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 **SUMANTH S** REGISTER NO. **2116231001224**

ACADEMIC YEAR 2024-25 **SEMESTER**- IV **BRANCH**: B. Tech Information Technology [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		

Course Outcomes (COs)

Course Name: Software Engineering

Course Code: CS23432

CO 1	Understand the software development process models.
CO 2	Determine the requirements to develop software
CO 3	Apply modeling and modeling languages to design software products
CO 4	Apply various testing techniques and to build a robust software products
CO 5	Manage Software Projects and to understand advanced engineering concepts

CO - PO – PSO matrices of course

PO/PSO / CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CS23432.1	2	2	3	2	2	2	2	2	2	2	3	2	1	3	-
CS23432.2	2	3	1	2	2	1	-	1	1	1	2	-	1	2	-
CS23432.3	2	2	1	1	1	1	1	1	1	1	1	1	2	2	1
CS23432.4	2	2	3	2	2	2	1	0	2	2	2	1	1	2	1
CS23432.5	2	2	2	1	1	1	1	0	2	1	1	1	2	1	-
Average	2.0	2.2	2.0	1.6	1.6	1.4	1.3	1.3	1.6	1.4	1.8	1.3	1.4	2.0	1.0

Correlation levels 1, 2 or 3 are as defined below:
1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High) No correlation: "-"

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.

Getting Started with Azure DevOps

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 prepare Problem Statement for your given project.

PROBLEM STATEMENT:

Educational institutions and organizations need a flexible, scalable solution to deliver high-quality learning experiences beyond the limitations of traditional classrooms. An E-Learning Platform will provide users with access to a variety of courses, quizzes, and certifications anytime and anywhere, enabling self-paced learning. The platform will allow students to track their progress, identify strengths and weaknesses, and receive personalized course recommendations to improve performance. Instructors will be able to create and manage course content easily, while administrators can monitor user activities and system performance, ensuring a smooth, effective, and data-driven learning environment that supports career and academic growth.

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 for E-Learning Platform Development

Agile planning is essential to building an E-Learning Platform that is flexible, user-centered, and capable of adapting to evolving learning needs. Following the Agile methodology, we avoid rigid, upfront plans and instead use an incremental, iterative approach. Continuous feedback from students, instructors, and administrators will guide improvements throughout the project.

In Agile planning for the E-Learning Platform, the project will be broken down into smaller, manageable pieces of work (features and improvements), with a clear vision in mind. Agile planning will include:

- Roadmaps to guide the platform's feature releases and development schedule.
- Sprints to focus on building and delivering specific modules (e.g., user authentication, course catalog, quizzes, progress tracking).
- Feedback loops to stay responsive to users' needs and quickly adapt the platform.

Tasks will be framed as user stories to capture real requirements from the perspective of platform users (students, teachers, admins). For example:

"As a student, I want to track my progress so that I can stay motivated."

By thinking in terms of user stories, the team ensures that we are always focused on delivering real value to end-users.

Steps in Agile Planning Process for the E-Learning Platform

1. Define Vision

   Create a clear vision for the E-Learning Platform: to offer an engaging, interactive, accessible, and personalized learning experience to a global audience.

2. Set Clear Expectations on Goals

   Outline project goals, such as:

   - Enable seamless course enrollment and management.
   - Support multimedia content (videos, PDFs, quizzes).

- Track and display learner progress.
- Facilitate communication between students and instructors.

3. Define and Break Down the Product Roadmap

   Develop a roadmap identifying major features (e.g., user authentication, course creation, learning analytics) and prioritize their release.

4. Create Tasks Based on User Stories

   Identify user stories, for example:

   "As an instructor, I want to create and manage courses."

   "As a learner, I want to view and complete assigned lessons."

   "As an admin, I want to manage user access rights."

5. Populate Product Backlog

   Create a backlog listing all user stories, tasks, bugs, and enhancements, prioritized based on importance and user impact.

6. Plan Iterations and Estimate Effort

   Organize work into 1–2 week sprints, estimating the effort for each task using story points or time estimates.

7. Conduct Daily Stand-Ups

   Hold short daily meetings to update on progress, discuss challenges, and plan the day's work.

8. Monitor and Adapt

   Continuously monitor progress, collect user feedback after each sprint, and adjust the backlog and roadmap to better meet users' needs.

RESULT:

Thus, the Agile plan was completed successfully.

EX NO: 4

DATE:18/02/2025

CREATE USER STORIES WITH ACCEPTANCE CRITERIA

AIM:

To create User Stories with Acceptance Criteria.

THEORY:

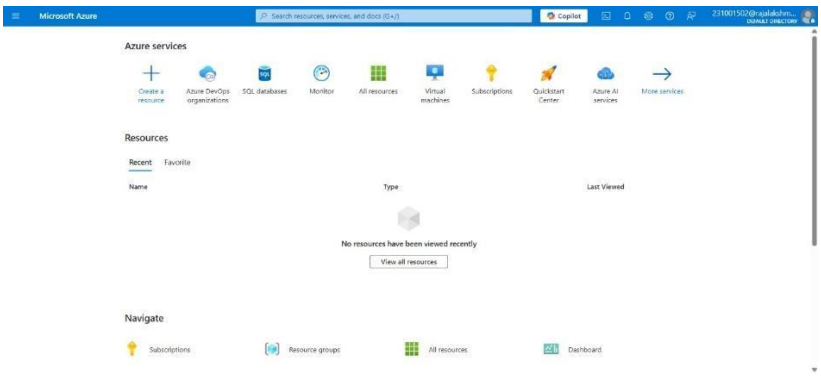
A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer.

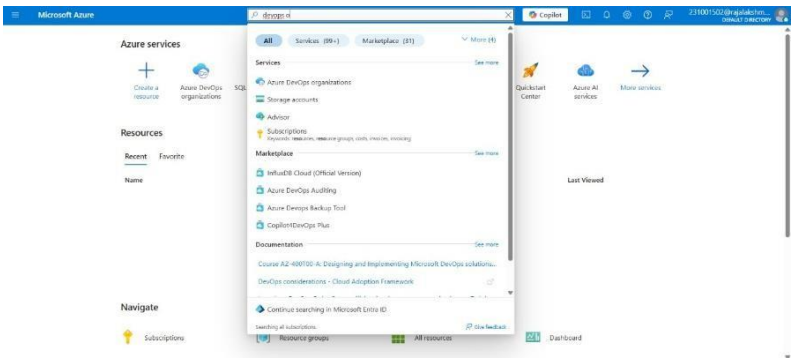
User story template

"As a [role], I [want to], [so that]."

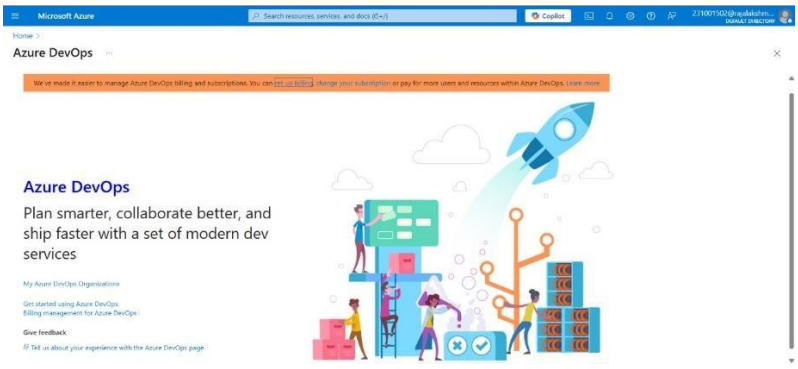
PROCEDURE:

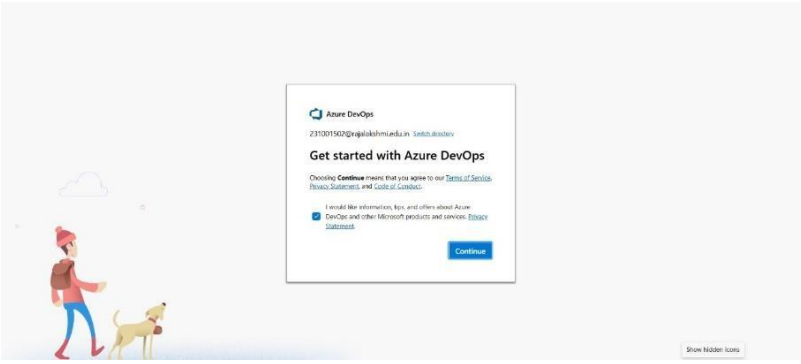
1. Open your web browser and go to the Azure website: *https://azure.microsoft.com/en-in* Sign in using your Microsoft account credentials. If you don't have an account, you'll need to create one.
2. If you don't have a Microsoft account, you can sign up for *https://signup.live.com/?lic=1*
3. Azure home page
4. Open DevOps environment in the Azure platform by typing Azure DevOps Organizations in the search bar.

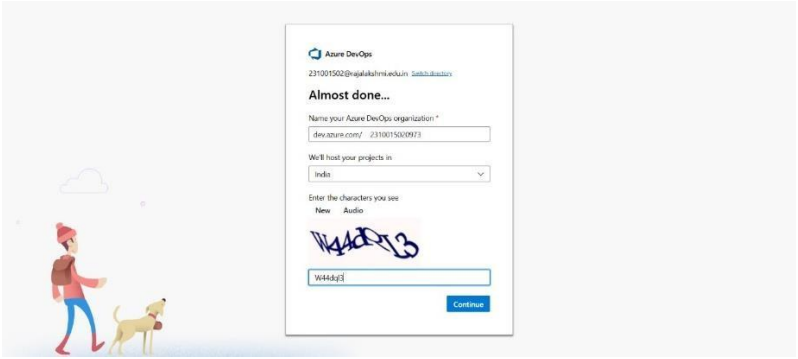




5. Click on the My Azure DevOps Organization link and create an organization and you should be taken to the Azure DevOps Organization Home page.







6. Create the First Project in Your Organization

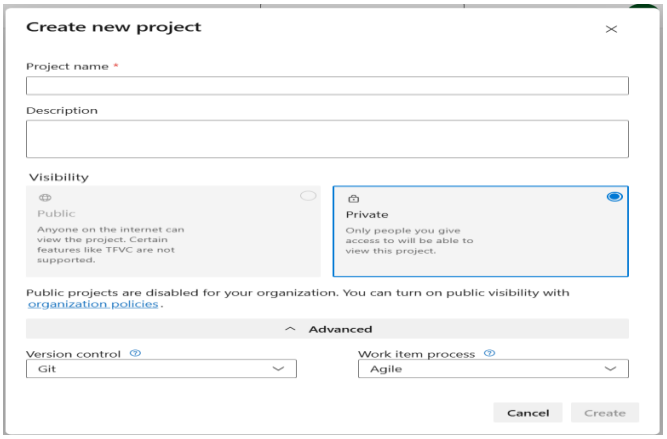
After the organization is set up, you'll need to create your first **project**. This is where you'll begin to manage code, pipelines, work items, and more.

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

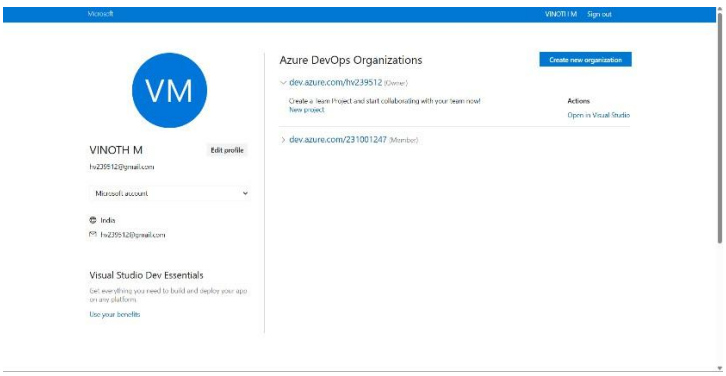
ii. Enter the project name, description, and visibility options:

- **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).

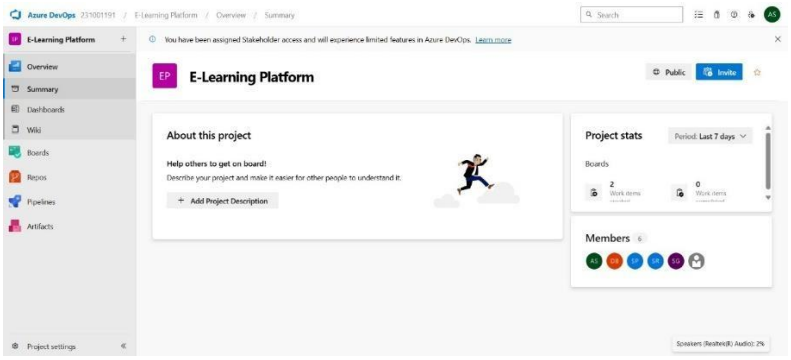
iii. Once you've filled out the details, click **Create** to set up your first project.



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 Organization Home page.



8. Project dashboard

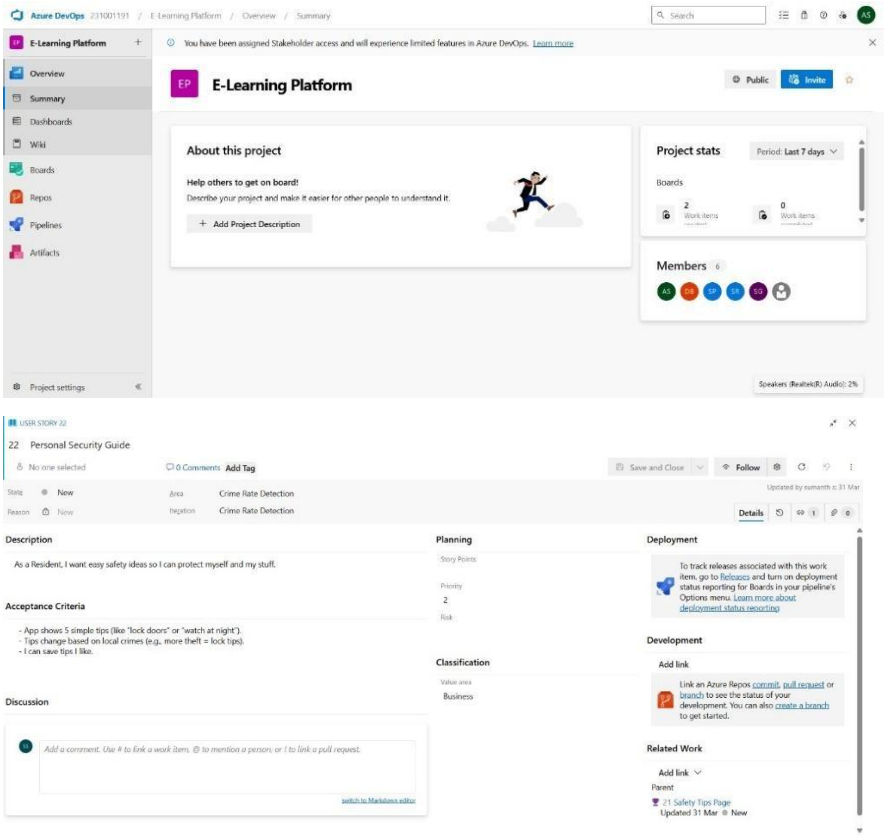


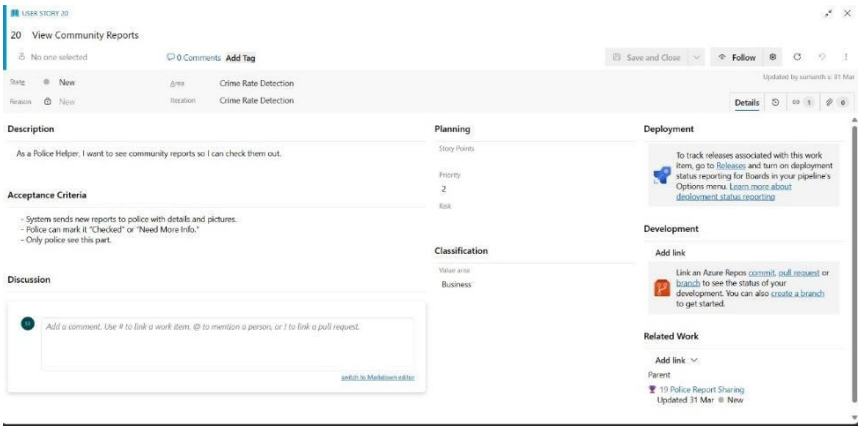
9. To manage user stories

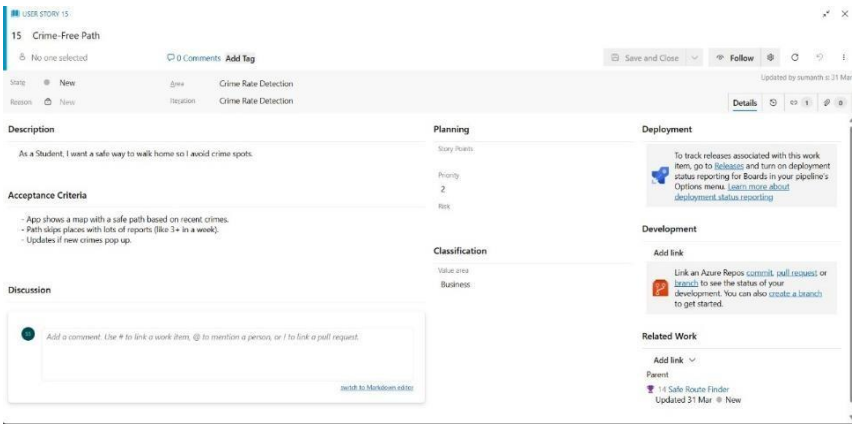
   a) From the **left-hand navigation menu**, click on **Boards**. This will take you to the main **Boards** page, where you can manage work items, backlogs, and sprints.

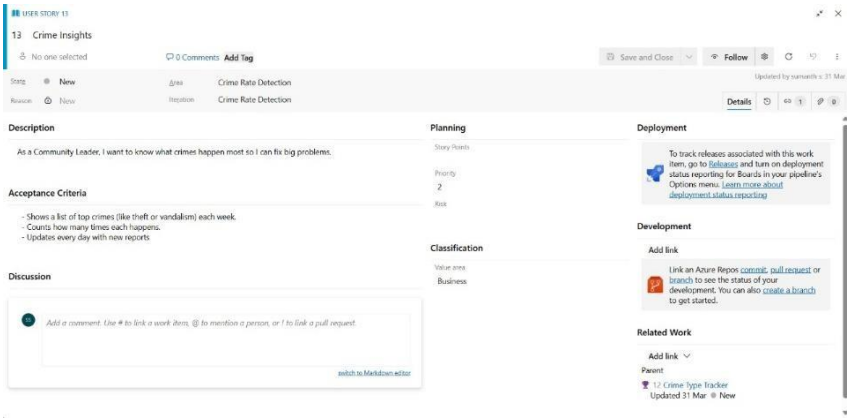
   b) On the **work items** page, you'll see the option to **Add a work item** at the top. Alternatively, you can find a **+** button or **Add New Work Item** depending on the view you're in. From the **Add a work item** dropdown, select **User Story**. This will open a form to enter details for the new User Story.

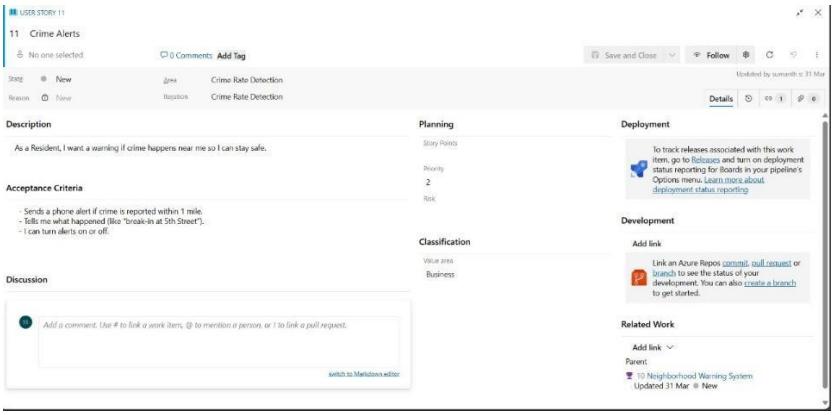
10. Fill in User Story Details

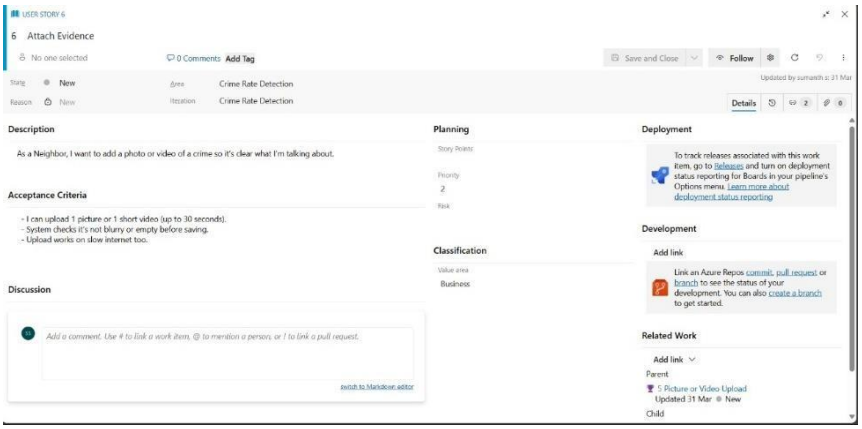


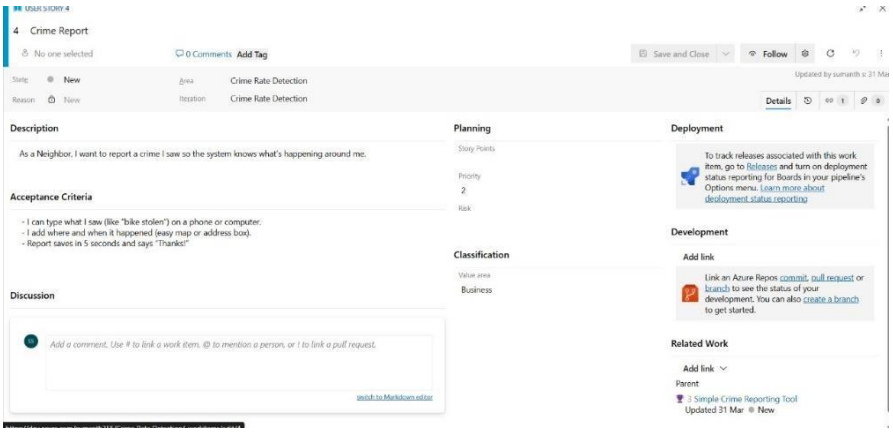

USER STORY 20
20 View Community Reports
No one selected
0 Comments Add Tag
Save and Close
Follow
State New
Reason New
Area Crime Rate Detection
Iteration Crime Rate Detection
Details
Description
As a Police Helper, I want to see community reports so I can check them out.
Acceptance Criteria
- System sends new reports to police with details and pictures.
- Police can mark it "Checked" or "Need More Info."
- Only police see this part.
Discussion
Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request.
switch to Markdown editor
Planning
Story Points
Priority
2
Risk
Classification
Value area
Business
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
Add link
Parent
19 Police Report Sharing
Updated 31 Mar New

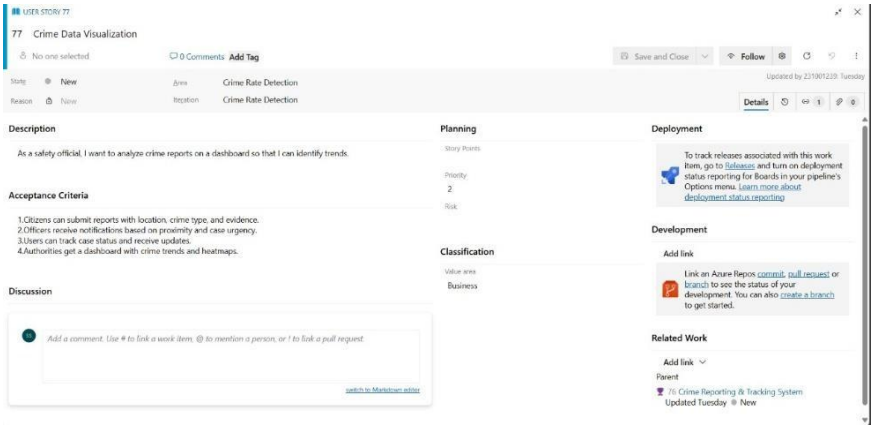

USER STORY 15
15 Crime-Free Path
No one selected
0 Comments Add Tag
Save and Close
Follow
State New
Reason New
Area Crime Rate Detection
Iteration Crime Rate Detection
Details
Description
As a Student, I want a safe way to walk home so I avoid crime spots.
Acceptance Criteria
- App shows a map with a safe path based on recent crimes.
- Path skips places with lots of reports (like 3+ in a week).
- Updates if new crimes pop up.
Discussion
Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request.
switch to Markdown editor
Planning
Story Points
Priority
2
Risk
Classification
Value area
Business
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
Add link
Parent
14 Safe Route Finder
Updated 31 Mar New

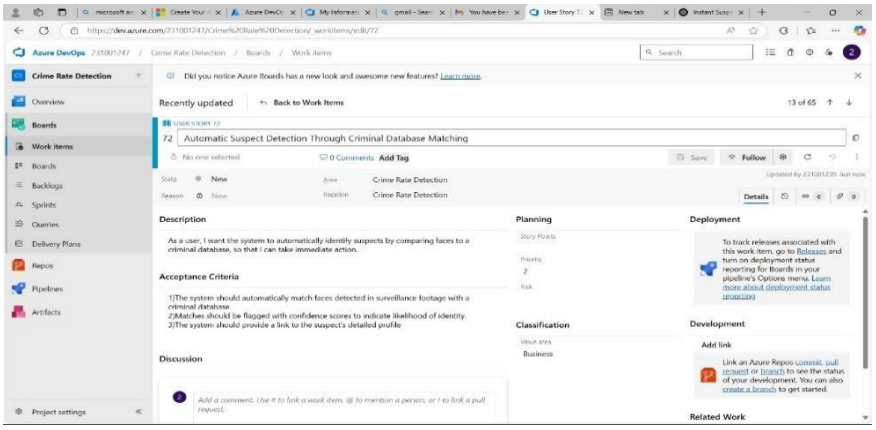

USER STORY 13
13 Crime Insights
No one selected
0 Comments Add Tag
Save and Close
Follow
State New
Reason New
Area Crime Rate Detection
Iteration Crime Rate Detection
Details
Description
As a Community Leader, I want to know what crimes happen most so I can fix big problems.
Acceptance Criteria
- Shows a list of top crimes (like theft or vandalism) each week.
- Counts how many times each happens.
- Updates every day with new reports
Discussion
Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request.
switch to Markdown editor
Planning
Story Points
Priority
2
Risk
Classification
Value area
Business
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
Add link
Parent
12 Crime Type Tracker
Updated 31 Mar New


USER STORY 11
11 Crime Alerts
No one selected
0 Comments Add Tag
Save and Close
Follow
State New
Reason New
Area Crime Rate Detection
Iteration Crime Rate Detection
Details
Description
As a Resident, I want a warning if crime happens near me so I can stay safe.
Acceptance Criteria
- Sends a phone alert if crime is reported within 1 mile.
- Tells me what happened (like "break-in at 5th Street").
- I can turn alerts on or off.
Discussion
Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request.
switch to Markdown editor
Planning
Story Points
Priority
2
Risk
Classification
Value area
Business
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
Add link
Parent
10 Neighborhood Warning System
Updated 31 Mar New









RESULT:

The user story was written successfully.

EX NO: 5

DATE:25/02/2025

DESIGNING SEQUENCE DIAGRAMS USING AZURE DEVOPS-WIKI

AIM:

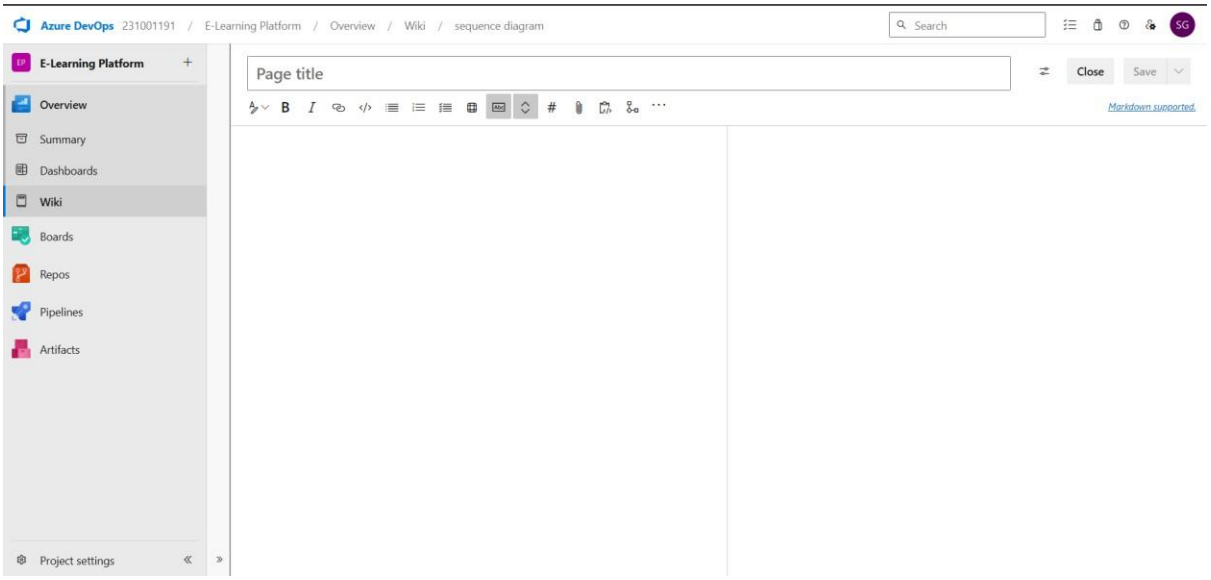
To design a Sequence Diagram by using Azure DevOps-WIKI

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



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

```
::: mermaid
sequenceDiagram
  participant Neighbor
    participant SafetyApp
    participant Police
    participant Resident
```

```
participant CommunityLeader

Neighbor->>SafetyApp: reportCrime()
SafetyApp->>Police: notify about new report
Police->>SafetyApp: markReportStatus()
SafetyApp->>CommunityLeader: update crime statistics
SafetyApp->>Resident: sendAlert()
```

Explanation:

participant defines the entities involved.

->> represents a direct message.

-->> represents a response message.

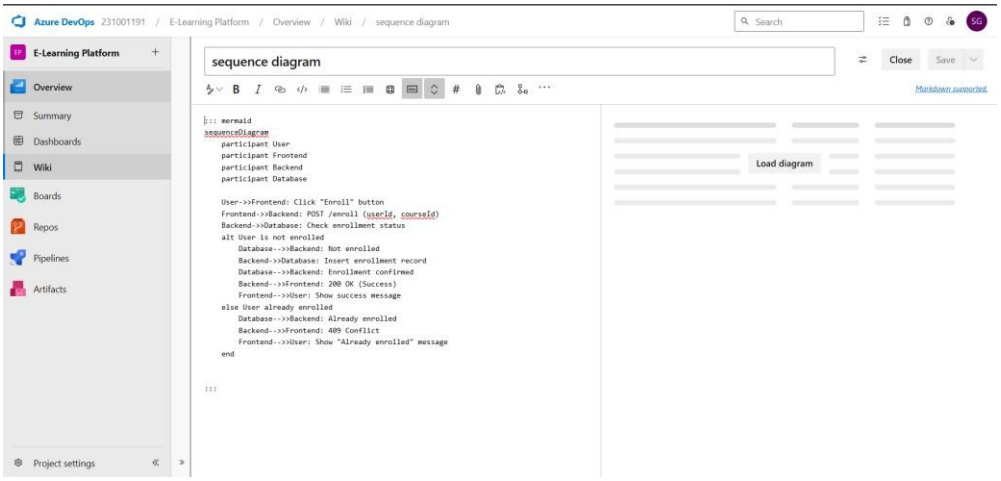
+ after ->> activates a participant.

- after -->> deactivates a participant.

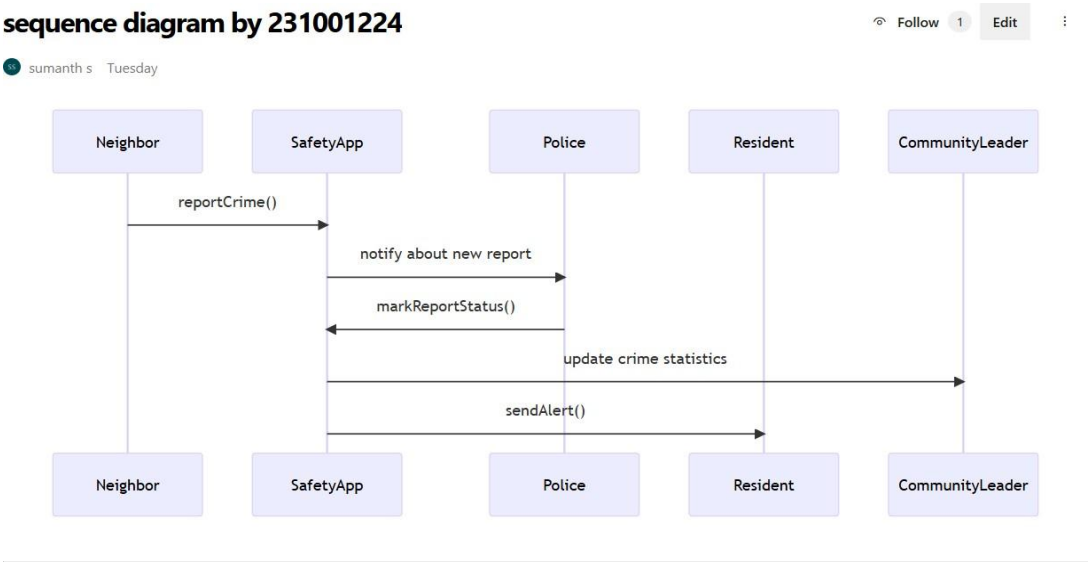
alt / else for conditional flows.

loop can be used for repeated actions.

->	Solid line without arrow
-->	Dotted line without arrow
->>	Solid line with arrowhead
-->>	Dotted line with arrowhead
<<->>	Solid line with bidirectional arrowheads (v11.0.0+)
<<-->>	Dotted line with bidirectional arrowheads (v11.0.0+)
-x	Solid line with a cross at the end
--x	Dotted line with a cross at the end
-)	Solid line with an open arrow at the end (async)
--)	Dotted line with a open arrow at the end (async)



4. click wiki menu and select the page



RESULT:

Thus, the sequence diagram was created successfully.

EX NO. 6

DATE:04/03/2025

DESIGNING CLASS DIAGRAM USING AZURE

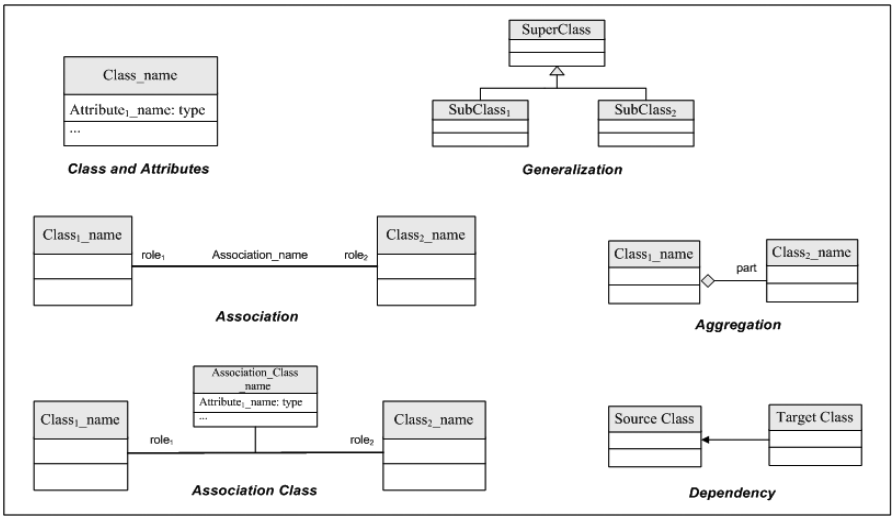
DEVOPS-WIKI

AIM:

To design a Class Diagram by using Azure DevOps-WIKI

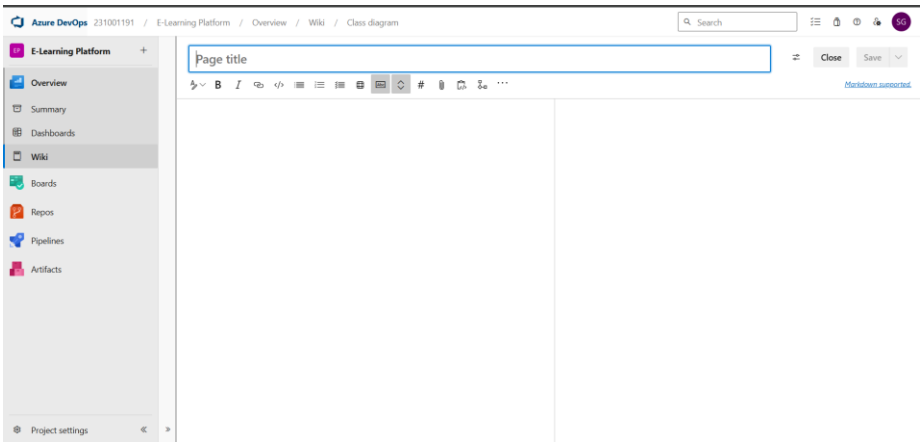
THEORY:

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.



PROCEDURE:

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



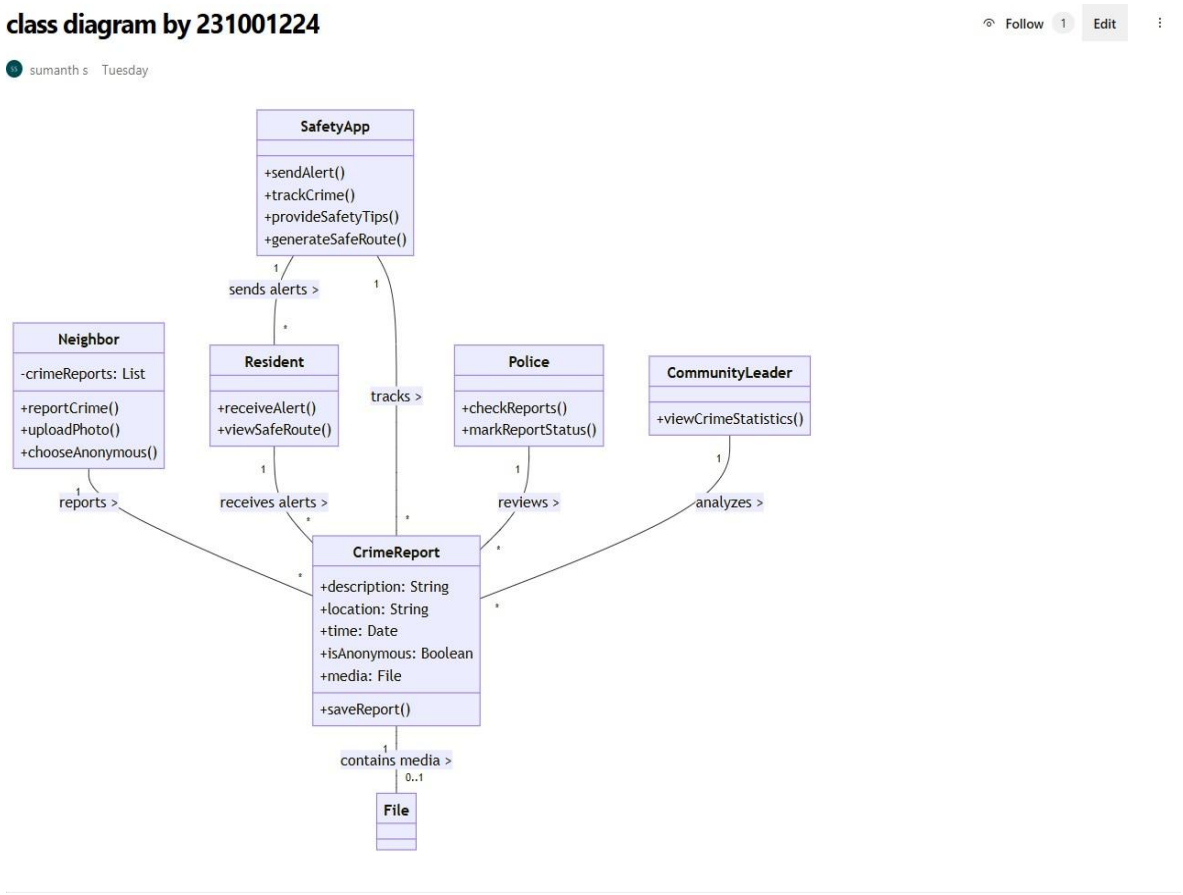
3. Write code for drawing class diagram and save the code

```
classDiagram
:::mermaid
classDiagram
    class Neighbor {
        +reportCrime()
        +uploadPhoto()
        +chooseAnonymous()
        -crimeReports: List<CrimeReport>
    }

    class CrimeReport {
        +description: String
        +location: String
        +time: Date
        +isAnonymous: Boolean
        +media: File
        +saveReport()
    }

    class Resident {
        +receiveAlert()
        +viewSafeRoute()
    }

    class Police {
        +checkReports()
```

```
    +markReportStatus()
}

class CommunityLeader {
    +viewCrimeStatistics()
}

class SafetyApp {
    +sendAlert()
    +trackCrime()
    +provideSafetyTips()
    +generateSafeRoute()
}

Neighbor "1" -- "*" CrimeReport : reports >
CrimeReport "1" -- "0..1" File : contains media >
Resident "1" -- "*" CrimeReport : receives alerts >
SafetyApp "1" -- "*" CrimeReport : tracks >
Police "1" -- "*" CrimeReport : reviews >
CommunityLeader "1" -- "*" CrimeReport : analyzes >
SafetyApp "1" -- "*" Resident : sends alerts >
```

Relationship Types

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



RESULT:

Thus, the class diagram was created successfully.

EX NO: 7

DATE:11/03/2025

DESIGNING USE CASE DIAGRAM USING AZURE DEVOPS-WIKI

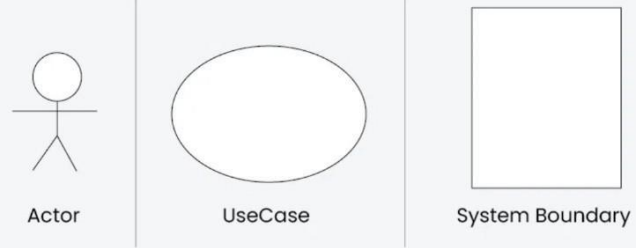
AIM:

Steps to draw the Use Case Diagram using Azure DevOps -WIKI

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: Open Azure DevOps Wiki

- Go to your Azure DevOps project.
- From the left sidebar, click on Wiki.

Step 2: Create or Edit a Wiki Page

- Click New Page to create a new page.
- Or, navigate to an existing page and click the Edit (, button.

Step 3: Insert a Mermaid Code Block

Use a fenced code block with the mermaid keyword:

::: mermaid

graph TD

actorUser([User])

actorInstructor([Instructor])

actorAdmin([Admin])

actorUser --> UC1[Register/Login]

actorUser --> UC2[Browse Courses]

actorUser --> UC3[Enroll in Course]

actorUser --> UC4[Attend Live Class]

actorUser --> UC5[Take Quiz]

actorUser --> UC6[Download Certificate]

actorInstructor --> UC1

actorInstructor --> UC7[Create Course]

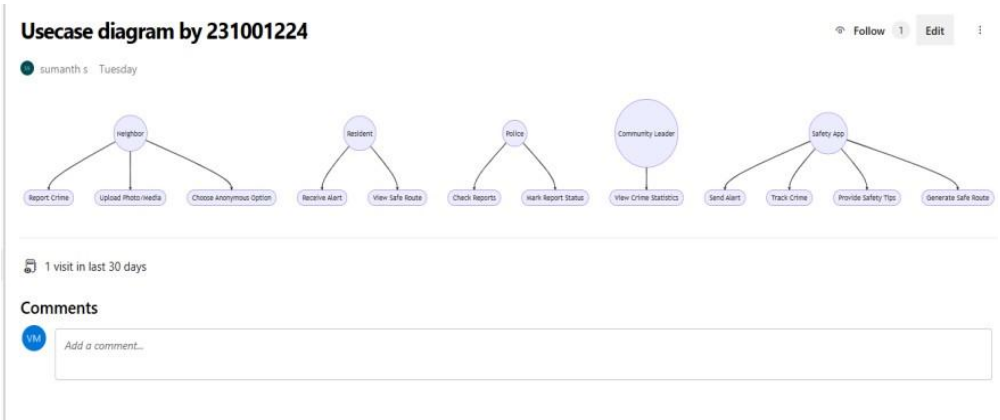
actorInstructor --> UC8[Create Quiz]

actorAdmin --> UC1

actorAdmin --> UC9[Approve Course]

actorAdmin --> UC10[Manage Users]

:::



RESULT:

The Use Case diagram was designed successfully.

EX NO. 8

DATE:18/03/2025

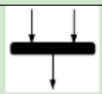
DESIGNING ACTIVITY DIAGRAMS USING AZURE DEVOPS-WIKI

AIM:

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

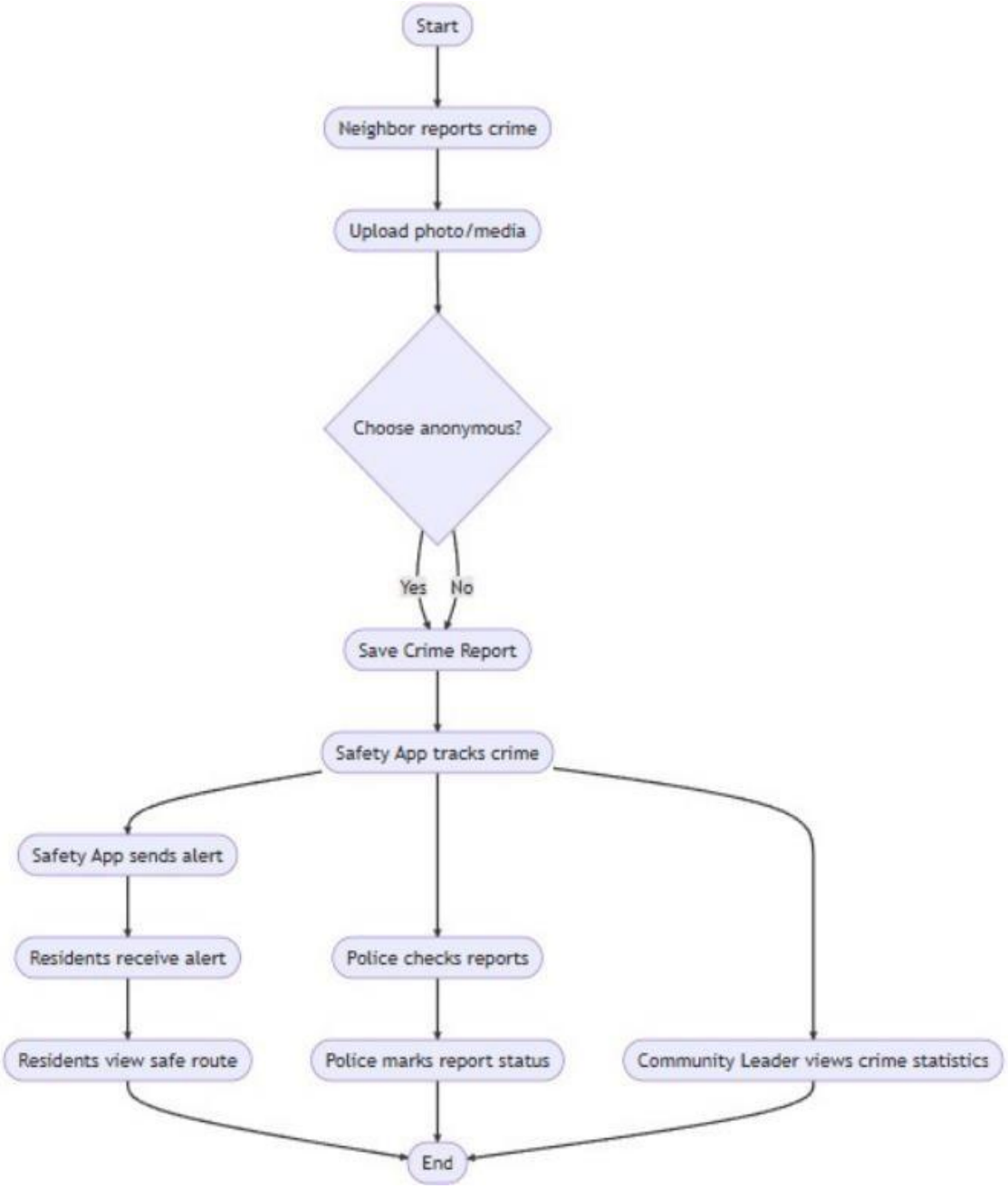
THEORY:

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 o 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:

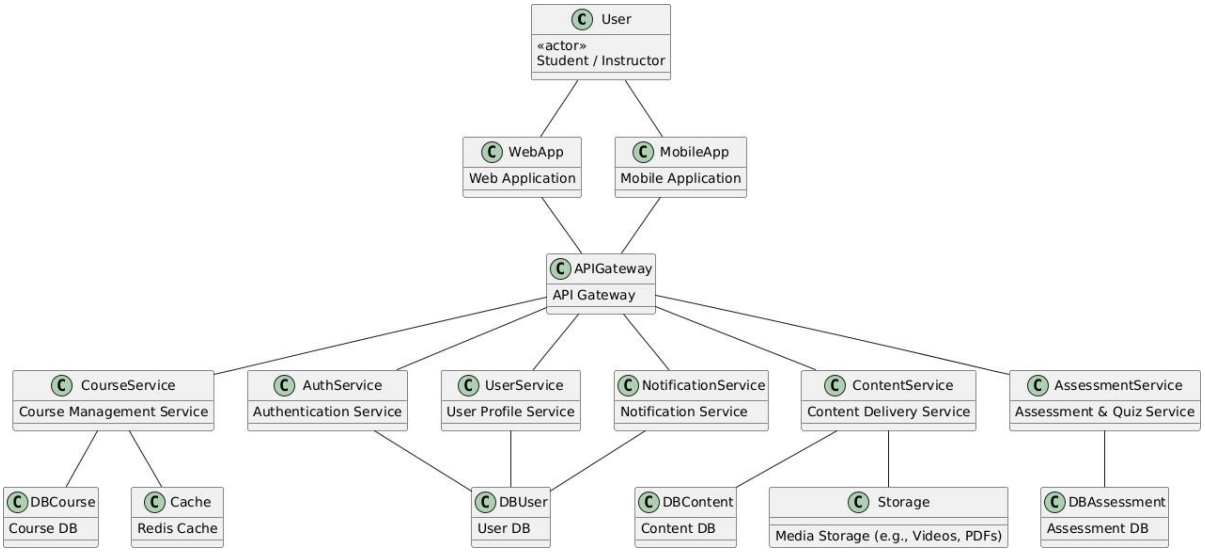
Steps to draw the Architecture Diagram using Star UML

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.

PROCEDURE:

1. Draw diagram in Star UML



RESULT:

The architecture diagram was designed successfully.

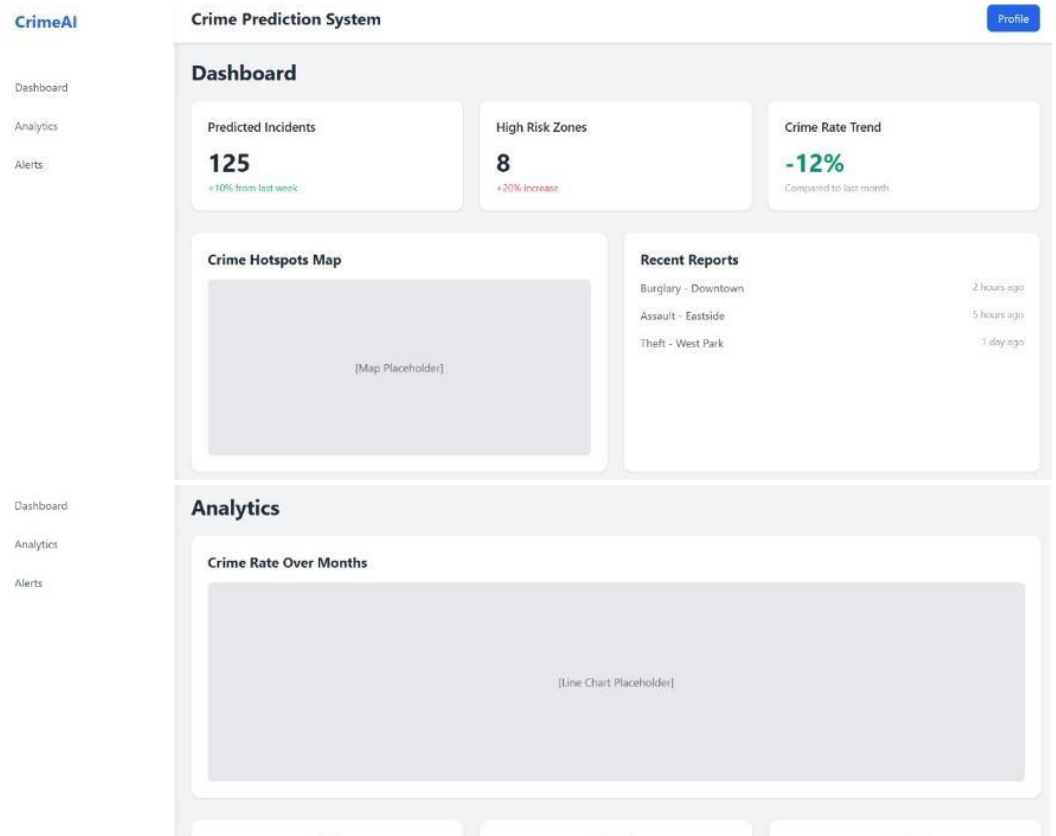
EX NO. 10

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.

EXP NO : 12

DATE : 15.04.2025

TESTING – TEST PLAN, TEST CASE, 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 behaviors 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.

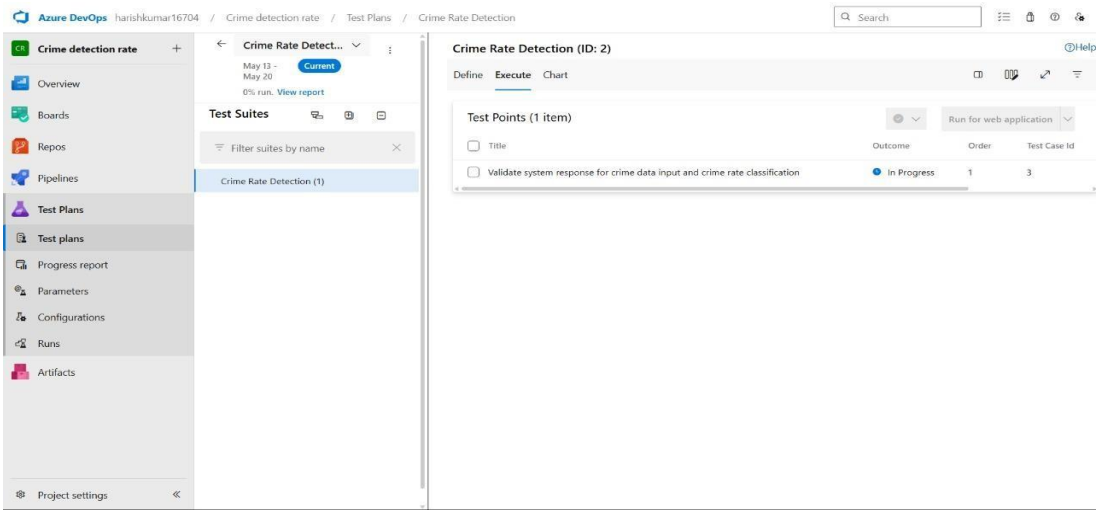
7. Separate Test Suites

- 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.

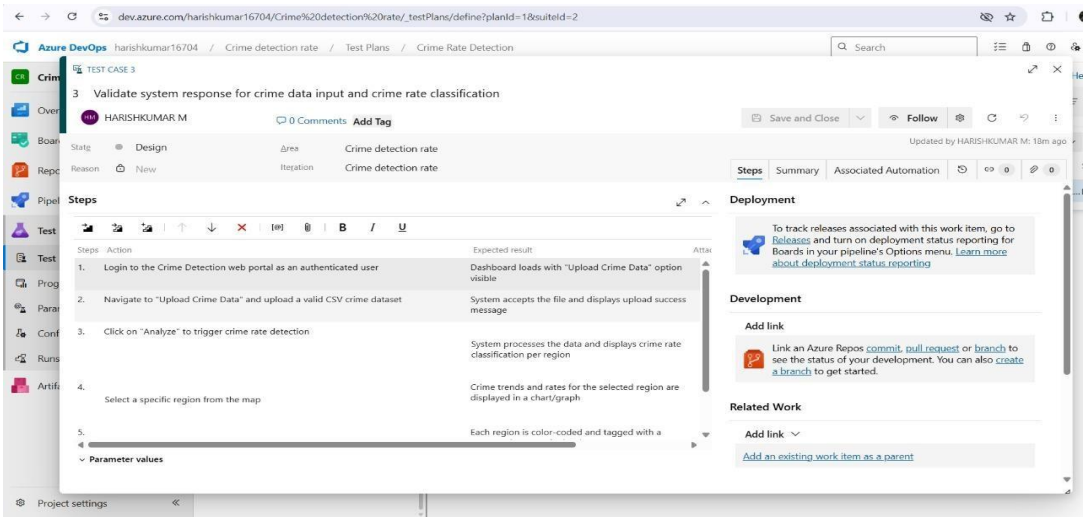
8. Prioritize and Review

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

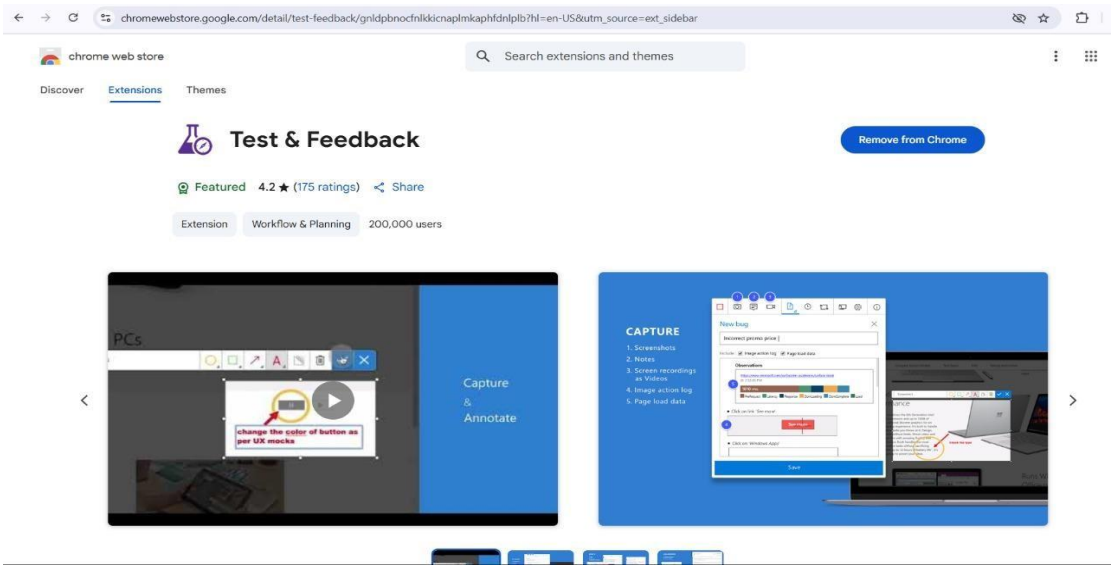
1. **New test plan**



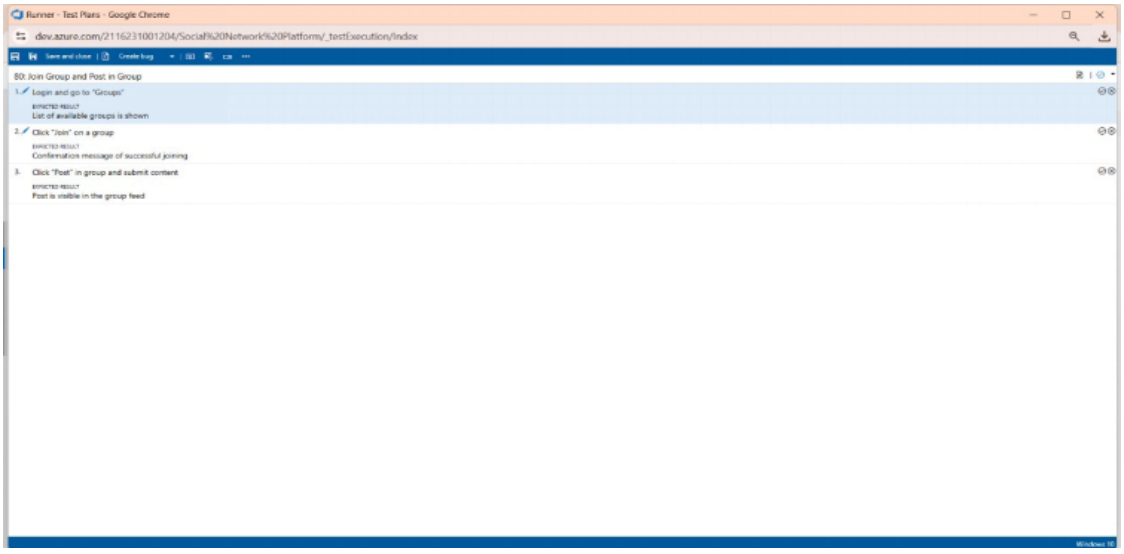
2. **Test case**



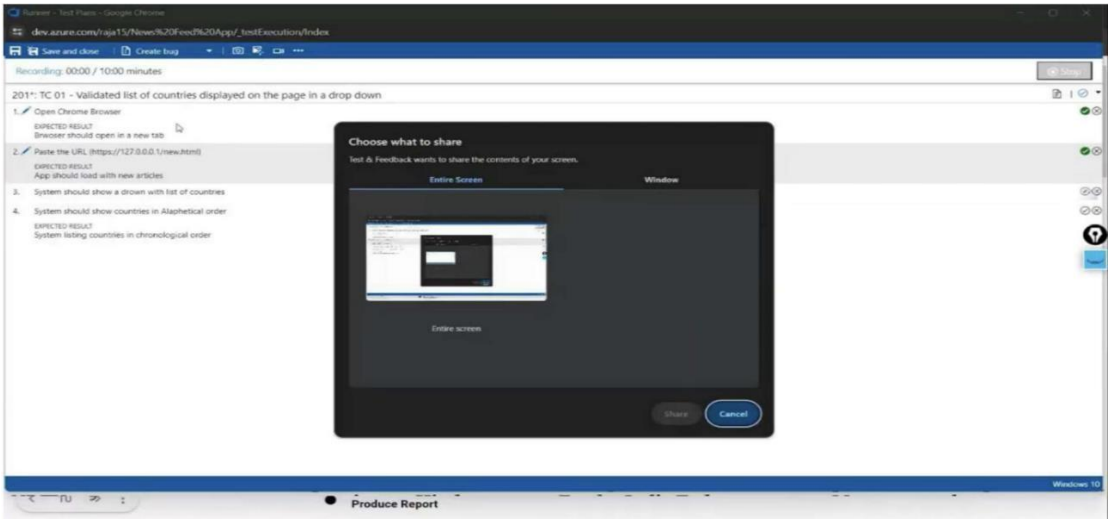
3. Installation of Test



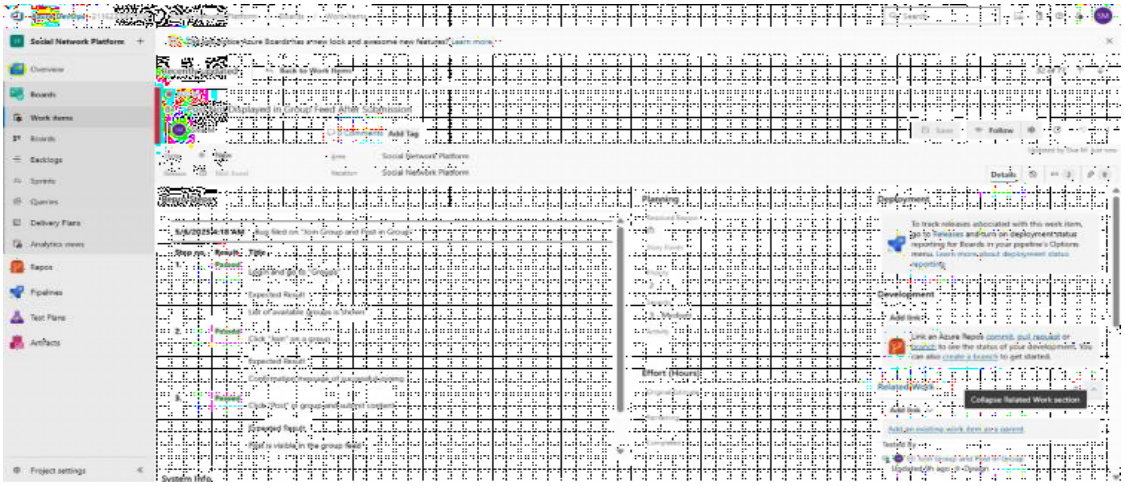
4. Running the Test Cases



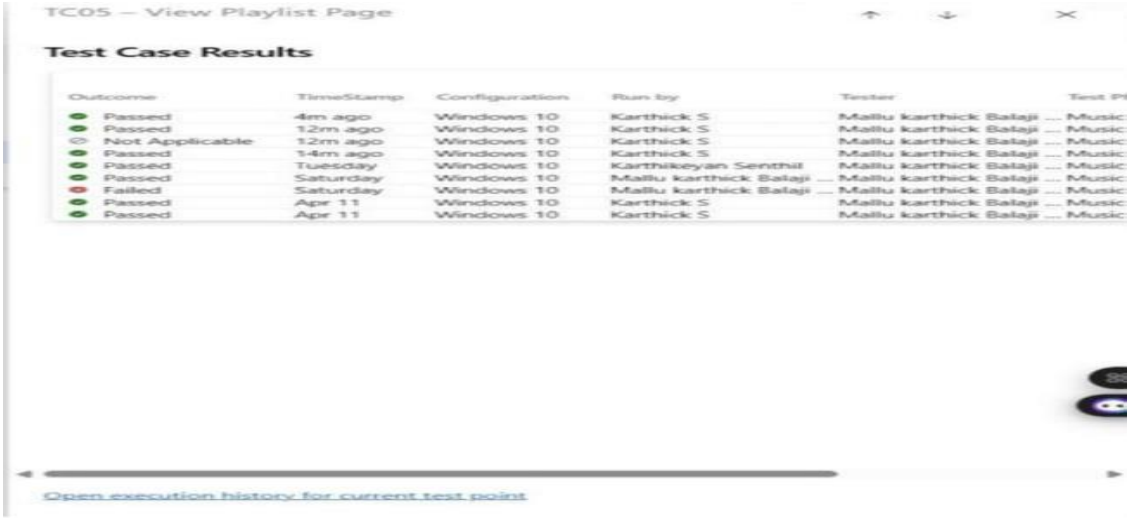
5. Recording the Test Cases



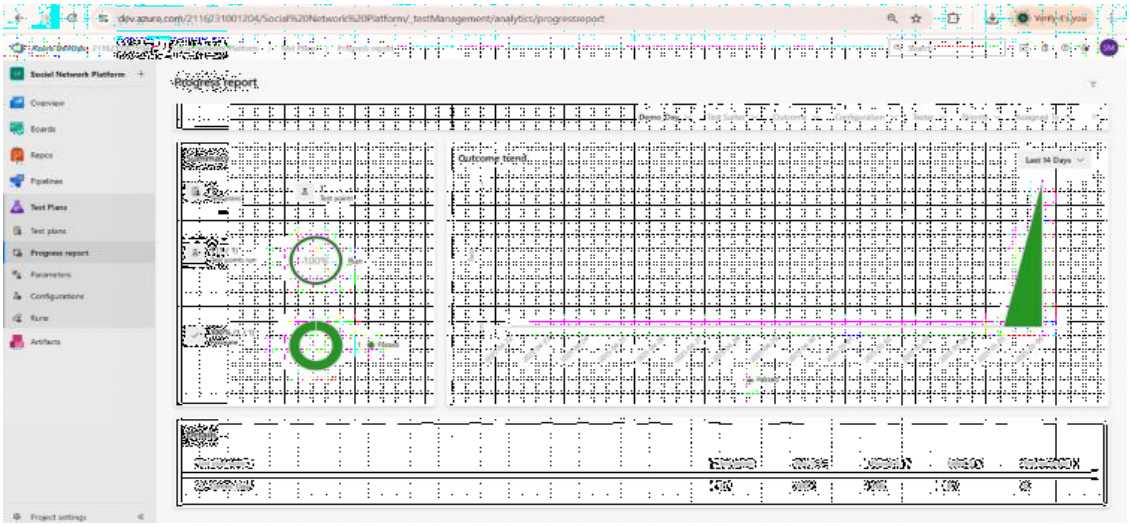
6. Creating Bugs



7. Test Case Results



6. 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.

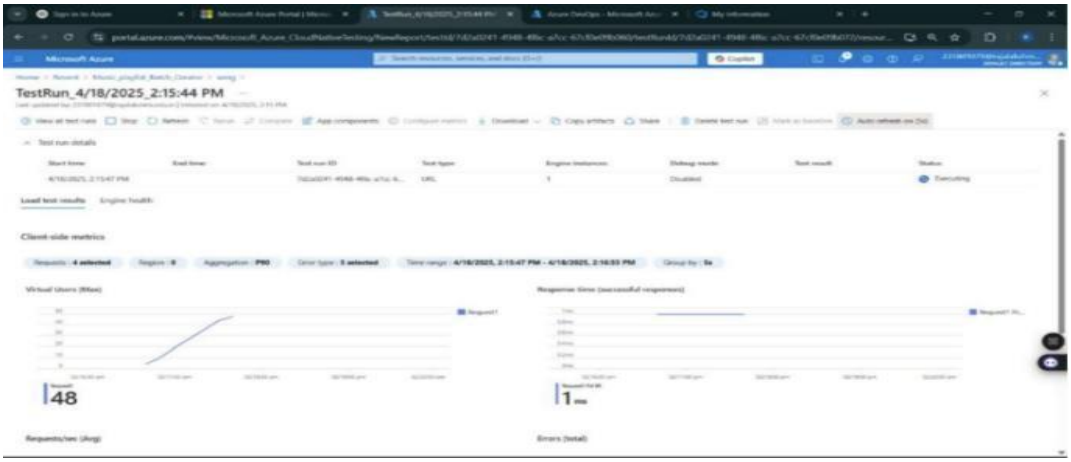
2. Create the Resource
   - Go to Create a resource — Search for "Azure Load Testing".
   - Select Azure Load Testing and click Create.
3. Fill in the Configuration Details
   - Subscription: Choose your Azure subscription.
   - Resource Group: Create new or select an existing one.
   - Name: Provide 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.

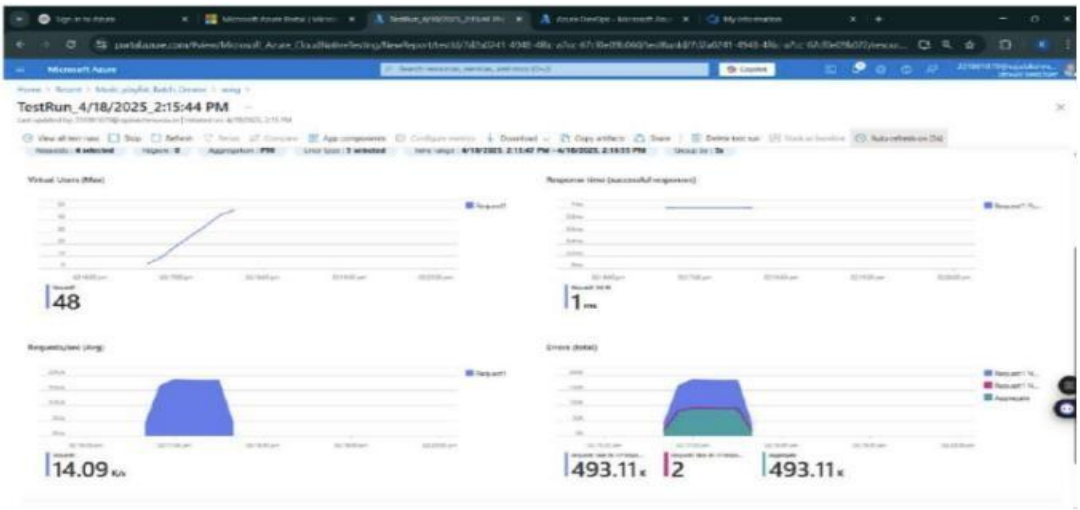
Steps to Create and Run a Load Test:

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.