

## INDEX

S.NO	DATE	TITLE
1.	21/1/25	Azure DevOps Environment Setup
2.	21/1/25	Azure DevOps Project Setup and User Story Management
3.	28/1/25	Setting Up Epics, Features and User Stories for Project Planning
4.	11/2/25	Sprint Planning
5.	18/2/25	Poker Estimation
6.	25/2/25	Designing Class and Sequence Diagrams for Project Architecture
7.	04/3/25	Designing Use-Case and Activity Diagrams for Project Architecture
8.	25/3/25	Testing – Test Plans and Test Cases
9.	15/4/25	CI/CD Pipelines in Azure
10.	22/4/25	GitHub: Project Structure & Naming Convention

**EXP NO: 1**

## **AZURE DEVOPS ENVIRONMENT SETUP**

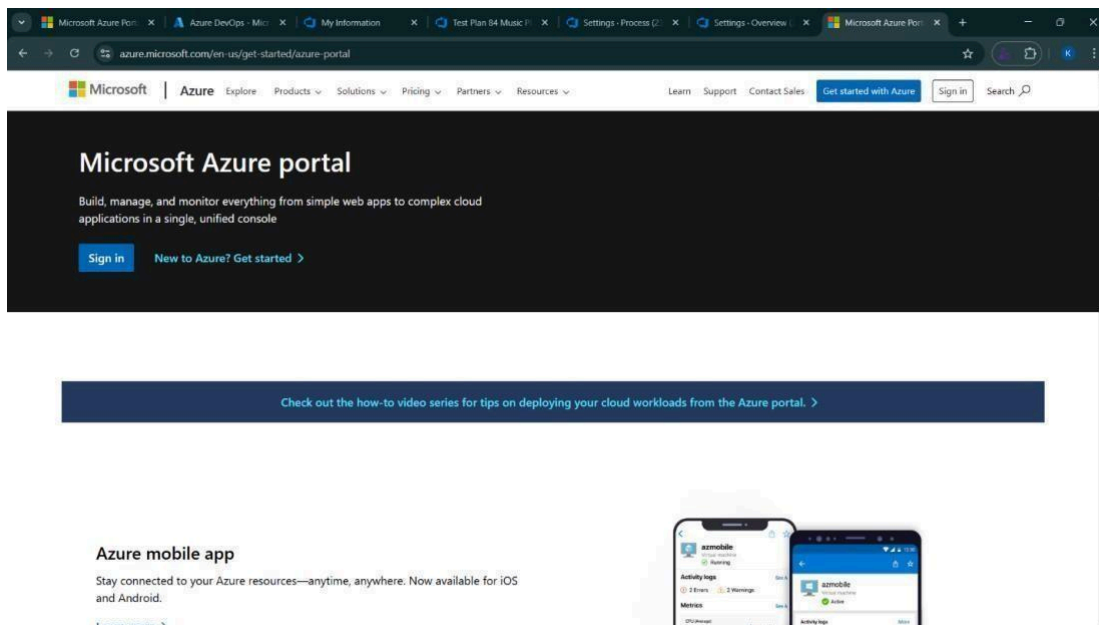
### **AIM**

To set up and access the Azure DevOps environment by creating an organization through the Azure portal.

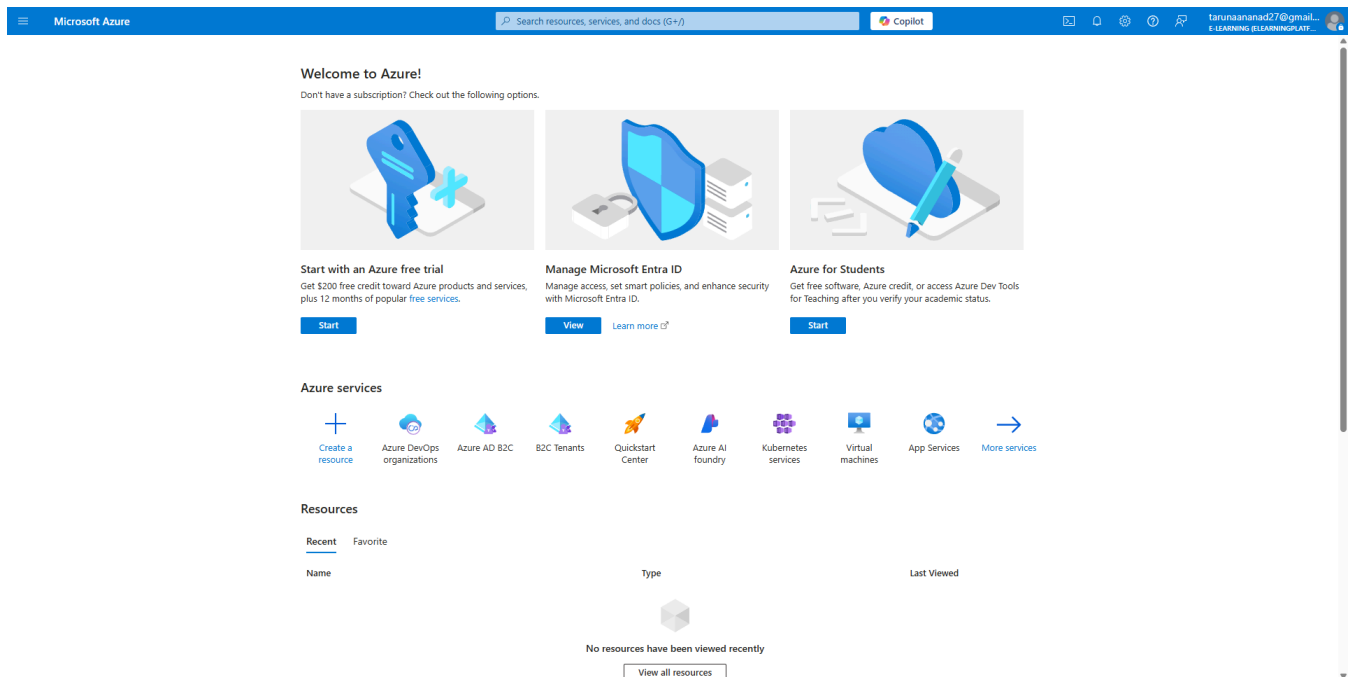
### **INSTALLATION**

1. Open your web browser and go to the Azure website: <https://azure.microsoft.com/enus/getstarted/azureportal>. Sign in using your Microsoft account credentials.

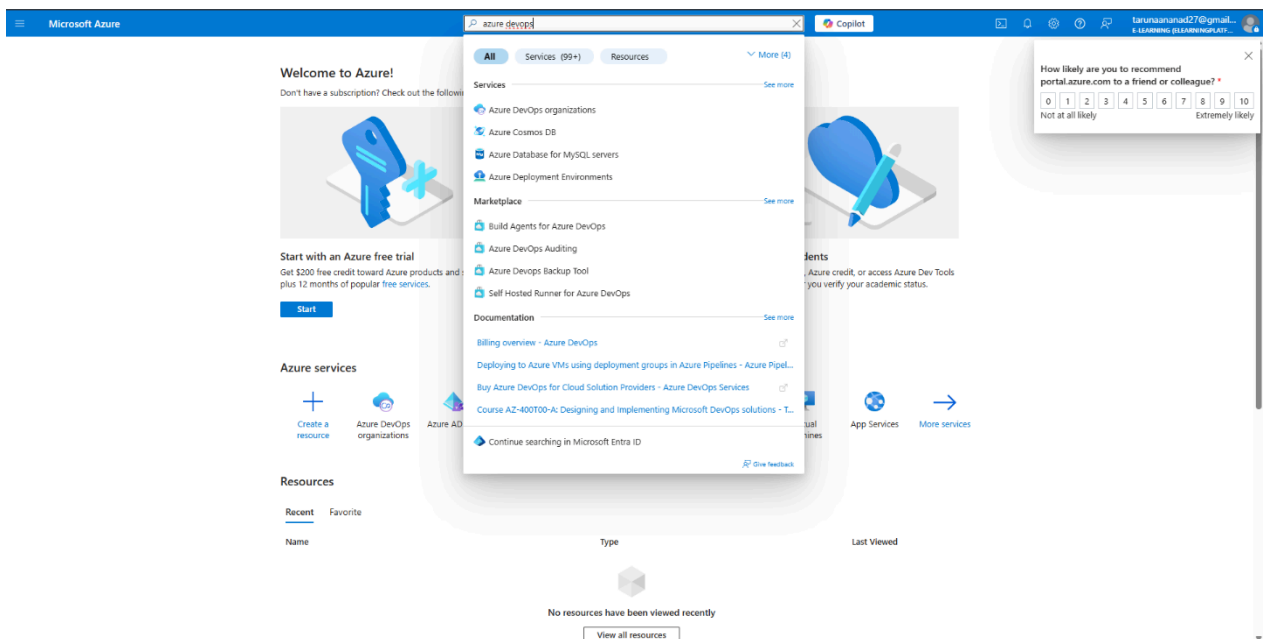
If you don't have a Microsoft account, you can create one here: <https://signup.live.com/?lic=1>



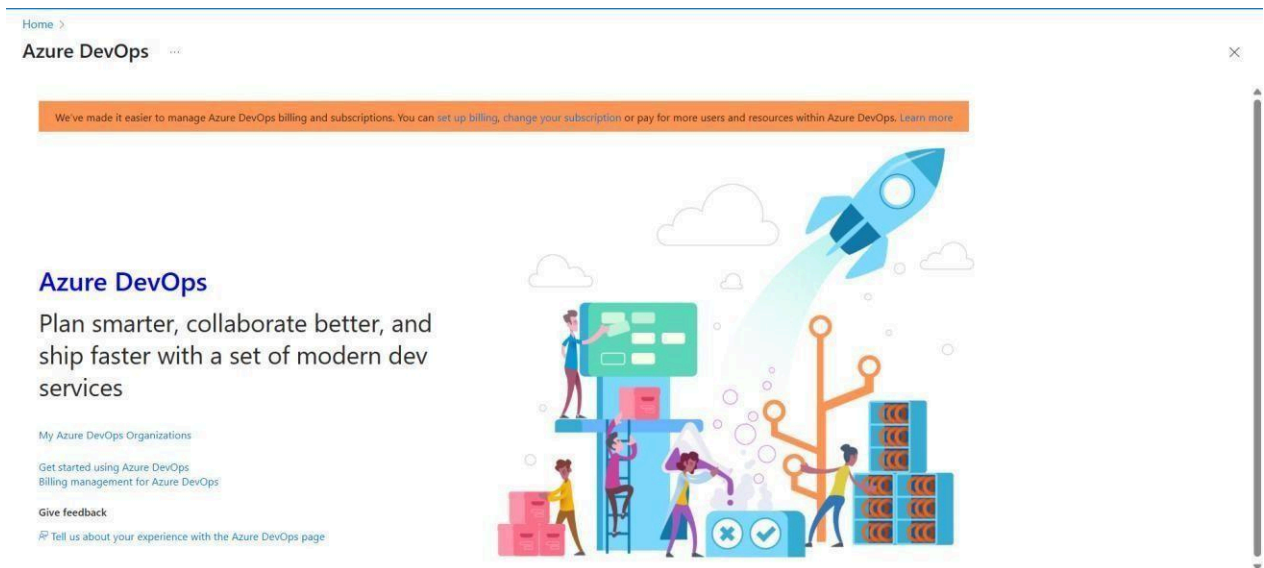
## 2. Azure home page



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



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



## RESULT

Successfully accessed the Azure DevOps environment and created a new organization through the Azure portal.

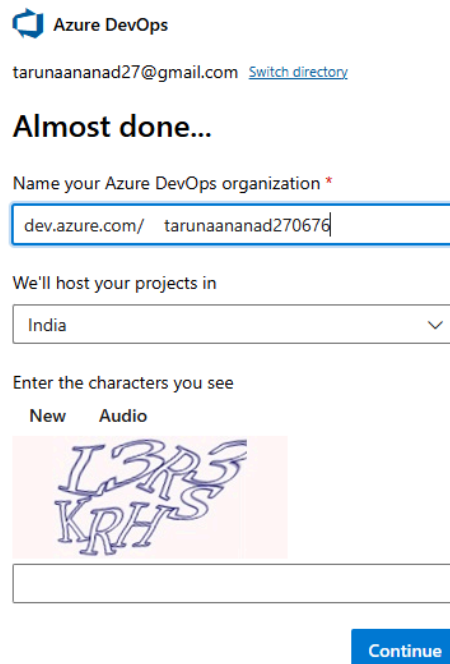
EXP NO: 2

## AZURE DEVOPS PROJECT SETUP AND USER STORY MANAGEMENT

### AIM

To set up an Azure DevOps project for efficient collaboration and agile work management.

### 1. Create An Azure Account



### 2. 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.
- On the organization's **Home page**, click on the **New Project** button.

c. 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).

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


## Create new project

Project name \*

OQS

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 ?

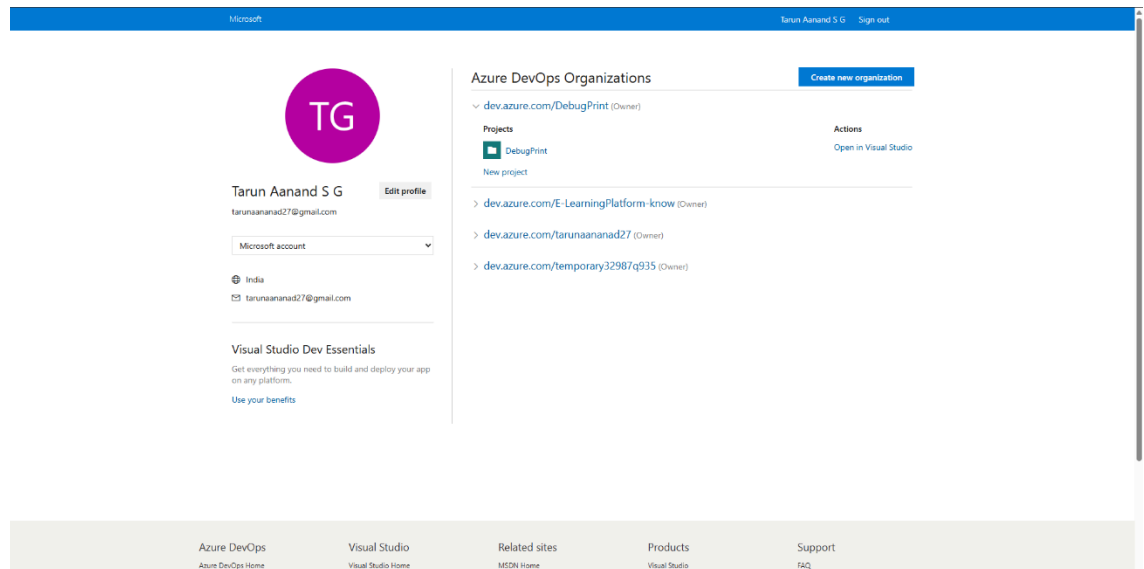
Git

Work item process ?

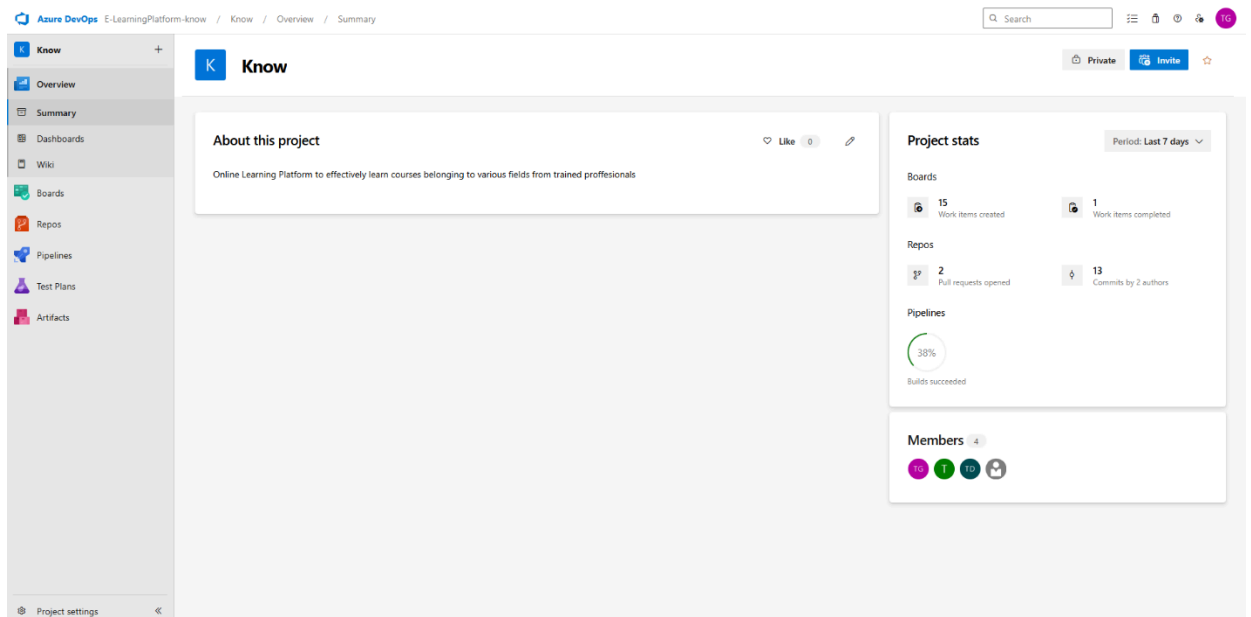
Basic

Cancel>Create

3. Once logged in, ensure you are in the correct organization. If you're part of multiple organizations, you can switch between them from the top left corner (next to your user profile). Click on the Organization name, and you should be taken to the Azure DevOps Organization Home page.



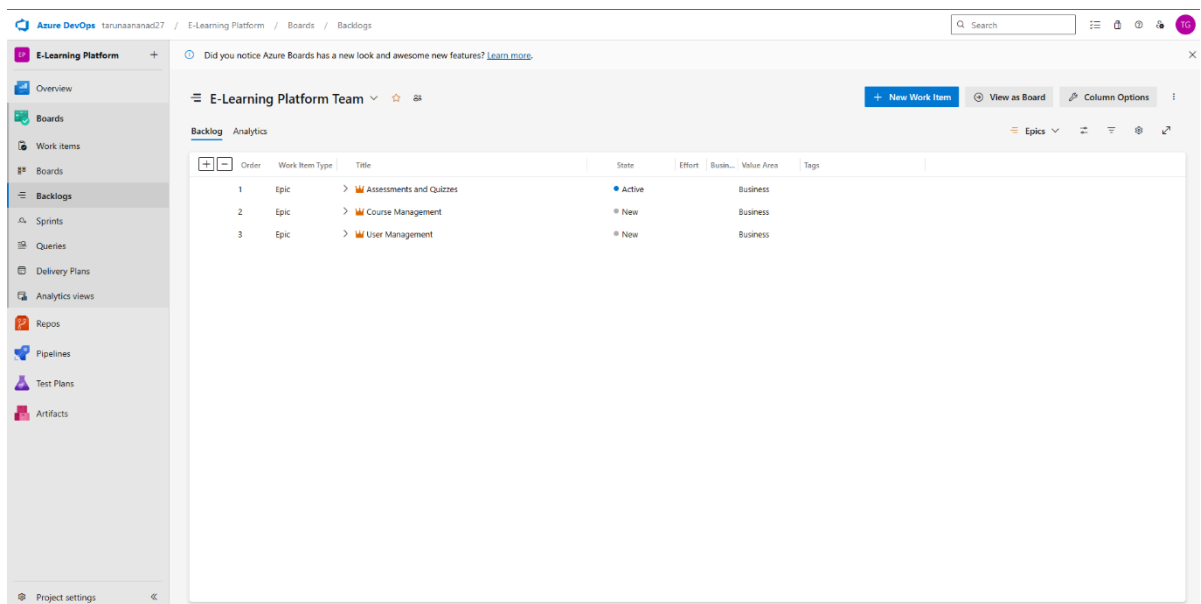
#### 4. Project dashboard



5.To manage user stories:

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.





## **RESULT**

Successfully created an Azure DevOps project with user story management and agile workflow setup.

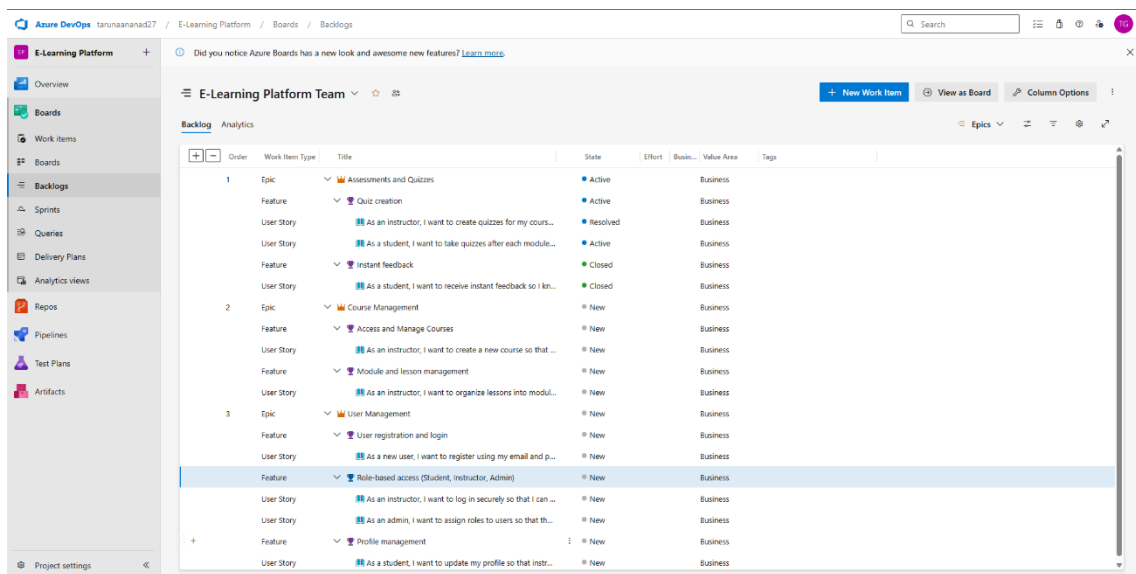
EXP NO:3

# SETTING UP EPICS, FEATURES, AND USER STORIES FOR PROJECT PLANNING

## AIM

To learn about how to create epics, user story, features, backlogs for your assigned project.

## Create Epic, Features, User Stories, Task



## 1.Fill in Epics

Did you notice Azure Boards has a new look and awesome new features? [Learn more.](#)

Work Items | Back to Work Items

NEW EPIC \* Field 'Title' cannot be empty.

Enter title

No one selected | 0 Comments | Add Tag

Save

State: New | Reason: New | Area: E-Learning Platform | Iteration: E-Learning Platform\Sprint 1

Description: Click to add Description.

Discussion: Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request. [switch to Markdown editor](#)

Planning: Priority: 2, Risk, Effort, Business Value, Time Criticality, Start Date, Target Date, Classification: 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. Add an existing work item as a parent

## 2.Fill in Features

FEATURE 2

2 User Registration & Login

No one selected | 0 Comments | Add Tag

Save and Close | Follow | 3 | 0

Updated by Ranjani Sai: Apr 25

State: New | Reason: New | Area: Online Quiz System | Iteration: Online Quiz System\sprint 1

Description: Click to add Description.

Discussion: Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request. [switch to Markdown editor](#)

Planning: Priority: 2, Risk, Effort, Business Value, Time Criticality, Start Date, Target Date, Classification: 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. Add an existing work item as a parent

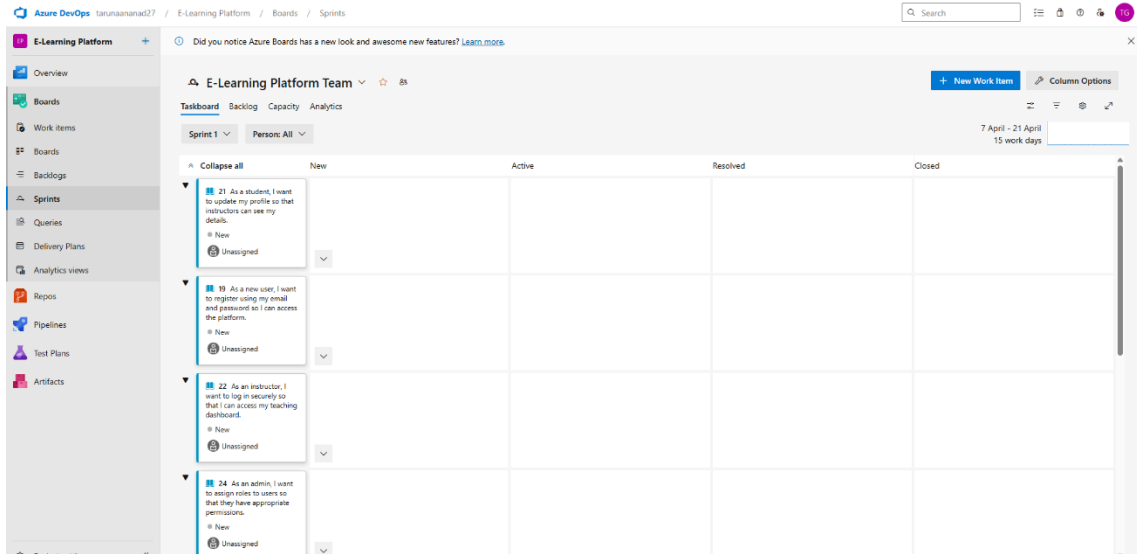
## 3.Fill in User Story Details



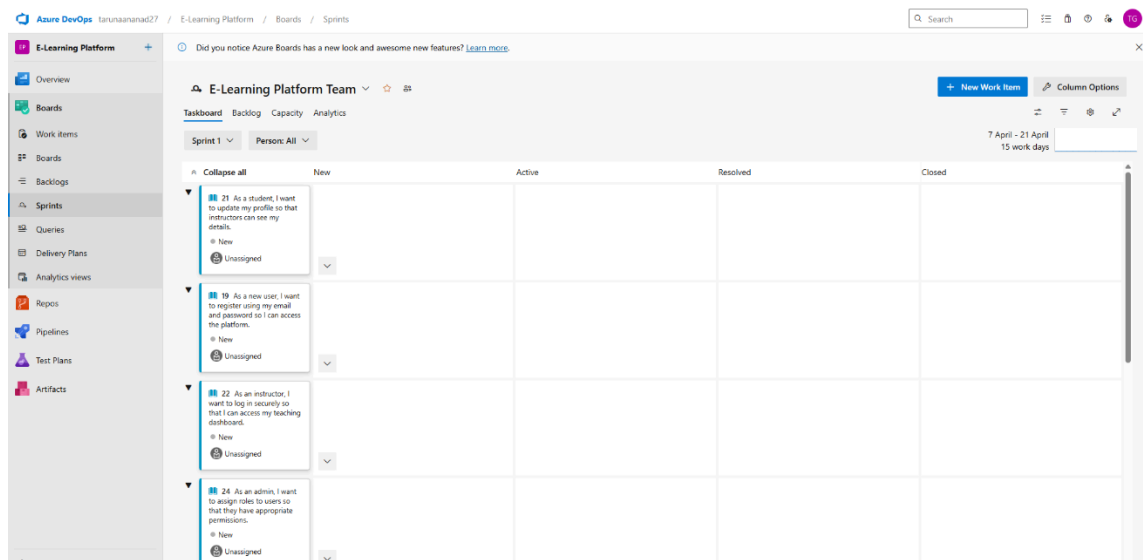
## AIM

To assign user story to specific sprint for the E-Learning Platform.

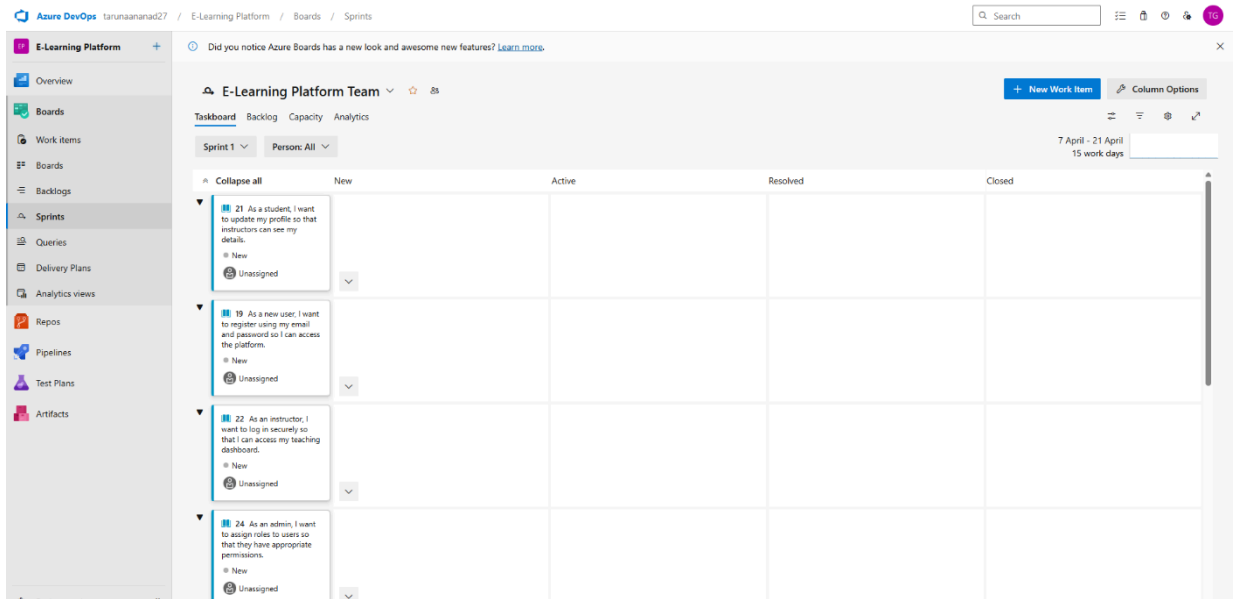
## Sprint Planning Sprint 1



## Sprint 2



## Sprint 3



## RESULT

The Sprints have been created for E-Learning Platform.

EXP NO:5

# POKER ESTIMATION

## AIM

Create Poker Estimation for the user stories – E-Learning Platform.

## Poker Estimation

USER STORY 13

13 As an admin, I want to manage a question bank so I can reuse questions easily.

No one selected0 CommentsAdd Tag

Save and CloseFollow

Updated by Ranjani Sai: Apr 25

Statg: NewArea: Online Quiz SystemReason: NewIteration: Online Quiz System\sprint 1

Details10

Description

Click to add Description.

Acceptance Criteria

Click to add Acceptance Criteria.

Discussion

Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request.

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

## **RESULT**

The Estimation/Story Points is created for the project using Poker Estimation.



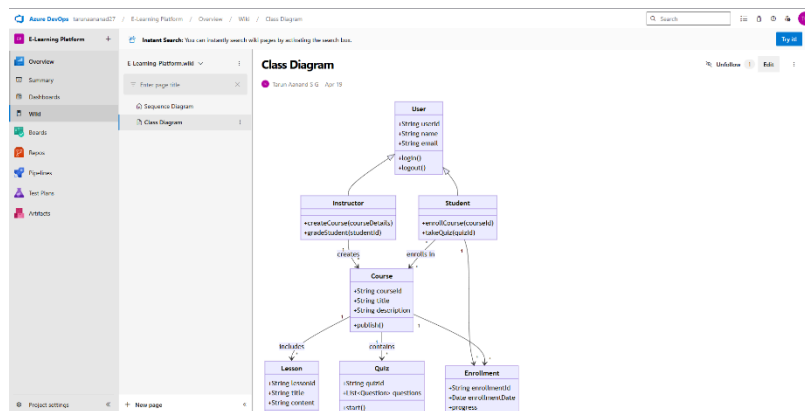
EXP NO: 6

# DESIGNING CLASS AND SEQUENCE DIAGRAMS FOR PROJECT ARCHITECTURE

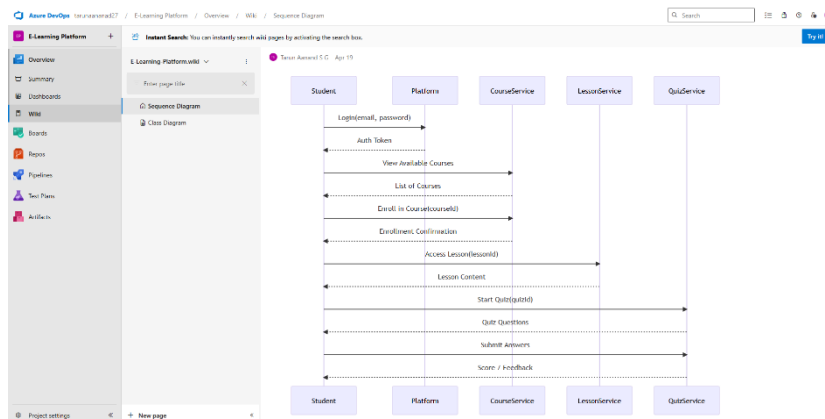
## AIM

To Design a Class Diagram and Sequence Diagram for the given Project.

### 6A. Class Diagram



### 6B. Sequence Diagram



## RESULT

The Class Diagram and Sequence Diagram is designed Successfully for the E-Learning Platform.

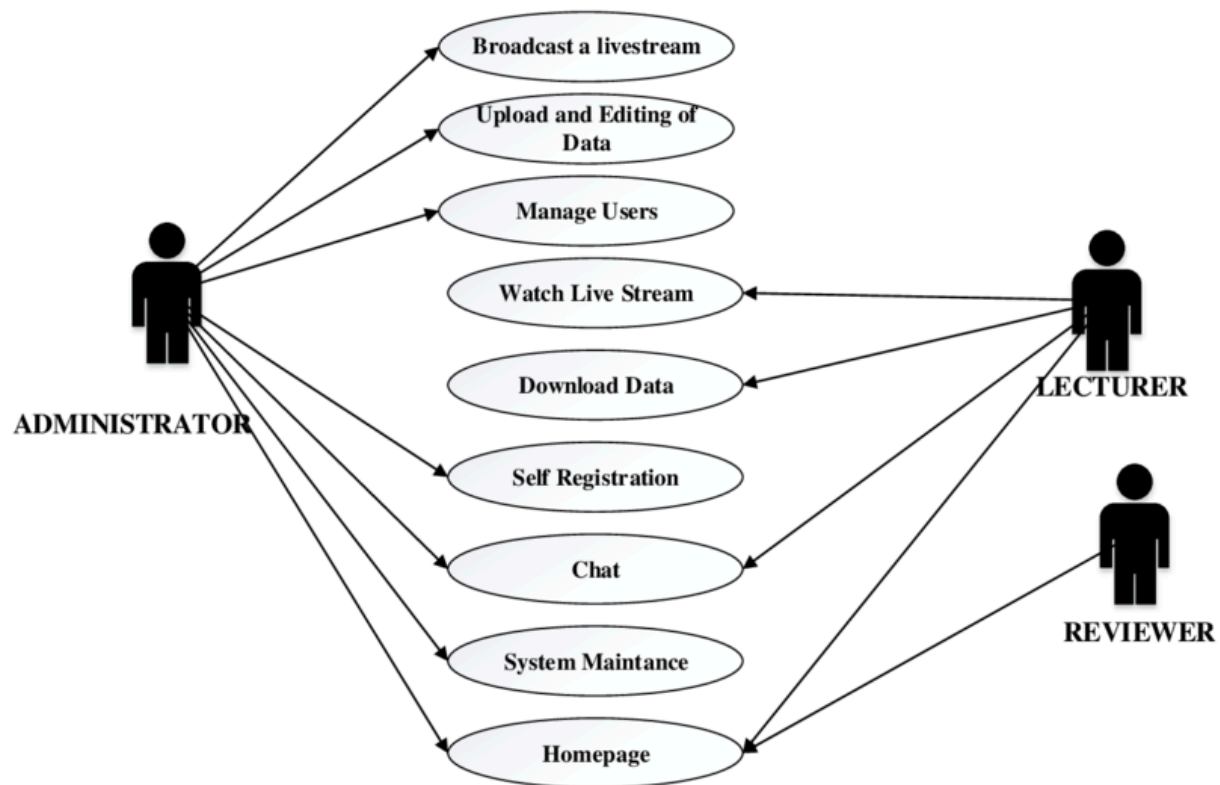
EXP NO: 7

## DESIGNING USE-CASE AND ACTIVITY DIAGRAMS FOR PROJECT STRUCTURE

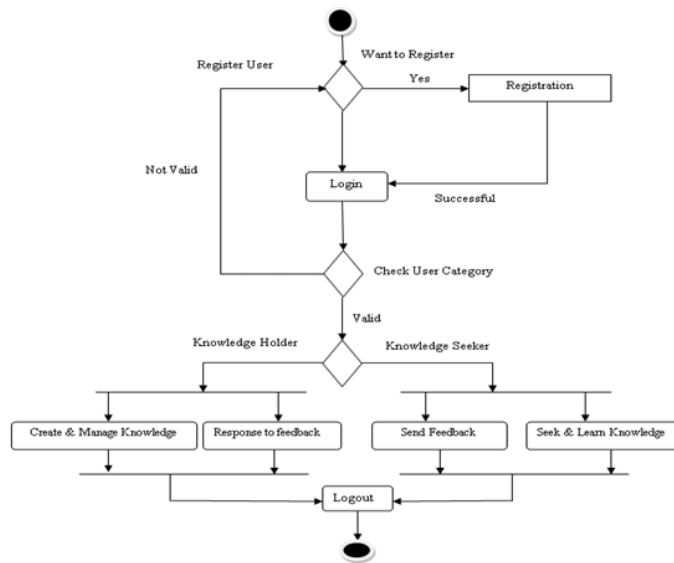
### AIM

To Design an Use-Case Diagram and Activity Diagram for the given Project. 7A.

### Use-Case Diagram



### 7B. Activity Diagram



## RESULT

The Use-Case Diagram and Activity Diagram is designed Successfully for the E-Learning Platform.

**EXP NO: 8**

**TESTING TEST PLANS AND 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 Login

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

- Each test case simulates a real user behaviour (e.g., logging in, submitting quizzes, viewing results)

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

- Focused on validating that all core functionalities work correctly under normal conditions
- Example: Player registers and logs in, submits quizzes and views results

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

- Simulate invalid inputs, system issues or failed actions to ensure proper error handling.
- Example: Login with invalid credentials, submission without attachments, unauthorized access to admin panel.

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

- Each test case includes a clear sequence of actions and expected results. · Ensures both manual testers and automation tools can follow the process easily.

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

- Test cases are uniquely identifies (e.g., TC01 – Valid Login, TC03 – Invalid Password).
- Facilities easy mapping to features and tracking in Azure DevOps.

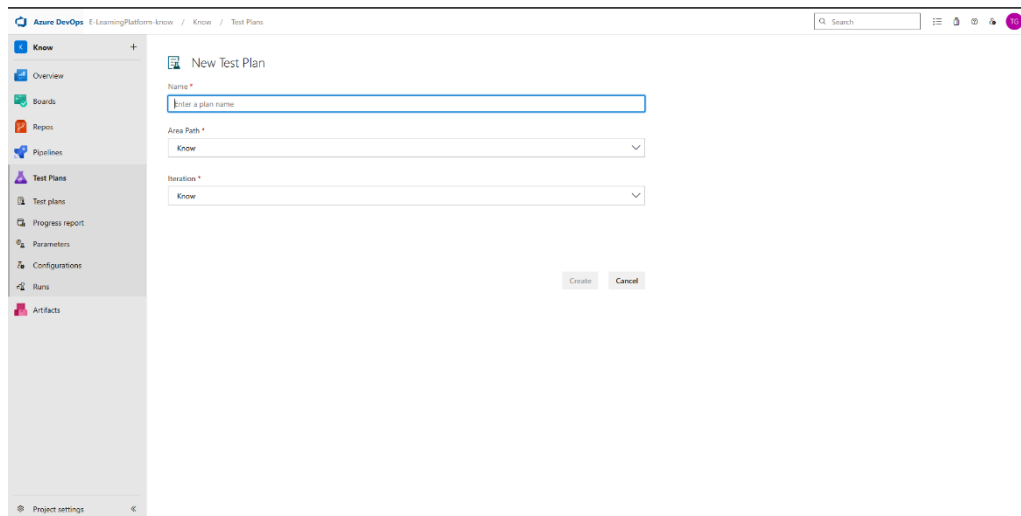
## 7. Separate Test

- Grouped by functionality such as:
  - Login and Registration
  - Quizzes Submission
  - Viewing Results
  - Admin Functions
- Improves organization and enables focused execution in Azure DevOps.

## 8. Prioritize and Review

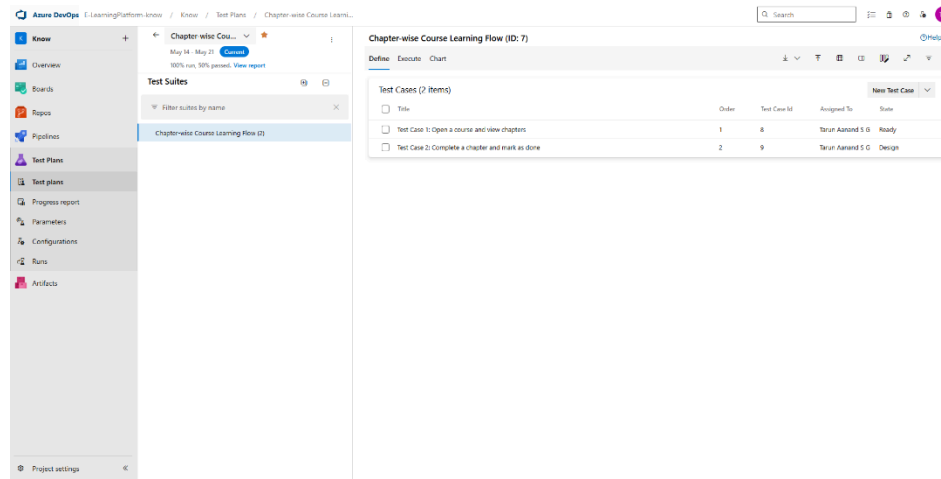
- High-priority assigned to critical workflows like login, quizzes and results.
- Reviewed for completeness, accuracy and alignment with user stories and features definition.

### 1.New test plan



The screenshot shows the 'New Test Plan' interface in Azure DevOps. The left sidebar contains a navigation menu with options: Know, Overview, Boards, Repos, Pipelines, Test Plans (selected), Test plans, Progress report, Parameters, Configurations, Runs, and Artifacts. The main area is titled 'New Test Plan' and contains three input fields: 'Name \*' with a placeholder 'Enter a plan name', 'Area Path \*' with a dropdown menu showing 'Know', and 'Iteration \*' with a dropdown menu showing 'Know'. At the bottom right of the form are 'Create' and 'Cancel' buttons. The top of the interface shows the Azure DevOps logo and the breadcrumb 'E-LearningPlatform-know / Know / Test Plans'.

## 2. Test suite



## 3. Test case

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

E-Learning Platform– Test Plans

### USER STORIES

- As an admin, I want to log in to manage quizzes and users (ID: 3).
- As a player, I want to register and log in so that I can access quizzes securely (ID: 4).
- As an admin, I want to give users the right access so they can use the system properly (ID: 6).
- As a player, I want to see only my quizzes and progress so that it's easy to use (ID: 7).
- As an admin, I want to create and configure quizzes with time limits so I can control quiz flow (ID: 10).

### TEST SUITES

#### Test Suite 1: User Registration and Login

##### Test Case 1.1: User Registration with Valid Details

- **Action:**
  1. Navigate to the registration page
  2. Enter valid name, email, password
  3. Click "Register"

- **Expected Result:**  
User account is created, and a confirmation message is shown.
  - **Type:** Happy Path
- 

#### **Test Case 1.2: Login with Valid Credentials**

- **Action:**
    1. Go to login page
    2. Enter registered email and correct password
    3. Click "Login"
  - **Expected Result:**  
User is redirected to dashboard
  - **Type:** Happy Path
- 

#### **Test Case 1.3: Logout from Application**

- **Action:**
    1. Click on user profile
    2. Click "Logout"
  - **Expected Result:**  
User is logged out and redirected to login page
  - **Type:** Happy Path
- 

### **Test Suite 2: Course Enrollment and Access**

#### **Test Case 2.1: View Available Courses**

- **Action:**
    1. Login to the platform
    2. Navigate to "Courses" tab
  - **Expected Result:**  
A list of available courses is displayed
  - **Type:** Happy Path
- 

#### **Test Case 2.2: Enroll in a Free Course**

- **Action:**

1. From course list, select a free course
  2. Click “Enroll”
- **Expected Result:**  
User is successfully enrolled in the course
  - **Type:** Happy Path
- 

#### **Test Case 2.3: Access Enrolled Course Content**

- **Action:**
    1. Go to “My Courses”
    2. Click on enrolled course
    3. Play a video lecture
  - **Expected Result:**  
Course content loads and plays correctly
  - **Type:** Happy Path
- 

#### **Test Suite 3: Assessment and Certification**

##### **Test Case 3.1: Start Quiz for a Completed Module**

- **Action:**
    1. Complete a module
    2. Click “Take Quiz”
  - **Expected Result:**  
Quiz page opens with questions related to the module
  - **Type:** Happy Path
- 

##### **Test Case 3.2: Submit Quiz Answers**

- **Action:**
    1. Select answers for all questions
    2. Click “Submit”
  - **Expected Result:**  
Submission is accepted and score is displayed
  - **Type:** Happy Path
- 

##### **Test Case 3.3: Download Certificate After Course Completion**

- **Action:**
  1. Complete all modules and pass final quiz



## 2. Click “Download Certificate”

- **Expected Result:**  
Certificate is downloaded in PDF format
- **Type:** Happy Path

## Test Cases

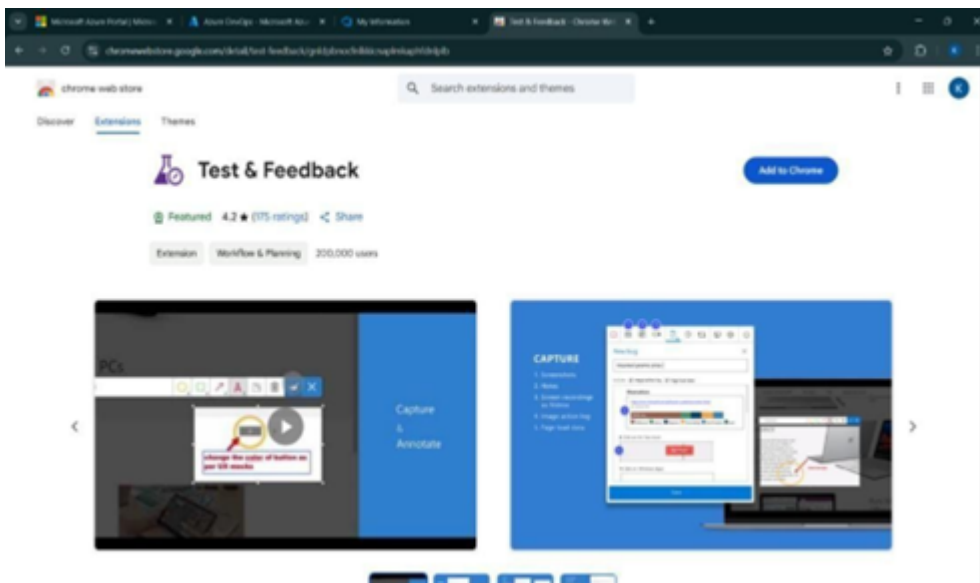
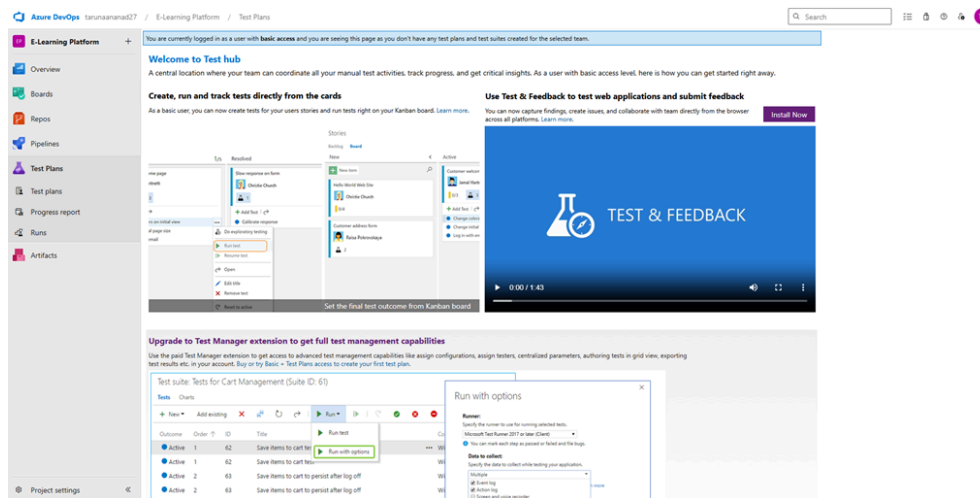
The screenshot shows the Azure DevOps Test Plans interface. The left sidebar contains navigation options: Know, Overview, Boards, Repos, Pipelines, Test Plans (selected), Test plans, Progress report, Parameters, Configurations, Runs, and Artifacts. The main area displays a table of test plans under the 'All' tab. The table has columns for Title, Test Plan ID, State, Area Path, Iteration, and Assigned To. There are four test plans listed: 'Chapter-wise Course Learning Flow' (ID 6, Active, Assigned to Tarun Aanand S G), 'Login verification' (ID 1, Active, Assigned to Tarun Aanand S G), 'Subject Selection Functionality' (ID 13, Active, Assigned to Tanish), and 'Role-based Access' (ID 10, Active, Assigned to Tanish). A '+ New Test Plan' button is in the top right.

Title	Test Plan ID	State	Area Path	Iteration	Assigned To
Chapter-wise Course Learning Flow	6	Active	Know	Know	Tarun Aanand S G
Login verification	1	Active	Know	Know	Tarun Aanand S G
Subject Selection Functionality	13	Active	Know	Know	Tanish
Role-based Access	10	Active	Know	Know	Tanish

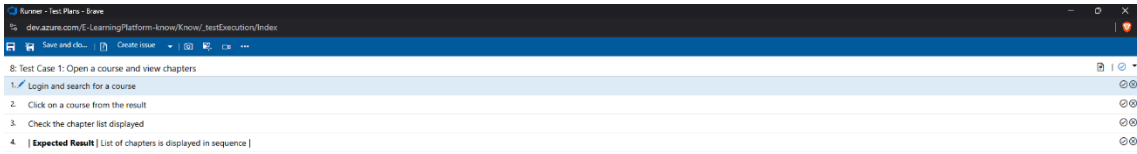
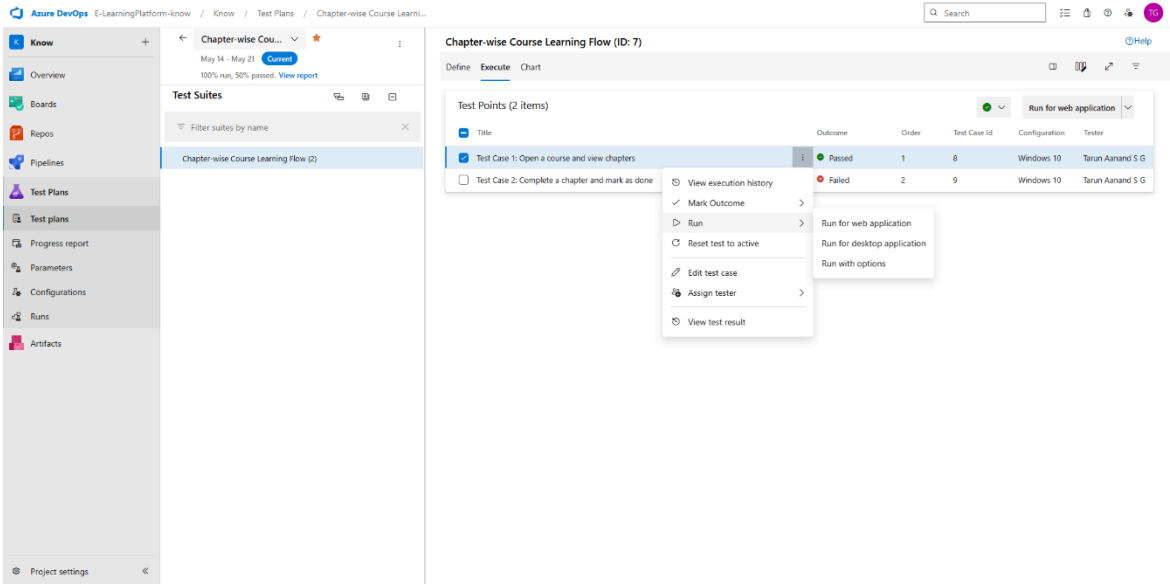
The screenshot shows the details of a test plan named 'Chapter-wise Course Learning Flow (ID: 7)'. The left sidebar is the same as the previous screenshot. The main area shows the 'Test Suites' section with a filter 'Filter suites by name'. Below the filter, 'Chapter-wise Course Learning Flow (7)' is listed. The right pane shows the 'Test Cases (2 items)' table. The table has columns for Title, Order, Test Case Id, Assigned To, and State. There are two test cases: 'Test Case 1: Open a course and view chapters' (Order 1, ID 8, Assigned to Tarun Aanand S G, State Ready) and 'Test Case 2: Complete a chapter and mark as done' (Order 2, ID 9, Assigned to Tarun Aanand S G, State Design). A '+ New Test Case' button is in the top right of the test cases table.

Title	Order	Test Case Id	Assigned To	State
Test Case 1: Open a course and view chapters	1	8	Tarun Aanand S G	Ready
Test Case 2: Complete a chapter and mark as done	2	9	Tarun Aanand S G	Design

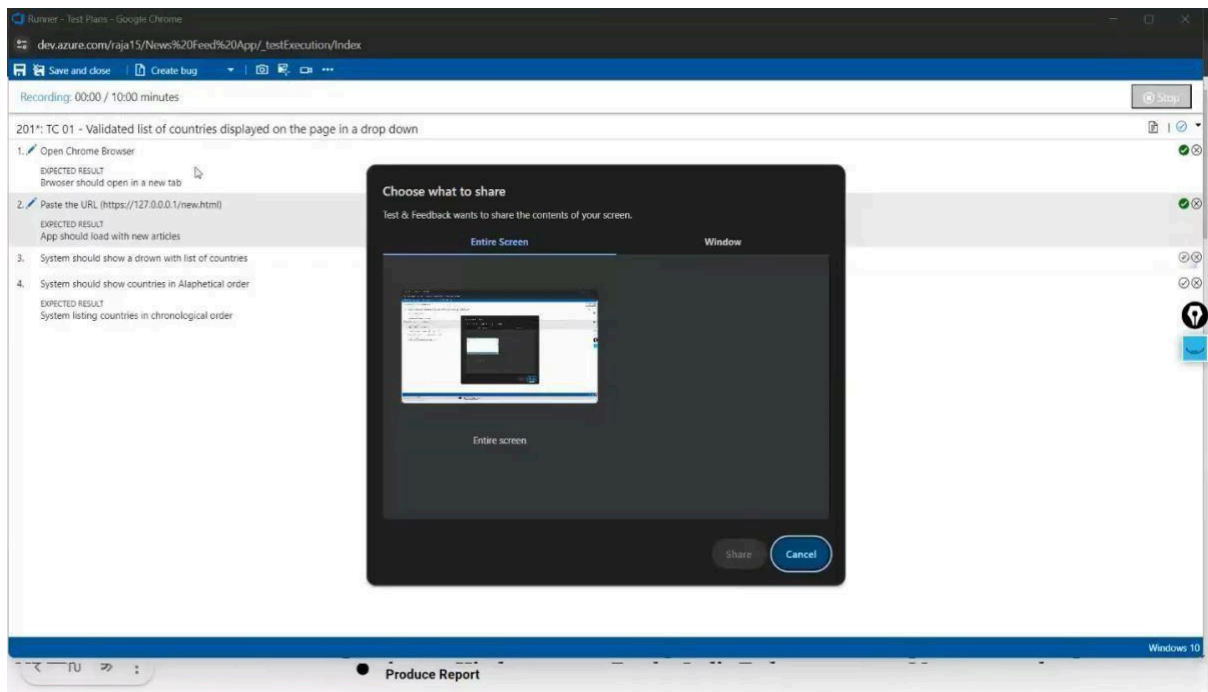
## 4.Installation of test



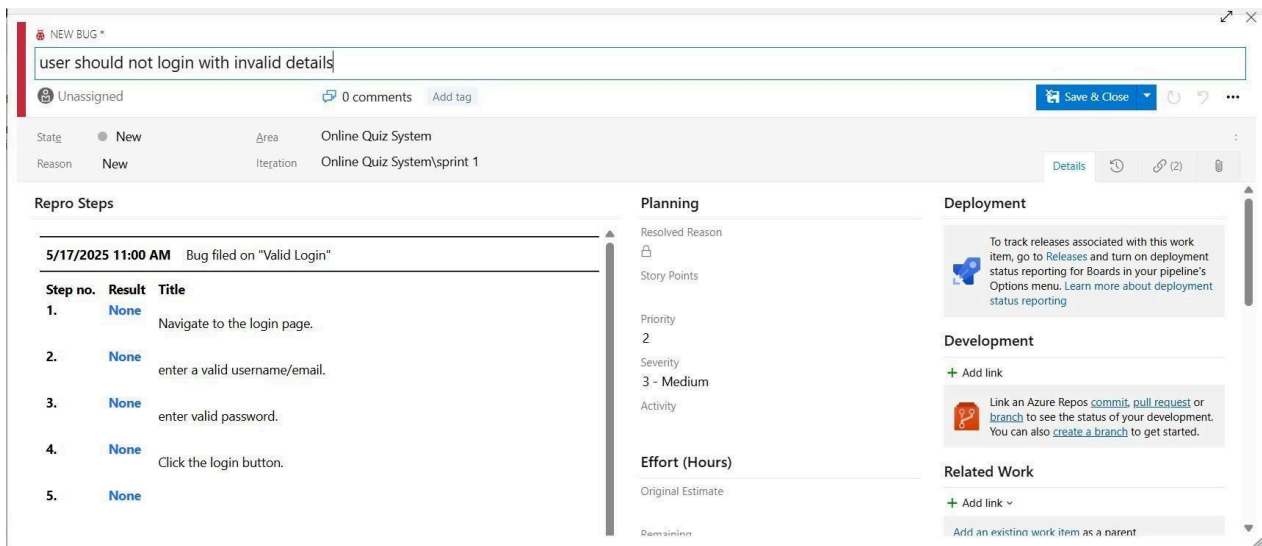
5. Running the test cases



## 6. Recording the test case



## 7. Creating the bug



NEW BUG \*

user should not enter invalid details

Unassigned0 commentsAdd tagSave & Close

StateNew

AreaOnline Quiz System

ReasonNew

IterationOnline Quiz System\sprint 1

System Info

Browser - Name	Google Chrome 136
Browser - Language	en-US
Browser - Height	816
Browser - Width	1536
Browser - User agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36
Operating system - Name	Windows NT 10.0; Win64; x64
Operating system - Architecture	x86_64
Operating system - Processor model	12th Gen Intel(R) Core(TM) i5-12450H
Operating system - Number of processors	12
Memory - Available	6998618112
Memory - Capacity	16831893504
Display - Pixels per inch (X axis)	120
Display - Pixels per inch (Y axis)	120

System info

Found in Build

Integrated in Build

## 8.Test case results

Azure DevOps2315011300815 / Online Quiz System / Test Plans / Login Verification

Online Quiz System

Overview

Boards

Repos

Pipelines

Test Plans

Test plans

Progress report

Parameters

Configurations

Runs

Artifacts

Project settings

Login Verification

Apr 28 - May 12

Past

100% run, 100% passed. View report

Test Suites

Filter suites by name

Login Verification (3)

Login Verification (ID: 23)

DefineExecuteChart

Test Points (3 items)

Title

Valid Login

Invalid Username

Invalid Password

Valid Login

Test Case Results

Outcome	TimeSta...	Configuration	Run by	Tester	Test Plan
Passed	Just now	Windows 10	Ranjani Sai	Ranjani Sai	Login V
Passed	4m ago	Windows 10	Ranjani Sai	Ranjani Sai	Login V
Paused	4m ago	Windows 10	Ranjani Sai	Ranjani Sai	Login V
Failed	6m ago	Windows 10	Ranjani Sai	Ranjani Sai	Login V
Passed	Thursd...	Windows 10	Ranjani Sai	Ranjani Sai	Login V
Failed	Thursd...	Windows 10	Ranjani Sai	Ranjani Sai	Login V

Open execution history for current test point

## 9. Test report summary

**NEW BUG**

User should not enter invalid details

Unassigned 0 comments Add tag Save & Close

State: New Area: Online Quiz System Reason: New Iteration: Online Quiz System\sprint 1

**Repro Steps**

5/17/2025 11:09 AM Bug filed on "Valid Login"

Step no.	Result	Title
1.	None	Navigate to the login page.
2.	None	enter a valid username/email.
3.	None	enter valid password.
4.	None	Click the login button.
5.	None	

**Planning**

Resolved Reason  
Story Points  
Priority: 2  
Severity: 3 - Medium  
Activity  
Effort (Hours)  
Original Estimate  
Permissions

**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**

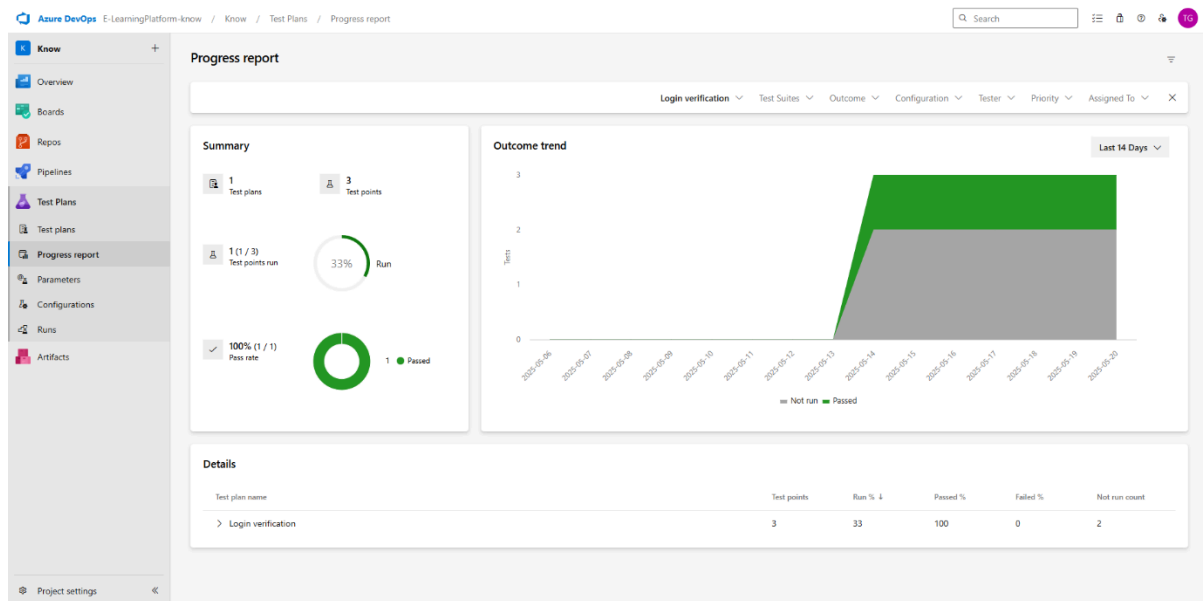
+ Add link  
Link an Azure Repos [commit](#), [pull request](#) or [branch](#) to see the status of your development. You can also [create a branch](#) to get started.

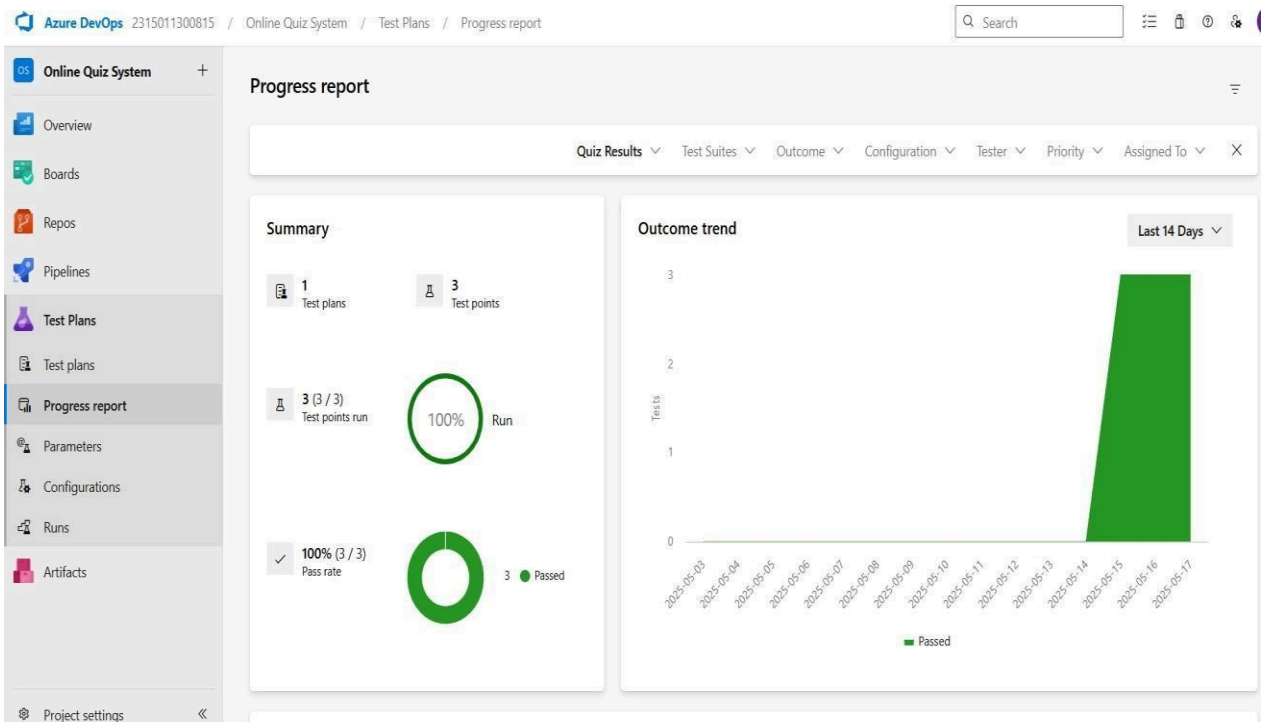
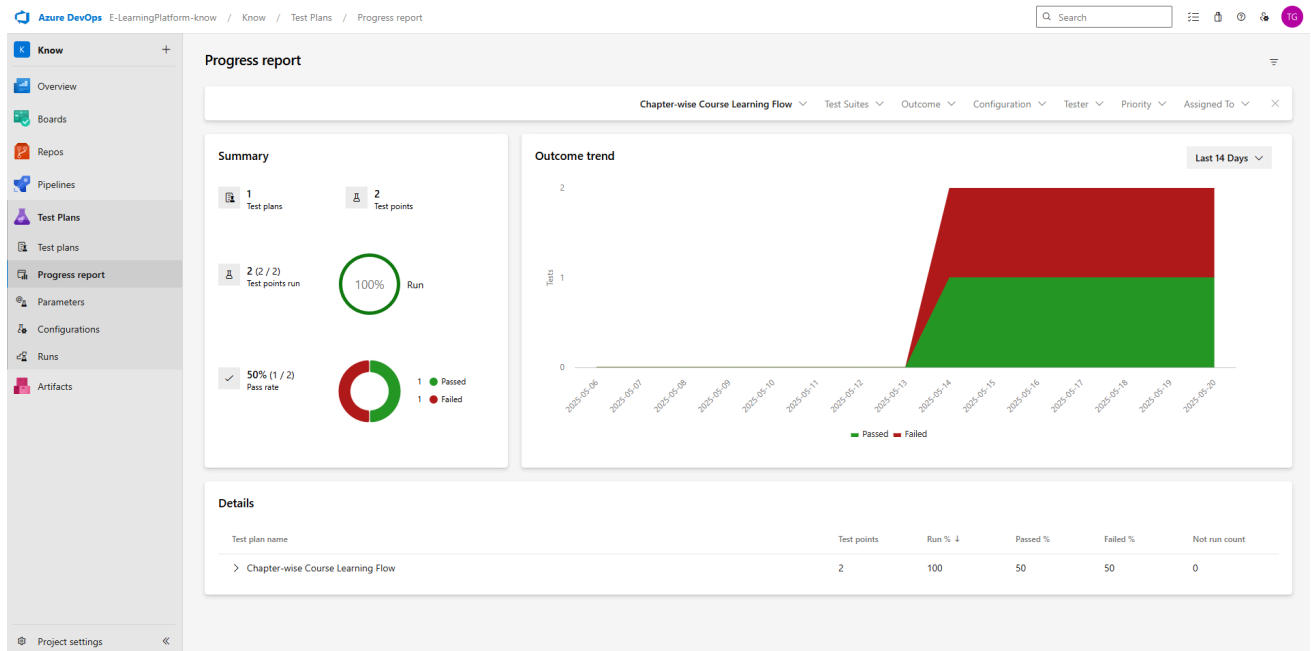
**Related Work**

+ Add link  
Add an existing work item as a parent

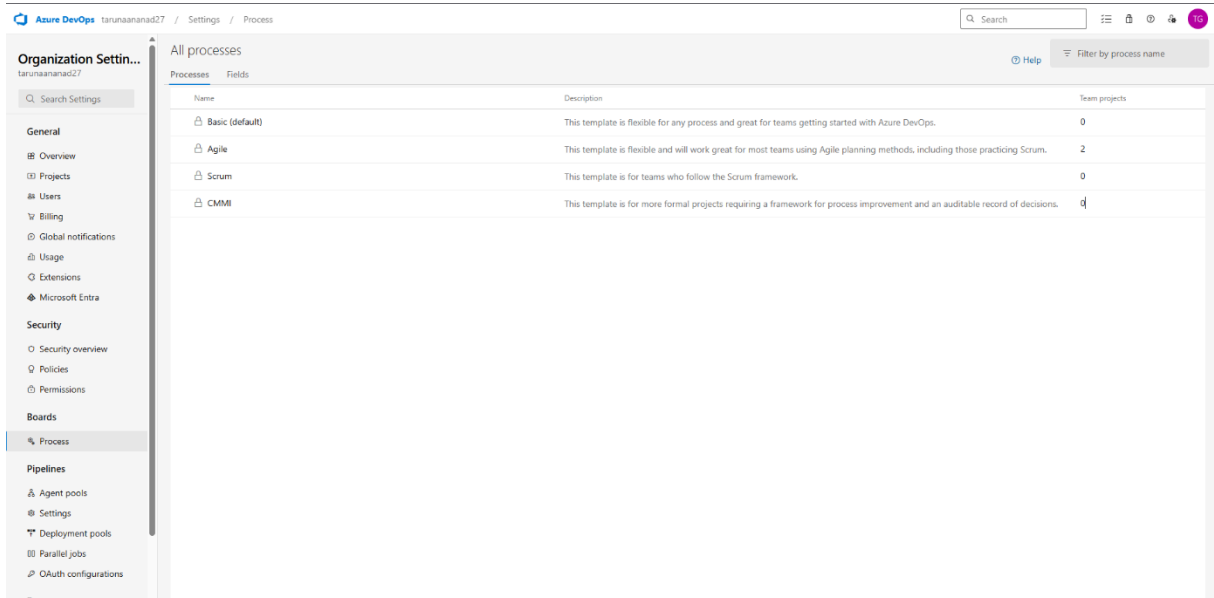
- Assigning bug to the developer and changing state

## 10. Progress report



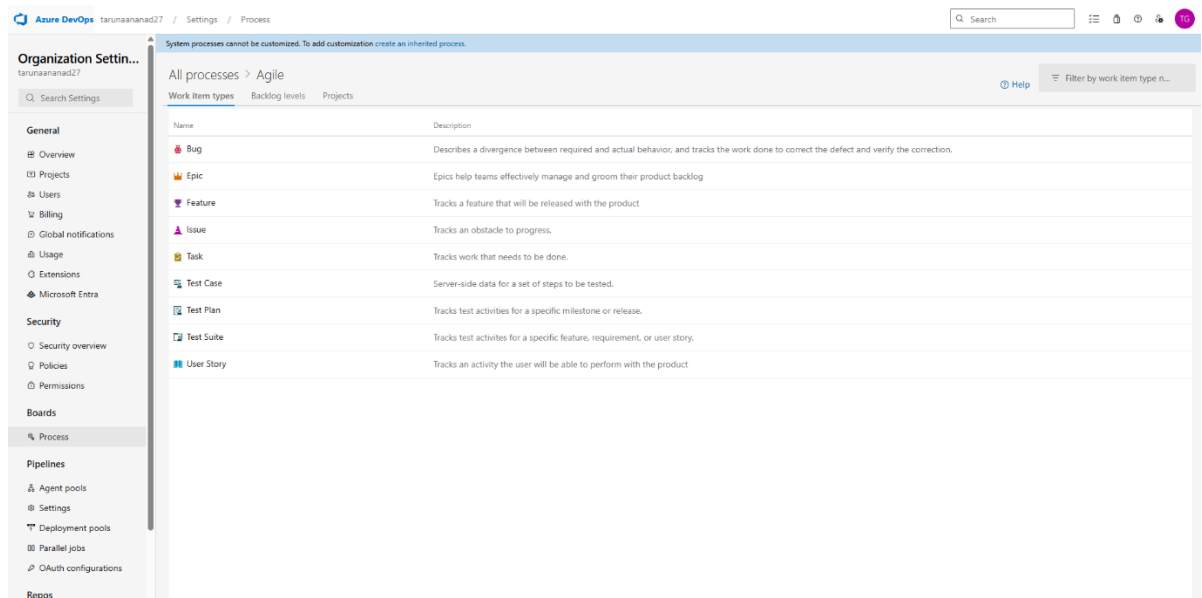


## 11.Changing the test template



The screenshot shows the Azure DevOps Settings page for the 'Process' section. The left sidebar contains the 'Organization Settings' menu with options like General, Overview, Projects, Users, Billing, Global notifications, Usage, Extensions, Microsoft Entra, Security, Boards, Pipelines, and Repos. The main content area is titled 'All processes' and shows a table of process templates. The table has columns for Name, Description, and Team projects. The templates listed are Basic (default), Agile, Scrum, and CMMI.

Name	Description	Team projects
Basic (default)	This template is flexible for any process and great for teams getting started with Azure DevOps.	0
Agile	This template is flexible and will work great for most teams using Agile planning methods, including those practicing Scrum.	2
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 improvement and an auditable record of decisions.	0



The screenshot shows the Azure DevOps Settings page for the 'Agile' work item types. The left sidebar is the same as the previous screenshot. The main content area is titled 'All processes > Agile' and shows a table of work item types. The table has columns for Name and Description. The work item types listed are Bug, Epic, Feature, Issue, Task, Test Case, Test Plan, Test Suite, and User Story.

Name	Description
Bug	Describes a divergence between required and actual behavior, and tracks the work done to correct the defect and verify the correction.
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

## RESULT

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



**EXP NO: 9**

## **CI/CD PIPELINES IN AZURE**

### **AIM**

To implement a Continuous Integration and Continuous Deployment (CI/CD) pipeline in Azure DevOps for automating the build, testing, and deployment process of the E-Learning Platform, ensuring faster delivery and improved software quality.

### **PROCEDURE**

#### **Steps to Create and implement pipelines in Azure:**

1. Sign in to Azure DevOps and Navigate to Your Project  
Log in to [dev.azure.com](https://dev.azure.com), select your organization, and open the project where your Student Management System code resides.
2. Connect a Code Repository (Azure Repos or GitHub)  
Ensure your application code is stored in a Git-based repository such as Azure Repos or GitHub. This will be the source for triggering builds and deployments in your pipeline.
3. Create a New Pipeline  
Go to the Pipelines section on the left panel and click “Create Pipeline”.  
Choose your source (e.g., Azure Repos Git or GitHub), and then select the repository containing your project code.
4. Choose the Pipeline Configuration  
You can select either the YAML-based pipeline (recommended for version control and automation) or the Classic Editor for a GUI-based setup. If using YAML, Azure DevOps will suggest a template or allow you to define your own.
5. Define Build Stage (CI - Continuous Integration) from YAML file.
6. Install dependencies (e.g., npm install, dotnet restore).

7. Build the application (dotnet build, npm run build).
8. Run unit tests (dotnet test, npm test).
9. Publish build artifacts to be used in the release stage.
10. Save and Run the Pipeline for the First Time

Save the YAML or build definition and click “Run”.

Azure will fetch the latest code and execute the defined build and test stages.
11. Configure Continuous Deployment (CD)

Navigate to the Releases tab under Pipelines and click “New Release Pipeline”. Add an Artifact (from the build stage) and create a new Stage (e.g., Development, Production).
12. Configure the CD stage with deployment tasks such as deploying to Azure App Service, running database migrations or scripts, and restarting services using the Azure App Service Deploy task linked to your subscription and app details.
13. Set Triggers and Approvals

Enable continuous deployment trigger so the release pipeline runs automatically after a successful build. For production environments, configure pre-deployment approvals to ensure manual verification before release.
14. Monitor Pipelines and Manage Logs

View all pipeline runs under the Runs section.

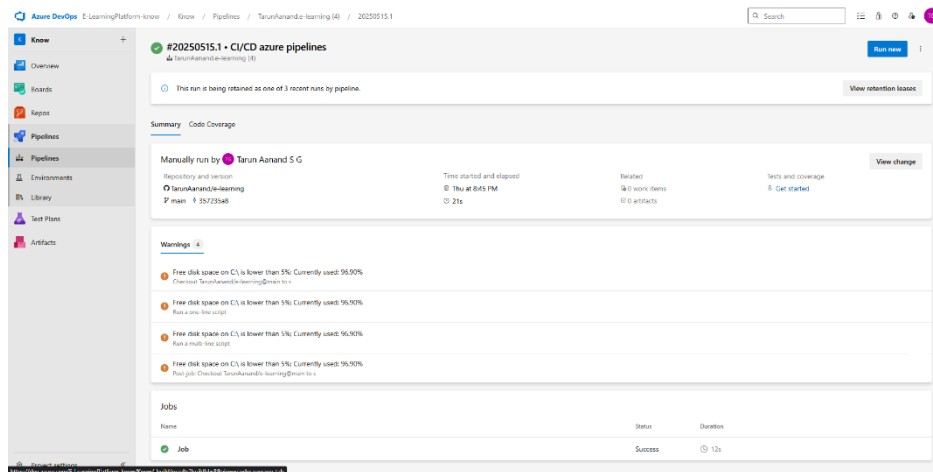
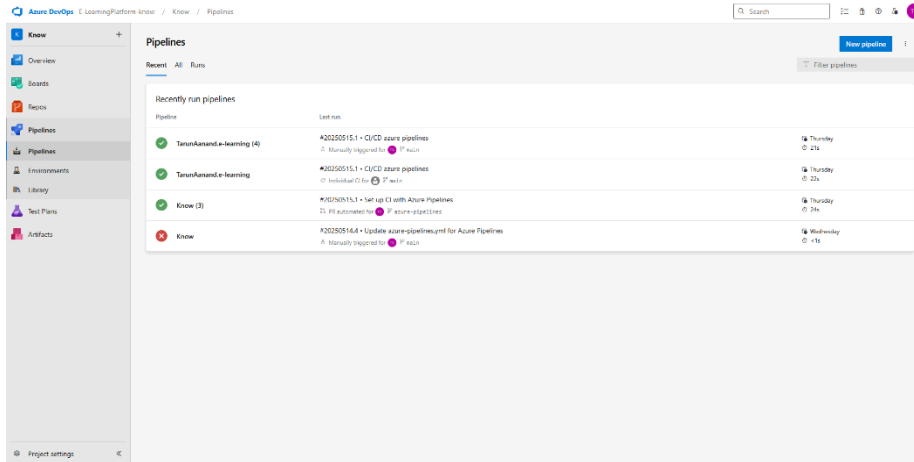
Check logs for build/test/deploy stages to debug any errors.

You can also integrate email alerts or Microsoft Teams notifications for build failures.
15. Review and Maintain Pipelines

Regularly update your pipeline tasks or YAML configurations as your application grows.

Ensure pipeline runs are clean and artifacts are stored securely.

Integrate quality gates and code coverage policies to maintain code quality.



## RESULT

Thus, the pipelines for the given project “E-Learning Platform” has been executed successfully.

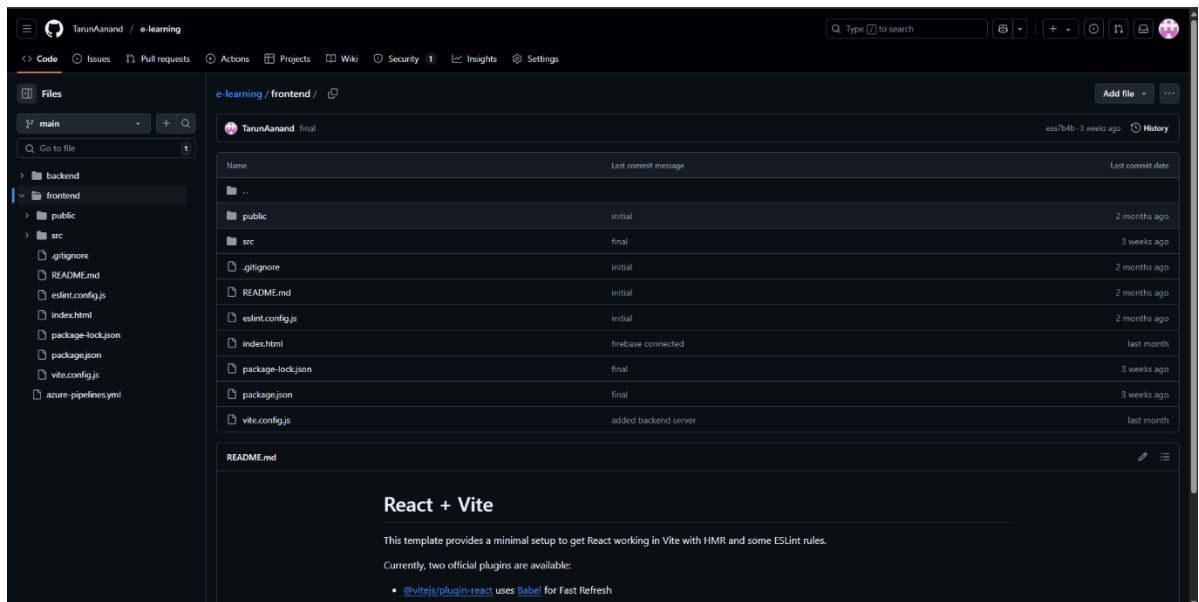
**EXP NO: 10**

## **GITHUB: PROJECT STRUCTURE & NAMING CONVENTIONS**

### **Aim:**

To provide a clear and organized view of the project's folder structure and file naming conventions, helping contributors and users easily understand, navigate, and extend the E-Learning platform.

### **GitHub Project Structure**



The screenshot displays the Azure DevOps web interface for a repository named 'E-LearningPlatform-know'. The left sidebar shows navigation options: Overview, Boards, Repos, Files, Commits, Pushes, Branches, Tags, Pull requests, Advanced Security, Pipelines, Test Plans, and Artifacts. The 'Files' view is active, showing a tree structure of the repository. The 'frontend' directory is selected, revealing its contents: public, src, .gitignore, eslint.config.js, index.html, package-lock.json, package.json, README.md, and vite.config.js. The right pane shows the 'frontend' directory's commit history table.

Name	Last change	Commits
public	Mar 26	3757a0b2 initial tarun
src	May 2	eea7b4bc final tarun
.gitignore	Mar 26	3757a0b2 initial tarun
eslint.config.js	Mar 26	3757a0b2 initial tarun
index.html	Apr 20	53ee6cfd firebase connected tarun
package-lock.json	May 2	eea7b4bc final tarun
package.json	May 2	eea7b4bc final tarun
README.md	Mar 26	3757a0b2 initial tarun
vite.config.js	Apr 5	81b8859d added backend server tarun

Below the table, there is a section titled 'React + Vite' which states: 'This template provides a minimal setup to get React working in Vite with HMR and some ESLint rules. Currently, two official plugins are available: @vitejs/plugin-react (uses Babel for Fast Refresh) and @vitejs/plugin-react-swc (uses SWC for Fast Refresh)'. It also includes a section for 'Expanding the ESLint configuration' with a recommendation to use TypeScript and enable type-aware lint rules.

## Result:

The GitHub repository clearly displays the organized project structure and consistent naming conventions, making it easy for users and contributors to understand and navigate the codebase.