IN 2900 - Industry Based Project
Level 2

Strix - Multi-tenant Issue Tracking and Reporting System

Group Name - The Court of Owls

184025L	M.D.N. Chandrasiri
184038E	E.Y.A.R. Edirisinghe
184116R	W.P.C.P. Pathirana
184152X	W.A.D. Sandarekha
184157R	I.K.G.D.S.N.Senanayake

Faculty of Information Technology
University of Moratuwa

April 2021

Strix - Multi-tenant Issue Tracking and Reporting System

Group Name - The Court of Owls

184025L	M.D.N. Chandrasiri
184038E	E.Y.A.R. Edirisinghe
184116R	W.P.C.P. Pathirana
184152X	W.A.D. Sandarekha
184157R	I.K.G.D.S.N.Senanayake

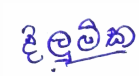
Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the partial fulfillment of the requirements of the Honours Degree of Bachelor of Science in Information Technology.

Supervised by: Dr. Priyanga D. Talagala
Faculty of Information Technology
University of Moratuwa

April 2021

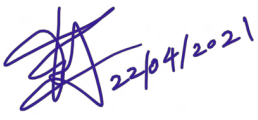
Declaration

We declare that this thesis is our own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Name of students	Signature of the students
M.D.N. Chandrasiri	
E.Y.A.R. Edirisinghe	
W.P.C.P. Pathirana	
W.A.D. Sandarekha	
I.K.G.D.S.N.Senanayake	

Date: 22nd April, 2021

Supervised by

Name of supervisor	Signature of supervisor
Dr. Priyanga D. Talagala	

Date: 22nd April, 2021

Dedication

We dedicate this dissertation to our supervisor Dr. Priyanga D. Talagala who was a constant source of inspiration throughout the project time period, Mr. Manujith Pallewatte who had faith in our abilities to deliver the desired project and to the academic staff and the family members who supported us throughout the process.

Acknowledgement

It is a great pleasure to present this section as a tribute to those who have always stood strong for us and acted as torchbearers for us. We offer our first and foremost thanks to our supervisor **Dr. Priyanga D. Talagala**, for pointing out all our shortcomings, providing a great deal of knowledge and supervision for the completion of our project. We are very happy with her excellent support, and the time she has given us. We are very grateful to our industrial mentor **Mr. Manujith Pallewatte**, COO / Co - Founder Avantrio (pvt) Ltd, for entrusting us with this project with great confidence and for his support during his precious time. We also would like to express our sincere gratitude to all our lecturers and instructors of the relevant departments of faculty of IT, university of Moratuwa, for their continued encouragement and guidance throughout the project completion period.

Abstract

Bugs and issues are a major problem consumers face when using software. No software product is perfect. Almost all the software consists of bugs or issues. But it does not prevent consumers from using it. The way to control bugs and issues is to let consumers report the bugs and issues they encounter to the developers. Then developers can identify the issues, fix them, and release a better version of the software. That is the path of software to its perfection. This is the moment where a bug tracker comes in. The task of a bug tracker is to make an intermediate platform between the user and the developer so they can communicate about the bugs properly and they can manage them properly.

Even though there are many bug trackers in the market, developers are still facing problems due to lack of customer engagement with bug trackers. As reasons, developers point out that existing bug trackers focus more on technical facilities rather than user-friendliness. That complexity forces customers to report their bugs just using communication platforms such as emails and messaging apps. This makes developer's work much harder as it is very hard to manage their workflow.

As a solution, we are developing our bug tracker to be more user-friendly, simple, and straightforward. It will be easier even for a non-technical person to engage with the system. The system has been developed using role based access control and there are 5 categories of external and internal as two main users. External users are customers and as a customer they can easily report a bug via a simple and user-friendly web user interface. After addressing customer engagement problems in the bug tracker we are focusing on developers. They are included in the internal user category. For developers, we are implementing essential facilities for sprint creating,reporting. As an internal user manager also can generate reports for the future work flow management. Thus, we are making this platform friendlier to both developers and customers compared to other existing bug tracking systems.

Table of Content

Declaration **3**

Dedication **4**

Acknowledgement **5**

Abstract **6**

Table of Content **7**

List of Figures **11**

List of Tables **12**

Introduction **1**

1.1 Introduction 1

1.2 Background and motivation 1

1.3 Aims and Objectives 2

1.4 Solution 2

1.5 Structure of the Report 3

1.6 Summary 4

Literature Review **5**

2.1 Introduction 5

2.2 Comparison between other systems 5

2.2.1 Comparison with other available software 8

2.3 Summary 9

Technologies used for the solution **10**

3.1 Introduction 10

3.2 Programming Languages 10

3.2.1 JavaScript 10

3.2.2 Python 12

3.3 Client-side technologies 13

3.3.1 React Library 13

3.3.2 Bootstrap 14

3.3.3 React hooks 15

3.3.4 Other Libraries 16
3.3.4.1 Axios 16
3.3.4.2 Formik and Yup 16
3.3.4.3 Material Table 16
3.3.4.4 Sweetalert2 16
3.3.4.5 React-icons 16
3.3.4.6 React-Router-Dom 16
3.3.4.7 React-beautiful-dnd 17
3.3.4.8 Styled-Components 17
3.3.4.9 Chart js and React-chartjs-2 17
3.3.4.10 React-datepicker 17
3.3.4.11 React-table 17
3.3.4.12 React-select 18
3.3.4.13 React-hook-form 18
3.3.4.14 Jspdf , jspdf-autotable, papaparse 18
3.4 Server side technologies 18
3.4.1 Django Framework 18
3.4.2 Django rest framework 19
3.4.3 Psycopg 20
3.5 Summary 20
Approach **21**
4.1 Introduction 21
4.2 Users of Strix Bug Management System 21
4.3 System inputs and outputs 22
4.4 Process within the system 23
4.5 Software Development Process Model used -Agile Scrum 23
4.6 User Stories According to Agile Scrum 25
4.6.1 Authentication and Authorization 25
4.6.2 Bug capture Log 26
4.6.3 Bug management system 26
4.6.4 Bug solution pool 28

4.6.5 Bug reporting system 28
4.7 Project Management Plan 29
4.7.1 Work breakdown structure (WBS) 29
4.7.2 Resource Allocation 30
4.8 Summary 30
Analysis and Design 31
5.1 Introduction 31
5.2 Analysis 31
5.2.1 Authentication and Authorization 31
5.2.2 Bug Capture Log 33
5.2.3 Bug Management System 34
5.2.4 Bug Solution Pool 36
5.2.5 Bug Reports 36
5.3 Design 37
5.4 Summary 58
Implementation 59
6.1 Introduction 59
6.2 Implementation Process 59
6.2.1 Frontend Implementation 60
6.2.2 Backend Implementation 60
6.2.3 Database Implementation 61
6.2.4 Working design of the system 62
6.3 Layered Architecture 74
6.4 Summary 74
Evaluation 75
7.1 Introduction 75
7.2 Testing 75
7.3 Summary 84
Conclusion 85
8.1 Introduction 85
8.2 How our system differs similar systems 85

8.3 Further Development and Limitations 85
8.4 Summary 86
Reference **87**
Individuals Contribution to the Project **89**
Name of student: M.D.N. Chandrasiri 89
Name of student: E.Y.A.R. Edirisinghe 91
Name of student: W.P.C.P.Pathirana 92
Name of student: W.A.D. Sandarekha 93
Name of student: I.K.G.D.S.N.Senanayake 94
Appendix **95**
Appendix A 95
Appendix B 95

List of Figures

Figure 5.3.1 – Use case Diagram 35
Figure 5.3.2 – Activity Diagram for Login & Password Reset 35
Figure 5.3.3 – Activity Diagram for Authentication & Authorization 36
Figure 5.3.4 - Activity Diagram for Bug Capture Log 37
Figure 5.3.5 – *Activity Diagram for Bug Management System* 38
Figure 5.3.6 – Activity Diagram for Bug Reporting 39
Figure 5.3.7 – Activity Diagram for Bug Solution Pool 40
Figure 5.3.8 – Activity Diagram for Bug Solution Pool Approvals 41
Figure 5.3.9 - Sequence Diagram for Login and Password reset 42
Figure 5.3.10 - Sequence Diagram for Authentication and authorization 43
Figure 5.3.11 - Sequence Diagram for Bug Capture Log 44
Figure 5.3.12 - Sequence Diagram for Bug Management System (Manager) 45
Figure 5.3.13 - Sequence Diagram for Bug Management System (QA) 46
Figure 5.3.14 - Sequence Diagram for Bug Management System (Developer) 47
Figure 5.3.15 - Sequence Diagram for Bug Solution Pool Approvals 48
Figure 5.3.16 - Sequence Diagram for Bug Solution Pool (QA, Developer) 50
Figure 5.3.17 - Sequence Diagram for Bug Solution Pool (Manager) 51
Figure 5.3.18 - Sequence Diagram for Bug Reporting 52
Figure 5.3.19 - Component Diagram 53
Figure 5.3.20 - Class Diagram 54
Figure 5.3.21 - ER Diagram 54
Figure 6. 1 – Layered Architecture 70

List of Tables

Table 2. 1 8
Table 7. 1 83

Chapter 1

Introduction

1.1 Introduction

This report is for the final evaluation of the “Strix bug tracking” system. This chapter sets the background for the project and provides an introduction to the report.

The rest of the chapter is organized as follows. Section 1.1 provides the background and the motivation for the project. Section 1.2 provides the aims and objectives of the project. Section 1.3 provides the proposed solution in terms of users, input, output, process, technology, and features.

1.2 Background and motivation

A bug tracking system is a tool that makes a developer's process easier and it should not make the debugging process and issue solving process harder for the developer. However, most of the existing bug trackers do not seem to fulfill that requirement due to various reasons. Most of the existing bug trackers provide tons of features for developers. During the initial discussions with the developers and field experts, it becomes obvious that customers most likely are not using bug trackers to report their issues. When we look through the customer's perspective in existing bug tracking systems, the issue-creating process (the process customers have to go through when they are reporting about an issue or a bug) is too complicated for a person who is not familiar with the technology. When we are considering developers we can further categorize them into Developer, Manager, QA, and Manager in terms of their role in the development process. Each of these roles has specific requirements in a bug tracking process (further discussed in Chapter 4.2). Those requirements also needed to be fulfilled without making the process more complicated.

1.3 Aims and Objectives

Aims

- Develop a user friendly, efficient system to bring all user roles to one platform and to solve issues regarding a selected product with the use of latest technologies.

Objectives

- Critical review of bug tracking systems.
- In depth study of latest developments in technologies such as
    - Reactjs
    - Django
    - Django Rest
    - Postgresql
- Develop a separate working environment for each user role.
- Providing a knowledge pool for developers.
- Evaluation of the proposed solution.
- Preparation of final documentation.

1.4 Solution

As mentioned in Section 1.1 most of the existing bug trackers do not focus on developers and also for the customers. Hence we developed the system from frontend react js and backend python based django technology and there are 5 categories of external and internal as two main users. Customers are included as external users. QA, developers and managers are included as internal users. Those categories should log in to the system under role based access control. To summarize the system requirements ,the process involves external users logging into the system and reporting a bug. The manager then reviews it and creates a timeline as Sprint. Then the developers assigned to those bugs can fix the problems. Finally QA reviews them. Through this system , customers can resolve their software errors easily. This can also be easily done if managers need to generate reports to provide feedback on past or current performance, and analyze the workflow management of their company. To make those things easier , we introduced and implemented

- Kanban Board system
- Sprint creation facility
- Easy reporting tools
- Proper bug solution pool as a knowledge repository for developer

To make the bug reporting process easier for customers

- Simple interface
- Ability to view their related projects
- Screen capturing and Screen recording facility was implemented

Our main focus was on making the bug reporting process easy to the customer and reducing additional effort in managing the bug tracking system to the developer. Thus, more quality and more effective software was developed with reduced time wastage and constantly updating customer feedback.

1.5 Structure of the Report

Chapter 2, entitled review of other's work, describes how our system differs from existing systems. Chapter 3 is based on technologies adapted to our solution. We have explained it well. Coming to Chapter 4, we wrote about our approach to the problem defined in terms of inputs, outputs and processes. Chapter 5 shows the design diagrams under the heading Analysis and Design. For each sub-module we have clearly shown the relevant diagrams. Chapter 6 also describes how the implementation has been done so far using layered architecture. Finally, we mentioned in Chapter 7 the complete system testing according to the testing methodologies. At the end of the report we mentioned the references we used and the individual contribution of our team to the project.

1.6 Summary

This chapter gives a brief introduction to the system and the structure of this document. It will be described in more detail in the following chapters.

Chapter 2

Literature Review

2.1 Introduction

This chapter summarizes the background information of the project and the state of the art. Here we compare several other approaches for the same techniques related to issue tracking to identify the merits and demerits of the existing systems.

2.2 Comparison between other systems

In this chapter, we mainly consider the Jira platform which was designed by Atlassian Corporation Plc in 2002 to manage agile and scrum practices, organizing project tasks and capture and record software bugs, etc [1].

Jira has several platforms such as

- Jira Core [2]
- Jira Software [3]
- Jira Service Desk [4]

As best agile practices there are both similarities and differences between our system (Strix) and Jira [5].

Considering them, based on a literature review we have listed few similarities in Jira with our proposed system,

- Powerful agile view

  Agile views mainly focus on flexibility, cross-functional aspects with collaboration, and continuous improvement of a particular task which can be seen in both of the systems.

- Kanban boards
    - Kanban boards are used to visualize the workflow especially in solving bugs. There are few lanes to display current progress with sticky notes like, ‘To do’, ‘In Progress’, ‘Review’, and ‘Done’. Workflow is visible in both systems as well.
- Detailed Reports
    - Reports are essential factors in bug reporting and project management to summarize the work
    - Reports like
        - Sprint summary
        - Project and developer timesheets
        - Monthly bug summary and performance charts are the major reports that can be seen.

Although Jira caters to many facilities it still has major drawbacks from the user's perspective. We have listed down those major limitations of Jira as follows.

- Jira is hard to set up and takes some considerable time to get used to
    - Since 2002 the time Jira was invented, it has not evolved that much making it so difficult to set up and use [6]
    - As the same, there are a large number of dashboards, lists, screens, and workflow options which has become such a difficult task for a new Jira user.
    - Along with that, each user has to learn Jira query language to do simple searches to search and find tasks and issues [7].

- Complicated UIs with a lot of extra settings and additional features
    - A major complaint of users is that there are so many extra settings and additional features which are difficult to understand by a new user.
    - There are several complaints from Jira long term users that UI/UI (User Interface and User Experience has been inconsistent over the years with updates [14].

- Jira does not have its built-in timeline views for its users
    - Since every issue or project needs to be done according to a timeline with constraints, time management is a very essential part of both Jira and Strix. But In Jira, there is no in-built timeline functionality. If we need one we have to import a time tracker with necessary requirements by falling under payment methods [8].
- Collaboration features are not available in Jira for free
    - Jira does not have a proper commenting and collaboration system that is freely offered [14]
    - User has to pay an additional amount to get the experience of them [15]
- Jira is an expensive tool for a small company that is growing slowly
    - Even though Jira has so many separate module different modules like Time tracking, Timeline, Notepad, almost all of these features are available as paid versions [9]
    - As of October 31st, 2020, Jira has 2 pricing plans with free trials
        - Simple Agile Management system for maximum of 10 members
        - Standard plan ($7/user/month) -250GB cloud, storage 5000 users
        - Premium plan ($14/user/month)- Unlimited storage,5000 users

2.2.1 Comparison with other available software

Table 2.1 provides a comparison of existing software tools for issue tracking[10, 11, 12]

	Bugzilla	Jira	Strix
Follow powerful agile practices	❌	✔	✔
Use Kanban boards to visualize the workflow	❌	✔	✔
Workflow management	✔	✔	✔
Detailed Reports	✔	✔	✔
Used for project management software Note: Strix uses agile methods inside issue tracking instead	✔	✔	❌
Issue tracking software	✔	✔	✔
UI/ UX less complexity	✔	❌	✔
Less difficult to set up and get used to software	✔	❌	✔
A built-in time tracker to check current progress	❌	❌	✔
Collaboration features • Comments • Tag team members, get notifications	 ❌ ❌	 Paid ❌ ❌	 ✔ ✔
Query Searching	✔	JQL[13] ✔	❌
Notepad apps to write down ideas, thoughts, rough plans	❌	❌	✔
Pricing methods	Free	✔	Free

Table 2. 1

Even Though some software tools use query languages for searching and filtering issues, as for the first phase, we did not include query language processing in our proposed system. But we suggest implementing an unique query language which includes advanced searching and filtering in our issue tracking system under the further development section.

2.3 Summary

In this chapter, we compare our system with the Jira platform and describe the similarities such as

- Powerful agile views
- Kanban boards
- Detailed reports

And Strix provides solutions to the limitations of the existing approaches by,

- Providing user-friendly UIs instead of complex interfaces
- Easy to set up even for a non-technical user solving all the constraints
- Providing detailed reports and timesheets free for tracking time and developer performance
- Adding comments section for more collaboration between users

Chapter 3

Technologies used for the solution

3.1 Introduction

The Strix bug management system consists of three major components,

1. Web-based user interface (frontend)
2. Rest API based backend
3. SQL based database

According to the company's needs, we considered many technologies for each of the major components. After a successful analysis, we chose major technologies for each of the components. In this chapter, the technologies used up to now are discussed.
The rest of the chapter is organized as follows. Section 3.2 provides a description of the programming languages used in the system. Section 3.3 provides details about the client-side technologies while the Section 3.4 provides details about the server-side technologies.

3.2 Programming Languages

3.2.1 JavaScript

JavaScript is one of the most powerful languages to develop web based applications. We used JavaScript as our main programming language in the frontend development and the framework that we used in the frontend development is fully dependent on JavaScript as well.

We use JavaScript due to the below reasons,

1. JavaScript is a cross-browser compatible language.

One of the main user roles in our system is the customers who do not have much knowledge about web browser technologies. Hence it is better to use cross-browser

compatible language when developing a frontend for the system. It allows us to run our system in lots of web browsers out there.

2. JavaScript comes with lots of libraries and frameworks.

There are lots of frontend development frameworks out there that are based on JavaScript. Hence it is easy to find well developed documentation for JavaScript frameworks. As well as there are lots of free libraries that can be used along with JavaScript. This helps us to develop fully functionalized user interfaces in a short time period because those libraries provide an abstract implementation of the above functionalities.

3. Simplify Complex modern web application development.

JavaScript is an interpreted programming language. Still, it simplifies the development of complex web applications by allowing the developers to simplify the application's composition. The developers can use JavaScript libraries to create DOM boundaries. This shadow DOM further makes web browsers deliver documents with widely used HTML tags like select, div, and input.

4. Responsive web design

Nowadays most developers do responsive web design to make a website accessible and work well across multiple browsers and devices like smartphones and desktops. Responsive web design enables the developers to optimize the site for both computers and mobile devices using the same code. The developers have to combine CSS3, HTML5, and JavaScript to make the web pages responsive. The developers can use JavaScript to optimize web pages for mobile devices.

3.2.2 Python

Python is a powerful programming language in complex computations. We use python as our main programming language in the backend development and the framework that we used in the backend development is also fully dependent on Python. We use python due to below reasons,

1. Python has many web application frameworks.

Python's countless resources come in many forms, including a wide variety of web application frameworks like Django, Flask, Bottle, Tornado, Hug, CherryPy. As well as all of these web frameworks have a wide variety of libraries.

2. Fastest growing major backend language.

According to the statistical reports, Python is the fastest growing backend language along with its vast use in complex computations applications. Python has garnered increasing traffic in Stack Overflow's site since late 2011, as well as in their Developer Survey. Back in 2013, Python ranked 6th in popularity with just under 22% of survey takers having declared to use it. By 2017, it had finally risen to 5th place and, in 2020, it has reached 3rd place among the most loved languages for Stack Overflow users.

3. Python is easy to work with and easy to read.

Among the lots of programming languages python is one of the easiest languages to both read and write. Its simplicity makes the size of codes much smaller.

4. Python is great for machine learning.

In this new era of technology, AI plays a major role in every application. Hence it is better to use language like python which has great support for various machine learning techniques.

3.3 Client-side technologies

3.3.1 React Library

There are many frameworks and libraries available for frontend development. Each framework and library have their own pros and cons. Among them React is one of the most popular and widely used libraries for frontend development and it is not a framework. React is an **open-source JavaScript library** used for frontend development, which was developed by Facebook. Its component-based library lets you build high quality user-interfaces for web apps. This library allows us to place HTML code inside JavaScript and it works with Virtual DOM. We chose to react over other libraries and frameworks due to the below reasons,

1. React enables developers to reuse components.

In react, normally our application comprises components. Ideally, in the initial stage, we will be having few components but as we go on writing more and more components we have to manage them according to a file structure. Each component in React has its own logic. There are some situations where we need to use the same component again and again. Hence with the aid of React we can easily reuse components which makes greater reusability. For example, in the Strix bug management system we use a navigation bar and a footer on every web page. But it is not necessary to build navigation and footer components every time when we need a webpage.

2. It's easier to learn for developers.

When we consider the strengths and weaknesses of the team and the time duration that we have to finish the project, it is quite hard to develop web applications while learning but react is easier to learn while developing.

3. React has a well-established vibrant ecosystem of developer tools.

React consists of a rich and vibrant ecosystem. Developers can find dozens of ready-made customizable charts, graphs, and other documentation tools that allow them to build a web app in less time. When we developed our web app we faced lots of

issues at every stage. But it was easy to find the issue and resolve it using react documentation tools.

3.3.2 Bootstrap

Bootstrap is a surprisingly powerful and effective tool for developers building responsive websites and web applications. We chose bootstrap due to the below reasons,

1. Allows building fully responsive websites without any hassle.

Responsive web design enables the developers to optimize the site for both computers and mobile devices using the same code. With the aid of bootstrap, we could easily make the Strix bug management system responsively. Fluid grid-based layouts, Flexible images, and media and CSS media queries helped to make a fully responsive web user interface.

2. Built-in pre styled components

Bootstrap allows us to use their pre-styled components when building our system without writing them from scratch. Writing style components is time-consuming and when it comes to responsive web design it is extremely difficult to do in a short period of time. But bootstrap's pre-styled components made the flow easy.

3. Ease of integration

When it comes to frontend styling frameworks there are many. Each of them has unique features when compared to others. Bootstrap is the easiest framework that can be implemented with raw HTML tags and its future compatibility is also high.

3.3.3 React hooks

React hook is not a framework or a library. It is a new addition to react in version 16.8 that allows you to use state and other react features, like lifecycle methods, without writing a class.

When we develop a user interface we have to develop two types of user interfaces.

1. State user interfaces - User interfaces that contain details to be submitted (Ex: Login Form of Strix bug management system)

2. Stateless user interfaces - User interfaces that do not contain details to be submitted

(Ex: Ticket view of the Strix bug management system)

In the past version of React (<16.8) we have to use two types of approaches to develop each of the methods.

1. State - Class based components

2. Stateless - Function based components

It is time consuming to use a class based component and readability is also less. But with the aid of React Hooks we can simply convert class base components into react functions. It significantly simplified our code and made it more readable. As well as it is easy to extract and share stateful logic in our apps. In the Strix bug management system, we used react hooks to simplify our code base.

3.3.4 Other Libraries

3.3.4.1 Axios

The Axios is used to make HTTP requests to the backend. Axios used promise-based request handling which helps to make requests in asynchronized manner and It is easy to handle errors with Axios as well.

3.3.4.2 Formik and Yup

Formik simplifies the process of form submitting and remapping form data. Yup simplifies the process of form validation and error control in forms. The Formik and Yup combination to achieve more realistic forms.

3.3.4.3 Material Table

Material-Table is a JavaScript library that has many functionalities to manipulate table data. It has built-in search, sort, filter, and pagination functionalities which is very helpful in table creation. It is easy to use.

3.3.4.4 Sweetalert2

Sweetalert2 is a JavaScript library that supports a vast number of alert services including Success, Error and Danger indication, etc.

3.3.4.5 React-icons

The system uses many icons to visualize certain attributes and items. React-icons library is a collection of icons that contains multiple icon libraries. It removes the need to install several libraries in order to obtain appropriate icons.

3.3.4.6 React-Router-Dom

React-router-dom is a JavaScript based library that was developed especially for react. It contains the DOM bindings for React Router. In other words, the router components for websites. It is very much useful when it comes to the routing of a website.

3.3.4.7 React-beautiful-dnd

React-beautiful-dnd library is specifically built to support drag and drop behavior. The library supports movements between lists which helps with the work state management of the tickets in the system. Furthermore, the library supports conditional dragging and conditional dropping. For the time being, the system only uses conditional dropping to manage workflow among user roles. Conditional dragging, on the other hand, would be useful if the company wants to restrict the workflow of users in a specific user role.

3.3.4.8 Styled-Components

The Kanban board entails specific components which are not used anywhere else in the system. Hence the styled components are used to create unique components in this particular submodule.

3.3.4.9 Chart js and React-chartjs-2

React-charts-2 is a library that provides various charts in react applications. This is a wrapper to the Chartjs library and because of that, we need to add both packages. Reasons for choosing these packages are that React-chartjs-2 makes it easy to display Charts.js charts in React applications. Using this library we can display different types of charts, configure them, and style as needed.

3.3.4.10 React-datepicker

For the filtering option by date selected option is highly required. With react-datepicker it was easy to handle the date selection by using various properties like selecting time, disabling required dates and it is user friendly and easy to implement

3.3.4.11 React-table

According to the npm trends popularity of this library is high among the react users. Since this library is constantly updated and issues are less it has a higher rating compared to other libraries. Features like lightweight (3 kb), column definition driven, client side pagination and sorting.

3.3.4.12 React-select

When selecting developers from the dropdown, the user needs to select multiple or single developer options, and rather than using a normal dropdown this library provides animated components using wrappers and displays in a user friendly way.

3.3.4.13 React-hook-form

To handle POST request by requesting body data in React without using regular inputs and attaching onClick listeners, React-hook form relies heavily on uncontrolled inputs which handle validation as well as registering inputs using controllers

3.3.4.14 Jspdf , jspdf-autotable, papaparse

In these timesheets, we need to export those tables in pdf and CSV formats. This jsPDF plugin adds the ability to generate PDF tables using Javascript data by parsing HTML tables. Papa Parse is the fastest in-browser CSV parser for js. Through this, we can parse CSV files directly. Here I used this library to parse CSV files to be exported just like pdf.

3.4 Server side technologies

3.4.1 Django Framework

Django is a high level python web framework mainly used in backend development that encourages rapid development and clean pragmatic design. It takes care of much of the hassle of web development and it is free and open source. We chose Django backend framework due to the below reasons,

1. Django allows fast backend development.

Our system requirements were not completely defined in the initial stage. Hence we decided to follow the Agile Software Development methodology. According to agile, we had to develop component by component completely in different stages. With the support of Django, it was much easier to develop backend functionalities fast and the rebuild was also fast.

2. Django has powerful authentication and authorization support.

Our system consists of five main user roles and the whole system works according to a user role. Hence it is necessary to have role based access control (RBAC) within our system. With the aid of Django authentication support, we could easily manage it. For example,

- CSRF token validation while handling POST requests.
- Django auth token to determine currently logged-in users.

3. Powerful ORM support.

In the Strix bug management system, we used a Postgresql database which is based on SQL to build our database. When it comes to database queries we didn't want to write SQL queries directly because Django has a powerful ORM which maps python commands into SQL queries.

4. High compatibility with cutting edge technologies.

Artificial intelligence is the most popular technology in the present.

3.4.2 Django rest framework

The biggest reason to use Django REST Framework is that it makes serialization so easy. In Django, we define our models for our database using Python. While we can write raw SQL, for the most part, the Django ORM handles all the database migrations and queries and sometimes we need to send payloads to the frontend or wherever a request is coming. Those payloads can be part of a table in the database or a whole table depending on the request. A common format of a payload is usually JSON format. In that case, we need to explicitly create a payload structure, query the database and filter relevant data and then send it. But with the aid of the Django rest framework, we can easily convert our database table into a JSON file without wasting lots of code lines.

3.4.3 Psycopg

Pyscopg is a python library that acts as a driver for the Postgresql database. Psycopg is used to connect our Strix database which was developed using Postgresql to the Django backend.

3.5 Summary

This chapter described the main technologies used when implementing the Strix bug management system. Mentioned languages and technologies will not change in the future but new libraries will be added to the system according to the user's needs.

Chapter 4

Approach

4.1 Introduction

Along with all the modern technologies which were discussed in Chapter 3, in this chapter we discuss how to utilize them with reference to users, outputs, inputs, and technologies used in our system.

Strix, the bug management system has come with a new approach that can be easily used by non-technical users when compared to other related software and our aim is to provide a proper platform for users to report their issues related to products.

The rest of the chapter is organized as follows. Section 4.2 provides details about the users of the system. Section 4.3 provides a brief description of the system inputs and the outputs. Section 4.4 describes the process within the system. Section 4.5 explains the software development process model used by the development team. Section 4.6 describes the user stories of each module of the system. Finally, in section 4.7 project management plan of the development team is described.

4.2 Users of Strix Bug Management System

With the role-based access control (RBAC), there are 5 categories within 2 major users as follows

- External users
  - Customer - the person who add issues to the system
- Internal users
  - Admin - who creates all the roles and permissions.
  - Manager - who tracks all the work done by developers and QA and creates sprints.
  - Developers - the person who is responsible for resolving a particular issue within a time range.

- Quality Assurance - the person who reviews the issue and mark it as done

4.3 System inputs and outputs

In this part, we discuss the various inputs and outputs to and from the system by different users within the system. These inputs and outputs are categorized according to the users of the system and list down below.

User Type	User Role	Input	Output
External user	Customer	Login credentials	User access to the system
		Form details - Issue name - Issue description - Issue tags - Screenshots - Screen records - Other files	Issue Created
		Comments	Comment added to the ticket
Internal user	Developer	Login credentials	User access to the system
		daily effort on a ticket	Data added to timesheet
		Comments	Comment added to the ticket
	QA	Login credentials	User access to the system
		Comments	Comment added to the ticket
		Corrected tags	Changed data in the system
		Review status	Changed status in the system
	Manager	Login credentials	User access to the system
		Sprint details	Sprint added to the system
		Total effort of a ticket	Data added to the system

	Admin	Login credentials	User access to the system
		User details	User added to the system
		Project details	Project added to the system

4.4 Process within the system

After assigning roles by the super admin, the customer opens an issue and stores the bug details as a ticket under the relevant project. The manager then looks at the incoming ticket details and creates a sprint to resolve them. It is the responsibility of the developers to add that sprint information to their schedule and complete it within the appropriate time frame. In the process, Kanban boards are updated, and QA also has a key role to play in reviewing issues and updating their status. Managers also have the ability to generate reports and provide feedback on past or current performance, and analyze workflow management, if needed later or during discussions. In addition, everyone except managers can add their comments to a particular ticket as needed.

4.5 Software Development Process Model used -Agile Scrum

In this system, our initial idea was to follow the waterfall model by gathering all the requirements related to all the modules. But in the meantime, our company needs us to implement the first stage of the bug management system as the client has other plans to make it an open source project for future versions. Therefore they stated 5 modules to complete by following the agile practices. We divided each module within our team after considering the difficulty level of each module and started to develop them by component wise. (Ex: Components - homepage for each user role, login page, database, etc.) Each component covers one or more user stories.

Since our team is not well versed in software development, using agile provides us an opportunity to learn required technologies with each sprint. Given that we complete a component in each sprint, we have the ability to test each component of its functionality.

Our team follows a thirty day sprint cycle and covers at least one component by the end of it. Work of the previous day, plans of the current day, and issues each member faced during the previous day was discussed and resolved during daily team meetings. We took meetings every other week with the supervisor and as needed with the mentor. Following the feedback of the mentor and the supervisor, we make corrections to the software.

4.6 User Stories According to Agile Scrum

4.6.1 Authentication and Authorization

- As an administrator, I want to log into the admin home, so that I can navigate to every other function.
- As an administrator, I want to reset my password, so that I can give a new password.
- As an administrator, I want to create external user accounts, so that I can add newly added external users to the system.
- As an administrator, I want to create internal user accounts, so that I can add newly added internal users to the system.
- As an administrator, I want to create projects, so that I can add newly created projects to the system.
- As an administrator, I want to assign users to a particular project, so that I can provide access to the users to a particular project.
- As an administrator, I want to update external user accounts, so that they have accurate details.
- As an administrator, I want to update internal user accounts, so that they have accurate details.
- As an administrator, I want to update project details, so that they have accurate details.
- As a Manager, I want to reset my password, so that I can give a new password.
- As a Manager, I want to log into the system, so that I can manage the system.
- As a QA, I want to log into the system, so that I can review issues.
- As a QA, I want to reset my password, so that I can give a new password.

- As a Developer, I want to reset my password, so that I can give a new password.
- As a Developer, I want to log into the system, so that I can solve issues.
- As a Customer, I want to log into the system, so that I can report an issue.
- As a Customer, I want to reset my password, so that I can give a new password.

4.6.2 Bug capture Log

- As a customer, I want to view related projects, so that I can go through each project and file a ticket related to an issue.
- As a customer, I want to create a new issue, so that I can add a new ticket to my existing project.
- As a customer, I want to take a screenshot of the particular part of the system, so that I can upload it with the form which I want to add as the ticket.
- As a customer, I want to upload files and record the screen, so that I can attach them with the form in the process of generating a ticket.
- As a customer, I want to comment on tickets, so that I can add more details about my issue for better communication.

4.6.3 Bug management system

- As a manager, I want to view the issue backlog, so that I can access issues.
- As a manager, I want to sort and filter issues in the backlog, so that I can access issues that match the conditions.
- As a manager, I want to view the sprint backlog, so that I can access sprints.
- As a manager, I want to create sprints and customize its due date so that I can add issues to it.
- As a manager, I want to add issues to the sprint so that developers and QAs can work on them.
- As a manager, I want to pin sprints to the sidebar, so that I can access certain sprints easily.
- As a manager, I want to end a sprint, so that I can update the status of the sprint.
- As a manager, I want a search option for issues and sprint backlogs so that I can search for a specific issue or a sprint.

- As a QA, I want to view the issue backlog, so that I can access issues.
- As a QA, I want to change tags of the bug if necessary so that developers get the correct idea of the issue.
- As a QA, I want to view the sprint backlog, so that I can access sprints.
- As a QA, I want to pin sprints to the sidebar, so that I can access certain sprints easily.
- As a QA, I want to add comments to the issue, so that I can get a better understanding of the issue.
- As a QA, I want to review issues which are in the “Review” lane of the kanban board, so that I can make decisions regarding which lane to move the issue into.
- As a QA, I want to move issues from “Review” lane to “Done” lane, so that the issue is marked as completed.
- As a QA, I want to move issues from “Done” lane to “Review” lane, so that I can re-review the issues that I mistakenly identified as completed.
- As a QA, I want to move issues from the “Review” lane to “In progress” lane, so that developers can work on them again.
- As a QA, I want to mark issues for Bug Solution Pool, so that the manager can approve or reject them.
- As a developer, I want to view the Issue backlog, so that I can access issues.
- As a developer, I want to add comments to issues, so that I can get a better understanding of the issue.
- As a developer, I want to view the sprint backlog, so that I can access sprints.
- As a developer, I want to pin sprints to the sidebar, so that I can access certain sprints easily.
- As a developer, I want to attach files to the issue, so that customers can view them.
- As a developer, I want to add time spent on an issue, so that the company can track my working hours.
- As a developer, I want to move issues from the “Open” lane to the “In progress” lane, so that I can work on it.

- As a developer, I want to move issues from the “In progress” lane to the “Open” lane, so that someone else can work on it.
- As a developer, I want to move issues from the “In progress” lane to the “Review” lane, so that a QA can review it.
- As a developer, I want to move issues from the “Review” lane to the “In progress” lane, so that I can rework on issues that I mistakenly moved to the “Review” lane.
- As a developer, I want to mark issues for Bug Solution Pool, so that the manager can approve or reject them.

4.6.4 Bug solution pool

- As a manager, I want to approve or reject issues that developers have marked for the Bug Solution Pool so that developers and QAs can view them in the Bug Solution Pool.
- As a manager, I want to view issues and previously added comments, so that I can understand the issue.
- As a manager, I want to search for issues, so that I can access the necessary issues.
- As a QA, I want to view issues and previously added comments, so that I can understand the issue.
- As a QA, I want to add new comments.
- As a QA, I want to search issues, so that I can access the necessary issues.
- As a developer, I want to view issues and previously added comments, so that I can understand the issue.
- As a developer, I want to add new comments.
- As a developer, I want to search for issues, so that I can access the necessary issues.

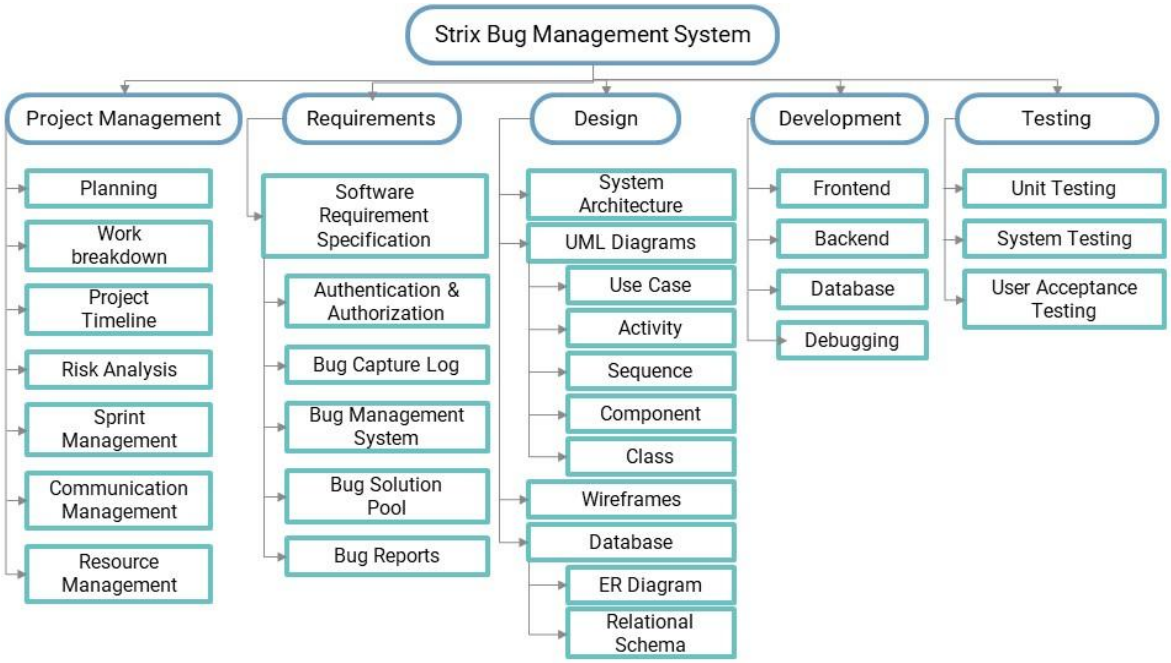
4.6.5 Bug reporting system

- As a manager, I want to view and download the “Project/Developer timesheet”, so that I can get information about issues worked on categorized by the developer and project along with the number of hours taken.

- As a manager, I want to view the summary of the sprints, so that I can see progress on multiple sprints.
- As a manager, I want to view the table “Monthly bug summary, so that I can see issues grouped by month of reporting.
- As a manager, I want to view a bar chart of the total number of bugs reported, so that I can see how many bugs have been reported in each month.
- As a manager, I want to view a stacked bar chart on projects, so that I can see total bugs assigned, total resolved issues, and total in progress along with the month.
- As a manager, I want to view a stacked bar chart from the developers' perspective, so that I can see total bugs assigned, total resolved issues, and total progress along with the month.
- As a manager, I want to view developer performance in a pie chart, so that I can see the resolved and in-progress tickets.
- As a manager, I want to view the developer's average effort in a bar chart, so that I can see the average hours spent on an issue along with the month.
- As a manager, I want to view the full bug summary, so that I can see total bugs reported, total projects with active bugs, and bug completion as a percentage.

4.7 Project Management Plan

4.7.1 Work breakdown structure (WBS)



4.7.2 Resource Allocation

Resource	**Status**
Team	There are five second year undergraduates who are working on the project. Three out of five members have experience in React development. One member has experience in the Django framework.
Time	Each member will put in 20 hours of effort into each month considering learning, development, and testing aspects. The project development and testing will go on for five months.
Equipment	The project will be done on the personal computers of team members.
Storage	During the development phase, the database will be hosted on Heroku.

	At the implementation phase, the database will be hosted on the client's cloud storage.
Monetary	The project will be done for academic purposes and thus no payment will be made.

4.8 Summary

In this Chapter, we discussed our approach to the problem defined in terms of inputs and outputs, and process. As the same, we describe the software development process model which we followed in this project.

Chapter 5

Analysis and Design

5.1 Introduction

The analysis of the Strix Bug Management System is carried out to understand the fundamental and non-fundamental requirements, to understand the most suitable software process model, to create the most effective plan, and to reduce risks that occur while developing the web application.

Various UML diagrams such as Use Case, Activity Diagram, Component Diagram, and Sequence Diagram have been used to visualize the system. An Entity Relationship Diagram has been developed to visualize the database of the system. System Requirement Specification and Project management plan are linked under Appendix A and B, respectively.

The rest of the chapter is organized as follows. Section 5.2 provides an analysis of the system modules. Section 5.3 consists of the design diagrams of the system.

5.2 Analysis

The application has been divided into five sub modules. (Eg: As given in Figure 5.1)

1. Authentication and Authorization
2. Bug Capture Log (BCL)
3. Bug Management System (BMS)
4. Bug Solution Pool (BSP)
5. Bug Reports (BR)

Each sub module has its unique functionalities and different access points for users.

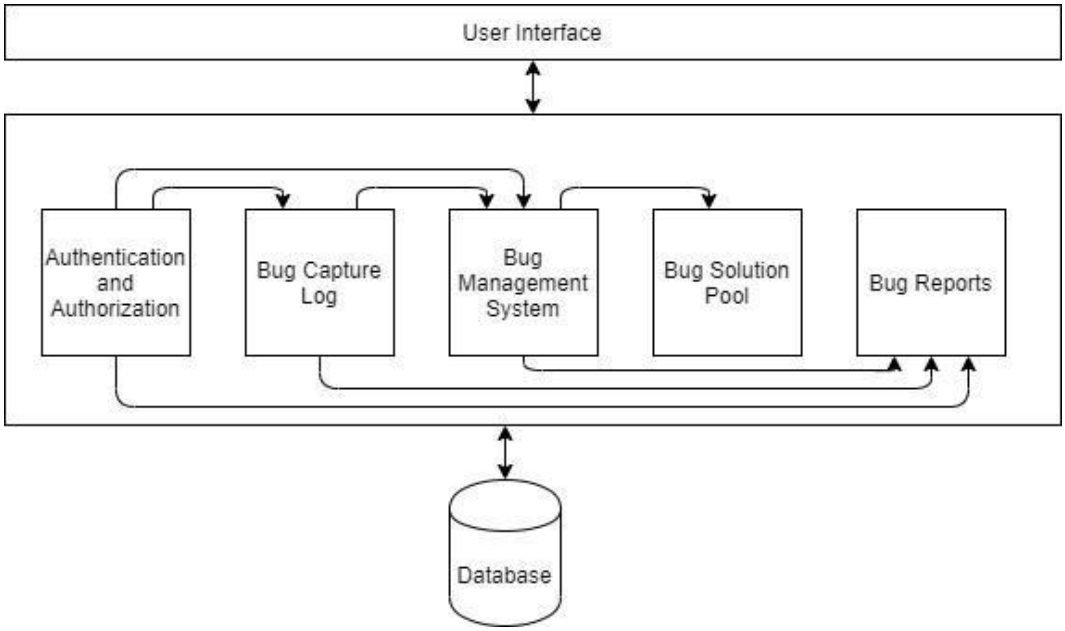


Figure 5. 1 - Architecture Diagram

GitHub Link For the Original Architecture Diagram

5.2.1 Authentication and Authorization

The Authentication and Authorization sub module can be specified as the initiation module of the system. Strix Bug Management System follows Role Based Access Control (RBAC) method.

The following functionalities of the system belong to this sub module.

- Users can log in to the system
- Credential validation and verification
- Users can make a password reset request
- User creation functionality for admins
- User role creation and assignment functionality for admins
- Permission creation and assignment functionality for admins
- Project creation and assignment functionality for admins.
- Users, Projects search option for admins

5.2.2 Bug Capture Log

BCL is the interface where external users (customers) can engage with the system. Customers who log into the system through the Authentication and Authorization sub module will be directed to the BCL module.

All the functions that are available in this sub module will be only allowed for the "customer" user role.

- Customers can view the project(s) where he/she is a part of
- Customers can select a project and view previously filed issues and any added comments.
- Customers can file an issue for the selected project
- Customers can add comments to the issue
- The issue filing process can be listed down as follows,
    - Customer selects "Create new Issue" button
    - Issue window requests input of a topic and a description
    - Issue window allows uploading files
    - Issue window allows screen recordings and screen capturing

5.2.3 Bug Management System

This sub module will be only available for internal users (managers, QAs and developers). Internal users who log into the system through the Authentication and Authorization sub module will be directed to the BMS module.

Issues created by customers in the BCL module will be visible in the BMS issue backlog which is a pool of all the bugs belonging to that particular project.

It should be noted that a sprint will consist of a Kanban board with four lanes. The four lanes will be,

- Open
- In progress
- Review
- Completed

Following sub functions will be available in BMS for the mentioned user roles.

- Manager
    - View issue backlog
    - Sort and filter issues in the backlog
    - Create sprints and customize its due date
    - Add issues to the sprint
    - View sprint backlog
    - Pin sprints to the sidebar
    - Approve or reject issues that developers have marked for the Bug Solution Pool
    - End sprint
    - Search option for the issue and sprint backlogs

- QA
    - View issue backlog
    - Change tags of the bug if necessary
    - View sprint backlog
    - Add comments
    - Review issues which are in the “Review” lane of the kanban board
    - Move issues from “Review” lane to “Done” lane or vice versa
    - Move issues from “Review” lane to “In progress” lane
- Developer
    - View Issue backlog
    - Add comments to issues
    - View Sprint backlog
    - Attach files to the issue
    - User can add time spent on a bug (time tracking)
    - Move issues from “Open” lane to “In progress” lane or vice versa
    - Move issues from “In progress” lane to “Review” lane or vice versa
    - Marking issues for Bug Solution Pool

5.2.4 Bug Solution Pool

Bug Solution Pool is a platform where resolved bugs are referenced as a knowledge resource. Managers, QAs, developers are allowed to use this sub module. Following functionalities are provided for the users.

- View issues and previously added comments
- Add comments
- Full-text search option

5.2.5 Bug Reports

Bug Reports sub module exists to provide graphical representation and an overall summary of the issues, users, sprints, and other details. The following reports are supported in this sub module.

- Project / Developer timesheet
- Developer x Bug development
- Developer performance
- Developer x Average effort
- Project x Bug Development
- Sprint summary
- Monthly bug summary
- Month x Total bugs
- Month x Bug development
- Full bug summary

5.3 Design

Figure 5.3.1 - Use Case Diagram

Figure 5.3.2 - Activity Diagram for Login and Password Reset

Figure 5.3.3 - Activity Diagram for Authentication and Authorization

Figure 5.3.4 - Activity Diagram for Bug Capture Log

Figure 5.3.5 - Activity Diagram for Bug Management System

Figure 5.3.6 - Activity Diagram for Bug Reporting

Figure 5.3.7 - Activity Diagram for Bug Solution Pool

Figure 5.3.8 - Activity Diagram for Bug Solution Pool Approvals

Figure 5.3.9 - Sequence Diagram for Login and Password Reset

Figure 5.3.10 - Sequence Diagram for Authentication and Authorization

Figure 5.3.11 - Sequence Diagram for Bug Capture Log

Figure 5.3.12 - Sequence Diagram for Bug Management System (Manager)

Figure 5.3.13 - Sequence Diagram for Bug Management System (QA)

Figure 5.3.14 - Sequence Diagram for Bug Management System (Developer)

Figure 5.3.15 - Sequence Diagram for Bug Solution Pool Approvals

Figure 5.3.16 - Sequence Diagram for Bug Solution Pool (QA, Developer)

Figure 5.3.17 - Sequence Diagram for Bug Solution Pool (Manager)

Figure 5.3.18 - Sequence Diagram for Bug Reporting

Figure 5.3.19 - Component Diagram

Figure 5.3.20 - Class Diagram

Figure 5.3.21 - ER Diagram

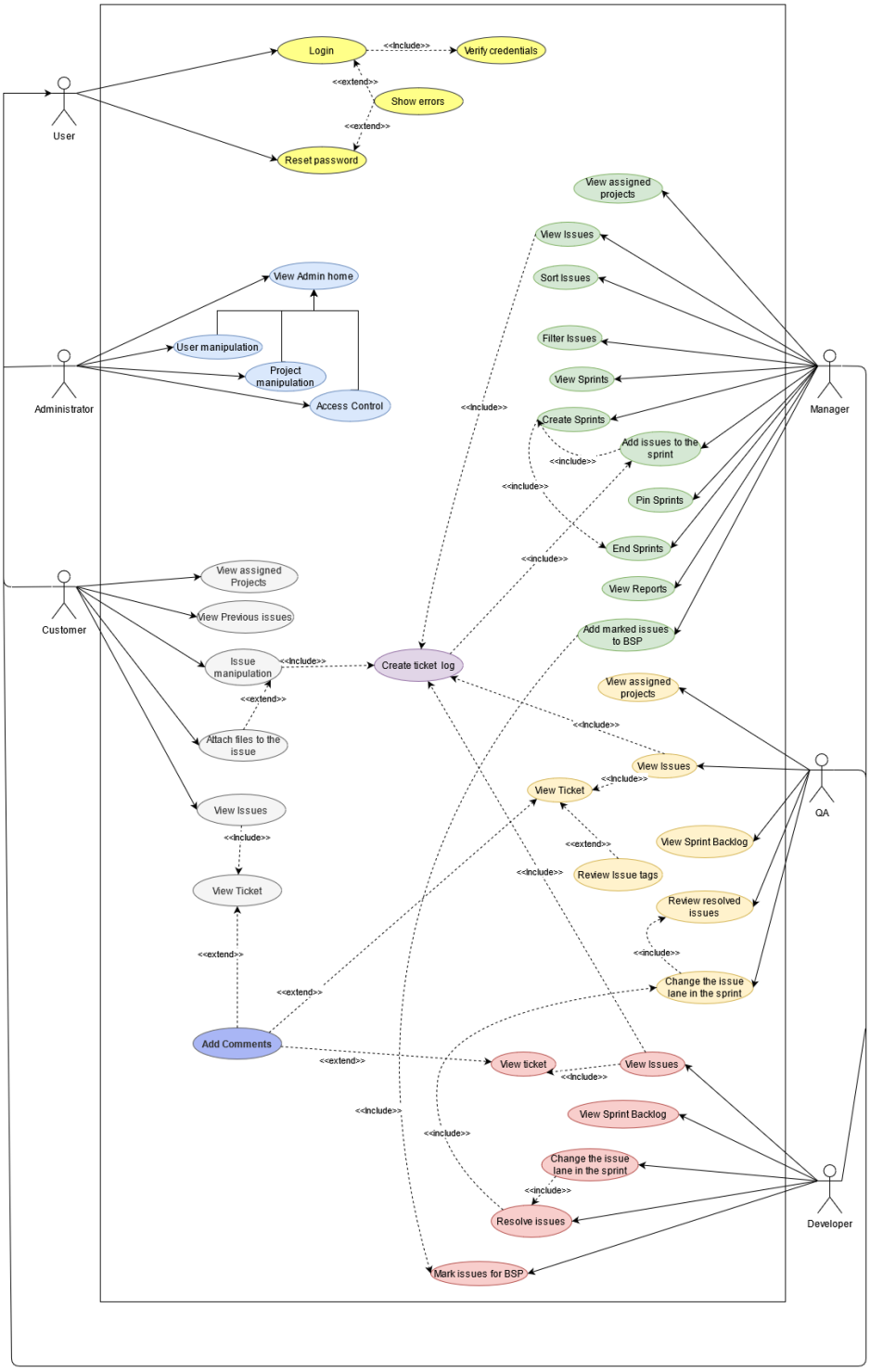


Figure 5.3.1 – Use case Diagram

GitHub Link For the Original Use Case Diagram

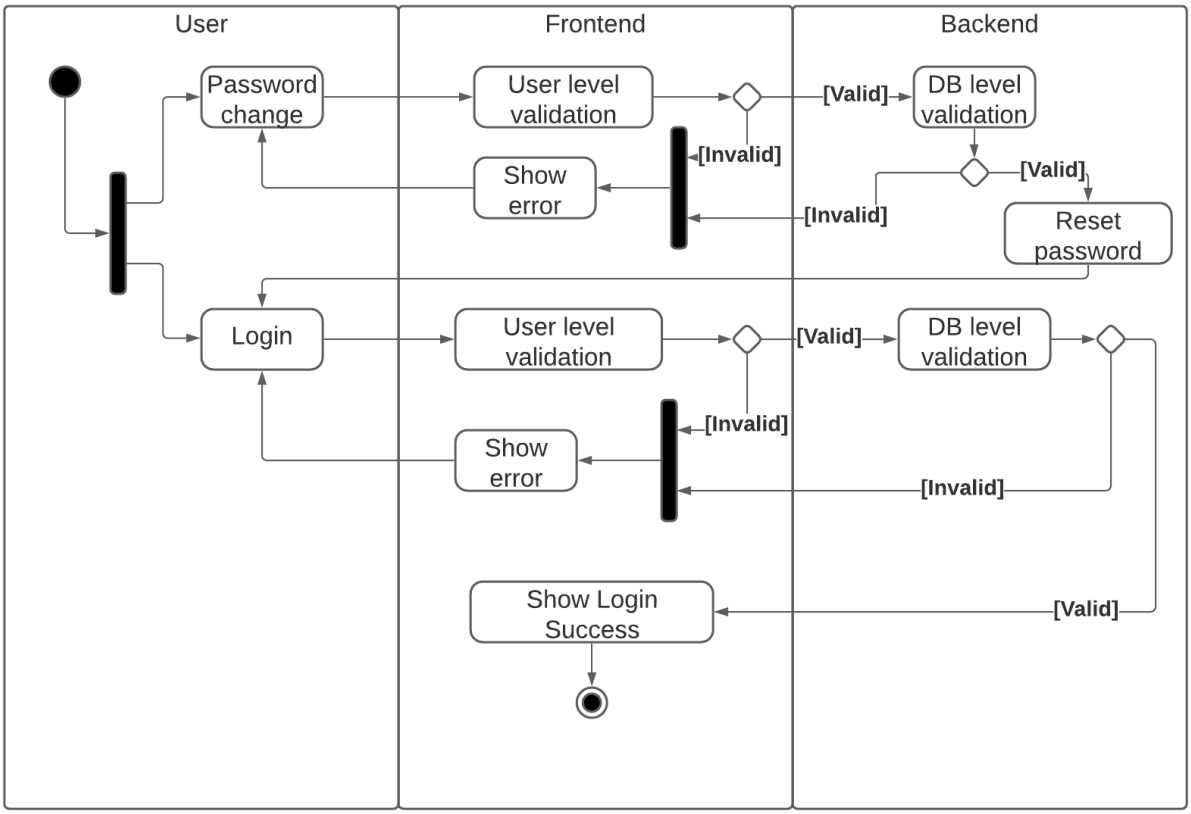


Figure 5.3.2 – Activity Diagram for Login & Password Reset

GitHub Link For the Original Activity Diagram for Login & Password Reset

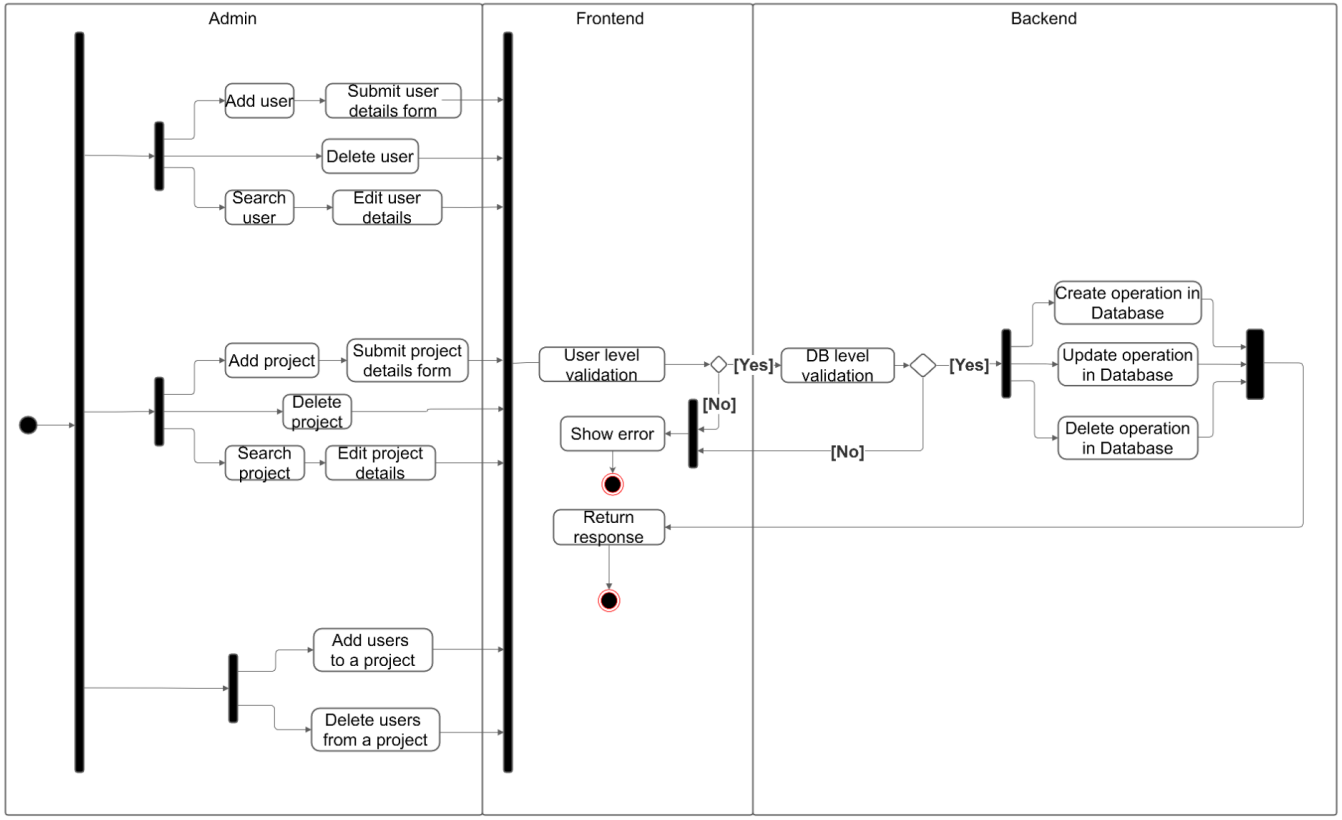


Figure 5.3.3 – Activity Diagram for Authentication & Authorization

GitHub Link For the Original Activity Diagram for Authentication and Authorization

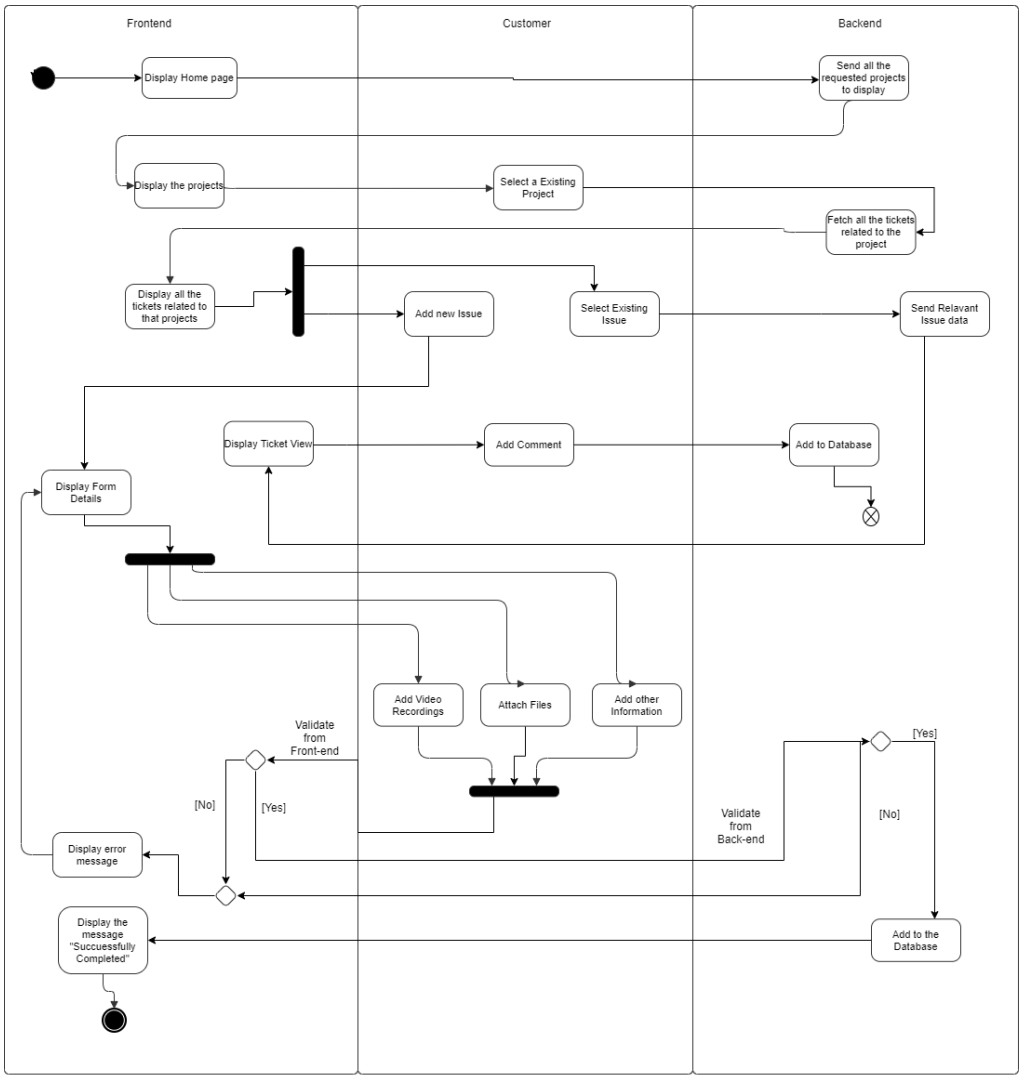


Figure 5.3.4 - Activity Diagram for Bug Capture Log

GitHub Link For the Original Activity Diagram for Bug Capture Log

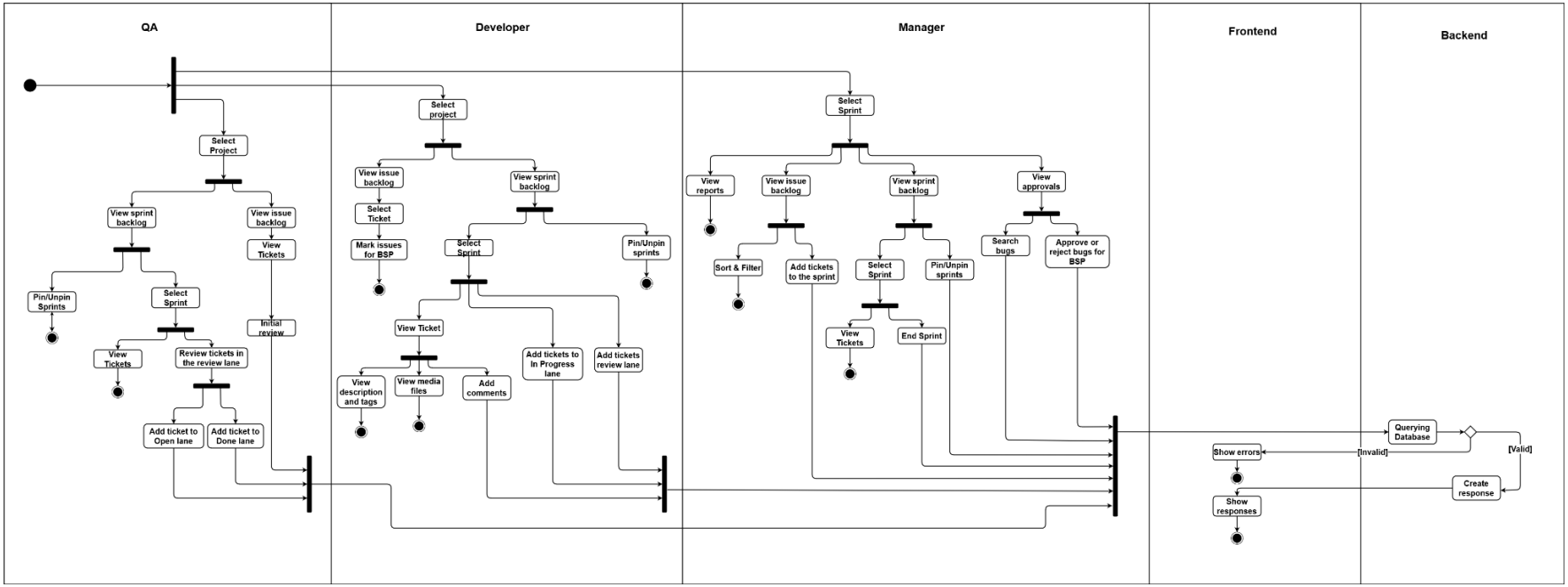


Figure 5.3.5 – Activity Diagram for Bug Management System

GitHub Link For the Original Activity Diagram for Bug Management System

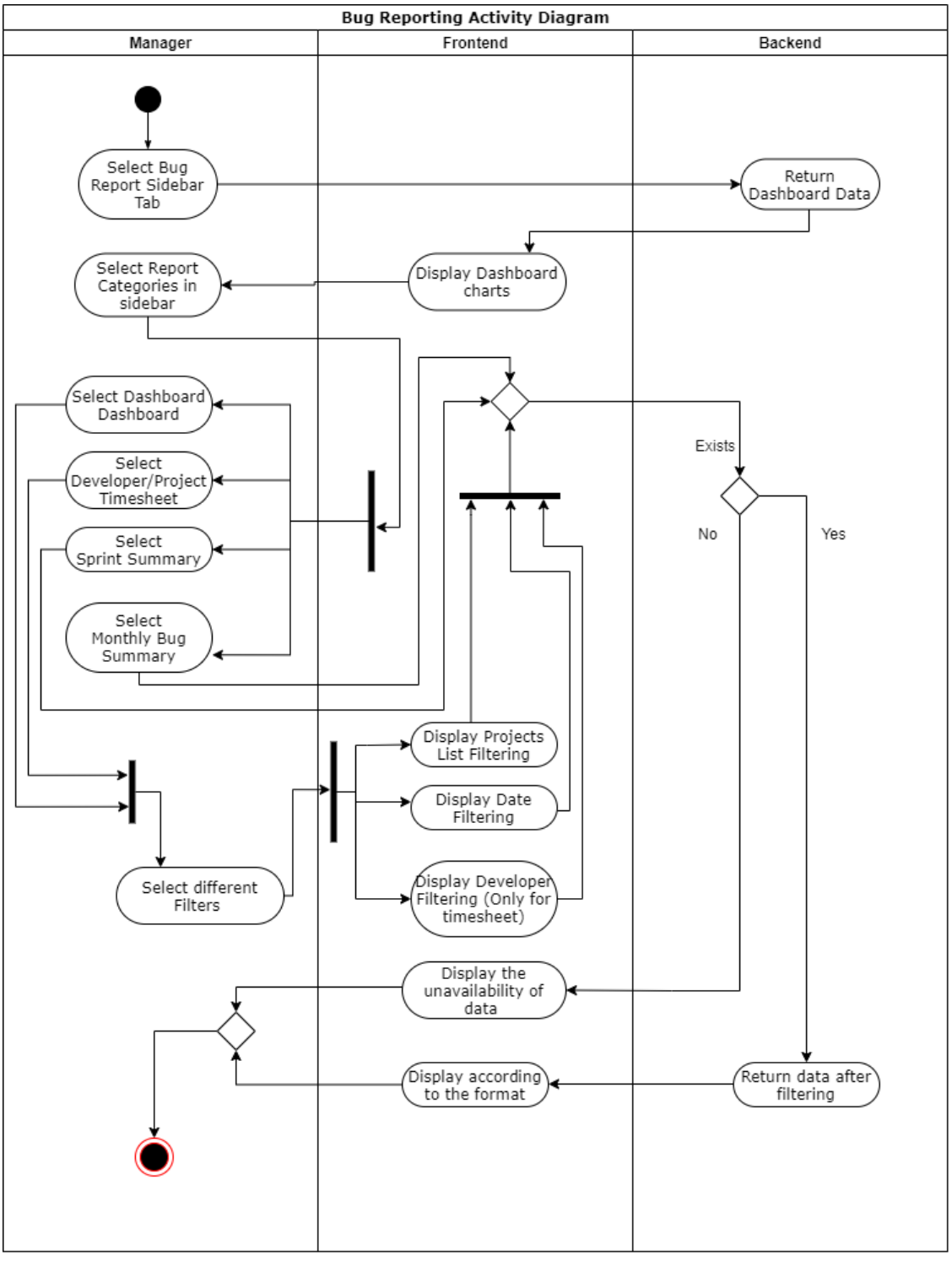


Figure 5.3.6 – Activity Diagram for Bug Reporting

GitHub Link For the Original Activity Diagram for Bug Reporting

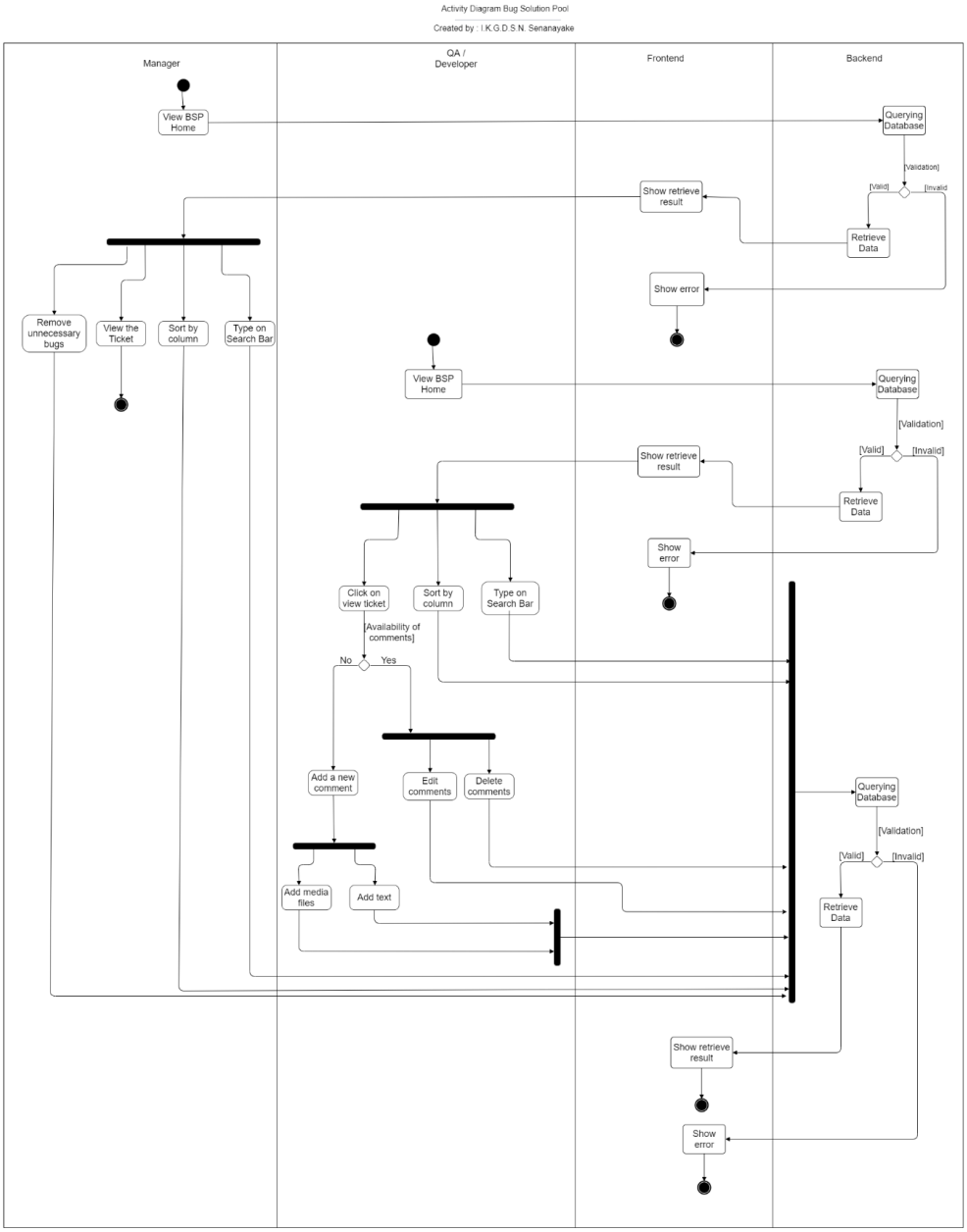


Figure 5.3.7 – Activity Diagram for Bug Solution Pool

GitHub Link For the Original Activity Diagram for Bug Solution Pool

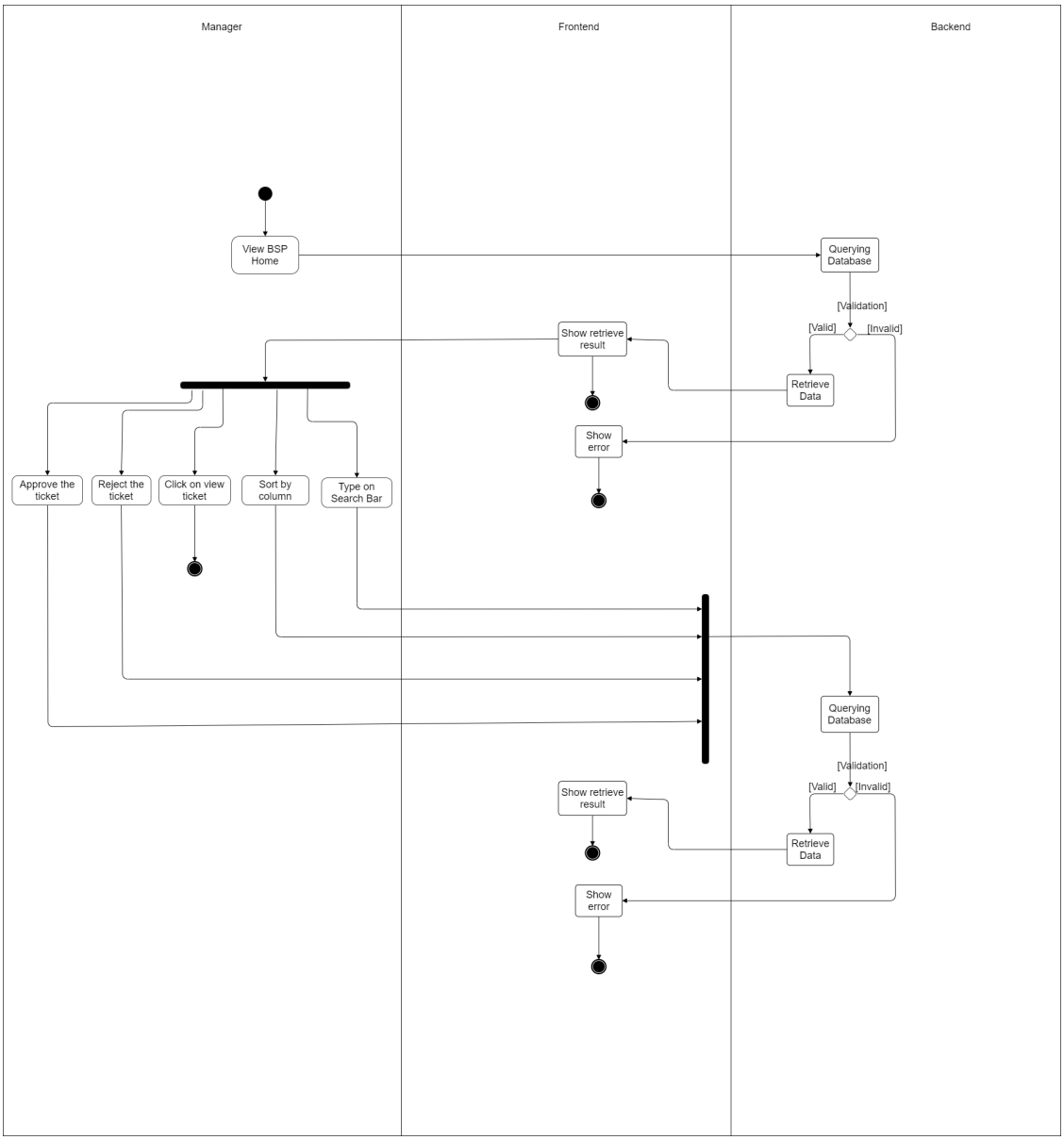


Figure 5.3.8 – Activity Diagram for Bug Solution Pool Approvals

GitHub Link For the Original Activity Diagram for Bug Solution Pool Approvals

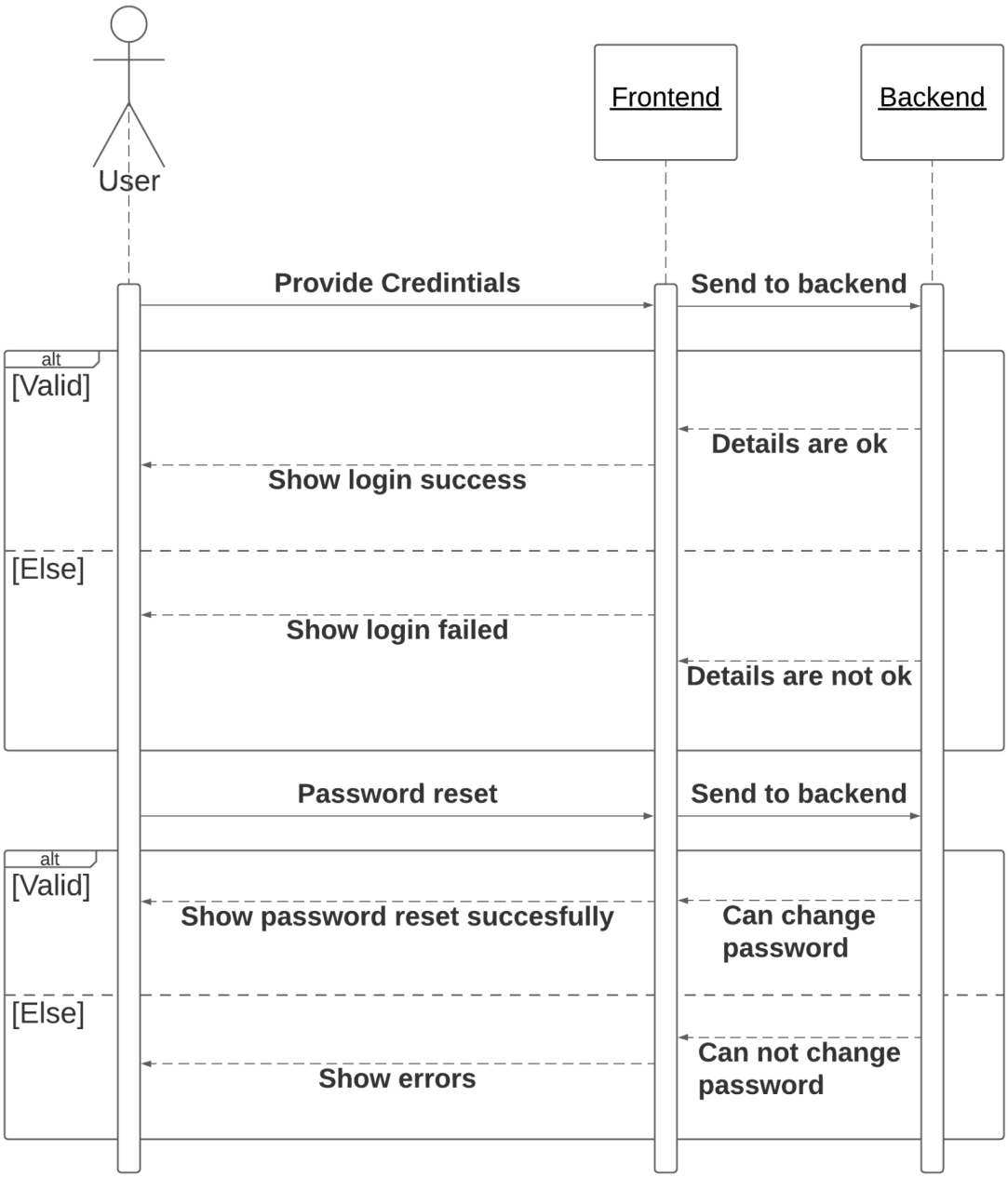


Figure 5.3.9 - Sequence Diagram for Login and Password reset

GitHub Link For the Original Sequence Diagram for Login and Password reset

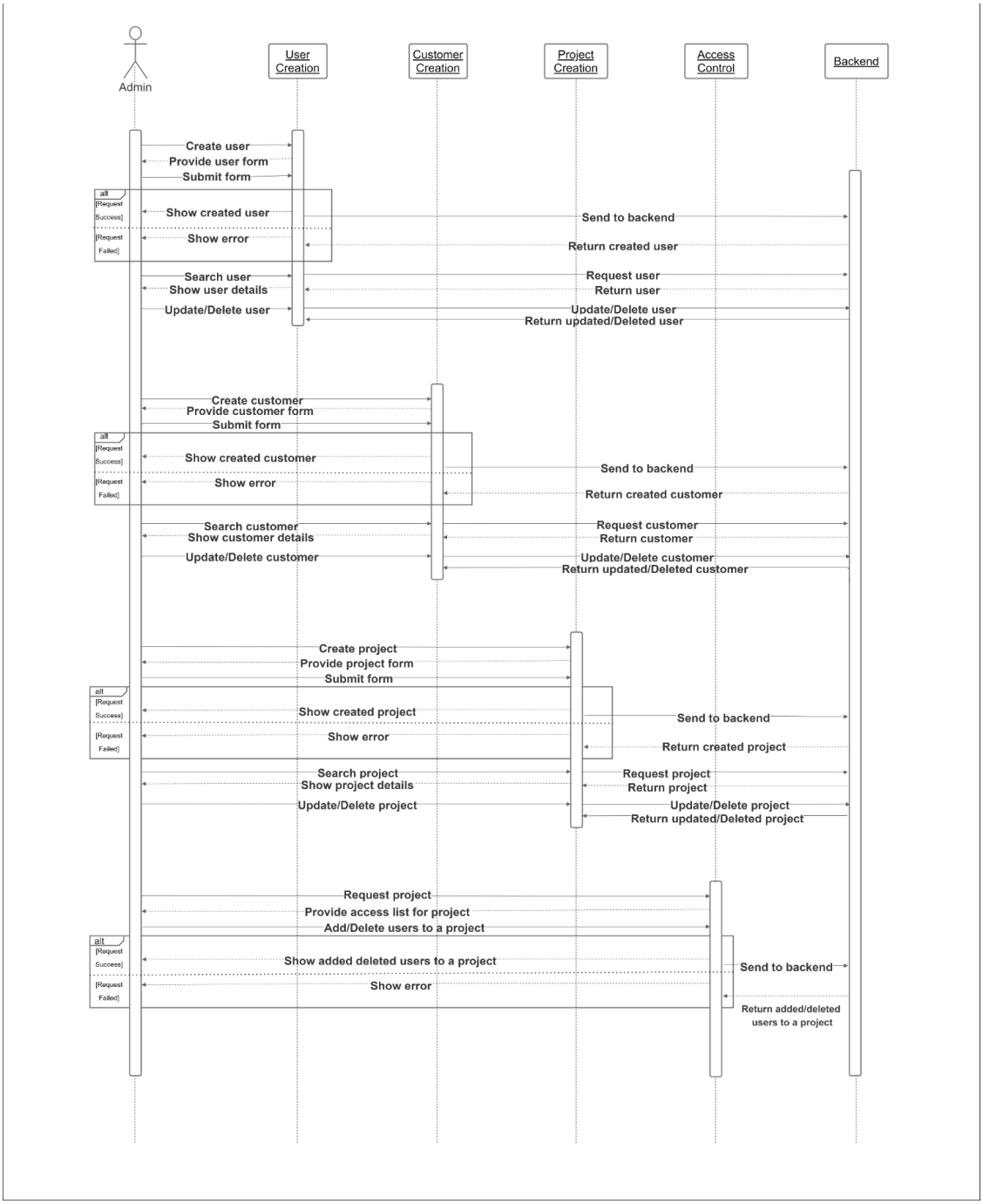


Figure 5.3.10 - Sequence Diagram for Authentication and authorization

GitHub Link For the Original Sequence Diagram for Authentication and Authorization

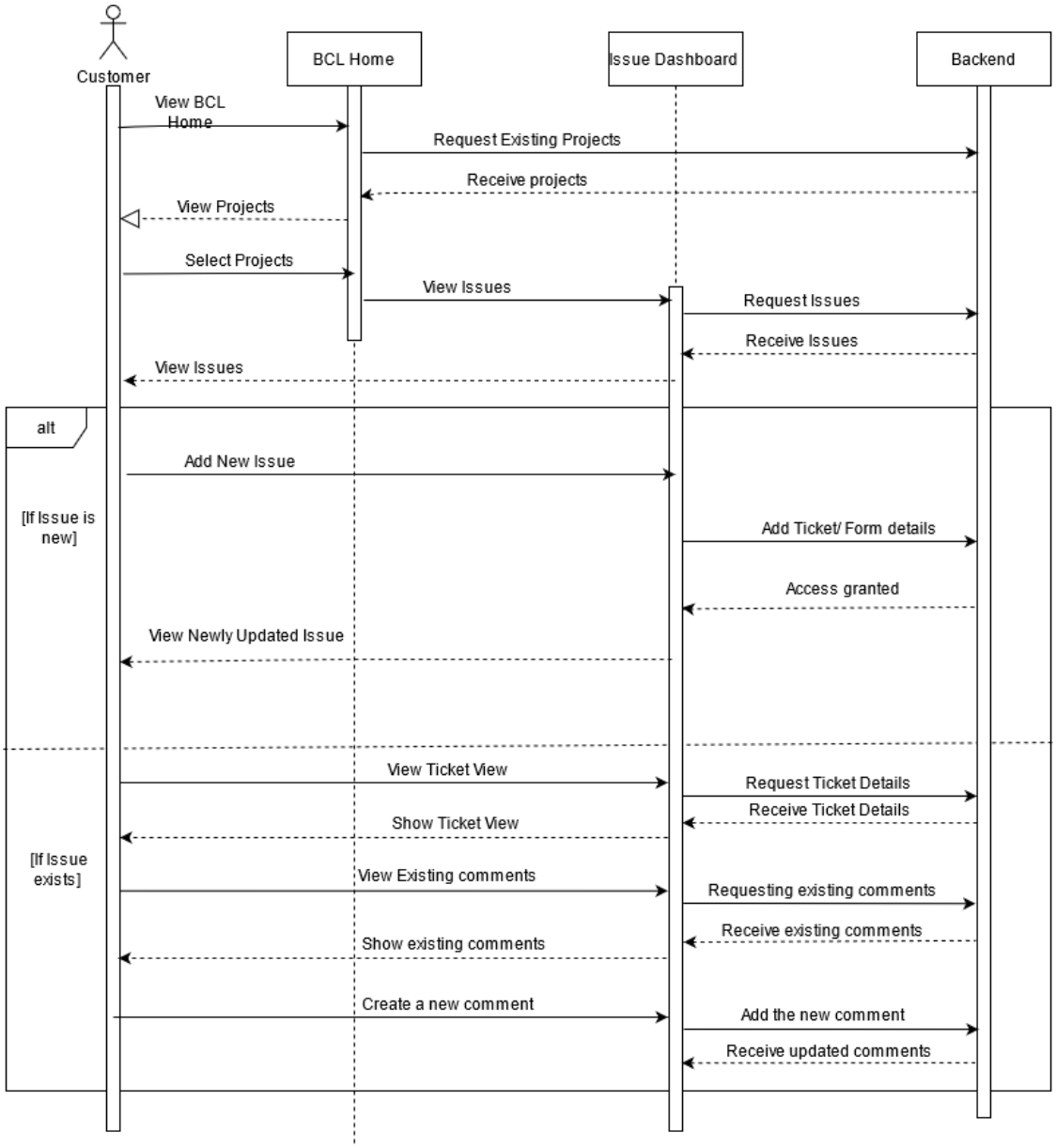


Figure 5.3.11 - Sequence Diagram for Bug Capture Log

GitHub Link For the Original Sequence Diagram for Bug Capture Log

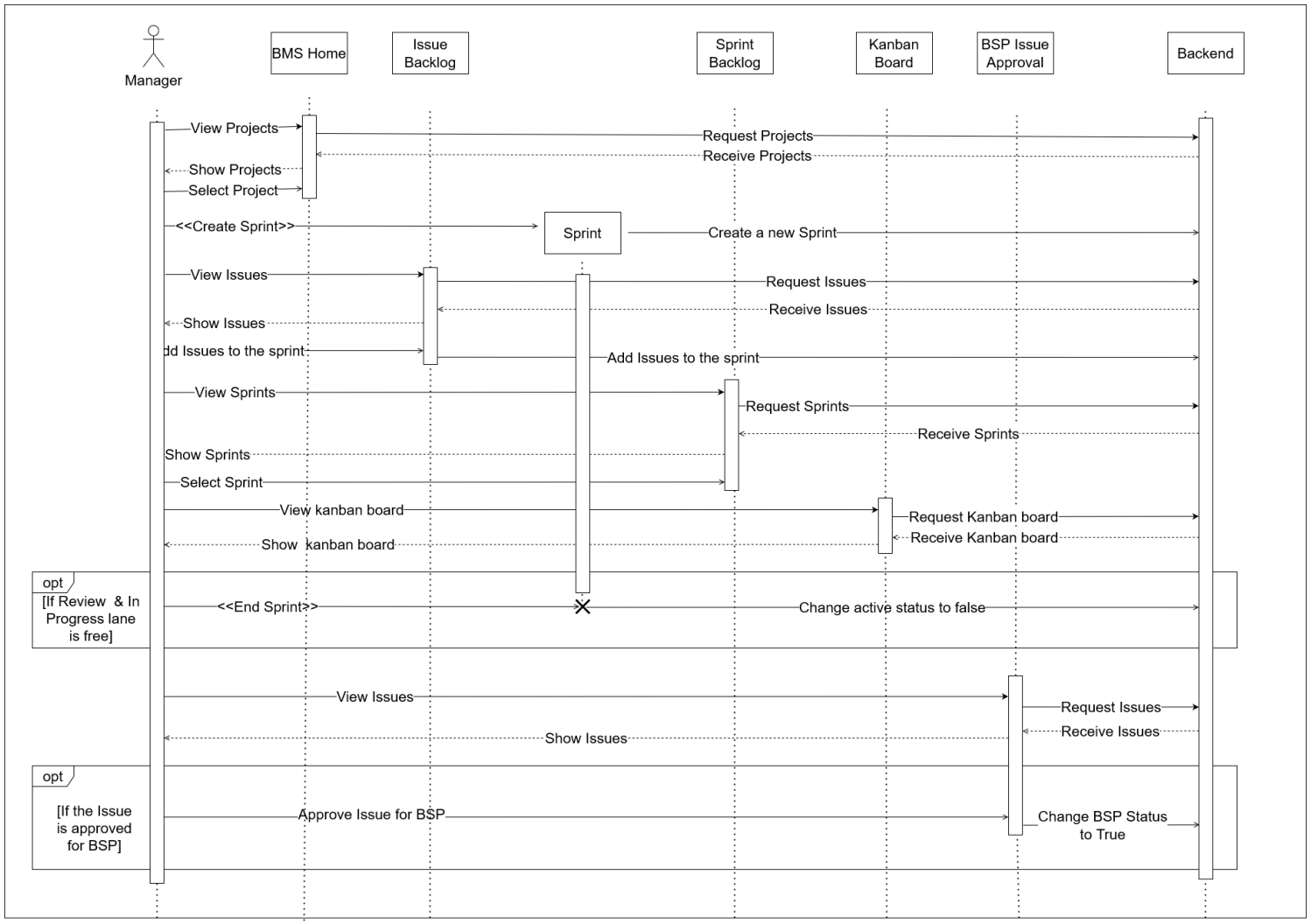


Figure 5.3.12 - Sequence Diagram for Bug Management System (Manager)

GitHub Link For the Original Sequence Diagram for Bug Management System (Manager)

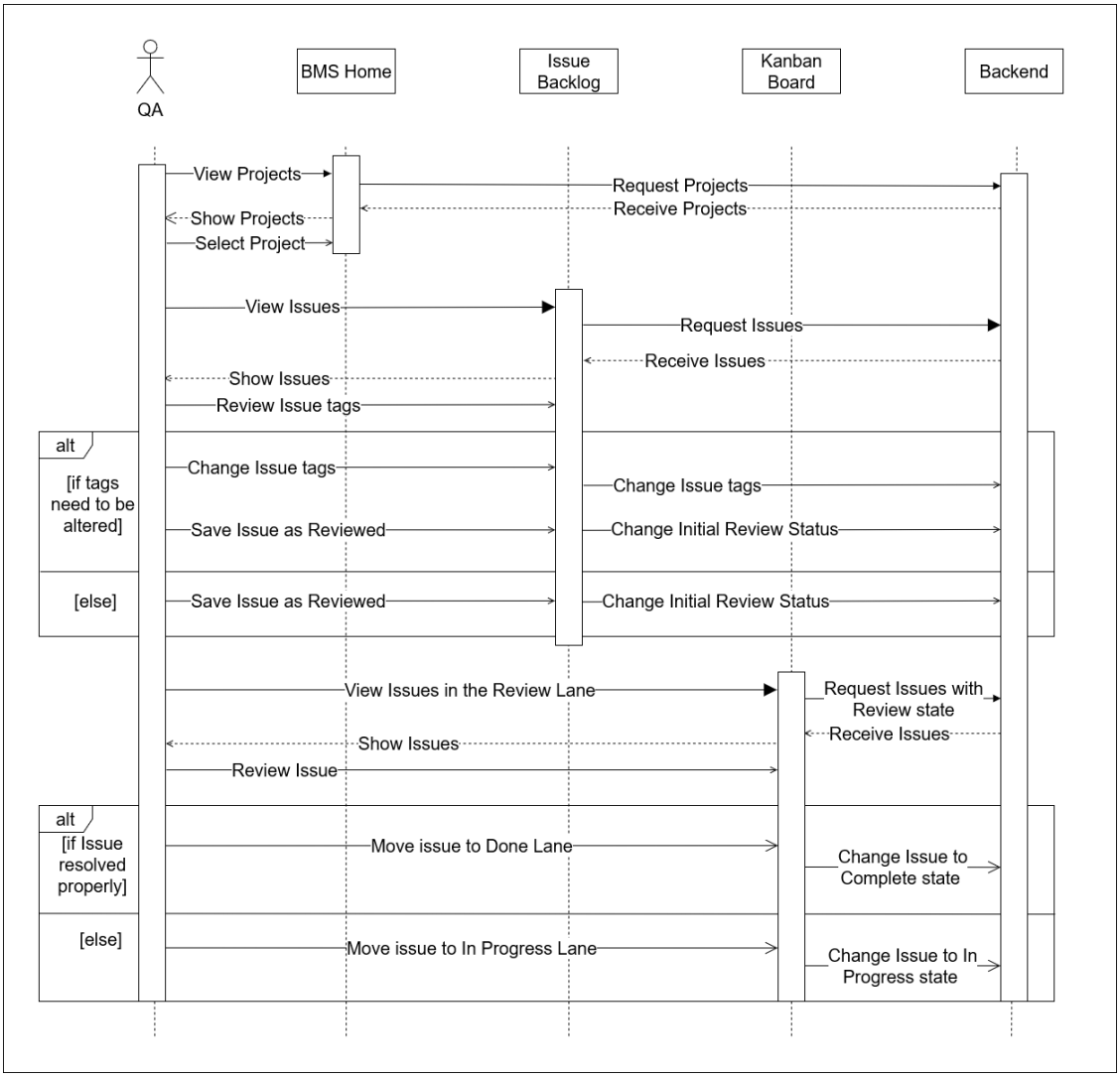


Figure 5.3.13 - Sequence Diagram for Bug Management System (QA)

GitHub Link For the Original Sequence Diagram for Bug Management System (QA)

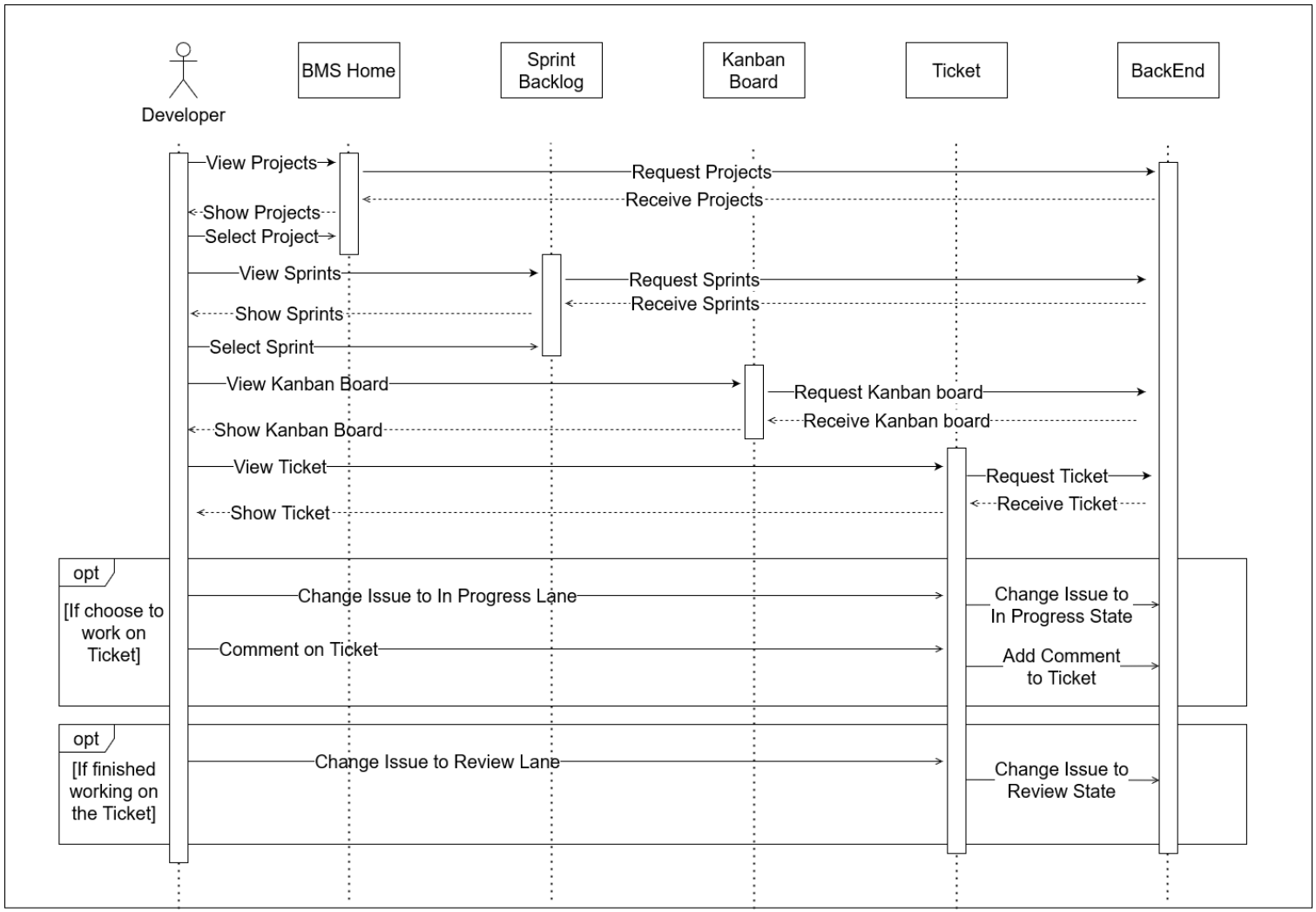


Figure 5.3.14 - Sequence Diagram for Bug Management System (Developer)

GitHub Link For the Original Sequence Diagram for Bug Management System (Developer)

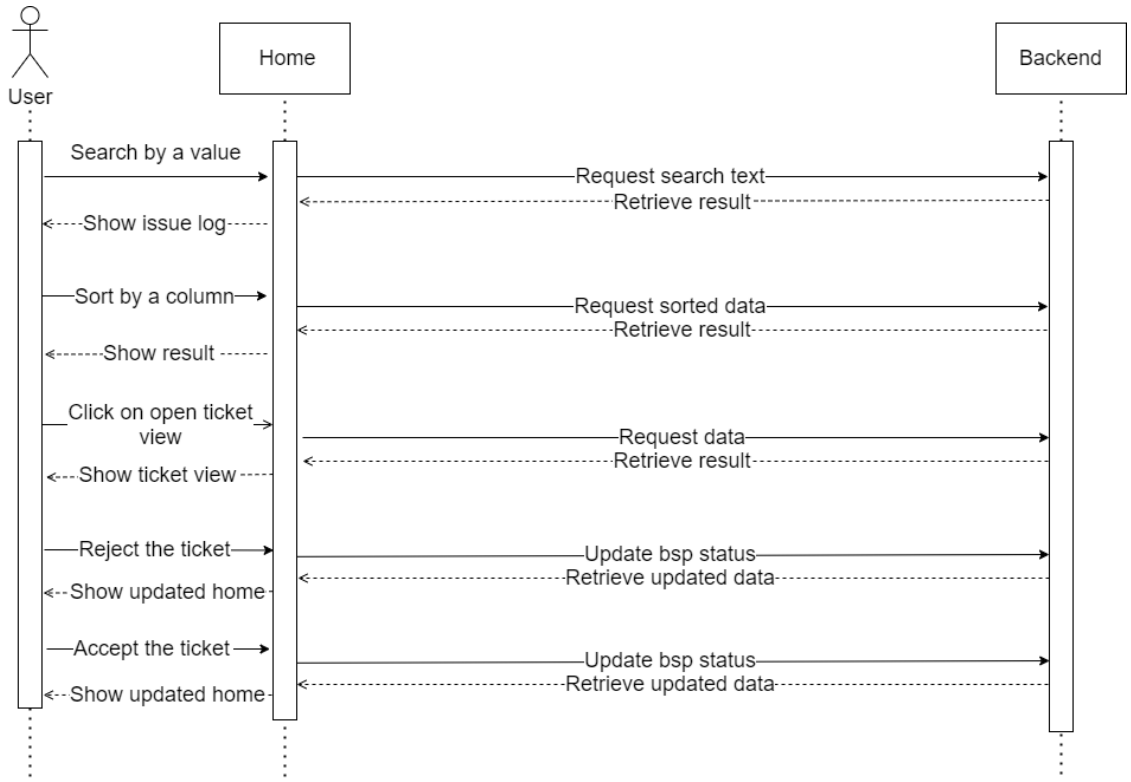


Figure 5.3.15 - Sequence Diagram for Bug Solution Pool Approvals

GitHub Link For the Original Sequence Diagram for Bug Solution Pool

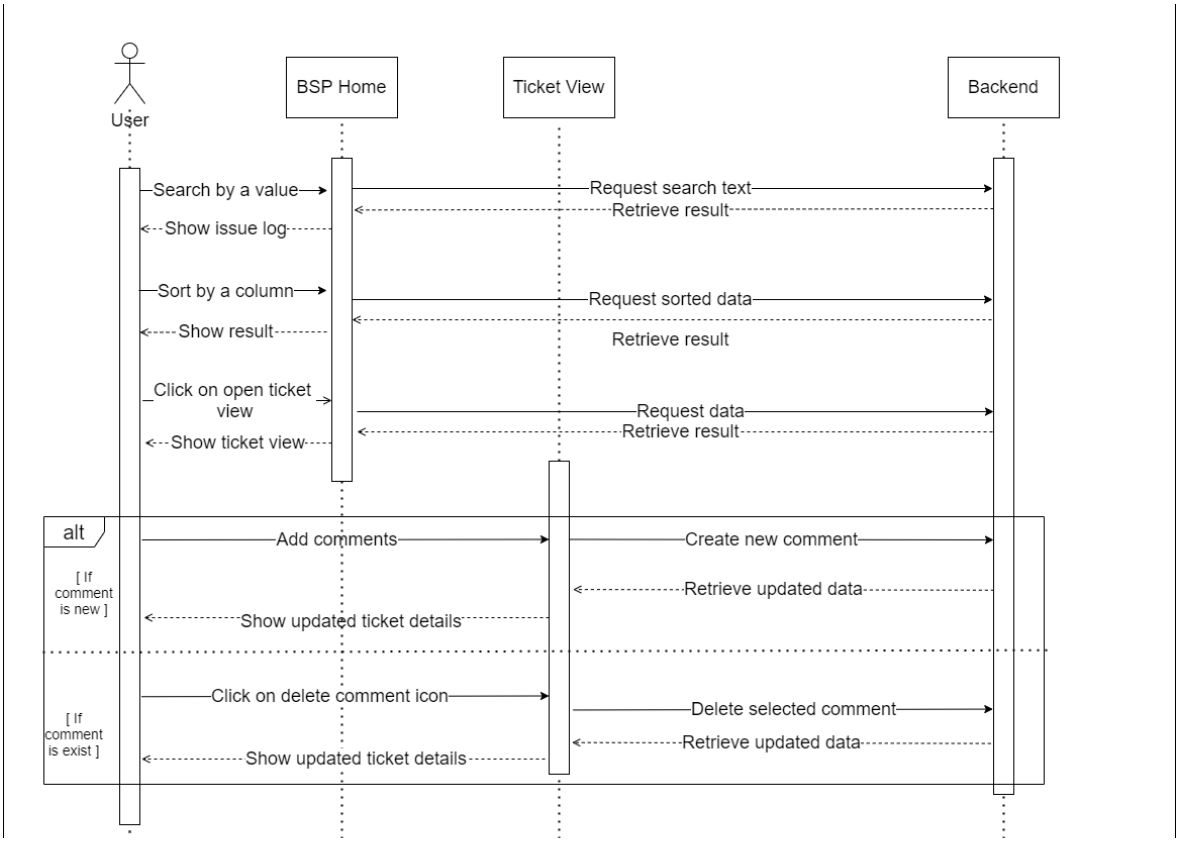


Figure 5.3.16 - Sequence Diagram for Bug Solution Pool (QA, Developer)

GitHub Link For the Original Sequence Diagram for Bug Solution Pool

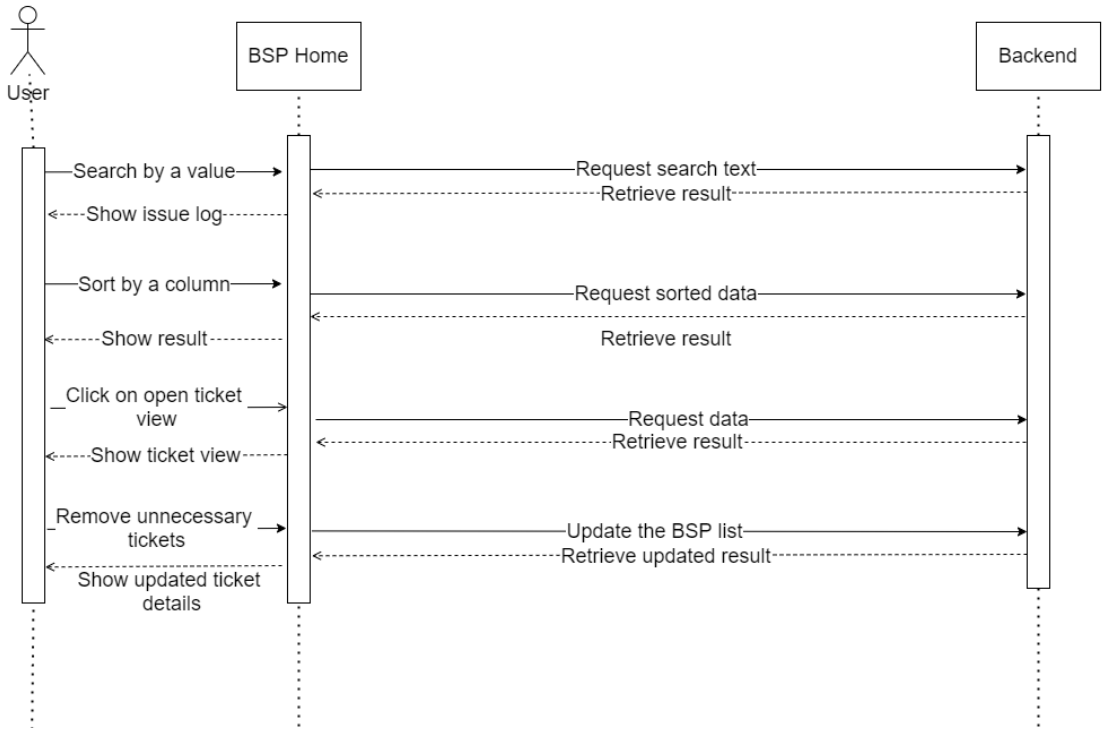


Figure 5.3.17 - Sequence Diagram for Bug Solution Pool (Manager)

GitHub Link For the Original Sequence Diagram for Bug Solution Pool

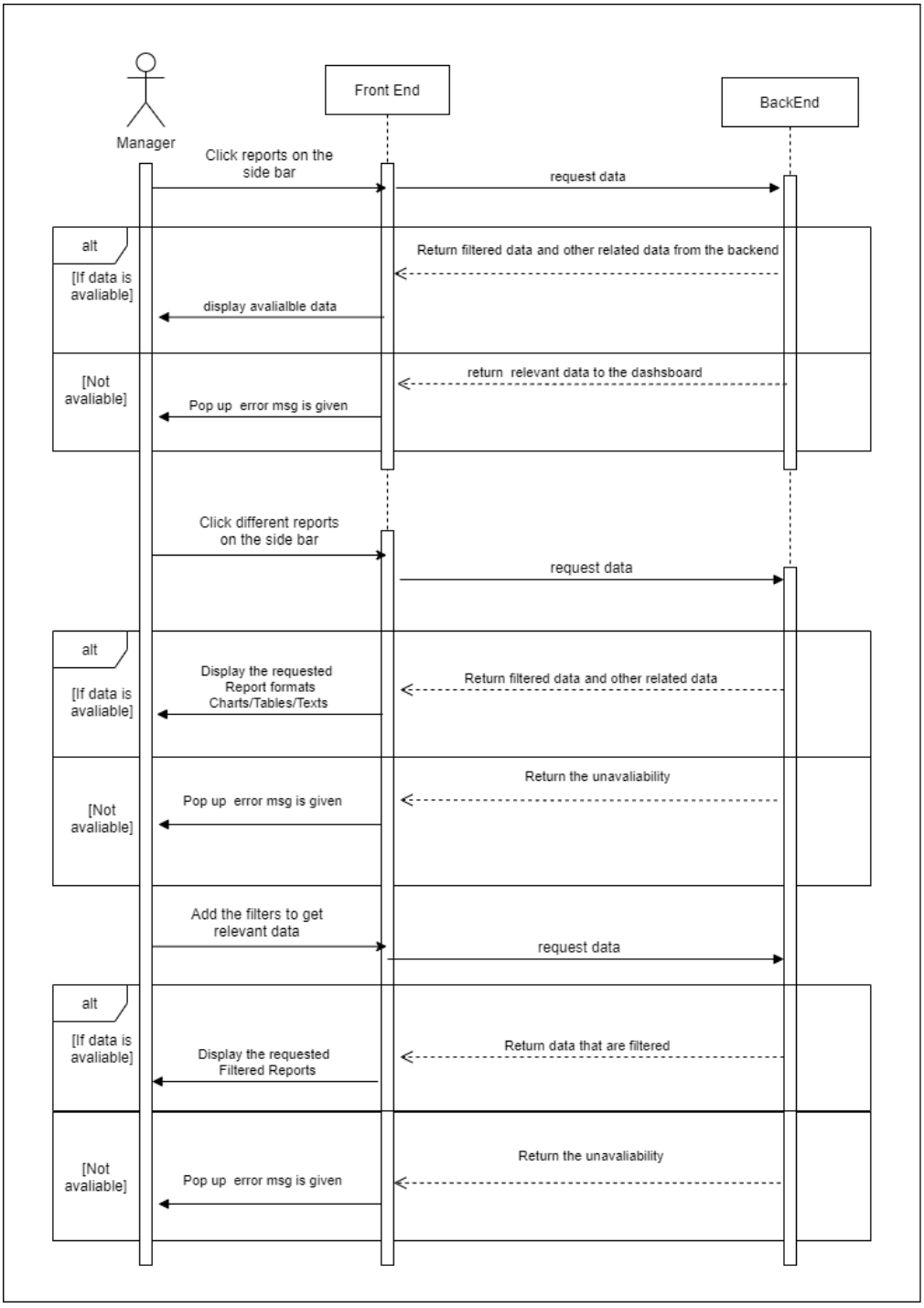


Figure 5.3.18 - Sequence Diagram for Bug Reporting

GitHub Link For the Original Sequence Diagram for Bug Reporting

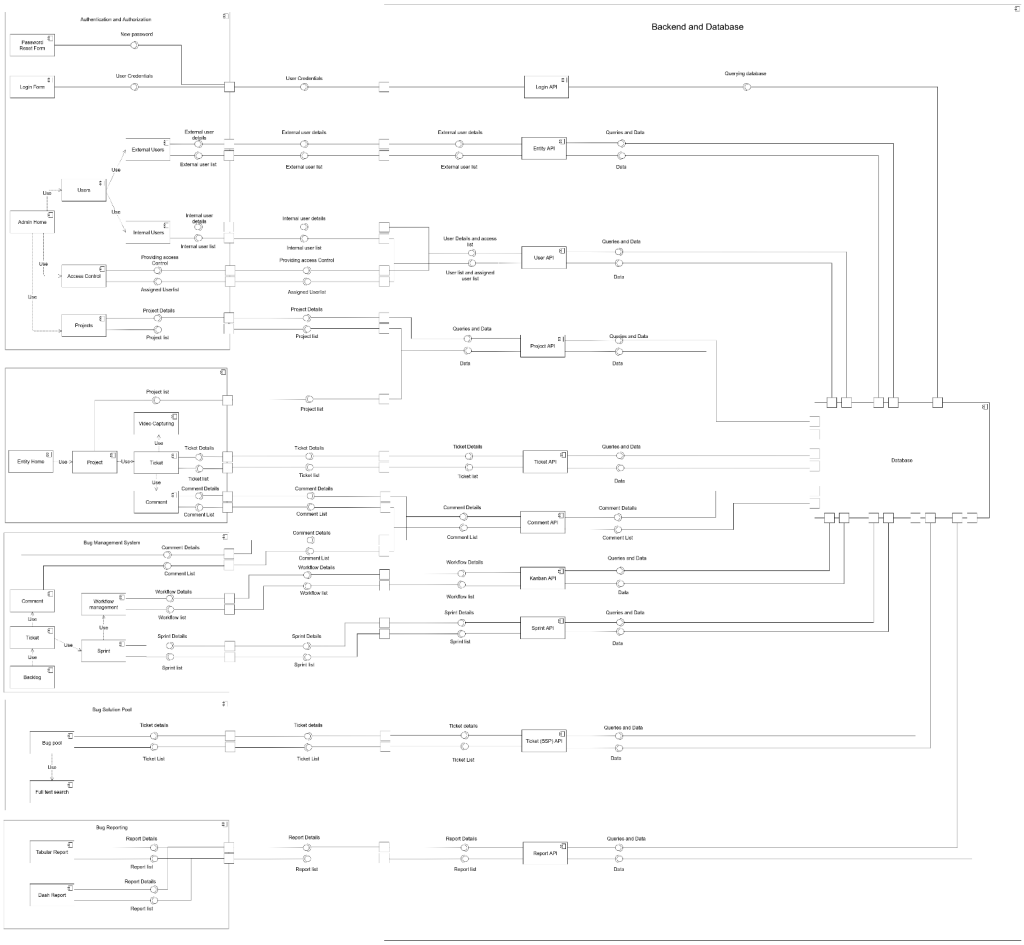


Figure 5.3.19 - Component Diagram

GitHub Link For the Original Component Diagram

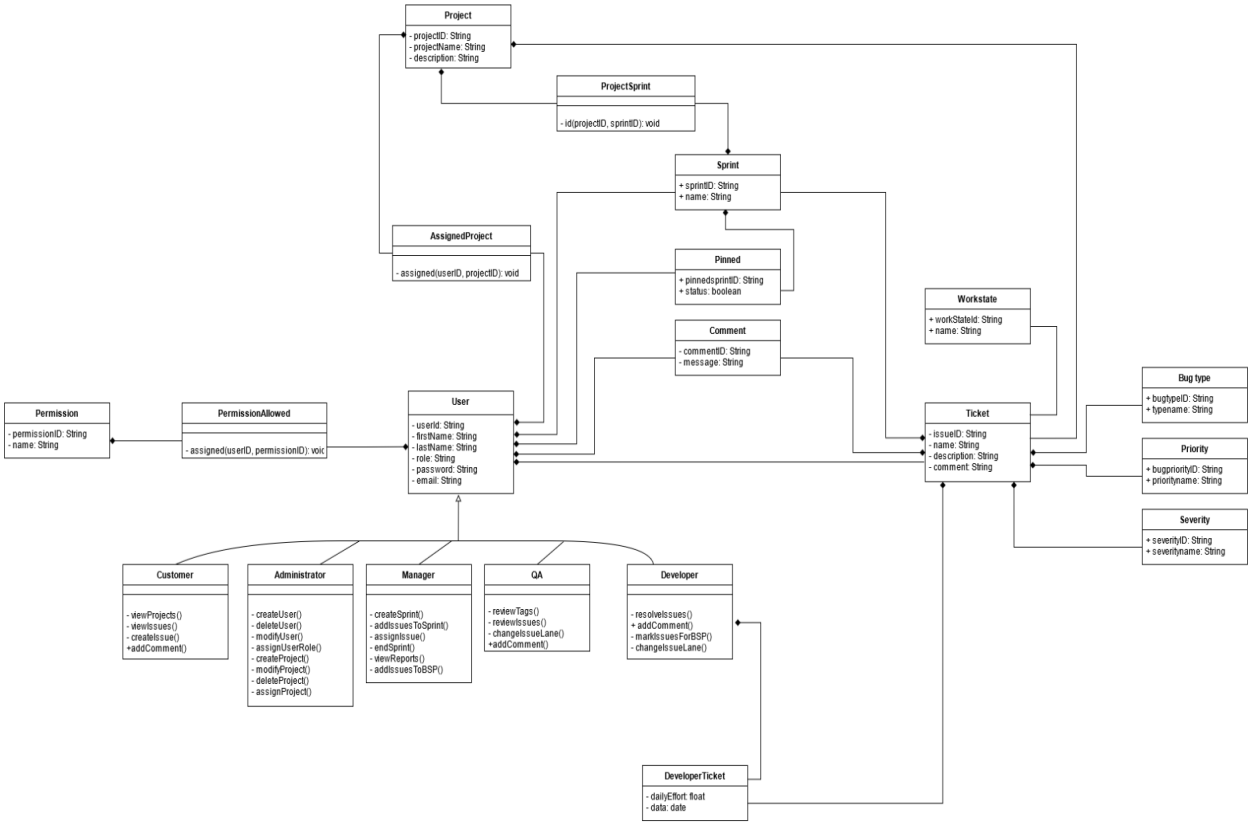


Figure 5.3.20 - Class Diagram

GitHub Link For the Original Class Diagram

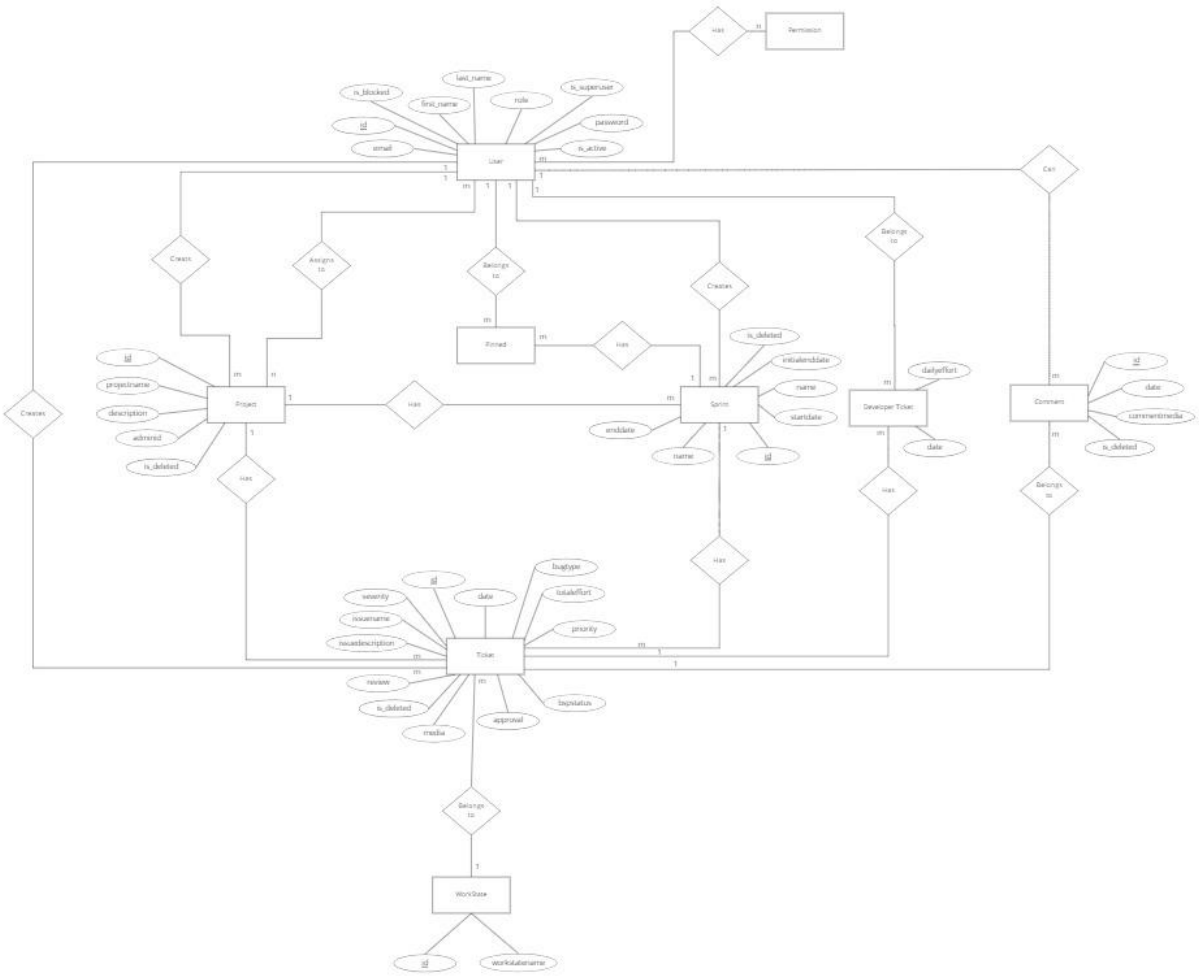


Figure 5.3.21 - ER Diagram

GitHub Link For the Original ER Diagram

5.4 Summary

In this Chapter we described the modules of our system. Furthermore, we provided detailed UML diagrams and the ER diagram to help readers understand the process of each of the modules.

Chapter 6

Implementation

6.1 Introduction

This chapter describes the process of implementing the software solution from identifying the system requirements followed by further analysis and the design process.

The rest of the chapter is organized as follows. Section 6.2 provides details about the implementation process. Section 6.3 provides a diagram explaining the layered architecture of the system.

6.2 Implementation Process

As the first step of the process, we identified the requirements of the project by going through project documentation provided by the client organization. However, to understand the problem and requirements properly, we held virtual and physical meetings with the client organization.

After considering the difficulty of each function that is needed, the system was then divided into modules as follows,

1. Authentication and Authorization
2. Bug Capture Log
3. Bug Management System
4. Bug Solution Pool
5. Bug Reports

Then each of these modules was divided into sub modules for ease of implementation. Those sub modules were assigned to each team member fairly. During the time period of project approval and interim evaluation we managed to complete a part of each module by ourselves. ER(Entity Relationship Diagram), relational mapping and actual database was designed and implemented before the

interim evaluation. After the interim evaluation, the complete project was developed and finally unit testing was accomplished.

6.2.1 Frontend Implementation

Initially, the browser renders the root component which is App.js (This is a common naming convention for root component) to the DOM(Document Object Model). The root component can access all the other components which are routed to the container directory.

The container directory contains all the routes which are used to redirect to a particular component. In our project, we used five different containers for each user role. Those individual container routes to different components.

The services directory contains all the service components. Service components connect backend services with frontend (Ex: System needs to authenticate users and the real user details are kept in the database. Hence frontend needs to get user details from backend).

6.2.2 Backend Implementation

The backend also follows a layered architecture (Generally Django follows a layered approach). URLs file contains all the endpoints and their relevant view. The frontend can invoke any URL with a defined endpoint where the URLs file can route that request to a view.

View file contains all the backend logic. Django has a built-in ORM that maps a python class to SQL. Models file contains all the class files which represent the database. In addition to those files, there are

1. middlewares.py files to create middlewares which are very often used (For example user validation, user exist).
2. EmailService.py file to create email sending functions.

6.2.3 Database Implementation

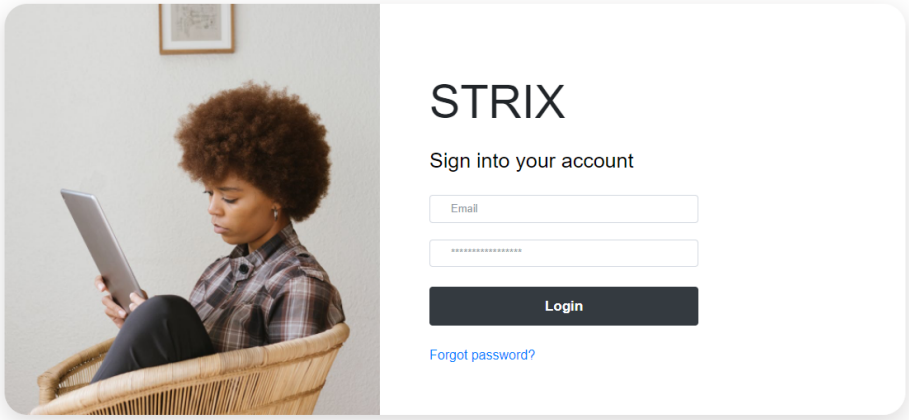
We used Postgresql as our database. Initially, we designed the ER of the Strix bug management system and then we mapped the designed ER into a real database. Since we used Django as our backend we had to follow code based database implementation where we need to implement a relation of our database as a python class. The completed database has fourteen relations and twelve of them are mapped relations and the remaining two were taken from Django default implementation. Since the hosted database was needed we hosted the database in the free tier (Hobby Dev) of Heroku Postgresql. Designed relations of the database are,

1. User table
2. Project table
3. Ticket Table
4. Ticket Media table
5. Workstate Table
6. Bug Type Table
7. Severity Table
8. Priority Table
9. Sprint Table
10. Pinned Table
11. Comment Table
12. DeveloperTicket Table
13. Groups Table
14. AuthToken Table

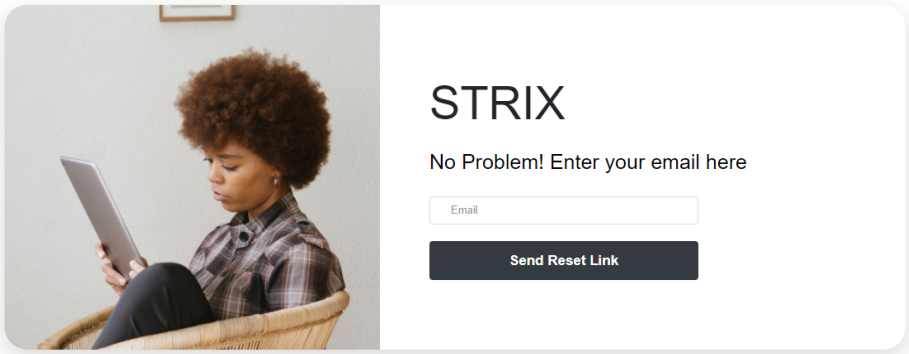
6.2.4 Working design of the system

All Users

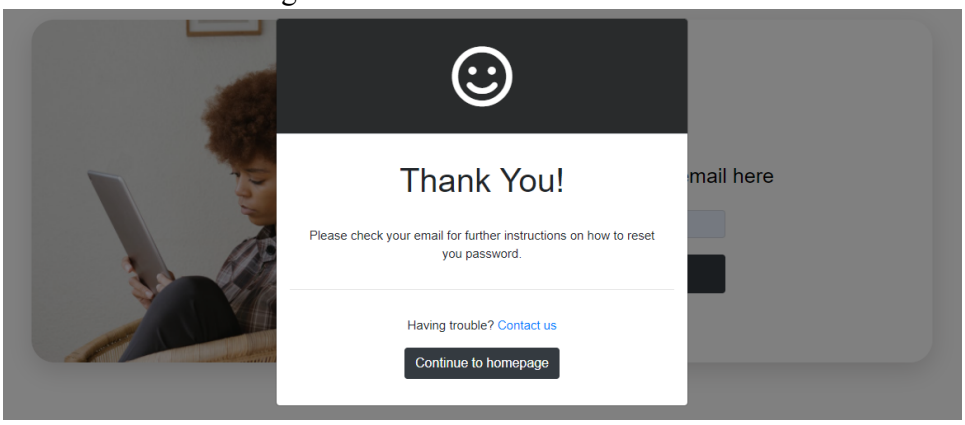
Login Page



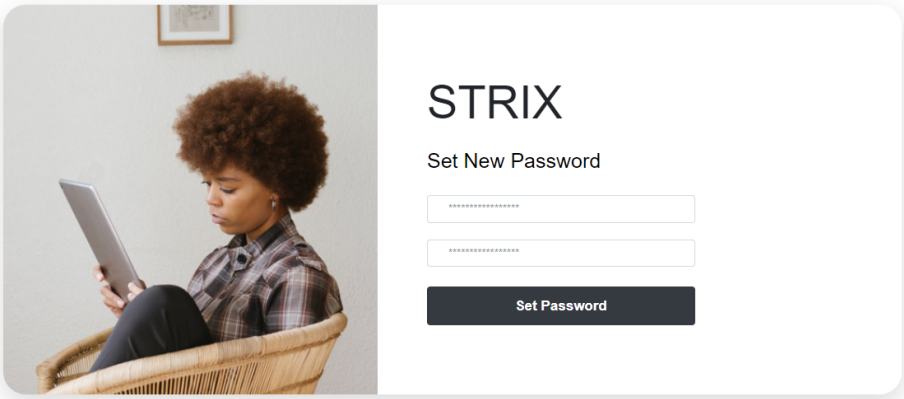
Forgot password Page



Email confirmation Page

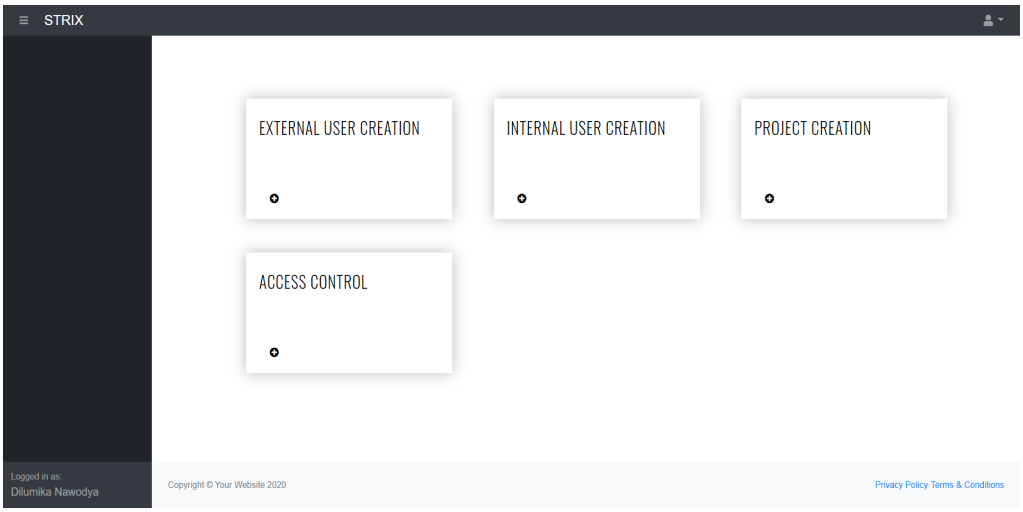


Set new password Page

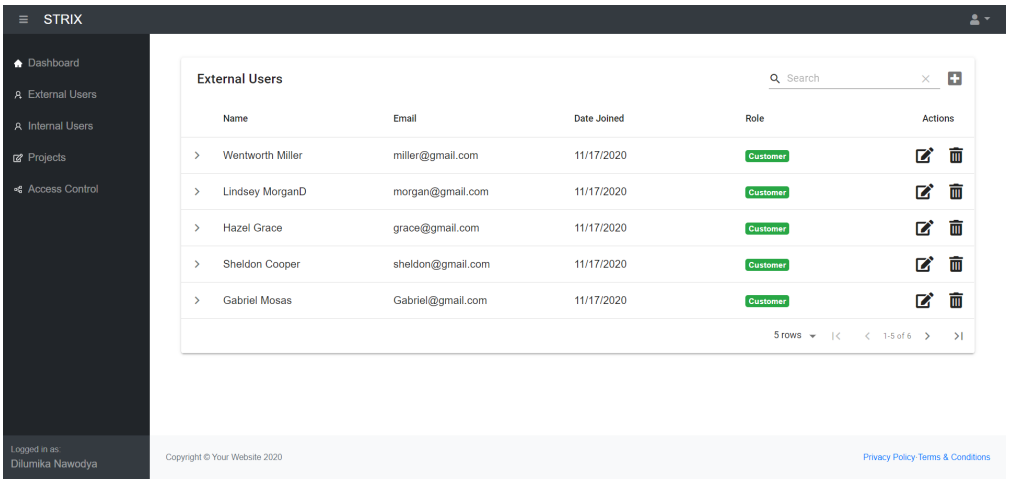


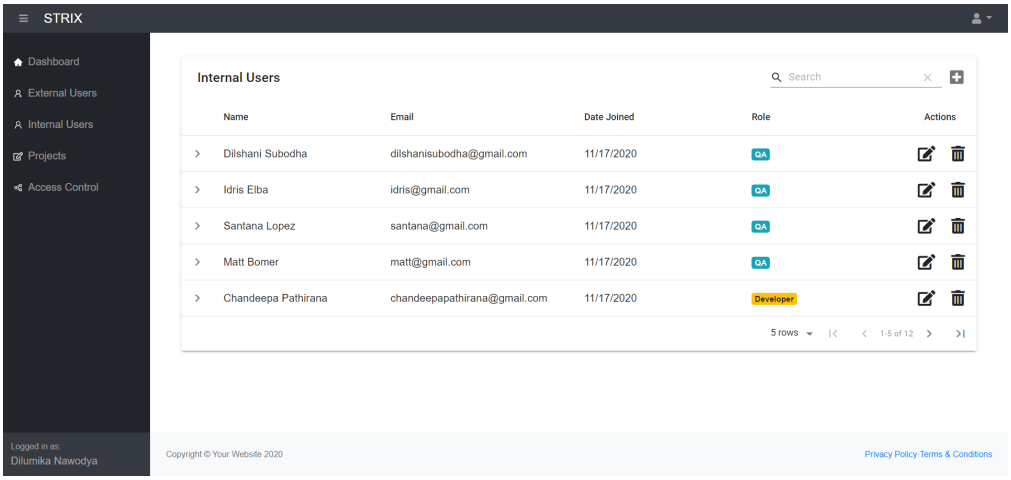
Admin

Admin Home

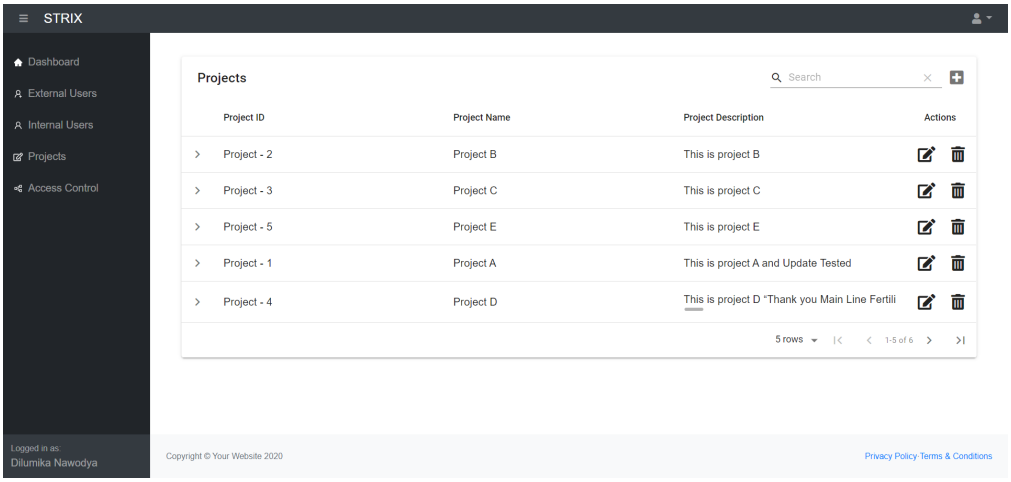


External User Creation / Internal User Creation

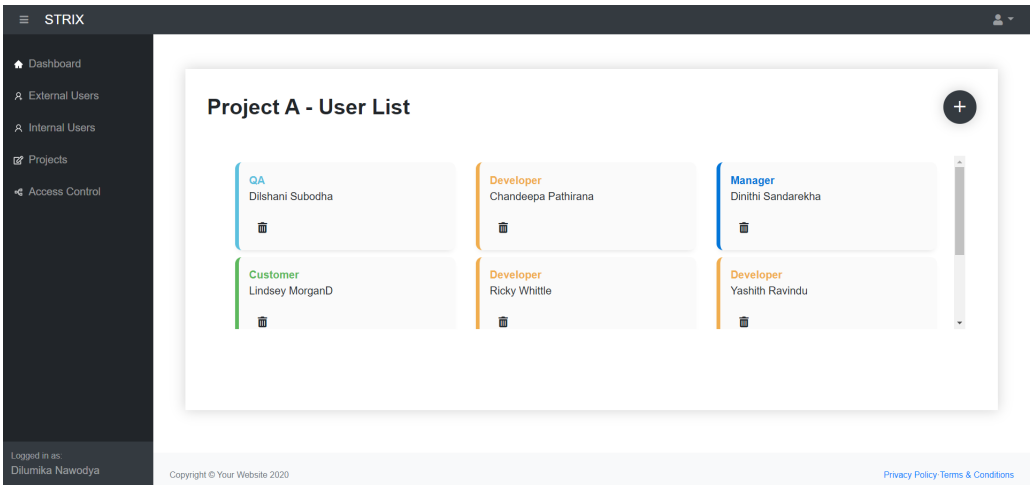


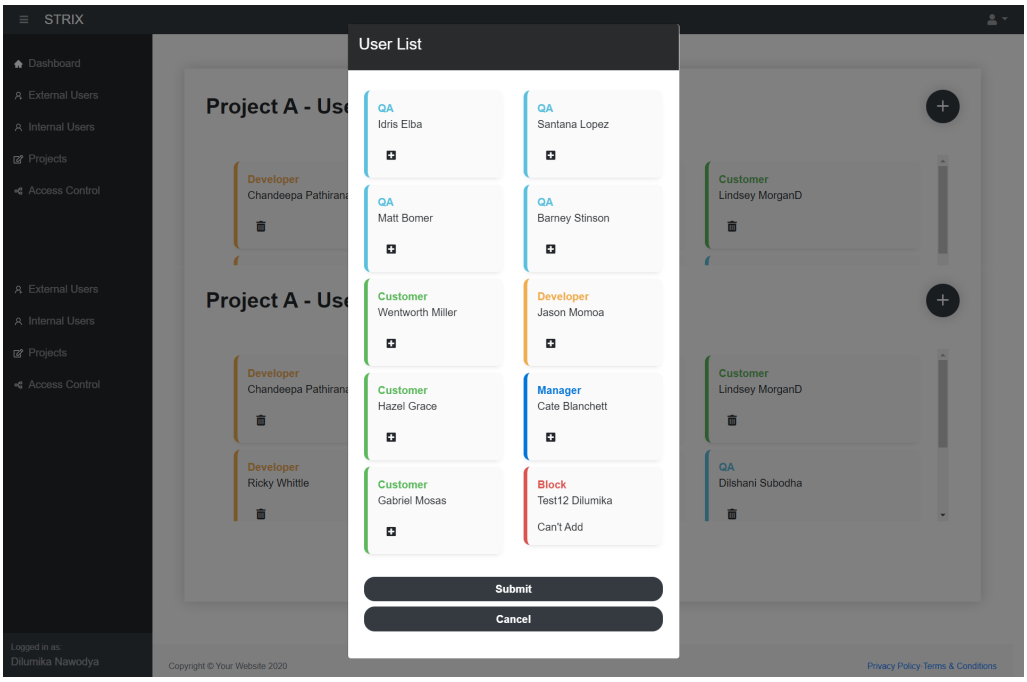


Project Creation



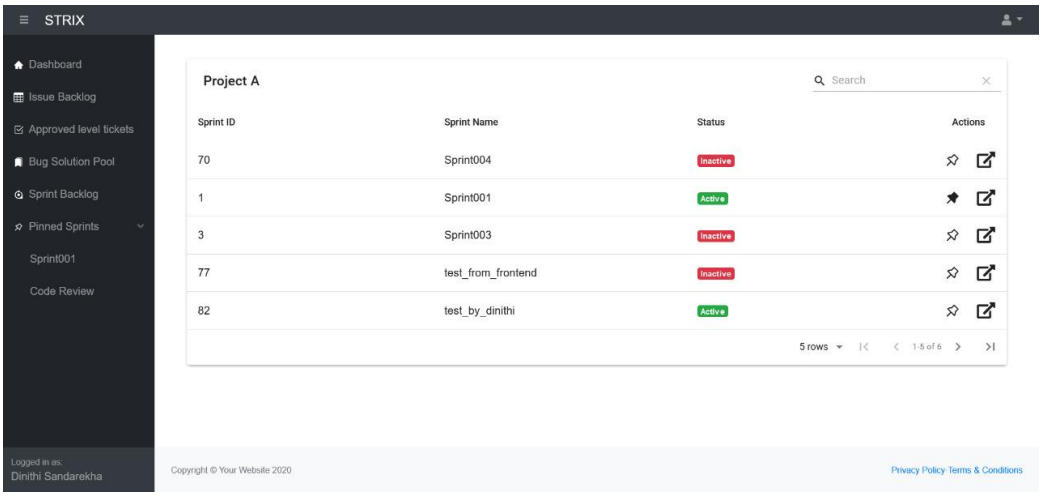
Access Control



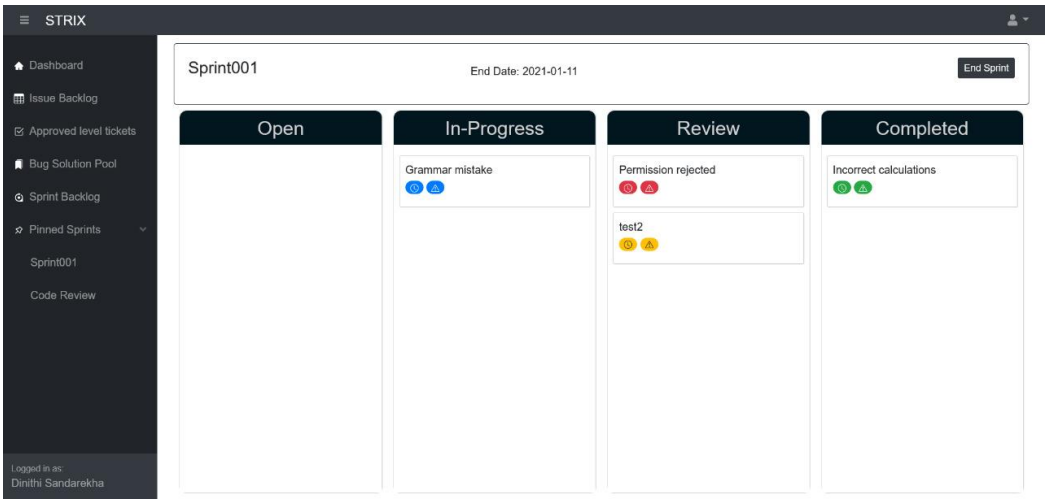


Manager

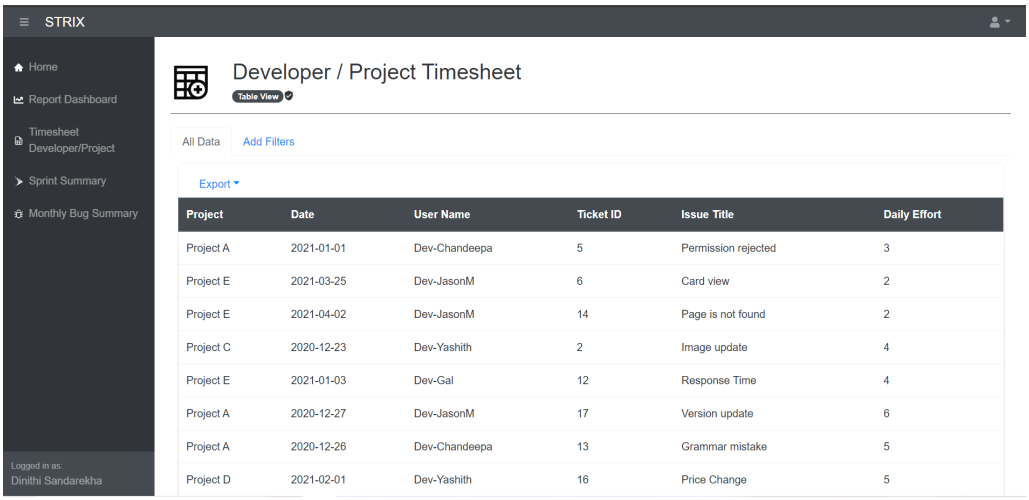
Sprint backlog



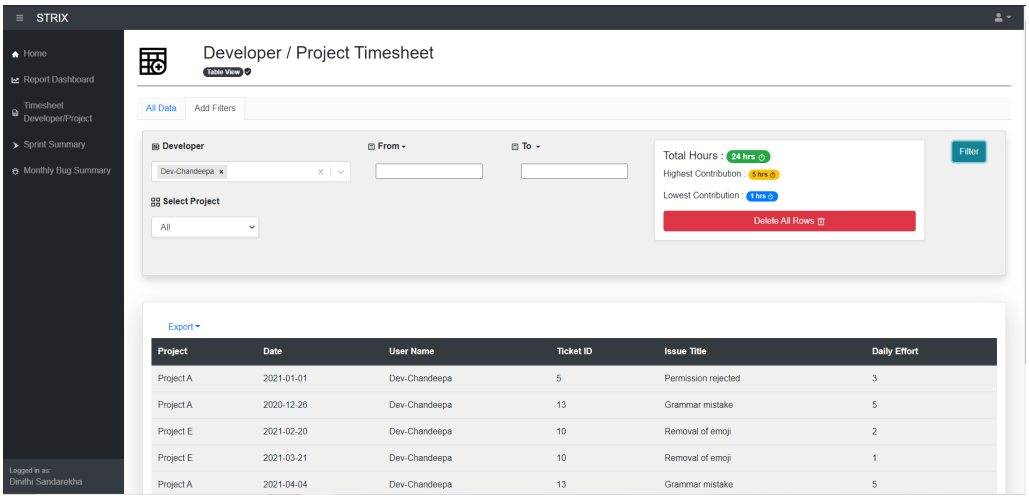
Kanban board



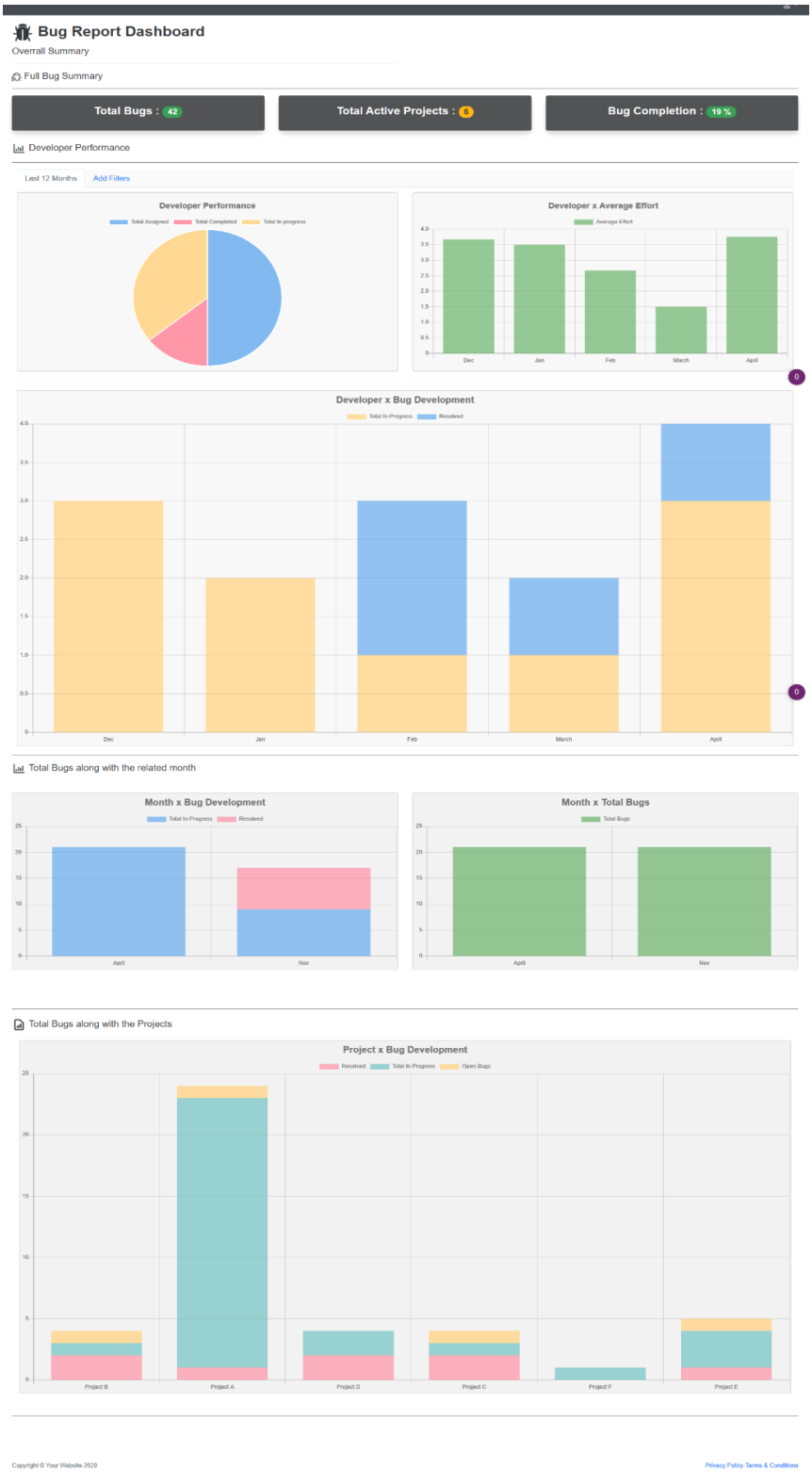
Developer / Project Timesheet



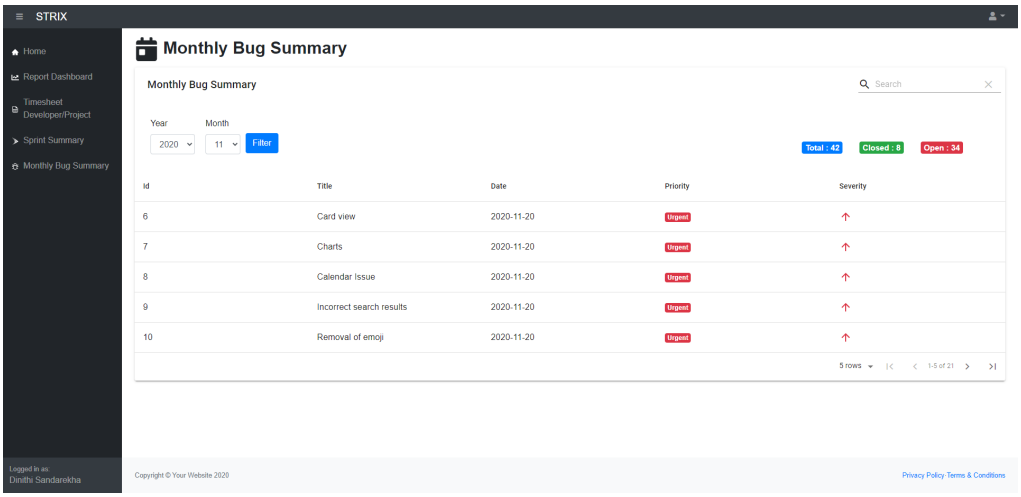
Developer / Project Timesheet filters



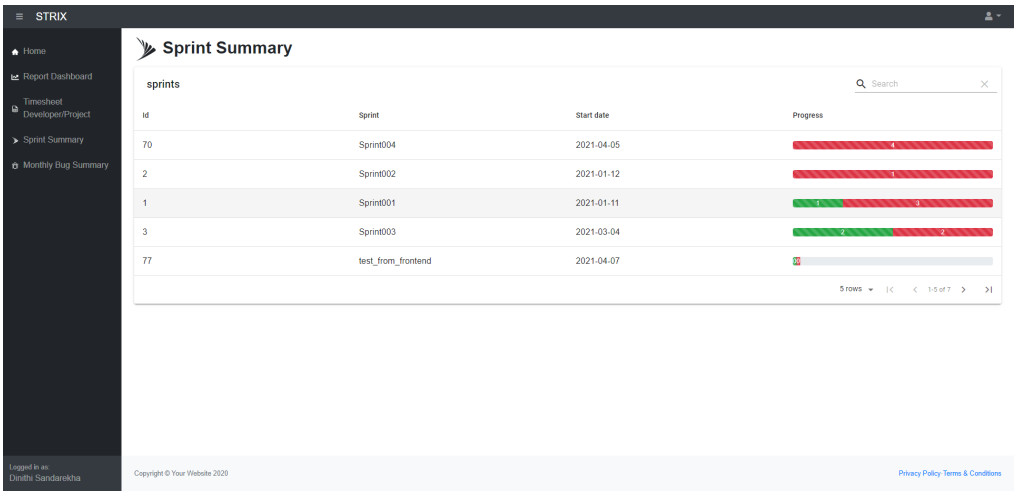
Report Dashboard



Monthly Bug Summary

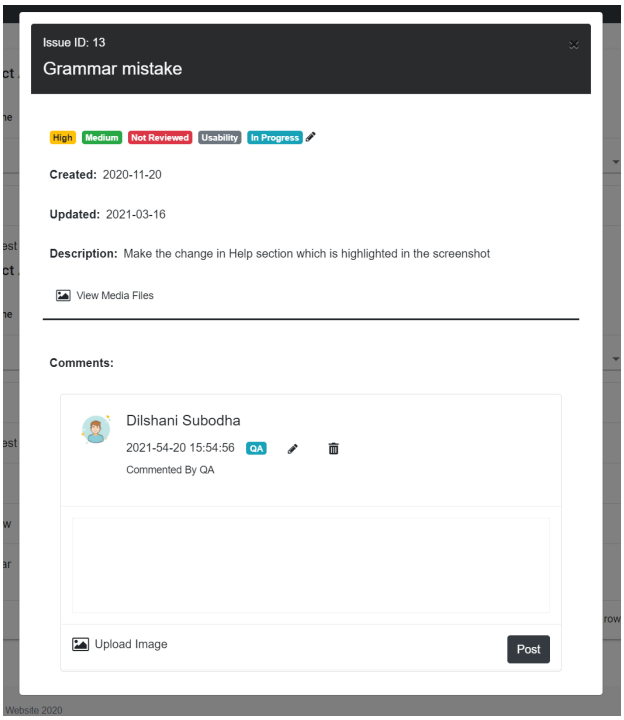


Sprint Summary

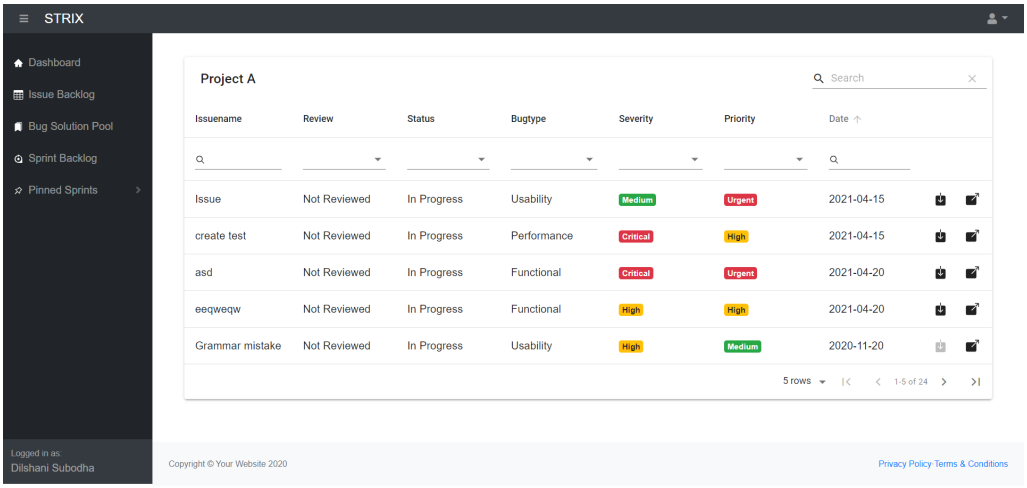


QA

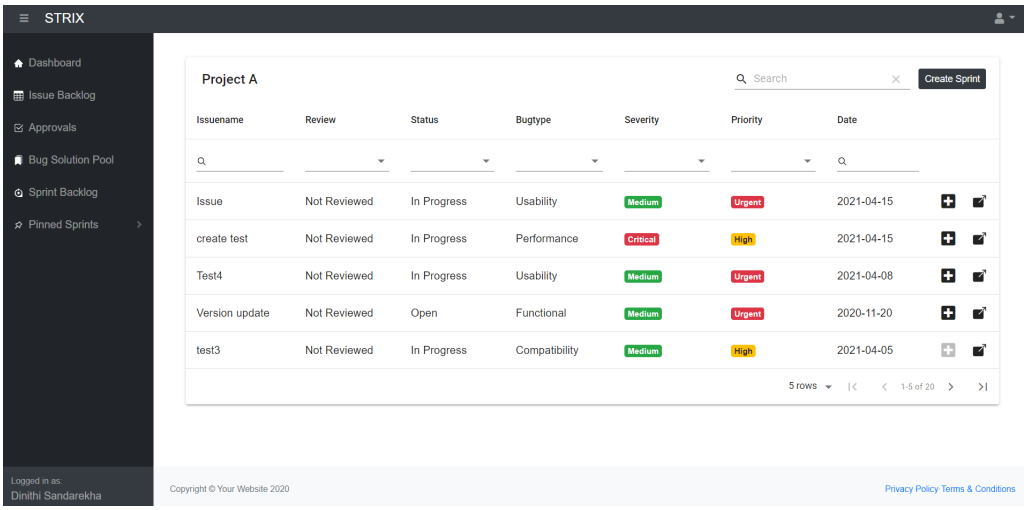
Ticket view



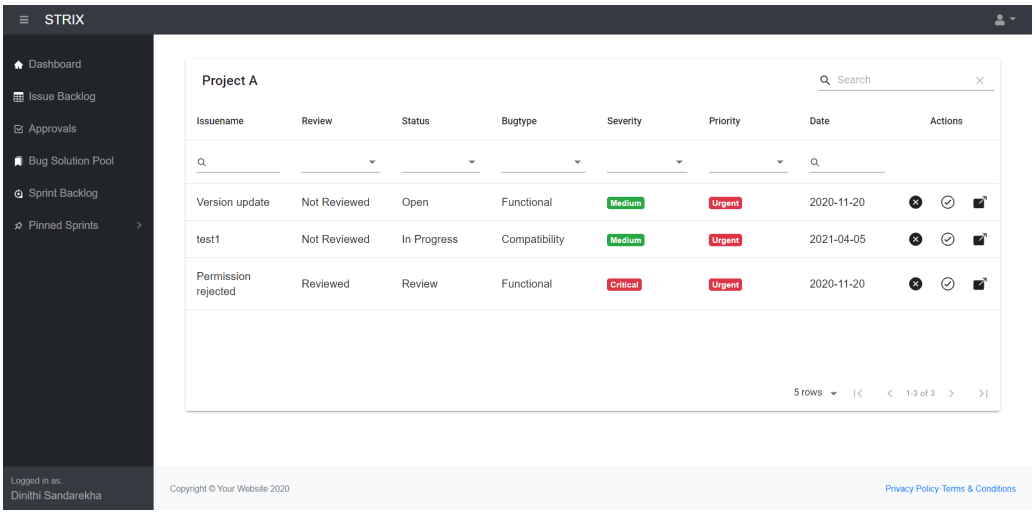
Issue Backlog QA



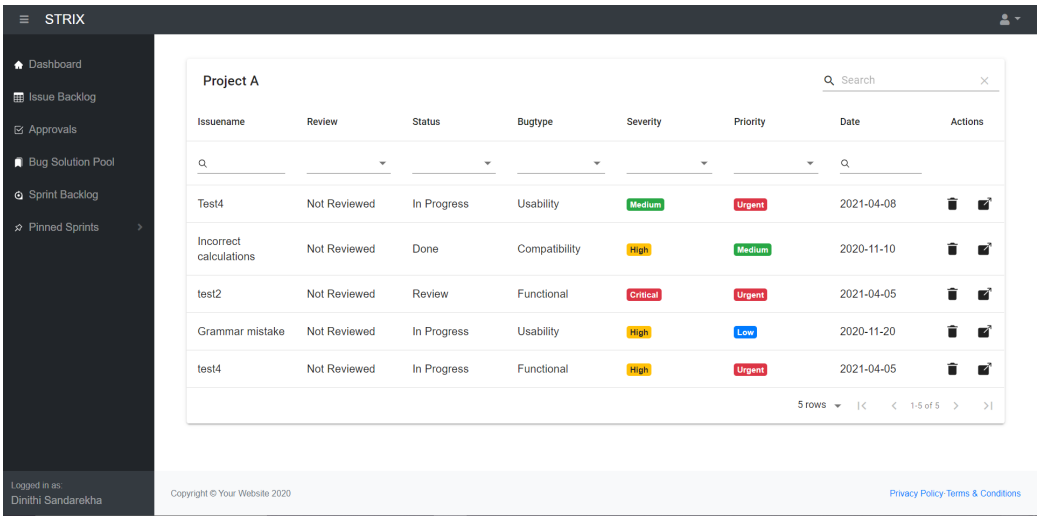
Issue Backlog Manager



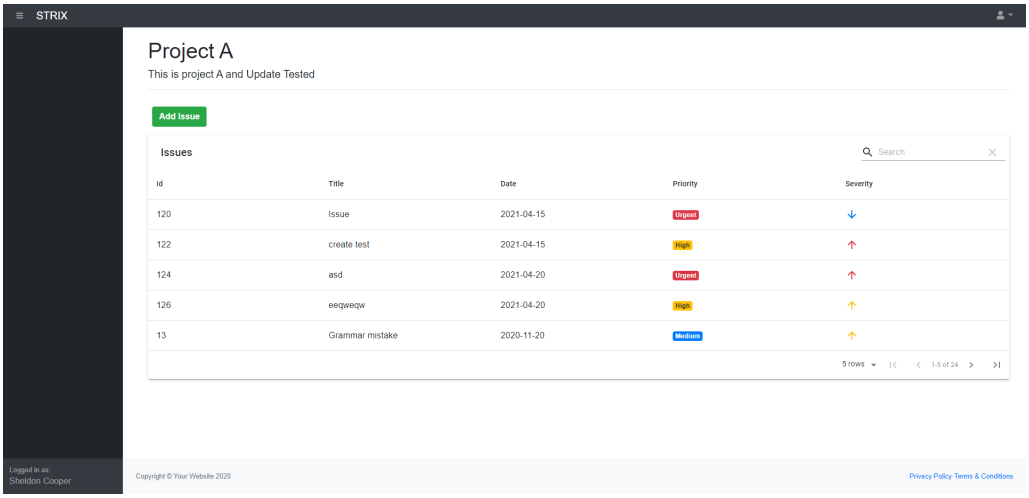
Approvals



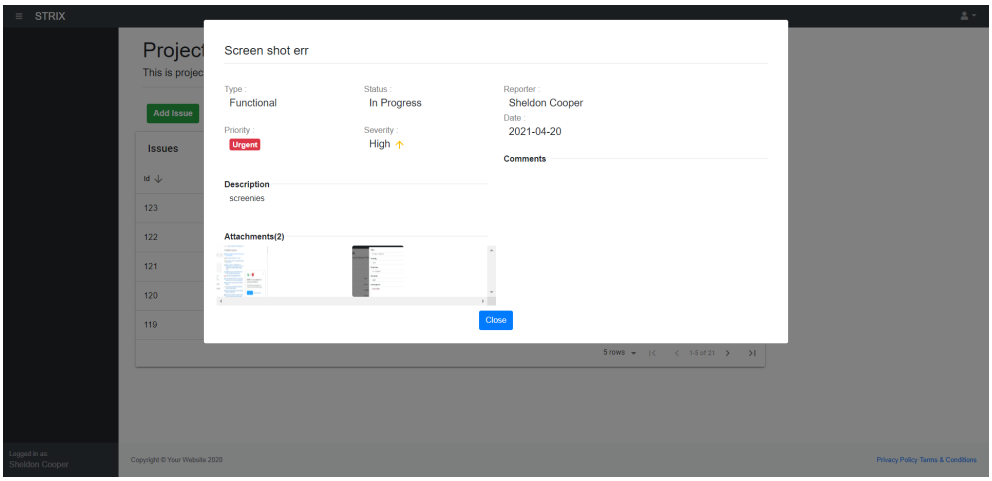
Bug Solution Pool



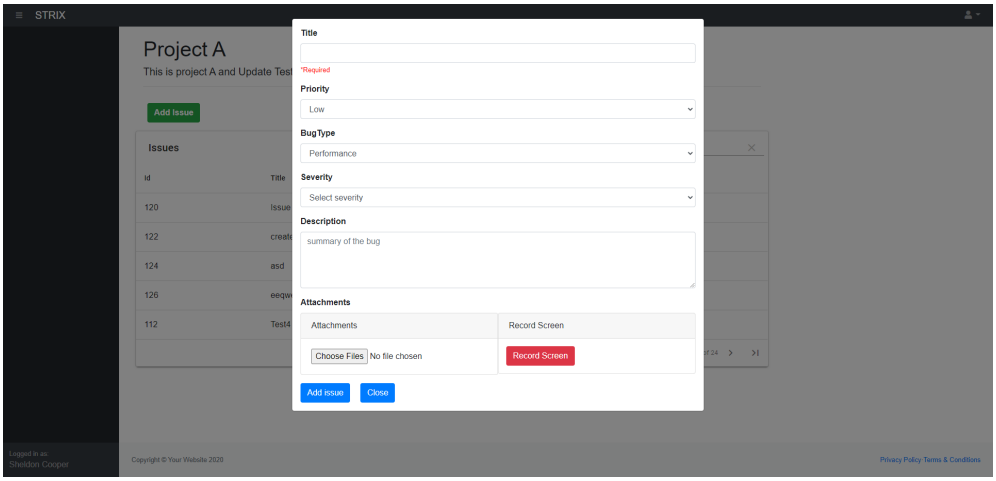
BCL Dashboard



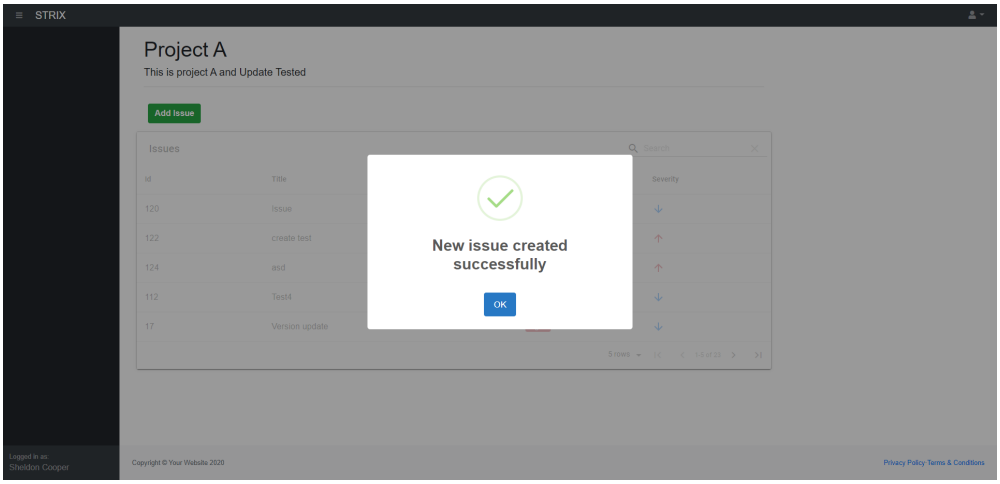
BCL Ticket View



BCL Issue submission form



Issue Submit Confirmation



6.3 Layered Architecture

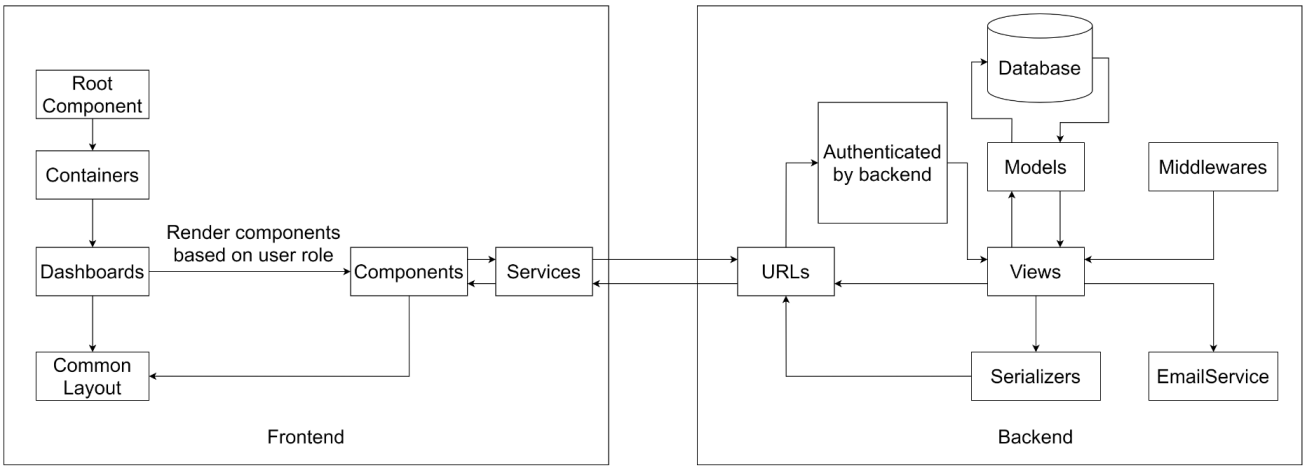


Figure 6. 1 – Layered Architecture

6.4 Summary

This chapter elaborated on how the implementation was done up to now with the use of layered architecture, the way the layers have been divided for easy access and decoupling using interfaces. Further it describes the general workflow for the system.

Chapter 7

Evaluation

7.1 Introduction

This chapter describes the evaluation and testing part of the system. To assure the quality performance of our system we have made different test cases based on the user roles and tested them. Within the scope, test results have been implemented.

7.2 Testing

We have tested our system grouped by user role . Each user performs specific tasks in the system and those tasks have been tested and results have been shown below.

Test Case No	Test Case	Expected results	Test Results
T1ALL_01	Reset Password (Email confirmation)	Confirmed modal or Rejected modal for email confirmation	Passed
T1ALL_02	Reset Password (Set new password)	Set new password and success or error modal	Passed
T1ALL_03	Issue Backlog search option	Preview searched user, project or ticket where they used	Passed
T1ALL_04	Sprint Backlog search option	Getting correct results for the searched phrase	Passed
T2ADMIN_01	Login	Able to access the bug management system and able	Passed

		to view admin functionalities as cards	
T2ADMIN_02	Create external users	New user added to the external user list	Passed
T2ADMIN_03	Update external users	Change details of the external user	Passed
T2ADMIN_04	Delete external users	Remove user from the external user list	Passed
T2ADMIN_05	Create internal users	New user added to the internal user list	Passed
T2ADMIN_06	Update internal users	Change details of the internal user	Passed
T2ADMIN_07	Delete internal users	Remove user from the internal user list	Passed
T2ADMIN_08	Change user role	Change user role to block or their default user role	Passed
T2ADMIN_09	Create Projects	New project added to the project list	Passed
T2ADMIN_10	Update Projects	Change details of the project	Passed
T2ADMIN_11	Delete Projects	Remove project from the project list	Passed
T2ADMIN_12	View all projects	View the list of projects as cards	Passed

T2ADMIN_13	Assign users to a project	Assigned users are able to access the project	Passed
T2ADMIN_14	Remove users from a project	Removed users are unable to access the project	Passed
T3BCL_01	Login	Able to access the bug management system	Passed
T3BCL_02	Select a project	Show related issues of the project	Passed
T3BCL_03	Select particular issue from the issue list	Show ticket view	Passed
T3BCL_04	Add a comment to a existing issue	Show added comment in the ticket view	Passed
T3BCL_05	Click add issue button	Display ticket form	Passed
T3BCL_06	Fill relevant information in the form	Validate the form	Passed
T3BCL_07	Attach files to issue form	Ready attached files to upload	Passed
T3BCL_08	Click on the screen record button	Show which window or screen to record	Passed
T3BCL_09	Click preferred screen to record and start recording	Start screen recording	Passed
T3BCL_10	Stop screen recording	Ask a place to save recording	Passed
T3BCL_11	Submit newly created issue	Confirm whether issue created or not	Passed

T4DEV_01	Login	Able to access the bug management system	Passed
T4DEV_02	View assigned projects	Directed to the assigned project dashboard	Passed
T4DEV_03	Upload files	Upload files in the comments	Passed
T4DEV_04	View media files	View previously uploaded media files	Passed
T4DEV_05	Add comments	Add new comments to the tickets	Passed
T4DEV_06	View issue Backlog	Display all the tickets with a table	Passed
T4DEV_07	Sort & Filter Issues	Show filtered or sorted data	Passed
T4DEV_08	View Sprint Backlog	Directed to the list of Sprints	Passed
T4DEV_09	View Tickets	View Issue details	Passed
T4DEV_10	Pin sprints to the sidebar	Pinned sprints visible in the sidebar	Passed
T4DEV_11	Access Kanban board	Directed to the Kanban board view	Passed
T4DEV_12	View Tickets in the Kanban board	View issue details by clicking a ticket	Passed
T4DEV_13	Add daily effort to the issue	submit daily effort through the ticket view	Passed

T4DEV_14	Move issues in the Kanban board except to the "Done" lane	Change workstate of the Issues	Passed
T4DEV_15	Mark issues to the Bug Solution Pool	Display alert message and disable the button	Passed
T4DEV_16	View tickets in the Bug Solution Pool	Display all the tickets which are accepted by the manager	Passed
T4DEV_17	Add comments to the issues in the Bug Solution Pool	View Comments Sections	Passed
T4DEV_18	Search issues in the Bug Solution Pool	Display tickets according to the search result	Passed
T5BMS_01	Login	Able to access the bug management system	Passed
T5BMS_02	View assigned projects	Directed to the assigned project dashboard	Passed
T5BMS_03	View media files	View previously uploaded media files	Passed
T5BMS_04	View issue Backlog	Display all the tickets with a table	Passed
T5BMS_05	Sort & Filter Issues	Show filtered or sorted data	Passed
T5BMS_06	View Sprint Backlog	Directed to the list of Sprints	Passed
T5BMS_07	Create Sprints	Appear new Sprints in the Sprint Backlog	Passed

T5BMS_08	View Tickets	View Issue details	Passed
T5BMS_09	Add issues to sprint	Appear added issues in the Kanban board	Passed
T5BMS_10	Pin sprints to the sidebar	Pinned sprints visible in the sidebar	Passed
T5BMS_11	End a sprint	Appear the Sprint as Inactive	Passed
T5BMS_12	Access Kanban board	Directed to the Kanban board view	Passed
T5BMS_13	View Tickets in the Kanban board	View issue details by clicking a ticket	Passed
T5BMS_14	Add total estimated effort to the issue	Submit total estimated effort through the ticket view	Passed
T5BMS_15	Approve issues to the Bug Solution Pool	Display a alert message and ticket will disappear from the table	Passed
T5BMS_16	Reject issues to the Bug Solution Pool	Display a alert message and ticket will disappear from the table	Passed
T5BMS_17	View tickets in the Bug Solution Pool	View all the tickets which are accepted before	Passed
T5BMS_18	Search issues in the Bug Solution Pool	Display tickets according to the search result	Passed
T5BR_01	Click on the 'Reports' tab on sidebar	Directed to the Report Dashboard	Passed

T5BR_02	Select a project , valid dates for 'From' and 'To' areas provided	Filter the charts according to all three categories	Passed
T5BR_03	Select a project name from the dropdown provided	Show all the data related to that project	Passed
T5BR_04	Select multiple projects from the dropdown	Filter all the data related to those projects	Passed
T5BR_05	Select valid dates in the given 'From' and 'To' area provided randomly	Filter the related charts according to date	Passed
T5BR_06	Select an incorrect date in 'From' field prior to 'To' date field	Pop up message will be shown for incorrect dates	Passed
T5BR_07	Click on the Developer/Project Timesheet tab on sidebar	Directed to the timesheet table	Passed
T5BR_08	Click on 'Export' tab on the top of the table	Show export table options in the dropdown	Passed
T5BR_09	Click 'Export Timesheet as PDF'	Download the table with the extension of PDF (.pdf)	Passed
T5BR_10	Click 'Export Timesheet as CSV'	Download the table with the extension of CSV (.csv)	Passed
T5BR_11	Click 'Export Timesheet as XLSX'	Download the table with the extension of xlsx	Passed

T5BR_12	Click on the 'Filtered Data' tab on the timesheet	Directed to the filtering options	Passed
T5BR_13	Select a project, a developer and valid dates for 'From' and 'To' areas provided	Filter the charts according to all four categories	Passed
T5BR_14	Select all 16 different states for 4 filtering options	Filter the data and map into the charts	Passed
T5BR_15	Select an incorrect date in 'From' field prior to 'To' date field	Pop up message will be shown for incorrect dates	Passed
T5BR_16	Click 'Export Timesheet as PDF' for filtered table	Download the table with the extension of PDF (.pdf)	Passed
T5BR_17	Click 'Export Timesheet as CSV' for filtered table	Download the table with the extension of CSV (.csv)	Passed
T5BR_18	Click 'Export Timesheet as XLSX' for filtered table	Download the table with the extension of xlsx	Passed
T5BR_19	Click on the Sprint Summary tab on sidebar	Directed to the Sprint Summary dashboard	Passed
T5BR_20	Click on a sprint in the sprint table	Show a modal of the summary of that particular sprint	Passed
T5BR_21	Click on the Monthly Bug Summary tab on the sidebar	Directed to the Monthly Bug Summary dashboard	Passed

T5BR_22	Add year and month to filter the table	Display the filtered table for the given month and the year	Passed
T6QA_01	Login	Able to access the bug management system	Passed
T6QA_02	View assigned projects	Directed to the assigned project dashboard	Passed
T6QA_03	View media files	View previously uploaded media files	Passed
T6QA_04	Add comments	Add new comments to the tickets	Passed
T6QA_05	View issue Backlog	Display all the tickets with a table	Passed
T6QA_06	Sort & Filter Issues	Show filtered or sorted data	Passed
T6QA_07	View Sprint Backlog	Directed to the list of Sprints	Passed
T6QA_08	View Tickets	View Issue details	Passed
T6QA_09	Pin sprints to the sidebar	Pinned sprints visible in the sidebar	Passed
T6QA_10	Change tags of the bug	Edit bug tags through the ticket view	Passed
T6QA_11	Access Kanban board	Directed to the Kanban board view	Passed
T6QA_12	View Tickets in the Kanban board	View issue details by clicking a ticket	Passed

T6QA_13	Move issues in the Kanban board	Change workstate of the Issues	Passed
T6QA_14	View tickets in the Bug Solution Pool	View all the tickets which are accepted before	Passed
T6QA_15	Add comments to the issues in the Bug Solution Pool	View Comments	Passed
T6QA_16	Search issues in the Bug Solution Pool	Display tickets according to the search result	Passed

Table 7.1

7.3 Summary

As for the conclusion, we defined 103 test cases based on each user role and tested the expected outcomes. According to our mentor's requirements we were able to complete and pass all the test cases our proposed solution is in a perfect condition to use as a bug management tool.

Chapter 8

Conclusion

8.1 Introduction

This chapter summarizes how the Strix bug management system differs from other bug tracking software and further development suggestions by the development team.

The rest of the chapter is organized as follows. Section 8.2 provides a description as to how our system differs from similar systems. Section 8.3 provides suggestions from the development team for further development of the system.

8.2 How our system differs similar systems

The majority of bug management systems are similar but differ in performance or deployment and user friendliness. Our system is more user friendly than others. It's because software like Jira and Bugzilla have complex user interfaces. It takes more time for users to become familiar with the software. Due to straightforward interface and due to inbuilt functions like screen-recording and kanban board our software is easy to use and anyone can quickly learn how it works. Our system also has an additional feature like screen recording. Clients can record the screen while the bug shown by the application is running. This allows the developer to identify the bug clearly. Moreover, our system is a free and open source role based application.

8.3 Further Development and Limitations

With the timeline, we were able to complete all the requirements which were given by our mentor. As for further development, we have few suggestions to optimize this bug management system.

1. Comment Section Optimization

a. Up to now except admin and manager, every user role (Customer, Developer, and QA) can add their comments as text or as attaching images. As for future development, we suggest adding video recording as well as cropped images by the comment module itself.

b. To avoid a particular user adding unnecessary, offensive, and harsh comments, we suggest training an AI module to filter and remove comments. When a particular comment gets flagged, it should be blocked for every user.

2. Include edit options in the uploaded images in the add ticket form

a. As for the given requirements, when adding a new ticket there are no edit options for the images. We suggest adding edit and crop options for the uploaded images to convey the message to the external users clearly.

3. Add exporting option for the charts

a. Up to now managers can only view and filter the chart data. As for future development, we suggest exporting those required charts and other tables into different file formats for the company usage.

4. Develop query language option for searching and filtering

5. Sprint task restrictions

a. As of now, the only conditional movement applied to the Kanban board is the restriction of developers to move issues to the “Done” lane. However, through further development, we suggest restricting the task movement based on the assigned user of the particular task.

8.4 Summary

In conclusion we believe we were able to develop the system to meet our company's requirements. By developing this system we were able to gain a good idea about the life cycle of the software development and we were able to learn a number of new technologies. We believe this product will be very useful when it comes to issue tracking due to its unique features and functionalities.

Reference

[1] "Jira | Issue & Project Tracking Software ...."
https://www.atlassian.com/software/jira. Accessed 6 Nov. 2020.

[2] "Jira Core - Project Management for business teams - Atlassian."
https://www.atlassian.com/software/jira/core. Accessed 6 Nov. 2020.

[3] "Jira | Issue & Project Tracking Software ...."
https://www.atlassian.com/software/jira. Accessed 6 Nov. 2020.

[4] "Jira Service Desk | IT Service Desk & ITSM Software - Atlassian."
https://www.atlassian.com/software/jira/service-desk. Accessed 6 Nov. 2020.

[5] "The Pros and Cons of Using Jira Software - Project ...." 6 Dec. 2019,
https://project-management.com/the-pros-and-cons-of-using-jira-software/.
Accessed 6 Nov. 2020.

[6] "7 reasons why people say they hate using Jira." 20 Feb. 2019,
https://deviniti.com/atlassian/why-people-say-they-hate-using-jira/. Accessed 6
Nov. 2020.

[7] "JQL: Get started with advanced search in Jira | Atlassian."
https://www.atlassian.com/software/jira/guides/expand-jira/jql. Accessed 6 Nov.
2020.

[8] "Timetracker - Time Tracking & Reporting | Atlassian Marketplace."
https://marketplace.atlassian.com/apps/1211243/timetracker-time-tracking-reportin
g. Accessed 6 Nov. 2020.

[9] "Jira Pricing - Monthly and Annual ...." https://www.atlassian.com/software/jira/pricing. Accessed 6 Nov. 2020.

[10] "Bugzilla." https://www.bugzilla.org/. Accessed 6 Nov. 2020.

[11] "Bugzilla Reviews & Ratings 2020 - TrustRadius." https://www.trustradius.com/products/bugzilla/reviews?qs=easy-functions. Accessed 6 Nov. 2020.

[12] "Jira Software vs Bugzilla Comparison | Atlassian." https://www.atlassian.com/software/jira/comparison/jira-vs-bugzilla. Accessed 6 Nov. 2020.

[13]“Jira Query Language” https://confluence.atlassian.com/jiracoreserver073/quick-searching-861257204.html#Quicksearching-Smartquerying . Accessed 6 Nov. 2020

[14]”7 reasons why use of jira can be frustrating” https://community.atlassian.com/t5/Jira-articles/7-reasons-why-use-of-Jira-can-be-frustrating-part-1/ba-p/1013802 .Accessed 7 Nov. 2020

[15]”Jira Pricing” https://www.atlassian.com/software/jira/pricing .Accessed 4 Aprl. 2021

Individuals Contribution to the Project

Name of student: M.D.N. Chandrasiri

Index: 184025L

At first, we had to prepare an initial report and a system requirements specification (SRS) for our system which is strix bug management system. Hence I contributed to both the report and SRS. As per the division of work I had to finish the authentication and authorization component of our system. It is basically a complete component where admins can create, update users and projects within the system. We decided to use react as our frontend library and django as our backend framework. Since I already had a basic understanding about JavaScript I started to learn react first. Then we realized that react hooks are better than react class components hence I learned react hooks as well. Apart from those I refreshed my knowledge about bootstrap, HTML and CSS.

Since I had a basic understanding about python and django I didn't work on learning python and django for a long time. But it was just a basic rather i had to learn to build a custom user model for strix database. Another important thing was we needed a cloud base database where all team members want to interact. Hence I followed some medium articles to establish a postgresql database in the cloud.

As per the work breakdown of the modules, I was assigned to complete the authentication and authorization module. I developed the Login and Logout submodule which has form validation using formik and yup. API was developed by using django rest APIView. Password reset is another needed module for any system out there. I developed a password reset page using formik and yup in frontend. For the backend I had to implement an email sending API using user credentials. For that I generate a random key along with a user token which was generated by the system and I generate a clickable link which redirects to the password reset form.

Apart from the authentication, I was assigned to create an admin panel which has both internal and external user creation, update and delete. Hence for that I developed tables which have sorting, filtering and pagination to view all users and each row of the tables represents a user where an action column also appears. I managed to add user update delete within that action column.

Though project creation is user activity, in our system project creation is handled by the administrator. Hence I developed a project creation sub module where administrators can add, edit and delete projects to the system.

I was assigned to complete the comment section submodule to the system as well. This module is used by bug capture log and bug management system module. Hence I developed a comment section and integrated it into the ticket view of a particular ticket. Within the comment section, according to the user role users have different levels of functionalities.

Ex: Managers can view comments but cannot add comments or delete others.

In addition to these, I learned about Github for version controlling and agile scrum framework.

Name of student: E.Y.A.R. Edirisinghe

Index: 184038E

Since I didn't have an understanding about how react works, I had to learn react first. I'm responsible for bug capture log and reporting modules in the product. Also while developing the project I had to learn about version controlling and learned about git version controlling theories. In those modules so far I've created bug reporting procedures for the customer, and implemented the screen capture facility and file uploading facility. While working on the project we had an issue regarding the styling of the UI. After discussions we agreed to use Bootstrap for styling and had to learn about it. Since React is based on modern JavaScript first I had various issues about js. Also after starting to work on issue form I had to figure out what is the content of the form. Since we are trying to create this more user friendly to customer I had to come up with a simple lay out for the form while providing all the details the developer wanted. After researching by looking and studying what other bug trackers' issue creating procedure, I came up with a simple layout. To create the form I had to figure out about react bootstrap and how it works too because bootstrap and react bootstrap were a bit different.
When it comes to form there is an issue because sometimes people submit their issue without providing essential details needed. While searching for a solution for that, I found a library called "Formik" and learned how to implement it for form validation. After implementing validation I started to work on file uploading and screen recording. For screen-recording I used a Media recorder web API supported by the browser. I developed a detailed view of an issue which shows the relevant details of a selected issue. Since we used the Material ui tables library for representing data, the in-build sorting part was already available in the library but it was limited to numerical and alphabetical sorting only. Because of that I manually implemented sorting by priority and by severity.

In addition to bug capture log, I contributed to reporting functionality by creating sprint summary and monthly bug summary. Basic functionality of sprint summary

is to summarize each sprint details into one place and functionality of monthly bug summary is to summarize details about bugs in a particular month.

Name of student: W.P.C.P.Pathirana

Index: 184116R

Our system is developed mainly with the technologies like react as the frontend, Django as the backend and PostgreSQL as the database. The modules I have taken are the BCL and reporting section. First I contributed to the project proposal and SRS while completing the wireframes of the above modules.

When it comes to BCL I have completed the BCL home which is a project view for a particular customer. As for the requirement, in the reporting section with the division of work, I handled 7 sub modules. In the report dashboard, I handled all the backend requests and frontend views including a full bug summary which includes total bugs, total active bugs and bug completion percentage.

As for the charts, I got several chart formats to handle which are bar, stacked bar and pie charts etc. For these charts I added filtering options as well. From that, charts can be filtered by date and project wise.

My next task is to develop a developer/ project timesheet to view details about the working hours and other information about developers. This also includes 4 filters which are date (to, from), project and developer filters. Finally these tables need to be exported in pdf and csv formats and I handled that option using external libraries.

Name of student: W.A.D. Sandarekha

Index: 184152X

As per the division of our work, I was assigned to contribute to the Bug Management System and the Bug Reports. Since I had a basic understanding of React and Python, I delved into learning additional concepts of React such as bootstrap and React Hooks and their implementations. We chose to use bootstrap as we decided that it made it easy for us to develop components. Then I moved on to learn about Python Object Oriented Programming concepts and the Django Framework.

During the development, I struggled with using bootstrap as I had not used it before. Furthermore, I faced the trouble of establishing a file structure and the integration of our codes. However, after having a discussion with our mentor, we managed to find solutions. Additionally, I had trouble with the initial database design as we had to change it frequently due to identifying new attributes and relationships.

During the development of the Sprint Backlog, the table component was changed many times as it was difficult to implement the sorting and searching functionalities. At last, we were able to find the material-table library which contained the necessary functions.

The Kanban board implementation was subjected to few changes according to the suggestions of the supervisor and the industry mentor. There were difficulties while figuring out the conditional workflow based on user roles as it was difficult to find proper documentation.

Apart from the frameworks, I also learned about the agile scrum development and the processes of Kanban boards extensively as it works as the concept behind our system. In order to integrate our code and to support version control I learned about Github as well.

Name of student: I.K.G.D.S.N.Senanayake

Index: 184157R

I was assigned to work on the Backlog in Bug Management System and the Bug Solution Pool module. I have some basic knowledge about HTML, CSS and JavaScript. I studied the React JavaScript framework. Then we decided to use react hooks. Because it is the most popular function-based implementation than class-based implementation. Then I learned and switched to react hooks. Also we have used Bootstrap css framework. Because it is much easier to use Bootstrap than to use css scripts. Such that I learned how to use Bootstrap in React and then styled the pages. When it comes to the Backend I need to work with the database on backlog and also on BSP. Therefore I learned about api and state management in react hooks to connect api url with frontend. When I implemented the frontend, I encountered some problems. Because the backend database was not created yet. Therefore I created a temporary fake backend api using a mock api site and temporarily connected the data with the frontend. After I implemented the frontend interfaces for the backlog, I moved on to learn Python OOP concepts and Django framework. I struggled sometimes because I did not work with Django before. I faced some issues regarding functionalities on my implementation part. After having discussion with our mentor and the supervisor, I managed to find solutions. During the implementation process of backlog tables, I tried multiple filtering options. But I failed to do that and I had that time included the filtering section limited to one column only. Because I did not use the Material table before. We were able to find the material-table library which contained the necessary functions. Then I switched to the Material table and it gave me a solution for the multiple filtering and also for the full text search. Apart from the frontend and backend technologies I learned about Github as well to integrate our code and to support version control.

Appendix

Note: To avoid making this report longer, we have provided only the links to the online materials of the following documents. Interested readers are referred to the online documentation for details.

Appendix A

System Requirements Specification Document (SRS)

Link: https://github.com/DiWeera98/IN2900Docs/blob/master/SRS.pdf

Appendix B

Project Management Plan

Link: https://github.com/DiWeera98/IN2900Docs/blob/master/Project%20Management%20Plan.pdf