

**RAJALAKSHMI ENGINEERING COLLEGE**  
**RAJALAKSHMI NAGAR, THANDALAM - 602 105**



**RAJALAKSHMI  
ENGINEERING COLLEGE**

**CS23432**

**SOFTWARE CONSTRUCTION**

**Laboratory Record Note Book**

Name : .....

Year / Branch / Section : .....

Register No. : .....

Semester : .....

Academic Year : .....



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

**BONAFIDE CERTIFICATE**

NAME    HARIKRASANTH P      REGISTER NO.    2116231001055

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

Technology [AD/AE]. This Certification is the Bonafide record of work done by the above student in the **CS23432- Software Construction** Laboratory during the year 2024-2025.

Signature of Faculty -in- Charge

Submitted for the Practical Examination held on \_\_\_\_\_

Internal Examiner

External Examiner

**LAB PLAN**  
**CS23432-SOFTWARE CONSTRUCTION LAB**

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

## Course Outcomes (COs)

**Course Name: Software Engineering**

**Course Code: CS23432**

|             |  |
|-------------|--|
| <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<br/>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****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:**

#### **Online Quiz System:**

In the rapidly evolving world of digital education, online platforms are increasingly adopting innovative tools to meet the dynamic needs of learners and educators. One such crucial area is the management of assessments. Institutions and educators are required to create, deliver, and evaluate numerous quizzes efficiently and effectively. However, the traditional manual approach to managing quizzes — including preparing questions, assigning difficulty levels, setting timers, evaluating responses, and generating results — is often time-consuming, error-prone, and lacks scalability.

Many educators face challenges such as inconsistent formatting, repetitive tasks, delayed evaluations, and lack of real-time feedback while conducting assessments. These issues not only reduce teaching efficiency but also affect the learner's experience due to slow feedback and poor performance tracking. A slow and inefficient quiz process can reduce student engagement and hinder academic progress.

To address these issues, there is a need for a smart and scalable **Online Quiz System** that simplifies and automates the entire quiz workflow. This system should offer features like bulk question uploads via CSV/Excel, timed quizzes, automatic grading, category-based quiz creation, real-time feedback, performance analytics, and error highlighting — ensuring faster, more accurate, and seamless assessments.

By implementing such a system, educators can manage assessments more efficiently, reduce manual effort, and concentrate on enhancing the quality of instruction. It also ensures that learners receive immediate and accurate feedback, promoting better understanding, motivation, and academic performance.

Overall, this project aims to develop a reliable and user-friendly **Online Quiz System** that not only streamlines the quiz management process but also contributes to the growth, engagement, and success of digital learning environments.

### **Result:**

The problem statement was written successfully.

**EX NO: 3****AGILE PLANNING****Aim:**

To prepare an Agile Plan.

**THEORY**

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

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

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

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

- Steps in Agile planning process
  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**

#### **Aim:**

To create User Stories

#### **THEORY**

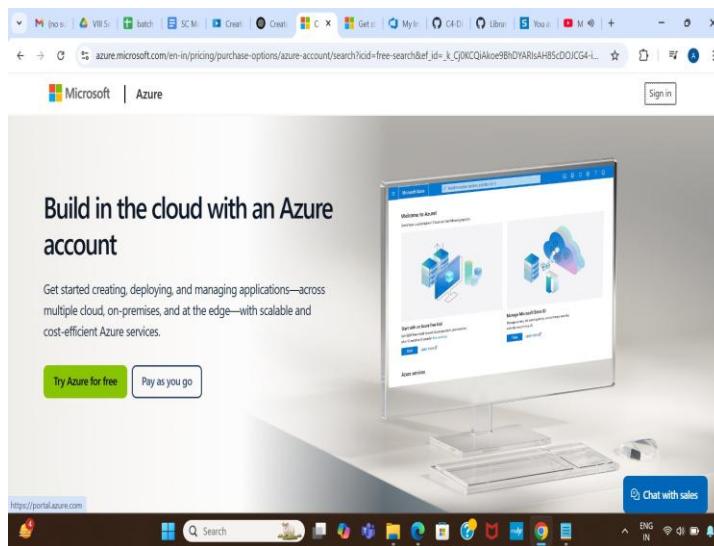
A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer.

User story template

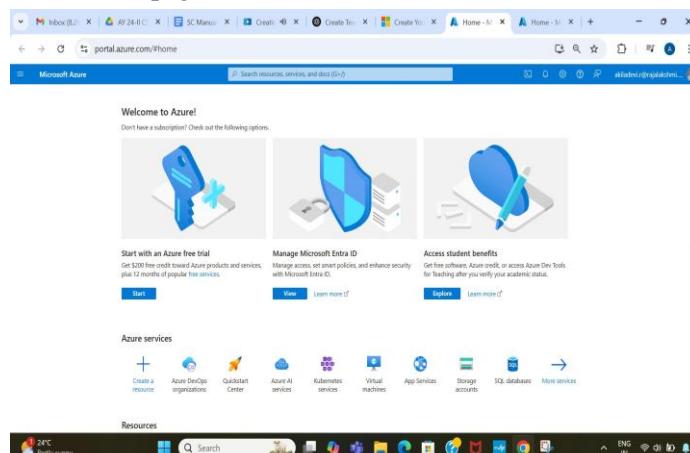
**"As a [role], I [want to], [so that]."**

#### **Procedure:**

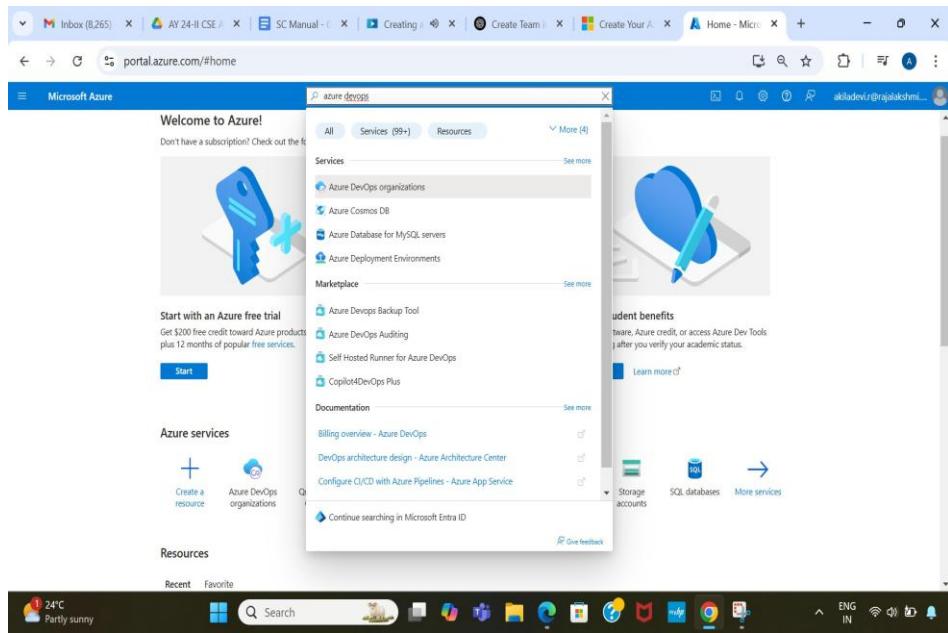
1. Open your web browser and go to the Azure website:  
<https://azure.microsoft.com/en-in> Sign in using your Microsoft account credentials. If you don't have an account, you'll need to create one.
2. If you don't have a Microsoft account, you can sign up for  
<https://signup.live.com/?lic=1>



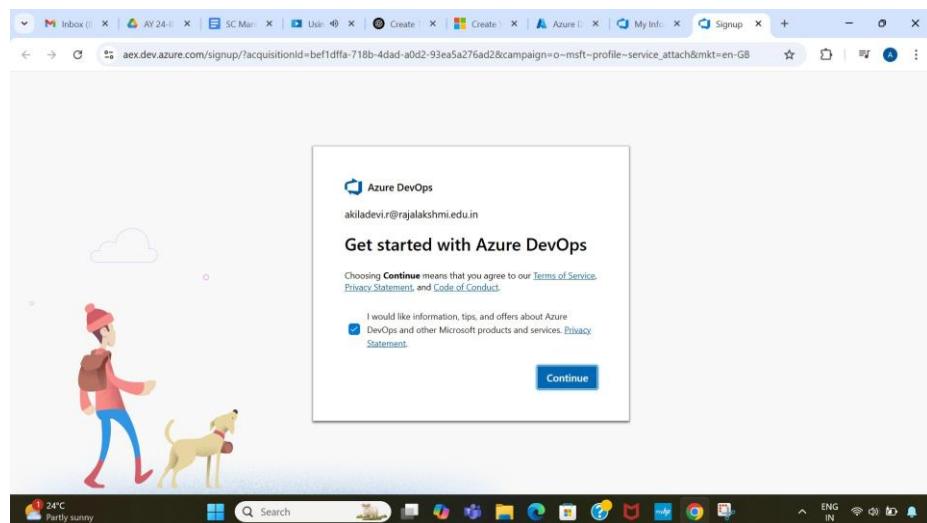
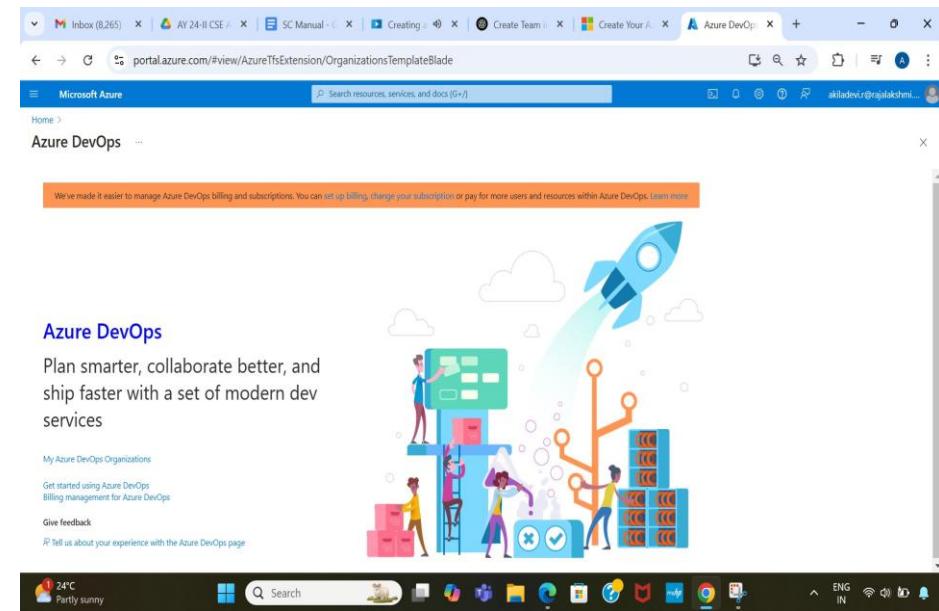
3. Azure home page

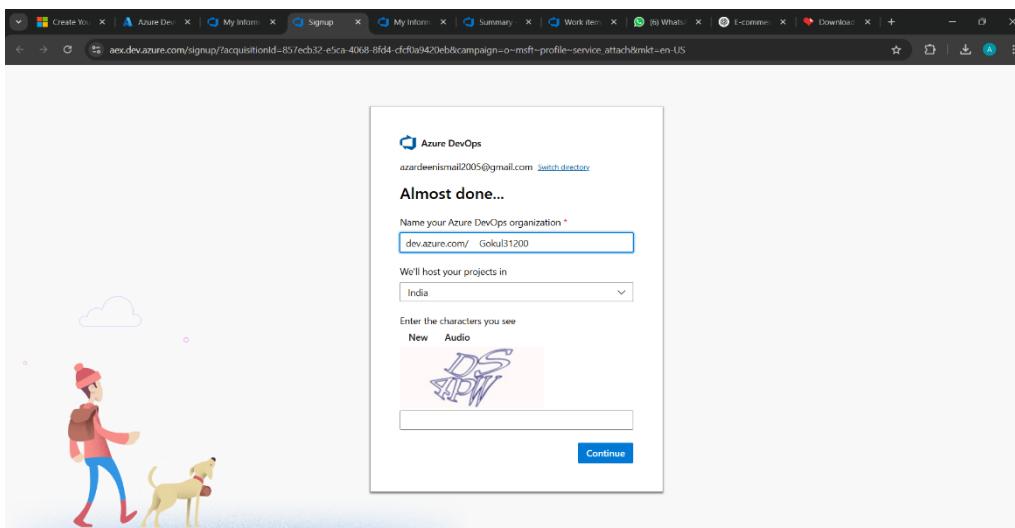


4. Open DevOps environment in the Azure platform by typing Azure DevOps Organizations in the search bar.



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.





## Organization

### 6. Create the First Project in Your

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.

The screenshot shows the Microsoft Azure DevOps Organizations interface. At the top, there's a navigation bar with 'Microsoft' on the left and 'Hariharan M' and 'Sign out' on the right. Below the navigation bar is a user profile card for 'Hariharan M' with a red circular icon containing 'HM'. The main content area is titled 'Azure DevOps Organizations' and shows two organization entries:

- dev.azure.com/231001052** (Owner)
  - Actions: Create new organization, Open in Visual Studio
  - Description: Create a Team Project and start collaborating with your team now! New project
- dev.azure.com/dineshkumard980** (Member)
  - Actions: Open in Visual Studio, Manage security, Browse extensions, Leave
  - Description: Projects: CAT, Online Quiz System

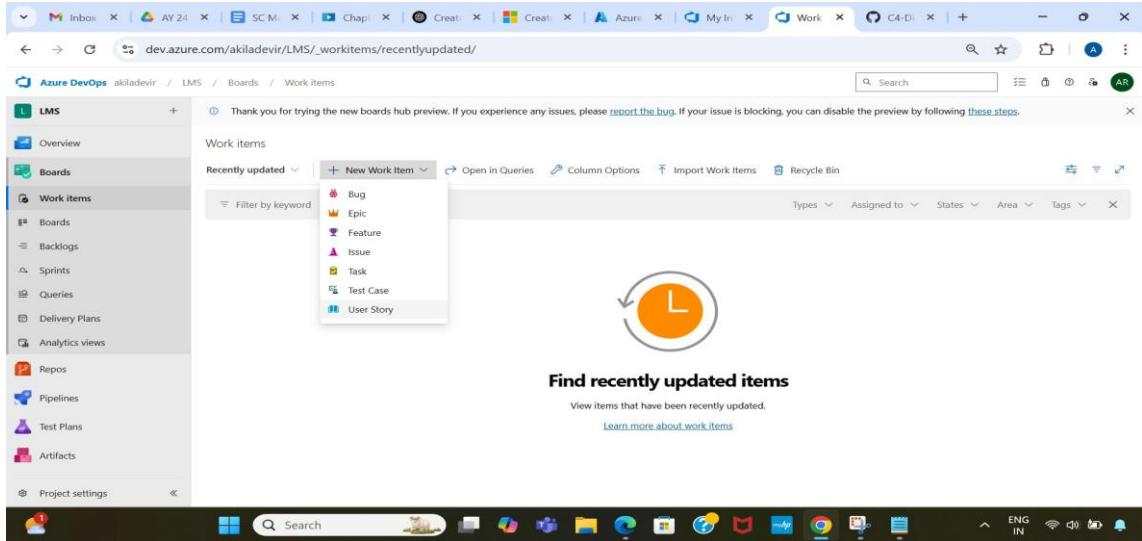
On the left side, there's a sidebar with 'Visual Studio Dev Essentials' and a 'Use your benefits' section.

## 8. Project dashboard

The screenshot shows the 'Online Quiz System' project dashboard in Azure DevOps. The top navigation bar includes 'Azure DevOps', 'dineshkumard980', 'Online Quiz System', 'Overview', and 'Summary'. The main content area has a title 'OS Online Quiz System' and a 'Project stats' section with 'Period: Last 7 days' and a 'Public' status indicator. The left sidebar lists project management sections: Overview, Summary, Dashboards, Wiki, Boards, Repos, Pipelines, and Artifacts. The 'About this project' section provides a detailed description of the platform's features, including its purpose, key features, use cases, and technical stack. The bottom of the screen shows a Windows taskbar with various icons and system status indicators.

## 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.



## 9. Fill in User Story Details

A screenshot of the Azure DevOps User Story creation screen. The URL is "dev.azure.com/dineshkumard980/\_workitems/recentlyupdated/16". The sidebar on the left shows links for Online Quiz System, Overview, Boards, Work items, Backlogs, Sprints, Queries, Delivery Plans, Artifacts, and Project settings. The main area is titled "16 'Admin Access for Quiz Creation and Student Monitoring'" and contains a "Description" field with the text: "As an admin, I want to access admin-only features like quiz creation and student monitoring, so that regular users cannot perform admin tasks." Below this is an "Acceptance Criteria" section with two bullet points: "Admin Access to Features" and "Restricted Access for Regular Users". To the right, there are four tabs: "Planning", "Deployment", "Classification", and "Development". The "Planning" tab shows "Story Points" (2), "Priority" (2), and "Risk". The "Deployment" tab has a note about tracking releases. The "Classification" tab shows "Value area" (Business). The "Development" tab has a note about linking to Azure Repos. At the bottom, there are buttons for "Save and Close", "Follow", and "Details".

## Result:

The user story was written successfully.

## **EX NO: 5**

### **SEQUENCEDIAGRAM**

#### **AIM:**

To design a Sequence Diagram by using Mermaid.js

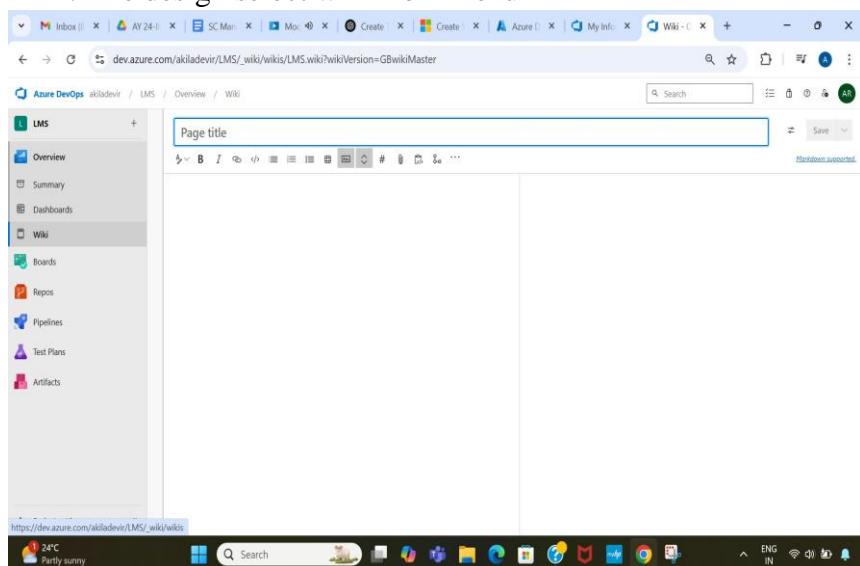
#### **THEORY:**

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

```
    participant Admin
    participant Student
    participant QMS as Quiz Management System
    participant DB as Database
```

%% Admin Flow

Admin->>QMS: Login

QMS->>DB: Validate Admin Credentials

DB-->>QMS: Admin Validated

QMS-->>Admin: Login Success

Admin->>QMS: Create Quiz (title, desc)

QMS->>DB: Insert Quiz Details

DB-->>QMS: Quiz Created  
Admin->>QMS: Add Question (Q, Options, Answer)  
QMS->>DB: Insert Question  
DB-->>QMS: Question Added  
Admin->>QMS: Publish Quiz  
QMS->>DB: Update Quiz Status  
DB-->>QMS: Status Updated

%% Student Flow  
Student->>QMS: Login  
QMS->>DB: Validate Student Credentials  
DB-->>QMS: Student Validated  
QMS-->>Student: Login Success  
Student->>QMS: View Available Quizzes  
QMS->>DB: Fetch Published Quizzes  
DB-->>QMS: Quiz List  
QMS-->>Student: Display Quiz List  
Student->>QMS: Attempt Quiz (selected)  
QMS->>DB: Fetch Questions  
DB-->>QMS: Questions  
QMS-->>Student: Display Questions  
Student->>QMS: Submit Answers  
QMS-->>QMS: Evaluate Answers  
QMS->>DB: Store Result  
DB-->>QMS: Result Stored  
Student->>QMS: View Result  
QMS->>DB: Fetch Result  
DB-->>QMS: Result  
QMS-->>Student: Display Result

### **Explanation:**

participant defines the entities involved.  
->> represents a direct message.  
-->> represents a response message.  
+ after ->> activates a participant  
- after -->> deactivates a participant.  
alt / else for conditional flows loop can  
be used for repeated actions.

-> Solid line without arrow  
--> Dotted line without arrow  
->> Solid line with arrowhead  
-->> Dotted line with arrowhead  
<<->> Solid line with bidirectional arrowheads (v11.0.0+)

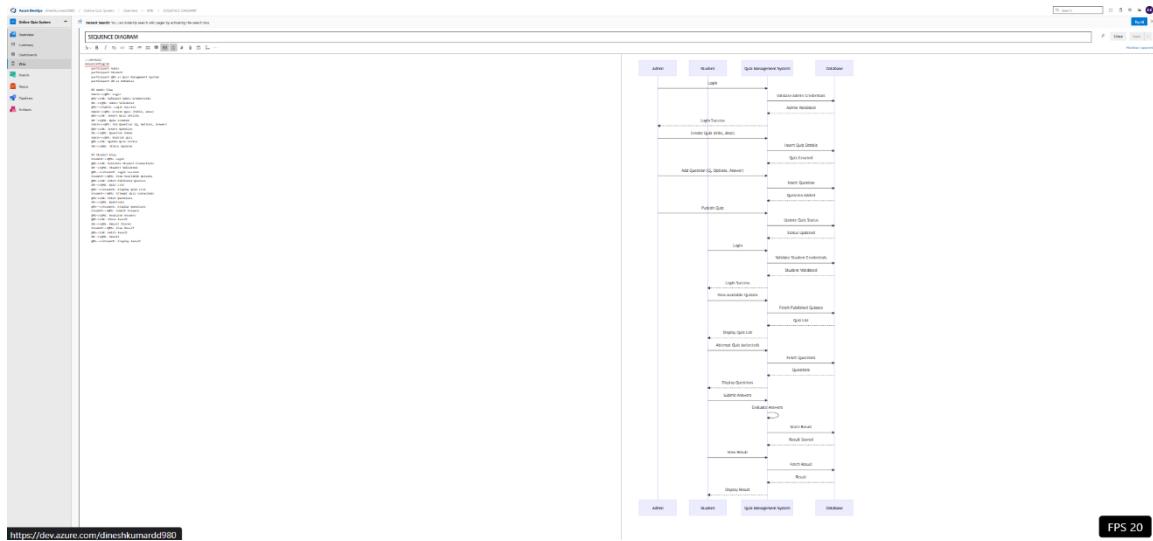
<<-->> Dotted line with bidirectional arrowheads (v11.0.0+)

-x Solid line with a cross at the end

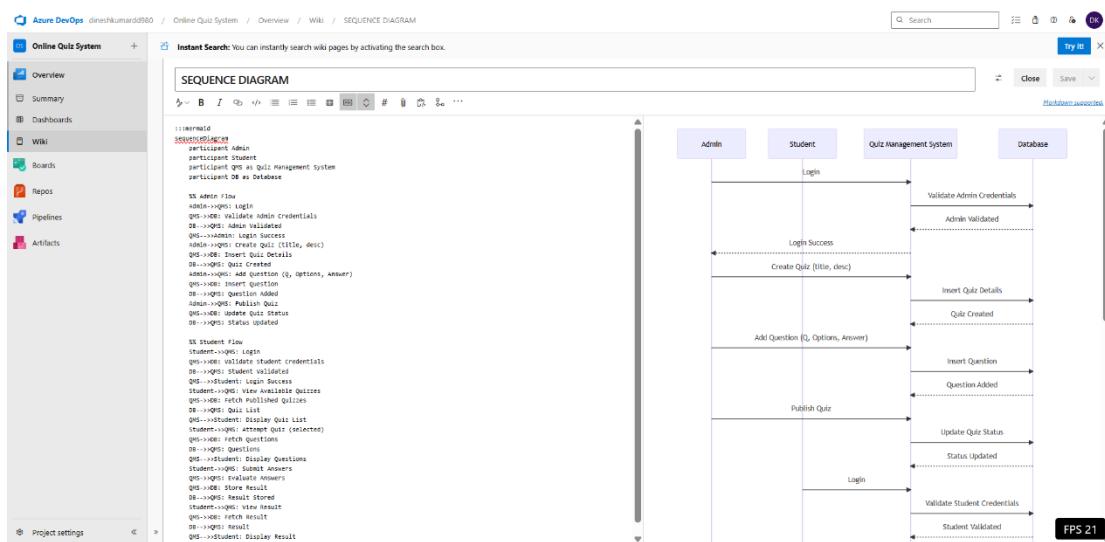
--x Dotted line with a cross at the end

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

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



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



#### Result:

The sequence diagram was drawn successfully.

## EX NO. 6

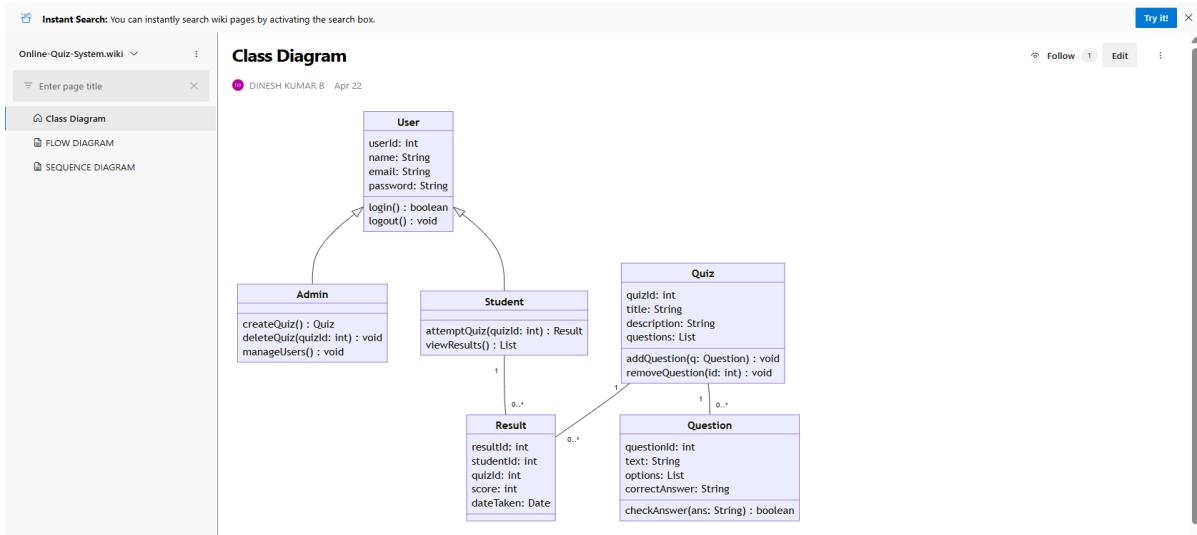
### CLASS DIAGRAM

#### AIM :-

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

#### THEORY

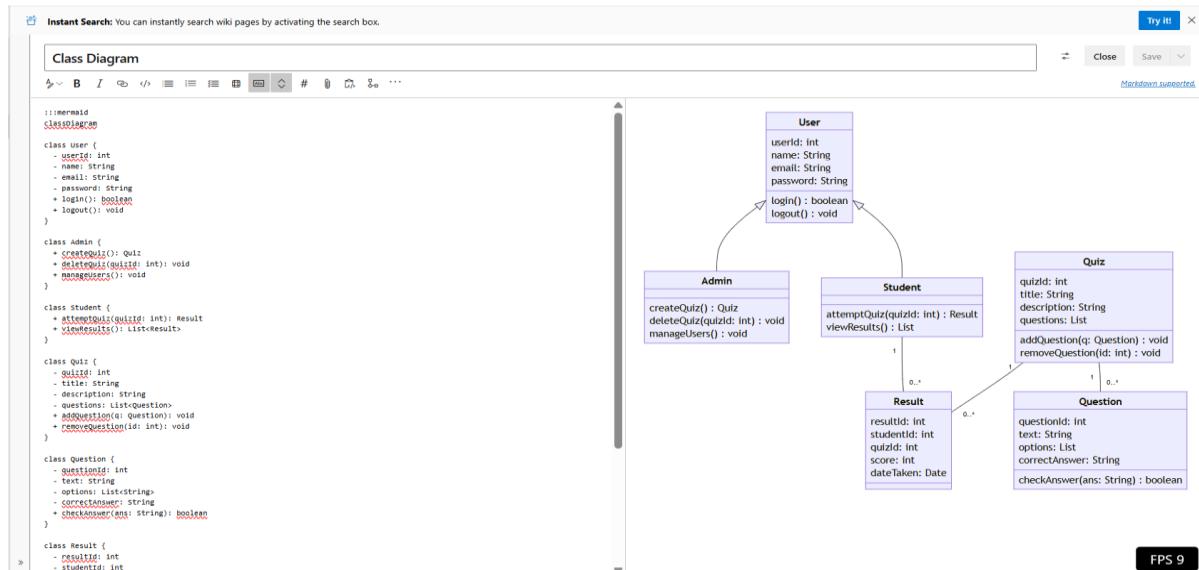
A UML class diagram is a visual tool that represents the structure of a system by showing its classes, attributes, methods, and the relationships between them.



Notations in class diagram

#### Procedure:

1. Open a project in Azure DevOps Organisations.



2. To design select wiki from menu
- 3.
4. Write code for drawing class diagram and save the code

```
:::mermaid
```

```
classDiagram
```

```
class User {  
    - userId: int  
    - name: String  
    - email: String  
    - password: String  
    + login(): boolean  
    + logout(): void  
}
```

```
class Admin {  
    + createQuiz(): Quiz  
    + deleteQuiz(quizId: int): void  
    + manageUsers(): void  
}
```

```
class Student {  
    + attemptQuiz(quizId: int): Result  
    + viewResults(): List<Result>  
}
```

```
class Quiz {  
    - quizId: int  
    - title: String  
    - description: String
```

```
- questions: List<Question>  
+ addQuestion(q: Question): void  
+ removeQuestion(id: int): void  
}
```

```
class Question {  
    - questionId: int  
    - text: String  
    - options: List<String>  
    - correctAnswer: String  
    + checkAnswer(ans: String): boolean  
}
```

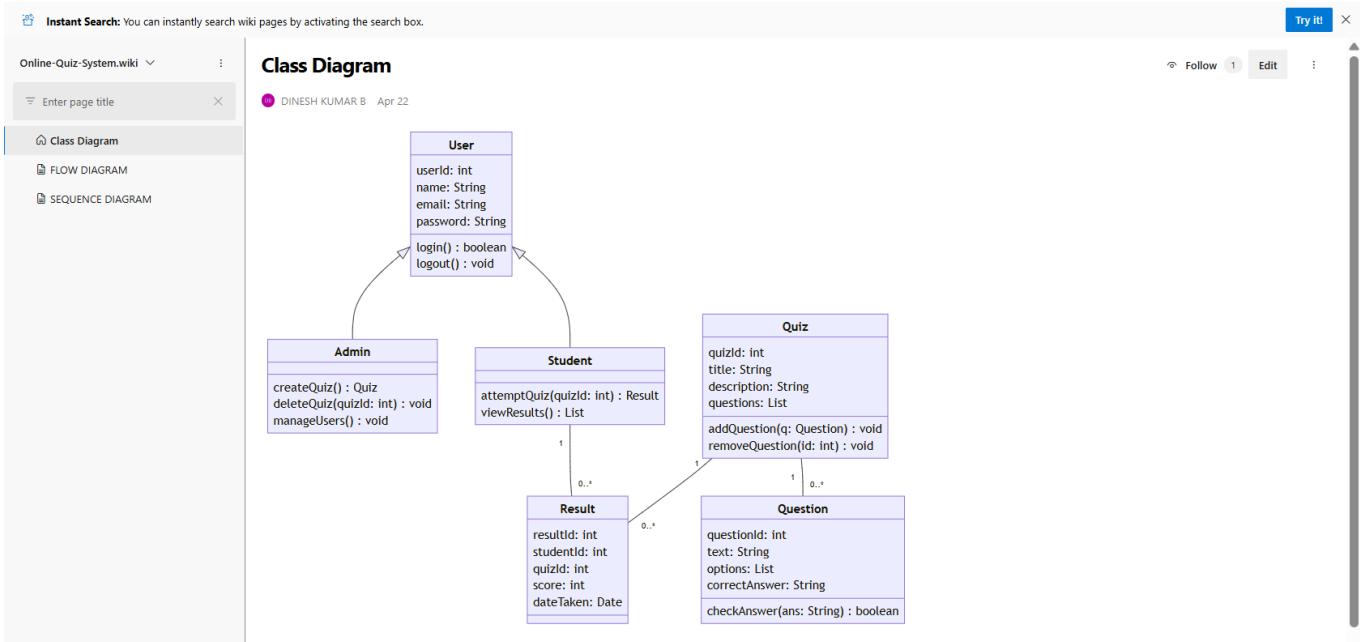
```
class Result {  
    - resultId: int  
    - studentId: int  
    - quizId: int  
    - score: int  
    - dateTaken: Date  
}
```

```
User <|-- Admin  
User <|-- Student  
Quiz "1" -- "0..*" Question  
Student "1" -- "0..*" Result  
Quiz "1" -- "0..*" Result
```

⋮

## Relationship Types

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



## Result:

The use case diagram was designed successfully.

## EX NO: 7

### USECASE DIAGRAM

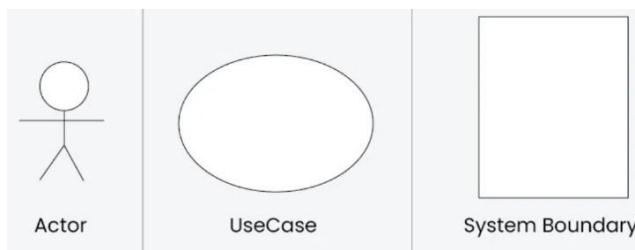
#### Aim:

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

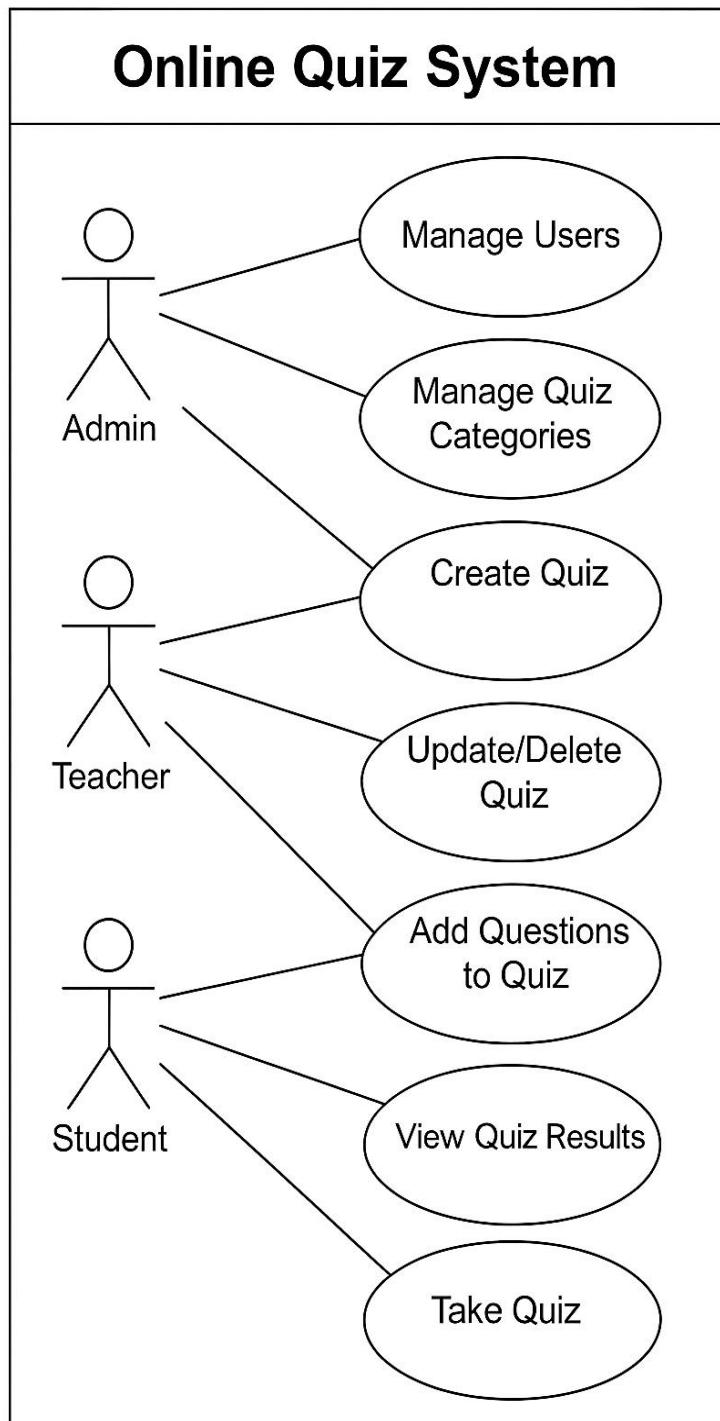
Option 1: Add to Azure DevOps Wiki

- 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

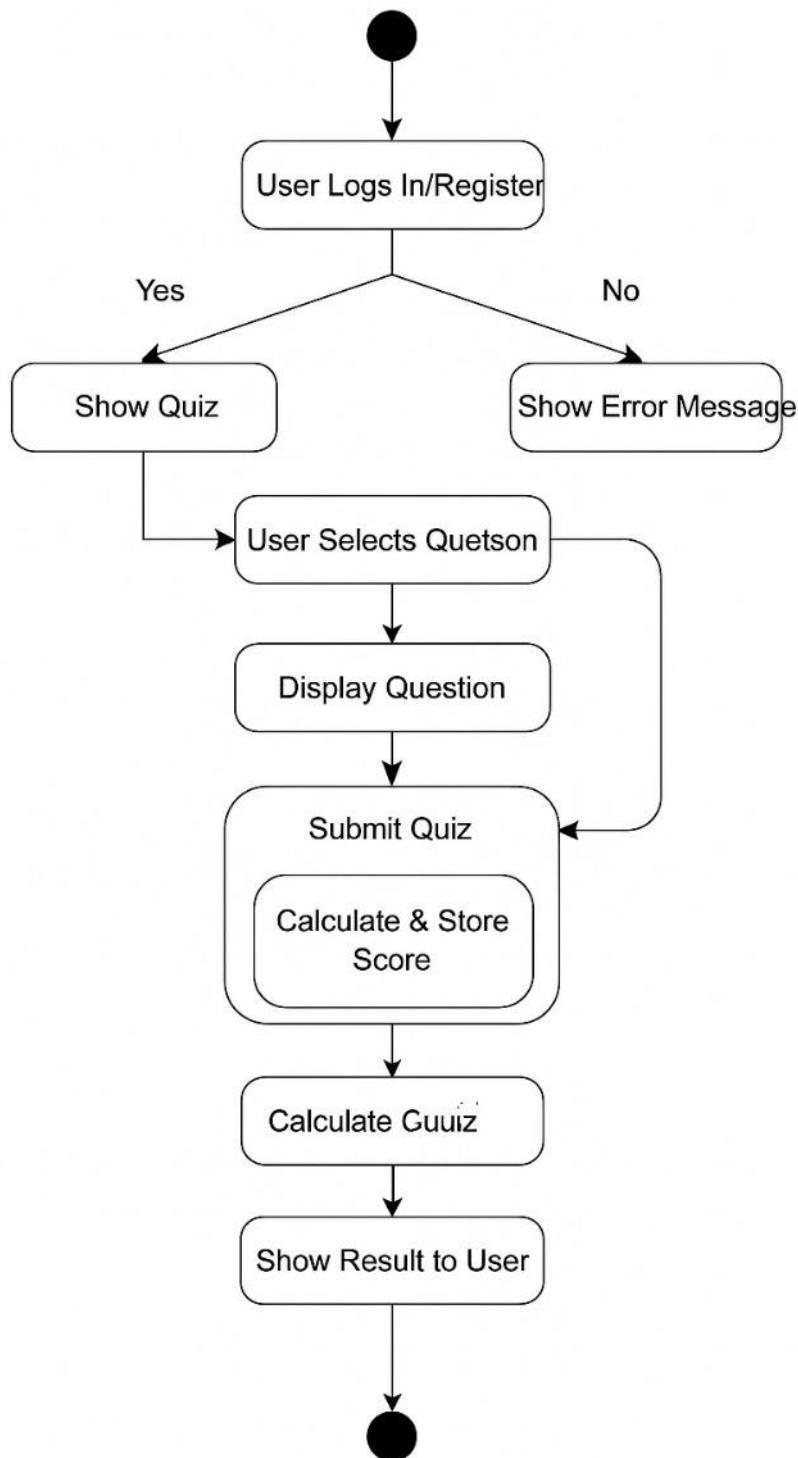
Activity diagrams are an essential part of the Unified Modelling Language (UML) that help visualize workflows, processes, or activities within a system. They depict how different actions are connected and how a system moves from one state to another.

| 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

# Online Quiz System



## Result:

The activity diagram was designed successfully

## EX NO. 9

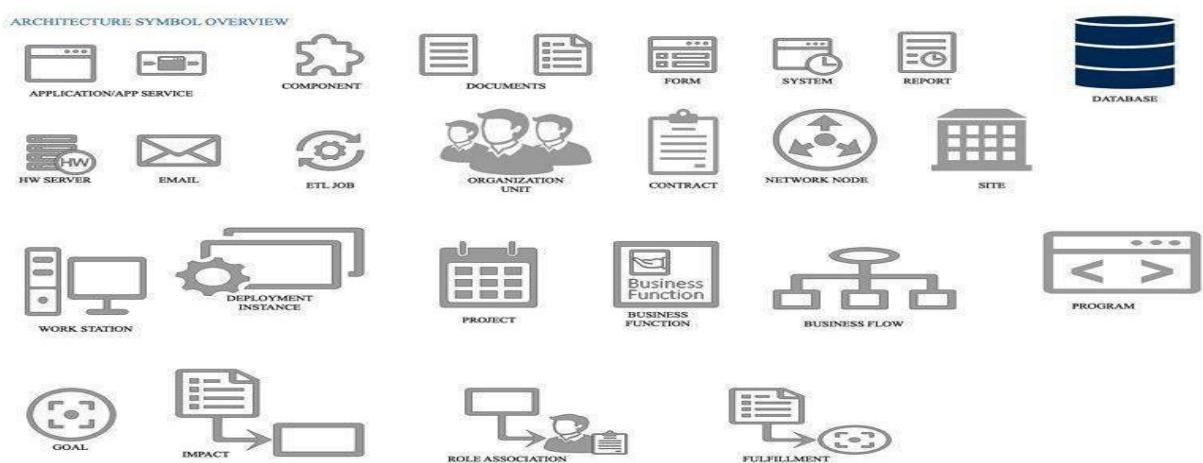
### ARCHITECTURE DIAGRAM

#### Aim:

Steps to draw the Architecture Diagram using draw.io.

#### Theory:

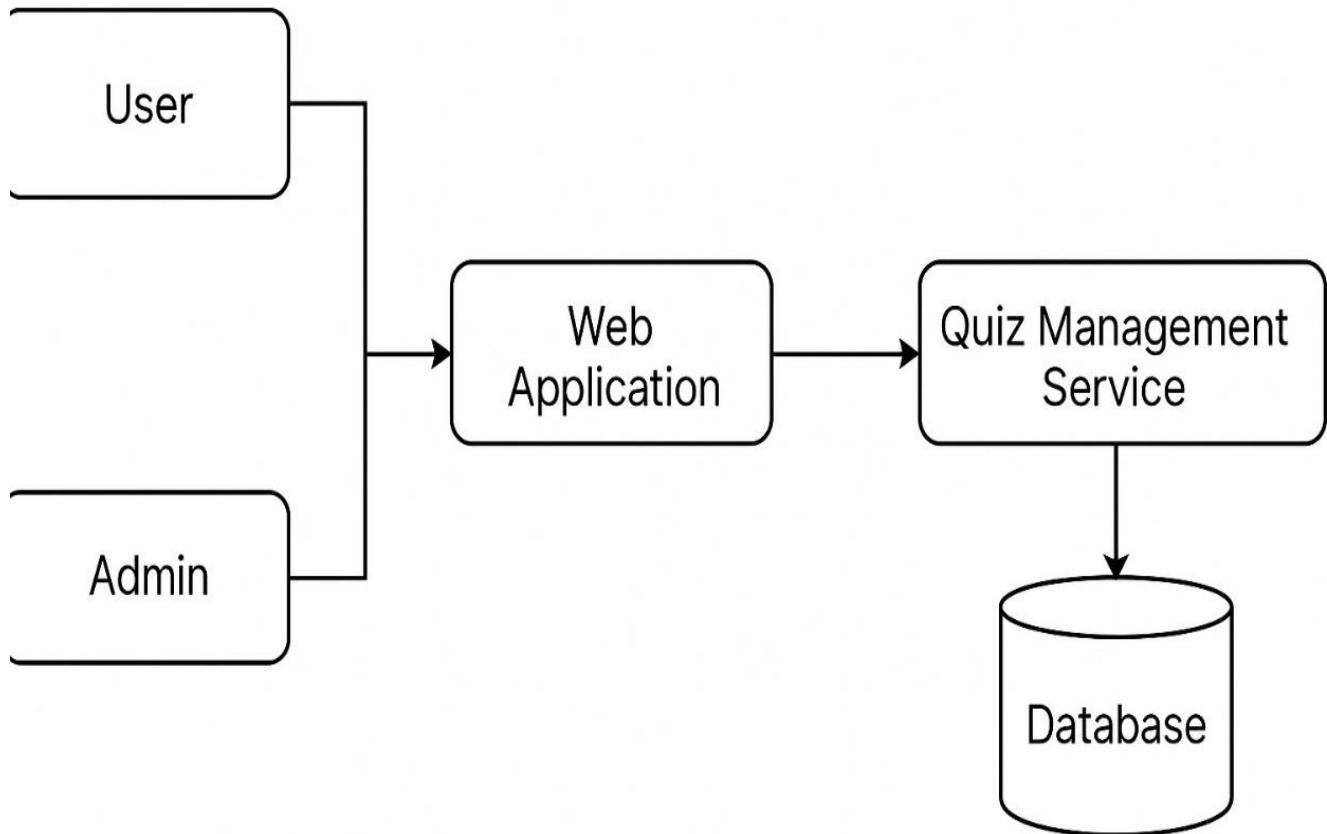
An architectural diagram is a visual representation that maps out the physical implementation for components of a software system. It shows the general structure of the software system and the associations, limitations, and boundaries between each element.



#### Procedure

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

## QUIZ MANAGEMENT SYSTEM



### Result:

The architecture diagram was designed successfully

## EX NO. 10

### USER INTERFACE

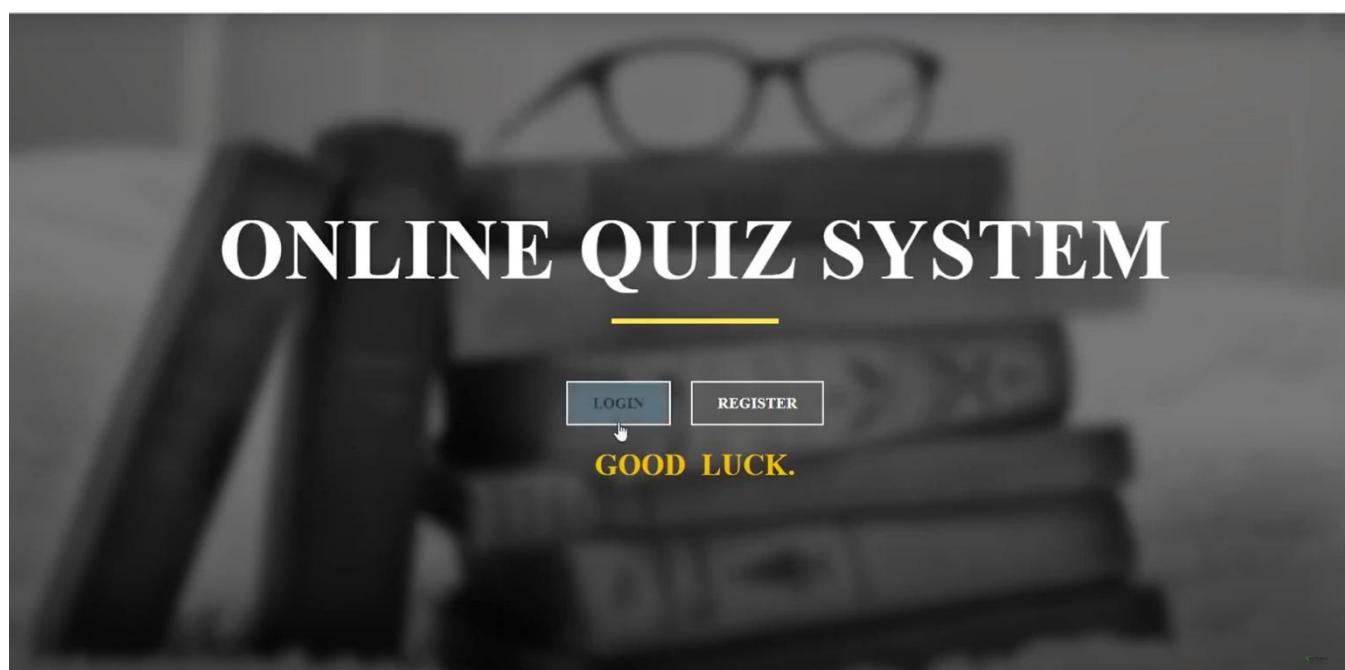
#### Aim:

Design User Interface for the given project

#### User Interface:

User Interface (UI) refers to the visual layout and interactive elements of a software application or website that allow users to interact with the system. It includes components like buttons, menus, input fields, icons, colors, typography, and the overall screen layout.

A well-designed UI ensures that users can easily and efficiently navigate, understand, and use the application to achieve their goals.



| S.N. | Topic         | Total question | Marks | Action                 |
|------|---------------|----------------|-------|------------------------|
| 1    | Ip Networking | 10             | 30    | <button>Start</button> |
| 2    | Php & Mysqli  | 10             | 30    | <button>Start</button> |

Online Quiz System    [Home](#)    [History](#)    [Ranking](#)    [Log out](#)

Question 1 ::

How long is an IPv6 address?

32 bits  
 128 bytes  
 64 bits  
 16 bytes

Online Quiz System    [Home](#)    [History](#)    [Ranking](#)    [Log out](#)

### Result

|                 |    |
|-----------------|----|
| Total Questions | 10 |
| right Answer    | 3  |
| Wrong Answer    | 7  |
| Score           | 2  |
| Overall Score   | 2  |

**Result:**

The UI was designed successfully.

## **EX NO. 11**

## **IMPLEMENTATION**

### **Aim:**

To implement the given project based on Agile Methodology.

### Procedure:

#### Step 1: Set Up an Azure DevOps Project

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

#### Step 2: Add Your Web Application Code

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

```
git clone <repo_url>
cd <repo_folder>
```

- Add web application code (HTML, CSS, JavaScript, React, Angular, or backend like Node.js, .NET, Python, etc.).
- Commit & push:

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

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

- Navigate to Pipelines → Click "New Pipeline".
- Select Git Repository (Azure Repos, GitHub, or Bitbucket).
- Choose Starter Pipeline or a pre-configured template for your framework.

Modify the azure-pipelines.yml file (Example for a Node.js app):

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

```
steps:
- task: UseNode@1
  inputs:
    version: '16.x'
- script: npm install
  displayName: 'Install dependencies'
- script: npm run build
  displayName: 'Build application'
- task: PublishBuildArtifacts@1
  inputs:
    pathToPublish:      'dist'
    artifactName: 'drop'
```

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

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

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

#### **Result**

Thus, the application was successfully implemented.

## **EX NO. 12**

### **TESTING**

#### **a) TESTING-TEST PLANS & TEST CASES**

##### **Aim:**

Test Plans and Test Case and write two test cases for at least five user stories showcasing the happy path and error scenarios in azure DevOps platform.

##### **Test Planning and Test Case**

##### **Test Case Design Procedure**

###### **1. Understand Core Features of the Application**

- User Signup & Login
- Viewing and Managing Playlists
- Fetching Real-time Metadata
- Editing playlists (rename, reorder, record)
- Creating smart audio playlists based on categories (mood, genre, artist, etc.)

###### **2. Define User Interactions**

- Each test case simulates a real user behaviour (e.g., logging in, renaming a playlist, adding a song).

###### **3. Design Happy Path Test Cases**

- Focused on validating that all features function as expected under normal conditions.
- Example: User logs in successfully, adds item to playlist, or creates a category-based playlist.

###### **4. Design Error Path Test Cases**

- Simulate negative or unexpected scenarios to test robustness and error handling.
- Example: Login fails with invalid credentials, save fails when offline, no recommendations found.

###### **5. Break Down Steps and Expected Results**

- Each test case contains step-by-step actions and a corresponding expected outcome.
- Ensures clarity for both testers and automation scripts.

###### **6. Use Clear Naming and IDs**

- Test cases are named clearly (e.g., TC01 – Successful Login, TC10 – Save Playlist Fails).
- Helps in quick identification and linking to user stories or features.

###### **7. Separate Test Suites**

- Grouped test cases based on functionality (e.g., Login, Playlist Editing, Recommendation System).

- Improves organization and test execution flow in Azure DevOps.

## 8. Prioritize and Review

- Critical user actions are marked high-priority.
- Reviewed for completeness and traceability against feature requirements.

### 1. New test plan

New Test Plan

Name \* Authentication

Area Path \* Online Quiz System

Iteration \* Online Quiz System

Create Cancel

### 2. Test suite

| Title  | Outcome | Order | Test Case Id |
|--|---------|-------|--------------|
| Displaying Available Quizzes to Students in Online Quiz System | Passed  | 1     | 58           |
| Account Deletion Flow for User-Requested Data Removal          | Passed  | 2     | 63           |
| Admin Access for Quiz Creation and Student Monitoring          | Passed  | 3     | 64           |
| Email Verification for Account Activation and Security         | Failed  | 4     | 65           |

### 3. Test case

Give two test cases for at least five user stories showcasing the happy path and error scenarios in Azure DevOps platform.

#### Online Quiz System – Test Plans

### USER STORIES

- US201** – As a student, I want to register and log in to the system.
- US202** – As a student, I want to take a quiz and submit answers.
- US203** – As a student, I want to view my quiz results.
- US204** – As an admin, I want to create a new quiz with multiple questions.
- US205** – As an admin, I should not be able to create a quiz without adding any questions.

#### Test Suite: TS01 – User Authentication (ID: 301)

##### Test Suites

###### 1. TC01 – Successful Registration and Login

- **Action:**
  - Navigate to registration page.
  - Enter valid name, email, and password.
  - Submit form.
  - Go to login page and enter credentials.
- **Expected Results:**
  - User account is created.
  - Login successful, user redirected to dashboard.
- **Type:** Happy Path

###### 2. TC02 – Registration with Missing Fields

- **Action:**
  - Navigate to registration page.
  - Leave the email field blank.
  - Submit form.
- **Expected Results:**
  - Validation fails.
  - Error “Email is required” is displayed.
  - Registration is not completed.
- **Type:** Error Path

###### 3. TC03 – Login with Invalid Credentials

- **Action:**
  - Navigate to login page.
  - Enter incorrect email and/or password.
  - Click "Login".
- **Expected Results:**
  - Login fails.
  - Message “Invalid email or password” is displayed.
- **Type:** Error Path

#### **4. TC04 – Password Validation During Registration**

- **Action:**
  - Navigate to registration page.
  - Enter valid email and name.
  - Enter a password less than 6 characters (e.g., "abc").
  - Click "Register".
- **Expected Results:**
  - Registration fails.
  - Error message “Password must be at least 6 characters long” is shown.
- **Type:** Error Path

**Test Suite: TS02 – Quiz Participation (ID: 302)**

#### **1. TC05 – View Quiz Results**

- **Action:**
  - Log in as a student.
  - Navigate to “My Results”.
  - Click on a completed quiz.
- **Expected Results:**
  - Quiz results are shown with score, correct answers, and feedback.
- **Type:** Happy Path

#### **2. TC06 - View Results Without Submitting Quiz**

- **Action:**
  - Attempt to view results for a quiz that hasn't been submitted.
- **Expected Results:**
  - Message “No results available – quiz not attempted” is displayed.
- **Type:** Error Path

#### **3. TC07 – Results Display Accuracy**

- **Action:**
  - Complete a quiz with known correct/incorrect answers.
  - View results.
- **Expected Results:**
  - Scores and feedback match the actual answers submitted.
  - Answer key is displayed accurately.
- **Type:** Happy Path

**Test Suit: TS03 – Upload Notifications (ID: 108)**

#### **1. TC08 – Successful Quiz Attempt**

- **Action:**
  - Log in as a student.
  - Select an available quiz and start it.
  - Answer all questions and click “Submit”.
- **Expected Results:**
  - Quiz is submitted.
  - Confirmation message “Quiz submitted successfully” is shown.
- **Type:** Happy Path

## Test Cases

The screenshot shows the Microsoft Azure DevOps interface for a test case titled "58 Displaying Available Quizzes to Students in Online Quiz System". The test case is assigned to DINESH KUMAR B and is in the "Design" state. It is associated with the "Online Quiz System" area and iteration. There are 0 comments and 1 step.

**Steps:**

1. Login as a student user and navigate to the Available Quizzes page. Expected result: The page loads and displays quizzes assigned to the student or marked as public.
2. Verify the quiz list shows Title, Subject, Duration, and Status. All quizzes display complete details, and statuses are shown (Attempted/Not).
3. Check if past/expired quizzes are visible. Expired quizzes should either not appear or should be clearly marked as expired.
4. Apply sort by date or filter by subject (if feature is available). Quizzes get sorted or filtered correctly as per selected criteria.

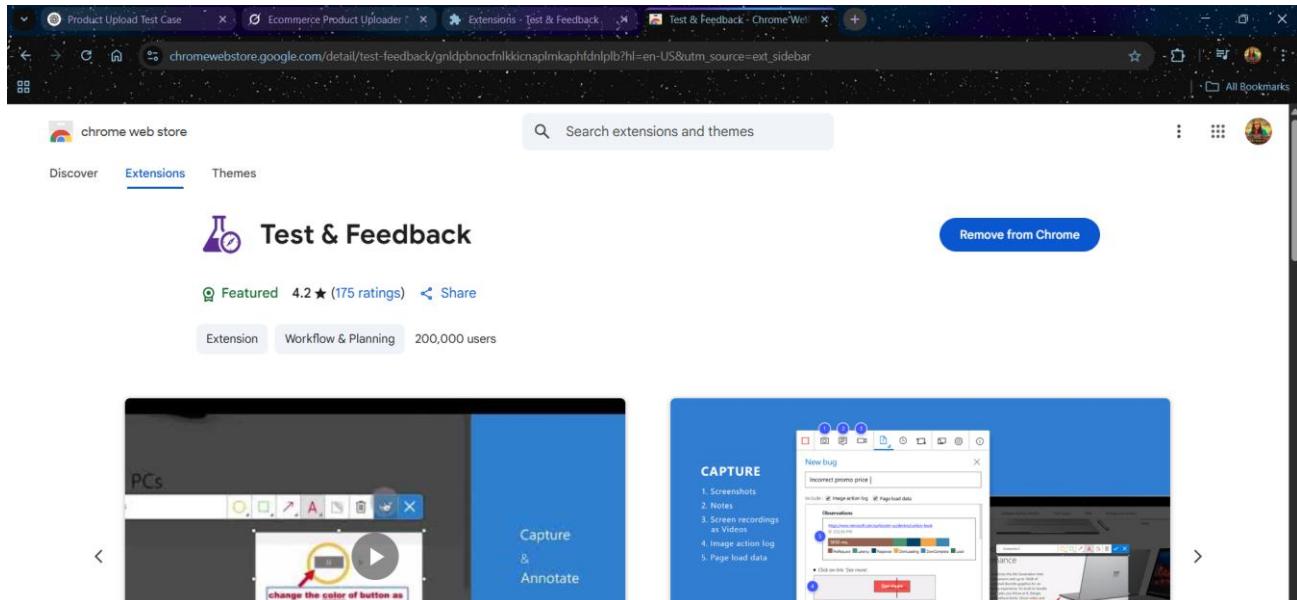
Below the steps, there is a link to "Parameter values". On the right side of the screen, there are sections for "Recent test results", "Deployment", and "Development".

## 4. Installation of test

The screenshot shows the Chrome Web Store page for the "Test & Feedback" extension. The extension has a rating of 4.2 stars from 175 ratings and over 200,000 users. It is categorized under "Extension" and "Workflow & Planning".

The main image on the page shows a screenshot of the extension in use, displaying a UI for capturing and annotating screenshots. Another smaller image shows the extension's interface for creating bugs.

## Test and feedback Showing it as an extension



This screenshot shows a Microsoft Azure DevOps work item for a bug titled 'Quiz List Fails to Filter Expired Quizzes Properly in Student View'. The work item is categorized as 'New' and assigned to 'Unassigned'. It includes sections for 'Repro Steps', 'Planning', 'Deployment', 'Development', and 'Related Work'. The 'Repro Steps' section details three steps: 1. Logging in as a student user and navigating to the Available Quizzes page, where the expected result is that the page loads and displays quizzes assigned to the student or marked as public. 2. Verifying the quiz list shows Title, Subject, Duration, and Status, with the expected result being that all quizzes display complete details and statuses are shown (Attempted/Not). 3. A third step is partially visible. The 'Planning' section shows a resolved reason, story points (2), priority (2), severity (3 - Medium), and activity. The 'Deployment' section provides instructions for tracking releases. The 'Development' section includes a link to Azure Repos. The 'Related Work' section allows adding existing work items as parents. The 'Tested By' section lists a user who tested the issue.

## 5. Running the test cases

The screenshot shows a test management interface. On the left, there's a sidebar titled "Test Suites" with a single entry: "dummy (4)". The main area is titled "dummy (ID: 61)" and has tabs for "Define", "Execute" (which is selected), and "Chart". Under "Test Points (4 items)", there is a table:

| Title  | Outcome | Order | Test Case Id |
|--|---------|-------|--------------|
| Displaying Available Quizzes to Students in Online Quiz System | Passed  | 1     | 58           |
| Account Deletion Flow for User-Requested Data Removal          | Passed  | 2     | 63           |
| Admin Access for Quiz Creation and Student Monitoring          | Passed  | 3     | 64           |
| Email Verification for Account Activation and Security         | Failed  | 4     | 65           |

## 6. Recording the test case

The screenshot shows a browser window with a recording interface. The address bar says "dev.azure.com/dineshkumard980/Online%20Quiz%20System/\_testExecution/Index". The recording status is "Recording: 00:00 / 10:00 minutes".

**58\*: Displaying Available Quizzes to Students in Online Quiz System**

1. Login as a **student user** and navigate to the **Available Quizzes** page.  
EXPECTED RESULT:  
The page loads and displays quizzes assigned to the student or mark.
2. Verify the quiz list shows **Title, Subject, Duration, and Status**.  
EXPECTED RESULT:  
All quizzes display complete details, and statuses are shown (Attempted).
3. Check if **past/expired quizzes** are visible.  
EXPECTED RESULT:  
Expired quizzes should either not appear or should be **clearly marked**.
4. **Apply sort by date or filter by subject** (if feature is available).  
EXPECTED RESULT:  
Quizzes get sorted or filtered correctly as per selected criteria.

A modal dialog titled "Choose what to share with Test & Feedback" is open. It asks "The site will be able to see the contents of your screen". It offers three options: "Chrome Tab", "Window", and "Entire Screen". "Entire Screen" is selected. At the bottom of the dialog are "Share" and "Cancel" buttons.

## 7.Creating the bug

The screenshot shows a Microsoft Azure DevOps bug tracking interface. A new bug has been created with the following details:

- Title:** BUG 62 Quiz List Fails to Filter Expired Quizzes Properly in Student View
- Status:** New
- Area:** Online Quiz System
- Iteration:** Online Quiz System
- Activity:** 3 - Medium
- Effort (Hours):** Original Estimate, Remaining, Completed
- Test Configuration:** Windows 10
- System Info:** Detailed table of system configuration including browser, operating system, processor, memory, and display.
- Discussion:** Placeholder for comments.

The bug description includes two test cases:

- Expected Result:** Expired quizzes should either not appear or should be clearly marked as expired.
- Actual Result:** Apply sort by date or filter by subject (if feature is available).

Comments section:

- A comment from DINESH KUMAR was added 2 hours ago, linking to a GitHub commit.
- Related work section: "Displaying Available Quizzes to Students in Online Quiz System" is listed as a task.

## 8.Test case results

The screenshot shows a test management interface for the 'dummy' test suite (ID: 61). The results are as follows:

- Test Suites:** dummy (4 items)
- Test Points (4 items):**

| Title  | Outcome | Order | Test Case Id |
|--|---------|-------|--------------|
| Displaying Available Quizzes to Students in Online Quiz System | Passed  | 1     | 58           |
| Account Deletion Flow for User-Requested Data Removal          | Passed  | 2     | 63           |
| Admin Access for Quiz Creation and Student Monitoring          | Passed  | 3     | 64           |
| Email Verification for Account Activation and Security         | Failed  | 4     | 65           |

## 9. Test report summary

BUG 62

62 Quiz List Fails to Filter Expired Quizzes Properly in Student View

No one selected 0 Comments Add Tag

Save and Close Follow Details Updated by DINESH KUMAR B: 2h ago

State: New Area: Online Quiz System Iteration: Online Quiz System

Reason: New

Repro Steps

5/14/2025 4:58 PM Bug filed on "Displaying Available Quizzes to Students in Online Quiz System"

**Step no.** **Result** **Title**

1. Passed Login as a student user and navigate to the Available Quizzes page.  
Expected Result  
The page loads and displays quizzes assigned to the student or marked as public.

2. Passed Verify the quiz list shows Title, Subject, Duration, and Status.  
Expected Result  
All quizzes display complete details, and statuses are shown (Attempted/Not).

3. Passed Check if past/expired quizzes are visible.  
Expected Result  
Expired quizzes should either not appear or should be clearly marked as expired.

Planning

Resolved Reason: Story Points  
Priority: 2  
Severity: 3 - Medium  
Activity:

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.

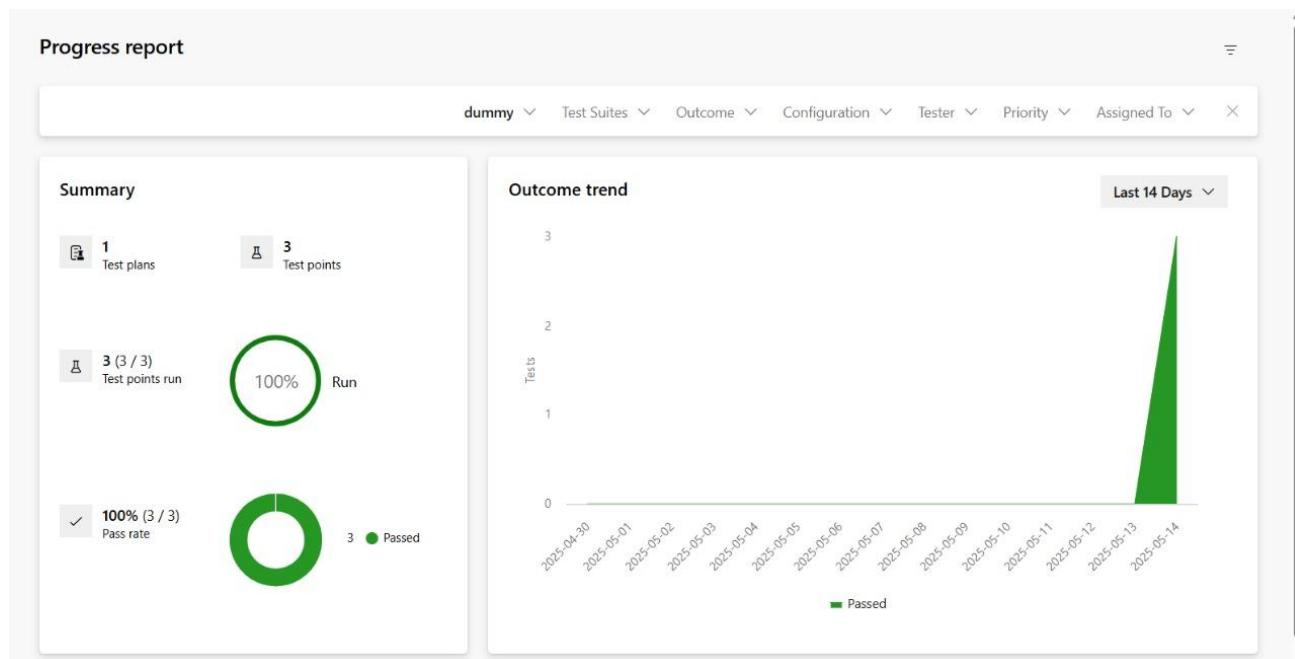
Effort (Hours)

Original Estimate: Remaining: Completed:

Related Work

Add link: Add an existing work item as a parent

## 10. Progress report



Azure DevOps dineshkumardd980 / Settings / Process

System processes cannot be customized. To add customization [create an inherited process](#).

### All processes > Agile

Work item types Backlog levels Projects

Help Filter by work item type n...

| Name       | Description   |
|------------|---|
| Bug        | Describes a divergence between required and actual behavior, and tracks the work done to correct the defect and verify the correct... |
| Epic       | Epics help teams effectively manage and groom their product backlog   |
| Feature    | Tracks a feature that will be released with the product   |
| Issue      | Tracks an obstacle to progress.   |
| Task       | Tracks work that needs to be done.  |
| Test Case  | Server-side data for a set of steps to be tested.   |
| Test Plan  | Tracks test activities for a specific milestone or release.   |
| Test Suite | Tracks test activities for a specific feature, requirement, or user story.  |
| User Story | Tracks an activity the user will be able to perform with the product  |

[https://dev.azure.com/dineshkumardd980/\\_settings/process?process-name=Agile&a=workitemtypes#process-name=Agile&a=workitemtypes](https://dev.azure.com/dineshkumardd980/_settings/process?process-name=Agile&a=workitemtypes#process-name=Agile&a=workitemtypes)

## 11.Changing the test template

Azure DevOps dineshkumardd980 / Settings / Process

### All processes

Processes Fields Help Filter by process name

| Name            | Description  | Team projects |
|-----------------|--|---------------|
| Basic (default) | This template is flexible for any process and great for teams getting started with Az... | 1             |
| Agile           | This template is flexible and will work great for most teams using Agile planning me...  | 1             |
| Scrum           | This template is for teams who follow the Scrum framework.                               | 0             |
| CMMI            | This template is for more formal projects requiring a framework for process improv...    | 0             |

Process

## 12. View the new test case template

The screenshot shows the 'Add a field to Test Case' dialog box over a background of the Azure DevOps settings interface. The dialog has tabs for 'Definition', 'Options', and 'Layout'. Under 'Definition', the 'Create a field' option is selected, showing fields for Name (empty), Type (Text (single line)), and Description (Optional provide a description for the field). The background shows the 'Steps' section of the process configuration.

The screenshot shows the 'Test Case' configuration page under the 'Gokul312005 Agile' process. It displays sections for 'Steps' (Text (multiple lines)), 'Recent test results' (Recent test case results), 'Deployment' (Deployments), 'Development' (Links), 'Related Work' (Links), and 'Status' (Priority: Integer, Automation status: Text (single line)). The 'Steps' section is currently active.

The screenshot shows the 'Organization Settings' page in Azure DevOps. On the left, there's a sidebar with sections like General, Security, and Boards. The main area is titled 'All processes > Agile' and shows a table with one row for 'Online Quiz System'. The table has columns for 'Name' and 'Description'. The 'Name' column contains 'Online Quiz System' and the 'Description' column contains '... Online Quiz System The Online Quiz System is a web-based platform built using PHP and MySQL to simplify quiz creation, participa...'. At the top of the main area, a blue bar says 'System processes cannot be customized. To add customization create an inherited process.'

| Name               | Description   |
|--------------------|---|
| Online Quiz System | ... Online Quiz System The Online Quiz System is a web-based platform built using PHP and MySQL to simplify quiz creation, participa... |

## Result:

The test plans and test cases for the user stories is created in Azure DevOps with Happy Path and Error Path

## b) Load Testing and Performance Testing

### Aim:

To create an Azure Load Testing resource and run a load test to evaluate the performance of a target endpoint.

### Load Testing

#### Steps to Create an Azure Load Testing Resource:

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

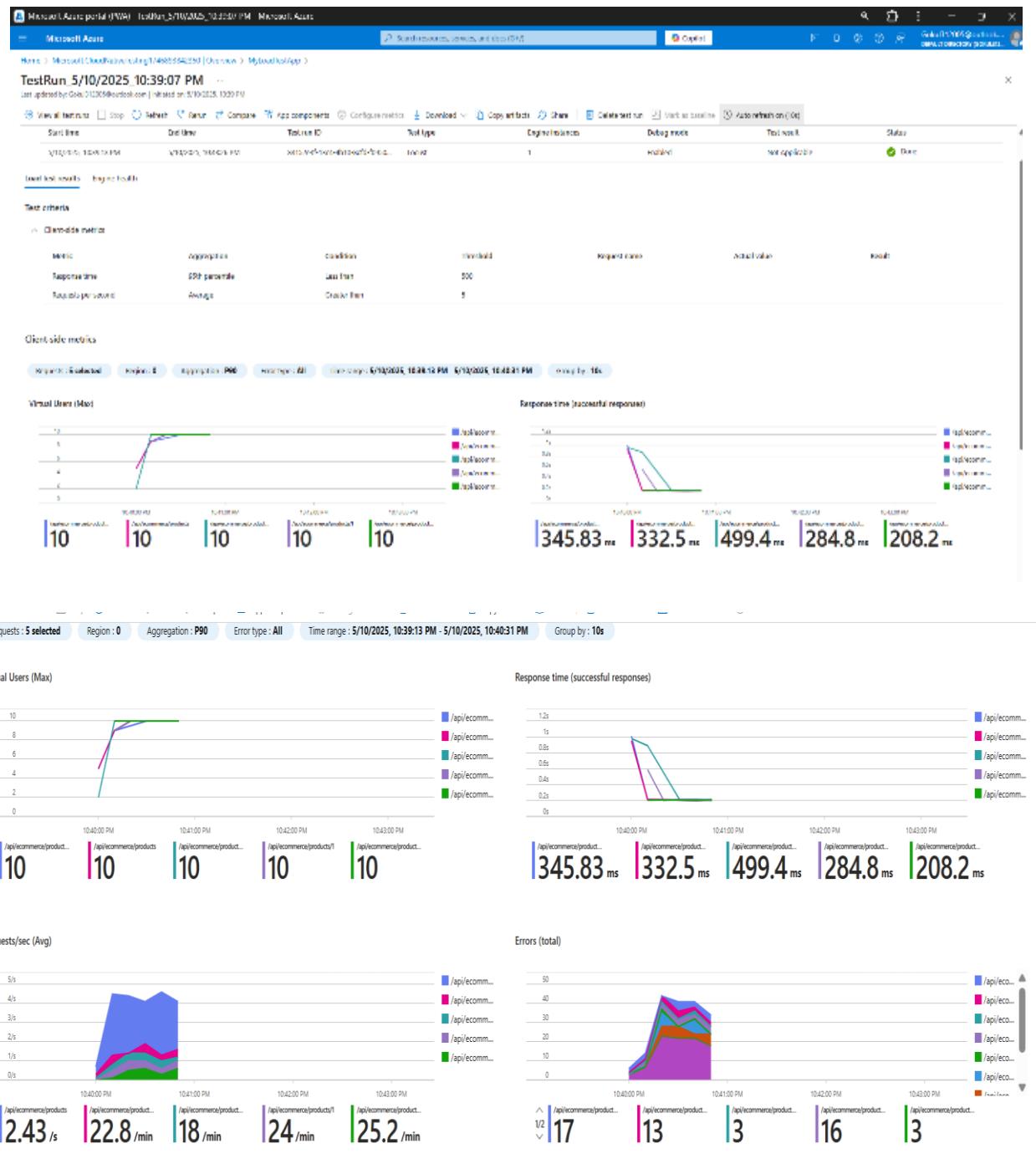
1. Sign in to Azure Portal  
Go to <https://portal.azure.com> and log in.
2. Create the Resource
  - o Go to *Create a resource* → Search for “Azure Load Testing”.
  - o Select Azure Load Testing and click Create.
3. Fill in the Configuration Details
  - o *Subscription*: Choose your Azure subscription.
  - o *Resource Group*: Create new or select an existing one.
  - o *Name*: Provide a unique name (no special characters).
  - o *Location*: Choose the region for hosting the resource.
4. (Optional) Configure tags for categorization and billing.
5. Click Review + Create, then Create.
6. Once deployment is complete, click Go to resource.

#### Steps to Create and Run a Load Test:

Once your resource is ready:

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

## Load Testing



The screenshot shows the homepage of an Online Quiz System. At the top, there is a navigation bar with links for Home, History, Ranking, and Log out. Below the navigation bar, there is a large green area containing a table with two rows of quiz information.

| S.N. | Topic         | Total question | Marks | Action                |
|------|---------------|----------------|-------|-----------------------|
| 1    | Ip Networking | 10             | 30    | <a href="#">Start</a> |
| 2    | Php & Mysqli  | 10             | 30    | <a href="#">Start</a> |

### Result:

Successfully created the Azure Load Testing resource and executed a load test to assess the performance of the specified endpoint.

