|  |  |
| --- | --- |
| **Project Title** | Plan, Schedule, Implement and Test Help desk system |
| **Qualification Name** | Bachelor of Technology in Software Engineering/  Higher Diploma in Software Engineering |
| **Product Name** | BDSE/HDSE - Capstone Project Web Development |
| **Module Name** | Capstone Project Web Development |

|  |  |  |  |
| --- | --- | --- | --- |
| **Student name** | | **Assessor name** | |
| Nithika Minlaka Jayarathne | | Ms.Ei Thandar Khaing | |
| **Date issued** | **Completion date** | | **Submitted on** |
| 21/11/2023 | 07/03/2024 | | 07/03/2024 |
|  | | | |
| **Project title** | **Plan, Schedule, Implement and Test Help desk system** | | |

|  |
| --- |
| **Learner declaration** |
| I certify that the work submitted for this assignment is my own and research sources are fully acknowledged.  Student signature: Date:07/03/2024 |

**TABLE OF CONTENTS**

**PROJECT PLANNING…………………………………………………………………..3**

Introduction ……………………………………………………………………………………………………..3

Problem Statement……………………………………………………………………………………………3

Project Objectives…………………………………………………………………………………………….6

Project Scope…………………………………………………………………………………………………….9

User Requirements………………………………………………………………………………………….16

System Requirements…………………………………………………………………………………….17

Gannett Chart…………………………………………………………………………………………………20

Milestone Schedule………………………………………………………………………………………..21

UML Design…………………………………………………………………………………………………….23

Logical Design………………………………………………………………………………………………..24

ER Design……………………………………………………………………………………………………….24

UML Case Diagram………………………………………………………………………………………..25

Flowchart 1.………………………………………………………………………………………………….26

Flowchart 2…………………………………………………………………………………………………..27

BSO……………………………………………………………………………………………………………….28

TSO……………………………………………………………………………………………………………….31

**DEVOLOPMENT AND DEPLOYMENT…………………………………………..32**

Used Back End Framework…………………………………………………………………………...36

Used Tool for Back End Framework ……………………………………………………………..39

Used Front End Framework…………………………………………………………………………...42

Used Tool for Front End Framework………………………………………………………………46

Database Framework……………………………………………………………………………………..50

**TESTINGS FOR HELP DESK PORTAL……………………………………………53**

**SCREENSHOTS OF HELP DESK PORTAL……………………………………….91**

**PROJECT SCENARIO**

* Manzaneque Limited is a real estate company that specializes in sustainable and eco-friendly properties. The company has been growing rapidly and is now expanding into new markets across the globe. As a result, the IT department is overwhelmed with requests from employees for IT support, and there is no formal system in place to handle these requests.
* To address this issue, the company is planning to implement a new IT Helpdesk system that will streamline the process of logging and tracking helpdesk queries. The new system will enable employees to submit requests online, which will be automatically assigned to helpdesk specialists based on their areas of expertise. The system will also provide real-time updates to employees on the status of their requests.

## **SYSTEM REQUIRED**

* When someone in the company has a problem, they can contact the helpdesk. One of the helpdesk operators will attempt to deal with the enquiry, but if an immediate answer cannot be given the problem is passed to one of several specialists. An Information System is needed to log and track the helpdesk queries. This will enable analysts to see how the equipment is performing overall, whether the helpdesk specialists are sufficiently resourced to solve problems in an acceptable time and whether there are subject areas where employee training is needed.

## 

## **PROPOSED SYSTEM OPERATION**

* When a new call comes into the helpdesk, the names of the caller and helpdesk operator are logged, along with the time of the call, the serial number of the computer and, if relevant, the operating system and software being used
* The caller’s name will be checked against a register of all personnel to retrieve the caller’s ID number, job title and department.
* Their equipment will also be checked against a register of equipment to find the equipment type and make. Their software will be checked to see if it is under a valid license.
* Every call is logged and each problem is given a problem number, which is supplied to the caller so it can be quoted on any subsequent calls about the same problem
* The helpdesk operator will also record notes and descriptions of the problem. A reason for each call is always recorded even if it is just a note to say how desperate the caller is getting (e.g., in the case of a follow-up call)
* When a problem is first reported, the helpdesk operator will also allocate a problem type, selecting it from a list of problem types. It is the skill of the operator to know what problem type is most relevant and how specific the problem is.
* Some problem types are refinements of more general problem types and so it is possible that the problem type allocation may be altered later if more information becomes available.
* When the problem area is identified the helpdesk operator can look up previous problems of the same type to see if the problem has occurred before and, if so, how it was resolved.
* It is also possible to look up previous problems with the same equipment or from the same caller to see if there were other related problems.
* If the problem can’t be solved immediately, the helpdesk operator will use the system to look up which specialist to refer the problem to.
* Each specialist will be an expert in one or more problem types.
* If there is no specialist listed for a more specific problem type, then a specialist from the more general problem type will be used.
* The system will also list how many problems the specialist is currently working on so that if there is more than one specialist for a problem type, the specialist who is currently the least loaded can be allocated.
* When a problem is eventually resolved, the helpdesk operator or the specialist will log the date and time it is resolved and record some indication of how the problem was resolved and the time taken to resolve the problem.

**PROJECT OBJECTIVES**

1. Efficient Problem Resolution:

* Enable the helpdesk operators to efficiently resolve hardware and software problems for the company's IT systems.
* Provide a systematic process for logging and tracking helpdesk queries.

1. Documentation and Analysis:

* Log essential information for each call, including caller's name, helpdesk operator, time of the call, computer serial number, and relevant software details.
* Retrieve additional information about the caller, such as ID number, job title, and department.
* Check equipment and software against registers to ensure validity and proper licensing.

1. Problem Categorization:

* Allow helpdesk operators to allocate a problem type to each reported issue, selecting from a predefined list.
* Enable refinement of problem types based on additional information, if available.

1. Knowledge Base and Historical Data:

* Maintain a knowledge base that allows operators to look up previous problems of the same type to facilitate quicker resolutions.
* Provide the ability to check historical problems related to the same equipment or from the same caller.

1. Specialist Allocation:

* Identify and assign specialists based on the type of problem reported.
* Consider specialists' workload to ensure optimal allocation and prevent overloading.

1. Resolution Tracking:

* Record the date and time when a problem is resolved.
* Document the resolution details and the time taken to resolve the problem.

1. User-Friendly Interface:

* Design an intuitive and user-friendly interface for helpdesk operators and specialists.

1. System Performance Optimization:

* Optimize the web application system for performance efficiency.

1. Training and Resource Planning:

* Provide insights into overall equipment performance and the sufficiency of resources for problem resolution.
* Identify areas where employee training is needed based on recurring problems.

1. Comprehensive Testing:

* Develop and implement a comprehensive testing plan to ensure the functionality, performance, and reliability of the Help Desk System.

1. Documentation and Reporting:

* Generate reports that offer insights into the overall performance of the helpdesk, specialists' workload, and problem resolution trends.

1. Scalability and Flexibility:

* Design the system to accommodate future expansion and changes in the company's IT infrastructure.

**PROJECT SCOPE**

The scope of the project involves the development of a comprehensive Help Desk System for Manzaneque Limited, focusing on the following key aspects:

1. **Help Desk Functionality:**

* Implement a system to log, track, and manage helpdesk queries related to hardware and software issues.

1. **User Authentication and Access Control:**

* Include secure user authentication mechanisms to ensure that only authorized personnel can access and manipulate data within the system.

1. **Data Logging and Retrieval:**

* Capture and store essential information for each helpdesk call, including caller details, operator information, time of the call, computer details, and relevant software information.

1. **Problem Categorization:**

* Develop a system for helpdesk operators to categorize and allocate problem types, with the ability to refine categories based on additional information.

1. **Knowledge Base and Historical Data:**

* Create a knowledge base that allows operators to look up previous problems of the same type and retrieve historical data related to equipment or specific callers.

1. **Specialist Allocation:**

* Design a mechanism for identifying and assigning specialists based on the type of problem reported, considering specialists' workload to ensure optimal allocation.

1. **Resolution Tracking:**

* Implement functionality to record the date and time of problem resolution, along with detailed information on how the problem was resolved and the time taken.

1. **User Interface:**

* Develop an intuitive and user-friendly interface for both helpdesk operators and specialists.

1. **Performance Optimization:**

* Optimize the web application system for efficient performance, considering the potential volume of helpdesk queries.

1. **Reporting and Insights:**

* Generate reports providing insights into the overall performance of the helpdesk, workload distribution among specialists, and trends in problem resolution.

**11. Training Recommendations:**

* Include features that can suggest areas where employee training is needed based on recurring problems.

**12. Testing:**

* Develop and execute a comprehensive testing plan to ensure the reliability, functionality, and security of the Help Desk System.

1. **Project Constraints:**
2. **Budgetary Constraints:**

* Develop the system within the allocated budget for IT projects.

1. **Time Constraints:**

* Adhere to the project timeline to ensure timely delivery of the Help Desk System.

1. **Resource Limitations:**

* Work within the existing IT infrastructure and resources available to Manzaneque Limited.

1. **Security and Compliance:**

* Ensure that the system complies with relevant security standards and data protection regulations.

1. **Scalability Challenges:**

* Address potential scalability challenges to accommodate future expansion and changes in the company's IT requirements.

1. **Integration with Existing Systems:**

* Integrate the Help Desk System with existing IT systems and databases to ensure seamless operations.

1. **User Training and Adoption:**

* Consider the need for user training and provide sufficient documentation for system adoption by helpdesk operators and specialists.

1. **Technical Constraints:**

* Work within the technical capabilities and limitations of the chosen development environment and technology stack.

1. **Feedback Mechanism:**

* Establish a feedback mechanism for continuous improvement based on user experiences and system performance.

1. **Legal and Ethical Considerations:**

* Adhere to legal and ethical considerations related to data privacy, confidentiality, and workplace regulations.

**RISK ANALYSIS AND EVALUATION**

| **Risk** | **Description** | **Impact** | **Mitigation** |
| --- | --- | --- | --- |
| **User Adoption** | Users may resist or struggle to adopt the new system, leading to inefficiencies. | Operational disruptions, reduced productivity. | - Conduct user training sessions. - Provide clear documentation. - Involve users in the design process. |
| **Data Security** | Unauthorized access to sensitive information in the helpdesk system. | Data breaches, loss of trust. | - Implement robust authentication and access controls. - Use encryption for sensitive data. - Regularly audit and monitor system access. |
| **Incomplete Requirements** | Users may not provide all necessary requirements initially, leading to feature gaps. | Project delays, rework. | - Engage in continuous communication with users. - Conduct thorough requirement gathering sessions. - Be prepared for iterative updates. |
| **Technological Changes** | Rapid changes in technology may affect the compatibility and longevity of chosen tools. | System obsolescence, increased development time. | - Regularly assess and update technology choices. - Build flexibility into the system architecture to accommodate future changes. |
| **Scope Creep** | Continuous changes to project scope beyond initial requirements. | Project delays, increased development costs. | - Clearly define and document the project scope. - Implement a change management process to evaluate and approve scope changes. |
| **Resource Constraints** | Insufficient resources (human, financial, technological) allocated for the project. | Delays, incomplete features. | - Conduct a thorough resource assessment before the project starts. - Continuously monitor and adjust resource allocation as needed. |
| **Integration Challenges** | Difficulties integrating the Help Desk System with existing systems or databases. | Data inconsistencies, operational disruptions. | - Conduct a comprehensive analysis of existing systems and databases. - Plan and test integration thoroughly before deployment. |
| **Dependency on Key Personnel** | Dependence on specific individuals for critical tasks. | Bottlenecks, project delays. | - Cross-train team members to handle critical tasks. - Document critical processes and procedures to reduce dependency on specific individuals. |
| **Quality Assurance Issues** | Insufficient testing leading to undetected bugs or performance issues. | System failures, user dissatisfaction. | - Develop a comprehensive testing plan. - Implement automated testing where applicable. - Conduct thorough testing at each development stage. |
| **Vendor Reliability** | Dependence on third-party vendors for critical components. | Delays, system instability. | - Select reputable vendors with a track record of reliability. - Have contingency plans in place in case of vendor-related issues. |
| **Change Resistance** | Resistance from employees to adapt to new processes and workflows. | Decreased productivity, employee dissatisfaction. | - Implement change management strategies. - Communicate the benefits of the new system to employees. - Provide training and support during the transition. |

**CHOSEN SDLC MODEL FOR THE HELP DESK SYSTEM**

Given the nature of the Help Desk System and the provided project details, the **Incremental Model** or the **Iterative Model** would be suitable.

**Incremental Model:**

**Reasons:**

* **Phased Implementation:** Since the project involves developing a Help Desk System with various components, an incremental approach allows for phased implementation. This means each phase (or increment) could represent a set of functionalities, making it easier to manage and test.
* **Early Deliverables:** The incremental model provides the advantage of delivering functional pieces of the system early in the development process. This can be beneficial for immediate testing and feedback.
* **Flexibility for Changes:** As the system evolves through increments, there is flexibility to accommodate changes or enhancements in later stages based on user feedback or changing requirements.
* **Risk Management:** Risks can be better managed as each increment is a mini-project with its own planning, design, implementation, and testing phases. This allows for early identification and mitigation of risks.
* **Continuous User Involvement:** Incremental development encourages continuous user involvement and feedback, aligning well with the iterative nature of help desk systems that often require user input for refinement.

**Iterative Model:**

**Reasons:**

* **Progressive Refinement:** The iterative model is well-suited for projects where requirements are expected to evolve. It allows for progressive refinement of the system through multiple iterations.
* **Feedback Loop:** It provides a continuous feedback loop, enabling adjustments to be made at each iteration based on the feedback received from users and stakeholders.
* **Risk Management:** Similar to the incremental model, the iterative model supports effective risk management by identifying and addressing issues early in the development process.
* **Adaptability:** Help desk systems may have evolving requirements or changing user needs. The iterative model's adaptability to changes makes it suitable for such scenarios.
* **Prototyping:** The iterative model often involves the creation of prototypes, which can be beneficial for refining user interfaces and ensuring that the system aligns closely with user expectations.

**USER & SYSTEM REQUIRMENTS**

User Requirements:

* The helpdesk system should be user-friendly, allowing users to easily submit and track their queries.
* Users should be able to submit queries through multiple channels, such as phone, email, and an online portal.
* The system should provide users with real-time updates on the status of their queries, including estimated resolution times.
* Users should be able to view a history of their past queries and their resolutions.
* The system should allow users to rate the quality of the support provided by the helpdesk specialists.
* The system must allow users to create and submit helpdesk tickets online.
* Users should be able to track the status of their helpdesk tickets and receive updates on progress.
* The system must provide an intuitive interface for users to describe their problems and request assistance.
* Users must be able to access the system from any device with an internet connection.
* The system should allow users to search for self-help resources to solve common issues.
* Users should have the option to escalate their tickets to higher levels of support if their problems are not resolved in a timely manner.
* The system must provide a secure and confidential way for users to submit sensitive information related to their issues.
* The system should allow users to rate their experience with the helpdesk support team and provide feedback.
* Users must be able to easily access information about their equipment, such as its warranty status and service history.
* The system should provide users with regular updates on system maintenance and downtime.

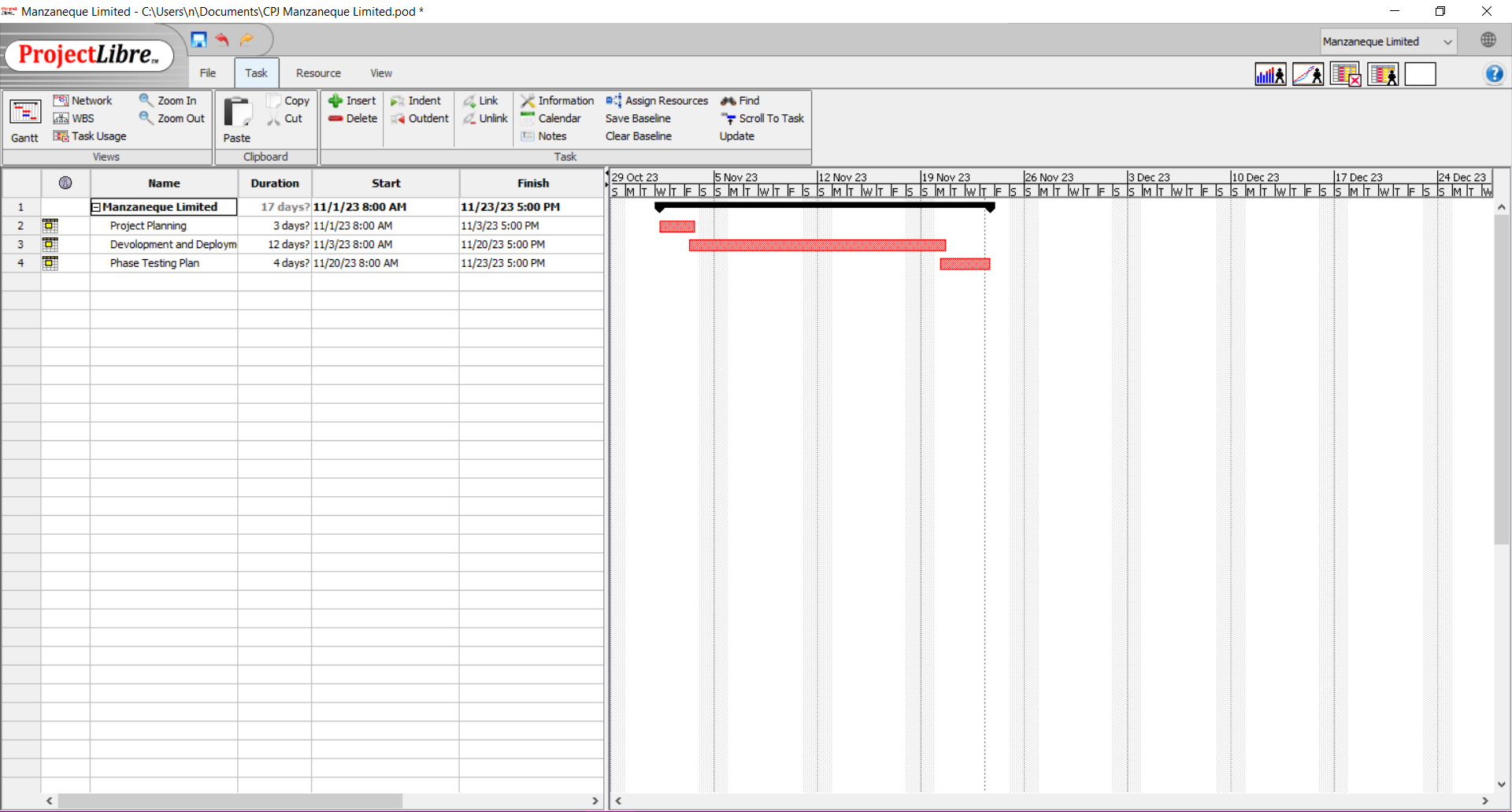
System Requirements:

* The system should be designed to handle a high volume of queries and be scalable to accommodate future growth.
* The system should have robust query management functionality, including the ability to assign, prioritize, and track queries.
* The system should be integrated with a knowledge base that allows helpdesk specialists to easily access information and resources to help them resolve queries efficiently.
* The system should provide detailed reporting and analytics on query resolution times, user satisfaction, and specialist performance.
* The system should have robust security measures in place to protect user data and prevent unauthorized access.
* The system should be able to handle a large volume of helpdesk queries efficiently and effectively.
* The system should be scalable to accommodate future growth and expansion of the company.
* The system should have a user-friendly interface that is easy to navigate and understand.
* The system should be accessible via a web-based interface, allowing users to access it from anywhere with an internet connection.
* The system should be able to integrate with other existing systems within the company such as inventory management and CRM systems.
* The system should be able to generate automatic notifications to users regarding the status of their queries and provide updates on progress.
* The system should provide the ability to assign queries to specific technicians or teams for resolution.
* The system should have a search function to allow users to quickly find previously logged queries and their resolution status.
* The system should provide real-time reporting and analytics on the performance of the helpdesk, including query resolution times, user satisfaction, and technician performance.
* The system should have robust backup and disaster recovery capabilities to ensure that data is not lost in the event of system failures or disasters.

Hardware Requirements:

* Servers: The IT Helpdesk system requires a dedicated server for hosting the database and application servers for processing user requests.
* Storage Devices: Adequate storage devices should be provided to store system data, such as databases, log files, and backups.
* Workstations: The IT Helpdesk operators and specialists require workstations equipped with the necessary hardware, such as computers, keyboards, and mice.
* Networking Equipment: The system requires routers, switches, firewalls, and other networking equipment to support communication between system components.
* Printers: The IT Helpdesk system requires printers to produce reports and other documentation.
* Backup Equipment: Backup equipment such as Uninterruptible Power Supply (UPS) should be installed to avoid data loss in case of power outages.
* Security Equipment: The system requires adequate security equipment such as surveillance cameras, biometric scanners, and access control systems to protect sensitive data and hardware.
* Audio/Visual Equipment: The system requires audio/visual equipment such as speakers, projectors, and monitors to support training sessions and presentations.
* Mobile Devices: The IT Helpdesk operators and specialists require mobile devices such as smartphones and tablets for remote access to the system.
* Virtualization Environment: The IT Helpdesk system may require a virtualization environment to support multiple instances of the application servers and databases to enhance scalability and high availability.

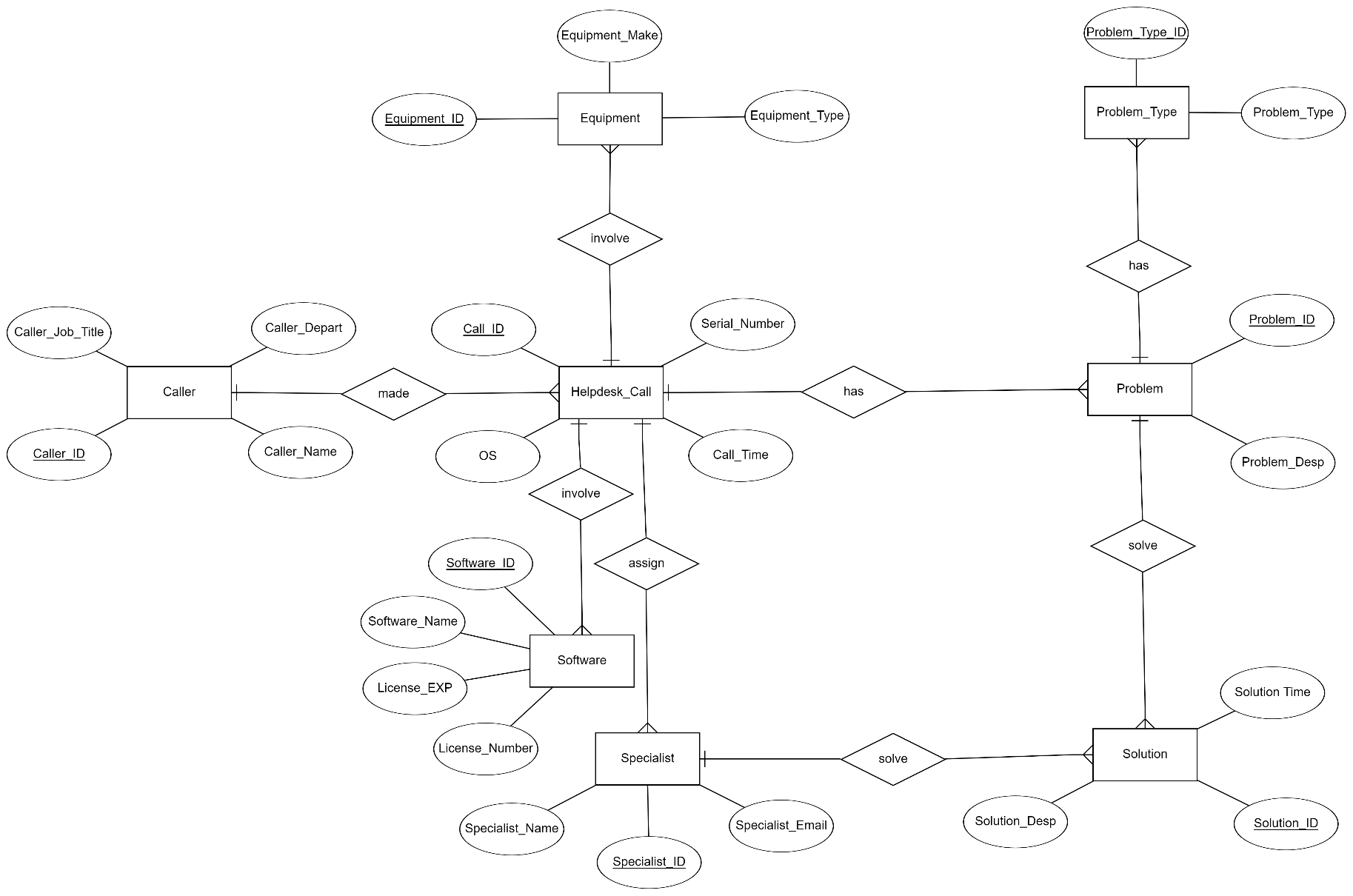
**GANETT CHART**



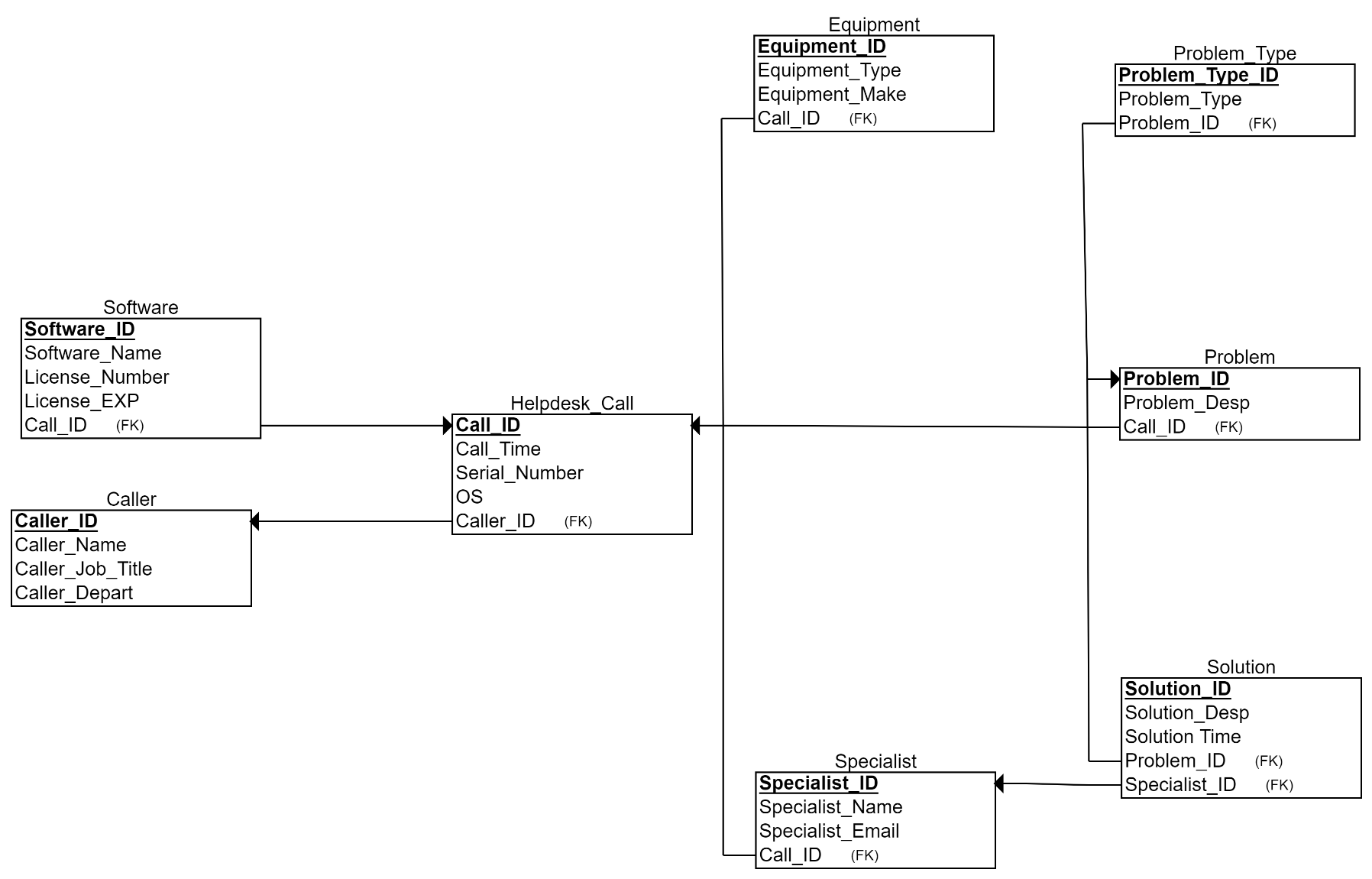
**MILE STONE SCHEDULE**

| **Milestone** | **Duration** | **Tasks** |
| --- | --- | --- |
| **1. Project Kickoff** | **Week 1-2** | - Define project scope, objectives, and stakeholders. |
|  |  | - Establish project team roles and responsibilities. |
|  |  | - Initial project planning. |
| **2. User and System Requirements** | **Week 3-4** | - Gather and finalize detailed user requirements. |
| Specification |  | - Document system requirements. |
|  |  | - Review and get approval from stakeholders. |
| **3. Design Phase** | **Week 5-8** | - Develop UML design representations (Use Case Diagram, Activity Diagram, DFD). |
|  |  | - Create algorithmic design (Flow Chart or Algorithm). |
|  |  | - Develop behavioral design (Use Case Diagram, Activity Diagram, or DFD). |
|  |  | - Create data design (ERD or EERD Diagram, Data Dictionary). |
| **4. Development and Deployment Environment** | **Week 9-10** | - Identify and set up a suitable development environment. |
| Setup |  | - Begin the development of the Help Desk System. |
| **5. Increment 1 Implementation** | **Week 11-14** | - Implement the first increment of the Help Desk System. |
|  |  | - Conduct initial testing and debugging. |
|  |  | - Collect feedback from users and stakeholders. |
|  |  |  |
| ... (Repeat for additional increments) |  |  |
| **Final Testing and Refinement** | **Week 15-16** | - Perform comprehensive testing of the entire system. |
|  |  | - Refine and address any identified issues. |
| **6. Evaluation and Report** | **Week 17-18** | - Review how the application meets the needs of the requirements and problem definition. |
|  |  | - Critical analysis of strengths and weaknesses. Discuss potential improvements and enhancements. |
| **7. Final Deployment** | **Week 19-20** | - Deploy the final version of the Help Desk System. |
|  |  | - Provide necessary training to end-users and support staff. |
| **8. Project Closure** | **Week 21-22** | - Finalize project documentation. |
|  |  | - Conduct a project review and lessons learned session. |
|  |  | - Obtain project sign-off from stakeholders. |

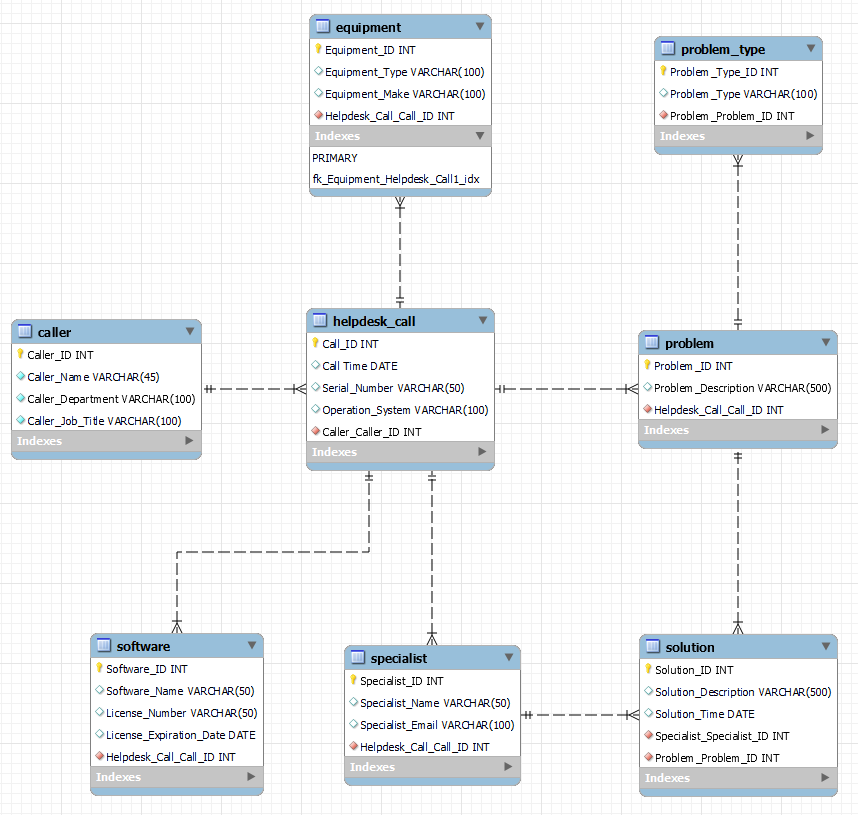
**UML DESIGN**



**Logical Design**



**ER DIAGRAM**

****

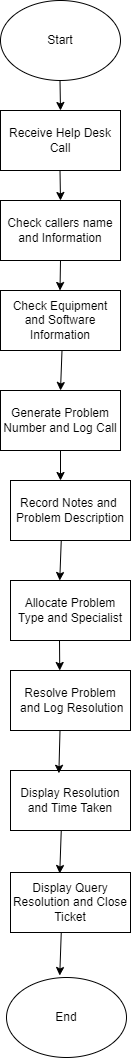
**UML CASE DIAGRAM**

-[Diagram, timeline

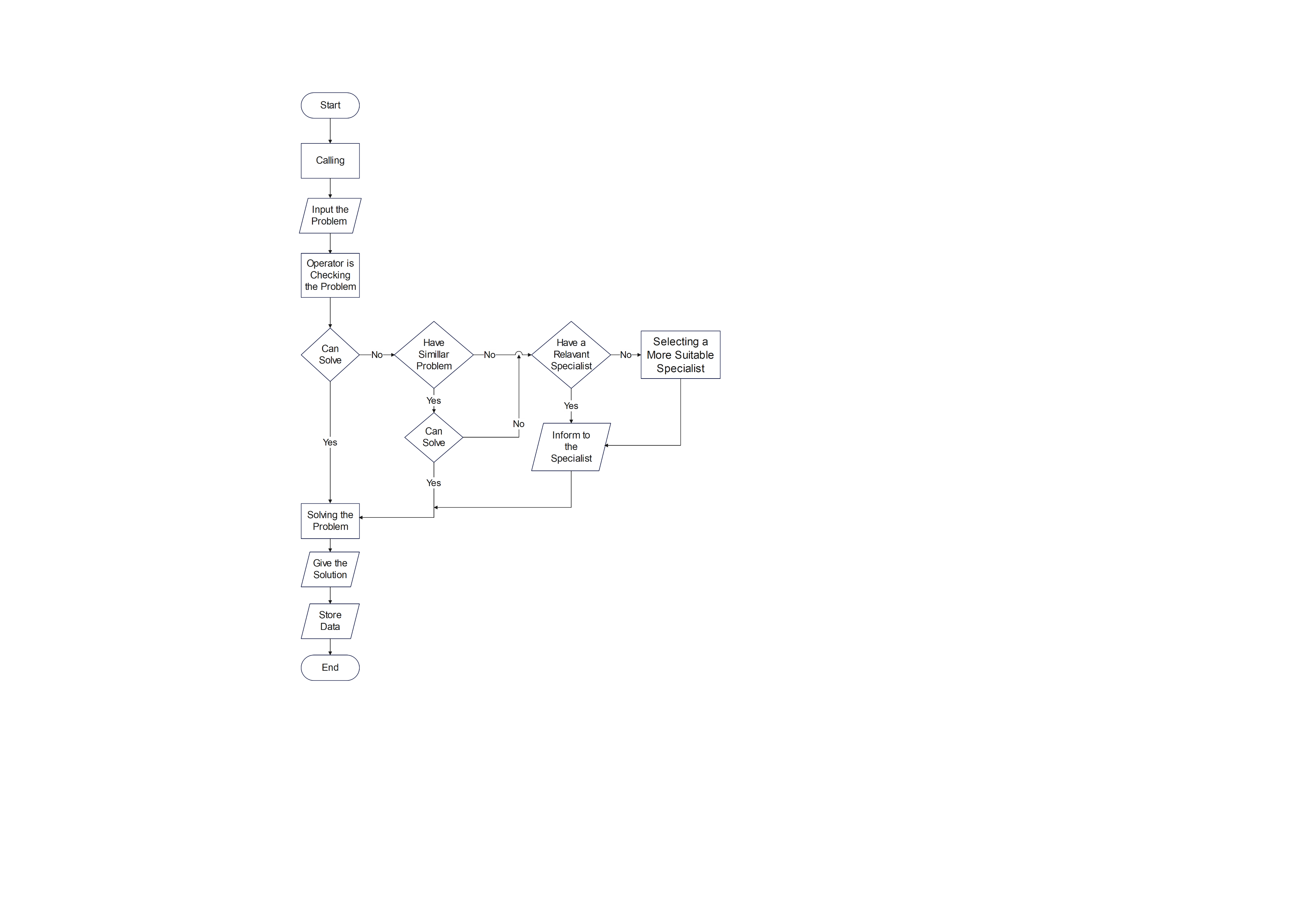
Description automatically generated

Diagram

Description automatically generated

Flow chart 

**FLOW CHART**



Business System Options (BSO) for Help Desk System:

* **Custom Web Application:**
* Developing a custom web-based application tailored to the specific needs of Manzaneque Limited's IT Helpdesk.

Advantages

* 1. Provides a fully customizable solution
  2. Aligning perfectly with the company's unique requirements.

Disadvantages

1. Longer development time and potentially higher initial costs.

* **Commercial Helpdesk Software:**
* Purchasing and implementing a pre-built helpdesk software solution available in the market.

Advantages:

1. Quicker implementation
2. Potential cost savings
3. May have features already aligned with industry best practices.

Disadvantages

* May require customization to meet all specific needs; ongoing licensing costs.
* **Integrated IT Service Management (ITSM) System:**
* Implementing a comprehensive ITSM system that not only handles helpdesk queries but integrates with other IT functions.

Advantages:

* Streamlines IT processes
* Allows for better tracking of overall IT performance
* Can offer scalability.

Disadvantages

* May involve a more significant change in existing processes
* Training for staff may be required.

**Business System Option 4: Open Source Helpdesk Solution**

* Utilizing an open-source helpdesk solution involves implementing a software system with publicly available source code. This option allows for customization and flexibility without the high costs associated with proprietary solutions.

**Advantages:**

* **Cost-Effective:** No licensing fees; potential cost savings.
* **Community Support:** Benefit from a community of developers and users.
* **Customization:** Can be tailored to specific business needs.
* **Transparency:** Full access to source code for transparency and control.
* **Scalability:** Adaptable to business growth without additional licensing costs.

**Disadvantages:**

* **Limited Vendor Support:** Relies on community support or in-house expertise.
* **Potential Security Concerns:** Security measures may require closer attention.
* **Integration Challenges:** Compatibility issues with other systems.
* **Documentation Variances:** Quality and completeness of documentation can vary.
* **Dependency on Community:** Updates and patches may depend on community contributions.

Business System Option 5: Hybrid Solution

* A hybrid solution combines elements of custom development with pre-built modules. This approach allows for flexibility and customization while leveraging existing solutions for certain functionalities.

**Advantages:**

* **Tailored Functionality:** Customized modules for specific business needs.
* **Quick Implementation:** Faster deployment for pre-built modules.
* **Cost Flexibility:** Control costs by selecting pre-built or custom components.
* **Scalability:** Easily expand and modify with changing business requirements.
* **Vendor Support for Modules:** Benefit from support for pre-built components.

**Disadvantages:**

* **Integration Complexity:** Requires seamless integration between custom and pre-built components.
* **Development Skills Needed:** Proficiency in both custom and pre-built solutions.
* **Dependency on Vendors:** Relies on vendors for support and updates.
* **Maintenance Challenges:** Ongoing maintenance may involve both custom and pre-built aspects.
* **Potential for Over-customization:** Risks of complicating the system with excessive customization.

Technical System Options (BSO) for Help Desk System:

Technical System Option 1: Web Application Framework (e.g., Django, Ruby on Rails)

* Using a web application framework provides a structured and efficient way to develop a custom web-based IT Helpdesk system. Frameworks like Django or Ruby on Rails offer pre-built components, speeding up development and ensuring best practices.

**Advantages:**

* **Rapid Development:** Frameworks offer ready-made components for faster development.
* **Scalability:** Frameworks often support scalability as the application grows.
* **Community Support:** Benefit from a community of developers for problem-solving.
* **Security Features:** Frameworks often come with built-in security features.
* **Code Organization:** Promotes code organization and maintainability.

**Disadvantages:**

* **Learning Curve:** Developers need to learn the specific framework.
* **Limited Flexibility:** Some frameworks may restrict certain design choices.
* **Potential Bloat:** Unused features in the framework may lead to unnecessary complexity.
* **Dependence on Framework:** Updates and changes in the framework may impact the system.
* **Compatibility Challenges:** Integrating with existing systems may require extra effort.

Technical System Option 2: Database Management System (DBMS) - MySQL or PostgreSQL

* Choosing an appropriate Database Management System (DBMS) is crucial for storing and managing data efficiently. MySQL or PostgreSQL, for example, can be employed to ensure data integrity and security.

**Advantages:**

* **Data Integrity:** DBMS ensures consistency and accuracy of data.
* **Scalability:** Capable of handling growing amounts of data.
* **Transaction Support:** ACID properties ensure reliable transactions.
* **Community Support:** Large user communities for both MySQL and PostgreSQL.
* **Compatibility:** Compatible with various programming languages and frameworks.

**Disadvantages:**

* **Learning Curve:** May require expertise in database design and management.
* **Costs:** Licensing fees or resource costs for enterprise-level systems.
* **Complexity:** Advanced features may introduce complexity.
* **Performance Tuning:** Requires tuning for optimal performance.
* **Vendor Dependence:** For commercial database solutions, reliance on the vendor for updates.

Technical System Option 3: User Authentication and Authorization System

* Implementing a secure user authentication and authorization system is essential for controlling access to the helpdesk application, ensuring data security, and compliance with privacy standards.

**Advantages:**

* **Security:** Protects sensitive data from unauthorized access.
* **Controlled Access:** Allows granular control over user permissions.
* **Audit Trail:** Tracks user actions for security and accountability.
* **Integration:** Can integrate with existing authentication systems.
* **Compliance:** Helps in meeting regulatory requirements for data protection.

**Disadvantages:**

* **Implementation Complexity:** Setting up a robust authentication system can be complex.
* **Maintenance Overhead:** Regular updates and maintenance are necessary.
* **User Training:** Users may need training on authentication processes.
* **Potential for Lockouts:** Risk of user lockouts if not implemented carefully.
* **Scalability Challenges:** May face challenges as the user base grows.

Technical System Option 4: Integration with Existing Systems (APIs or Middleware)

* Ensuring seamless integration with existing systems such as HR databases and equipment registers is crucial. This can be achieved through the use of APIs or middleware solutions that facilitate communication between different systems.

**Advantages:**

* **Data Consistency:** Avoids data duplication and ensures consistency.
* **Efficiency:** Streamlines processes by leveraging existing data.
* **Improved Decision-Making:** Access to a unified view of information.
* **Flexibility:** Allows for the adoption of new technologies without a full system overhaul.
* **Cost-Effective:** Maximizes the use of existing infrastructure and data.

**Disadvantages:**

* **Integration Complexity:** May require significant development effort.
* **Data Security:** Potential security risks during data transfer between systems.
* **Dependency on External Systems:** Relies on the stability of integrated systems.
* **Customization Challenges:** Integration may not fully accommodate unique business processes.
* **Maintenance Overhead:** Regular updates and monitoring for potential issues.

Technical System Option 5: Performance Optimization Tools

* Implementing performance optimization tools is crucial to ensure the IT Helpdesk system operates efficiently. These tools help identify and address performance bottlenecks proactively.

**Advantages:**

* **Proactive Issue Identification:** Identifies performance issues before they impact users.
* **Scalability Planning:** Assists in planning for system growth.
* **Resource Allocation:** Optimizes resource usage for improved efficiency.
* **User Experience:** Enhances the overall user experience by reducing latency.
* **Stability:** Improves system stability by addressing performance-related issues.

**Disadvantages:**

* **Learning Curve:** Staff may need training to use performance tools effectively.
* **Implementation Time:** Setting up and configuring performance tools may take time.
* **Resource Consumption:** The tools themselves may consume system resources.
* **False Positives:** Potential for misinterpreting data and addressing non-issues.
* **Cost:** Investment in performance tools and ongoing maintenance.

**BACK END FRAMEWORK USED FOR HELP DESK SYSTEM IS SPRING BOOT**

**REASONS FOR CHOOSING SPRING BOOT AS THE BACK END FRAMEWORK**

Features of Spring Boot for Helpdesk Scenario:

* **Dependency Management:** Spring Boot provides robust dependency management through its built-in dependency management system. This simplifies the management of project dependencies, including libraries and frameworks required for building the helpdesk system.
* **Auto-Configuration:** Spring Boot offers auto-configuration, which automatically configures Spring applications based on the dependencies present in the classpath. This reduces the need for manual configuration, making it easier to set up the helpdesk system.
* **Embedded Server:** Spring Boot comes with an embedded servlet container (Tomcat, Jetty, or Undertow) that allows you to deploy applications as standalone executable JAR files. This simplifies deployment and eliminates the need for external application servers.
* **Production-Ready Features:** Spring Boot includes several production-ready features, such as metrics, health checks, and externalized configuration. These features enable you to monitor and manage the health and performance of the helpdesk system in production environments.
* **Spring Security Integration:** Spring Boot seamlessly integrates with Spring Security, providing robust authentication and authorization mechanisms for securing the helpdesk system. This ensures that sensitive information is protected from unauthorized access.
* **RESTful Web Services:** Spring Boot simplifies the creation of RESTful web services using Spring MVC and Spring Webflux. This allows you to expose APIs for submitting and managing helpdesk tickets, enabling seamless integration with frontend and third-party systems.
* **Database Access:** Spring Boot offers seamless integration with various databases through Spring Data JPA and Spring Data JDBC. This simplifies database access and allows you to store and retrieve helpdesk tickets and related information efficiently.
* **Testing Support:** Spring Boot provides robust testing support with features like Spring Test and Spring Boot Test. This allows you to write unit tests, integration tests, and end-to-end tests to ensure the reliability and quality of the helpdesk system.
* **Monitoring and Management:** Spring Boot Actuator provides endpoints for monitoring and managing the helpdesk system at runtime. This includes endpoints for health checks, metrics, environment information, and more, facilitating better operational visibility and control.
* **Community and Ecosystem:** Spring Boot has a large and active community, which provides extensive documentation, tutorials, and support resources. Additionally, the Spring ecosystem offers a wide range of libraries and tools that complement Spring Boot, enabling you to extend and customize the helpdesk system as needed.

Advantages of using Spring Boot as the Back End Framework

* **Rapid Development:** Spring Boot's convention-over-configuration approach and auto-configuration features enable rapid development of the helpdesk system.
* **Reduced Boilerplate Code:** Spring Boot reduces boilerplate code, allowing developers to focus on implementing business logic rather than infrastructure concerns.
* **Increased Productivity:** With Spring Boot's features like embedded server and dependency management, developers can build, deploy, and iterate on the helpdesk system more efficiently.
* **Simplified Deployment:** The ability to package applications as standalone JAR files simplifies deployment, reducing deployment overhead and ensuring consistency across environments.
* **Scalability:** Spring Boot's architecture and support for distributed systems make it scalable, allowing the helpdesk system to handle increased workload and user demand as the company grows.
* **Robust Security:** Integration with Spring Security ensures that the helpdesk system is secure, protecting sensitive information and preventing unauthorized access.
* **Flexible Integration:** Spring Boot's support for various protocols and standards enables seamless integration with existing systems, databases, and third-party services.
* **Easy Maintenance:** Spring Boot's modular and well-structured architecture makes it easy to maintain and update the helpdesk system over time, reducing maintenance costs and efforts.
* **Comprehensive Testing:** Spring Boot's testing support facilitates comprehensive testing of the helpdesk system, ensuring reliability, stability, and compliance with business requirements.
* **Enterprise-Grade Reliability:** With its production-ready features and robust ecosystem, Spring Boot provides enterprise-grade reliability, ensuring that the helpdesk system meets the company's operational needs and standards.

**TOOL USED TO IMPLEMENT SPRING BOOT FOR HELP DESK SYSTEM SPRING TOOL SUITE 3**

**REASONS FOR CHOOSING SRING TOOL SUITE AS THE TOOL**

Spring Tool Suite (STS) is an integrated development environment (IDE) based on Eclipse that provides comprehensive tooling support for building Spring-based applications

Features of Spring Tool Suite for Helpdesk Scenario:

* **Spring Project Wizards:** STS offers project wizards and templates for quickly creating Spring-based projects, including Spring Boot applications. This streamlines the setup process for the helpdesk system.
* **Code Assist and Auto-Completion:** STS provides intelligent code assist and auto-completion features, which help developers write code faster and with fewer errors when implementing the helpdesk system

.

* **Spring Configuration Editors:** STS includes editors for Spring configuration files (XML and annotations), making it easier to configure Spring beans, MVC controllers, and other components required for the helpdesk system.
* **Dependency Management:** STS integrates with Maven and Gradle, providing advanced dependency management capabilities. This allows developers to manage project dependencies efficiently, including libraries and frameworks required for the helpdesk system.
* **Server Integration:** STS seamlessly integrates with various application servers, including Apache Tomcat and Pivotal tc Server. This enables developers to deploy and debug the helpdesk system directly from within the IDE.
* **Spring Boot Support:** STS offers extensive support for Spring Boot, including project creation, auto-configuration, and run/debug capabilities. This facilitates rapid development and testing of the helpdesk system using Spring Boot.
* **Code Refactoring Tools:** STS provides powerful code refactoring tools, such as rename, extract method, and inline method refactorings. These tools help improve code quality and maintainability when developing the helpdesk system.
* **Integrated Testing Frameworks:** STS supports integration with popular testing frameworks like JUnit and TestNG, allowing developers to write and execute unit tests, integration tests, and end-to-end tests for the helpdesk system directly within the IDE.
* **Spring Data and Database Tools:** STS includes tools for working with Spring Data repositories and databases. This enables developers to easily create and manage data access components for storing and retrieving helpdesk tickets and related information.
* **Extensive Plugin Ecosystem:** STS supports a wide range of plugins and extensions, allowing developers to customize and extend the IDE's functionality to meet the specific requirements of the helpdesk system.

Advantages of Using Spring Tool Suite for Helpdesk Scenario:

* **Seamless Integration:** STS seamlessly integrates with Spring frameworks and tools, providing a cohesive development experience for building Spring-based applications like the helpdesk system.
* **Increased Productivity:** With features like code assist, project wizards, and testing support, STS helps developers work more efficiently, reducing development time and effort.
* **Streamlined Development Workflow:** STS offers a streamlined development workflow, from project creation to deployment and testing, making it easy to develop and iterate on the helpdesk system.
* **Comprehensive Tooling:** STS provides comprehensive tooling support for all aspects of Spring development, including configuration, dependency management, and testing, ensuring a smooth development experience for the helpdesk system.
* **Built-in Server Support:** STS comes with built-in support for deploying and debugging applications on various servers, simplifying the development and testing process for the helpdesk system.
* **Rich Ecosystem:** STS benefits from the rich ecosystem of Eclipse plugins and extensions, allowing developers to enhance the IDE's functionality as needed for the helpdesk system.
* **Robust Testing Capabilities:** With support for popular testing frameworks and integrated testing tools, STS enables developers to write and execute tests effectively, ensuring the reliability and quality of the helpdesk system.
* **Easy Collaboration:** STS supports collaboration features like version control integration and team sharing, enabling developers to work together seamlessly on the helpdesk system.
* **Continuous Integration:** STS integrates with continuous integration (CI) servers like Jenkins and Hudson, facilitating automated builds and deployments for the helpdesk system.
* **Community Support:** STS benefits from a large and active community of developers, providing access to resources, tutorials, and support forums to help developers overcome challenges and optimize their workflow when building the helpdesk system.

**FRONT END FRAMEWORK USED FOR THE HELP DESK SCENARIO REACT JS**

**REASONS TO WHY REACT JS IS USED AS A FRONT END FRAMEWORK FOR HELP DESK SYSTEM**

Features of React JS:

* ***Component-Based Architecture***: React utilizes a component-based architecture, allowing developers to create reusable UI components. This feature is beneficial for building a help desk portal with consistent UI elements like forms, buttons, and cards.
* ***Virtual DOM***: React uses a virtual DOM to efficiently update and render UI components. This feature enhances performance by minimizing DOM manipulation, which is advantageous for handling real-time updates and interactions in the help desk portal.
* ***JSX***: React's JSX syntax allows developers to write HTML-like code within JavaScript, making it easier to create and maintain UI components. This feature facilitates the creation of dynamic and interactive user interfaces for the help desk portal.
* ***Unidirectional Data Flow***: React follows a unidirectional data flow, where data flows in one direction from parent to child components. This feature simplifies state management and reduces the risk of data inconsistency, ensuring data accuracy in the help desk portal.
* ***React Hooks***: React Hooks provide a way to use state and other React features without writing classes. This feature simplifies state management and component lifecycle in functional components, making it easier to manage complex logic in the help desk portal.
* ***Reusable Components***: React encourages the creation of reusable components, promoting code reusability and maintainability. This feature allows developers to build a modular help desk portal with reusable UI components for different sections like user authentication, ticket management, and reporting.
* ***Support for Server-Side Rendering (SSR)****:* React supports server-side rendering, enabling faster initial page loads and improved SEO performance. This feature is beneficial for rendering the help desk portal on the server and delivering pre-rendered HTML to clients, enhancing the overall user experience.
* ***Rich Ecosystem and Community Support****:* React has a rich ecosystem of libraries, tools, and community-driven resources, making it easier for developers to build and extend the functionality of the help desk portal. This feature ensures ongoing support, updates, and enhancements for the project.
* ***Integration with State Management Libraries****:* React can be seamlessly integrated with state management libraries like Redux and MobX, providing efficient state management solutions for complex applications. This feature is advantageous for managing global state and data flow in the help desk portal.
* ***Mobile App Development****:* React Native, a framework based on React, allows developers to build cross-platform mobile applications using JavaScript and React. This feature enables the development of companion mobile apps for the help desk portal, extending its accessibility and usability to mobile users.

Advantages of using React JS:

* ***Enhanced Performance***: React's virtual DOM and efficient rendering mechanism contribute to improved performance and responsiveness, ensuring a seamless user experience in the help desk portal.
* ***Scalability***: React's component-based architecture and modular design facilitate scalability, allowing the help desk portal to accommodate future growth and feature enhancements with ease.
* ***Developer Productivity***: React's declarative syntax, reusable components, and extensive tooling enhance developer productivity, enabling faster development cycles and easier maintenance of the help desk portal.
* ***Code Reusability***: React's emphasis on component reusability promotes code reusability, reducing development time and effort required to build and maintain the help desk portal.
* ***Community and Support***: React has a vibrant community and extensive documentation, providing access to resources, tutorials, and best practices to support the development of the help desk portal.
* ***Flexibility***: React's flexibility allows developers to integrate it with other libraries, frameworks, and tools, enabling customization and extending functionality as per the requirements of the help desk portal.
* ***SEO-Friendly***: React's support for server-side rendering improves SEO performance by generating pre-rendered HTML content, making the help desk portal more discoverable and accessible to search engines.
* ***Cross-Platform Compatibility***: React's compatibility with different platforms and environments, including web browsers and mobile devices, ensures broad accessibility and reach for the help desk portal.
* ***Code Stability and Predictability***: React's unidirectional data flow and component-based architecture contribute to code stability and predictability, reducing the likelihood of bugs and errors in the help desk portal.
* ***Modern Development Practices***: React promotes the adoption of modern development practices like componentization, functional programming, and reactive programming, enabling the creation of robust and maintainable solutions for the help desk scenario.

**USED TOOL FOR IMPLEMENTING THE FRONT END OF HELP DESK PORTAL WHICH IS VISUAL STUDIO CODE**

**REASONS FOR CHOOSING VISUAL STUDIO CODE TO IMPLEMENT THE FRONT END OF THE VISUAL STUDIO CODE**

Features of Visual Studio Code :

* **Intelligent Code Editor**: VS Code provides an intelligent code editor with features like syntax highlighting, autocompletion, and code formatting, enhancing productivity and code quality when developing React components for the help desk portal.
* **Integrated Terminal**: VS Code includes an integrated terminal that allows developers to run commands, execute scripts, and interact with the command-line interface directly within the editor, streamlining development tasks such as running React scripts, managing dependencies, and debugging.
* **Extensions Marketplace**: VS Code offers a rich ecosystem of extensions through its marketplace, providing access to a wide range of tools, linters, and extensions tailored for React development. This feature enables developers to customize their development environment and enhance productivity with React-specific extensions.
* **Git Integration**: VS Code integrates seamlessly with Git version control, allowing developers to manage source code changes, collaborate with team members, and track revisions effectively. This feature facilitates version control and collaboration when building and maintaining the help desk portal.
* **Debugging Support**: VS Code provides robust debugging support for React applications, including breakpoints, watch expressions, and interactive debugging tools. This feature helps developers identify and troubleshoot issues in the help desk portal's frontend code efficiently.
* **Live Server**: VS Code offers extensions like Live Server that enable developers to launch a local development server and automatically refresh the browser whenever changes are made to the React code. This feature streamlines the development process and provides instant feedback when building and testing the help desk portal.
* **IntelliSense**: VS Code's IntelliSense feature provides intelligent code completion, suggesting relevant React components, props, and methods based on the context, reducing errors and speeding up development when writing React code for the help desk portal.
* **Task Automation**: VS Code allows developers to define and execute custom tasks, such as building, testing, and deploying React applications, using task runners like npm scripts or Gulp. This feature automates repetitive tasks and enhances workflow efficiency in the help desk scenario.
* **Integrated Version Control**: VS Code includes built-in support for version control systems like Git, enabling developers to commit, push, pull, and manage code changes directly from the editor. This feature promotes collaboration and ensures code integrity in the development of the help desk portal.
* **Cross-Platform Compatibility**: VS Code is available for multiple operating systems, including Windows, macOS, and Linux, ensuring consistent development experience across different platforms for building the help desk portal's frontend with React JS.
* Advantages:
* **Familiarity and Adoption**: VS Code is widely adopted by developers and has a large user base, making it easier to onboard new team members and leverage community resources, tutorials, and support when developing the help desk portal with React JS.
* **Performance and Responsiveness**: VS Code is known for its lightweight and fast performance, offering responsive editing and debugging capabilities, even when working with large React codebases for the help desk scenario.
* **Customization and Extensibility**: VS Code's extensibility allows developers to customize the editor with themes, plugins, and settings tailored to their preferences and workflow requirements, enhancing comfort and productivity when working on the help desk portal's frontend.
* **Integration with Development Tools**: VS Code seamlessly integrates with various development tools and services, such as npm, ESLint, Prettier, and GitHub, providing a cohesive development environment for building, testing, and deploying React applications for the help desk scenario.
* **Community Support and Documentation**: VS Code benefits from an active community of developers who contribute plugins, extensions, and resources to enhance its functionality and address common use cases, including React development for the help desk portal.
* **Continuous Improvement**: VS Code is developed and maintained by Microsoft, with regular updates and improvements released to address bugs, add new features, and enhance performance, ensuring a modern and reliable development experience for building the help desk portal.
* **Accessibility Features**: VS Code includes accessibility features and options for customization, making it accessible to developers with diverse needs and preferences, ensuring inclusivity in the development process of the help desk portal's frontend.
* **Integration with Azure Services**: VS Code integrates seamlessly with Microsoft Azure services, enabling developers to deploy, host, and monitor React applications on Azure, leveraging cloud infrastructure for scaling and managing the help desk portal.
* **Ease of Collaboration**: VS Code supports collaborative editing and real-time collaboration through extensions like Live Share, allowing multiple developers to work together on the same React codebase simultaneously, facilitating teamwork and knowledge sharing in building the help desk portal.
* **Cost-Effective Solution**: VS Code is free and open-source software, providing a cost-effective solution for frontend development with React JS for the help desk scenario, without requiring expensive licensing fees or subscriptions.

**DATABASE FRAMEWORK TOOL USED FOR THE HELP DESK SCENARIO IS MYSQL WORKBENCH 8.0**

**REASONS AS TO WHY THE DATABASE FRAMEWORK TOOL USED FOR THE HELP DESK SCENARIO IS MYSQL WORKBENCH 8.0**

MySQL Workbench is a powerful visual tool for database design, development, and administration.

Features of MySQL Workbench for Help Desk Scenario:

* **Database Design:** MySQL Workbench provides a graphical interface for designing and modeling databases, allowing developers to create and visualize database schemas for the help desk system.
* **Entity-Relationship Diagrams (ERDs):** MySQL Workbench supports creating Entity-Relationship Diagrams (ERDs) that represent the relationships between different entities and attributes in the database schema, providing a clear understanding of the data structure.
* **Data Modeling Tools:** MySQL Workbench offers various data modeling tools, including tables, views, indexes, and foreign keys, allowing developers to define and manage the data model for the help desk system efficiently.
* **Schema Synchronization:** MySQL Workbench enables developers to synchronize the database schema with changes made in the visual design, ensuring consistency between the database model and the actual database implementation.
* **SQL Development:** MySQL Workbench includes a built-in SQL editor with syntax highlighting, code completion, and formatting features, facilitating the development of SQL queries, stored procedures, and triggers for the help desk system.
* **Query Execution:** MySQL Workbench allows developers to execute SQL queries directly within the IDE and view the results in tabular format, enabling quick testing and validation of database operations for the help desk system.
* **Database Administration:** MySQL Workbench offers administration features such as user management, server configuration, and performance monitoring, empowering administrators to manage and optimize the database environment for the help desk system.
* **Backup and Restore:** MySQL Workbench provides tools for performing database backups and restores, including full, incremental, and partial backups, ensuring data integrity and disaster recovery for the help desk system.
* **Migration Support:** MySQL Workbench supports database migration from other database management systems (DBMS) to MySQL, allowing developers to migrate existing data and schemas to MySQL for use in the help desk system.
* **Collaboration and Version Control:** MySQL Workbench integrates with version control systems such as Git, enabling developers to collaborate on database projects and track changes to the database schema and SQL scripts for the help desk system.

Advantages of Using MySQL Workbench for Help Desk Scenario:

* **Visual Database Design:** MySQL Workbench offers a visual interface for designing and modeling databases, making it easier to conceptualize and create the database schema for the help desk system.
* **Efficient Development Workflow:** With features like schema synchronization and SQL development tools, MySQL Workbench streamlines the database development process, allowing developers to iterate quickly and efficiently.
* **Data Integrity and Consistency:** MySQL Workbench helps maintain data integrity and consistency by enforcing relationships, constraints, and validations defined in the database schema for the help desk system.
* **Improved Productivity:** MySQL Workbench enhances developer productivity with its intuitive interface, code completion, and query execution features, enabling developers to work faster and more effectively.
* **Centralized Administration:** MySQL Workbench centralizes database administration tasks, providing administrators with a single interface for managing users, configurations, backups, and monitoring for the help desk system.
* **Data Security:** MySQL Workbench includes features for managing user permissions and access control, ensuring that sensitive data in the help desk system is protected from unauthorized access.
* **Scalability and Performance:** With performance monitoring and optimization tools, MySQL Workbench helps optimize database performance and scalability, ensuring that the help desk system can handle growing data and user loads.
* **Data Backup and Recovery:** MySQL Workbench simplifies the process of backing up and restoring databases, providing peace of mind and ensuring that critical data for the help desk system is protected against loss or corruption.
* **Flexibility and Compatibility:** MySQL Workbench supports a wide range of MySQL versions and configurations, making it suitable for various deployment environments and scenarios, including the help desk system.
* **Community Support and Resources:** MySQL Workbench benefits from a large and active community of developers and database administrators, providing access to resources, tutorials, and forums for troubleshooting and assistance when building and managing the help desk system's database.

**TESTING FOR THE HELP DESK PORTAL**

1. Functional Testing (UAT Testing)

|  |  |  |  |
| --- | --- | --- | --- |
| Requirements | Test Scenario Id | Test Scenario | Number of Test Cases |
| Functional Testing | TS001– UAT Testing | Registration | 2 |
| TS002– UAT Testing | Login | 2 |
| TS003– UAT Testing | Search Car | 2 |
| TS004– UAT Testing | Post car | 2 |
| TS005– UAT Testing | View Car | 2 |

* + 1. Functional Testing (UAT)

|  |  |
| --- | --- |
| Test Scenario | Registration |
| TS001 |
| Test Cases | Registration in Help Desk Portal |
| TC001 | Validating data that has entered into registration page |
| TC002 | Once data has entered it should save the data into database |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Scenario ID | Test Case ID | Test Priority | Pre-condition | Expected Result | Actual Result | Pass/  Fail |
| TS001 | TC001 | High | Help Desk Portal website has completed and uploaded on server | After submitting the registration data, will show error email format ‘Please include a ‘@’ in the email address. | Result is come out  as expected | Pass |
| Test Step & Test data | In Registration Page, input test data  1. Name: thinaya1  2. Email: 1234  3. Password: 12345678@  4. Click sign up button | | | | | |
| Evidence | The user's email is in the wrong format, that way it won't register an account. The page will show that “please include an @ in the Email Address”. | | | | | |
| Test Scenario ID | Test Case ID | Test Priority | Pre-condition | Expected Result | Actual Result | Pass/  Fail |
| TS001 | TC002 | High | ABC Cars Portal website has completed and uploaded on server | After successfully register, see whether the data added to the data base | Result is come out  as expected | Pass |
| Test Step & Test data | In Registration Page, input test data  1. Name: duminda@1  2. Email: duminda@gmail.com  3. Password: 12345678@  5.Role:Operator  6. Click sign up button | | | | | |
| Evidence | **Refer SS- SS01** | | | | | |

|  |  |
| --- | --- |
| Test Scenario | Login |
| TS002 |
| Test Cases | Login in Help Desk Portal |
| TC001 | Login as user and limited features |
| TC002 | Login as Admin with full access |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Scenario ID | Test Case ID | Test Priority | Pre-condition | Expected Result | Actual Result | Pass/  Fail |
| TS001 | TC001 | High | Help Desk Car Portal website has completed and uploaded on server | Entering the credentials as user | Result is come out  as expected | Pass |
| Test Step & Test data | In login Page, user data  1. Email : hashan@gmail.com  2. Password: 12345678@  3. Click sign in button | | | | | |
| Evidence |  | | | | | |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  | | | | | |
|  |  | | | | | |

|  |  |
| --- | --- |
| Test Scenario | Provide Solution |
| TS003 |
| Test Cases | Providing the Solution by the Specialist to the Operator |
| TC001 | Providing the Solution by the Specialist to the Operator |
| TCOO2 | Giving a Problem to the Specialist |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Scenario ID | Test Case ID | Test Priority | Pre-condition | Expected Result | Actual Result | Pass/  Fail |
| TS001 | TC001 | moderate | Help Desk Portal website has completed and uploaded on server | Providing the Solution by the Specialist to the Operator | Result is come out  as expected | Pass |
|  |  | | | | | |
| Evidence |  | | | | | |
| Test Scenario ID | Test Case ID | Test Priority | Pre-condition | Expected Result | Actual Result | Pass/  Fail |
| TS001 | TC002 | moderate | Help Desk Portal website has completed and uploaded on server | Providing a Problem to the Specialist | Result is come out  as expected | Pass |
| TOOC2 | Giving a Problem to the Specialist | | | | | |
|  |  | | | | | |

1. Performance Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Requirements | Test scenario id | Test scenario | Number of test cases |
| Non-Functional Testing | PM001 – Performance Testing | Testing the performance of registration page | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Scenario id | Test case id | Test case | Preconditions | Expected results | Actual result | Pass/fail |
| PM001 | TC001 | To test the loading time speed of the registration page using Chrome DevTools | Implementation of Help Desk Protal is complete with all webpages and navigation links and uploaded on localhost server | After testing, The loading time is expected to be less than 2s. | as expected, | pass |
| steps | 1. Enter URL localhost:8080  2. Use Chrome DevTools by right click and choose inspect  3.Click Network, choose No throttling, Fast 3G, Slow 3G  4. Click “Register” on website navigation bar |  | | | | |
| Test data |  |  | | | | |
| Evidence | Fast 3G    Slow 3g | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Scenario id | Test case id | Test case | Preconditions | Expected results | Actual result | Pass/fail |
| PM001 | TC002 | To check the overall performance of the registration page using Chrome DevTools | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | A performance report is generated with performance expected to be above 80 – 90% | Not as expected, | fail |
| steps | 1. Enter URL localhost:8080  2. Click “Register” on website navigation bar  3. Use Chrome DevTools by right click and choose lighthouse  4.Click ‘Analyze Page Load’ |  | | | | |
| Test data |  |  | | | | |
| Evidence |  | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Requirem  ents | Test Scenario Id | Test Scenario | Number of Test Cases |
| Non- Functional Testing | CB001 – Cross Browser Testing | Ensure register page maintain a consistent look and feel in different browser (Chrome, Edge, Firefox) | 3 |
| CB001 – Cross Browser Testing | Ensure login page maintain a consistent look and feel in different browser (Chrome, Edge, Firefox) | 3 |
| CB001 – Cross Browser Testing | Ensure home page maintain a consistent look and feel in different browser (Chrome, Edge, Firefox) | 3 |
| CB001 – Cross Browser Testing | Ensure car detail page maintain a consistent look and feel in different browser (Chrome, Edge, Firefox) | 3 |
| CB001 – Cross Browser Testing | Ensure profile page maintain a consistent look and feel in different browser (Chrome, Edge, Firefox) | 3 |

1. Compatibility Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Scenario id | Test case id | Test case | Preconditions | Expected results | Actual result | Pass/fail |
| CB001 | CBCC001 | Check a consistent look and feel of the register page in Google Chrome | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | After testing, Register page maintain a consistent look and feel in Google Chrome | as expected, | pass |
| steps | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Click ‘Register’ |  | | | | |
| Test data | Forwarding URL = https://517d118-100-204- 247.ap.ngrok.io |  | | | | |
| Evidence |  | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Scenario id | Test case id | Test case | Preconditions | Expected results | Actual result | Pass/fail |
| CB001 | CBCC001 | Check a consistent look and feel of the register page in Firefox | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | After testing, Register page maintain a consistent look and feel in Google Chrome | as expected, | pass |
| steps | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Click ‘Register’ |  | | | | |
| Test data | Forwarding URL = https://517d118-100-204- 247.ap.ngrok.io |  | | | | |
| Evidence |  | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Scenario id | Test case id | Test case | Preconditions | Expected results | Actual result | Pass/fail |
| CB001 | CBCC001 | Check a consistent look and feel of the register page in Edge | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | After testing, Register page maintain a consistent look and feel in Google Chrome | as expected, | pass |
| steps | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Click ‘Register’ |  | | | | |
| Test data | Forwarding URL = https://517d118-100-204- 247.ap.ngrok.io |  | | | | |
| Evidence |  | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | | Expected Results | Actual  Results | | Pass/Fail |
| CB002 | CBCC001 | Check a consistent look and feel of the login page in Google Chrome | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Click ‘Login’ | | Forwarding URL = https://517d118-100-204- 247.ap.ngrok.io | After testing, Login page maintain a consistent look and feel in Google Chrome | | As expected, | Pass |
| Evidence |  | | | | | | | | | |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | | Expected Results | Actual  Results | | Pass/Fail |
| CB002 | CBCC001 | Check a consistent look and feel of the login page in firefox | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Click ‘Login’ | | Forwarding URL = https://517d118-100-204- 247.ap.ngrok.io | After testing, Login page maintain a consistent look and feel in firefox | | As expected, | Pass |
| Evidence |  | | | | | | | | | |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | | Expected Results | Actual  Results | | Pass/Fail |
| CB002 | CBCC001 | Check a consistent look and feel of the login page in edge | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Click ‘Login’ | | Forwarding URL = https://517d118-100-204- 247.ap.ngrok.io | After testing, Login page maintain a consistent look and feel in edge | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  **ID** | Test  **Case ID** | **Test Case** | **Preconditions** | **Steps** | **Test Data** | | **Expected Results** | Actual  **Results** | | **Pass/Fail** |
| CB003 | CBCC001 | Check a consistent look and feel of the home page in Google Chrome | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8082 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Home page maintain a consistent look and feel in Google Chrome | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  **ID** | Test  **Case ID** | **Test Case** | **Preconditions** | **Steps** | **Test Data** | | **Expected Results** | Actual  **Results** | | **Pass/Fail** |
| CB003 | CBCC002 | Check a consistent look and feel of the home page in firefox | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8082 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Home page maintain a consistent look and feel in firefox | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  **ID** | Test  **Case ID** | **Test Case** | **Preconditions** | **Steps** | **Test Data** | | **Expected Results** | Actual  **Results** | | **Pass/Fail** |
| CB003 | CBCC003 | Check a consistent look and feel of the home page in edge | Implementation of ABC Car Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8082 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Home page maintain a consistent look and feel in edge | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | | Expected Results | Actual  Results | | Pass/Fail |
| CB004 | CBCC001 | Check a consistent look and feel of the car detail page in Google Chrome | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal, click ‘View Car Details’ | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Car detail page maintain a consistent look and feel in Google Chrome | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | | Expected Results | Actual  Results | | Pass/Fail |
| CB004 | CBCC002 | Check a consistent look and feel of the car detail page in firefox | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal, click ‘View Car Details’ | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Car detail page maintain a consistent look and feel in firefox | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | | Expected Results | Actual  Results | | Pass/Fail |
| CB004 | CBCC003 | Check a consistent look and feel of the car detail page in edge | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal, click ‘View Car Details’ | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Car detail page maintain a consistent look and feel in edge | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  **ID** | Test  **Case ID** | **Test Case** | **Preconditions** | **Steps** | **Test Data** | | **Expected Results** | Actual  **Results** | | **Pass/Fail** |
| CB005 | CBCC001 | Check a consistent look and feel of the profile page in Google Chrome | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal, click ‘Profile’ | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Profile page maintain a consistent look and feel in Google Chrome | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  **ID** | Test  **Case ID** | **Test Case** | **Preconditions** | **Steps** | **Test Data** | | **Expected Results** | Actual  **Results** | | **Pass/Fail** |
| CB005 | CBCC002 | Check a consistent look and feel of the profile page in firefox | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal, click ‘Profile’ | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Profile page maintain a consistent look and feel in firefox | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  **ID** | Test  **Case ID** | **Test Case** | **Preconditions** | **Steps** | **Test Data** | | **Expected Results** | Actual  **Results** | | **Pass/Fail** |
| CB005 | CBCC003 | Check a consistent look and feel of the profile page in edge | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1.Download ngrok for windows and add authtoken to terminal, enter ngrok http 8080 2.Paste forwarding URL in Lambdatest.com  3. Sign in to the portal, click ‘Profile’ | | Forwarding URL = https://517d-118- 100-204- 247.ap.ngrok.io Username = Faith Password = faith | After testing, Profile page maintain a consistent look and feel in edge | | As expected, | Pass |
| Evidence |  | | | | | | | | | |

1. Portability Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Requirements | Test Scenario ID | Test Scenario | Number of Test Cases |
| Non-Functional Testing | PB001 – Portability Testing | Ensure register page maintain a consistent look and feel in different devices (laptop, iPad, phone) | 3 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | Expected Results | Actual  Results | Pass/Fail |
| PB001 | PBCC001 | To ensure all feature in the register page is visible to the user in laptop | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1. Enter URL localhost:8082 2. navigate to navigation bar   3.Click Register |  | After testing, all register page feature is visible to the user in laptop devices | As expected, | Pass |
| Evidence |  | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | Expected Results | Actual  Results | Pass/Fail |
| PB001 | PBC002 | To ensure all feature in the register page is visible to the user in iPad | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1. Enter URL localhost:8082 2. navigate to navigation bar 3.Click Register 4.right click choose inspect, Toggle device toolbar |  | After testing, all register page feature is visible to the user in iPad devices | As expected, | Pass |
| Evidence |  | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | Expected Results | Actual  Results | Pass/Fail |
| **PB001** | PBC003 | To ensure all feature in the register page is visible to the user in phone | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1. Enter URL localhost:8082 2. navigate to navigation bar 3.Click Register 4.right click choose inspect, Toggle device toolbar |  | After testing, all register page feature is visible to the user in phone devices | As expected, | Pass |
| **Evidence** |  | | | | | | | |

1. Usability Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Requirements | Test Scenario ID | Test Scenario | Number of Test Cases |
| **Functional Testing** | UB001 – Usability Testing | Test whether the register page is user-friendly in design | 1 |
| UB002 – Usability Testing | Test whether the login page is user-friendly in design | 1 |
| UB003 – Usability Testing | Test whether the home page is user-friendly in design | 1 |
| UB004 – Usability Testing | Test whether the car detail page is user-friendly in design | 1 |
| UB005 – Usability Testing | Test whether the profile page is user-friendly in design | 1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | | Test  Case ID | | Test Case | | Preconditions | | Steps | | Test Data | | Expected Results | Actual  Results | Pass/Fail | |
| **UB001** | | UBC001 | | Check all navigation in header and footer works properly and the content is visible in register page and user friendliness | | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | | 1. Enter URL localhost:8082 2. Click “Register” on website navigation bar | |  | | All navigations links is working correctly in the register page and loading the right web pages and having a clear flow easily understood by user | As expected, | Pass | |
| **Evidence** | |  | | | | | | | | | | | | | |
| Scenario  ID | Test  Case ID | | Test Case | | Preconditions | | Steps | | Test Data | | Expected Results | | Actual  Results | Pass/Fail |
| UB002 | UBC001 | | Check all navigation in header and footer works properly and the content is visible in login page and user friendliness | | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | | 1. Enter URL localhost:8082 2.Click ‘Register | |  | | All navigations links is working correctly in the login page and loading the right web pages and having a clear flow easily understood by user | | As expected, | Pass |
| **Evidence** |  | | | | | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | Expected Results | Actual  Results | Pass/Fail |
| UB003 | UBC001 | Check all navigation in header and footer works properly and the content is visible in the home page and user friendliness | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1. Enter URL localhost:8082  2.Sign in to the portal | Email = nithikajayarathne@gmail.com  Password = 12345678@ | All navigations links is working correctly in the home page and loading the right web pages and having a clear flow easily understood by user | Not as expected, confusing how to find the home page and there is no tooltip for the logo and car icon | Fail |
| **Evidence** |  | | | | | | | |

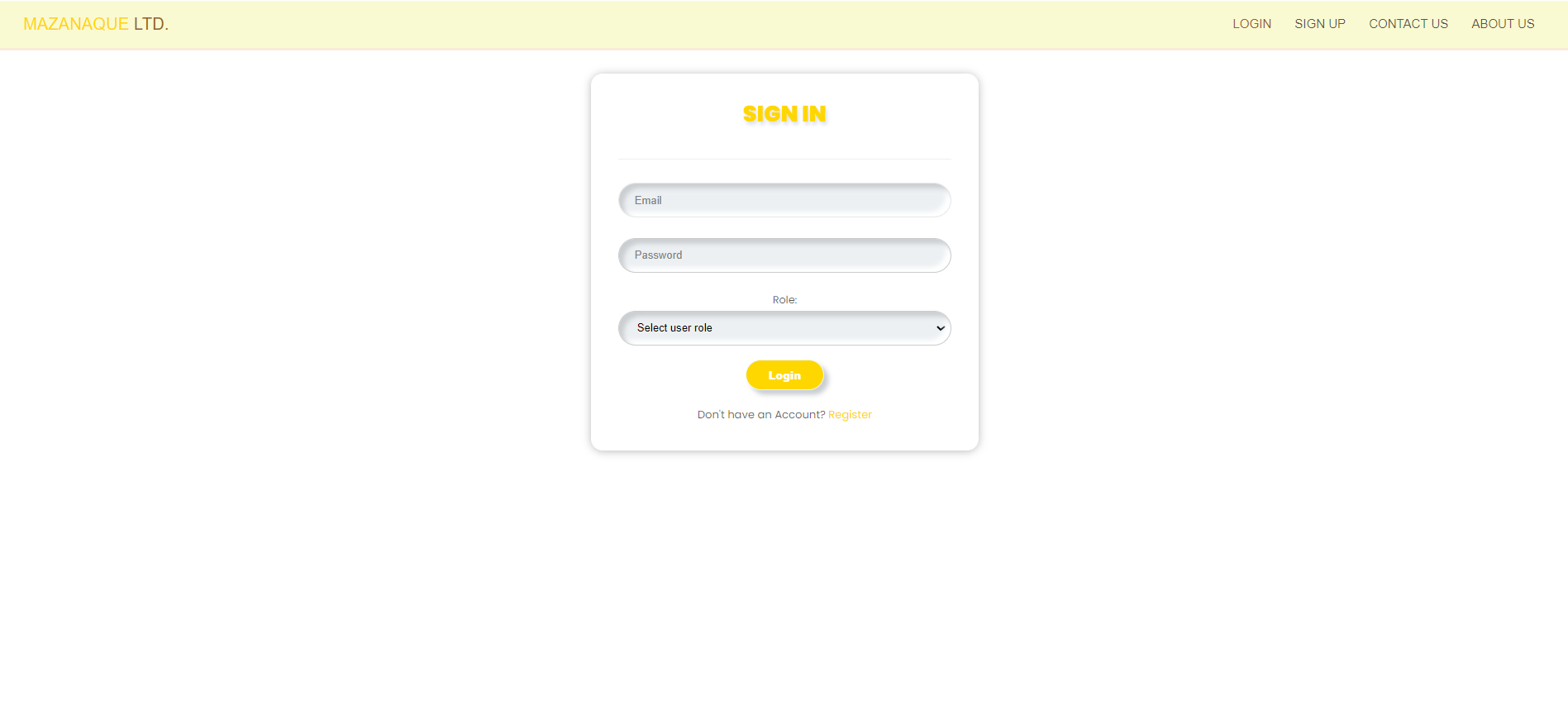
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | Expected Results | Actual  Results | Pass/Fail |
| UB004 | UBC001 | Check all navigation in header and footer works properly and the content is visible in the car detail page and user friendliness | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1. Enter URL localhost:8082 2.Sign in to the portal  3. Click View Car details on any car shown in the home page | Email = nithikajayarathne@gmail.com  Password = 12345678@ | All navigations links is working correctly in the car detail page and loading the right web pages and having a clear flow easily understood by user | As expected, | Pass |
| **Evidence** |  | | | | | | | |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | Expected Results | Actual  Results | Pass/Fail |
| UB005 | UBC001 | Check all navigation in header and footer works properly and the content is visible in the profile page and user friendliness | Implementation of Help Desk Portal is complete with all webpages and navigation links and uploaded on localhost server | 1. Enter URL localhost:8082 2.Sign in to the portal  3. Navigate to home page navigation bar 4.Click Profile | Email = nithikajayarathne@gmail.com  Password = 12345678@ | All navigations links is working correctly in the profile page and loading the right web pages and having a clear flow easily understood by user | As expected, | Pass |
| **Evidence** |  | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario  ID | Test  Case ID | Test Case | Preconditions | Steps | Test Data | Expected Results | Actual  Results | Pass/Fail |

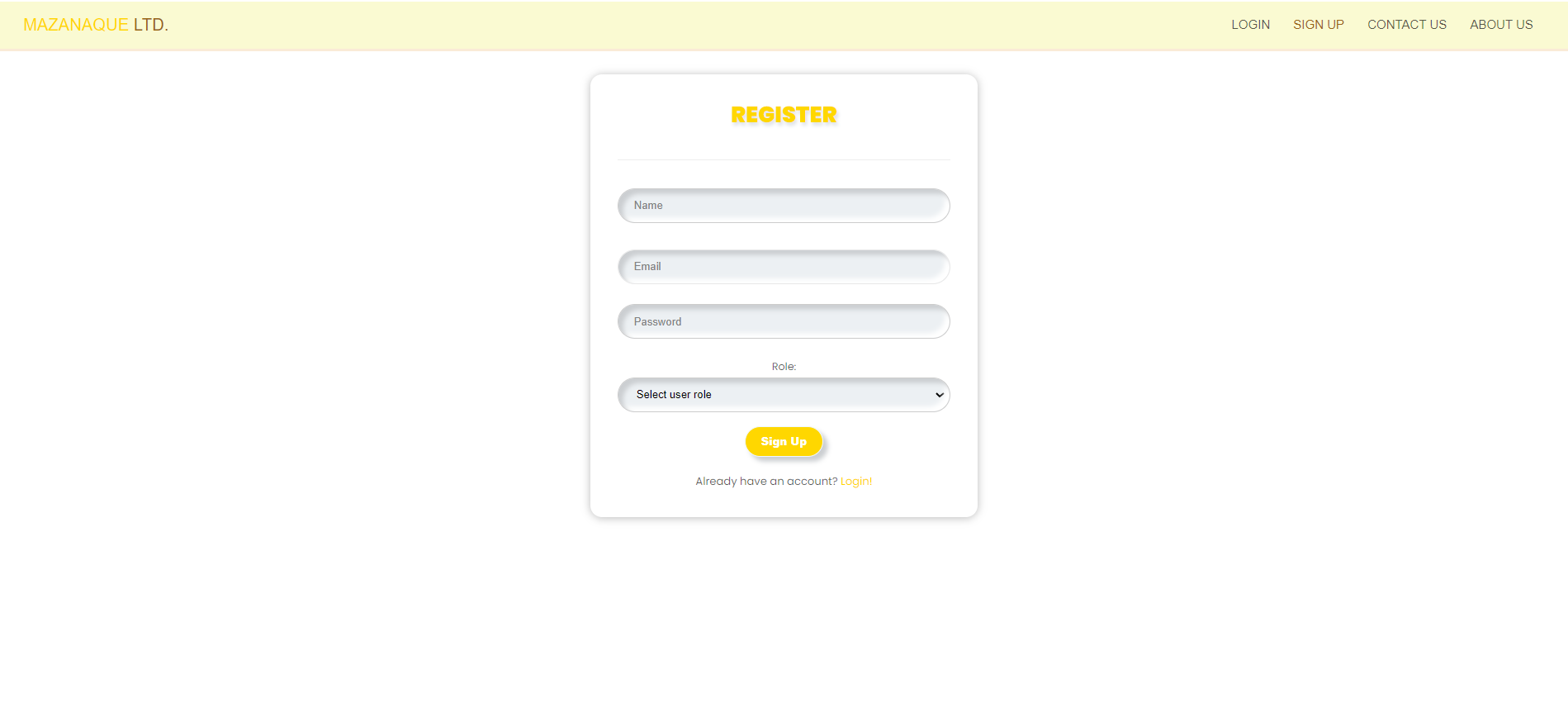
|  |  |
| --- | --- |
| Name of Project | Help Desk Portal Project |
| Reported By: | Nithika Minlaka Jayarathne |
| Report Date: | 06/03/2024 |
|  |  |
| Tested By: | Nithika Minlaka Jayarathne |
| Tested Date: | 06/03/2024 |
| Testing Name/ Test Scenario ID | Functional testing: -Unit Testing/ TS001 / TS002 / TS003 / TS004 /TS005 |
| Total Number of Test Cases | 5 |
| No. of Pass: | 4 |
| No. of Failures: | 1 |
| Elapsed Time: | 00:30:00 |
|  |  |
| Tested By: | Nithika Minlaka Jayarathne |
| Tested Date: | 06/03/2024 |
| Testing Name/ Test Scenario ID | Functional testing: -UAT Testing/ UA001 / UA002 / UA003 / UA004 /UA005 |
| Total Number of Test Cases | 10 |
| No. of Pass: | 9 |
| No. of Failures: | 1 |
| Elapsed Time: | 00:30:00 |
|  |  |
| Tested By: | Nithika Minlaka Jayarathne |
| Tested Date: | 06/03/2024 |
| Testing Name/ Test Scenario ID | Performance Testing/ PM001 |
| Total Number of Test Cases | 2 |
| No. of Pass: | 2 |
| No. of Failures: | 0 |
| Elapsed Time: | 00:10:00 |
|  |  |
| Tested By: | Nithika Minlaka Jayarathne |
| Tested Date: | 06/03/2024 |
| Testing Name/ Test Scenario ID | Compatibility Testing/ CB001 / CB002 / CB003 / CB004 / CB005 |
| Total Number of Test Cases | 15 |
| No. of Pass: | 15 |
| No. of Failures: | 0 |
| Elapsed Time: | 00:40:00 |
|  |  |
| Tested By: | Nithika Minlaka Jayarathne |
| Tested Date: | 06/03/2024 |
| Testing Name/ Test Scenario ID | Usability Testing/ UB001/  UB002/UB003/UB004/UB005 |
| Total Number of Test Cases | 5 |
| No. of Pass: | 4 |
| No. of Failures: | 1 |
| Elapsed Time: | 00:20:00 |
|  |  |
| Tested By: | Nithika Minlaka Jayarathne |
| Tested Date: | 06/03/2024 |
| Testing Name/ Test Scenario ID | Security Testing/ SE001/ SE002/ SE003 |
| Total Number of Test Cases | 5 |
| No. of Pass: | 4 |
| No. of Failures: | 1 |
| Elapsed Time: | 00:20:00 |
|  |  |
| Tested By: | Nithika Minlaka Jayarathne |
| Tested Date: | 06/03/2024 |
| Testing Name/ Test Scenario ID | Portability Testing/ PB001 |
| Total Number of Test Cases | 3 |
| No. of Pass: | 3 |
| No. of Failures: | 0 |
| Elapsed Time: | 00:30:00 |
| SUMMARY LOG STATUS | |
| Total Number of Test Cases | 45 |
| Total Number of Pass | 41 |
| Total Number of Failures | 4 |
| Pass % | 91% |
| Total Elapsed Time | 03:00:00 |

Screenshots of the Web Application

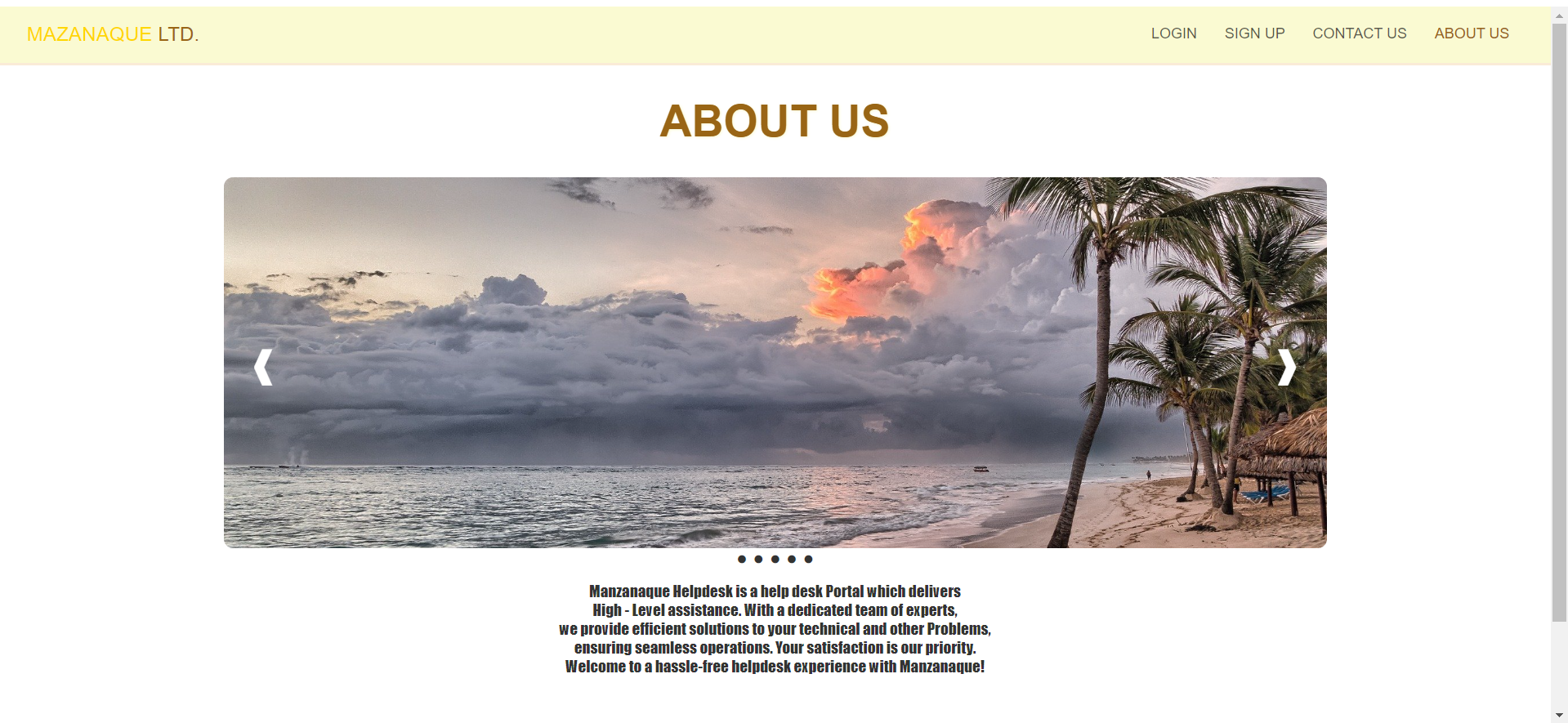
Login Page



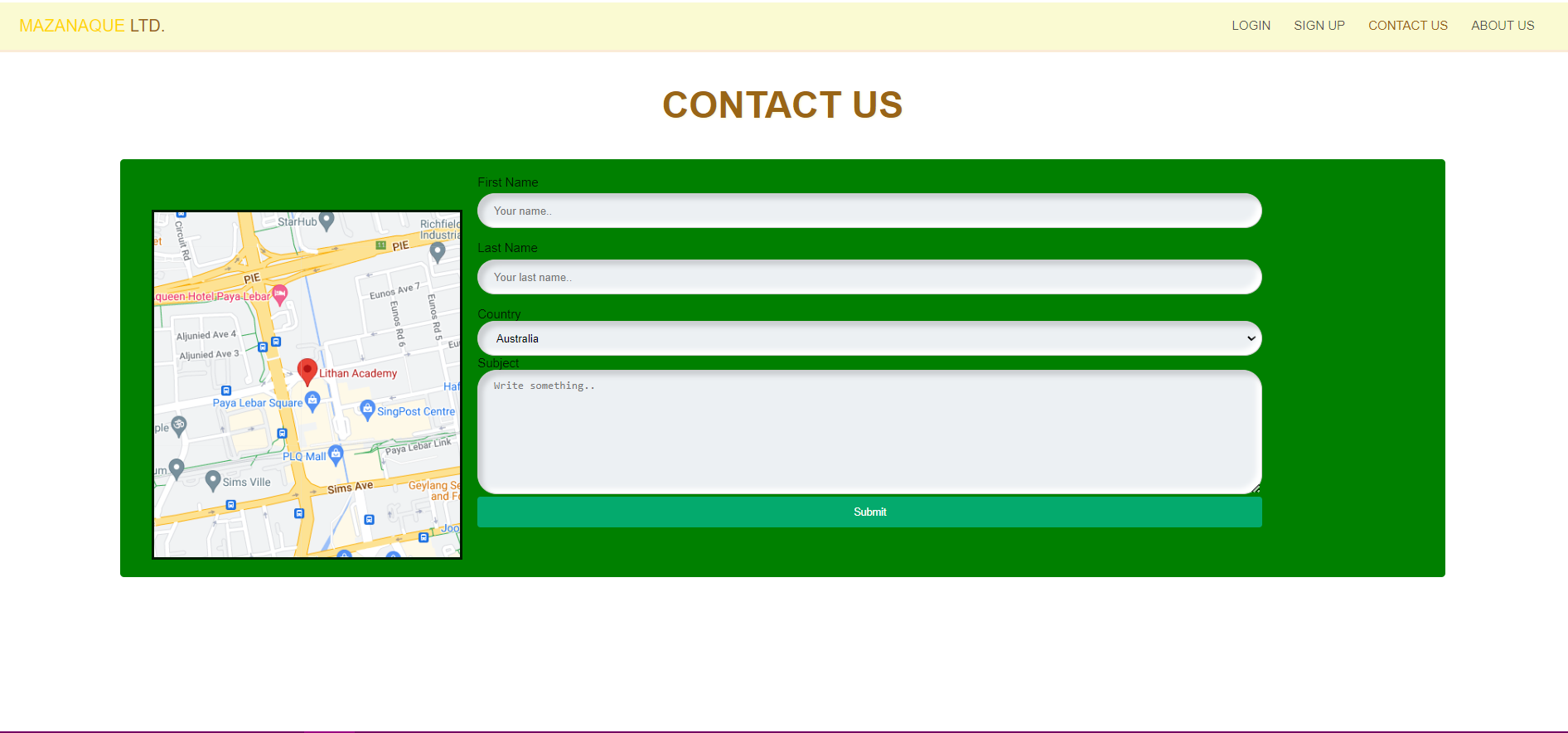
Sign Up Page



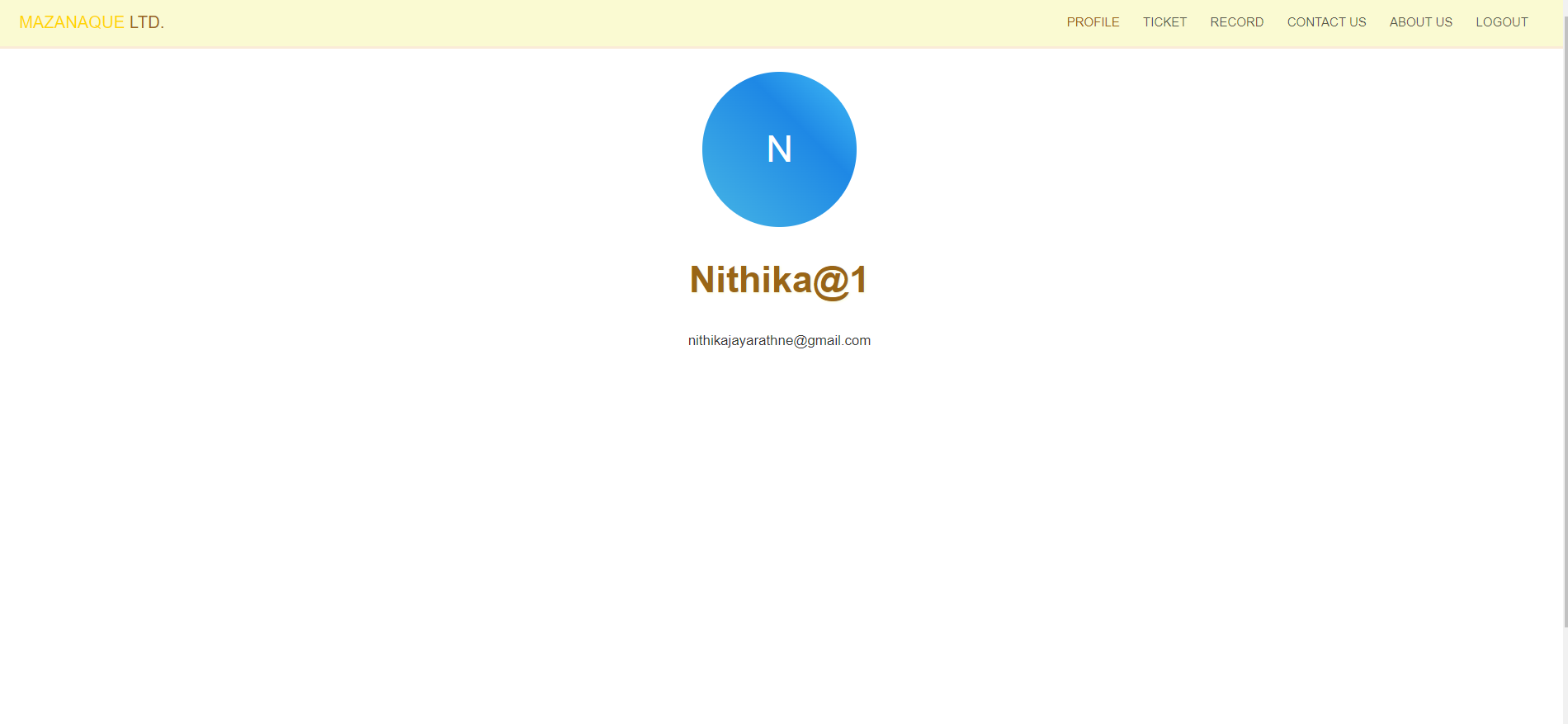
About Us Page



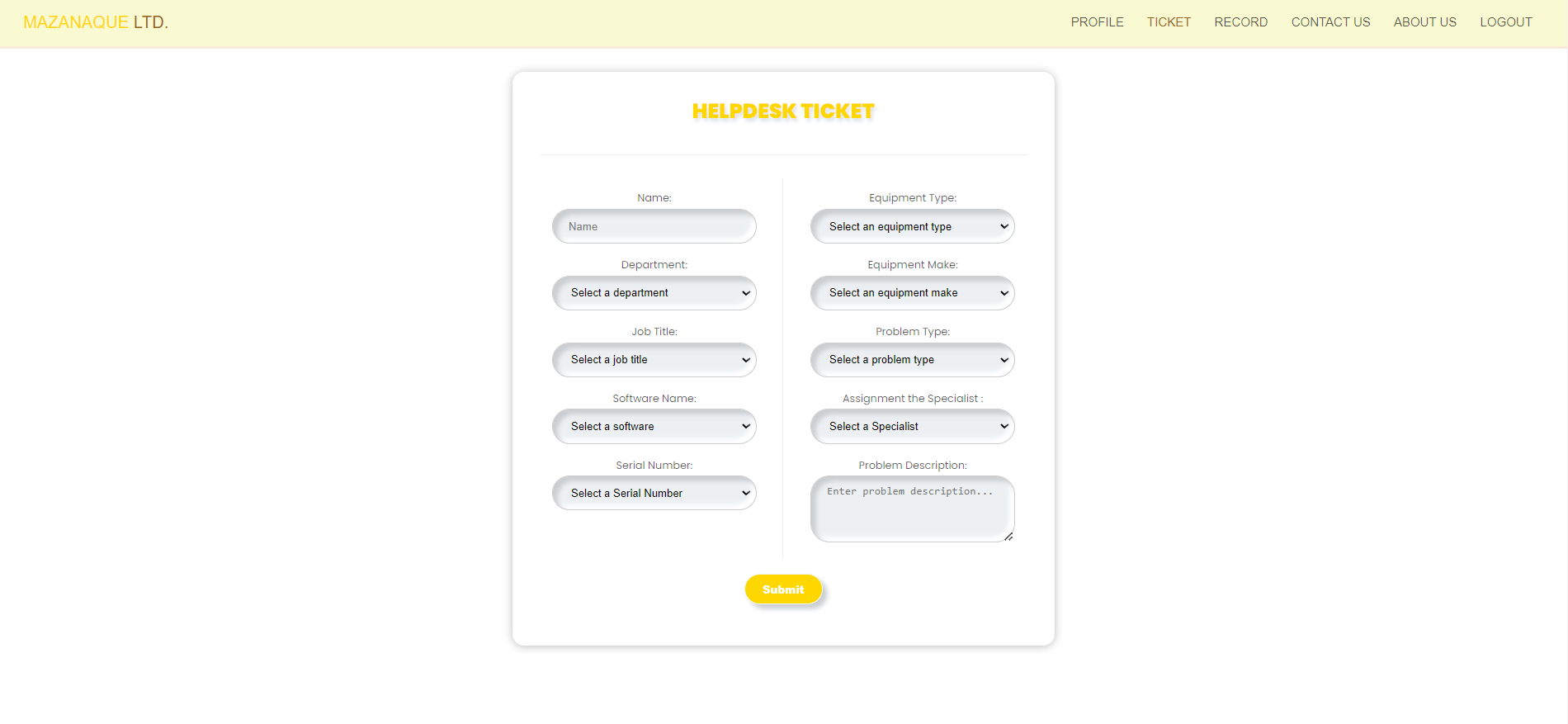
Contact Us Page



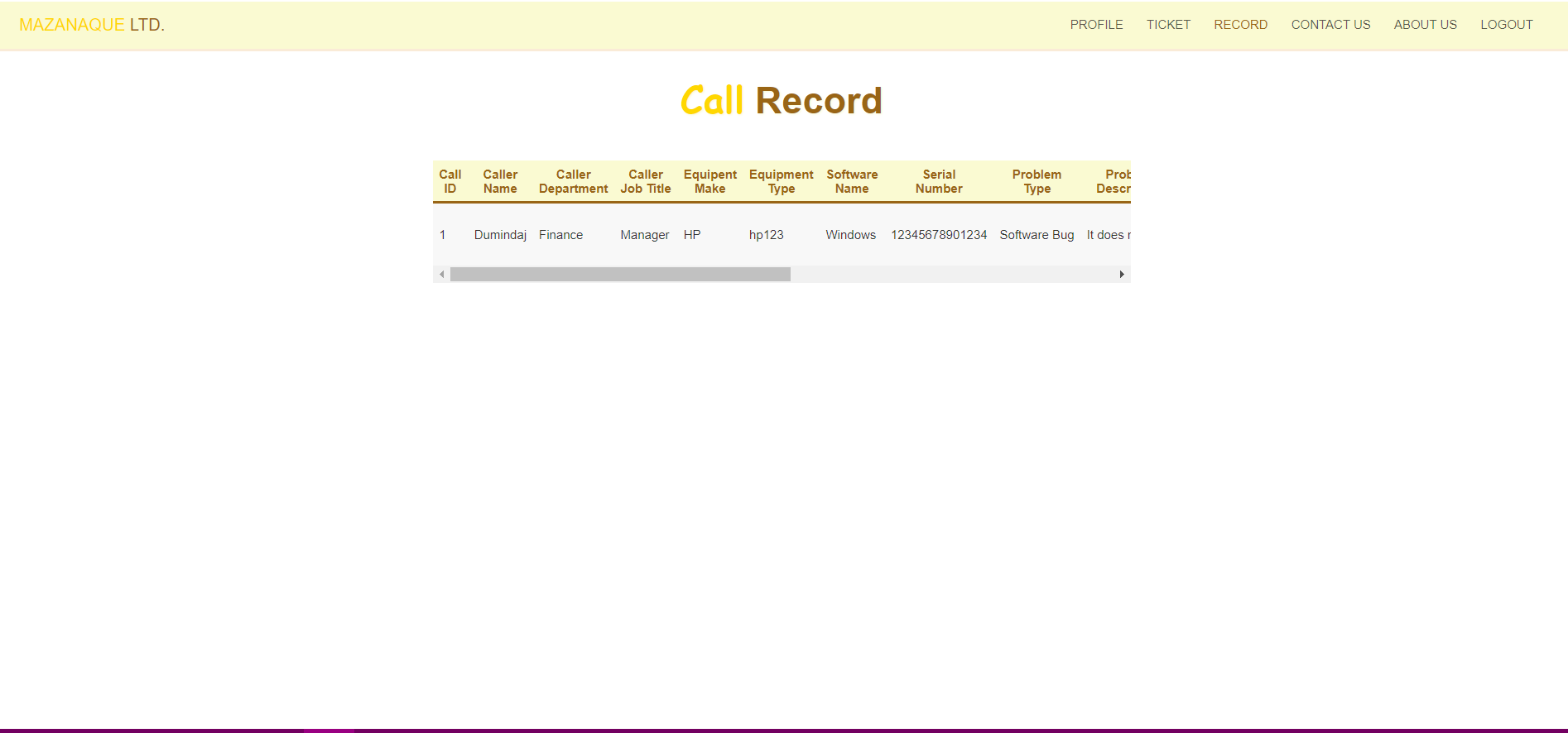
User Profile Page



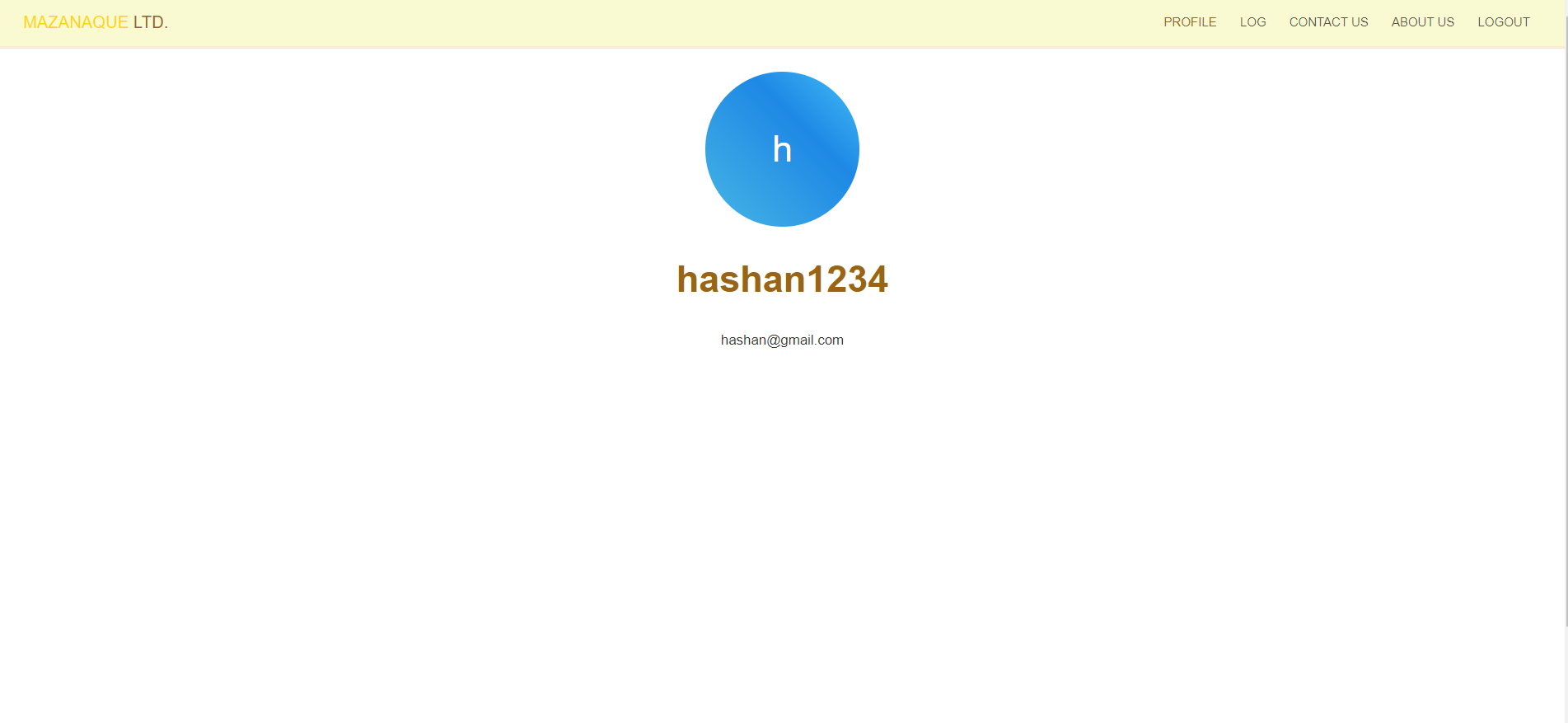
Ticket Page



Record Page



Specialist Page



Call Log Page

