 | Sprint Planning. |
| 5. | 18/2/25 | Poker Estimation. |
| 6. | 25/2/25 | Designing Class and Sequence Diagram for Project Architecture. |
| 7. | 4/3/25 | Designing Activity and Use Case Diagram for Project Structure. |
| 8. | 25/3/25 | Testing – Test Plans and Test Cases. |
| 9. | 15/4/25 | Loading Testing and Pipelines. |
| 10. | 22/4/25 | GitHub: Project Structure and Naming Conventions. |

**Index**

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

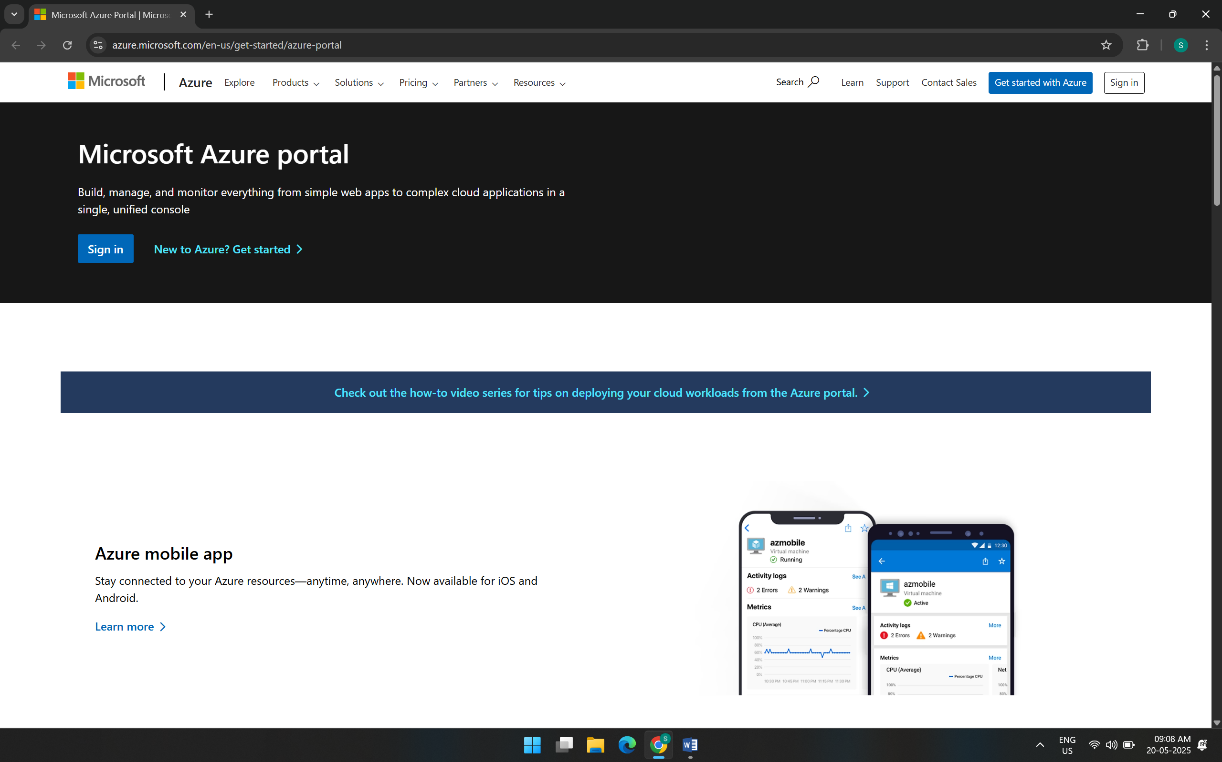
# AIM:-

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-](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>

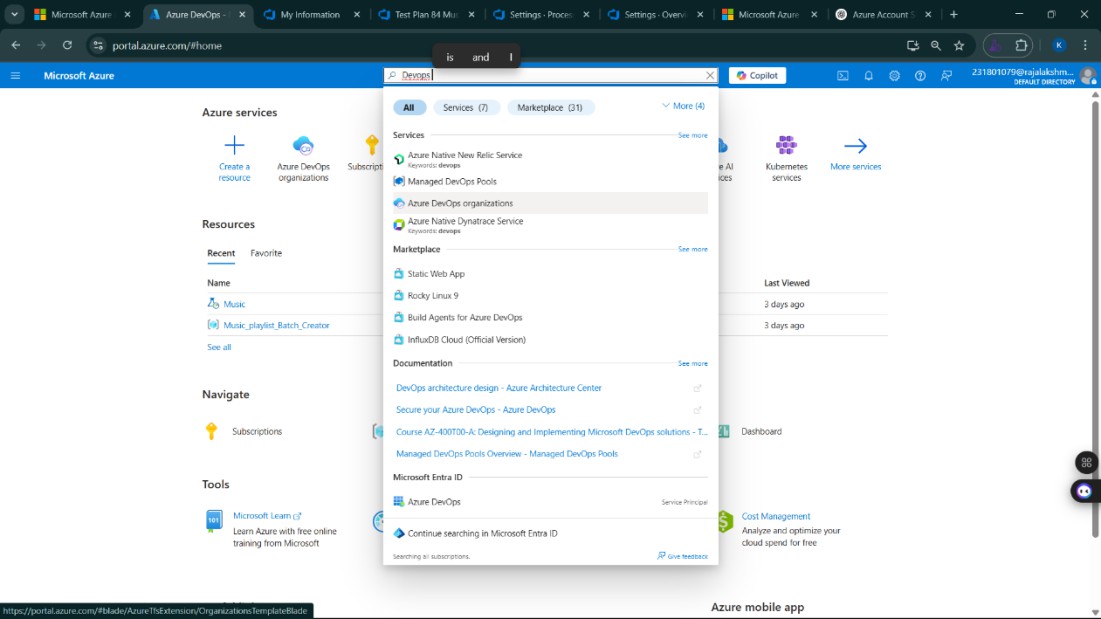


1. Azure home page

A screenshot of a computer

AI-generated content may be incorrect.

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.

A screenshot of a computer

AI-generated content may be incorrect.

# RESULT:

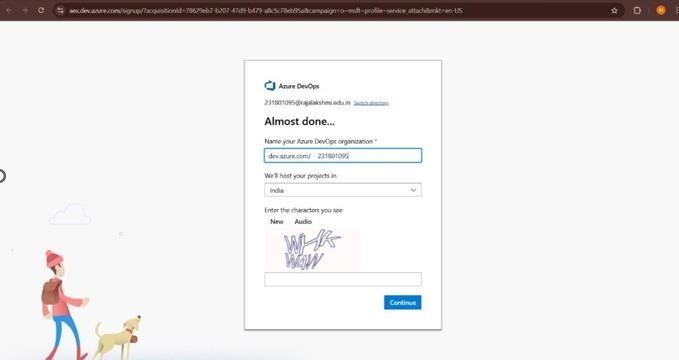
Successfully accessed the Azure DevOps environment and created a new organization through the Azure portal.

|  |  |
| --- | --- |
| **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

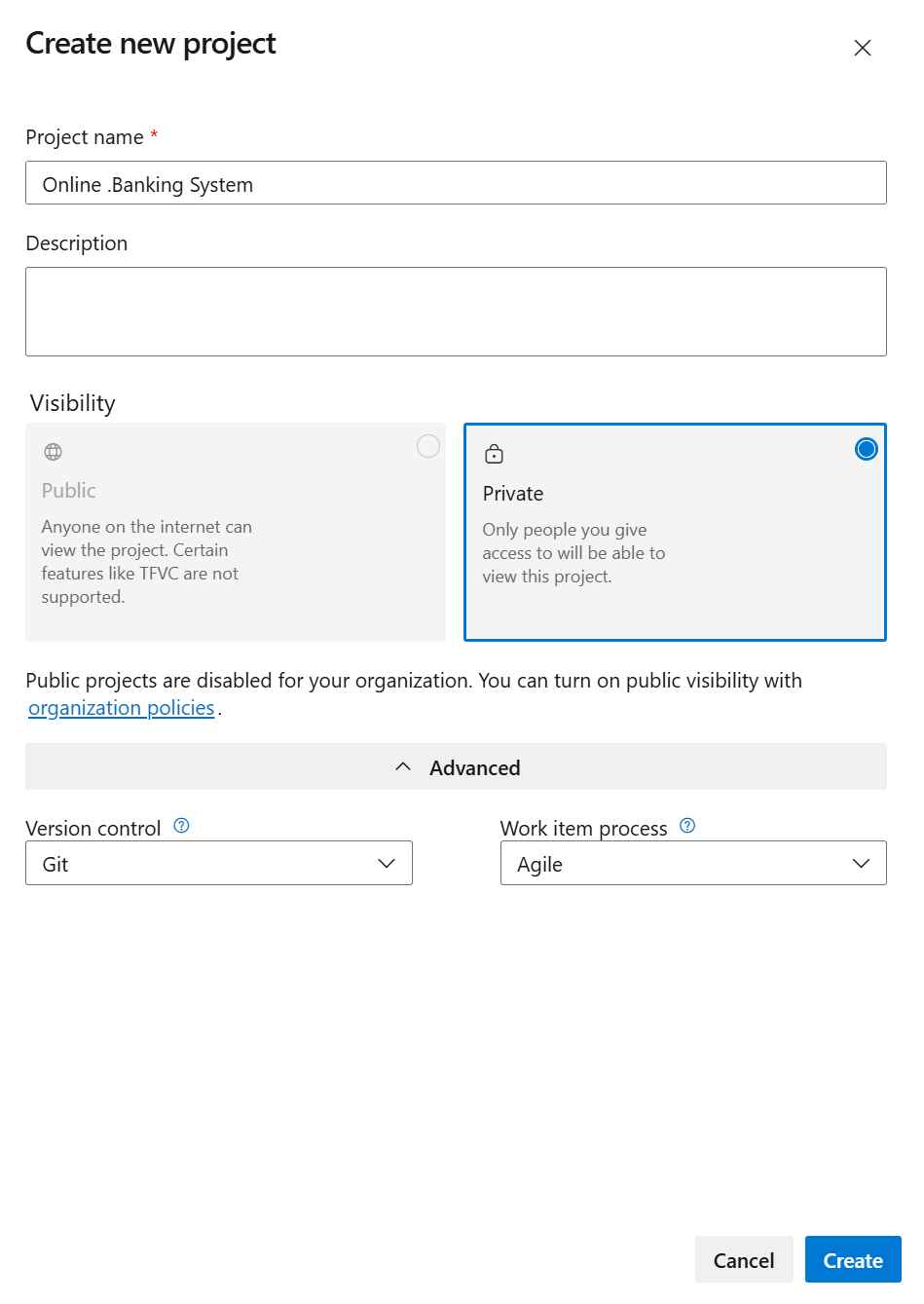
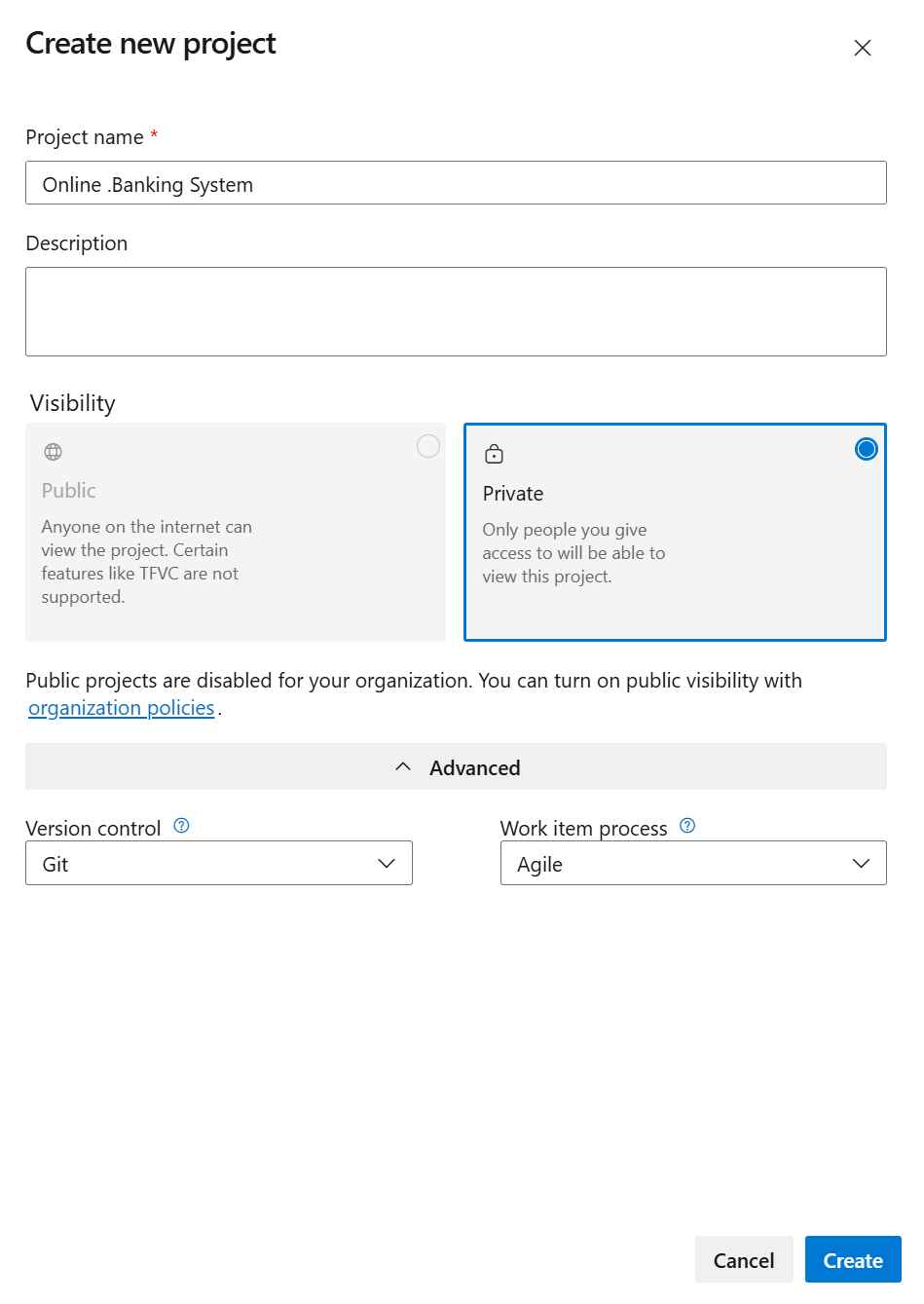


1. Create the First Project in Your Organization
   1. After the organization is set up, you’ll need to create your first **project**. This is where you'll begin to manage code, pipelines, work items, and more.
   2. On the organization’s **Home page**, click on the **New Project** button.
   3. Enter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

A screenshot of a computer

AI-generated content may be incorrect.

1. Project dashboard

A screenshot of a computer

AI-generated content may be incorrect.

1. To manage user stories:
2. 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.

A screenshot of a computer

AI-generated content may be incorrect.

**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

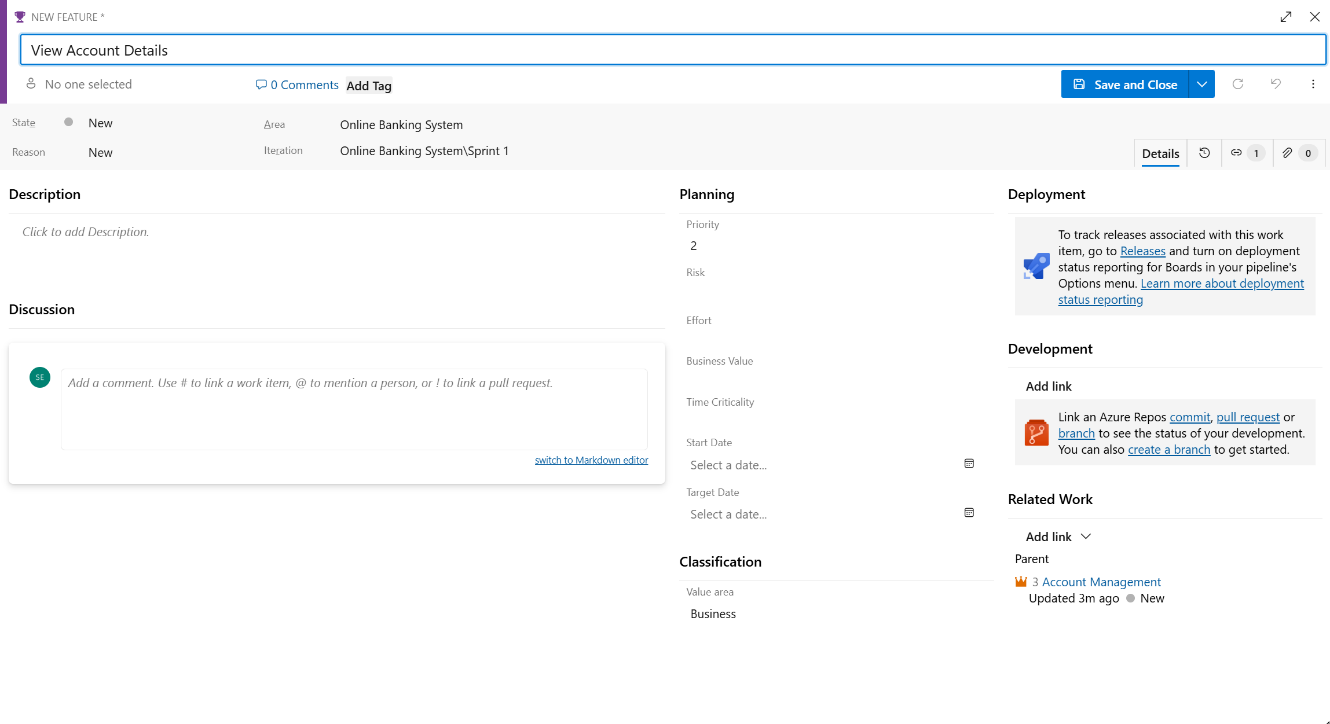
**A screenshot of a computer

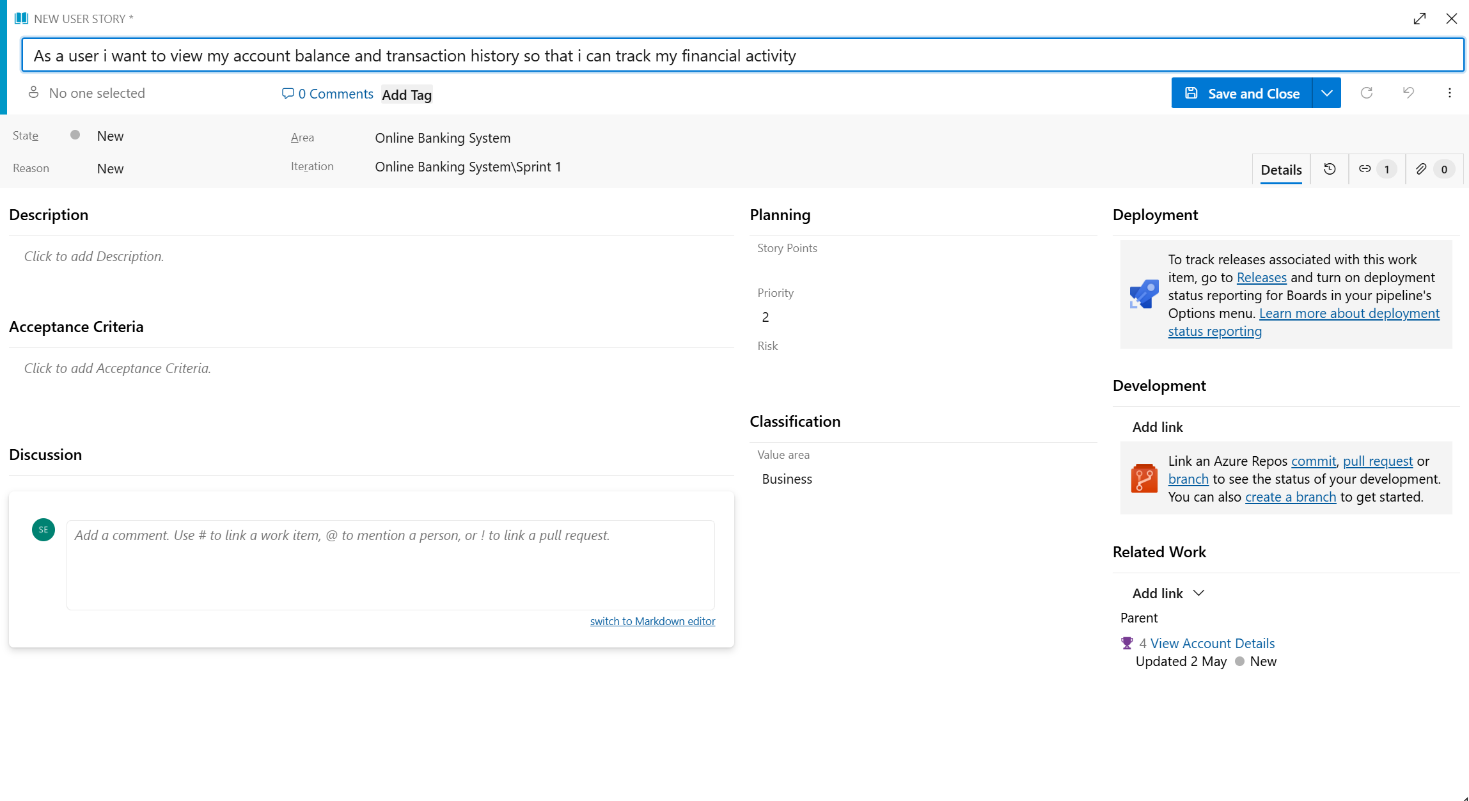
AI-generated content may be incorrect.**

1. **Fill in Epics**

**A screenshot of a computer

AI-generated content may be incorrect.**

1. **Fill in Features**
2. **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 Online Banking System Project.

## Sprint Planning:-

## Sprint 1:

**A screenshot of a computer

AI-generated content may be incorrect.**

**Sprint 2**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Sprint 3**

**A screenshot of a computer

AI-generated content may be incorrect.**

# RESULT:-

The Sprints are created for the Online Banking System.

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

# AIM:-

Create Poker Estimation for the user stories -Online Banking System.

**Poker Estimation:-**

**A screenshot of a computer

AI-generated content may be incorrect.**

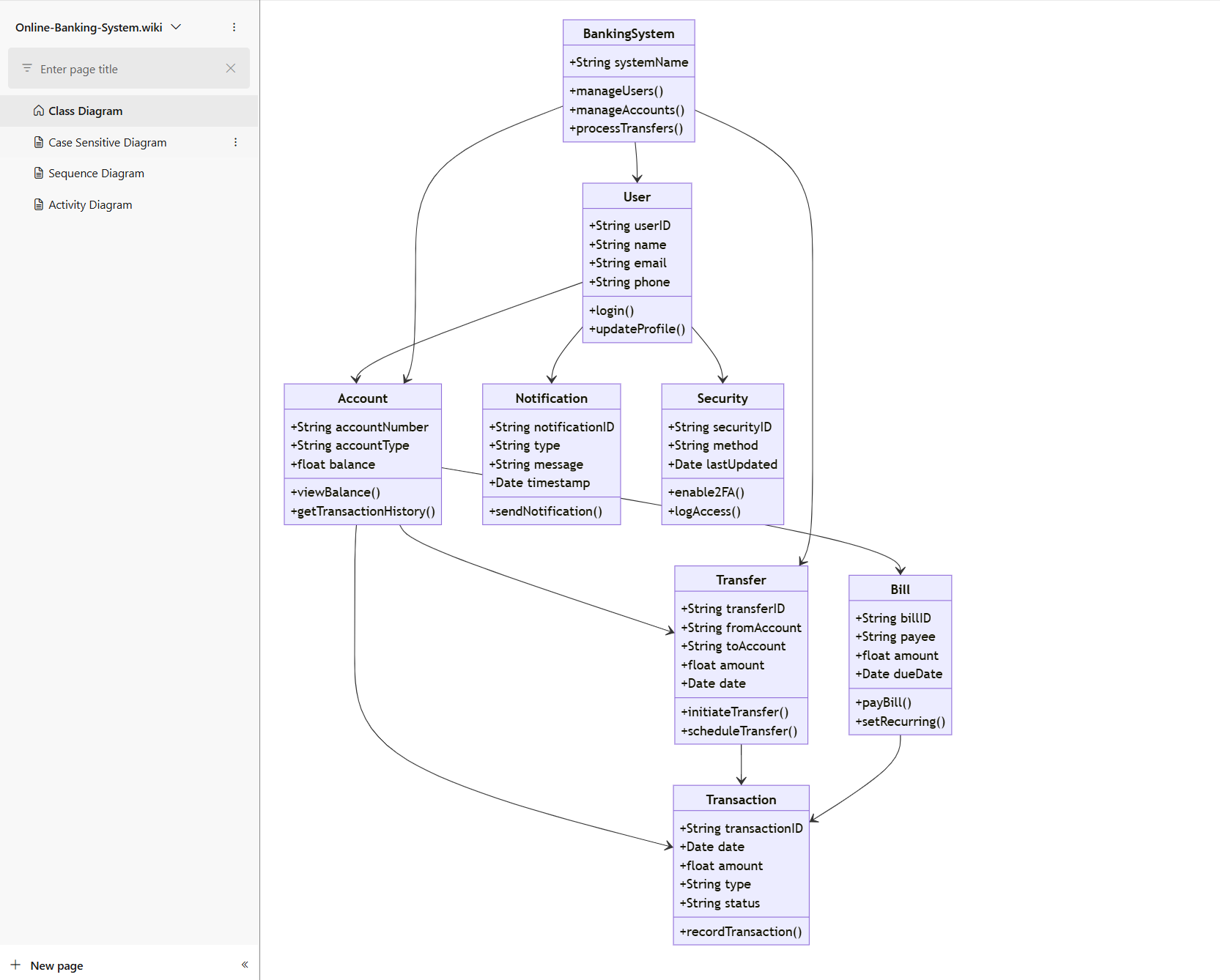
# RESULT:-

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

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

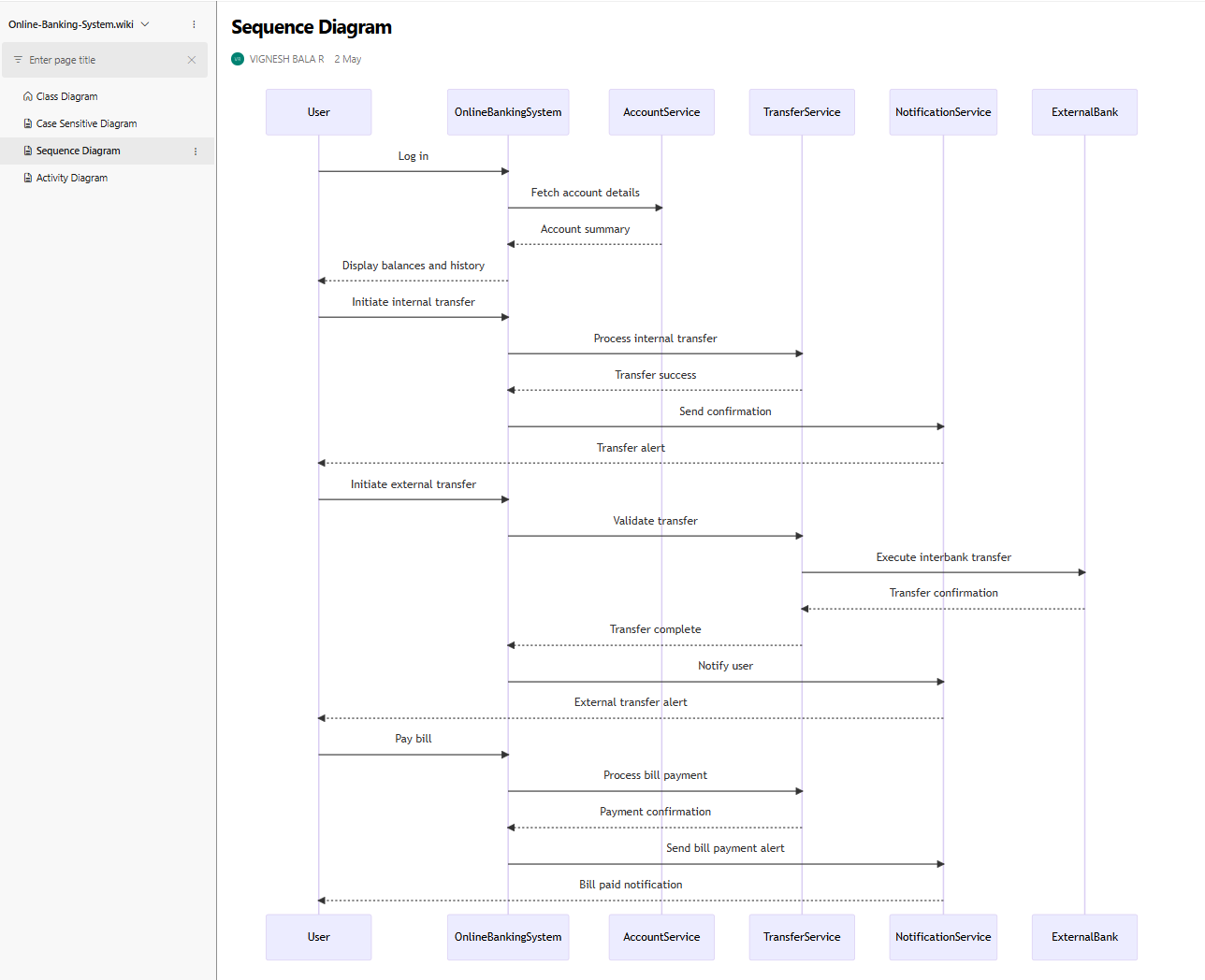
# AIM:-

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 the Online Banking System.

|  |  |
| --- | --- |
| **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:-**

**A diagram of a banking system

AI-generated content may be incorrect.**

# 7B. USE CASE DIAGRAM:-

**A diagram of a person's work flow

AI-generated content may be incorrect.**

**RESULT:-**

Thus activity and use case diagram has been designed successfully for Online Banking 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

### 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

### Define User Interactions

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

### 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.

### 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.

### 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.

### 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.

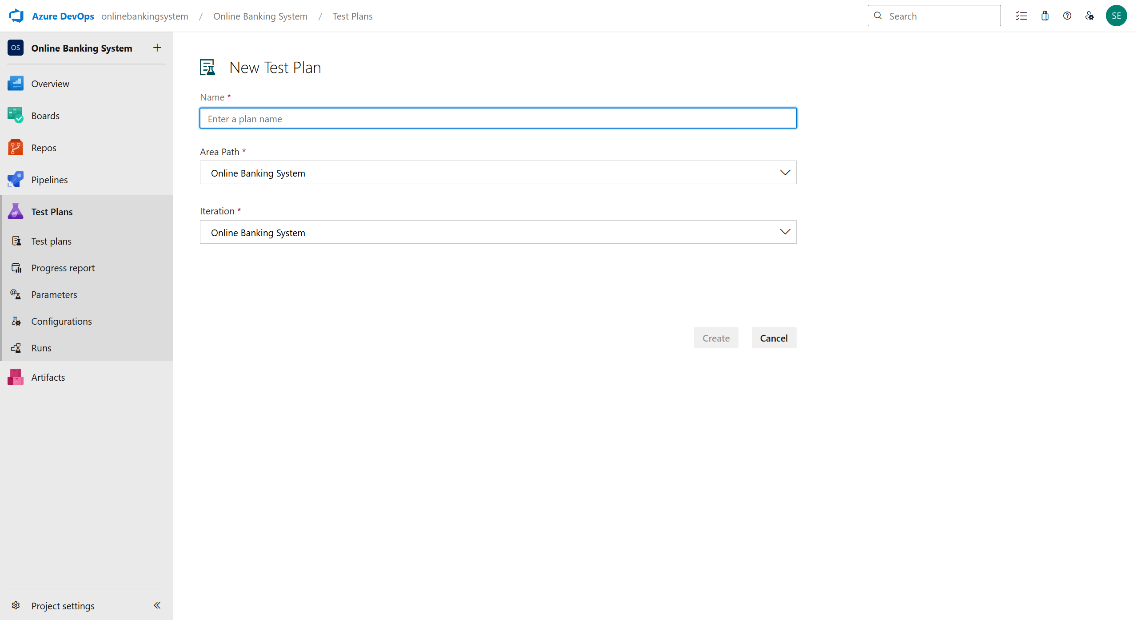
### Separate Test Suites

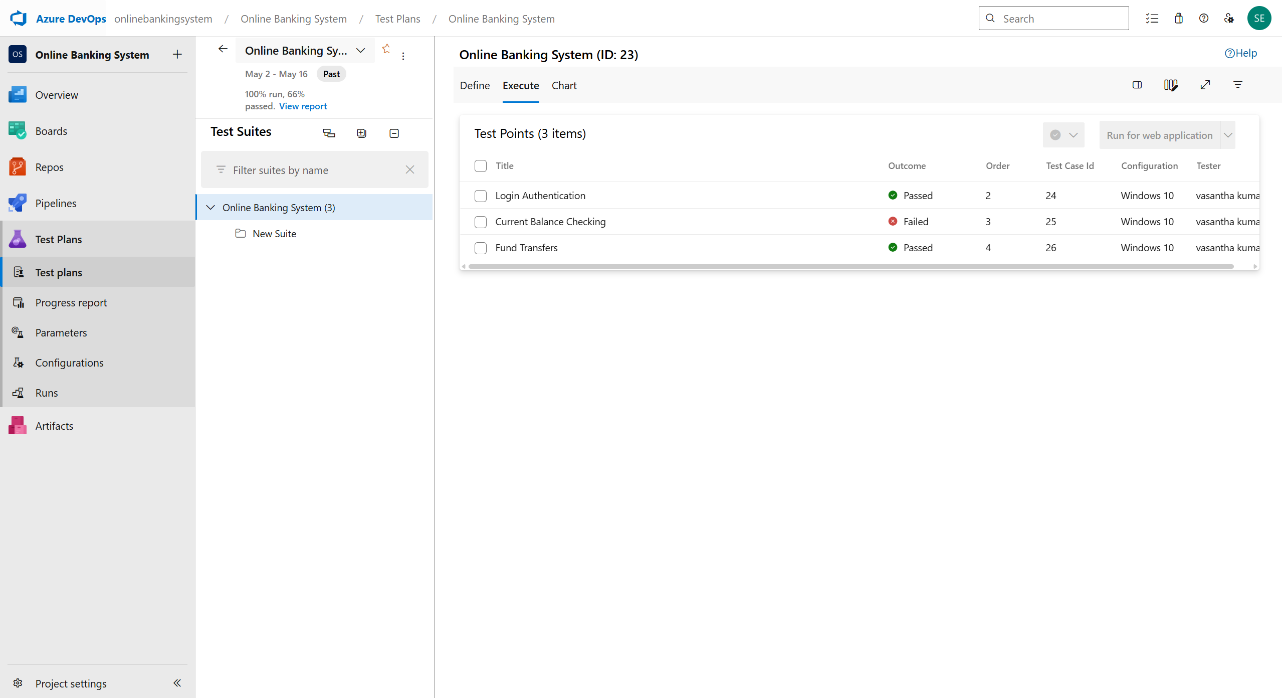
* + Grouped by functionality such as:
    - Login and Registration
    - Course Enrollment
    - Assignment Submission
    - Report Generation
    - Admin Functions
  + Improves organization and enables focused execution in Azure DevOps.

### 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.

### New test plan

****

1. **Test suite**
2. **Test case**

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

Online Banking System– Test Plans

**USER STORIES**

**US101: As a user, I want to register and log in securely to access my bank account.**

**US102: As a user, I want to view my account balance and transaction history.**

**US103: As a user, I want to transfer funds to another account securely.**

**US104: As a user, I want to pay utility bills directly from my account.**

**US105: As an admin, I want to manage user accounts and monitor transaction logs.**

**Test Suite: TS01 – User Registration & Login (ID: 301)**

**TC01 – Successful User Sign-Up**

**Action:**

**Go to Sign-Up page**

**Enter valid full name, email, and password**

**Click "Register"**

**Expected Result:**

**User account is created and redirected to dashboard**

**Type: Happy Path**

**TC02 – Login with Invalid Password**

**Action:**

**Go to Login page**

**Enter valid email and incorrect password**

**Click "Login"**

**Expected Result:**

**Error: "Invalid email or password"**

**Type: Error Path**

**Test Suite: TS02 – Account Overview (ID: 302)**

**TC03 – View Account Balance**

**Action:**

**Login → Navigate to "My Accounts"**

**Select account**

**Expected Result:**

**Balance and recent transactions are displayed**

**Type: Happy Path**

**TC04 – View Transactions Without Login**

**Action:**

**Open account dashboard without logging in**

**Try to access transaction history**

**Expected Result:**

**Error: "Please log in to view account details"**

**Type: Error Path**

**Test Suite: TS03 – Fund Transfer (ID: 303)**

**TC05 – Transfer Funds Successfully**

**Action:**

**Login → Navigate to "Transfer Funds"**

**Enter valid recipient account, amount, and purpose**

**Click "Submit"**

**Expected Result:**

**Confirmation: "Transfer successful"**

**Type: Happy Path**

**TC06 – Transfer Funds with Insufficient Balance**

**Action:**

**Login → Navigate to "Transfer Funds"**

**Enter recipient and amount greater than balance**

**Click "Submit"**

**Expected Result:**

**Error: "Insufficient balance"**

**Type: Error Path**

**Test Suite: TS04 – Bill Payments (ID: 304)**

**TC07 – Pay Utility Bill Successfully**

**Action:**

**Login → Go to "Pay Bills"**

**Select biller, enter amount**

**Click "Pay"**

**Expected Result:**

**Message: "Payment successful"**

**Type: Happy Path**

**TC08 – Pay Bill with Expired Card**

**Action:**

**Login → Pay Bills**

**Use expired debit card credentials**

**Click "Pay"**

**Expected Result:**

**Error: "Payment failed – card expired"**

**Type: Error Path**

**Test Suite: TS05 – Admin Management (ID: 305)**

**TC09 – Deactivate User Account**

**Action:**

**Login as Admin → Go to "User Management"**

**Select user → Click "Deactivate"**

**Expected Result:**

**Confirmation: "User account deactivated"**

**Type: Happy Path**

**TC10 – Monitor Logs Without Privileges**

**Action:**

**Login as normal user**

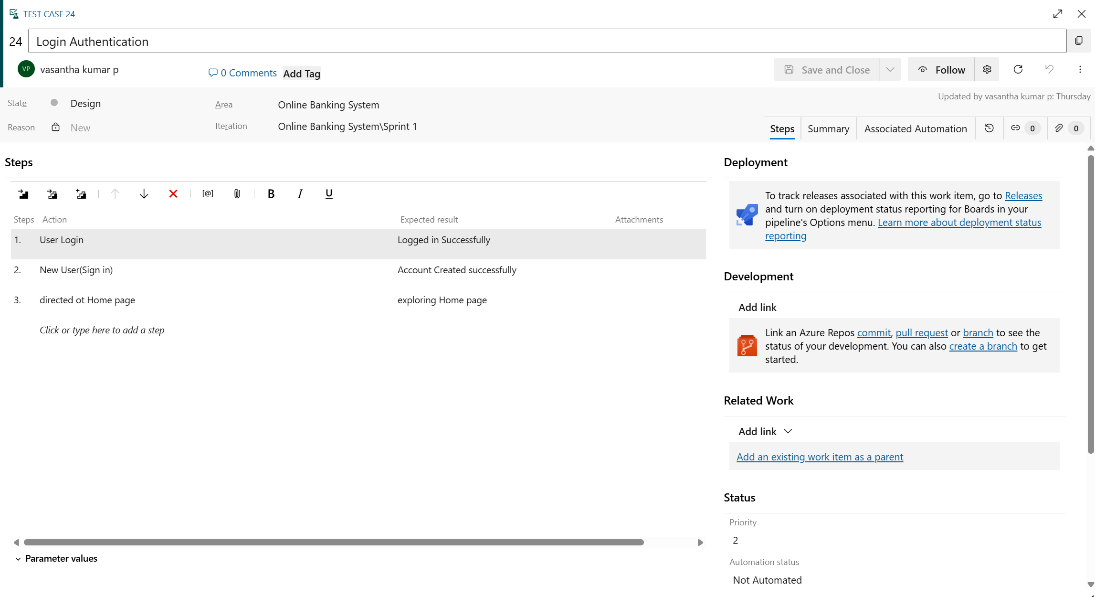
**Try accessing "Transaction Logs"**

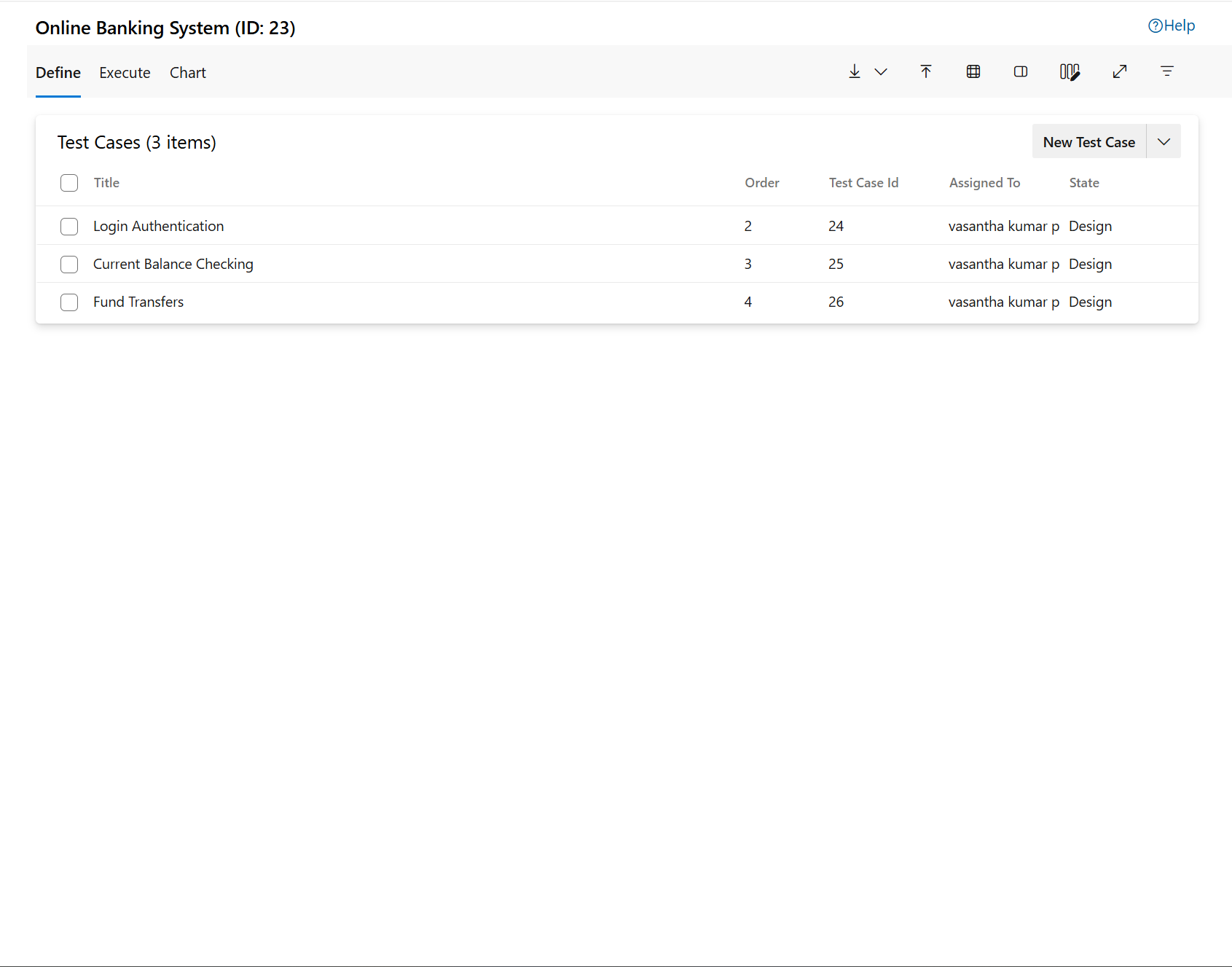
**Expected Result:**

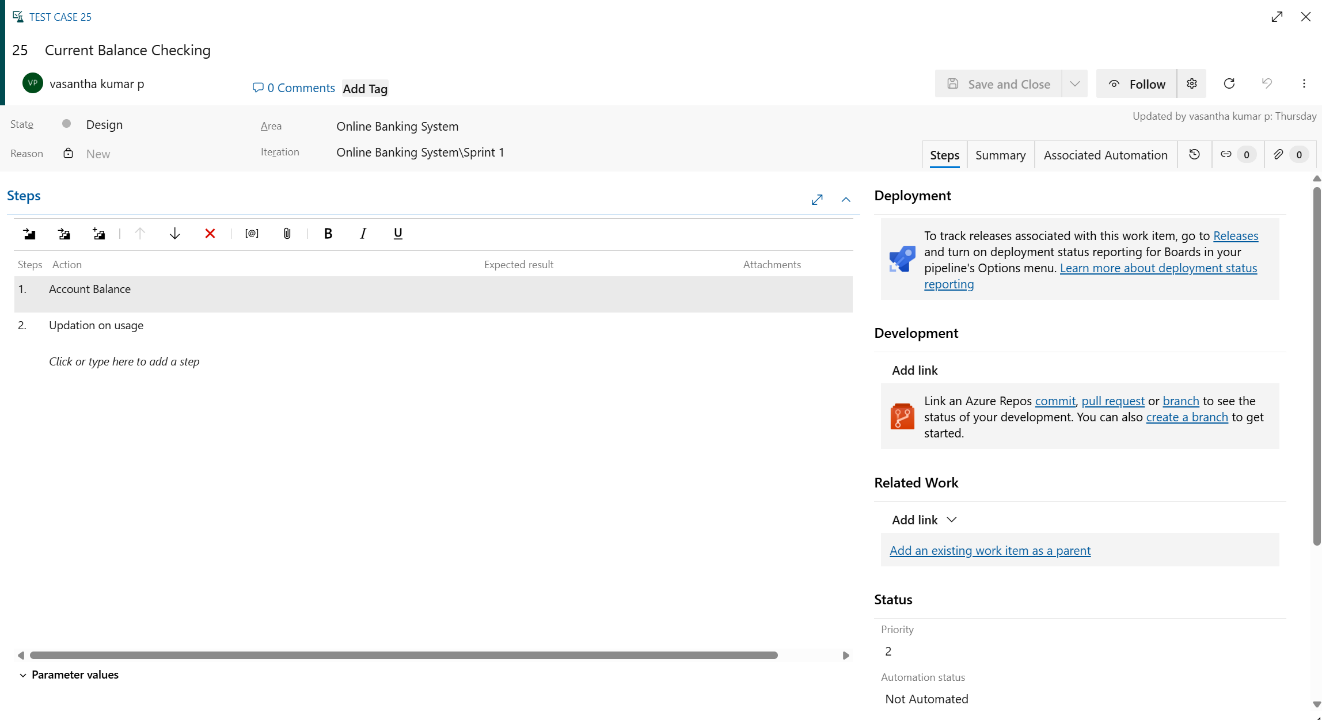
**Error: "Access denied – admin privileges required"**

**Type: Error Path**

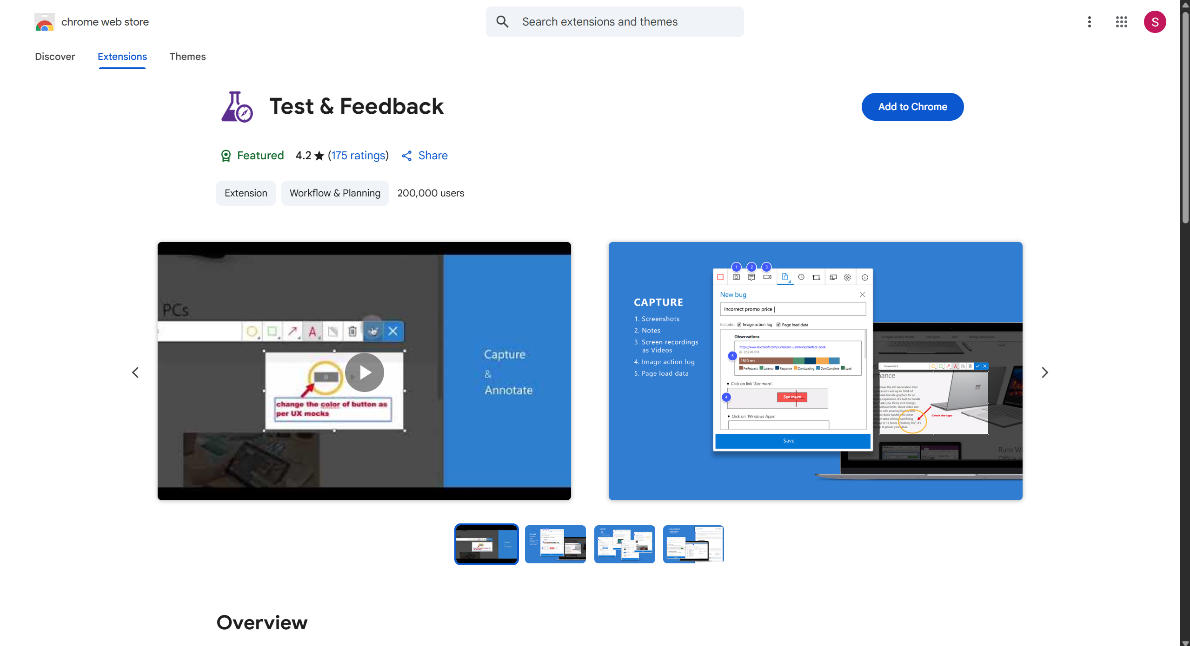
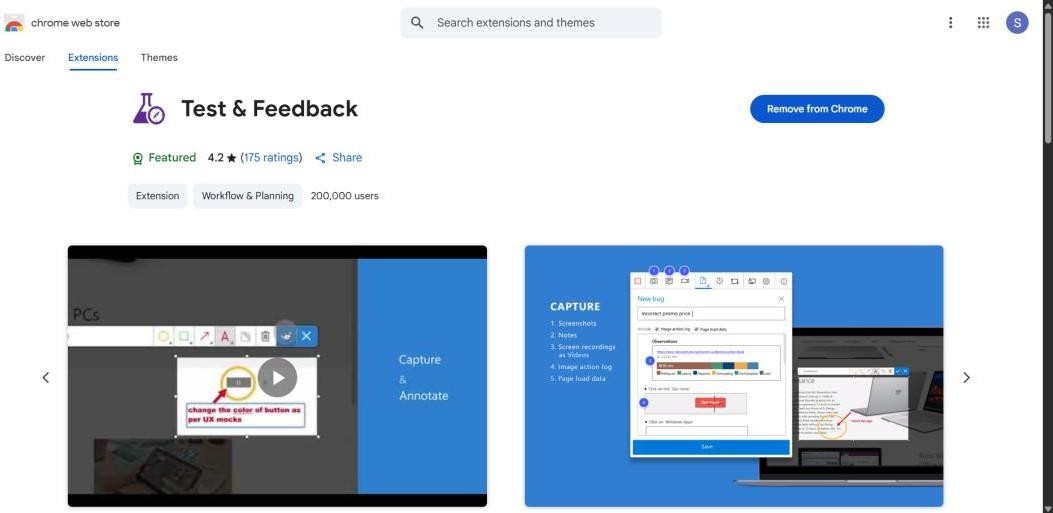
1. **Test Cases**

****

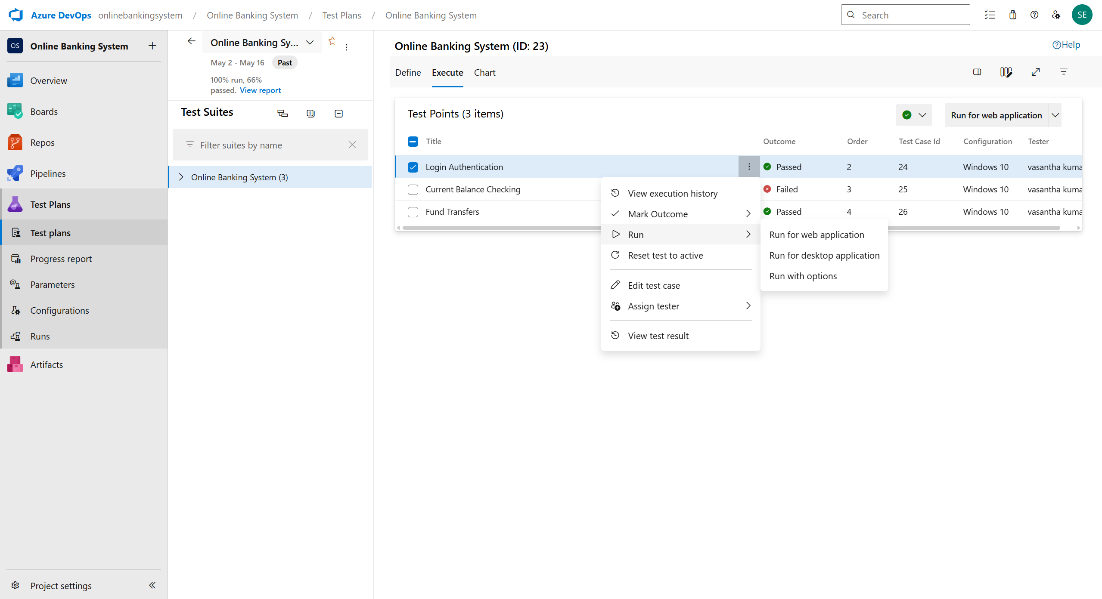
****

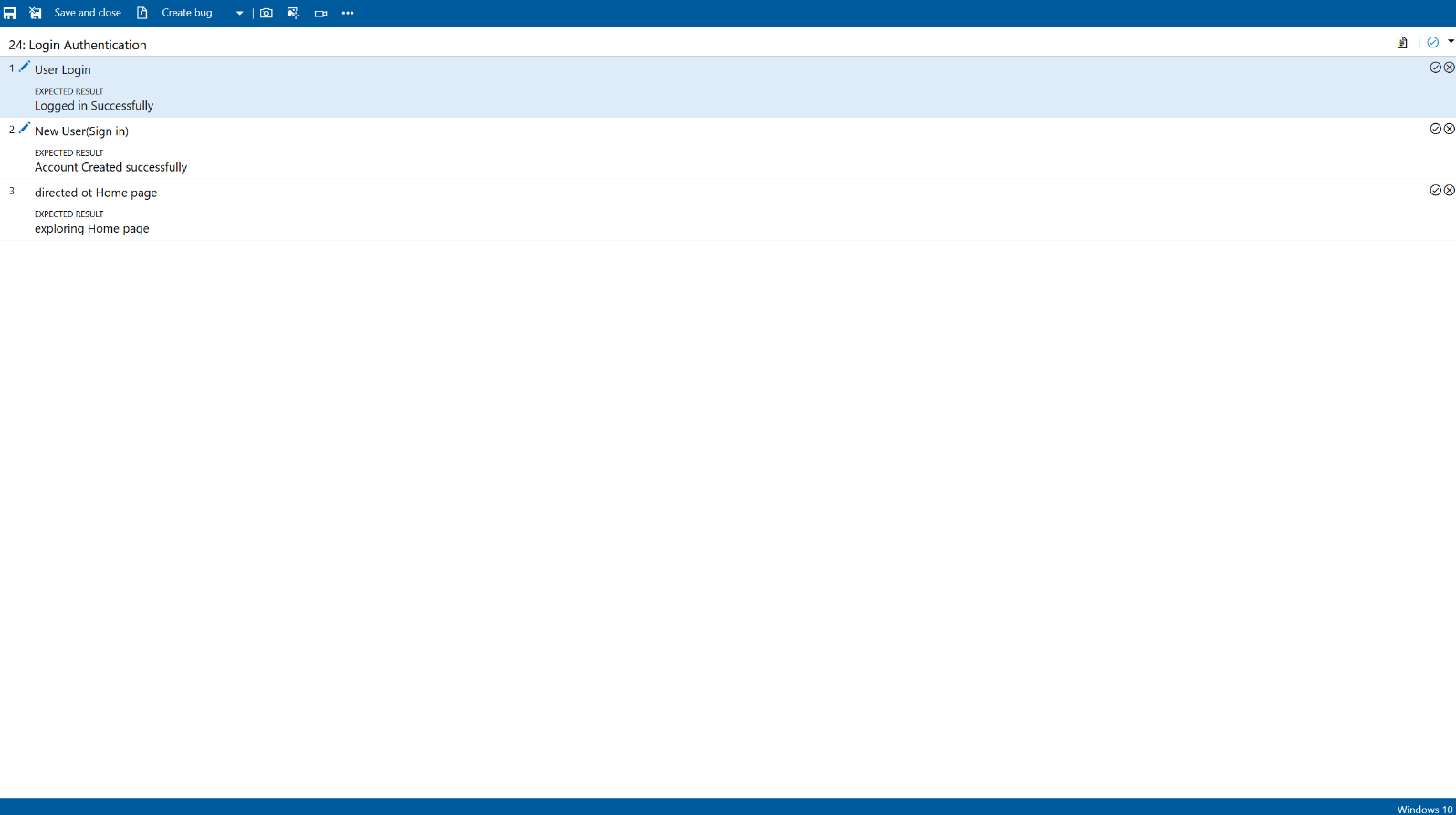


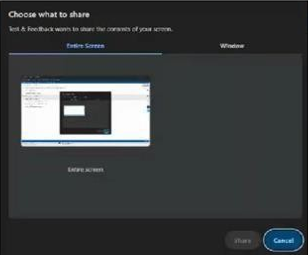
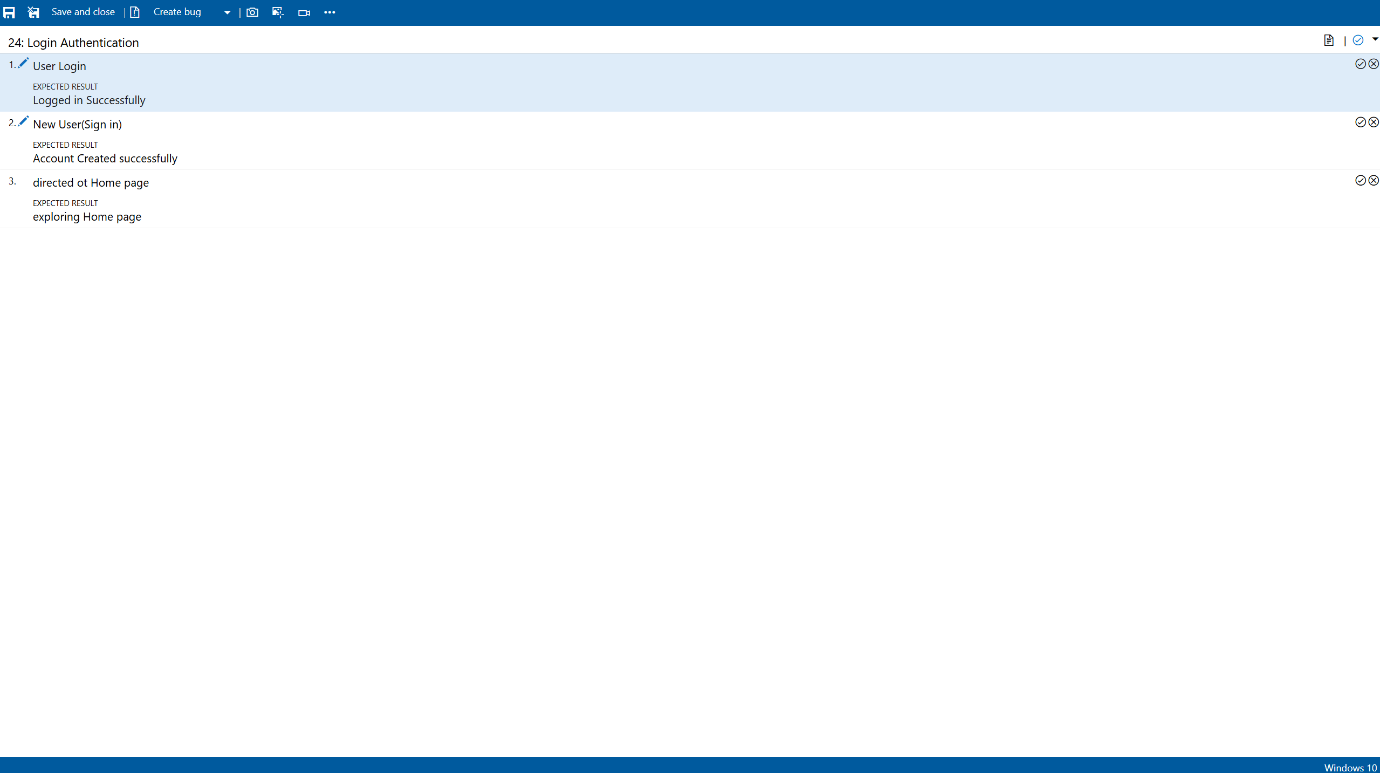
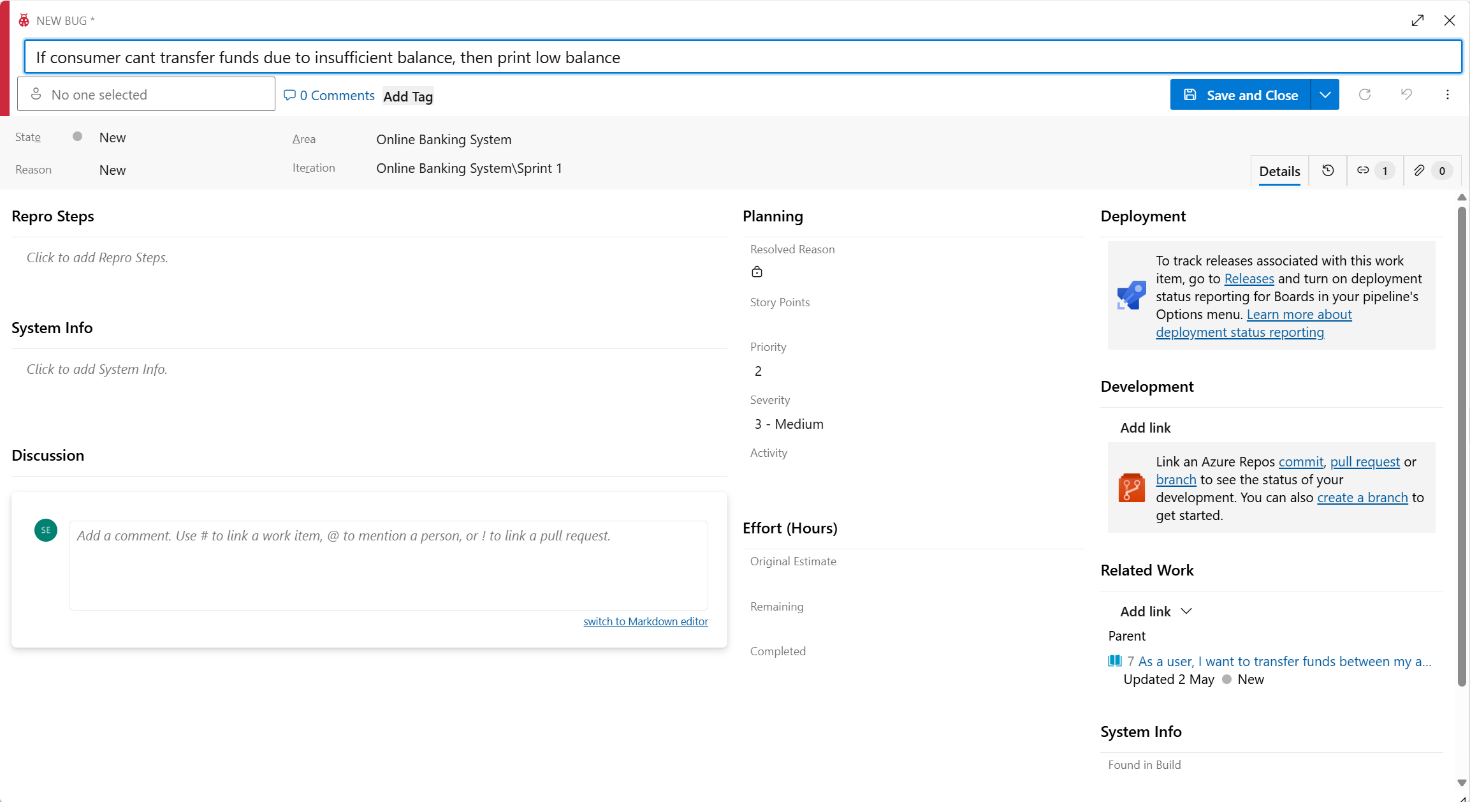
****

1. **Installation of test**
2. ****

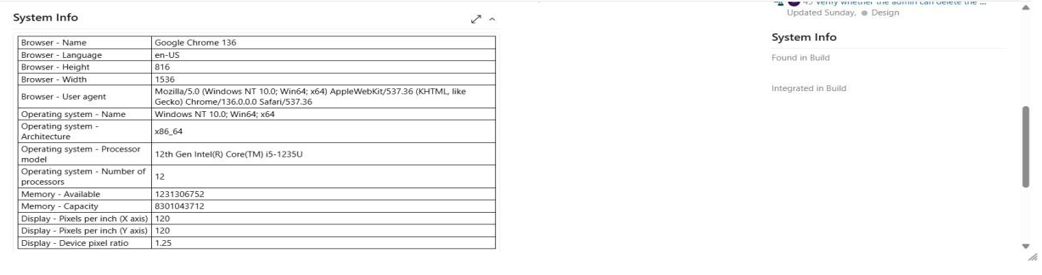
****

1. **Running the test cases**

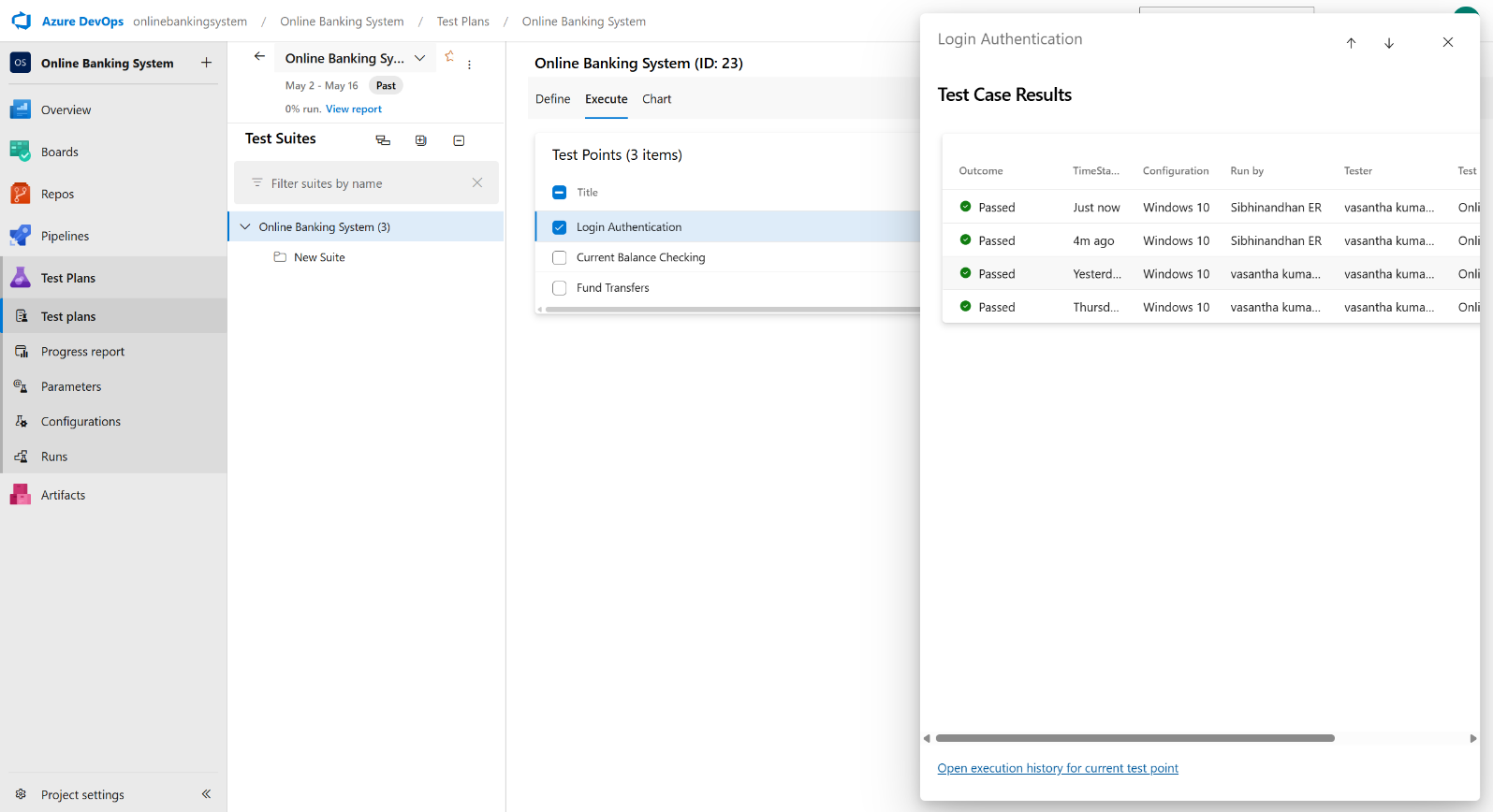
****

1. **Recording the test case**
2. **Creating the bug**

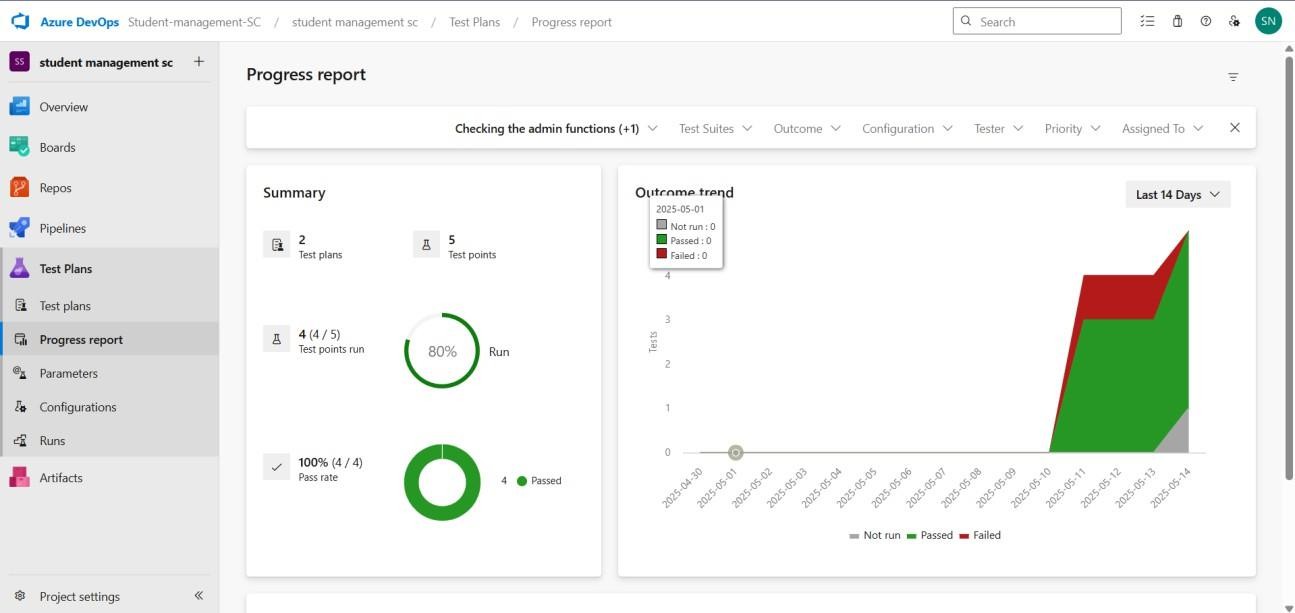
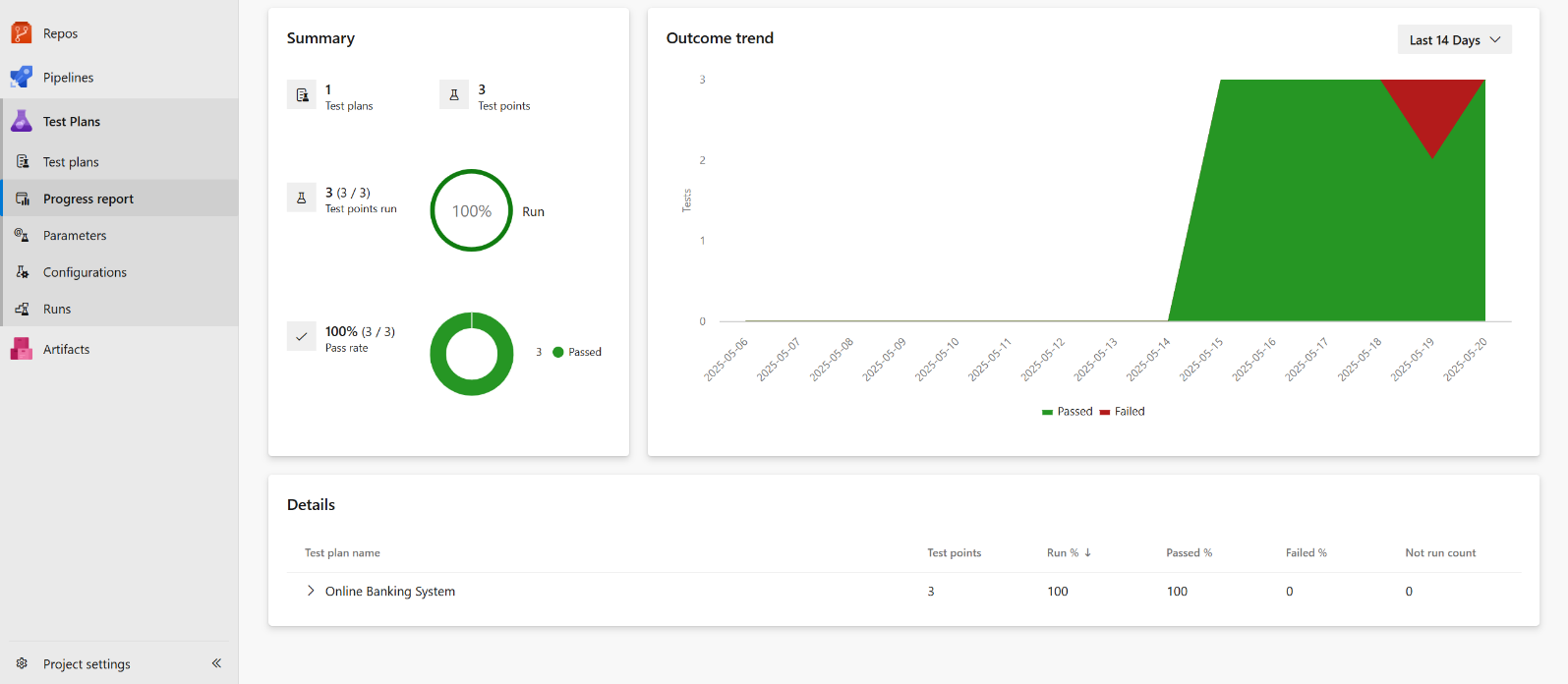
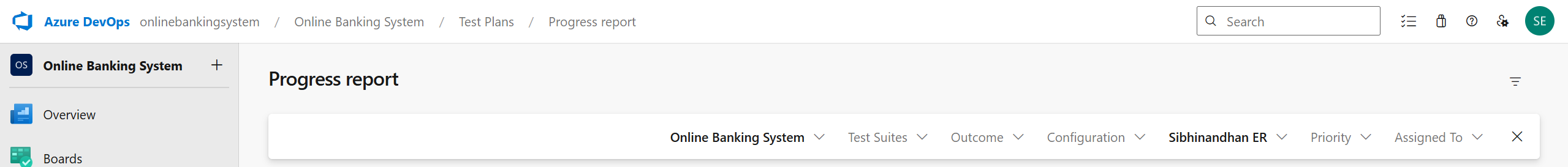


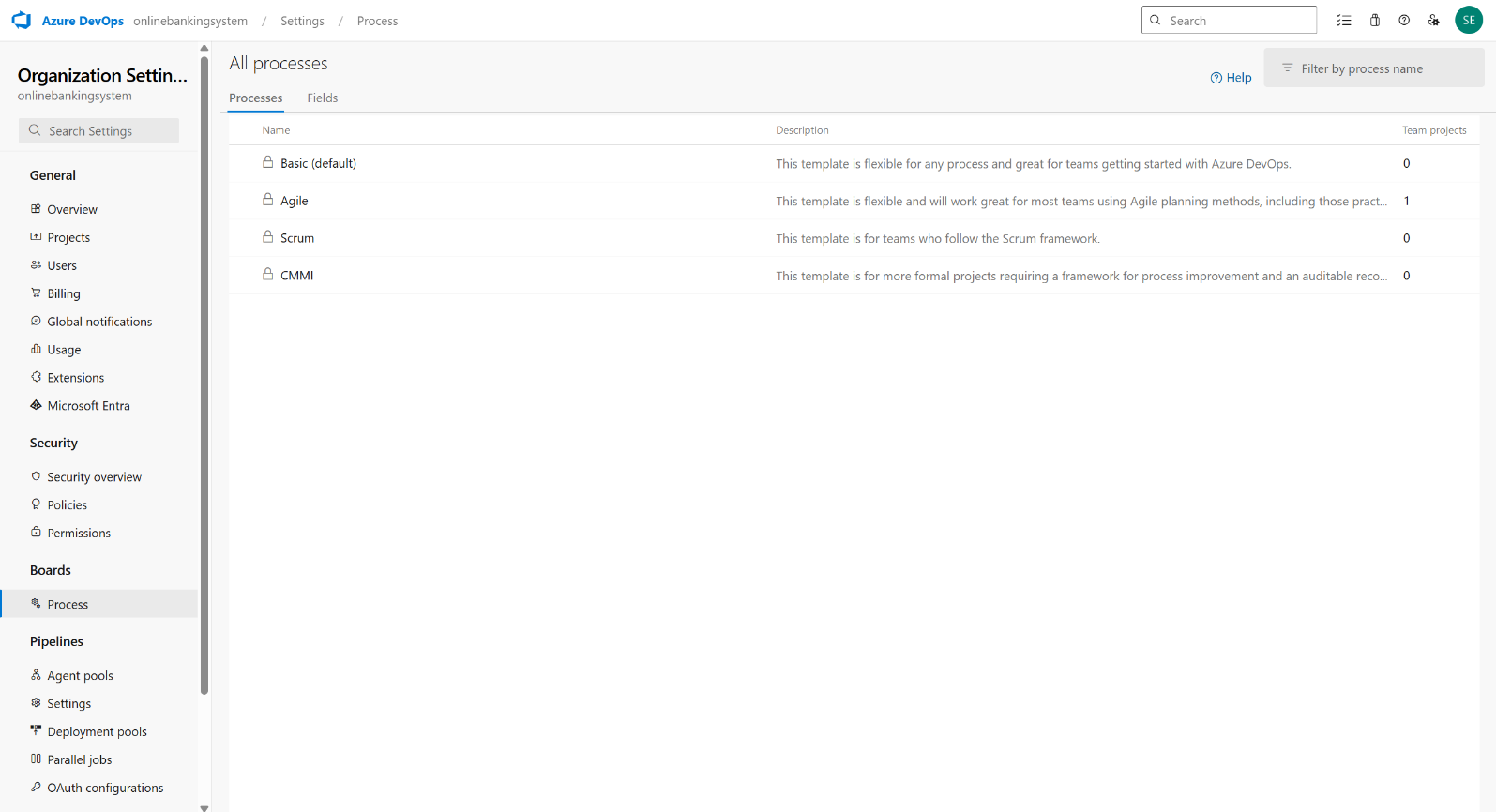
****

**Test Results**

****

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 Online Banking 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 Online Banking 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
2. Install dependencies (e.g., npm install, dotnet restore)
3. Build the application (dotnet build, npm run build)
4. Run unit tests (dotnet test, npm test)
5. Publish build artifacts to be used in the release stage
6. 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.
2. 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.

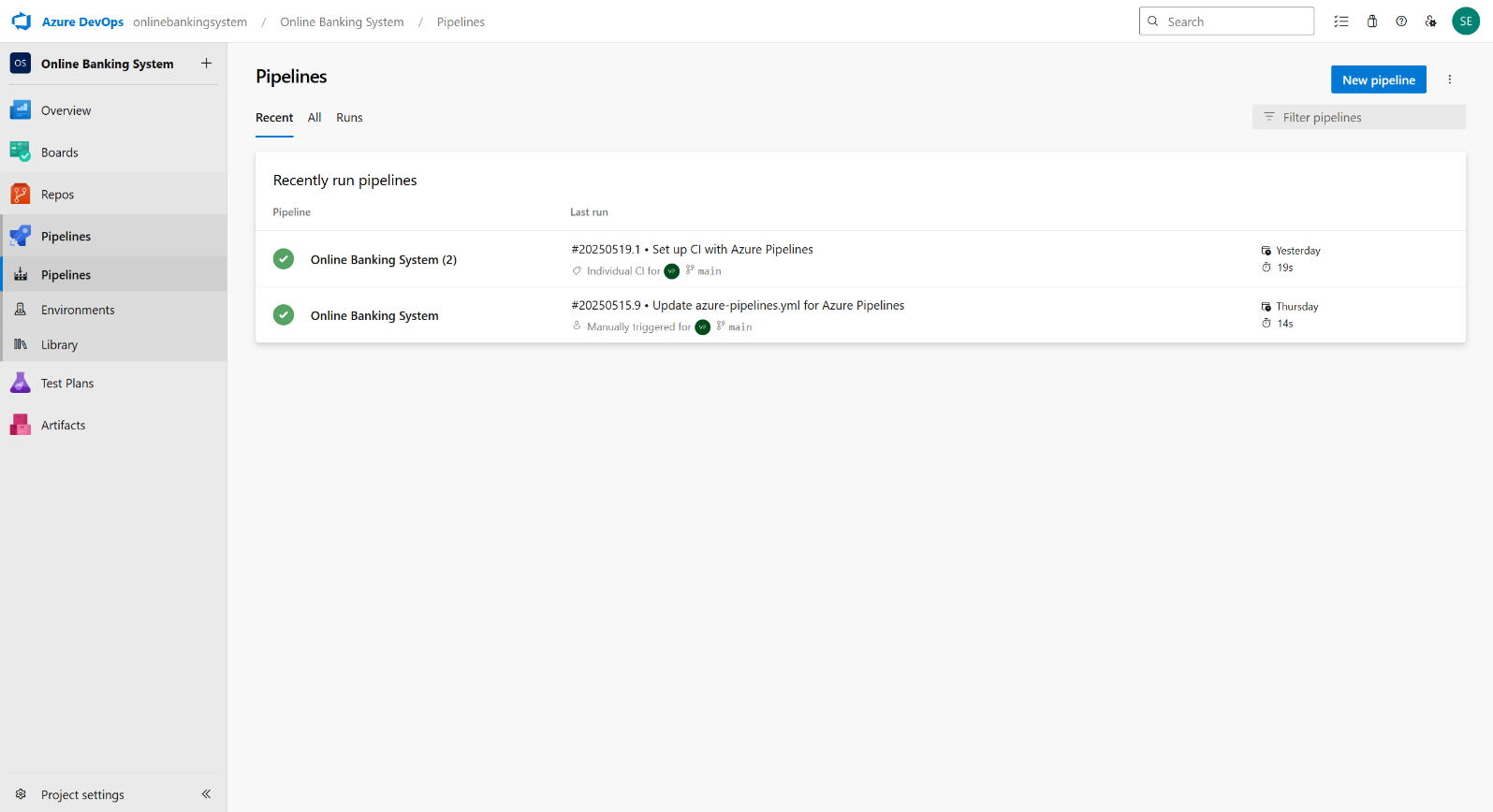
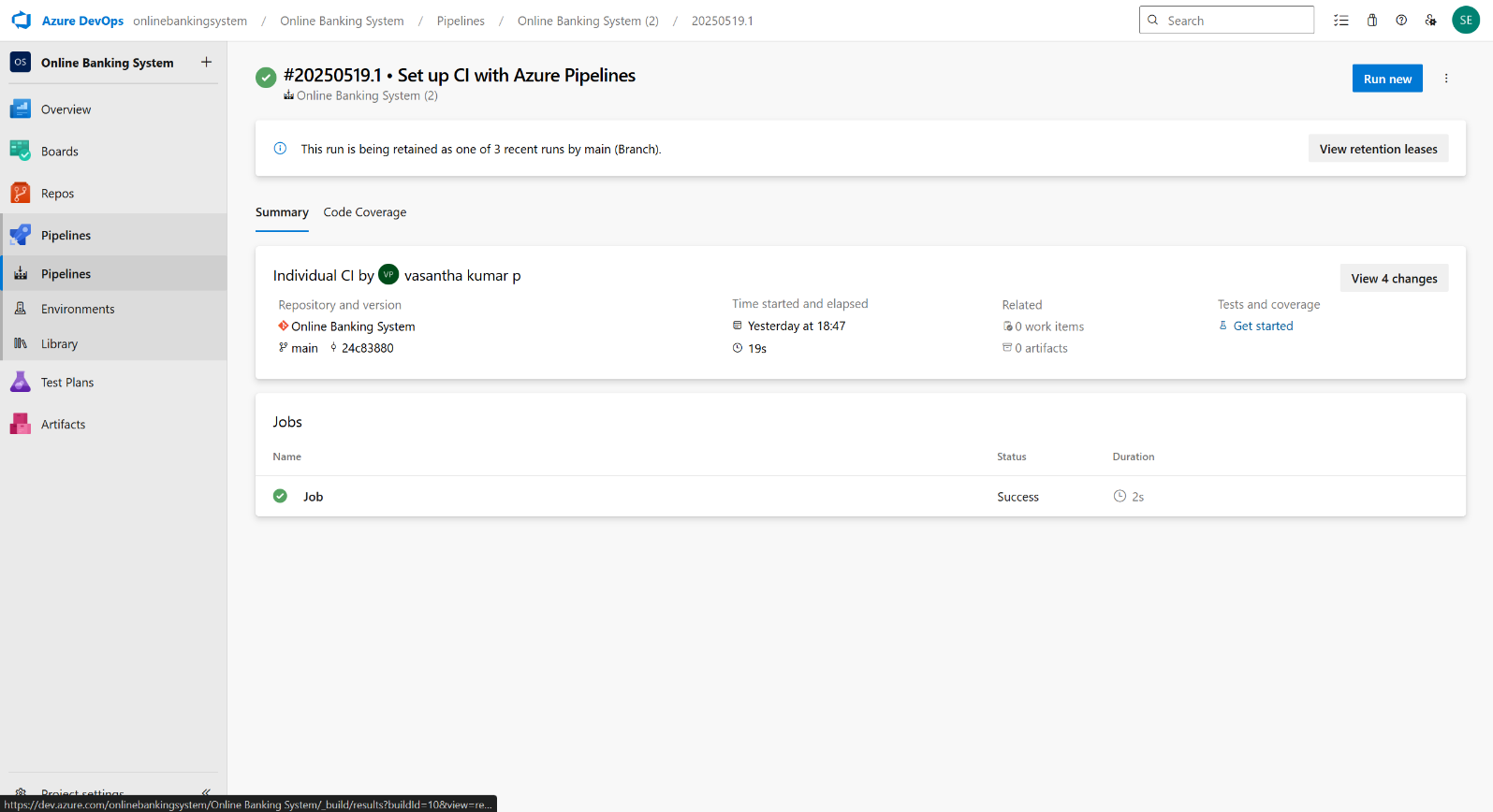
Check logs for build/test/deploy stages to debug any errors.

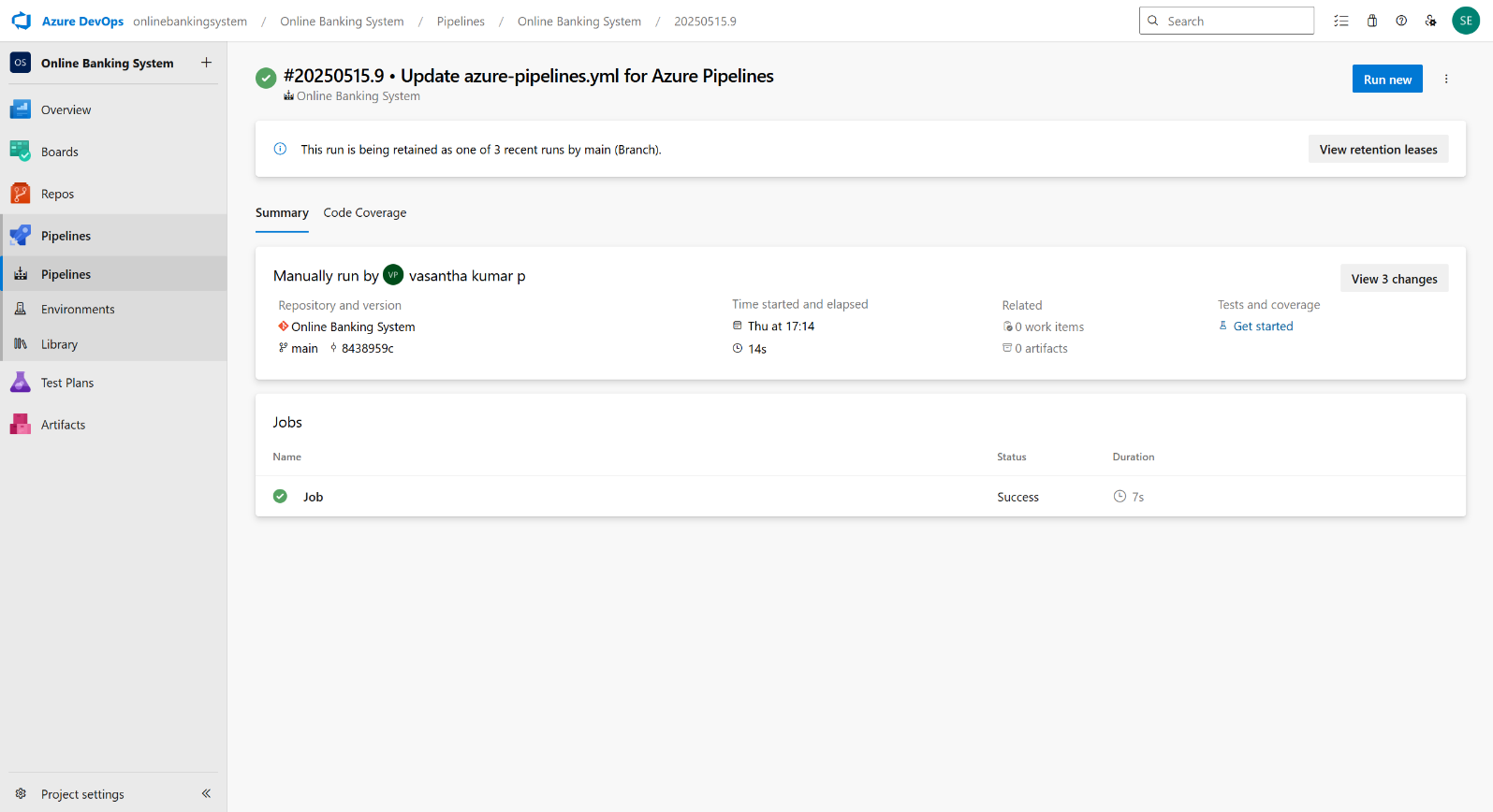
You can also integrate email alerts or Microsoft Teams notifications for build failures.

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:

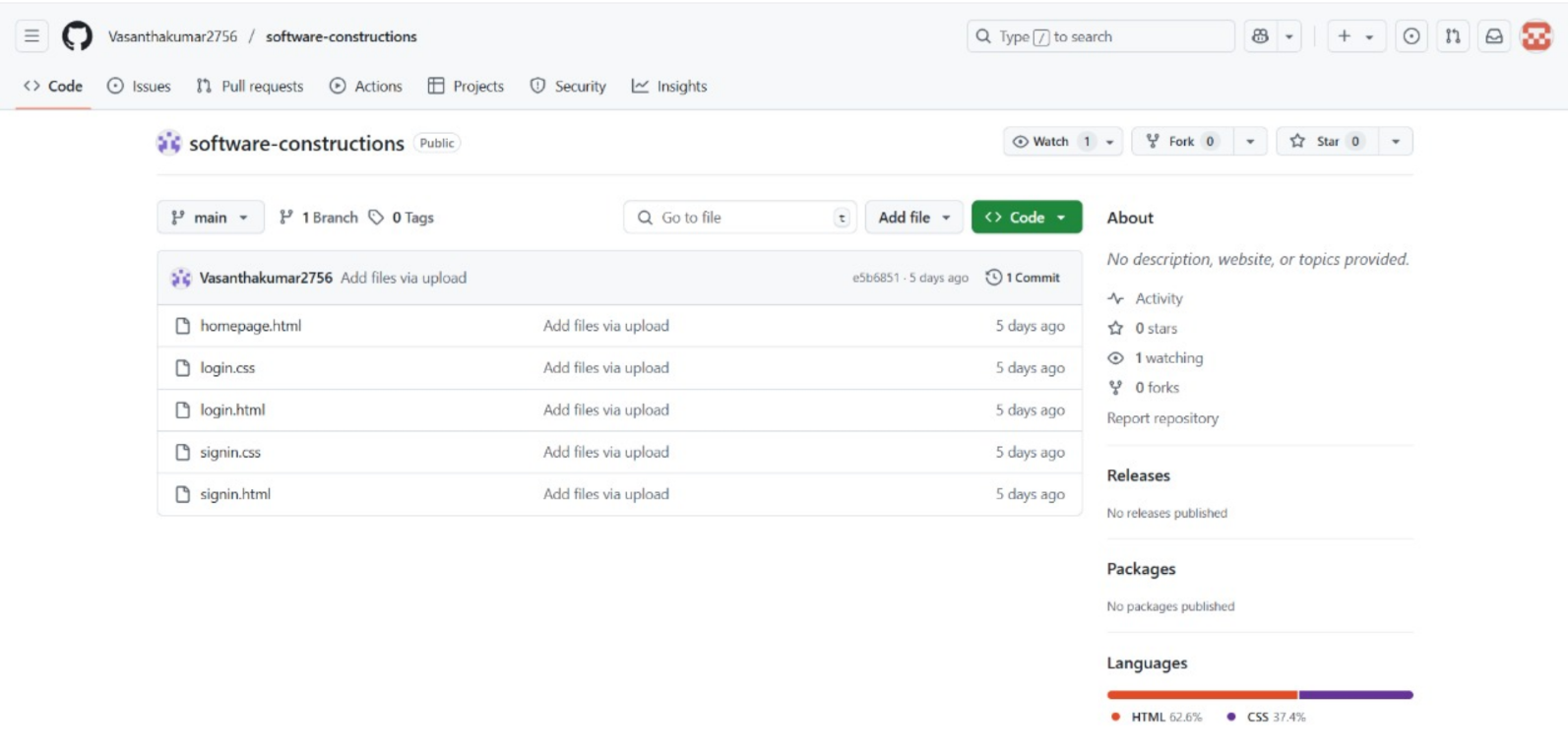
Thus the pipelines for the given project “Online Banking System has been executed successfully

|  |  |
| --- | --- |
| **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 Online Banking 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.