



**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

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

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: “-”

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

PROBLEM STATEMENT

AIM:

To prepare PROBLEM STATEMENT for your given project.

PROBLEM STATEMENT:

E-CHALLAN GENERATION

Manual enforcement of illegal parking is inefficient, error-prone, and limited in scale, leading to traffic congestion, safety hazards, and enforcement challenges. There is a need for an automated, AI-driven system to detect, verify, and issue fines accurately, ensuring transparency, efficiency, and better urban traffic management.

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.

CREATE USER STORIES

AIM:

To create User Stories

THEORY

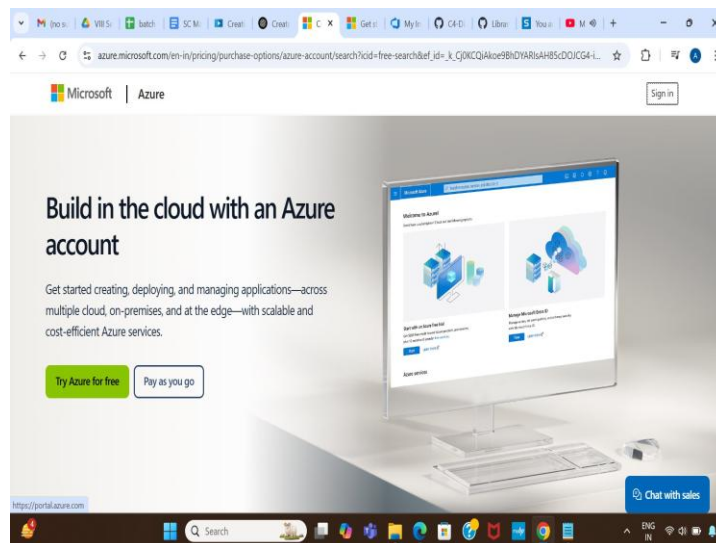
A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer.

User story template

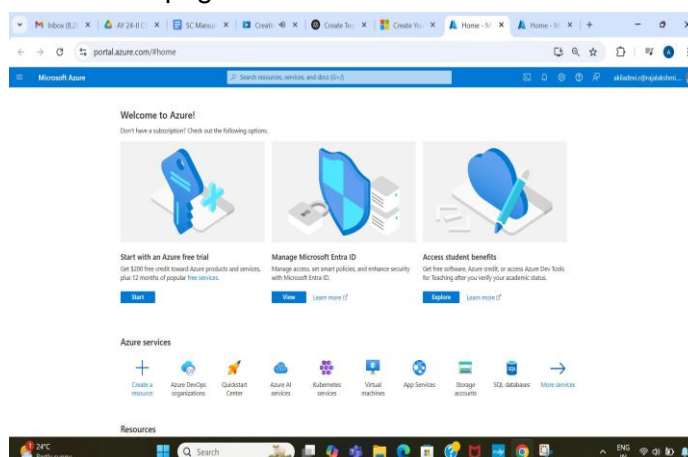
"As a [role], I [want to], [so that]."

PROCEDURE:

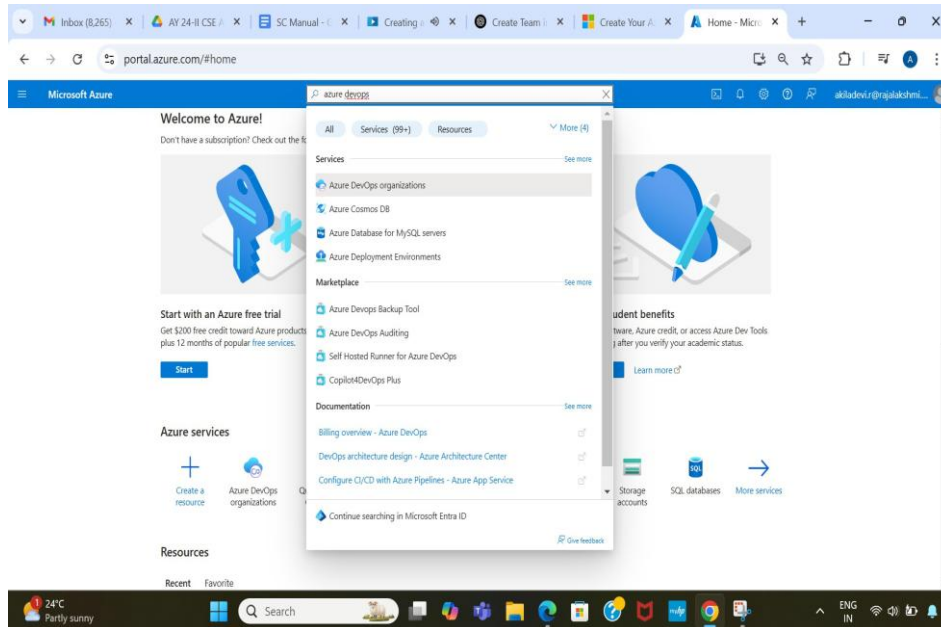
1. Open your web browser and go to the Azure website:
<https://azure.microsoft.com/en-in> Sign in using your Microsoft account credentials. If you don't have an account, you'll need to create one.
2. If you don't have a Microsoft account, you can sign up for
<https://signup.live.com/?lic=1>



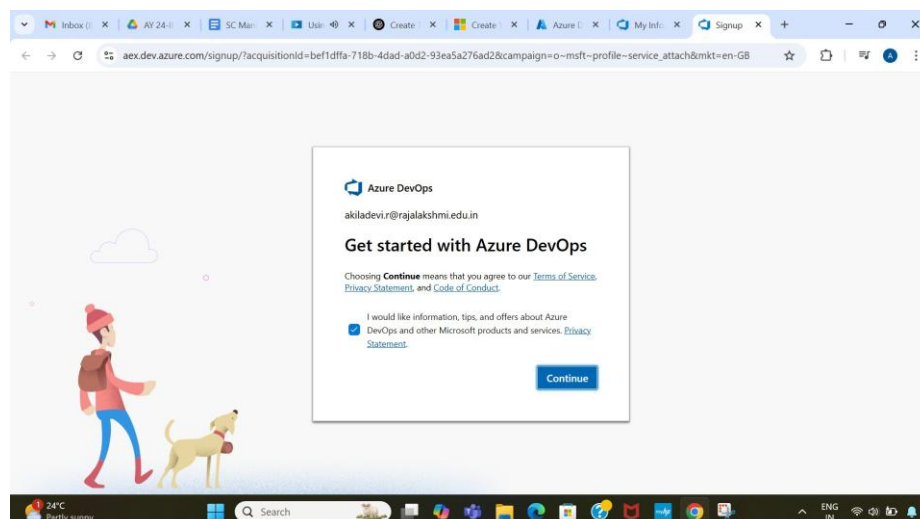
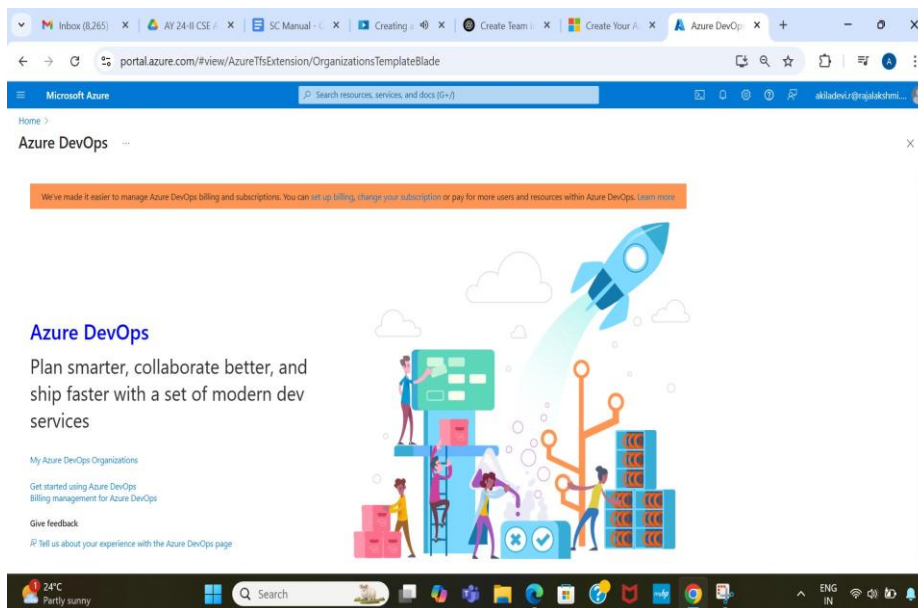
3. Azure home page

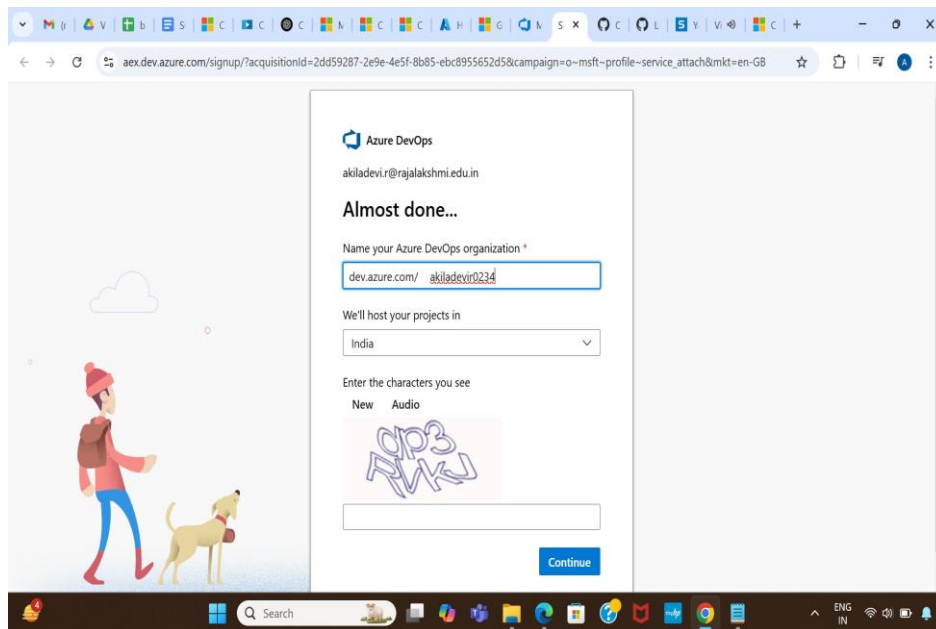


4. Open DevOps environment in the Azure platform by typing Azure DevOps Organizations in the search bar.



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.



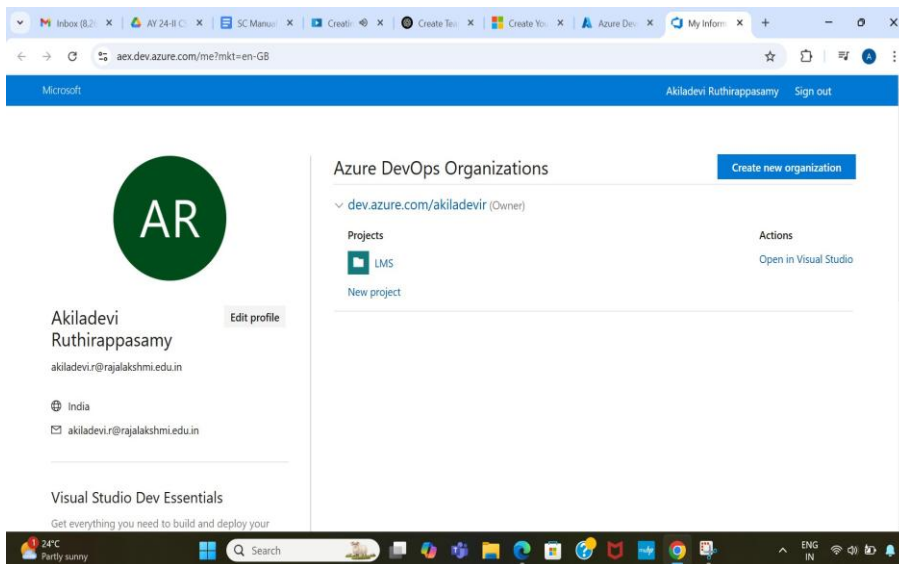


6. Create the First Project in Your Organization

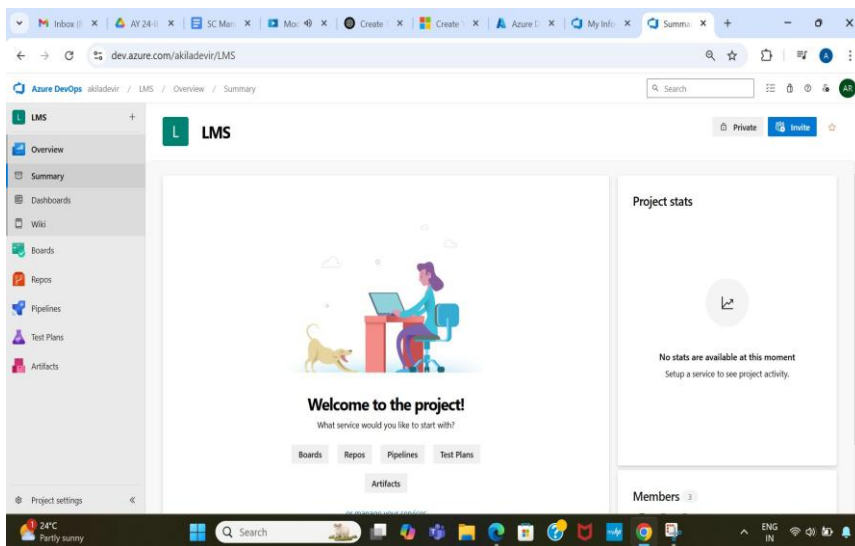
After the organization is set up, you'll need to create your first **project**. This is where you'll begin to manage code, pipelines, work items, and more.

- i. On the organization's **Home page**, click on the **New Project** button.
- ii. Enter the project name, description, and visibility options:
 - 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.

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.

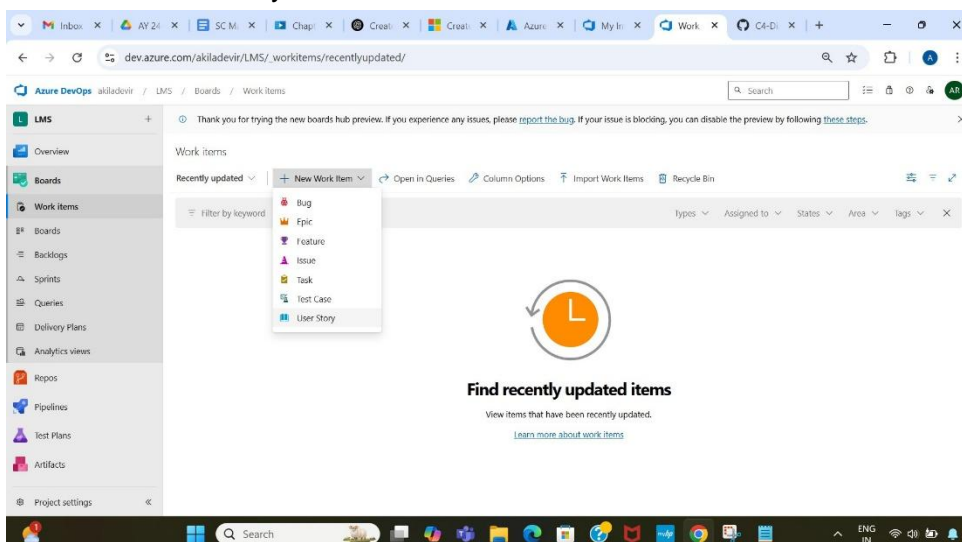


8. Project dashboard



9. To manage user stories

- 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.



10. Fill in User Story Details

The screenshot shows the Azure DevOps interface for a project named 'Expert System'. The left sidebar contains navigation options: Overview, Boards, Work items, Backlogs, Sprints, Queries, Delivery Plans, Analytics views, Repos, Pipelines, Test Plans, Artifacts, and Project settings. The main area displays the details of a user story titled 'Login' (USER STORY 17) created by Akiladevi Ruthirappasamy. The story is in the 'New' state and is associated with the 'Expert System' area and 'Expert System/Sprint 1' iteration. The description is: 'As a registered user, I want to log into my account so that I can access my personal dashboard and features.' The acceptance criteria are listed as follows:

1. The user should be able to access the login page from the homepage or a direct URL.
2. The login form must have fields for email/username and password.
3. If the user submits an empty field, they should see a validation error.
4. If the email format is invalid, an error message should be displayed.
5. If valid credentials are provided, the user should be redirected to their dashboard.
6. If incorrect credentials are entered, an appropriate error message should be displayed.
7. If the user enters incorrect credentials more than 5 times, their account should be temporarily locked for 10 minutes.
8. The password input field should be masked by default but allow users to toggle visibility.
9. Users should have an option to reset their password via a "Forgot Password?" link.

The right sidebar contains sections for Planning (Story Points: 3, Priority: 2, Risk: 2 - Medium), 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.), and Related Work.

OUTPUT

The screenshot shows the Azure DevOps interface for a project named 'E-challan generation'. The left sidebar contains navigation options: Overview, Boards, Work items, Backlogs, Sprints, Queries, Delivery Plans, Pipelines, Artifacts, and Project settings. The main area displays a list of work items under the 'Work items' section. The list is filtered by keyword and shows the following items:

ID	Title	Assigned to	State	Area Path	Tags
8	AI Based License Plate Recognition	Unassigned	New	E-challan generation	
7	View Past Reports	Unassigned	New	E-challan generation	
6	Submit a Traffic Violation Report	Unassigned	New	E-challan generation	
5	Manage Officer Performance & Prevent Misuse	Unassigned	New	E-challan generation	
4	Approve or Reject Violation Reports	Unassigned	New	E-challan generation	
3	Dispute a Challan	Unassigned	New	E-challan generation	
2	View & Pay a Challan	Unassigned	New	E-challan generation	
1	Receive a Challan notification	Unassigned	New	E-challan generation	

RESULT:

The user story was written successfully.

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

```

::: mermaid
sequenceDiagram
    CUSTOMER ->> BOOK-LOAN : borrows
    CUSTOMER ->> LIBRARY-MEMBER : registers
    CUSTOMER ->>{ FINE : incurs
    BOOK-LOAN ->> BOOK : contains
    BOOK-LOAN ->> LIBRARY-MEMBER : issued-to
    BOOK ->> BOOK-LOAN : "borrowed in"
    LIBRARY-MEMBER ->> BOOK-LOAN : "issued in"
    LIBRARY-MEMBER ->> LIBRARY-STAFF : manages
    BOOK-CATEGORY ->> BOOK : belongs-to
:::
```

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

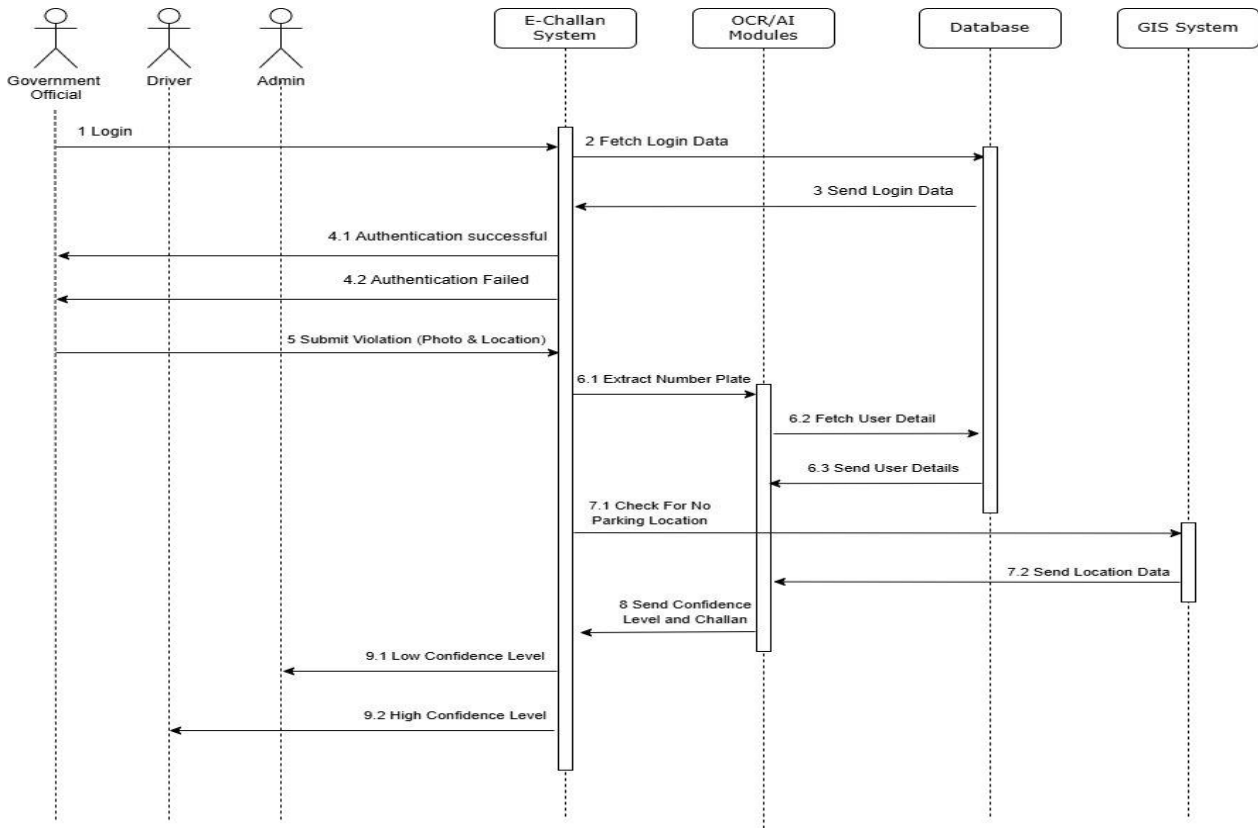
--x Dotted line with a cross at the end

-) Solid line with an open arrow at the end (async)

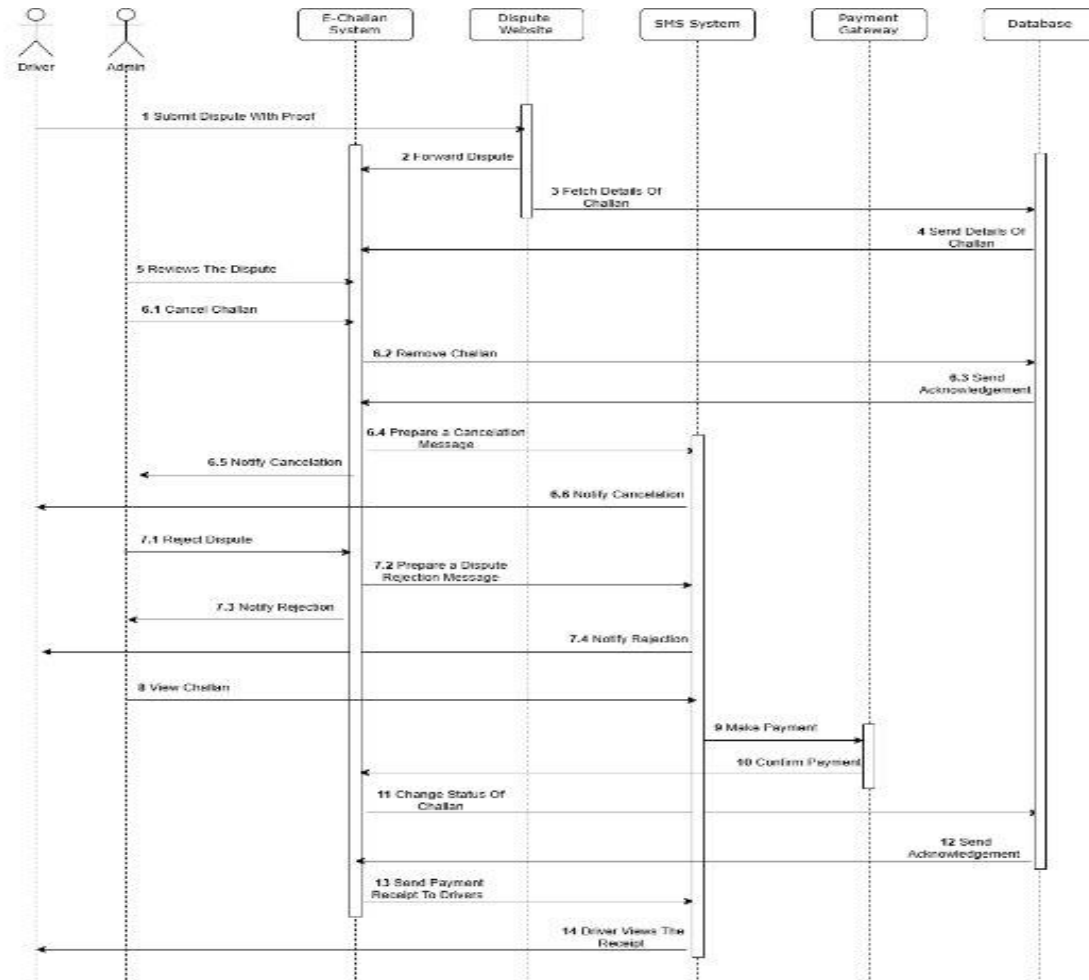
--) Dotted line with an open arrow at the end (async)

4. click wiki menu and select the page

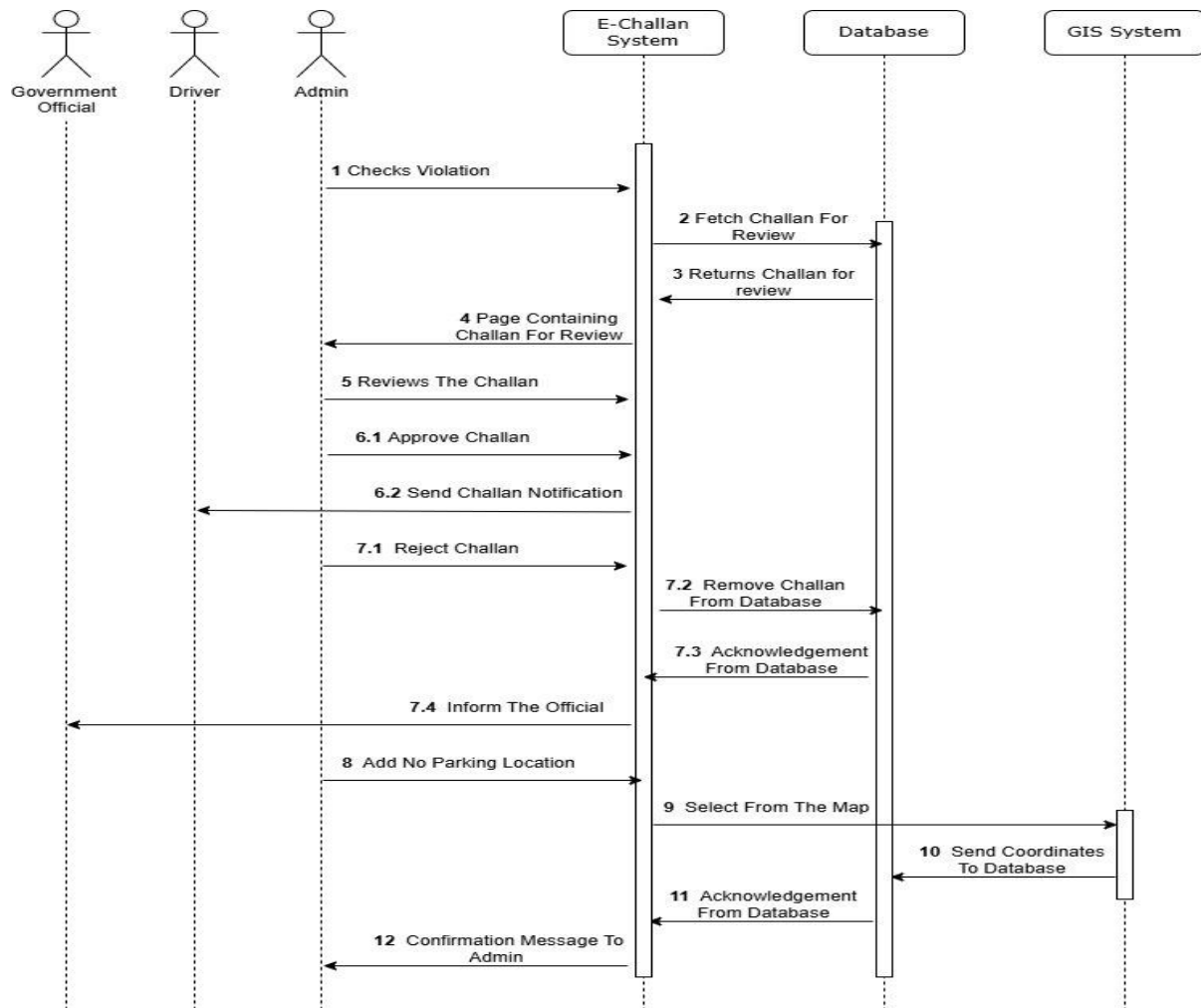
OUTPUT:
SEQUENCE DIAGRAM (VIOLATION REPORTING):



SEQUENCE DIAGRAM (PAYMENT AND DISPUTE):



SEQUENCE DIAGRAM (ADMIN):



RESULT:

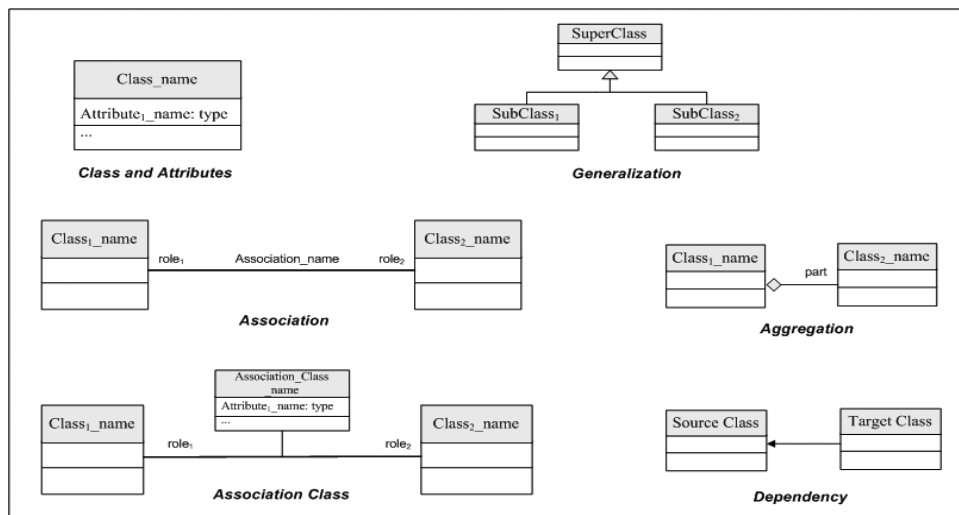
The Sequence diagram was designed successfully.

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
class Person {
    +String name
    +int age
    +walk()
    +talk()
}

class Employee {
    +String company
    +double salary
    +work()
}

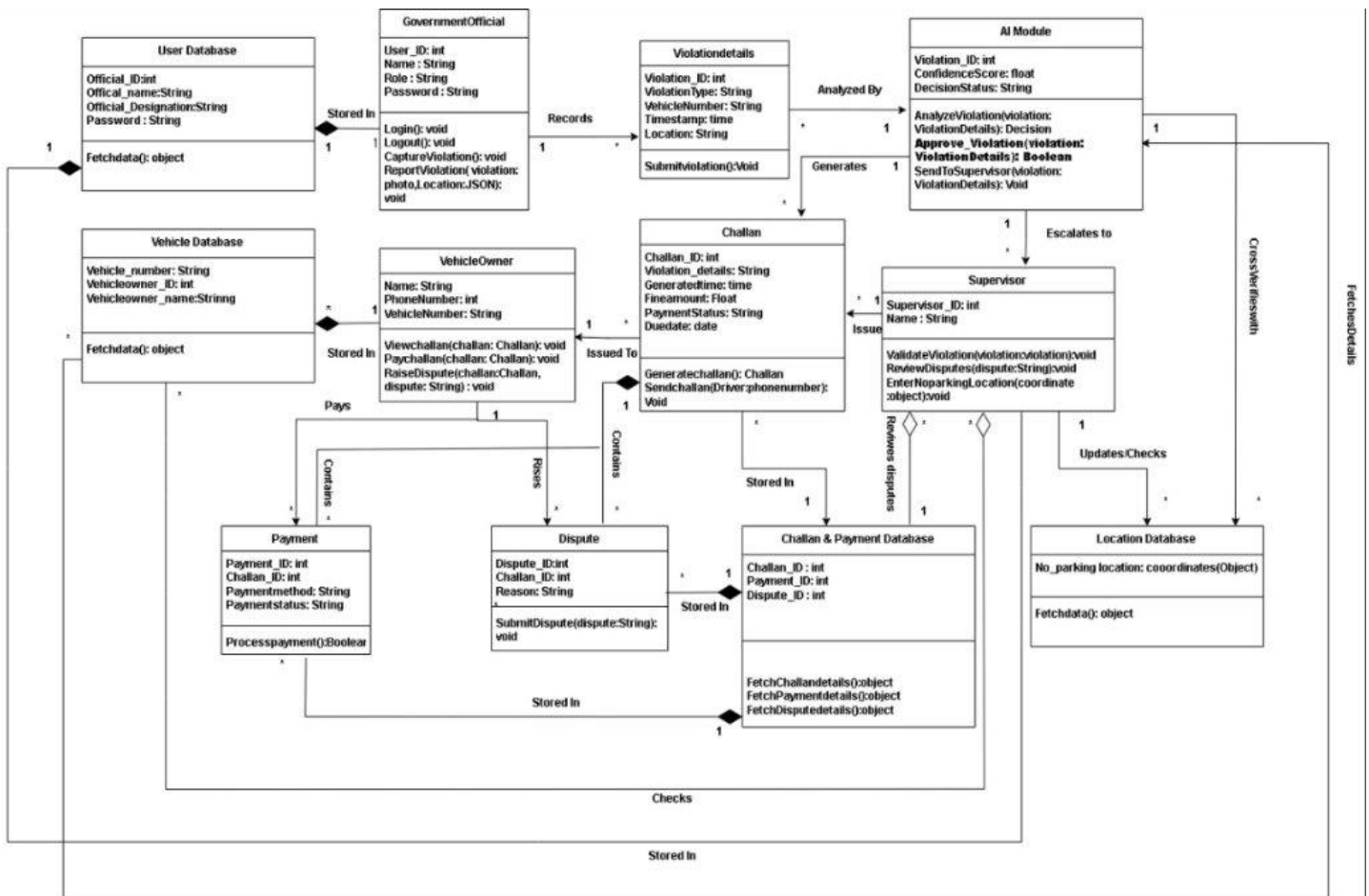
class Manager {
    +String department
    +manageTeam()
}

Person <|-- Employee // Inheritance
Employee <|-- Manager // Inheritance
Person --> Employee : "has a" // Association
Manager --* Employee : "manages" // Composition
  
```

Relationship Types

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

CLASS DIAGRAM:



RESULT:

The Class diagram was designed successfully.

USECASE DIAGRAM

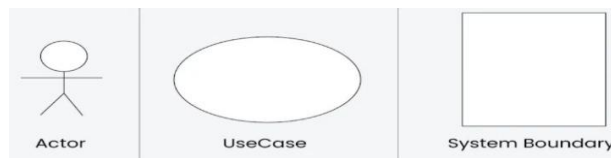
AIM:

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

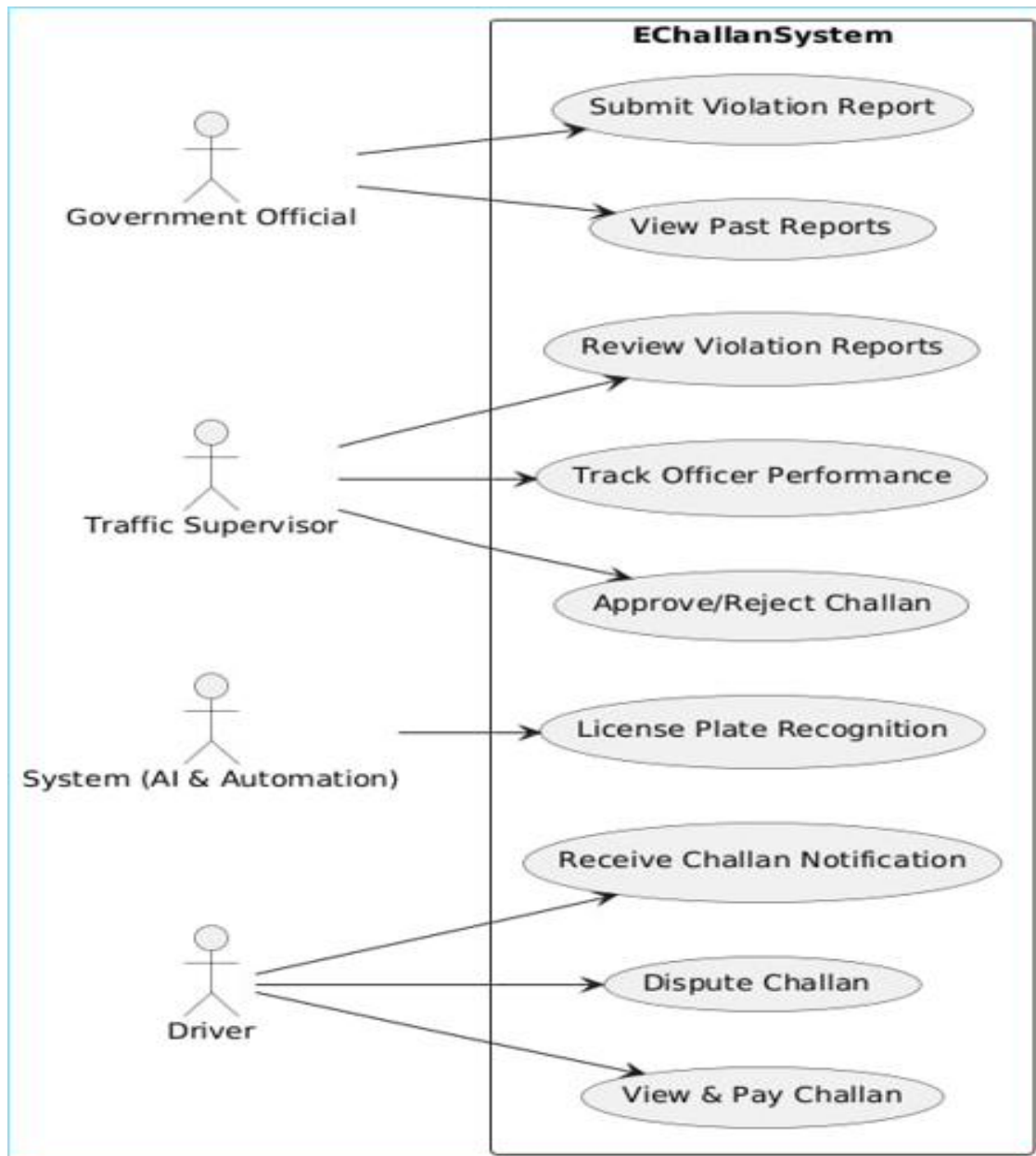
- Open Azure DevOps and go to your project.
- Navigate to Wiki (Project > Wiki).
- Click "Edit Page" or create a new page.
- Drag & Drop the exported PNG/JPG image.
- Use Markdown to embed the diagram:
• ![Use Case Diagram](attachments/use_case_diagram.png)

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:



RESULT:

The use case diagram was designed successfully

EX NO. 8



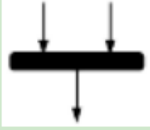



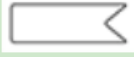




ACTIVITY DIAGRAM

AIM :-

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

THEORY

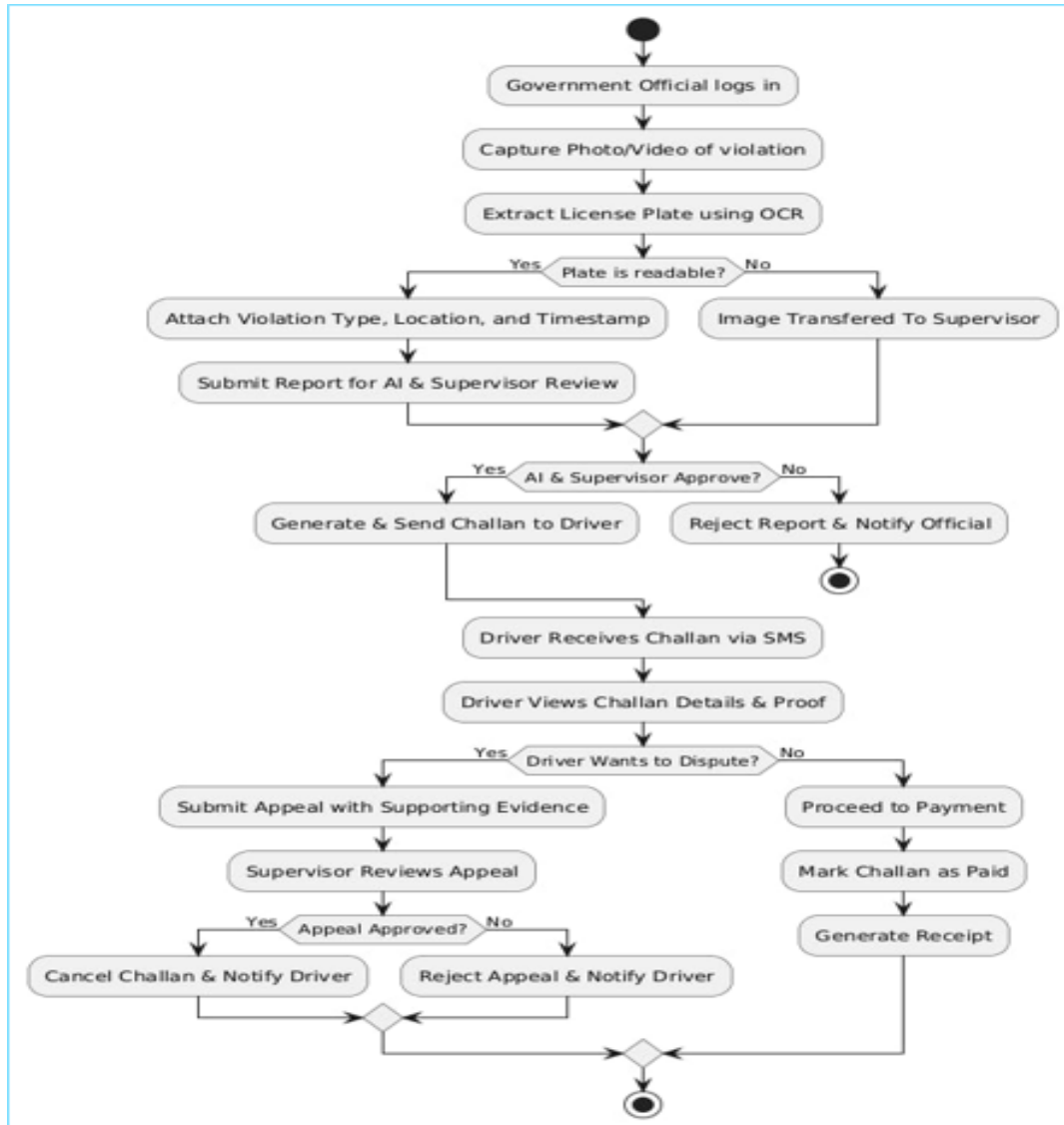
Activity diagrams are an essential part of the Unified Modelling Language (UML) that help visualize workflows, processes, or activities within a system. They depict how different actions are connected and how a system moves from one state to another.

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

ACTIVITY DIAGRAM



RESULT:

The activity diagram was designed successfully

EX NO. 9

ARCHITECTURE DIAGRAM

AIM:

Steps to draw the Architecture Diagram using draw.io.

THEORY:

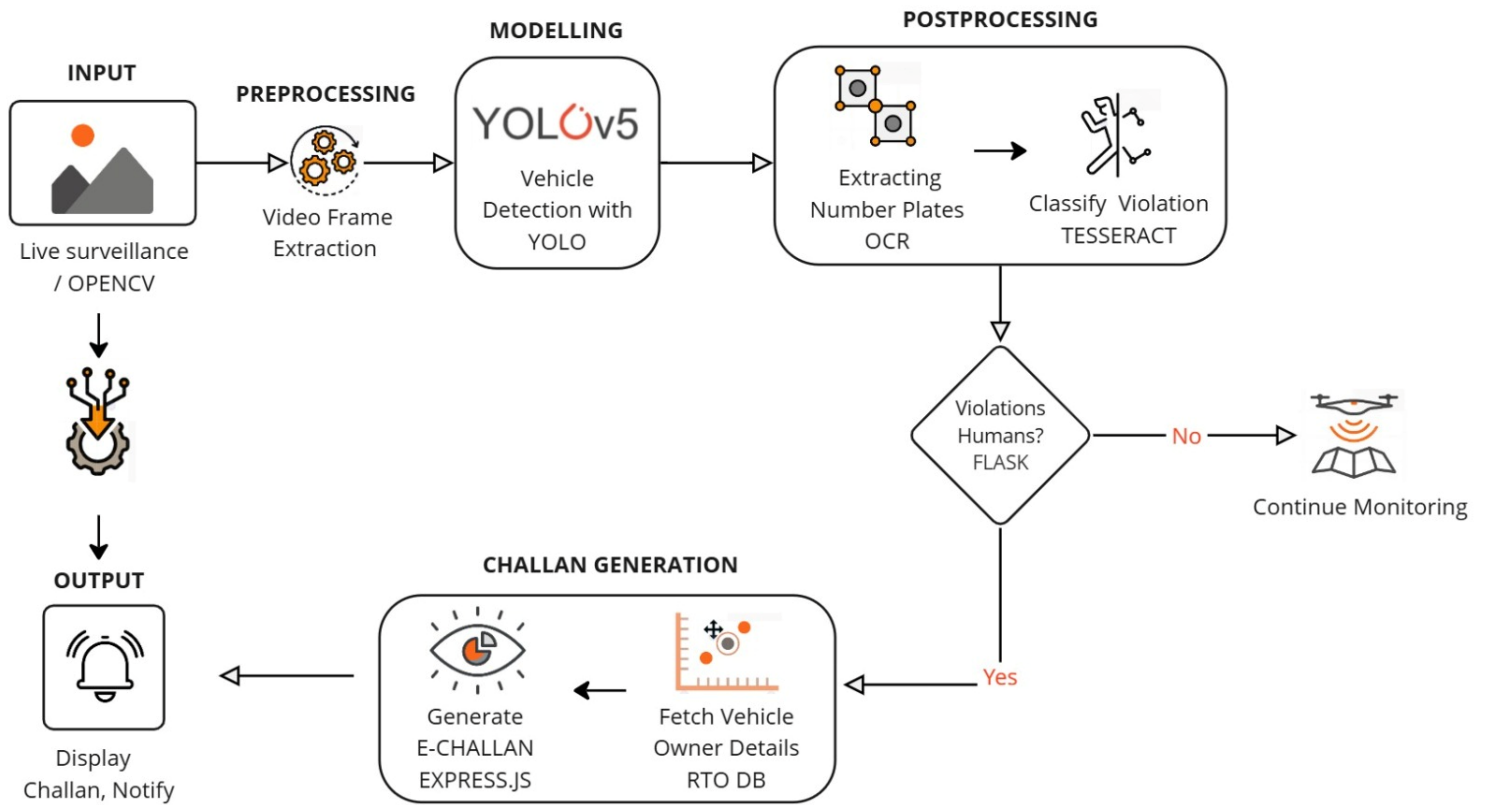
An architectural diagram is a visual representation that maps out the physical implementation for components of a software system. It shows the general structure of the software system and the associations, limitations, and boundaries between each element.



PROCEDURE:

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

ARCHITECTURE DIAGRAM:



RESULT:

The architecture diagram was designed successfully

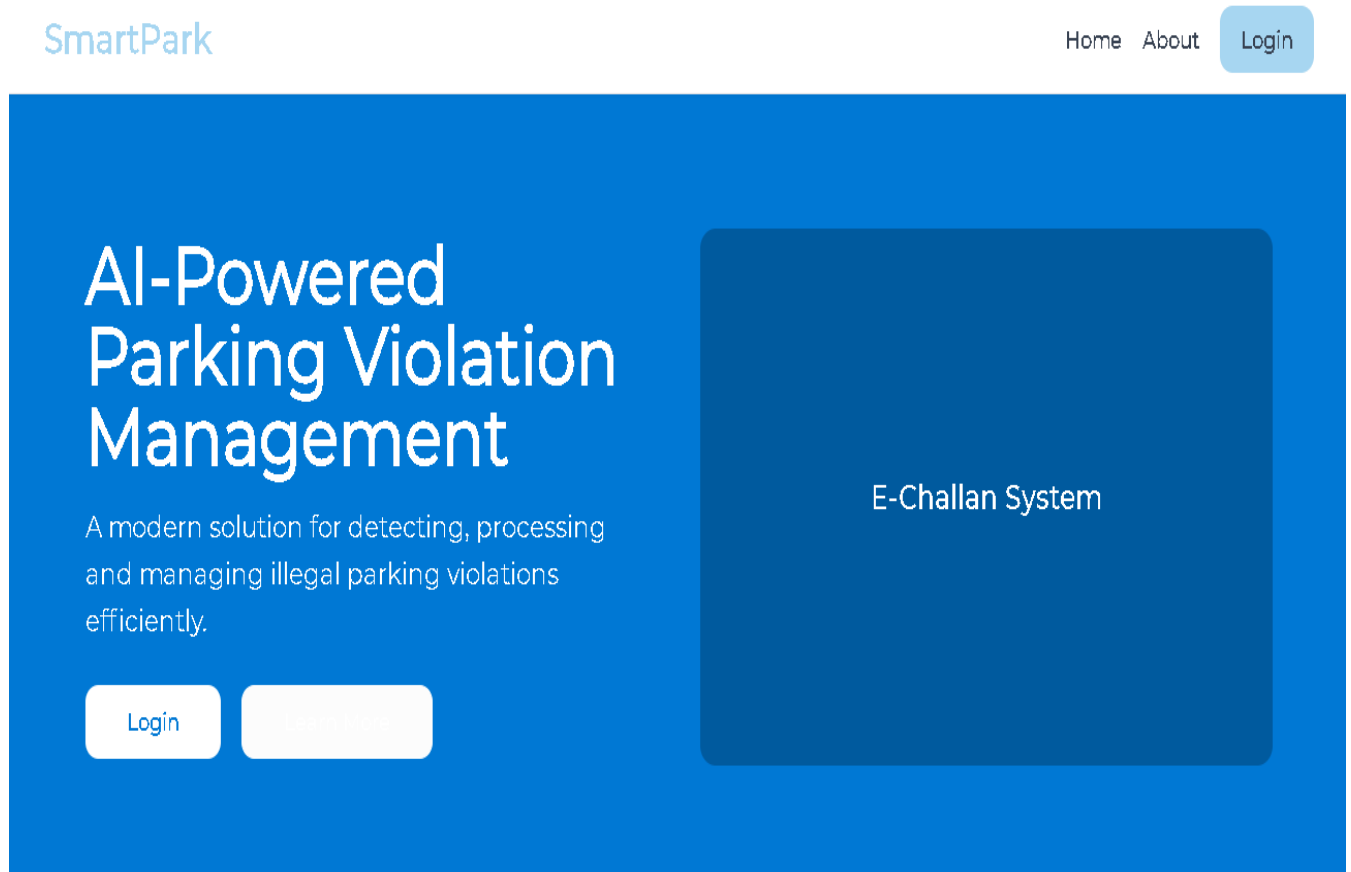
EX NO. 10

USER INTERFACE

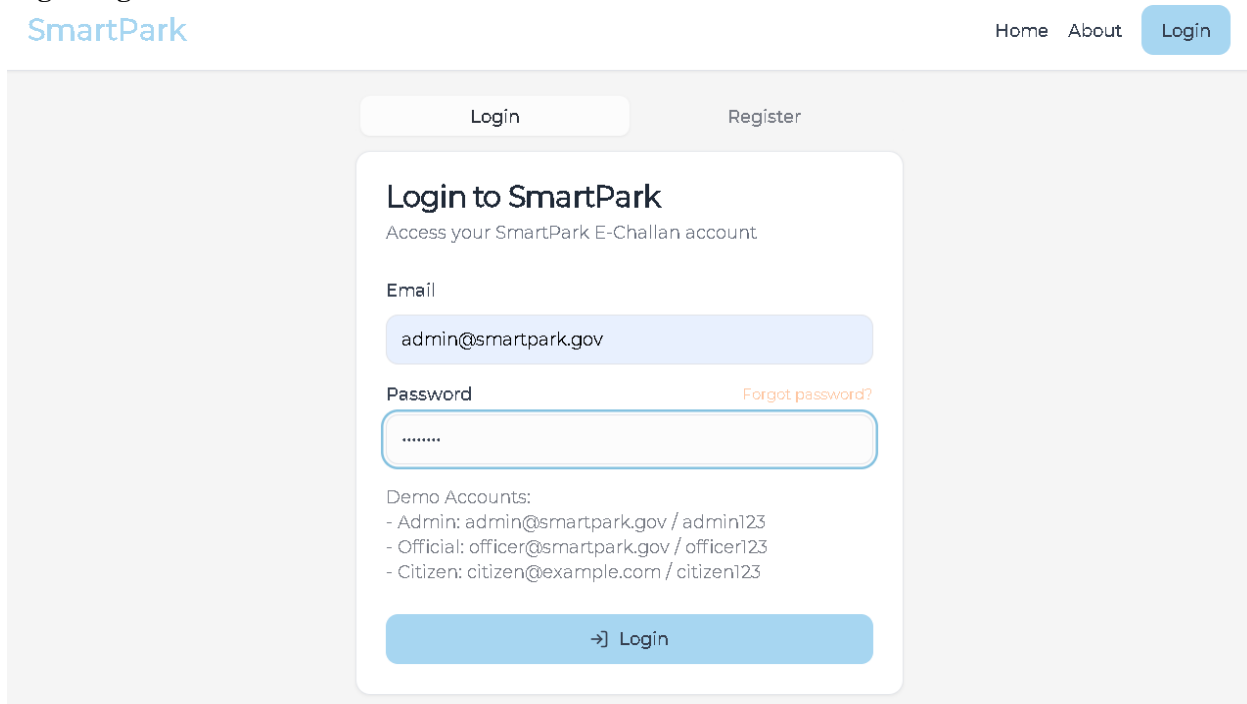
AIM:

Design User Interface for the given project

Home Page:



Login Page:



Admin Dashboard:

SmartPark

Admin User

Admin

Overview

User Management

Review Violations

Challans

Disputes

Reports

Settings

Logout

Admin Dashboard

Total Challans

1,248

+12% from last month

Pending Review

36

-5% from last month

Open Disputes

28

+8% from last month

Total Users

157

+3% from last month

Recent Violations

Violation #1001

Reported by Officer 101

4/30/2025

Violation #1002

Reported by Officer 102

4/30/2025

Violation #1003

Reported by Officer 103

4/30/2025

Violation #1004

Reported by Officer 104

4/30/2025

Recent Disputes

Dispute for Challan #2001

Filed by Citizen 201

4/30/2025

Dispute for Challan #2002

Filed by Citizen 202

4/30/2025

Dispute for Challan #2003

Filed by Citizen 203

4/30/2025

Dispute for Challan #2004

Filed by Citizen 204

4/30/2025

SmartPark

Admin User

Admin

User Management

Review Violations

Challans

Disputes

Reports

Settings

Logout

User Management

Add User

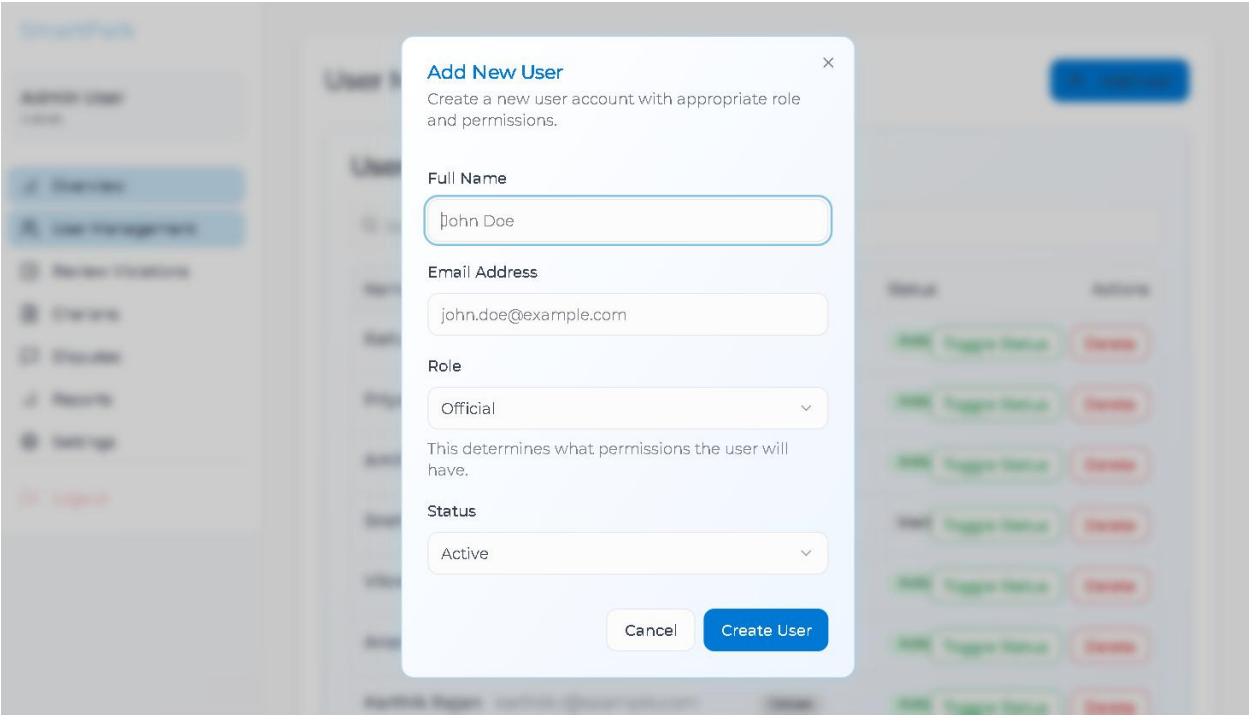
Users

Search users...

Name	Email	Role	Status	Actions
Rahul Sharma	rahul.s@example.com	Admin	Active	<div>Toggle Status</div> <div>Delete</div>
Priya Patel	priya.p@example.com	Official	Active	<div>Toggle Status</div> <div>Delete</div>
Amit Kumar	amit.k@example.com	Official	Active	<div>Toggle Status</div> <div>Delete</div>
Sneha Reddy	sneha.r@example.com	Official	Inactive	<div>Toggle Status</div> <div>Delete</div>
Vikram Singh	vikram.s@example.com	Citizen	Active	<div>Toggle Status</div> <div>Delete</div>
Ananya Gupta	ananya.g@example.com	Citizen		

User status updated

Rahul Sharma's status is now active.



SmartPark

Admin User
Admin

Overview

User Management

Review Violations

Challans

Disputes

Reports

Settings

Logout

Review Violations

Pending Review (5)

Approved (2)

Rejected (2)

Violation by KA-01-AB-1234

KA-01-AB-1234

ID: V-1001

Location
MG Road, Bengaluru

Reported By
Officer Kumar

Date & Time
28 Apr 2024, 10:15 am

Reject

Approve

Violation by MH-02-CD-5678

MH-02-CD-5678

ID: V-1002

Location
Linking Road, Mumbai

Reported By
Officer Sharma

Date & Time
28 Apr 2024, 11:30 am

Reject

Approve

- Overview
- User Management
- Review Violations
- Challans
- Disputes
- Reports
- Settings

Logout

Challans Management

Search challans...

All (7) Pending (3) Paid (2) Disputed (2)

Challan ID	Vehicle Number	Location	Date	Amount	Status	
CH-1001	KA-01-AB-1234	MG Road, Bengaluru	20 Apr 2024, 10:15 am	₹500	Paid	Download
CH-1002	MH-02-CD-5678	Linking Road, Mumbai	21 Apr 2024, 11:30 am	₹500	Pending	Download
CH-1003	DL-03-EF-9012	Connaught Place, Delhi	22 Apr 2024, 09:45 am	₹500	Pending	Download
CH-1004	TN-04-GH-3456	Anna Salai, Chennai	23 Apr 2024, 02:20 pm	₹500	Disputed	Download
CH-1005	WB-05-IJ-7890	Park Street, Kolkata	24 Apr 2024, 01:10 pm	₹500	Paid	Download

- Overview
- User Management
- Review Violations
- Challans
- Disputes
- Reports
- Settings

Logout

Analytics & Reports

2024 All Data

Total Violations
1,248
+12% from last month

Challans Issued
1,156
+10% from last month

Revenue Collected
₹5,78,000
+15% from last month

Dispute Rate
8.6%
-2% from last month

Overview

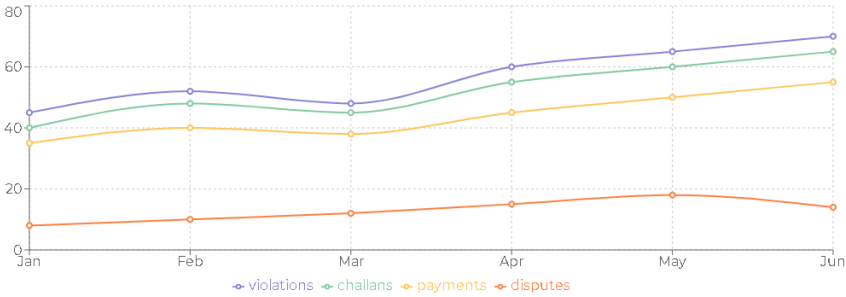
Locations

Officers

Payment

Monthly Performance

Violations, challans, payments, and disputes over time



Admin User
Admin

Overview

User Management

Review Violations

Challans

Disputes

Reports

Settings

Logout

Analytics & Reports

2024

All Data

Total Violations
1,248
+12% from last month

Challans Issued
1,156
+10% from last month

Revenue Collected
₹5,78,000
+15% from last month

Dispute Rate
8.6%
-2% from last month

Overview

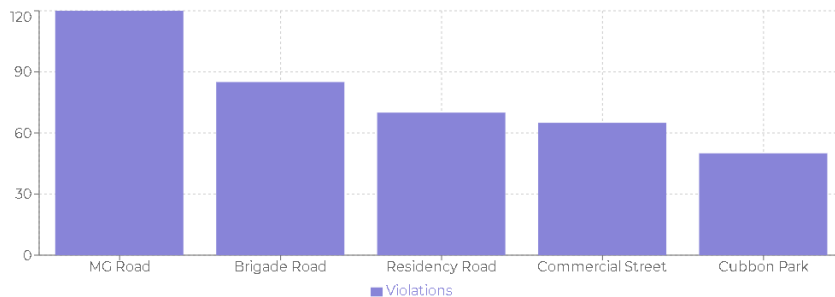
Locations

Officers

Payment

Top Violation Locations

Areas with the highest number of parking violations



Admin User
Admin

Overview

User Management

Review Violations

Challans

Disputes

Reports

Settings

Logout

Analytics & Reports

2024

All Data

Total Violations
1,248
+12% from last month

Challans Issued
1,156
+10% from last month

Revenue Collected
₹5,78,000
+15% from last month

Dispute Rate
8.6%
-2% from last month

Overview

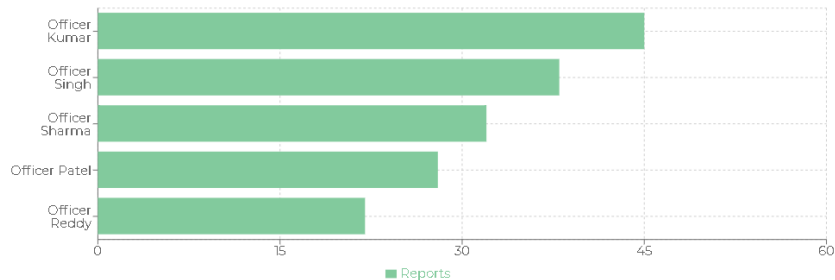
Locations

Officers

Payment

Officer Performance

Reports submitted by each traffic officer



Admin User
Admin

Overview

User Management

Review Violations

Challans

Disputes

Reports

Settings

Logout

Analytics & Reports

2024

All Data

Total Violations

1,248

+12% from last month

Challans Issued

1,156

+10% from last month

Revenue Collected

₹5,78,000

+15% from last month

Dispute Rate

8.6%

-2% from last month

Overview

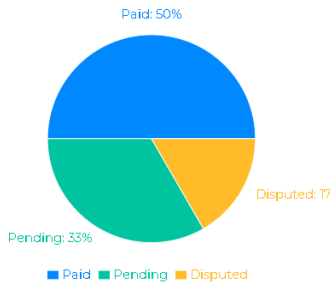
Locations

Officers

Payment

Payment Status

Distribution of challan payment statuses



Citizen Dashboard:

Citizen User
Citizen

Overview

My Challans

Pay Challan

Disputes

Profile

Logout

Citizen Dashboard

Total Challans

3



Pending Payment

1



Disputes Filed

1



Your Challans

Challan #CH-1001
Date: 2023-12-15

₹500
Paid

Challan #CH-1002
Date: 2024-01-20

₹800
Pending

Challan #CH-1003
Date: 2024-02-05

₹1,000
Disputed

Citizen User
Citizen

Overview

My Challans

Pay Challan

Disputes

Profile

Logout

Pay Challan

Select Challan to Pay

Choose from your pending challans

CHN-2023-4562

Vehicle: MH01AB1234 | Date: 2023-10-15
Violation: No Parking Zone

₹500
Due Amount

CHN-2023-4623

Vehicle: MH01AB1234 | Date: 2023-11-22
Violation: Wrong Side Parking

₹1000
Due Amount

CHN-2023-4891

Vehicle: MH01CD5678 | Date: 2023-12-01
Violation: Footpath Parking

₹200
Due Amount

Payment Options

Choose your payment method

Card UPI NetBanking

Card Number

1234 5678 9012 3456

Expiry Date

MM/YY

CVV

123

Name on Card

Pay Now

Payment Summary

Challan ID:

CH-1002

Vehicle Number:

Violation:

Total Amount:

₹

Secure Payment

All your payment information is encrypted and secure. We do not store your card details.

Citizen User
Citizen

Overview

My Challans

Pay Challan

Disputes

Profile

Logout

Pay Challan

Select Challan to Pay

Choose from your pending challans

- CHN-2023-4562

MH01AB1234

Date: 2023-10-15

₹500

Due Amount

Violation: No Parking Zone
- CHN-2023-4623

MH01AB1234

Date: 2023-11-22

₹1000

Due Amount

Violation: Wrong Side Parking
- CHN-2023-4891

MH01CD5678

Date: 2023-12-01

₹200

Due Amount

Violation: Footpath Parking

Payment Options

Choose your payment method

Card

UPI

NetBanking

Card Number

1234 5678 9012 3456

Expiry Date

MM/YY

CVV

123

Name on Card

Pay Now

Payment Summary

Challan ID:	CHN-2023-4623
Vehicle Number:	MH01AB1234
Violation:	Wrong Side Parking
Total Amount:	₹1000

Secure Payment
All your payment information is encrypted and secure. We do not store your card details.

Citizen User
Citizen

Overview

My Challans

Pay Challan

Disputes

Profile

Logout

My Challans

- CHN-2023-4562

MH01AB1234

Date: 2023-10-15

₹500

Due Amount

Violation: No Parking Zone
- CHN-2023-4623

MH01AB1234

Date: 2023-11-22

₹1000

Due Amount

Violation: Wrong Side Parking
- CHN-2023-4891

MH01CD5678

Date: 2023-12-01

₹200

Due Amount

Violation: Footpath Parking

File a Dispute

Provide details about why you believe this challan was issued incorrectly.

Challan ID: CH-1002
Vehicle: MH-02-CD-5678

Reason for Dispute

Explain why you believe this challan was wrongly issued...

Supporting Evidence (Optional)

Click to upload or drag and drop
Support for images, PDF documents

Submit Dispute

Citizen User
Citizen

Overview

My Challans

Pay Challan

Disputes

Profile

Logout

Dispute Management

Pending (2)

Approved (1)

Rejected (1)

D-1001
for Challan CH-1004

Pending

Vehicle Number
TN-04-OH-3456

Submitted
25 Apr 2024, 04:45 pm

Reason for Dispute

I was not parked in a no-parking zone. There was no sign visible at the location.

Evidence Provided

parking_sign.jpg

D-1002
for Challan CH-1007

Pending

Vehicle Number
MH-07-MN-5678

Submitted
28 Apr 2024, 11:15 am

Reason for Dispute

I have a valid parking permit for this location that was displayed on my dashboard.

Evidence Provided

parking_permit.jpg

Official Dashboard:

Traffic Official
Official

Overview

Report Violation

My Reports

Profile

Logout

Officer Dashboard

My Reports

37



Approved

31

+5 this week



Pending Review

6



Recent Reports

Vehicle: KA-01-AB-1001
Location: Main Street

Approved

Vehicle: KA-01-AB-1002
Location: Park Avenue

Approved

Vehicle: KA-01-AB-1003
Location: Broadway

Pending

Vehicle: KA-01-AB-1004
Location: 5th Avenue

Approved

Vehicle: KA-01-AB-1005
Location: Market St

Approved

Traffic Official
Official

Overview

Report Violation

My Reports

Profile

Logout

Report Parking Violation

Upload Evidence

Take a photo or upload an image of the parking violation



Click to upload or drag and drop

Select Image

Violation Details

Enter information about the parking violation

Vehicle Plate Number

e.g., KA-01-AB-1234

Location

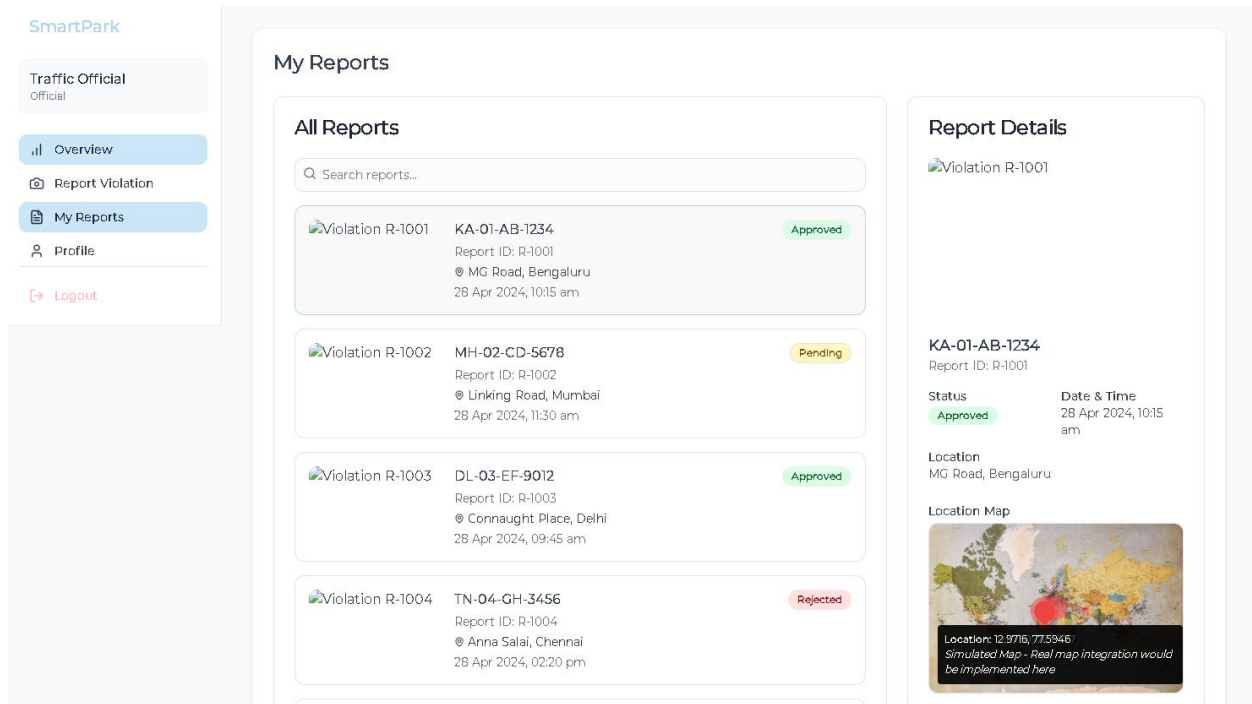
Enter location or use GPS

GPS

Additional Notes

Provide any additional details about the violation

Submit Violation Report



RESULT:

The UI was designed successfully.

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.