Software Requirements Specification (SRS)

for
Skill Swap

Prepared by

Abd AL Mohsin Siraj (2254901009),
Mridula Mozid (2254901037),
Samiul Hossen Sarkar Santo (2254901079),
Nishat Tasneem (2254901085),
Md.Shehab Uddin (2254901121),
Md. Mubstasim Abedin Jaowad(22540901123)

October 7, 2025

Contents

Document History

Version	Date	Changes	Author(s)
SRS - 1.0	September 16, 2025	Initial draft of the SRS document.	[Mohsin, Mridula, Santo, Nishat, Shehab, Jaowad]
SRS - 2.0	October 07, 2025	Pre-Approved Final Draft of the SRS document.	[Mohsin, Mridula, Santo, Nishat, Shehab, Jaowad]

Chapter 1

Introduction

The SkillSwap is a mutual knowledge exchange platform designed for learners who seek to acquire skills without financial barriers. It offers a trustworthy, flexible, and well-structured learning experience that connects users through verified expertise and micro-learning roadmaps. Unlike costly online courses or gig-based platforms, the SkillSwap introduces an equitable, non-monetary system built entirely on the value of shared human knowledge and verified contributions to learn, teach, and grow together.

1.1 Purpose

This document outlines the software requirements for *SkillSwap(Version 1.0)*, a knowledge exchange platform designed to challenge traditional monetary learning systems.

Inspired by platforms like Fiverr and Coursera, SkillSwap focuses on mutual growth, inclusivity, and accessibility through a "knowledge exchange" model. In this system, individuals can both request the learning of categorized skills and offer to teach their own expertise. The primary purpose of this is to establish a secure and interactive platform for direct skill exchange, focusing on flexible micro-learning, roadmap-based progression, and trustworthy collaboration achieved through verification, reviews, and smart matchmaking. This document aims to give a complete outline of the required features, performance expectations, limits, and connections necessary for developing, implementing, and maintaining the SkillSwap software.

1.2 Intended Audience

The Software Requirements Specification is designed to cater to a diverse audience involved in the development and utilization of the SkillSwap platform. The primary stakeholders include:

1. **Potential Investors:** Interested in gaining a clear idea of the project's scope, potential, and operational framework for easier decision-making.
2. **Project Managers:** Responsible for tracking the project efficiently with a clear understanding of the work and resources.
3. **Developers:** Entrusted with the coding and implementation, requiring detailed project features, logic, and coding standards.
4. **UI/UX Designers:** Understanding the user's point of view to design features and the UI flow.

5. **Business Analysts:** Tasked with understanding business goals and ensuring they are properly aligned with requirements.

6. **QA/QC Team:** Ensuring product quality by creating test plans, cases, and acceptance measures.

7. **Administrators**: Responsible for overall system management.

8. **Moderators**: Responsible for understanding and managing the skill verification process efficiently.

9. **End-users:** Understanding the features and best use of the system.

1.3 Intended Use

This SRS is intended to serve as a guideline for all stakeholders involved in the Skillswap project. This subsection describes how they should use the SRS for a better appreciation of the purposes and features of the project.

1. **Potential Investors:** They can use this SRS to get an understandable and accurate knowledge of the scope, possibility, work structure, and risk of the project, thus making sound investment decisions.

2. **Project Managers:** This document gives a clear understanding of the scope of the project, resources, and budget, thus allowing them to track progress and manage the project effectively.

3. **Developers:** The document is a technical manual that offers a vivid specification of project features, logic, and coding conventions to be followed.

4. **UI/UX Designers:** They would look at the SRS for understanding the different categories of users, their needs, and the system features. This understanding is crucial for developing an intuitive user interface (UI) as well as a smooth user experience (UX) flow.

5. **Business Analysts:** The SRS will be used to ensure that the business goals are properly mapped against the documented requirements and that those requirements are best communicated to the development team.

6. **QA/QC Team:** They will refer to the SRS in creating comprehensive test plans, cases, and acceptance criteria to guarantee that every feature runs flawlessly and adheres to the requirements laid out.

7. **Administrators**: The SRS helps them comprehend the system management functionality, their role, and how to oversee the platform operations in an efficient way.

8. **Moderators**: They will use this document to understand the procedure of skill verification, platform rules, and their role in the quality and integrity of the platform.

9. **End-users:** The SRS gives end-users a clear vision of the features of the platform and how they can most effectively use the system to teach or learn skills.

1.4 Product Scope

SkillSwap is like a global classroom where people can share their skills with others. It lets you learn specific parts of a skill from someone who already knows it, and also helps you teach your own skills in return. The platform provides roadmaps, matchmaking, and secure collaboration tools to make learning simple, trustworthy, and interactive.

1.4.1 Purpose

The purpose of SkillSwap is to provide a secure and interactive platform where people can exchange skills directly, focusing on flexible micro-learning, roadmap-based progress, and trustworthy collaboration through verification, reviews, and smart matchmaking.

1.4.2 Benefits and Objectives

The purpose of SkillSwap is to provide a secure and interactive platform where people can exchange skills directly, focusing on flexible micro-learning, roadmap-based progress, and trustworthy collaboration through verification, reviews, and smart matchmaking.

1. **Affordable learning**: Exchange skills instead of paying for costly courses.
2. **Micro-learning flexibility**: Focus only on the parts of a skill you need.
3. **Roadmap-based progression**: Learn complex skills step by step through structured sub-skills.
4. **Personalized experience**: Build profiles, showcase verified skills, and find the right partners.
5. **Trust and credibility** : Ensure reliability with verification, reviews, and ratings.
6. **Seamless collaboration**: Use messaging, scheduling, and file/resource sharing.
7. **Smart matchmaking**: Connect learners and teachers by skill, availability, and goals.
8. **Secure and scalable system**: protect accounts and support global users.
9. **Admin support**: Manage community, analytics, and skill catalog moderation.

1.5 Risk Definition

While SkillSwap aims to create a smooth, secure, and engaging platform for mutual learning, several risks must be considered to reflect real-life usage scenarios. Each of these risks will be paired with mitigation plans such as regular monitoring, verification badges, and proper structure of the database to ensure reliability.

1. **Technical Risks**
   (a) **Server Downtime:** If the server goes down during peak hours, learners and trainers may lose access to ongoing sessions, chats, or uploaded resources. This could break trust and discourage consistent use.

(b) **Data Loss or Corruption:** If database backups fail or data becomes corrupted, users might lose resumes, certificates, or uploaded resources, which are vital for credibility and smooth skill swaps.

2. **Security Risks:**

   (a) **Malicious File Uploads:** Since users can upload files (resumes, study resources, assignments), there is a risk of harmful files (e.g., malware, corrupted PDFs) being uploaded, which could compromise user systems.

   (b) **Data Privacy Breach:** Unauthorized access to personal details (emails, resumes, certificates) could damage user trust and cause legal implications.

3. **User Risks:**

   (a) **Fake Credentials or Misrepresentation:** Users might upload false certificates or exaggerate their expertise, which could lead to poor learning experiences or disputes.

   (b) **Inappropriate Behavior:** Misuse of chat, resource sharing, or reporting tools may create a hostile environment, requiring strong moderation policies.

4. **Operational Risks:**

   (a) **Scalability Issues:** If the platform grows faster than expected (e.g., hundreds of users joining within weeks), the system may face performance lags, slow file uploads, or delayed notifications.

   (b) **Adoption Gap:** Some users may sign up but remain inactive, leading to fewer real swaps and making the platform appear less engaging to newcomers.

5. **External Risks**

   (a) **Email Delivery Failures:** Notifications might land in spam folders or fail to deliver, leaving users unaware of deadlines or swap requests.

   (b) **Legal/Policy Changes:** New data protection or digital learning laws (e.g., GDPR compliance) could force major changes in how user data and resources are handled.

Chapter 2

Overall Description

SkillSwap is built using Node.js for the backend, React, HTML, CSS for the frontend, and PostgreSQL for the database. It will run smoothly on modern browsers, both desktop and mobile, and will be hosted on cloud services to ensure stability.

The main classes of users are:

1. Learners
2. Trainers
3. Dual-role participants
4. Admins

We assume that users have basic digital literacy and internet access. For now, the platform will support only English, though expansion is possible in the future.

2.1 User Classes and Characteristics

In the SkillSwap platform, user classes are defined to address the specific roles and requirements of learners, skill providers, moderators, and administrators, ensuring a tailored experience for each group.

2.1.1 User Class: Learners

Characteristics:

1. Learners actively explore the platform to identify and select skills they wish to acquire.
2. They engage in the matching process, evaluating potential skill providers based on profiles and ratings.
3. Learners initiate interactions by sending swap requests to skill providers, commencing the skill exchange process.
4. They communicate to discuss session details, clarify skill requirements, and coordinate learning schedules.
5. They maintain their accounts, ensuring profiles are updated with accurate personal and skill-related information.

2.1.2 User Class: Skill Providers

Characteristics:

1. Skill providers actively list their expertise on the platform, providing detailed descriptions, proficiency levels, and supporting evidence.
2. They manage the availability and status of their offered skills, indicating readiness for swap sessions.
3. Skill providers engage in communication with potential learners, responding to inquiries, negotiating terms, and scheduling sessions.
4. Upon agreement, they confirm swap requests, initiating the learning period.
5. They ensure their skills are adequately demonstrated and supported (e.g., through verified credentials or resources) during exchanges.
6. They adhere to platform processes for completing sessions, including providing feedback and managing account updates.
7. Skill providers maintain their user accounts, keeping profiles current with evolving expertise.

2.1.3 User Class: Moderators

Characteristics:

1. Moderators possess comprehensive access to verification tools, enabling them to assess and validate skill submissions.
2. They manage the skill verification process, ensuring compliance with quality standards and accuracy of claims.
3. They address and resolve issues reported during verification, maintaining a consistent user experience.
4. Moderators enforce platform guidelines, ensuring submissions meet ethical and proficiency criteria.
5. They implement security protocols to protect the integrity of the verification process and user data.
6. Moderators serve as a key point of contact for escalation, facilitating communication between users and administrators.

2.1.4 User Class: Administrators

Characteristics:

1. Administrators have full control over system management, enabling oversight of all platform operations.

2. They manage user accounts, ensuring adherence to policies, security standards, and data accuracy.
3. They handle and resolve escalated issues reported by users or moderators, ensuring operational continuity.
4. Administrators enforce platform regulations, maintaining a secure and ethical environment.
5. They implement and oversee security measures to protect user information, transaction processes, and system stability.
6. Administrators act as the central communication nexus, coordinating updates and support across the platform.

2.2 User Needs

This section of the Software Requirements Specification (SRS) delineates the specific requirements and expectations of the end-users within the SkillSwap platform, tailored to the distinct roles of learners, skill providers, moderators, and administrators.

2.2.1 Learners

1. **Profile Management:** Learners require an intuitive interface to create and maintain profiles with essential data (e.g., personal details, skills sought, location), facilitating effective matching with skill providers.
2. **Matching and Communication:** Learners need a robust matching algorithm and communication tools (e.g., messaging, notifications) to identify and engage with suitable skill providers for session coordination.
3. **Feedback Submission:** Learners necessitate the ability to submit evaluations and feedback on learning experiences, enhancing the accuracy of future matches and fostering community trust.
4. **Device Compatibility:** The system interface must be responsive across various devices, ensuring accessibility for learners on mobile or desktop platforms.

2.2.2 Skill Providers

1. **Skill Listing and Verification:** Skill providers require a streamlined process to list their expertise with detailed descriptions and submit evidence for verification, ensuring professional credibility.
2. **Session Management:** Providers need tools to manage session availability, accept or decline swap requests, and track ongoing exchanges effectively.
3. **Interaction Capabilities:** A dedicated communication channel is essential, enabling providers to respond to inquiries, negotiate terms, and coordinate sessions with learners.
4. **Device Compatibility:** The system interface must be responsive to skill providers' devices, supporting both mobile and desktop usage for profile and session management.

2.2.3 Moderators

1. **Verification Management:** Moderators require a centralized dashboard to review skill evidence, provide detailed feedback, and manage verification workflows to maintain platform quality.

2. **Issue Resolution:** Moderators need tools to address reported issues during verification and escalate complex cases, ensuring a consistent user experience.

3. **Performance Monitoring:** Moderators necessitate periodic reports on verification accuracy to support continuous improvement and operational efficiency.

4. **Device Compatibility:** The system interface must be optimized for desktop use, accommodating the detailed tasks performed by moderators.

2.2.4 Administrators

1. **System Oversight:** Administrators require comprehensive controls to monitor user activities, manage escalations, and maintain platform operations.

2. **Security Implementation:** Administrators need robust security features to protect user data, ensure transaction integrity, and manage authentication processes.

3. **Analytics and Updates:** Administrators necessitate access to usage analytics and the capability to disseminate critical system updates to all users.

4. **Device Compatibility:** The system interface must support secure desktop access, tailored to the administrative functions performed.

2.2.5 General User Needs

1. **User Authentication and Authorization:** A secure and efficient authentication system is imperative, enabling users to access and manage their accounts with confidence.

2. **Accessibility and Usability:** The platform must be accessible to users with varying technical proficiencies, providing an intuitive interface to enhance the overall user experience.

3. **Notifications:** Users require timely notifications via email or in-app alerts for swap requests, session updates, verification statuses, and other pertinent information to remain engaged.

2.3 Operating Environment

The SkillSwap platform will be deployed on a web-based architecture supported by scalable cloud infrastructure. The hardware requirements are as follows:

2.3.1 Server Side

- **Linux-based Servers:**
  1. Ubuntu Server 20.04 LTS / 22.04 LTS
  2. CentOS 8 / Red Hat Enterprise Linux 9
- **Windows-based Servers:**
  1. Windows Server 2019
  2. Windows Server 2022
- **Containerization support:**
  1. Docker and Kubernetes (for scalable deployment)

2.3.2 Client Side (End Users)

- Windows 10 / 11
- macOS 11 (Big Sur) or later
- Linux Desktop (Ubuntu 20.04+, Fedora 35+)
- Android 10+
- iOS 14+

2.3.3 Software Components and Applications

- Google Chrome (latest stable version)
- Mozilla Firefox (latest stable version)
- Microsoft Edge (latest stable version)
- Safari (latest stable version)

2.4 Constraints

2.4.1 Technical Constraints

- The system must be web-based and mobile-friendly to support multiple device types.
- PostgreSQL will be used as the primary database due to its scalability and advanced querying capabilities.
- The application must ensure real-time data synchronization for matching, availability, and notifications.
- Authentication must integrate with third-party providers (e.g., Google, Facebook) for ease of access.
- The platform must comply with common web security practices (e.g., HTTPS, encrypted passwords, secure APIs).

2.4.2 Data Constraints

- User personal data (profile details, ratings, reviews) must be stored securely and comply with data protection regulations.
- Search and filter queries must be optimized to handle large datasets without performance degradation.
- Only supported languages will be available for communication (to be expanded gradually).
- Transaction records of point trading must be immutable to prevent disputes.

2.4.3 Integration Constraints

- The system must integrate with external APIs for authentication, payment/point top-up, and notifications (email/push).
- Compatibility with multiple browsers (Chrome, Firefox, Edge, Safari) must be ensured.
- Limited dependency on third-party services to reduce vendor lock-in.

2.4.4 Time Constraints

- Initial MVP must be delivered within the semester timeframe for evaluation.
- Features may need to be prioritized, with non-essential modules deferred to later iterations.

2.4.5 Budget Constraints

- Development will rely on free or community versions of tools, frameworks, and hosting (where possible).
- Paid services such as cloud hosting, SMS/email notifications, or premium APIs may only be adopted if sponsors or grants are available.

2.4.6 Regulatory and Compliance Constraints

- The system must comply with relevant data privacy laws (e.g., GDPR principles for international users).
- User consent must be explicitly collected before storing personal or sensitive data.
- Copyright and licensing terms must be respected for any third-party integrations.

2.4.7 Resource Constraints

- Development is limited to a team of 6 members with predefined roles.
- Team members are students; hence, working hours are limited by academic workload.
- Limited access to high-performance servers during initial deployment; may rely on shared hosting solutions initially.

2.5 Assumptions

The Software Requirement Specification relies on the expectation that users will actively participate in platform activities, such as listing skills, initiating skill swaps, and managing their accounts. This assumption forms the basis for the seamless borrowing and swapping processes envisioned for the Skill Swap System.

- **User Proficiency:** It is assumed that users (both Learners and Skill Providers) have a basic understanding of online platforms and digital literacy, allowing them to navigate and utilize the system effectively.
- **Consistent Connectivity:** It is assumed that users, both learner and mentor, have consistent and reliable internet connectivity for seamless interaction with the system and with each other.
- **Language Proficiency:** It is assumed that users selecting a communication language are sufficiently fluent to conduct meaningful sessions.
- **Administrator Authority :**It is assumed that administrators have the necessary skills and authority to manage and monitor all aspects properly to ensure a secure and compliant user base.
- **Fair Trade Participation:**It is assumed that users will provide accurate availability schedules and respect agreed-upon session timings. The users will also trade points fairly and not attempt to exploit or manipulate the system.
- **Review Honesty:** It is assumed that users will provide ratings and reviews honestly and in good faith.
- **Aligned Objectives:** It is assumed that the information provided in the SRS accurately aligns with the objectives and expectations of developers, testers, project managers, stakeholders, and users.

Chapter 3

Requirements

This section outlines the functional and non-functional requirements of the SkillSwap system.

3.1 Functional Requirements

The system must provide the following functionalities:

3.1.1 Session Conduction:

Session Scheduling:

As a user, I want to arrange a meeting with a matched peer, so that we can schedule our skill-sharing session.

1. Success:
    - After a match is confirmed, users can access an *"Arrange Meeting"* interface from their chat or dashboard. The interface allows proposing a date, time, and duration for the session.
    - If *"Virtual"* is selected, the system provides an option to generate and attach a meeting link (e.g., Zoom, Google Meet).,
    - Once both users mutually agree on the details, the session is confirmed and added to their respective calendars within the platform.
    - Both users receive a confirmation notification and a scheduled reminder before the session.,
2. Failure:
    - If a proposed time is rejected by the other user, the system prompts the initiator to propose a new time.
    - If the system fails to generate a virtual meeting link, it will display an error: *"Could not generate meeting link. Please try again or add one manually."*
    - If any other system error occurs, it will display a message: "Something error occurred. *Error code: [Server error/network error].*"

Post-Session Management and Feedback:

As a user, I want to complete a session and provide feedback, so that I can track my progress and help others by rating my experience.

- **Success**
  - After a session's scheduled end time, users are prompted to mark the session as "Finished".
  - Upon marking it as finished, the user is presented with an interface to:
    1. Add private notes about the session for personal reference.
    2. Provide a public rating and written feedback for the user who taught the skill.
    3. Provide private feedback on the overall session experience to the platform. The feedback is successfully submitted and associated with the other user's profile and the session record.
- **Failure**
  - If the feedback form fails to submit, the system will display a message: "Failed to submit feedback. Please try again."

Session History Viewing and Modification:

As a user, I want to view my session history and my upcoming session, so that I can track all my learning activities and change schedules if I need to.

- **Success:**
  - Users can navigate to a *"My Sessions"* or *"History"* section in their profile. This section displays two distinct lists: *"Upcoming Sessions"* and *"Past Sessions."* Each entry shows key details: the skill, the other user's name, the date, and the time. From the "Upcoming Sessions" list or the chat, either user can initiate a "Cancel" or "Reschedule" request.
  - If a reschedule is requested, the user can propose a new time and date and other user receives a notification.
  - If both agree, the session is rescheduled or removed from both user's calendars.
- **Failure:**
  - The system cannot retrieve the session history, it will display a message: *"Could not load session history. Please refresh the page."*
  - If the other user denies the request, the session remains unchanged, and the initiator is notified.
  - If the system fails to process the request, it will display a message: *"Could not process your request. Please try again."*

In-Session Issue Reporting:

As a user, I want to report an issue during a session, so that the platform can intervene in case of problems.

- **Success:**
  - During a session's active time window, a *"Report Issue"* button is available on the session details page. Clicking the button opens a form where the user can describe the problem. The report is submitted to the admin/moderator team for review. The user receives a confirmation that their report has been received.
- **Failure:**
  - If the report fails to submit, the system displays: *"Failed to submit the report. Please check your connection and try again."*

3.1.2 Admin Profile:

Admin Dashborad:

As an admin, I want to view a dashboard with key metrics, so that I can get a quick overview of the platform.

- **Success:**
  - Upon logging in, the admin is directed to a central dashboard.
  - The dashboard displays key metrics: user growth, number of active disputes, and trending skills.
  - A persistent sidebar provides navigation to other admin functions: "Manage Moderators," *"Generate Reports," "Skill Moderation,"* and *"Create Announcement."*
- **Failure:**
  - If the analytics data cannot be loaded, the respective widgets will show an error: *"Could not load data."*
  - If any other system error occurs, it will display a message: "Something error occurred. Error code: *[Server error/network error]"*.

Moderator Management:

As an admin, I want to create, manage, and override moderators, so that I can delegate tasks while maintaining ultimate control.

- **Success:**
  - From the *"Manage Moderators"* section, an admin can create new moderator accounts. The admin can assign specific permissions to moderators (e.g., can only moderate skills, can handle disputes).
  - The admin can view a log of all actions taken by moderators.

  - The admin can select any moderator's decision (e.g., a skill approval) and override it (e.g., reject the skill).

- **Failure:**
  - If the system fails to create a new moderator, it displays an error: *"Could not create moderator account."*
  - If an override action fails, it displays: *"Failed to override moderator action."*

Report Generation:

As an admin, I want to generate and export reports, so that I can analyze platform data for decision-making.

- **Success:**
  - In the "Generate Report" section, the admin can select report types.
  - The admin can specify a date range for the report.
  - The system generates the report and provides options to export it as a PDF file or view it in the platform.
- **Failure:**
  - If report generation fails, the system will show a message: *"Error generating report. Please try again."* If the export fails, it will show: *"Export failed."*

Community Announcements:

As an admin, I want to send announcements to the community, so that I can give important updates or information regarding the platform.

- **Success:**
  - In the "Create Announcement" section, the admin can compose a message.
  - The admin can choose the audience: "All Users" or specific user groups.
  - The announcement is successfully dispatched as a platform notification to the audience.
  - The "Skill Moderation" section displays a list of all skills on the platform (pending, approved, rejected).
  - The admin can directly edit a skill's name, category, or description.
  - Admin and Moderators can approve pending skills or remove existing skills from the catalog.
- **Failure:**
  - If an edit or removal action fails, the system will display a message: *"Action failed. The skill could not be modified/removed."*

3.1.3 Cross Matching:

As a user, I want to be matched with people who can learn from me and teach me, so that I can get the best possible learning experience.

Success:

1. The system identifies users with overlapping teach/learn skills.,
2. The system validates that both users share at least one common language.,
3. If multiple candidates exist, the system prioritizes matches by relevance (rating, proximity, availability).,
4. Both users receive notifications about the match and must confirm before the session begins.,
5. The system allows skill-trade matching using trading points if exact reciprocal interest does not exist.,

Failure:

1. If no suitable matches exist, the system suggests alternatives or displays a message without breaking the experience.,
2. If language mismatch occurs, the system informs the user and suggests alternatives.,
3. If skill levels do not align, the system avoids unsuitable matches and suggests better alternatives.,
4. If the user profile is incomplete, the system prevents sending match requests until profile completion.,
5. If the user is not logged in, the system redirects them to log in or create an account before sending match requests.,

3.1.4 Skill Category, Search, and Filters:

As a user, I want to search for people teaching a skill and filter results by rating, availability, language, session time, category, trading points, and skills, so that I can find the most suitable person.

Success:

- The system displays a list of users teaching the searched skill.,
- Filters refine search results according to criteria (rating, availability, etc.).,
- The system suggests auto-complete options for partial keywords.,
- If filters are removed, the system restores the full set of matching results.,
- Search results are ranked by relevance (rating, availability, or proximity).,

Failure:

- If no users match, the system shows a friendly message and suggests alternatives (e.g., related skills or upcoming sessions).,
- If the search query is invalid or empty, the system prompts the user to re-enter valid keywords.,

3.1.5 Skill Verification:

As a skill provider, I want my skill verification request to first be reviewed by moderators from related fields, and if no suitable moderator is available, I want my skill to remain visible with an "Unverified" label until an admin assigns an appropriate moderator, so that my profile remains transparent while waiting for verification.

Success:

- The user submits a new skill (e.g., Kotlin Programming) for verification.
- The system automatically searches for moderators from related fields (e.g., Java Programming moderators).
- The verification request is assigned to the most relevant available moderator.
- If a moderator accepts, they review and verify the submitted skill.
- Once verification is complete, the skill status changes from "Unverified" to "Verified." The user receives a confirmation notification after successful verification.
- The user receives a confirmation notification after successful verification.

Failure:

- If no suitable moderator is available, the skill remains visible with the label *"Unverified – Awaiting Moderator."*
- The system automatically notifies admins to assign or onboard a qualified moderator.
- If the moderator rejects the request, the system notifies the user with reasons for rejection.
- If the skill submission data is incomplete, the system prompts the user to provide the missing information.
- If the system experiences an assignment error, it logs the issue and informs admins for manual review.

3.1.6 Review & Feedback:

- As a learner, I want to review the person who taught me a skill after each session, so that their teaching and communication abilities are fairly evaluated.
- As a teacher, I want to rate the learner based on attentiveness and engagement during the session, so that future skill providers know how cooperative the learner is.

- As a platform user, I want all reviews to be authentic and fair, so that trust is maintained, and I can appeal if I feel a review is unjust.

Success:

- After completing a session, the learner is prompted to submit a review for the teacher.
- The learner rates the teacher on teaching ability, communication, and subject knowledge, with an optional written comment.
- Once submitted, the review is saved and becomes visible on the teacher's public profile.
- The teacher is also prompted to review the learner, rating their attentiveness, willingness to learn, and overall engagement.
- The teacher's review is stored in the system and becomes visible to other potential teachers who may interact with that learner.
- The system ensures both reviews are timestamped, linked to the completed session, and contribute to the user's overall rating.
- All reviews are automatically checked for inappropriate language or spam before being published.
- If a user appeals a review, the system forwards it to a moderator for evaluation.
- The moderator can approve, edit, or remove the review based on the platform's fairness and authenticity policies.
- After moderator action, both parties are notified of the final decision.

Failure:

- If either party exits before submitting a review, the system sends a reminder notification.
- If a review contains offensive or prohibited content, it is automatically flagged and sent for moderation before posting.
- If the review submission fails due to a network or server issue, the system temporarily stores it and retries automatically.
- If a user attempts to review without completing a session, the system prevents submission and displays an error message.
- If the appeal process is misused (e.g., repeated false appeals), the system limits future appeal submissions for that user.

3.1.7 User Profile

User Dashboard:

As a new user, I want to create a profile with basic information (name, photo, bio, education details, uploading resume, skills with proficiency levels, language proficiency, and location), so that I can attract matches from users seeking those exact abilities and get potential swap partners.
The system shall allow new users to create a profile by inputting and uploading the specified basic information. Upon successful creation, the profile shall be stored in the database and become visible for matching purposes, enabling other users to view and initiate swaps based on the provided details.

- **Success:**
  - Given a new user is signing up for the platform, when the user enters all required fields correctly and submits the form, then the profile is created successfully, and the user is redirected to the dashboard with a confirmation message.
- **Failure:**
  - If the upload fails due to file size limits or invalid input formats, then the system shall display an error message prompting the user to correct the input or use a smaller file, preventing submission until resolved.

External Profile Integration:

As a registered user, I want to link my social media or external profiles (e.g., LinkedIn, GitHub), so that I can enhance my credibility.
The system shall enable registered users to add links to social media or external profiles. Upon successful integration, the links shall be displayed on the user's profile page, allowing other users to access them for verification of expertise.

- **Success:**
  - Given a registered user aims to boost credibility, when the user enters valid URLs and submits, then the links are added to the profile, and a success notification is shown.
- **Failure:**
  - If the link fails due to an invalid URL, then the system shall prompt the user to enter a valid link, rejecting the submission until the URL is corrected.

Availability Scheduling

As a registered user, I want to link my social media or external profiles (e.g., LinkedIn, GitHub), so that I can enhance my credibility.
The system shall enable registered users to add links to social media or external profiles. Upon successful integration, the links shall be displayed on the user's profile page, allowing other users to access them for verification of expertise.

- **Success:**
  - Given a registered user needs efficient scheduling, when the user selects and saves free slots for specific days and times, then the calendar is updated and visible to other users.
- **Failure:**
  - If the calendar update fails due to system errors or conflicting inputs, then the user shall receive a notification to retry, with the previous state preserved to avoid data loss.

Badge Display

As a registered user, I want to display badges for completed swaps, so that I can build trust with other users.
The system shall automatically update and display badges on registered users' profiles upon completion of swap sessions. The badges shall indicate metrics such as the number of completed swaps, enhancing profile visibility.

- **Success:**
  - Given a registered user has completed successful swap sessions, when the system processes the completion, then the profile is updated to display the badge, attracting interest from potential partners.
- **Failure:**
  - If the badge update fails due to database issues, then the user shall be notified to contact support, with the system logging the error for administrative review.

Profile Editing:

As a registered user, I want to edit my profile details (e.g., add skills, username, bio, skill proficiency levels), so that I can keep my information current as my interests evolve.
The system shall allow registered users to modify profile details through an editing interface. Upon successful edit, the changes shall be reflected immediately, improving matching accuracy.

- **Success:**
  - Given a registered user's interests have evolved after acquiring new expertise, when the user updates and saves the details, then the profile is revised successfully, with a confirmation displayed.
- **Failure:**
  - If the edit is incomplete due to missing required fields, then the system shall prompt the user to fill all required fields, preventing save until completion.

Skill Categorization:

As a new or registered user, I want to pick a skill category to organize my abilities, so that my skills are easily categorized for matching.
The system shall enable users to select from predefined skill categories during profile creation or editing. Upon successful assignment, the categories shall organize skills for efficient system matching.

- **Success:**
  - Given a new user during signup or a registered user during editing, when the user selects and assigns categories to skills, then the skills are categorized and ready for matching.
- **Failure:**
  - If a selected category is unavailable or invalid, then the system shall suggest the closest alternative, guiding the user to complete the assignment.

3.1.8 Messaging

Message Initiation:

As a potential swap partner, I want to send a direct message to another user, so that I can initiate a conversation about a specific skill match.
The system shall allow users to compose and send direct messages to other users. Upon successful delivery, the message shall appear in the recipient's inbox, starting the conversation.

- **Success:**
  - Given a user wants to learn a skill and has found a suitable profile, when the user composes and sends the message, then the conversation is initiated successfully.
- **Failure:**
  - If the message fails to send due to network issues, then the system shall display a retry option, preserving the drafted message.

Threaded Communication:

As a user, I want to receive and reply to messages in a threaded chat, so that I can keep track of ongoing discussions about session details.
The system shall support threaded chats for receiving and replying to messages. Upon successful reply, the thread shall update to include the new message, maintaining discussion continuity.

- **Success:**
  - Given a user has started a conversation, when the recipient replies and the sender responds, then the threaded chat tracks the discussion effectively.
- **Failure:**
  - If the thread loads incorrectly due to caching issues, then the system shall prompt the user to refresh the chat, reloading the content.

Real Time Notifications:

As a busy user, I want real-time notifications for new messages, so that I can respond quickly.
The system shall deliver real-time notifications for incoming messages. Upon successful notification, the user shall be alerted promptly, facilitating quick responses.

-
- **Success:**
  - Given a busy user is engaged in a conversation, when another user sends a message, then the notification is delivered in real-time, allowing immediate response.
- **Failure:**
  - If notifications are disabled by user settings, then the alert shall be withheld until the user logs in, with messages queued in the inbox.

Message Tagging:

As a user, I want to star important messages, so that I can quickly access key conversations or agreements about skill swaps.
The system shall enable users to tag (star) messages for easy reference. Upon successful tagging, the message shall be added to a starred section for quick access.

- **Success:**
  - Given a user wants to access key details, when the user stars a received message, then it is tagged and accessible quickly.
- **Failure:**
  - If the star feature malfunctions due to system errors, then the user shall see an error message and be able to retry the action.

Message Management:

As a user, I want to delete irrelevant or outdated message threads, so that I can declutter my inbox and focus on active skill swap discussions.
The system shall allow users to delete selected message threads. Upon successful deletion, the inbox shall be updated to reflect the removal, improving organization.

- **Success:**
  - Given a user has an old, irrelevant chat, when the user selects and deletes the thread, then the inbox is decluttered, focusing on active discussions.
- **Failure:**
  - If deletion fails due to permissions or errors, then the user shall be notified to try again, with the thread remaining intact.

User Safety Control:

As a user, I want to report or block another user from messaging me, so that I can protect myself from inappropriate or spam interactions.
The system shall provide reporting and blocking options for users. Upon successful report or block, the offending user shall be restricted, and administrators notified if required.

- Success:
  - Given a user encounters spam, when the user reports and blocks the sender, then the interaction is halted, ensuring safety.
- **Failure:**
  - If the report fails due to submission errors, then the system shall offer an option to submit manually to support, logging the attempt.

3.1.9 Email Notifications:

As a user

- The system must send an email notification to the user's registered email address for the following events:
  1. Password recovery
  2. Skill verification approval or rejection
  3. New skill swap request
  4. Session link generation
  5. Platform updates (e.g., terms & conditions)
- The email should include relevant dynamic content based on the event, such as:
  1. Requester's name, requested skill, and link to respond (for swap requests)
  2. Link to view shared resource (for received resources)
  3. Assignment/task details and due date (for deadline reminders)
  4. Verification status and link to view verified certificates (for certificate verification)
  5. Summary of updates and link to the full terms (for platform updates)
- The system must allow users to:
  1. Resend verification emails if not received within 5 minutes.
  2. Resend password reset emails if not received within 5 minutes.
  3. Update their registered email address if it is incorrect.
- For undelivered emails, the system must display a help tip suggesting users to check their spam/junk folders.
- When a certificate verification or session link is successfully processed, an automatic email should be triggered with the relevant details and a direct link to view or join.
- For task deadlines, the system must automatically schedule and send reminder emails 24 hours before the due date.

Failure Scenarios

- If an email fails to send due to a network or SMTP error, the system must:
    1. Retry sending up to 3 times automatically.
    2. Log the failure reason.
    3. Display a message: "We couldn't send your email right now. Please try again later."
- If the verification or password reset email is still not received after resending, the system must also:
    1. Provide an emergency contact option for user support.
- If an email address is invalid or not verified, the system must display: *"Invalid or unverified email address. Please update your email to continue."*
- If spam filters block the email, the system should still send it.

3.1.10 File Sharing:

User Story

As a new user, I can upload my resume, certificates, or any sort of proof of expertise to my profile, so that other users can learn about my background and skills before initiating a swap.
As a participant in a skill swap, I want to upload study resources, assignments, and add deadlines, so that my partner can access the materials on time and stay on track during the learning process.
As a user chatting with my skill partner, I want to quickly send files through the chat window, so that I can share references, examples, or quick notes without leaving the conversation (with limited file types for security purposes).
As a user and participant in a skill swap, I want the ability to upload supporting documents when reporting issues or defending myself in a dispute, so that I can provide proof and ensure fair resolution.
As a participant in a skill swap, I want to attach files while giving reviews and feedback, so that others can clearly understand my learning experience and benefit from additional context or evidence.

Success Scenarios

1. In the Profile section, users can upload files such as resumes, certificates, or skill proofs that will be visible on their profile.
2. In an Active Skill Swap Session, users can upload resources, assignments, or materials and set valid deadlines for submissions.
3. In the Chat Window, users can upload and send files (limited to supported formats such as .pdf, .jpg, .png, .docx, .pptx, etc.) to their swap partner.

4. In the Report or Dispute section, users can upload supporting documents as evidence for issue resolution.
5. In the Feedback/Review section, users can attach files to provide additional context about their experience.
6. The system validates the uploaded file format and size before submission.
7. The system shows a progress bar or loading indicator during file upload.
8. The system stores uploaded files securely and links them to the relevant section (profile, chat, feedback, or report).
9. When users set a deadline, the system ensures the date selected is not earlier than the current date.
10. Once a file is successfully uploaded, the user sees a confirmation message such as "File uploaded successfully."

Failure Scenarios

1. If a user uploads a file exceeding the allowed size limit, the system displays:
   "File size exceeds the limit. Please try again with a smaller file."
2. If a user uploads an unsupported file type (e.g., .exe), the system displays:
   "Invalid file type. Please upload a supported format."
3. If a user tries to send an unsupported file type via chat, the system displays:
   "This file type cannot be shared in chat."
   The file will not be sent.
4. If the connection drops or a server error occurs during upload, the system displays:
   "Upload failed. Please retry."
   The file will not appear until the upload is successful.
5. If a user sets a deadline in the past, the system displays:
   "Deadline cannot be earlier than today. Please choose a valid date."

3.1.11 Roadmap:

As a user of the platform,

- **AI-Based Roadmap Generation**

  As a user of the platform, I want to view an AI-generated roadmap for a large skill (e.g., Web Development), so that I can follow a structured, step-by-step learning path.

  1. **Success:**
     - When the user selects a main skill, the AI automatically generates sub-skills such as HTML, CSS, and JS.
     - The roadmap is displayed visually in phases (e.g., Phase 1 – HTML, Phase 2 – CSS).

- Users can see their current level and progress in each phase.

2. **Failure:**
   - If the AI fails to generate a roadmap, an error message ("Unable to generate roadmap. Please try again later.") is shown.
   - If network issues occur, the user is prompted to retry.

- **Level Customization**

  As a user, I want to set my skill level (Beginner, Intermediate, or Advanced), so that the roadmap matches my learning needs.

  1. **Success:**
     - After selecting a level, the system adjusts the roadmap starting point and complexity.
     - The user is redirected to the most suitable sub-skills based on their level.
  2. **Failure:**
     - If the level is not selected, the system shows a prompt: "Please choose your current level to continue."

- **Manual Mentor Search**

  As a user, I want to manually search for teachers or mentors for each sub-skill, so that I can select the best instructor based on rating or experience.

  1. **Success:**
     - When a user clicks a sub-skill (e.g., CSS), the system displays a list of available mentors.
     - The user can sort or filter results (by rating, price, availability).
  2. **Failure:**
     - If no mentor is available, a message appears: "No mentors available for this skill right now."
     - If there's a connection error, the system prompts: "Unable to load mentor list. Please try again."

- **Progress Tracking**

  As a learner, I want to mark sub-skills as completed, so that my roadmap progress updates automatically.

  1. **Success:**
     - When a sub-skill is marked done, the progress bar updates immediately.
     - The user can view completion percentage and remaining phases.
  2. **Failure:**
     - If a progress update fails, the system notifies the user: "Progress not saved. Please refresh."

3.1.12 Login/Signup Authentication

As a user of the platform,

- **User Registration and Verification**

  As a user of the platform, I want to register using my email and provide basic information (name, email, password, and skills I want to teach/learn), So that I can create a verified account and access the platform securely.

  1. **Success:**
     - User enters valid details (name, email, password, skills) and clicks "Register."
     - System sends a verification email.
     - After successful verification, the account becomes active.
     - Optional OAuth signup (Google/social login) successfully links the user's account.
  2. **Failure:**
     - If any required field is missing, the system shows "Please fill out all required fields."
     - If the email is already registered, an alert appears: "This email already exists."
     - If verification email expires or fails, the user can request a new one.

- **Login**

  As a user of the platform, I want to log in securely using my email and password or Google/social account, So that I can access my dashboard and manage my teaching and learning activities.

  1. **Success:**
     - When valid credentials or OAuth login are provided, the system redirects the user to the homepage/dashboard.
     - If the user had previously checked "Remember Me," the system maintains the session.
     - If a specific feature requires login, unregistered users are redirected to the signup page, and then back to the login page once registration is complete.
  2. **Failure:**
     - If credentials are invalid, the system displays "Incorrect email or password."
     - If password format is invalid, the user is prompted to re-enter it correctly.
     - After five failed attempts, login is temporarily locked for security reasons.

- **Logout and Session Management**

  As a logged-in user, I want to log out from the current or all devices, So that my session ends securely and my account remains protected.

  1. **Success:**
     - When the user selects "Logout," the session token is invalidated.
     - Option to "Log out from all devices" terminates all active sessions.

- The user is redirected to the login page after logout.

2. **Failure:**
   - If session termination fails, the system displays "Unable to log out. Please try again."
   - Inactivity for 30 minutes automatically ends the session (auto-logout).

- **Forgot/Reset Password**

  As a user, I want to reset my password if I forget it, So that I can regain access to my account securely.

  1. **Success:**
     - User clicks "Forgot Password" and enters registered email.
     - A reset link with a secure token (valid for 30 minutes) is sent.
     - On clicking the link, user can set a new password.
  2. **Failure:**
     - If the email doesn't exist, an error "Email not found" is shown.
     - If the reset link expires, user must request a new one.

- **Two-Factor Authentication (Optional)**

  As a user, I want to enable two-factor authentication (2FA), So that I can add an extra layer of account protection.

  1. **Success:**
     - When enabled, system sends a one-time code (OTP) via email or authenticator app.
     - Login is successful only after entering the correct OTP.
  2. **Failure:**
     - If OTP is incorrect or expired, system shows "Invalid or expired code."

3.2 Non-Functional Requirements

The non-functional requirements (NFRs) of the SkillSwap platform define the quality standards, operational constraints, and performance expectations that the system must satisfy. These requirements ensure usability, maintainability, scalability, security, and reliability across all modules of the system.

3.2.1 Performance Requirements

- The system should be capable of supporting at least 500 concurrent users during peak hours without noticeable degradation in performance.
- Average page load time should not exceed 3 seconds on a stable internet connection (10 Mbps or higher).
- The system must maintain a response time of less than 1 second for major user interactions such as login, skill search, or messaging.

- Database queries, including search and filter operations, must be optimized to return results within 2 seconds.
- The platform should maintain at least 99.5% uptime annually, excluding scheduled maintenance periods.

3.2.2 Scalability Requirements

- The architecture must allow horizontal and vertical scaling to accommodate future user growth.
- The backend should be modular to support future integration with mobile applications or additional APIs.
- Cloud deployment (e.g., AWS, Render, or Vercel) should allow for auto-scaling during high-traffic events.
- Caching mechanisms such as Redis or in-memory caching should be implemented for frequently accessed data.

3.2.3 Security Requirements

- All communications between client and server must be encrypted using HTTPS (SSL/TLS protocols).
- Passwords must be stored using a strong hashing algorithm (e.g., bcrypt or Argon2).
- Authentication tokens (JWT) must be securely managed and invalidated upon logout or token expiry.
- The system must implement input validation and sanitization to prevent common vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF).
- User data, including personal and session information, must comply with global privacy standards such as GDPR principles.

3.2.4 Usability Requirements

- The interface must be intuitive and consistent, ensuring users can perform key actions within three clicks.
- The design should adhere to WCAG 2.1 accessibility guidelines, ensuring compatibility with screen readers and accessible color contrast ratios.
- All primary user actions must provide feedback (e.g., success, failure, or progress states).
- The platform should provide responsive design, maintaining usability on both mobile and desktop screens.
- User documentation, tooltips, and onboarding guides must be provided for first-time users.

3.2.5 Reliability and Availability Requirements

- The system should be capable of recovering from unexpected shutdowns without data loss through regular database backups.
- Backups must be taken automatically at least once every 24 hours and stored in a secure off-site location.
- Failover mechanisms must be established to ensure minimal downtime during server crashes or maintenance.
- All critical operations (login, point transactions, verification, etc.) must include fail-safe handling to prevent corruption.

3.2.6 Maintainability Requirements

- The system codebase must follow modular and component-based design principles for easy debugging and updates.
- Proper source code documentation (using tools like JSDoc) must be maintained for all major components, functions, and APIs.
- The system should support environment-based configurations (development, testing, production) using tools like dotenv.
- Version control using Git and GitHub must be strictly followed, with branches for each feature and code reviews before merging.
- Automated testing (unit, integration, and regression) should be implemented to ensure stability after updates.

3.2.7 Portability Requirements

- The application should be compatible across major browsers, including Chrome, Firefox, Edge, and Safari.
- The web version should be accessible from mobile browsers without loss of functionality.
- The backend should be containerized using Docker for consistent deployment across environments.

3.2.8 Privacy Requirements

- User consent must be explicitly obtained before collecting or storing personal information.
- Personal identifiable information (PII) must be anonymized or pseudonymized wherever possible.
- Users must be able to request data deletion or download under compliance with privacy policies.
- Session cookies should use secure and HttpOnly flags to prevent unauthorized access.

3.2.9 Supportability Requirements

- The system should provide clear and descriptive error messages to aid troubleshooting.
- Logs must be maintained for critical system events, errors, and user activities for debugging and audit purposes.
- The platform should allow administrators to monitor performance and user activity through an internal dashboard.

3.2.10 Ethical and Social Requirements

- User-generated content (e.g., feedback, messages, resources) must adhere to community guidelines to maintain a respectful environment.
- Any reported misconduct must be handled promptly by moderators or administrators.
- Users must agree to terms of service and privacy policies during registration.