Project Report

On

IssueQ

Submitted as a part of course curriculum for

Edwisor

Submitted by
SUBHANSHU BIGASIA

DECLARATION

I hereby declare that this submission is my own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person. Except where due acknowledgement has been made.

Signature

SUBHANSHU BIGASIA
Date:- 26/01/2020

ACKNOWLEDGEMENT

It gives me a great sense of pleasure to present the report of IssueQ undertaken during Edwisor training. I owe special debt of gratitude to Edwisor , for its constant support and guidance throughout the course of our work.

Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Signature

SUBHANSHU BIGASIA
Date: 26/01/2020

ABSTRACT

IssueQ is an issue tracking tool which allows the users to track the issues which are encountered during the software development life cycle. It is specially designed for I.T industry but is not limited to them. The applications consist of following modules-:

Login and Signup

This module allows users to login with their existing account or create a new account if the user has not registered.

Home

This module is the first screen that is shown to user upon login. It shows the different issues that are currently assigned to the user. If no issue is assigned to user, adequate message is shown to the user.

View All Issues

This module allows the user to see all the issues that are currently saved in database. The user can view any issue simply by clicking on the issue column.

Create Issue

This module allows the user to create an issue and assign it to someone. The user can access this module simply by clicking on Create Issue button from the taskbar.

Notification

This module allows the user to see his/her notifications. The notification contains information of activities that take place on issues. The user only receives notifications of issue which are added to his/her watchlist.

Logout

This module simply allows the user to logout of the application.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

During developing an application, the developer faces multiple issues. Be it an individual developer or a team of developers. These issues can range between 10-60 issues per sprint. Remembering all these issues is a tremendous task. Hence an application is needed to solve this problem.

This application solves this problem by providing specialized tools to keep a track of issues. It allows the user to report an issue, assign it to some other user. This issue can further be assigned to other users.

The users can also add issues to their watchlist so that they can keep an eye on the activities that are taking place in the issue. The application provides real-time notification to users and if by chance the user is not online, the activities are saved in the notifications. The user can access those notifications simply by clicking on the notifications tab.

1.1 Problem

The biggest problem that is faced while developing a project is the issues that come during the development. As we know no product is perfect in the first go and many issues are faced by the developer. But the issues can range from 10-50, hence an application is needed so that these issues can be tracked and the users can coordinate with each other on different issues.

1.2 Motivation

The idea of having an issue tracking tool where users can keep a track on the issues faced during development and coordinate with each other simplifies a developers life. It also allows to make efficient software as all the issues are tracked during development.

1.3 Goal and Criteria

The aim of this project would be to provide with an application which would simplify the process of resolving issues during development of software by providing tools to track and share issues and make an efficient software.

1.4 Overview

Chapter 2 describes LITERATURE REVIEW of Design & Tools used, description of the implementation, special knowledge required for this project and discusses about the selection criteria and describes why special frameworks, clients and tools have been chosen. Chapter 3 describes METHODOLOGY USED FOR IMPLEMENTATION, defines the features and requirements of the system which can solve the problem. Chapter 4 includes the description about the RESULTS obtained out of various observations and therefore discusses the other related factors involved in the complete procedure. Chapter 5 describes the CONCLUSION, overview of the solution for implementing the requirements and discusses about the project overall architecture and design. And it also includes References that describes the sources used while the course of the project that led to successful completion of this project.

CHAPTER 2

LITERATURE REVIEW

2.1 Selection Criteria

This Section discusses about the different selection opportunities for choosing the development frameworks, the web operating system and the possible tools which could have been considered reaching the target.

2.1.1 Frameworks

There have been some framework's attributes considered before making the final decision. First of all, I needed to choose open source frameworks. Proprietary frameworks are not appropriate since their codes cannot be accessed. Another reason of using an open source framework was to avoid an additional overhead which was required for buying the license. After that, it was advantageous to find the most popular frameworks. Another point was the framework's structure or language in which the framework has been written. Here I have chosen JavaScript language and its frameworks for the development of our projects and the languages and frameworks used as listed below.

Visual Studio Code

Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control and Github, syntax highlighting, intelligent code completion, snippets, and code refactoring. It us highly customizable, allowing users to change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

MongoDB

MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schema. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License (SSPL).

Angular 6

Angular is a TypeScript-based open-source web application framework led by the Angular Team at Google and by a community of individuals and corporations. Angular is a complete rewrite from the same team that built AngularJS.

NodeJS

Node.js is an open-source, cross-platform, JavaScript runtime environment that executes JavaScript code outside of a browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server- and client-side scripts.

ExpressJS

Express.js, or simply Express, is a web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node.js.

MLAB

mLab is a fully managed cloud database service that hosts MongoDB databases. mLab runs on cloud providers Amazon, Google, and Microsoft Azure, and has partnered with platform-as-a-service providers.

Amazon AWS

Amazon Web Services (AWS) is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. In aggregate, these cloud computing web services provide a set of primitive abstract technical infrastructure and distributed computing building blocks and tools. One of these services is Amazon Elastic Compute Cloud, which allows users to have at their disposal a virtual cluster of computers, available all the time, through the Internet.

Heroku

Heroku is a cloud platform as a service (PaaS) supporting several programming languages. One of the first cloud platforms, Heroku has been in development since June 2007, when it supported only the Ruby programming language, but now supports Java, Node.js, Scala, Clojure, Python, PHP, and Go.

2.2 Modules Overview

This section goes through the various modules that are included in this project and thus discussing them in detail and highlighting their importance and need for this project and the need and the scope of the project for the present time, and how it plays a game winner comparing to the classical methods used. The various modules included are as listed below.

2.2.1 Login and Register

This module allows the user to login into application with valid credentials or register themselves on the application.

2.2.2 Home

This is the first page that the user sees when he/she logs in. This module shows all the expenses that are assigned to the user. The user can view, delete or edit the expense.

2.2.2 View All Issues

This module shows all the issues that are currently present in the database. These issues are visible to every user. The user can view, edit or delete any issue. The user can further search for an issue from the search box.

2.2.3 View Issue

This module allows the user to view the details of a particular issue simply by clicking on that issue.

2.2.4 Notification

This module provides all the notifications to a user. The notifications are any activities that take place with an issue that is in the user's watchlist. The idea of introducing this module is to make the user never miss a single activity that took place with an issue even if he/she is offline.

2.2.5 Logout

This module simply allows the user to logout of the application.

CHAPTER 3

PROPOSED METHODOLOGY

This chapter discusses about features and requirements of the system that can solve the problem. The first section considers how the “IssueQ” platform process looks like in general. In this section, the use case diagram will show required features in detail. In the second section, the functional and non-functional requirements will be addressed for the mentioned features while making it easy in the upcoming sections to reference to different requirements.

As the choice of development, the waterfall method was chosen since the project's plan or project's requirements were unlikely to change and the testing results or customer's feedback were not supposed to effect the project. So each model's phase (project planning, analysis, design, implementation, etc.) was accomplished before moving on to the next phase.

3.1 Product Perspective

3.1.1 Product Functions

IssueQ provides the users with the following functions. Detailed use cases are given in the further section Functional requirements

- **Login**

This is the first screen encountered by user. If the user is registered with the database, he/she can login else he/she would be required to Register first.

- **Register**

Users that are not registered with the database can register by going to registration page. After providing the basic information, the user can register and further login.

- **View Issue**

This is the first screen encountered by a user after he/she has logged in. It shows all the issues that are currently assigned to the user. The user can view, edit, or delete the issues. The user can view any issue simply by clicking on that issue. The user can further edit or delete the issue by clicking on the buttons provided. The user can also comment on an issue.

- **View All Issues**

This function allows the user to view all the issues that are currently saved in the database. The user can view, edit or delete any issue. The user can view any issue simply by clicking on that issue. The user can further edit or delete the issue by clicking on the buttons provided. The user can also sort the columns by clicking on a column. The user can also search for an issue by using the search box provided.

- **Edit Issue**

This module allows the user to edit any issue. The user can edit an issue by clicking on edit button. The user is than required to fill the required information and click submit. If the user is wishing to assign the issue to another user, he/she can do that by using this module. Once the user has filled all the required details, he/she should click on Submit button.

- **Delete Issue**

This module allows the user to delete any issue. To do that the user simply has to click on the delete button. An alert message is popped up which shows that the issue has been deleted.

- **Notifications**

This feature allows the user to receive notification of issues. The notifications consist of any activities that take place with the issues that are currently in the user's watchlist. The idea of introducing this module is that the user does not miss any activities that take place with an issue.

- **Logout**

This feature simply allows the user to logout of the application.

3.2 Use Cases

IssueQ provides the following features to any end-user and the use cases for the same are given further.

- Login
- Register
- View Issue
- View All Issue
- Edit Issue
- Delete Issue
- Comment on an Issue
- Notifications
- Search Issues
- Logout

3.2.1 Login

Use Case Name	Login
Brief Description	The login process consists of getting the username and password of the customer and checking the authorization with the system.
Flow Of Events	1. Customer should click the sign in button. 2. The system displays login dialog box. 3. Customer should enter username and password. 4. Customer should click the login button.
Actors	Customer
Preconditions	The customer's account should exist on the system.
Post conditions	Customer will be able to use the other functionalities of the system.
Exception conditions	If the username and/or password are not correct, the login process is canceled and the customer would be notified.

Table 3 .1: Login use case description

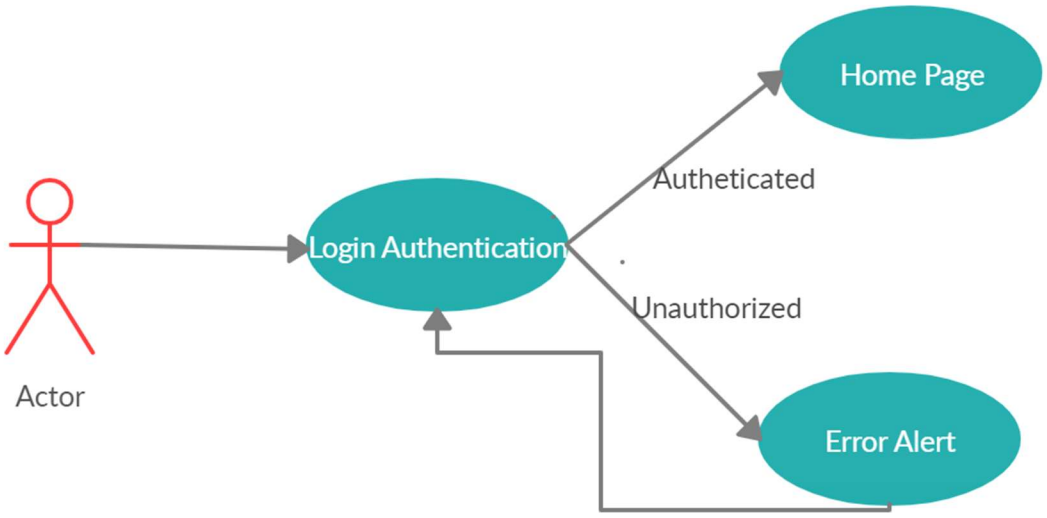

Fig3.1

3.2.2 Register

Use Case Name	Register
Brief Description	Customer should be able to register.
Flow Of Events	1. Customer should click on a register button. 2. Customer should fill valid details and click on Sign Up button.
Actors	Customer
Preconditions	Customer is on registration page and does not have an account.
Post conditions	Customer would be able to login using his/her details.

Table 3 .2: Register use case description.

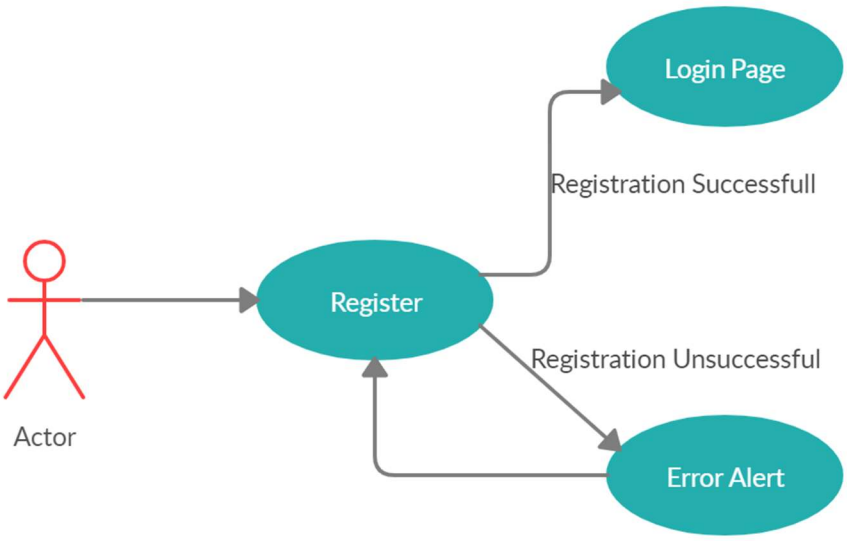

Fig3.2

3.2.3 View Issue

Use Case Name	View Issue
Brief Description	Customer should be able to view a particular issue.
Flow Of Events	1. Customer should click on a row to view that particular issue.
Actors	Customer
Preconditions	Customer should be logged in.
Post conditions	Customer would be able to view a particular issue.

Table 3.3: View Issue use case description

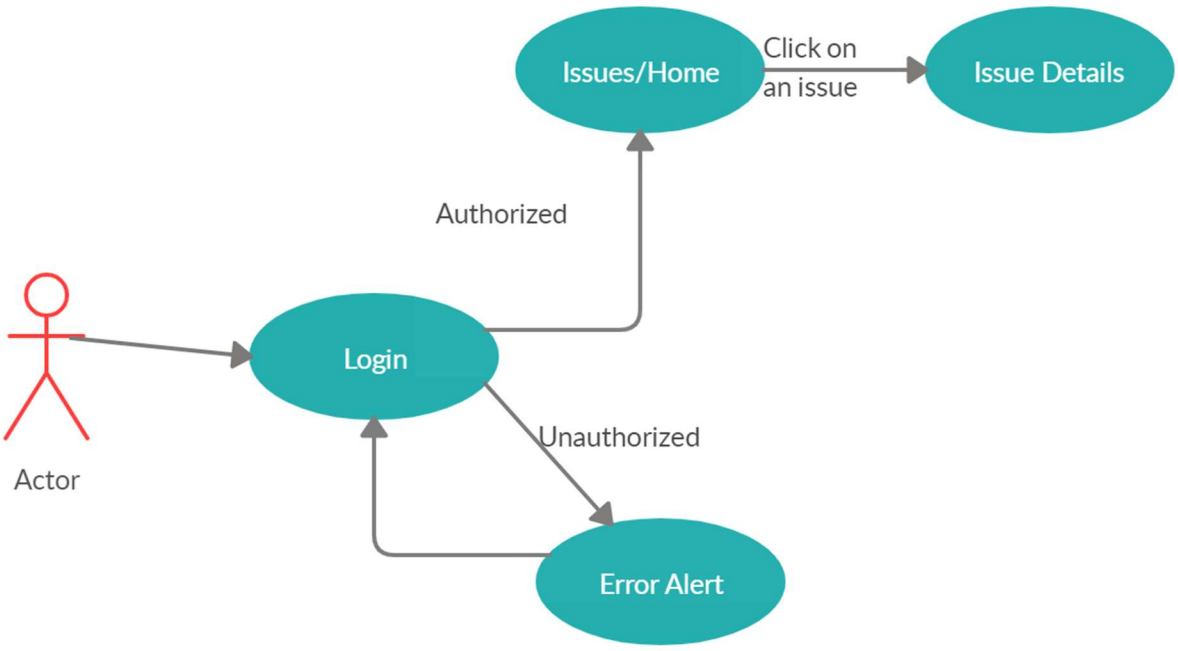

Fig3.3

3.2.4 View All Issue

Use Case Name	View all issues.
Brief Description	Customer would be able to view all issues.
Flow Of Events	1. Customer should click on view all issues button.
Actors	Customer
Preconditions	Customer should be logged in.
Post conditions	Customer would be able to view all issues.

Table 3.4: View All Issue use case description

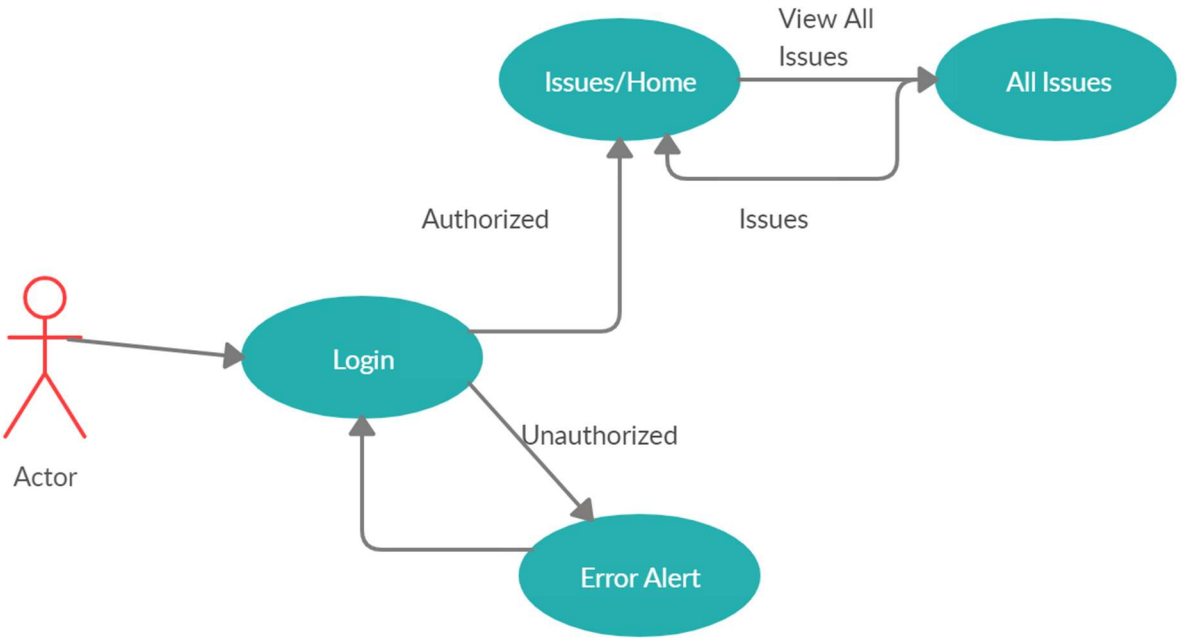

Fig3.4

3.2.5 Edit Issue

Use Case Name	Edit Issue
Brief Description	Customer should be able to edit an issue.
Flow Of Events	1. Customer should click on edit button of particular issue. 2. Customer should fill the valid details and click on submit.
Actors	Customer
Preconditions	Customer should be logged in.
Post conditions	Customer would be able to edit an issue.

Table 3.5: Edit Issue use case description

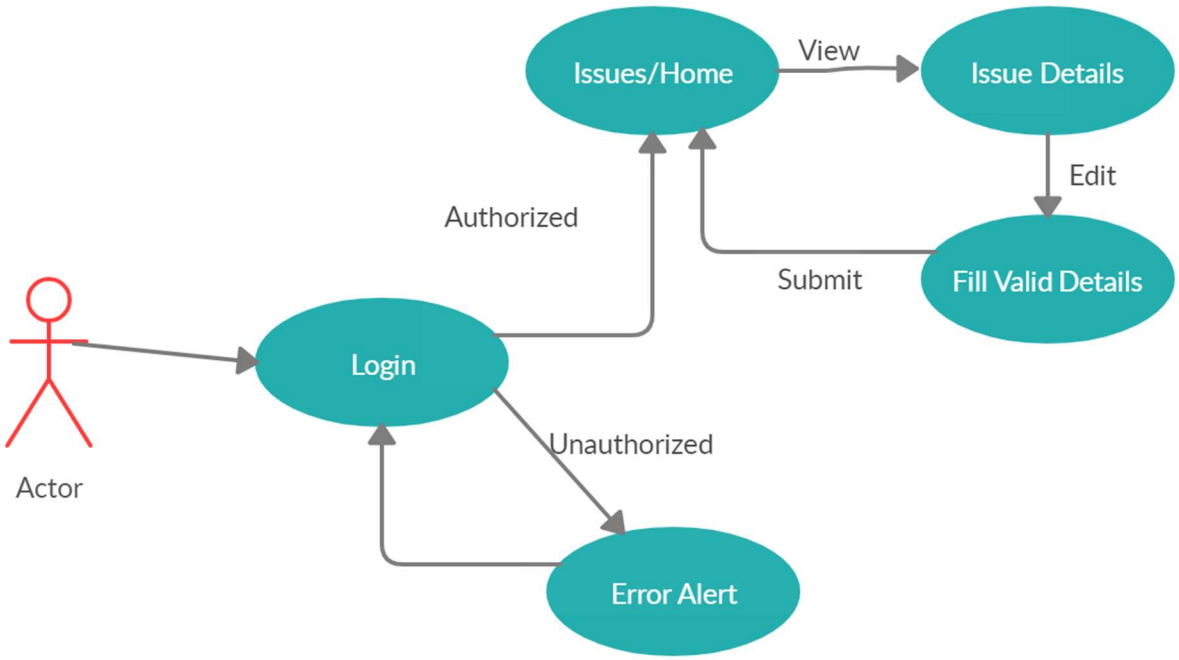

Fig.3.5

3.2.6 Delete Issue

Use Case Name	Delete Issue
Brief Description	Customer should be able to delete an issue.
Flow Of Events	1. Customer should click on delete button of particular issue.
Actors	Customer
Preconditions	Customer should be logged in.
Post conditions	Customer would be able to delete an issue.

Table 3.6: Delete Issue use case description

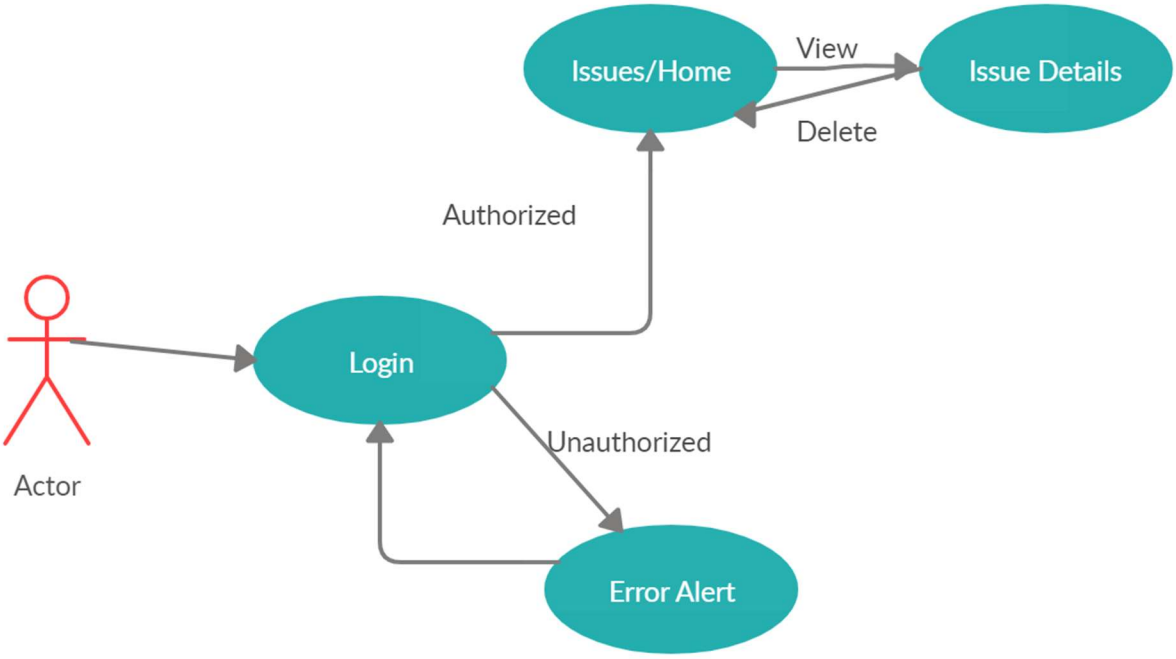

Fig.3.6

3.2.7 Comment on an Issue

Use Case Name	Comment on an Issue
Brief Description	User should be able to comment on an issues.
Flow Of Events	1. Customer should click on delete button of particular issue.
Actors	Customer
Preconditions	Customer should be logged in.
Post conditions	Customer would be able to delete an issue.

Table 3.7: Comment on an issue use case description

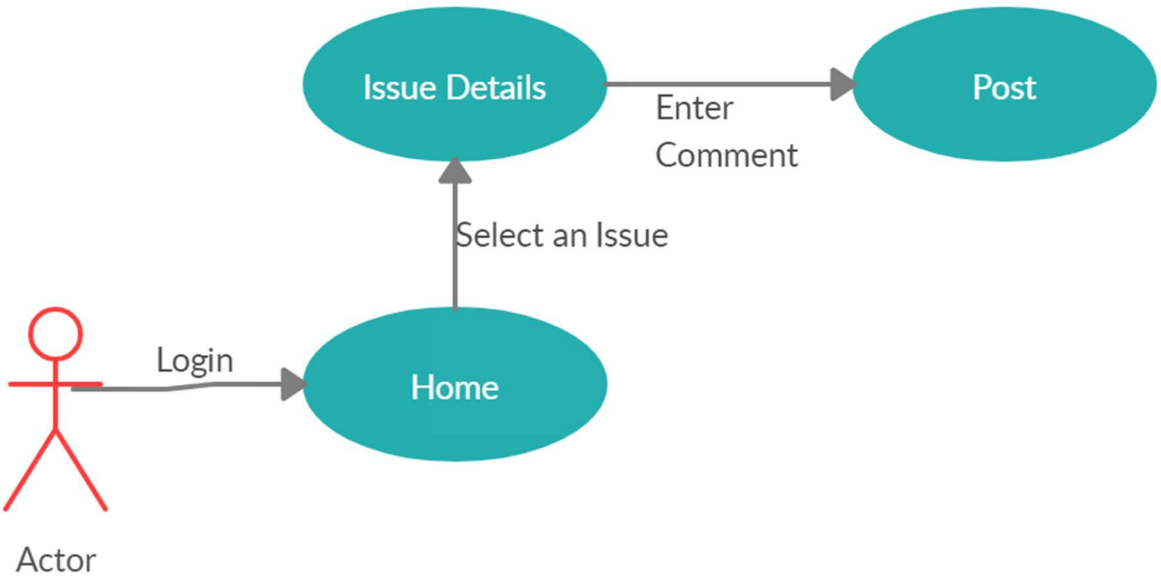

Fig.3.7

3.2.8 Notifications

Use Case Name	Notifications
Brief Description	Customer should be able to view all the notifications.
Flow Of Events	1. Customer should click on notifications button to view all the notifications.
Actors	Customer
Preconditions	Customer should be logged in.
Post conditions	Customer would be able to view all the notifications.

Table 3.8: Notifications use case description

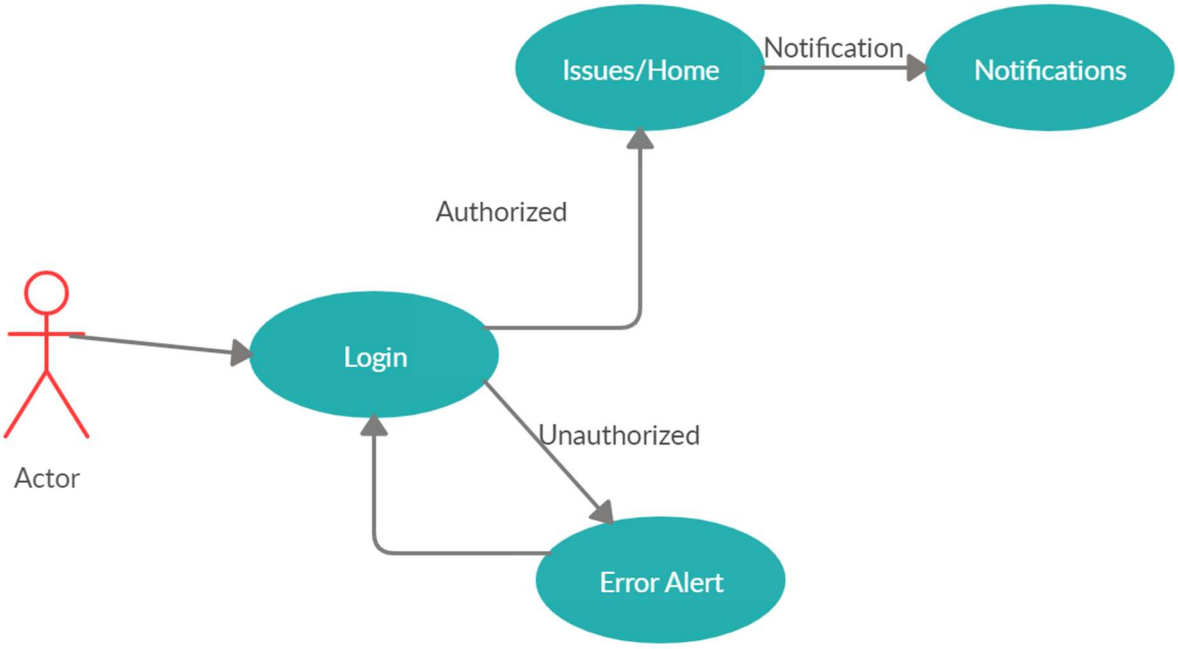

Fig.3.8

3.2.9 Search Issue

Use Case Name	Search Issue
Brief Description	Customer should be able to search a particular issue.
Flow Of Events	1. Customer should click on view all issue to view all issues. 2. Customer should type characters in the search box to search. 3. The search results are pushed inside the dropdown from where the user can go to a particular issue.
Actors	Customer
Preconditions	Customer should be logged in.
Post conditions	Customer would be able to search for an issue.

Table 3.9: Search Issue use case description

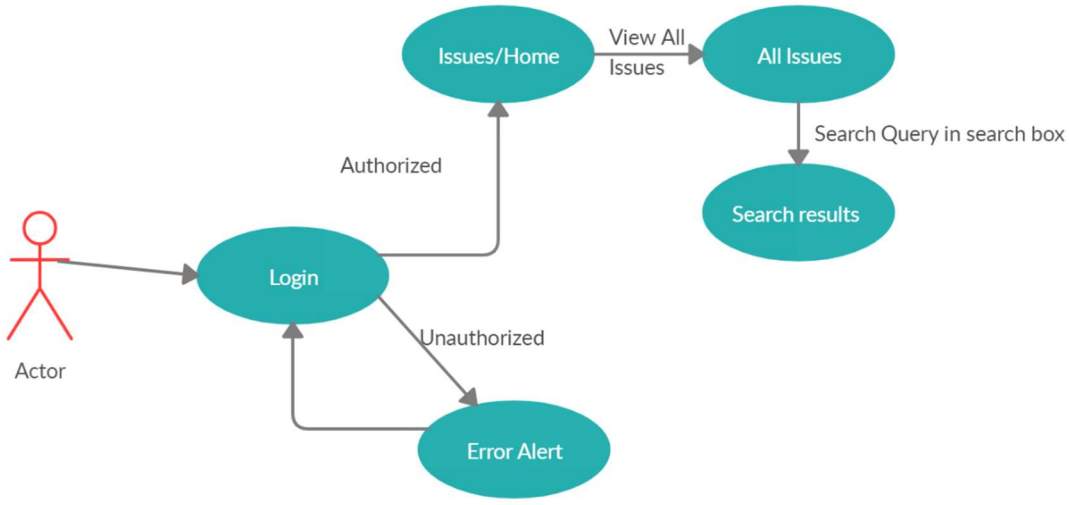

Fig.3.9

3.2.10 Logout

Use Case Name	Logout
Brief Description	Customer should be able logout from his/her account.
Flow Of Events	1. Customer should click on logout button.
Actors	Customer
Preconditions	Customer should be logged in.
Post conditions	Customer would be able to logout from his/her account.

Table 3.10: Logout use case description

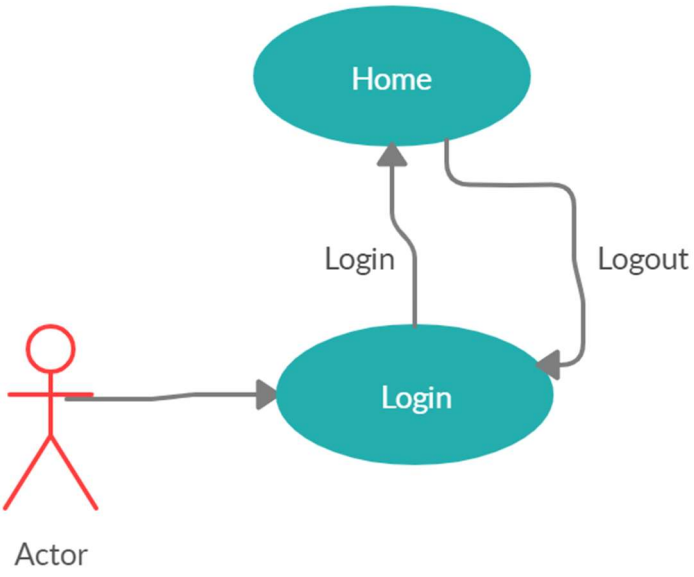

Fig.3.10

3.3 Non-Functional Requirements

In this section design constraints and software attributes are mentioned.

3.3.1 Design Constraints

In this section necessary hardware requirements are listed for the system. This includes database and the server that are needed to run a web application therefore I am now preferring for the ExperessJS server and MongoDB database server for our application.

3.3.2 Software Attributes

In this section software attributes are specified in order to verify them objectively.

1) Usability

As far as possible, the system should provide a simple, responsive interface. The system should be configurable from a single source at a central administrative position, and should provide a central, easy-to-use interface that will allow administrators to configure the user interface and features in a way that reduces page clutter. A system will be considered to meet this requirement if it has a single administrative interface rather than individual links for editing each page.

2) Reliability

The System must be backed up on a configurable schedule. Back-up requirements will need to be determined, based on individual components of the system. The system should be backed up with a frequency that ensures system and all data is protected. Since the updates and changes will be done to the database via web-interface, it should be backed up on a nightly basis, with options for weekly, as well as off-site backup when necessary. The system must have the ability and capacity to restore back-up data within five hours so that the system is never offline for an inordinate period of time.

3) Availability

The system should be available 24 hours a day, 7 days a week This statement provides a general sense of system availability. It is not intended to demand the system maintain reliability, or to require the system to be highly available. It should not exclude scheduled downtime.

4) Maintainability

The system must be maintainable without substantial modification. Due to limited number of administrators and support staff, it is important that the system be mostly self-sustaining. This will reduce the number of hours spent maintaining the system and simplify maintenance tasks.

5) Performance

The system should support at least 1000 concurrent users. This statement provides a general sense of reliability when the system is under load. It is important that a substantial number of actors be able to access the system at the same time.

6) Security

The system must comply with the permission roles Security is the most important attribute of system. System should not allow unauthorized accesses.

CHAPTER 4

RESULTS & DISCUSSIONS

4.1 Workflow & Results

This observation was made to know the working of the project and how does the application performs under varied scenario of input parameters i.e. keeping the track of the application performance, credibility and reliability as a whole. And as per the observation the application performs quiet fine over multiple range of problems tested over it with a view to cover a full-fledged range of inputs so as to test its performance for a regular use over a scalable load for the network.

4.2 Future Work

The future work of IssueQ would be consisting of integrating a chat application so that all the users can discuss issues on the platform itself. Also integrating the application with E-mail id's would be a great addition to this application so that mails regarding the activities that take place on issues can be send to mail id's to keep the users notified. Also to expand the ways to login, social login can be integrated in this application. The issue details can also be shared to other social networks.

CHAPTER 5

CONCLUSION & REFERENCES

5.1 Conclusion

This report aimed to show my project's requirement details in terms of several aspects. First, a brief summary of IssueQ is introduced. Then, a market and literature research is carried out and results are established. And at the main part, the requirement details of the project are described. As a last work, project's schedule is presented. This report tried to focus on the aspects which thought to be important. So, there is no part that reflects irrelevant or useless information. This report was very useful for clarifying the project's scope. Also, it'll be beneficial for further planning.

From the beginning to the end, the whole process will be heavy and challenging for me, but I will try to do my best. I believe that this product will take its place in the market and users will see what they expected.

5.2 References

[1.] Creately (https://www.creately.com)

[2.] Stack Overflow (https://stackoverflow.com/)

[3.] Wikipedia (https://www.wikipedia.org/)

[4.] Angular Documentation (https://angular.io/)