

# **RAJALAKSHMI ENGINEERING COLLEGE**

**RAJALAKSHMI NAGAR, THANDALAM – 602 105**



**RAJALAKSHMI  
ENGINEERING COLLEGE**

## **CS23432 SOFTWARE CONSTRUCTION LABORATORY**

### **Laboratory Note Book**

Name: Vijay R

Year / Branch / Section: 2nd YEAR / AIML/AC

University Register No. :2116231501183 College Roll

No: 231501183

Semester: 4rd SEMESTER Academic

Year: 2024-2025

**RAJALAKSHMI ENGINEERING COLLEGE**  
**[AUTONOMOUS]**

**RAJALAKSHMI NAGAR, THANDALAM – 602 105**

**BONAFIDE CERTIFICATE**

Name : ... Vijay R.....

Academic Year : .... 2024-2025.....

Semester : 04

Branch : AIML

**Register No.**

**2116231501183**

Certified that this is the bonafide record of work done by the above student in the

**CS23432 – SOFTWARE CONSTRUCTION** during the year 2024 - 2025.

**Signature of Faculty in-charge**

Submitted for the Practical Examination held on .....

**Internal Examiner**

**External Examiner**

<b>Ex. No.</b>	<b>Title</b>	<b>Page No.</b>
<b>1</b>	<b>Study of Azure DevOps</b>	<b>4</b>
<b>2</b>	<b>Writing Problem Statement – <i>Online Rental System</i></b>	<b>6</b>
<b>3</b>	<b>Designing Project Using Agile-Scrum Methodology with Azure</b>	<b>7</b>
<b>4</b>	<b>Agile Planning – Epics, User Stories, and Sprint Planning</b>	<b>9</b>
<b>5</b>	<b>User Story Creation in Azure DevOps</b>	<b>13</b>
<b>6</b>	<b>Sequence Diagram – Online Banking Transaction Flow</b>	<b>19</b>
<b>7</b>	<b>Class Diagram – Structural Design Using Mermaid.js</b>	<b>23</b>
<b>8</b>	<b>Use Case Diagram – Functional Overview</b>	<b>30</b>
<b>9</b>	<b>Activity Diagram – System Workflow</b>	<b>32</b>
<b>10</b>	<b>Architecture Diagram – Azure Deployment Design</b>	<b>33</b>
<b>11</b>	<b>User Interface Design</b>	<b>34</b>
<b>12</b>	<b>Implementation of Online Banking System in Azure</b>	<b>37</b>

<b>EXP NO: 1</b>  <b>Date :</b>	<b>AZURE DEVOPS ENVIRONMENT SETUP</b>
---------------------------------------	---------------------------------------

## 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:**

- o Azure Repos (Version Control)
  1. Supports Git repositories and Team Foundation Version Control (TFVC).
  2. Provides features like branching, pull requests, and code reviews.
- o Azure Pipelines (CI/CD)
  1. Automates build, test, and deployment processes.
  2. Supports multi-platform builds (Windows, Linux, macOS).
  3. Works with Docker, Kubernetes, Terraform, and cloud providers (Azure, AWS, GCP).
- o Azure Boards (Agile Project Management)
  1. Manages work using Kanban boards, Scrum boards, and dashboards.
  2. Tracks user stories, tasks, bugs, sprints, and releases.
- o Azure Test Plans (Testing)
  1. Provides manual, exploratory, and automated testing.
  2. Supports test case management and tracking.
- o Azure Artifacts (Package Management)
  1. Stores and manages NuGet, npm, Maven, and Python packages.
  2. 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:

Thus, the study for the given problem statement was successfully completed.

**AIM:**

To prepare PROBLEM STATEMENT for your given project.

**PROBLEM STATEMENT:**

In the rapidly evolving urban landscape of India, especially in metropolitan areas like Chennai, finding the right property—be it for purchase, rent, or investment—can be a daunting task. Traditional property search mechanisms are often fragmented, lacking real-time insights, standardized reviews, and a unified user experience for both home-seekers and real estate professionals.

The absence of a centralized, intelligent platform to connect buyers, investors, and builders restricts transparency and efficiency in the real estate ecosystem. Builders struggle to showcase their projects effectively, investors lack data-driven tools for evaluating opportunities, and end-users often experience a disjointed and outdated user interface.

The primary goal of this project is to design and develop “RentXpert” — a futuristic, responsive, and AI-integrated real estate portal for the Chennai metropolitan area. This platform enables users to search, explore, and review residential properties while empowering builders and investors with smart dashboards and visual analytics powered by Microsoft Azure.

RentXpert will feature: ( <https://rentxpert.netlify.app> )

- ❖ A professional and light-themed UI with touch responsiveness and teal aesthetics.
- ❖ Location-based smart search with real-time filters and widgets.
- ❖ Builder and investor dashboards for efficient project management and decision-making.
- ❖ An integrated review and rating system for transparency.
- ❖ All data and content localized to Chennai, Tamil Nadu, India.

The project addresses the core problem of digital real estate inefficiency and aims to revolutionize how users interact with property data using intuitive design and intelligent tools.

## **RESULT:**

Thus, the problem statement for the given problem is successfully written.

## EX NO: 3 DESIGNING PROJECT USING AGILE-SCRUM METHODOLOGY BY USING AZURE.

### **AIM:**

To plan and implement an Agile model for the development of the RentXpert real estate web application.

### **THEORY:**

Agile planning is a core principle of the Agile methodology—a project management approach that embraces incremental and iterative development, constant user feedback, and adaptability to change. Unlike traditional waterfall models, Agile planning avoids rigid up-front designs and instead fosters collaboration, continuous improvement, and high-quality deliverables.

In the RentXpert project, Agile planning was utilized to manage and develop the entire application—from ideation to deployment. The project was divided into meaningful user stories, each corresponding to a specific feature such as property search, builder dashboard, review system, and investor insights. These user stories reflected real-world use cases and guided sprint-based development cycles.

Agile planning for RentXpert included:

- Roadmapping the release of modules (Homepage, Listings, Dashboards, etc.)
- Creating focused sprint plans for UI/UX design, backend integration, and testing
- Incorporating user stories for each widget such as “As a user, I want to search for properties in Velachery” or “As a builder, I want to post new listings”
- Conducting regular feedback sessions and adapting based on usability inputs

This iterative method ensured that each stage of the RentXpert portal aligned with user needs, technological constraints, and design goals.

### **STEPS IN AGILE PLANNING PROCESS:**



- ❖ Define Vision
  - To build a centralized, responsive, and localized real estate platform for Chennai.
- ❖ Set Clear Expectations on Goals
  - Deliver a user-focused web application with smart search, dashboards, and review features.
- ❖ Define and Break Down the Product Roadmap
  - Split the system into modules: Homepage, Listings, Details, Builder Dashboard, Investor Dashboard, Reviews, and About Us.
- ❖ Create Tasks Based on User Stories
  - Example stories:
    - “As a buyer, I want to filter properties by location and price.”
    - “As a builder, I want to add and manage my project listings.”
- ❖ Populate Product Backlog
  - Add user stories into a prioritized backlog hosted on Azure Boards.
- ❖ Plan Iterations and Estimate Effort
  - Break tasks into 1–2 week sprints for UI design, API integration, data modeling, and testing.
- ❖ Conduct Daily Stand-Ups
  - Regular update discussions were held to track progress and blockers.
- ❖ Monitor and Adapt
  - Post-sprint reviews and testing feedback helped in refining design and fixing bugs.

## RESULT:

Thus, the RentXpert web application was successfully designed and developed using Agile-Scrum methodology on the Microsoft Azure platform.

**AIM:**

To build a secure, responsive, and user-friendly real estate web platform (RentXpert) that allows users in Chennai to search and view properties, post listings, write reviews, access investment insights, and interact with builders through a unified web-based interface.

**SCOPE:**

MVP (Minimum Viable Product) Features:

- ❖ User login and registration
- ❖ Smart property search with filters (location, price, BHK, type)
- ❖ Property listing cards with image and key details
- ❖ Property detail view with map and amenities
- ❖ Builder dashboard to post and manage projects
- ❖ Investor dashboard with Smart Investment Score & trends
- ❖ Review and rating system
- ❖ About Us section with team info and mission

**AGILE EPICS & USER STORIES:****[?] Epic 1: User Management**

- ❖ As a user, I want to register and log in securely.
- ❖ As a user, I want to update my personal details.

**[?] Epic 2: Property Listings**

- ❖ As a user, I want to search for properties in different areas of Chennai.
- ❖ As a user, I want to filter properties by price, type, and size.
- ❖ As a user, I want to view detailed information of a selected property.

**[?] Epic 3: Builder Project Posting**

- ❖ As a builder, I want to post new property listings.
- ❖ As a builder, I want to edit or remove my listings.
- ❖ As a builder, I want to view all posted projects in a dashboard.

#### 🔍 **Epic 4: Investor Dashboard**

- ❖ As an investor, I want to view Smart Investment Scores of properties.
- ❖ As an investor, I want to track price trends and potential ROI.

#### 🔍 **Epic 5: Reviews & Ratings**

- ❖ As a user, I want to leave reviews and ratings for properties and builders.
- ❖ As a user, I want to read feedback from other users.

#### 🔍 **Epic 6: Static & Informative Pages**

- ❖ As a user, I want to know about the team and the vision of RentXpert.
- ❖ As a user, I want to access contact and support information.

### **SPRINT PLANNING:**

#### 🔍 **Sprint 1: Project Setup & Authentication**

- ❖ Initialize frontend and backend
- ❖ User registration and login system
- ❖ Teal-themed responsive UI
- ❖ Header sidebar with widget-based navigation

#### 🔍 **Sprint 2: Property Search & Listing View**

- ❖ Smart filters (location, budget, BHK)
- ❖ Property cards with images and details
- ❖ Functional "View Details" modal

#### 🔍 **Sprint 3: Builder Dashboard**

- ❖ Builder login
- ❖ Post/edit/delete property listings
- ❖ Builder's table view of their projects

## 📌 Sprint 4: Investor Dashboard

- ❖ ROI calculation logic
- ❖ Property scoring UI
- ❖ Trend charts (placeholder or sample data)

## 📌 Sprint 5: Reviews & Ratings

- ❖ Review submission form
- ❖ Review listing by users
- ❖ Star-based rating system

## 📌 Sprint 6: Final Polishing & About Us

- ❖ About Us content (team, mission)
- ❖ Footer with clickable LinkedIn developer credit
- ❖ Teal-colored widgets, consistent design styling
- ❖ Make all nav buttons open their respective widget on the same page.

## **FUTURE ENHANCEMENTS:**

- ❖ AI-based property recommendations
- ❖ Integrated chatbot for buyer assistance
- ❖ Scheduling visits with calendar UI
- ❖ Location-aware smart search
- ❖ Email/WhatsApp notification for new listings
- ❖ Admin dashboard for property moderation

## **RESULT:**

Thus, the Agile plan for the RentXpert real estate web application was completed successfully using sprint-based iterations and modular widget design.

## EXP:5

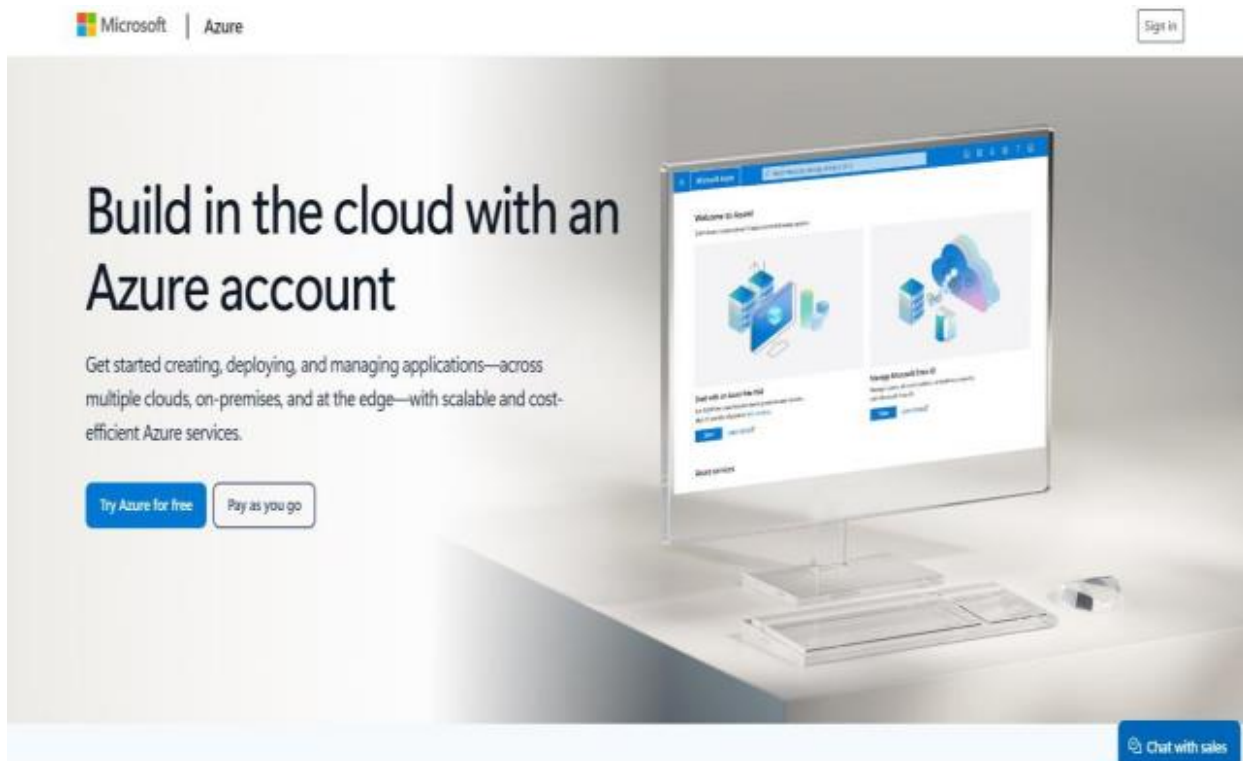
## USER STORIES-CREATION

### AIM:

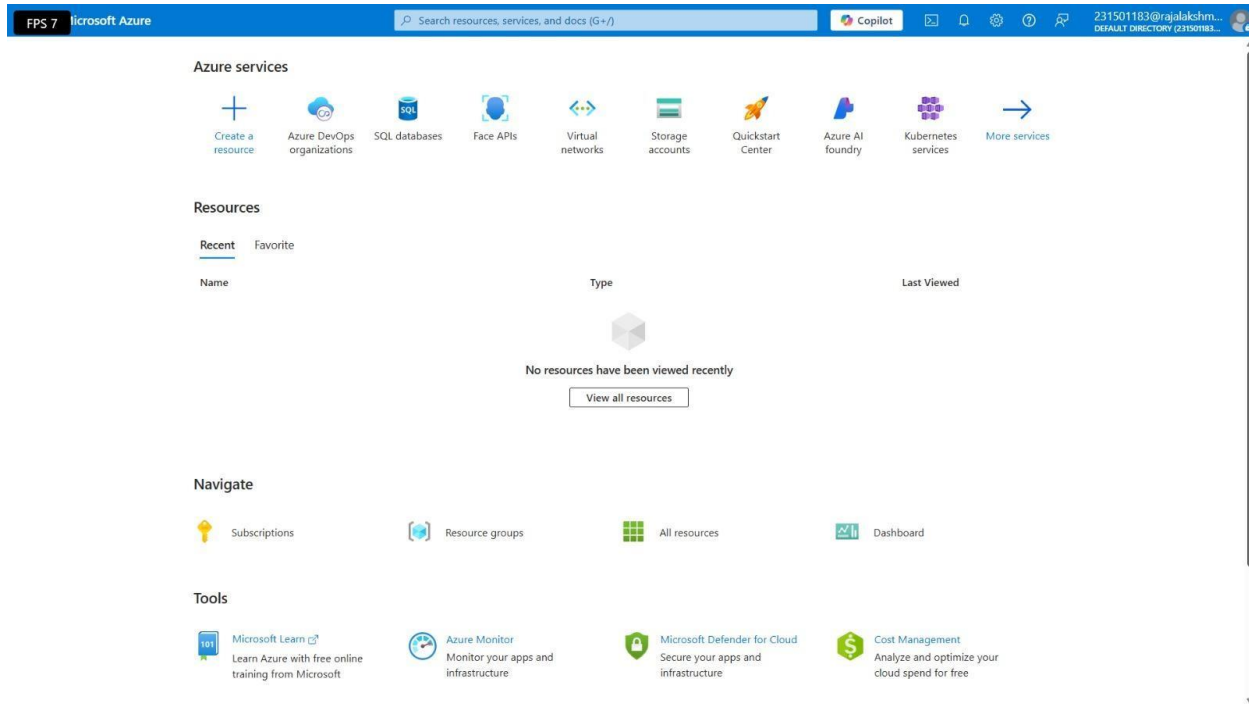
To create user stories for the RentXpert real estate portal using Azure DevOps.

### THEORY:

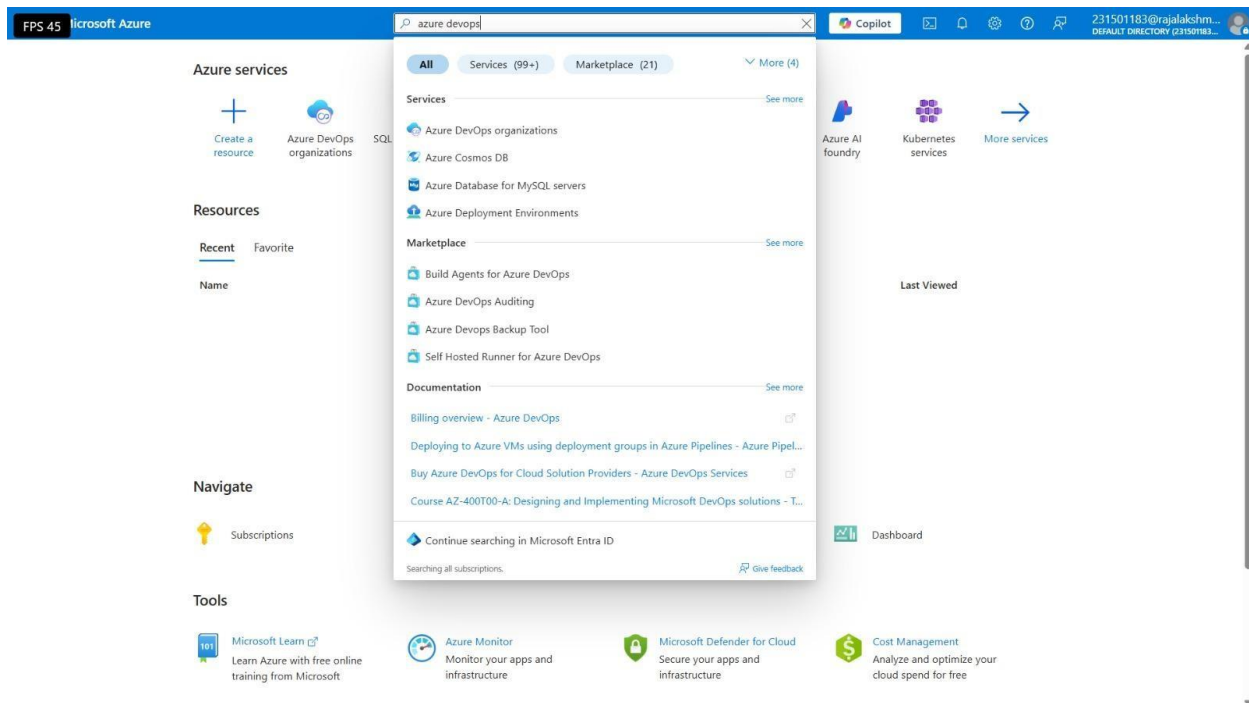
1. Open your web browser and go to the Azure website: <https://azure.microsoft.com/en-in> Sign in using your Microsoft account credentials. If you don't have an account, you'll need to create one.
2. If you don't have a Microsoft account, you can sign up for <https://signup.live.com/?lic=1>



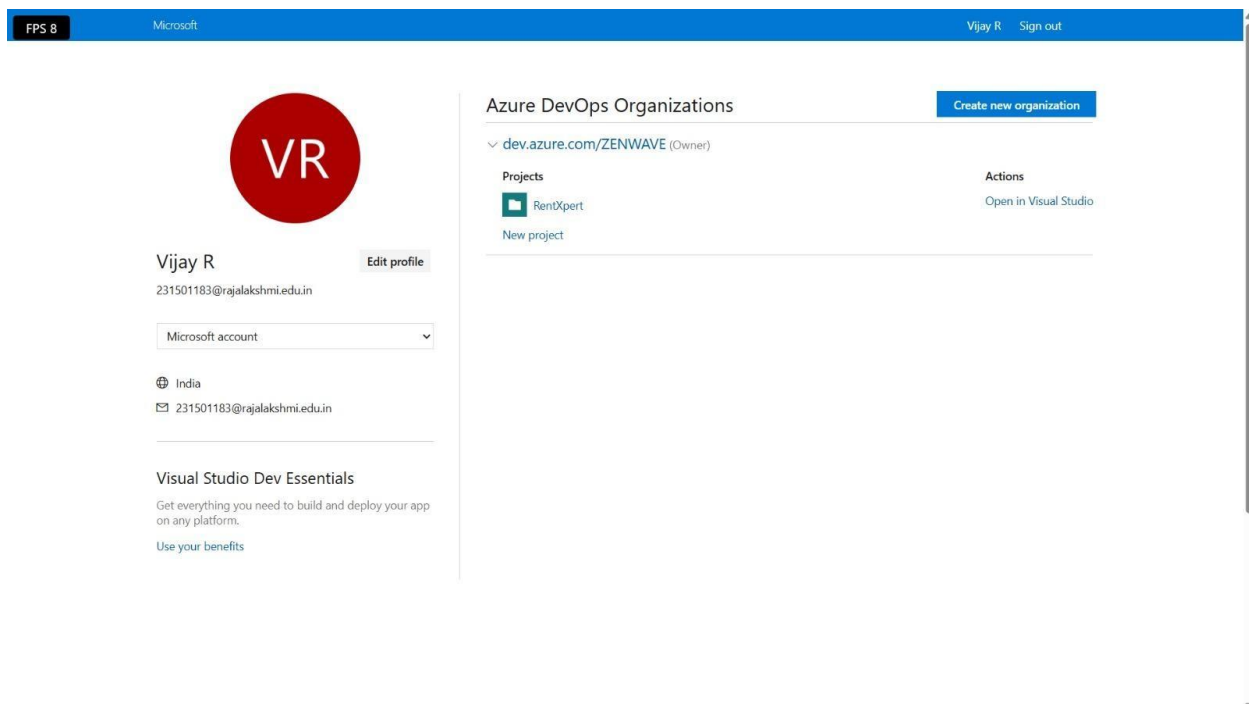
## 1. Azure home page :



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



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



1. 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). 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: Git

Work item process: Agile

Cancel Create

1. 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.

FPS 8 Microsoft Vijay R Sign out

**Vijay R**  
231501183@rajalakshmi.edu.in  
Edit profile

Microsoft account

India  
231501183@rajalakshmi.edu.in

Visual Studio Dev Essentials  
Get everything you need to build and deploy your app on any platform.  
Use your benefits

**Azure DevOps Organizations** Create new organization

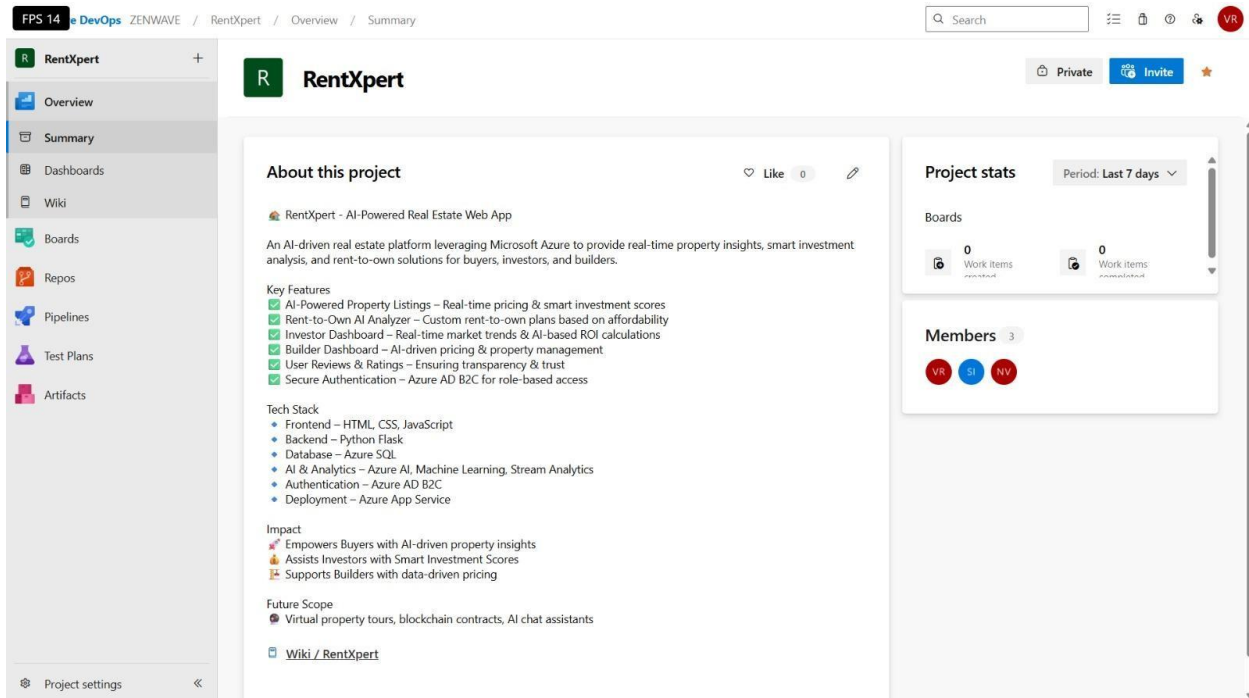
dev.azure.com/ZENWAVE (Owner)

**Projects**

RentXpert  
New project

**Actions**  
Open in Visual Studio

## 1. Project dashboard



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.

FPS 11

DevOps

ZENWAVE

RentXpert

Boards

Work items

Search

RentXpert

Overview

Boards

Work items

Boards

Backlogs

Sprints

Queries

Delivery Plans

Analytics views

Repos

Pipelines

Test Plans

Artifacts

Project settings

Work items

Recently updated

New Work Item

Open in Queries

Column Options

Import Work Items

Recycle Bin

Filter by keyword

Bug

Epic

Feature

Issue

Task

Test Case

User Story

Title	Assigned To	State	Area Path	Tags
RentXpert	Unassigned	New	RentXpert	
Property	Vijay R	Active	RentXpert	
Login & Regist	Vijay R	Active	RentXpert	
Property Detail Page ( )	Vijay R	New	RentXpert	
Property Listing Page ( )	Vijay R	New	RentXpert	
User Login & Registration ( )	Vijay R	New	RentXpert	
Smart Investment Score	Sajiv Jess B I	Active	RentXpert	
Real-Time Market Trends	Sajiv Jess B I	Active	RentXpert	
Homepage ( )	Sajiv Jess B I	New	RentXpert	
Landing Page	Sajiv Jess B I	Active	RentXpert	
Smart Investment Score	Sajiv Jess B I	New	RentXpert	
Investor Dashboard ( )	Sajiv Jess B I	New	RentXpert	
About Us Page ( )	NAVITHA J V	New	RentXpert	
Review System & About Us Page ( )	NAVITHA J V	Active	RentXpert	
Builder Dashboard ( )	NAVITHA J V	New	RentXpert	

# 1. Fill in User Story Details

FPS 10

DevOps

ZENWAVE

RentXpert

Boards

Work items

Search

RentXpert

Overview

Boards

Work items

Boards

Backlogs

Sprints

Queries

Delivery Plans

Analytics views

Repos

Pipelines

Test Plans

Artifacts

Project settings

Work items

Recently updated

Back to Work Items

6 of 17

USER STORY 22

22 User Login & Registration ( )

Vijay R

0 Comments

Add Tag

Save

Follow

Details

State

New

Area

RentXpert

Reason

New

Iteration

RentXpert\Sprint 1

Updated by Vijay R: Apr 8

Description

As a **user**, I want to **securely register and log in** so that I can access my personalized features like saved searches, reviews, and property postings.

Acceptance Criteria

- Users can sign up/log in using **email, phone, or social media**.
- Secure authentication is managed via **Azure AD B2C**.
- Users can recover their password via **forgot password** option.
- Role-based authentication** allows builders, buyers, and investors to access their specific dashboards.

Discussion

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

21 Login & Registration

Updated Apr 14

Active

## RESULT:

The user story for the given problem statement was written successfully.

**AIM:**

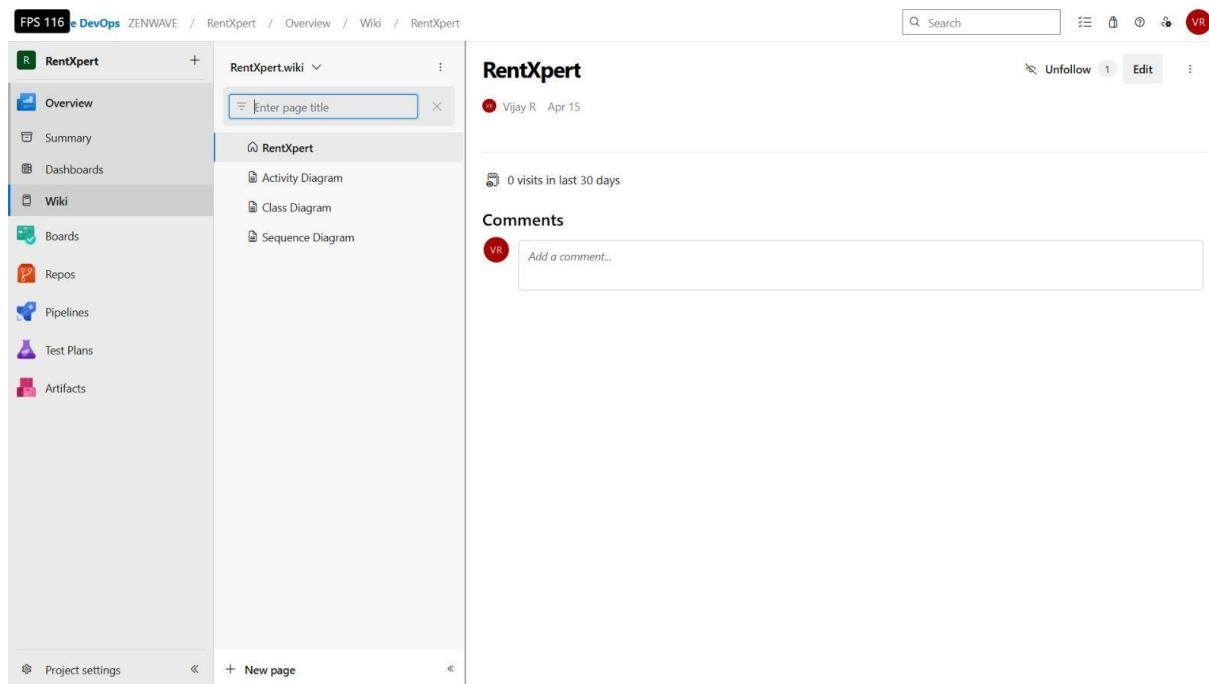
To design a Sequence Diagram by using Mermaid.js for the given problem statement.

**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



## 1. Write code for drawing sequence diagram and save the code.::: mermaid sequence Diagram

```
sequenceDiagram
    participant User
    participant PropertyService
    participant Property
    participant SmartInvestmentScore
    participant Review
    participant Investor
    participant MarketTrends

    Note over User: User logs into the system
    User->>User: login()

    Note over User, PropertyService: User searches for properties
    User->>PropertyService: searchProperties(location, BHK, etc.)
    PropertyService->>Property: getDetails()
    Property->>PropertyService: return details
    PropertyService->>User: return property list

    Note over User, Property: User views property detail
    User->>Property: getDetails()

    alt User is Investor
        User->>Investor: login() as Investor
        Investor->>SmartInvestmentScore: calculateScore(propertyID)
        SmartInvestmentScore-->>Investor: return score
        Investor-->>User: viewSmartScore()
    end

    Note over MarketTrends: System fetches real-time data
    MarketTrends->>Property: getRealTimeTrends()
    Property-->>MarketTrends: data updated

    Note over User, Review: User writes a review
    User->>Review: writeReview(propertyID, rating, comment)

    ...
```

### EXPLANATION:

#### 1. User Login

- User logs into the system.
- Function: login()
- If the credentials were correct.

## 2. Property Search

- User calls search Properties (location, BHK, etc.) on Property Service.
- Property Service fetches property details from Property.
- List of properties is returned to the User.
- Run When a user initiates.

## 3. Viewing a Property

- User views a specific property.
- Calls getDetails() on Property.
- Always runs when a listing is selected.

## 4. IF CONDITION: Is the User an Investor?

If true:

- User logs in as Investor.
- Calls calculateScore(propertyID) on SmartInvestmentScore.
- Score is returned to Investor.
- Investor displays viewSmartScore() to User.

If false:

- Skip this entire block.
- User doesn't see investment **scores**.

## 5. Real-Time Market Trend Updates

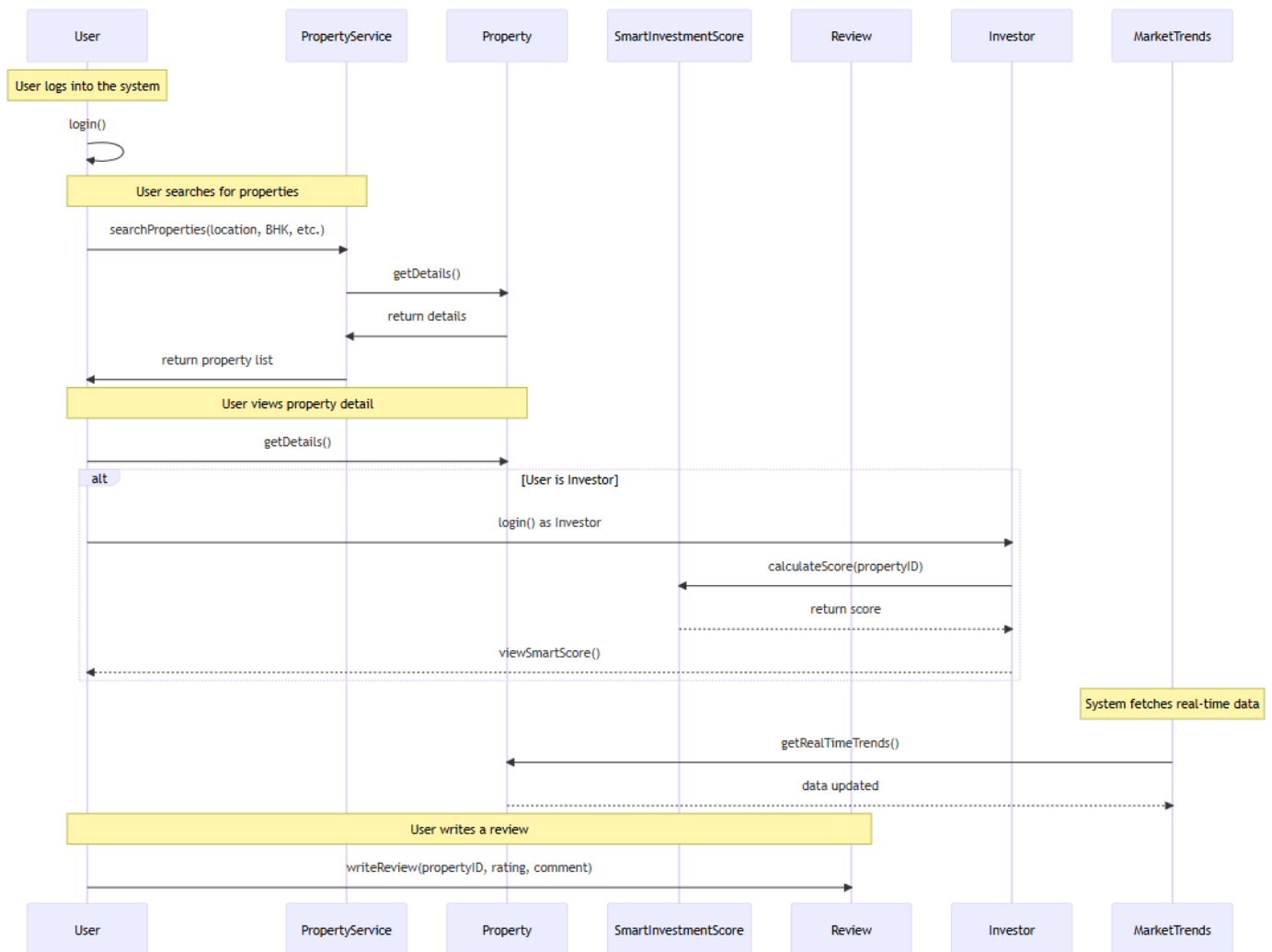
- MarketTrends service auto-fetches real-time data.
- Sends it to Property, which updates its internal dataset.
- This is *not user-triggered*.
- Happens automatically in the background.

## 6. User Writes a Review

- User writes a review for a property.
- Calls writeReview(propertyID, rating, comment) on Review.
- Optional action. User may or may not choose to do this.

4. Click wiki menu and select page:

## SEQUENCE DIAGRAM:



## RESULT:

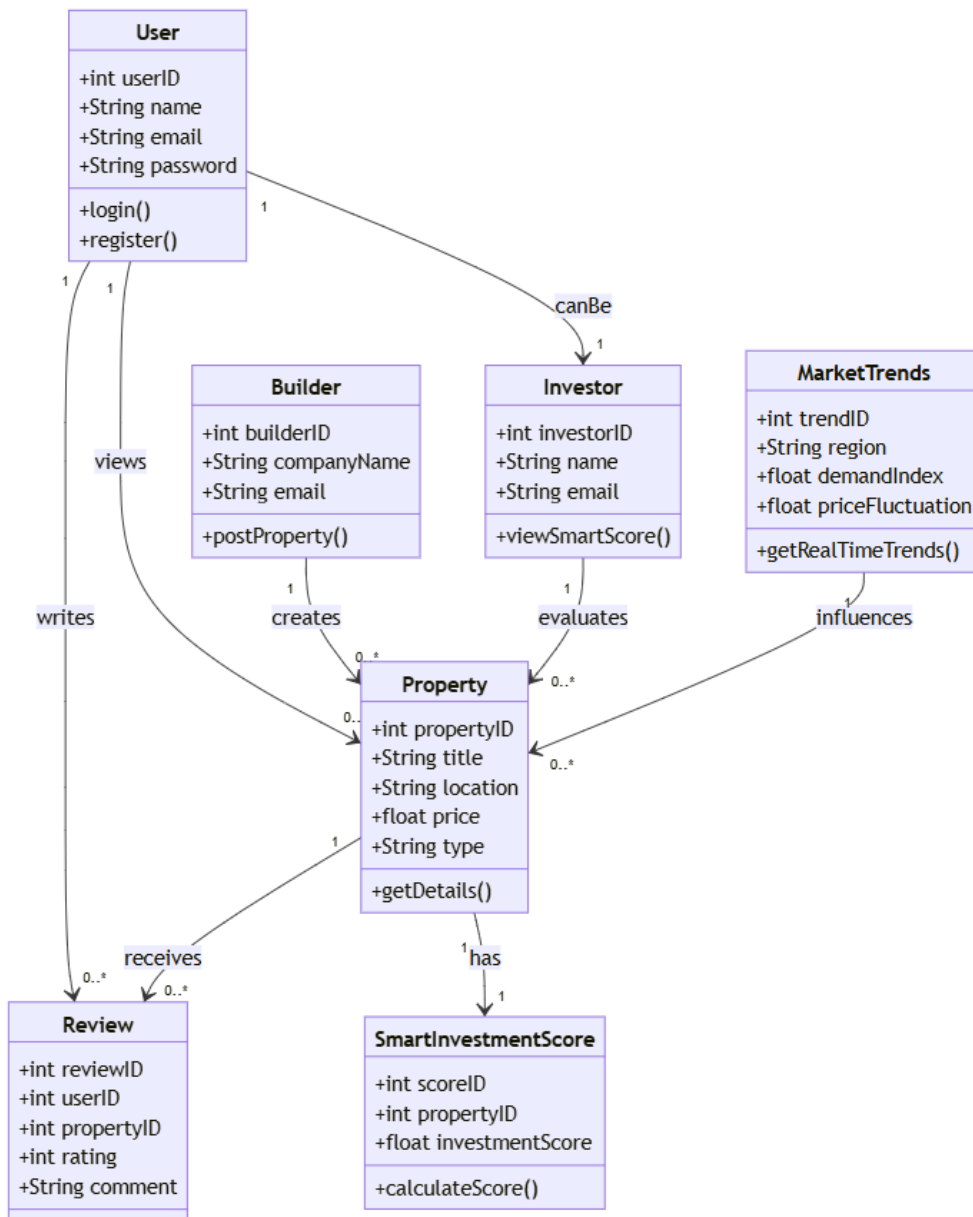
Thus, the sequence diagram for the given problem statement was drawn successfully.

**AIM:**

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

**THEORY:**

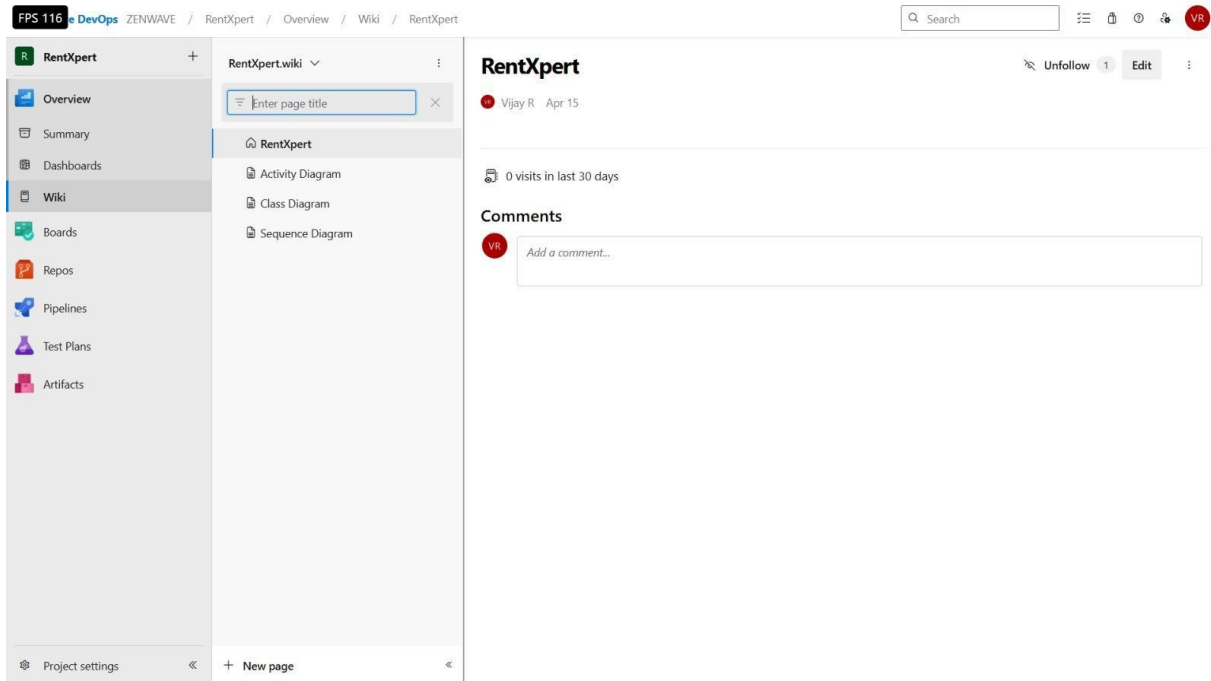
A UML class diagram is a visual tool that represents the structure of a system by showing its classes, attributes, methods, and the relationships between them.





## PROCEDURE:

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



3. Write the mermaid code for the class diagram.

```
::: mermaid
```

```
classDiagram
```

```
%% === Core Classes ===
```

```
class User {
```

```
    +int userID
```

```
    +String name
```

```
    +String email
```

```
    +String password
```

```
    +login()  
2116231501183
```

```
    +register()
```

CS23432

```
}
```

```
class Property {  
    +int propertyID  
    +String title  
    +String location  
    +float price  
    +String type  
    +getDetails()  
}
```

```
class Builder {  
    +int builderID  
    +String companyName  
    +String email  
    +postProperty()  
}
```

```
class Investor {  
    +int investorID  
    +String name  
    +String email  
    +viewSmartScore()  
    2116231501183  
}
```

```
class SmartInvestmentScore {  
  
    +int scoreID  
  
    +int propertyID  
  
    +float investmentScore  
  
    +calculateScore()  
  
}
```

```
class Review {  
  
    +int reviewID  
  
    +int userID  
  
    +int propertyID  
  
    +int rating  
  
    +String comment  
  
}
```

```
class MarketTrends {  
  
    +int trendID  
  
    +String region  
  
    +float demandIndex  
  
    +float priceFluctuation  
  
    +getRealTimeTrends()  
  
}
```

2116231501183

CS23432

%% === Relationships ===

User "1" --> "0..\*" Review : writes

User "1" --> "0..\*" Property : views

User "1" --> "1" Investor : canBe

Property "1" --> "1" SmartInvestmentScore : has

Property "1" --> "0..\*" Review : receives

Builder "1" --> "0..\*" Property : creates

Investor "1" --> "0..\*" Property : evaluates

MarketTrends "1" --> "0..\*" Property : influences

...

## **RESULT:**

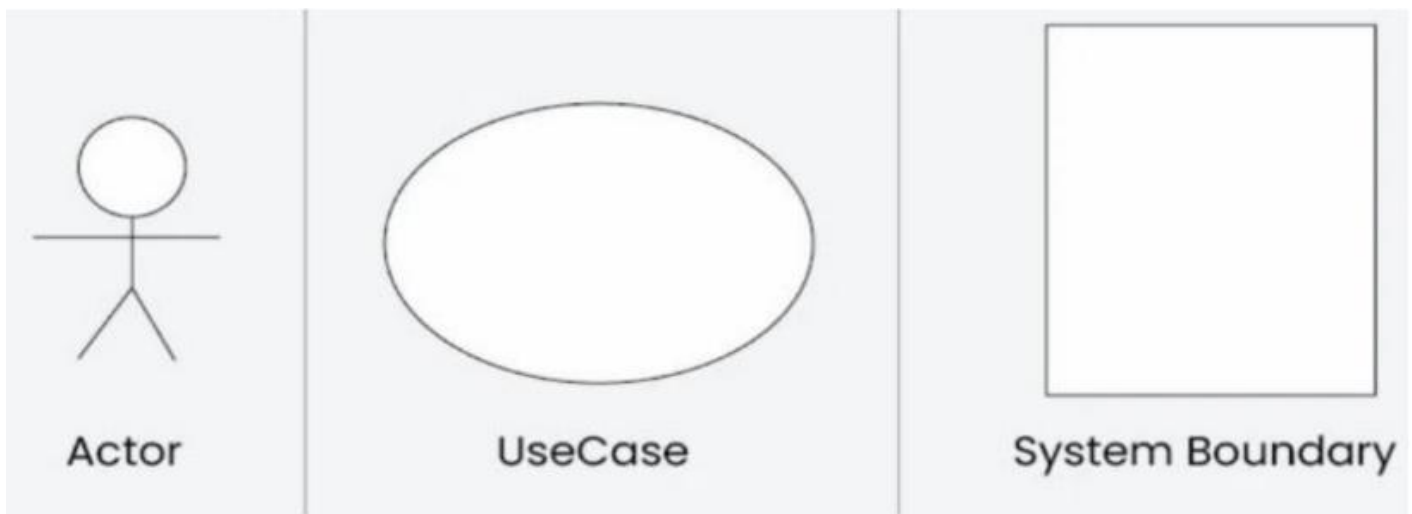
Thus, the use case diagram for the given problem statement was designed successfully.

**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.

2116231501183

CS23432

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 Diagram.

**RESULT:**



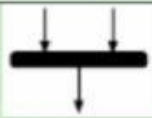


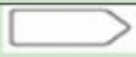





The use case diagram for the given problem statement was designed successfully.

**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 the Azure Wiki

**RESULT:**

Thus, the Activity diagram for the above problem statement done successfully.  
2116231501183

**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

**RESULT:**

Thus, the architecture diagram for the given problem statement was designed successfully.

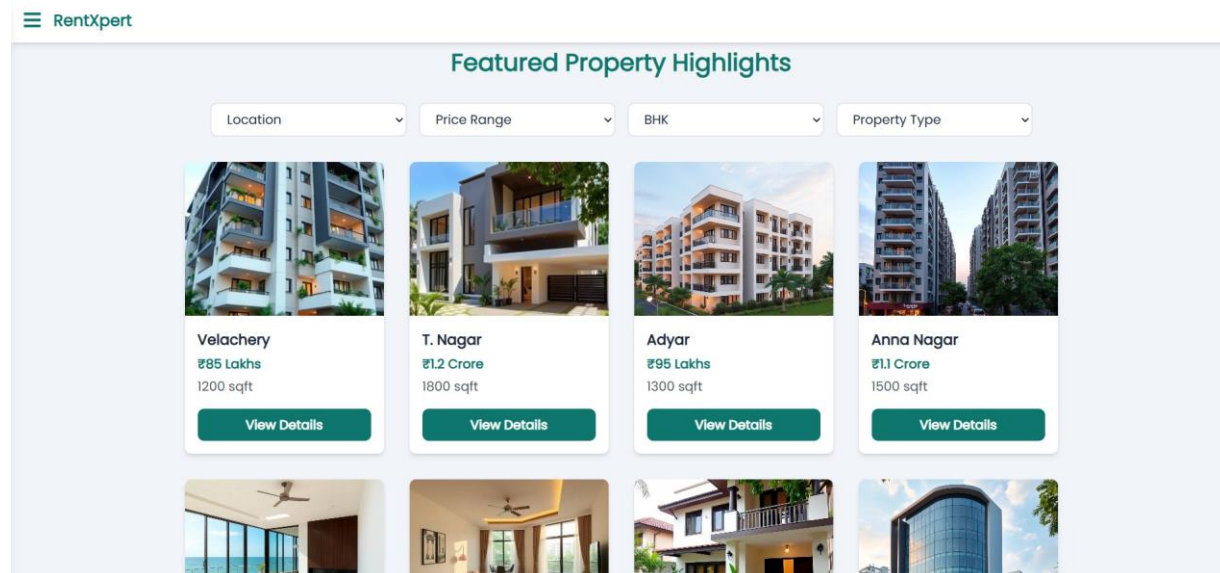
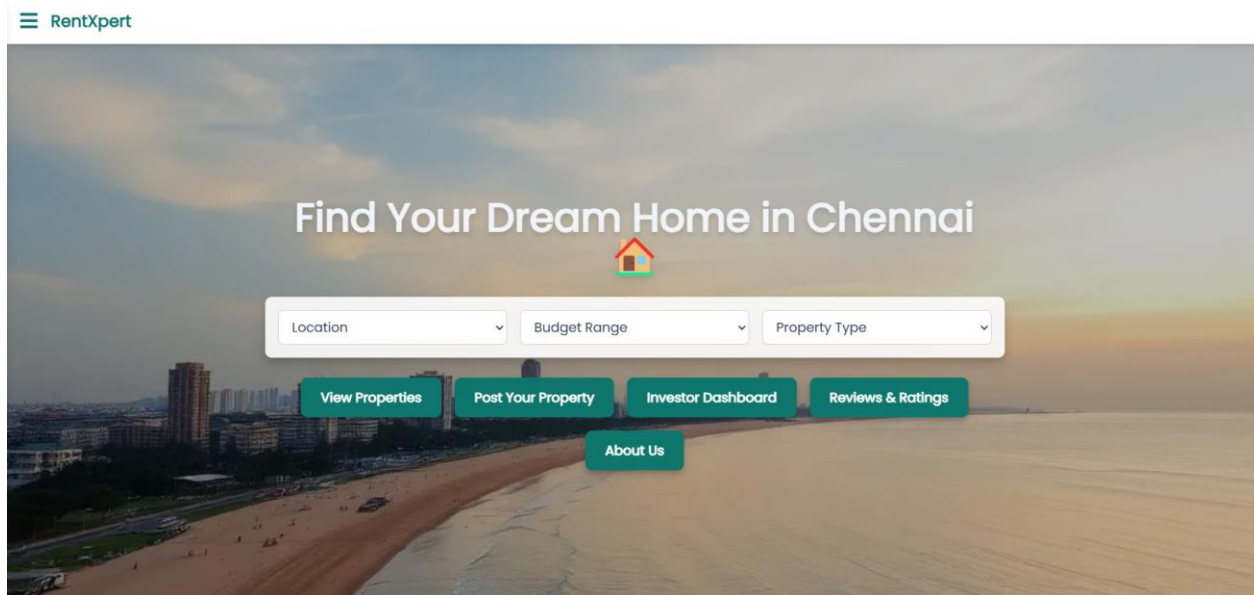


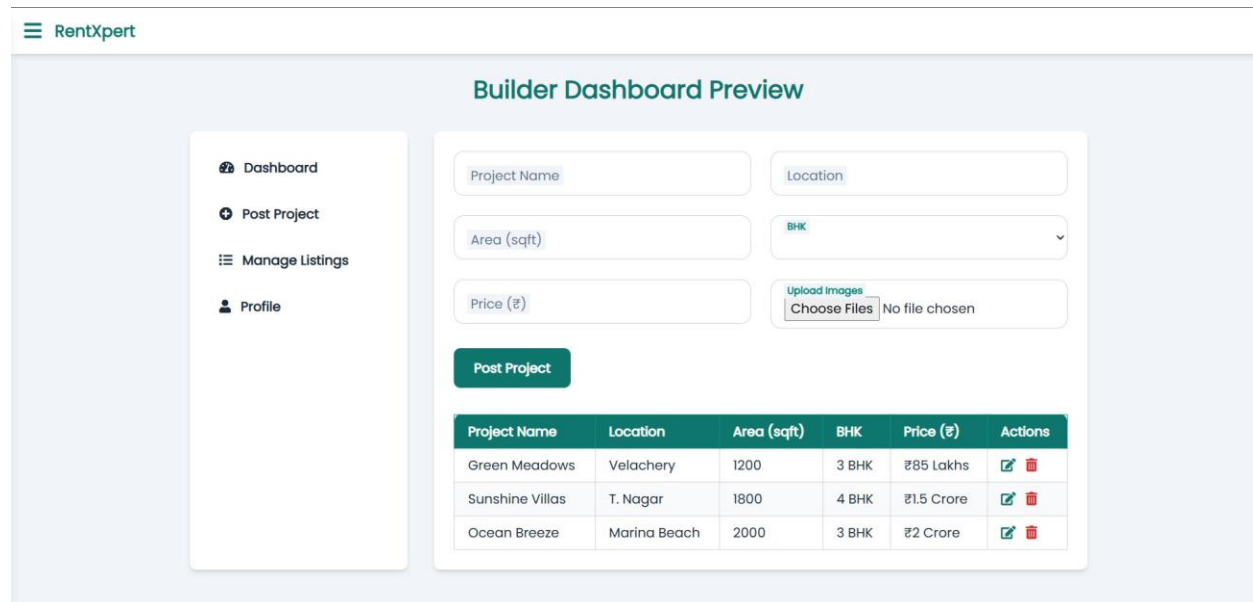
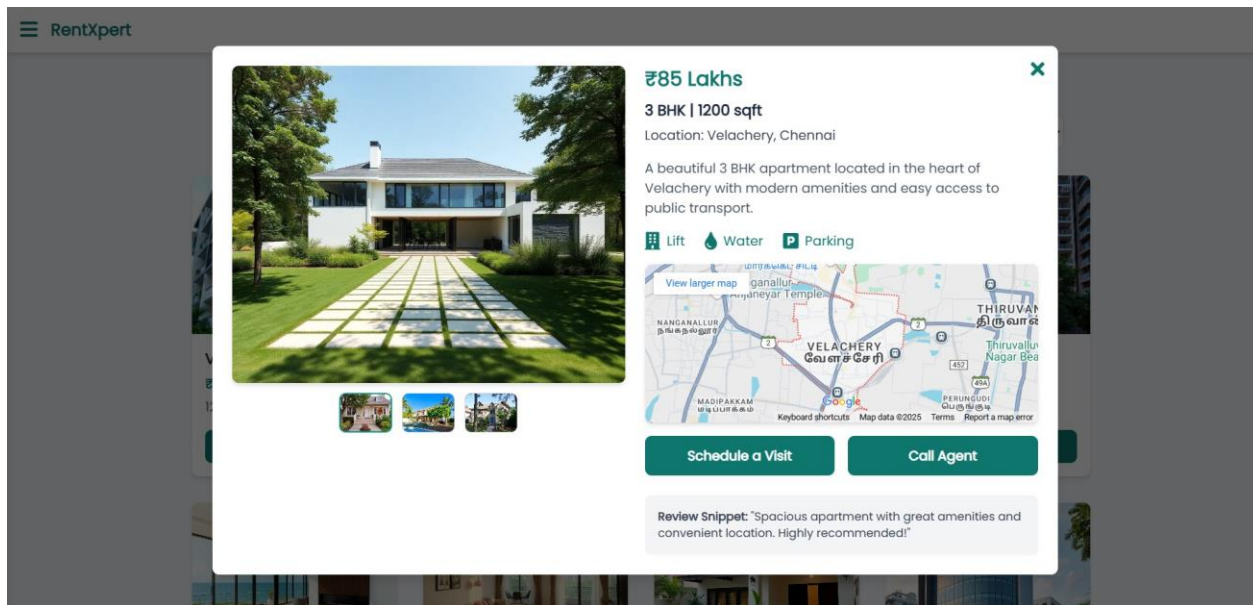
**EX NO: 11**

## **USER INTERFACE**

### **AIM:**

Design User Interface for the given project.





## RESULT:

Thus, the UI for the given problem statement is completed 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 cd`
- 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 implementation of the given problem statement is done 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` `cd`
- 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.