

RAJALAKSHMI ENGINEERING COLLEGE
RAJALAKSHMI NAGAR, THANDALAM – 602 105



RAJALAKSHMI
ENGINEERING COLLEGE

CS23432
SOFTWARE CONSTRUCTION

Laboratory Record Note Book

Name :

Year / Branch / Section :

Register No. :

Semester :

Academic Year :



RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)
RAJALAKSHMI NAGAR, THANDALAM – 602 105

BONAFIDE CERTIFICATE

NAME _____ REGISTER NO. _____

ACADEMIC YEAR 2024-25 **SEMESTER- IV** **BRANCH:** B. Tech Information
Technology [AD/AE]. This Certification is the Bonafide record of work done by the
above student in the **CS23432- Software Construction** Laboratory during the
year 2024-2025.

Signature of Faculty -in – Charge

Submitted for the Practical Examination held on _____

Internal Examiner

External Examiner

LAB PLAN
CS23432-SOFTWARE CONSTRUCTION LAB

Ex No	Date	Topic	Page No	Sign
1	21/01/2025	Study of Azure DevOps		
2	28/01/2025	Problem Statement		
3	04/02/2025	Agile Planning		
4	18/02/2025	Create User stories with Acceptance Criteria		
5	25/02/2025	Designing Sequence Diagrams using Azure DevOps-WIKI		
6	04/03/2025	Designing Class Diagram using Azure DevOps-WIKI		
7	11/03/2025	Designing Use case Diagram using Azure DevOps-WIKI		
8	18/03/2025	Designing Activity Diagrams using Azure DevOps-WIKI		
9	25/03/2025	Designing Architecture Diagram Using Star UML		
10	01/04/2025	Design User Interface		
11	08/04/2025	Implementation – Design a Web Page based on Scrum Methodology		
12	15/04/2025	Testing-Test Plan, Test Case and Load Testing		

EX NO: 1

DATE:21/01/2025

Study of Azure DevOps

AIM:

To study how to create an agile project in Azure DevOps environment.

STUDY:

Azure DevOps is a cloud-based platform by Microsoft that provides tools for DevOps practices, including CI/CD pipelines, version control, agile planning, testing, and monitoring. It supports teams in automating software development and deployment.

1. Understanding Azure DevOps

Azure DevOps consists of five key services:

1.1 Azure Repos (Version Control)

Supports Git repositories and Team Foundation Version Control (TFVC). Provides features like branching, pull requests, and code reviews.

1.2 Azure Pipelines (CI/CD)

Automates build, test, and deployment processes.

Supports multi-platform builds (Windows, Linux, macOS).

Works with Docker, Kubernetes, Terraform, and cloud providers (Azure, AWS, GCP).

1.3 Azure Boards (Agile Project Management)

Manages work using Kanban boards, Scrum boards, and dashboards. Tracks user stories, tasks, bugs, sprints, and releases.

1.4 Azure Test Plans (Testing)

Provides manual, exploratory, and automated testing. Supports test case management and tracking.

1.5 Azure Artifacts (Package Management)

Stores and manages NuGet, npm, Maven, and Python packages. Enables versioning and secure access to dependencies.

Step 1: Create an Azure DevOps

Account Visit Azure DevOps.

Sign in with a Microsoft Account.

Create an Organization and a

Project. Step 2: Set Up a Repository

(Azure Repos)

Navigate to Repos.

Choose Git or TFVC for version

control. Clone the repository and

push your code.

Step 3: Configure a CI/CD Pipeline (Azure Pipelines)

Go to Pipelines → New Pipeline.

Select a source code repository (Azure Repos, GitHub, etc.).

Define the pipeline using YAML or the Classic

Editor. Run the pipeline to build and deploy the

application.

Step 4: Manage Work with Azure Boards

Navigate to Boards.

Create work items, user stories, and tasks.

Organize sprints and track progress.

Step 5: Implement Testing (Azure Test

Plans) Go to Test Plans.

Create and run test cases

View test results and track bugs.

RESULT:

The study was successfully completed.

EX NO: 2

DATE:28/01/2025

PROBLEM STATEMENT

AIM:

To develop a social network platform that helps people connect, share content, and communicate easily through a seamless digital experience.

PROBLEM STATEMENT:

People often struggle to stay connected and share meaningful content in one place. Existing platforms are cluttered or lack personalization. There is a need for a simple, user social network that allows users to connect, interact, and share content easily based on their interests.

RESULT:

The problem statement was written successfully.

EX NO: 3

DATE:04/02/2025

AGILE PLANNING

AIM:

To prepare an Agile Plan.

THEORY:

Agile planning is a part of the Agile methodology, which is a project management style with an incremental, iterative approach. Instead of using an in-depth plan from the start of the project—which is typically product-related—Agile leaves room for requirement changes throughout and relies on constant feedback from end users.

With Agile planning, a project is broken down into smaller, more manageable tasks with the ultimate goal of having a defined image of a project's vision. Agile planning involves looking at different aspects of a project's tasks and how they'll be achieved, for example:

- Roadmaps to guide a product's release ad schedule
- Sprints to work on one specific group of tasks at a time
- A feedback plan to allow teams to stay flexible and easily adapt to change

User stories, or the tasks in a project, capture user requirements from the end user's perspective Essentially, with Agile planning, a team would decide on a set of user stories to action at any given time, using them as a guide to implement new features or functionalities in a tool. Looking at tasks as user stories is a helpful way to imagine how a customer may use a feature and helps teams prioritize work and focus on delivering value first.

- Steps in Agile planning process
- Define vision

1. Set clear expectations on goals
2. Define and break down the product roadmap
3. Create tasks based on user stories
4. Populate product backlog
5. Plan iterations and estimate effort
6. Conduct daily stand-ups
7. Monitor and adapt

RESULT:

Thus, the Agile plan was completed successfully.

EX NO: 4

DATE:18/02/2025

CREATE USER STORIES

AIM:

To create User Stories

THEORY:

A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer.

EPIC 1: Stories That Disappear

Users should be able to post short updates that disappear after some time.

FEATURE:

1)24-hour temporary posts with viewer tracking.

USER STORY 1:

As a user, I want to share short updates that disappear in 24 hours so that I don't clutter my profile.

ACCEPTANCE CRITERIA:

- 1) Users can post short updates that disappear in 24 hours.
- 2) Stories support text, images, and videos.
- 3) Users can see how many people viewed their story.

USER STORY 2:

As a user, I want to react to stories so that I can express my thoughts.

ACCEPTANCE CRITERIA:

- 1) Users can like or react to stories.
- 2) Story reactions appear in the viewer list.
- 3) Users receive notifications for reactions.

USER STORY 3:

As a user, I want to see who viewed my story so that I know who's interested.

ACCEPTANCE CRITERIA:

- 1) Users can see who viewed their story.

2) Story insights include total views and interactions.

3) Users can hide their story from specific users.

EPIC 2: Easy Post Sharing

Users should be able to share posts quickly and easily.

FEATURE:

1) Quick post creation with text, images, and videos.

USER STORY 1:

As a user, I want to edit my posts so that I can correct mistakes.

ACCEPTANCE CRITERIA:

1) Users can edit a post's text or media after posting.

2) Changes appear in real time without losing engagement.

3) An "Edited" label appears on modified posts

USER STORY 2:

As a user, I want to delete my posts if I change my mind.

ACCEPTANCE CRITERIA:

1) Users can delete posts from their profile and feed.

2) Deleted posts are permanently removed.

3) A confirmation message appears before deletion.

USER STORY 3:

As a user, I want to post text, images, and videos so that I can share my thoughts.

ACCEPTANCE CRITERIA:

1) Users can create posts with text, images, or videos.

2) Posts appear instantly on the user's profile and feed.

3) Users can preview posts before sharing.

EPIC 3: Notification

Users should get alerts about important updates.

FEATURE:

- 1) Alerts for likes, comments, messages, and events.

USER STORY 1:

As a user, I want to get notifications when someone likes my post so that I can see who interacted with me.

ACCEPTANCE CRITERIA:

- 1) Users receive notifications for likes, comments, and shares.
- 2) Notifications appear in the app and as push alerts.
- 3) Users can disable specific types of notifications.

USER STORY 2:

As a user, I want to be notified when someone comments on my post so that I can reply.

ACCEPTANCE CRITERIA:

- 1) Users receive alerts for new messages.
- 2) Clicking the notification opens the chat directly.
- 3) Users can mute message notifications.

USER STORY 3:

As a user, I want to turn off notifications if I don't want them.

ACCEPTANCE CRITERIA:

- 1) Users can turn notifications on or off for specific activities.
- 2) Notification preferences are saved automatically.
- 3) Users can reset notification settings to default.

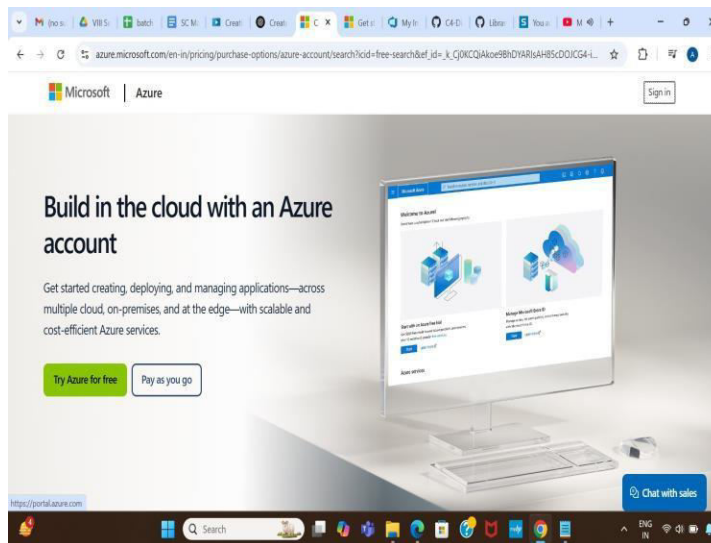
PROCEDURE:

1) Open your web browser and go to the Azure website:

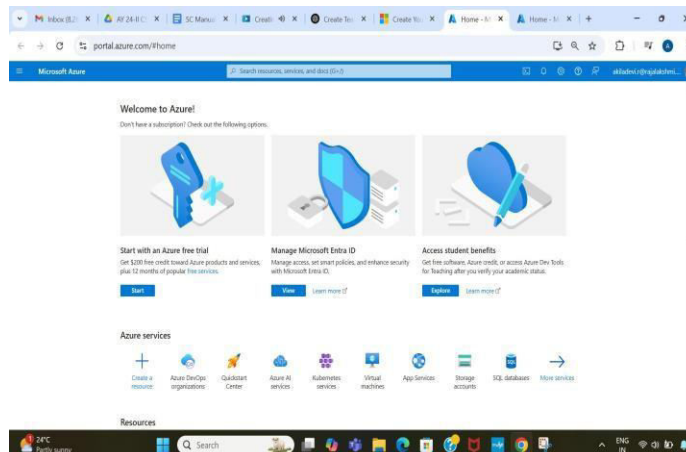
<https://azure.microsoft.com/en-in> Sign in using your Microsoft account credentials. If you don't have an account, you'll need to create one.

2) If you don't have a Microsoft account, you can sign up for

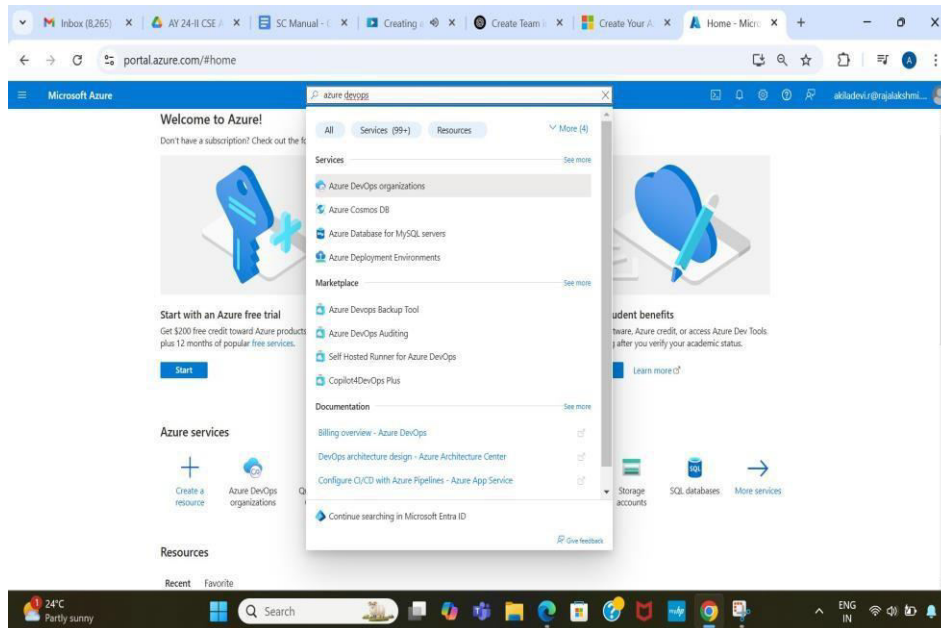
<https://signup.live.com/?lic=1>



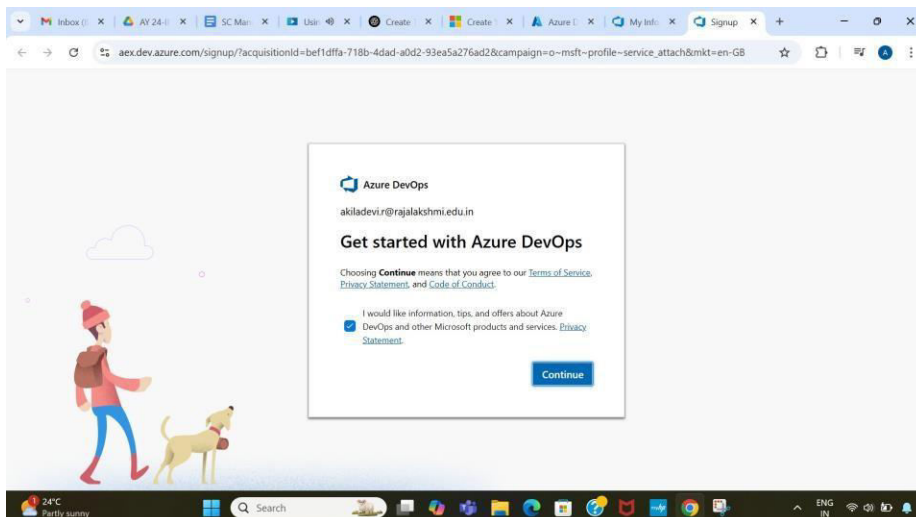
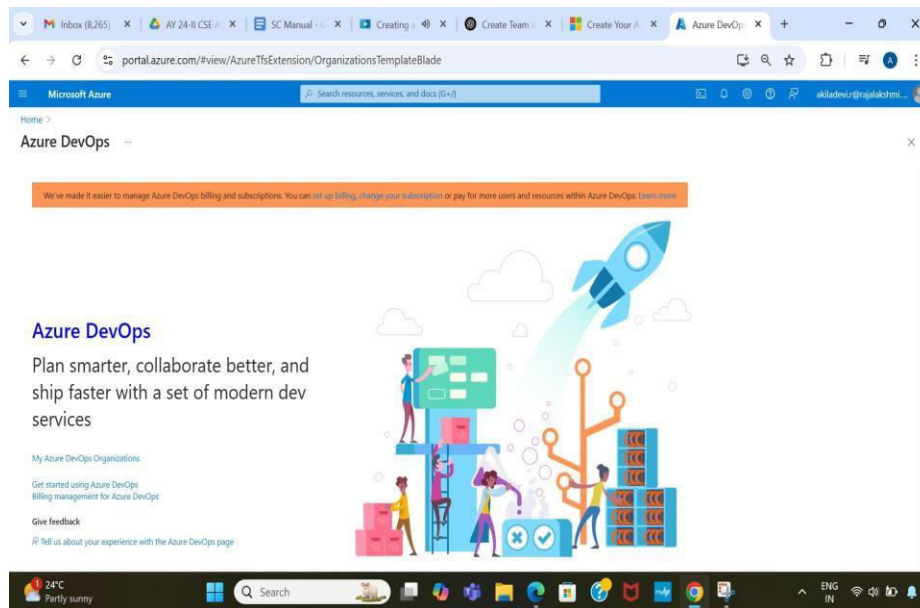
3) Azure home page



4) Open DevOps environment in the Azure platform by typing Azure DevOps Organizations in the search bar.



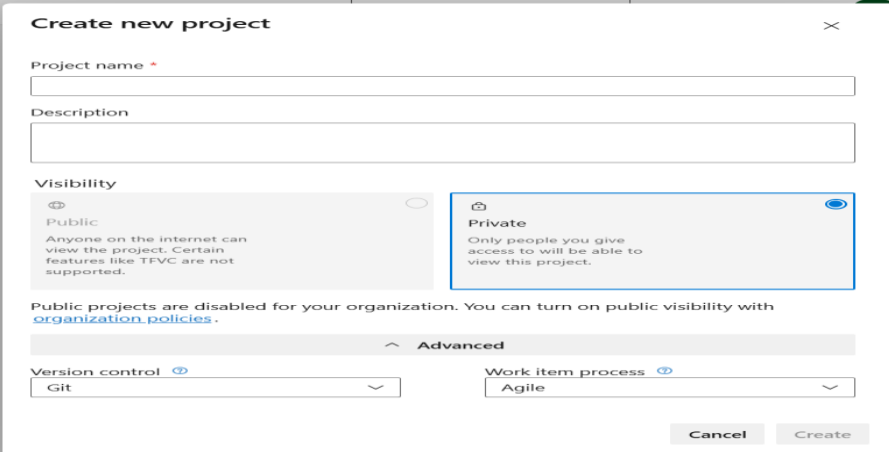
5) Click on the My Azure DevOps Organization link and create an organization and you should be taken to the Azure DevOps Organization Home page.



6) Create the First Project in Your Organization

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.

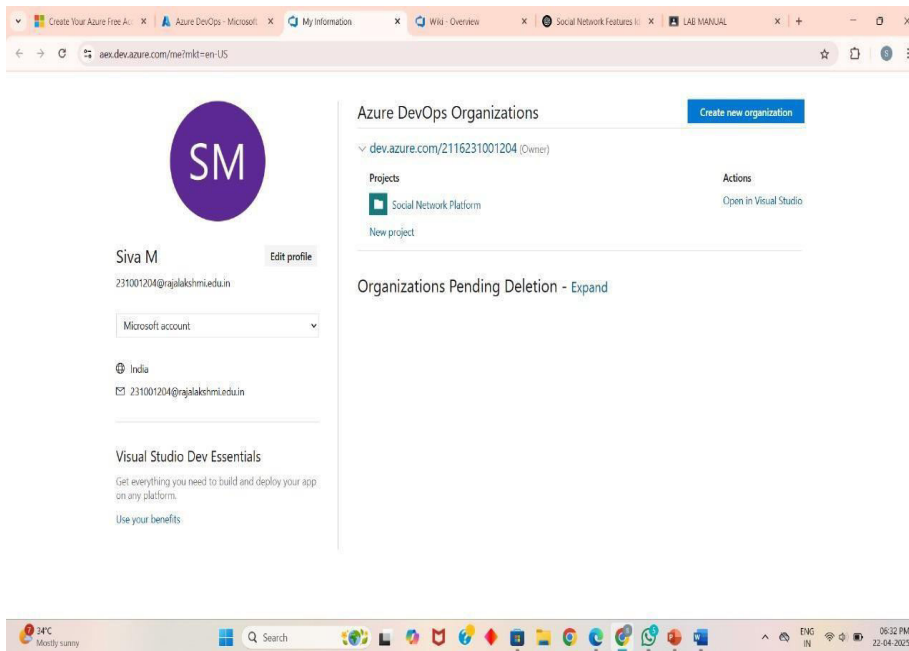
- i. On the organization's **Home page**, click on the **New Project** button.
- ii. Enter the project name, description, and visibility options:
 - o **Name:** Choose a name for the project (e.g., **LMS**).
 - o **Description:** Optionally, add a description to provide more context about the project.
 - o **Visibility:** Choose whether you want the project to be **Private** (accessible only to those invited) or **Public** (accessible to anyone).
- iii. Once you've filled out the details, click **Create** to set up your first project.



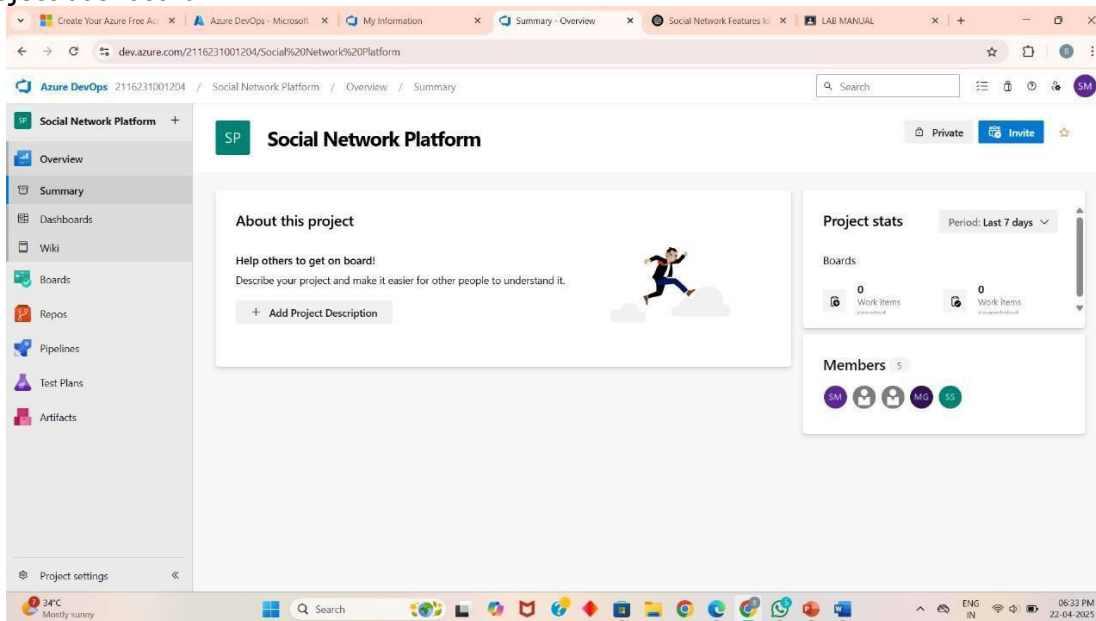
The screenshot shows the 'Create new project' dialog box. It has a title bar with a close button. The form includes a 'Project name' field with a red asterisk, a 'Description' field, and a 'Visibility' section. The 'Visibility' section has two radio buttons: 'Public' (disabled) and 'Private' (selected). Below the radio buttons, there is a note: 'Public projects are disabled for your organization. You can turn on public visibility with [organization policies](#).' Below this is an 'Advanced' section with two dropdown menus: 'Version control' (set to 'Git') and 'Work item process' (set to 'Agile'). At the bottom right are 'Cancel' and 'Create' buttons.

7) 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

8) Organization Home page.

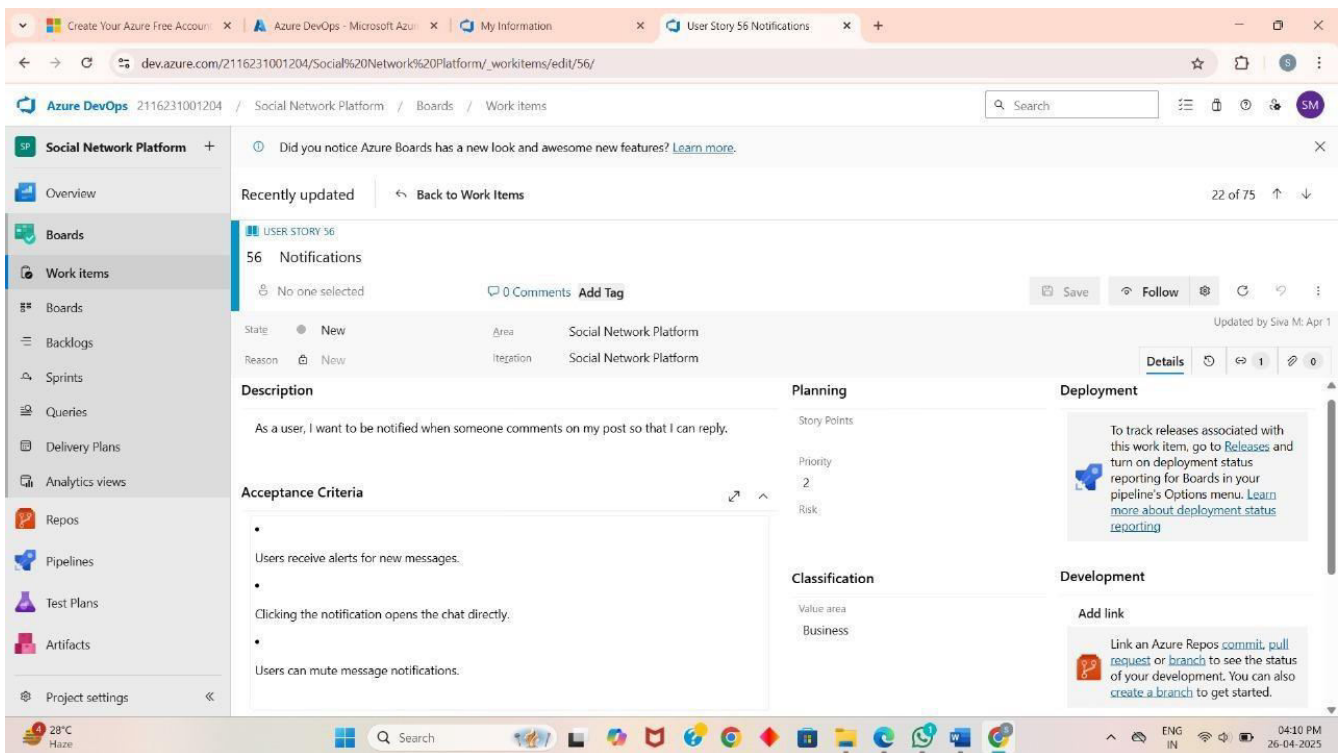
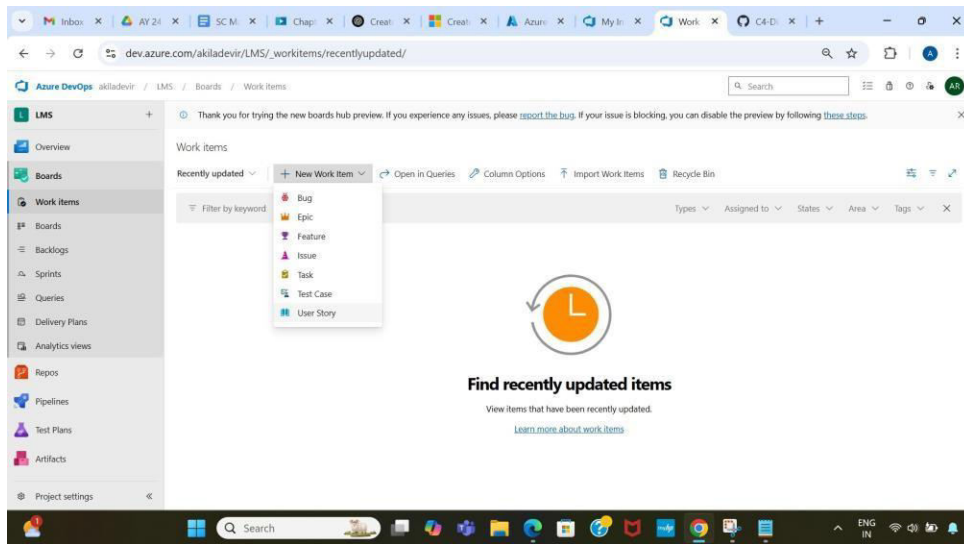


9) Project dashboard



10) To manage user stories

- 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.
- 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:

The user story was written successfully.

EX NO:5

DATE:25/02/2025

SEQUENCE DIAGRAM

AIM:

To design a Sequence Diagram by using Mermaid.js

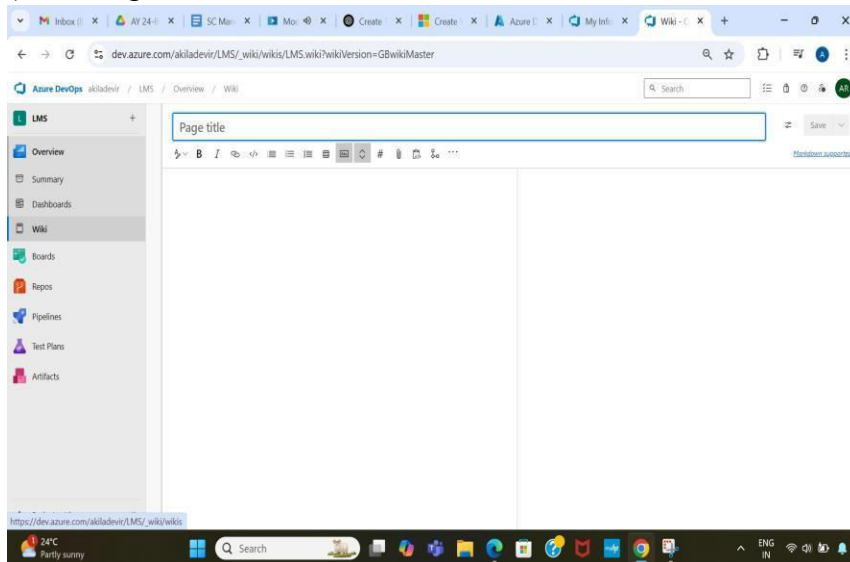
THEORY:

A Sequence Diagram is a key component of Unified Modelling Language (UML) used to visualize the interaction between objects in a sequential order. It focuses on how objects communicate with each other over time, making it an essential tool for modelling dynamic behaviour in a system.

PROCEDURE:

1) Open a project in Azure DevOps Organisations.

2) To design select wiki from menu



Write code for drawing sequence diagram and save the code.

```
sequenceDiagram
    participant User
    participant Frontend
    participant Backend
    participant Database
```

User->>Frontend: Login with email & password

Frontend->>Backend: Send login request

Backend->>Database: Validate credentials

Database-->>Backend: Send user data

Backend-->>Frontend: Auth success response

Frontend-->>User: Display homepage

User->>Frontend: Create a new post
 Frontend->>Backend: Send post content
 Backend->>Database: Save post
 Database-->>Backend: Confirm post saved
 Backend-->>Frontend: Post success response
 Frontend-->>User: Show post on feed

...

Explanation:

User ->> Frontend:

→ Solid line with arrowhead — User sends a direct message to Frontend.

Frontend ->> Backend:

→ Solid line with arrowhead — Frontend sends a direct message to Backend.

Backend ->> Database:

→ Solid line with arrowhead — Backend sends a direct message to Database.

Database -->> Backend:

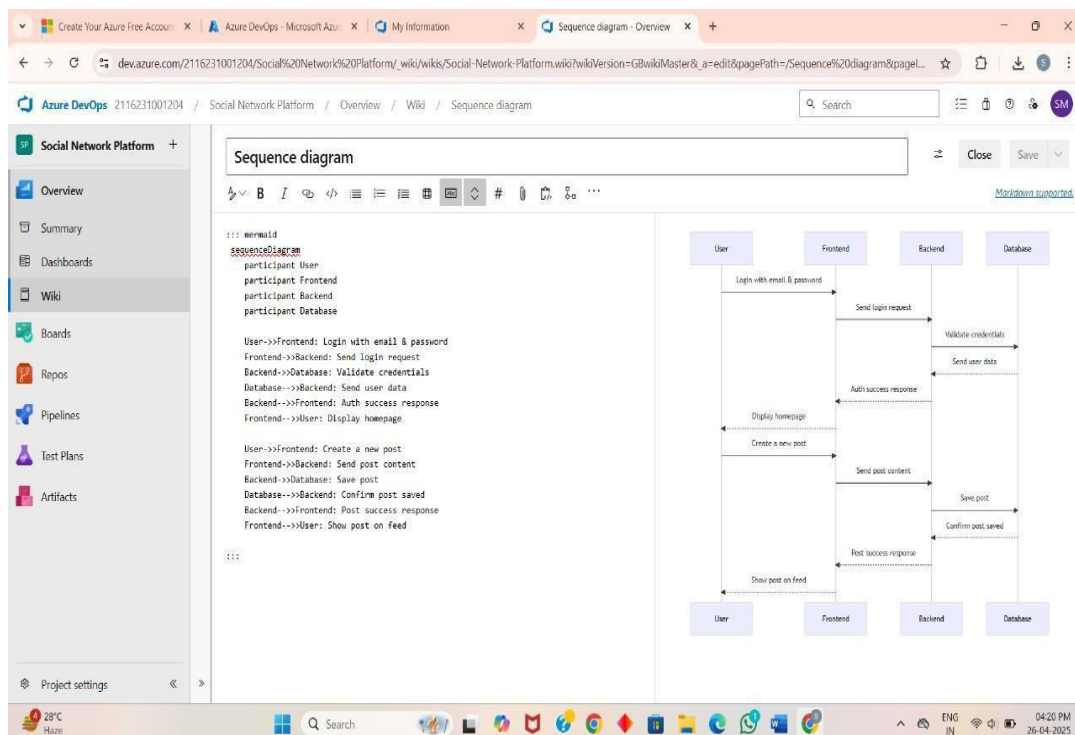
→ Dotted line with arrowhead — Database responds with a message to Backend.

Backend -->> Frontend:

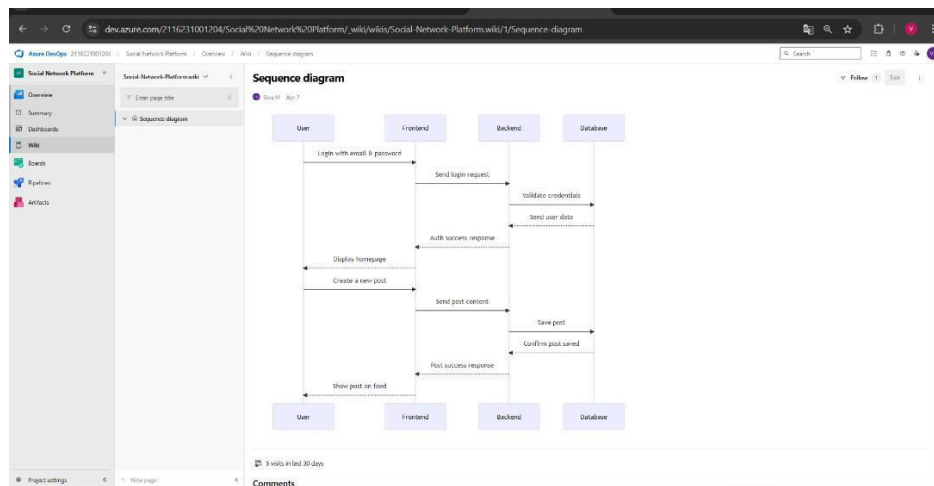
→ Dotted line with arrowhead — Backend responds to Frontend.

Frontend -->> User:

→ Dotted line with arrowhead — Frontend responds to User.



1. click wiki menu and select the page



RESULT:

The sequence diagram was drawn successfully.

EX NO:6

DATE:04/03/2025

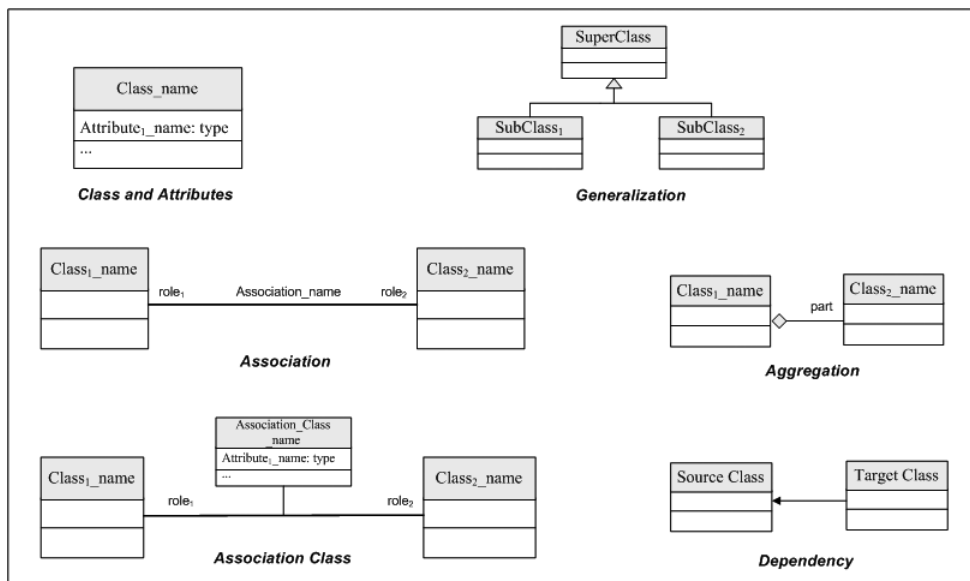
CLASS DIAGRAM

AIM:

To draw a sample class diagram for your project or system.

THEORY:

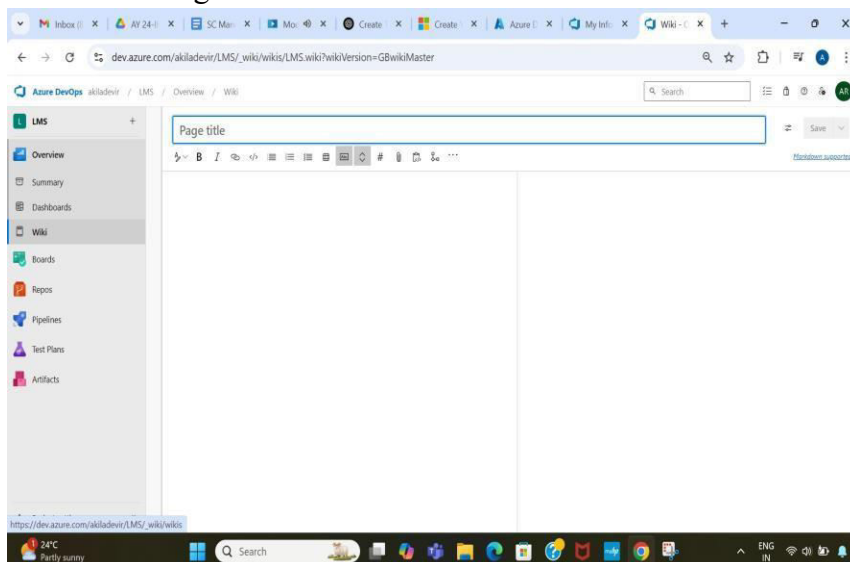
A UML class diagram is a visual tool that represents the structure of a system by showing its classes, attributes, methods, and the relationships between them.



Notations in class diagram

PROCEDURE:

1. Open a project in Azure DevOps Organisations.
2. To design select wiki from menu



3. Write code for drawing class diagram and save the code

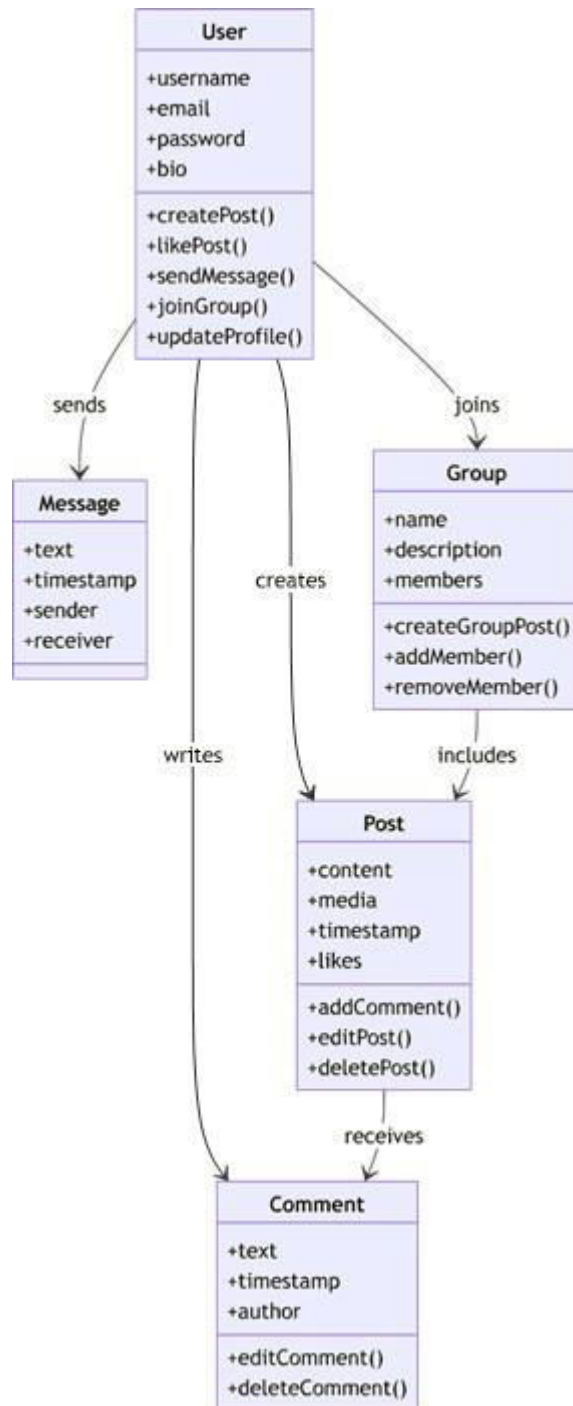
```
::: mermaid
classDiagram
```

```
class User {
+username
+email
+password
+bio
+createPost()
+likePost()
+sendMessage()
+joinGroup()
+updateProfile()
}
```

```
class Post {
+content
+media
+timestamp
+likes
+addComment()
+editPost()
}
```

Relationship Types

Type	Description
<	Inheritance
*	Composition
o	Aggregation
>	Association
<	Association
>	Realization



RESULT:

The use case diagram was designed successfully.

EX NO: 7

DATE:11/03/2025

USECASE DIAGRAM

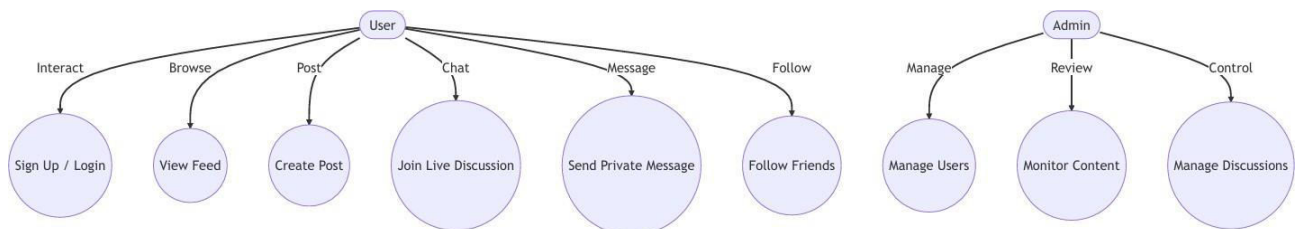
AIM:

Steps to draw the Use Case Diagram using draw.io

THEORY:

- UCD shows the relationships among actors and use cases within a system which Provide an overview of all or part of the usage requirements for a system or organization in the form of an essential model or a business model and communicate the scope of a development project

- **Use Cases**
- **Actors**
- **Relationships**
- **System Boundary Boxes**



PROCEDURE:

Step 1: Create the Use Case Diagram in Draw.io

- Open Draw.io (diagrams.net).
- Click "Create New Diagram" and select "Blank" or "UML Use Case" template.
- Add Actors (Users, Admins, External Systems) from the UML section.
- Add Use Cases (Functionalities) using ellipses.
- Connect Actors to Use Cases with lines (solid for direct interaction, dashed for <<include>> and <<extend>>).
- Save the diagram as .drawio or export as PNG/JPG/SVG.

Step 2: Upload the Diagram to Azure DevOps

Option 1: Add to Azure DevOps Wiki

- Open Azure DevOps and go to your project.
- Navigate to Wiki (Project > Wiki).
- Click "Edit Page" or create a new page.
- Drag & Drop the exported PNG/JPG image.
- Use Markdown to embed the diagram:

Option 2: Attach to Work Items in Azure Boards

- Open Azure DevOps → Navigate to Boards (Project > Boards).
- Select a User Story, Task, or Feature.
- Click "Attachments" → Upload your Use Case Diagram.
- Add comments or descriptions to explain the use case.

RESULT:

The use case diagram was designed successfully

EX NO. 8

DATE:18/03/2025



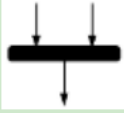








ACTIVITY DIAGRAM

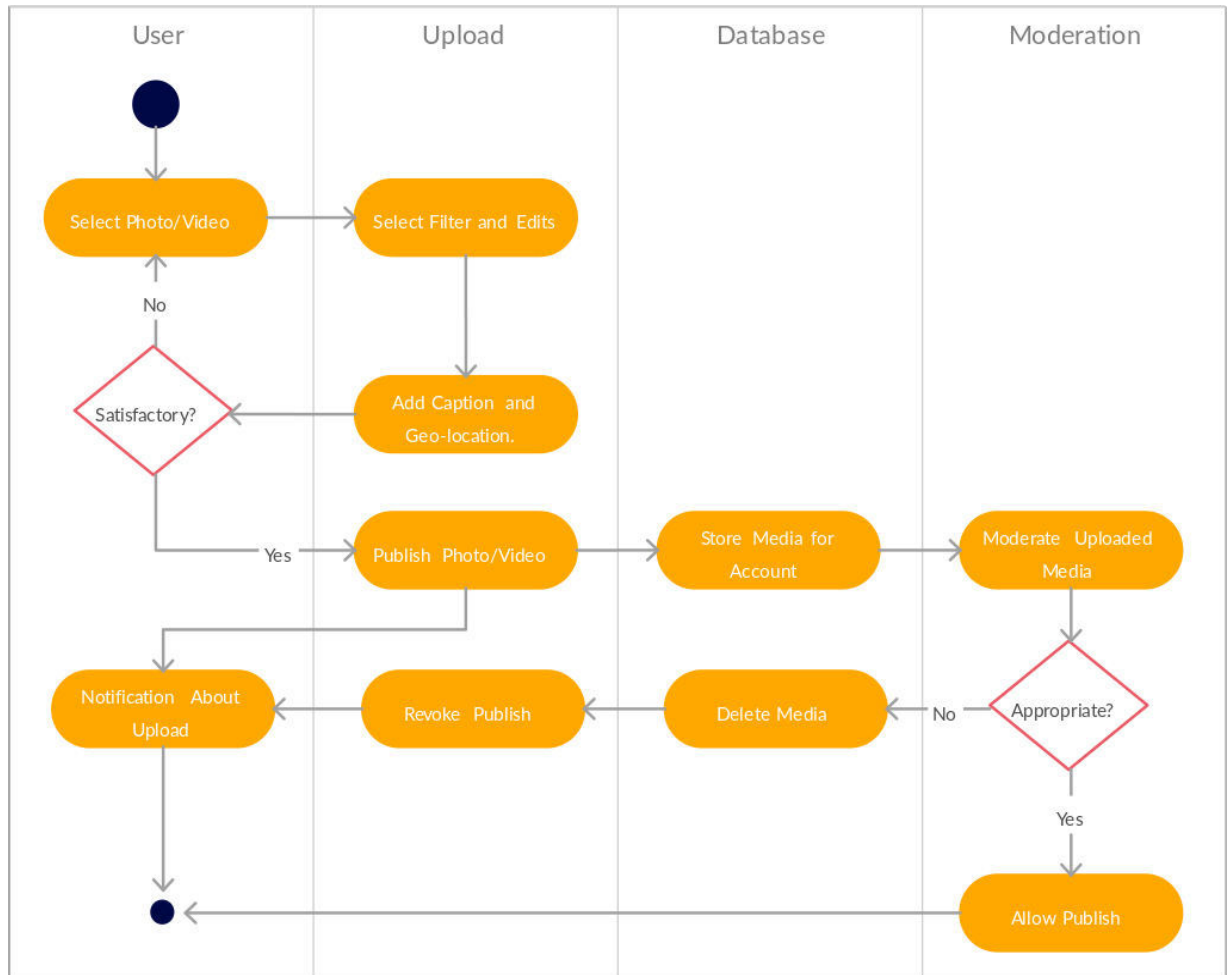
AIM:

To draw a sample activity diagram for your project or system.

THEORY:

Activity diagrams are an essential part of the Unified Modelling Language (UML) that help visualize workflows, processes, or activities within a system. They depict how different actions are connected and how a system moves from one state to another.

Notations	Symbol	Meaning
Start		Shows the beginning of a process
Connector		Shows the directional flow, or control flow, of the activity
Joint symbol		Combines two concurrent activities and re-introduces them to a flow where one activity occurs at a time
Decision		Represents a decision
Note		Allows the diagram creators to communicate additional messages
Send signal		Show that a signal is being sent to a receiving activity
Receive signal		Demonstrates the acceptance of an event
Flow final symbol		Represents the end of a specific process flow
Option loop		Allows the creator to model a repetitive sequence within the option loop symbol
Shallow history pseudostate		Represents a transition that invokes the last active state.
End		Marks the end state of an activity and represents the completion of all flows of a process



PROCEDURE:

1. Draw diagram in draw.io
2. Upload the diagram in Azure DevOps wiki

RESULT:

The activity diagram was designed successfully

EX NO. 9

DATE:25/03/2025

ARCHITECTURE DIAGRAM

AIM:

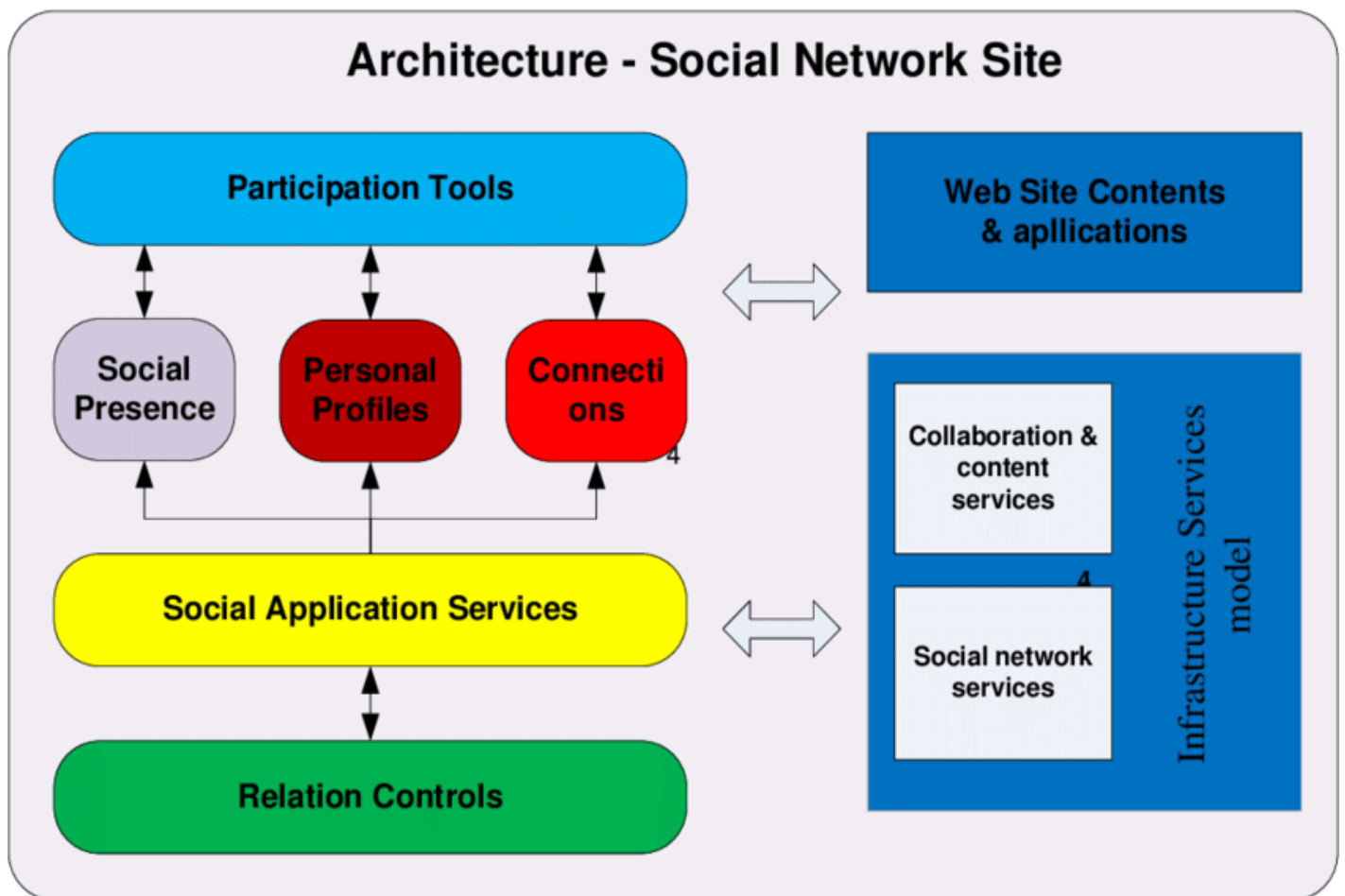
Steps to draw the Architecture Diagram using draw.io.

THEORY:

An architectural diagram is a visual representation that maps out the physical implementation for components of a software system. It shows the general structure of the software system and the associations, limitations, and boundaries between each element.

ARCHITECTURE SYMBOL OVERVIEW





PROCEDURE:

1. Draw diagram in draw.io
2. Upload the diagram in Azure DevOps wiki

RESULT:

The architecture diagram was designed successfully

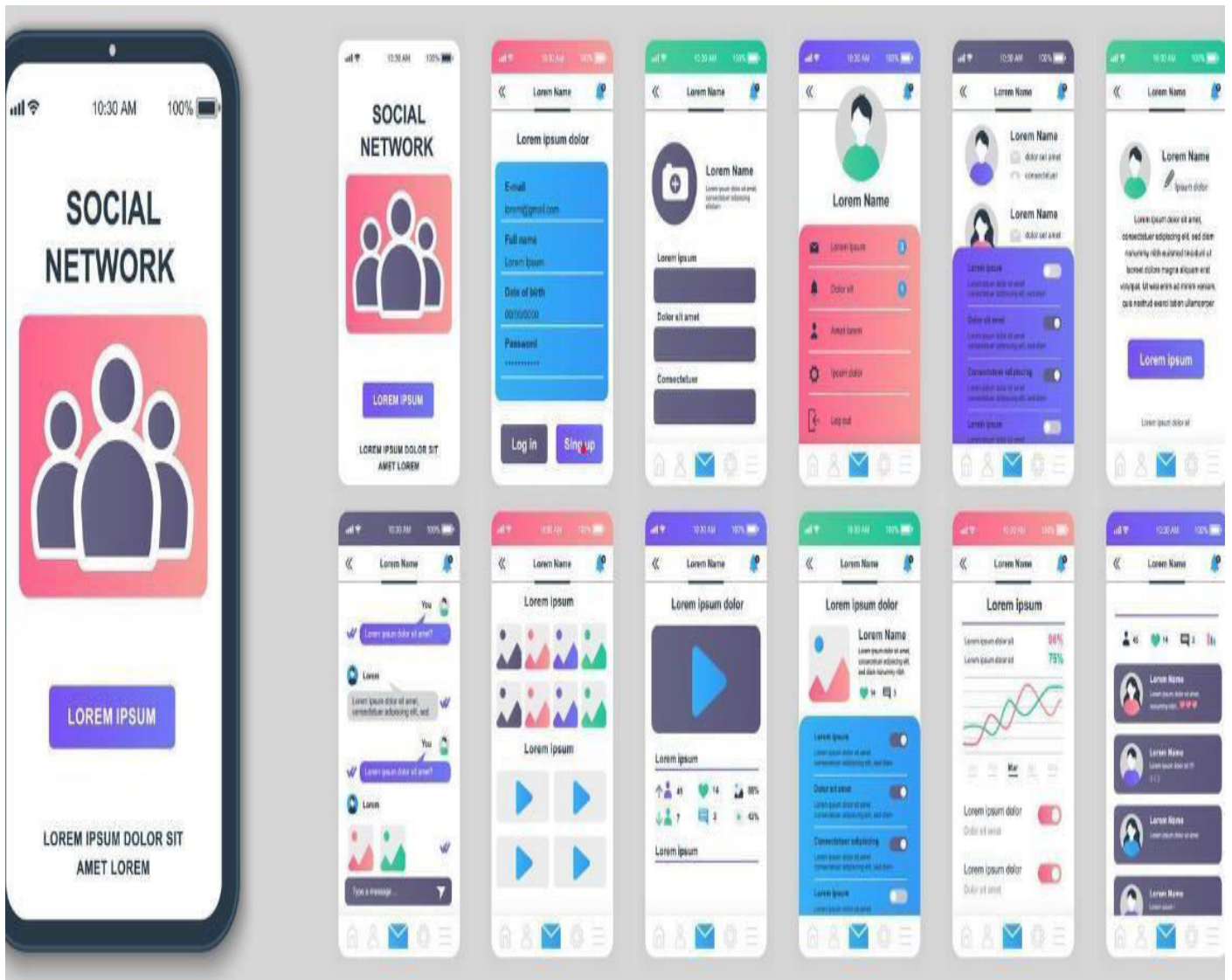
EX NO. 10

DATE:01/04/2025

USER INTERFACE

AIM:

Design User Interface for the given project



RESULT:

The UI was designed successfully.

EX NO. 11

DATE: 08/04/2025

IMPLEMENTATION

AIM:

To implement the given project based on Agile Methodology.

PROCEDURE:

Step 1: Set Up an Azure DevOps Project

- Log in to Azure DevOps.
- Click "New Project" → Enter project name → Click "Create".
- Inside the project, navigate to "Repos" to store the code.

Step 2: Add Your Web Application Code

- Navigate to Repos → Click "Clone" to get the Git URL.
- Open Visual Studio Code / Terminal and run:

```
git clone <repo_url>
cd <repo_folder>
```
- Add web application code (HTML, CSS, JavaScript, React, Angular, or backend like Node.js, .NET, Python, etc.).
- Commit & push:

```
git add .
git commit -m "Initial commit"
git push origin main
```

Step 3: Set Up Build Pipeline (CI/CD - Continuous Integration)

- Navigate to Pipelines → Click "New Pipeline".
- Select Git Repository (Azure Repos, GitHub, or Bitbucket).
- Choose Starter Pipeline or a pre-configured template for your framework.
- Modify the azure-pipelines.yml file (Example for a Node.js app):

```
trigger:
- main
pool:
vmImage: 'ubuntu-latest'

steps:
- task: UseNode@1
  inputs:
    version: '16.x'

- script: npm install
  displayName: 'Install dependencies'

- script: npm run build
  displayName: 'Build application'

- task: PublishBuildArtifacts@1
  inputs:
    pathToPublish: 'dist'
    artifactName: 'drop'
```

Click "Save and Run" → The pipeline will start building app.

Step 4: Set Up Release Pipeline (CD - Continuous Deployment)

- Go to Releases → Click "New Release Pipeline".
- Select Azure App Service or Virtual Machines (VMs) for deployment.
- Add an artifact (from the build pipeline).
- Configure deployment stages (Dev, QA, Production).
- Click "Deploy" to push your web app to Azure.

RESULT:

Thus, the application was successfully implemented.

EX.NO: 12

DATE: 15/04/2025

TESTING-TEST PLAN, TEST CASE AND
LOAD TESTING

AIM:

To design and manage structured test plans and test cases in Azure DevOps for validating core user stories through both happy path and error scenarios and evaluate the performance of the application's endpoint by creating and executing load tests using Azure Load Testing.

PROCEDURE:

TEST CASE DESIGN PROCEDURE

1. Understand Core Features of the Application

- Review requirement documents and user stories.
- Identify all main functionalities of the application.
- Ensure complete coverage of modules before test case creation.

2. Define User Interactions

- Determine common user behaviours based on application flow.
- Translate user actions into testable scenarios.
- Ensure each test case mimics a real user operation.

3. Design Happy Path Test Cases

- Create test cases for expected and correct user actions.
- Ensure each functionality works under normal conditions.
- Add these cases under the relevant Test Suite in Azure DevOps.

4. Design Error Path Test Cases

- Identify edge cases, invalid inputs, and system failures.
- Test how the system handles incorrect or unexpected behavior.
- Add these test cases to the same or a separate Test Suite in Azure DevOps.

5. Break Down Steps and Expected Results

- Write step-by-step instructions in the "Steps" section of the test case.
- Provide expected results for each action.

- Ensure clarity for both manual execution and automation mapping.

6. Use Clear Naming and IDs

- Name test cases clearly using a defined naming convention (e.g., TC01, TC02, etc.).
- Ensure titles reflect the purpose of the test case.
- Azure DevOps auto-generates test case IDs for tracking.

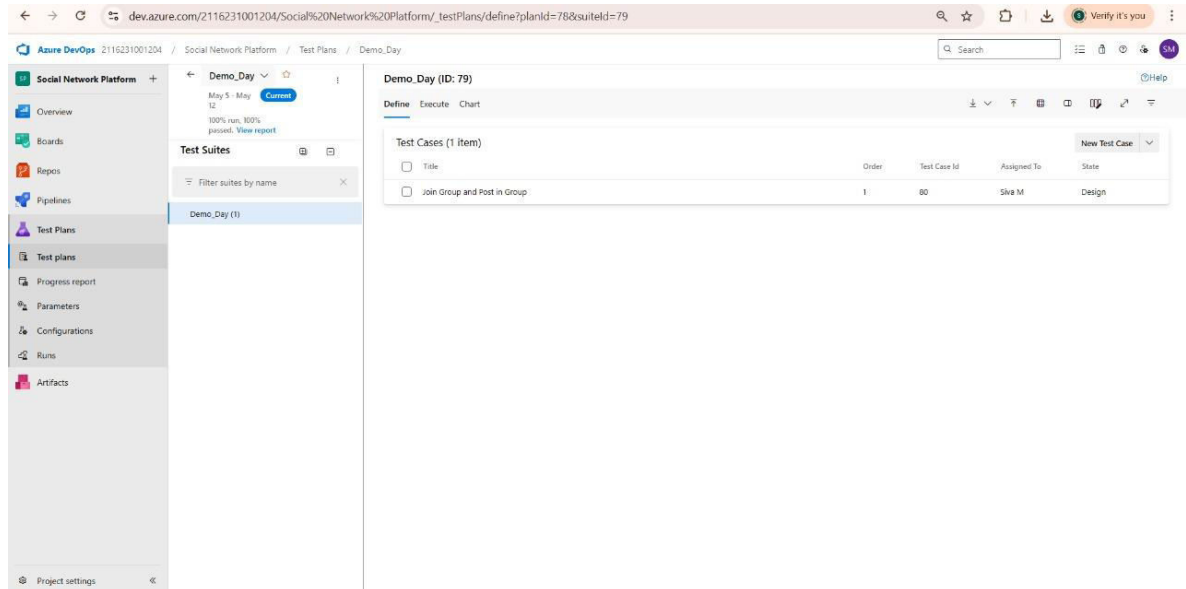
7. Separate Test Suites

- Group test cases based on functionality (e.g., Login, Playlist, Recommendations).
- Use Static, Requirement-based, or Query-based suites in Azure DevOps.
- Improves traceability and execution flow.

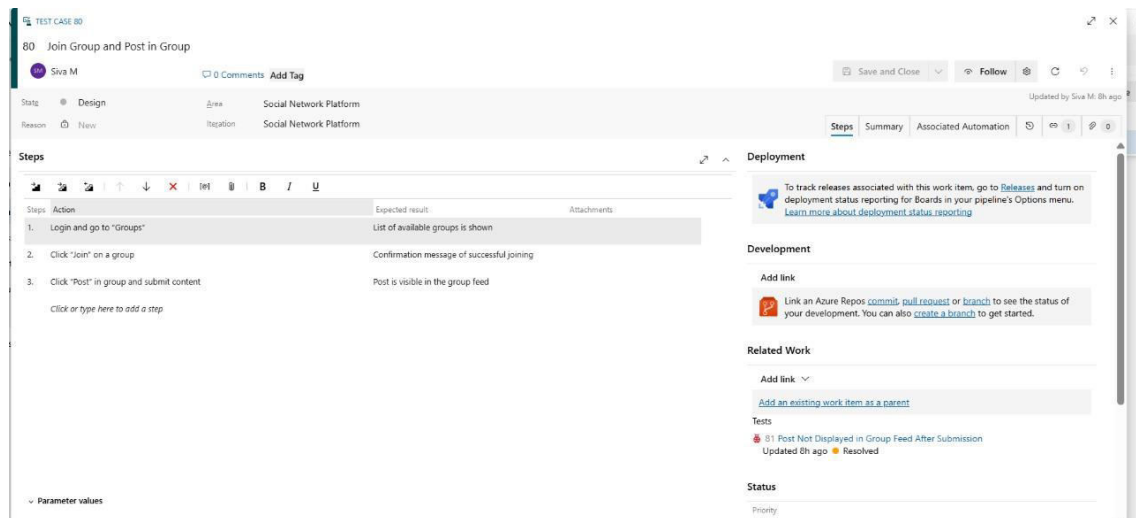
8. Prioritize and Review

- Mark test cases with priority (High, Medium, Low).
- Review test cases for completeness and correctness.
- Ensure alignment with associated user stories or features.

1. New test plan



2. Test case



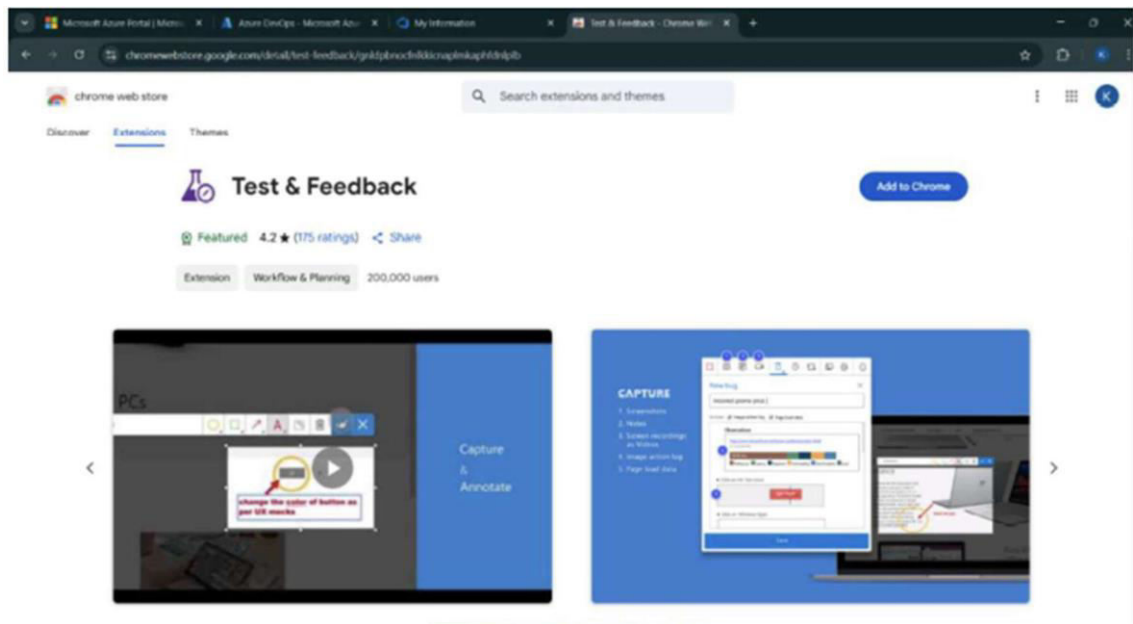
The screenshot shows a Test Case titled "Join Group and Post in Group" in the Azure DevOps interface. The test case is associated with the "Social Network Platform" project. It is currently in the "Design" state. The test steps are as follows:

Steps	Action	Expected result	Attachments
1.	Login and go to "Groups"	List of available groups is shown	
2.	Click "Join" on a group	Confirmation message of successful joining	
3.	Click "Post" in group and submit content	Post is visible in the group feed	

Below the steps, there is a prompt: "Click or type here to add a step".

On the right side of the interface, there are sections for "Deployment", "Development", "Related Work", "Tests", and "Status". The "Tests" section shows a list of test results, including "81 Post Not Displayed in Group Feed After Submission" which was updated 8h ago and is in a "Resolved" state.

3. Installation of Test

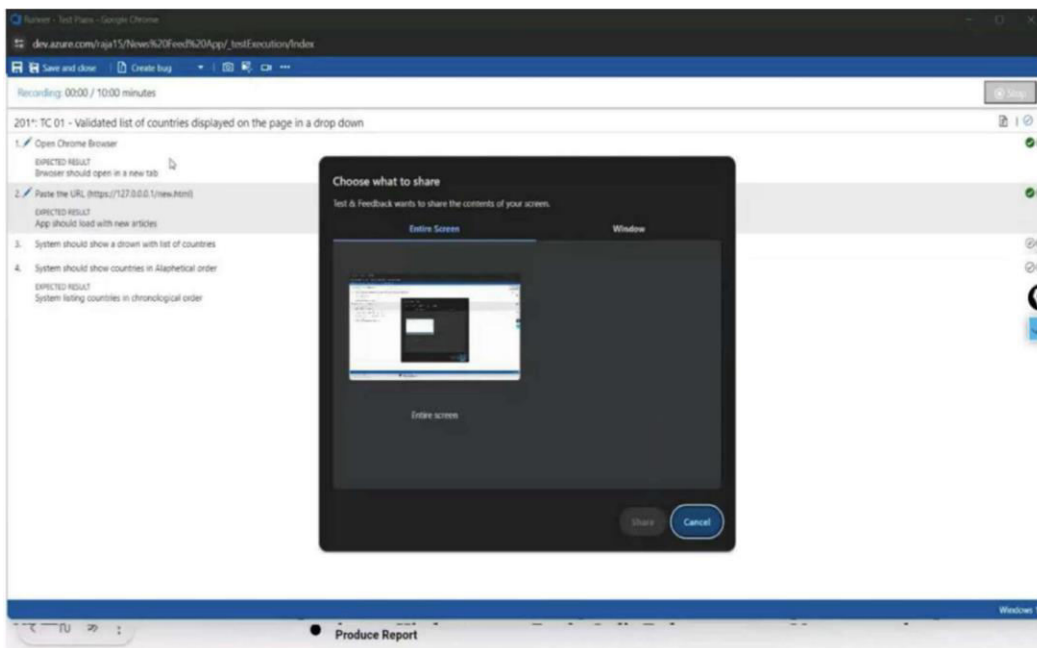


The screenshot shows the Chrome Web Store page for the "Test & Feedback" extension. The extension is featured with a 4.2 star rating from 175 ratings. It is categorized as an "Extension" and "Workflow & Planning" tool, with over 200,000 users. The page includes a "Add to Chrome" button. Below the header, there are two preview images: one showing a "Capture & Annotate" interface with a play button and a text box, and another showing a "CAPTURE" interface with a list of steps and a "Start" button.

4. Running the Test Cases

21: User Sign-Up and Login		
1. Enter valid email/password and click "Register"		
EXPECTED RESULT	Verification email is sent	
2. Click "Sign up with Google" and select account		
EXPECTED RESULT	User is signed up and logged in	
3. Click "Forgot Password" and submit email		
EXPECTED RESULT	Password reset email is sent	
4. Enter incorrect login credentials		
EXPECTED RESULT	Error message is displayed	

5. Recording the Test Cases



6. Creating Bugs

Azure DevOps 231001191 / E-Learning Platform / Boards / Work Items

Search

E-Learning Platform

Overview

Boards

Work items

Boards

Backlogs

Sprints

Queries

Delivery Plans

Analytics views

Repos

Pipelines

Test Plans

Artifacts

Project settings

Recently updated

Back to Work Items

1 of 8

40 User Sign-Up does not respond

No one selected

0 Comments

Add Tag

Save

Follow

Details

Updated by Saravanan G just now

Repro Steps

5/6/2025 1:36 PM Bug filed on "User Sign-Up and Login"

Step no.	Result	Title
1.	Passed	Enter valid email/password and click "Register"
		Expected Result Verification email is sent
2.	Passed	Click "Sign up with Google" and select account
		Expected Result User is signed up and logged in
3.	Passed	Click "Forgot Password" and submit email
		Expected Result Password reset email is sent
4.	Passed	Enter incorrect login credentials
		Expected Result

Planning

Resolved Reason

Fixed

Story Points

Priority

2

Severity

3 - Medium

Activity

Effort (Hours)

Original Estimate

Remaining

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

Link an Azure Repos [commit](#), [pull request](#) or [branch](#) to see the status of your development. You can also [create a branch](#) to get started.

Related Work

7. Test Case Results

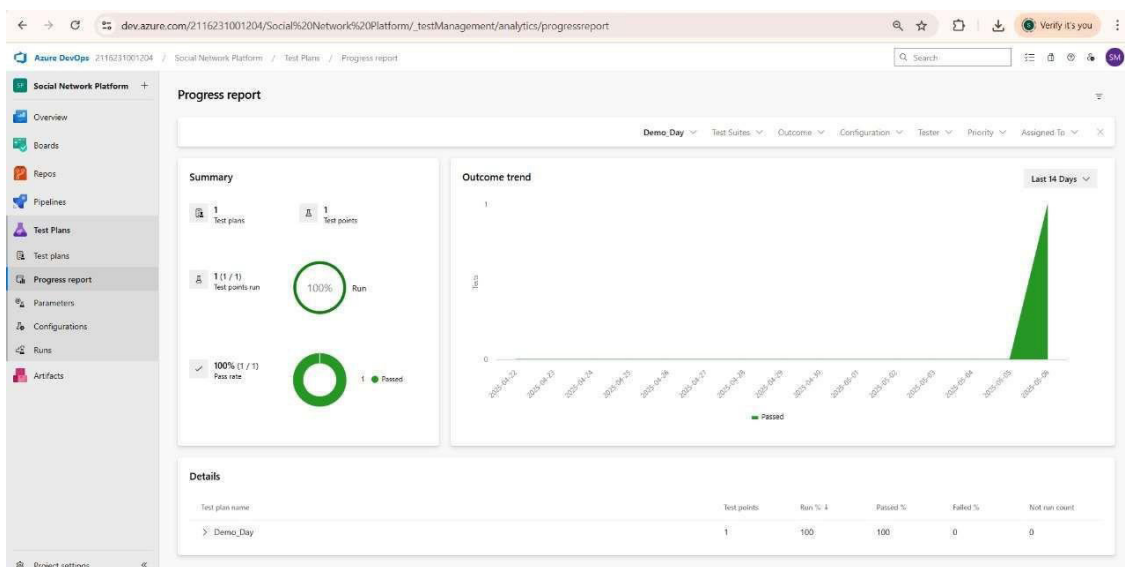
TC05 – View Playlist Page

Test Case Results

Outcome	TimeStamp	Configuration	Run by	Tester	Test PI
Passed	4m ago	Windows 10	Karthick S	Mallu karthick Balaji ...	Music
Passed	12m ago	Windows 10	Karthick S	Mallu karthick Balaji ...	Music
Not Applicable	12m ago	Windows 10	Karthick S	Mallu karthick Balaji ...	Music
Passed	14m ago	Windows 10	Karthick S	Mallu karthick Balaji ...	Music
Passed		Windows 10	Karthikeyan Senthil	Mallu karthick Balaji ...	Music
Passed	Saturday	Windows 10	Mallu karthick Balaji	Mallu karthick Balaji ...	Music
Failed	Saturday	Windows 10	Mallu karthick Balaji	Mallu karthick Balaji ...	Music
Passed	Apr 11	Windows 10	Karthick S	Mallu karthick Balaji ...	Music
Passed	Apr 11	Windows 10	Karthick S	Mallu karthick Balaji ...	Music

[Open execution history for current test point](#)

8. Progress Report



LOAD TESTING PROCEDURE :

Steps to Create an Azure Load Testing Resource:

Before you run your first test, you need to create the Azure Load Testing resource:

1. Sign in to Azure Portal

Go to <https://portal.azure.com> and log in.

2. Create the Resource

- Go to Create a resource — Search for “Azure Load Testing”.
- Select Azure Load Testing and click Create.

3. Fill in the Configuration Details

- Subscription: Choose your Azure subscription.
- Resource Group: Create new or select an existing one.
- Name: Provide a unique name (no special characters).
- Location: Choose the region for hosting the resource.

4. (Optional) Configure tags for categorization and billing.

5. Click Review + Create, then Create.

6. Once deployment is complete, click Go to resource.

Steps to Create and Run a Load Test:

Once your resource is ready:

1. Go to your Azure Load Testing resource and click Add HTTP requests > Create.

2. Basics Tab

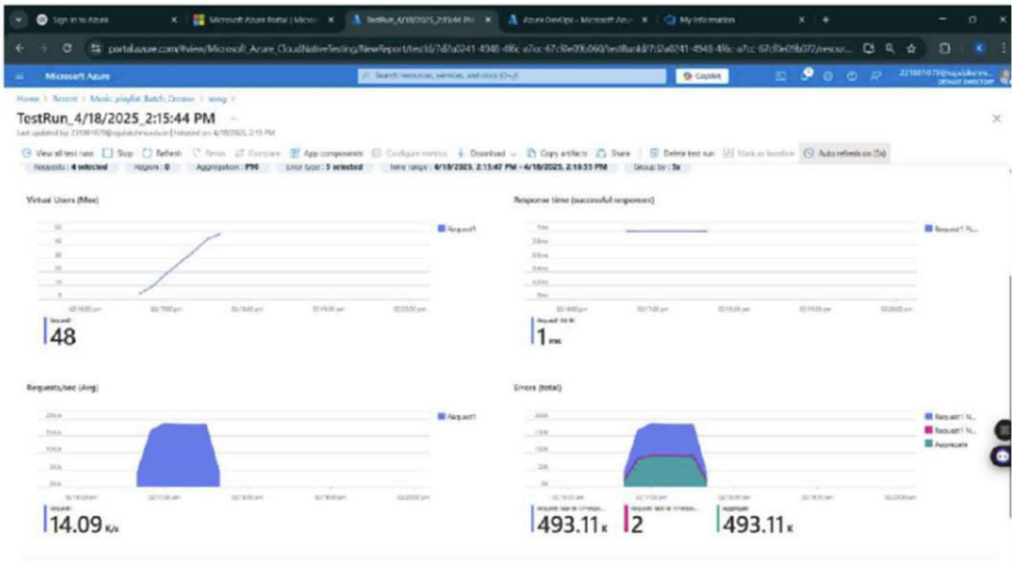
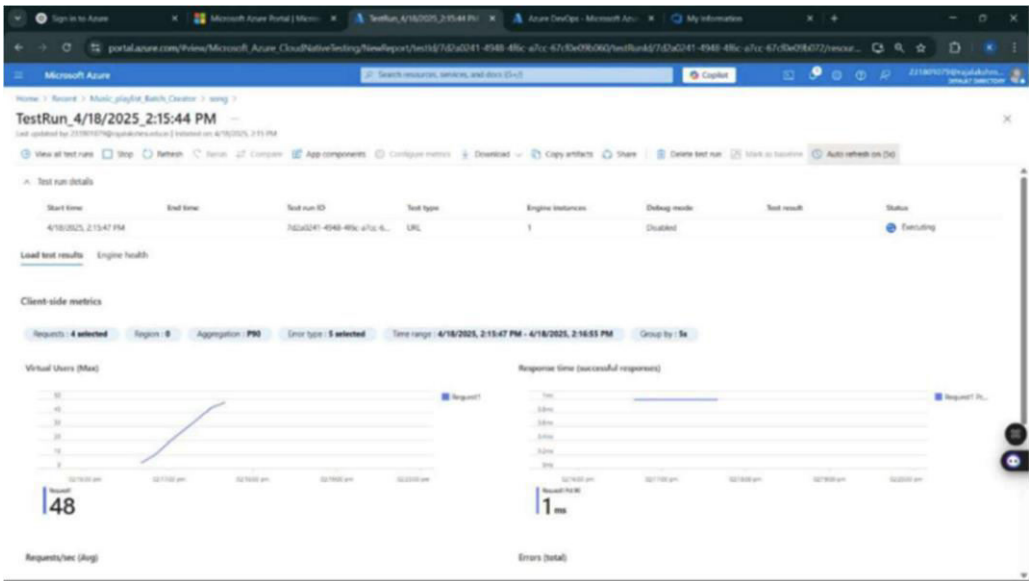
- Test Name: Provide a unique name.
- Description: (Optional) Add test purpose.
- Run After Creation: Keep checked.

3. Load Settings

- Test URL: Enter the target endpoint (e.g., <https://yourapi.com/products>).

4. Click Review + Create — Create to start the test.

Load Testing



RESULT:

Test plans and test cases for selected user stories were created in Azure DevOps, covering both happy and error paths and an Azure Load Testing resource was also set up, and a load test was successfully run to evaluate the performance of the target endpoint.