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 Diagram and Sequence Diagram.
7.	**04/3/25**	Designing Use Case Diagram and Activity Diagram.
8.	**25/3/25**	Testing – Test Plans and Test Cases.
9.	**15/4/25**	Load Testing and Pipelines.
10.	**22/4/25**	GitHub: Project Structure & Naming Conventions.

EXP NO: 1	AZURE DEVOPS ENVIRONMENT SETUP

Aim:

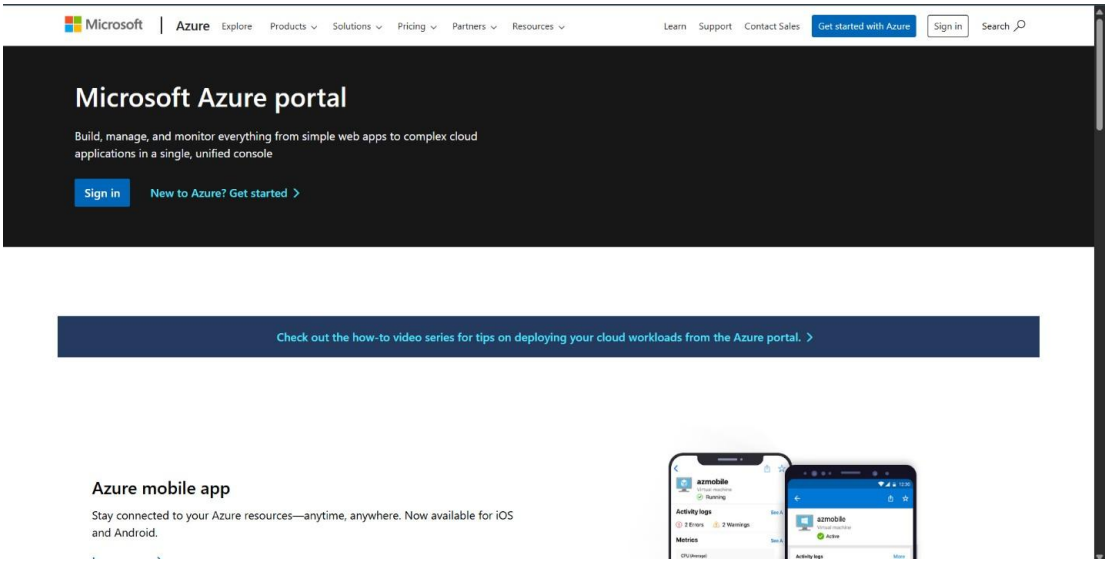
To set up and access the Azure DevOps environment by creating an organization through the Azure portal.

INSTALLATION

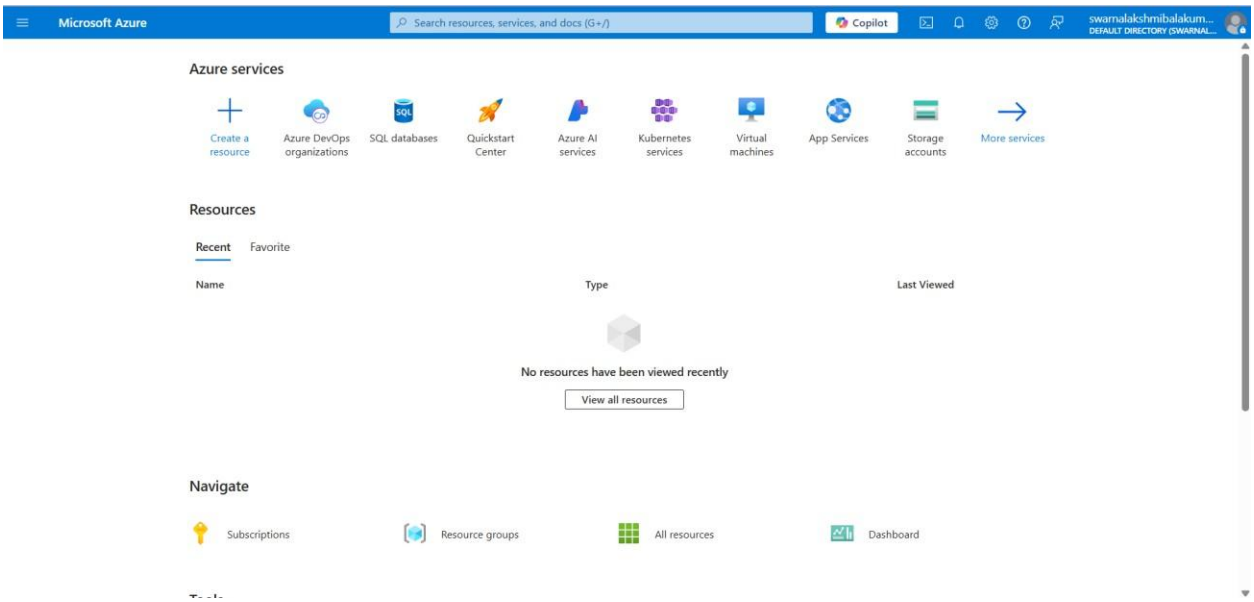
1. Open your web browser and go to the Azure website: https://azure.microsoft.com/en-us/getstarted/azure-portal.

Sign in using your Microsoft account credentials.

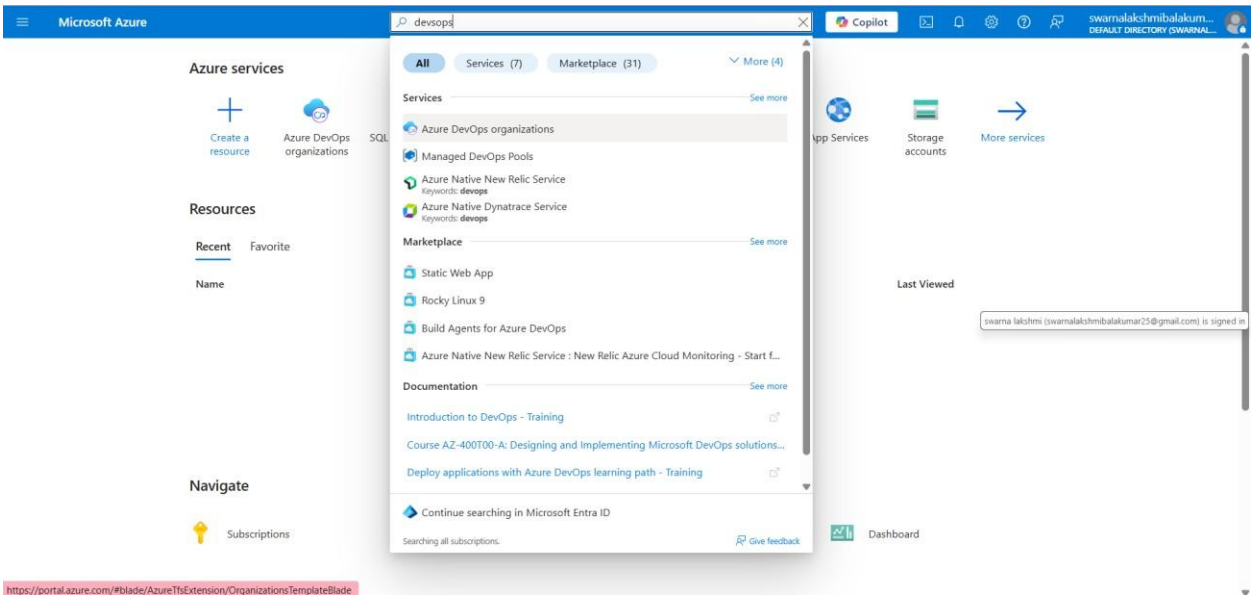
If you don't have a Microsoft account, you can create one here: https://signup.live.com/?lic=1



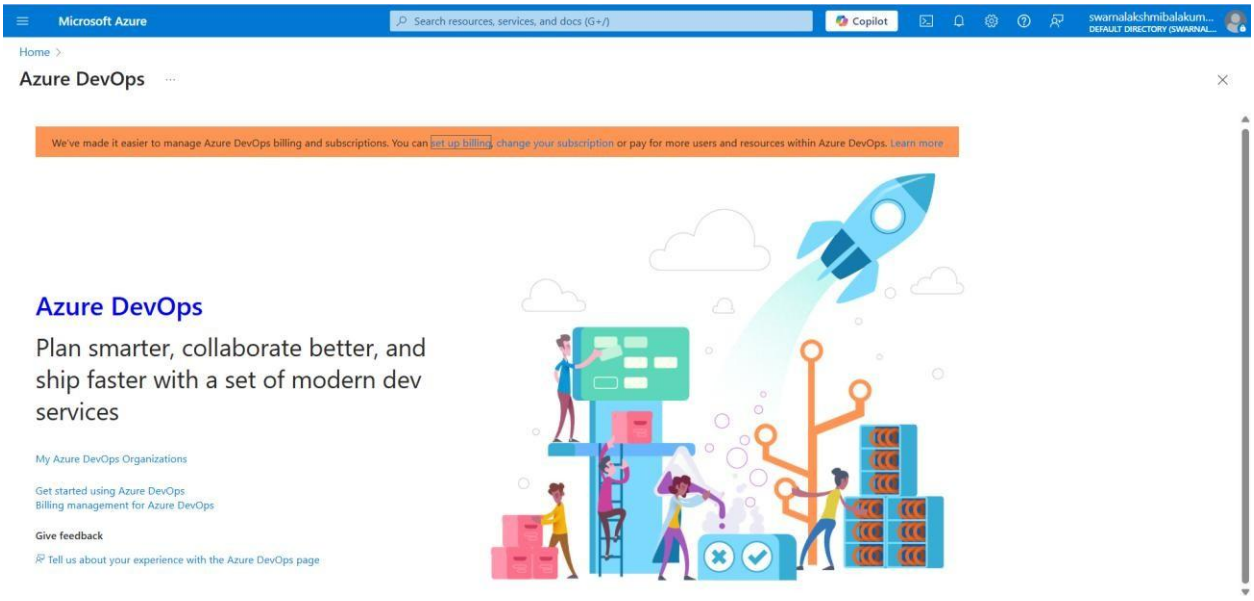
2. Azure home page



3. Open a 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:

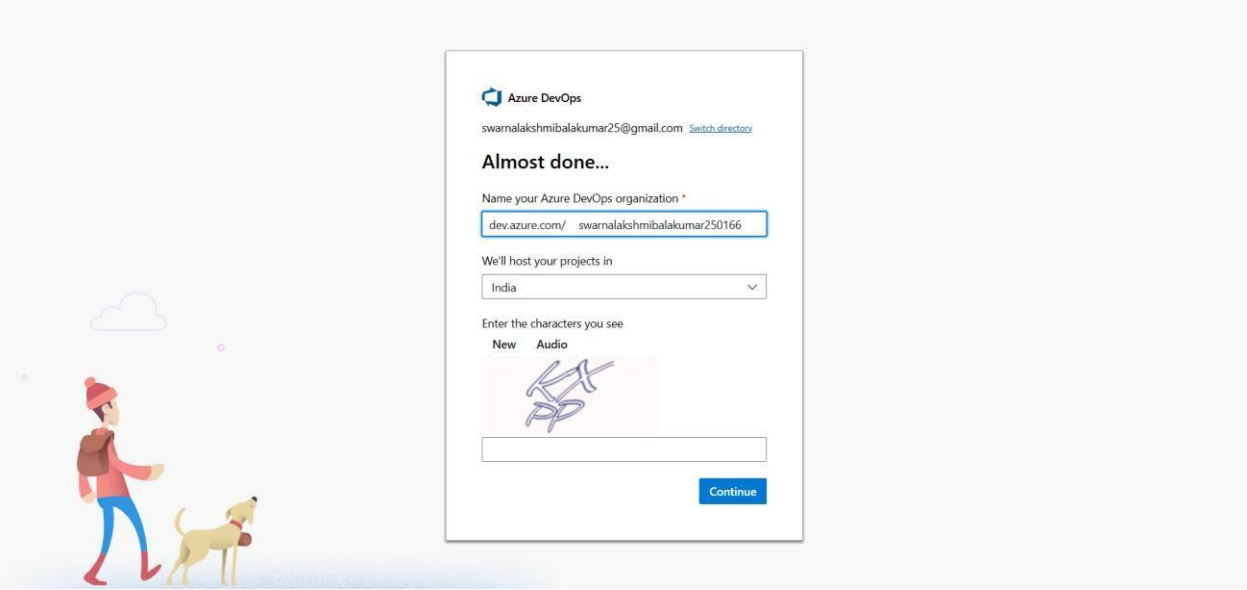
Successfully accessed the Azure DevOps environment and created a new organization through the Azure portal.

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

a. 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.

b. On the organization's **Home page**, click on the **New Project** button.

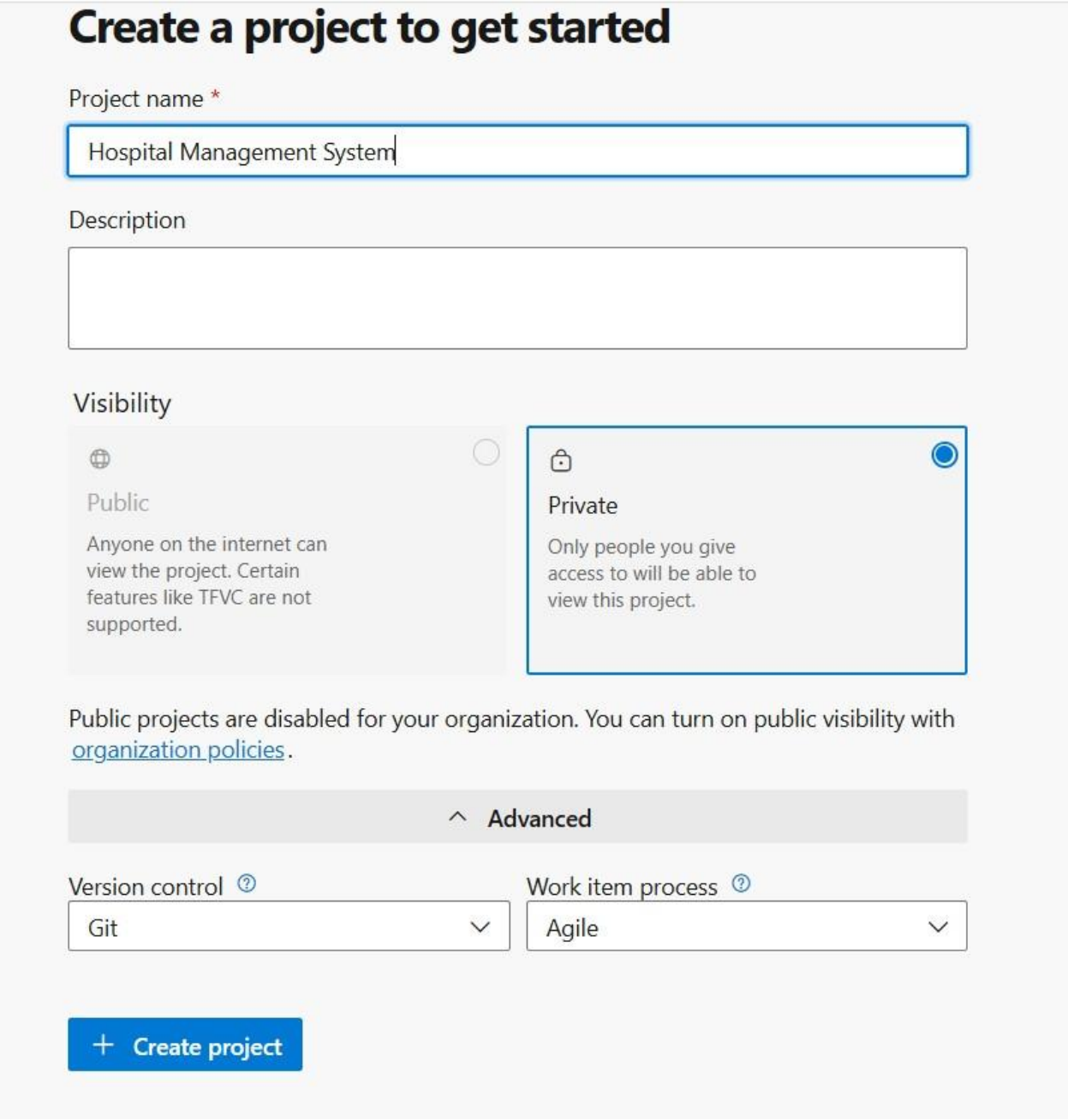
c. Enter the project name, description, and visibility options:

***Name**:* Choose a name for the project (e.g., LMS).

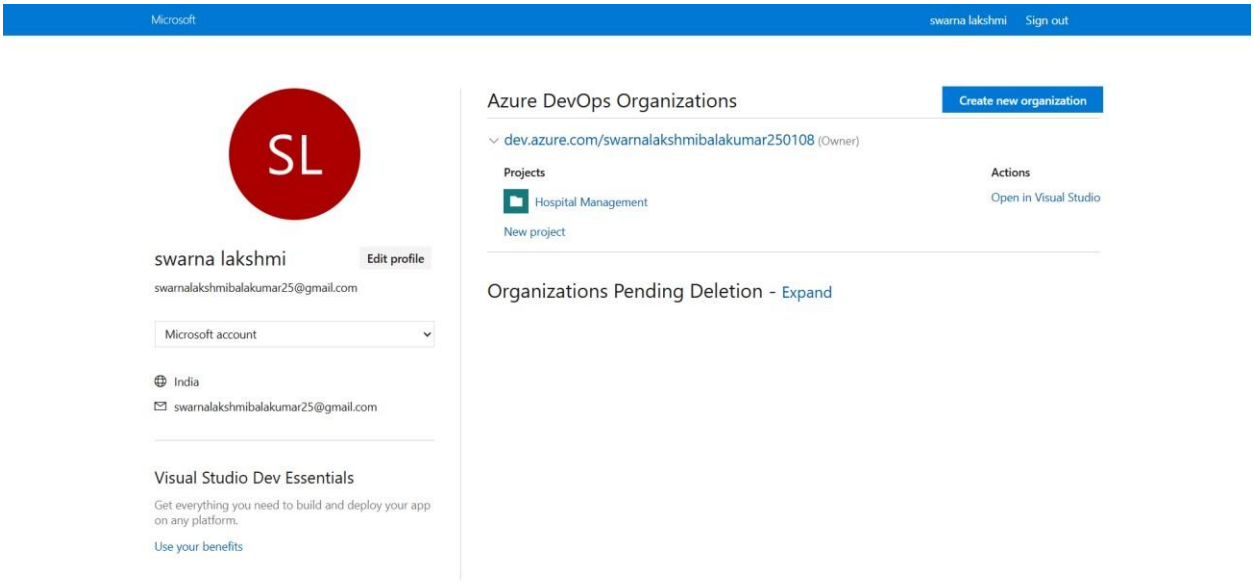
***Description**:* Optionally, add a description to provide more context about the project.

***Visibility**:* Choose whether you want the project to be **Private** (accessible only to those invited) or **Public** (accessible to anyone).

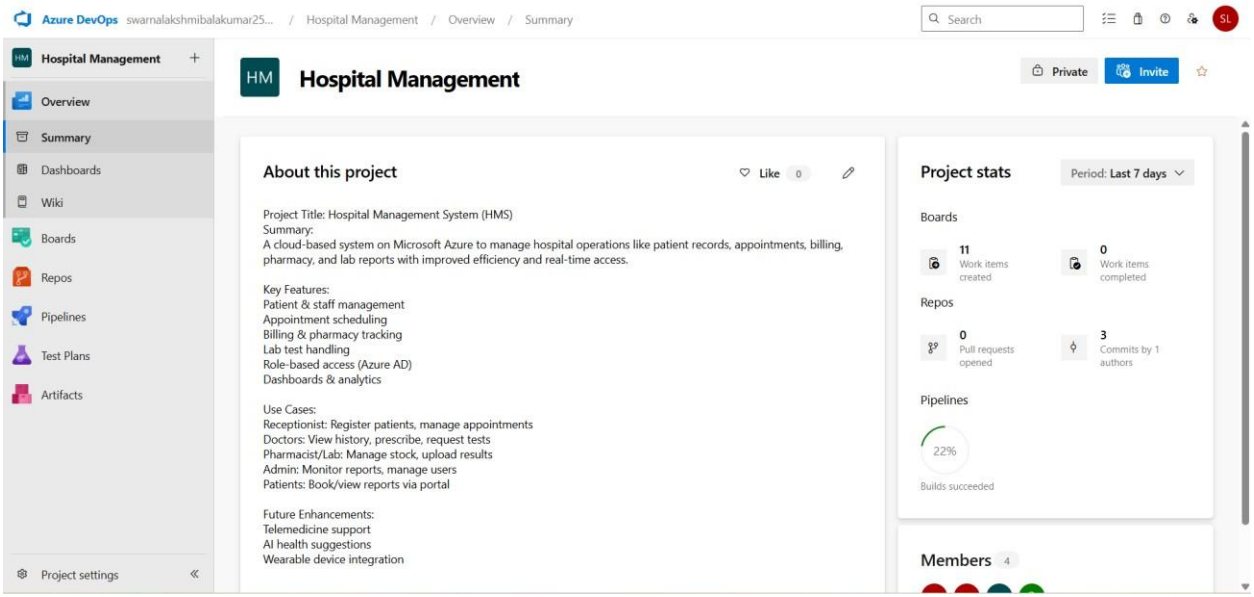
d. Once you've filled out the details, click **Create** to set up your first project.



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.



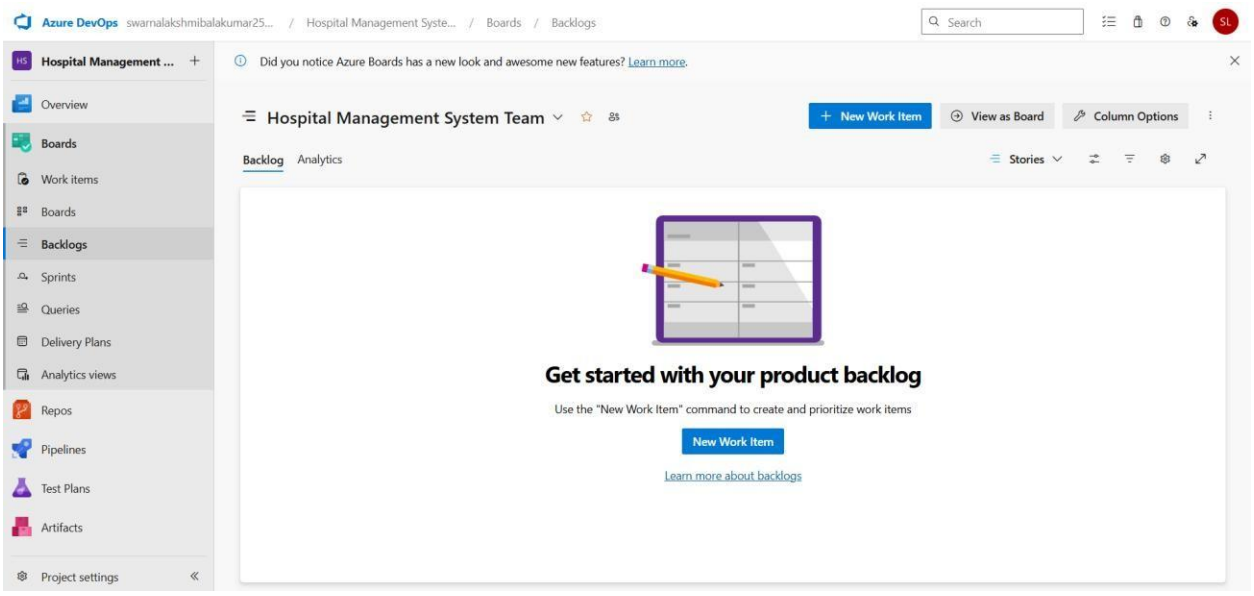
4. Project dashboard

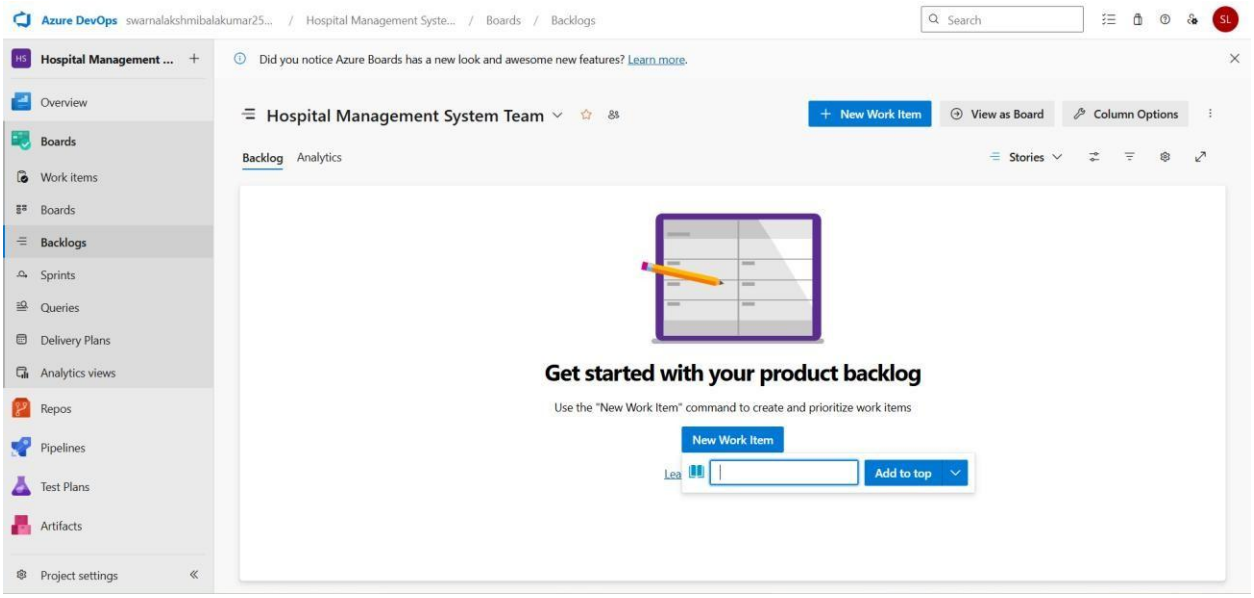


5. To manage user stories:

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:

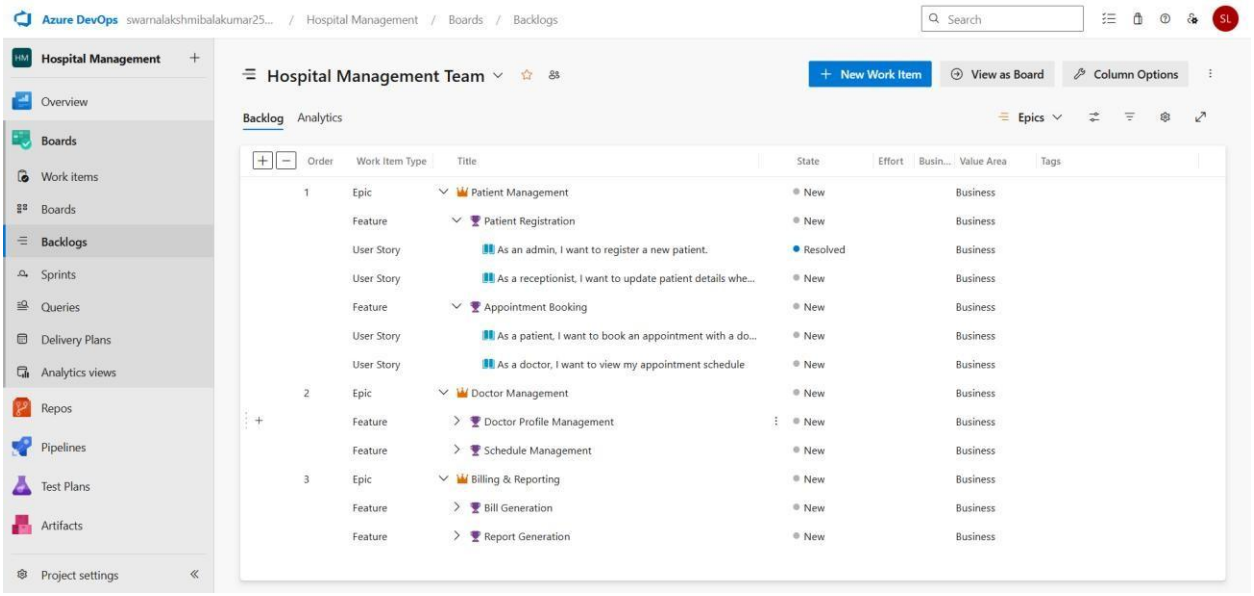
Successfully created an Azure DevOps project with user story management and agile workflow setup.

EXP NO: 3	SETTING UP EPICS, FEATURES, AND USER STORIES FOR PROJECT PLANNING

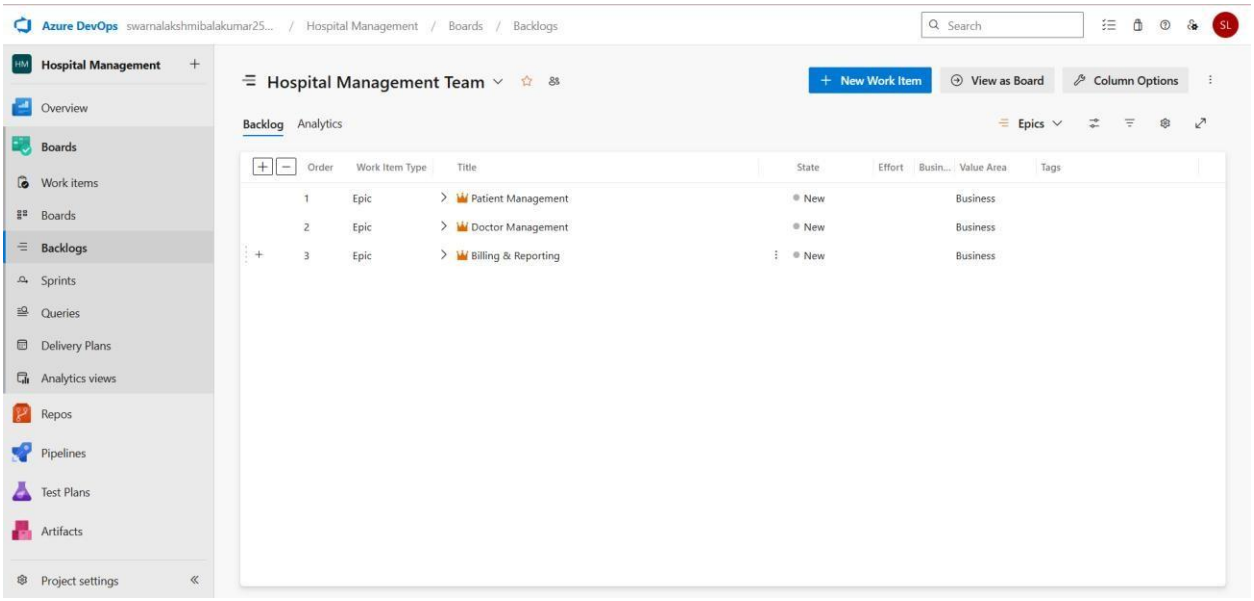
Aim:

To create epics, user stories, features, and tasks for the project, Hospital Management System.

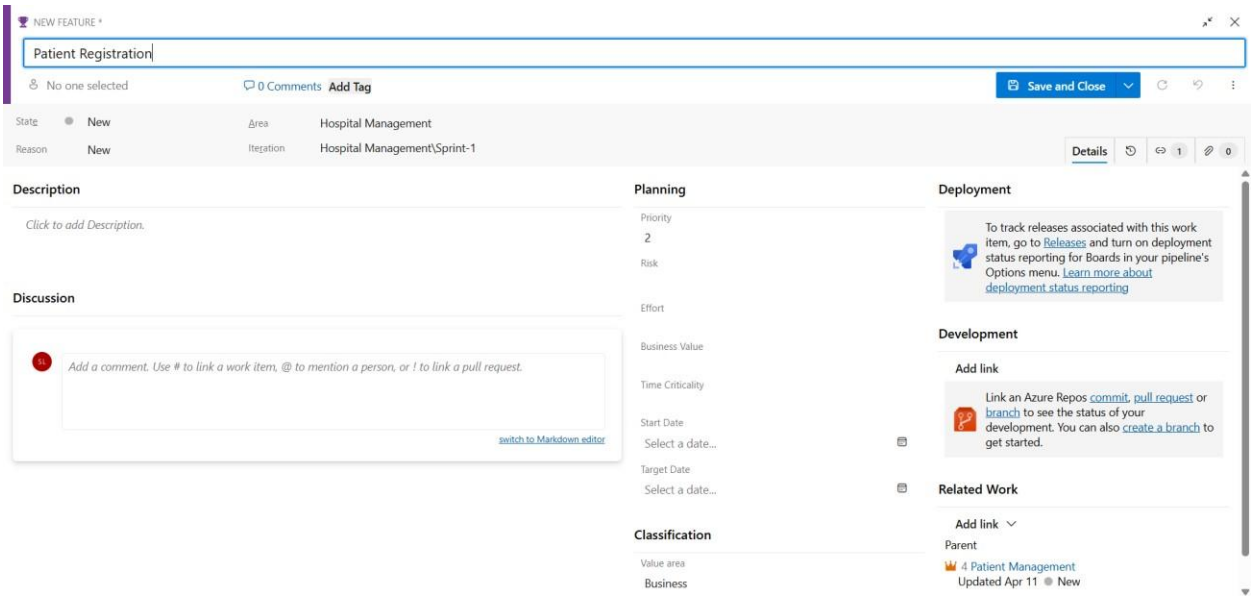
Create Epic, Features, User Stories, Task



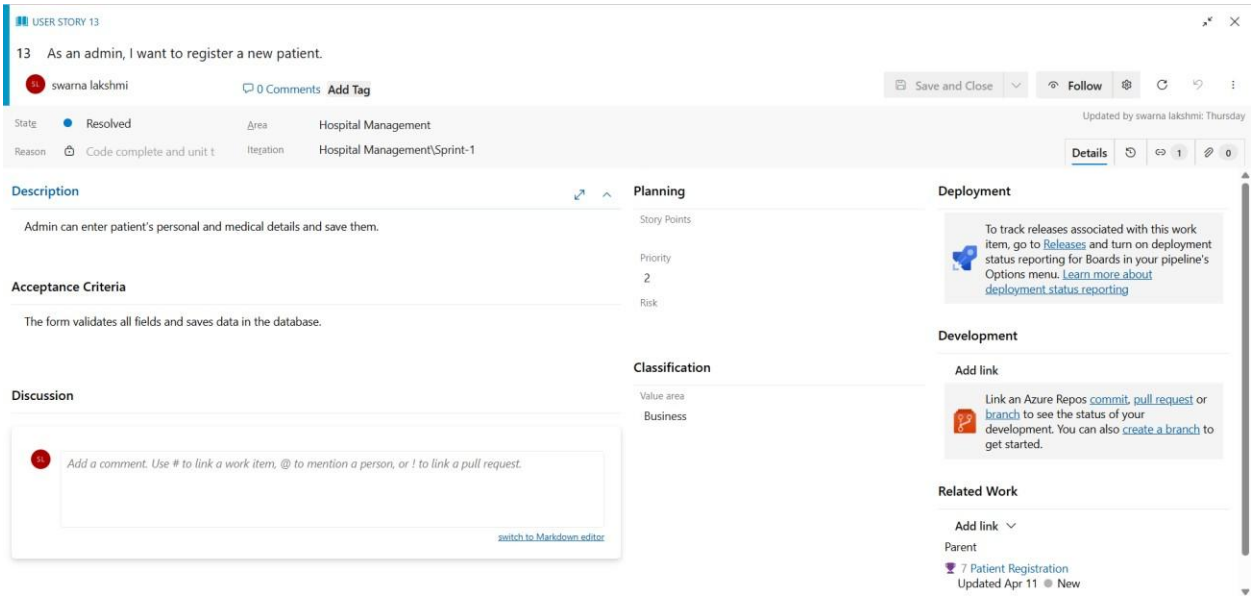
1. Fill in Epics



2. Fill in Features



3. Fill in User Stories



Result:

Thus, epics, features, user stories, and tasks have been created successfully.

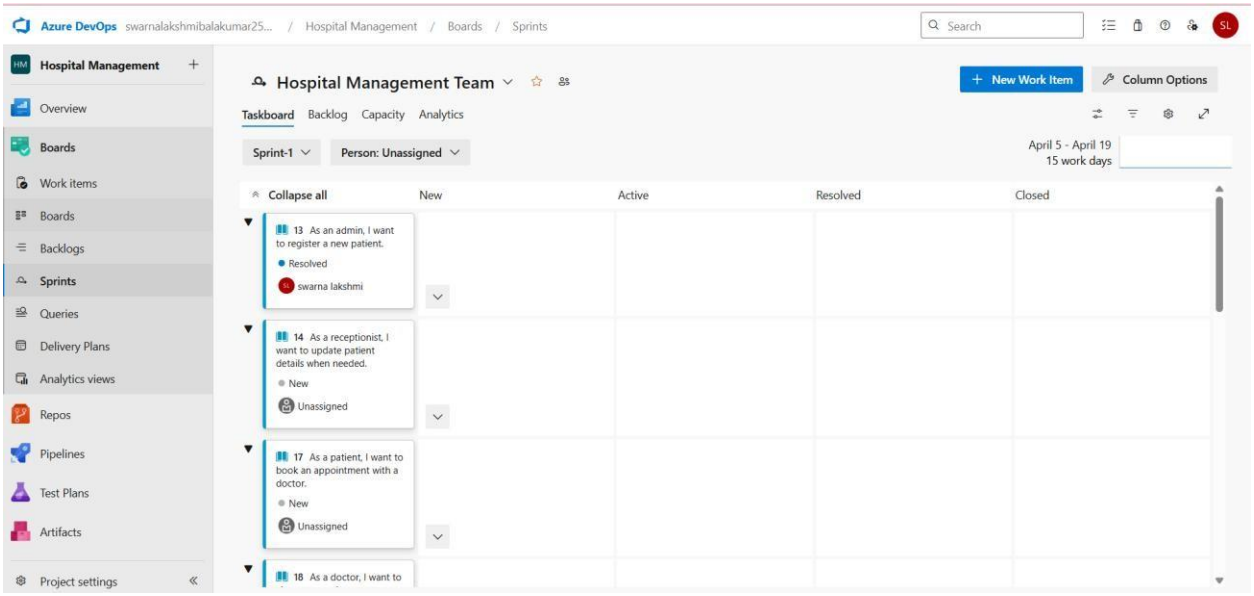
EXP NO: 4	SPRINT PLANNING

Aim:

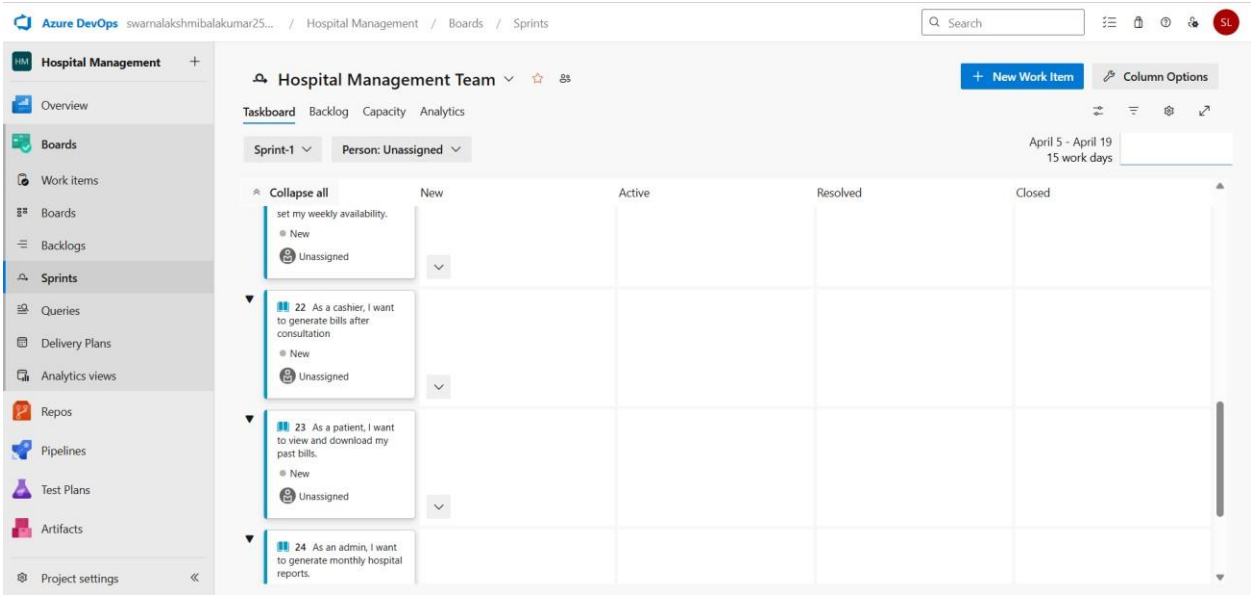
To assign a user story to a specific sprint for the project, Hospital Management System.

SPRINT PLANNING

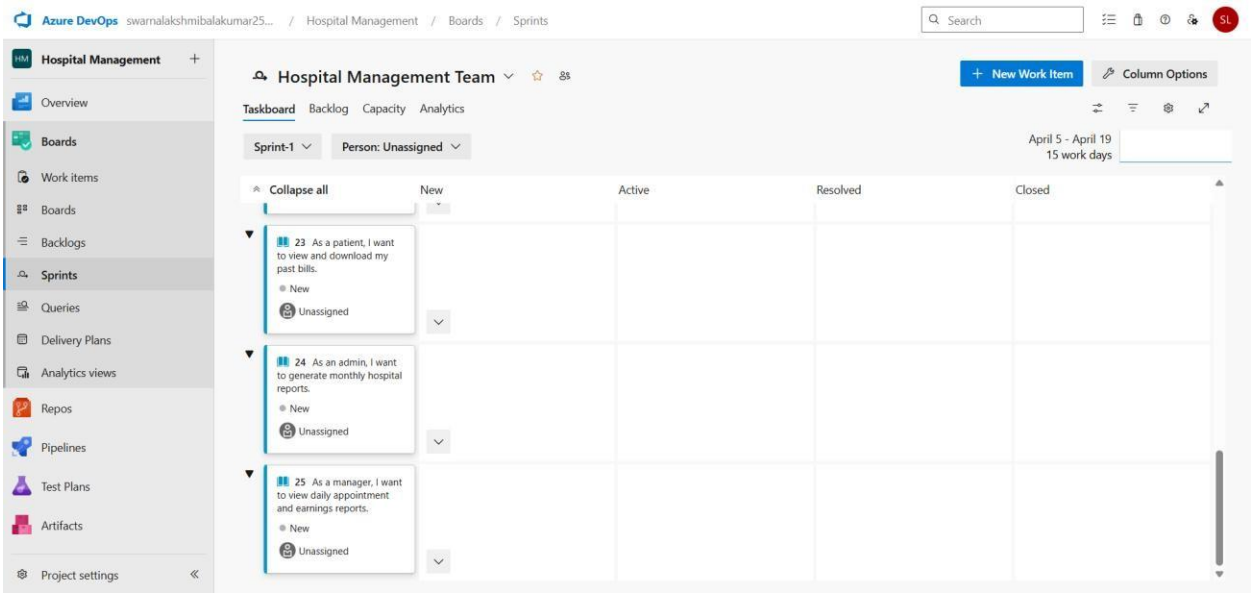
Sprint 1



Sprint 2



Sprint 3



Result:

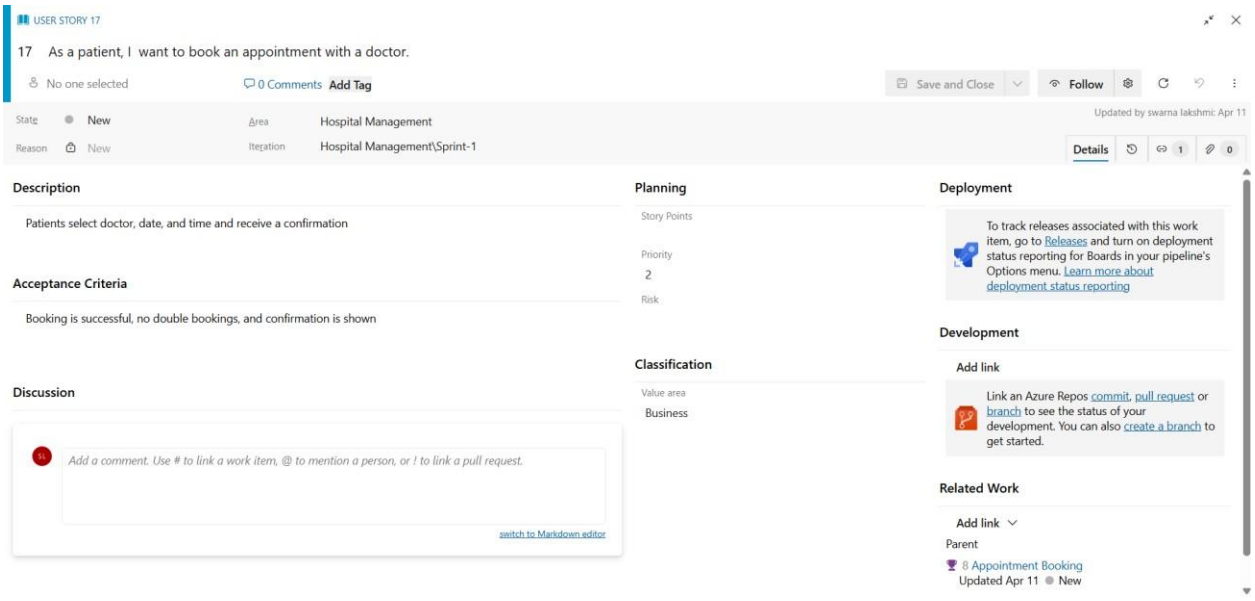
The Sprints are created for the project, Hospital Management System.

EXP NO: 5	POKER ESTIMATION

Aim:

Create Poker Estimation for the user stories for the project, Hospital Management System.

Poker Estimation



Result:

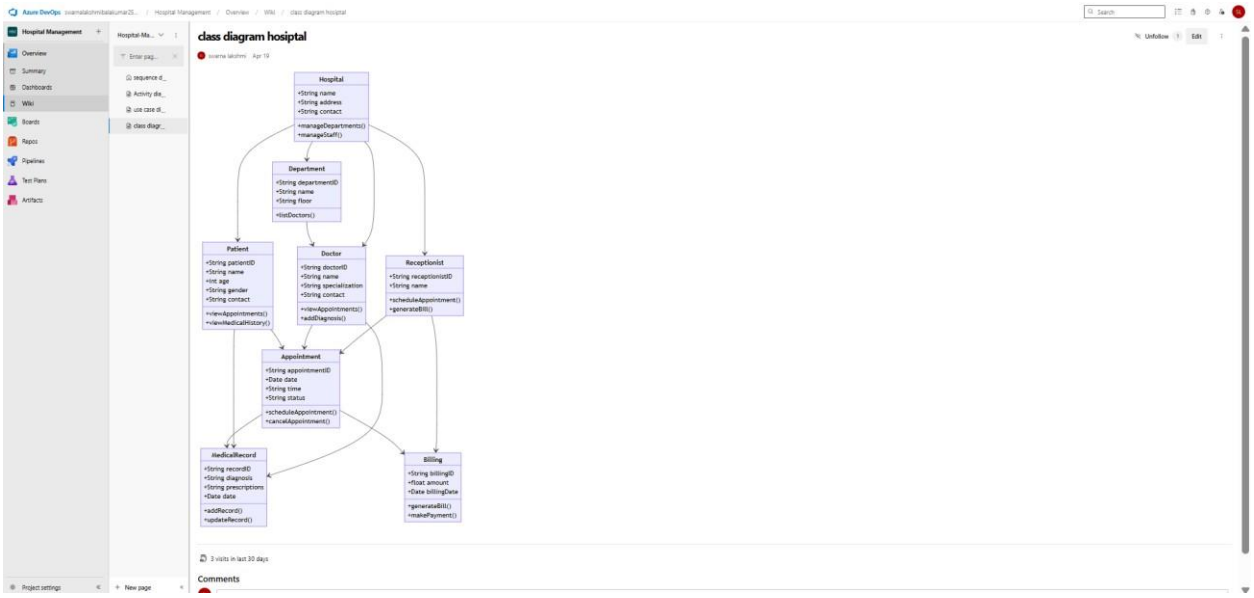
The Estimation/Story Points is created for the project using Poker Estimation.

EXP NO: 6	DESIGNING CLASS DIAGRAM AND SEQUENCE DIAGRAM

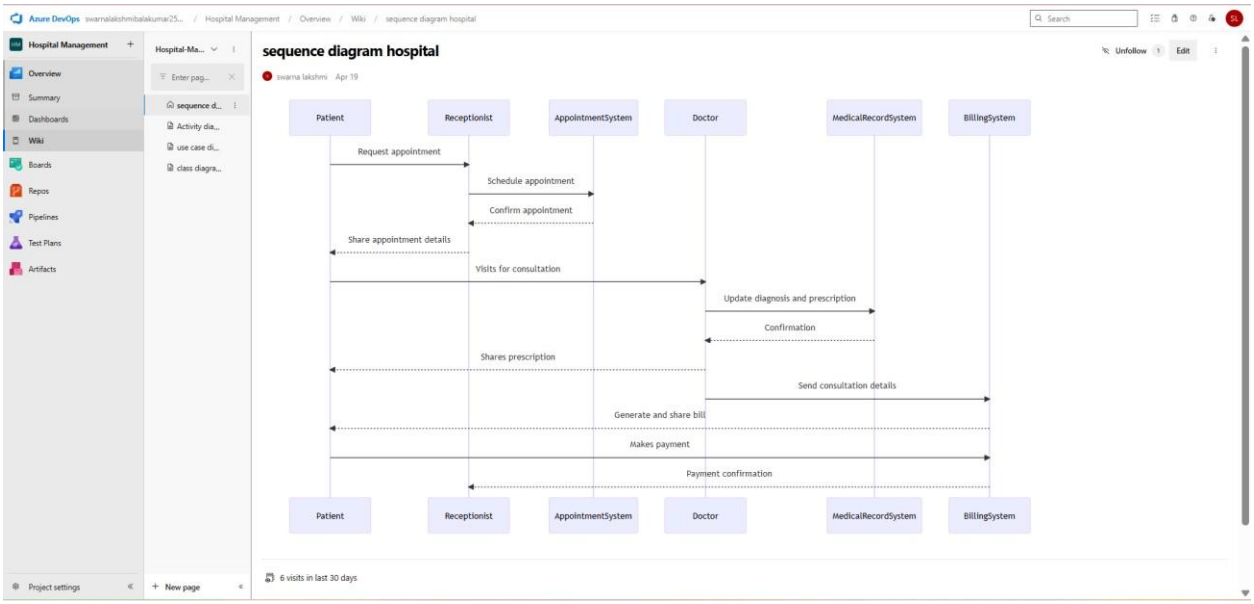
Aim:

To design a Class Diagram and Sequence Diagram for the project, Hospital Management System.

6A. Class Diagram



6B. Sequence Diagram



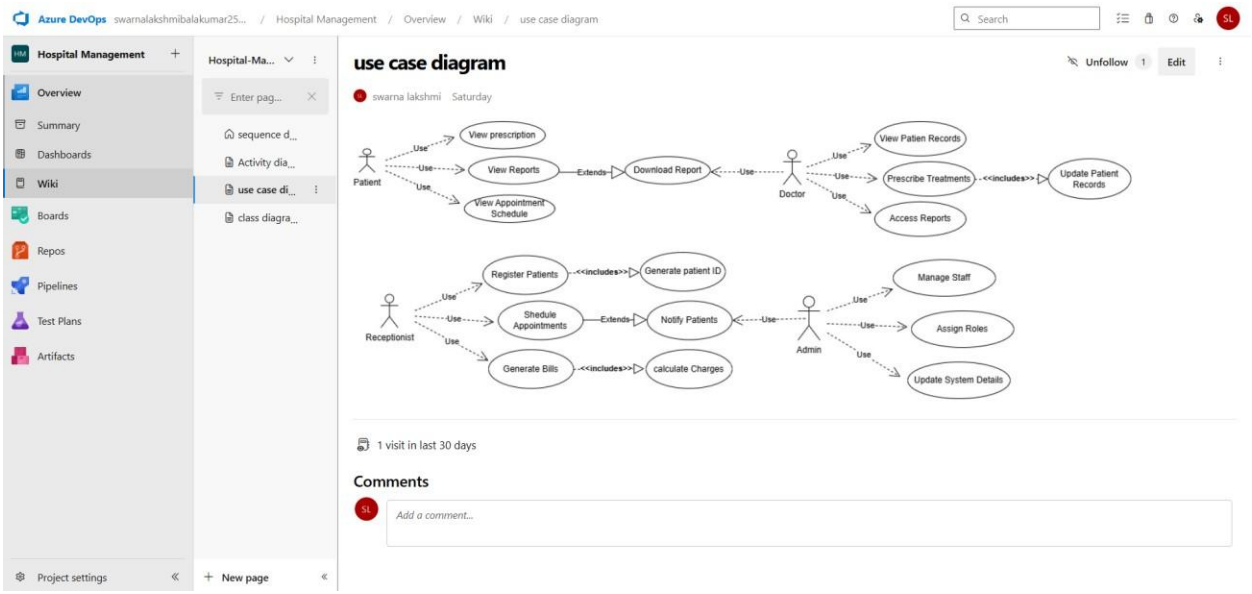
Result: The Class and Sequence Diagrams are designed successfully for the project, Hospital Management System.

EXP NO: 7	DESIGNING USE CASE DIAGRAM AND ACTIVITY DIAGRAM

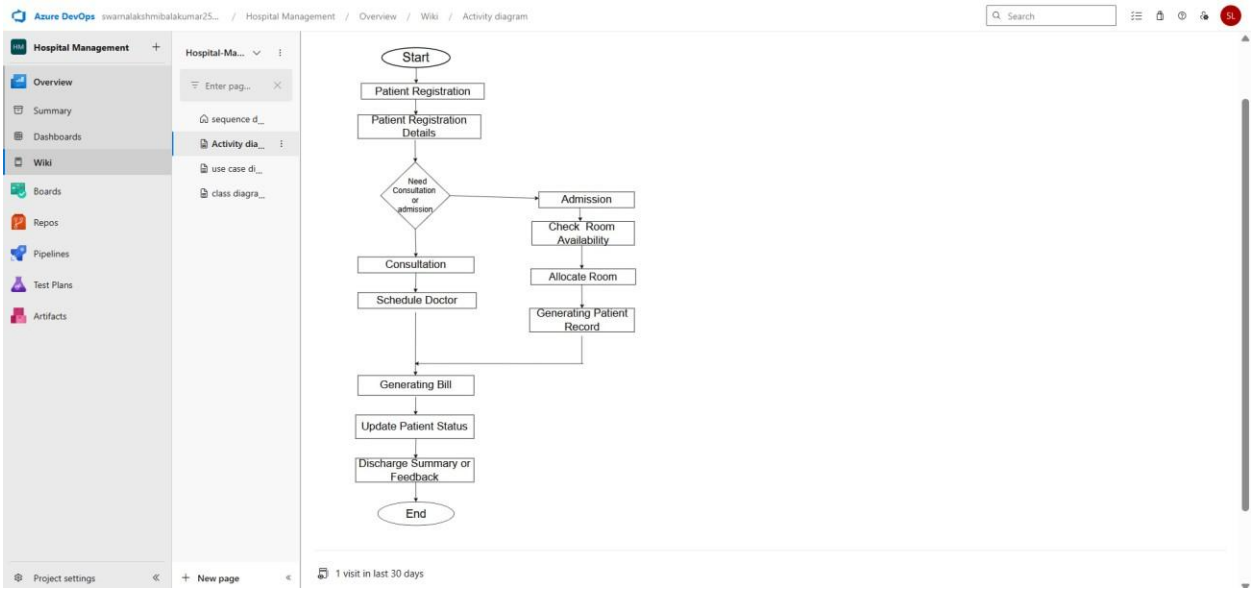
Aim:

To design a Use Case Diagram and an Activity Diagram for the project, Hospital Management System.

7A. Use Case Diagram



7B. Activity Diagram



Result: The Use Case and Activity Diagrams are designed successfully for the project, Hospital Management System.

EXP NO: 8	TESTING – TESTING – TEST PLANS AND TEST CASES

Aim:

Test Plans and Test Cases for the Hospital Management System (HMS), showcasing **happy path** and **error scenarios** for **at least five user stories**, designed and documented for execution on **Azure DevOps platform**.

Test Planning and Test Case

Test Case Design Procedure

- **Understand Core Features of the Application**
  - Patient Registration
  - Appointment Booking
  - Billing and Report Generation
  - Doctor Availability Management
  - Prescription Management

1. **Define User Interactions**

Simulate real scenarios (e.g., register patient, book appointment, generate bill, update availability).

2. **Design Happy Path Test Cases**

Validate expected flows (e.g., successful patient registration, successful booking).

3. **Design Error Path Test Cases**

Simulate invalid inputs or system limitations (e.g., missing fields, double booking, unavailable doctor).

4. **Break Down Steps and Expected Results**

Each test case includes **step-by-step actions** and **clearly defined expected outcomes**.

5. **Use Clear Naming and IDs**

Example: TC_PM_001 – Register New Patient Successfully

Naming is consistent with Azure DevOps standards.

6. **Separate Test Suites**

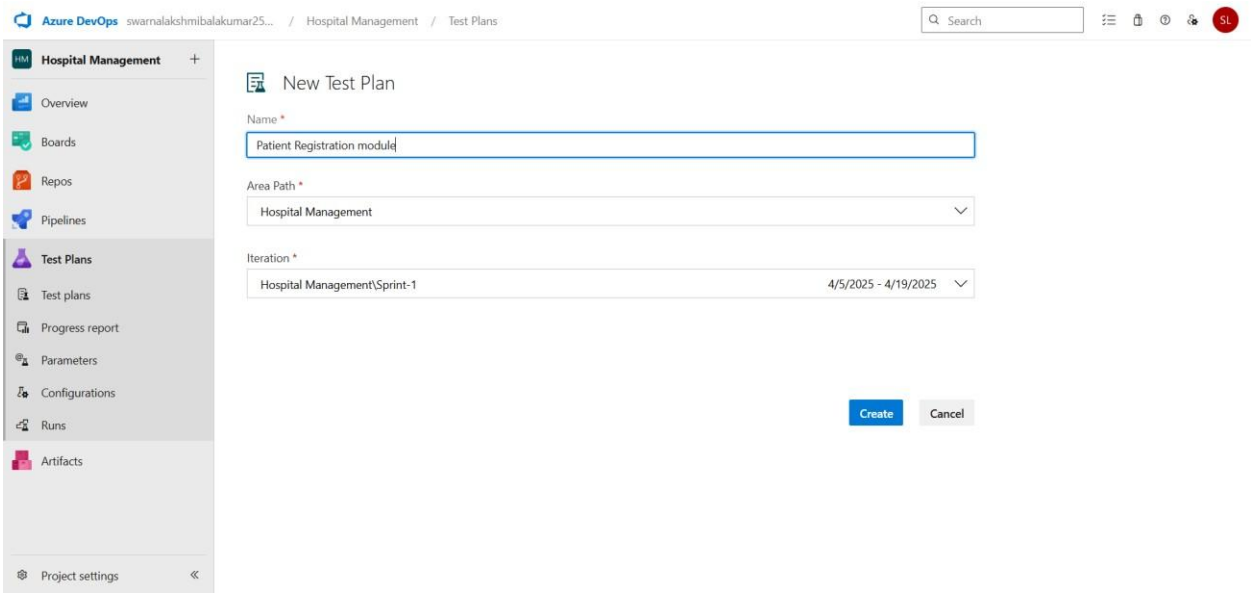
Suites are modular: Registration, Appointment, Billing, Availability, Prescription.

7. **Prioritize and Review**

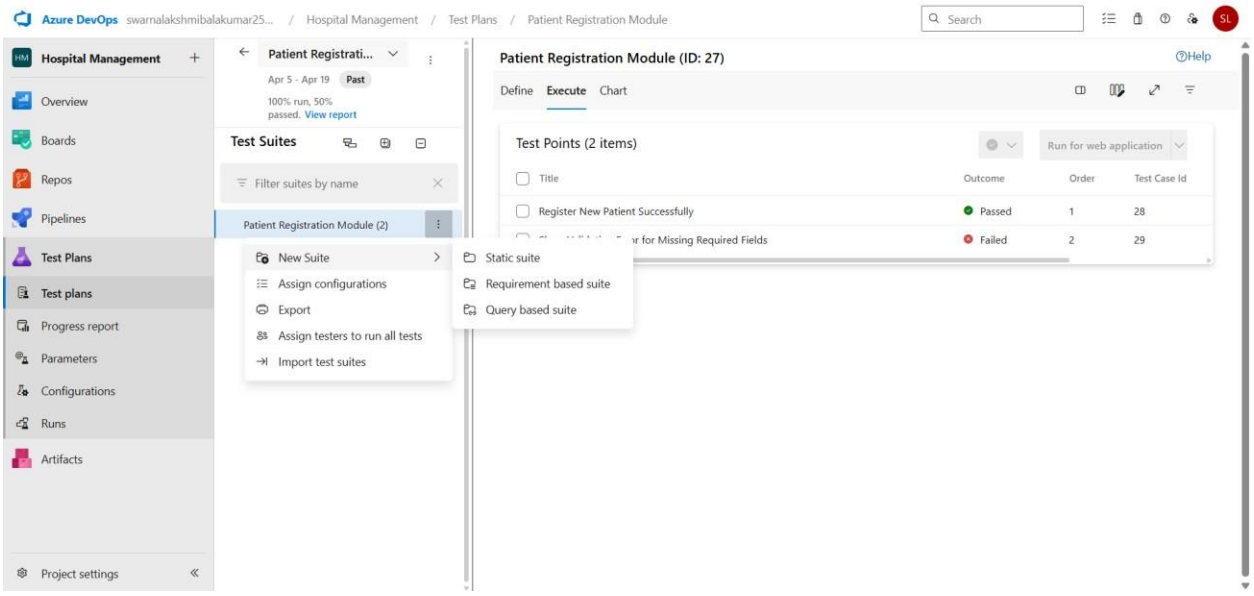
Happy path scenarios marked **High Priority**

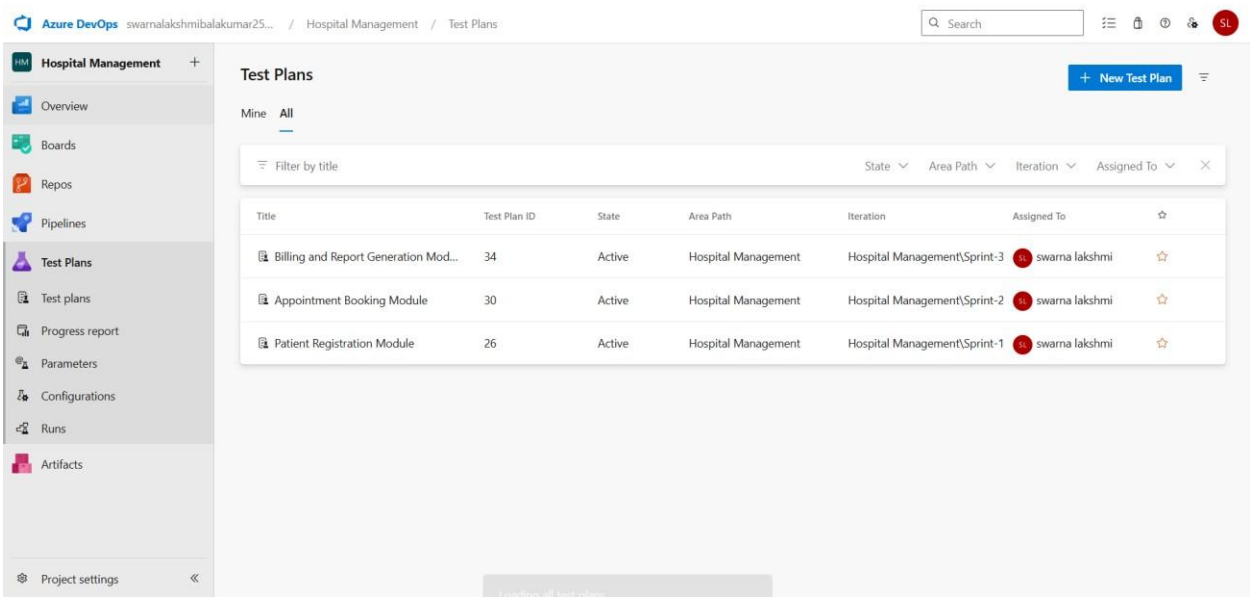
Test cases mapped to **linked user stories** in Azure DevOps.

1. New test plan



2. Test suite





3. Test case

Give two test cases for at least three user stories showcasing the happy path and error scenarios in azure DevOps platform.
Hospital Management System – Test Plans

USER STORIES

- As an admin, I want to register a new patient.
- As a patient, I want to book an appointment with a doctor.
- As a cashier, I want to generate bills after consultation.

Test Suites

Test Suite: TS01 - Patient Registration Module

1. **TC01 – Register New Patient Successfully**
    - **Action:**
        - Login as Admin to the Azure-hosted HMS portal.
        - Navigate to the “Register New Patient” form.
        - Fill in valid details: Name, Age, Gender, Contact, Medical History.
        - Click "Save".
    - **Expected Results:**
        - A success message is shown: "Patient Registered Successfully".
    - **Type**: Happy Path
2. **TC02 – Show Validation Error for Missing Required Fields.**
    - **Action:**
        - Open the registration form.
        - Leave required fields (e.g., Name, Contact) blank

- Click "Save".

- **Expected Results:**
  - Form displays message: "Name and Contact are required."
- **Type:** Error Path

Test Suite: TS02 - Appointment Booking Module

1. **TC03 – Book Appointment Successfully**
   - **Action:**
     - Login to the Azure-hosted patient portal.
     - Navigate to "Book Appointment".
     - Select Doctor, Date, and Available Time Slot.
     - Click "Confirm".
   - **Expected Results:**
     - Confirmation message displayed: "Appointment Confirmed".
   - **Type:** Happy Path
2. **TC04– Prevent Double Booking** ○

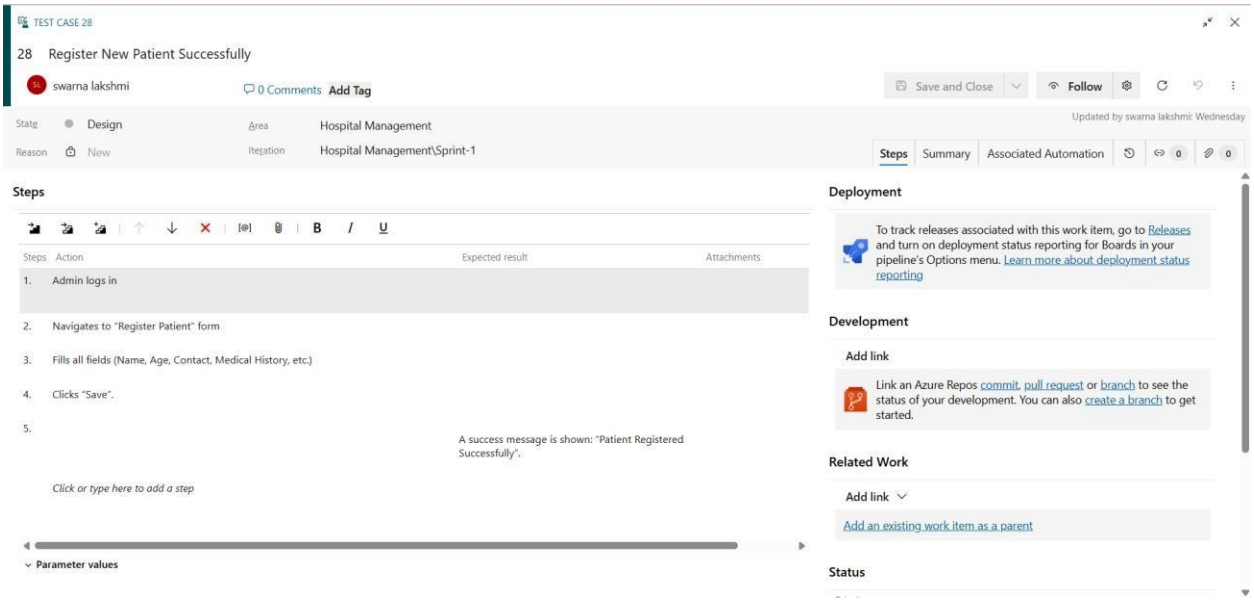
   Action:

   - Patient A books Dr. Smith at 10:00 AM.
   - Patient B selects the same doctor and time slot.
   - Clicks "Book".
   - **Expected Results:**
     - System returns error: "Time slot already booked."
   - **Type:** Error Path

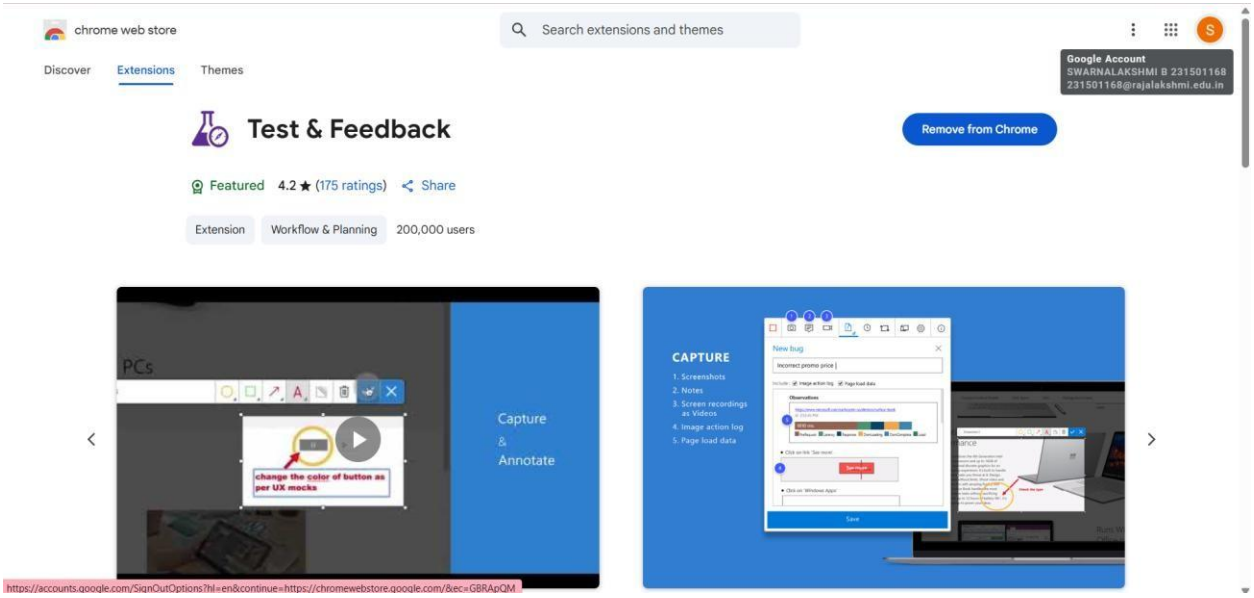
Test Suite: TS03 - Billing and Report Generation

1. **TC05 – Generate Consultation Bill**
   - **Action:**
     - Login as Cashier.
     - Select a completed consultation
     - Enter service fees, medicine charges, and lab test fees.
     - Click "Generate Bill"
   - **Expected Results:**
     - Confirmation message and downloadable PDF bill are displayed.
   - **Type:** Happy Path

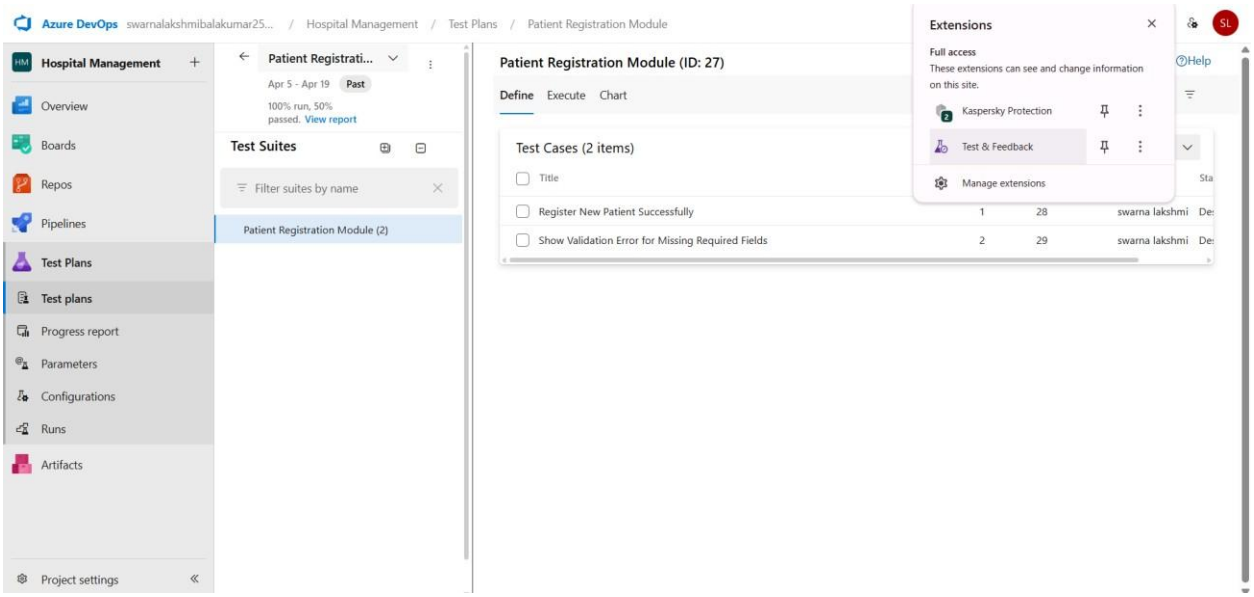
Test Cases



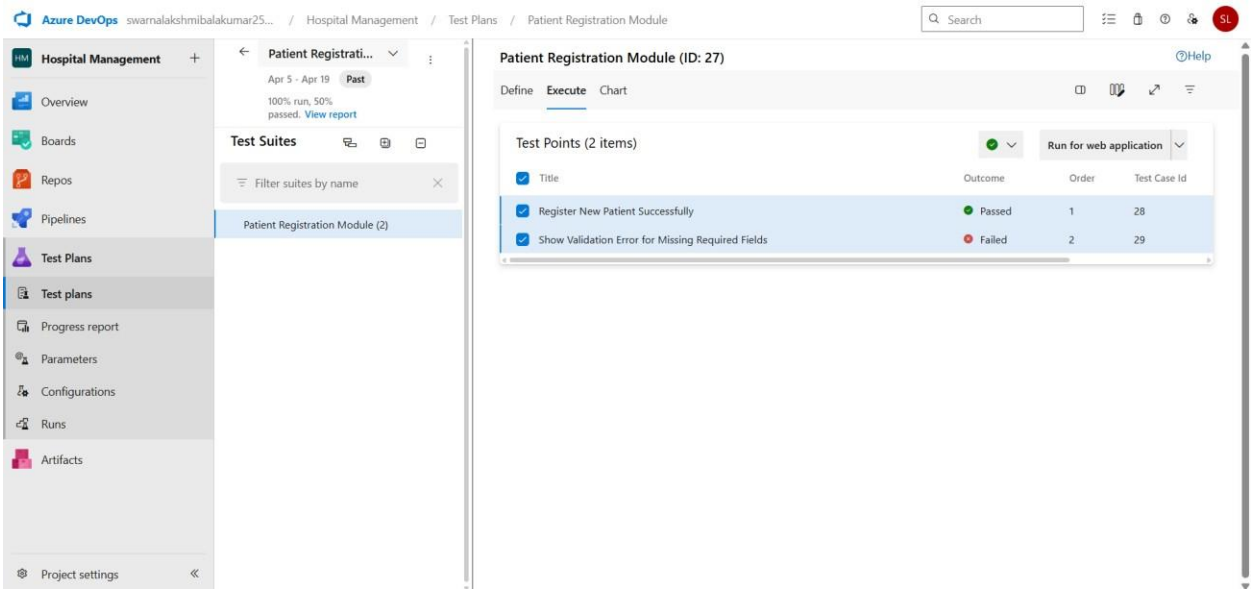
4. Installation of test

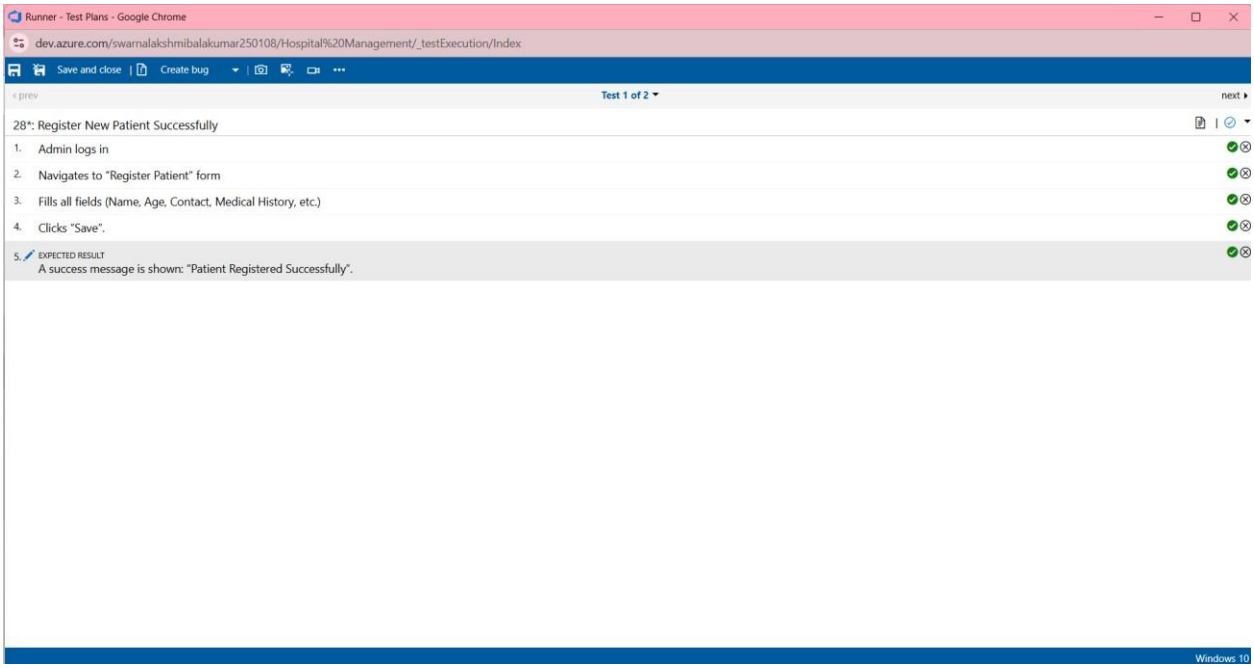


Test and feedback
Showing it as an extension

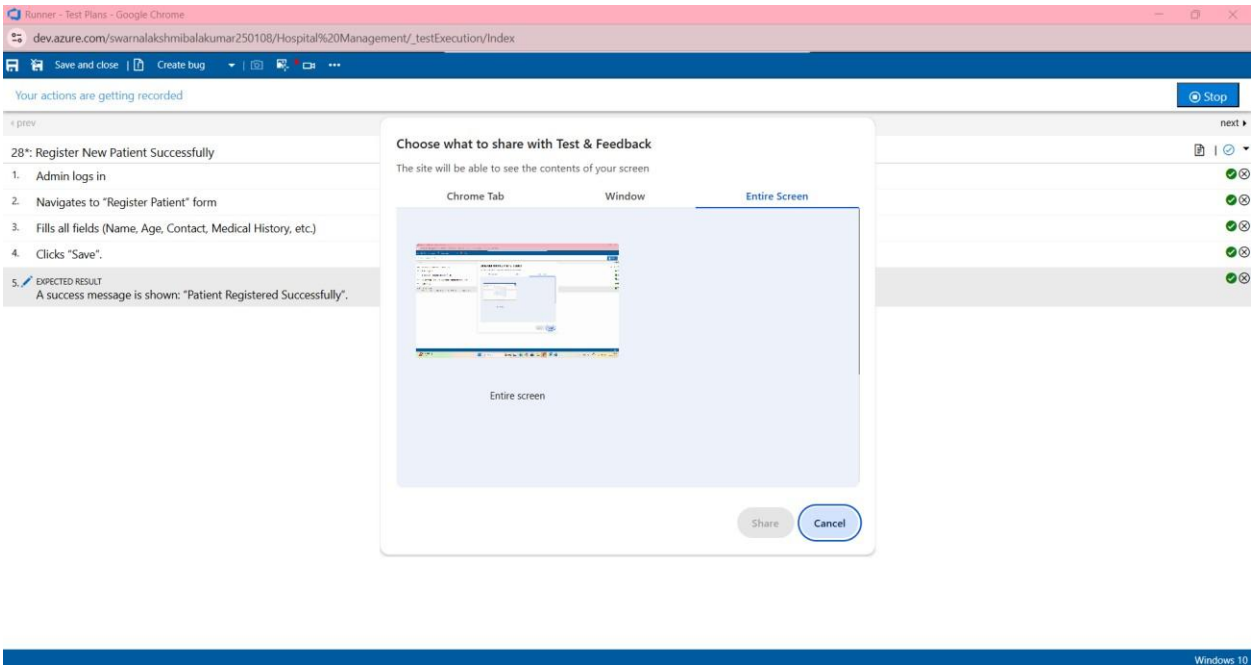


5. Running the test cases

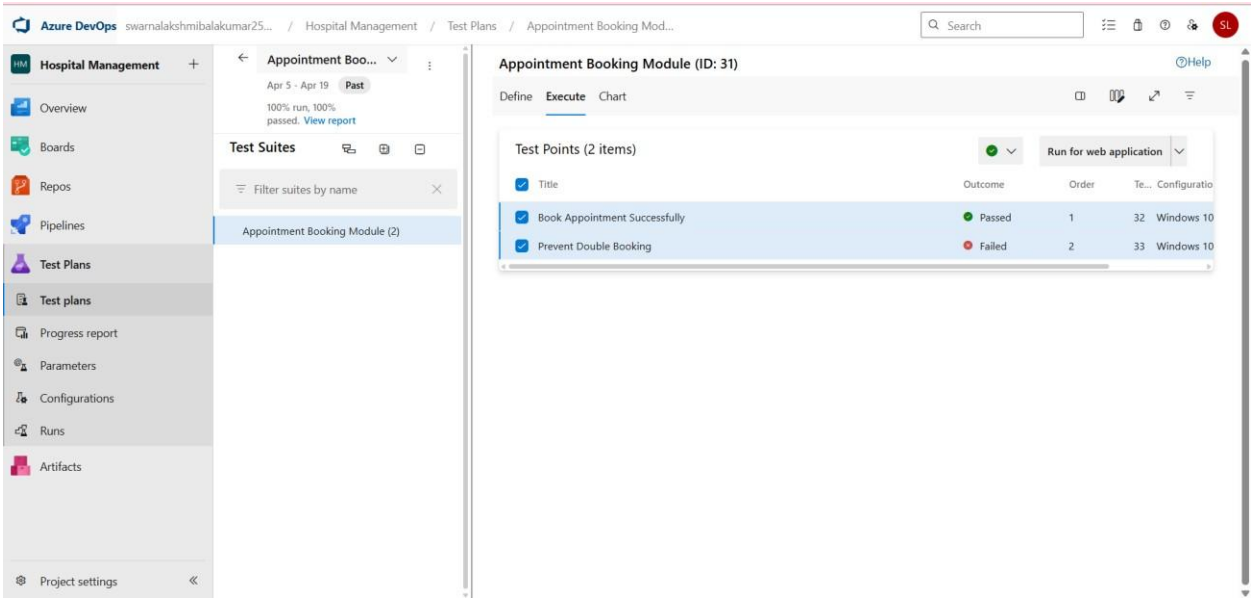




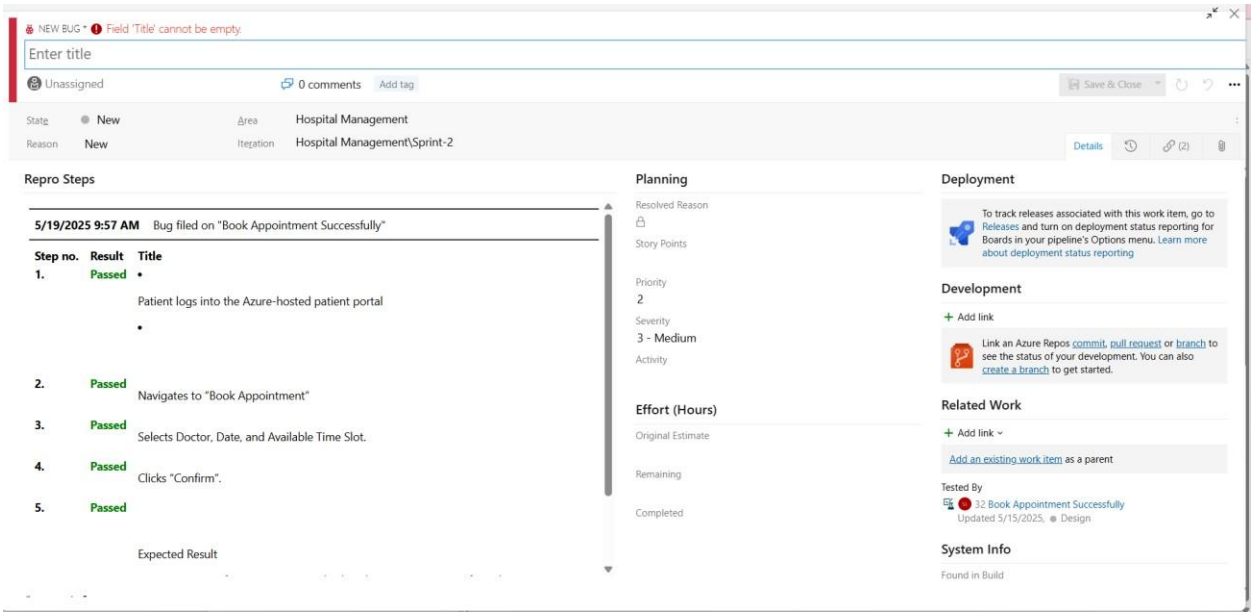
6. Recording the test case



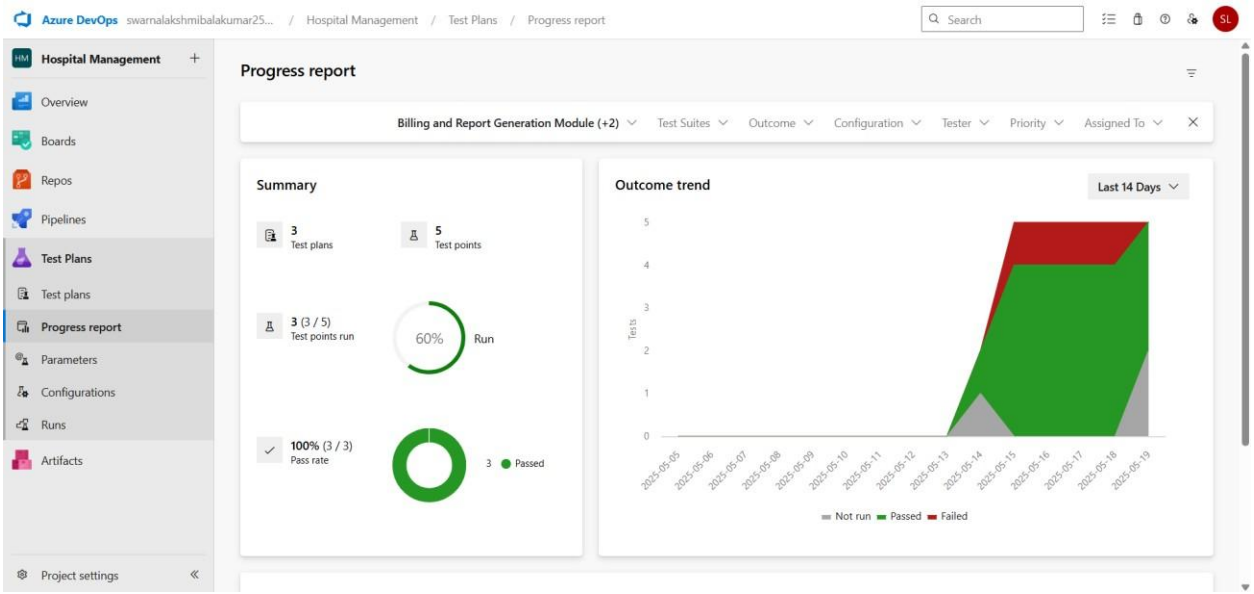
7. Test case results



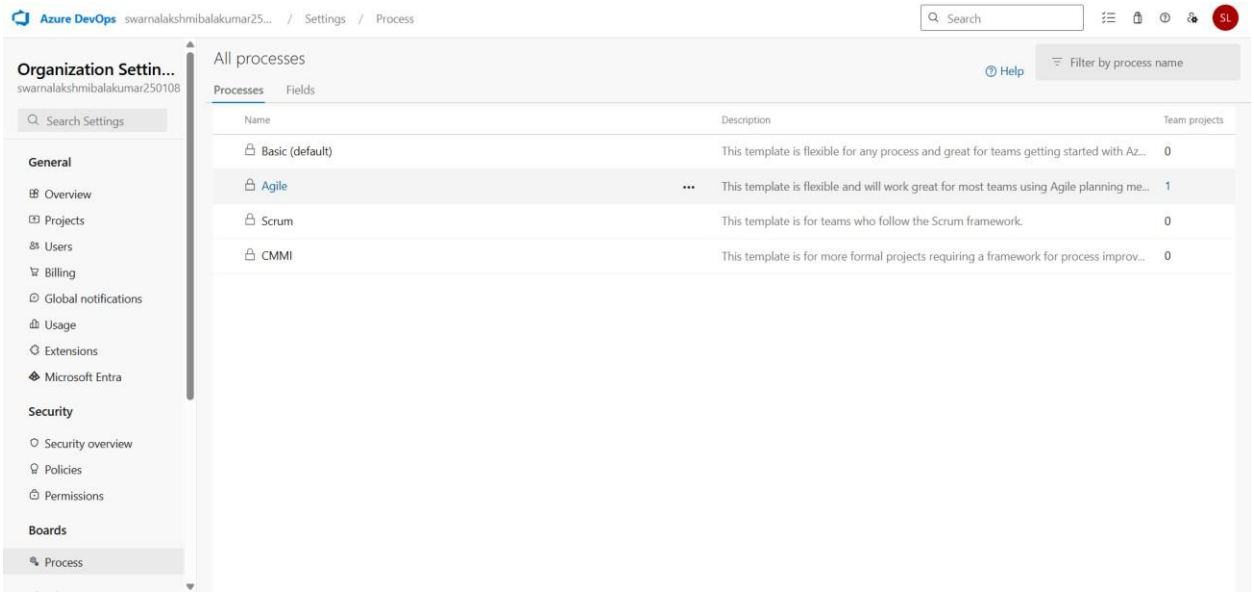
8.Test report summary



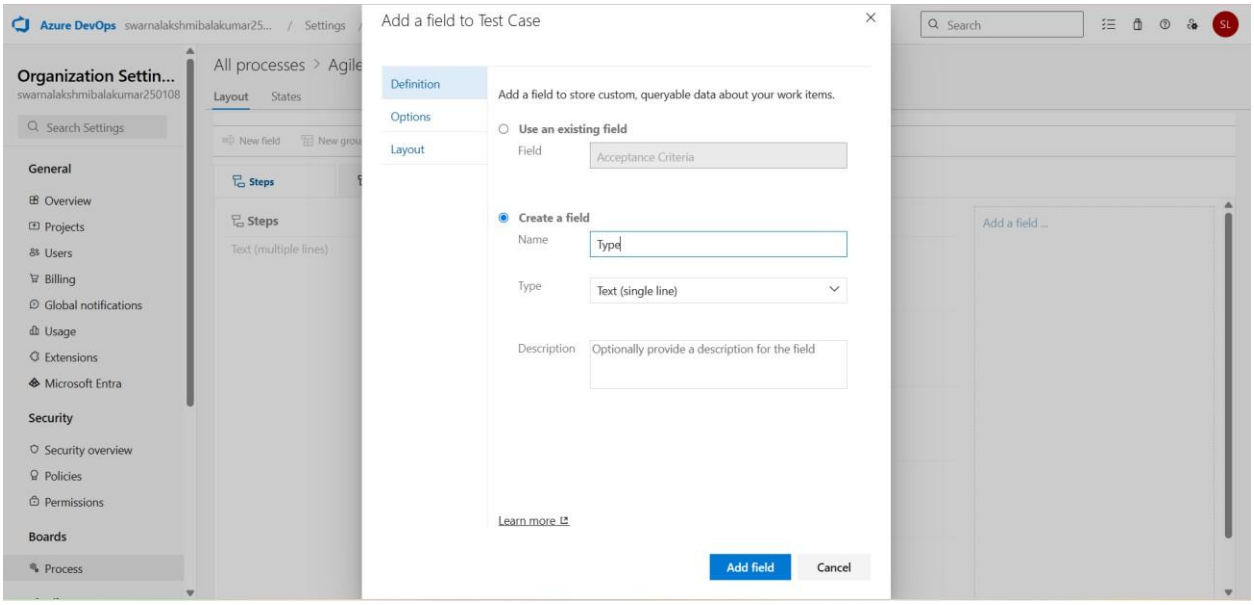
9.Progress report

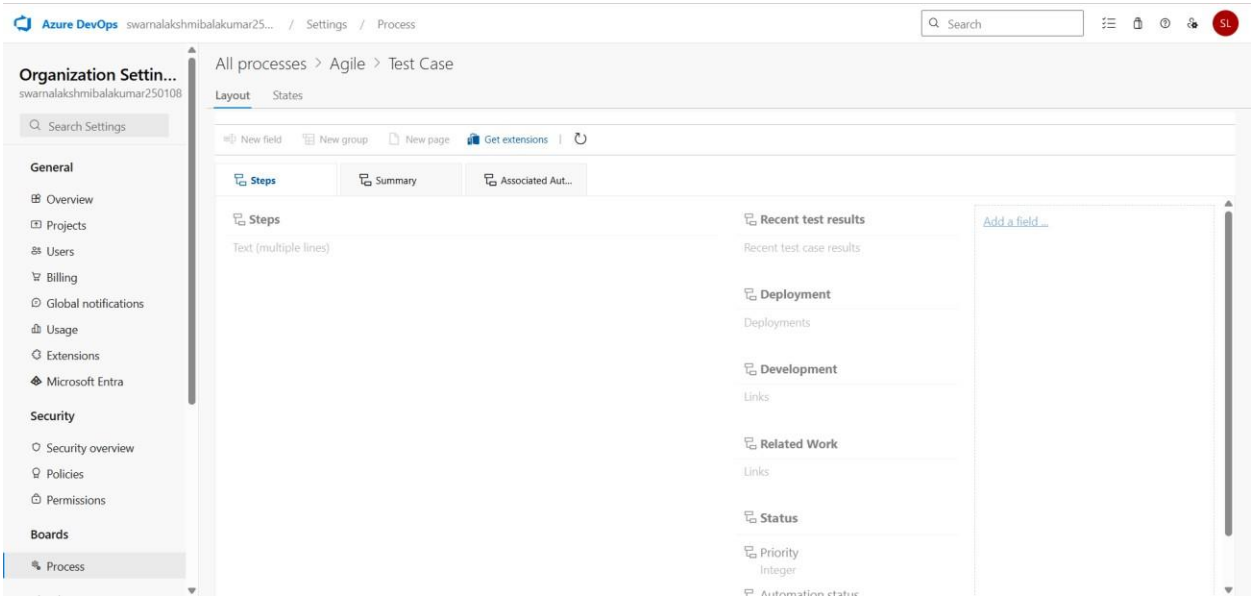


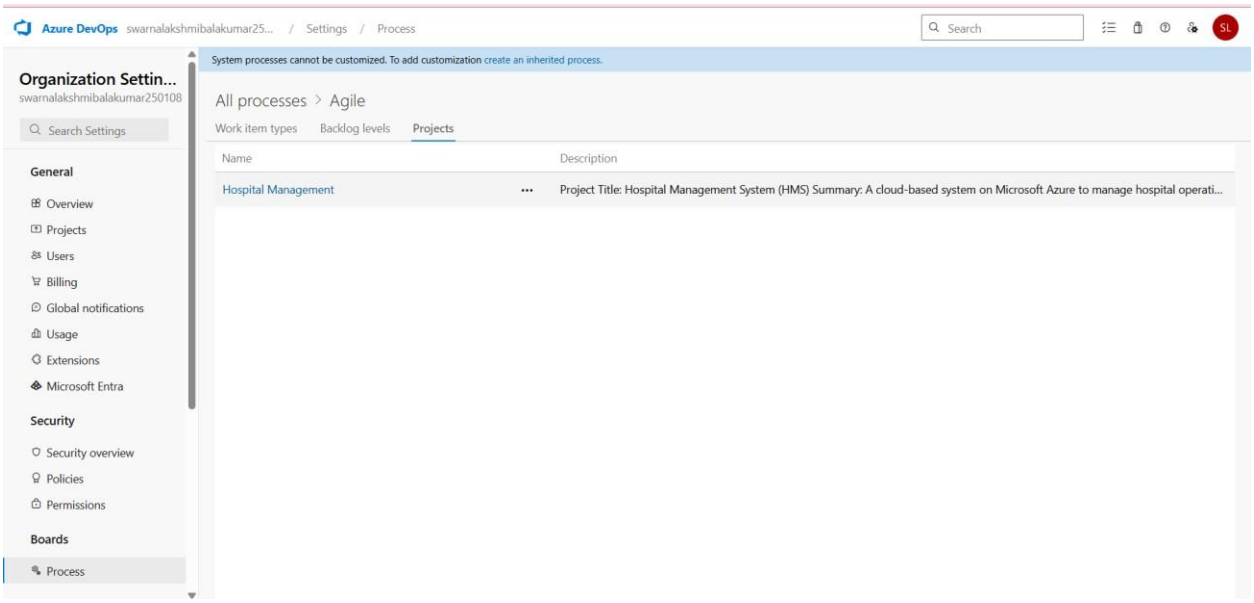
10.Changing the test template



11.View the new test case template







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 Hospital Management System, 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, select your organization, and open the project where your Hospital 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 the 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)

1. 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 the task linked to your subscription and app details.

13. Set Triggers and Approvals

    Enable a 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.

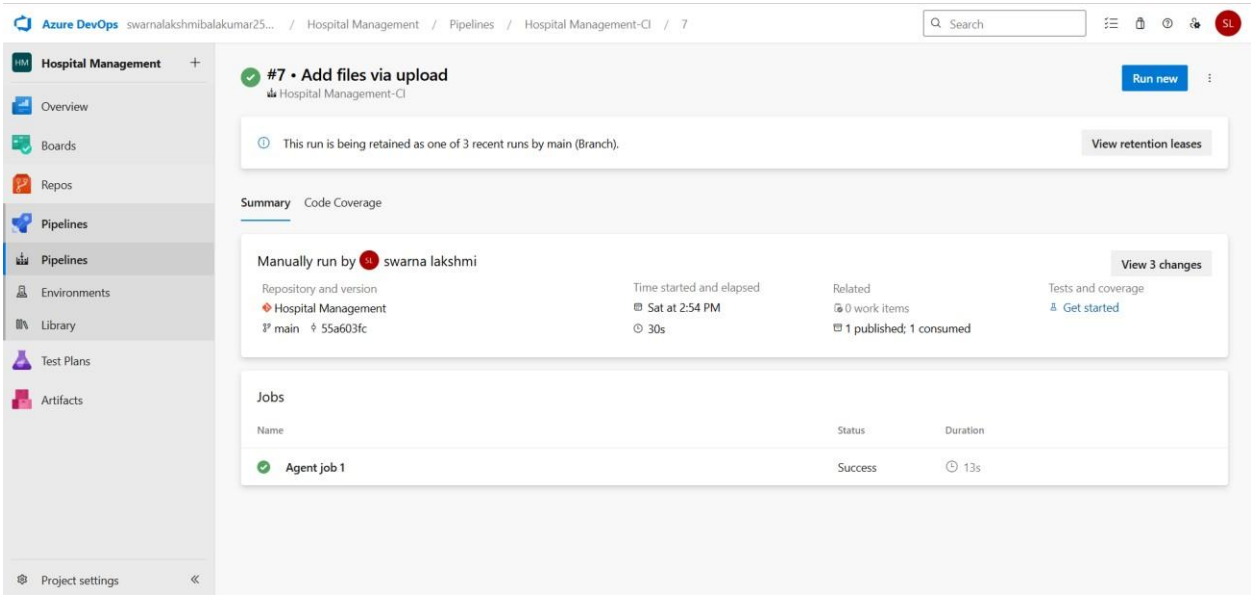
14. Monitor Pipelines and Manage Logs

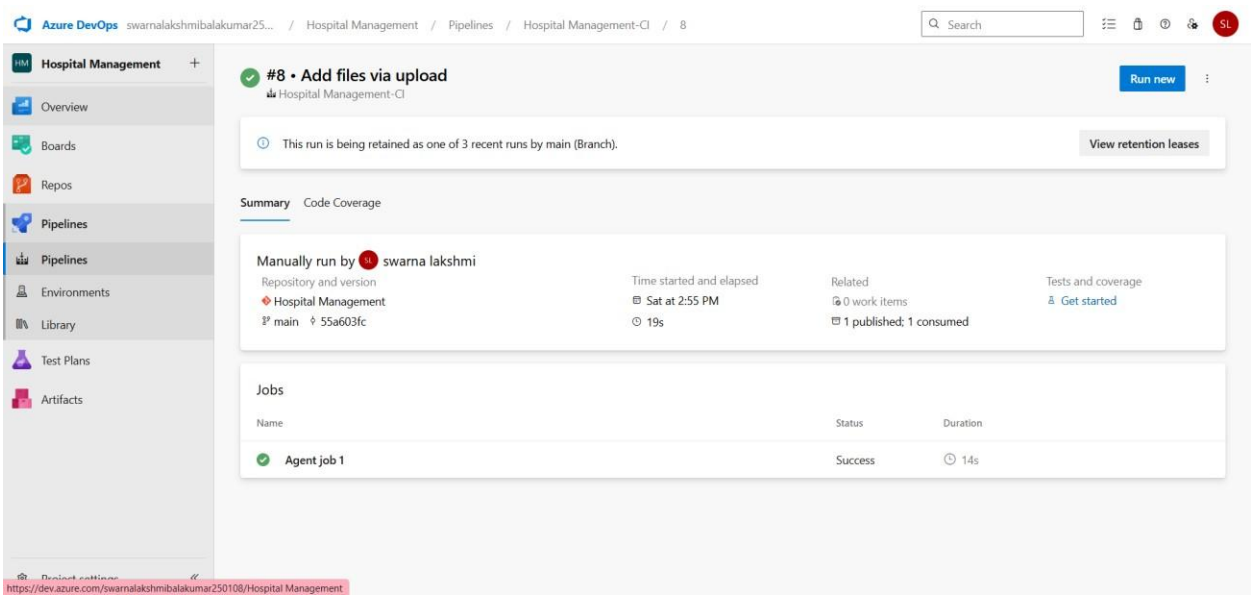
    View all pipeline runs under the Runs section.
    Check logs for build/test/deploy stages to debug any errors.
    You can also integrate email alerts or Microsoft Teams notifications for build failures.

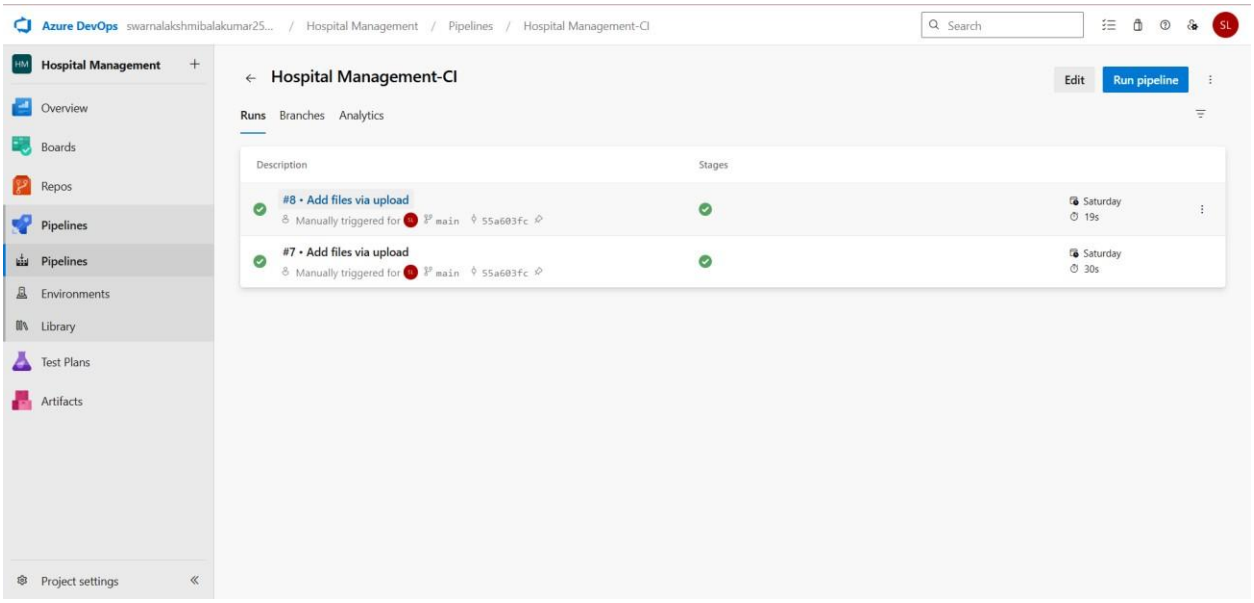
15. Review and Maintain Pipelines

    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.

Pipeline


Azure DevOps swarnalakshmibalakumar25... / Hospital Management / Pipelines / Hospital Management-CI / 7
Search
Hospital Management
Overview
Boards
Repos
Pipelines
Pipelines
Environments
Library
Test Plans
Artifacts
Project settings
#7 • Add files via upload
Hospital Management-CI
Run new
This run is being retained as one of 3 recent runs by main (Branch).
View retention leases
Summary
Code Coverage
Manually run by swarna lakshmi
View 3 changes
Repository and version
Hospital Management
main
55a603fc
Time started and elapsed
Sat at 2:54 PM
30s
Related
0 work items
1 published; 1 consumed
Tests and coverage
Get started
Jobs
Name
Status
Duration
Agent job 1
Success
13s


Azure DevOps swarnalakshmibalakumar25... / Hospital Management / Pipelines / Hospital Management-CI / 8
Search
Hospital Management
Overview
Boards
Repos
Pipelines
Pipelines
Environments
Library
Test Plans
Artifacts
#8 • Add files via upload
Hospital Management-CI
Run new
This run is being retained as one of 3 recent runs by main (Branch).
View retention leases
Summary
Code Coverage
Manually run by swarna lakshmi
Repository and version
Hospital Management
main
55a603fc
Time started and elapsed
Sat at 2:55 PM
19s
Related
0 work items
1 published; 1 consumed
Tests and coverage
Get started
Jobs
Name
Status
Duration
Agent job 1
Success
14s
https://dev.azure.com/swarnalakshmibalakumar250108/Hospital Management



Result:

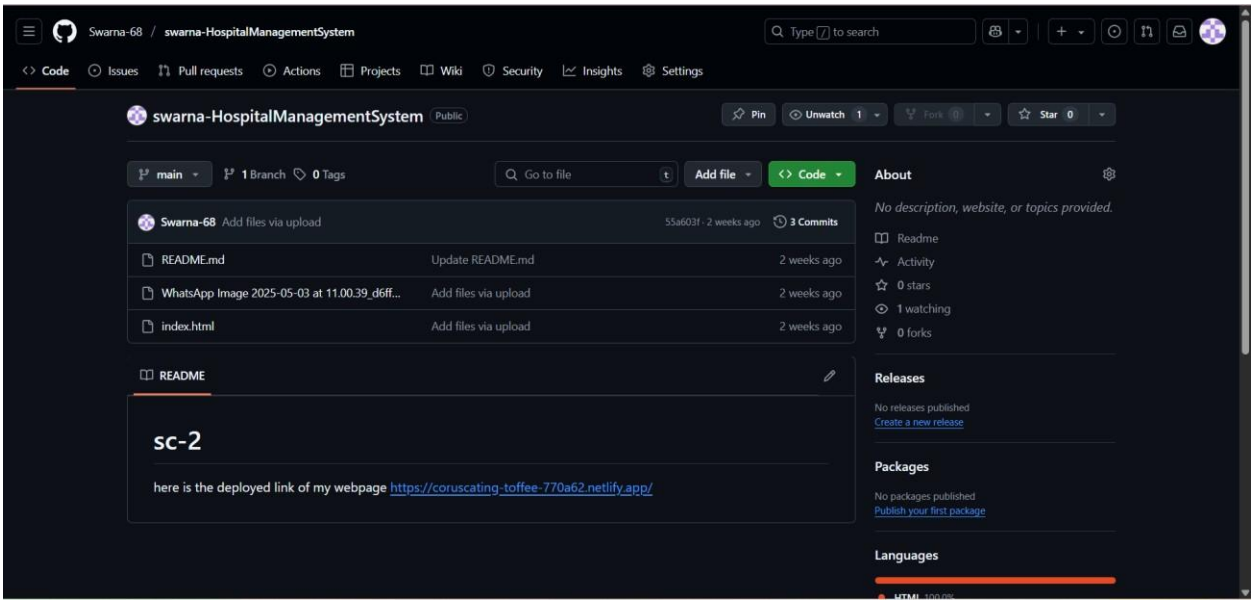
Thus, the pipelines for the given project, “Hospital Management System” has been executed successfully

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 project- Hospital Management System.

GitHub Project Structure



Result:

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