INDEX

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 Epics, 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 Diagrams for Project Architecture
7.	04/3/25	Designing Use-Case and Activity Diagrams for Project Architecture
8.	25/3/25	Testing – Test Plans and Test Cases
9.	15/4/25	CI/CD Pipelines in Azure
10.	22/4/25	GitHub: Project Structure & Naming Convention

EXP NO: 1	AZURE DEVOPS ENVIRONMENT SETUP

Aim:

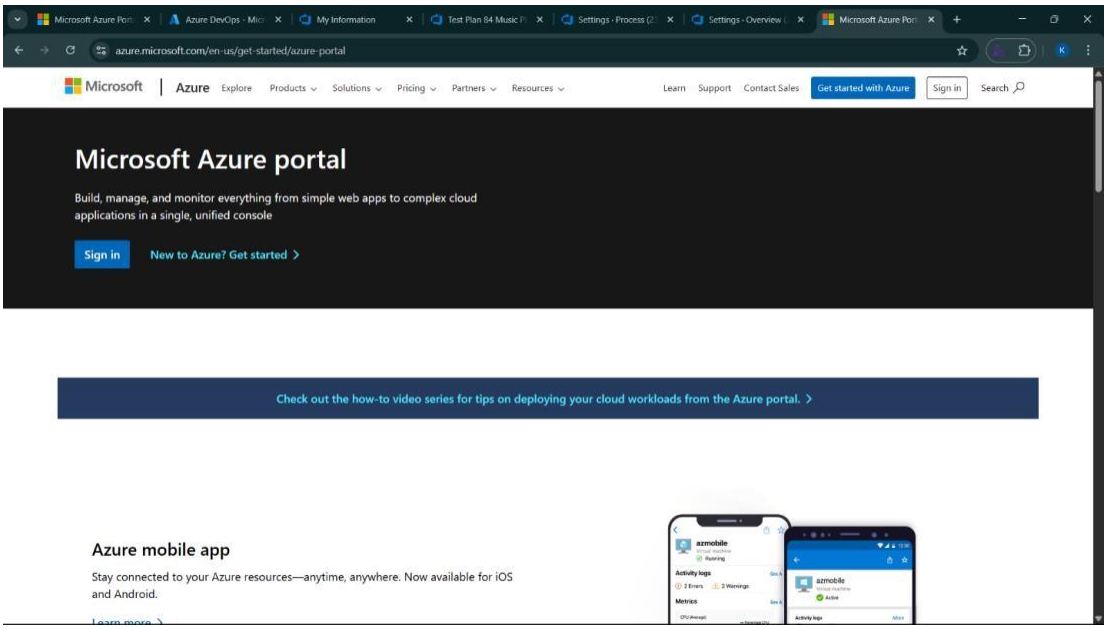
To set up and access the Azure DevOps environment by creating an organization through the Azure portal.

INSTALLATION

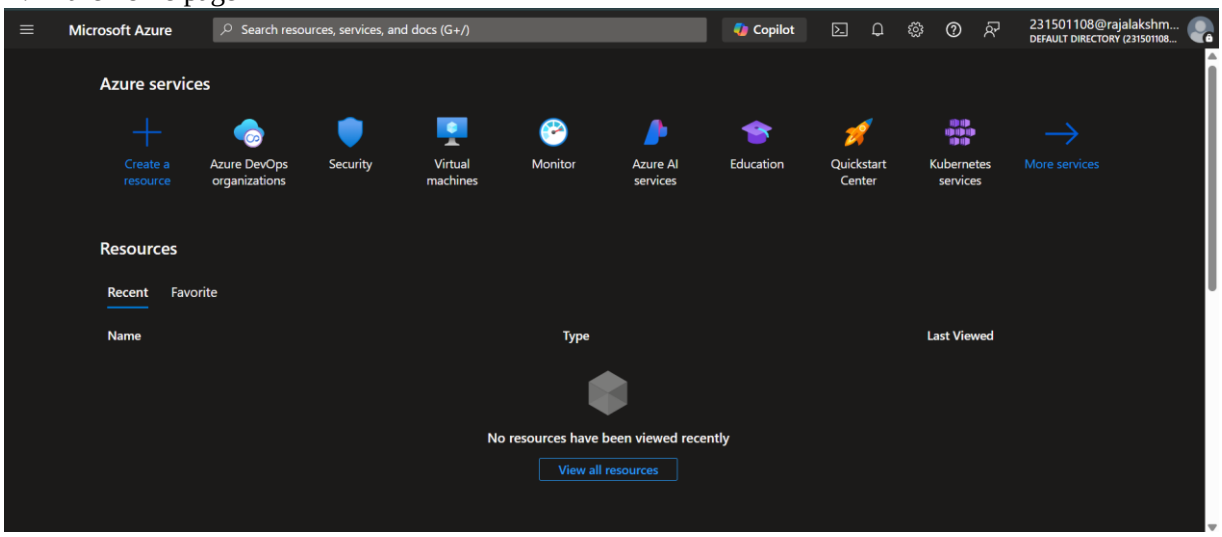
1. Open your web browser and go to the Azure website: https://azure.microsoft.com/en-us/get-started/azure-portal.

Sign in using your Microsoft account credentials.

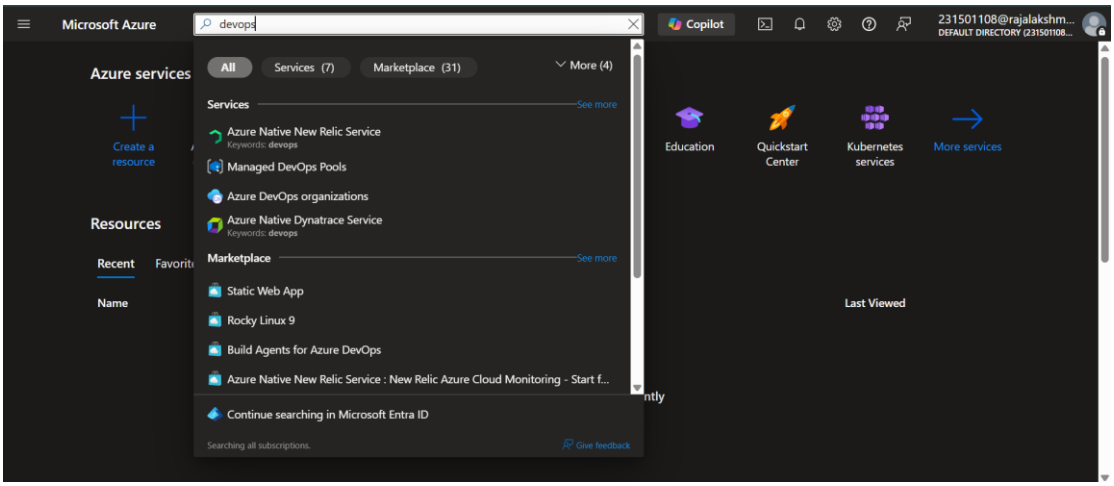
If you don't have a Microsoft account, you can create one here: https://signup.live.com/?lic=1



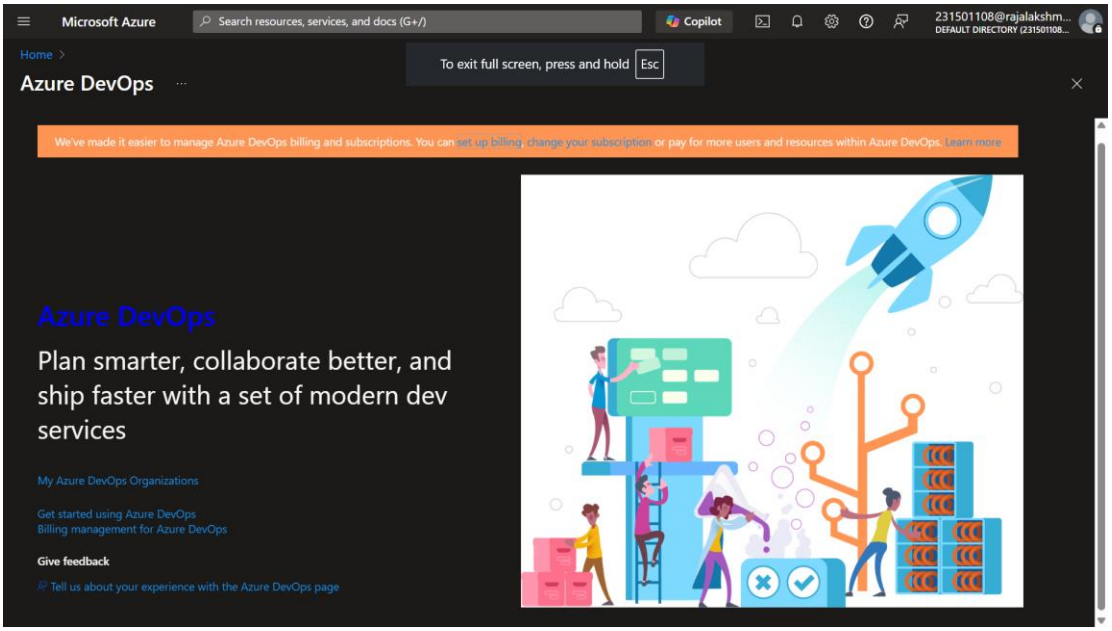
2. Azure home page



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:

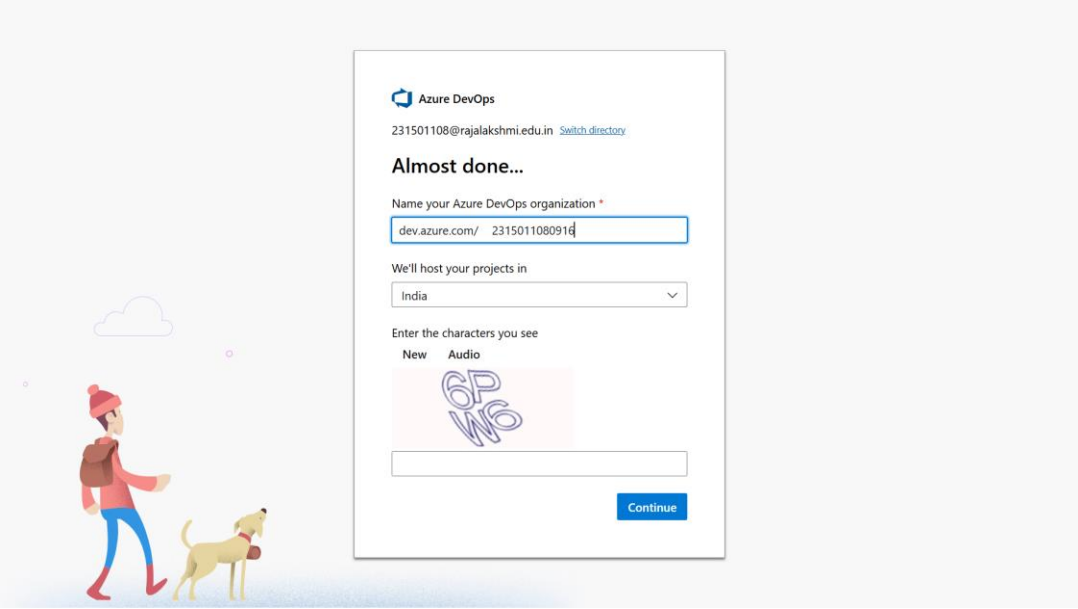
Successfully accessed the Azure DevOps environment and created a new organization through the Azure portal.

EXP NO: 2	AZURE DEVOPS PROJECT SETUP AND USER STORY 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

a. 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.

b. On the organization's **Home page**, click on the **New Project** button.

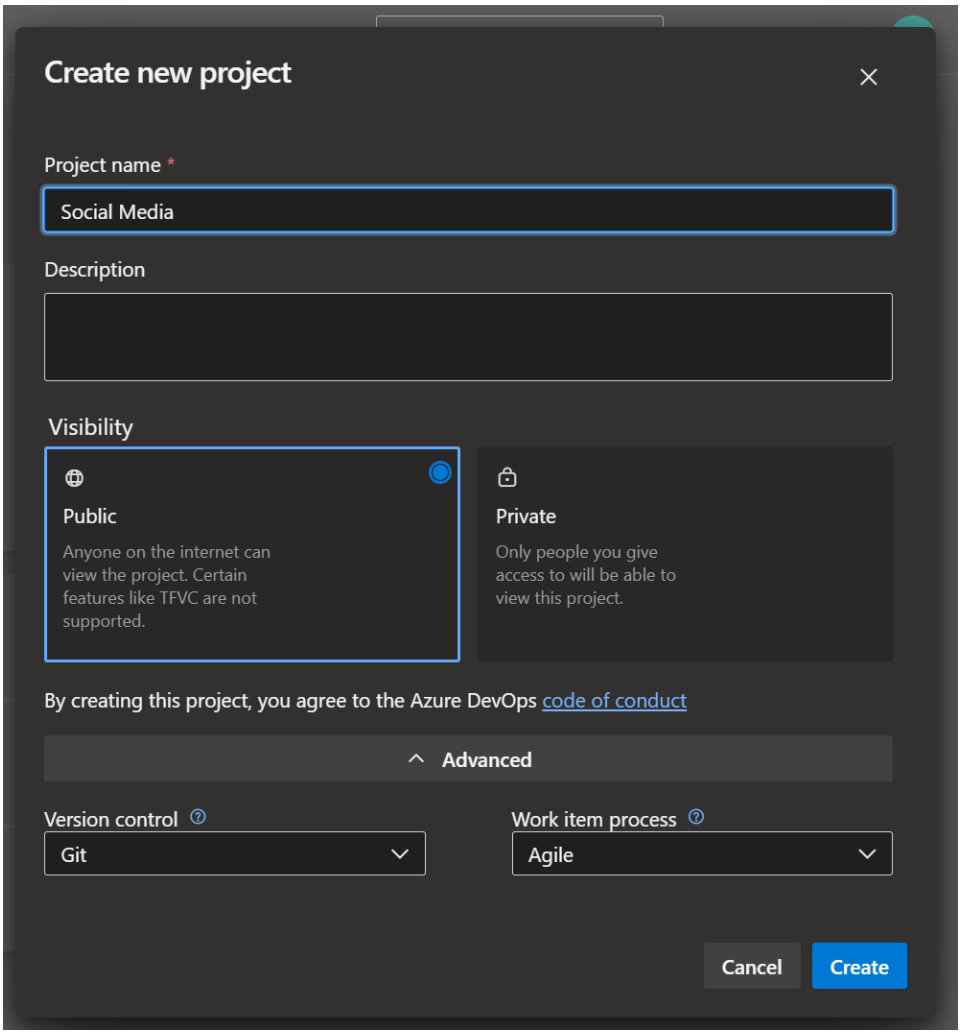
c. Enter the project name, description, and visibility options:

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

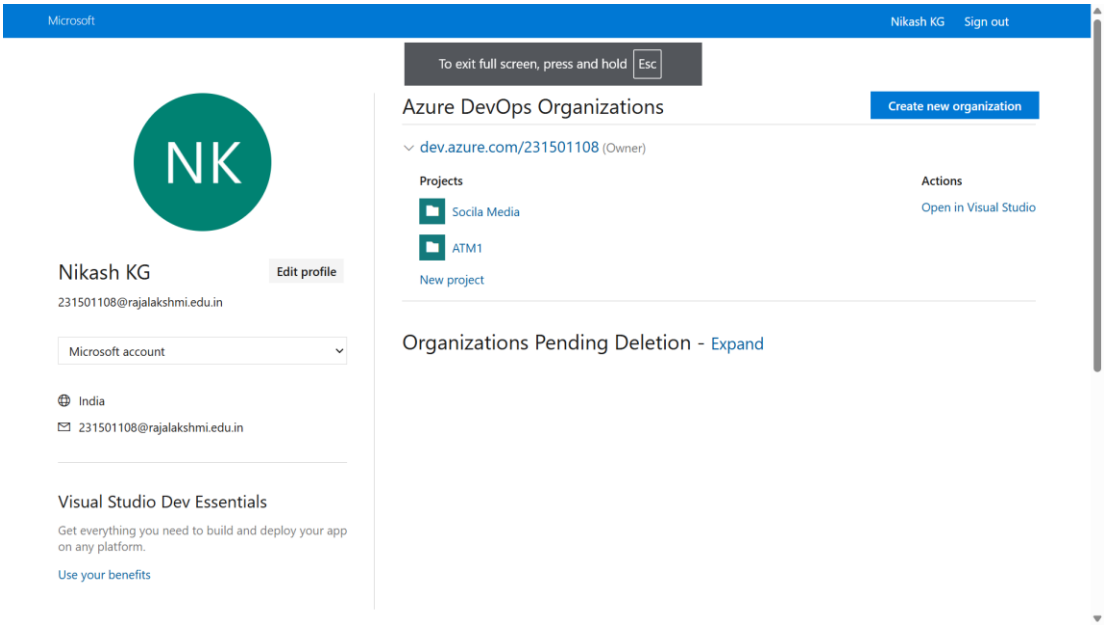
Description: Optionally, add a description to provide more context about the project.

Visibility: Choose whether you want the project to be **Private** (accessible only to those invited) or **Public** (accessible to anyone).

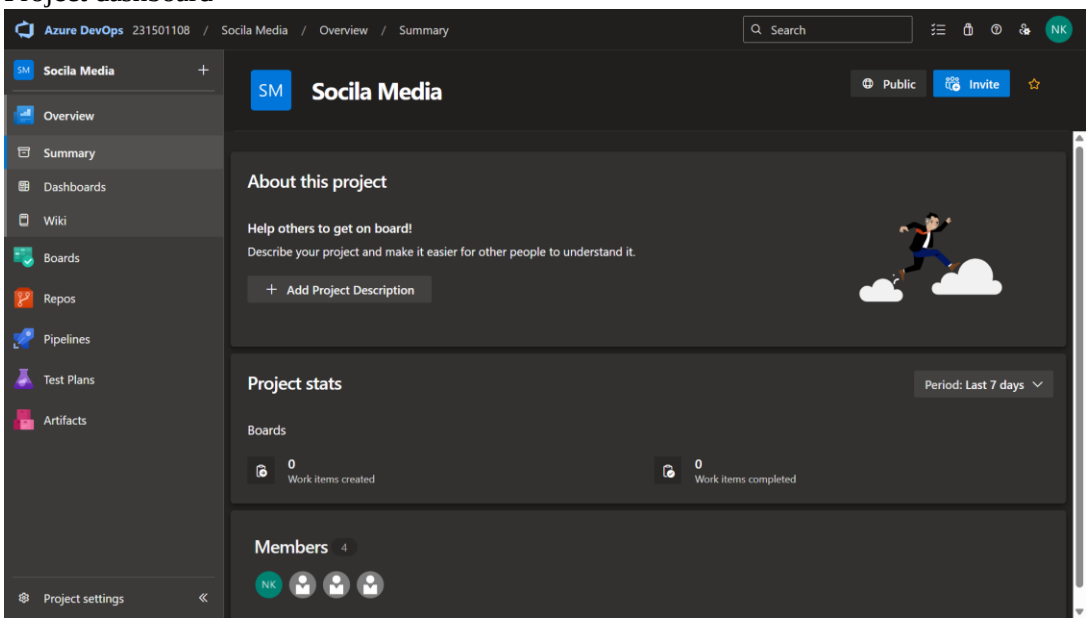
d. Once you've filled out the details, click **Create** to set up your first project.



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.



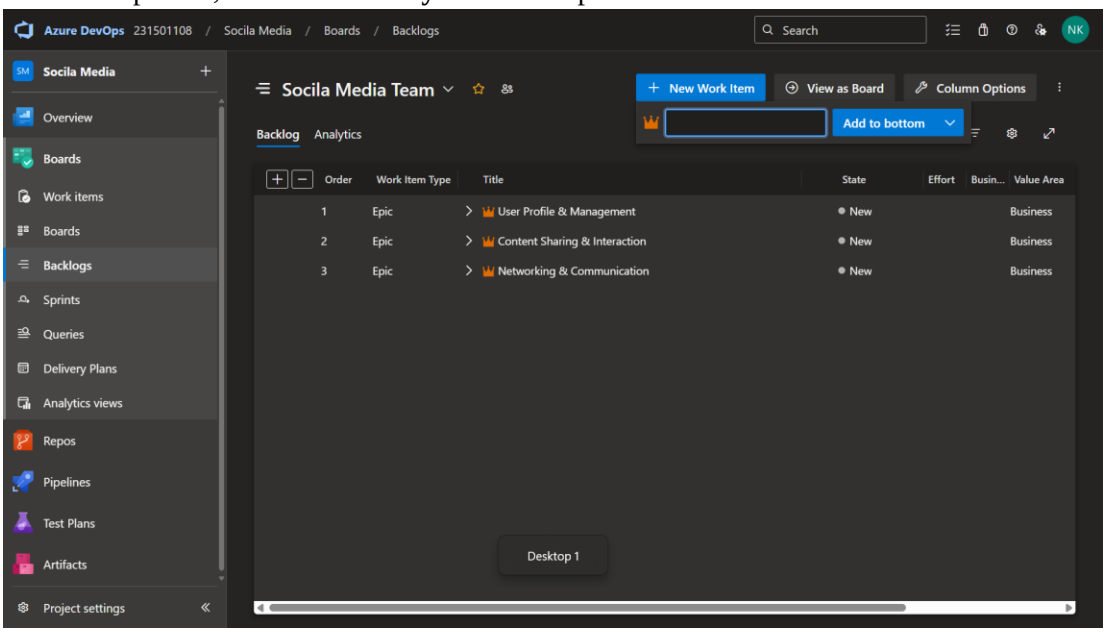
4. Project dashboard

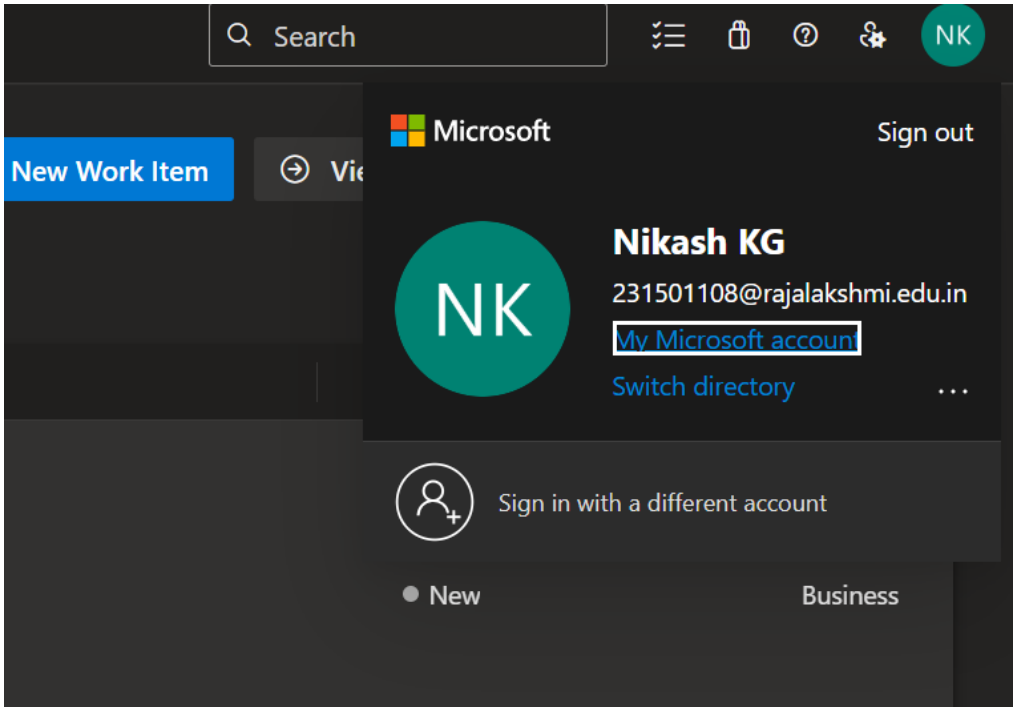
5. 

6. To manage user stories:

a. 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.

b. 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:

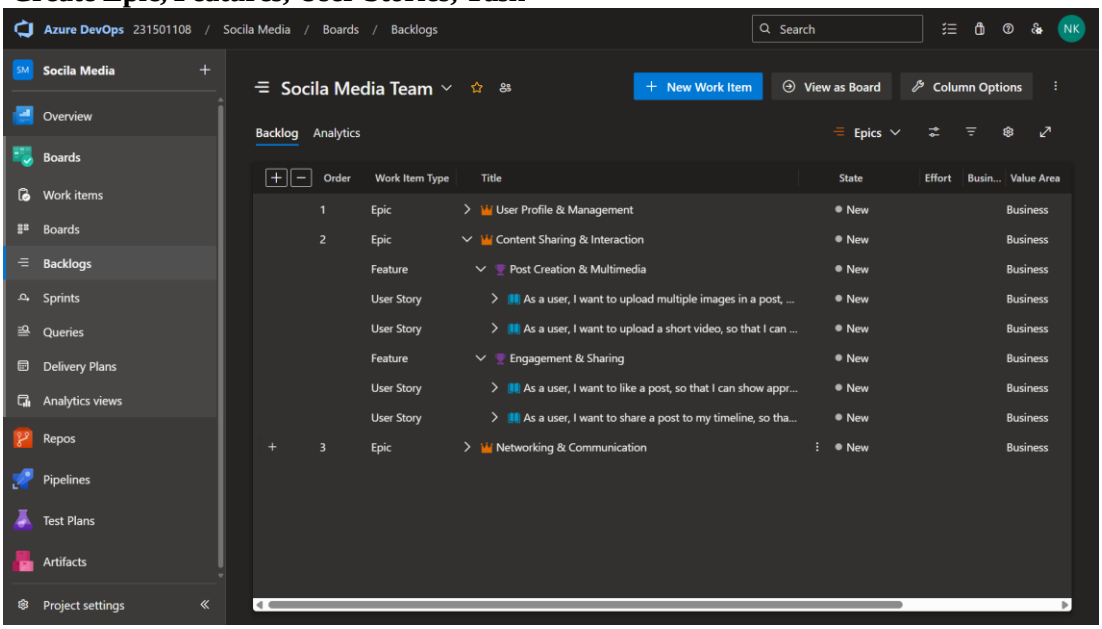
Successfully created an Azure DevOps project with user story management and agile workflow setup.

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

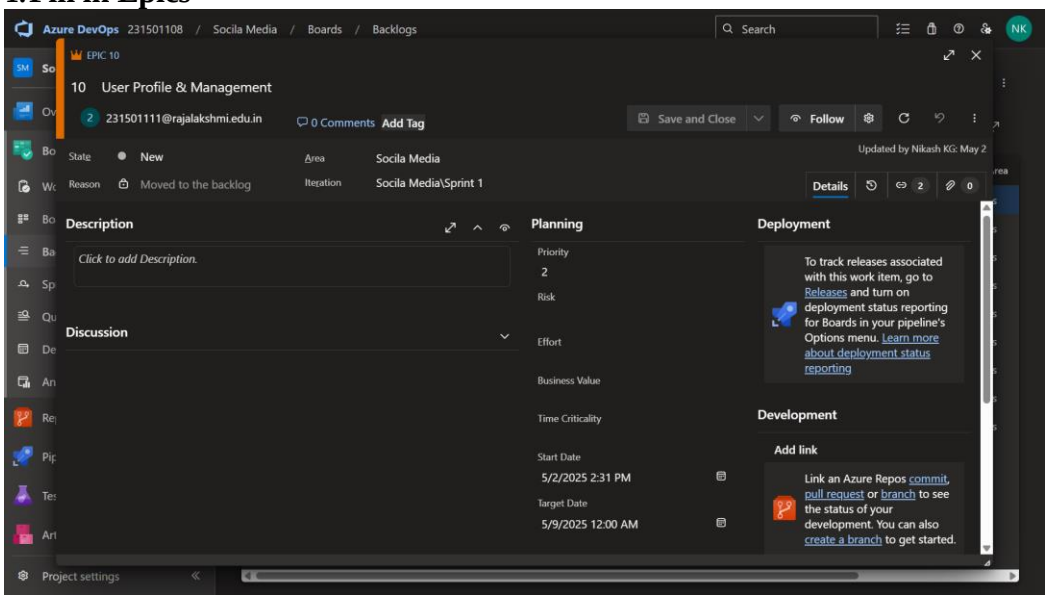
Aim:

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

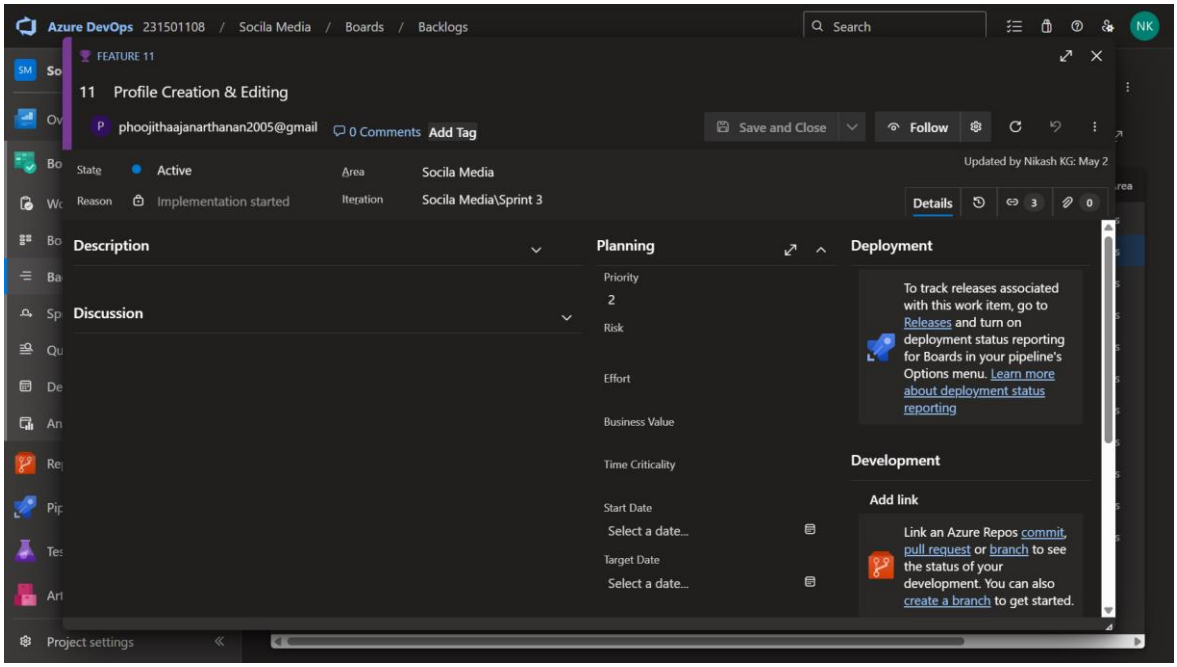
Create Epic, Features, User Stories, Task



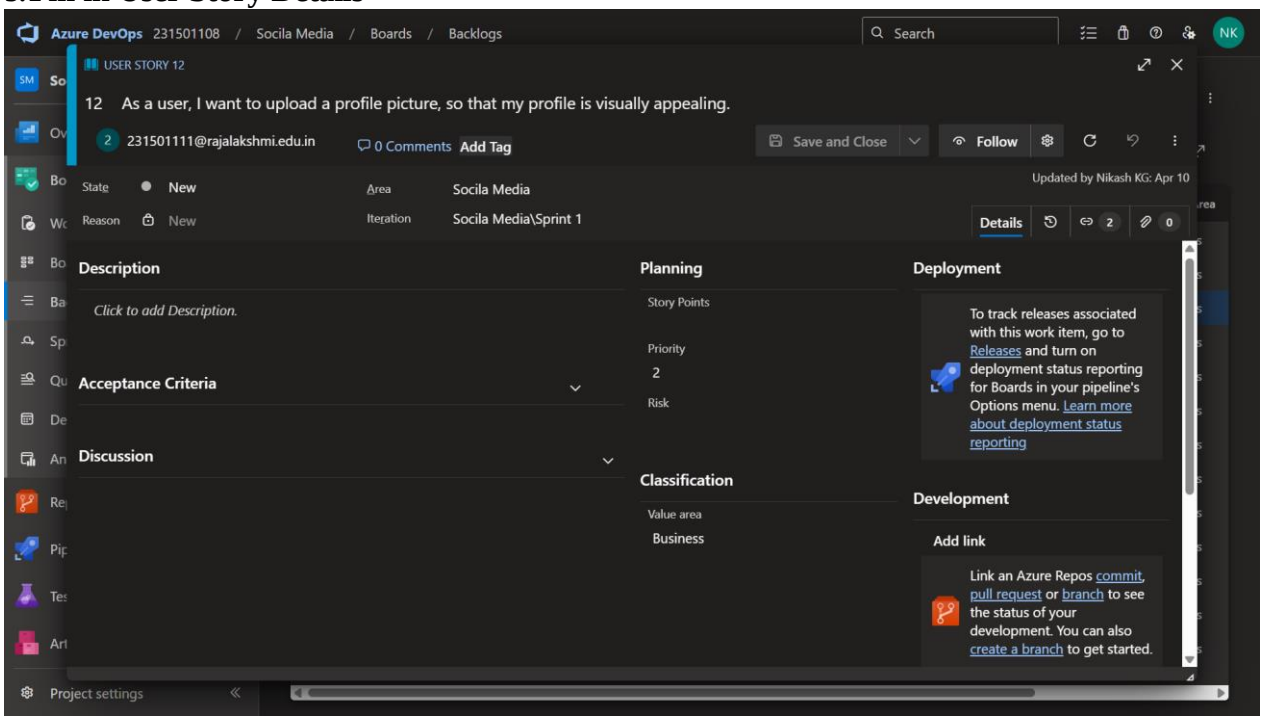
1. Fill in Epics



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.

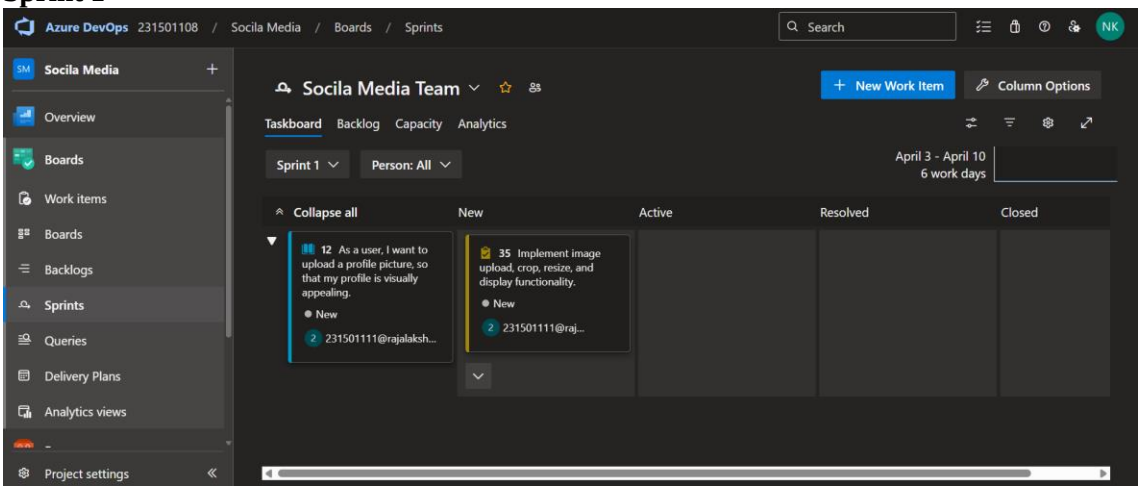
EXP NO: 4	SPRINT PLANNING

Aim:

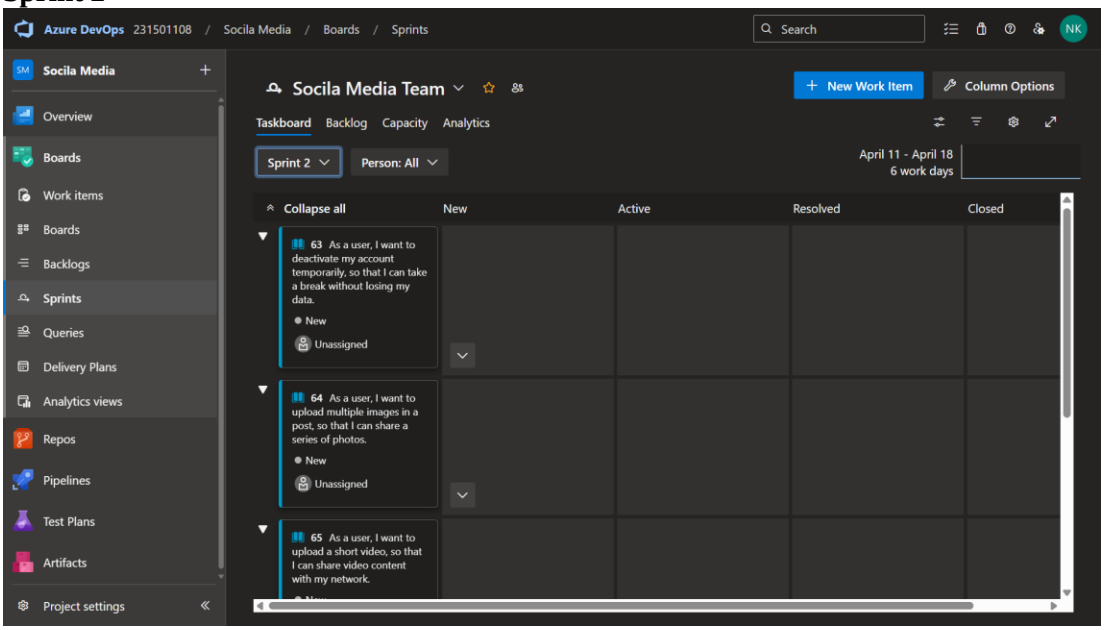
To assign user story to specific sprint for the Music Playlist Batch Creator Project.

Sprint Planning

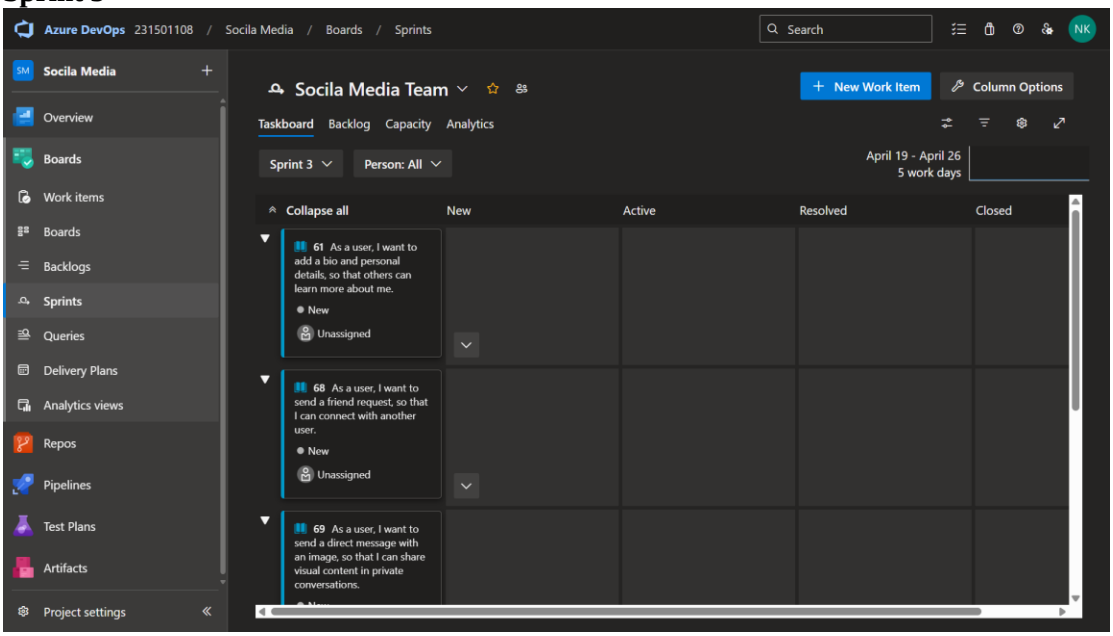
Sprint 1



Sprint 2



Sprint 3



Result:

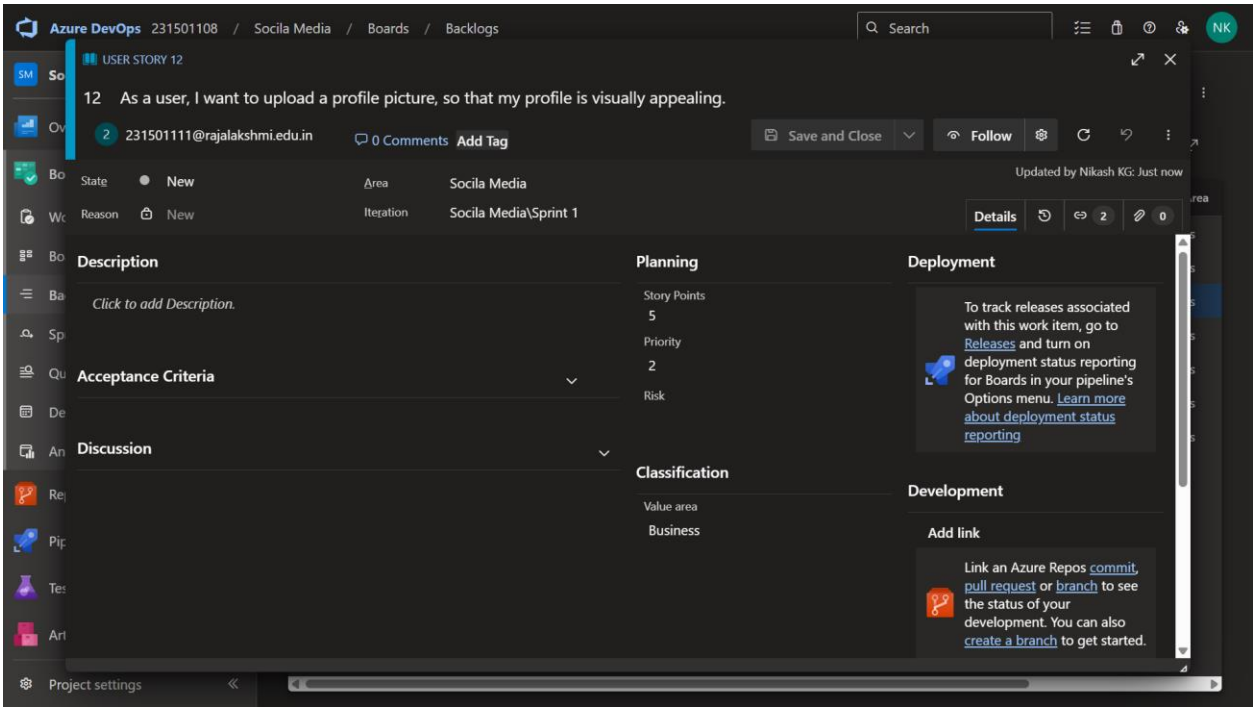
The Sprints are created for the Music Playlist Batch Creator Project.

EXP NO: 5	POKER ESTIMATION

Aim:

Create Poker Estimation for the user stories - Music Playlist Batch Creator Project.

Poker Estimation



Result:

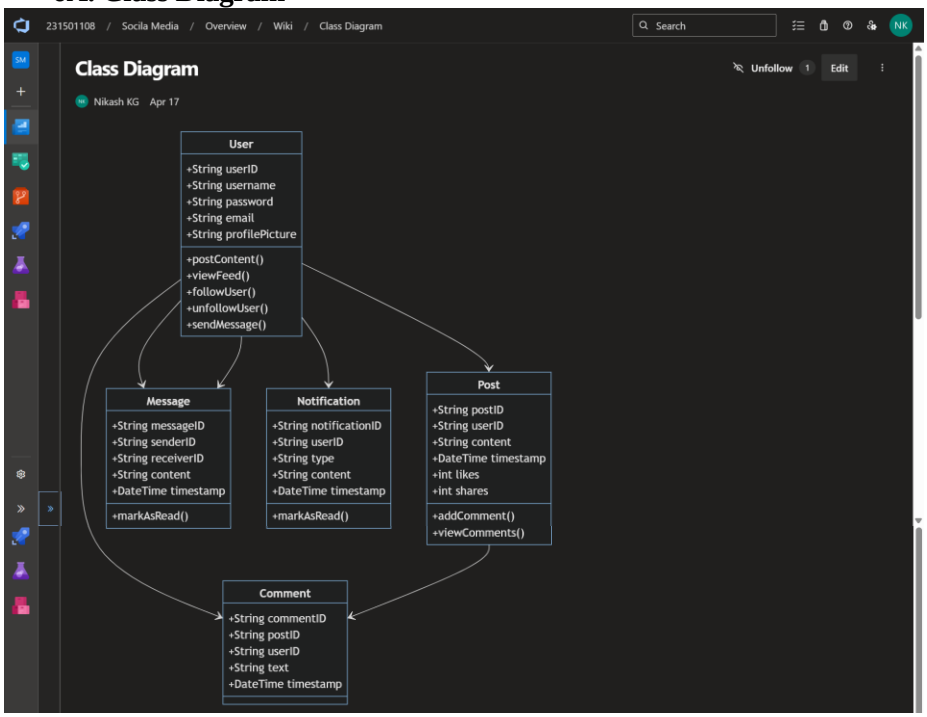
The Estimation/Story Points is created for the project using Poker Estimation.

EXP NO: 6	DESIGNING CLASS AND SEQUENCE DIAGRAMS FOR PROJECT ARCHITECTURE

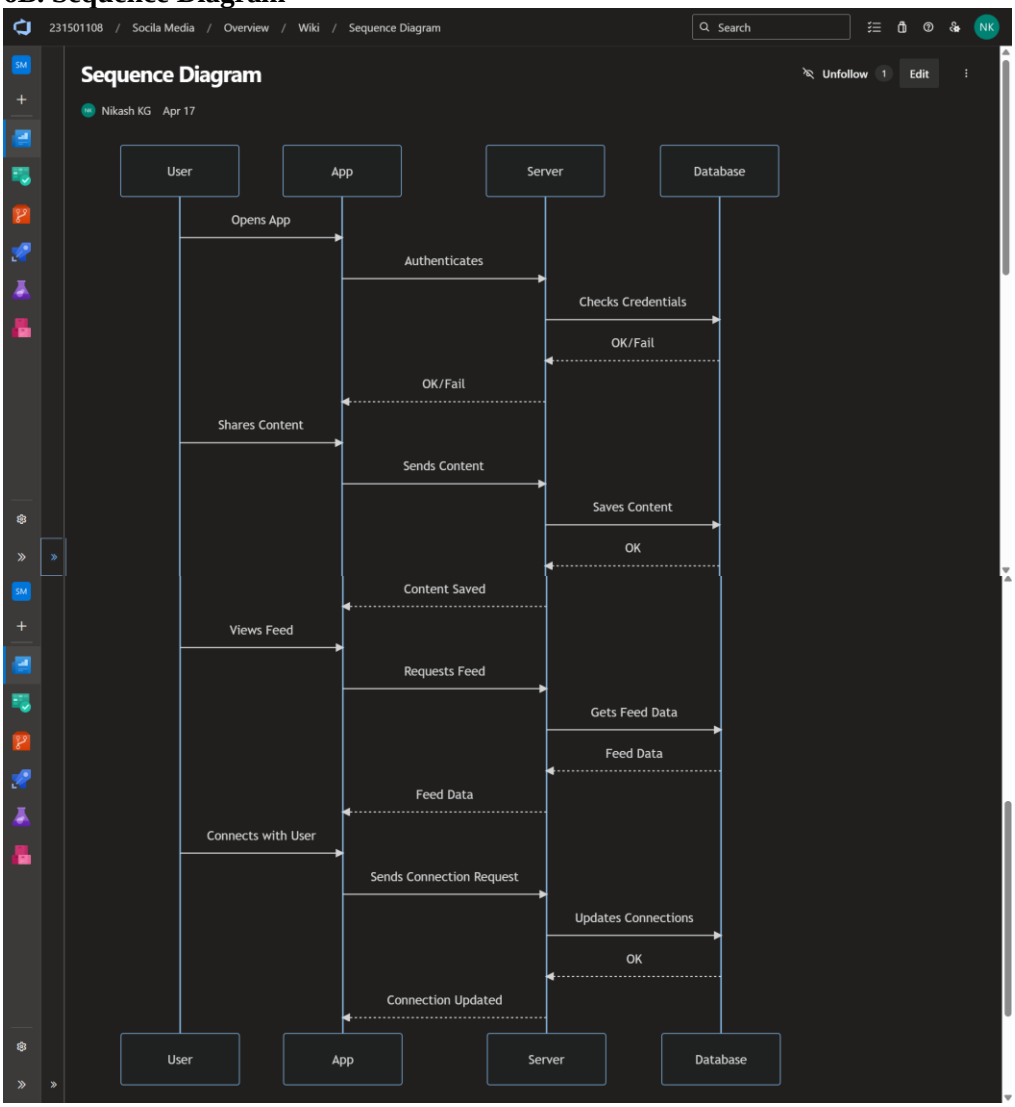
Aim:

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

6A. Class Diagram



6B. Sequence Diagram



Result:

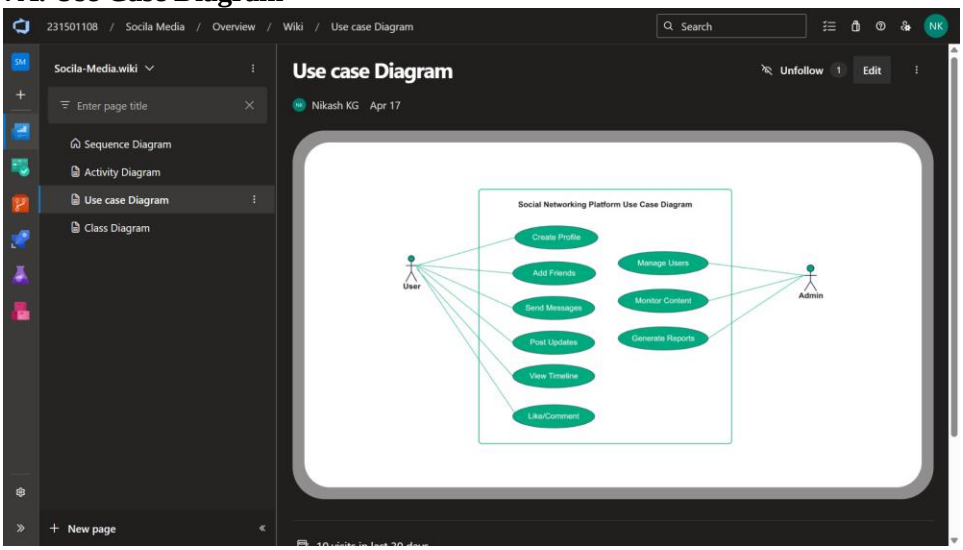
The Class Diagram and Sequence Diagram is designed Successfully for the Music Playlist Batch Creator.

EXP NO: 7	DESIGNING ACTIVITY AND USECASE DIAGRAMS FOR PROJECT ARCHITECTURE

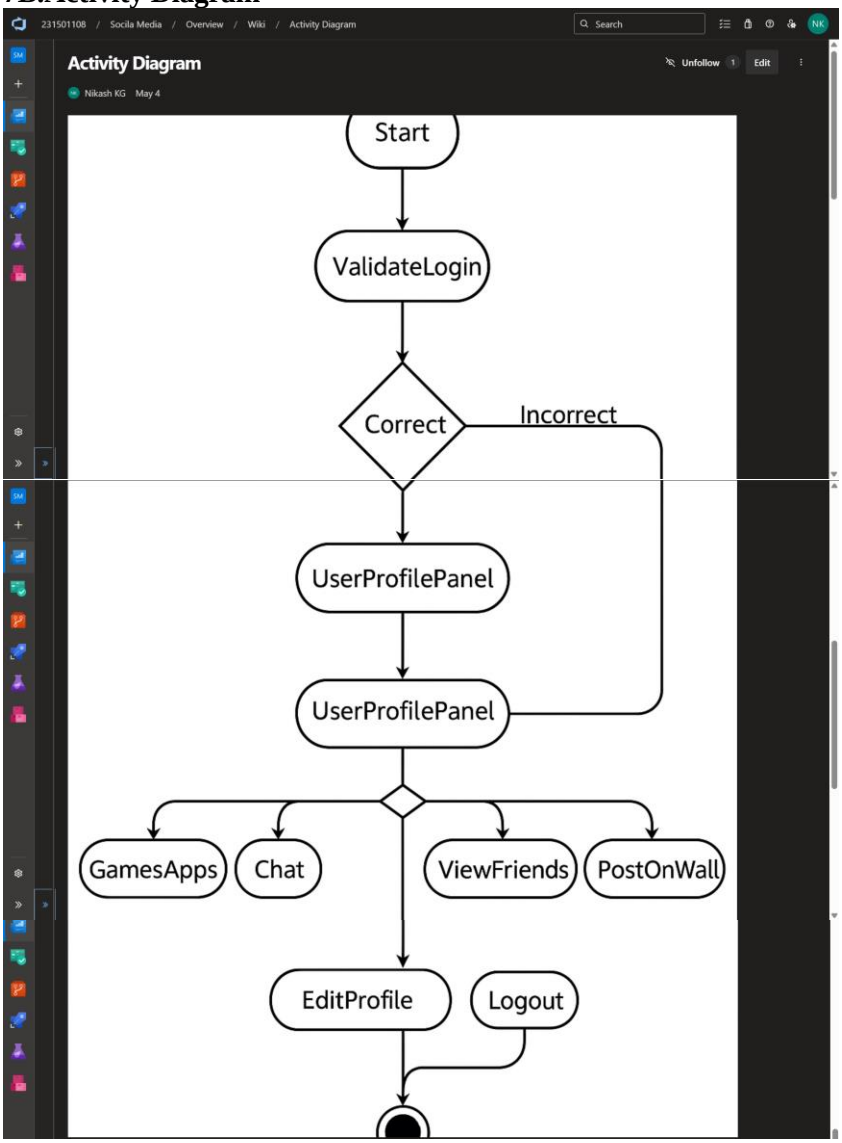
Aim:

To Design an Activity and Use Case Diagram for the given Project.

7A. Use Case Diagram



7B.Activity Diagram



Result:

The Architecture Diagram and ER Diagram is designed Successfully for the Music Playlist Batch 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

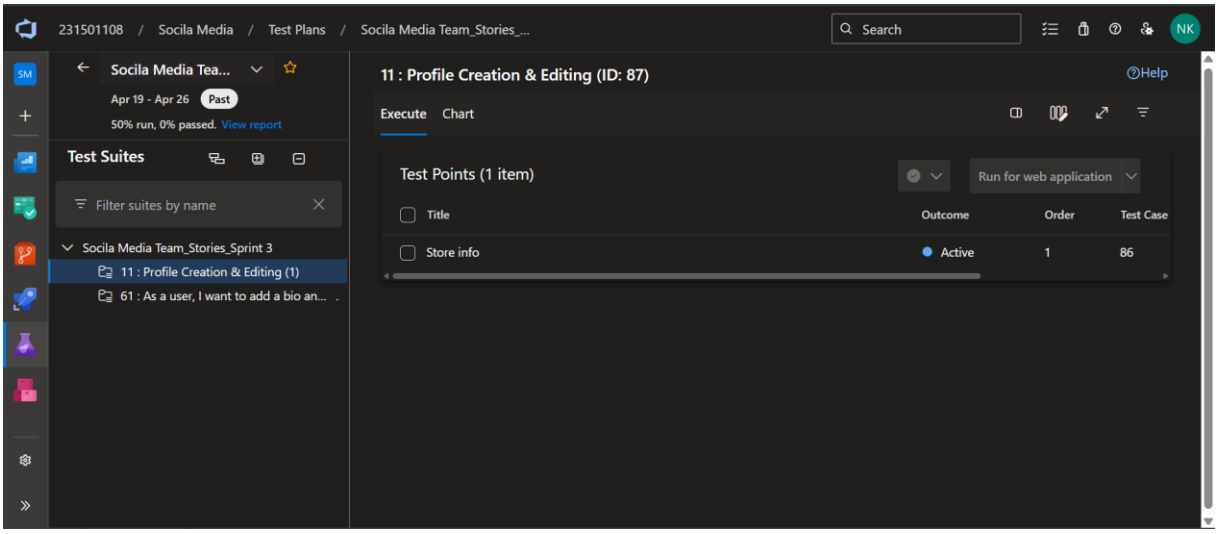
1. **Understand Core Features of the Application**
    - User Signup & Login
    - Viewing and Managing Playlists
    - Fetching Real-time Metadata
    - Editing playlists (rename, reorder, record)
    - Creating smart audio playlists based on categories (mood, genre, artist, etc.)
2. **Define User Interactions**
    - Each test case simulates a real user behaviour (e.g., logging in, renaming a playlist, adding a song).
3. **Design Happy Path Test Cases**
    - Focused on validating that all features function as expected under normal conditions.
    - Example: User logs in successfully, adds item to playlist, or creates a category-based playlist.
4. **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.
5. **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.
6. **Use Clear Naming and IDs**
    - Test cases are named clearly (e.g., TC01 – Successful Login, TC10 – Save Playlist Fails).
    - Helps in quick identification and linking to user stories or features.
7. **Separate Test Suites**
    - Grouped test cases based on functionality (e.g., Login, Playlist Editing, Recommendation System).

- Improves organization and test execution flow in Azure DevOps.

8. Prioritize and Review

- Critical user actions are marked high-priority.
- Reviewed for completeness and traceability against feature requirements.

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.
Music Playlist Batch Creator – Test Plans

USER STORIES

- As a user, I want to sign up and log in securely so that I can access my playlists (ID: 79).
- As a user, I need to see my playlist in one place (ID: 76).
- As a user, I should be able to create an audio playlist as needed (ID: 73).
- As a user, I should be able to rename, record, and change the playlist (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 Suit: TS02 - View Playlists (ID: 87)

1. **TC05 – View Playlist Page**
    - **Action:**
        - Log in successfully.
        - Navigate to "My Playlists" section.
    - **Expected Results:**
        - All created playlists are displayed clearly.
    - **Type:** Happy Path
2. TC06 – Playlist Loading Failure
    - **Action:**
        - Disconnect from the internet.
        - Navigate to "My Playlists".
    - **Expected Results:**
        - Network is offline.
        - Error message "Unable to load playlists" is shown.
    - **Type:** Error Path

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

1. **TC07 – Real-Time Metadata Display**
    - **Action:**
        - Play a song.
        - Observe the metadata panel.
    - **Expected Results:**
        - Metadata (title, artist, album, duration) is displayed and updates in real time.
    - **Type:** Happy Path
2. TC08 – Metadata Not Updating
    - **Action:**

        - Play a different song.
        - Observe the metadata panel.
    - **Expected Results:**
        - Metadata remains static or shows default/fallback message.
    - **Type:** Error Path

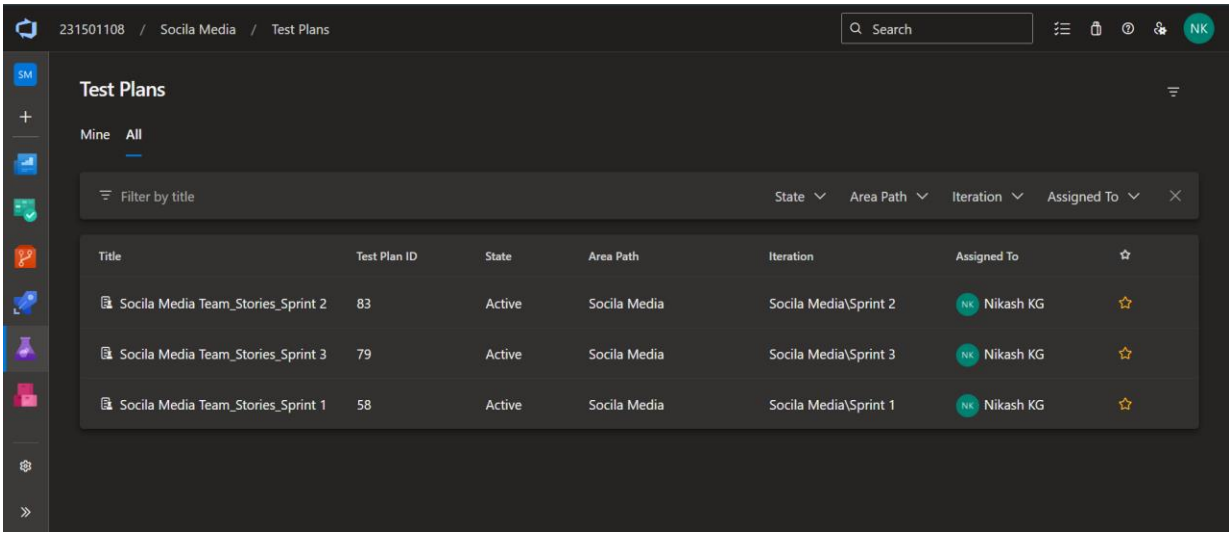
Test Suit: TS04 - Playlist Editing (ID: 89)

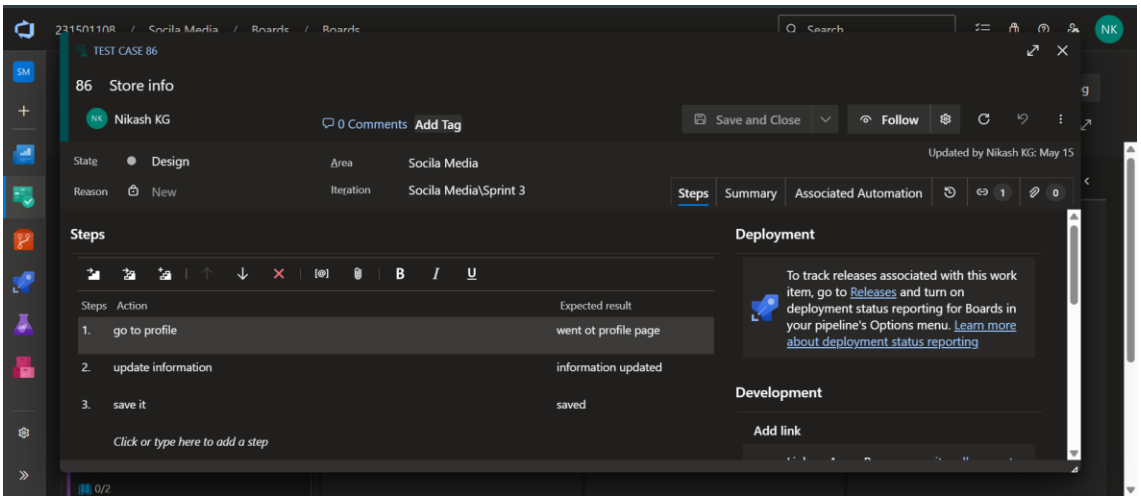
1. **TC09 – Rename Playlist Successfully**
    - **Action:**
        - Navigate to "My Playlists".
        - Click "Rename" next to a playlist.
        - Enter a new name and click "Save".
    - **Expected Results:**
        - Playlist name updates successfully.
    - **Type:** Happy Path
2. **TC10 – Rename with Blank Name**
    - **Action:**
        - Click "Rename" on a playlist.
        - Leave the field blank.
        - Click "Save".
    - **Expected Results:**
        - Error message "Playlist name cannot be empty" is shown.
    - **Type:** Error Path
3. **TC11 – Change Playlist Order**
    - **Action:**
        - Open a playlist.
        - Drag and drop songs to reorder.
        - Click "Save".
    - **Expected Results:**
        - Playlist order is updated and saved.
    - **Type:** Happy Path
4. **TC12 – Change Playlist Order Fails**
    - **Action:**
        - Login and go to “My Playlists”.
        - Select a playlist.
        - Go offline or simulate server error.
        - Reorder songs and click “Save Order”.
    - **Expected Results:**
        - Error message: "Failed to update order. Please check your connection".
    - **Type:** Error Path

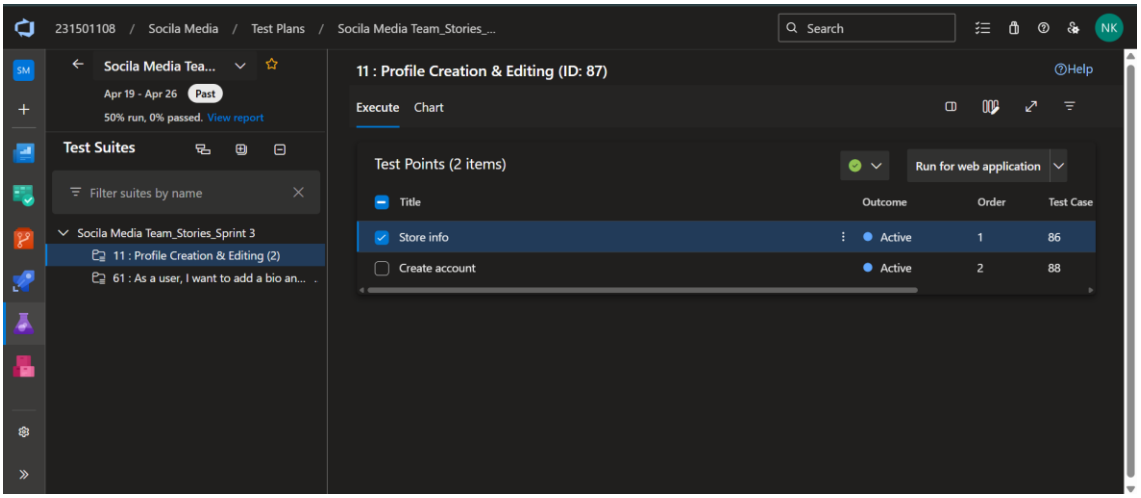
Test Suit: TS05 - Smart Playlist Creation (ID: 90)

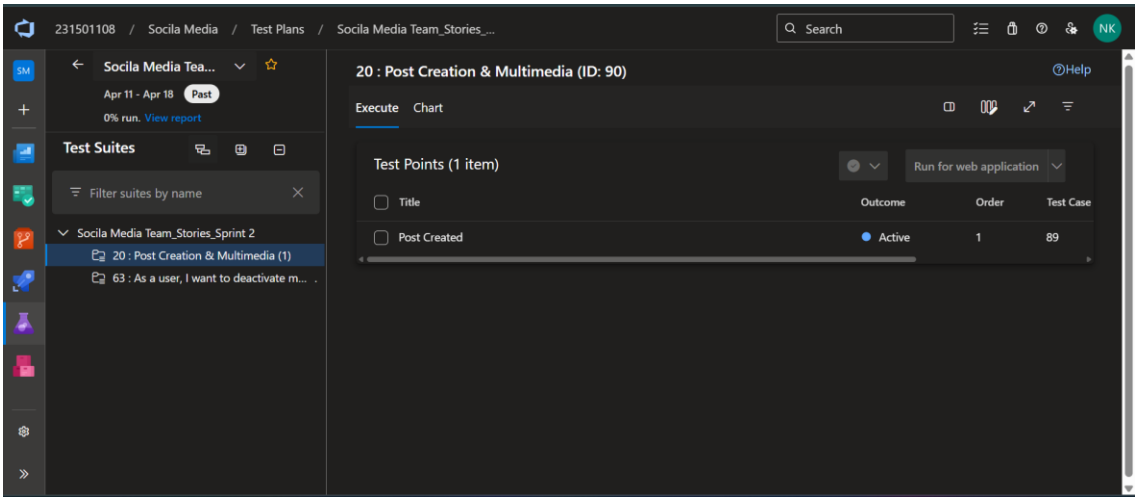
1. **TC13 – Generate Playlist Based on Various Categories**
    - **Action:**
        - Login with valid credentials.
        - Click on "Generate Playlist".
        - Select categories.
        - Click "Generate Playlist".
    - **Expected Results:**
        - Playlist is generated based on selected mood and categories.
    - **Type:** Happy Path
2. **TC14 – Fail to Generate Playlist Due to Missing Category Selection or Invalid Input**
    - **Action:**
        - Login with valid credentials.
        - Click on "Generate Playlist".
        - Select categories.
        - Click "Generate Playlist".
    - **Expected Results:**
        - Error message: "Please select at least one valid category" or "No recommendations found for the selected filters".
    - **Type:** Error Path

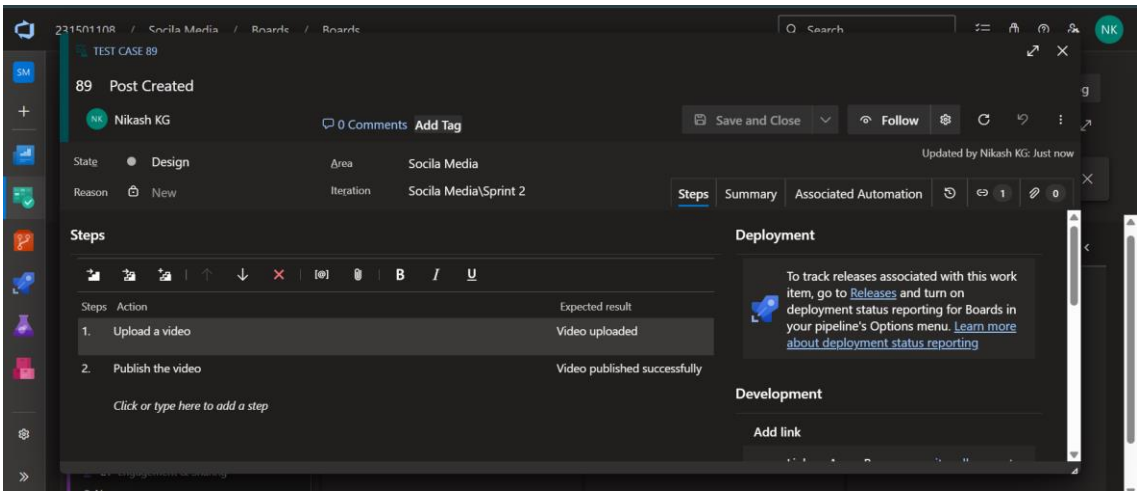
Test Cases



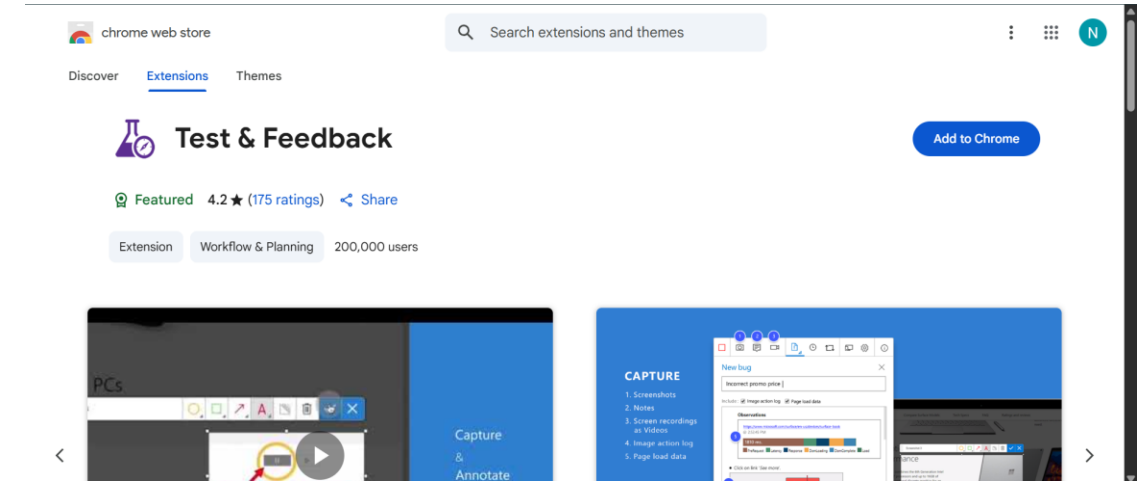

TEST CASE 86
86 Store info
Nikash KG
0 Comments Add Tag
Save and Close
Follow
State Design
Reason New
Area Socila Media
Iteration Socila Media\Sprint 3
Updated by Nikash KG: May 15
Steps Summary Associated Automation
Steps
Steps Action Expected result
1. go to profile went ot profile page
2. update information information updated
3. save it saved
Click or type here to add a step
Deployment
To track releases associated with this work item, go to Releases and turn on deployment status reporting for Boards in your pipeline's Options menu. Learn more about deployment status reporting
Development
Add link

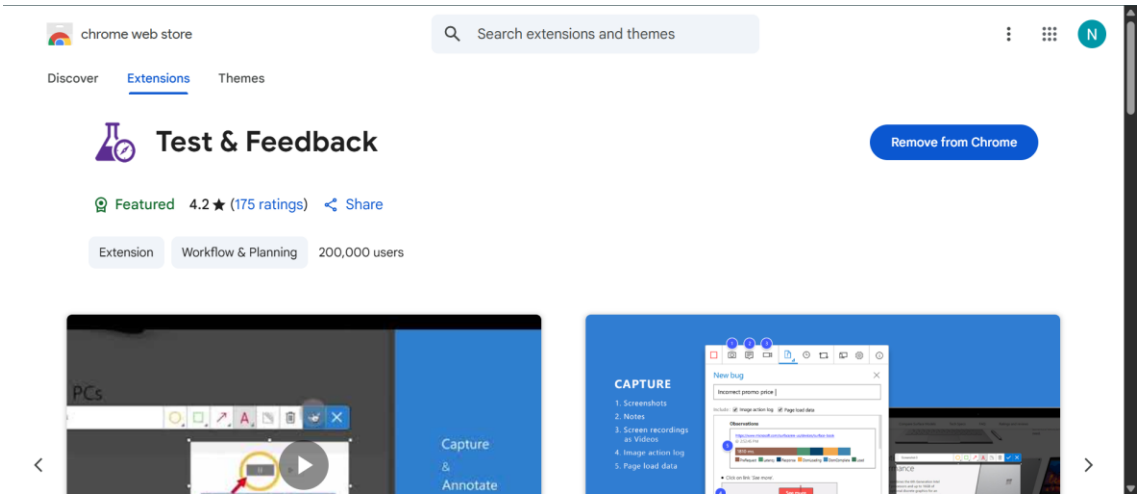

231501108 / Socila Media / Test Plans / Socila Media Team_Stories_...
Search
Socila Media Tea...
Apr 19 - Apr 26 Past
50% run, 0% passed. View report
Test Suites
Filter suites by name
Socila Media Team_Stories_Sprint 3
11 : Profile Creation & Editing (2)
61 : As a user, I want to add a bio an...
11 : Profile Creation & Editing (ID: 87)
Help
Execute Chart
Test Points (2 items)
Run for web application
Title Outcome Order Test Case
Store info Active 1 86
Create account Active 2 88


231501108 / Socila Media / Test Plans / Socila Media Team_Stories_...
Search
Socila Media Tea...
Apr 11 - Apr 18 Past
0% run. View report
Test Suites
Filter suites by name
Socila Media Team_Stories_Sprint 2
20 : Post Creation & Multimedia (1)
63 : As a user, I want to deactivate m...
20 : Post Creation & Multimedia (ID: 90)
Help
Execute Chart
Test Points (1 item)
Run for web application
Title Outcome Order Test Case
Post Created Active 1 89

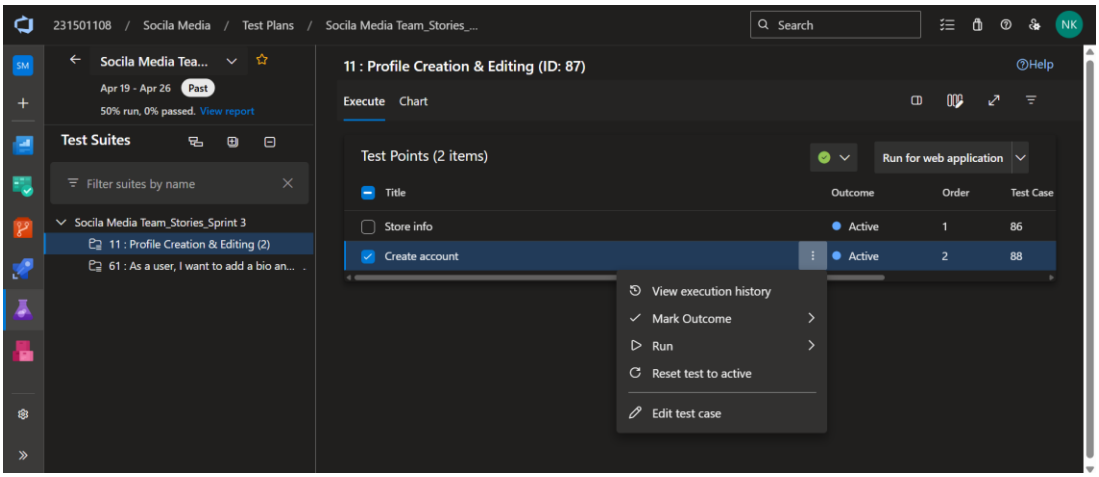


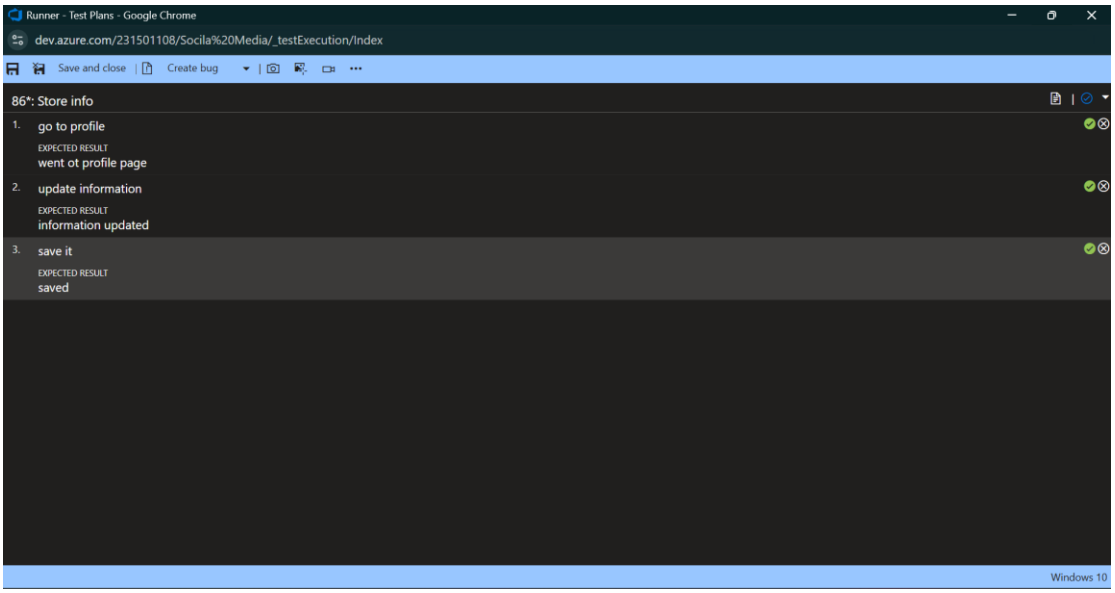
3. Installation of test



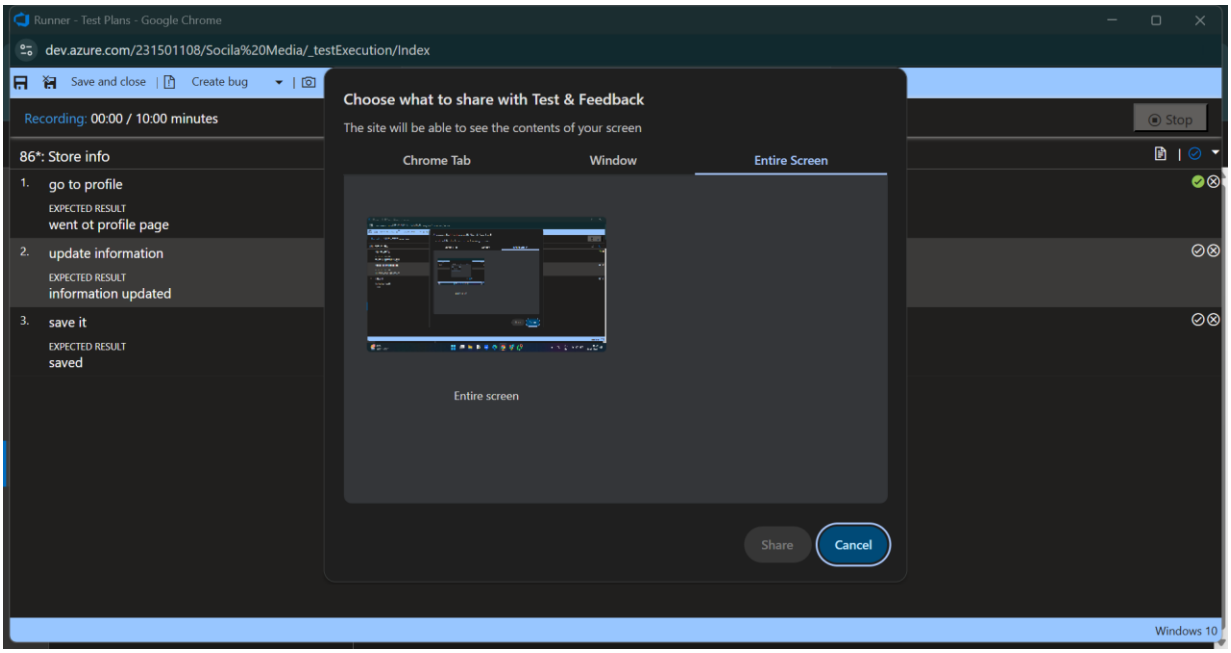


4. Running the test cases

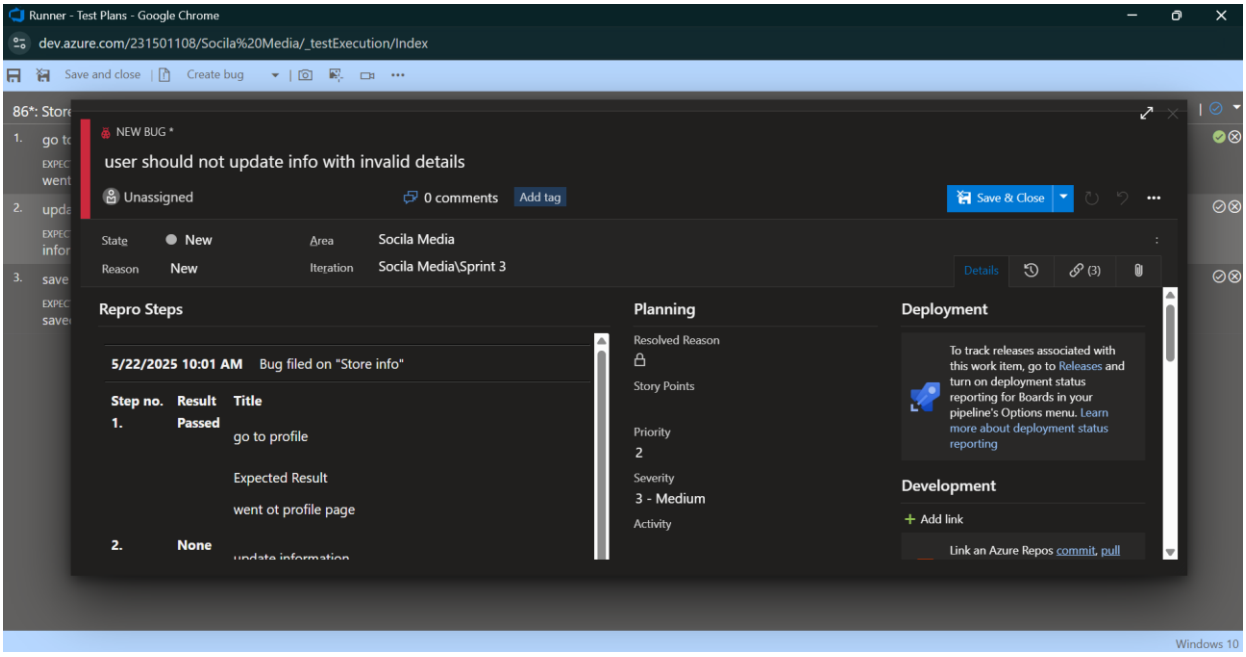


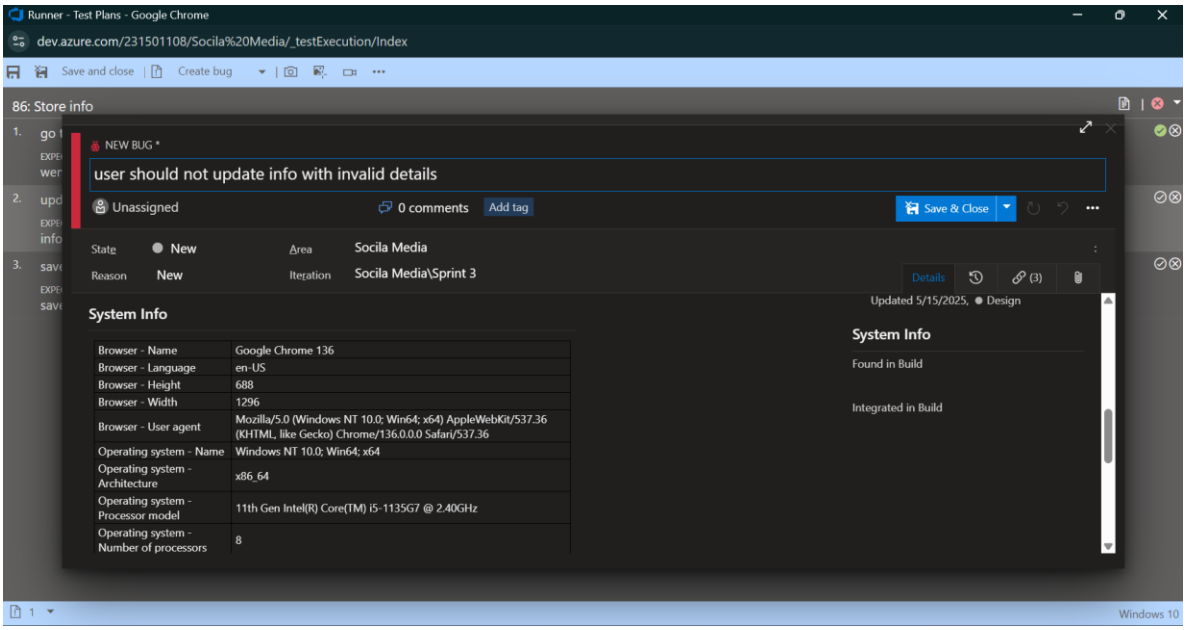


5. Recording the test case

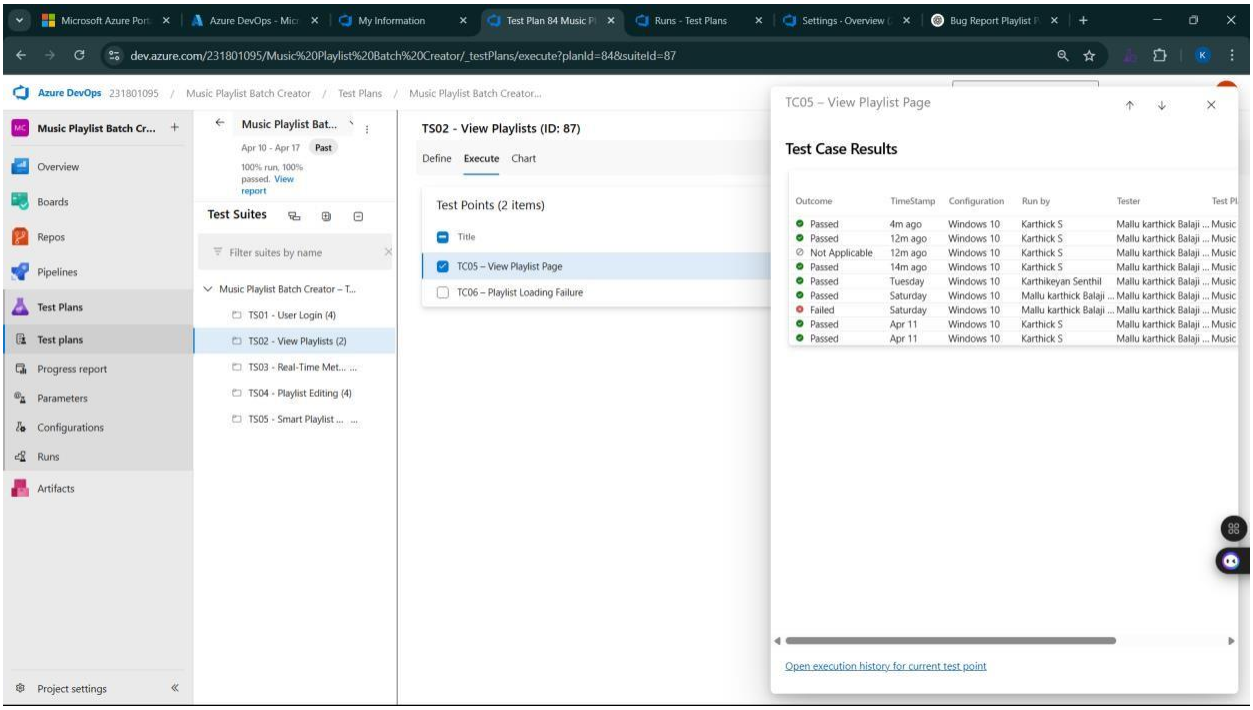


6. Creating the bug

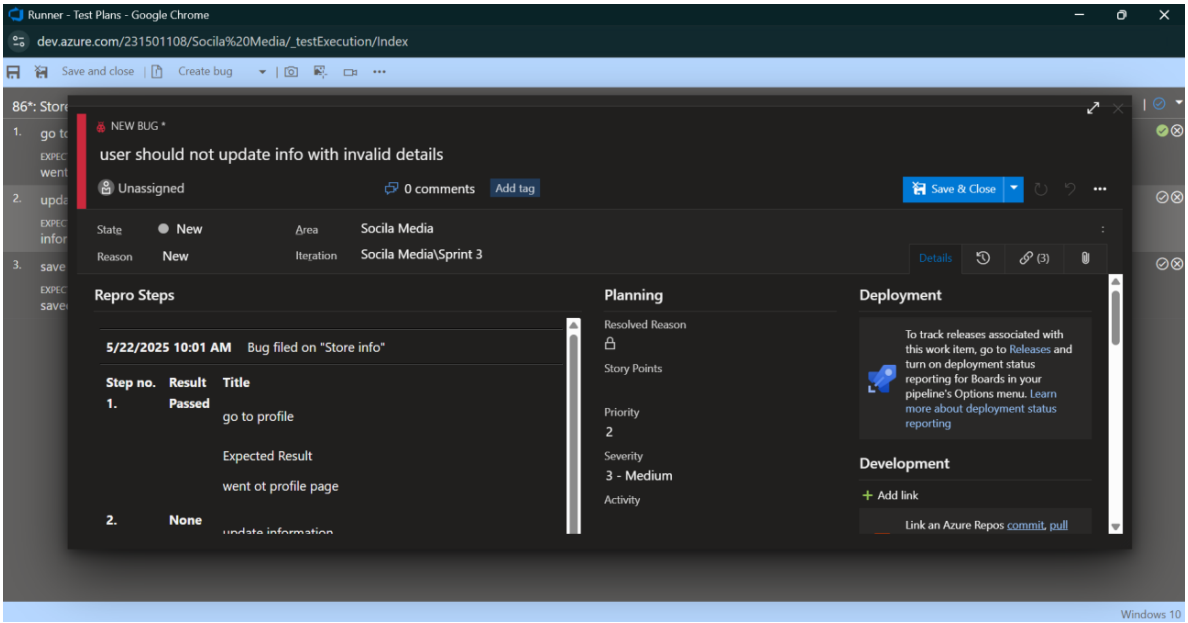



Runner - Test Plans - Google Chrome
dev.azure.com/231501108/Socila%20Media/_testExecution/Index
Save and close
Create bug
86: Store info
NEW BUG *
user should not update info with invalid details
Unassigned
0 comments
Add tag
Save & Close
State
New
Area
Socila Media
Reason
New
Iteration
Socila Media\Sprint 3
Details
(3)
Updated 5/15/2025, Design
System Info
Browser - Name
Google Chrome 136
Browser - Language
en-US
Browser - Height
688
Browser - Width
1296
Browser - User agent
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36
Operating system - Name
Windows NT 10.0; Win64; x64
Operating system - Architecture
x86_64
Operating system - Processor model
11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz
Operating system - Number of processors
8
System Info
Found in Build
Integrated in Build
Windows 10

7. Test case results

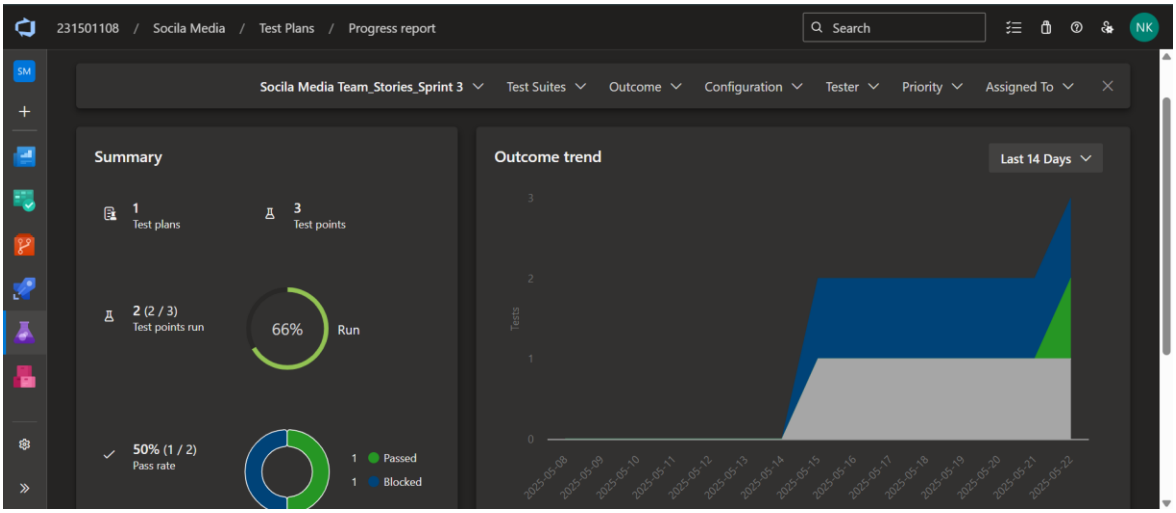


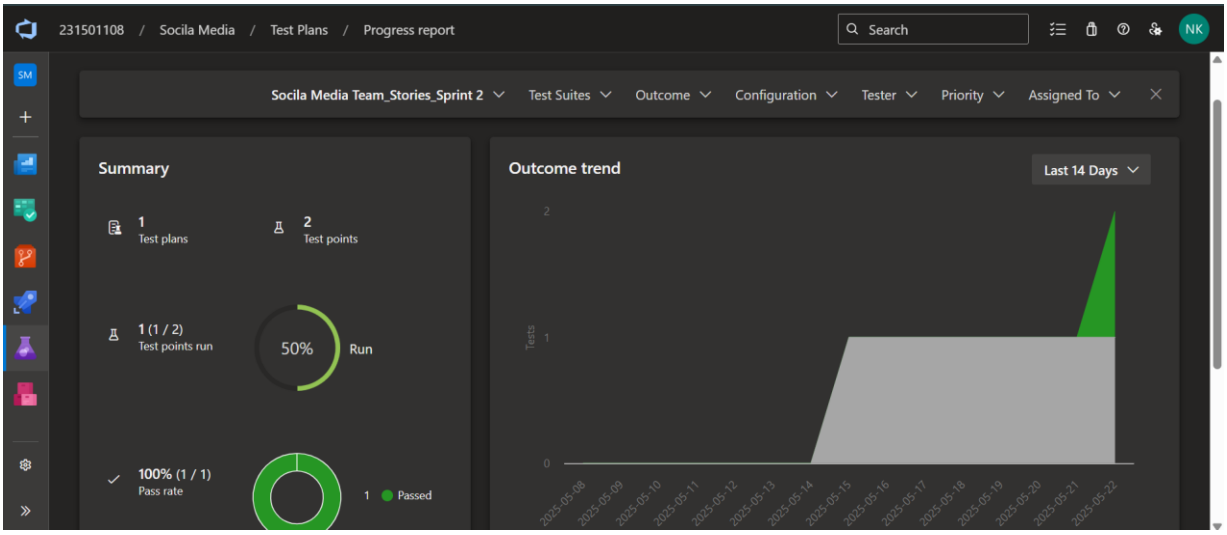
8. Test report summary



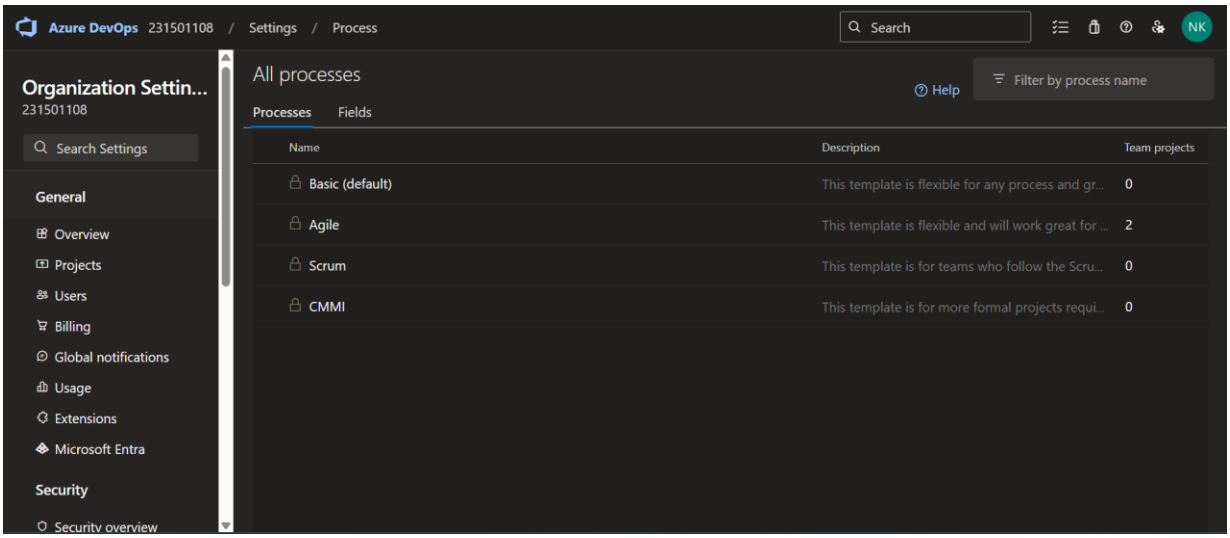
- Assigning bug to the developer and changing state

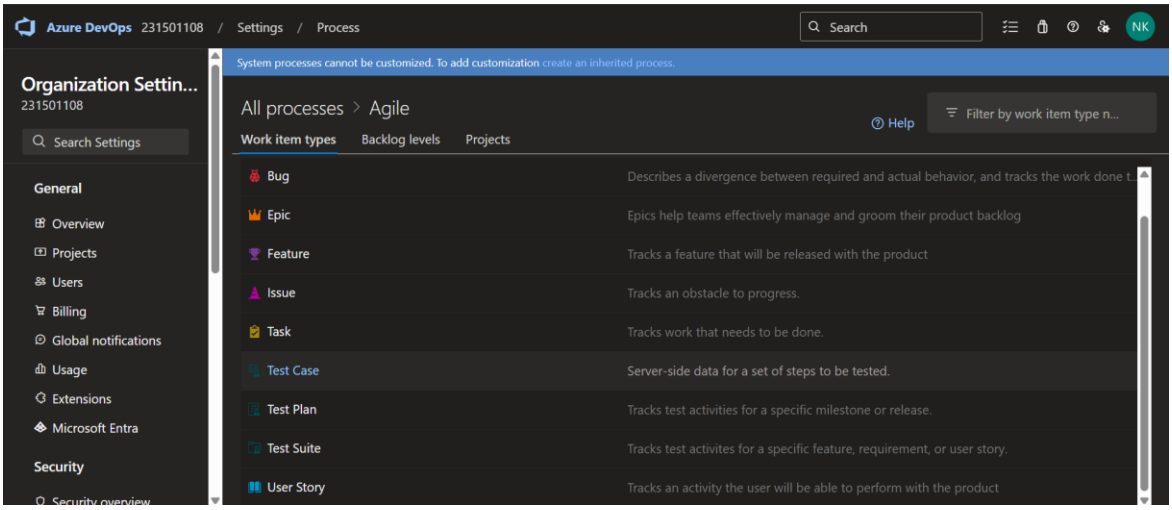
9. Progress report





10. 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	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, select your organization, and open the project where your Student Management System code resides.

2. 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.

3. 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.

4. 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.

5. Define Build Stage (CI - Continuous Integration) from YAML file.

6. Install dependencies (e.g., npm install, dotnet restore).

7. Build the application (dotnet build, npm run build).

8. Run unit tests (dotnet test, npm test).

9. Publish build artifacts to be used in the release stage.

10. 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.

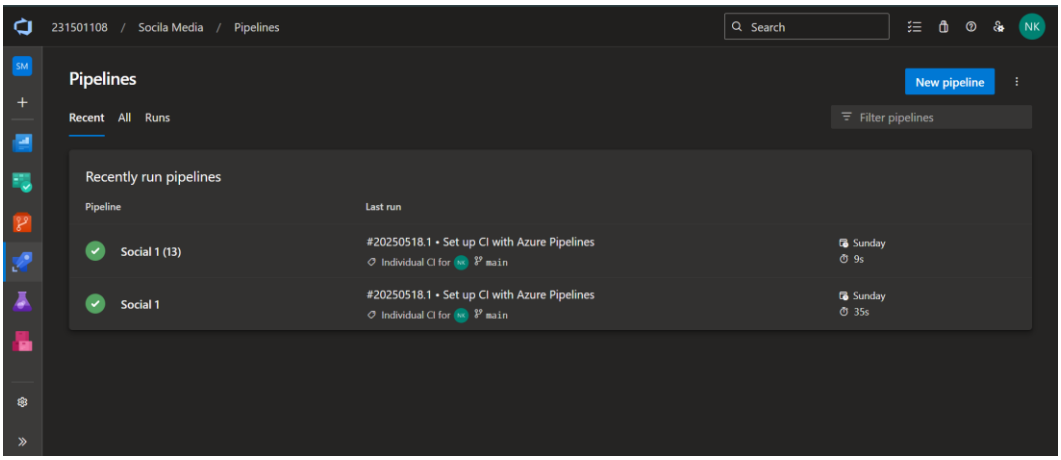
11. Configure Continuous Deployment (CD)
    Navigate to the Releases tab under Pipelines and click “New Release Pipeline”. Add an Artifact (from the build stage) and create a new Stage (e.g., Development, Production).

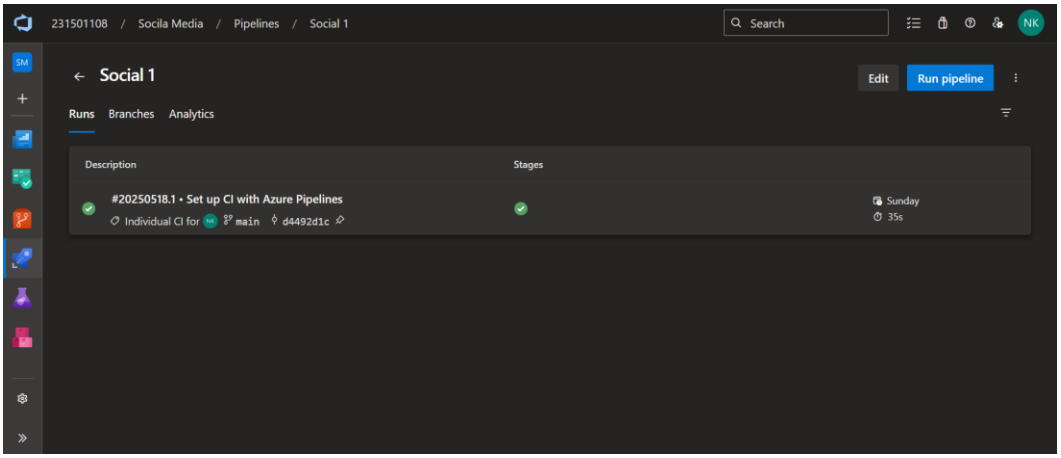
12. Configure the CD stage with deployment tasks such as deploying to Azure App Service, running database migrations or scripts, and restarting services using the Azure App Service Deploy task linked to your subscription and app details.

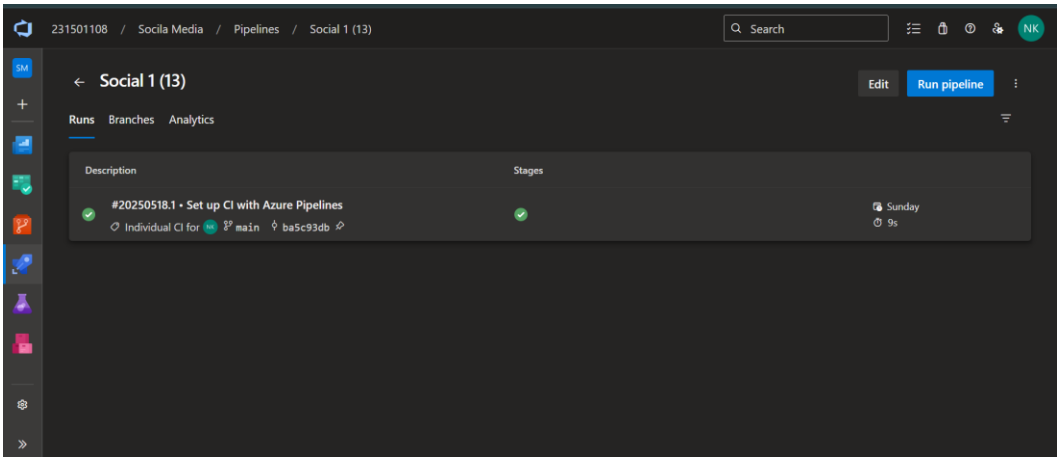
13. Set Triggers and Approvals
    Enable continuous deployment trigger so the release pipeline runs automatically after a successful build. For production environments, configure pre-deployment approvals to ensure manual verification before release.

14. 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.

15. 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

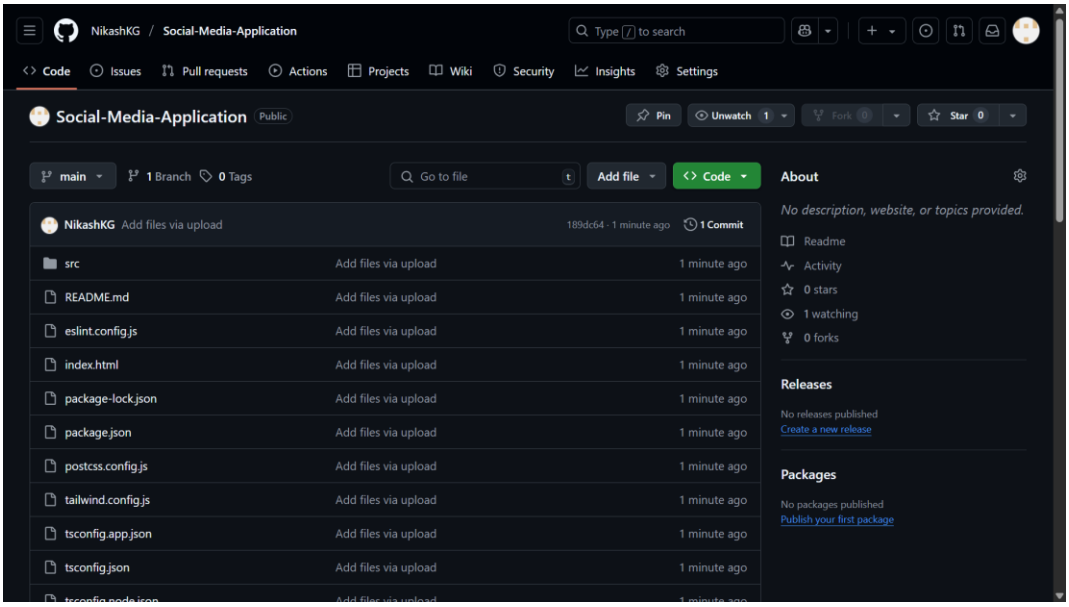
Thus, the pipelines for the given project **"Online Quiz System"** has been executed successfully.

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 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.