Software Requirements Specification

for

CLIENT TICKETING SYSTEM

By

Vansh Bajaj

Makam Ranga Rakshith

Prithak Gajurel

Duddu Bala Guru Venkata Arjun

Table of Contents

Table of Contents ii

1. Abstract 1

2. Scope 1

2.1 Product Perspective 1

2.2 Product Features 1

2.3 User Classes and Characteristics 2

3. Functional Requirments 2

3.1 Ticket Creation and Mangement 2

3.2 Ticket Assignment 3

3.3 Real-time Status Updates 3

3.4 Report and Analytics 3

4. Non-Functional Requirements 3

4.1 Security 3

4.2 Performance 3

4.3 Reliability 4

4.4 Usability 4

4.5 Compatibility 4

4.6 Notifications 4

4.7 Scalaility 4

5. Diagram 5

5.1 ER diagrams 5

5.2 Sequence Diagrams 6

5.2.1 log-in module……………………………………………………………………………6

5.2.2 Raising Ticket and Assigning module…………………………………………………..6

5.2.3 Updating Status and Notification module……………………………………………….7

5.3 Data diagram 7

# Abstract

In the realm of customer service and support, effective management of client inquiries and issues is paramount for organizational success. A client ticketing system serves as a structured approach to streamline this process, ensuring timely resolution and customer satisfaction. The client ticketing system acts as a centralized platform where clients can submit their queries or problems, which are then organized into tickets. Each ticket is assigned a unique identifier and categorized based on priority and type of issue. The ticket is passed on to the consultancy firm where the manager assigns the issue to the developer/consultant. The developer/consultant needs to update the status of the issue to the manager who is further answerable to the client. The system works with the help of flask and SQL for database management. By prioritizing efficiency, communication, and data-driven insights, businesses can foster stronger client relationships and achieve operational excellence.

# Scope

The Client Ticketing System is designed to streamline the process of managing client inquiries and issues, ensuring timely and efficient resolution. The scope of this project includes the following key aspects:

## Objectives

* **Centralized Ticket Management**: Create a unified platform where clients can submit their queries and issues, which are then organized into tickets for streamlined management.
* **Ff Efficient Ticket Assignment:** Facilitate the assignment of tickets to appropriate developers or consultants based on issue type and priority.
* **Real-time Status Updates**: Enable developers or consultants to update the status of tickets, providing managers and clients with real-time visibility into issue resolution progress.
* **Enhanced Communication**: Improve communication between clients, managers, and developers/consultants to ensure clarity and transparency throughout the issue resolution process.
* **Data Insights**: Generate reports and analytics to provide insights into ticket handling efficiency, types of issues and aiding in continuous improvement.

## Product Features

* **Ticket Creation**: Clients can submit tickets with a unique identifier, categorizing them by priority and type of issue.
* **Ticket Assignment**: Managers can assign tickets to developers or consultants based on their expertise and availability.
* **Status Updates**: Developers/consultants can update the status of tickets, which is communicated back to the manager and client.
* **Reporting and Analytics**: The system generates reports to track ticket resolution times, issues and overall performance metrics

## In scope

* **User Authentication and Authorization**: Implementing secure login mechanisms for clients, managers, and developers/consultants.
* **Database Management**: Utilizing MySQL for efficient data storage and retrieval of ticket information.
* **REST API Development**: Creating RESTful APIs using Python Flask for seamless interaction between the front-end and back-end components.
* **Data Processing**: Employing Pandas and NumPy for data manipulation and analysis.
* **User Interface Design**: Developing an intuitive user interface for clients, managers, and developers/consultants to interact with the system.
* **Notification System**: Implementing email or in-app notifications to inform users of ticket status changes.

### 3. Functional Requirements

#### 3.1 Ticket Creation and Management

* **Description**: Clients can submit tickets through a user-friendly interface. Each ticket includes essential details such as a unique identifier, issue description, priority level, and issue type.
* **Components**:
  + **Ticket Submission Form**: Clients fill out a form with relevant information.
  + **Ticket Storage**: The system stores ticket data in a MySQL database.
  + **Ticket Organization**: Managers can view and organize submitted tickets based on priority and issue type.

#### 3.2 Ticket Assignment

* **Description**: Managers assign tickets to developers or consultants for resolution.
* **Components**:
  + **Developer/Consultant Assignment Interface**: Managers select a developer or consultant for each ticket.

#### 3.3 Real-time Status Updates

* **Description**: Developers or consultants update ticket status, providing real-time visibility to managers and clients.
* **Components**:
  + **Status Options**: Developers can set a ticket’s status to “In Progress,” “Pending,” or “Resolved.”
  + **Notifications**: Managers and clients receive notifications when status changes occur.

#### 3.4 Reporting and Analytics

* **Description**: Generate reports and analytics to gain insights into ticket handling efficiency and performance.
* **Components**:
  + **Report Generation**: Automatically create reports on metrics like average resolution time and issue types.
  + **Analytics Dashboard**: Managers access visual representations of data.

# 4. Non-Functional Requirements

#### 4.1 Security

* **Description**: Ensure data security and user privacy.
* **Components**:
  + **Authentication Mechanism**: Secure login for clients, managers, and developers/consultants.
  + **Credential Storage**: Safely store user credentials.

#### 4.2 Performance

* **Description**: Optimize system responsiveness and scalability.
* **Components**:
  + **Quick Loading**: Efficiently retrieve ticket details.
  + **Scalability**: Handle a growing number of tickets and users.

#### 4.3 Reliability

* **Description**: Minimize downtime and ensure data integrity.
* **Components**:
  + **Backup and Recovery**: Regularly back up data and have recovery procedures in place.

#### 4.4 Usability

* **Description**: Create an intuitive user interface.
* **Components**:
  + **User-Friendly Forms**: Easy ticket submission.
  + **Consistent Terminology**: Clear labels and terminology.

#### 4.5 Compatibility

* **Description**: Ensure compatibility across devices and browsers.
* **Components**:
  + **Cross-Browser Compatibility**: Web interface works on major browsers.

#### 4.6 Notifications

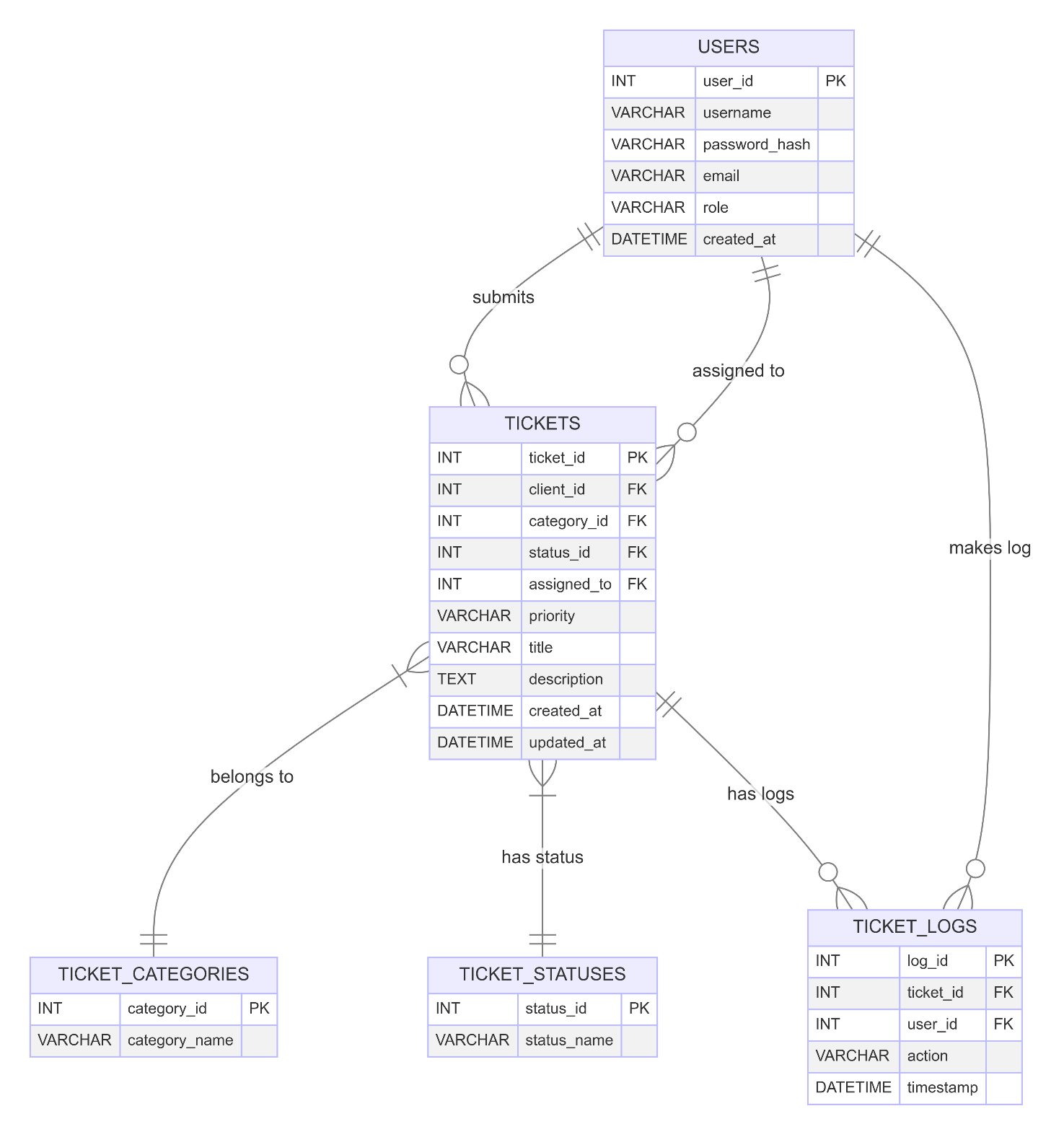
* **Description**: Keep users informed.
* **Components**:
  + **Email Notifications**: Notify users of new tickets, status changes, and comments.

#### 4.7 Scalability

* **Description**: Handle many concurrent users and tickets.
* **Components**:
  + **Load Balancing**: Distribute traffic efficiently.
  + **Resource Optimization**: Efficiently use server resources.

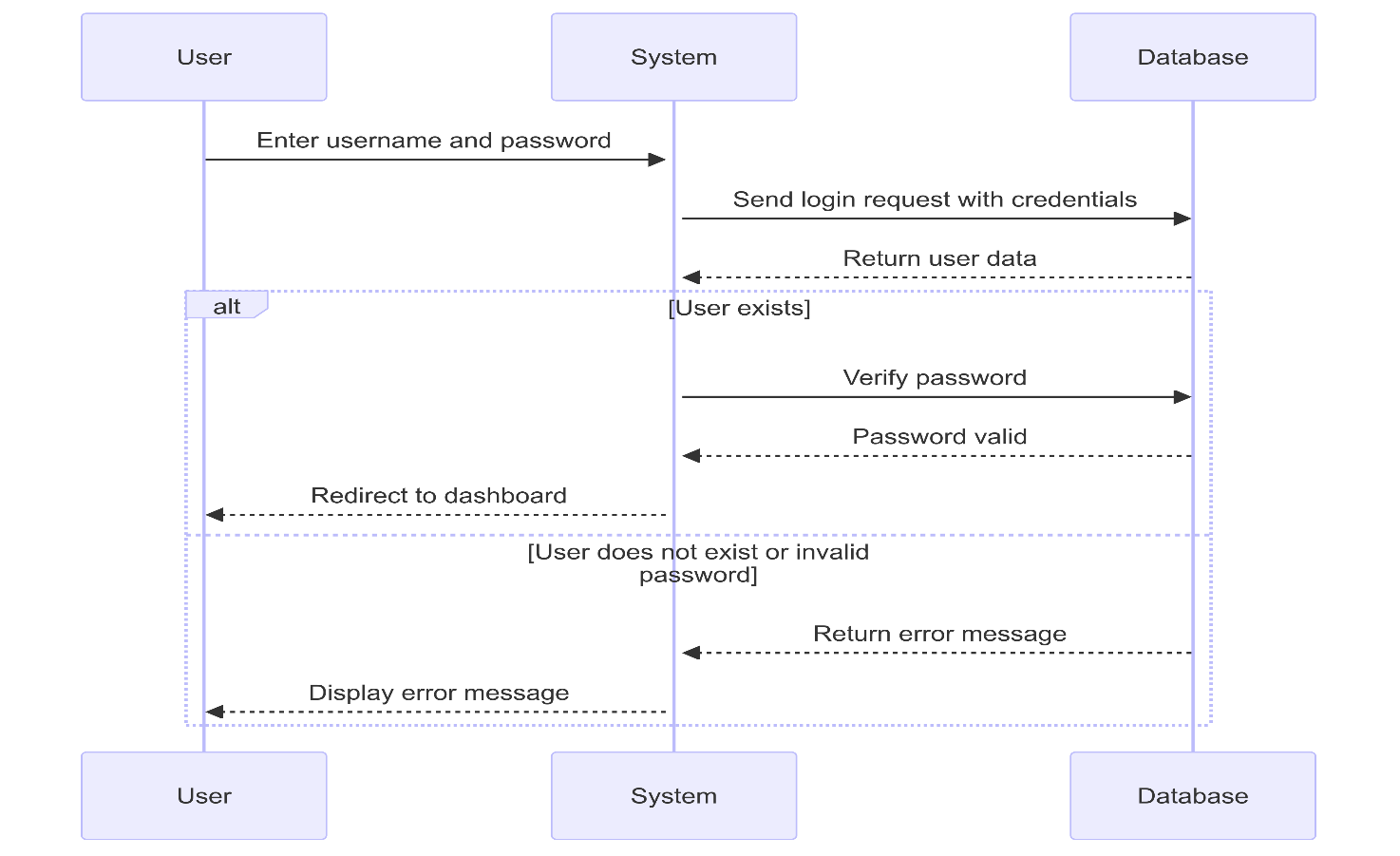
# 5. Diagrams

**5.1 ER Diagram**

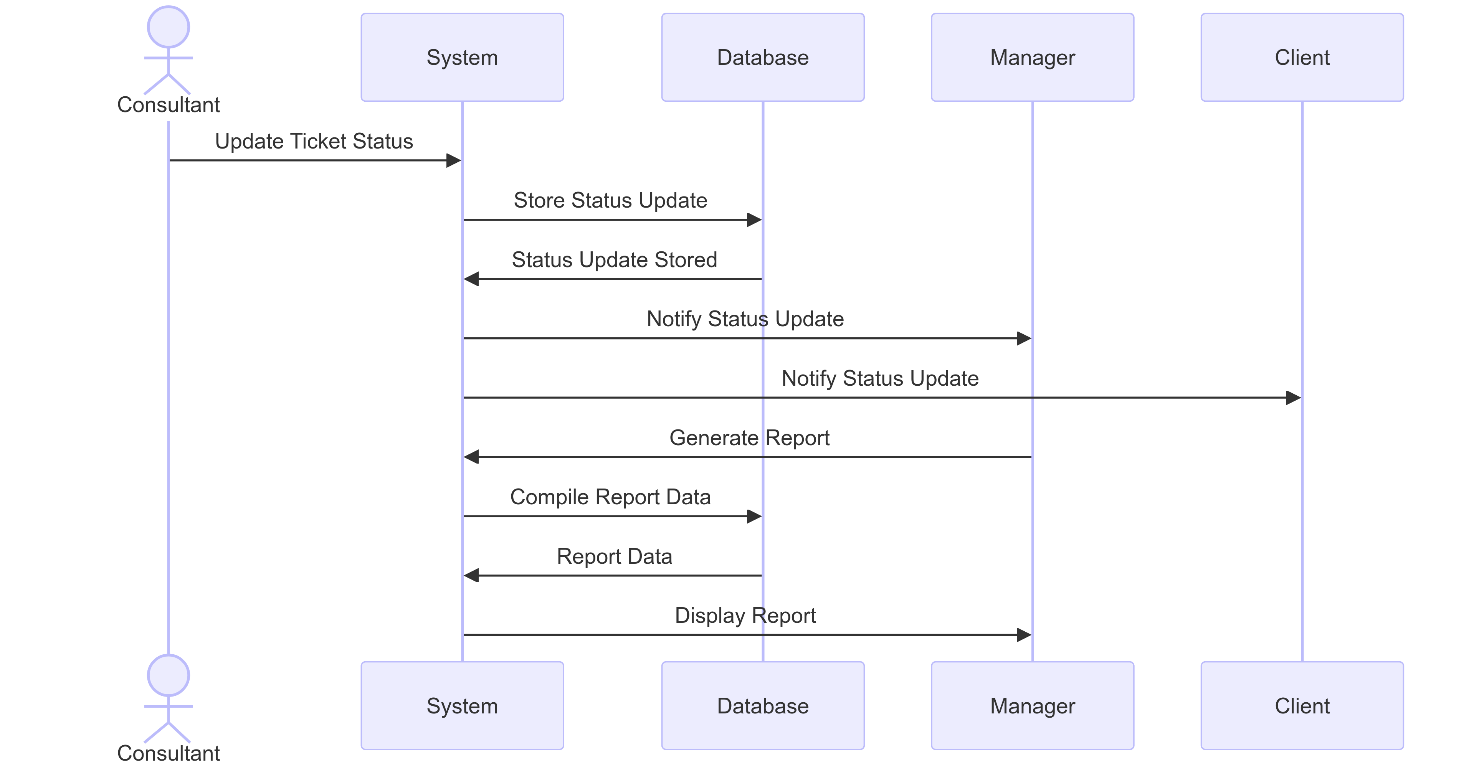


**5.2 Sequence Diagram**

5.2.1 log-in module



5.2.2 Raising Ticket and Assigning module



5.2.3 Updating Status and Notification module

A diagram of a company

Description automatically generated with medium confidence

A diagram of a flowchart

Description automatically generated**5.3 Data Diagram**