



**RAJALAKSHMI  
ENGINEERING COLLEGE**

An AUTONOMOUS Institution  
Affiliated to ANNA UNIVERSITY, Chennai

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**ACADEMIC YEAR 2024-2025**

**EVEN SEMESTER**



**CS23432 SOFTWARE ENGINEERING**

**LAB MANUAL**

**SECOND YEAR**

**FOURTH SEMESTER**

**2024- 2025**

**EVEN SEMESTER**

Ex No	List of Experiments
1	Study of Azure DevOps
2	Designing Project using AGILE-SCRUM Methodology.
3	Agile Planning
4	User stories – Creation
5	Architecture Diagram Using AZURE
6	Designing Usecase and Class Diagram
7	Designing Interaction Diagrams
8	Design Interface
9	Implementation – Design a Web Page based on Scrum Methodology
10	Testing using Azure.
11	Deployment

Requirements	
Hardware	Intel i3, CPU @ 1.20GHz 1.19 GHz, 4 GB RAM, 32 Bit Operating System
Software	StarUML , Azure

## LAB PLAN

### CS19442-SOFTWARE ENGINEERING LAB

Ex No	Date	Topic	Page No	Sign
1		Study of Azure DevOps		
2		Writing Problem Statement		
3		Designing Project using AGILE-SCRUM Methodology by using Azure.		
4		Agile Planning		
5		User stories – Creation		
6		Architecture Diagram Using AZURE		
7		Designing Usecase Diagram using StarUML		
8		Designing Activity Diagrams using StarUML		
9		Designing Sequence Diagrams using StarUML		
10		Design Class Diagram		
10		Design User Interface		
11		Implementation – Design a Web Page based on Scrum Methodology		
12		Testing		
13		Deployment		

## Course Outcomes (COs)

Course Name: Software Engineering

Course Code: CS23432

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

### CO - PO – PSO matrices of course

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

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:**

### **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 bug.

### **Result:**

The study was successfully completed.

**EX NO: 2**

**DATE:**

## **PROBLEM STATEMENT**

**AIM:**

To prepare PROBLEM STATEMENT for your given project.

**Problem Statement:**

### **CRIME FORESIGHT DASHBOARD**

Communities deserve to feel safe, but crime prevention often falls short due to delayed responses and scattered information. Law enforcement teams face challenges in identifying crime patterns early, predicting potential hotspots, and deploying resources where they're needed most. Without real-time insights and reliable forecasting tools, crime trends go unnoticed until it's too late. On the other side, the public remains unaware of risks in their surroundings. To bridge this gap, there's a growing need for a smart, data-driven system that not only tracks and analyzes crime data but also predicts future incidents. By combining machine learning with socio-economic insights and delivering it all through a user-friendly dashboard, this system empowers authorities to act before problems escalate — creating safer, more informed, and better-prepared communities.

**Result:**

The problem statement was written successfully.

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



**Result:**

Thus the Agile plan was completed successfully.

## EX NO: 4      CREATE USER STORIES

DATE:

Aim:

To create User Stories

### Theory:

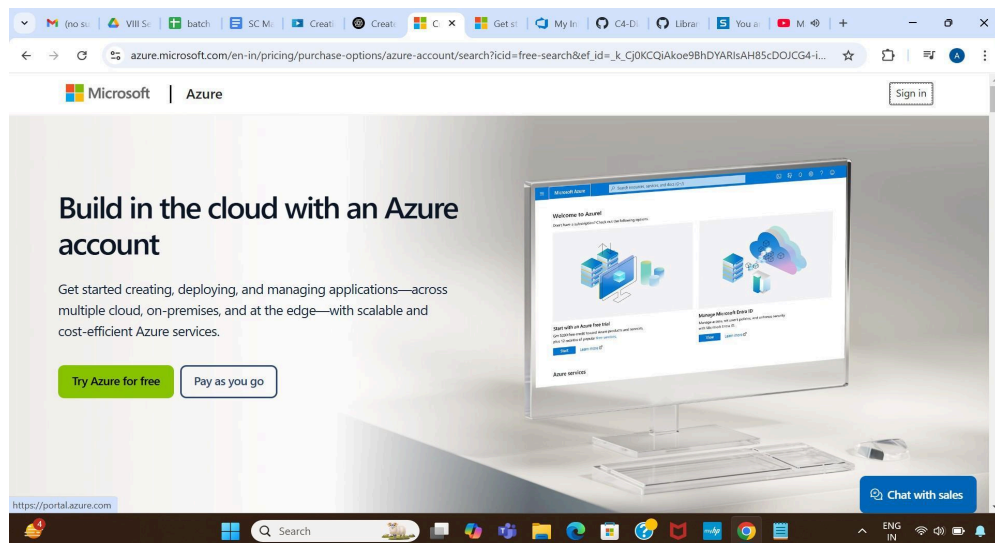
A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer.

User story template

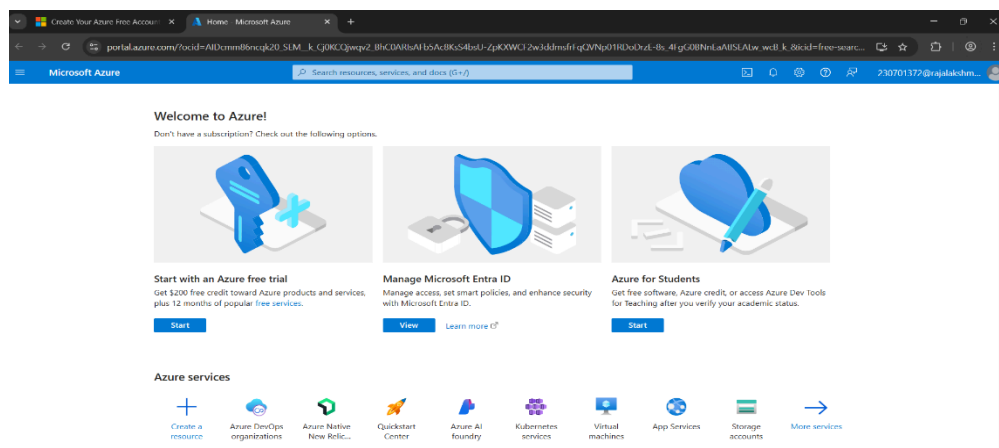
**"As a [role], I [want to], [so that]."**

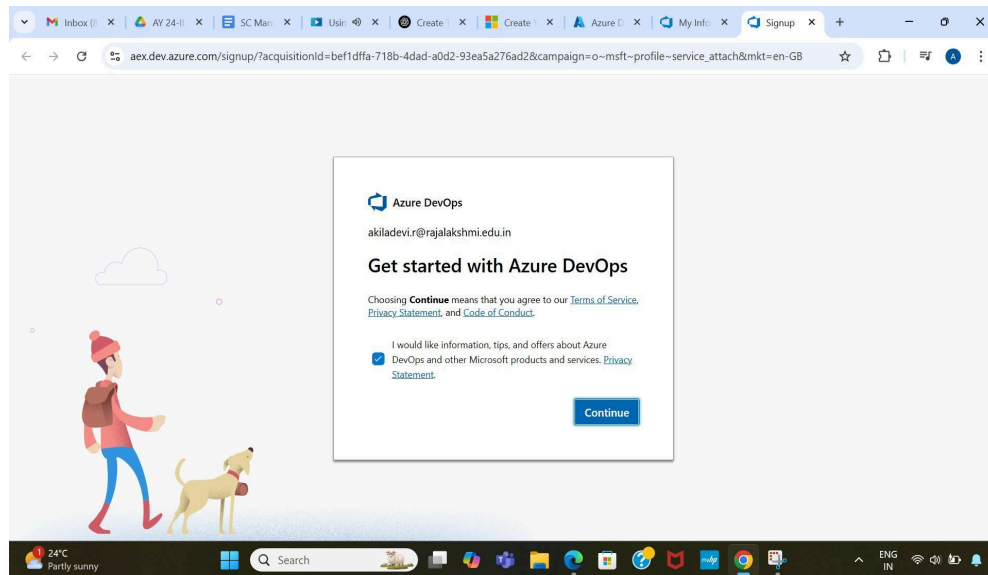
### Procedure:

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>

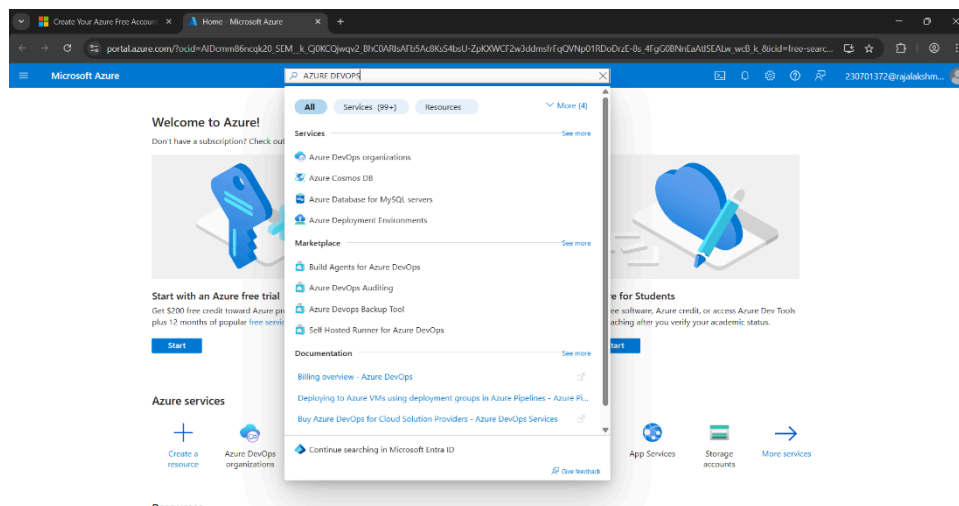


### 3. Azure home page

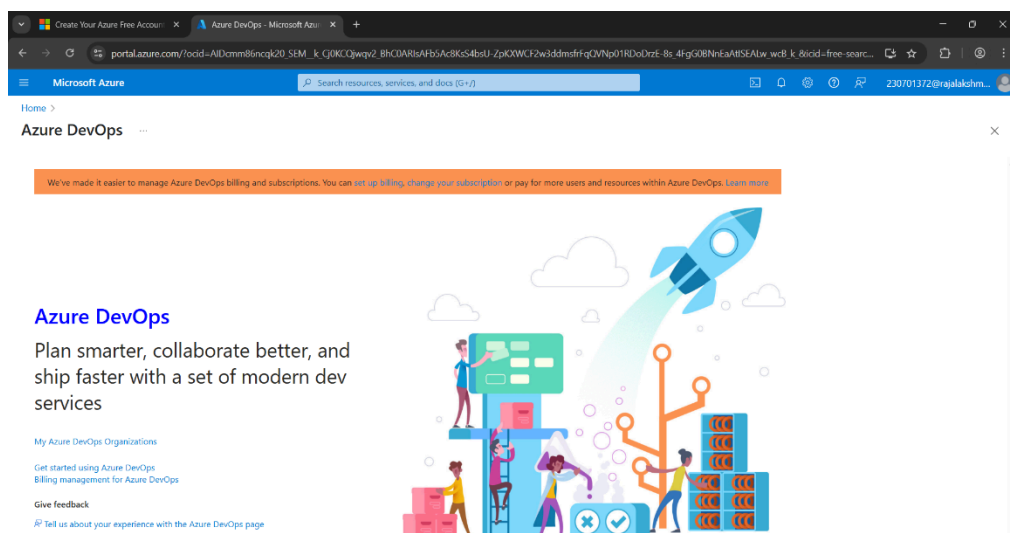


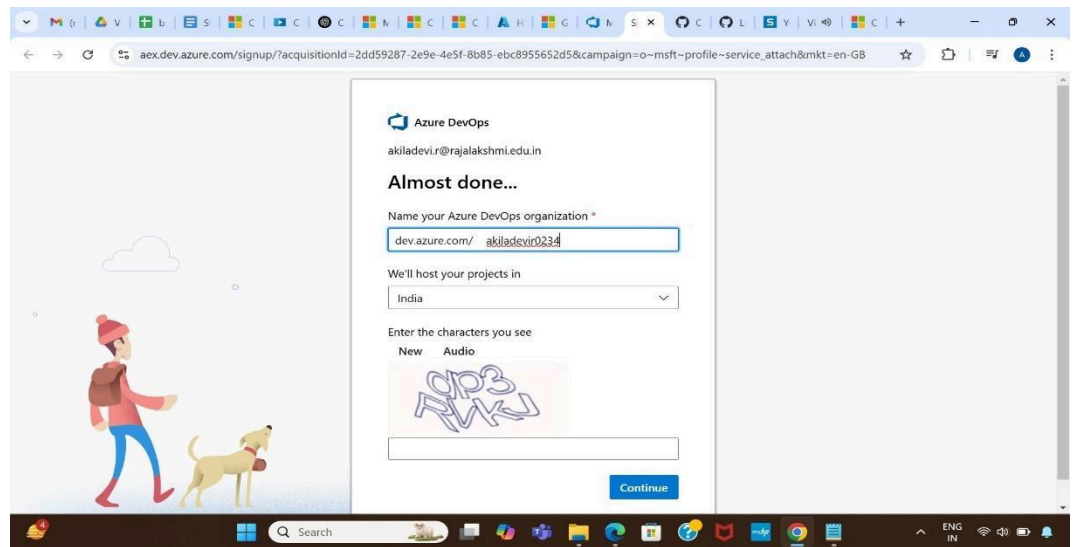


4. Open DevOps environment in the Azure platform by typing Azure DevOps Organizations in the search bar.



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.





## 7. 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.

Create new project

Project name \*

Description

Visibility

Public

Anyone on the internet can view the project. Certain features like TFVC are not supported.

Private

Only people you give access to will be able to view this project.

Public projects are disabled for your organization. You can turn on public visibility with [organization policies](#).

Advanced

Version control ②

Git

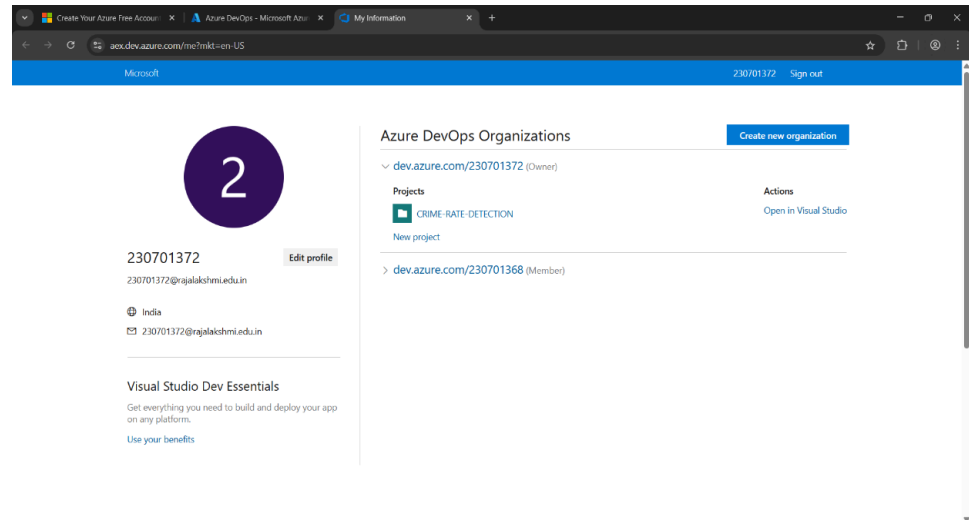
Work item process ②

Agile

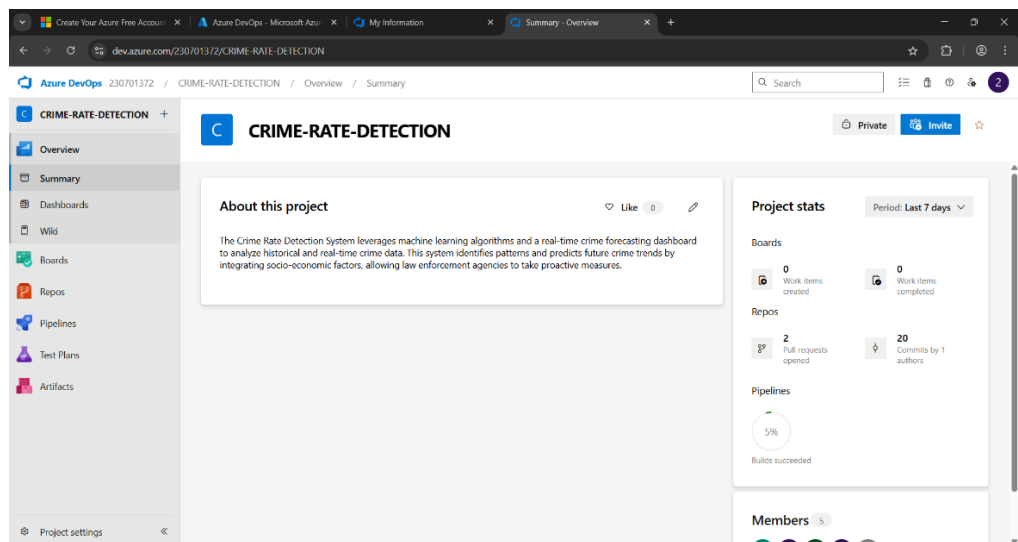
Cancel

Create

8. 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.

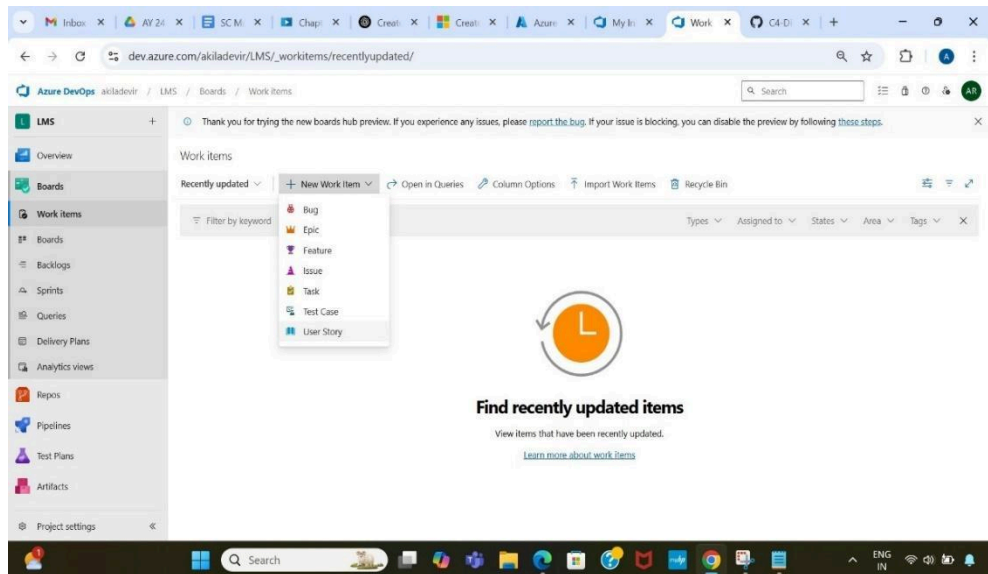


#### 4. Project Dashboard

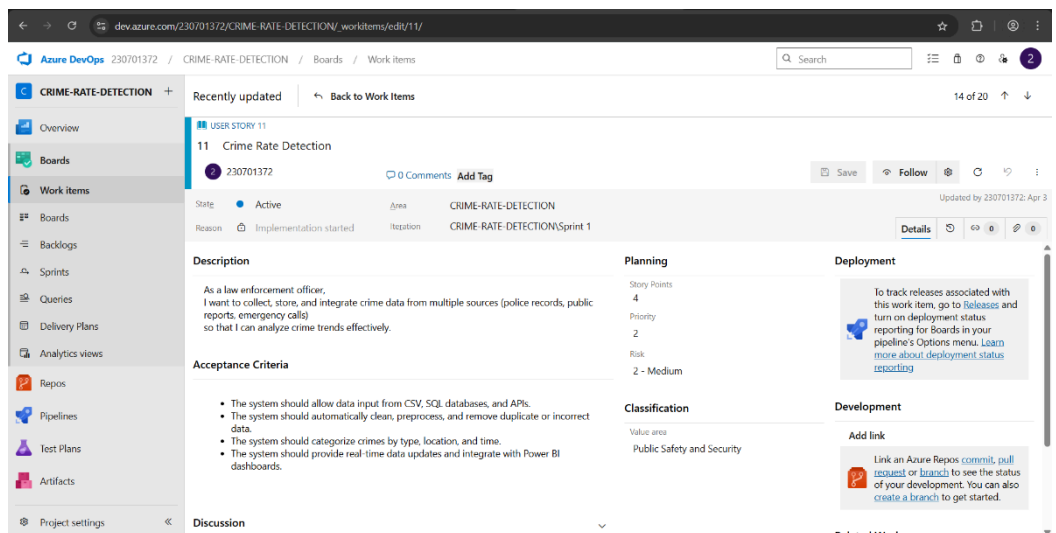


## 5. To manage user stories

- 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.



## 6. Fill in User Story Details



**Result:**

The user story was written successfully.



**DATE:****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
3. Write code for drawing sequence diagram and save the code.

```
sequenceDiagram
```

```
sequenceDiagram
```

```
participant User
```

```
participant LoginPage
```

```
participant Dashboard
```

```
participant Analytics
```

```
participant MapService
```

```
participant ReportsDB
```

```
%% --- Step 1: Authentication Phase ---
```

```
rect rgb(240, 240, 255)
```

```
User->>LoginPage: Enter username & password
```

```
LoginPage->>Dashboard: Authenticate user
```

```
Dashboard-->>User: Redirect to dashboard
```

```
end
```

```
%% --- Step 2: Dashboard Loads System Data ---
```

```
rect rgb(230, 255, 230)
```

```
Dashboard->>Analytics: Fetch stats (Crime Rate, Accuracy)
```

```
Dashboard->>ReportsDB: Get recent crime incidents
```

```
Dashboard->>MapService: Load hotspot map
```

```
MapService-->>Dashboard: Error (Missing API Key)
```

```
end
```



```

%% --- Step 3: Dashboard Display to User ---
rect rgb(255, 250, 230)
User->>Dashboard: View Charts & Predictions
Dashboard-->>User: Display crime trends, charts, predictions
end

%% --- Step 4: Deep Dive into Reports & Analysis ---
rect rgb(255, 240, 240)
User->>ReportsDB: Navigate to detailed reports
ReportsDB-->>User: Show reports (Cybercrime, Robbery, etc.)

User->>Analytics: Open detailed analytics
Analytics-->>User: Show Total Crime, Violent Crime, Accuracy, Hotspots
End

```

### 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 an open arrow at the end (async)

#### 4. Click wiki menu and select the page

The screenshot shows the Azure DevOps Wiki page for the 'SEQUENCE Diagram' in the 'CRIME-RATE-DETECTION' project. The page is titled 'SEQUENCE Diagram' and contains a Mermaid code block and a visual representation of the diagram.

**Mermaid Code:**

```
sequenceDiagram
    participant User
    participant LoginPage
    participant Dashboard
    participant Analytics
    participant MapService
    participant ReportDB

    %% --- Step 1: Authentication Phase ---
    rect rgb(240, 240, 255)
    User->>LoginPage: Enter username & password
    LoginPage->>Dashboard: Authenticate user
    Dashboard->>User: Redirect to dashboard
    end

    %% --- Step 2: Dashboard Loads System Data ---
    rect rgb(230, 255, 230)
    Dashboard->>Analytics: Fetch stats (Crime Rate, Accuracy)
    Dashboard->>ReportDB: Get recent crime incidents
    Dashboard->>MapService: Load hotspot map
    MapService-->>Dashboard: Error (Missing API Key)
    end

    %% --- Step 3: Dashboard Display to User ---
    rect rgb(255, 250, 230)
    User->>Dashboard: View Charts & Predictions
    Dashboard->>User: Display crime trends, charts, predictions
    end

    %% --- Step 4: Deep Dive into Reports & Analysis ---
    User->>ReportDB: Navigate to detailed reports
    ReportDB-->>User: Show reports (ColarCrime, Robbery, etc.)
    User->>Analytics: Open detailed analytics
    Analytics-->>User: Show Total Crime, Victim Crime, Accuracy, Hotspots
```

The visual representation of the diagram shows the following sequence of events:

- Authentication Phase:** User enters username and password on the LoginPage, which authenticates the user and redirects to the Dashboard.
- Dashboard Loads System Data:** The Dashboard fetches stats (Crime Rate, Accuracy) from Analytics, gets recent crime incidents from ReportDB, and loads a hotspot map from MapService. MapService returns an error (Missing API Key) to the Dashboard.
- Dashboard Display to User:** The User views charts and predictions on the Dashboard, which displays crime trends, charts, and predictions.
- Deep Dive into Reports & Analysis:** The User navigates to detailed reports on the ReportDB, which shows reports (ColarCrime, Robbery, etc.). The User then opens detailed analytics on the Analytics page, which shows total crime, victim crime, accuracy, and hotspots.

#### Result:

The sequence diagram was drawn successfully.

## EX NO. 6

DATE :

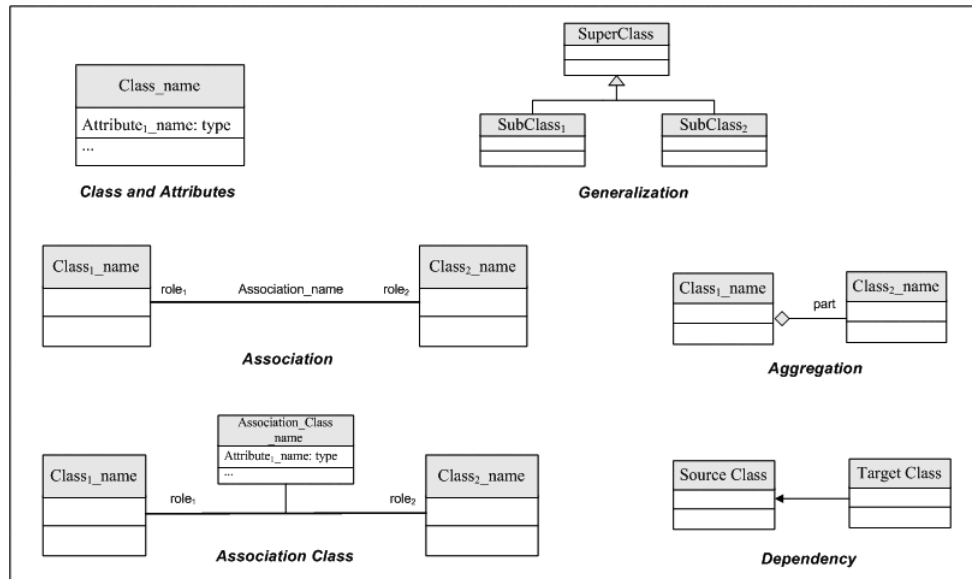
## CLASS DIAGRAM

### 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

```
classDiagram
::: mermaid
classDiagram
class User {
+String username
+String password
+String role
+login()
+viewDashboard()
+viewReports()
+viewAnalytics()
}

class LoginPage {
+String sessionID
+authenticate()
+redirectToDashboard()
}
```

```

class Dashboard {
    +Float totalCrimeRate
    +Float predictionAccuracy
    +loadStats()
    +loadMap()
    +loadReports()
    +displayTrends()
}

class Analytics {
    +Integer totalCrimes
    +Integer violentCrimes
    +fetchCrimeRate()
    +fetchPredictionAccuracy()
    +showDetailedStats()
}

class ReportsDB {
    +List incidentList
    +getRecentIncidents()
    +getCrimeReports()
}

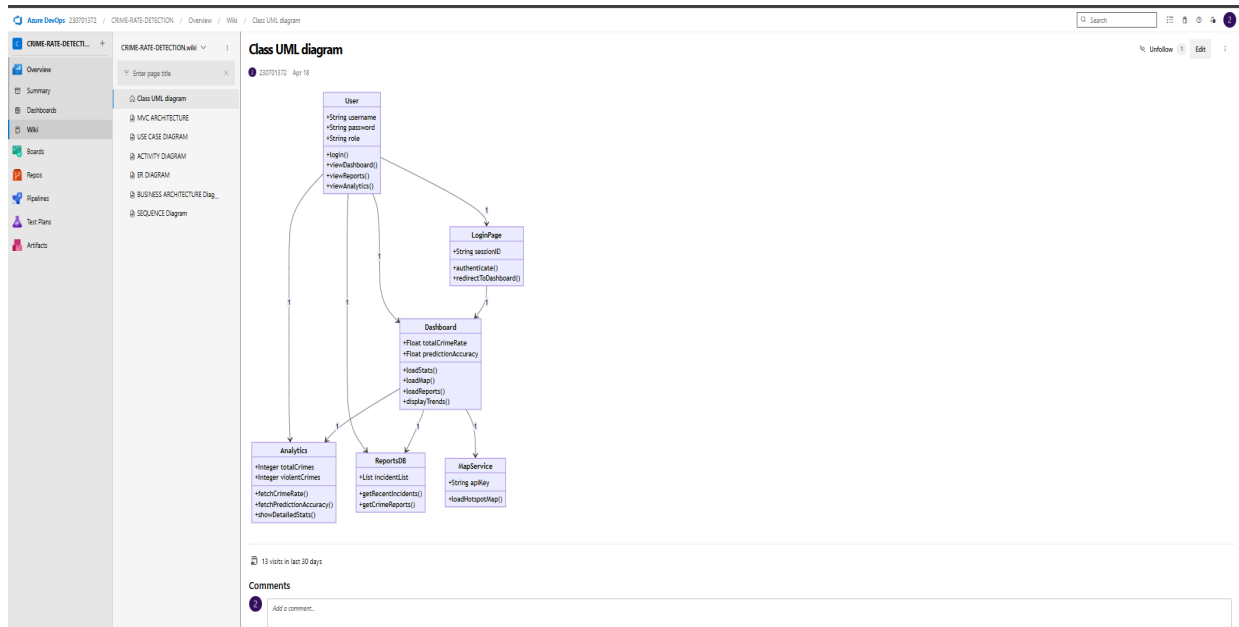
class MapService {
    +String apiKey
    +loadHotspotMap()
}

%% Relationships (1-to-1 style, straight lines)
User --> LoginPage : 1
LoginPage --> Dashboard : 1
User --> Dashboard : 1
Dashboard --> Analytics : 1
Dashboard --> ReportsDB : 1
Dashboard --> MapService : 1
User --> ReportsDB : 1
User --> Analytics : 1

```

## Relationship Types

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



Visit : <https://mermaid.js.org/syntax/classDiagram.html>

## Result:

The use case diagram was designed successfully.

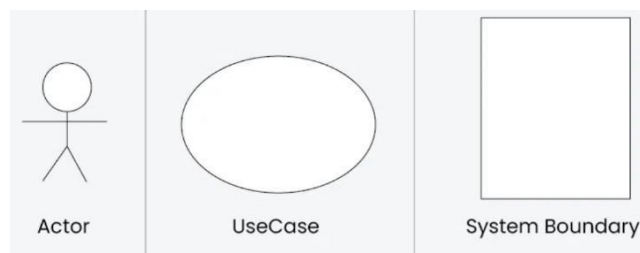
**DATE:****Aim:**

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:
- ![Use Case Diagram](attachments/use\_case\_diagram.png)

### 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.

## USE CASE DIAGRAM

2 230701372 May 3



### Result:

The use case diagram was designed successfully

## EX NO. 8

DATE:



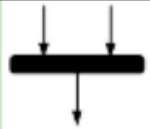


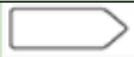





## ACTIVITY DIAGRAM

### Aim :-

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

### Theory:-

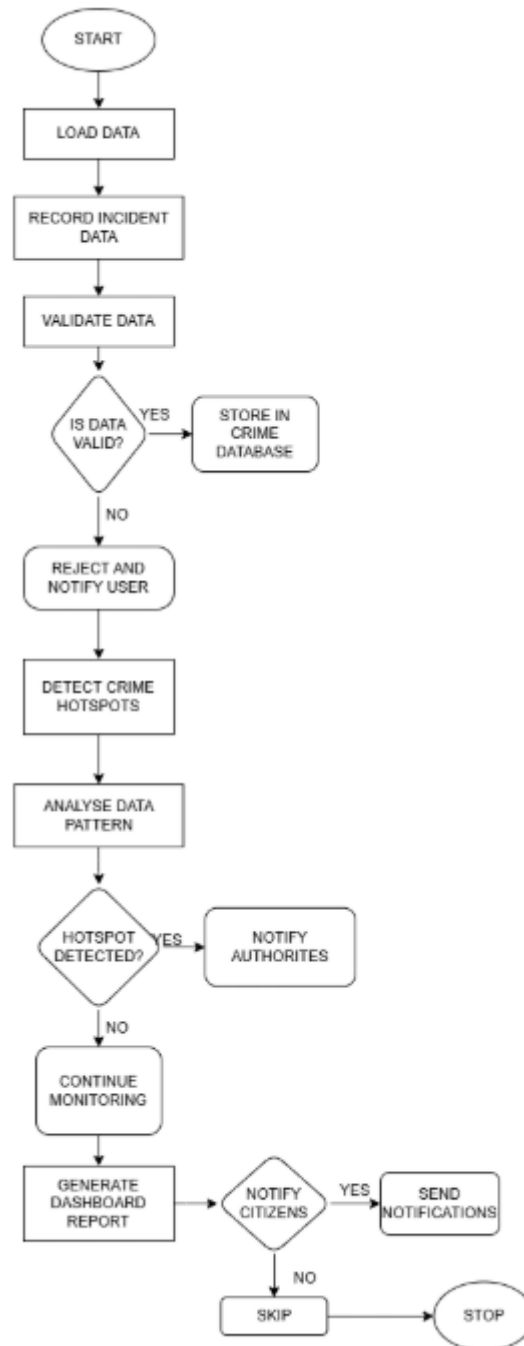
Activity diagrams are an essential part of the Unified Modelling Language (UML) that help visualize workflows, processes, or activities within a system. They depict how different actions are connected and how a system moves from one state to another.

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:**

## **ARCHITECTURE DIAGRAM**

### **Aim:**

Steps to draw the Architecture Diagram using draw.io.

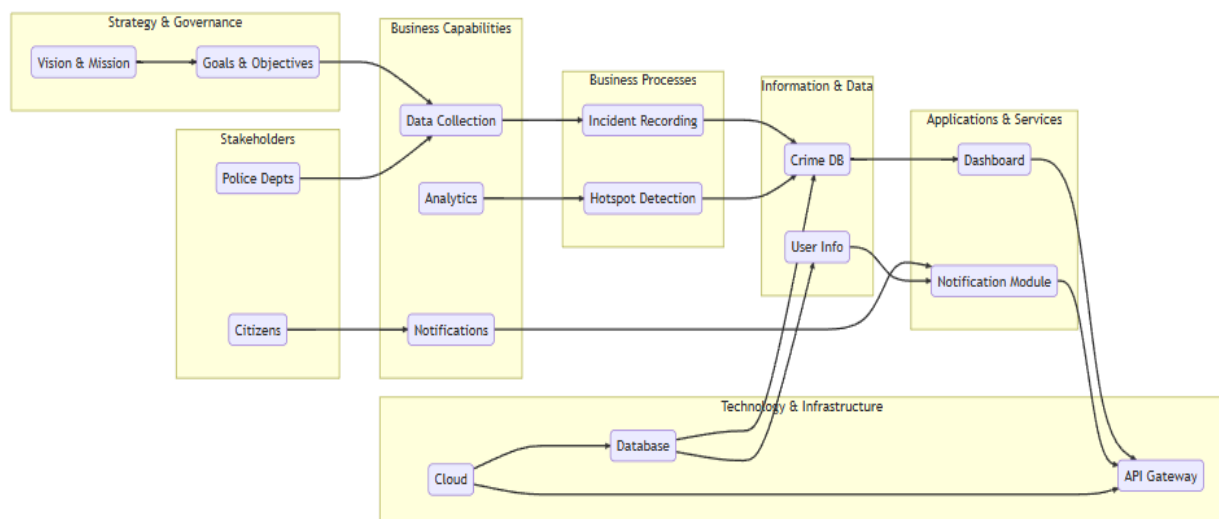
### **Theory:**

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.

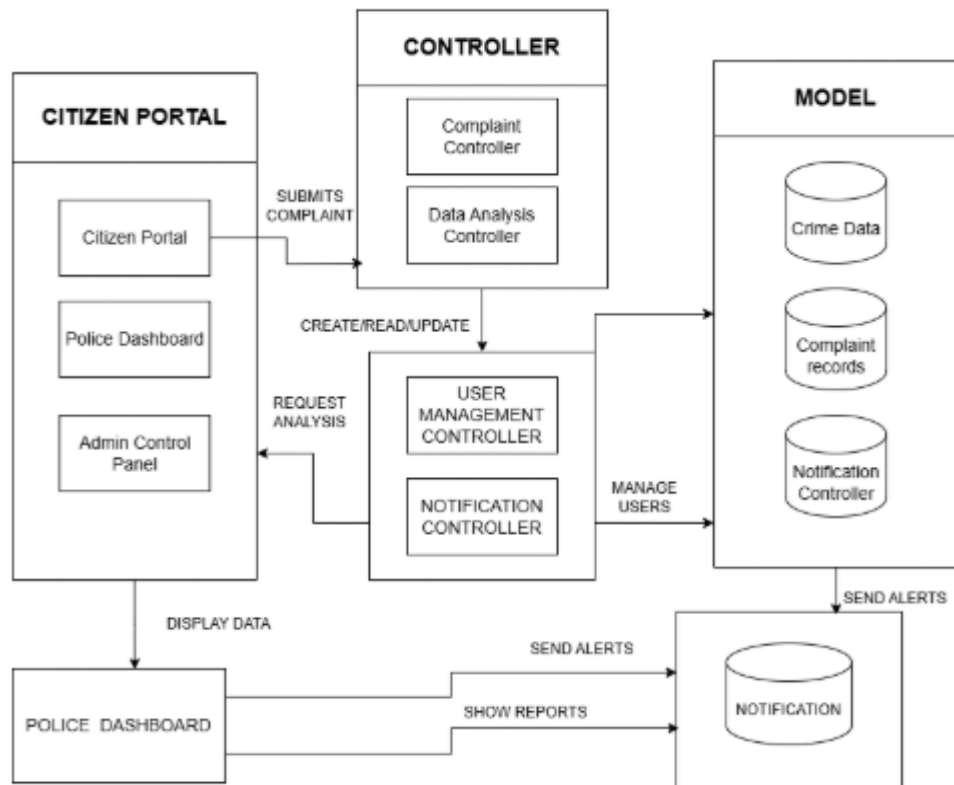


### **Procedure:-**

1. Draw diagram in draw.io



2. Upload the diagram in Azure DevOps wi



### Result:

The architecture diagram was designed successfully

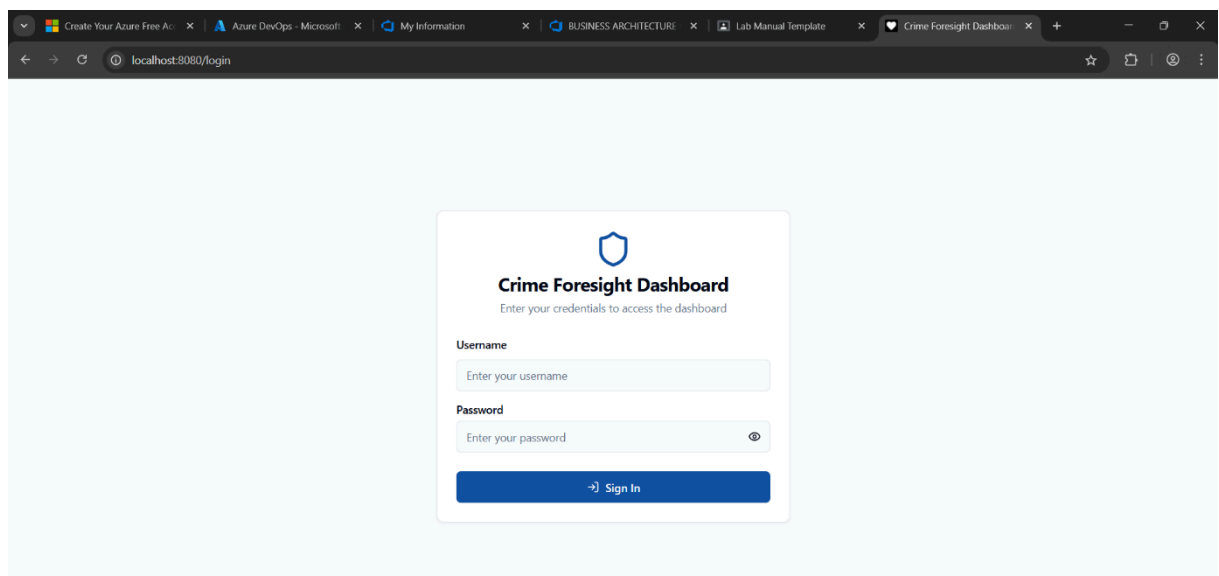
**EX NO. 10**

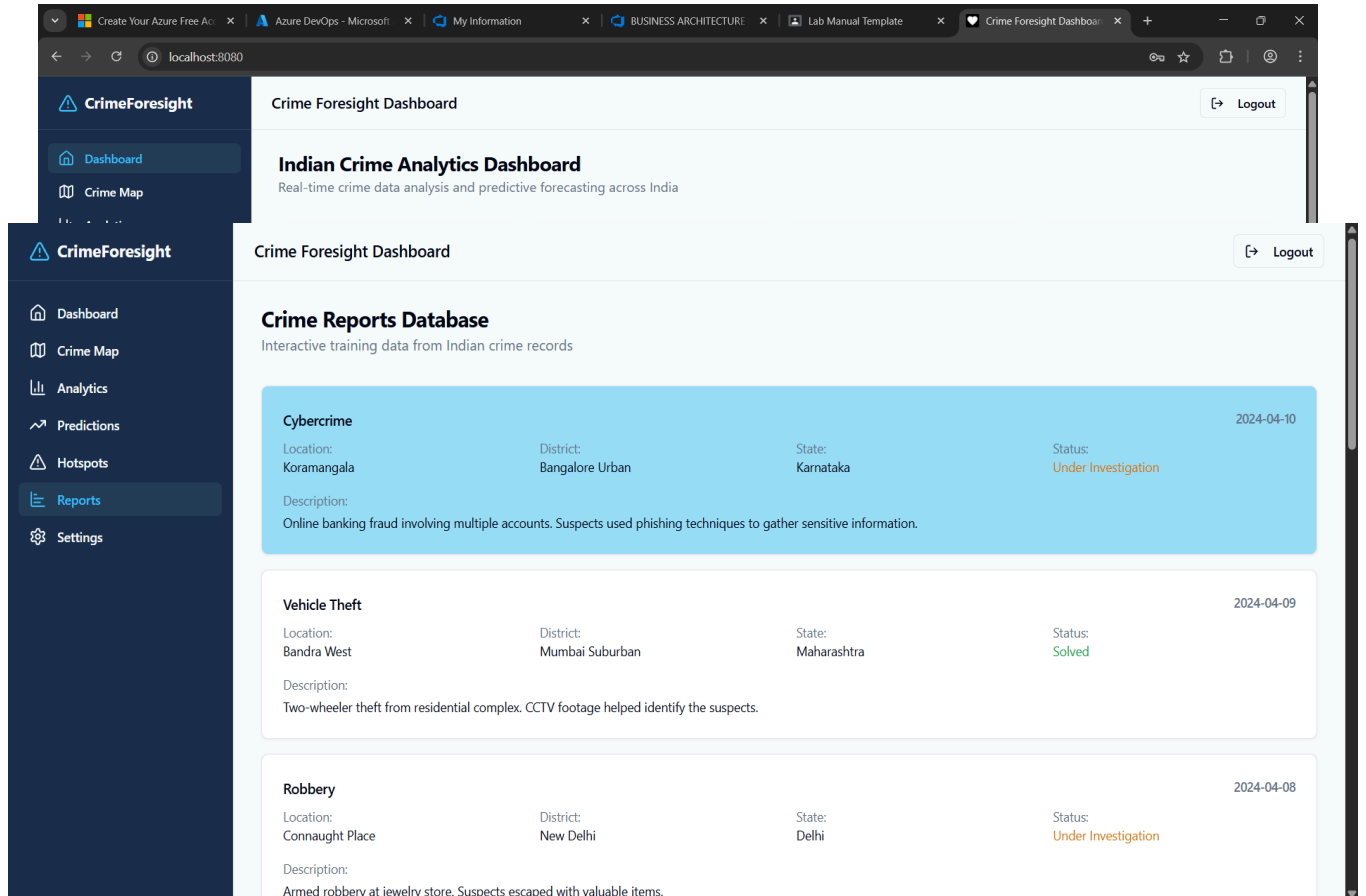
**DATE:**

**USER INTERFACE**

### Aim:

Design User Interface for the given project





## Result:

The UI was designed successfully.

**DATE:****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:
  name: Default

steps:
  # Checkout your repository
  - checkout: self

  # Install Node.js version 20.x
  - task: NodeTool@0
    inputs:
      versionSpec: '20.x'
      displayName: 'Install Node.js'
```

```
# Install dependencies and build project
```

```
- script: |
```

```
  npm install
```

```
  npm run build
```

```
  displayName: 'Install Dependencies and Build'
```

```
  workingDirectory:
```

```
  '$(Build.SourcesDirectory)/crime-foresight-dashboard-FINAL/crime-foresight-dashboard-main'
```

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.

### Implementation:

The screenshot displays the Azure DevOps web interface for a pipeline named 'CRIME-RATE-DETECTION'. The left sidebar shows the navigation menu with options like Overview, Boards, Repos, Pipelines, Environments, Library, Test Plans, and Artifacts. The main area is titled 'Review your pipeline YAML' and shows the YAML configuration for a pipeline named 'CRIME-DETECTION-SC-PROJECT'. The pipeline is configured with a trigger on the 'main' branch, a pool named 'Default', and several steps: 'Checkout your repository', 'Install Node.js version 20.x', and 'Install dependencies and build project'. The 'Install dependencies and build project' step is highlighted, showing its script and working directory.

```
1 # Node.js
2 # Build a general Node.js project with npm.
3 # Add steps that analyze code, save build artifacts, deploy, and more:
4 # https://docs.microsoft.com/azure/devops/pipelines/languages/javascript
5
6 trigger:
7   - main
8
9 pool:
10  - name: Default
11
12 steps:
13   # Checkout your repository
14   - checkout: self
15
16   # Install Node.js version 20.x
17   Settings
18   - task: NodeTool@0
19     inputs:
20       - versionSpec: '20.x'
21     displayName: 'Install Node.js'
22
23   # Install dependencies and build project
24   - script: |
25     npm install
26     npm run build
27     displayName: 'Install Dependencies and Build'
28     workingDirectory: '$(Build.SourcesDirectory)/crime-foresight-dashboard-FINAL/crime-foresight-dashboard-main'
```

Azure DevOps 230701372 / CRIME-RATE-DETECTION / Pipelines

CRIME-RATE-DETECTION +

Overview

Boards

Repos

Pipelines

Pipelines

Environments

Library

Test Plans

Artifacts

Project settings

Connect Select Configure Review

New pipeline

**Review your pipeline YAMI**

Save and run

Saving will commit azure-pipelines-3.yml to the repository.

Commit message

Set up CI with Azure Pipelines

Azure DevOps 230701372 / CRIME-RATE-DETECTION / Pipelines / CRIME-DETECTION-SC-PRO... / 20250520.1

Search

Jobs in run #20250520.1

CRIME-DETECTION-SC-PROJECT (17)

Jobs

Job	Duration
Initialize job	<1s
Checkout CRIME-DETE...	6s
Install Node.js	3m 17s
Install Dependencies a...	26s
Post-job: Checkout CR...	<1s
Finalize Job	<1s
Report build status	<1s

Job

1 Pool: Default

2 Agent: ASUS

3 Started: Today at 12:32 PM

4 Duration: 3m 51s

5

6 Job preparation parameters

41 Job live console data:

42 Starting: Job

43 Async Command Start: WindowsPreInstalledGitTelemetry

44 Async Command End: WindowsPreInstalledGitTelemetry

45 Async Command Start: WindowsPreInstalledGitTelemetry

46 Async Command End: WindowsPreInstalledGitTelemetry

47 Async Command Start: WindowsPreInstalledGitTelemetry

48 Async Command End: WindowsPreInstalledGitTelemetry

49 Async Command Start: WindowsPreInstalledGitTelemetry

50 Async Command End: WindowsPreInstalledGitTelemetry

51 Finishing: Job

View raw log

Command Prompt - run.cmd

Microsoft Windows [Version 10.0.26100.4061]  
(c) Microsoft Corporation. All rights reserved.

C:\Users\Varsha Thomas>cd C:\Users\Varsha Thomas\Downloads\vsts-agent-win-x64-4.255.0

C:\Users\Varsha Thomas\Downloads\vsts-agent-win-x64-4.255.0>run.cmd

Scanning for tool capabilities.

Connecting to the server.

2025-05-20 06:57:20Z: Listening for Jobs

2025-05-20 06:57:24Z: Running job: Job

2025-05-20 06:58:32Z: Job Job completed with result: Failed

2025-05-20 06:58:41Z: Running job: Job

2025-05-20 06:58:59Z: Job Job completed with result: Failed

2025-05-20 06:59:01Z: Running job: Job

2025-05-20 06:59:27Z: Job Job completed with result: Failed

2025-05-20 06:59:29Z: Running job: Job

2025-05-20 06:59:52Z: Job Job completed with result: Failed

2025-05-20 06:59:55Z: Running job: Job

2025-05-20 07:00:29Z: Job Job completed with result: Failed

2025-05-20 07:00:31Z: Running job: Job

2025-05-20 07:00:57Z: Job Job completed with result: Canceled

2025-05-20 07:00:59Z: Running job: Job

2025-05-20 07:01:27Z: Job Job completed with result: Failed

2025-05-20 07:01:29Z: Running job: Job

2025-05-20 07:01:59Z: Job Job completed with result: Failed

2025-05-20 07:02:01Z: Running job: Job

2025-05-20 07:02:44Z: Job Job completed with result: Failed

2025-05-20 07:02:46Z: Running job: Job

2025-05-20 07:06:49Z: Job Job completed with result: Succeeded

Result

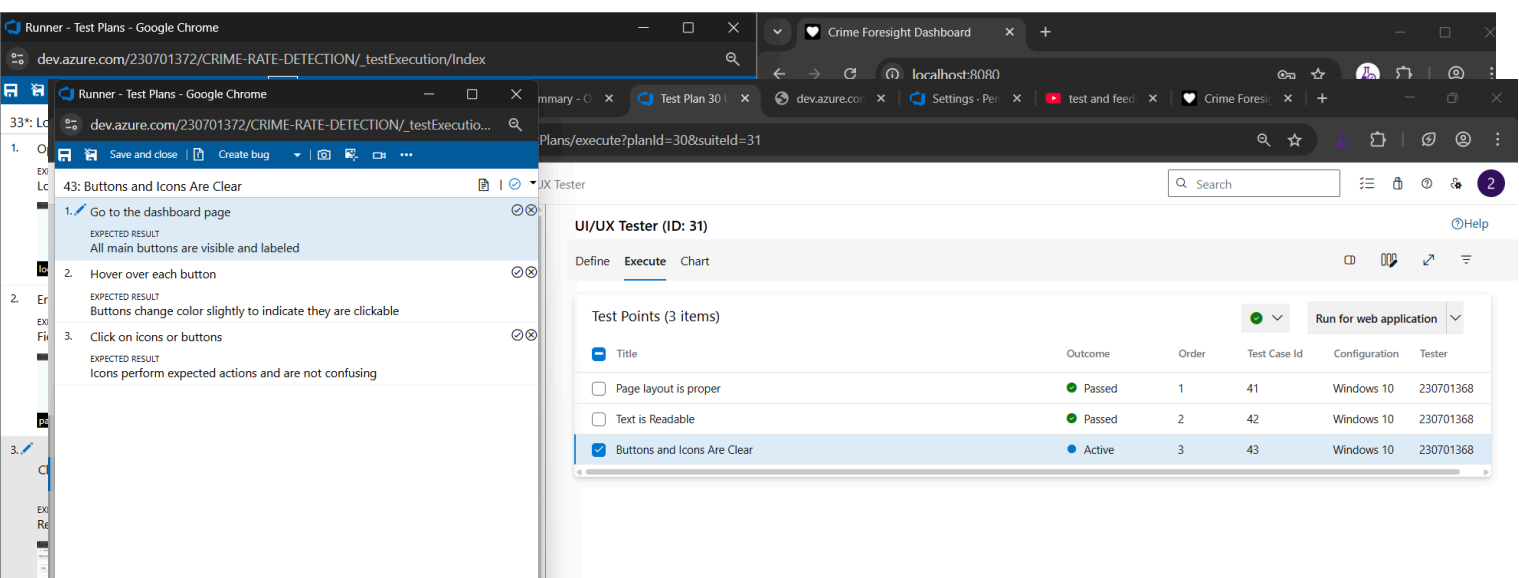
Thus the application was successfully implemented.

EXP NO.12

DATE:TESTING

Aim:

To test the functionality, performance, and reliability of the application using automated pipelines within Azure DevOps.





dev.azure.com/230701372/CRIME-RATE-DETECTION/\_testManagement/runs?\_a=runQuery

Azure DevOps 230701372 / CRIME-RATE-DETECTION / Test Plans / Runs

CRIME-RATE-DETECTION +

- Overview
- Boards
- Repos
- Pipelines
- Test Plans
- Test plans
- Progress report
- Parameters
- Configurations
- Runs
- Artifacts

Enter Run ID... Go

Recent test runs

Test runs Filter

Recent exploratory sessions

15 runs (1 selected)

State	Run L...	Title	Completed Date	Build Number	Failed	Pass Rate
Completed	36	UI/UX Tester (Manual)	5/20/2025 5:01:59 PM		0	100%
Completed	35	UI/UX Tester (Manual)	5/20/2025 4:58:24 PM		0	100%
Completed	34	UI/UX Tester (Manual)	5/20/2025 4:56:10 PM		0	100%
Completed	33	Functionality Tester (Manual)	5/20/2025 4:49:16 PM		0	100%
Completed	32	Functionality Tester (Manual)	5/20/2025 4:45:18 PM		0	100%
Completed	30	Functionality Tester (Manual)	5/20/2025 4:42:39 PM		0	100%
Completed	28	Functionality Tester (Manual)	5/20/2025 4:38:52 PM		0	100%
Completed	27	Functionality Tester (Manual)	5/20/2025 4:31:51 PM		0	100%
Completed	25	Functionality Tester (Manual)	5/20/2025 4:29:21 PM		0	100%
In progress	24	Functionality Tester (Manual)	5/20/2025 4:26:37 PM		0	0%
Completed	19	Functionality Tester (Manual)	5/20/2025 4:10:10 PM		0	100%
Completed	18	Functionality Tester (Manual)	5/20/2025 4:09:26 PM		0	0%
Completed	17	Functionality Tester (Manual)	5/20/2025 4:09:22 PM		0	0%
Completed	12	Functionality Tester (Manual)	5/20/2025 3:57:34 PM		0	100%
In progress	8	UI/UX Tester (Manual)	5/20/2025 3:52:10 PM		0	0%

## Result:

Thus, the application was successfully tested and validated in Azure DevOps pipelines.