

SkillSage

Connecting Companies with Expert Freelance Examiners

A.K.M. Ahsanul Haque

Assistant Professor

CSE Department, Southeast University

CSE474.1

Software Development and Project Management Lab

Prepared By:

Group Name: SkillSage Team

<u>Mst Tamanna Jannat</u>	<u>2022100010004</u>
<u>Sharifuzzaman Hasan</u>	<u>2022100010023</u>
<u>Md Mahabub Alam</u>	<u>2018000011001</u>
<u>Nashit Hossain Emon</u>	<u>2020000000080</u>



Department of Computer Science & Technology

Executive Summary:

SkillSage is an innovative platform designed to transform the hiring process by connecting companies with freelance examiners who specialize in candidate assessments. In today's competitive job market, hiring the right talent efficiently and effectively is critical. SkillSage streamlines this process by enabling businesses to access a pool of experienced examiners who conduct remote evaluations, offering companies valuable insights into candidate abilities.

This solution addresses the challenges of time, cost, and quality in recruitment by offering a flexible, on-demand model where companies can scale hiring support based on their needs. Examiners can be hired for single or multiple assessments, making it ideal for businesses with variable recruitment demands. Additionally, SkillSage's structured, two-stage feedback system allows companies to provide immediate and follow-up evaluations, ensuring accountability and improved hiring outcomes.

Through SkillSage, companies benefit from a more thorough vetting process without overextending resources, while candidates experience a transparent and streamlined hiring journey. This proposal outlines how SkillSage's platform, powered by the MERN stack, will deliver a user-friendly, secure, and efficient hiring solution that aligns with modern recruitment trends and requirements.

Contents

Introduction	5
Project Introduction.....	5
Key Features:.....	6
Background of the Study.....	7
Project Background.....	7
Objectives	8
Methodology [Process Model].....	9
The Project	10
Communication.....	10
Planning	10
Project Planning for SkillSage.....	10
3. Timeline & Phases:.....	10
Project Timeline and PERT Diagram.....	11
Analysis of Critical Path.....	12
Conclusion.....	12
Final Critical Path	12
Gantt Chart	13
Project Estimation.....	14
1. Scope Definition	14
Time Estimate (for a 5-6 month project):	17
Cost Estimate:	17
Cost-Benefit Analysis for SkillSage	17
Conclusion:.....	18
Training	18
Modeling.....	20
Function Definitions and Descriptions.....	21
Use Case Diagram	24

Use Case Narratives:	24
1. Client Registers an Account	24
2. Client Searches for Examiners.....	24
3. Examiner Sets Up Profile.....	24
4. Candidate Joins Interview	25
Class Diagram.....	26
Activity Diagram.....	27
Data Flow Diagram.....	28
State Diagram.....	29
Sequence Diagram	30
User Interface (UI) Design.....	30
User Authentication module:.....	31
Conversation Module.....	32
Evaluation Process	32
Entities and Attributes	33
Normalized Schema	35
Schema Diagram	36
ERD (Entity Relationship Diagram).....	37
Construction.....	38
1. MERN Stack Components	38
2. Redis (In-memory Caching).....	39
3. Socket.IO	39
Integration Overview	39
Deployment of SkillSage	40
a. Deployment Environment.....	40
Learning Experiences from SkillSage Development.....	41
Bibliography/References.....	41
Conclusion.....	42

Introduction

Project Introduction

SkillSage is a cutting-edge platform designed to connect companies with experienced freelance examiners to assess job seekers. Our mission is to streamline the hiring process by providing businesses with access to qualified professionals who can offer objective and comprehensive evaluations of candidates' skills.

In today's competitive job market, traditional hiring methods often fall short in providing detailed and unbiased assessments. SkillSage addresses this challenge by offering a seamless, user-friendly platform where companies can easily find and engage with freelance examiners specializing in various fields.



Traditional Recruitment

- **Limited Scalability**
- **Time-Consuming**
- **Bias Risks**



SkillSage

- **Expert Examiners**
- **Efficiency and Flexibility**
- **Objective Evaluations**

Key Features:

1. Expert Examiner Network:

- **Diverse Talent Pool:** Access a broad range of freelance examiners with expertise across various industries and disciplines.
- **Specialized Skills:** Find examiners with specific skills and qualifications tailored to your company's needs.

2. Smart Matching System:

- **Advanced Algorithms:** Utilize sophisticated algorithms to match companies with the most suitable examiners based on job requirements and candidate profiles.
- **Custom Filters:** Apply filters to refine searches and ensure a perfect fit between examiners and job roles.

3. Assessment Quality:

- **Accuracy:** Ratings on how accurately the examiner evaluates candidates' skills and competencies

4. User-Friendly Interface:

- **Intuitive Dashboard:** Navigate through a streamlined and easy-to-use dashboard for both companies and examiners.
- **Seamless Communication:** Engage with examiners through integrated messaging and scheduling tools.

5. Secure and Reliable Platform

6. Analytics and Reporting:

7. Transparent Pricing Models

8. Outcome Impact

Technology Stack: SkillSage is built using the MERN stack (MongoDB, Express.js, React, Node.js), ensuring a robust and scalable platform. The system leverages modern development practices and tools to deliver a high-performance, secure, and intuitive user experience.

Business Model: SkillSage operates on a subscription-based model for companies and a commission-based model for examiners. This dual revenue stream ensures a sustainable and scalable business while providing value to both parties involved.

Market Opportunity: With the increasing demand for specialized skill assessments and the growing freelance workforce, SkillSage is positioned to capture a significant share of the market. Our platform not only fills a critical gap in the hiring process but also offers a flexible and efficient solution that aligns with the evolving needs of businesses and professionals.

Vision: Our vision is to become the leading platform for skill assessments, fostering a more effective and transparent hiring process. We aim to empower companies with the tools they need to make informed hiring decisions and provide freelance examiners with opportunities to showcase their expertise.

Background of the Study

Project Background

The modern recruitment process has become increasingly complex, especially as companies strive to find candidates who possess not only the necessary technical skills but also the right cultural fit for their organization. Traditional methods of assessment, such as in-house interviews and standardized tests, often fail to provide an accurate and unbiased evaluation of a candidate's capabilities. This gap in the recruitment process has led to the need for more specialized and objective assessments, conducted by experts who can provide a deeper insight into a candidate's potential.

How It works?



Objectives

Primary Objective

To develop a platform that enables companies to easily find and engage with freelance examiners for assessing job seekers.

Primary Objective

- To build a user-friendly interface for both companies and examiners.
 - To ensure secure and confidential handling of candidate information.
 - To provide a scalable platform that can accommodate various industries and types of assessments.
-

Methodology [Process Model]

Project Requirements

a. Scalability and Performance

- **Project Requirement:** SkillSage will likely need to handle a growing number of users, including companies, freelance examiners, and job seekers. The platform should be able to scale efficiently to accommodate increased traffic and data.
- **Technology Choice:** **Node.js** is non-blocking and event-driven, making it highly efficient for handling multiple requests simultaneously. It's well-suited for I/O-heavy operations, such as real-time communication and data streaming, which might be integral to the platform.

b. Real-Time Communication

- **Project Requirement:** The platform might require real-time features like live chat between companies and examiners, instant notifications, or live updates during assessments.
- **Technology Choice:** **Node.js** with **Socket.io** or similar libraries is ideal for building real-time applications, as it can maintain persistent connections with clients, enabling fast and responsive communication.

c. Modern, Interactive User Interface

- **Project Requirement:** SkillSage needs to offer a responsive, interactive, and user-friendly interface to ensure a smooth user experience across different devices.
- **Technology Choice:** **React.js** is a powerful front-end library that allows developers to build complex, dynamic user interfaces with ease. Its component-based architecture enables reusability, making the UI development process more efficient and maintainable.

d. Single-Page Application (SPA)

- **Project Requirement:** SkillSage may benefit from a Single-Page Application (SPA) architecture, where the main interface dynamically loads content without requiring full page reloads, leading to a more seamless user experience.
- **Technology Choice:** **React.js** excels at building SPAs, enabling fast navigation and a smooth, responsive user experience. The use of React Router for managing routes within the app further enhances the ability to build complex, navigable interfaces.

SkillSage represents a significant step forward in streamlining the hiring process by providing companies with access to skilled freelance examiners. The platform's robust features and user-friendly design ensure that companies can confidently assess job seekers, leading to better hiring decisions and ultimately, more successful teams.

The Project

Communication

Objective: Ensure clear and effective communication throughout the project lifecycle to align team members, stakeholders, and clients.

- **Channels:** Specify the primary communication channels (e.g., email, Slack, project management tools) and their intended use.
- **Stakeholders:** Identify who needs to be communicated with regularly (e.g., team members, clients, project sponsors) and their communication preferences.
- **Documentation:** Outline how project documentation, decisions, and changes will be recorded and shared (e.g., using a shared document repository or project management tool).
- **Feedback Mechanisms:** Describe how feedback will be gathered and addressed to ensure continuous improvement and stakeholder satisfaction.

Planning

Project Planning for SkillSage

1. Objectives:

- Connect companies with freelance examiners for candidate assessments.
- Enable easy interview scheduling, real-time communication, and feedback.
- Manage examiner payments and allow post-interview reviews.

2. Key Features:

- **User Roles:** Admin, Company (Client), Freelance Examiner, Candidate.
- **Core Functionalities:**
 - User registration/authentication
 - Profile management for examiners and companies
 - Scheduling and interview links
 - Payment processing and feedback system

3. Timeline & Phases:

- **Requirements Gathering:** 2 weeks

- **System Design:** 2 weeks
- **Development:** 3 phases, each lasting 4 weeks
- **Testing & QA:** 2 weeks
- **Deployment:** 1 week
- **Maintenance:** Ongoing

4. Resources & Technologies:

- **Team:** Frontend & Backend Developers, UI/UX Designer, Database Admin.
- **Tech Stack:** React.js, Next.js, Tailwind CSS, Node.js, MongoDB, Stripe (Payments), Socket.IO (Real-time).

5. Risks & Mitigation:

- **Development Delays:** Track progress with milestones.
- **Security Risks:** Use encryption and secure APIs.
- **Payment Failures:** Test integration rigorously.

6. Budget:

- Estimated total: \$15,500 - \$26,000

Conclusion: SkillSage will streamline hiring by connecting companies with freelance examiners, simplifying interview scheduling, communication, and payments. A structured approach ensures a secure, user-friendly experience.

Project Timeline and PERT Diagram

To visualize and optimize the project's timeline, a **PERT (Program Evaluation and Review Technique) diagram** is used. This diagram outlines each phase, task dependencies, and expected completion times, helping to identify critical paths and potential bottlenecks.

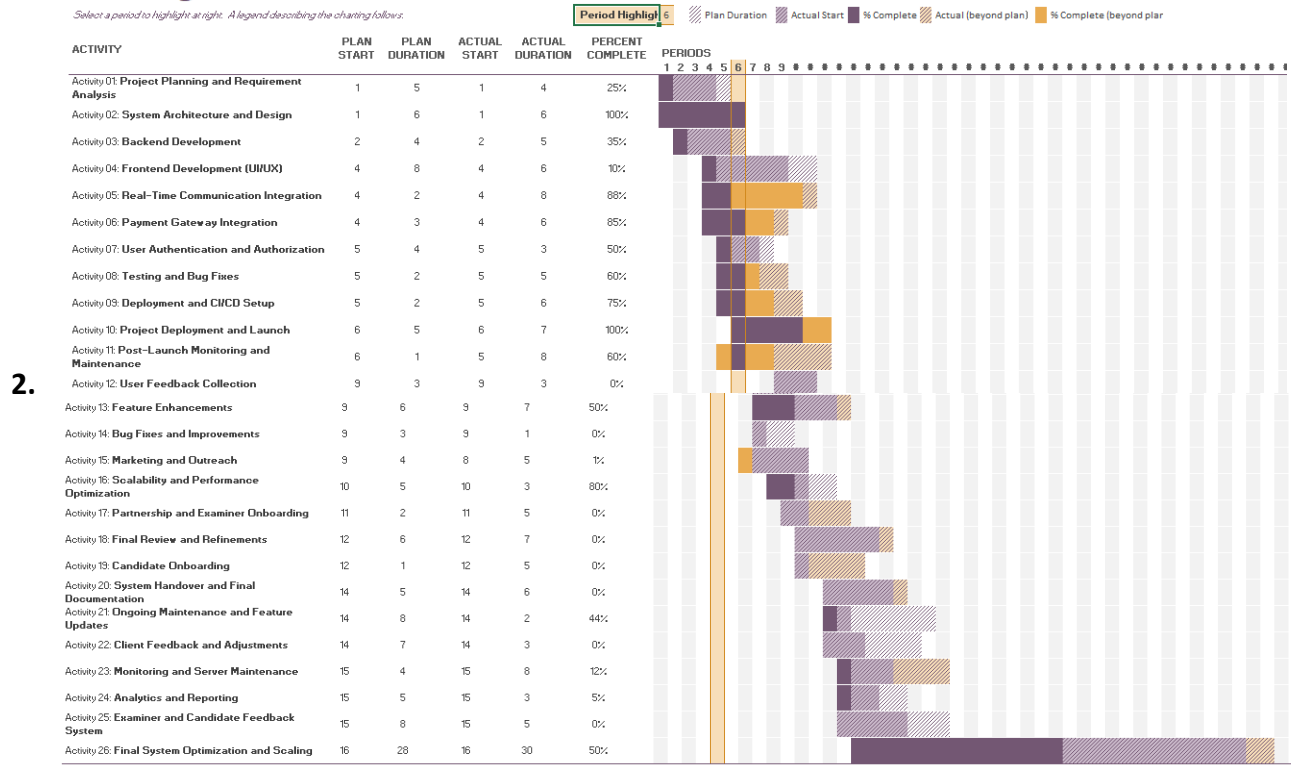
Gantt Chart

1. Here's a basic outline of a Gantt chart for the **SkillSage** project. This will detail phases from planning and design to deployment and testing. Assuming a project timeline of approximately 4-6 months, each phase has key tasks and milestones to keep the development on track.

Phase	Task	Duration	Timeline
1. Project Planning	Requirements Gathering	1 week	Week 1
	Define Scope and Features	1 week	Week 1
	Set Project Milestones	1 week	Week 2
2. Design	UI/UX Design for Client Interface	2 weeks	Weeks 3-4
	UI/UX Design for Examiner Interface	2 weeks	Weeks 3-4
	UI/UX Design for Candidate Portal	1 week	Week 5
	Feedback and Iteration	1 week	Week 6
3. Frontend Development	Setup Project Structure	1 week	Week 7
	Develop Client Dashboard	2 weeks	Weeks 8-9
	Develop Examiner Dashboard	2 weeks	Weeks 8-9
	Candidate Interface	1 week	Week 10
	Integrate Tailwind/MUI for Styling	1 week	Week 10
4. Backend Development	Set Up Express Server and MongoDB	1 week	Week 11
	Develop Authentication (JWT)	1 week	Week 12
	Build API Endpoints	2 weeks	Weeks 12-13
	Implement Real-Time Notifications	1 week	Week 13
	Payment Integration	2 weeks	Weeks 14-15
5. Testing & QA	Unit Testing	1 week	Week 16
	Integration Testing	1 week	Week 17
	User Acceptance Testing	2 weeks	Weeks 18-19
6. Deployment	Prepare for Deployment	1 week	Week 20
	Deploy to Production	1 week	Week 21
	Post-Deployment Monitoring	1 week	Week 22
7. Documentation & Review	Create Documentation	1 week	Week 23
	Project Review and Feedback	1 week	Week 24

SkillSage

Select a period to highlight at right. A legend describing the charting follows.



3.

Project Estimation

Project estimation for SkillSage involves calculating various metrics to predict the time, cost, and resources required to complete the project successfully. Estimation involves considering the project size, complexity, team capabilities, risks, and project-specific factors like technologies and features. The main steps of the estimation process for SkillSage could include the following:

1. Scope Definition

Clearly define the full scope of the project:

- **Key features:** User accounts, examiner profiles, scheduling, payment gateway, real-time communication, feedback system, etc.
- **Technologies:** Node.js, Express.js, React.js, MongoDB, Redis, Socket.IO, etc.
- **Key Milestones:** Planning, development, testing, deployment, post-launch, etc.

2. Breakdown into Phases and Tasks

Divide the project into smaller, manageable tasks (like the activities listed earlier) to ensure you estimate accurately:

- **Requirement Analysis & Planning**
- **System Architecture & Design**
- **Frontend Development**
- **Backend Development**
- **Real-Time Features Integration**
- **Testing & Debugging**
- **Deployment**
- **Post-Launch Monitoring & Maintenance**

3. Effort Estimation (Time & Resources)

Estimate the effort for each phase:

- **Requirement Analysis & Planning:** 2-3 weeks
- **System Design:** 3-4 weeks
- **Frontend Development:** 6-8 weeks (depending on UI complexity)
- **Backend Development:** 8-10 weeks (with APIs, database setup, security)
- **Real-Time Features:** 4-5 weeks (Socket.IO, messaging)
- **Testing & Debugging:** 4 weeks (unit testing, integration testing, QA)
- **Deployment & Launch:** 1-2 weeks (CI/CD setup, final checks)
- **Post-Launch Monitoring & Maintenance:** Ongoing after launch (first 2-3 months more intensive)

4. Cost Estimation

Cost is usually a function of time, team size, and hourly rates. Here's a basic structure:

- **Project Manager:** \$30–50/hour
- **Frontend Developer:** \$25–45/hour
- **Backend Developer:** \$30–50/hour

- **Full-Stack Developer:** \$35–55/hour
- **QA Engineer:** \$20–40/hour
- **Designer (UI/UX):** \$25–45/hour
- **DevOps Engineer:** \$35–55/hour

Sample Estimation (for a team of 5)

- **Total hours per developer per week:** 40 hours
- **Project duration:** 6 months (rough estimate)

Role	Hourly Rate	Estimated Hours	Cost Estimate
Project Manager	\$40	400 hours	\$16,000
Frontend Developer	\$35	600 hours	\$21,000
Backend Developer	\$40	600 hours	\$24,000
QA Engineer	\$30	240 hours	\$7,200
UI/UX Designer	\$35	200 hours	\$7,000
Total Cost			\$75,200

5. Risk Estimation

Identify potential risks that could affect the project timeline and cost:

- **Feature creep:** The addition of features during development can increase both cost and time.
- **Technical complexity:** Integrating complex systems (real-time chat, payment gateway, etc.) can lead to delays or unexpected issues.
- **Team skill gaps:** Ensure that your team has the right skills, or you may face delays.
- **Unforeseen issues:** Bugs, third-party tool failures, etc., can cause delays.

6. Buffer for Contingencies

Include a contingency buffer of **10-20%** to cover unforeseen issues, team illness, or unexpected delays.

For example:

- **Buffer for delays:** Add 10-15% of total project cost and timeline as a contingency.

7. Schedule Estimate

- **Total Project Duration:** ~ 5–6 months (this depends on team capacity, complexity of features, and overlapping tasks)
- **Buffer for Delays:** ~10% of total time (another 2–3 weeks)

8. Final Estimation Example

Let's assume you want to have an overall estimate for the SkillSage project:

Time Estimate (for a 5-6 month project):

- **Planning & Design:** 1 month
- **Development (Backend & Frontend):** 3 months
- **Testing & QA:** 1 month
- **Deployment & Post-Launch Monitoring:** 1 month
- **Buffer:** 2-3 weeks (for unforeseen issues)

Cost Estimate:

If you factor in a team of 5 (Project Manager, Frontend, Backend, QA, and Designer):

Total cost: \$75,200 (estimated cost for a 6-month project)

This approach provides a rough estimate of the project's cost, timeline, and resources. You can adjust based on team size, hourly rates, and task duration specifics. Would you like to refine any of these figures or activities further?

Cost-Benefit Analysis for SkillSage

Costs (Year 1):

- **Development Costs** (Frontend, Backend, Design, Testing): \$75,200
- **Operational Costs** (Server, Maintenance, Marketing): \$96,000
- **Contingency:** \$7,520
- **Total Costs:** \$178,720

Benefits (Year 1):

- **Revenue from Subscriptions** (100 companies @ \$500/month): \$600,000

- **Revenue from Transaction Fees** (1,000 exams @ \$10): \$10,000
- **Total Revenue:** \$610,000

Net Benefit:

- **Net Benefit** = \$610,000 - \$178,720 = **\$431,280**

ROI:

- **ROI** = $(\$431,280 / \$178,720) \times 100 = \mathbf{241.5\%}$

Break-Even Point:

- Achieved with **~30 companies** subscribing.

Conclusion:

SkillSage offers a **241.5% ROI** in the first year, with a **break-even point** after about 30 companies. The project is highly profitable and financially viable.

Training

Here's a streamlined training plan for SkillSage users and team members:

1. Client Training (Company Owners)

- **Platform Navigation:** Introduction to the SkillSage features and user interface.
- **Hiring Process:** Guide on searching, hiring examiners, and scheduling candidate interviews.
- **Payment & Feedback:** Managing payments and providing examiner feedback post-hire.

Format: Interactive webinar, user guides, and in-app FAQs.

2. Freelance Examiner Training

- **Profile Setup:** Creating and optimizing examiner profiles.
- **Client Interaction:** Guidelines on responding to client messages and confirming interviews.
- **Payment System:** Overview of payment process and withdrawal management.

Format: Video tutorials and support documentation.

3. Candidate Training

- **Interview Access:** Instructions for accessing the interview link and participation tips.
- **Notifications:** Overview of notifications and reminders for interview dates.

Format: Quick-start guide and automated reminders.

4. Admin and Support Team Training

- **Platform Mastery:** Deep understanding of platform features and resolving user issues.
- **System Monitoring:** Training on analytics tools to monitor system health and user metrics.

Format: Hands-on workshops and documentation.

5. Resources for All Users

- **Help Center:** Centralized FAQs, guides, and troubleshooting.
- **Live Webinars & Q&A:** Monthly updates and feature tutorials.
- **Automated In-App Tips:** Tooltips to guide new users through essential features.

This approach ensures each group has targeted resources for using SkillSage effectively.

Modeling

Objective: Develop models that represent the system's design, functionality, and architecture to guide development and ensure alignment with project requirements.

1. Project Features

Here's a summary of key features for the **SkillSage** project:

1. **User Profiles:** Dedicated profiles for clients, freelance examiners, and candidates with customizable information.
2. **Examiner Search & Hiring:** Clients can search for examiners, view profiles, and initiate hiring directly through the platform.
3. **Interview Scheduling:** Integrated calendar for scheduling candidate interviews with examiners, complete with reminders.
4. **Payment Processing:** Secure payment gateway for handling examiner fees and invoicing.
5. **Messaging & Notifications:** Real-time messaging and notifications for seamless communication and updates.
6. **Review System:** Clients can provide feedback on examiners after hiring and optionally follow up after three months.
7. **Admin Dashboard:** Comprehensive admin panel for monitoring activity, managing users, and overseeing transactions.
8. **Performance Analytics:** Reporting tools to track user engagement, performance metrics, and platform health.
9. **User Feedback Collection:** System for gathering user feedback to improve services and address issues.
10. **Security & Compliance:** Data security measures, including authentication, role-based access, and compliance with privacy standards.

These features make SkillSage a comprehensive solution for managing freelance examiners and candidate assessments.

Function Definitions and Descriptions

1. User Management

- **registerUser(type, userDetails)**
 - **Description:** Registers a new user on SkillSage. Takes the user type (client, examiner, candidate) and user details (e.g., name, email, password) as input, validating and saving the information to the database.
 - **Returns:** A success message and user ID or an error if registration fails.
 - **loginUser(email, password)**
 - **Description:** Authenticates the user based on email and password. If successful, returns a session token or JWT for secure access to platform features.
 - **Returns:** Session token/JWT or an error message on failure.
 - **getUserProfile(userId)**
 - **Description:** Retrieves the profile information of a specified user by user ID, ensuring profile customization for each user type.
 - **Returns:** User profile data.
 - **updateUserProfile(userId, updatedDetails)**
 - **Description:** Updates the user profile with new details provided by the user.
 - **Returns:** Confirmation of profile update or an error if updating fails.
-

2. Examiner Search and Matching

- **searchExaminers(filters)**
 - **Description:** Enables clients to search for examiners using filters like skills, experience, location, and ratings. Returns a list of matching examiners.
 - **Returns:** List of examiners matching the filter criteria.
 - **smartMatchExaminers(jobDetails)**
 - **Description:** Suggests the best examiners for a job based on job-specific requirements and examiner expertise.
 - **Returns:** List of recommended examiners.
 - **viewExaminerProfile(examinerId)**
 - **Description:** Allows clients to view a detailed profile of a specific examiner.
 - **Returns:** Examiner profile details, including past performance, skills, and feedback ratings.
-

3. Candidate Application and Management

- **applyForJob(candidateId, jobId)**
 - **Description:** Allows candidates to apply for available job positions posted by clients.

- **Returns:** Confirmation of the application or an error if unsuccessful.
 - **getCandidateApplications(clientId)**
 - **Description:** Retrieves all candidates who have applied for the client's job postings.
 - **Returns:** List of candidate applications for client review.
-

4. Assessment and Scheduling

- **scheduleAssessment(examinerId, candidateId, scheduleDetails)**
 - **Description:** Allows an examiner to schedule an assessment with a candidate based on provided details (date, time, platform).
 - **Returns:** Confirmation of the scheduled assessment or an error if scheduling fails.
 - **submitAssessmentFeedback(examinerId, candidateId, feedback)**
 - **Description:** After an assessment, the examiner submits feedback on the candidate's performance.
 - **Returns:** Confirmation that feedback was submitted successfully.
 - **getAssessmentReport(clientId, candidateId)**
 - **Description:** Allows the client to retrieve an examiner's assessment report for a candidate.
 - **Returns:** Detailed assessment report.
-

5. Payment and Review

- **processPayment(clientId, examinerId, paymentDetails)**
 - **Description:** Processes a payment from the client to the examiner for completed assessments.
 - **Returns:** Confirmation of payment or an error message if the transaction fails.
 - **leaveReview(clientId, examinerId, review)**
 - **Description:** Allows the client to leave a review and rating for the examiner's performance.
 - **Returns:** Confirmation of review submission.
-

6. Candidate Shortlisting

- **shortlistCandidates(examinerId, jobId, shortlistedCandidates)**
 - **Description:** Allows examiners to submit a list of shortlisted candidates based on the assessment results for a specific job.
 - **Returns:** Confirmation of shortlisted candidates or an error if shortlisting fails.
- **getShortlistedCandidates(clientId, jobId)**

- **Description:** Retrieves a list of shortlisted candidates for a specific job, submitted by the examiner.
 - **Returns:** List of shortlisted candidates.
-

7. Notifications and Feedback Follow-Up

- **sendNotification(userId, message)**

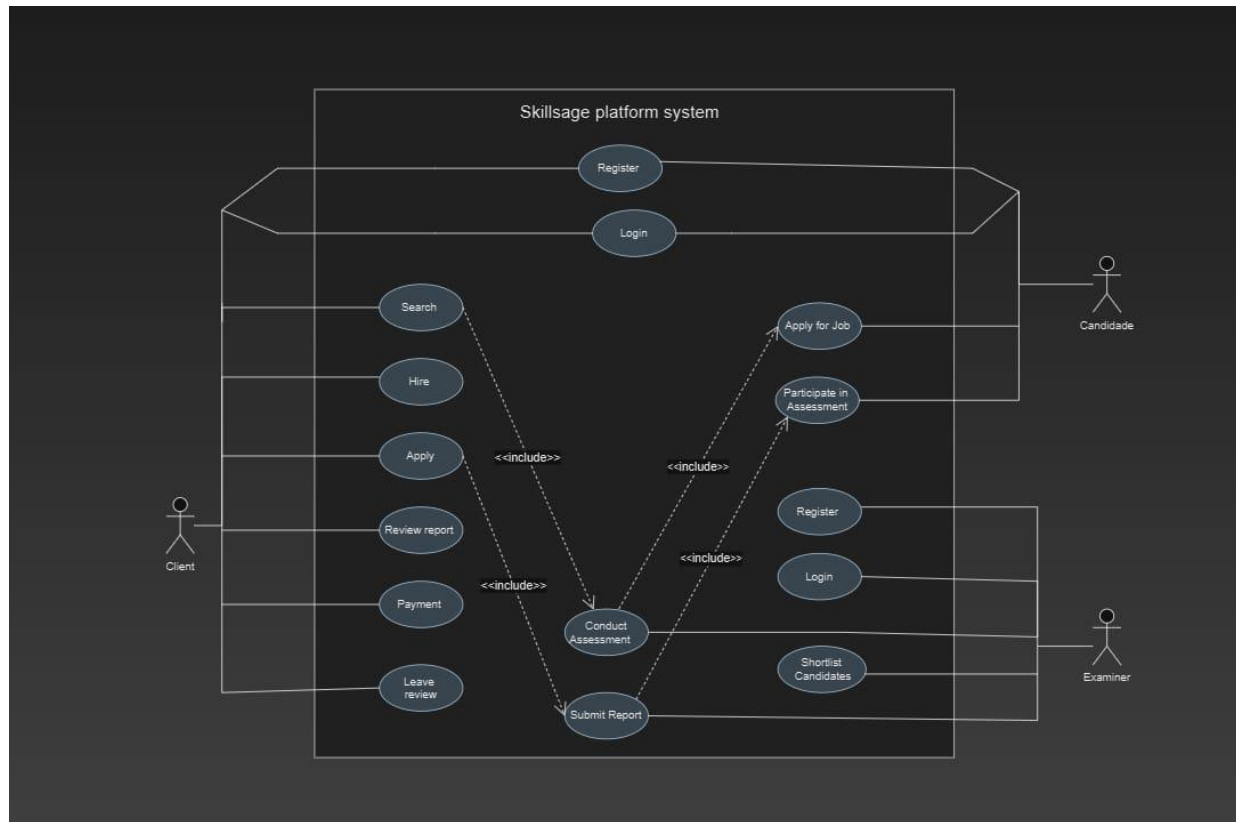
- **Description:** Sends a notification to a specified user about actions such as scheduling changes, assessment results, or payment confirmations.
- **Returns:** Confirmation of notification sent.

- **followUpFeedback(clientId, examinerId)**

- **Description:** Allows clients to provide follow-up feedback after an initial review, typically three months post-recruitment.
 - **Returns:** Confirmation of follow-up feedback or an error if the process fails.
-

These functions provide a comprehensive set of operations necessary to manage the SkillSage platform, covering all key user actions for clients, examiners, and candidates.

Use Case Diagram



Use Case Narratives:

1. Client Registers an Account

- **Goal:** Create a new account.
- **Outcome:** Account created and ready for login.

2. Client Searches for Examiners

- **Goal:** Find examiners by criteria.
- **Outcome:** Examiner selected for hire.

3. Examiner Sets Up Profile

- **Goal:** Create a profile.
- **Outcome:** Profile visible to clients.

4. Candidate Joins Interview

- **Goal:** Join scheduled interview.
- **Outcome:** Interview completed.

5. Client Pays Examiner

- **Goal:** Pay for interview.
- **Outcome:** Payment processed.

6. Admin Manages Accounts

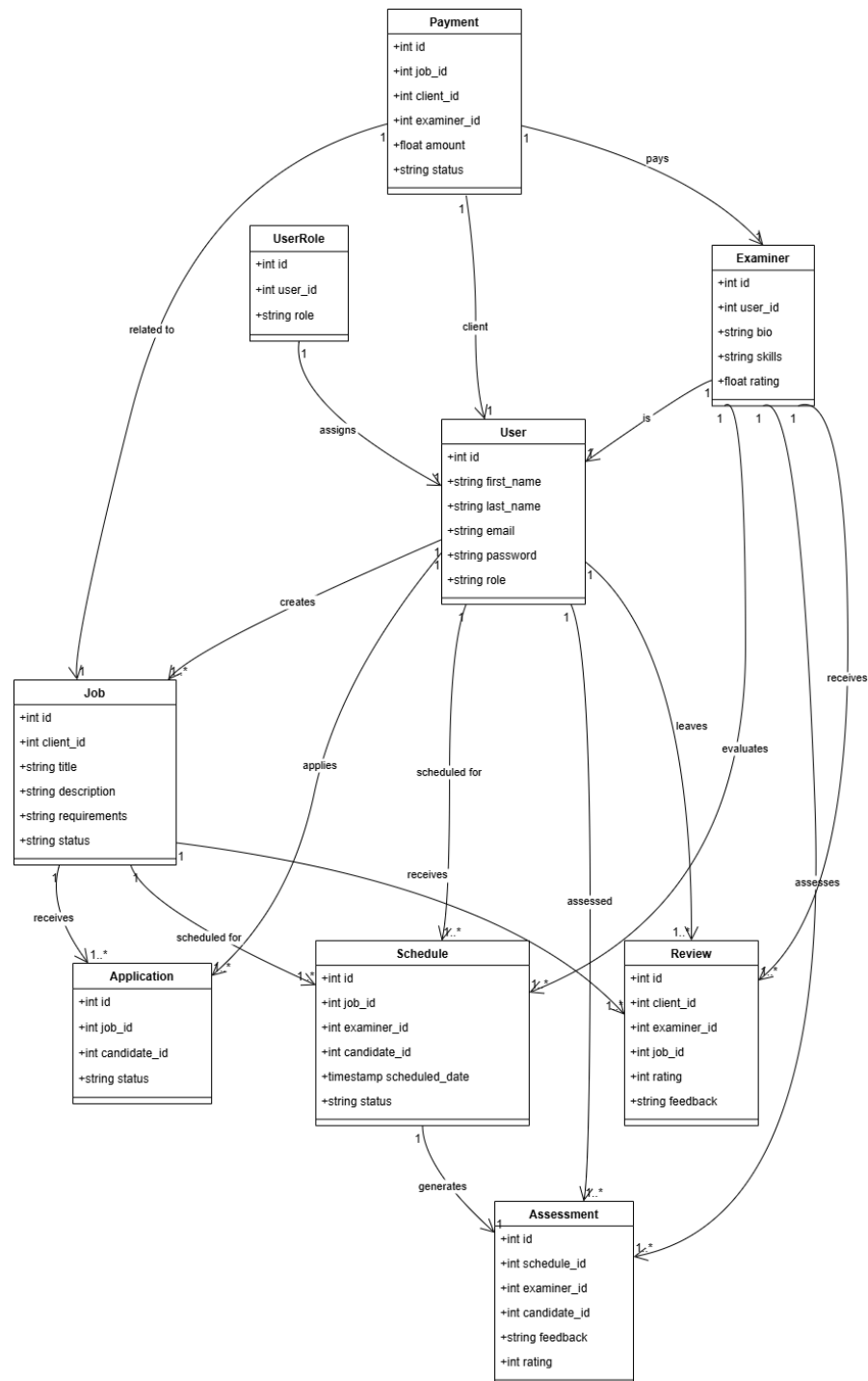
- **Goal:** Manage user accounts.
- **Outcome:** Accounts updated.

7. Admin Monitors Performance

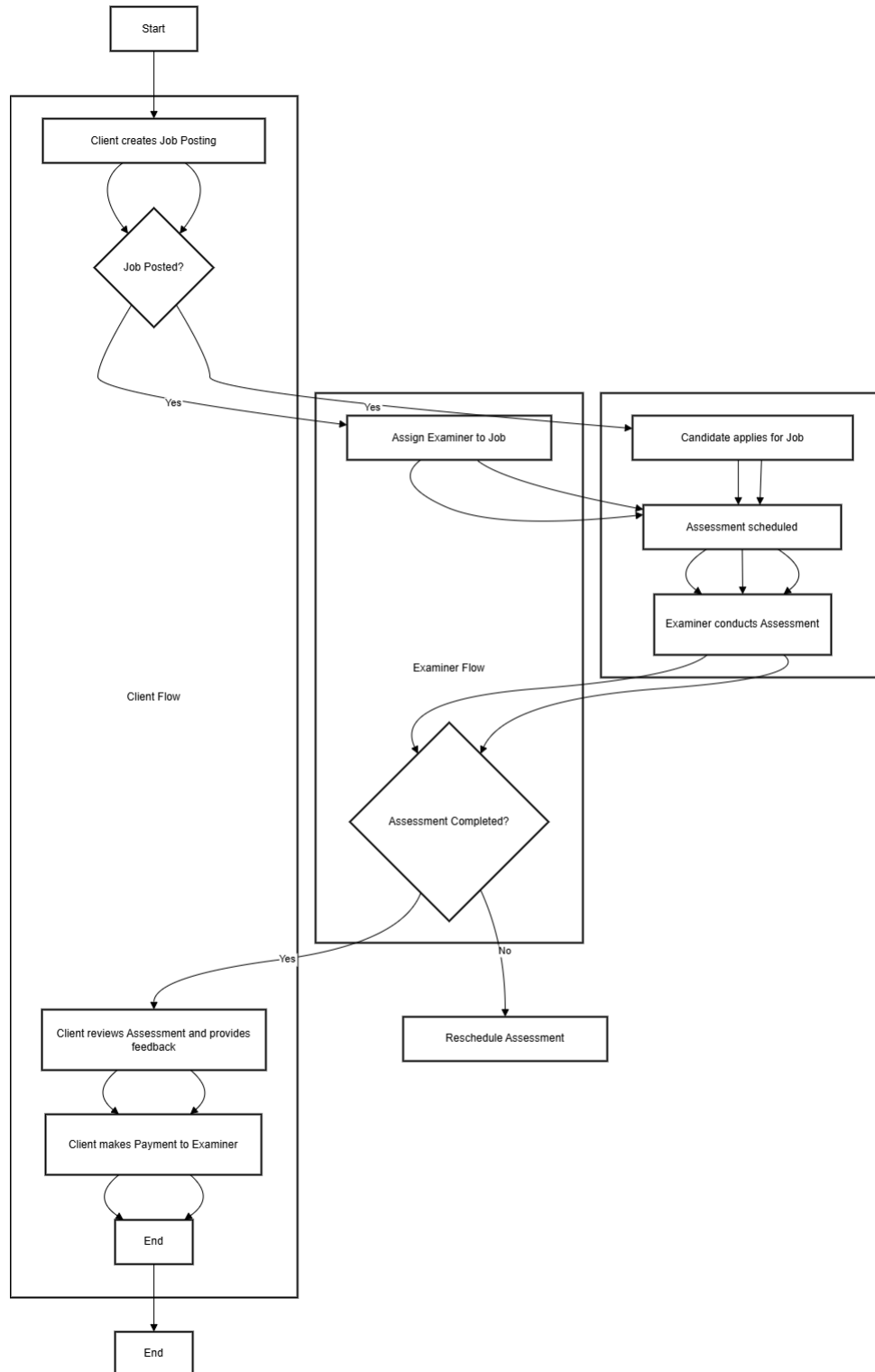
- **Goal:** Ensure platform is running smoothly.
- **Outcome:** Platform health maintained.

This is a highly condensed version of the key use cases.

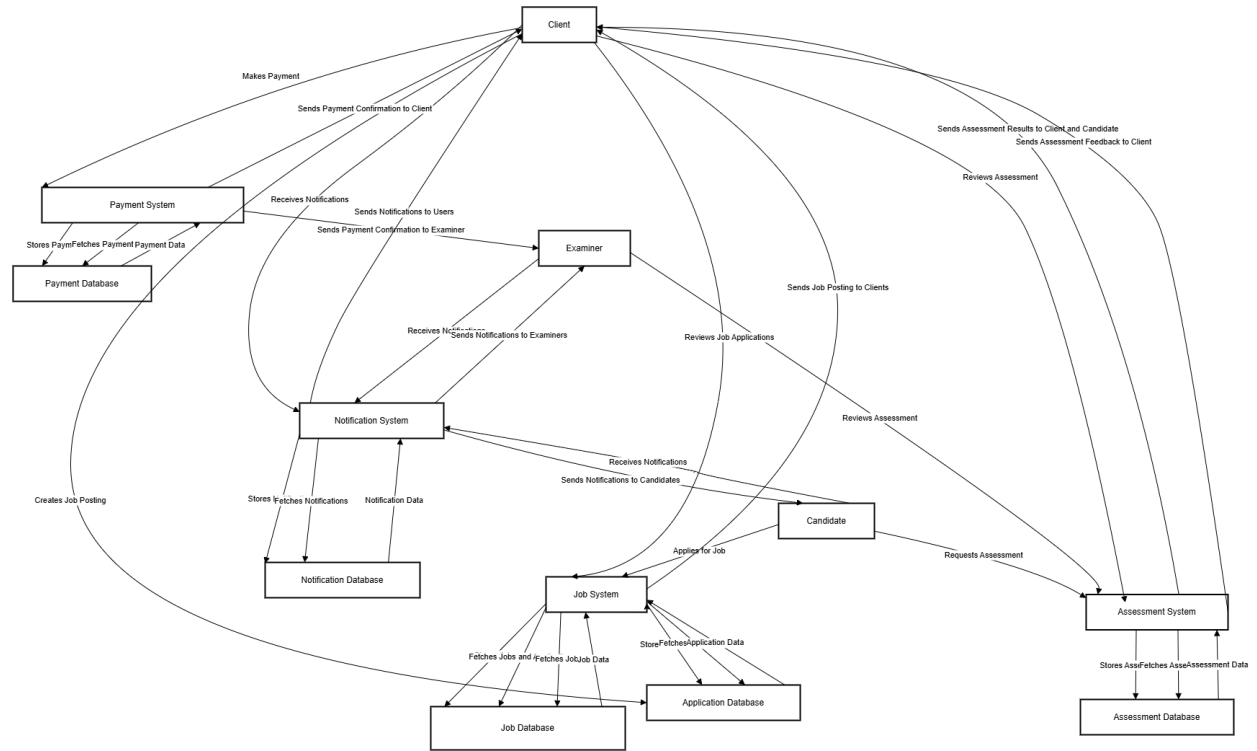
Class Diagram



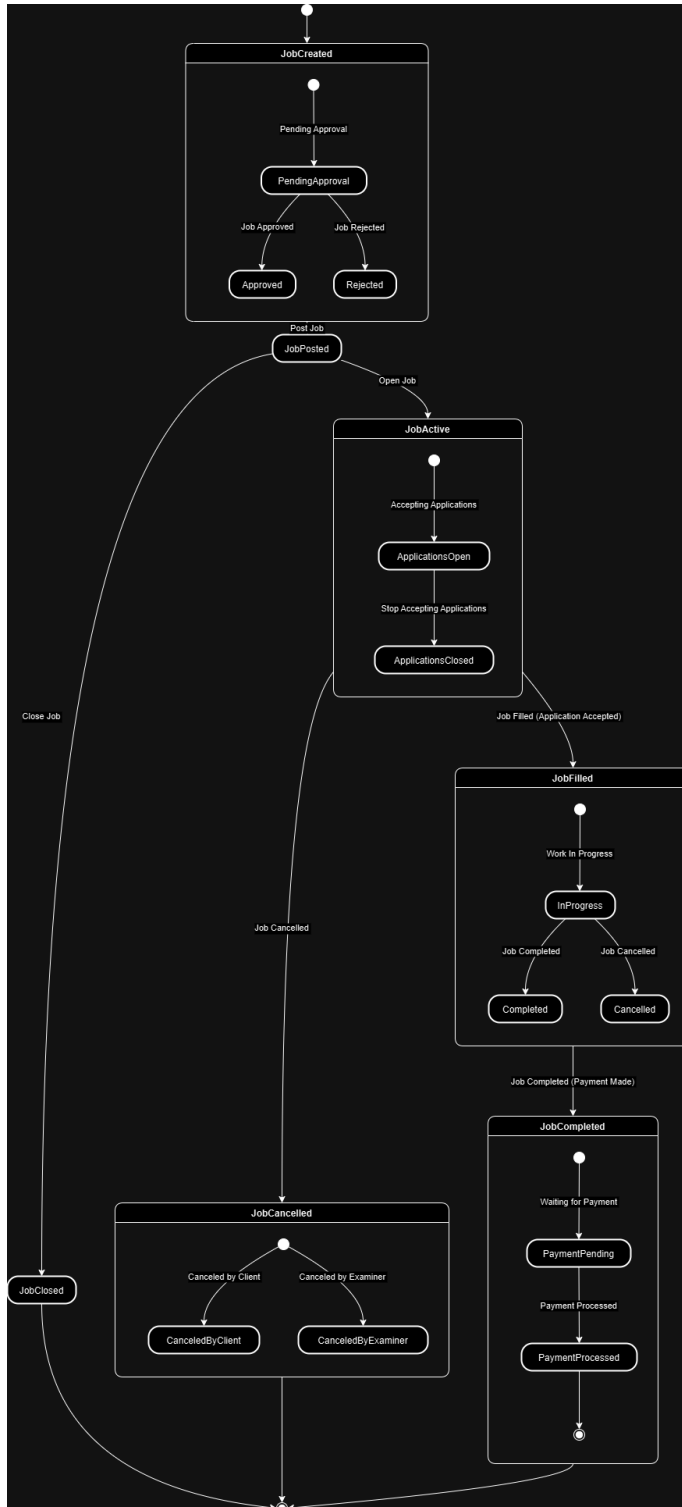
Activity Diagram



Data Flow Diagram



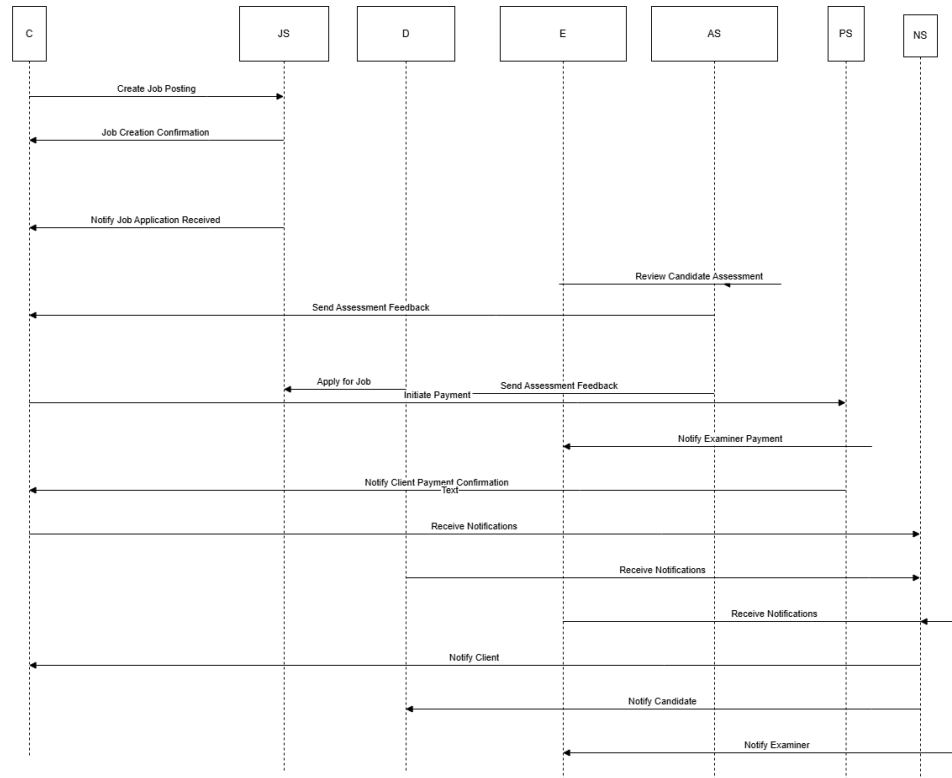
State Diagram



Sequence Diagram

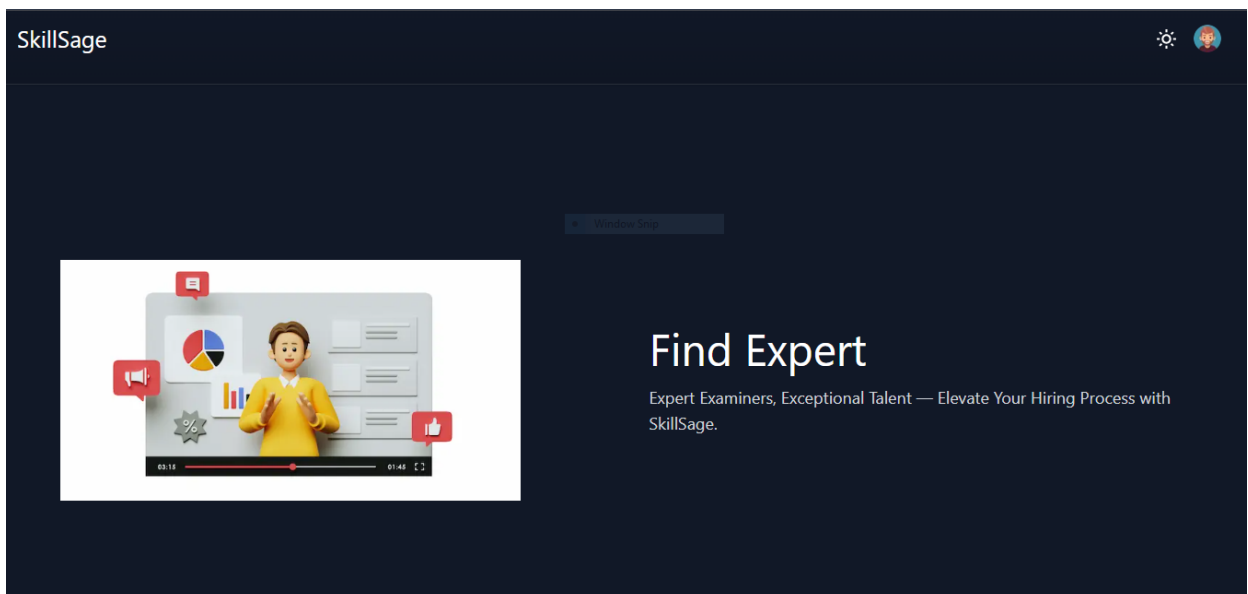
Sequence Diagram

- participant Client as C
- participant Candidate as D
- participant Examiner as E
- participant JobSystem as JS
- participant AssessmentSystem as AS
- participant PaymentSystem as PS
- participant NotificationSystem as NS



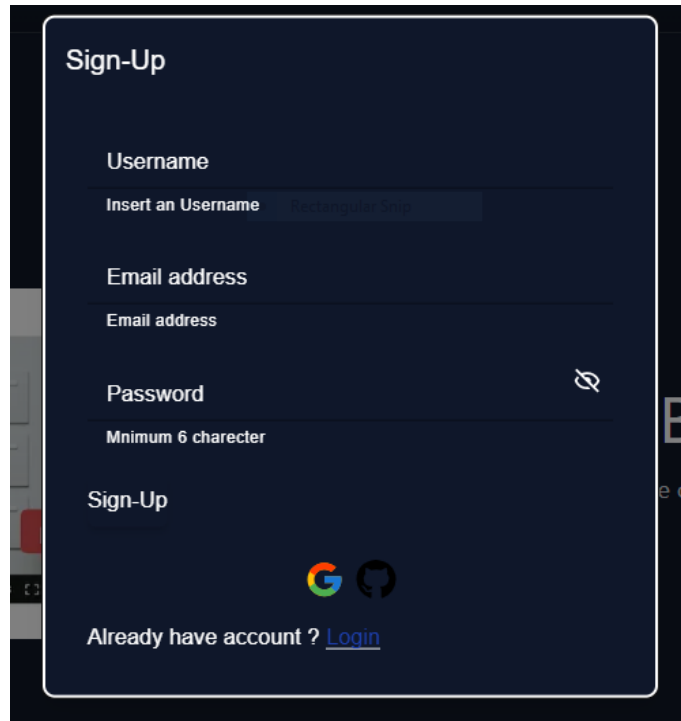
User Interface (UI) Design: Design wireframes and prototypes for the user interface to visualize the user experience and interaction.

User Home Page:



User Authentication module:

Sign-up module:



A dark-themed sign-up form with a white border. The title "Sign-Up" is at the top left. Below it are three input fields: "Username" with a placeholder "Insert an Username" and a "Rectangular Snip" button; "Email address" with a placeholder "Email address"; and "Password" with a placeholder "Mnimum 6 charecter" and a toggle icon. A "Sign-Up" button is at the bottom left. Below the button are the Google and GitHub logos. At the bottom, it says "Already have account ? [Login](#)".

Sign-Up

Username

Insert an Username Rectangular Snip



Email address

Email address

Password

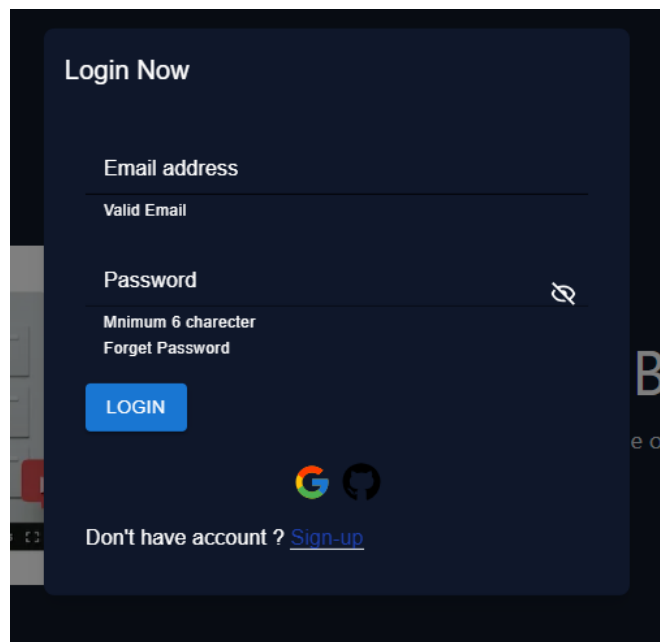
Mnimum 6 charecter

Sign-Up

Already have account ? [Login](#)

Login module:



A dark-themed login form with a white border. The title "Login Now" is at the top left. Below it are two input fields: "Email address" with a placeholder "Valid Email" and a "Valid Email" button; and "Password" with a placeholder "Mnimum 6 charecter" and a toggle icon. A "LOGIN" button is at the bottom left. Below the button are the Google and GitHub logos. At the bottom, it says "Don't have account ? [Sign-up](#)".

Login Now

Email address

Valid Email



Valid Email

Password

Mnimum 6 charecter

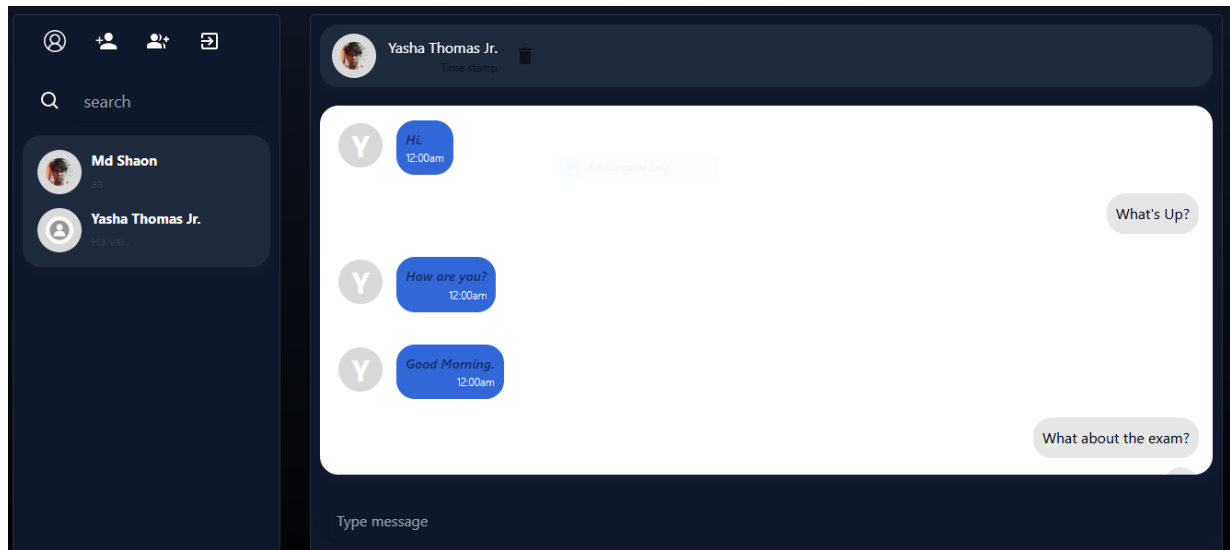
Forget Password

LOGIN

Don't have account ? [Sign-up](#)

Conversation Module



Evaluation Process

Application	Application review	Personal Interview	Documents & trial...	Background check	Hi	Edit
99	15	34	17	3		
5 candidates await next step						
Name		Status	In stage	Actions		
<input type="checkbox"/>	Alberta Flores	Completed	1 day	<input type="button" value="Reject"/>	<input type="button" value="Next stage"/>	<input type="button" value="More"/>
<input type="checkbox"/>	Brooklyn Simmons	Completed	1 day	<input type="button" value="Reject"/>	<input type="button" value="Next stage"/>	<input type="button" value="More"/>
<input type="checkbox"/>	Ralph Edwards	Completed	1 day	<input type="button" value="Reject"/>	<input type="button" value="Next stage"/>	<input type="button" value="More"/>
<input type="checkbox"/>	JC Jane Cooper	Completed	1 day	<input type="button" value="Reject"/>	<input type="button" value="Next stage"/>	<input type="button" value="More"/>
<input type="checkbox"/>	CW Cameron Williamson	Completed	1 day	<input type="button" value="Reject"/>	<input type="button" value="Next stage"/>	<input type="button" value="More"/>
<input type="checkbox"/>	Bessie Cooper	In-Progress	4 days	<input type="button" value="Reject"/>	<input type="button" value="Next stage"/>	<input type="button" value="More"/>
<input type="checkbox"/>	AM Arlene McCoy	In-Progress	4 days	<input type="button" value="Reject"/>	<input type="button" value="Next stage"/>	<input type="button" value="More"/>
<input type="checkbox"/>	Floyd Miles	In-Progress	4 days	<input type="button" value="Reject"/>	<input type="button" value="Next stage"/>	<input type="button" value="More"/>

Entities and Attributes

Here is a list of the main **entities** and their **attributes** for the **SkillSage** system based on the previously discussed schema and system components:

1. User

- id: Integer (Primary Key)
- first_name: Varchar (User's first name)
- last_name: Varchar (User's last name)
- email: Varchar (User's email, unique, not null)
- password: Varchar (User's password, not null)
- role: Varchar (Role of the user: client, examiner, or candidate)

2. Job

- id: Integer (Primary Key)
- client_id: Integer (Foreign Key to User.id for the client)
- title: Varchar (Job title)
- description: Text (Job description)
- requirements: Text (Job requirements)
- status: Varchar (Status of the job: open, closed)

3. Application

- id: Integer (Primary Key)
- job_id: Integer (Foreign Key to Job.id)
- candidate_id: Integer (Foreign Key to User.id for the candidate)
- status: Varchar (Application status: pending, shortlisted, or rejected)

4. Examiner

- id: Integer (Primary Key)
- user_id: Integer (Foreign Key to User.id, unique)
- bio: Text (Examiner's biography)
- skills: Text (Skills of the examiner)
- rating: Decimal(2,1) (Rating of the examiner, between 0 and 5)

5. Schedule

- id: Integer (Primary Key)
- job_id: Integer (Foreign Key to Job.id)
- examiner_id: Integer (Foreign Key to Examiner.id)
- candidate_id: Integer (Foreign Key to User.id)
- scheduled_date: Timestamp (Scheduled date for the interview)
- status: Varchar (Status: pending, completed, or canceled)

6. Assessment

- id: Integer (Primary Key)
- schedule_id: Integer (Foreign Key to Schedule.id)
- examiner_id: Integer (Foreign Key to Examiner.id)
- candidate_id: Integer (Foreign Key to User.id)
- feedback: Text (Feedback for the candidate)
- rating: Integer (Rating between 1 and 5)

7. Review

- id: Integer (Primary Key)
- client_id: Integer (Foreign Key to User.id for the client)
- examiner_id: Integer (Foreign Key to Examiner.id)
- job_id: Integer (Foreign Key to Job.id)
- rating: Integer (Rating between 1 and 5)
- feedback: Text (Feedback from the client)

8. Payment

- id: Integer (Primary Key)
- job_id: Integer (Foreign Key to Job.id)
- client_id: Integer (Foreign Key to User.id for the client)
- examiner_id: Integer (Foreign Key to Examiner.id)
- amount: Decimal(10,2) (Payment amount)
- status: Varchar (Payment status: pending, completed, or refunded)

9. UserRole

- id: Integer (Primary Key)
- user_id: Integer (Foreign Key to User.id)
- role: Varchar (Role: client, examiner, or candidate)

Normalized Schema

After normalization to **3NF**, the schema remains largely the same, but with minor considerations:

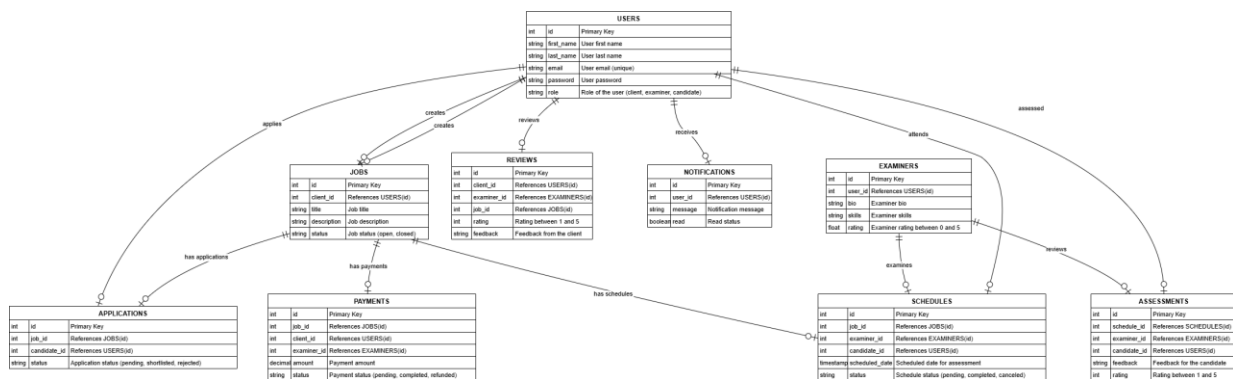
1. We have separated roles from the **Users** table and put them into the **UserRoles** table, removing redundancy and ensuring that role assignments are stored separately.
2. Tables like **Applications**, **Schedules**, **Assessments**, **Reviews**, **Payments**, and **UserRoles** all have primary keys and don't have redundant or transitive dependencies.

Here's a brief overview of the **normalized schema** in 3NF:

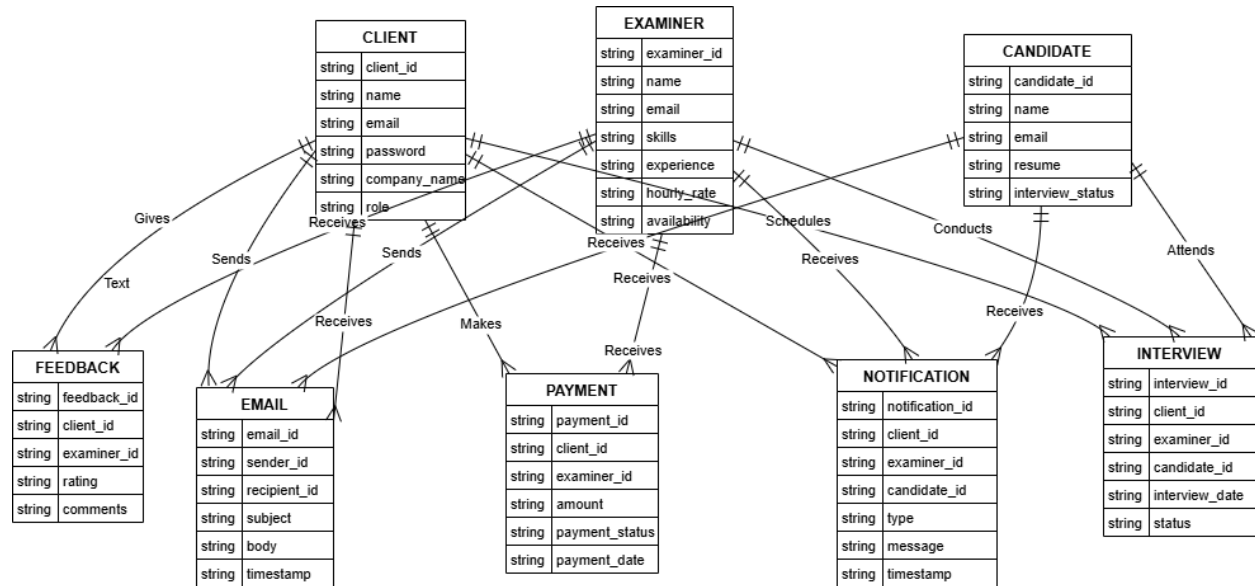
- **Users**
 - id (PK)
 - first_name
 - last_name
 - email
 - password
- **Jobs**
 - id (PK)
 - client_id (FK to Users)
 - title
 - description
 - requirements
 - status
- **Applications**
 - id (PK)
 - job_id (FK to Jobs)
 - candidate_id (FK to Users)
 - status
- **Examiners**
 - id (PK)
 - user_id (FK to Users)
 - bio
 - skills
 - rating
- **Schedules**
 - id (PK)
 - job_id (FK to Jobs)
 - examiner_id (FK to Examiners)
 - candidate_id (FK to Users)
 - scheduled_date
 - status
- **Assessments**
 - id (PK)
 - schedule_id (FK to Schedules)

- examiner_id (FK to Examiners)
 - candidate_id (FK to Users)
 - feedback
 - rating
- **Reviews**
 - id (PK)
 - client_id (FK to Users)
 - examiner_id (FK to Examiners)
 - job_id (FK to Jobs)
 - rating
 - feedback
- **Payments**
 - id (PK)
 - job_id (FK to Jobs)
 - client_id (FK to Users)
 - examiner_id (FK to Examiners)
 - amount
 - status
- **UserRoles**
 - id (PK)
 - user_id (FK to Users)
 - role

Schema Diagram



ERD (Entity Relationship Diagram)



Construction

Technical Architecture

1. System Architecture Diagram

- **Frontend:** React-based user interface providing a dynamic and responsive user experience.
- **Backend:** Node.js and Express.js for managing application logic, APIs, and server-side operations.
- **Database:** MongoDB for storing user profiles, assessments, and other data in a scalable and flexible NoSQL database.
- **Socket.IO Server:** Manages real-time communication, enabling features such as live updates, notifications, and chat functionality. **Diagram:** [Include a detailed diagram of the system architecture here]

Development Environment(Technology Stack)

1. MERN Stack Components

MongoDB

- **Role in SkillSage:** MongoDB is the NoSQL database for storing flexible data like user profiles, interviews, and transactions. It's scalable and supports complex queries for generating reports.
- **Features:** Schema-less design, horizontal scaling, aggregation framework, and indexing for improved performance.

Express.js

- **Role in SkillSage:** Express.js handles server-side logic and routing for APIs in the application, including secure authentication and user management.
- **Features:** RESTful routing, middleware for logging and error handling, integration with MongoDB for database interaction, and secure API endpoints.

React

- **Role in SkillSage:** React builds the user interface with a component-based architecture, ensuring a dynamic and responsive experience across dashboards and portals.

- **Features:** Component reusability, state management, React Router for navigation, and hooks for managing side effects and context.

Node.js

- **Role in SkillSage:** Node.js serves as the server-side runtime for building scalable, high-performance APIs and handling real-time interactions.
- **Features:** Non-blocking, event-driven architecture for handling concurrent requests and integration with Express and Socket.IO for real-time updates.

2. Redis (In-memory Caching)

- **Role in SkillSage:** Redis caches frequently accessed data like user sessions and examiner profiles, reducing database load and improving performance.
- **Features:** High-performance caching, session management, and pub/sub for real-time updates and notifications.

3. Socket.IO

- **Role in SkillSage:** Socket.IO enables real-time communication between the server and clients for features like live interview status updates and messaging.
- **Features:** Bidirectional communication, event-based messaging, real-time notifications, and integration with Redis for scaling.

Integration Overview

- **Frontend-Backend Communication:** React (frontend) uses REST APIs to communicate with Node.js/Express (backend) for data transactions, with Socket.IO used for real-time updates.
- **Backend-Database Communication:** Node.js interacts with MongoDB, and Redis is used to cache data and manage sessions.
- **Real-Time Features:** Socket.IO powers live notifications and updates for interviews, messages, and user availability.

This architecture ensures SkillSage is fast, scalable, and capable of providing real-time communication and seamless user experiences.

Deployment of SkillSage

a. Deployment Environment

SkillSage will be hosted on **AWS** or **Google Cloud** for scalability and reliability. The app will use **Node.js** for the backend, **Next.js** for the frontend, and **MongoDB Atlas** for database management. CI/CD tools like **GitHub Actions** will automate deployment, ensuring smooth updates and zero downtime. SSL certificates and firewalls will secure communication and protect data.

b. Rollout Plan

1. **Beta Release** (1-2 months): Limited release to collect feedback, fix bugs, and refine features.
2. **Full Release** (1 month): Public launch with full access to all users.
3. **Post-launch Support** (Ongoing): Monitor performance, fix issues, and roll out updates based on feedback.

c. AMC (Annual Maintenance Contract)

The AMC will cover:

- **Bug fixes** and **system updates**.
- **Performance optimization** and **user support**.
- **Database backups** and **security monitoring**.

d. Support and Maintenance

- **User support**: 24/7 helpdesk and knowledge base.
- **Technical support**: Regular server updates, system monitoring, and database optimization.
- **Security audits**: Regular vulnerability scans and patches.

SkillSage's deployment plan ensures a smooth launch, ongoing updates, and long-term system stability with robust support and maintenance.

In conclusion, the SkillSage platform represents a significant advancement in the recruitment industry by addressing the limitations of traditional assessment methods. By leveraging the MERN stack and integrating real-time communication via Socket.IO, SkillSage provides a robust, scalable, and dynamic solution for companies seeking to enhance their hiring processes.

Learning Experiences from SkillSage Development

1. **Full-Stack Development:** Gained hands-on experience with the **MERN stack** (MongoDB, Express.js, React.js, Node.js) for building the front-end and back-end of the app.
2. **UI/UX Design:** Worked on creating a responsive, user-friendly interface using **Next.js** and **Tailwind CSS**.
3. **Database Management:** Managed and optimized a **MongoDB Atlas** database for performance and scalability.
4. **Testing & Debugging:** Developed skills in **unit testing** and **integration testing** to ensure feature functionality.
5. **Performance Optimization:** Improved app performance with **load balancing**, **auto-scaling**, and React optimization.
6. **Security Best Practices:** Implemented **JWT authentication**, **SSL encryption**, and regular security audits.
7. **Deployment & CI/CD:** Automated deployments using **AWS** and **Docker**, ensuring smooth feature rollouts.
8. **User Feedback:** Focused on iterative improvements through continuous **user feedback** after launch.

Bibliography/References

Books

- Pressman, Roger S., and Bruce R. Maxim. *Software Engineering: A Practitioner's Approach*. 8th ed., McGraw-Hill Education, 2014.
 - Wieringa, Roel. *Design Science Methodology for Information Systems and Software Engineering*. Springer, 2014.
-

Conclusion

SkillSage is designed to be a comprehensive and scalable platform that bridges the gap between companies, freelance examiners, and candidates. By leveraging the MERN stack with Redis for caching and Socket.IO for real-time communication, SkillSage ensures a smooth, fast, and interactive experience for all users.

The integration of MongoDB, Express.js, React, and Node.js provides a robust foundation for handling dynamic user data, secure authentication, and efficient server-side logic. Redis enhances performance by caching frequently accessed data, while Socket.IO powers real-time notifications, enabling live updates and interaction between examiners, candidates, and clients.

This technology stack offers the flexibility and scalability necessary to handle growth and increasing user interactions, ensuring that SkillSage will be able to accommodate future requirements while delivering a seamless and efficient experience. With careful planning, iterative design, and thorough testing, SkillSage is positioned to become a reliable and innovative platform in the freelance examiner and interview scheduling space.