***Developers Guide***

Basic Chatting Site with User Authentication (SPJ Chatting)

Jimmy Hopkins, Shreejil Patel, Prabesh Mishra

**1.0 Introduction**

**1.1 Goals and Objectives**

The goal of the project is to develop a basic chatting site where users can communicate with others through a real-time messaging system. The objectives of the project are to ensure the project has an easy-to-use front-end experience, a secure system for storing users’ account information, and an appealing design.

**1.2 Statement of Scope**

The site will offer a real-time messaging system after registering/logging in to an account. Each created username will be unique as this will be used as their identity in the chatroom. The project will need to gather two inputs: user credentials (username and password) and messages sent from the user. The output will include displaying the messages sent in from the user.

**1.3 Software Context**

The basic chatting site contains HTML, CSS, and JavaScript to build an appealing front-end design. The back-end design is constructed with Node.js. Socket.io is used for the messaging system, bcrypt is used to securely hash passwords, and express is used to start the website server. MySQL is used to store user accounts securely in the local database.

 **1.4 Diagram of Project**

A diagram of a message

Description automatically generated

**2.0 Information for Developers**

**2.1 Goals and Objectives**

The goal of SPJ, the basic chatting site, is to develop a real-time communication platform with user authentication. This project allows users to register a secure account using bcrypt, a password hashing library. This safely protects a user’s account in the MySQL database. Additionally, these users can message other people in real time while the messages stored from previous sessions. The three developers accomplished this in the two-month timeframe.

**2.2 Performance Enhancement / Maintain Software**

A basic chatting site requires moderation. Users could potentially cause issues because it was created in a two-month timeframe. There could be potential bugs to arise from many users on the website at once. Developers can easily update how much the website can handle by cloud-hosting the website instead of hosting it locally depending on the traffic of the site. A feature that will ensure proper security is adding a change password feature in case a user needs to do this for any reason.

**2.3 Additional Information**

The website data can be accessed through my local MySQL workbench. Logging into this allows me to perform an SQL query to select each table with the updated information. This accounts for the chat messages and account user information. This is securely protected by a username and password on my local device to ensure security. If this needs to be changed in the future of the project, it can be hosted on a cloud server so that other moderators can access it securely.

Having a basic understanding of HTML, CSS, and JS for the front end as well as Node.js and MySQL for the backend is sufficient to grasp what our website accomplishes. Even if a developer is not familiar with all of the libraries in the project the concept is simple and can be understood by reviewing the code.