Software Requirements Specification

for

MentorQuest

Version 1.0 approved

Prepared by Tirth Shah

September 8 2024

Table of Contents

1. Introduction

1.1 Purpose

The software requirements specified in this document pertain to **MentorQuest**, a sophisticated platform designed to facilitate mentorship and knowledge exchange between 'Learners' and 'Advisors.' This document covers the initial release (Version 1.0) of the software.

Scope: MentorQuest is intended to provide a seamless interface for 'Learners' to engage with 'Advisors' through a forum where questions can be asked and answered. The system features an invite-based mechanism for 'Advisors,' allowing them to invite others based on a quota of invites that refresh monthly. 'Learners' do not require an invite to participate. Advanced Natural Language Processing (NLP) algorithms will be employed to match questions with suitable Advisors based on relevant tags. Advisors will earn mentor badges as recognition for their contributions when 'Learners' assign them as mentors.

1.2 Intended Audience and Reading Suggestion

Intended Audience: This Software Requirements Specification (SRS) document is intended for the following reader types:

- **Developers:** To understand the technical and functional requirements necessary for implementing MentorQuest. This includes details on user management, question matching, and badge assignment features.
- **Project Managers:** To oversee the development process, ensure alignment with the project's goals, and manage resources and timelines.
- **Marketing Staff:** To grasp the features and functionalities of MentorQuest for promotional purposes and to articulate the product's benefits to potential users.
- **Users:** To gain insight into the features and capabilities of MentorQuest from a user perspective, including how they can engage with the platform as 'Learners' or 'Advisors.'
- **Testers:** To derive test cases and ensure the software meets the specified requirements and performs as expected.
- **Documentation Writers:** To create user manuals, help guides, and other supporting documentation based on the specifications outlined in this document.

Document Organization and Reading Suggestions: The SRS document is organized as follows:

1. **Overview Sections:**
    - **1.1 Purpose**: Describes the purpose of the MentorQuest software and the scope covered by this SRS.
    - **1.2 Intended Audience and Reading Suggestions**: Provides guidance on who should read this document and how to navigate it.
2. **Functional Requirements:**
    - Details the core functionalities of MentorQuest, including user roles, question matching algorithms, and badge assignments.
    - Recommended for Developers and Testers to understand the system's behavior.
3. **Non-Functional Requirements:**
    - Specifies performance, security, and usability criteria.
    - Important for Project Managers and Developers to ensure the system meets quality standards.
4. **System Design and Architecture:**
    - Outlines the design and architecture of MentorQuest.
    - Useful for Developers and Project Managers to understand system components and their interactions.
5. **User Interface Requirements:**
    - Describes the interface requirements and user interactions with the system.
    - Beneficial for Marketing Staff and Documentation Writers to create user guides and promotional materials.
6. **Testing and Validation:**
    - Defines the testing strategy and validation criteria for MentorQuest.
    - Essential for Testers to develop and execute test cases.
7. **Glossary and Appendices:**
    - Provides definitions of terms and additional information relevant to the SRS.
    - Useful for all readers to ensure a common understanding of terms and concepts.

Suggested Reading Sequence:

1. Start with the **Overview Sections** to understand the purpose and audience of the document.
2. Move on to **Functional Requirements** to get detailed insights into what the system will do.
3. Review the **Non-Functional Requirements** to understand the performance and quality criteria.
4. Examine the **System Design and Architecture** for a comprehensive view of the system's structure.
5. Check the **User Interface Requirements** to see how users will interact with the system.
6. Go through the **Testing and Validation** section to understand how the system will be tested.
7. Refer to the **Glossary and Appendices** for additional context and definitions.

1.3 Project Scope

Description: MentorQuest is a comprehensive platform designed to facilitate meaningful mentorship and knowledge exchange between 'Learners' and 'Advisors.' The primary purpose of MentorQuest is to create an interactive environment where users can ask questions, receive guidance, and engage in professional development. The system features an invite-based mechanism for 'Advisors,' advanced Natural Language Processing (NLP) for question matching, and a badge system to recognize valuable contributions.

Benefits:

- **Enhanced Learning Experience:** Provides 'Learners' with access to a wide range of expertise and personalized advice, improving their learning and problem-solving capabilities.
- **Efficient Expertise Utilization:** Allows 'Advisors' to manage their invites and expand their network, enhancing their ability to share knowledge and contribute to the community.
- **Recognition and Motivation:** Implements a badge system to acknowledge 'Advisors' for their mentoring efforts, fostering motivation and continuous engagement.

Objectives:

- **Create a User-Friendly Platform:** Design an intuitive interface that enables seamless interactions between 'Learners' and 'Advisors.'
- **Implement Advanced Matching Algorithms:** Utilize NLP to accurately match questions with the most suitable 'Advisors' based on tags and expertise.
- **Develop a Robust Recognition System:** Establish a badge system to reward 'Advisors' for their mentorship contributions and encourage high-quality engagement.

Goals:

- **Launch Version 1.0:** Deliver a functional version of MentorQuest with core features including user management, question matching, and badge assignment.
- **Achieve User Satisfaction:** Ensure that both 'Learners' and 'Advisors' find the platform valuable and easy to use, leading to high levels of engagement and satisfaction.
- **Align with Business Strategies:** Support corporate goals by fostering a culture of learning and expertise sharing, contributing to professional development and knowledge growth.

Relation to Corporate Goals: MentorQuest aligns with corporate goals by enhancing professional development, supporting continuous learning, and leveraging expertise within the organization. By providing a platform for effective knowledge exchange and recognition, MentorQuest contributes to overall business strategies focused on fostering a knowledgeable and engaged workforce.

1.4 References:

- **Pressbook Publication**

  - **Title:** Software Requirements Specification (SRS)
  - **Author:** SLCC Technical Writing
  - **Source/Location:** Pressbook Publication

- **Geeks For Geeks**

  - **Title:** Software Requirement Specification (SRS) Format
  - **Author:** Geeks For Geeks

- **Source/Location:** Geeks For Geeks

- **University of Texas Dallas**

  - **Title:** SRS Document (Version 4.0)
  - **Author:** University of Texas Dallas
  - **Source/Location:** University of Texas Dallas SRS

2. Overall Description

2.1 Product Perspective

Context and Origin: MentorQuest is a new, self-contained product designed to facilitate mentorship and knowledge exchange between 'Learners' and 'Advisors.' It is not a follow-on member of an existing product family nor a replacement for any existing systems. Instead, MentorQuest represents an innovative solution aimed at enhancing professional development and expertise sharing within a community.

Product Relationship: MentorQuest is designed as an independent platform with its own distinct features and functionalities. However, it is part of a broader strategy to support continuous learning and knowledge management within organizations. While it does not directly interface with existing systems, it may integrate with other tools or platforms in the future to expand its capabilities or enhance user experience.

System Overview: MentorQuest functions as a standalone system with core components that include:

- **User Management:** Handles user registration, role assignments (Learners and Advisors), and invite management.
- **Question Matching:** Utilizes Natural Language Processing (NLP) to match questions with appropriate Advisors based on expertise tags.
- **Badge System:** Tracks and awards badges to Advisors based on their mentoring contributions.

Interfaces:

- **User Interface:** Provides access to various features including forums, question submission, and badge tracking. This is the primary point of interaction for both Learners and Advisors.
- **NLP Engine:** Interfaces with the question matching algorithm to process and analyze user queries.
- **Notification System:** Sends updates and alerts to users about new questions, responses, and badge awards.

2.2 Product Features

Overview: MentorQuest provides many features designed to facilitate effective mentorship and knowledge exchange between 'Learners' and 'Advisors.' The following is a high-level summary of the major features and functionalities of the product:

1. **User Management:**
    - **Registration and Login:** Allows users to create accounts and log in to the system.
    - **Role Assignment:** Differentiates users as 'Learners' or 'Advisors' and manages permissions based on roles.
    - **Invite System:** Enables 'Advisors' to invite other users based on a monthly quota.
2. **Question Submission and Forum:**
    - **Question Posting:** Allows 'Learners' to submit questions for guidance.
    - **Discussion Forum:** Provides a space for ongoing discussions and responses to questions.
3. **Question Matching:**
    - **Natural Language Processing (NLP):** Analyzes questions and matches them with appropriate 'Advisors' based on expertise tags.
    - **Tagging System:** Uses tags to categorize questions and areas of expertise.
4. **Badge System:**
    - **Badge Awarding:** Recognizes 'Advisors' for their contributions and mentoring efforts.
    - **Badge Tracking:** Keeps track of earned badges and displays them on user profiles.
5. **Notification System:**
    - **Alerts and Updates:** Sends notifications to users about new questions, responses, and badge awards.
    - **Email Notifications:** Optionally notifies users via email about important updates.

2.3 User Classes and Characteristics

2.3.1. Learners

- **Description:** Users who seek guidance and mentorship from Advisors. They can ask questions and participate in discussions within the forum.
- **Characteristics:**
    - **Frequency of Use:** Regular users who frequently engage with the platform to seek answers and advice.
    - **Subset of Product Functions Used:** Primarily use the Question Submission and Forum features.
    - **Technical Expertise:** Varies from novice to advanced; generally, they have basic technical skills for interacting with the platform.
    - **Security/Privilege Levels:** Standard user privileges; can view, post questions, and engage in discussions.
    - **Educational Level:** May range from high school to advanced degrees; no specific educational level required.
    - **Experience:** May have varying levels of experience in seeking online mentorship or participating in forums.

2.3.2. Advisors

- **Description:** Users who provide answers and mentorship to Learners. They are responsible for offering valuable insights and guidance based on their expertise.
- **Characteristics:**
    - **Frequency of Use:** Regularly use the platform to respond to questions and manage their invite quota.
    - **Subset of Product Functions Used:** Utilize all features including Question Matching, Badge System, and User Management.
    - **Technical Expertise:** Advanced; must have substantial knowledge in their field of expertise.
    - **Security/Privilege Levels:** Elevated privileges compared to Learners; can manage invites and earn badges.
    - **Educational Level:** Typically holds advanced degrees or has extensive professional experience.
    - **Experience:** Experienced in their field and in mentoring or providing guidance.

2.3.3. Administrators

- **Description:** Users responsible for managing the overall platform, including user accounts, system settings, and overseeing the functionality of the system.
- **Characteristics:**
    - **Frequency of Use:** Less frequent but crucial for maintaining and managing the system.
    - **Subset of Product Functions Used:** Access to all administrative functions including User Management, system configuration, and monitoring.
    - **Technical Expertise:** High; should have technical skills related to system administration and troubleshooting.
    - **Security/Privilege Levels:** Highest privileges; full access to all features and system settings.
    - **Educational Level:** Often has a background in IT or system management.
    - **Experience:** Experienced in system administration and user support.

2.4 Operating Enviornment

2.4.1. Website:

- **Hardware Platform:**
    - **Server Requirements:** Hosted on web servers with sufficient capacity to handle concurrent user access, data processing, and media storage. Recommended specifications include multi-core processors, 16TB or more RAM, and 512TB SSD storage.
- **Operating System and Versions:**
    - **Server OS:** Ubuntu 20.04 LTS
        - **Reason:** A stable and widely used Linux distribution known for its reliability and support, making it ideal for hosting web applications.
- **Web Server Software:**
    - **Software:** Nginx
        - **Reason:** Nginx is efficient in handling high concurrency and is well-suited for serving static files, making it a robust choice for applications with Django and React with Next.js.
- **Web Technologies:**
    - **Frontend:** React with Next.js

        - **Reason:** React provides a dynamic and responsive user interface, while Next.js offers server-side rendering and static site generation, enhancing performance and SEO.
    - **Backend:** Django
        - **Reason:** Django is a high-level Python web framework that facilitates rapid development and includes built-in features for security, database handling, and an admin interface.
    - **Database:** PostgreSQL
        - **Reason:** PostgreSQL is a powerful, open-source relational database that handles complex queries and large datasets efficiently, making it suitable for managing invitor-invitee connections and storing media.

2.4.2. iOS App:

- **Hardware Platform:**
    - **Devices:** iPhone and iPad devices with support for iOS 14 and later.
        - **Reason:** Ensures compatibility with the latest features and security updates available on modern Apple devices.
- **Operating System and Versions:**
    - **iOS Version:** Compatible with iOS 14 and later.
        - **Reason:** Supports the most recent updates and features while maintaining a broad user base.
- **Development Environment:**
    - **IDE:** Xcode
        - **Reason:** Xcode is the official IDE for iOS development, providing all the tools needed to build, test, and debug iOS apps.
    - **Programming Language:** Swift
        - **Reason:** Swift is a modern, safe, and efficient programming language designed specifically for iOS development.

2.4.3. Android App:

- **Hardware Platform:**
    - **Devices:** Android smartphones and tablets with support for Android 10 and later.

        - **Reason:** Ensures compatibility with a wide range of devices while leveraging the latest Android features and security updates.
- **Operating System and Versions:**
    - **Android Version:** Compatible with Android 10 and later.
        - **Reason:** Supports recent updates and provides a good balance between modern features and user coverage.
- **Development Environment:**
    - **IDE:** Android Studio
        - **Reason:** Android Studio is the official IDE for Android development, offering a comprehensive suite of tools for building and testing Android applications.
    - **Programming Languages:** Kotlin, Java
        - **Reason:** Kotlin is a modern language that improves productivity and safety, while Java remains widely used and supported, providing flexibility in development.

Common Considerations:

- **Cross-Platform Compatibility:** Ensure consistent functionality and user experience across the website, iOS app, and Android app.
- **Interoperability:** Backend services should support APIs required for both mobile apps and the website.
- **Security:** Implement security measures across all platforms, including HTTPS for the website and secure data handling for mobile apps.

Coexistence with Other Software:

- **Third-Party Integrations:**
    - **Authentication Providers:**
        - **Examples:** OAuth 2.0
        - **Purpose:** To handle user authentication and authorization securely, enabling single sign-on (SSO) and integration with social login services.

        - **Reason:** Provides a secure and scalable solution for managing user identities and access controls, enhancing user convenience and security.
    - **Analytics Tools:**
        - **Examples:** Google Analytics
        - **Purpose:** To track and analyze user interactions, behaviors, and engagement across the website and mobile apps.
        - **Reason:** Offers insights into user behavior and application performance, enabling data-driven decisions for improving user experience and optimizing features.

2.5 Design and Implementation Constraints

- **Language Considerations:**

    - **Bad Language Filtering:** Implement robust mechanisms for detecting and filtering inappropriate or offensive language within user-generated content. The system must ensure that language used in forums, messages, and other interactive elements adheres to community standards and policies.
        - **Reason:** To maintain a respectful and safe environment for all users, it is essential to prevent and address abusive or harmful language.

- **Content Moderation Team:**

    - **User Reporting and Moderation:** Establish a team responsible for reviewing reports of inappropriate behavior or content. This team will assess whether reported users should be banned or otherwise sanctioned based on the content and context of the reports.
        - **Reason:** Ensures that content moderation is carried out consistently and fairly, with human oversight to evaluate complex situations and enforce community guidelines effectively.

- **Technologies and Tools:**

- **Predefined Tech Stack:** Use the specified technologies and tools, including Nginx for the web server, Django for the backend, React with Next.js for the frontend, and PostgreSQL for the database.
    - **Reason:** To align with the chosen technology stack that has been evaluated for compatibility and performance with the project requirements.

- **Security Considerations:**

    - **Data Protection:** Implement strong security measures, including encryption, secure authentication, and access controls, to protect user data and prevent unauthorized access.
        - **Reason:** Ensures the safety and integrity of user data and system operations, mitigating risks of data breaches and unauthorized access.

- **Design Conventions and Standards:**

    - **Coding Standards:** Follow established coding standards and design conventions to maintain code quality and consistency, facilitating easier maintenance and scalability.
        - **Reason:** Adhering to standards promotes uniformity and readability, making the codebase more manageable for current and future development.

- **Communication Protocols:**

    - **API and Data Exchange:** Use standardized protocols for communication between the frontend, backend, and external services to ensure compatibility and reliable data exchange.
        - **Reason:** Consistent communication protocols prevent integration issues and ensure smooth operation of the application components.

2.6 Assumptions and Dependencies

- **Assumptions:**

    - **Third-Party Components:** The project assumes the availability and reliability of third-party authentication and analytics services, such as OAuth 2.0 for user management and Google Analytics for data analysis.

    - **Impact:** If these services experience outages or change their APIs, it could impact the functionality of the application. For example, if OAuth 2.0 were to face an outage lasting more than 24 hours, it could disrupt user login processes, requiring alternative authentication solutions.
- **User Base and Load:** The project assumes an initial user base of approximately 10,000 active users, with expected growth to 100,000 users within the first year. It is anticipated that the peak concurrent users could reach 1,000 at any given time. Data growth is estimated to be around 50GB per month, including user-generated content such as images and videos.
    - **Impact:** Changes in user load or data volume may necessitate scaling adjustments or optimizations. For example, if user concurrency exceeds 1,500 users, additional server resources or load balancing may be required to maintain performance.

- **Dependencies:**

    - **External Services:**
        - **Authentication Providers:** Dependence on services like OAuthfor handling user authentication and authorization.
        - **Analytics Tools:** Dependence on platforms such as Google Analytics for tracking and analyzing user interactions.
        - **Impact:** If these services were to discontinue their offerings or increase pricing, the project might need to find alternative solutions. For instance, if Google Analytics were to double its pricing, it could increase the project's operational costs significantly.
    - **Integration with Other Systems:** The project relies on the ability to integrate seamlessly with other systems and services, such as email providers.
        - **Impact:** Issues with integration or changes in third-party systems could require adjustments. For example, if the email provider discontinues their services then the user might not get notified about their updates, especially the web users.
    - **Technology Stack:** The project depends on the specified technologies (Django, React with Next.js, PostgreSQL) being supported and well-maintained throughout the development lifecycle.

- **Impact:** If any of these technologies were to become deprecated or encounter critical issues, it might require changing the technology stack. For example, if PostgreSQL faced a critical security flaw that wasn't patched, it might necessitate migrating to an alternative database, potentially extending the development timeline by several months.

2.7 Suggested Development Model

Suggested Development Model: Agile Development Model

The Agile model is the most suitable development approach for **MentorQuest** for several reasons:

1. **Iterative and Incremental Approach:** MentorQuest is a complex system involving multiple user types (Learners, Advisors, Admins) and features (invite system, NLP-based query matching, etc.). Agile's iterative cycles allow the team to develop and release the system incrementally, gathering feedback from users at each stage. This is especially important for refining the invite-based expert system and improving NLP-based matching.
2. **Adaptability:** The scope of the system could evolve as requirements may change based on feedback from the initial releases. The Agile methodology is flexible enough to adapt to changing project goals, feature adjustments, and shifting user needs.
3. **Rapid Feedback and Adjustment:** MentorQuest's success depends heavily on user satisfaction (both Learners and Advisors), and Agile enables continuous testing and improvements based on real user feedback, minimizing risk.
4. **Collaboration Focus:** With a multidisciplinary team (developers, data scientists for NLP, UI/UX designers), Agile fosters collaboration and ensures that cross-functional teams work cohesively through short sprints.
5. **Faster Time-to-Market:** Early delivery of a Minimum Viable Product (MVP), even with basic features like user invitations and initial question-answering capabilities, can help establish the platform and provide value sooner rather than waiting for the full feature set.

3. System Features

3.1 User Authentication

REQ-1: The system must allow users to register with a valid email address and password, and optionally using social media accounts (e.g., Google, Facebook).

REQ-2: The system must support user login using email/username and password, and optionally through social media accounts.

REQ-3: The system must provide multi-factor authentication (MFA) as an optional security feature during login.

REQ-4: The system must enable users to recover forgotten passwords by sending a password reset link to their registered email address.

REQ-5: The system must encrypt passwords and other sensitive data both in transit and at rest to ensure security.

REQ-6: The system must lock user accounts after a specified number of failed login attempts to prevent brute-force attacks.

REQ-7: The system must provide users with the ability to update their passwords through their account settings.

3.2 User Profile Management

REQ-1: The system must allow users to view and edit their profile information, including their name, email, and contact details.

REQ-2: The system must allow users to upload and update their profile picture, with validation for file type and size.

REQ-3: The system must enable users to set and modify their notification preferences.

REQ-4: The system must display a user-friendly interface for managing account settings and viewing user activity history.

REQ-5: The system must provide secure access to user profile information, ensuring that only authorized users can view or edit their profiles.

3.3 Messaging and Notifications

REQ-1: The system must support sending and receiving private messages between users, including text and attachments.

REQ-2: The system must deliver notifications for new messages, mentions, or relevant activities to the user's inbox or notification center.

REQ-3: The system must allow users to customize their notification preferences, including which types of notifications they receive and how they are delivered (e.g., email, in-app).

REQ-4: The system must provide a real-time messaging interface with notifications for incoming messages and updates.

REQ-5: The system must allow users to view their message history, including past conversations and attachments.

3.4 Content Moderation

REQ-1: The system must allow users to report inappropriate or abusive content, including messages, posts, or profile information.

REQ-2: The system must log all content reports and alert the moderation team for review.

REQ-3: The system must provide moderation tools for reviewing reported content, including options to hide, delete, or flag content.

REQ-4: The system must provide a mechanism for moderators to take action on reported content, such as issuing warnings or banning users.

REQ-5: The system must notify users of moderation decisions related to their content, including reasons for content removal or user bans.

REQ-6: The system must ensure that moderation processes comply with community guidelines and legal requirements, maintaining transparency and fairness.

3.5 User and Expert Interaction

REQ-1: The system must allow Learners to post questions in forums without needing an invite.

REQ-2: The system must enable Advisors to accept or decline mentoring invitations and manage their list of invitees.

REQ-3: The system must provide a mechanism for Advisors to earn badges based on Learner interactions and mentor assignments.

REQ-4: The system must match questions from Learners with relevant Advisors using Natural Language Processing (NLP) based on tags and expertise.

REQ-5: The system must allow Advisors to track their mentoring history and performance metrics.

3.6 Inviter and Invitee Management

REQ-1: The system must support an invitation-based model for Advisors to invite other Advisors to join the platform.

REQ-2: The system must track and manage the number of invites each Expert has, including invite usage and expiration.

REQ-3: The system must notify Advisors when they receive an invitation and allow them to accept or reject the invitation.

REQ-4: The system must enable invited Advisors to complete their registration and join the platform upon accepting the invitation.

REQ-5: The system must provide administrative tools for managing invite requests, including reviewing and approving pending invitations.

3.7 Data Management and Storage

REQ-1: The system must securely store user-generated content, including images, videos, and messages.

REQ-2: The system must provide efficient data retrieval and management capabilities to handle growing user data and activity.

REQ-3: The system must implement backup and recovery procedures to ensure data integrity and availability.

REQ-4: The system must provide access control mechanisms to protect sensitive data from unauthorized access.

REQ-5: The system must comply with data privacy regulations, such as GDPR or CCPA, ensuring user data is handled and stored appropriately.

4. External Interface Requirement

4.1 User Interface

4.1.1 Overview

The MentorQuest software will include multiple user interfaces, tailored to each platform (web, iOS, and Android) but maintaining consistency in design principles and functionality. The interfaces will be designed to ensure an intuitive user experience while adhering to established GUI standards and accessibility guidelines.

4.1.2 General Characteristics

- **Consistency:** All interfaces across web, iOS, and Android platforms will adhere to a unified design language to ensure a cohesive user experience. This includes color schemes, typography, and branding elements.
- **Responsiveness:** User interfaces will be responsive and adapt to various screen sizes and orientations, ensuring usability on desktops, tablets, and smartphones.
- **Accessibility:** Interfaces will follow accessibility guidelines to ensure that users with disabilities can navigate and interact with the system effectively. This includes support for screen readers, keyboard navigation, and sufficient color contrast.

4.1.3 Web Interface (React with Next.js)

- **Home Page:** Features a clean layout with navigation menus, login/register options, and access to key functionalities such as forums, user profiles, and notifications.
    - **Navigation Bar:** Includes links to the Home page, Forum, Profile, Messages, and Notifications.
    - **Footer:** Contains links to Terms of Service, Privacy Policy, and Contact Support.
- **Profile Management:** Allows users to view and edit their profile information, including profile picture, bio, and contact details.
    - **Edit Profile Button:** Accessible on the profile page for updating personal information.

    - **Notification Preferences:** Accessible via the profile settings to customize notification settings.
- **Messaging Interface:** Provides a real-time chat experience with options for sending text, images, and files.
    - **Message List:** Displays recent conversations and notification for new messages.
    - **Message Input Field:** Allows users to compose and send messages.
- **Content Moderation:** Features tools for reporting inappropriate content and reviewing reported items.
    - **Report Button:** Available on content items to report issues.
    - **Moderation Dashboard:** Accessible by moderators to review and manage reports.

4.1.4 iOS Interface (Swift)

- **Home Screen:** Provides access to key functionalities such as login/register, user profile, and notifications.
    - **Tab Bar:** Includes icons for Home, Forums, Profile, and Messages.
    - **Navigation Controls:** Standard iOS navigation controls for seamless movement between screens.
- **Profile Screen:** Users can view and edit their profile information.
    - **Edit Button:** Located at the top-right corner for profile updates.
    - **Profile Image Upload:** Allows users to add or change their profile picture.
- **Messaging Screen:** Supports sending and receiving messages with real-time updates.
    - **Chat Bubbles:** Display incoming and outgoing messages.
    - **Attachment Options:** Allows users to attach photos, videos, and files.
- **Notifications:** Displays alerts for new messages, mentions, or other relevant activities.
    - **Notification Center:** Accessible from the Home screen for viewing recent notifications.
- **Content Moderation:** Tools for reporting and reviewing content.
    - **Report Option:** Available on content items for reporting issues.
    - **Moderator Dashboard:** For managing and reviewing reported content.

4.1.5 Android Interface (Kotlin, Java)

- **Home Screen:** Features a layout with quick access to login/register, forums, user profiles, and notifications.
    - **Navigation Drawer:** Provides access to Home, Forums, Profile, Messages, and Settings.
    - **Floating Action Button:** Allows quick access to common actions such as creating a new post or message.
- **Profile Management:** Users can manage their profile information and settings.
    - **Profile Edit Option:** Available from the profile page for making changes.
    - **Notification Settings:** Customizable from the profile settings.
- **Messaging Interface:** Allows users to interact with messages and attachments.
    - **Chat List:** Shows recent conversations with notification badges for unread messages.
    - **Message Composition:** Provides options for text and media attachments.
- **Content Moderation:** Tools for reporting and reviewing content.
    - **Report Option:** Available on content items for reporting issues.
    - **Moderator Dashboard:** For managing and reviewing reported content.

4.1.6 Common User Interface Components

- **Standard Buttons:** Include common actions such as Save, Cancel, Edit, and Delete, ensuring uniformity across platforms.
- **Error Messages:** Display clear, concise messages for user errors and system issues, with guidance on how to resolve them.
- **Help and Support:** Accessible from any screen, providing users with resources and contact options for assistance.

4.1.7 User Interface Design Document

A detailed user interface design document will be created separately, outlining specific screen layouts, component designs, and user interaction flows. This document will include wireframes, mockups, and design guidelines to ensure consistency and usability across all platforms.

4.2 Hardware Interface

4.2.1 Overview

The MentorQuest software interacts with various hardware components depending on the platform it is running on. These interactions involve logical and physical characteristics specific to each device type.

4.2.2 Web Platform (React with Next.js)

- **Supported Device Types:**
  - **Desktops and Laptops**
  - **Tablets**
- **Physical Characteristics:**
  - **Display:** Supports a range of resolutions from standard to high-definition.
  - **Input Devices:** Includes keyboard and mouse.
  - **Peripheral Devices:** Compatibility with external storage and printers for data export and backup.
- **Communication Protocols:**
  - **HTTP/HTTPS:** Used for secure communication between the web browser and the backend server.
  - **WebSockets:** Enables real-time communication, such as live chat and instant notifications.

4.2.3 iOS Platform (Swift)

- **Supported Device Types:**
  - **iPhones:** Various models with different screen sizes and resolutions.
  - **iPads:** Tablets with large screens.

- **Physical Characteristics:**
  - **Display:** Optimized for Retina and other high-resolution displays.
  - **Touchscreen:** Supports multi-touch gestures and on-screen keyboard input.
  - **Sensors:** Integration with device sensors like camera, accelerometer, and gyroscope for enhanced functionality.
- **Communication Protocols:**
  - **HTTP/HTTPS:** For secure data transmission with backend services.

4.2.4 Android Platform (Kotlin, Java)

- **Supported Device Types:**
  - **Smartphones:** Various models with different screen sizes and resolutions.
  - **Tablets:** Larger devices with varied resolutions.
- **Physical Characteristics:**
  - **Display:** Adapted for different resolutions (e.g., HD, Full HD, Quad HD).
  - **Touchscreen:** Supports multi-touch and gesture-based interactions.
  - **Sensors:** Interaction with GPS, camera, and accelerometer for enhanced functionality.
- **Communication Protocols:**
  - **HTTP/HTTPS:** Ensures secure communication with the backend server.

4.3 Software Interface

4.3.1 Overview

MentorQuest interacts with several software components, including databases, operating systems, and external libraries. These interfaces define how the software components communicate and share data.

4.3.2 Software Components

- **Backend Server:**
  - **Name:** Django
  - **Version:** 4.1
  - **Purpose:** Handles application logic, user authentication, and data processing.
- **Frontend Framework:**
  - **Name:** React with Next.js
  - **Version:** 18.x and 13.x respectively
  - **Purpose:** Provides the user interface and interacts with the backend via API calls.
- **Database:**
  - **Name:** PostgreSQL
  - **Version:** 15.x
  - **Purpose:** Stores user data, including profiles, messages, and media content.
- **Operating Systems:**
  - **Server OS:** Ubuntu 20.04 LTS
  - **iOS:** iOS 16.x and above
  - **Android:** Android 12.x and above
- **Libraries and Tools:**
  - **Authentication:** OAuth 2.0 for secure user authentication.
  - **Analytics:** Google Analytics for tracking user interactions and engagement.
  - **Natural Language Processing:** SpaCy f or advanced natural language processing tasks, including question matching and intent recognition.

4.3.3 Data Items and Messages

- **Incoming Data:**
    - **User Data:** From user registrations, logins, and profile updates.
    - **Messages:** Real-time chat messages and notifications.
- **Outgoing Data:**
    - **Response Data:** Data sent back to users as a result of their requests (e.g., confirmation messages, search results).
    - **Analytics Data:** Interaction metrics sent to analytics tools for reporting.

4.3.4 Data Sharing and Communication

- **Shared Data:**
    - **User Profiles:** Shared between frontend and backend.
    - **Messages:** Stored in the database and accessed by both backend services and frontend interfaces.
- **Data Sharing Mechanisms:**
    - **APIs:** RESTful APIs for data access and manipulation.
    - **Database Queries:** Direct interactions with the PostgreSQL database for data retrieval and updates.

4.4 User Interface

4.4.1 Overview

MentorQuest requires various communication functions to operate effectively, including network protocols, data formats, and security measures.

4.4.2 Communication Functions

- **E-Mail:**
    - **Purpose:** For account verification, password recovery, and notifications.
- **Network Server Communications:**
    - **Protocols:**
        - **HTTP/HTTPS:** For secure client-server communication.
        - **WebSockets:** For real-time data updates and messaging.

4.4.3 Data Formatting

- **Message Formatting:**
    - **JSON:** Used for data interchange between frontend and backend.
    - **HTML:** For rendering web content in the browser.
- **Communication Standards:**
    - **HTTP/HTTPS:** Ensure secure data transfer.
    - **FTP:** For file transfer if needed.

4.4.4 Security and Encryption

- **Encryption:**
    - **Data Transmission:** HTTPS for secure communication over the web.
    - **Data Storage:** Encryption of sensitive user data in the database.
- **Synchronization Mechanisms:**
    - **Real-Time Updates:** Implemented via WebSockets for instant message delivery and notifications.

5. Other Non Functional Requirements

5.1 Performance Requirements

- **Response Time:**
  - The system must provide real-time responses to user queries within 1-2 seconds, including NLP-processed responses, to ensure a seamless user experience.
- **Concurrent Users:**
  - The platform should support at least 10,000 concurrent users without significant performance degradation.
- **Server Uptime:**
  - The system must maintain an uptime of 99.9% to ensure continuous availability for both Learners and Advisors.
- **NLP Processing Time:**
  - SpaCy-powered question matching and tagging should complete within 500 milliseconds to allow for efficient real-time query processing.
- **Media Upload and Retrieval:**
  - The platform must handle media (images, videos) upload and retrieval within 3 seconds, even for high-resolution files, to avoid delays in user interactions.
- **Database Queries:**
  - All database operations (e.g., fetching invites, storing messages) should be completed within 300 milliseconds to prevent bottlenecks during high traffic periods.

5.2 Safety Requirements

The software must implement safeguards to prevent any harm or damage due to potential failures. In particular:

- **Data Backup:** The system must have an automated backup process to prevent loss of data due to server failures or crashes.

- **User Reporting and Moderation:** Any offensive content reported by users must be reviewed within 24 hours, and a moderation team should be responsible for determining if the user should be warned, suspended, or banned based on the violation.
- **Data Integrity:** Integrity checks will be performed regularly to avoid corruption of critical data.
- **Compliance:** The system must adhere to relevant industry safety regulations and certifications such as ISO, etc..

5.3 Security Requirements

To ensure data privacy and system integrity, the following security measures are mandatory:

- **Authentication:** User identity must be verified through a secure login system. Multi-factor authentication (MFA) should be implemented for Advisors and other sensitive roles.
- **Data Encryption:** All sensitive user data, such as personal information, must be encrypted both at rest and in transit using TLS encryption.
- **Access Control:** Role-based access control (RBAC) will restrict access to specific system functionalities. Learners, Advisors, and Admins will have different privilege levels.
- **Audit Logging:** The system must log all critical operations (e.g., login attempts, changes in user roles, reported content). These logs should be available for review by security personnel.
- **Third-party Integrations:** External authentication providers (Google, Facebook) will be used for login. APIs used in the application must adhere to the industry security standards.

5.4 Software Quality Attributes

The following quality characteristics are part of the initial release of the product:

- **Availability:** The system is designed to be available at least 99.5% of the time, ensuring that users have access to the platform for expert consultations and learning sessions without frequent disruptions.

- **Correctness:** All core functionalities, including user registration, login, question posting, and expert invitations, are validated to follow defined business logic, preventing erroneous operations or data corruption.

- **Maintainability:** The initial release has a modular codebase, allowing developers to easily update components, fix bugs, and add new features without introducing issues in other parts of the system.

- **Reliability:** The software has been tested under typical usage conditions and can handle expected loads without crashing. It provides reliable performance for posting and answering questions, as well as processing invites.

- **Usability:** The user interface is intuitive, designed to ensure easy navigation for both learners and Advisors. The platform adheres to standard UI/UX practices, making it simple for users to ask questions, search for answers, and interact with Advisors.

- **Security:** Basic security features, such as encrypted password storage and secure login, have been implemented in the initial release to protect user data from unauthorized access.

- **Portability:** The application runs across different platforms, including web, iOS, and Android, with minimal differences in user experience, ensuring consistent functionality across devices.

- **Usability:** The interface should be intuitive and user-friendly, ensuring that users can easily navigate through different features without confusion.

Appendix A: Glossary

- **SRS (Software Requirements Specification):** A document that describes the functional and non-functional requirements of the system.

- **Learner:** A user of the platform who seeks guidance by asking questions or interacting with Advisors (advisors).

- **Expert:** A user who answers questions and provides mentorship to learners. Also referred to as an Advisor.

- **Invite:** A system feature allowing Advisors to invite others to join the platform, with a limited number of invites available each month.

- **Mentor Badge:** A reward given to Advisors when a learner assigns them as a mentor after receiving guidance.

- **NLP (Natural Language Processing):** A technology used to analyze and understand user queries to match them with the most relevant Expert.

- **Frontend:** The part of the application that the user interacts with, built with React and Next.js for this project.

- **Backend:** The server-side logic of the application, managed by Django in this case.

- **API (Application Programming Interface):** A set of functions allowing one software to communicate with another, such as between frontend and backend components.

Appendix B: Issues List

- **Decision Pending On:**
    - Establishing the number of invites Advisors receive each month and whether this amount can be dynamically increased based on other specific criteria (e.g., engagement level).
    - Deciding whether to include a probation period for new Advisors where after they answer a limited number of questions before earning the ability to invite others to the platform.