A
Project Report
Entitled

"SkillSwap: Peer Skill Exchange"

A project report submitted in partial fulfillment
of the requirement for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS

RANI CHANNAMMA UNIVERSITY, BELAGAVI.

Submitted by

Tej Hagargi	**[U15EW22S0032]**
Dhanush Poojary	**[U15EW22S0052]**
Arman Shaikh	**[U15EW22S0057]**

Under the Guidance of
Prof. Akshata Pethe

COLLEGE OF BACHELOR OF COMPUTER APPLICATION GOKAK

2024 - 25

K.L.E Society's

COLLEGE OF BACHELOR OF COMPUTER APPLICATION GOKAK

Certificate

Certified that the project entitled

"SkillSwap: Peer Skill Exchange"

Is a bonafide work carried out by

Tej Hagargi	**[U15EW22S0032]**
Dhanush Poojary	**[U15EW22S0052]**
Arman Shaikh	**[U15EW22S0057]**

In partial fulfillment for the award of degree of Bachelor of Computer Application of the Rani Channamma University, Belgavi, during the year 2024-2025.It is certified that all corrections/suggestions indicated for the internal assessment have been incorporated in the report deposited in the department library. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Computer Application Degree.

GUIDE	**Coordinator**	**Principal**
Prof. Akshata Pethe	**Prof. Malikjan Bagawan**	**Prof. Prashant Kivati**

Name of Examiners Signature with Date

1.____________________

2.____________________

ACKNOWLEDGEMENT

At the outset we would like to thanks **Prof. Akshata Pethe** our guide, for their co-operation in making our project a reality. As the project supervisors of our project their guidance was inspiration for successful completion of our project. **Prof. Akshata Pethe** was with us right from conceptualizing the theory of this project to final accomplishment.

We are grateful to our **Coordinator Prof. Malikjan Bagawan**, for the active interest showed in our project.

We express our sincere thanks to our **Principal Prof. Prashant Kivati** for supporting us throughout the project.

We would like to extend our sincere gratitude to both teaching and non-teaching staff of our department for providing us the basic infrastructures used for this project.

Our sincere and humble gratitude to our parents and friends without whose support this would not have been a success in the stipulated tim*e*.

CONTENTS

. SI.no Title Page. no

1	Introduction	1
2	Literature Survey	2
3	Software Requirement Specification	3
4	Modularization Details	4-5
5	Tools & Technologies Used	5-14
6	Use Case Diagram	15
7	DFD Diagram	16
8	ER Diagram	17
9	Screen Layouts	18-27
10	Database Tables	28-33
11	System Testing	34-37
12	Feasibility Study	38-40
13	Project Planning	41-43
14	Data and Database Security	44-46
15	User Profiles and Access Rights	47-48
16	Cost Estimation of the Project	49-50
17	Conclusion	51
18	Bibliography	52

Abstract

Project describes an open architecture for the development of an Advance Digital Library. The many users of such a system, even those with only limited or even no knowledge of information technology, can benefit enormously from quick and easy access to the information it contains.

This project has many features which are generally not available in normal library management systems like facility of user login and a facility of teachers login .It also has a facility of admin login through which the admin can monitor the whole system .

It also has facility of an online notice board where teachers can put up information about workshops or seminars being held in our colleges or nearby colleges and librarian after proper verification from the concerned institution organizing the seminar can add it to the notice board .

It has also a facility where student after logging in their accounts can see list of books issued and its issue date and return date and also the students can request the librarian to add new books by filling the book request form. The librarian after logging into his account i.e. admin account can generate various reports such as student report, issue report, teacher report and book report

1. Introduction

SkillSwap: Connect, Learn, and Grow Through Skill Exchange

In today's fast-paced world, many individuals possess valuable skills but lack access to resources or people who can help them develop new ones. While there are numerous platforms for professional networking and online learning, there are very few centralized systems designed specifically to facilitate direct, one-on-one skill exchange without the need for money. **SkillSwap** aims to fill this gap by providing a dedicated platform where users can find and connect with others to exchange skills in real-time through an intuitive chat-based system.

Many people want to learn new skills—be it a language, coding, painting, cooking, or music—but are unable to afford formal courses or simply prefer a more personalized, peer-to-peer learning environment. At the same time, many others are eager to share what they know. However, without a proper system, it's hard for these individuals to find each other and coordinate exchanges effectively.

With **SkillSwap**, users can search for available skill offers and requests, view user profiles, initiate chat-based discussions, and agree on mutual sessions. This reduces the need for complex scheduling, expensive lessons, or endless searches across unrelated platforms. It also promotes a community-driven ecosystem where knowledge is accessible, valuable, and shareable.

Whether you're looking to improve your guitar skills, teach graphic design, or practice French with a fluent speaker, **SkillSwap** empowers users to learn and teach by connecting the right people at the right time—making skill sharing as easy and accessible as a conversation.

2. Literature Survey

Existing System:

- Static platforms like job boards or online course websites that don't support direct skill trading.
- Social media groups and forums where skill exchange is unorganized and hard to track.
- Paid platforms that focus only on learning, not exchanging skills.
- Lack of real-time communication and tracking of accepted requests or scheduled sessions.

Proposed System:

- Enables users to search and connect with others based on the skills they offer or want to learn.
- Supports chat-based communication for easy coordination.
- Displays accepted skill swap requests in a dedicated chat section.
- Builds a self-sustaining learning community through mutual collaboration.
- Eliminates the need for money or complex scheduling tools.
- Especially useful for students, freelancers, and hobbyists looking for flexible learning opportunities.

3. Software Requirement Specifications

1. Hardware Requirements

Monitor	LED
Processor	Intel i3 or above with a minimum speed of 2.5 GHz
Hard Disk	Minimum 10 GB of free disk space
RAM	Minimum 4 GB RAM or above
Keyboard	Standard Keyboard
Mouse	Optical Mouse

2. Software Requirements

Component	Specification
Computer OS	Windows / Linux / macOS
Languages	React.js, Tailwind, Express.js, Node.js
Database	MongoDB
Keyboard	Standard Keyboard
Web Server	Express.js Server

4. Modularization Details

3. User (Learner / Sharer):

- Can register and log in using email and password.
- Create and update profile with skills they can teach and want to learn.
- Search for users based on skill categories or names.
- View matching users based on skill interest.
- Send skill swap requests to matched users.
- Accept or reject incoming skill swap requests.
- Chat with connected users in real-time.
- Share messages, learning goals, and resources through chat.

5. Tools and Technology Used

1. Frontend Technologies

1.1 React.js

- React.js is a widely used open-source JavaScript library developed and maintained by Meta (formerly Facebook). It is used for building fast, interactive, and scalable user interfaces, particularly for single-page applications (SPAs). The core idea of React is the concept of **components**, which are modular and reusable blocks of code that manage their own state.
- React uses a **Virtual DOM** (Document Object Model) to optimize rendering. When a user interacts with the UI, React updates only the parts of the page that change, instead of reloading the entire page. This leads to better performance and a smoother user experience.
- In my project, React.js was used to:
- Build reusable components like forms, chat boxes, and request cards.
- Manage routing and navigation using libraries like react-router-dom.
- Handle state locally and across components for interactivity.

1.2 Tailwind CSS

- Tailwind CSS is a utility-first CSS framework that enables developers to design directly in their HTML (or JSX) by applying utility classes. Instead of writing custom CSS for each component, Tailwind provides low-level utility classes that can be composed to build any design, without leaving the HTML structure.
- Tailwind improves development speed and consistency, offering benefits like:
- No need to switch between CSS and JSX files.
- Predefined design system with responsive design utilities.
- Easy customization via configuration.
- In my project, Tailwind was used to style all frontend components including layout, colors, typography, spacing, and responsiveness. It significantly reduced the time required for UI design and made the application mobile-friendly with minimal effort.

1.3 Axios and Fetch

- Both **Axios** and the native **Fetch API** are tools used to make HTTP requests from the frontend to the backend server.

- **Axios** is a promise-based HTTP client for the browser and Node.js. It provides a simple API for sending asynchronous HTTP requests and handling responses or errors.
- **Fetch** is built into modern browsers and allows developers to make network requests. It returns promises but does not automatically handle errors or timeouts, which Axios does better.
- In my project:
- Axios was primarily used for API calls such as submitting login/signup forms and fetching accepted requests.
- Fetch was used in some simpler interactions or where customization wasn't required.

1.4 Lucide-react

- **Lucide-react** is an icon library consisting of clean and consistent SVG icons built for React. It is the spiritual successor of Feather Icons and is widely appreciated for its simplicity and flexibility.
- Using Lucide-react in my project helped:
- Improve visual clarity with icons for actions like messages, notifications, etc.
- Add recognizable UI elements that enhance user experience.

- Customize icon size, stroke width, and color using props.

1.5 Framer Motion

- **Framer Motion** is a production-ready animation library for React. It allows developers to implement smooth transitions and interactive animations easily. It abstracts complex animation behavior and offers simple declarative APIs to animate components on hover, tap, scroll, or mount.
- In my project:
- Framer Motion was used to animate UI components such as modals, page transitions, and buttons.
- It added a dynamic and engaging user experience.
- Helped highlight user interactions and improve the app's aesthetics.

2. Backend Technologies

2.1 Node.js

- **Node.js** is a JavaScript runtime environment built on Chrome's V8 engine. It enables JavaScript to be executed outside the browser, making it suitable for building server-side applications. Node.js is event-driven and non-blocking, which makes it lightweight and efficient, especially for real-time applications.

- In my project:
- Node.js was used to build the backend API that handles user authentication, database interactions, and request routing.
- It allowed writing the entire application in JavaScript (frontend + backend)

2.2 Express.js

- **Express.js** is a minimal and flexible Node.js web application framework that provides robust features for building web applications and APIs. It simplifies tasks such as routing, handling requests/responses, middleware setup, and error handling.
- In my project:
- Express was used to define backend routes such as /api/register, /api/login, and /api/requests.
- Middleware like body-parser and cors were added via Express.
- Helped structure backend code clearly and efficiently.

2.3 Bcrypt

- **Bcrypt** is a password-hashing function that incorporates salting to ensure stored passwords are secure. It is resistant to brute-force attacks and rainbow table attacks.
- In my project:
- Bcrypt was used to hash user passwords before storing them in the database.
- During login, the submitted password is compared with the hashed password using Bcrypt's compare function.
- This ensures user authentication is secure.

2.4 Nodemon

- **Nodemon** is a development utility that automatically restarts the Node.js server when it detects file changes in the source code. This significantly improves the development experience by removing the need to manually restart the server after every change.
- In my project:
- Nodemon was used during development for rapid testing and debugging.
- It saved time and improved productivity.

2.5 CORS (Cross-Origin Resource Sharing)

- **CORS** is a security feature implemented in web browsers that restricts web pages from making requests to a different domain than the one that served the web page.
- In my project:
- The backend server needed to accept requests from the frontend running on a different port (e.g., frontend on port 3000, backend on 5000).
- Using the cors middleware in Express allowed my backend to securely accept and respond to requests from the frontend.

3. Database

3.1 MongoDB

- **MongoDB** is a popular NoSQL database that stores data in a flexible, JSON-like format called BSON (Binary JSON). It is ideal for applications that require rapid development and need to handle large amounts of unstructured data.
- Key benefits:

- Schema-less: You can store documents with different structures in the same collection.
- Scalable: MongoDB supports horizontal scaling through sharding.
- High performance for read/write operations.
- In my project:
- MongoDB was used to store users, skill requests, and chat-related data.
- Mongoose (a MongoDB ODM) was used to define schemas and interact with the database in an organized manner.
- It enabled fast and flexible data storage and retrieval.

4. Development Tools

4.1 Visual Studio Code (VSCode)

- **VSCode** is a lightweight but powerful source code editor developed by Microsoft. It supports debugging, syntax highlighting, Git integration, extensions, and more.
- In my project:
- VSCode was used as the primary code editor.

- Extensions such as ESLint, Prettier, and Tailwind IntelliSense improved productivity.
- Git and GitHub integration helped with version control.

4.2 Postman

- **Postman** is a collaborative platform used for API development and testing. It allows developers to send requests, view responses, test endpoints, and debug issues in RESTful APIs.
- In my project:
- Postman was used to test all backend API routes.
- It helped verify authentication, database CRUD operations, and error handling.
- Simplified testing before connecting frontend and backend.

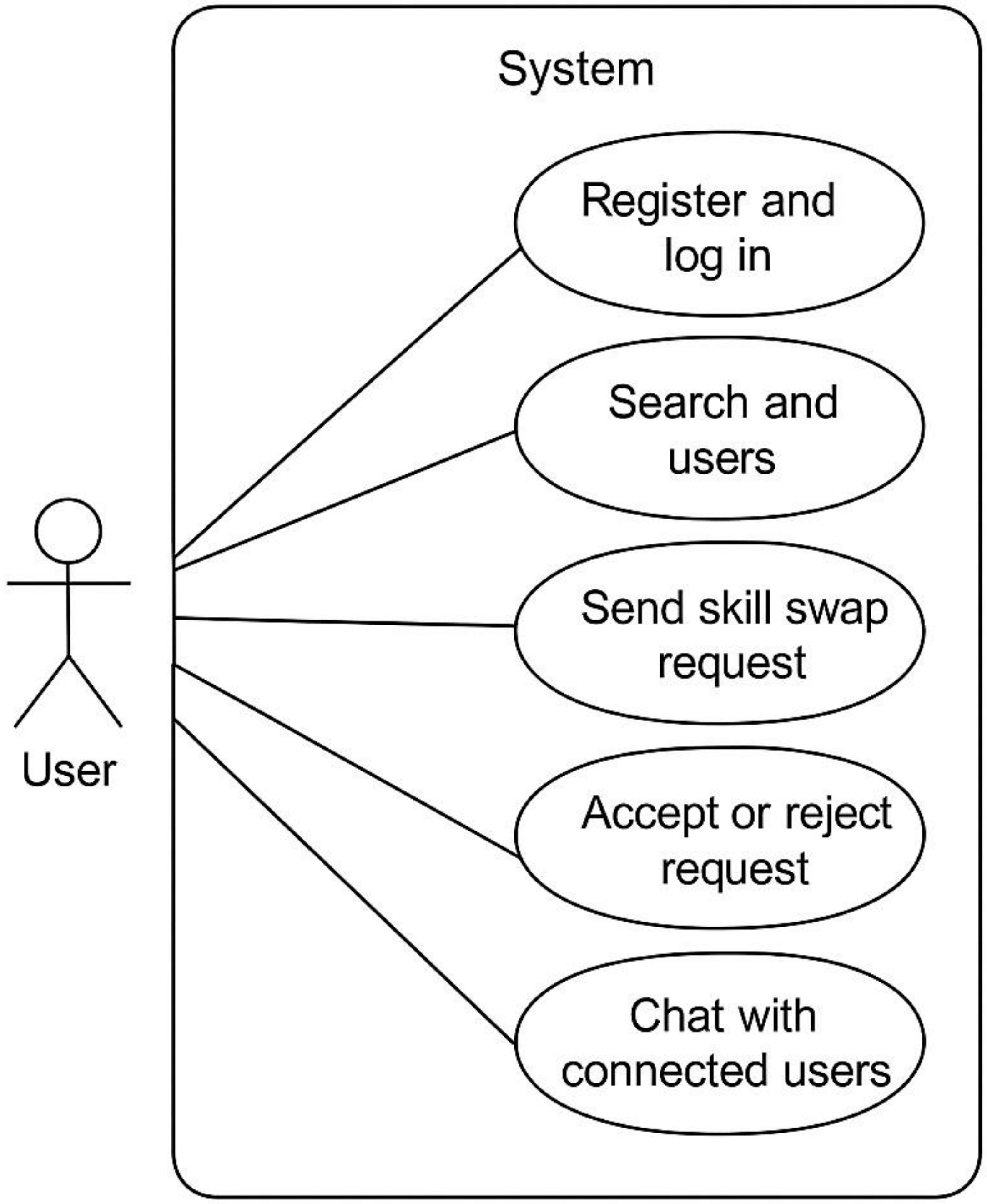
4.3 Excalidraw

- **Excalidraw** is a virtual whiteboard tool used to create hand-drawn style diagrams and sketches. It is especially useful for visualizing project structure, database schemas, or UI mockups.

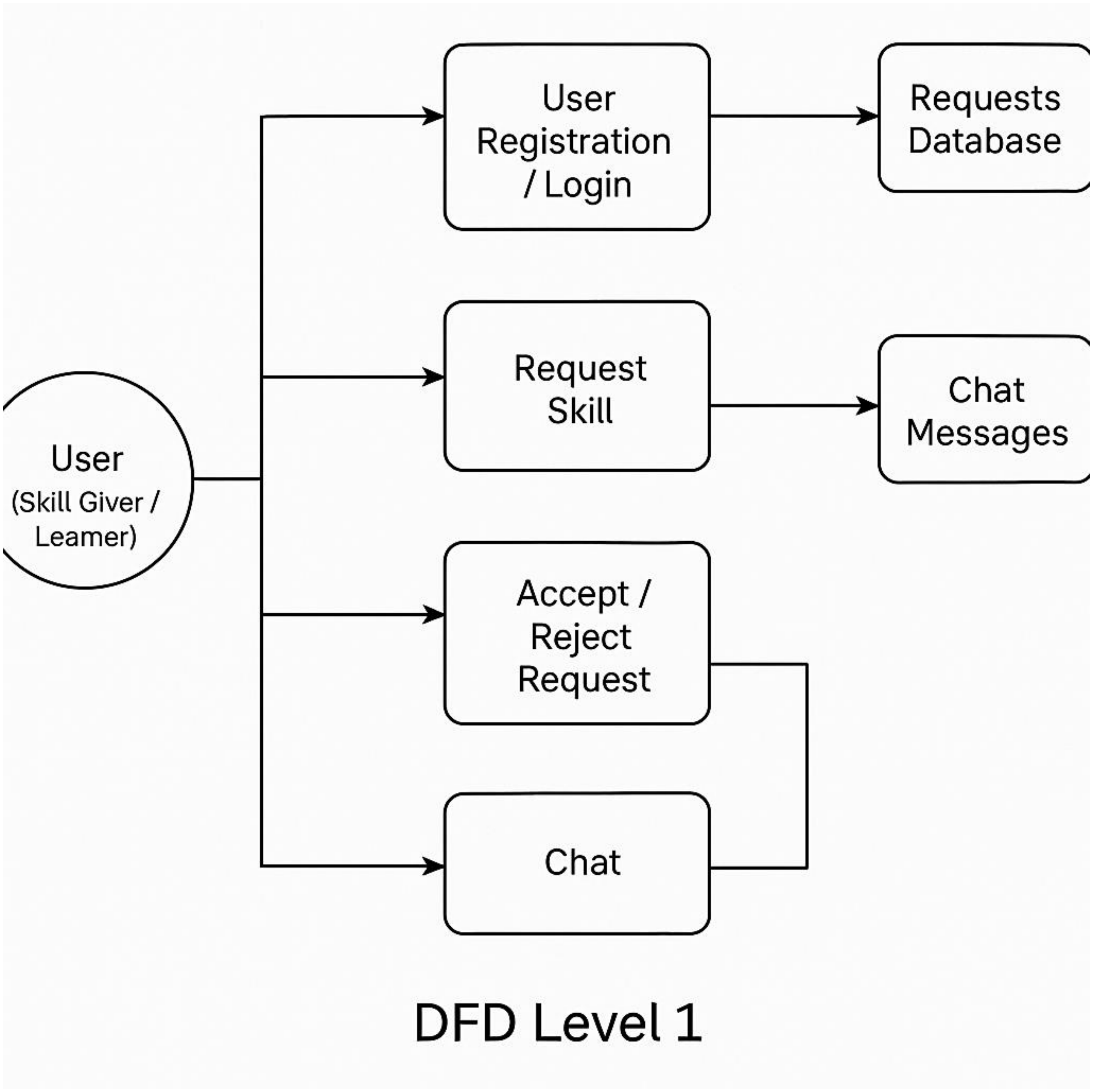
- In my project:
- Excalidraw was used to design flowcharts, entity-relationship diagrams, and frontend wireframes.
- It helped in the planning and documentation phase of the project.

- The combination of these technologies provided a robust, full-stack foundation for my project. React and Tailwind enabled a fast and modern frontend, while Node.js, Express, and MongoDB handled backend logic and data storage efficiently. Tools like VSCode, Postman, and Excalidraw supported smooth development and planning. Together, these technologies ensured that the application was scalable, secure, and user-friendly.

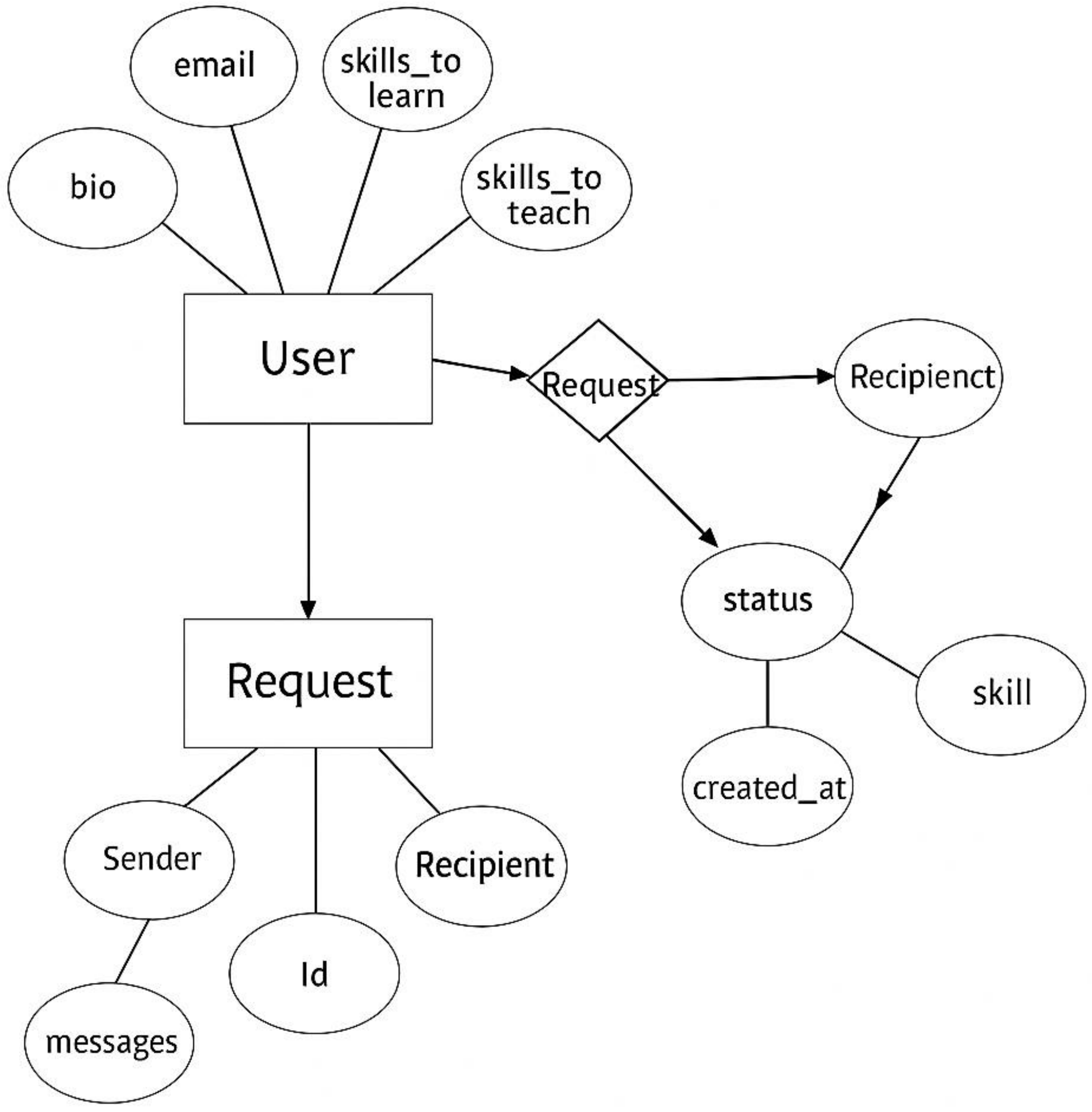
6. Use Case Diagram



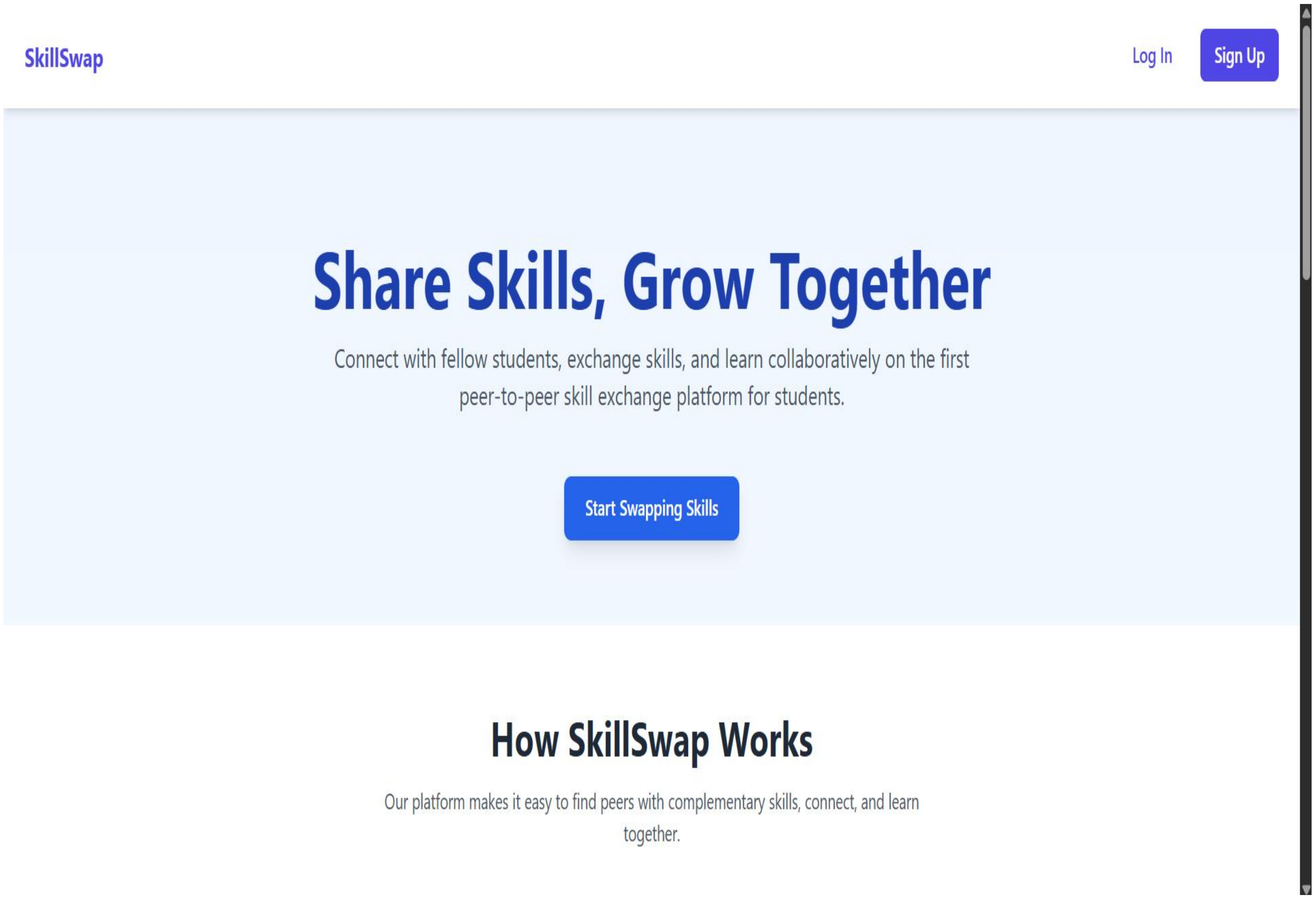
7. DFD Diagram



8. ER Diagram


email
skills_to learn
bio
skills_to teach
User
Request
Recipienct
status
skill
created_at
Request
Sender
Recipient
Id
messages

9. Screen Layouts


SkillSwap

Log In

Sign Up

Share Skills, Grow Together

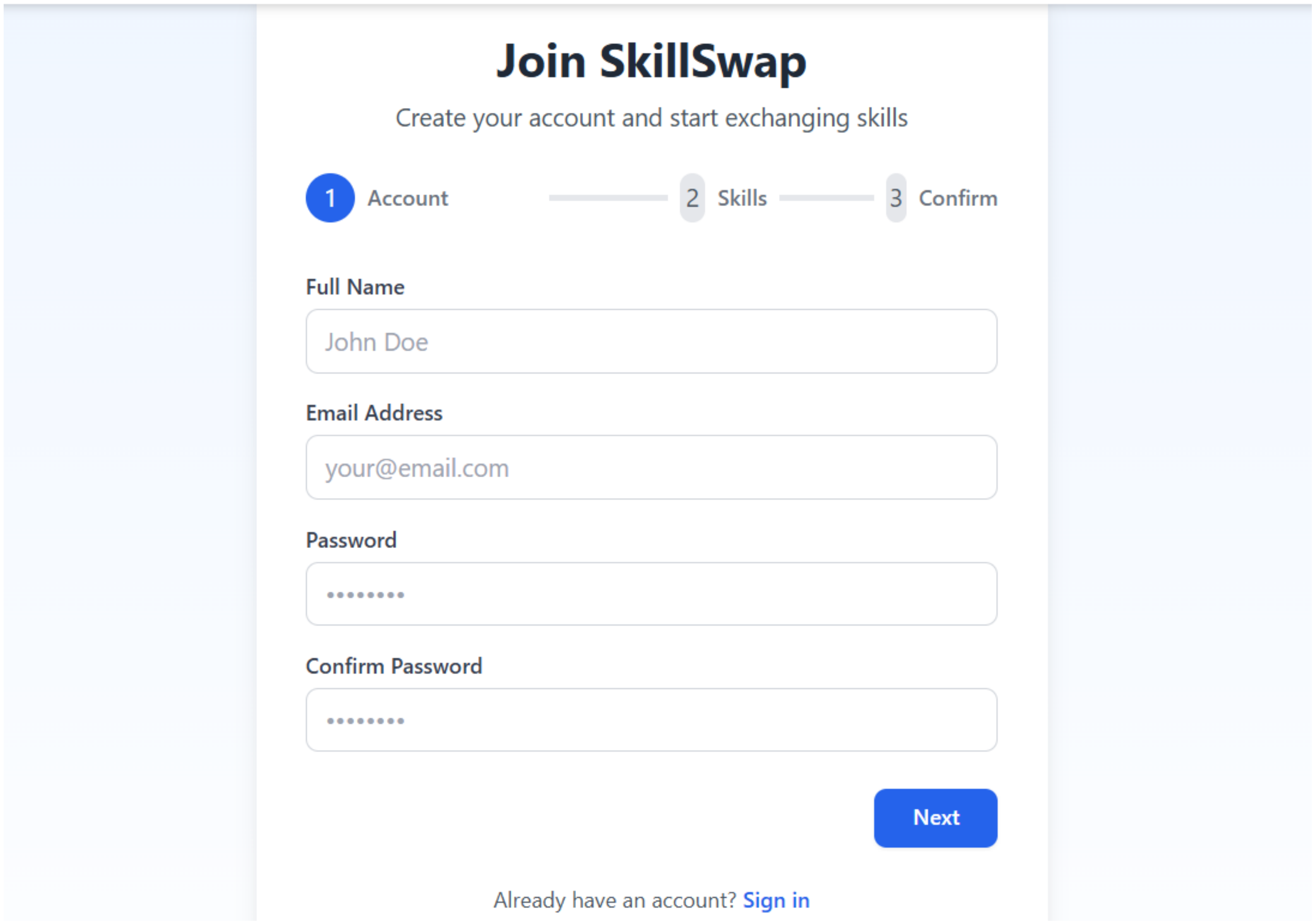
Connect with fellow students, exchange skills, and learn collaboratively on the first peer-to-peer skill exchange platform for students.

Start Swapping Skills

How SkillSwap Works

Our platform makes it easy to find peers with complementary skills, connect, and learn together.

Home Page



Join SkillSwap

Create your account and start exchanging skills

Full Name

John Doe

Email Address

your@email.com

Password

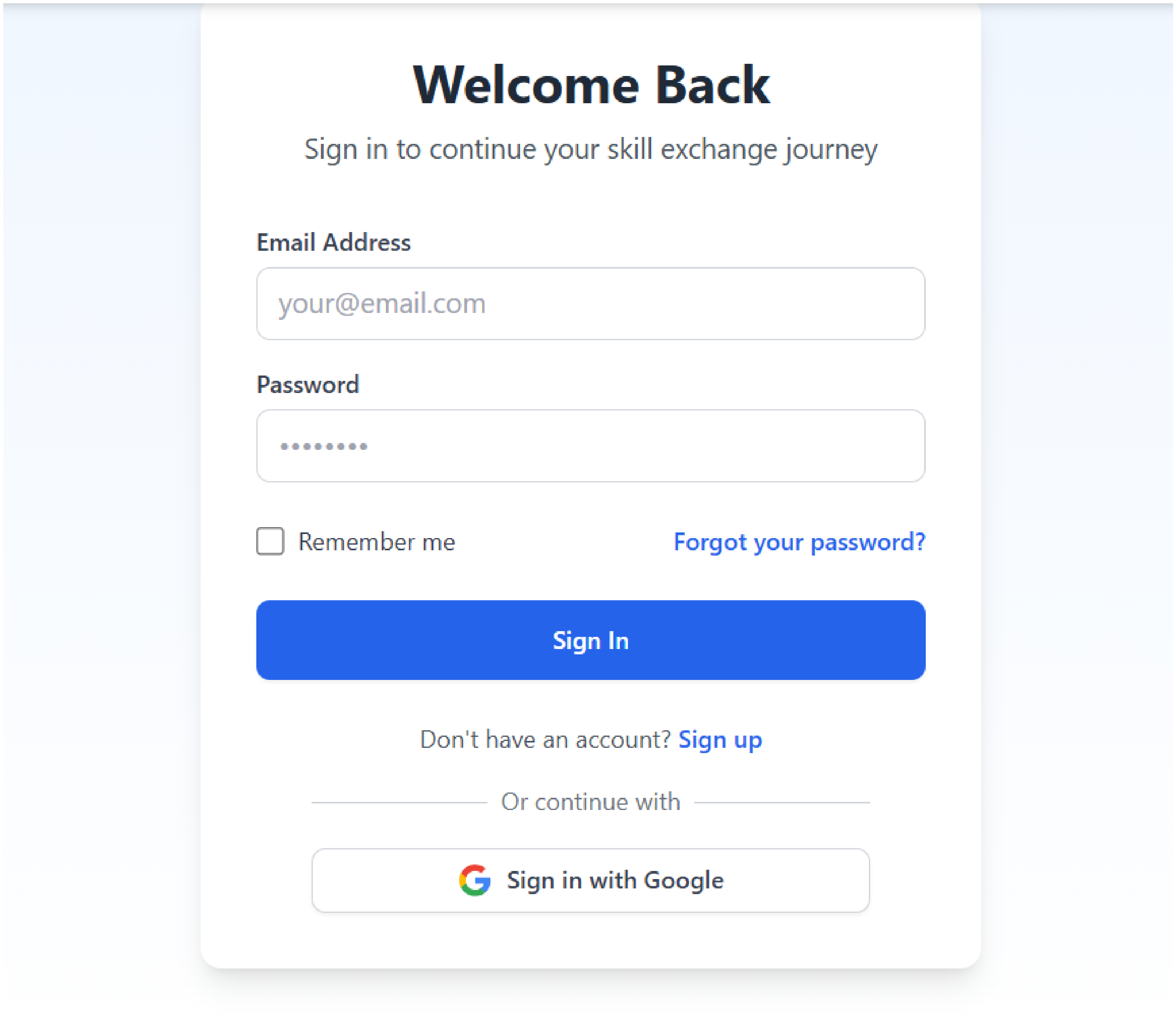
••••••••

Confirm Password

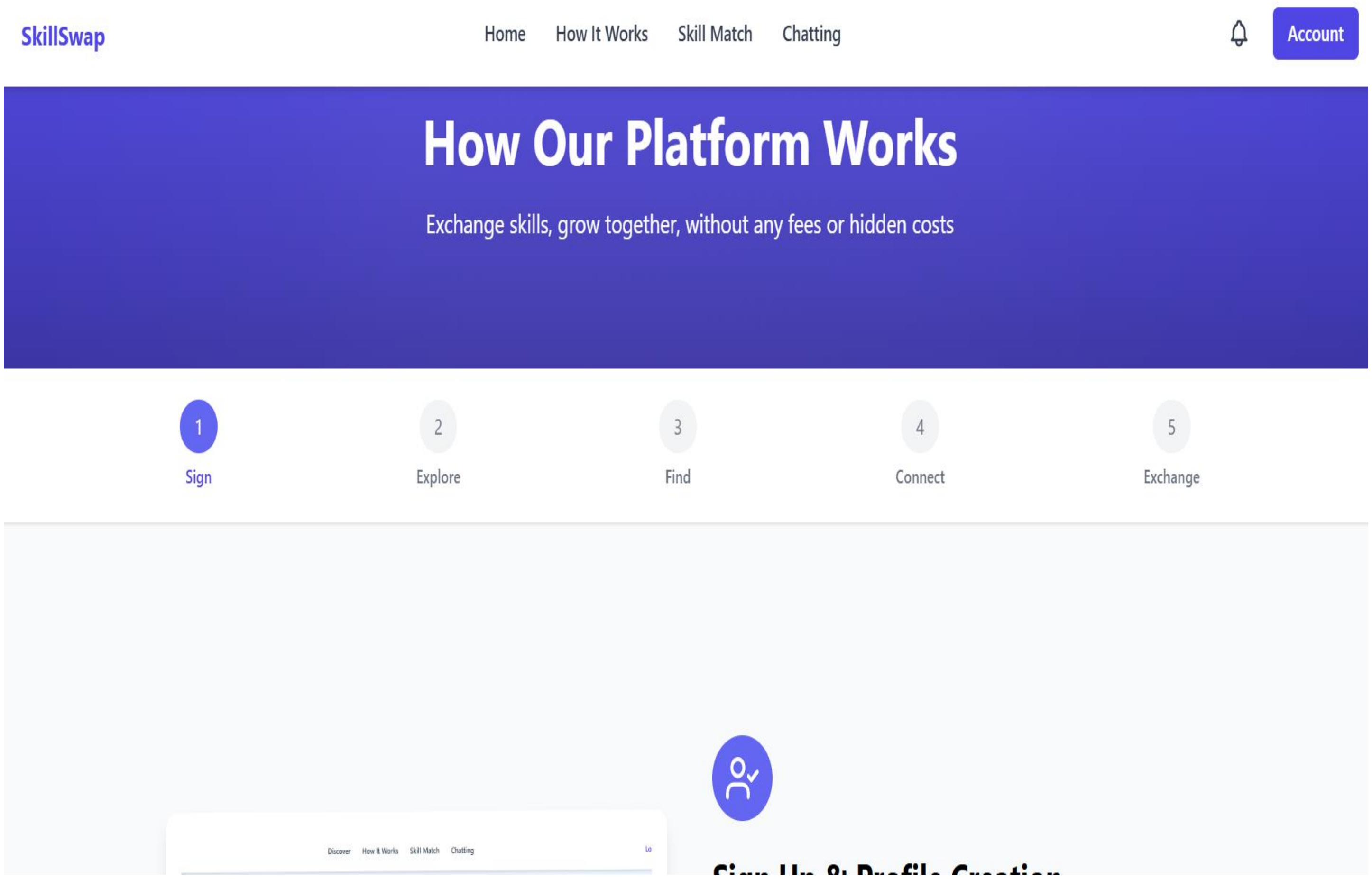
••••••••

Already have an account? Sign in

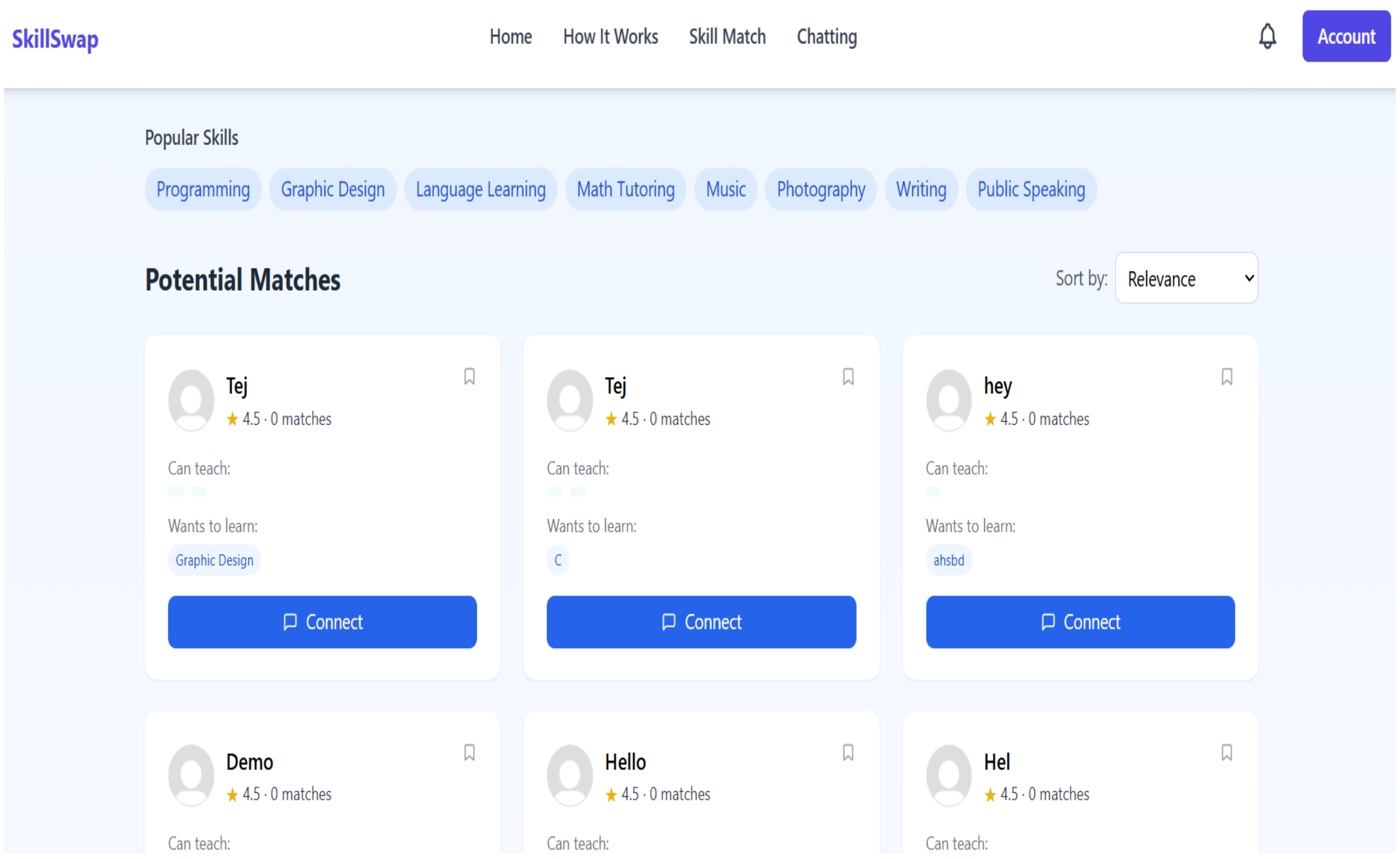
Signup Page


Welcome Back
Sign in to continue your skill exchange journey
Email Address
your@email.com
Password
••••••••
Remember me
Forgot your password?
Sign In
Don't have an account? Sign up
Or continue with
Sign in with Google

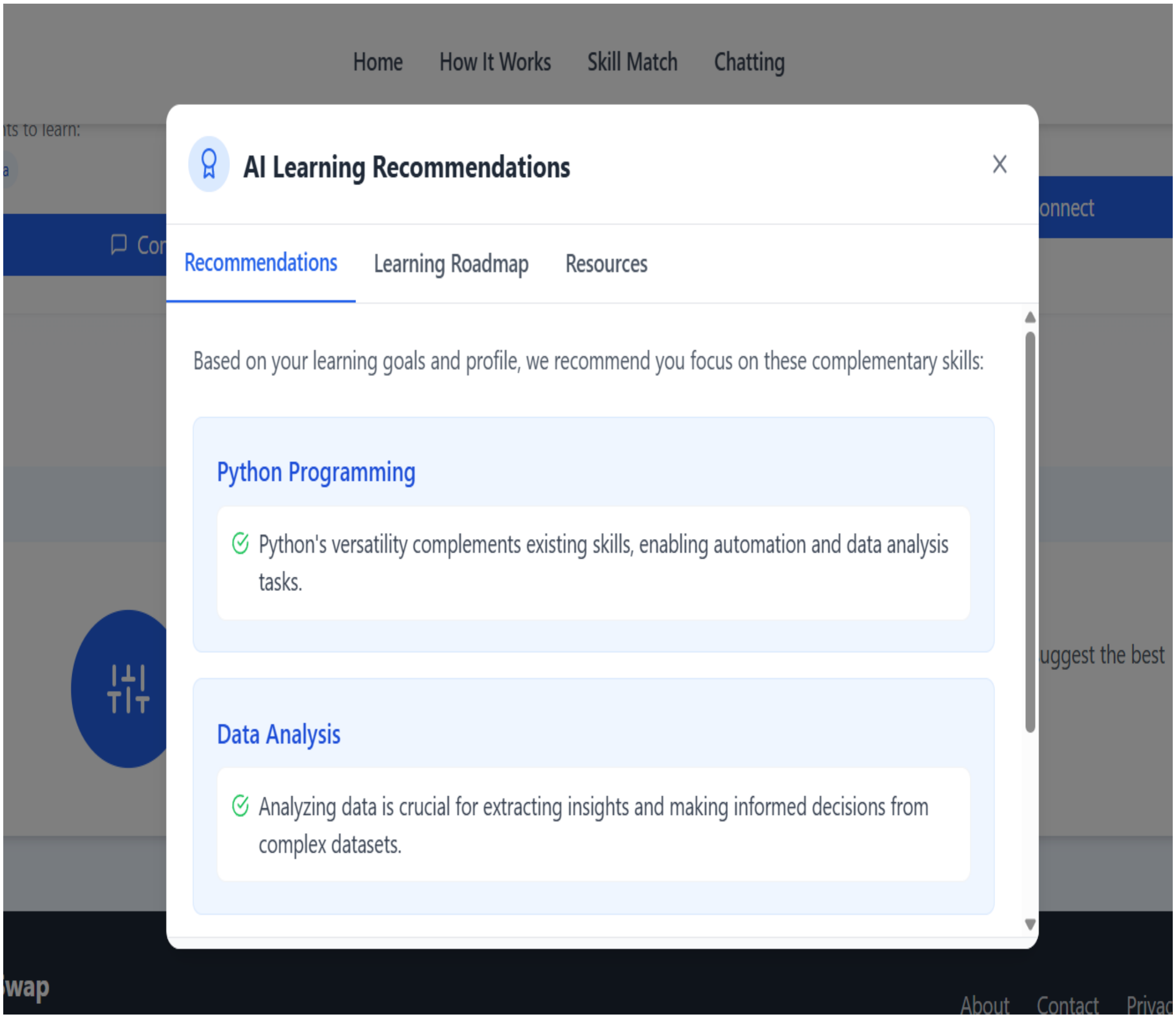
Login Page



How it works Page



Skill Matching Page



AI Learning Recommendations

Recommendations | Learning Roadmap | Resources

Based on your learning goals and profile, we recommend you focus on these complementary skills:

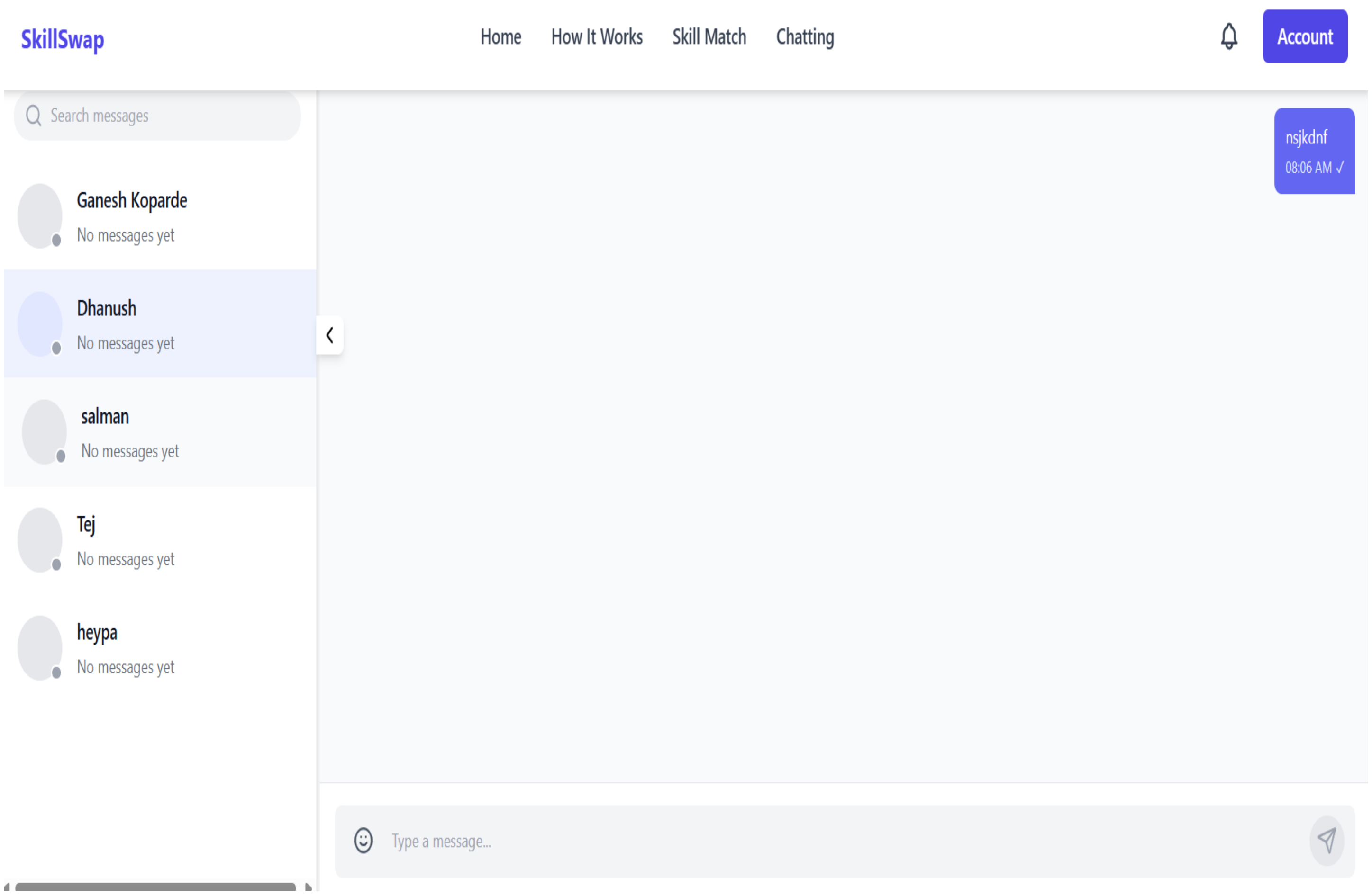
Python Programming

Python's versatility complements existing skills, enabling automation and data analysis tasks.

Data Analysis

Analyzing data is crucial for extracting insights and making informed decisions from complex datasets.

AI Recommendation Modal



Ganesh Koparde
No messages yet

Dhanush
No messages yet

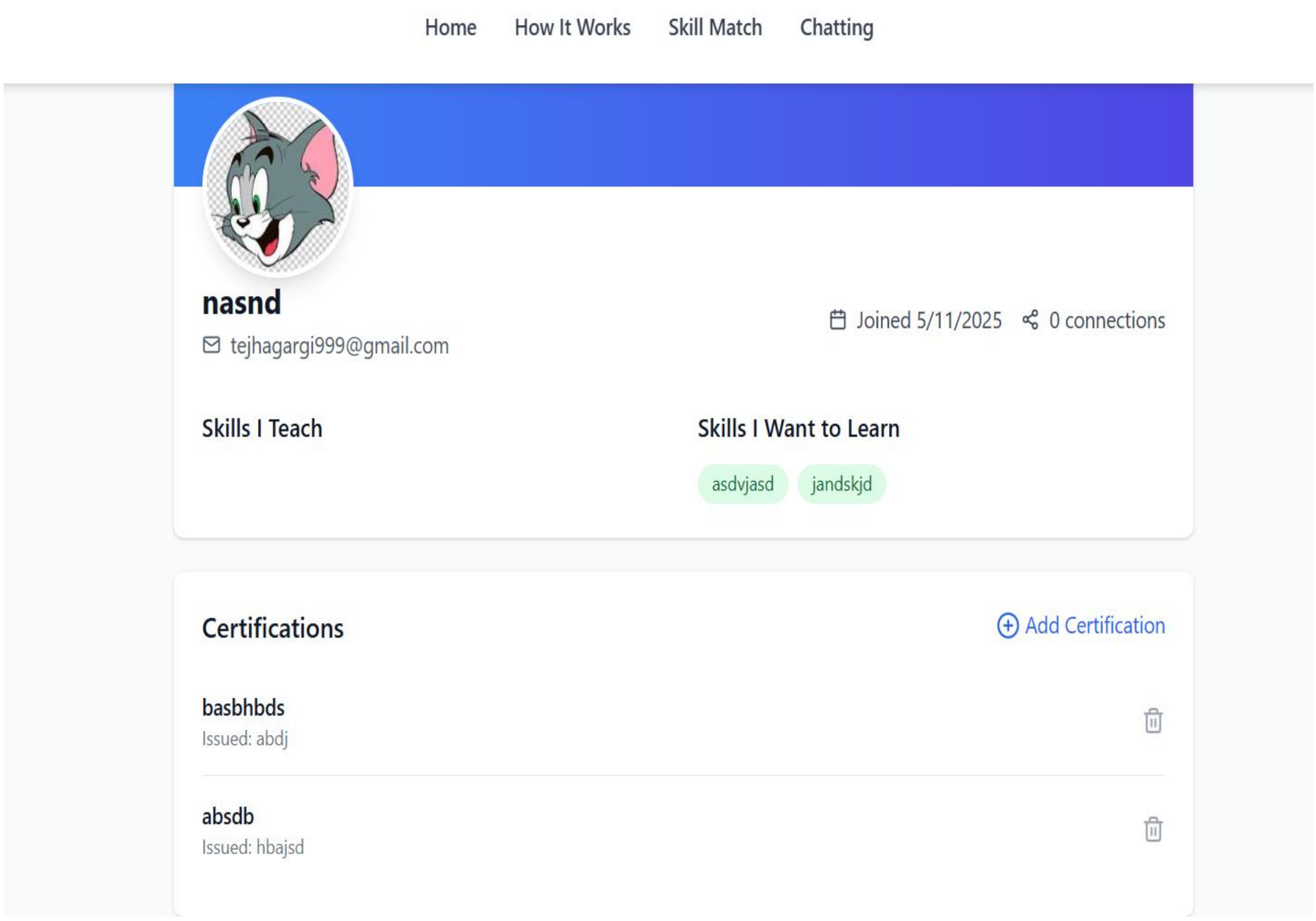
salman
No messages yet

Tej
No messages yet

heypa
No messages yet

nsjkdnf
08:06 AM ✓

Chatting Interface



Home How It Works Skill Match Chatting

nasnd

tejhagargi999@gmail.com

Joined 5/11/2025 0 connections

Skills I Teach

Skills I Want to Learn

asdvjasd jandskjd

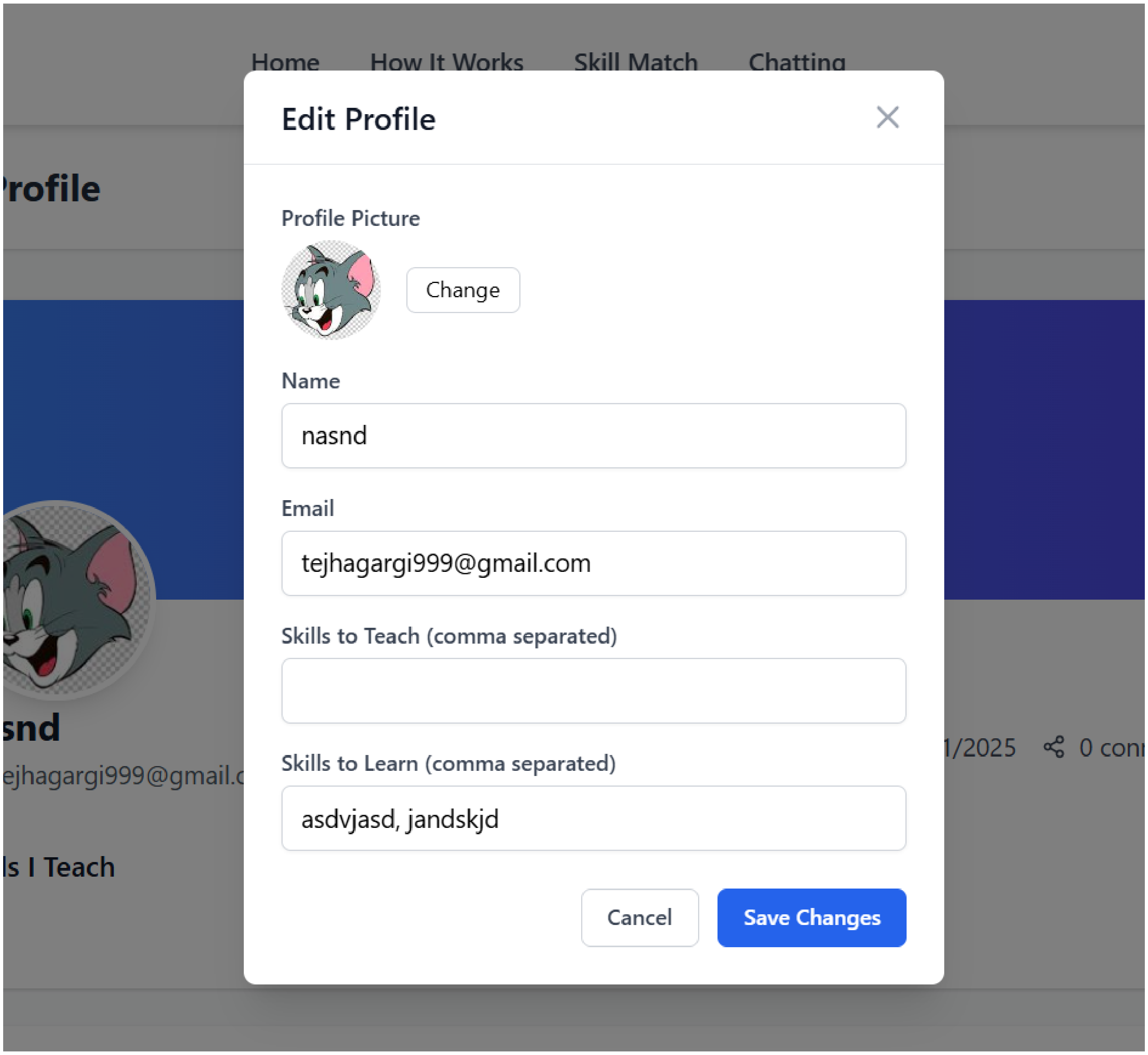
Certifications

Add Certification

basbhbds
Issued: abdj

absdb
Issued: hbajsd

Account Page



Edit Profile

Profile Picture

Change

Name

nasnd

Email

tejhagargi999@gmail.com

Skills to Teach (comma separated)

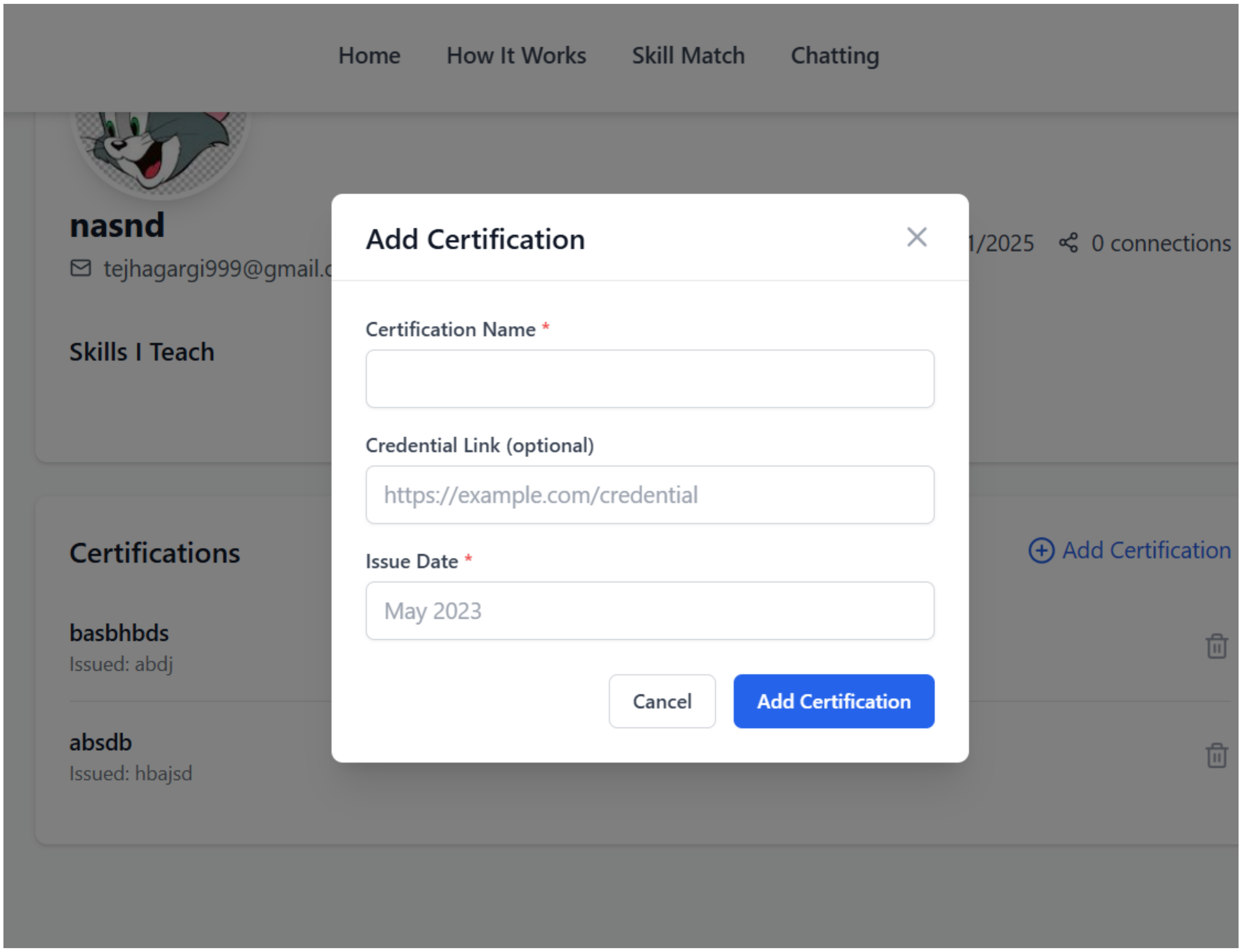
Skills to Learn (comma separated)

asdvjasd, jandskjd

Cancel

Save Changes

Profile Edit Modal



nasnd

tejhagargi999@gmail.c

1/2025 0 connections

Skills I Teach

Certifications

Add Certification

basbhbds
Issued: abdj

absdb
Issued: hbajsd

Add Certification

Certification Name *

Credential Link (optional)

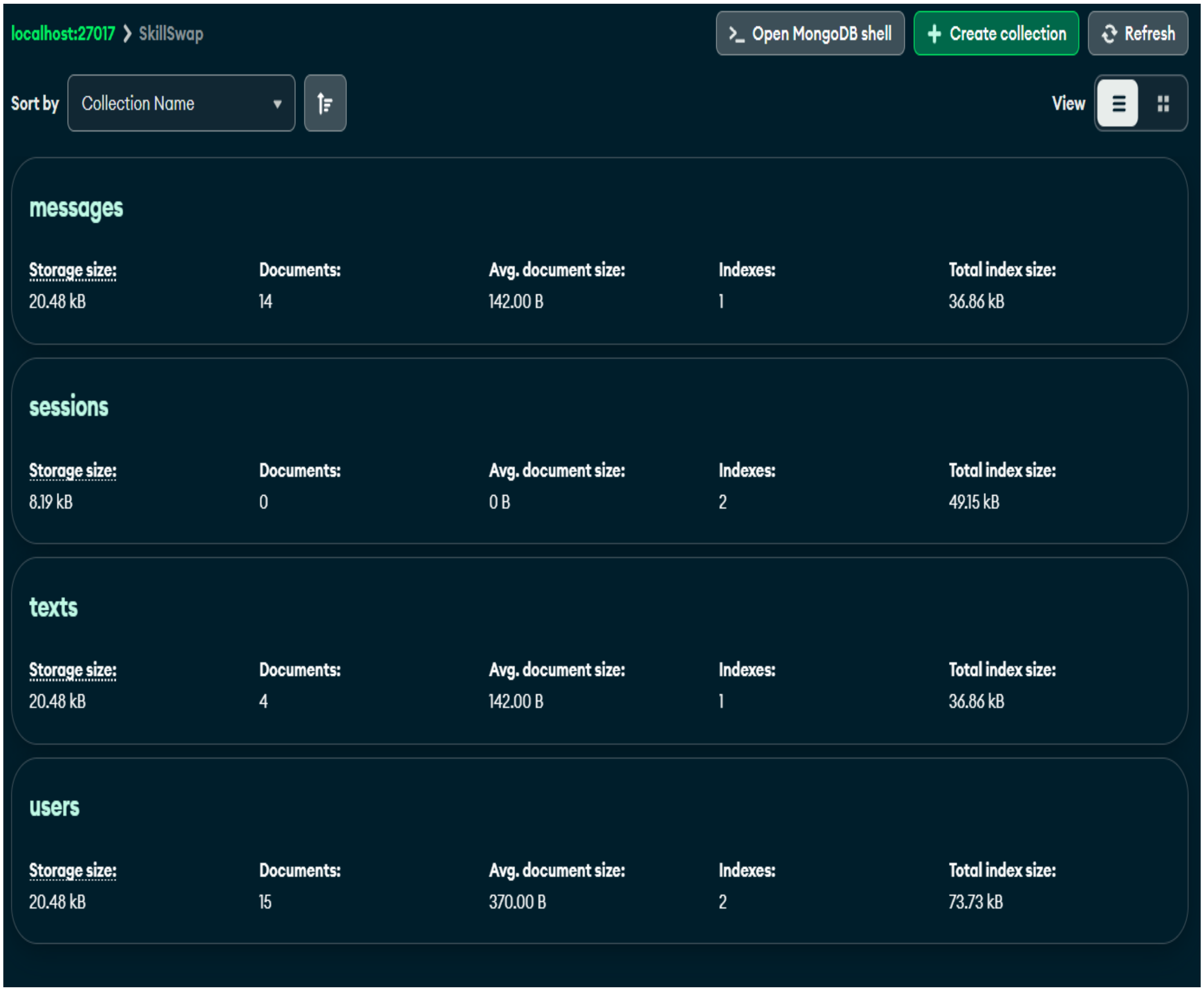
https://example.com/credential

Issue Date *

May 2023

Certification Adding Modal

10. Database modals



messages

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
20.48 kB	14	142.00 B	1	36.86 kB

sessions

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
8.19 kB	0	0 B	2	49.15 kB

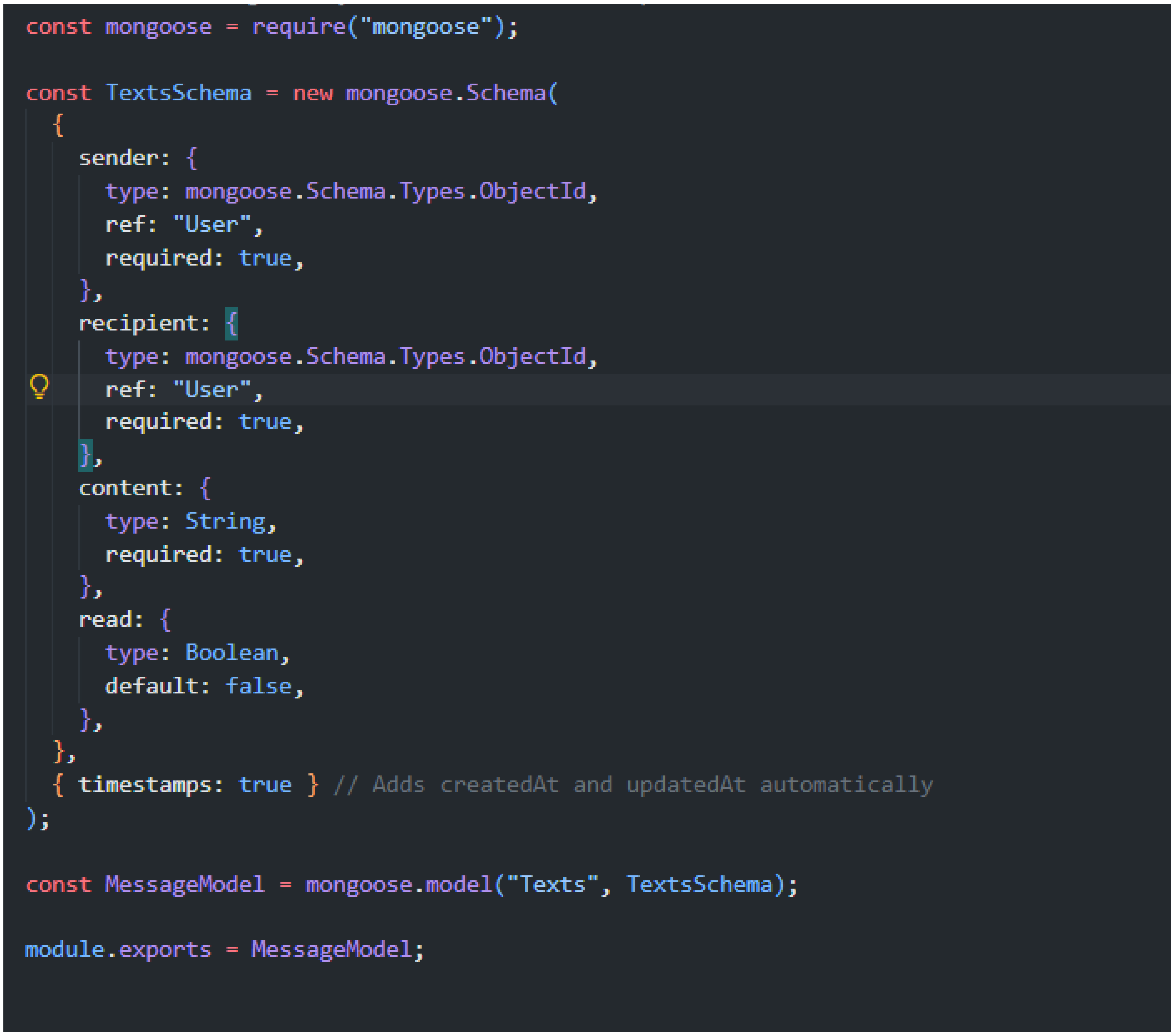
texts

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
20.48 kB	4	142.00 B	1	36.86 kB

users

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
20.48 kB	15	370.00 B	2	73.73 kB

Tables of the project



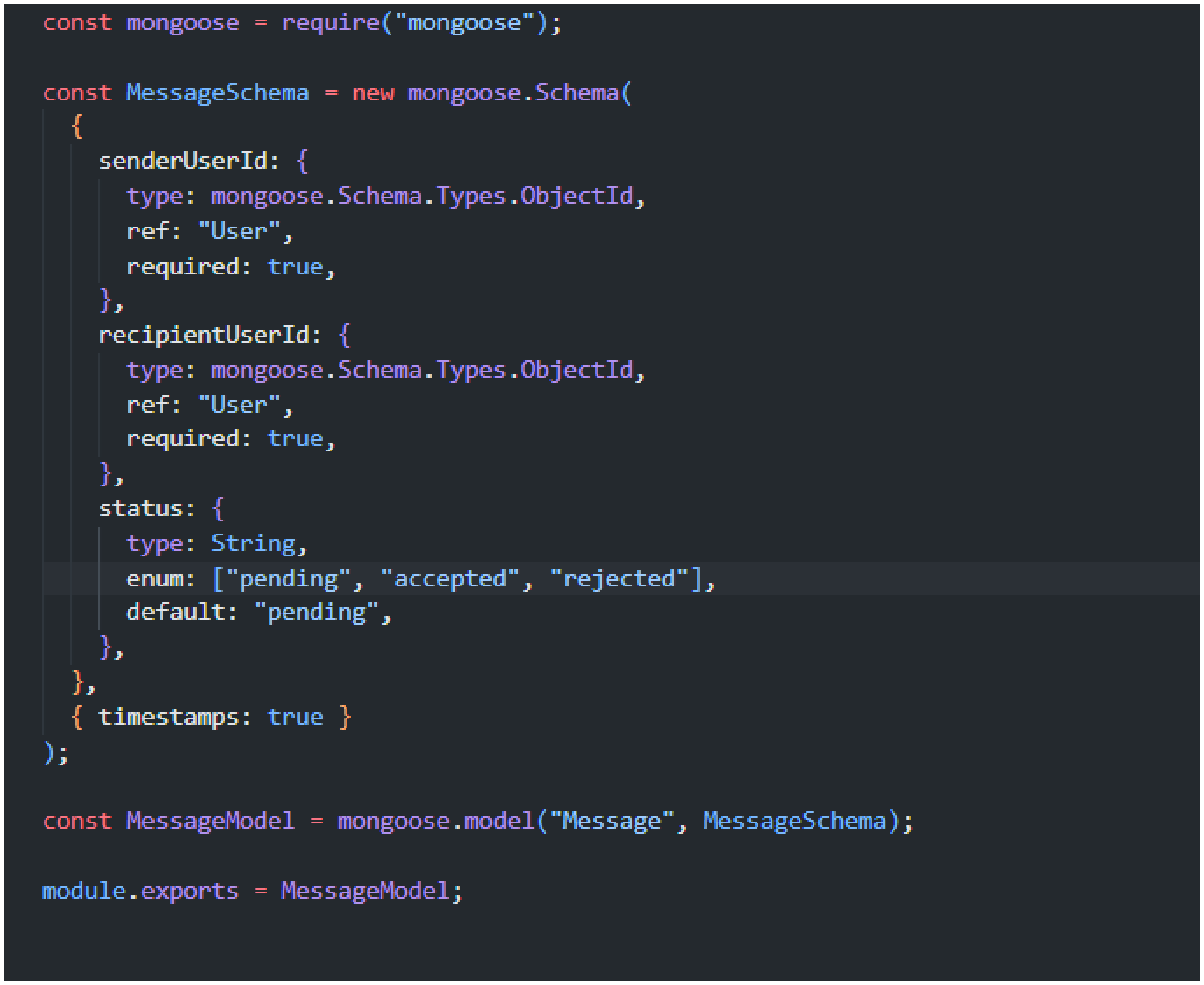
```
const mongoose = require("mongoose");

const TextsSchema = new mongoose.Schema(
  {
    sender: {
      type: mongoose.Schema.Types.ObjectId,
      ref: "User",
      required: true,
    },
    recipient: {
      type: mongoose.Schema.Types.ObjectId,
      ref: "User",
      required: true,
    },
    content: {
      type: String,
      required: true,
    },
    read: {
      type: Boolean,
      default: false,
    },
  },
  { timestamps: true } // Adds createdAt and updatedAt automatically
);

const MessageModel = mongoose.model("Texts", TextsSchema);

module.exports = MessageModel;
```

Schema of the Message Modal/Chat Modal



```
const mongoose = require("mongoose");

const MessageSchema = new mongoose.Schema(
  {
    senderUserId: {
      type: mongoose.Schema.Types.ObjectId,
      ref: "User",
      required: true,
    },
    recipientUserId: {
      type: mongoose.Schema.Types.ObjectId,
      ref: "User",
      required: true,
    },
    status: {
      type: String,
      enum: ["pending", "accepted", "rejected"],
      default: "pending",
    },
  },
  { timestamps: true }
);

const MessageModel = mongoose.model("Message", MessageSchema);

module.exports = MessageModel;
```

Schema for the Requests

```
const mongoose = require("mongoose");

const userSchema = new mongoose.Schema({
  fullName: {
    type: String,
    required: [true, "Please add a name"],
  },
  email: {
    type: String,
    required: [true, "Please add an email"],
    unique: true,
    match: [
      /^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/,
      "Please add a valid email",
    ],
  },
  googleId: String,
  password: {
    type: String,
  },
  bio: {
    type: String,
    trim: true,
  },
  certifications: [
    {
      name: {
        type: String,
        required: true,
      },
      link: {
        type: String,
      },
      date: {
        type: String,
        required: true,
      },
      createdAt: {
        type: Date,
        default: Date.now,
      },
    },
  ],
  image: {
    type: String,
    default:

"https://upload.wikimedia.org/wikipedia/commons/thumb/2/2c/Default_pfp
.svg/2048px-Default_pfp.svg.png",
  },
  teachSkills: [
    {
      skill: { type: String, required: true },
      tag: { type: String },
      proficiency: {
        type: String,
        enum: ["Beginner", "Intermediate", "Advanced"], // optional
      },
    },
  ],

  learnSkills: {
    type: [String],
    default: [],
  },
  ratings: {
    type: Number,
    default: 0,
  },
  matches: [
    {
      type: mongoose.Schema.Types.ObjectId,
      ref: "User",
    },
  ],
  createdAt: {
    type: Date,
    default: Date.now,
  },
});

const User = mongoose.model("User", userSchema);

module.exports = User;
```

Schema for the User Modal

11. System Testing

TESTING:

Testing is a process, which reveals errors in program. It is the major quality measure employed during software development. During testing, the program is executed with a set of conditions known as test cases and output is evaluated to determine whether the program is performing as expected.

The Primary and Larger objective of testing is to deliver quality software. Quality software is one that is devoid of errors and meets with a customer's stated requirements.

If errors are found, then the software must be debugged to locate these errors in the various programs. Corrections are then made. The program/system must be tested once again after corrections have been implemented - this time with an additional objective of finding out whether or not corrections in one part of the system have introduced any new errors elsewhere in the system.

Once all errors are found, then another objective must be accomplished that is check whether or not the system is doing what it is supposed to do. So, another aspect of testing is that it must also ensure that the system meets with user requirements.

- Techniques of testing
- Black Box Testing
- White Box Testing
- Equivalence Portioning
- Boundary Value Analysis
- Ad-hoc Testing

Specialized Testing done for this Project are:

- Volume Testing - This was done to determine whether or not the system is able to handle a large volume of data. The volume was a representative of the real-life volume with some provision for future growth.
- Performance testing - This is corollary to volume testing. This testing was done to focus on the performance of the System under large volumes and not just the ability to handle it.
- Security Testing - This attempt to verify that the protection mechanisms built into the system actually protects the system from unauthorized access or not.

- Regression Testing - This was basically done to see if any changes are made to one part of a Program whether it affects another part of System and also to check the deviations in behavior of unchanged parts of system
- Unit testing - Unit testing is normally considered as an adjunct to the coding step. After source level code has been developed, reviewed and verified for correspondence to component level design. A review of design information provides guidance for establishing test cases that are likely to uncover errors in each of the categories. Unit testing is responsible for testing each module in software structure independently.
- Integration testing - Tested modules are put together and tested in their integrity. Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objectives are to take unit tested components and build a program structure that has been discarded by design.

Testing strategies :

A testing strategy is general approach to the testing process rather than a method of devising particular system or components tests. Different strategies a may be adopted depending on the type of system to be tested and the development process used.

The testing strategies which discuss in this are:

- Top-down testing where testing starts with the most abstract component and works downwards.
- Bottom-up testing where testing starts with the fundamental components and works upwards.

12. Feasibility Study

Feasibility study is test of system proposal according to its workability, impact on the organization, ability to meet user needs and effective use of resources.

Three key considerations are:

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility

Technical Feasibility:

Technical Feasibility is method that determines whether the technology need for the proposed system is available and how can it be integrated within the organization.

- The proposed system can be implemented with the existing technical resources.

As the existing system is manual one, the user would readily implement the system as the software requirements could be easily met.

Economic Feasibility:

Economic feasibility is method used for evaluating the effectiveness of the proposed system. It is used to determine the benefits and savings that are expected from a proposed system and to whether the expenditure incurred for developing the new system will be cash effective or not.

- It can be implemented with existing system resources so no additional cost involved in implementing the system.
- The cost of paper work is reduced.
- The proposed system has only one-time investment.

Operational Feasibility

Operational Feasibility is method used for evaluating the effectiveness of the proposed system. It is used to determine whether the operations of the system can be carried out by the user easily or not.

- The system is easy to operate.
- No training is required
- The system will provide the user high speed & more accuracy.

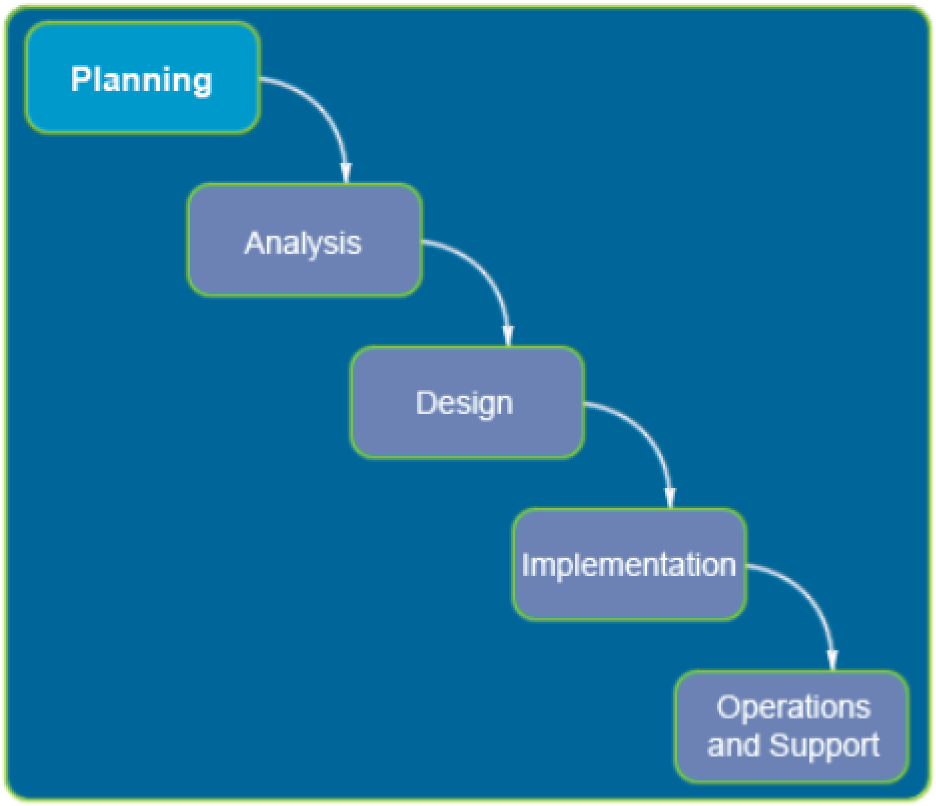
Behavioral Feasibility:

People are resistant to change and computers have been known to facilitate change. An estimate should be made by how strong a reaction the user staff is likely to have towards the development of computerized system.

- Time saving and cost effective.
- This system will work with existing set of tools; no other parts of the organization will be disturbed implementing this system.

13. Project Planning

SDLC



System Security Measures

System Security Measures apply to all systems at NYU; regardless of the level of their *System Classification* (see "Definitions," below). It is a baseline, which all systems must meet. Note that for most personal workstations, these are the only measures that apply. The requirements are:

1. **Password Protection:** All accounts and resources must be protected by passwords which meet the following requirements, which must be automatically enforced by the system:
    a. Must be at least eight characters long.
    b. Must NOT be dictionary or common slang words in any language, or be readily guessable.
    c. Must include at least three of the following four characteristics, in any order: upper case letters, lower case letters, numbers, and special characters, such as *!@#$%^&*.
    d. Must be changed at least once per year.

Software Updates: Systems must be configured to automatically update operating system software, server applications (web server, mail server, database server, etc.), client software (web browsers, mail clients, office suites, etc.), and malware protection software (antivirus, anti-spyware, etc). For Medium or High Availability systems, a plan to manually apply new updates within a documented time period is an acceptable alternative.

Firewall: Systems must be protected by a firewall that allows only those incoming connections necessary to fulfill the business need of that system. Client systems which have no business need to provide network services must deny all incoming connections. Systems that provide network services must limit access to those services to the smallest reasonably manageable group of hosts that need to reach them.

Malware Protection: Systems running Microsoft or Apple operating systems must have antivirus and anti-spyware software installed and it must be configured to automatically scan and update.

14. Data/database security

Database security concerns the use of a broad range of information security controls to protect databases (potentially including the data, the database applications or stored functions, the database systems, the database servers and the associated network links) against compromises of their confidentiality, integrity and availability.

It involves various types or categories of controls, such as technical, procedural/administrative and physical. *Database security* is a specialist topic within the broader realms of computer security, information security and risk management.

Security risks to database systems include, for example:

- Unauthorized or unintended activity or misuse by authorized database users, database administrators, or network/systems managers, or by unauthorized users or hackers (e.g. inappropriate access to sensitive data, metadata or functions within databases, or inappropriate changes to the database programs, structures or security configurations);

- Malware infections causing incidents such as unauthorized access, leakage or disclosure of personal or proprietary data, deletion of or damage to the data or programs, interruption or denial of authorized access to the database, attacks on other systems and the unanticipated failure of database services;
- Overloads, performance constraints and capacity issues resulting in the inability of authorized users to use databases as intended;
- Physical damage to database servers caused by computer room fires or floods, overheating, lightning, accidental liquid spills, static discharge, electronic breakdowns/equipment failures and obsolescence;
- Design flaws and programming bugs in databases and the associated programs and systems, creating various security vulnerabilities (e.g.

unauthorized privilege escalation), data loss/corruption, performance degradation etc.;

- Data corruption and/or loss caused by the entry of invalid data or commands, mistakes in database or system administration processes, sabotage/criminal damage etc.

Many layers and types of information security control are appropriate to databases, including:

- Access control
- Auditing
- Authentication
- Encryption
- Integrity controls
- Backups
- Application security

15. User profiles and access rights

User groups:

To control access for a large group of users at once, there's the concept of user groups. There are four predefined groups: Guests, Members, Global Moderators and Administrators. You can define your own groups in addition to this with the Groups Editor in the AdminCP. "Guests" is in fact not a real user group, but any user that is either not logged in or member of no group (not yet validated users) will be considered a guest and can be addressed as such with access rules.

You should understand that the "Members" group has a somewhat special meaning in that it contains all 'validated users. This is part of the user validation concept. When a user registers to the board and it is configured to first check their e-mail address before they can use their account, a user will be assigned to the "Members" group upon successful validation. Only this gives the user their default members' rights. Moderators, Administrators or members of other defined user groups need to be assigned to this particular user group in addition to their "Members" membership. It is possible – and in some cases required – for a user to be assigned to multiple user groups.

Access Rights:

Most networks will have been set up with 'access rights'. This means the administrator has set up each person who can log on, with the right to access certain files and folders. For instance, you may have a personal folder in which you have the right to open, read, write, create and delete files.

Other parts of the network may have files you can only read but not write. And finally, there will be areas that you cannot enter at all. In which case a message will often pop up to inform you "You do not have sufficient access rights" or something similar.

You may also inherit access rights by being assigned to a 'group'. For example, the admin may have set up a student group and set access rights to the group. Then, once you become a member, you automatically get the same access rights.

16. Cost Estimation of the Project

The Cost of a Project is derived not only from the estimates of effort and size but from other parameters such as hardware, travel expenses, tele-communication costs, training cost etc, should also be taken into account.

Software cost estimation is the process of predicting the amount of effort required to build a software system. Models provide one or more mathematical algorithms that compute cost as a function of a number of variables.

Size is a primary cost factor in most models and can be measuring using lines of code or function points. Models used to estimate cost can be categorized as either cost models or constraint models.

COCOMO is an example of a cost model. Software cost estimation is the process of predicting the amount of effort required to build a software system. Cost estimates are needed throughout the software lifecycle. Preliminary estimates are required to determine the feasibility of a project.

Detailed estimates are needed to assist with project planning. The actual effort for individual tasks is compared with estimated and planned values, enabling project managers to reallocate resources when necessary. The article is intended for those who are new to project cost estimation techniques, and those who would like to have a feedback on COCOMO II model.

My objective is to describe in a simple way basic cost estimation steps, tools and assumptions, having a real project in mind, and supplying only necessary details on the project itself. COCOMO (Constructive Cost Model) is a model that allows software project managers to estimate project cost and duration.

- The model is simple and well tested
- Provides about 20% cost and 70% time estimate accuracy

In general, COCOMO estimates project cost, derived directly from person-months effort, by assuming the cost is basically dependent on total physical size of all project files, expressed in thousands single lines of code (KSLOC).

The estimation formulas have the form:

Effort (in person-months) = a x KSLOCb

Where

SLOC: Single lines of code

a : is a coefficient

b: Scaling factor

17. Conclusion

The project has been successfully completed and tested in the long term; this project will make the use of library more efficient, user friendly, easier and effective. It is a real improvement over the existing manual system as it effectively overcomes all the drawbacks of the existing system.

In short, we can say that the digital libraries can save the time of the society by its services we can use this time and money in other developmental work. The implementation of the system in the organization will considerably reduce data entry, time and also provide readily calculated reports.

18. Bibliography

☐ **Stanford University – SkillSwap Final Report**
An in-depth study detailing the design process, user research, and iterative development of a peer-to-peer learning platform. Stanford University

☐ **International Journal of Modern Research in Science & Technology – SkillSwap**
This paper discusses the architecture and features of a SkillSwap platform, emphasizing user-centric design and scalability. IJMRSET

☐ **An-Najah National University – SkillSwap Project**
A comprehensive report on developing a mobile and web-based SkillSwap application using Flutter and Dart, focusing on community-driven learning. Najah Repository

☐ **SkillSwap by HarshSharma20503**
A MERN stack web platform facilitating collaborative learning through peer-to-peer guidance, featuring real-time chat, Google OAuth, and JWT authentication. GitHub

☐ **SkillSwap-Essentials by SkillSwap Network**
A centralized repository containing critical information, project details, and design elements to optimize workflow and ensure smooth collaboration. GitHub

☐ **Skillswap by Wellitsabhi**
An innovative platform where users can register, list their skills and interests, and find others to teach and learn from, promoting community-driven skill exchange. GitHub