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:

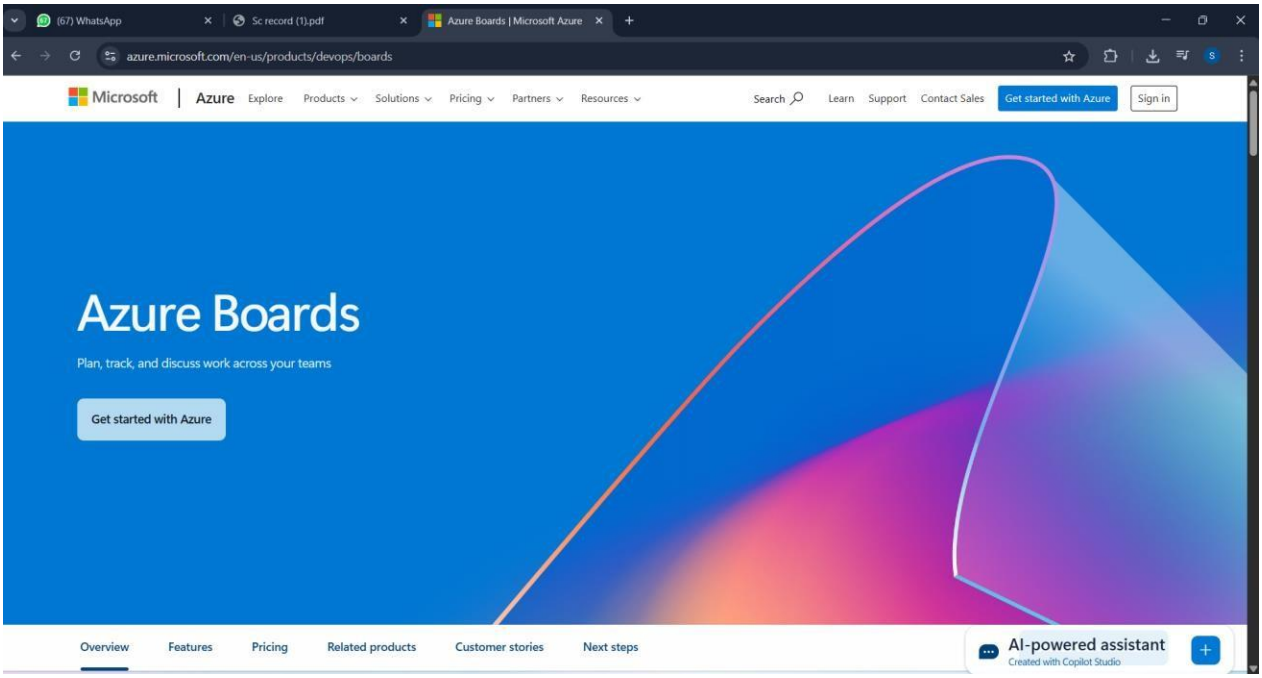
To set up and access the Azure DevOps environment by creating an organization through the Azure portal.

INSTALLATION

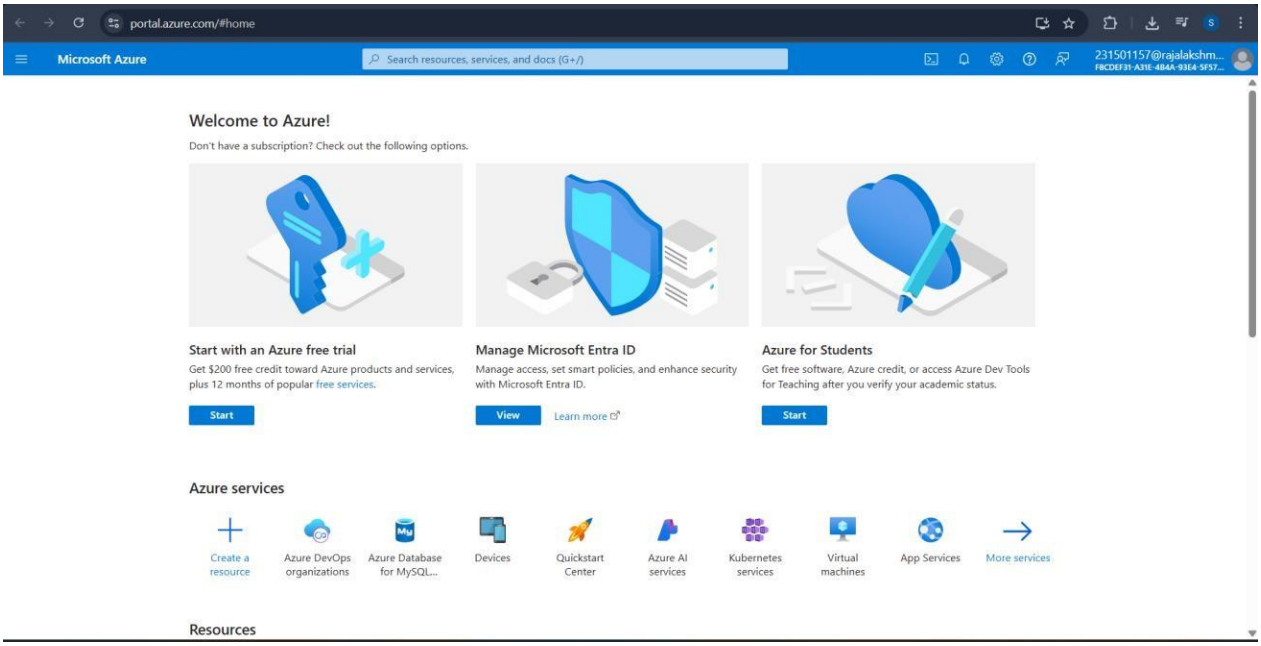
1. Open your web browser and go to the Azure website: https://azure.microsoft.com/en-us/getstarted/azure-portal.

Sign in using your Microsoft account credentials.

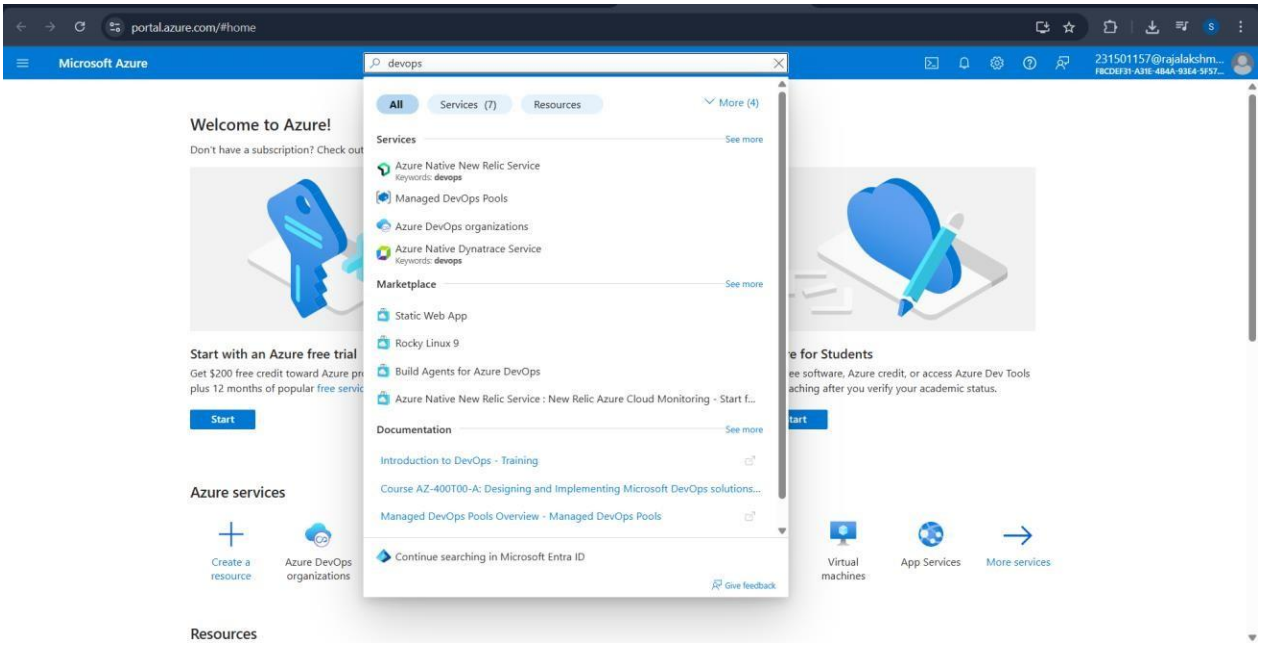
If you don't have a Microsoft account, you can create one here: https://signup.live.com/?lic=1



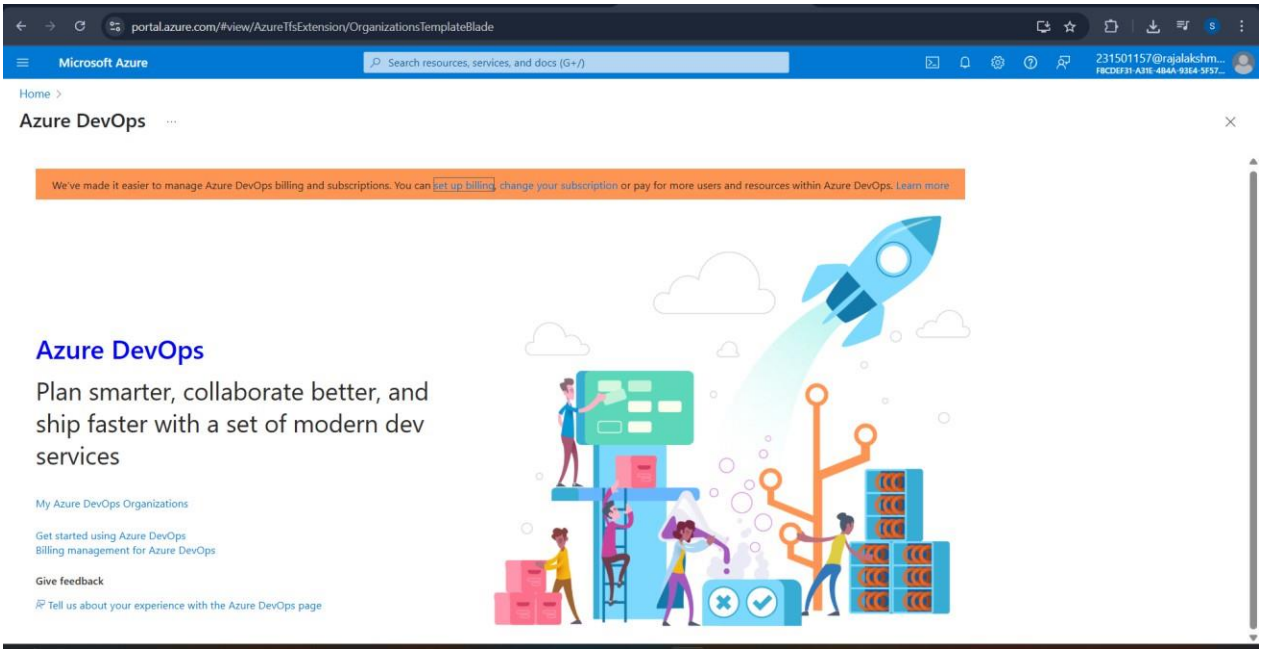
2. Azure home page



3. Open 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:

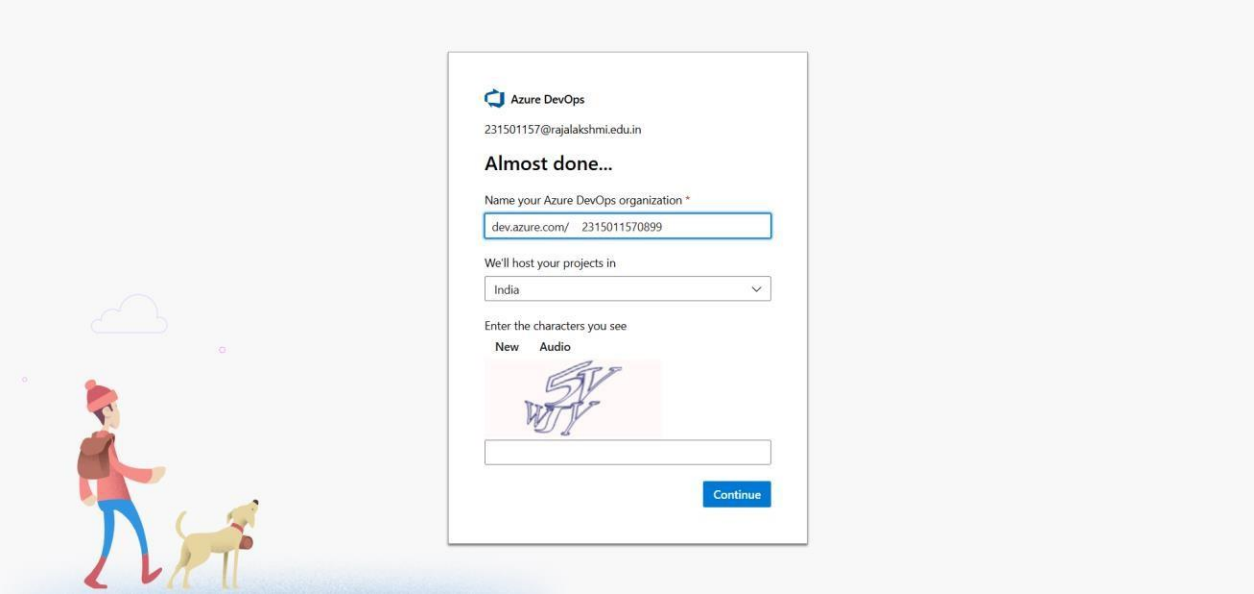
Successfully accessed the Azure DevOps environment and created a new organization through the Azure portal.

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.

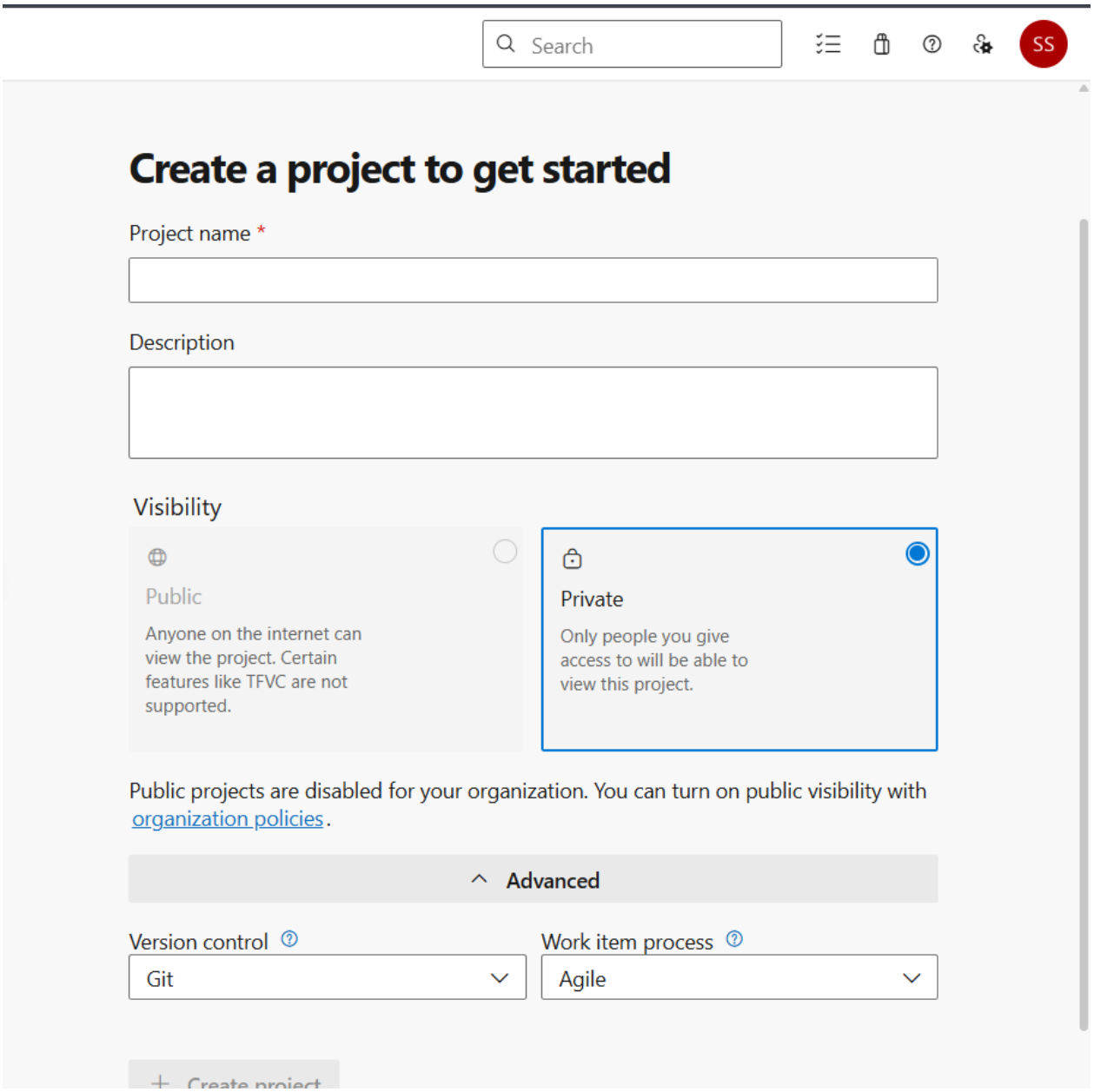
c. Enter the project name, description, and visibility options:

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

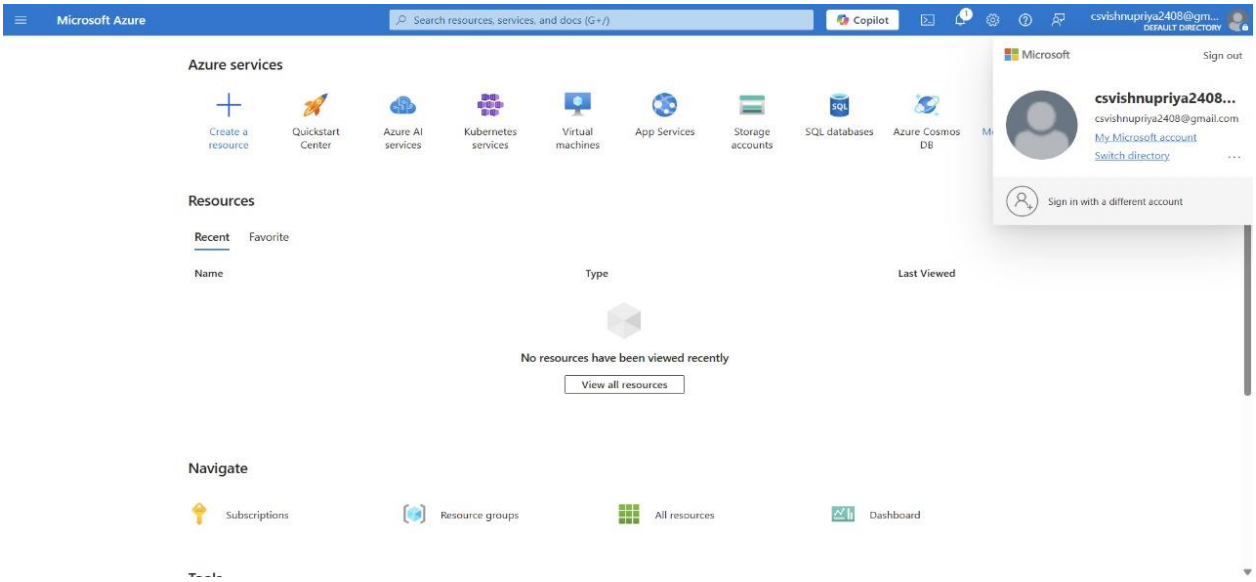
Description*:* Optionally, add a description to provide more context about the project.

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

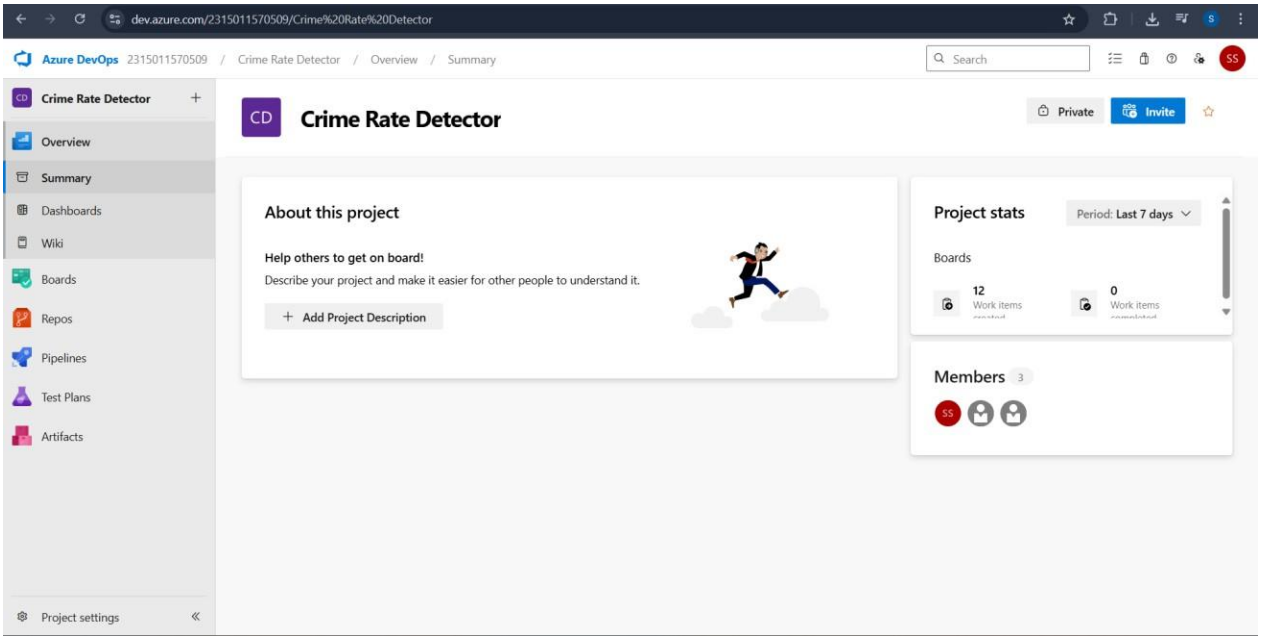
d. Once you've filled out the details, click **Create** to set up your first project.



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.



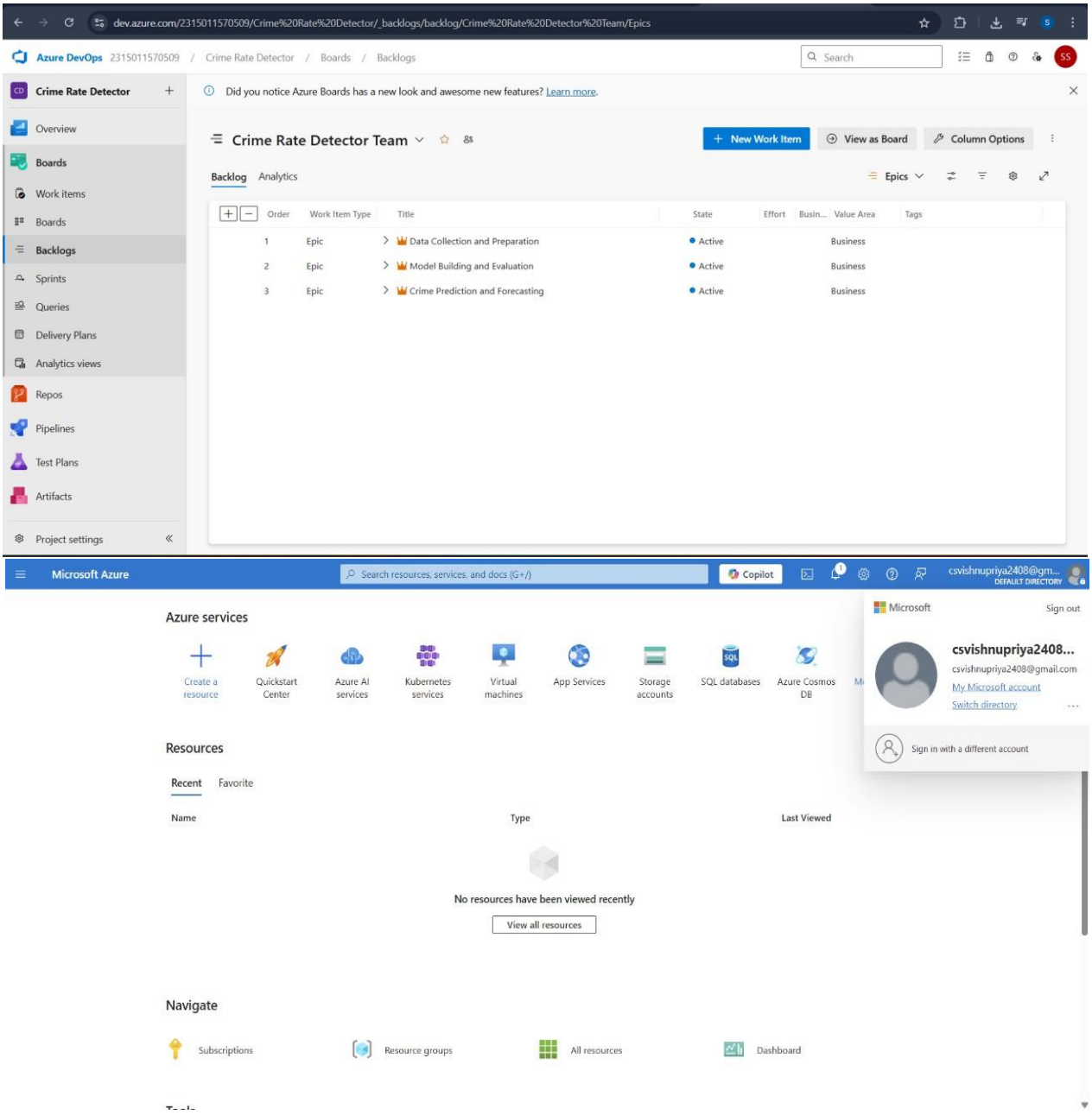
4. Project dashboard



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:

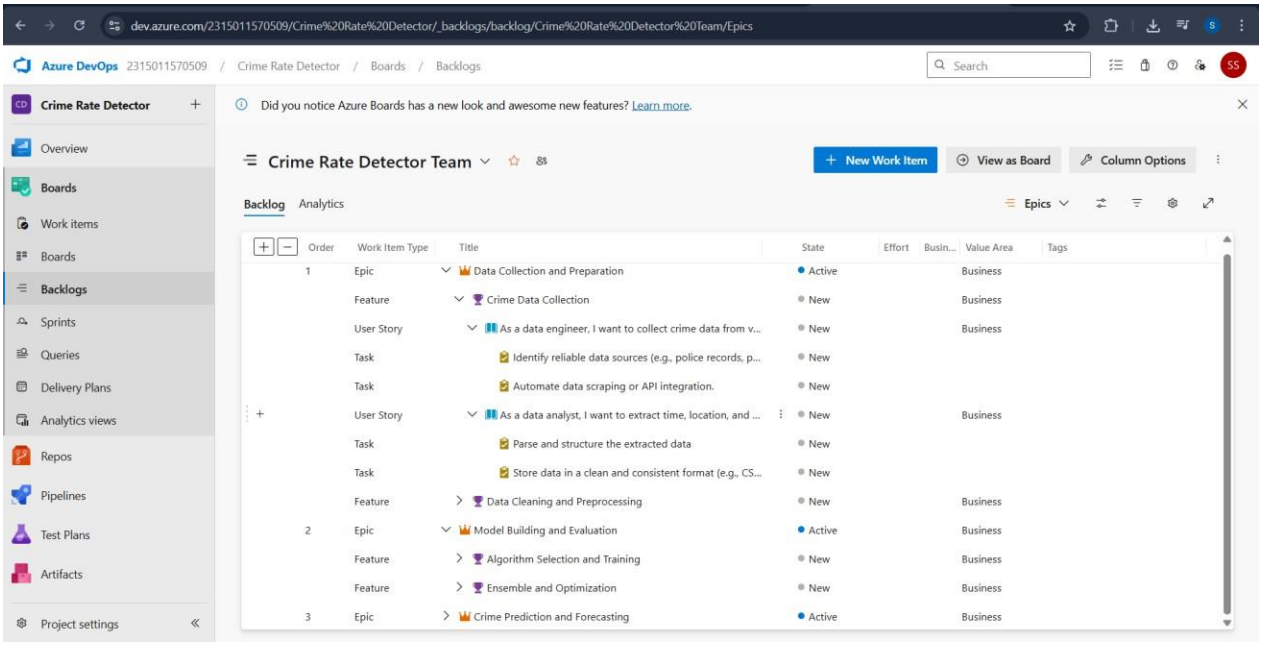
Successfully created an Azure DevOps project with user story management and agile workflow setup.

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

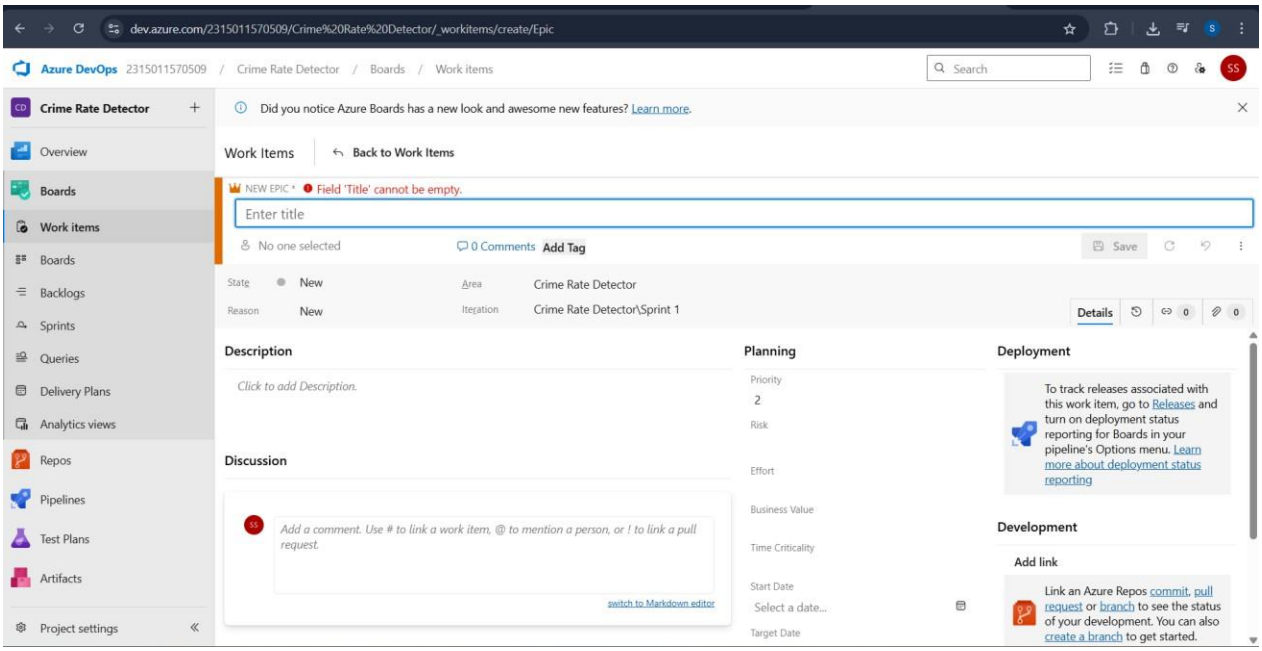
Aim:

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

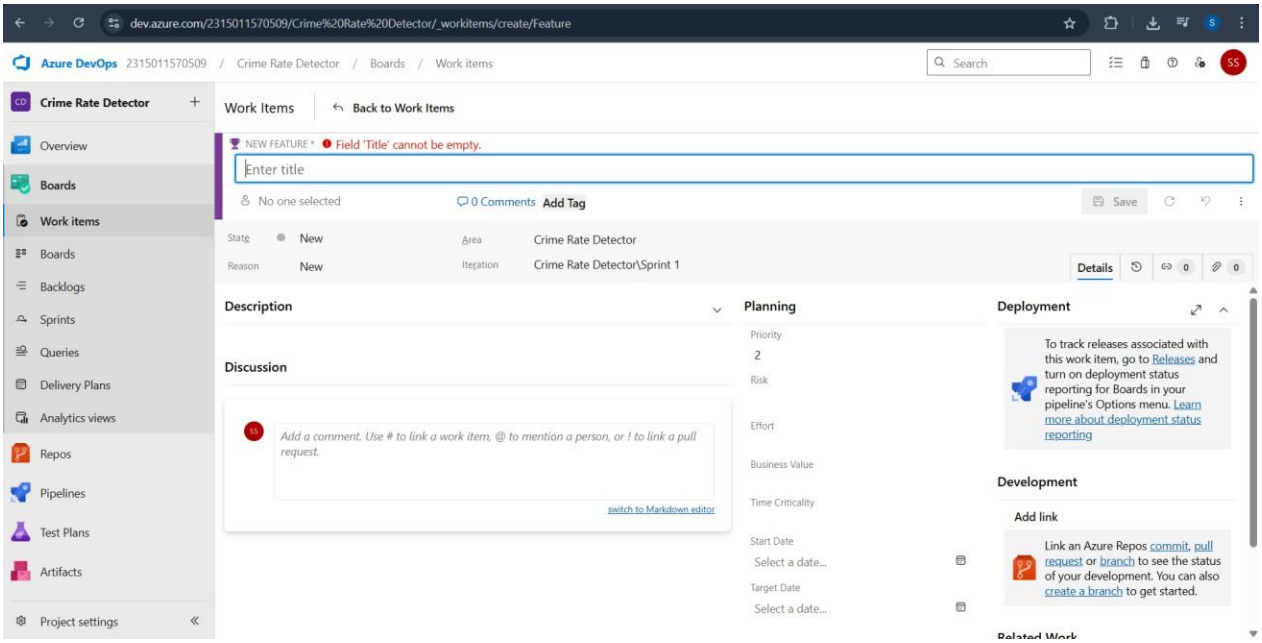
Create Epic, Features, User Stories, Task



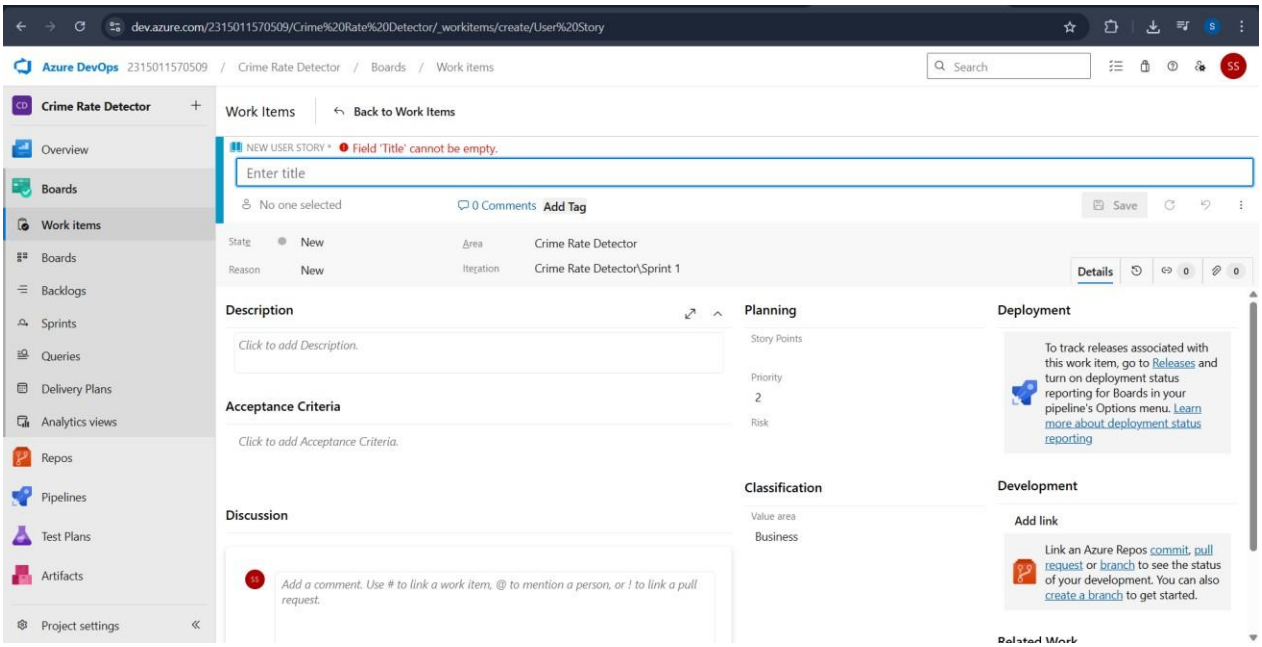
1. Fill in Epics



2. Fill in Features



3. Fill in User Story Details



Result:

Thus, the creation of epics, features, user story and task has been created successfully.

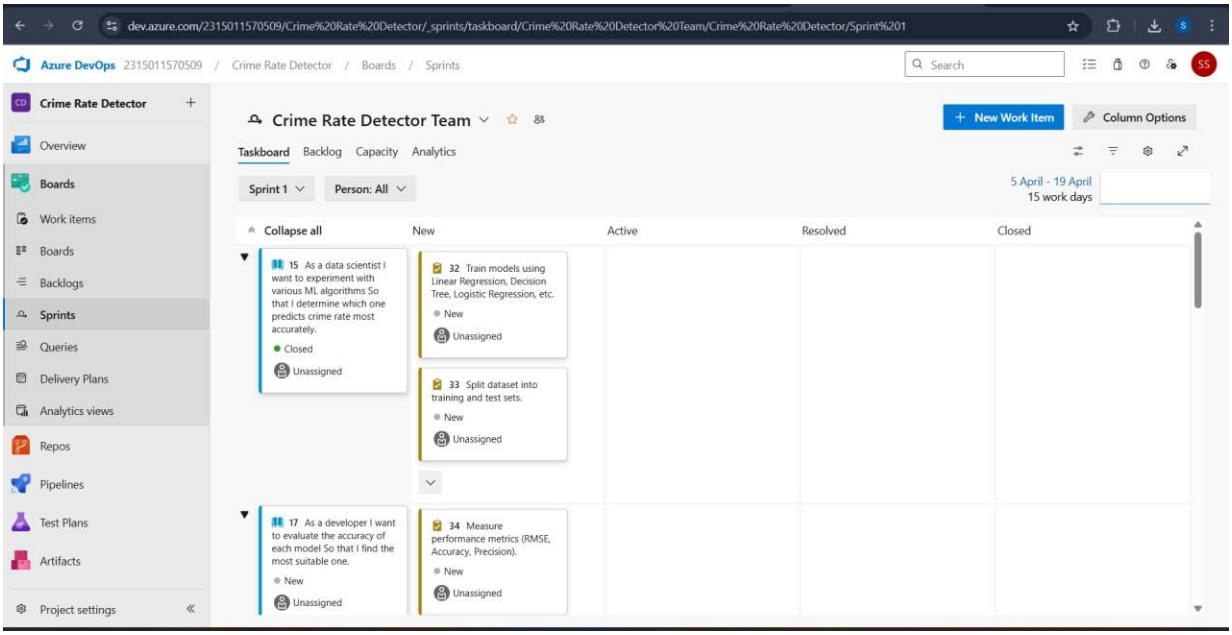
EXP NO: 4	SPRINT PLANNING

Aim:

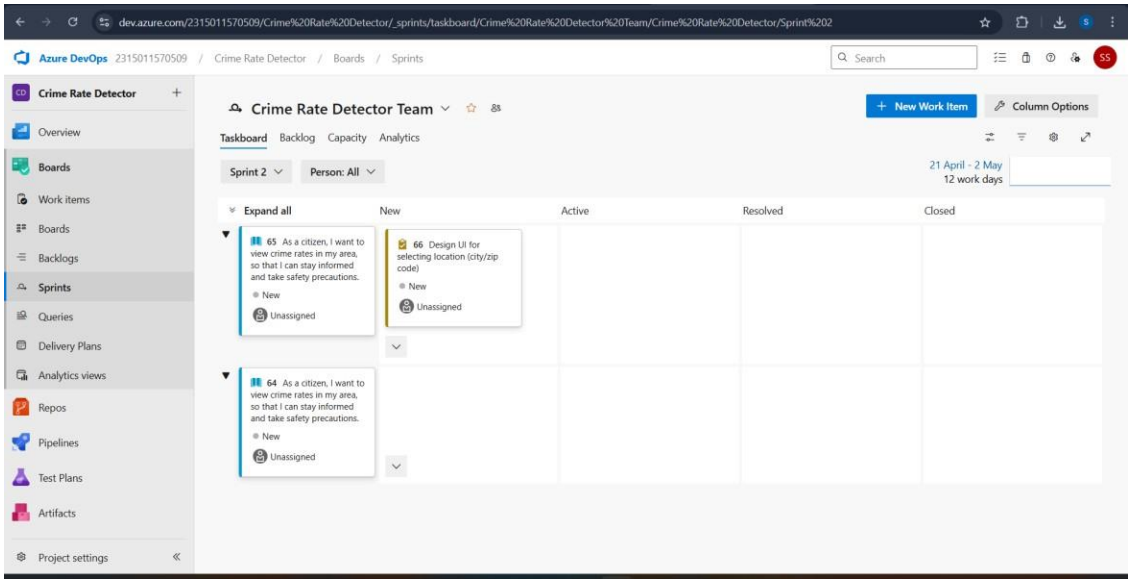
To assign user story to specific sprint for the Crime Rate Detector App Project.

Sprint Planning

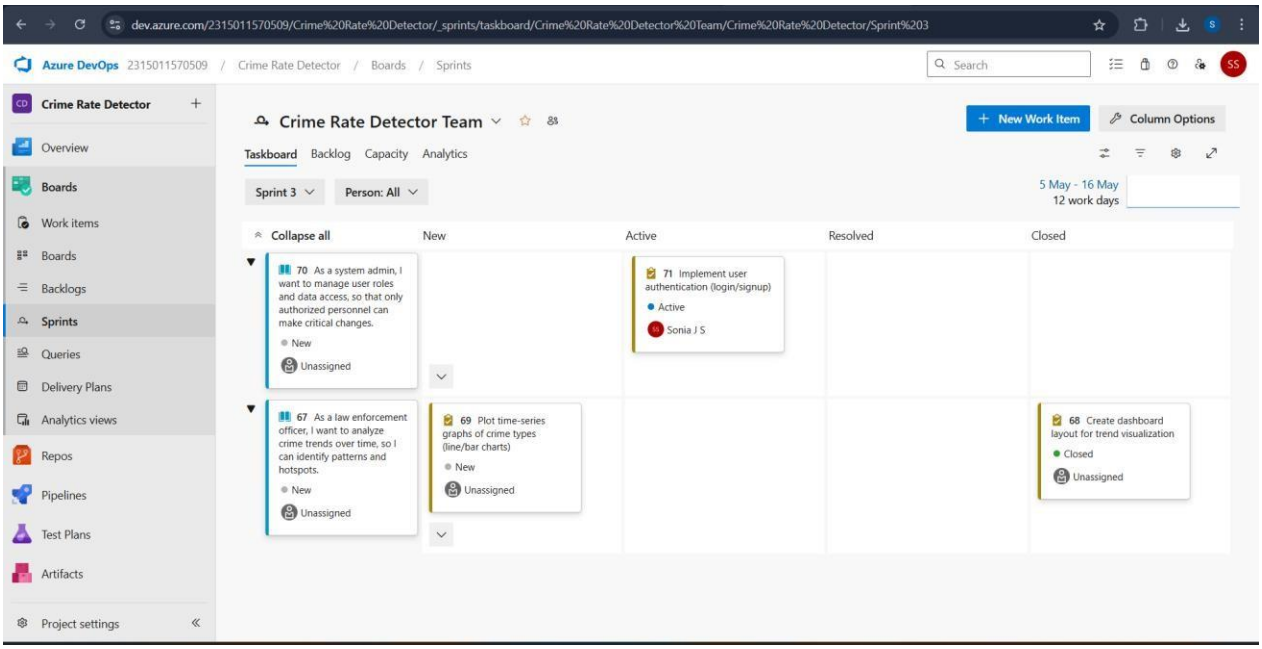
Sprint 1



Sprint 2



Sprint 3



Result:

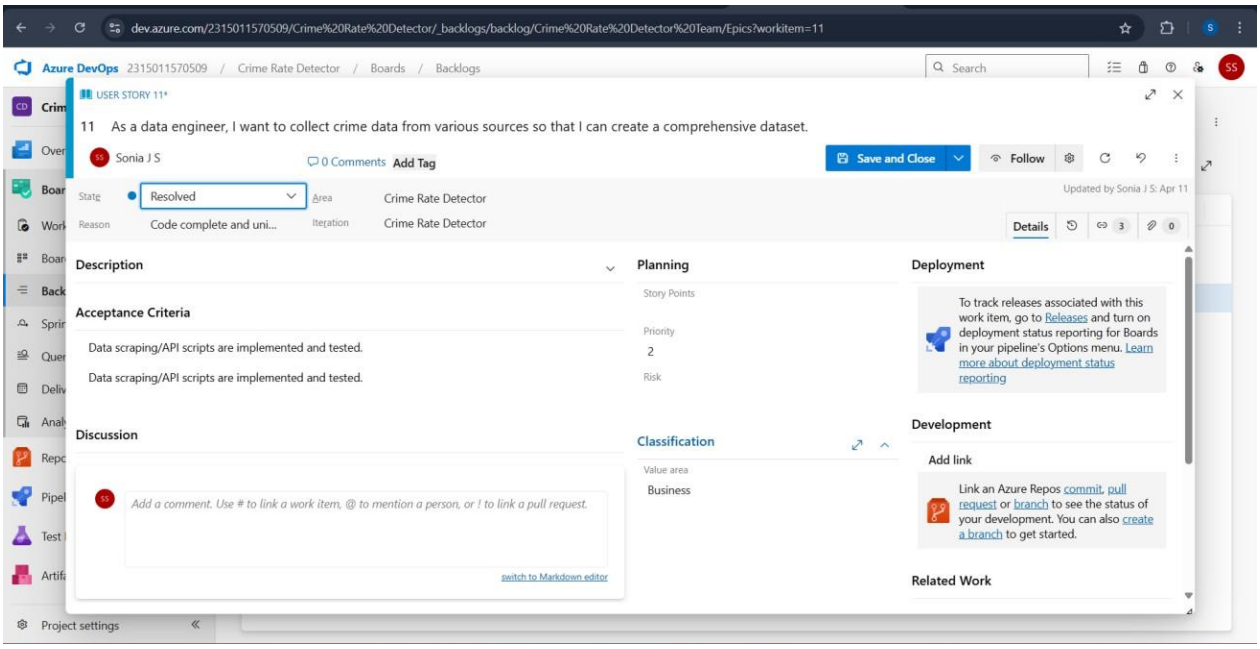
The Sprints are created for the Crime Rate Detector App Project.

EXP NO: 5	POKER ESTIMATION

Aim:

Create Poker Estimation for the user stories - Crime Rate Detector App Project.

Poker Estimation



Result:

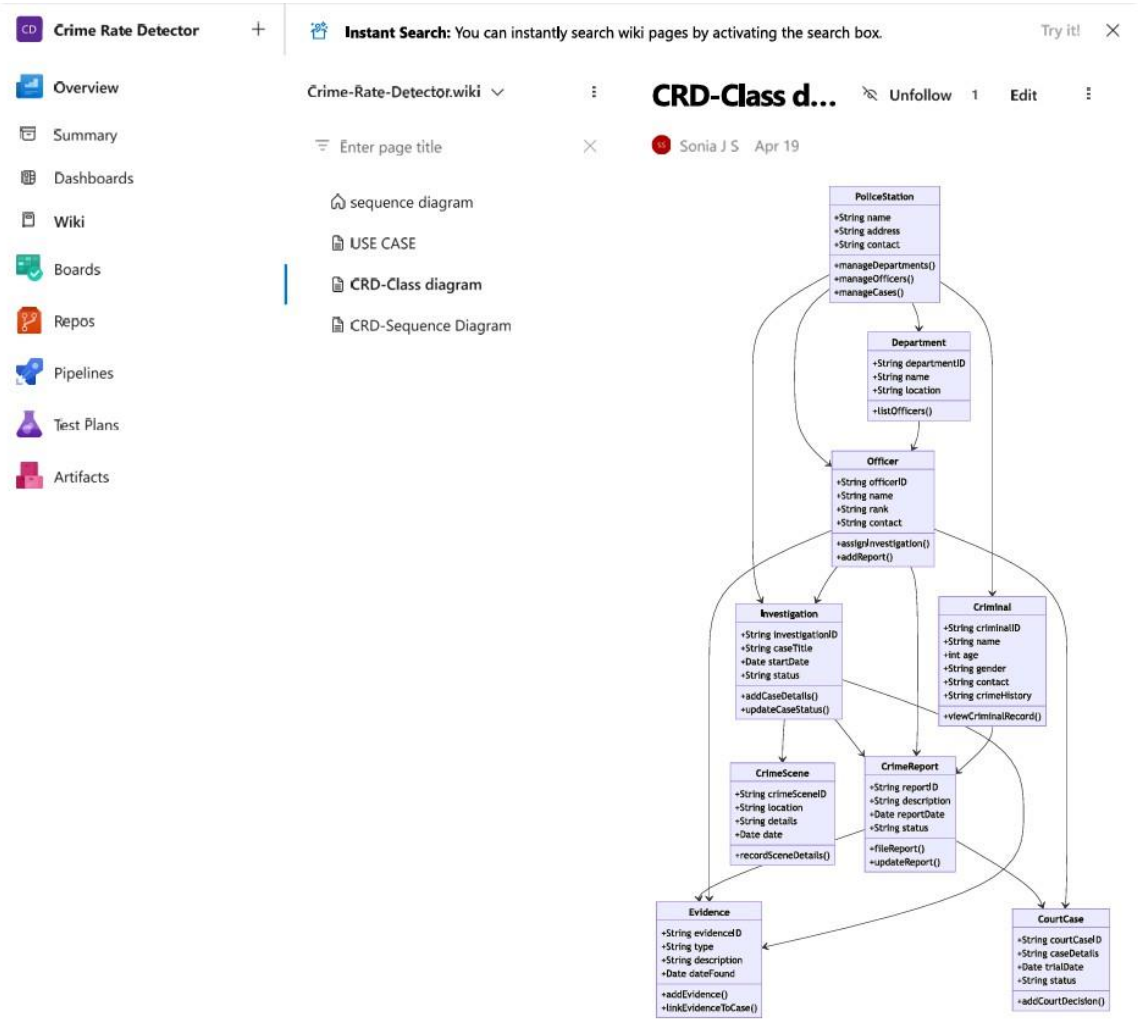
The Estimation/Story Points is created for the project using Poker Estimation.

EXP NO: 6	DESIGNING CLASS AND SEQUENCE DIAGRAMS FOR PROJECT ARCHITECTURE

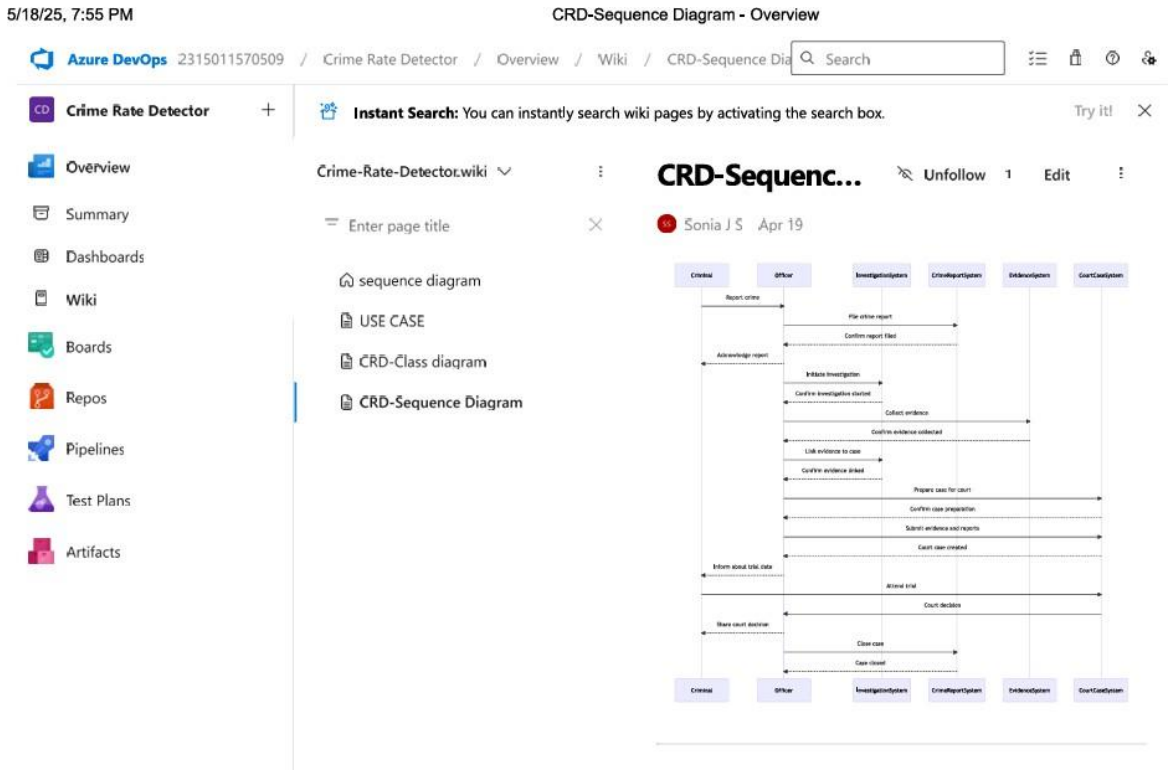
Aim:

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

6A. Class Diagram



6B. Sequence Diagram



Result:

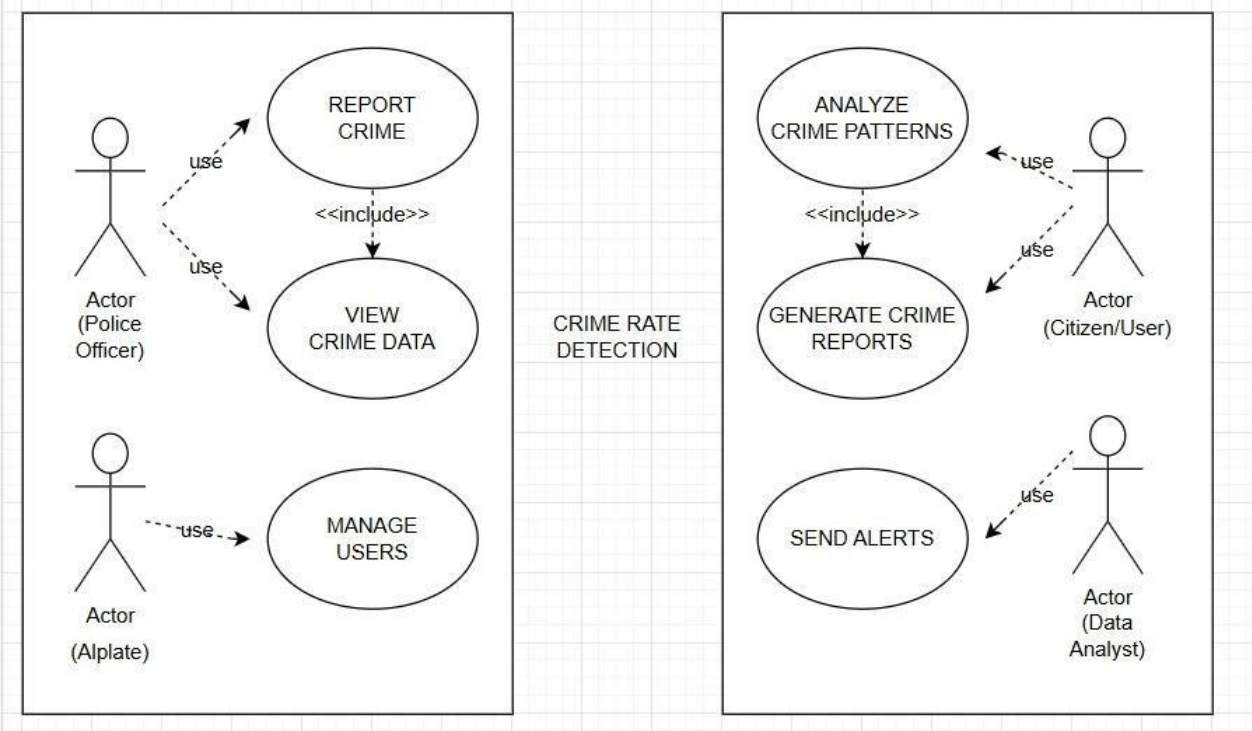
The Class Diagram and Sequence Diagram is designed Successfully for the Crime Rate Detector App

EXP NO: 7	DESIGNING USE-CASE AND ACTIVITY DIAGRAMS FOR PROJECT STRUCTURE

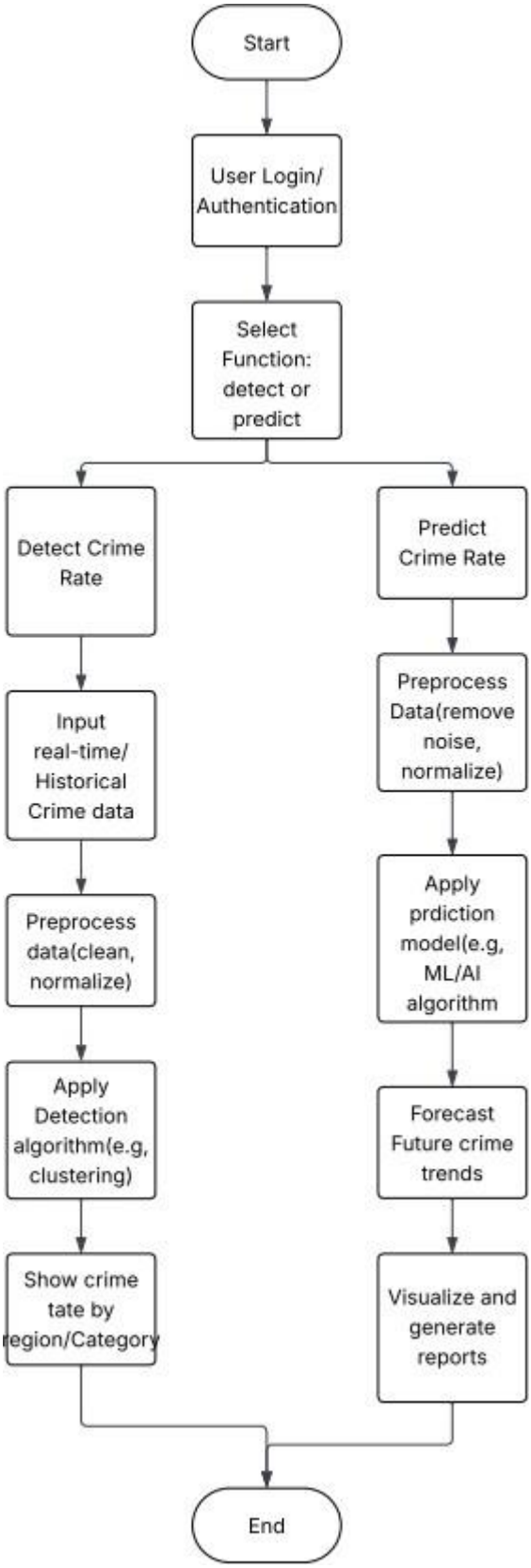
Aim:

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

7A. Use-Case Diagram



7B. Activity Diagram



Result:

The Use Case and Activity is designed Successfully for the Crime Rate Detector App

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

- Data Collection: Collect crime data from APIs, CSVs, public records, and databases.
- Data Preprocessing: Clean, normalize, and structure data (time, location, crime type).
- Machine Learning Prediction: Predict crime probability based on inputs like location and time.
- Risk Zone Identification: Highlight high-crime zones visually using maps or heatmaps.
- Reporting and Logs: Generate reports and logs for crime trends and predictions.

2. Define User Interaction

- A data engineer collecting data from a new source.
- A user submitting a location and date for crime prediction.
- A security analyst viewing the high-risk zone dashboard.

3. Design Happy Path Test Cases

- User enters valid input and receives a correct prediction.
- High-risk zones accurately visualized based on recent data.
- ML model processes cleaned and transformed data correctly.

4. Design Error Path Test Cases

- Missing input fields (e.g., location not provided).
- Invalid or corrupt data fails during preprocessing.
- System tries to predict with outdated or untrained model.

5. Break Down Steps and Expected Results

- **Step-by-step instructions** (e.g., "Enter location, select date, click 'Predict'").
- **Expected outcome** (e.g., "System displays crime risk level: Low/Medium/High").
  This ensures clarity for manual testers and consistency in automation scripts.

6. Use Clear Naming and IDs

- TC01 – Successful Crime Data Collection
- TC05 – Crime Prediction with Valid Input
- TC10 – Handle Missing Location Input
  This helps in mapping test cases to user stories or feature requirements.

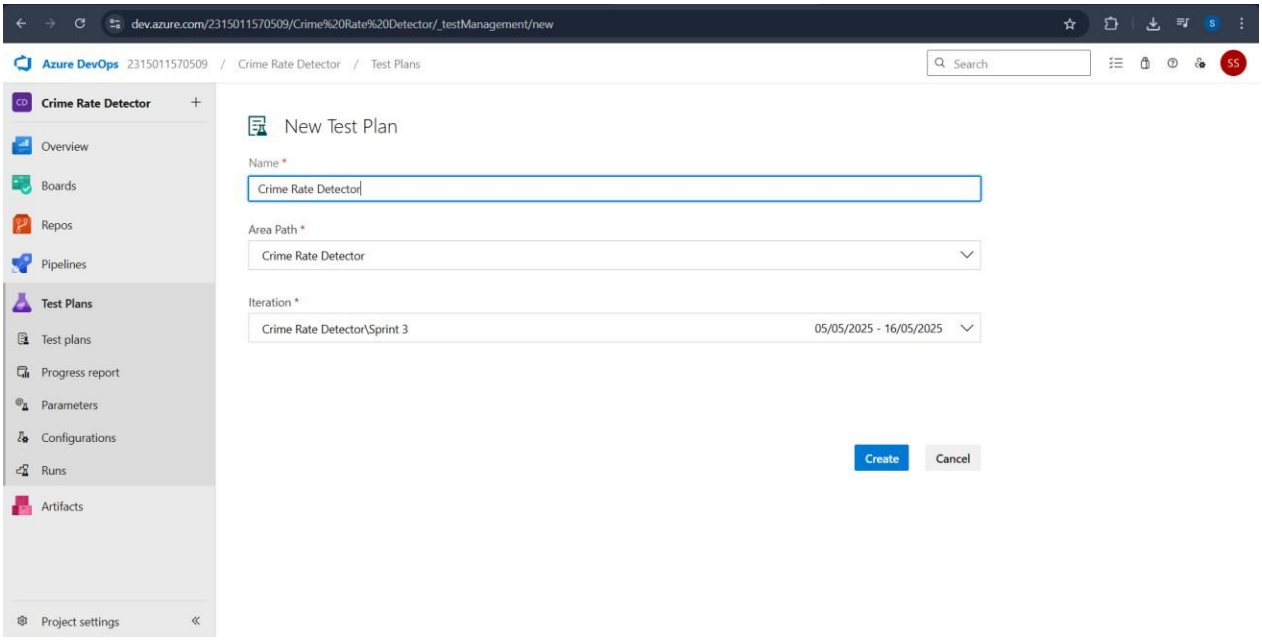
7. Separate Test Suites

- Grouped test cases based on functionality (e.g., Data Collection ML Prediction Engine, Data Preprocessing).
- Improves organization and test execution flow in Azure DevOps.

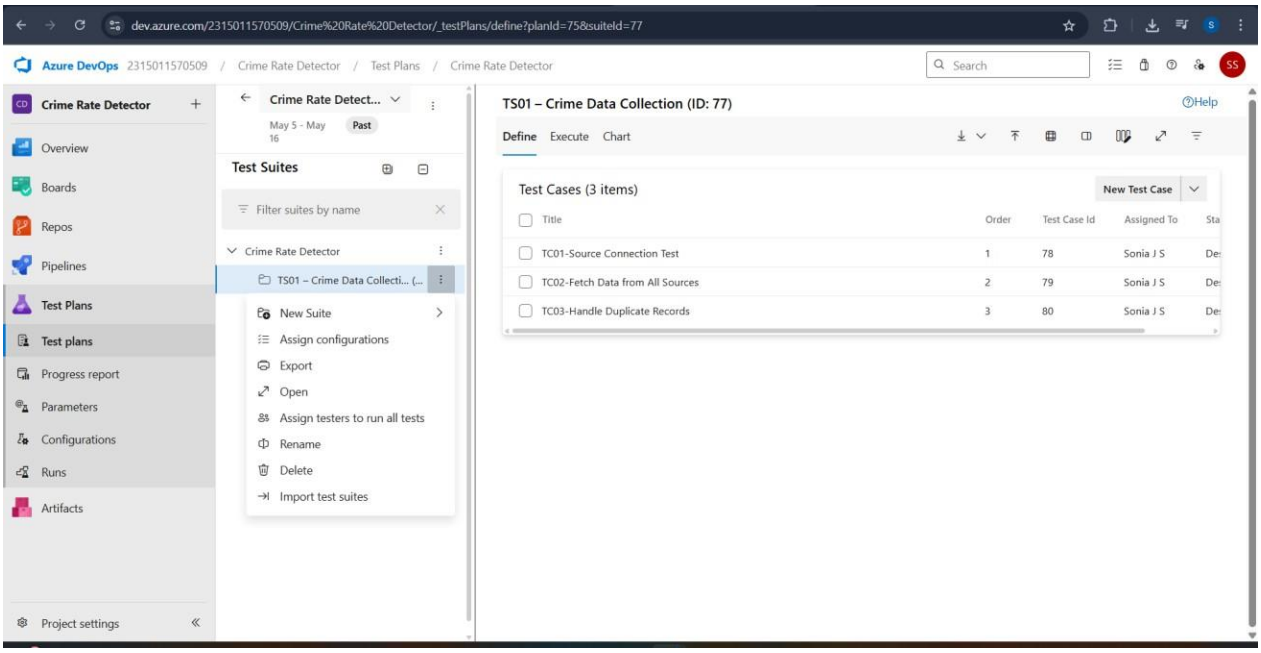
8. Prioritize and Review

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

1. New test plan



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.

Crime Rate Detector App – Test Plans

USER STORIES

1. As a data engineer, I want to collect crime data from various sources so that I can create a comprehensive dataset.

2. As a data analyst, I want to extract time, location, and crime type from raw crime data so that I can structure it properly for accurate and efficient analysis.

3. As a data scientist, I want to encode categorical data and scale numerical features to make the data ML-ready.

4. As a developer, I want to clean inconsistent or missing values in the dataset so that I can ensure high-quality inputs for the machine learning model.

5. As a security analyst I want to identify high-risk zones So that I can allocate resources effectively.

Test Suites

Test Suite: TS01 – Crime Data Collection (ID: 77)

TC01 – Source Connection Test

- **Action:**
  - Attempt connection to data source
  - Check for timeout or error

- **Expected Results:**
  - Connection established successfully
  - No timeout or error observed
- **Type:** Happy Path

TC02 – Fetch Data from All Sources

- **Action:**
  - Trigger data retrieval
  - Log number of records retrieved
- **Expected Results:**
  - Data retrieval process initiates
  - Record count is logged correctly
- **Type:** Happy Path

TC03 – Handle Duplicate Records

- **Action:**
  - Load data from multiple sources
  - Check for duplicate crime IDs or timestamps
- **Expected Results:**
  - All sources are successfully loaded
  - Duplicates flagged or removed
- **Type:** Happy Path

Test Suite: TS02 – Data Structuring (ID: 81)

TC04 – Extract Timestamps

- **Action:**
  - Upload raw data
  - Extract timestamp field
- **Expected Results:**
  - Data loads without errors
  - Dates standardized to YYYY-MM-DD format
- **Type:** Error Path

TC05 – Parse Location Data

- **Action:**
  ▪ Process raw data
  ▪ Extract location fields
- **Expected Results:**
  ▪ Data processed successfully
  ▪ City, ZIP, and coordinates extracted correctly
- **Type:** Happy Path

TC06 – Classify Crime Types

- **Action:**
  ▪ Analyze crime descriptions
  ▪ Match against predefined categories
- **Expected Results:**
  ▪ Descriptions are readable and processed
  ▪ Each record is assigned a crime type
- **Type:** Happy Path

Test Suite: TS03 – Feature Engineering (ID: 85)

TC07 – Encode Categorical Features

- **Action:**
  ▪ Identify categorical fields
  ▪ Apply Label/One-Hot encoding
- **Expected Results:**
  ▪ All categorical fields are found
  ▪ Encoding completes without errors
- **Type:** Happy Path

TC08 – Scale Numerical Features

- **Action:**
  ▪ Load dataset
  ▪ Apply MinMax or StandardScaler
- **Expected Results:**
  ▪ Dataset loads correctly
  ▪ Values fall within expected scaled range
- **Type:** Happy Path

TC09 – Validate ML Input Schema

- **Action:**
  - ▪ Run schema validator
  - ▪ Verify dataset
- **Expected Results:**
  - ▪ Schema is read and applied
  - ▪ Dataset contains required fields with no nulls or incorrect formats
- **Type:** Happy Path

Test Suite: TS04 – Data Cleaning (ID: 89)

TC10 – Check for Missing Values

- **Action:**
  - ▪ Scan dataset
  - ▪ Identify null/empty fields
- **Expected Results:**
  - ▪ Scan completes
  - ▪ Missing values highlighted or flagged
- **Type:** Happy Path

TC11 – Format Consistency Check

- **Action:**
  - ▪ Load data
  - ▪ Validate formats (date, address, type)
- **Expected Results:**
  - ▪ File loads successfully
  - ▪ All fields meet format specifications
- **Type:** Happy Path

TC12 – Remove Outliers

- **Action:**
  - ▪ Apply outlier detection
  - ▪ Remove or normalize

- **Expected Results:**
  - Outliers detected correctly
  - Dataset updated with outliers handled
- **Type:** Error Path

Test Suite: TS05 – High-Risk Zone Detection (ID: 93)

TC13 – Calculate Zone Risk Score

- **Action:**
  - Aggregate crimes by location
  - Compute risk using frequency/severity
- **Expected Results:**
  - Crime counts grouped correctly
  - Risk score reflects accurate level
- **Type:** Error Path

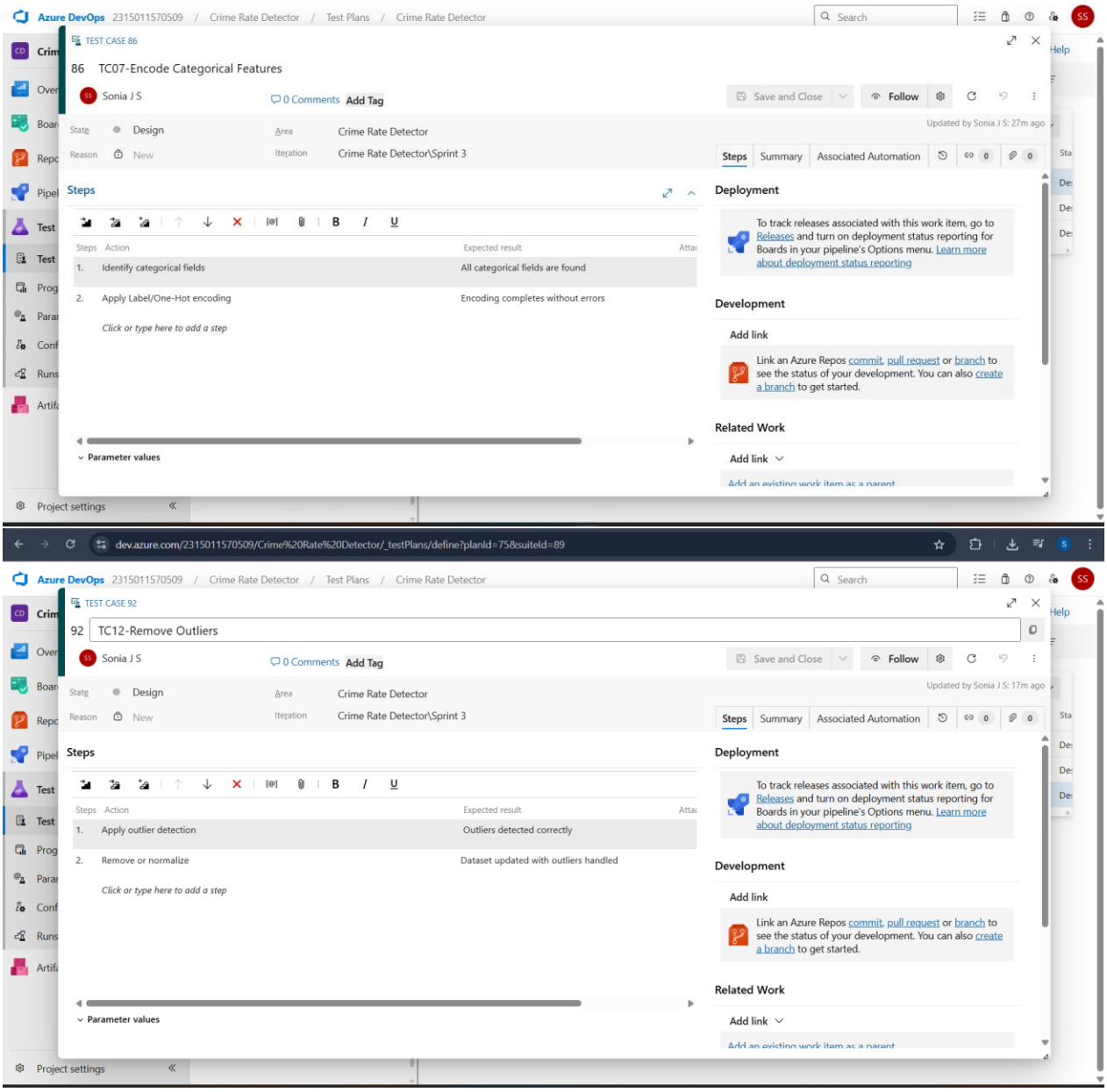
TC14 – Generate Risk Heatmap

- **Action:**
  - Input data into visualization tool
  - Display heatmap
- **Expected Results:**
  - Data loads to map tool
  - High-risk areas colored with appropriate intensity
- **Type:** Happy Path

TC15 – List Top High-Risk Areas

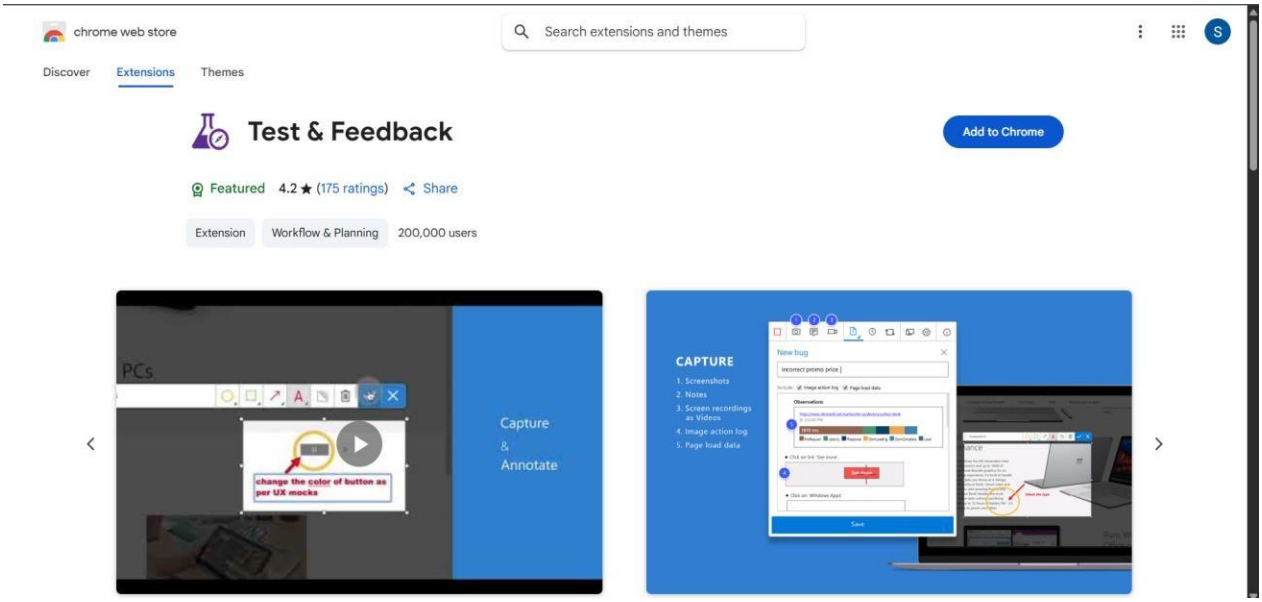
- **Action:**
  - Sort areas by crime score
  - Display top 5 zones
- **Expected Results:**
  - Areas ranked correctly
  - List includes top 5 zones with scores shown
- **Type:** Happy Path

Test Cases

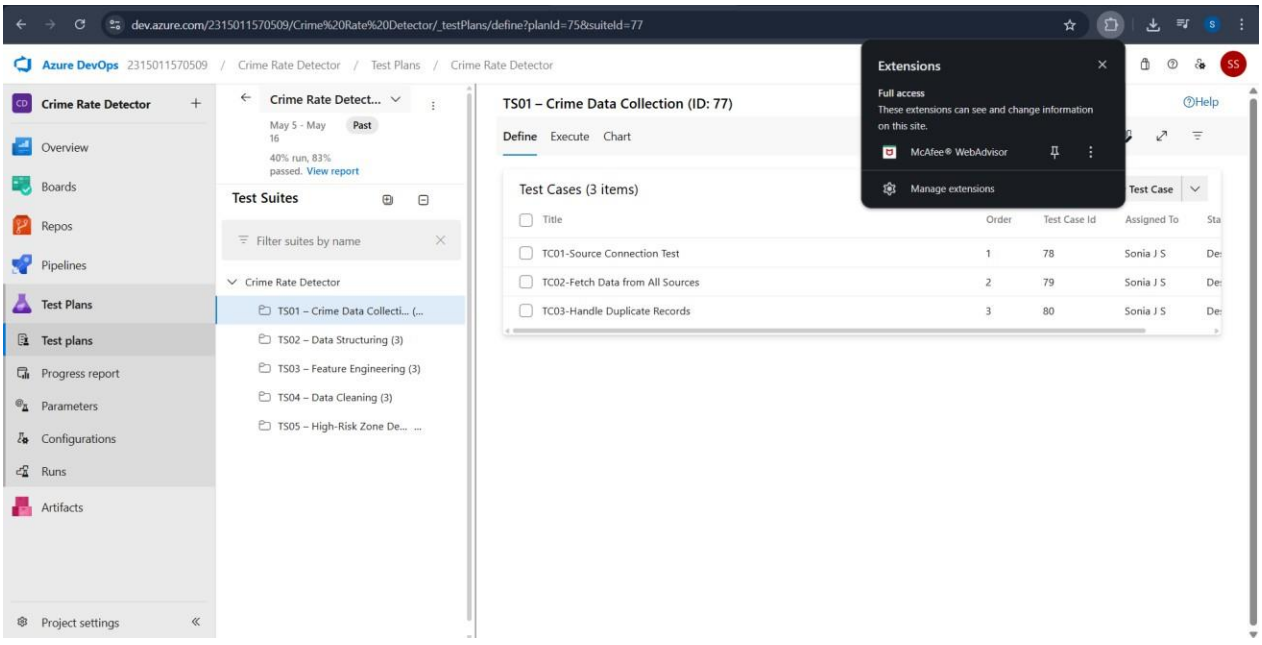

Azure DevOps 2315011570509 / Crime Rate Detector / Test Plans / Crime Rate Detector
Search
TEST CASE 86
86 TC07-Encode Categorical Features
Sonia J S
0 Comments Add Tag
Save and Close
Follow
State Design
Area Crime Rate Detector
Reason New
Iteration Crime Rate Detector\Sprint 3
Updated by Sonia J S: 27m ago
Steps Summary Associated Automation
Steps
Steps Action Expected result
1. Identify categorical fields All categorical fields are found
2. Apply Label/One-Hot encoding Encoding completes without errors
Click or type here to add a step
Parameter values
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
Project settings

dev.azure.com/2315011570509/Crime%20Rate%20Detector/_testPlans/define?planId=75&suiteId=89
Azure DevOps 2315011570509 / Crime Rate Detector / Test Plans / Crime Rate Detector
Search
TEST CASE 92
92 TC12-Remove Outliers
Sonia J S
0 Comments Add Tag
Save and Close
Follow
State Design
Area Crime Rate Detector
Reason New
Iteration Crime Rate Detector\Sprint 3
Updated by Sonia J S: 17m ago
Steps Summary Associated Automation
Steps
Steps Action Expected result
1. Apply outlier detection Outliers detected correctly
2. Remove or normalize Dataset updated with outliers handled
Click or type here to add a step
Parameter values
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
Project settings

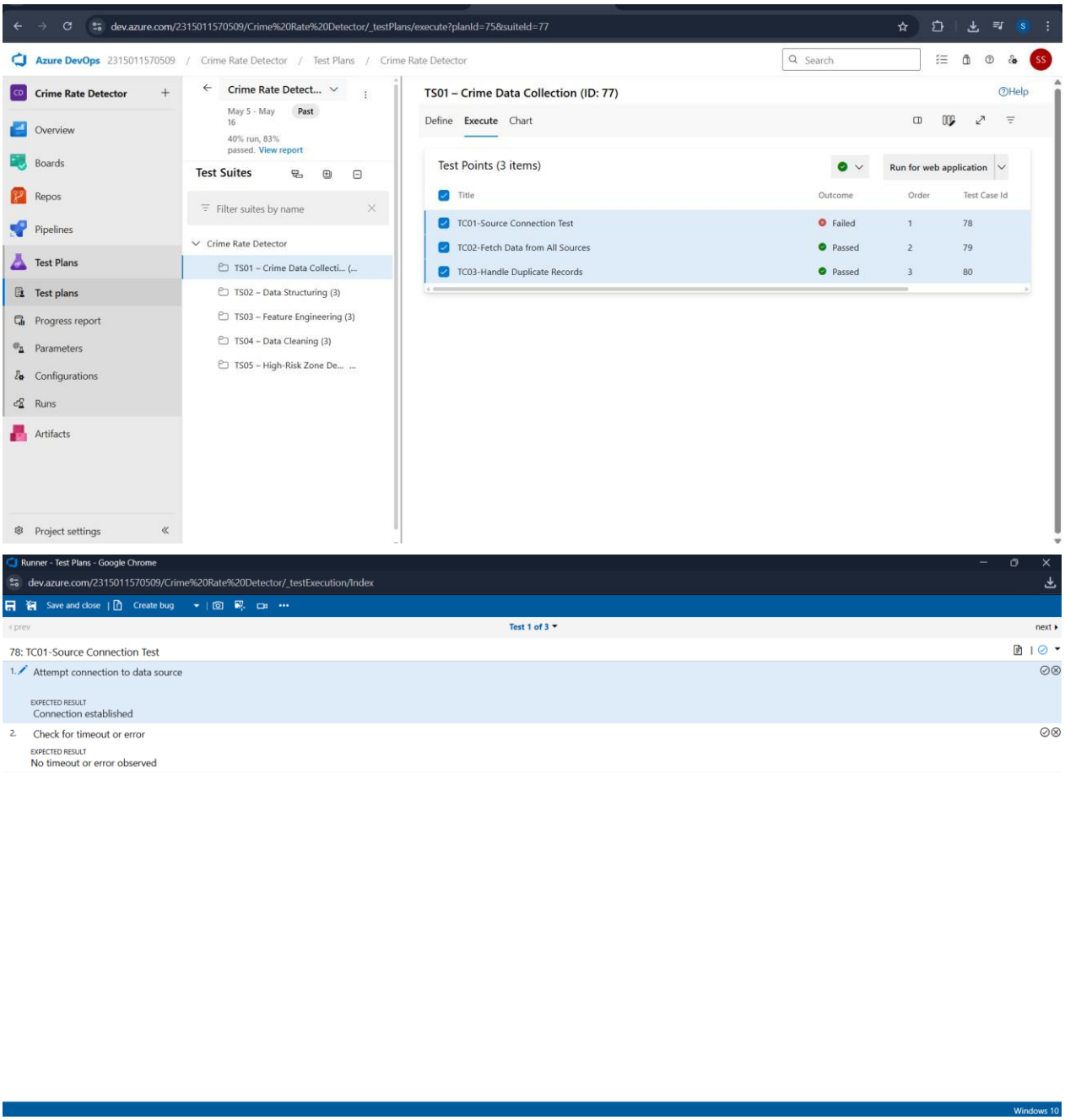
4. Installation of test



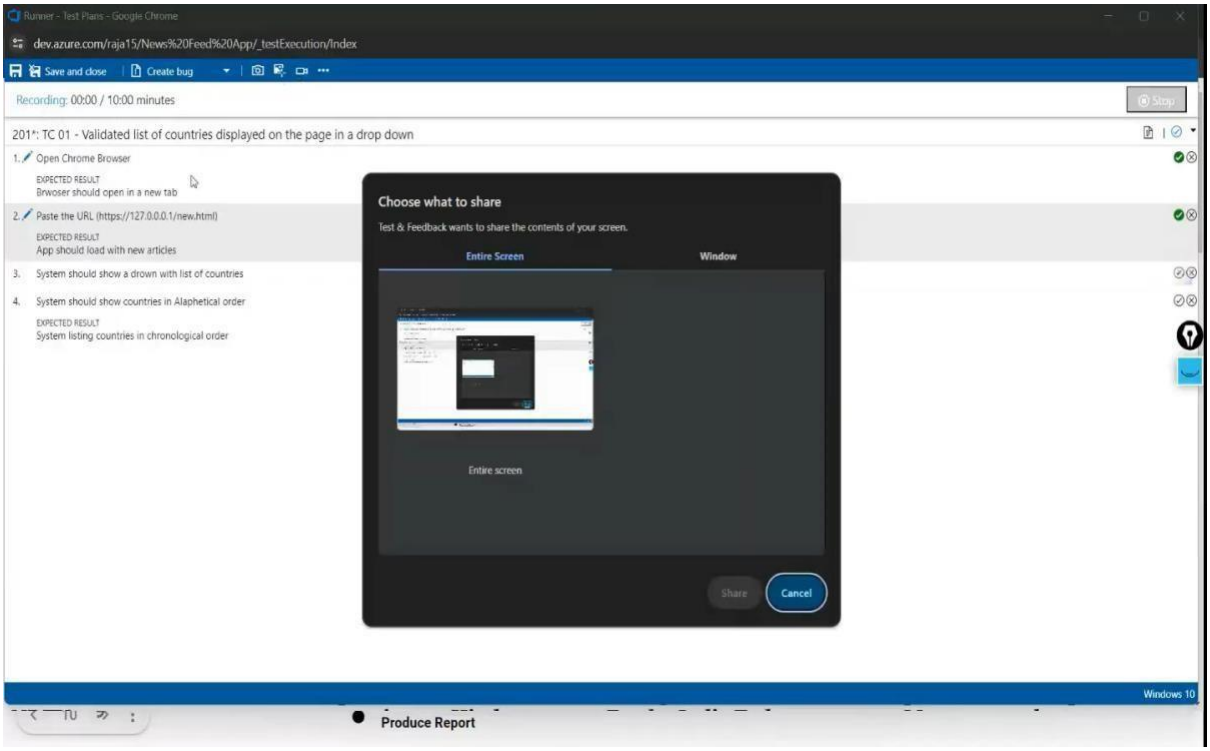
Test and feedback
Showing it as an extension



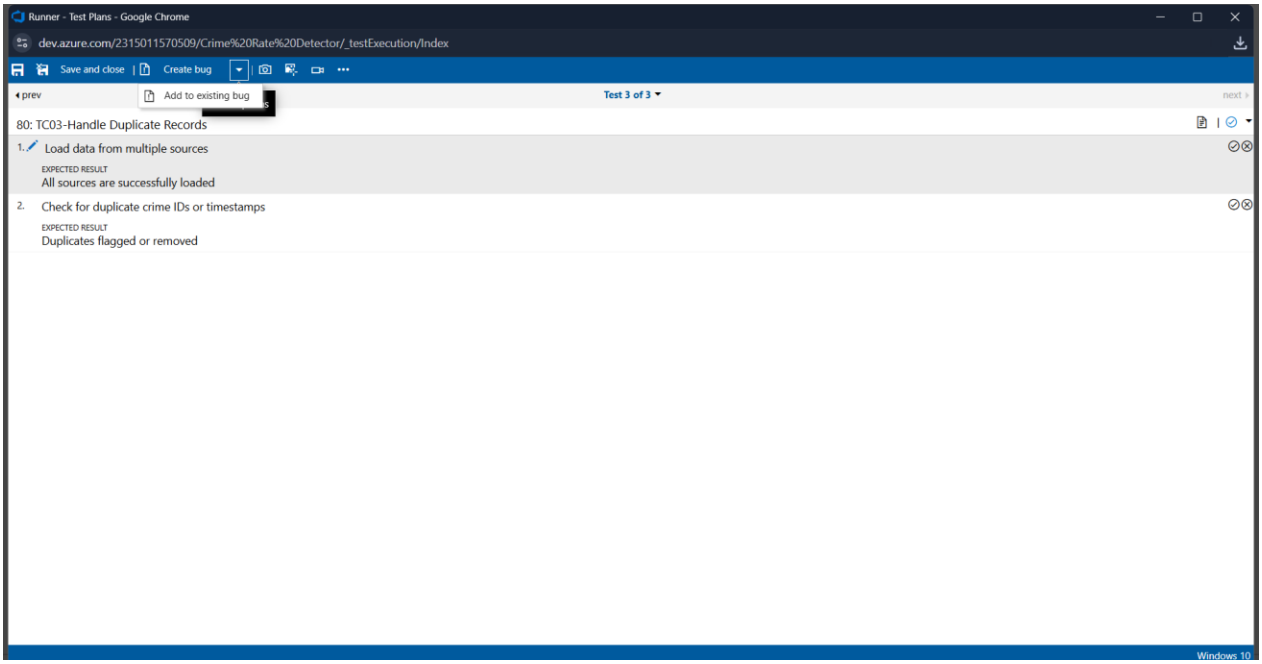
5. Running the test cases

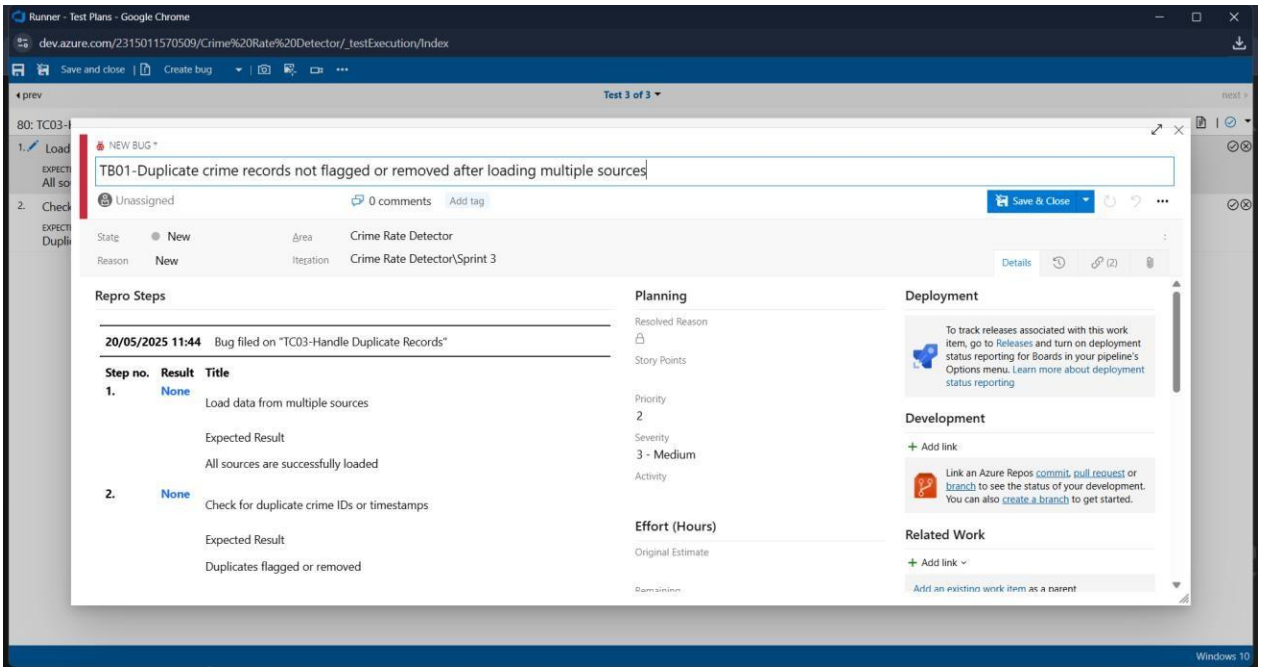


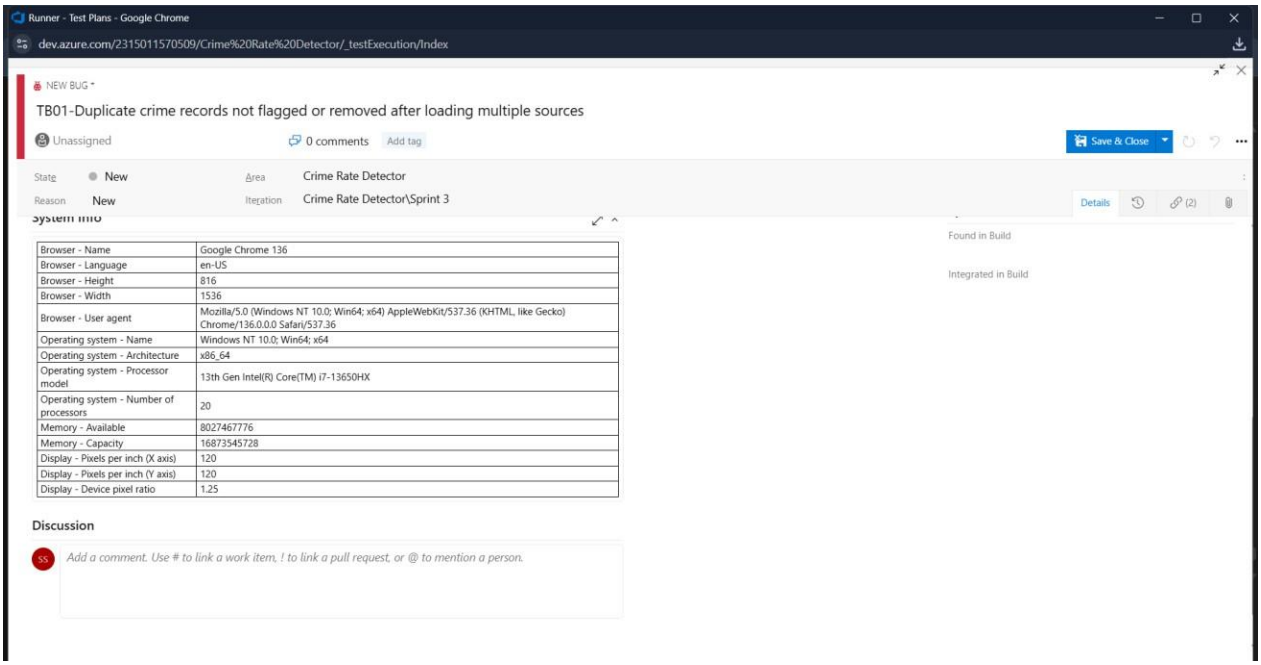
6. Recording the test case



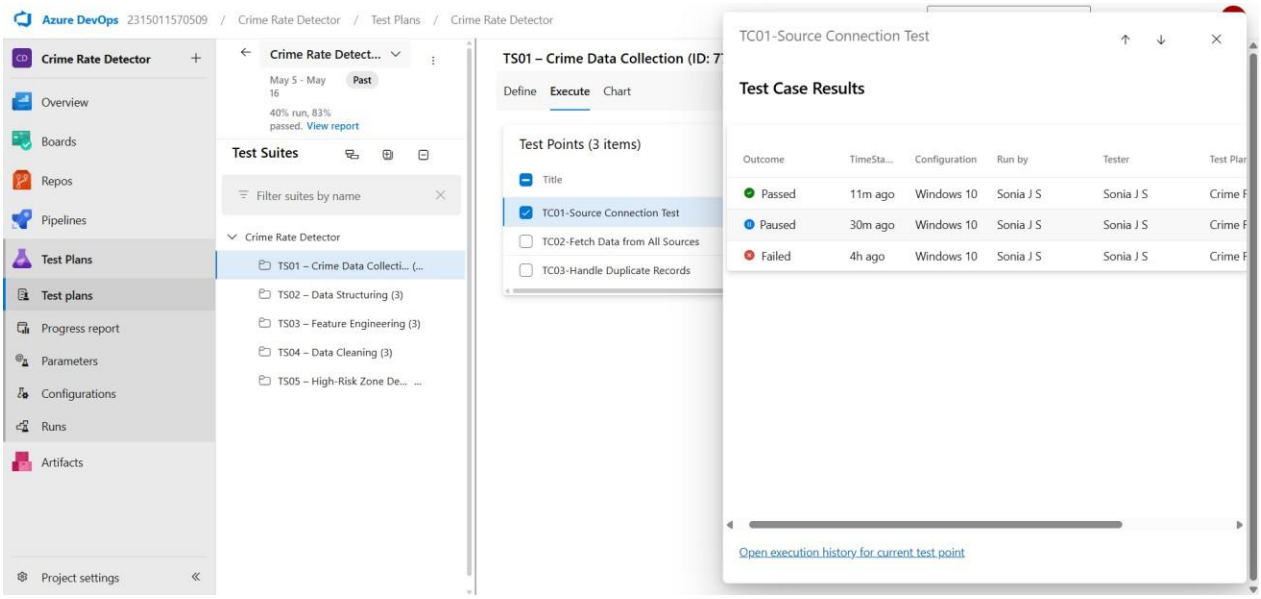
7. Creating the bug



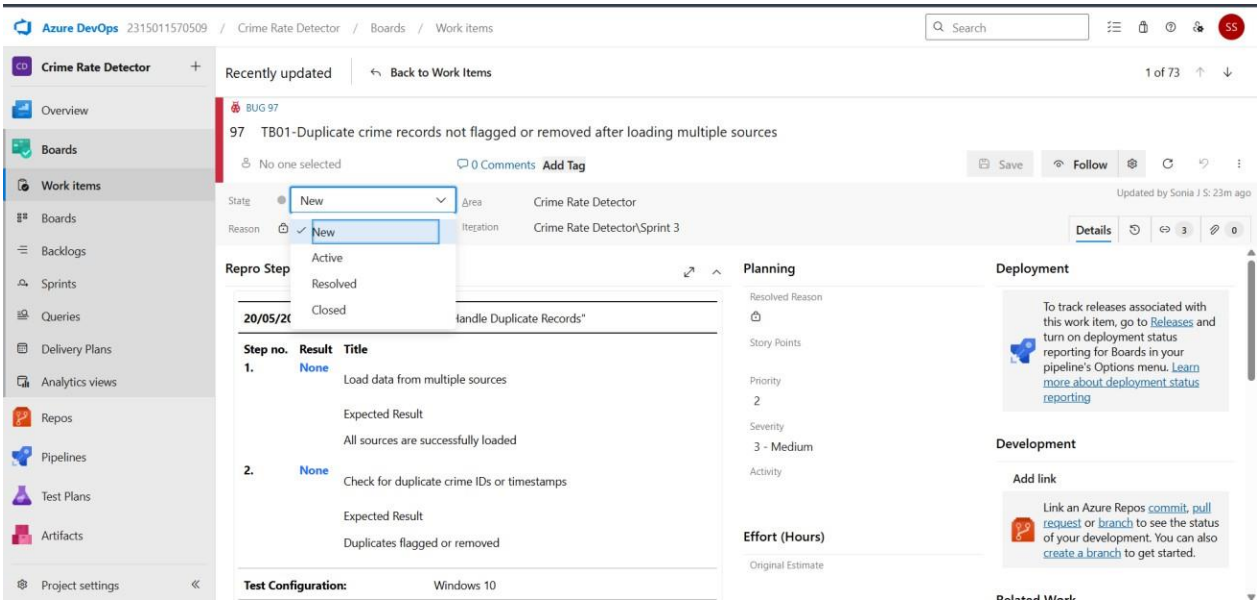

Runner - Test Plans - Google Chrome
dev.azure.com/2315011570509/Crime%20Rate%20Detector/_testExecution/Index
Save and close
Create bug
prev
Test 3 of 3
NEW BUG *
TB01-Duplicate crime records not flagged or removed after loading multiple sources
Unassigned
0 comments
Add tag
Save & Close
State
New
Area
Crime Rate Detector
Reason
New
Iteration
Crime Rate Detector\Sprint 3
Details
Repro Steps
20/05/2025 11:44 Bug filed on "TC03-Handle Duplicate Records"
Step no. Result Title
1. None
Load data from multiple sources
Expected Result
All sources are successfully loaded
2. None
Check for duplicate crime IDs or timestamps
Expected Result
Duplicates flagged or removed
Planning
Resolved Reason
Story Points
Priority
2
Severity
3 - Medium
Activity
Effort (Hours)
Original Estimate
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
Windows 10


Runner - Test Plans - Google Chrome
dev.azure.com/2315011570509/Crime%20Rate%20Detector/_testExecution/Index
NEW BUG *
TB01-Duplicate crime records not flagged or removed after loading multiple sources
Unassigned
0 comments
Add tag
Save & Close
State
New
Area
Crime Rate Detector
Reason
New
Iteration
Crime Rate Detector\Sprint 3
Details
Found in Build
Integrated in Build
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 | 13th Gen Intel(R) Core(TM) i7-13650HX
Operating system - Number of processors | 20
Memory - Available | 8027467776
Memory - Capacity | 16873545728
Display - Pixels per inch (X axis) | 120
Display - Pixels per inch (Y axis) | 120
Display - Device pixel ratio | 1.25
Discussion
SS
Add a comment. Use # to link a work item, ! to link a pull request, or @ to mention a person.

8. Test case results

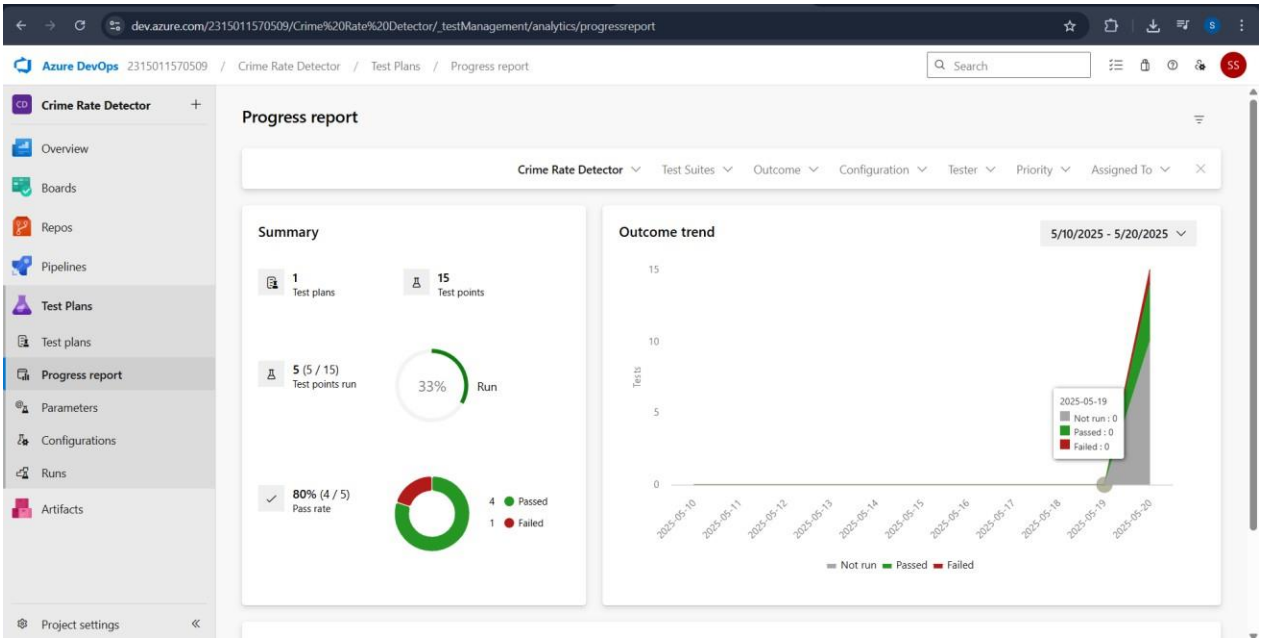


9. Test report summary

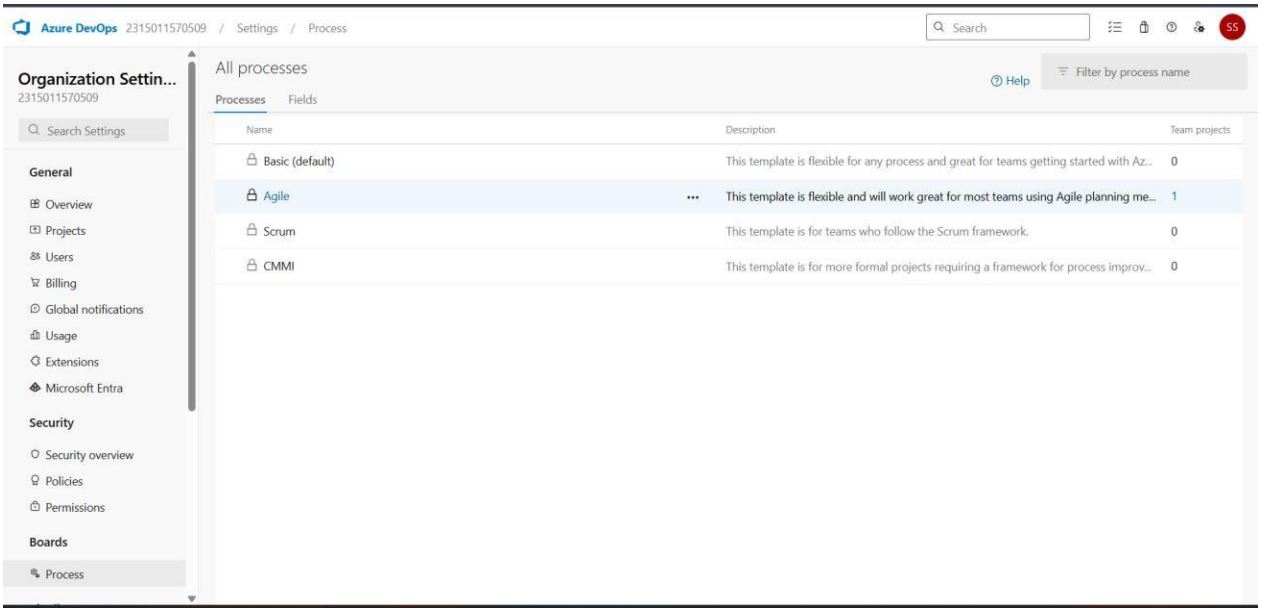


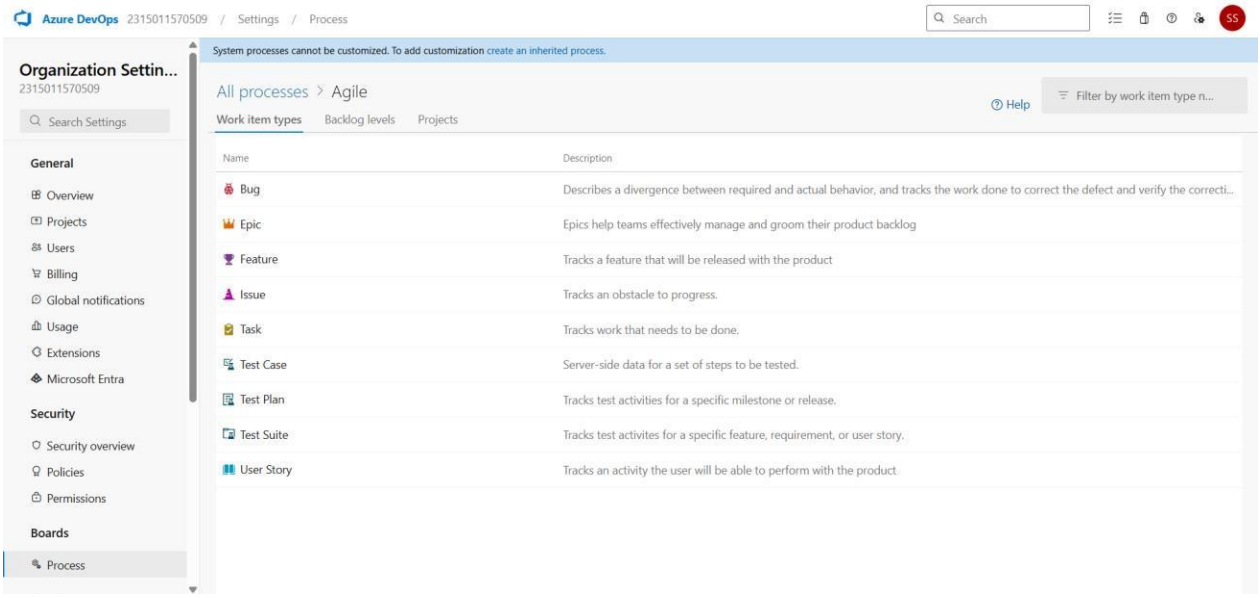
- Assigning bug to the developer and changing state

10. Progress report



11. Changing the test 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 Student 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 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 - ContinuousIntegration) 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.

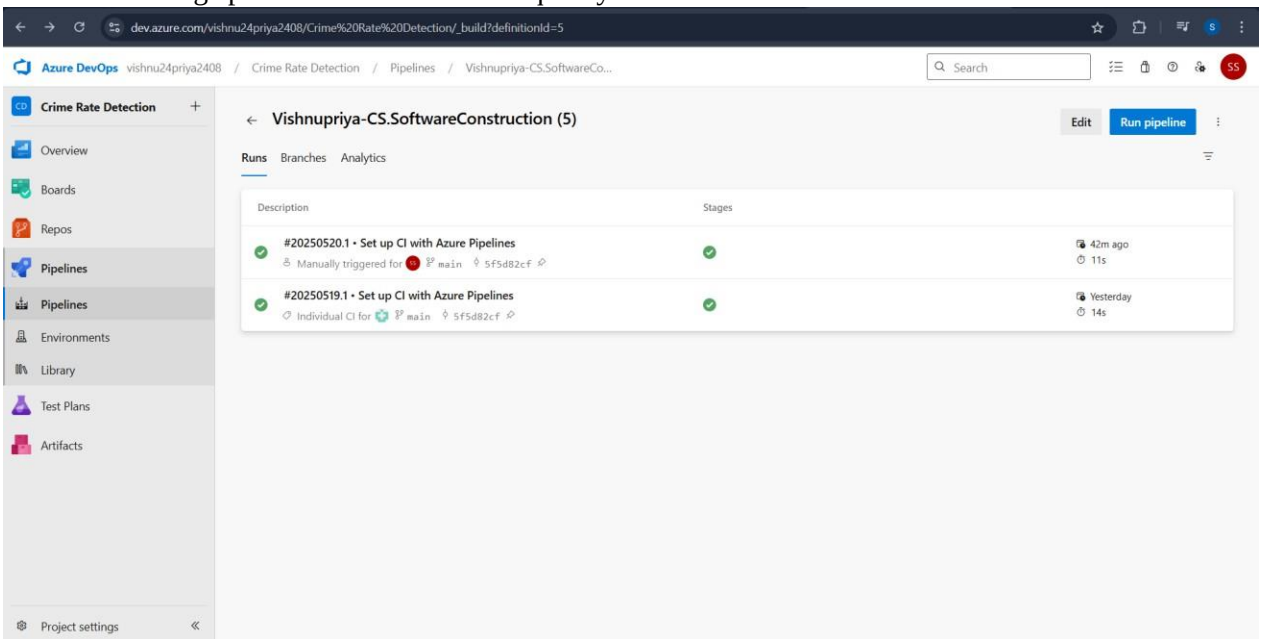
11. Configure Continuous Deployment (CD) Navigate to the Releases tab under Pipelines and click “New Release Pipeline”. Add an Artifact (from the build stage) and create a new Stage (e.g., Development, Production).

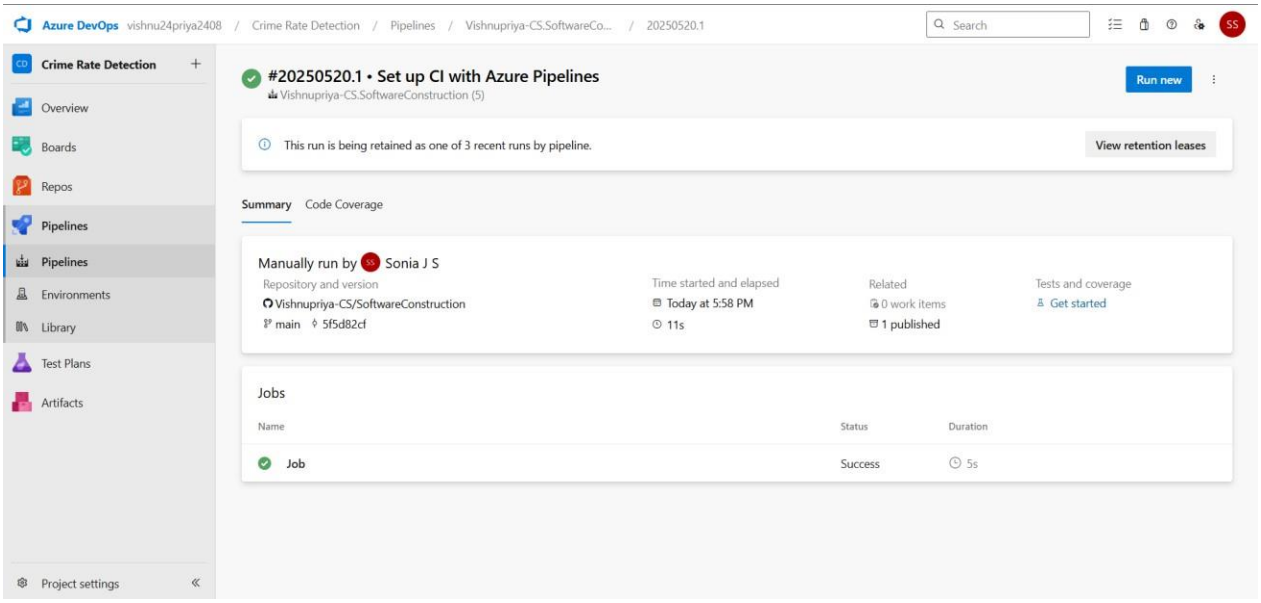
12. Configure the CD stage with deployment tasks such as deploying to Azure App Service, running database migrations or scripts, and restarting services using the Azure App Service Deploy task linked to your subscription and app details.

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:

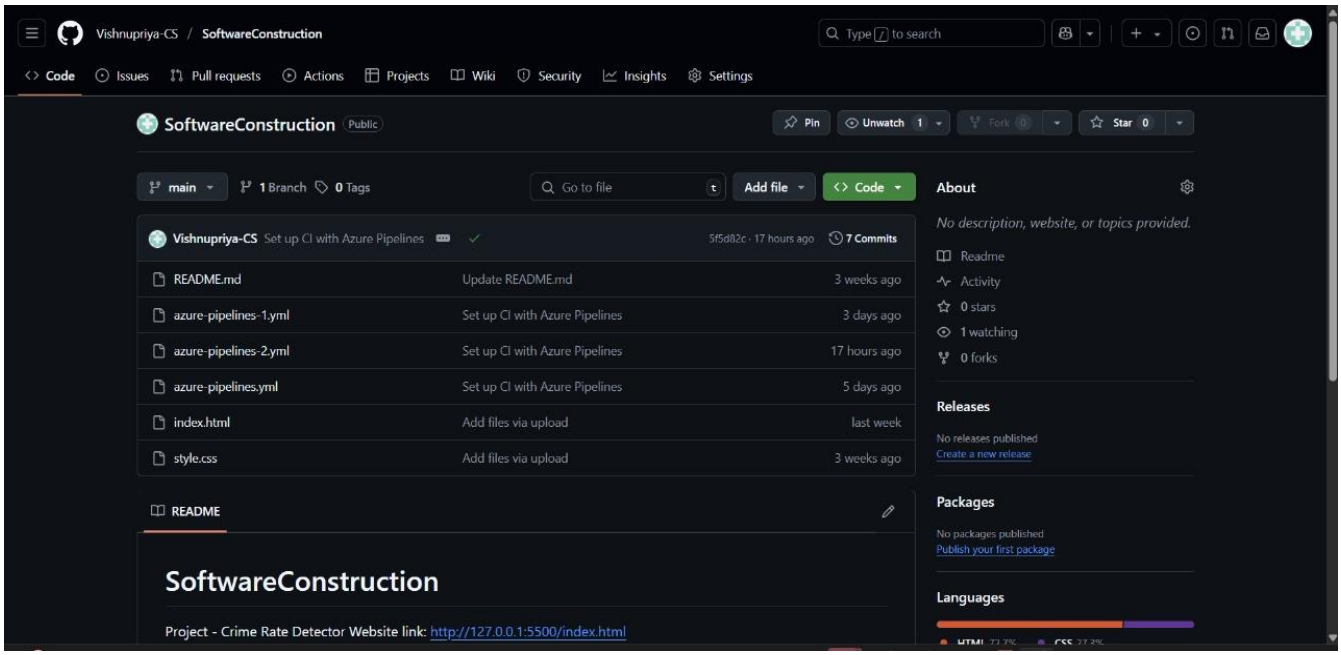
Thus the pipelines for the given project “Crime Rate Detector” has been executed successfully

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 Music Playlist Batch Creator project.

GitHub Project Structure



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.