**E-FAWATERCOM SYSTEM**

**Project Documentation**

# Index

# Introduction

## **Objective**

Testing's primary objective is to find as many bugs as possible in software products while determining whether the program satisfies user requirements or not.

## **System Overview**

Functional and non-functional requirements plus interface design need to be tested using appropriate tools, some parts of attached project need to be automated.

# E-Fawatercom Project Mini-SRS

Many people find it difficult to go to government departments and pay bills, so this site is designed to save time and effort.

**E-Fawatercom** is an integrated program developed to save time and effort. It is a dynamic and responsive system. This site provides a quick and easy way to create and pay invoices online.

## **Interface Design**

· Use the recognized colors in the design.

· Users’ sessions must be designed.

· Home Page must contain about us, contact us, header, footer, and background or slider.

· Each part in the home must change dynamically using Database.

## **Admin**

* Login
* View user registration details.
* Obtain a report showing the number of bills that have been paid and the bills that have not yet been paid.
* Update his profile.
* Obtains Financial monthly and annual reports on the website and (PDF or Excel) documents.
* Manage pages (Home, About Us, Contact Us).

## **User**

* Enter the home page.
* Register on the Website
* Login.
* Search for the invoice number.
* View his bills and pay them via Visa.
* Can view his invoice and download it.
* Can view and update his profile.

# Testing Requirements

## **Before You Start**

* Install Oracle Express edition and configure the database locally using SQL Developer
* Create a new user in the database and execute the attached SQL program on it.
* Install Visual Studio
* Unzip the attached API project and open it on visual studio.
* Change the connection string in the API project.
* Run the API project.
* Install Visual Studio Code and Node.js.
* From CMD, install the angular CLI package using npm install -g @angular/cli.
* Unzip the attached Angular project and open it on visual studio code.
* Open the terminal and execute this command to bypass policies set-executionPolicy -scope process -executionpolicy bypass
* Run Angular Project using ng serve -o command.
* Create an account on Azure DevOps to use as a management tool to create your test cases under each of the following PBIs and add the bug report under each test case if there is any.
* Customize the test case and bug templets by adding the following fields for each one if not exist:

Bug Template

1. Bug title 2. Bug Description

3. Steps to Reproduce 4. Actual Result

5. Expected Result 6. Severity

7. Priority 8. Type 9. Source

10. Attachments 11. Acceptance criteria

12- Environment

Test Case Template

1. Title  2. Description 3. Steps 4. Test Data

5. Actual Result 6. Expected Result 7. Status

[Follow this user manual to make your customization.](#_Customize_Test_case)

## **Manual Testing**

* The database must be used to get the actual outcome.
* Use your creativity and critical thinking to create the test cases.

Now, we'll break the project into epics, break them down into features, and further break those features into Product Backlog Items (PBIs).

**Epic:** **Home Pages**

**Description:** Improve the existing application to provide users with more accurate and detailed information about the services provided by the system.

**Features:**

1. **Home page**
   * Description: Integrates with dynamic data upgraded by admin all time including feedback from users.
   * Acceptance Criteria: The application should display feedback from registered users and dynamic information about the services provided by it.
2. **Contact us page.**
   * Description: Enable users to send messages to admin.
   * Acceptance Criteria: Any user can send messages and get replies on mobile phone or as email.
3. **Login**
   * Description: Users and admin can login to the system and use the provided services.
   * Acceptance Criteria: Authenticated users can login to the application in addition to the administrator.
4. **Register.**
   * Description: Enable users to sign up in the application to use its services.
   * Acceptance Criteria: Enable users to sign up in the application after inserting their correct data.

**Product Backlog Items (PBIs) under "Home Page" Feature:**

1. **PBI:** Display all information dynamically.
   * Description: Content displayed on the home page must be dynamic.
   * Acceptance Criteria: Admin must have the privileges which enable him to control all content displayed on the home page.

**Product Backlog Items (PBIs) under “Contact us page” Feature:**

1. **PBI:** Sending messages to contact the owners of application.
   * Description: Users must be able to send messages to Administrator.
   * Acceptance Criteria: All users must be able to send messages and get replies via email or phone number.
2. **PBI:** Display all information dynamically.
   * Description: Content displayed on the contact us page must be dynamic.
   * Acceptance Criteria: Admin must have the privileges which enable him to control all content displayed on the contact page.

**Product Backlog Items (PBIs) under “Login” Feature:**

1. **PBI:** All Authenticated users can login to the application.
   * Description: Registered users can login to the application.
   * Acceptance Criteria: Only registered users can login to the application and use its services securely.

**Product Backlog Items (PBIs) under “Register” Feature:**

1. **PBI:** Sign up into the application after filling in correct data.
   * Description: Guest users can sign up into the application to enjoy its services.
   * Acceptance Criteria: Users who enter valid data with correct email address, phone number, strong password can register successfully and use application’s services.

**Epic:** **User Pages**

**Description:** He is a user who enters the site to view his bills and pays them via Visa.

**Features:**

1. **Pay a new bill!**
   * Description: Users can pay their bills in different categories.
   * Acceptance Criteria: Users can search for specific category and enter the bill number to get more description, then complete the payment process.
2. **Payment history.**
   * Description: Payment list is saved in user dashboard.
   * Acceptance Criteria: each bill paid by a specific user is saved in a list in the user dashboard.
3. **Manage Testimonial** 
   * Description: Users can write their own feedback about system services.
   * Acceptance Criteria: User can send their feedback with rate from 1 to 5 but it will not be displayed in home page until the admin accepts it.
4. **Manage Account.**
   * Description: Ability to edit profile.
   * Acceptance Criteria: User must be able to edit his profile: name, password, location, etc...
5. **Manage Logout.**
   * Description: Logout securely from application.
   * Acceptance Criteria: User logouts from application in secure way.

**Product Backlog Items (PBIs) under “Pay new bill” Feature:**

1. **PBI:** Search for a specific category and search for a bill belonging to.
   * Description: Search for a specific category to find bill in to pay it.
   * Acceptance Criteria: After searching for a specific category and entering the bill number all details about it will be returned.
2. **PBI:** Complete the payment process.
   * Description: Determine the total amount to pay and choose the payment method.
   * Acceptance Criteria: after determining the total amount, the user can choose the payment method by visa card or by bank account.

**Product Backlog Items (PBIs) under “Payment History” Feature:**

1. **PBI:** Displaying list with all payment operations.
   * Description: all payment operations are saved in the user dashboard
   * Acceptance Criteria: all payment details are displayed in the user dashboard including the payment date, bill name, total amount, etc…

**Product Backlog Items (PBIs) under “Manage Testimonials” Feature:**

1. **PBI:** Send Testimonials about the services provided.
   * Description: users can send testimonials with alias names and evaluate the application.
   * Acceptance Criteria: user can send his own feedback with alias name and rate the services from 1 to 5 and add his message, but it will not be displayed till the admin accepts it.

**Product Backlog Items (PBIs) under “Manage Account” Feature:**

1. **PBI:** All users can edit their profiles.
   * Description: all users can update their profile and change their data.
   * Acceptance Criteria: Users can update their profile but after confirming the password, they can change their passwords anytime.

**Product Backlog Items (PBIs) under “Manage Logout” Feature:**

1. **PBI:** Logout correctly from application.
   * Description: Logout from application.
   * Acceptance Criteria: Admin must be able to logout correctly from application.

## **Automation Testing**

* Create an automated system based on the requirements.
* Run your test and generate an extent report containing bug details and screenshots of the bugs found.
* Apply Data-Driven Testing by reading the test data from an Excel file.
* Apply POM design pattern to improve the code quality and organization.

**Epic:** **Admin Pages**

**Description:** is a user who follows the procedures of invoices on the site and matters related to financial services and obtains complete reports about the users.

**Features:**

1. **Manage Categories**
   * Description: Adding new bill categories.
   * Acceptance Criteria: Admin must be able to add new bill categories and create bills that belong to them. In addition to CRUD operations.
2. **Manage Report**
   * Description: Display Detailed Information about bills in each category.
   * Acceptance Criteria: Admin must be able to get detailed information about any bill he searched for by name or between specific intervals.
3. **Manage Profile**
   * Description: Ability to edit profile.
   * Acceptance Criteria: Admin must be able to edit his profile: name, password, location, etc...

**Product Backlog Items (PBIs) under “Manage Category” Feature:**

1. **PBI:** Create new bill on a specific category.
   * Description: Adding new bills on categories randomly.
   * Acceptance Criteria: Admin can add new bill on each category and then display it under this category.

**Product Backlog Items (PBIs) under “Manage Report” Feature:**

1. **PBI:** Search by Category name.
   * Description: Search categories by names.
   * Acceptance Criteria: When Admin searches for category by its name, just the information related is displayed.
2. **PBI:** Search by date
   * Description: Search categories by payment date for bills in each one.
   * Acceptance Criteria: When admin searches for categories between date, the categories that have bills are paid in this interval must be displayed.

**Product Backlog Items (PBIs) under “Manage Profile” Feature:**

1. **PBI:** Change name.
   * Description: The ability to change name.
   * Acceptance Criteria: Admin must be able to change his name after confirming his password.

## **Performance Testing**

* Use the BlazeMeter as a recording tool.
* Create the test plan using the GUI mode.
* Use the appropriate controllers as you learned.
* Execute your test in non-GUI mode and generate HTML Report.

**Now, let’s create the following test plan:**

* + Number of threads: 3
  + Ramp-up period: 3
  + Loop count: 1

**Each user (thread) should do the following:**

* + Register
  + Login
  + Add a testimonial three times.
  + Edit the name field on the profile page twice.

**Note that:**

* + Each user (thread) should register with a different username.
  + Extract the username and password to use them in the Login transaction.
  + Adding testimonials and updating the name in the profile must be executed in random order. (Requirements 3 & 4)
  + Make sure the transaction will not be executed if the previous one failed.

## **API Testing**

* Use the Postman to test the following APIs if they are working correctly.
* The database must be used to get the actual outcome.
* Use the collection to organize your testing.

1. Get All Advantages:

* **HTTP Method:**

Get

* **API Request:**

<https://localhost:44318/api/Madadvantage/GetAllmadadvantage>

* **Request Data**

None

* **Response Data**

[

    {

        "advantageId": 2,

        "advantage": "INTUITIVE, INSTANT, FRICTION-FREE",

        "description": "Replaces paper bills and payments with user-friendly services on the web and mobile. This accelerated the adoption of this payment system ",

        "pictuer": "45612d2e-8964-448e-80cd-7f422184b8e6\_resize.jpg"

    },

    {

        "advantageId": 3,

        "advantage": "SCALABLE",

        "description": "Supports the billing and payment needs of small, mid-sized aend entrprise companies. Adaptable to any industry challenges in any market setting. Smart API-enabled solution scales to your growing needs",

        "pictuer": "56bde4ea-2fa1-4684-bb68-649351175ccf\_SCALABLE.jpg"

    },

    {

        "advantageId": 4,

        "advantage": "SECURE",

        "description": "Meeting the highest industry standards in protecting sensitive information. We?re ISO 27001-certified and use best industry practices in compliance with Automated Clearing House Association (ACHA) and PCI DSS regulations",

        "pictuer": "fa4d22f1-0cb3-45e4-8fc3-efffa93b878c\_SECURE.jpg"

    },

    {

        "advantageId": 61,

        "advantage": "SAVES MONEY And TIME",

        "description": "Reduces collection cycle time and billing expenses, which in turn makes payment faster and improves cash flow\n",

        "pictuer": "fed6bdcd-d8c4-49bb-8594-2f3ace27b9a4\_savemony.jpg"

    }

]

1. Create New Advantage:

* **HTTP Method:**

Post

* **API Request:**

<https://localhost:44318/api/Madadvantage>

* **Request Data**

{

    "advantage": "Test INTUITIVE, INSTANT, FRICTION-FREE",

    "description": "Replaces paper bills and payments with user-friendly services on the web and mobile. This accelerated the adoption of this payment system ",

    "pictuer": "45612d2e-8964-448e-80cd-7f422184b8e6\_resize.jpg"

}

* **Response Data**

Verify that the operation was completed successfully from the database.

1. Update existing Advantage:

* **HTTP Method:**

Put

* **API Request:**

<https://localhost:44318/api/Madadvantage>

* **Request Data**

 {

        "advantageId": 1,

        "advantage": "Test2 INTUITIVE, INSTANT, FRICTION-FREE",

        "description": "Replaces paper bills and payments with user-friendly services on the web and mobile. This accelerated the adoption of this payment system ",

        "pictuer": "45612d2e-8964-448e-80cd-7f422184b8e6\_resize.jpg"

    }

* **Response Data**

Verify that the operation was completed successfully from the database.

1. Delete existing Advantage:

* **HTTP Method:**

Delete

* **API Request:**

https://localhost:44318/api/Madadvantage/{id}

* **Request Data**

Check the database for an existing ID.

* **Response Data**

Verify that the operation was completed successfully from the database.

1. Searching for payment transactions during a specific interval:

* **HTTP Method:**

Post

* **API Request:**

https://localhost:44318/api/Paymenthistory/AggregatProfetsSearch

* **Request Data**

{

    "dateFrom": "2022-03-04",

    "dateTo": "2022-03-06"

}

* **Response Data**

[

{

"count": 1,

"sum": 5,

"billercategoryname": "Water",

"paymentDate": "2022-03-04",

"dateFrom": **null**,

"dateTo": **null**

},

{

"count": 1,

"sum": 2,

"billercategoryname": "Telecommunication",

"paymentDate": "2022-03-04",

"dateFrom": **null**,

"dateTo": **null**

},

{

"count": 1,

"sum": 3,

"billercategoryname": "Education",

"paymentDate": "2022-03-05",

"dateFrom": **null**,

"dateTo": **null**

}

]

1. Get All categories:

* **HTTP Method:**

Get

* **API Request:**

<https://localhost:44318/api/Billercategory/GetAllBillercategory>

* **Request Data**

None

* **Response Data**

[

    {

        "billercategoryid": 1,

        "billercategoryname": "Water",

        "pictuer": "5fd366be-acdb-40cb-941f-316941009120\_download (1).jpg",

        "deducted": 5,

        "billername": **null**

    },

    {

        "billercategoryid": 2,

        "billercategoryname": "Electricity",

        "pictuer": "6cee9894-7106-4425-944a-5bb01c8d3dfe\_R.jpg",

        "deducted": 2,

        "billername": **null**

    },

    {

        "billercategoryid": 3,

        "billercategoryname": "Telecommunication",

        "pictuer": "fdfab6c9-05f5-49a9-af48-e8f0733a1a1e\_OIP.jpg",

        "deducted": 2,

        "billername": **null**

    },

    {

        "billercategoryid": 4,

        "billercategoryname": "Education",

        "pictuer": "352208ac-7fa0-4b0c-afc4-870bb0ef99fa\_download.jpg",

        "deducted": 3,

        "billername": **null**

    }

]

1. Get All categories:

* **HTTP Method:**

Get

* **API Request:**

<https://localhost:44318/api/Billercategory/GetAllBillercategory>

* **Request Data**

None

* **Response Data**

[

    {

        "billercategoryid": 1,

        "billercategoryname": "Water",

        "pictuer": "5fd366be-acdb-40cb-941f-316941009120\_download (1).jpg",

        "location": "amman",

        "email": "water@gmail.com",

        "deducted": 5,

        "billername": **null**

    },

    {

        "billercategoryid": 2,

        "billercategoryname": "Electricity",

        "pictuer": "6cee9894-7106-4425-944a-5bb01c8d3dfe\_R.jpg",

        "location": "irbid",

        "email": "electric@gmail.com",

        "deducted": 2,

        "billername": **null**

    },

    {

        "billercategoryid": 3,

        "billercategoryname": "Telecommunication",

        "pictuer": "fdfab6c9-05f5-49a9-af48-e8f0733a1a1e\_OIP.jpg",

        "location": "amman",

        "email": "telcom@gmail.com",

        "deducted": 2,

        "billername": **null**

    },

    {

        "billercategoryid": 4,

        "billercategoryname": "Education",

        "pictuer": "352208ac-7fa0-4b0c-afc4-870bb0ef99fa\_download.jpg",

        "location": "amman",

        "email": "education@gmail.com",

        "deducted": 3,

        "billername": **null**

    }

]

1. Create new category:

* **HTTP Method:**

Post

* **API Request:**

<https://localhost:44318/api/Billercategory/GetAllBillercategory>

* **Request Data**

 {

        "billercategoryname": "Water-Test",

        "pictuer": "5fd366be-acdb-40cb-941f-316941009120\_download (1).jpg",

        "location": "amman",

        "email": "water@gmail.com",

        "deducted": 5,

        "billername": **null**

    }

* **Response Data**

Verify that the operation was completed successfully from the database.

1. Update existing category:

* **HTTP Method:**

Put

* **API Request:**

<https://localhost:44318/api/Billercategory>

* **Request Data**

  {

        "billercategoryid": 1,

        "billercategoryname": "Water-Test",

        "pictuer": "5fd366be-acdb-40cb-941f-316941009120\_download (1).jpg",

        "location": "amman",

        "email": "water@gmail.com",

        "deducted": 5,

        "billername": **null**

    }

* **Response Data**

Verify that the operation was completed successfully from the database.

1. Delete existing category:

* **HTTP Method:**

Delete

* **API Request:**

[https://localhost:44318/api/Billercategory/{id}](https://localhost:44318/api/Billercategory/%7bid%7d)

* **Request Data**

Check the database for an existing ID.

* **Response Data**

Verify that the operation was completed successfully from the database.

1. Login:

* **HTTP Method:**

Get

* **API Request:**

<https://localhost:44318/api/user/GetUserprofile/1>

* **Request Data**

 {

        "username": "noor",

        "password": "123456"

    }

* **Response Data**

Token for valid username and password, unauthorized for invalid username and password

1. Get User By ID:

* **HTTP Method:**

Post

* **API Request:**

[https://localhost:44318/api/user/GetUserprofile/{id}](https://localhost:44318/api/user/GetUserprofile/%7bid%7d)

* **Request Data**

Check the database for an existing ID.

* **Response Data**

{

    "userid": 1,

    "fullname": "Oday Alsanad",

    "phonenumber": "07895857",

    "email": "odayalsanad6374@gmail.com",

    "address": "irbid",

    "pictuer": "6af78a0f-443c-47d7-9282-358a1dc71fd8\_person\_2.jpg",

    "username": "Admin",

    "password": "123456"

}

# For You

* Test cases should include the user experience in all circumstances and environments.
* As a tester, use your creative thinking and critical eye to elicit as many test cases as possible to detect potential defects.
* In some cases, you may need to use the techniques we have learned to elicit test cases.
* Keep in mind the concepts and principles we learned in the quality assurance fundamental course.

# Project Programming Language and Technologies

* Visual Studio Program.
* Selenium Web driver.
* Microsoft Excel
* JMeter
* Postman
* BlazeMeter
* DevOps
* C# Programming Language.

# Project Uploaded Requirements

* Upload all works as a single zip file and ensure it contains:

1. selenium project.
2. JMeter test plane
3. All Excel files.
4. All HTML Reports

* Share the DevOps organization to [bayanalhassoun96@gmail.com](mailto:bayanalhassoun96@gmail.com)

# Deadline

The project must be submitted on the portal: on 3/10/2023.

# Customize Test case and Bug Templates

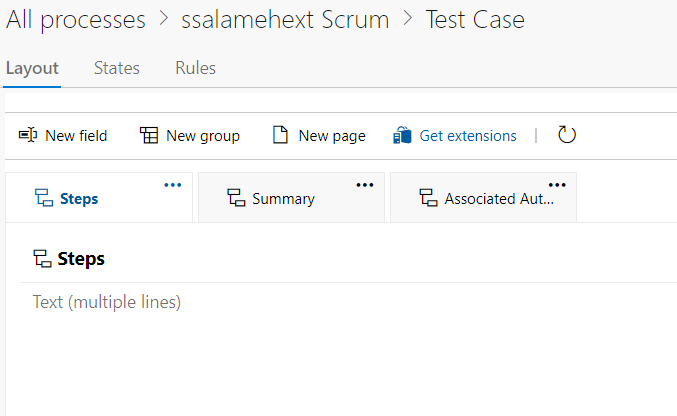
To customize the test case template:

* Add your epic, feature, and PBI under it.
* Add test case under PBI and click three dots then customize.

A screenshot of a computer

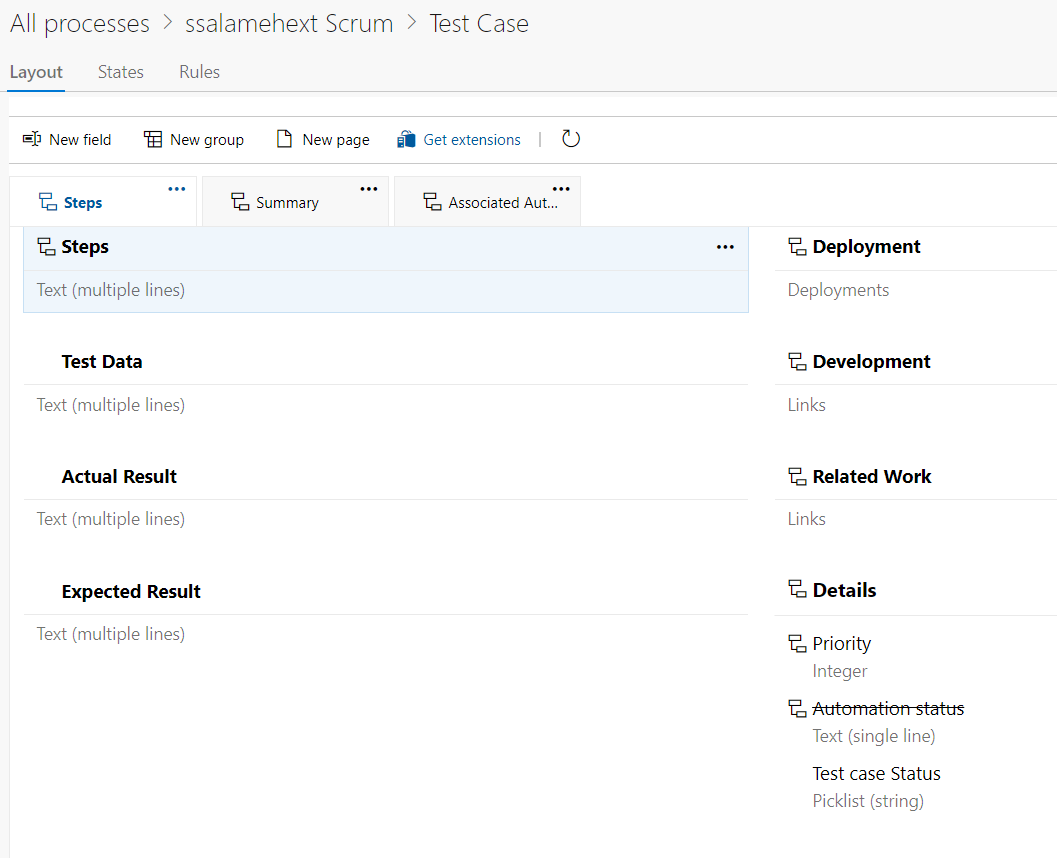
Description automatically generated

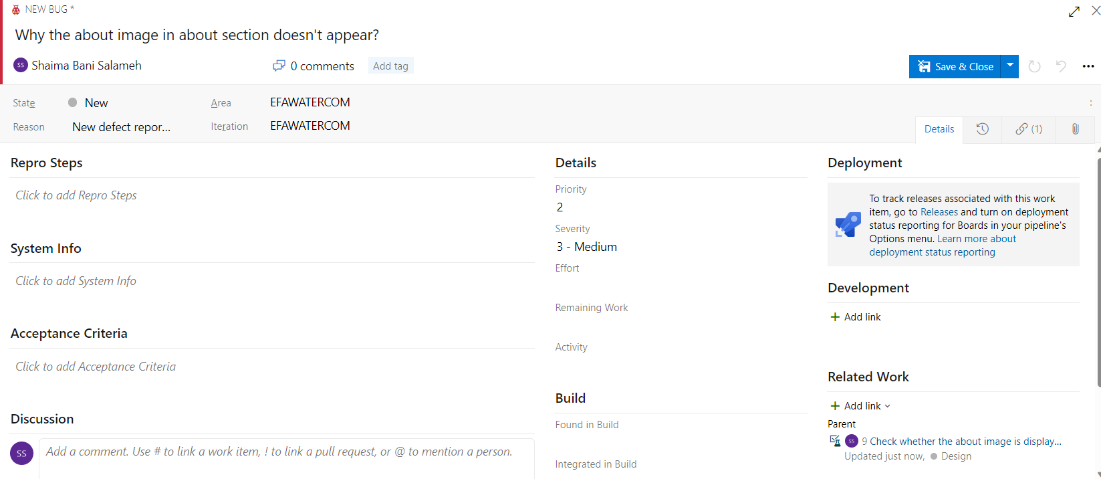
* Add new Field.



A screenshot of a computer

Description automatically generated

* Test case template will be like the following:
* Then, add bug on test case.
* The same, click on three dots then customize.



* Add new field.

A screenshot of a computer

Description automatically generated

* Bug template will be as the following.

