SOFTWARE REQUIREMENTS SPECIFICATION

for

Issue Tracker

Version 1.0

Prepared by : Riley Wium

December 29, 2020

Contents

1 Introduction

1.1 Purpose

This custom issue tracker was primarily made a personal tool and portfolio project. If the system is built robustly enough it can even be reworked into a bug tracker, project manager, or scrum board. The systems purpose is to allow for the management of tasks making scheduling easier and gaining data on the time and effort it takes to complete certain tasks. The system should allow for the creation, manipulation, and completion of task objects. The task objects are to be stored on a database and accessed via website. The project will include some functionality which is superfluous to my personal use of the tool but will serve it as a portfolio piece. Information about the design and progress of the issue tracker will be stored in this document.

1.2 Project Scope

The project will mainly use C#, ASP.NET, SQL, and the MVC design pattern. It will also likely incorporate Bootstrap and a third party encryption service OAuth. A small amount of jquery and javascript may be present as well.
The application will keep track of the task name, task due date, task priority, task descriptions, completion status, and the people assigned to the task. Users will be able to login with accounts assigned ID's and host projects, invite others, and be invited themselves. The user's can either be project owners or contributors, with the former allowing for more control over the project. The project will be focused on the Windows platform, but will be tested to see how it functions on mobile environments. All created data will persist in the database and only be accessible by the projects team via a browser.

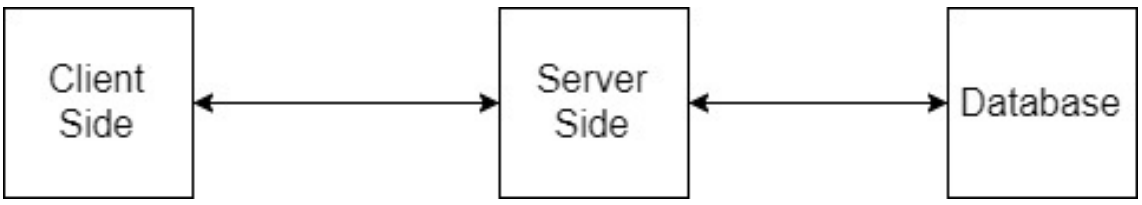

Figure 1.1: Issue Tracker System

2 Overall Description

2.1 Product Perspective

The project must be accessed via browser of which Chrome, Firefox, and Edge will be tested. Windows will be the main platform the project is developed for but Linux and Mac systems will be tested as well.

2.2 User Classes and Characteristics

The issue tracker has two types of users.

- Admins
- Project Owners
- Contributors

Project owners have all the permissions for their given project, contributors do as well except for Editing or Deleting the overall project.
Admins act as Project Owners for every project and are aptly equipped to administrate.

2.3 Product Functions

This Issue Tracker should include the following functionality.

2.3.1 User Interfaces

Issue Screen: Covers main issue tracking functionality. The user can post a issue for permitted projects to a list of total list of issues. The list of issues can be filtered in a variety of ways. A screen for adding issues will prompt the user for a name and additional info supplied.

Login Screen: Allows for users to access their account information and projects. Accompanied by a registration screen. The layout will be based on how the third party software functions.

Project Screen: Covers main project and team creation functionality. A project can be created or modified from this screen, as well as providing a link to join the project.

2.3.2 Synchronization Server

The server software will open a port on the network and will accept incoming connections with clients. Upon connection, the server will serve requested data to the client program. If the client program attempts to update a project, the server software will check the user's credentials to make sure that they have the rights to edit the projects. If they do, the server will update the database accordingly. If the data is modified on both server and client, the user will be prompted if they wish to push their changes.

2.3.3 Potential Functions

- email accessibility
- ticket dependency (ticket 2 can only be completed after ticket 1)
- email ticket creation
- easy to rework to bug tracker, project manager, or scrum board
- time tracker
- file attachments
- more user permission customization
- greater use of modals
- url encryption

3 System Features

The system will be a three-tiered design with a Server Application, a Database Engine, and Client Applications. The user will only interact with the system through client applications running on a PC. The general user interface will be forms with buttons and other familiar interactive tools. The user will use this interface to check-out a project, which will interact with the server application (via http) to retrieve the necessary data. The user will view and manipulate that data locally and then check-in the project, again through the server. The server will interact with the database to centrally store all the data for the system.

3.1 Description and Priority

The project has features that are main and also some are sub. Here is listed all the features necessary for this software.

Issue Board	
Purpose	Track, create, and manipulate issues for the project your working on
Priority	High
Input	Entered by the user. Includes: name, description, assignee, reporter, date reported, and comment
Processing	Validate input is of right type and format
Output	Recall previously stored information from database

Project Board	
Purpose	Track, create, and manipulate projects
Priority	High
Input	Input name to make a project. Input other username to invite to project
Processing	Validates data entered
Output	list of permitted projects

Login Page	
Purpose	Ensure account security in creation and login
Priority	Medium
Input	username and password to both create and login
Processing	checks if username is taken and if password is valid
Output	access to account

Synchronize Server	
Purpose	A server will open a port on the network and will accept connections with clients. Wait for incoming connections from a client computer
Priority	Low
Input	Inputs will come as predefined commands from the client computer
Processing	Server will follow the commands and output its result. If an invalid code is recieved the server will send an error message
Output	The server will send predefined codes and results to the client

3.2 Functional Requirements

Back-End - .NET framework, C# language, jQuery, SQL.
Font-End - BootStrap. ASP.NET MVC, JavaScript, HTML-CSS.
Database - SQL possibly MongoDB, or Azure.

4 Bibliography

- Latex Template by Md. Yasmi Tohabar and Akaash
- https://www.cs.utah.edu/ jamesj/ayb2005/docs/SRS.html#1.2
- https://www.programmer-books.com/wp-content/uploads/2019/07/ASP.NET-MVC-with-Entity-Framework-and-CSS.pdf
- https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/reading-related-data-with-the-entity-framework-in-an-asp-net-mvc-application
- https://www.youtube.com/watch?v=bIiEv__QNxwlist=PLJlxOV76kVjkJcSDKzQUD2UEbexV7DRJ channel=IAmTimCorey
- https://www.youtube.com/watch?v=phyV-OQNeRMlist=PLJlxOV76kVjkJcSDKzQUD2UEbexV7D channel=IAmTimCorey
- https://www.youtube.com/watch?v=weE1CwfS9rIlist=PLJlxOV76kVjkJcSDKzQUD2UEbexV7DRJ channel=IAmTimCorey