CS390 Software Engineering – Fall 2024 Assignment- part 3- Final document, and project deliverables

Team leader: ***Ahmed Abdelbary 2221191298***
Member 1: ***Lamiaa Abushara 2211150120***
Member 2: ***Seba Alzaher 2201114862***

Table of Contents

1.Introduction

This document is, hence, the final document on the development of the CSARCS (Computer Science Assistance Request and Credit System). The system was developed to aid computer science students in making academically related requests of one another in an organized fashion by creating a platform in which junior students can request help while the seniors, under the supervision of their faculty, would be able to provide mentoring.

This is a role-based access system whereby:
Junior Students submit assistance requests. Senior Students review the requests from junior students and have the option of accepting a request to provide academic support. Faculty Members oversee the process, manage sessions, and provide the necessary final approvals. The core functionalities of CSARCS are task management, session tracking, and feedback-collecting capabilities, which assure transparency and efficiency. CSARCS ensures user privacy and security via user authentication. The system has been designed based on different dashboards, namely the junior dashboard, senior dashboard, and faculty dashboard, ensuring end-user experience tailored to each role.

The entire lifecycle of CSARCS has been documented in this report, which includes requirements analysis for the developed system, design, implementation, testing, and deployment phases. The platform, bringing structured collaboration among students and faculty, enhances academic productivity.

2.Glossary

The list includes 35 definitions arranged in alphabetical order:

- **API (Application Programming Interface):** A set of rules that allows different parts of the system (or external systems) to interact with each other programmatically.
- **Authentication:** The process of verifying the identity of a user to grant them access to the system based on their role.
- **Authorization:** The process of ensuring that a user has the appropriate permissions to perform certain actions within the system.
- **Client-Server Model:** An architectural pattern where the client (e.g., a web browser) interacts with the server to perform actions and retrieve data.
- **Credit Points:** Extra points or marks given to final year students for properly playing the role of the mentor.
- **CS ARCS:** Computer Science Assistance Request and Credit System – a platform that facilitates peer-to-peer mentoring within the Computer Science department.
- **Dashboard:** A user interface element that provides an overview of the most relevant tasks, notifications, or statuses tailored to the user's role.
- **Faculty Member:** A supervising teacher in charge of the mentoring session’s quality output and adherence to the department’s regulations.

- **Faculty Supervision Console:** The interface used by faculty to oversee mentoring sessions, monitor progress, and allocate credits to senior mentors based on performance.

- **Junior Feedback:** A structured review submitted by Junior Students to evaluate the quality of tutoring assistance provided by Senior Students.

- **Junior Student:** A student in an earlier year of study (typically first or second year) who requests help on challenging subjects or topics.

- **Mentorship Assignment:** This is the process whereby junior students' requests for assistance are brought to assist eligible senior students according to their expertise and availability.

- **Mentorship Dashboard:** An interface from which senior mentors can see the requests they have been assigned, monitor the progress of their mentorship activities and communicate with junior students.

- **Notification:** A system-generated message that updates users about key events such as task assignment, completion, or feedback submission.

- **Peer-to-Peer Learning:** A learning approach where students learn from each other through mentorship and collaboration.

- **PostgreSQL:** A robust and scalable database recommended for the production environment of the system.

- **Progress Tracking:** A system characteristic whereby the users can see the progress and result of mentorship sessions.

- **Reliability:** The system's ability to function without disruptions, providing consistent access to assistance sessions.

- **Request Assistance Portal:** The application where junior students create and submit help requests indicating the subject and the form of help needed.

- **Responsiveness:** The adaptability of the system across various devices and screen sizes, allowing for easy access on desktops, tablets, and smartphones.

- **Role-Based Access Control (RBAC):** A security mechanism used to restrict system access to users based on their roles (e.g., Junior, Senior, Faculty).

- **Security:** Measures to protect user data and ensure access control within the system.

- **Senior Student:** A student of an betted year such as third year or fourth year whose role is to assist the junior students in a controlled environment and guidance too.

- **Session Feedback:** A feature allowing junior students to provide feedback on the assistance they received, which helps faculty assess mentor performance.

- **SQLite:** A lightweight database used during the development phase for managing structured data.

- **System Modularity:** The design principle of dividing the system into independent modules to improve maintainability and scalability.

- **System Scalability:** The ability of the system to handle increasing workloads, such as additional users or tasks, without a decline in performance.

- **Task:** A request for assistance created by Junior Students and fulfilled by Senior Students.

- **Task Allocation:** A system characteristic whereby requests are fairly shared among the qualified mentors to avoid injustice.

- **Testing:** The process of evaluating the system to ensure that it functions as expected and meets its requirements.

- **Token:** A secure digital key used for authenticating a user session, typically generated during login.

- **Traceability Matrix:** A document that maps test cases to their corresponding use cases to ensure all requirements are covered.

- **Use Case:** A specific scenario describing how a user interacts with the system to achieve a particular goal.

- **User:** Any individual accessing the system, including junior students, senior mentors, and faculty members.

- **User Interface (UI):** The part of the system that users interact with, including dashboards, forms, and notifications.

3.Customer Statement of Requirements (part 1)

Overview

As a client representing the Computer Science Department, we aim to implement a system that supports well-structured peer-to-peer learning. Junior students can ask for assistance with subjects that is difficult for them and senior students can provide guidance in a supervised environment. This program should be advantageous for all stakeholders involved (juniors, seniors, and faculty) and integrate into our CS department's operations.

Objectives

1. Facilitate Academic Support: We are keen to provide junior students with a user-friendly interface, where they can request for assistance in the specific difficult courses.

2. Encourage Senior Mentorship: The seniors should not only be the passive recipients of knowledge but instead be encouraged to serve as mentors to the juniors and in the process pass on their knowledge. By awarding additional credits for tutors, we envision creating a positive and nurturing

environment within our department.

3. Ensure Faculty Supervision: Faculty supervision of the interaction is essential so that the support is in accordance with departmental and academic integrity policies.

4. Create a User-Friendly System: The system should be user friendly to enable the students and even the faculty members to work on it without any difficulties. We envision a design that is simple with few instructions, easy task allocation and a straightforward follow up of the sessions.

Core Features

1. Request Assistance Portal:

   - Junior students should be able to submit with the chosen subject, topic and the kind of help that they seek.
   - Browse file / request specific link section would be a welcome addition as optional.

2. Mentorship Assignment:

   - Senior students who are available and qualified in particular subjects should receive notifications of relevant assistance requests.
   - Assigning should be proportionate so that each senior takes on a certain number of mentorships, promoting equity in the assigning of responsibilities.

3. Mentorship Dashboard:

   - Senior mentors should have to access a dashboard which allows them to see all the assignments that are active as well as the history of the mentors and the juniors under them.
   - Further provisions, like allowing people being mentored by juniors to rate the help they received and allowing seniors to have a courtesy of expressing their thoughts regarding the help they gave, would enrich the system.

4. Faculty Supervision Console:

   - Faculty members should be able to oversee all ongoing sessions, monitor conversation history, and view feedback to ensure quality control.
   - Faculty should have the ability to award credit points to seniors based on participation and feedback scores.

5. Progress Tracking and Reporting:

   - Requests and sessions progress should also be viewed both by juniors and seniors.

- Automated reports on mentorship hours and assistance provided should be accessible to faculty for assessment purposes.

4.System Requirements (part 1)

4.1. Functional Requirements

ID	Priority weight	Requirement description
F-01	5	System should authenticate users and differentiate roles. (Log in/sign in)
F-02	5	Junior students can create and manage task requests. Senior students should be able to browse and accept tasks.
F-03	4	System must verify senior students' tasks align with courses.
F-04	3	Track task status, documents exchanged, and scheduling details.
F-05	5	Faculty can review tasks and grant extra credits if standards met.
F-06	2	Notify users about task postings, acceptances, and completions.
F-07	4	Implement task prioritization and fair distribution mechanism
F-08	2	Enable feedback and rating system for tutoring sessions.
F-09	3	Allow faculty to manually override system decisions on credits
F-10	2	Enable document sharing and collaboration tools within tasks.
F-11	2	Provide a search function for users to find specific tasks.
F-12	3	Include a calendar system for scheduling and reminders.

4.2. Non-Functional Requirements

ID	Priority Weight	Requirement Description
NF-01	5	Ensure system availability 24/7 with minimal downtime.
NF-02	4	Interface should be user-friendly and tailored to each user role.

NF-03	5	System should be responsive on various devices including mobile.
NF-04	5	Secure user data and prevent unauthorized access.
NF-05	4	Ensure data integrity and prevent loss or corruption.
NF-06	4	Comply with accessibility standards to support all users.
NF-07	2	Integrate smoothly with existing university systems.
NF-08	4	Designed for easy maintenance and future updates.
NF-09	3	System should maintain a log of all activities for auditing.
NF-10	3	System updates should be deployable without downtime.
NF-11	3	Provide multilingual support for diverse student bodies.
NF-12	2	Ensure system's user interface adheres to the latest UI trends.

5. Functional Requirements Specification (part 1)

5.1. Stakeholders

Junior Students: These are users who create task requests in the system to get tutorial help on certain computer science topics. Their intention, most of the time, is to seek help in areas that are difficult for them.

Senior Students: These are users, whose enrolled courses allow them to accept task requests from juniors and provide tutorial help. Their aim is to help the juniors and earn some points for the help provided.

Faculty Members: Users whose duty is to oversee the quality of the assistance provided, ensuring it meets the department's standards, and giving scores to the senior students for the tasks done.

University Library System: An off the shelf system that tracks access to rooms for junior and senior students when they are attending classes as conclusive attendance records.

5.2. Actors

Junior Student: Commences requests for help regarding a given task and the subject and type of assistance required.

Senior Student: Attends to and takes task requests within his or her area of expertise, engages in tutoring, and is awarded bonus marks.

Faculty Member: Helps in the fulfilling of the tasks set, lawfully assesses the feedback of the sessions held, and assigns senior students with additional credit points.

Library Room Access System: A database system that logs students every time they enter controlled areas with the aim of ensuring that the students in question carried out the mentorship within approved university premises.

5.3. Use Cases

5.3.1. Use Case Diagram

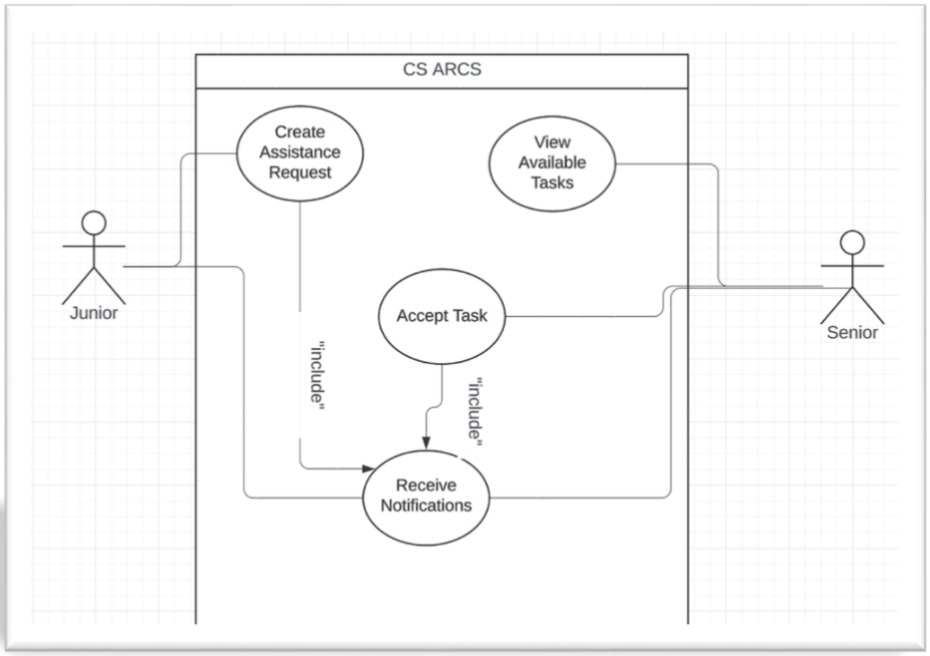


Fig. (5.1): Student Task Management Diagram:

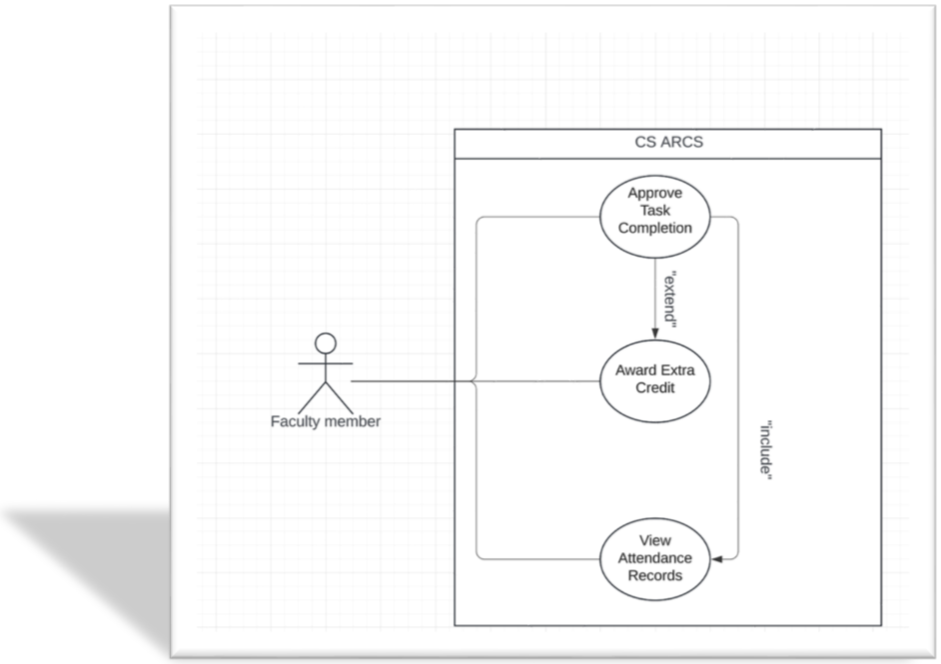


Fig. (5.2): Task Approval and Credit Allocation Diagram

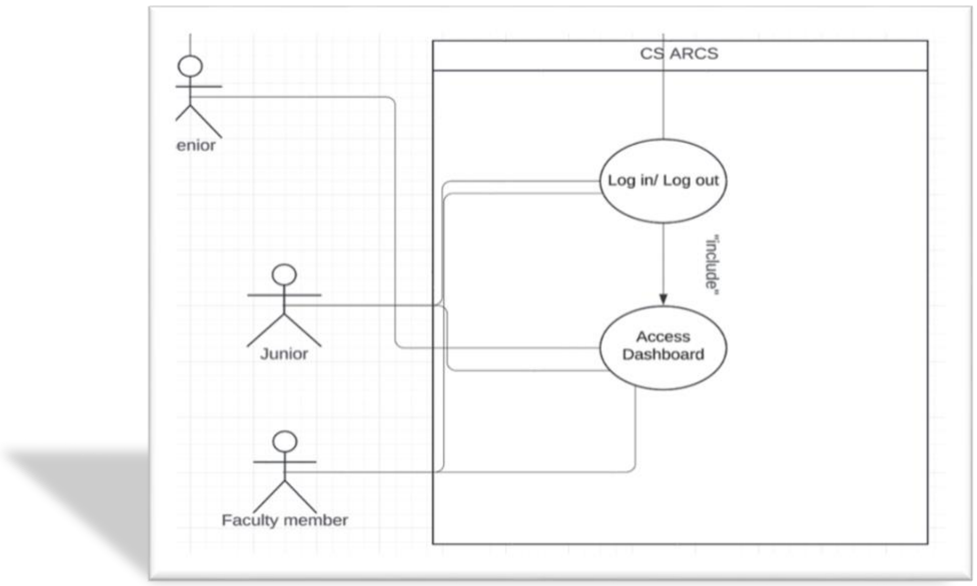


Fig. (5.3) : User Authentication and Access Control Diagram

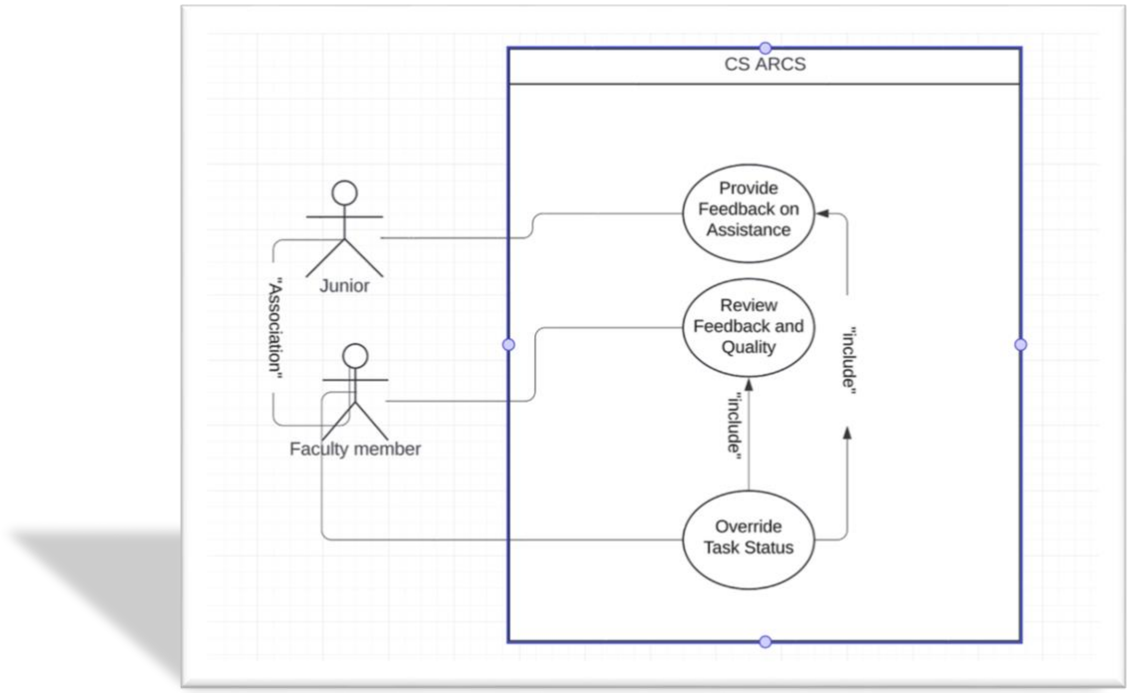


Fig. (5.4): Feedback and Quality Assurance Diagram

5.3.2. Use Case Summaries

ID	Name	Description
UC-1	Create Assistance Request	The Junior Student submits a request for tutoring, specifying the subject area and the type of guidance needed. This request is then posted to the system where eligible Senior Students can view and accept it.
UC-2	View Available Tasks	The Senior Student browses the list of open requests submitted by Junior Students. The system filters tasks based on the Senior Student's enrolled courses to ensure only relevant tasks are displayed.
UC-3	Accept Task	The Senior Student selects and accepts a task that aligns with their expertise. Once accepted, the task is assigned to the Senior Student, and

		notifications are sent to both the Junior Student and Faculty Member.
UC-4	Approve Task Completion	The Faculty Member reviews the completed task and assesses its quality. Based on the assessment, the faculty member approves or rejects the task completion and awards extra credit if the standards are met.
UC-5	Verify Room Access	This system use case records the room entry and exit times of both Junior and Senior students in designated library rooms, providing proof of attendance for the tutoring session.
UC-6	Provide Feedback on Assistance	After receiving tutoring, the Junior Student provides feedback on the session. This feedback is submitted to the Faculty Member for quality control purposes.
UC-7	Award Extra Credit	Based on the completed task and feedback, the Faculty Member awards extra credit to the Senior Student. This credit is linked to the student's coursework and incentivizes high-quality tutoring.
UC-8	Receive Notifications	The system sends notifications to Junior and Senior students when key events occur, such as when a task is posted, accepted, or completed. This ensures that users stay updated on task progress.
UC-9	Log In	The user (Junior, Senior, or Faculty) logs into the system, authenticating their role to gain access to relevant features. Each role has a unique dashboard, enabling specific functionalities aligned with their needs.

UC-10	Monitor Task Progress	The Faculty Member can view the status of all active tasks, including tutoring session schedules and progress updates, to ensure that tasks are on track and comply with departmental standards.
UC-11	Access Dashboard	users (Junior Student, Senior Student, Faculty Member) access a dashboard tailored to their role, where they can see relevant tasks, updates, or approvals.
UC-12	Review Feedback and Quality	Faculty Members review feedback submitted by Junior Students and assess the quality of the assistance provided by Senior Students. This use case supports quality control for awarded credits.

5.3.3. Traceability Matrix

	UC - 1	UC - 2	UC - 3	UC - 4	UC - 5	UC - 5	UC - 6	UC - 7	UC - 8	UC - 9	UC - 10	UC - 11	UC – 12
F-01										X			
F-02	X	X	X									X	
F-03			X										
F-04				X							X		
F-05				X				X					
F-06									X				
F-07		X									X		
F-08							X						X
F-09				X				X					X

F-10				X									
F-11		X											
F-12												X	

5.3.4. Fully dressed use case diagrams

UC-1: Create Assistance Request

Initiating Actor: Junior Student

Actor Goal: To submit a request for assistance on a specific course topic**.**

Stimulus: The Junior Student logs into the system and navigates to create a new assistance request.

Precondition: The Junior Student is authenticated and has access to the "Request Assistance" feature.

Postcondition: A new assistance request is created and visible to eligible Senior Students.

Flow of Events (Main Success Scenario):

1. Junior Student logs into the CS ARCS system.
2. Junior Student navigates to the "Request Assistance" page.
3. Junior Student specifies the course, topic, and type of help required.
4. Junior Student optionally uploads any relevant files or links.
5. Junior Student submits the assistance request.
6. The system saves the request and notifies eligible Senior Students.

Alternative Scenario(s):

Missing Information: If the Junior Student leaves the required fields blank, the system prompts them to fill in all mandatory details before submission.

UC-2: View Available Tasks

Initiating Actor: Senior Student

Actor Goal: To browse available assistance requests and choose tasks.

Stimulus: The Senior Student logs into the system and views the list of available tasks.

Precondition: The Senior Student is authenticated and has access to browse tasks.

Postcondition: The Senior Student has reviewed available tasks and can accept an appropriate one.

Flow of Events (Main Success Scenario):

1. Senior Student logs into the CS ARCS system.
2. Senior Student navigates to the "Available Tasks" section.
3. The system displays tasks that match the Senior Student’s courses.
4. Senior Student reviews the list of tasks and selects a suitable one for assistance.

Alternative Scenario(s):

No Matching Tasks: If no tasks are available that match the Senior Student’s courses, the system displays a message indicating no suitable tasks.

UC-3: Accept Task

Initiating Actor: Senior Student

Actor Goal: To accept a task and commit to provide assistance.

Stimulus: The Senior Student selects a task from the available tasks list.

Precondition: The Senior Student is authenticated and has permission to accept the task.

Postcondition: The task is assigned to the Senior Student, and the Junior Student is notified.

Flow of Events (Main Success Scenario):

1. Senior Student selects a task from the "Available Tasks" section.
2. Senior Student clicks on "Accept Task."
3. The system verifies the Senior Student’s enrollment in the relevant course.
4. The system assigns the task to the Senior Student.
5. The system notifies the Junior Student that their task has been accepted.

Alternative Scenario(s):

Task Already Taken: If another Senior Student has already accepted the task, the system displays an error and removes it from the available list.

UC-4: Approve Task Completion

Initiating Actor: Faculty Member

Actor Goal: To review completed tasks and award extra credit if standards are met.

Stimulus: Faculty Member logs into the system and reviews a completed task.

Precondition: The task is marked as "Completed" by the Senior Student and ready for faculty review.

Postcondition: The task is approved, and extra credit is awarded to the Senior Student.

Flow of Events (Main Success Scenario):

1. Faculty Member logs into the CS ARCS system.
2. Faculty Member navigates to the "Review Completed Tasks" section.
3. Faculty Member selects a task for review.
4. Faculty Member reviews the task details, session logs, and feedback.
5. If the task meets standards, Faculty Member approves the task and awards extra credit.
6. The system updates the task status to "Approved" and notifies the Senior Student.

Alternative Scenario(s):

Does Not Meet Standards: If the task does not meet standards, Faculty Member marks it as "Requires Improvement," and the system notifies the Senior Student.

UC-5: Verify Room Access

Initiating Actor: Library Room Access System

Actor Goal: To verify that tutoring sessions occur in designated library rooms.

Stimulus: Junior and Senior students access the library room for a tutoring session.

Precondition: Junior and Senior students are registered for a tutoring session.

Postcondition: Attendance is recorded in the system for both students.

Flow of Events (Main Success Scenario):

1. Junior and Senior Students arrive at the designated library room.
2. Each student scans their ID upon entering the room.
3. The library system records the entry time and confirms both students' presence.

4. The system saves attendance data as proof of participation for faculty review.

Alternative Scenario(s):

No Match Found: If either student does not scan in, the system logs incomplete attendance for the session.

UC-6: Provide Feedback on Assistance

Initiating Actor: Junior Student

Actor Goal: To provide feedback on the assistance received from the Senior Student.

Stimulus: The Junior Student completes a session with the Senior Student.

Precondition: The assistance session has concluded.

Postcondition: Feedback is recorded and available for faculty review.

Flow of Events (Main Success Scenario):

1. Junior Student logs into the CS ARCS system.
2. Junior Student navigates to "My Tasks" and selects the completed session.
3. Junior Student rates the assistance and leaves additional comments.
4. The system records the feedback and notifies the Faculty Member.

Alternative Scenario(s):

Skip Feedback: If the Junior Student skips feedback, the system sends a reminder after a set time.

UC-7: Award Extra Credit

Initiating Actor: Faculty Member

Actor Goal: To award extra credit based on task completion and quality.

Stimulus: Faculty Member reviews completed tasks with satisfactory ratings.

Precondition: Task has been reviewed and marked as approved.

Postcondition: Extra credit is added to the Senior Student's record.

Flow of Events (Main Success Scenario):

1. Faculty Member reviews the approved task.

2. Faculty Member assigns extra credit points.
3. The system updates the Senior Student's record with extra credit awarded.
4. The Senior Student is notified of the extra credit.

Alternative Scenario(s):

Credit Denied: If feedback is unsatisfactory, Faculty Member denies extra credit, and the system notifies the Senior Student.

UC-8: Receive Notifications

Initiating Actor: Junior Student, Senior Student

Actor Goal: To stay updated on task status changes.

Stimulus: A relevant event occurs (task posting, acceptance, or completion).

Precondition: The student has an active profile in the system.

Postcondition: Notification is sent to the relevant student(s).

Flow of Events (Main Success Scenario):

1. System detects a task status change.
2. System sends a notification to the relevant students (Junior/Senior).
3. Students receive the notification on their dashboard or via email (if enabled).

Alternative Scenario(s):

Notification Disabled: If notifications are disabled, students can check status manually on their dashboard.

UC-9: Log In

Initiating Actor: Junior Student, Senior Student, Faculty Member

Actor Goal: To securely access the system and view role-specific functionalities.

Stimulus: The user opens the CS ARCS login page and enters their credentials.

Precondition: The user has a valid account in the system.

Postcondition: The user is logged in and directed to their specific dashboard (Junior, Senior, or Faculty).

Flow of Events (Main Success Scenario):

1. User opens the CS ARCS login page.
2. User enters their username and password.
3. The system verifies the credentials.
4. If credentials are correct, the system logs in the user and directs them to their specific dashboard.
5. The user gains access to functionalities specific to their role.

Alternative Scenario(s):

Invalid Credentials: If the entered credentials are incorrect, the system displays an error message and prompts the user to try again.

Forgot Password: If the user cannot remember their password, they can initiate a password reset.

UC-10: Monitor Task Progress

Initiating Actor: Faculty Member

Actor Goal: To track the progress and status of all active tasks.

Stimulus: Faculty Member logs into the system and navigates to the "Monitor Task Progress" section.

Precondition: Faculty Member is authenticated and logged in.

Postcondition: The Faculty Member has a clear view of all ongoing tasks and their current statuses.

Flow of Events (Main Success Scenario):

1. Faculty Member logs into the CS ARCS system.
2. Faculty Member navigates to "Monitor Task Progress."
3. The system displays a list of all active tasks, with information on task status, assigned students, and scheduling details.
4. Faculty Member reviews task statuses and can drill down into specific task details if needed.

Alternative Scenario(s):

No Active Tasks: If there are no active tasks, the system displays a message indicating that all tasks are complete or awaiting approval.

UC-11: Access Dashboard

Initiating Actor: Junior Student, Senior Student, Faculty Member

Actor Goal: To access a personalized dashboard with relevant options based on their role.

Stimulus: The user logs into the system and is directed to their dashboard.

Precondition: User is successfully logged into the system.

Postcondition: The user sees role-specific functionalities and options on their dashboard.

Flow of Events (Main Success Scenario):

1. User logs into the CS ARCS system.
2. The system identifies the user's role (Junior Student, Senior Student, or Faculty Member).
3. The system directs the user to their role-specific dashboard.
4. The dashboard displays relevant options such as task requests (Junior), available tasks (Senior), or task review (Faculty).

Alternative Scenario(s):

Role-Switching (Admin Only): If there's an administrative role with access to multiple dashboards, the user can switch between views

UC-12: Review Feedback and Quality

Initiating Actor: Faculty Member

Actor Goal: To review feedback submitted by Junior Students on the assistance provided and assess the quality of completed tasks.

Stimulus: The Faculty Member navigates to the task feedback section.

Precondition: Completed tasks have feedback submitted by Junior Students.

Postcondition: Feedback is reviewed, and task quality is assessed.

Flow of Events (Main Success Scenario):

1. Faculty Member logs into the CS ARCS system.
2. Faculty Member navigates to the "Review Feedback and Quality" section.
3. Faculty Member selects a task with completed feedback.
4. Faculty Member reviews the Junior Student's feedback and any notes on the assistance session.
5. Faculty Member assesses the quality and decides if extra credit should be awarded or if further follow-up is needed.

Alternative Scenario(s):

Negative Feedback: If feedback is negative, the Faculty Member may mark the task as "Requires Improvement" and request a follow-up from the Senior Student.

5.4. System sequence diagrams

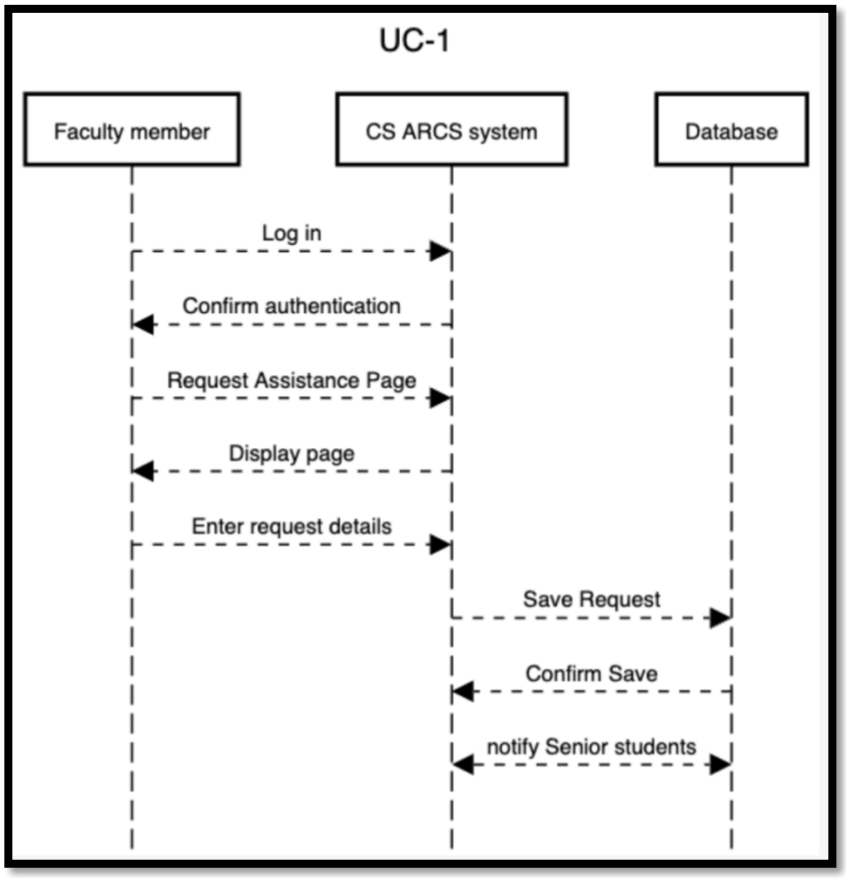


Fig. (5.5): sequence diagram for UC-1

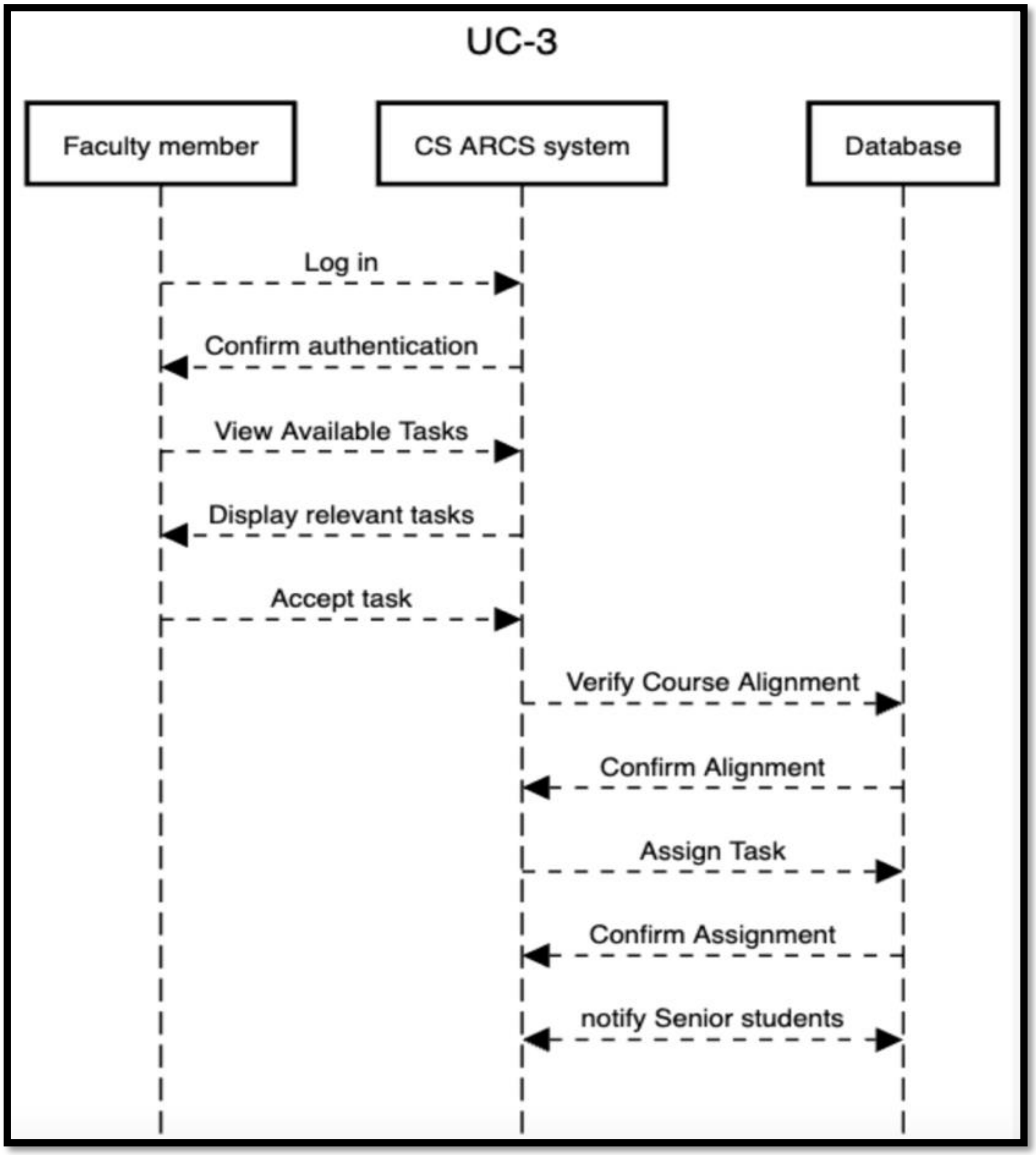


Fig. (5.6): sequence diagram for UC-3

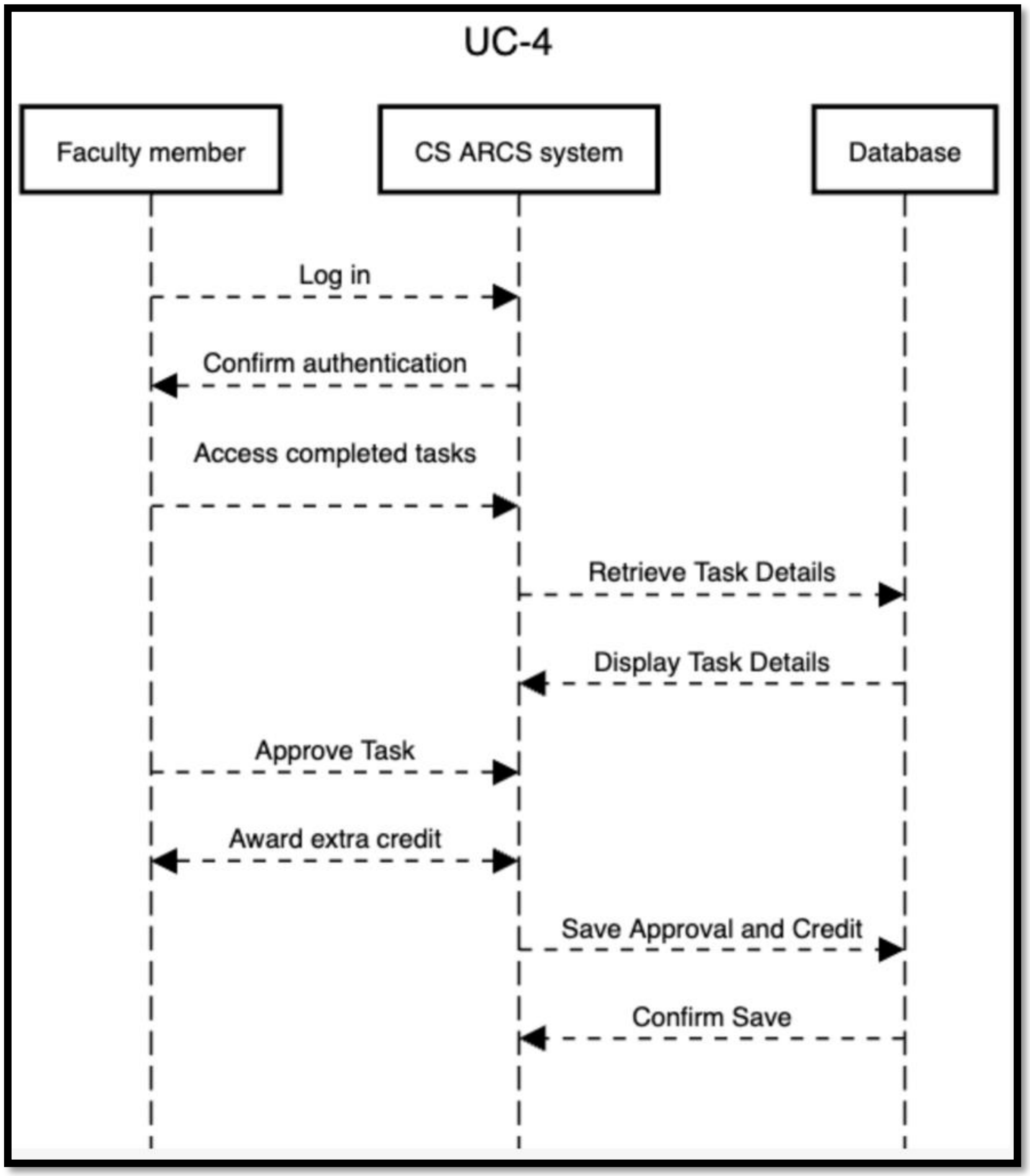


Fig. (5.7): sequence diagram for UC-4

6.System Architecture and System Design (part 2)

6.1. Architectural Styles

In our CS ARCS system, we utilize layered architectural pattern and client-server architecture. These styles fulfill the system's non-functional properties such as scalability, modularity, and maintainability.
The client-server model divides the system into its basic two parts:
Client: Users (e.g. Faculty, juniors and seniors) interacting with the system through a web interface
Server: backend system responsible for processing tasks, managing data and assuring data

integrity

The client-server model satisfies the non-functional system requirements such as:

- **Scalability (NF-01, NF-03):** The client-server design guarantees that the server can support multiple clients concurrently, ensuring that it can be accessed at any time and for long periods with very little downtime. additional servers can be deployed to handle the increased load.
- **Security (NF-04, NF-05):** the core operations like authentication and data storage are executed at the server side, protecting much of the sensitive information from exposure.
- **Responsiveness (NF-03):** tailored responses are given to a variety of client devices including mobile ones by utilizing server-side processing.

The three-layered system architecture eliminates complication and divides the system into three distinctive layers:

- **Presentation Layer:** This is user specific element of the system design which focuses on making the system functional and accessible to the users promptly. Moreover, it presents controls and interfaces, according to different users' (juniors, seniors, faculties) to the system as per the specification of NF-02.
- **Business Logic Layer:** processes the main features of the application, like managing tasks, notifications and submitting feedback.
- **Data Layer:** Manages storage, ensuring data integrity and availability.

This layered approach helps to meet the system non-functional requirements such as the following:

Modularity and Maintainability (NF-08): Each layer is independent, making it easier to implement updates or changes to specific parts of the system without impacting others.

Accessibility Standards (NF-06): The presentation layer has inbuilt features that allow it to meet the benchmarks set by the standards for the ease of use of the systems for all types of users.

Future Proofing (NF-10): It is possible to introduce changes in layers one at a time making it less disruptive.

Ability to Keep Track of Changes and Review Them (NF-09): Specific logs are kept in the said layer to make sure that all users of the system can be traced and their actions logged.

6.2. Subsystems

CS ARCS consists of many subsystems:

Authentication Subsystem:

Manages user logins, signups and assigns users to roles of junior, senior and faculty.

Concerned with checking users' identities, managing users' permissions, and preserving user sessions safely.

Task Management Subsystem:

Allows junior students to create tasks, senior students to accept and complete them, and faculty to review.

In charge of task creation/assignment, monitoring the task completion and ensuring internal equity in tasks distribution among seniors.

Admin/Faculty Console Subsystem:

Provides faculty members with tools to monitor tasks, review feedback, and award credits.

Responsible for displaying task statuses for review and enabling manual overrides for task assignments and credit allocation.

Notification Subsystem:

Notifies users of task updates, feedback reminders, and system changes.

Responsible for delivering in-app and email notifications and tracking notifications sent to users for auditing purposes.

Feedback Subsystem:

Collects feedback from junior students and allows faculty to review it for quality assurance.

Responsible for enabling juniors to rate their experience with tutors and providing tools for faculty to review feedback and award extra credits.

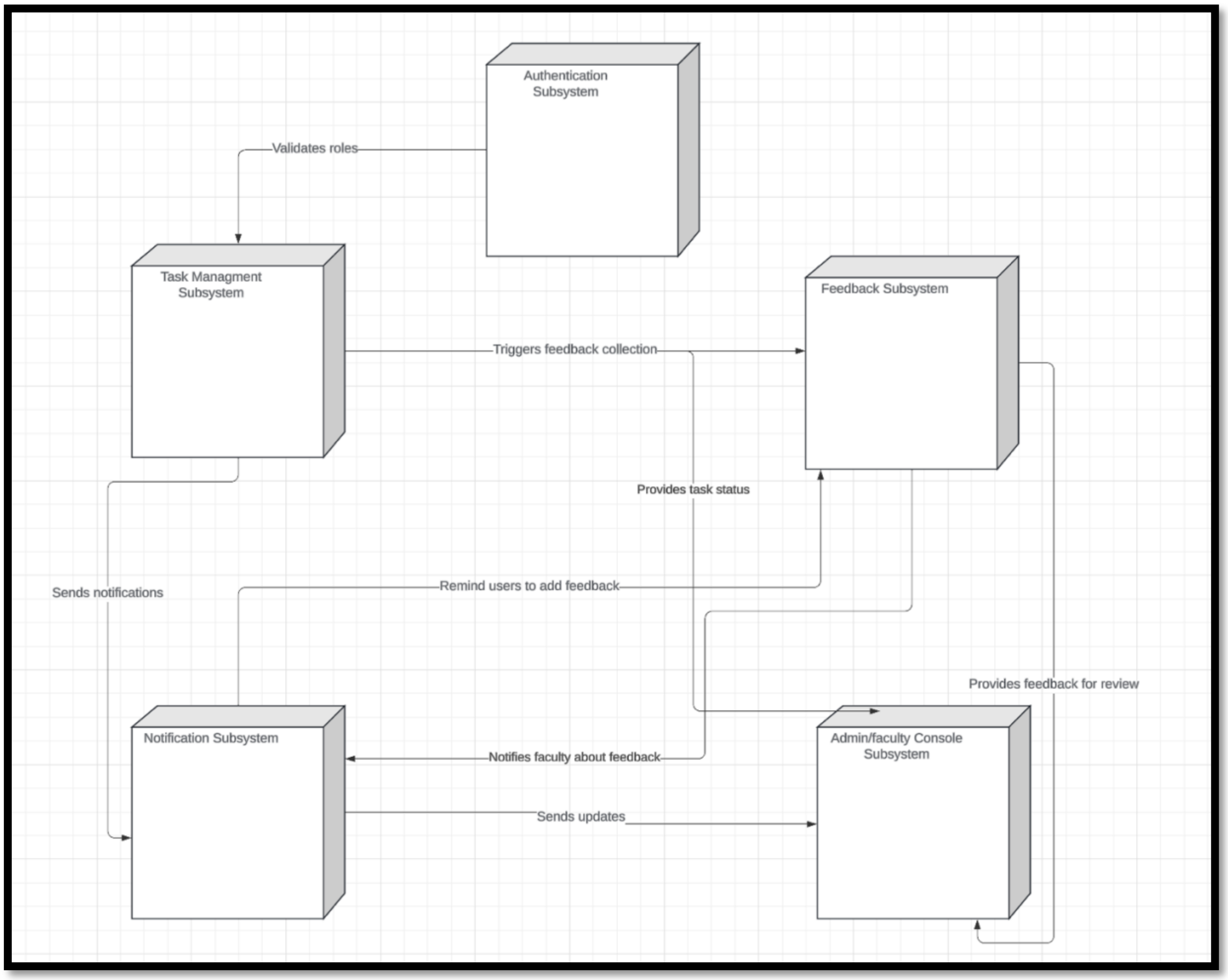


Fig. (6.1): UML diagram explaining the subsystems and their interactions with each other

6.3. Mapping subsystems to hardware

First, we will need to identify the hardware components that will be needed in this mapping for our CS ARCS system:

- Client devices such as computers, mobile phones and tablets
- Application server
- Database server
- Email server (for sending email notification to users)

• Subsystem	Hardware Component	Reason

Authentication Subsystem	Application Server	Handles user authentication (e.g., login, role validation) and manages secure user sessions.
Task Management Subsystem	Application Server and Database Server	Processes task creation/updates on the application server and stores tasks persistently in the database.
Feedback Subsystem	Application Server and Database Server	Collects feedback on the application server and saves feedback data in the database for admin review.
Notification Subsystem	Application Server, Email Server, and Client Devices	Sends notifications to users via the application server; email server handles external notifications (optional).
Admin/Faculty Console Subsystem	Client Devices (faculty desktops/laptops) and Application Server	Displays feedback and task statuses using a browser on client devices; connects to the server for logic processing.

Table (6): subsystems mapped to hardware components and the reason

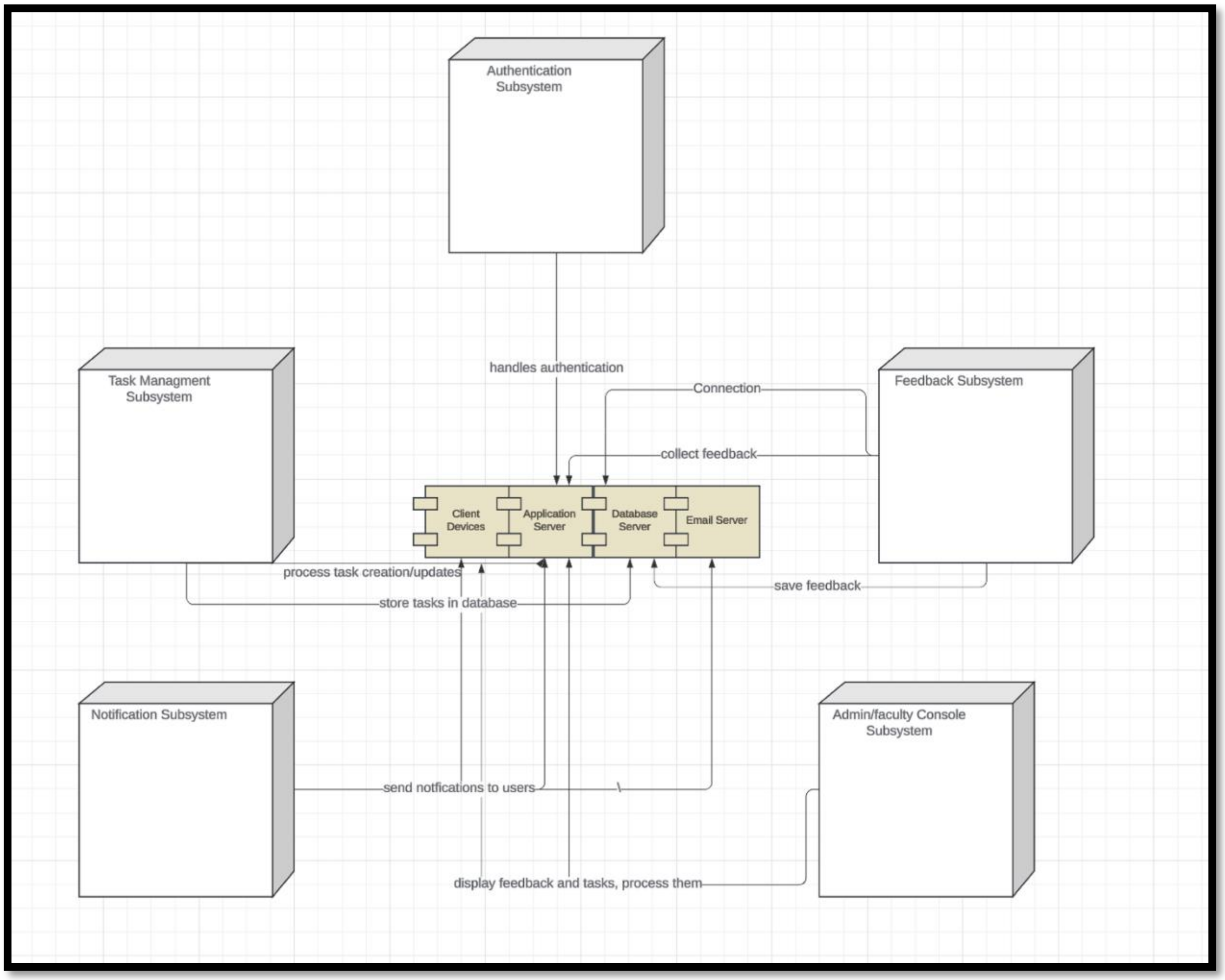


Fig. (6.2): UML diagram showing the mapping of subsystems to hardware components

6.4. Persistent data storage

Our CS ARCS system uses a relational database to store structured data. During development, we will use SQLite, and for production, we recommend PostgreSQL for scalability and robustness. We chose SQLite for development database because it is light-weight and file-based (does not require a server setup). Also, we choose PostgreSQL for production database because it is secure and supports advanced queries.

The system maintains the following main entities in the database:

1. **User table** which stores user details and role. It consists of the following fields**:**

   Id: identifier for each user

   Username: name of the user

   Email: email of the user

Password: encrypted user password

Role: junior/senior/faculty

2. **Tasks table tracks tasks created by juniors and accepted bt seniors. It consists of the following fields:**

    Id: identifier for each task

    Title: title of the task

    Description: brief description of the task

    Request_id: id of the junior who requested this task

    Accept_id: id of the senior who accepted this task

    Status: open/assigned/complete

    Time_created: Timestamp when the task was created

    Time_updated: Timestamp for the last update to the task

3. **Feedback table: Stores feedback data submitted by juniors about seniors and tasks**

    Id: identifier for each feedback

    task_id: id of the task the feedback is about

    user_id: id of the user the feedback is about

    rating: numeric rating (like the five-star system)

    comments: Additional comments about the task or the senior

4. **Notifications Table: Logs notifications sent to users. It consists of the following fields:**

    Id: identifier for each notification

    User_id: the id of the user the notification will be sent to

    Message: content of the notification

    Time_created: timestamp when the notification was sent

The database supports the following operations:

1. **User management:** create/retrieve/update/delete/assign role
2. **Task management:** add/delete/update/track status
3. **Feedback management:** collect/store/retrieve
4. **Notification:** log/retrieve

We also must ensure data integrity and reliability. We will achieve this through regular backups (for SQLite local file-based backups & cloud backups for PostgreSQL).

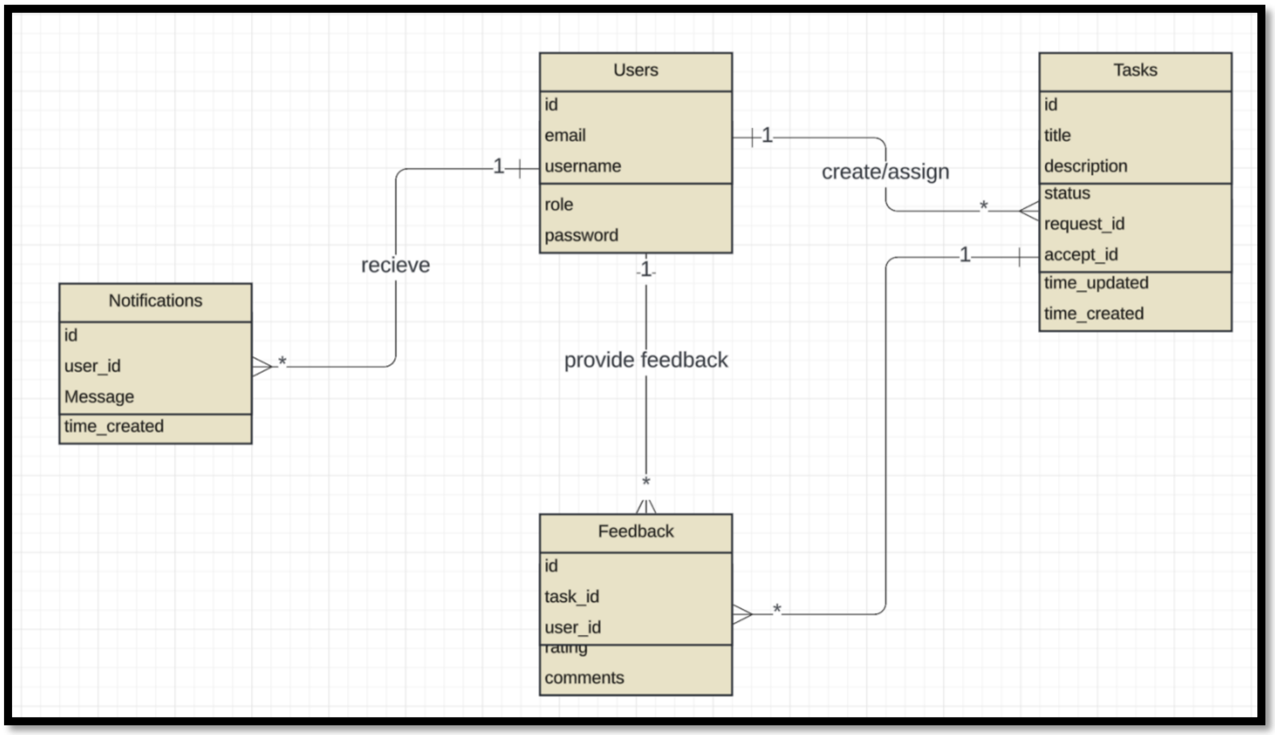


Fig. (6.3): ERD showing the relationships between the tables in the database system, where an arrow marked with 1à* means a one-to-many relationship

6.5. Network protocol

CS ARCS works over networks since it is a web-based application. It uses the following network protocols:

1. **HTTPS:** Enables communication between client devices (browsers) and the application server. Also, we chose the secure version of HTTP for security purposes.
2. **TCP/IP:** Handles reliable communication between the application server and the database server. It also guarantees data integrity and delivery.
3. **SMTP:** Used to send email notifications to users about task updates, reminders, and feedback.

6.6. Hardware requirements

CS ARCS requires low hardware requirements for clients since it is a web-based application. This achieves our goal of our program being reachable for everyone. The client devices' minimum hardware requirements are:

- Any device that runs a modern web browser (desktop, laptop, smartphone etc.)
- Modern web browser (e.g. Google Chrome, Mozilla Firefox)
- Any OS that can run a modern web browser (MacOS, iOS, Windows, Linux, etc.)
- A moderate internet speed (preferably above 2 Mbps)

As for the application server's recommended hardware requirements:

- Quad-core processor
- Minimum of 8 GB RAM
- Minimum of 100 GB SSD storage
- Modern operating system

These choices guarantee a fast and reliable server that can serve multiple clients concurrently.

7.Class Diagram and Interface specification (part 2)

7.1.Class Diagram

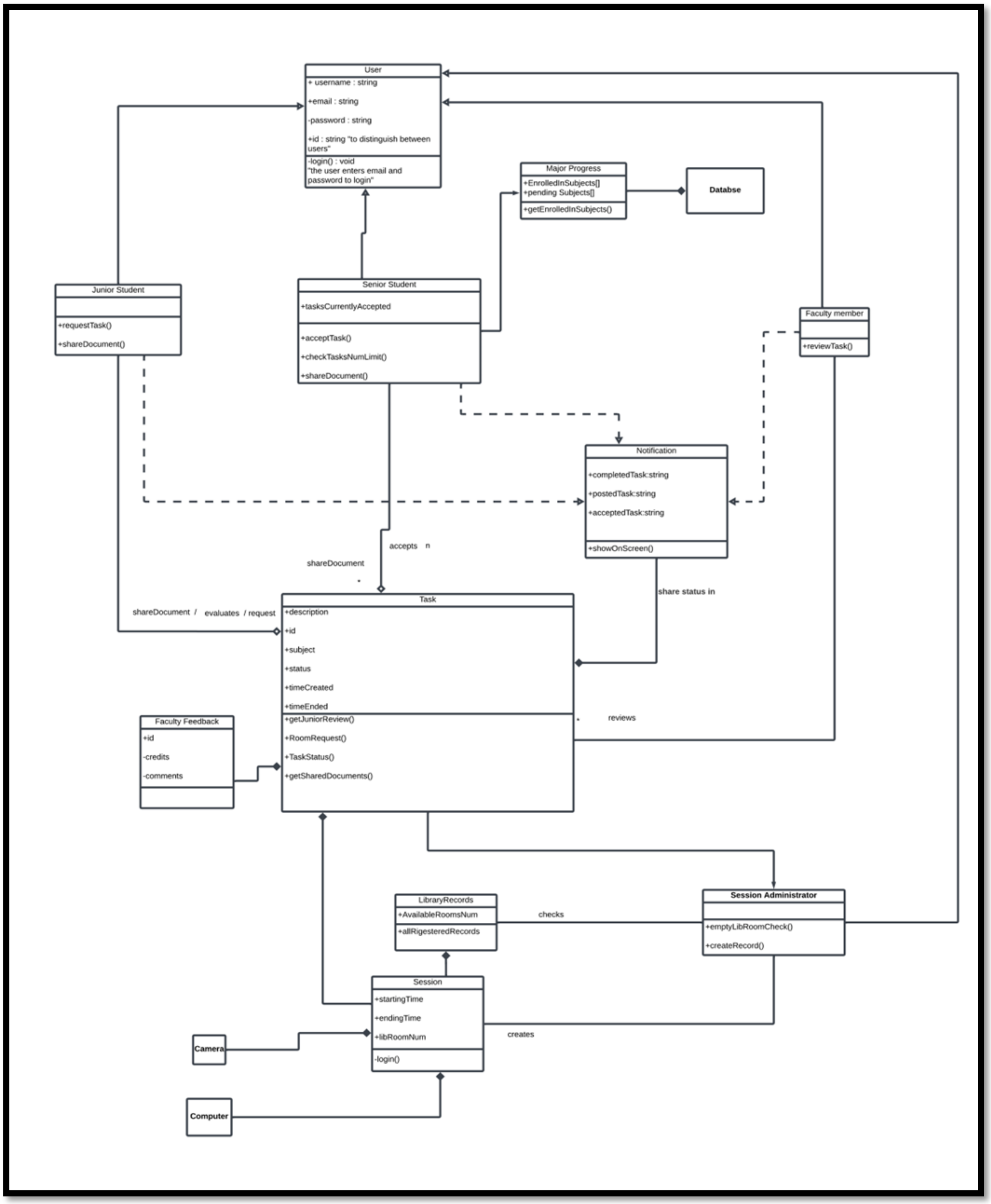
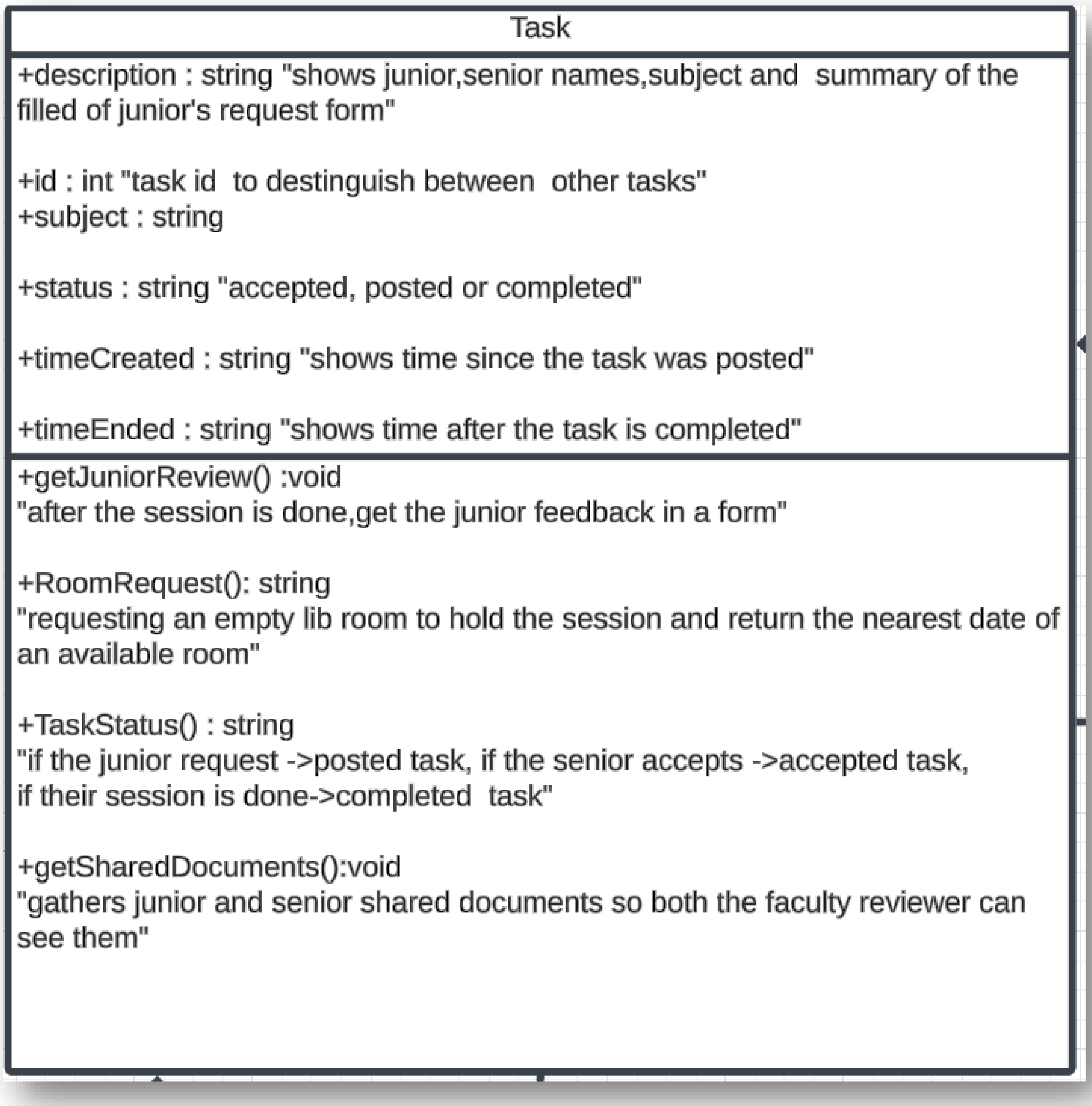


Fig. (7.1): class diagram

7.2. Data types and signatures



Task

+description : string "shows junior,senior names,subject and summary of the filled of junior's request form"

+id : int "task id to destinguish between other tasks"
+subject : string

+status : string "accepted, posted or completed"

+timeCreated : string "shows time since the task was posted"

+timeEnded : string "shows time after the task is completed"

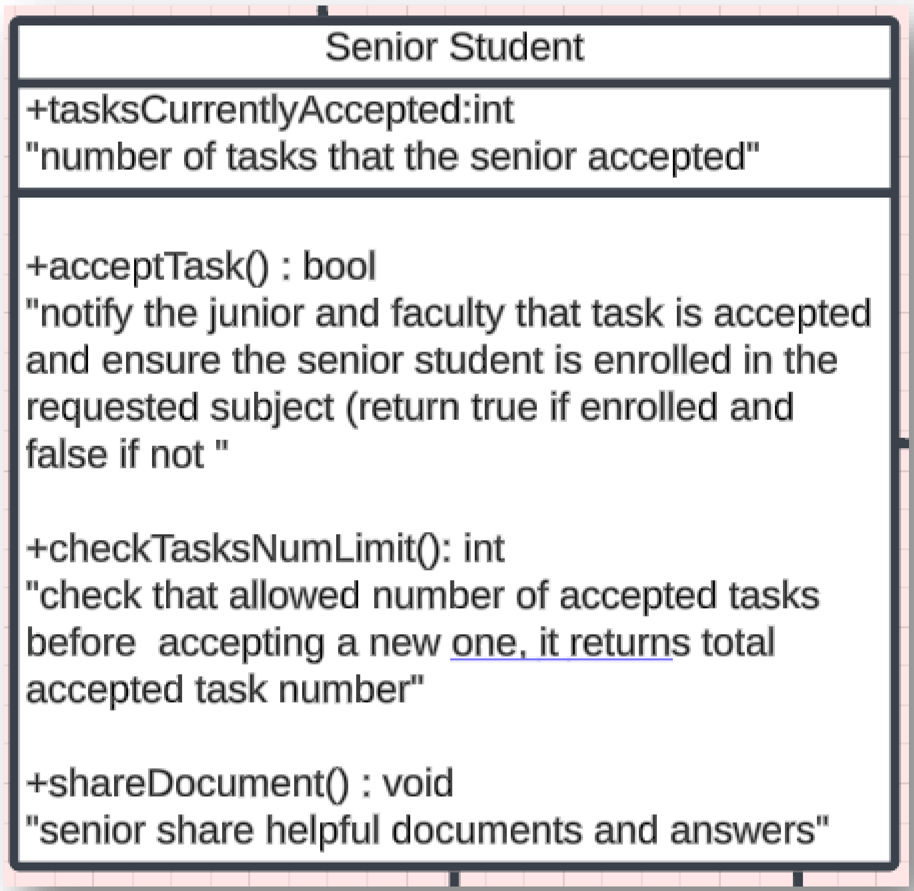
+getJuniorReview() :void
"after the session is done,get the junior feedback in a form"

+RoomRequest(): string
"requesting an empty lib room to hold the session and return the nearest date of an available room"

+TaskStatus() : string
"if the junior request ->posted task, if the senior accepts ->accepted task,
if their session is done->completed task"

+getSharedDocuments():void
"gathers junior and senior shared documents so both the faculty reviewer can see them"

Fig. (7.2): task description



Senior Student

+tasksCurrentlyAccepted:int
"number of tasks that the senior accepted"

+acceptTask() : bool
"notify the junior and faculty that task is accepted and ensure the senior student is enrolled in the requested subject (return true if enrolled and false if not "

+checkTasksNumLimit(): int
"check that allowed number of accepted tasks before accepting a new one, it returns total accepted task number"

+shareDocument() : void
"senior share helpful documents and answers"

Fig. (7.3): Senior description

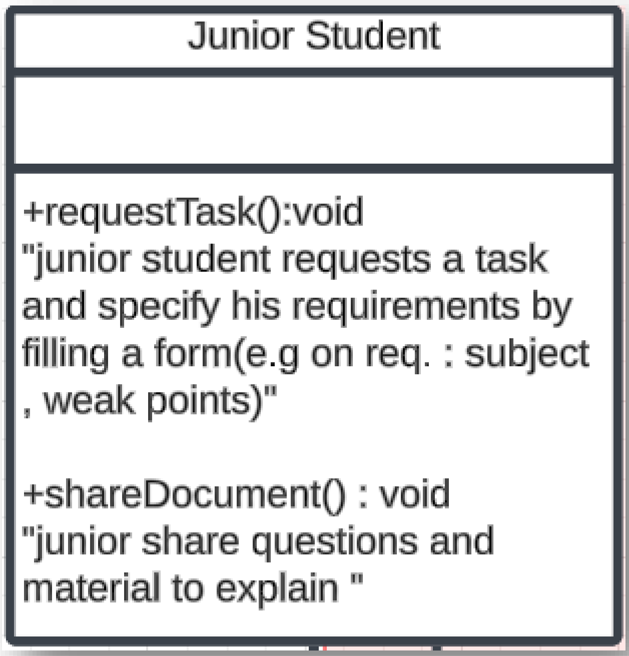

Junior Student
+requestTask():void
"junior student requests a task and specify his requirements by filling a form(e.g on req. : subject , weak points)"
+shareDocument() : void
"junior share questions and material to explain "

Fig. (7.4): Junior description

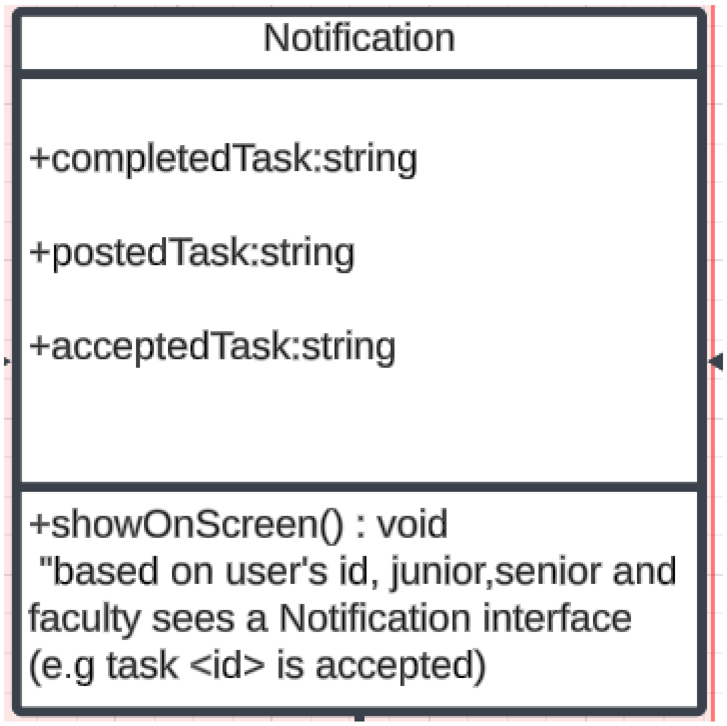

Notification
+completedTask:string
+postedTask:string
+acceptedTask:string
+showOnScreen() : void
"based on user's id, junior,senior and faculty sees a Notification interface (e.g task <id> is accepted)

Fig. (7.5): Notification description

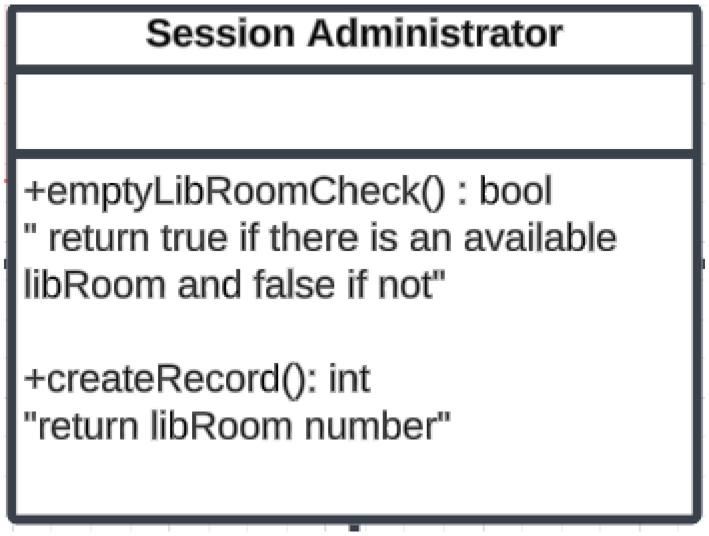

Session Administrator
+emptyLibRoomCheck() : bool
" return true if there is an available libRoom and false if not"
+createRecord(): int
"return libRoom number"

Fig. (7.6): Session Admin description

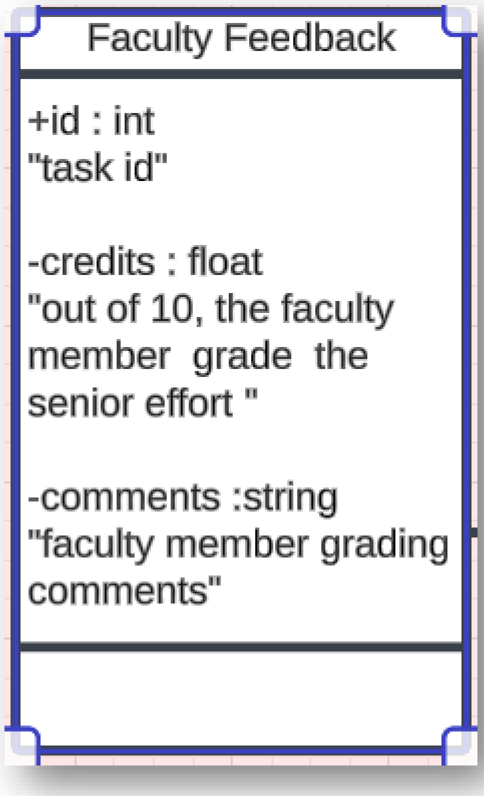

Faculty Feedback
+id : int
"task id"
-credits : float
"out of 10, the faculty member grade the senior effort "
-comments :string
"faculty member grading comments"

Fig. (7.7): Faculty feedback description

8.Interaction Diagrams (part 2)

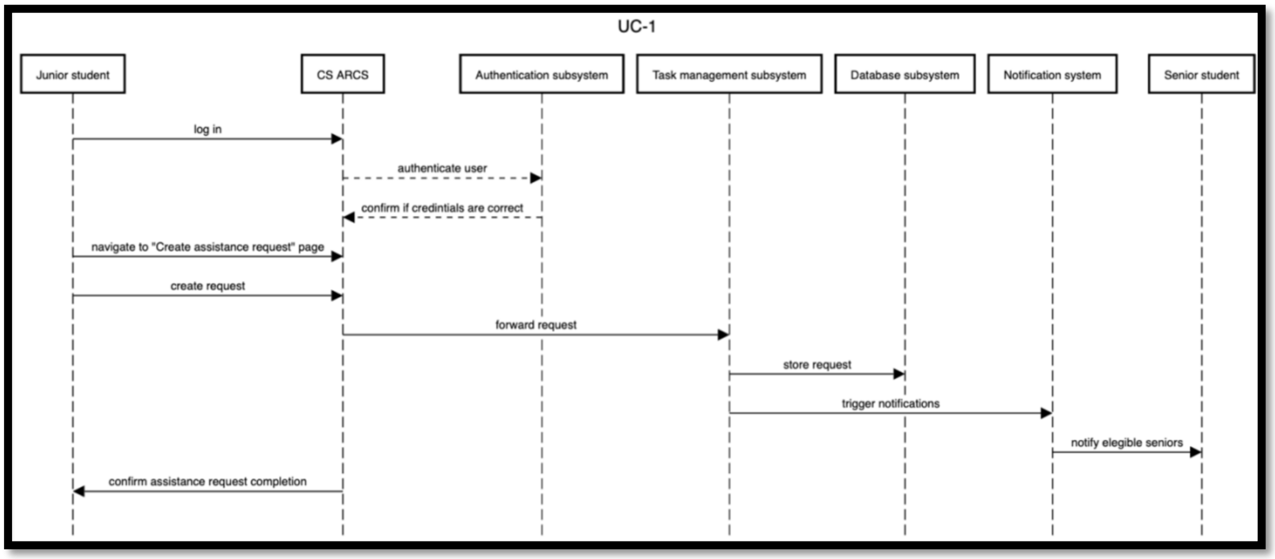


Fig. (8.1): detailed sequence diagram for UC-1 regarding creating assistance request

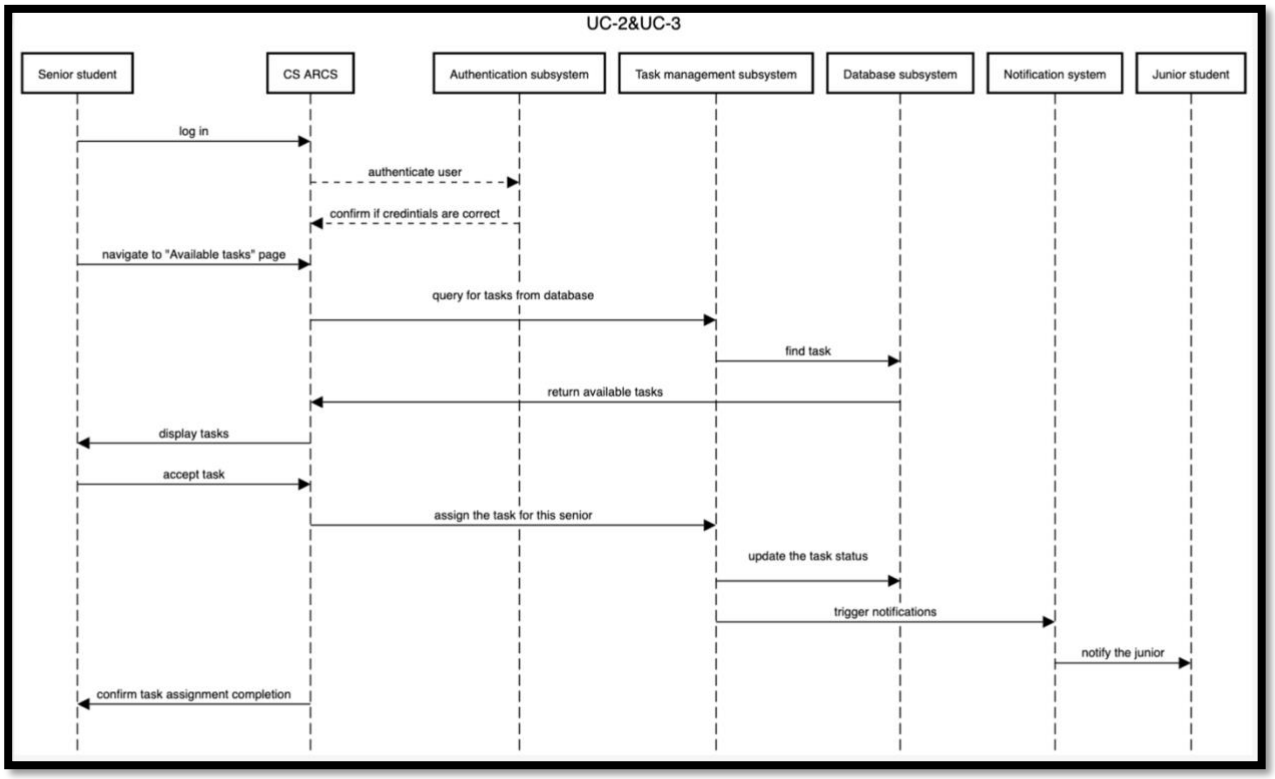


Fig. (8.2): detailed sequence diagram for UC-2 & UC-3 regarding navigating available tasks and accepting tasks

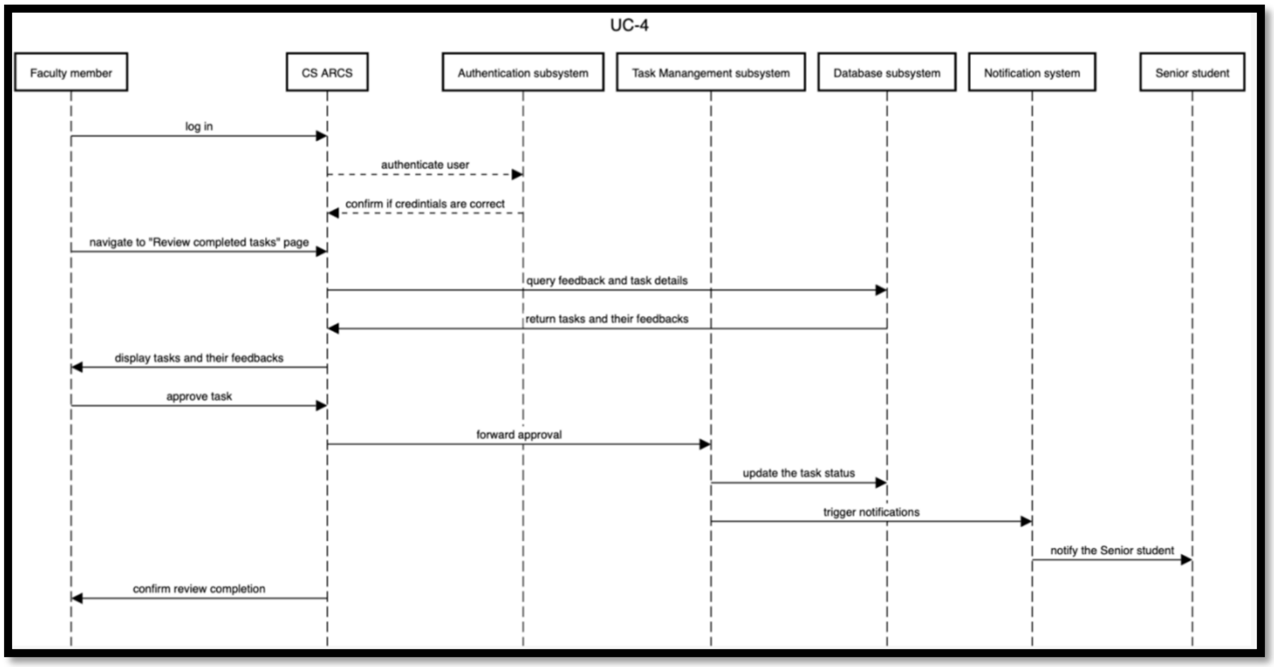

UC-4
Faculty member
CS ARCS
Authentication subsystem
Task Manangement subsystem
Database subsystem
Notification system
Senior student
log in
authenticate user
confirm if credintials are correct
navigate to "Review completed tasks" page
query feedback and task details
return tasks and their feedbacks
display tasks and their feedbacks
approve task
forward approval
update the task status
trigger notifications
notify the Senior student
confirm review completion

Fig. (8.3): detailed sequence diagram for UC-4 regarding reviewing feedback and awarding extra credit

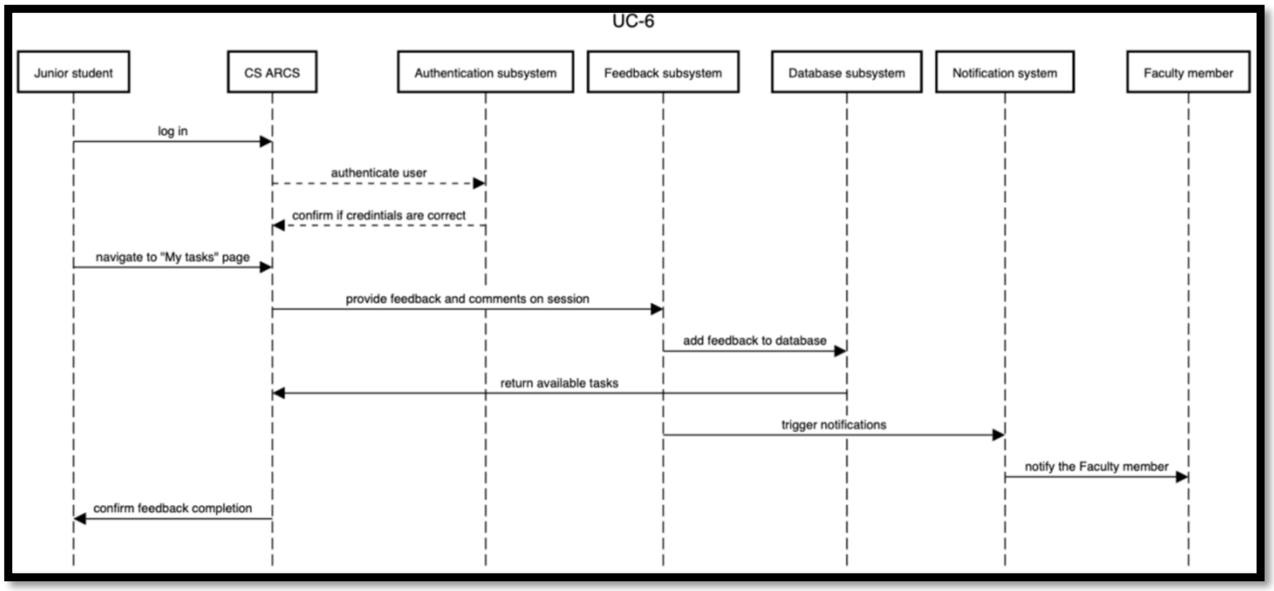


Fig. (8.4): detailed sequence diagram for UC-6 regarding providing feedback about completed tasks

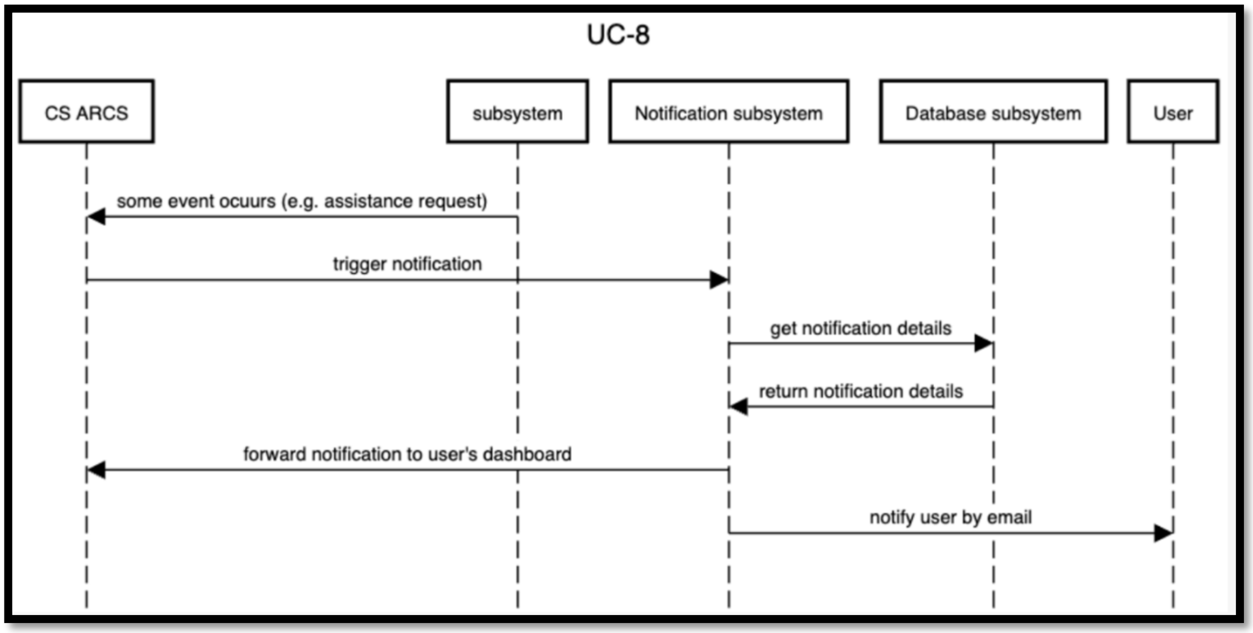


Fig. (8.5): detailed sequence diagram for UC-8 regarding notifying users

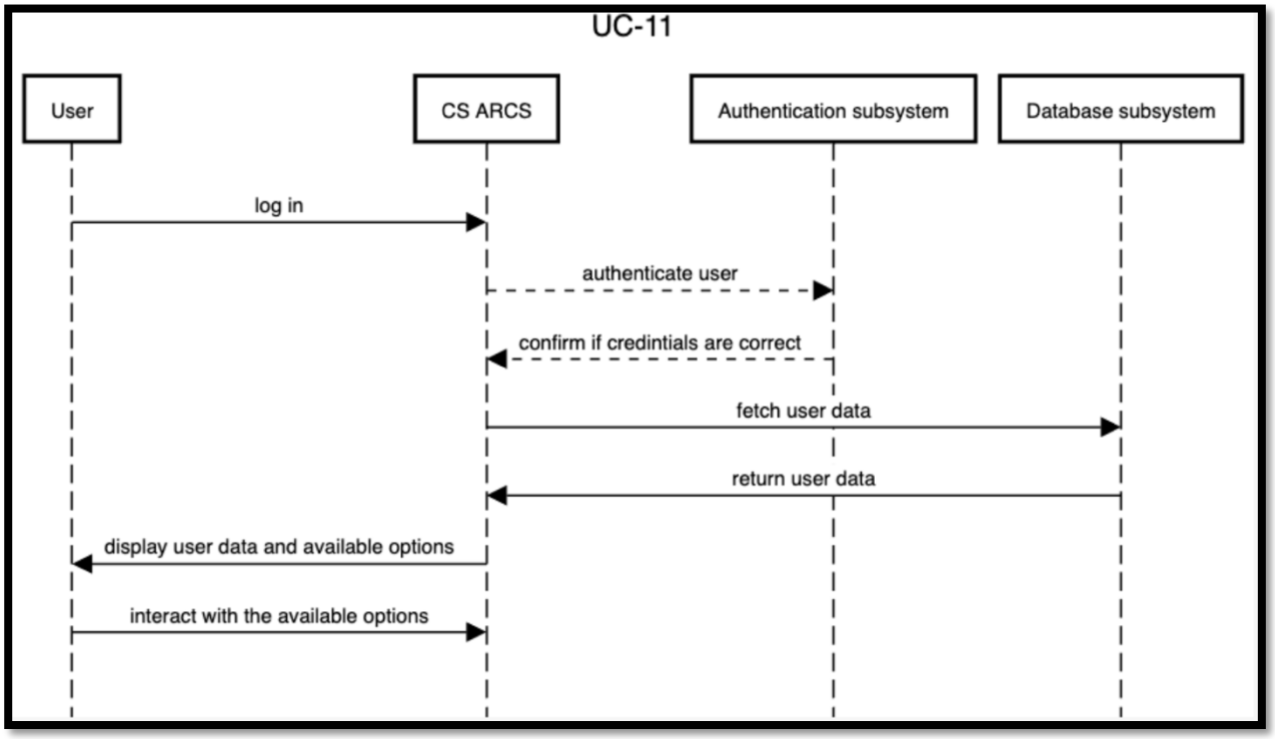


Fig. (8.6): detailed sequence diagram for UC-11 regarding logging in and authentication

9.User Interface

9.1.Initial User Interface Specification

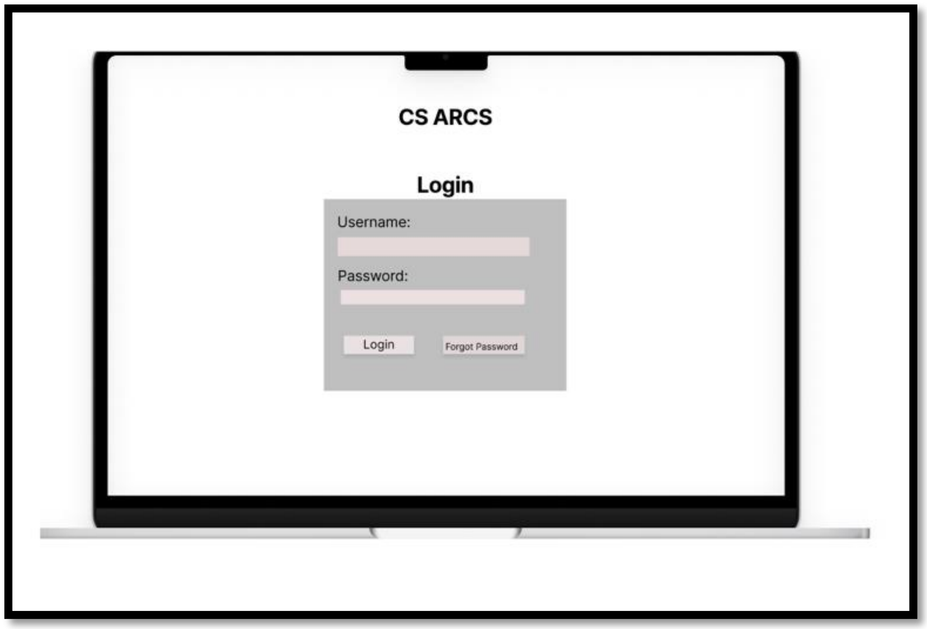


Fig. (9.1): main login page on a PC/Laptop (UC-9)

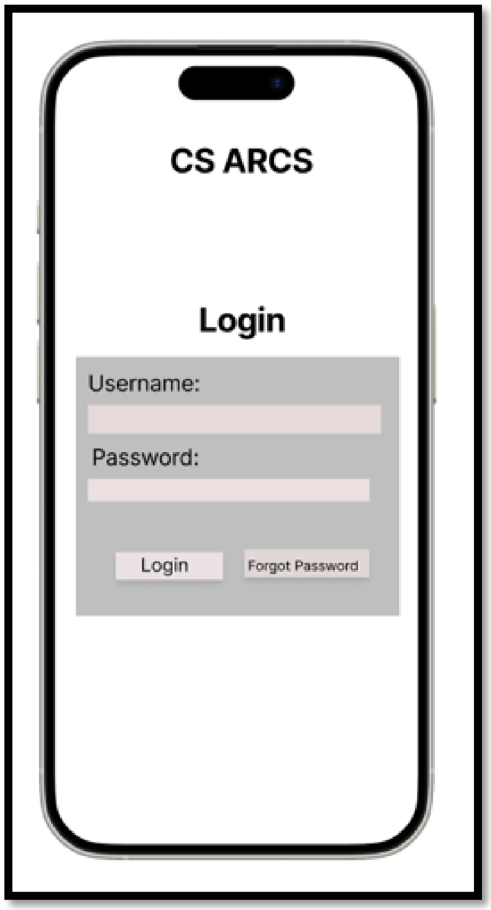


Fig. (9.2) : main login page on a smartphone (UC-9)

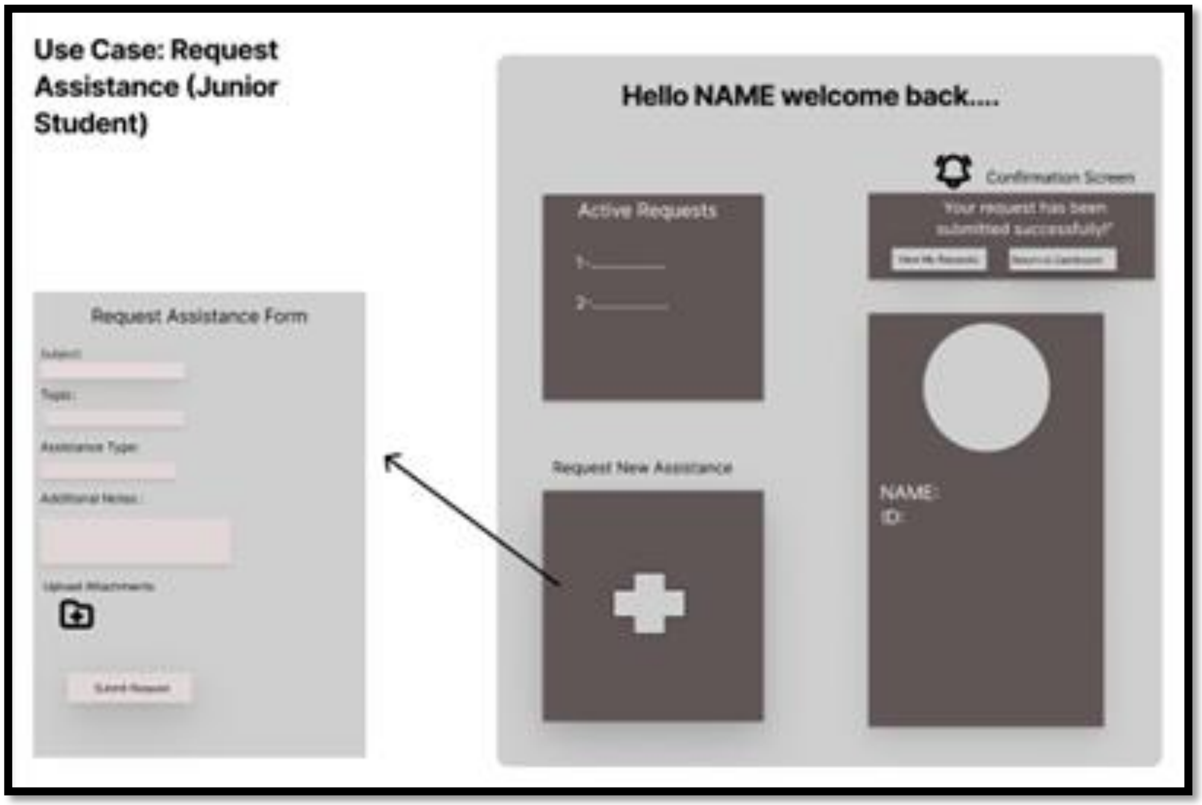


Fig. (9.3): Dashboard for Junior students (UC-1& UC-11)

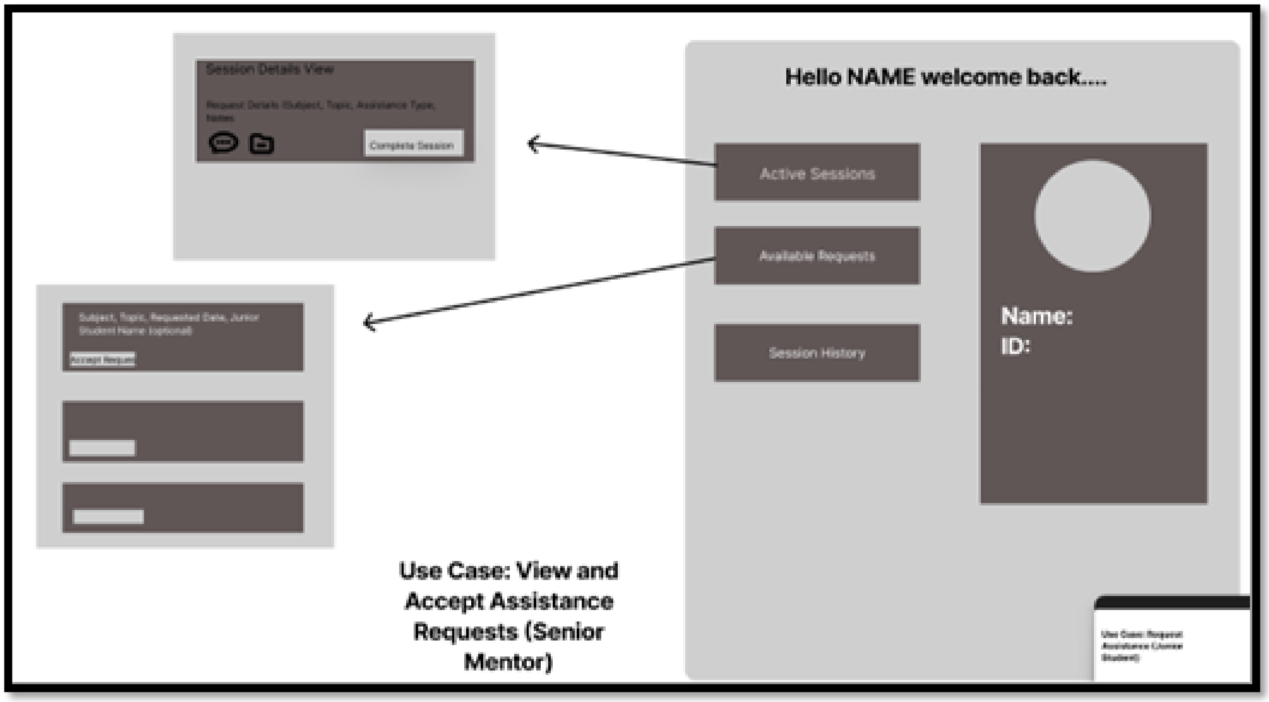

Session Details View
Request Details (Subject, Topic, Assistance Type, Notes)
Complete Session
Hello NAME welcome back....
Active Sessions
Available Requests
Session History
Subject, Topic, Requested Date, Junior Student Name (optional)
Accept Request
Name:
ID:
Use Case: View and Accept Assistance Requests (Senior Mentor)
Use Case: Request Assistance (Junior Student)

Fig. (9.4): Dashboard for Senior students (UC-2 & UC-3 & UC-11)

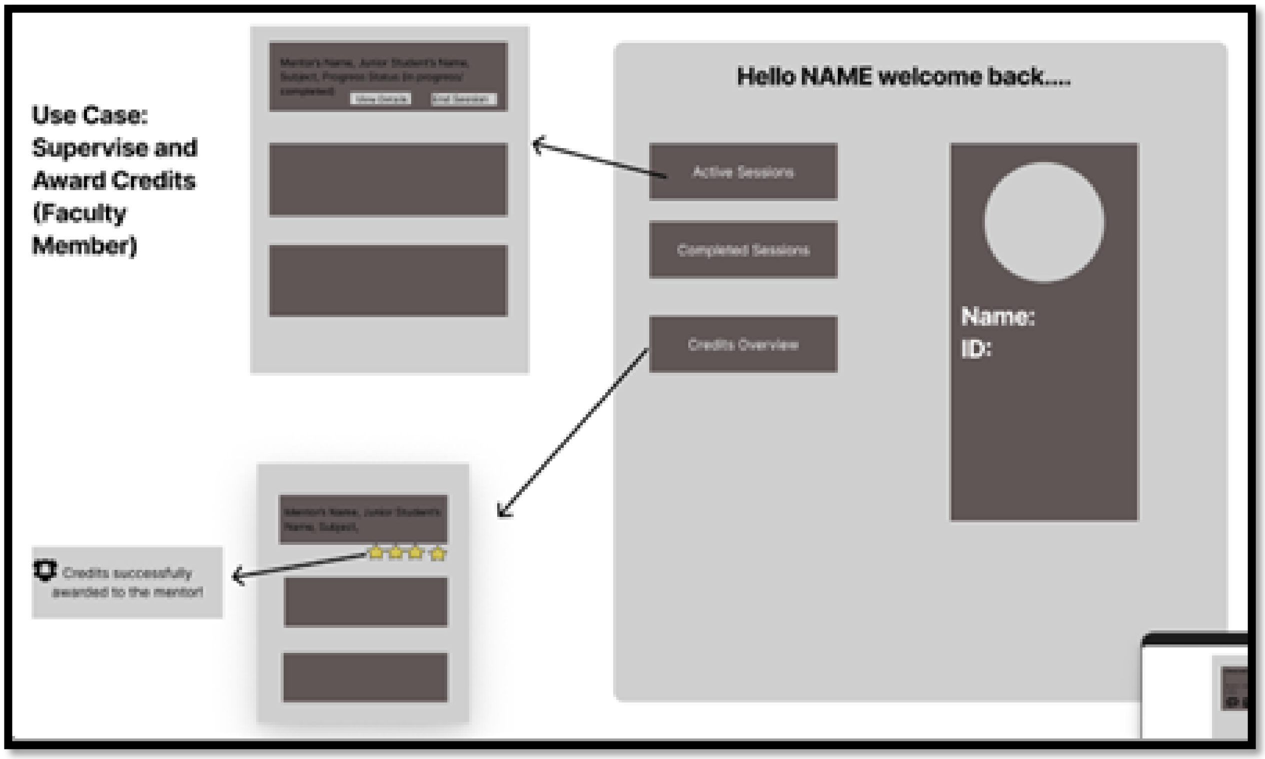

Use Case: Supervise and Award Credits (Faculty Member)
Mentor's Name, Junior Student's Name, Subject, Progress Status (in progress/ completed)
View Details
End Session
Hello NAME welcome back....
Active Sessions
Completed Sessions
Credits Overview
Name:
ID:
Mentor's Name, Junior Student's Name, Subject,
Credits successfully awarded to the mentor!

Fig. (9.5): Dashboard for Faculty members (UC-4 &UC-11 & UC-12)

9.2.User Interface Design and Implementation

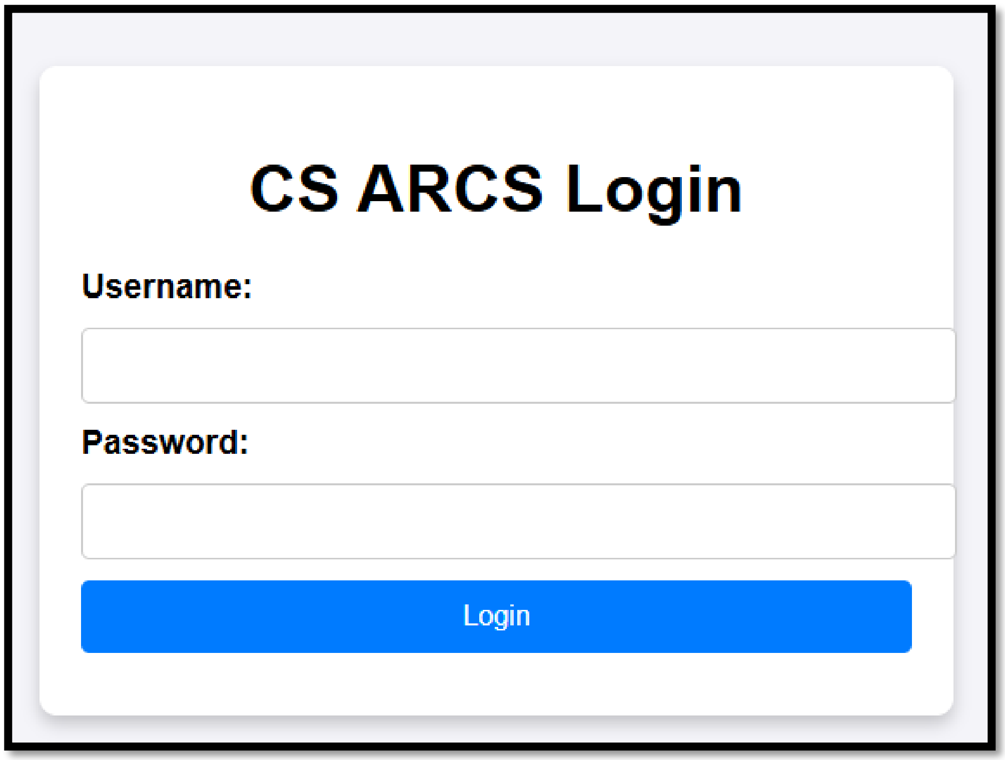
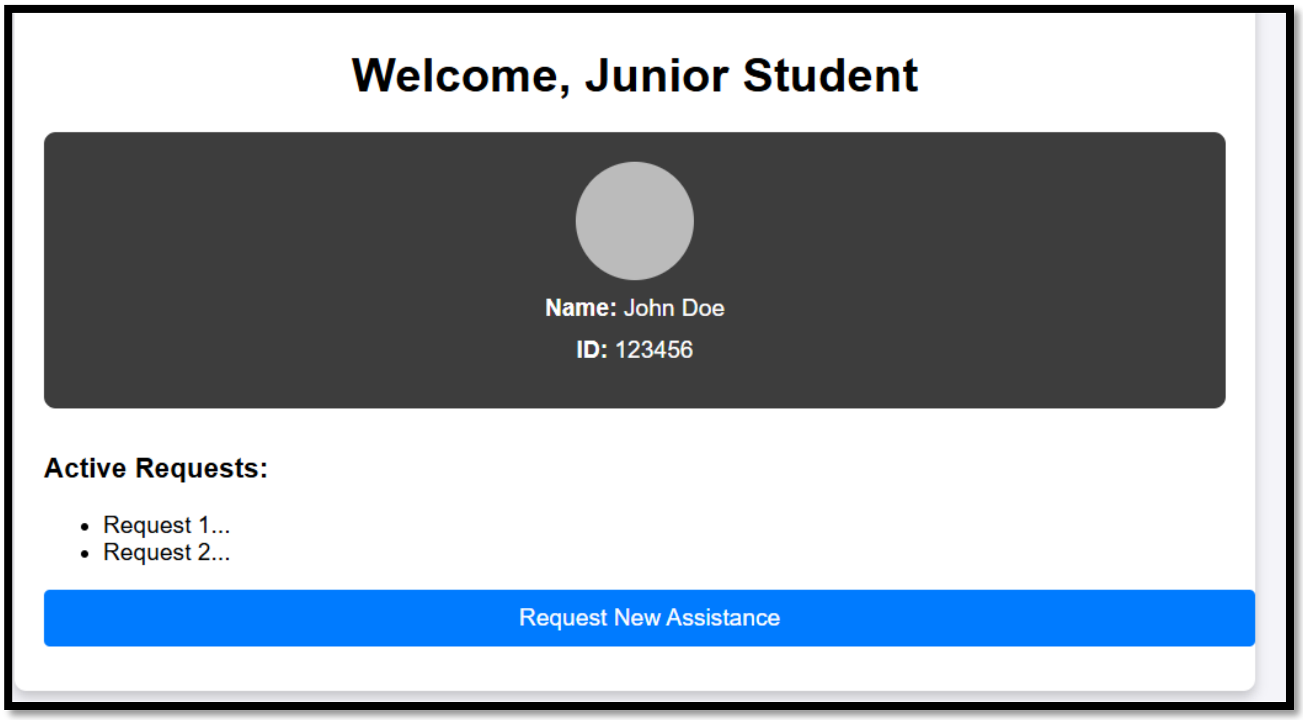


Fig. (9.6): main login page on a PC/Laptop (UC-9)



Welcome, Junior Student

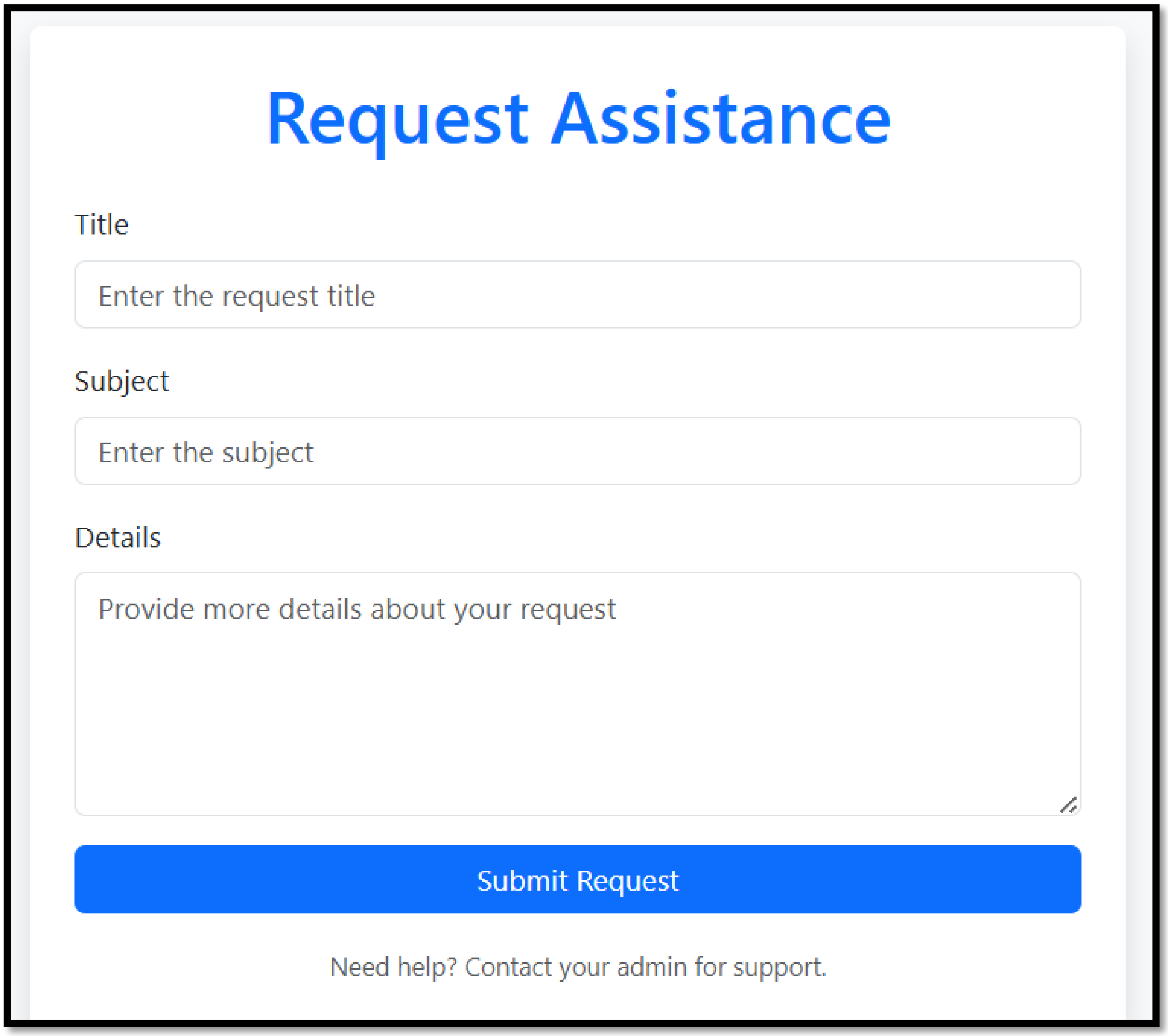
Name: John Doe

ID: 123456

Active Requests:

- Request 1...
- Request 2...

Fig. (9.7): Dashboard for Junior students (UC-1& UC-11)



Request Assistance

Title

Enter the request title

Subject

Enter the subject

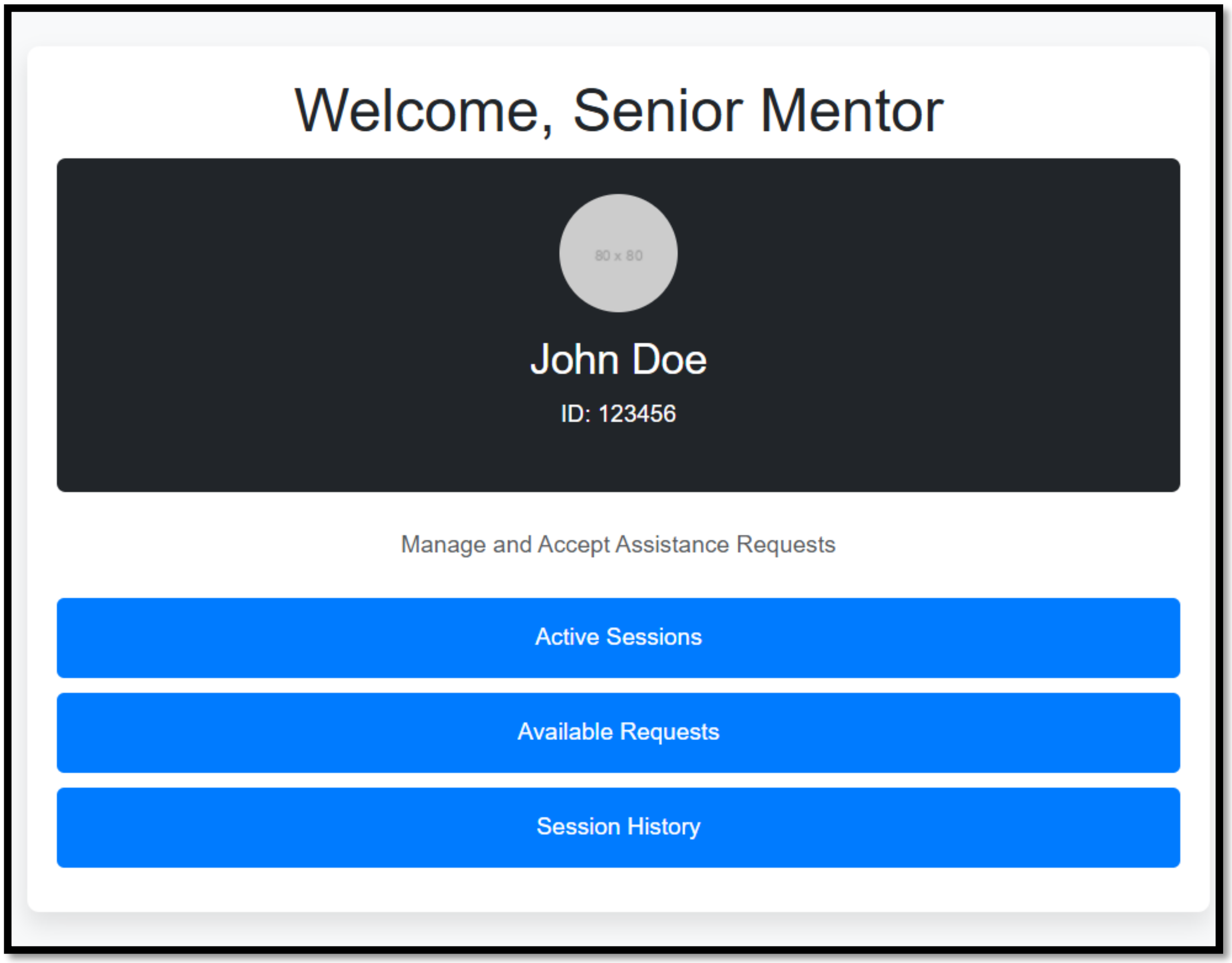
Details

Provide more details about your request

Submit Request

Need help? Contact your admin for support.

Fig. (9.8): Assistance request page for Junior students

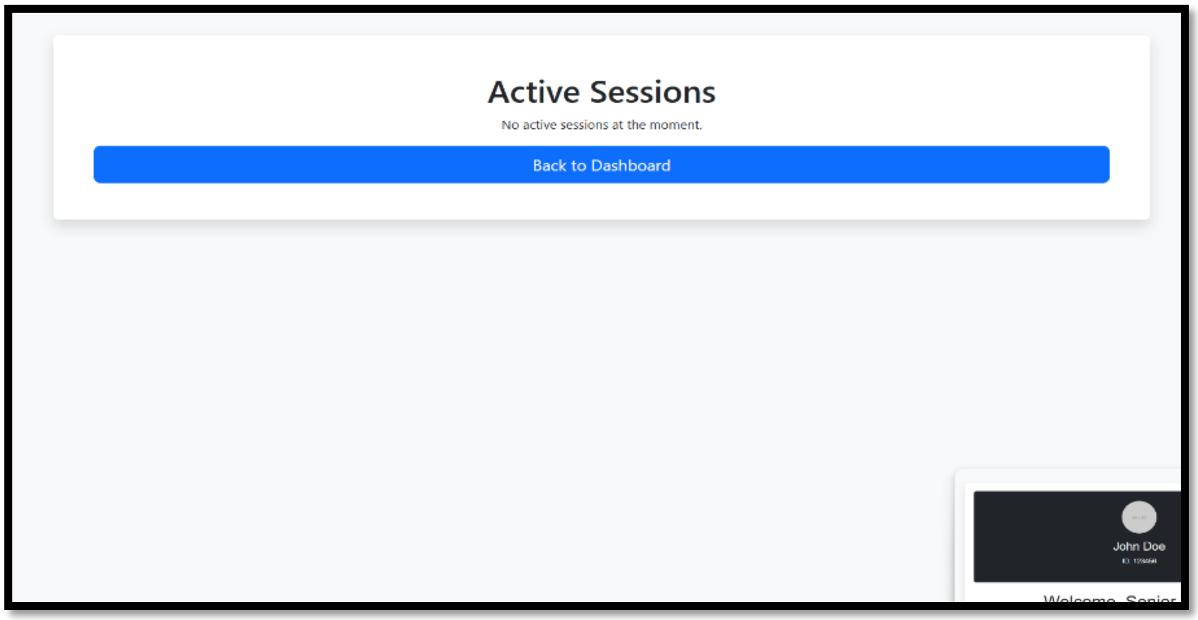

Welcome, Senior Mentor

John Doe

ID: 123456

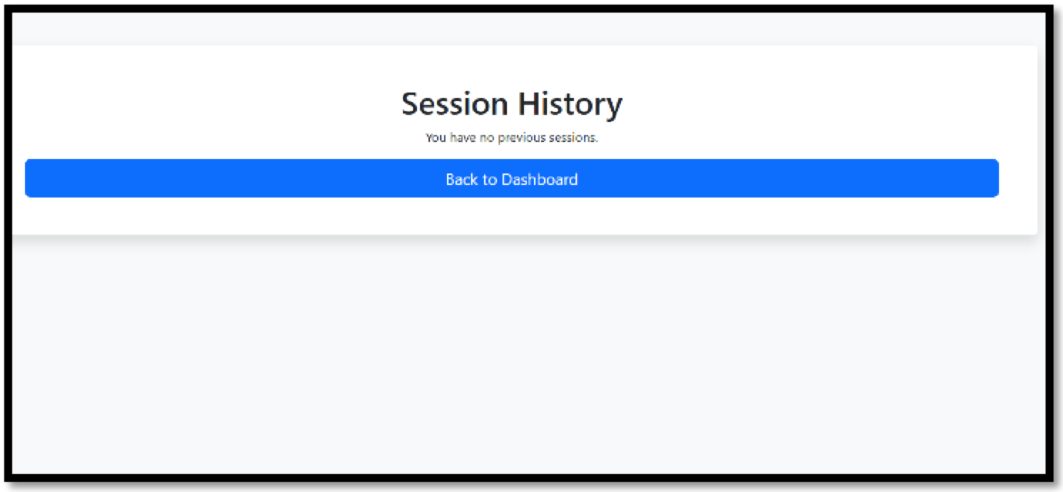
Manage and Accept Assistance Requests

Fig. (9.9): Dashboard for senior students


Active Sessions

No active sessions at the moment.

Fig. (9.10): Active sessions page for senior students


Session History

You have no previous sessions.

Fig. (9.11): Session History for Senior students

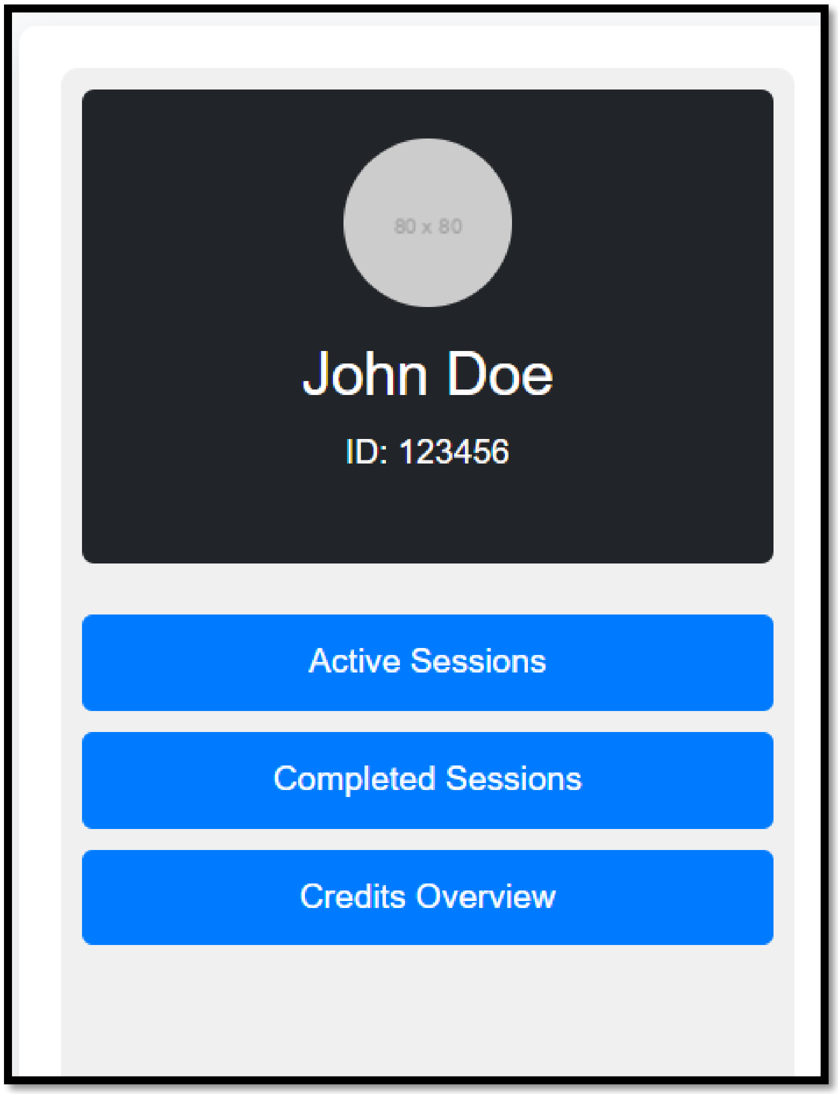
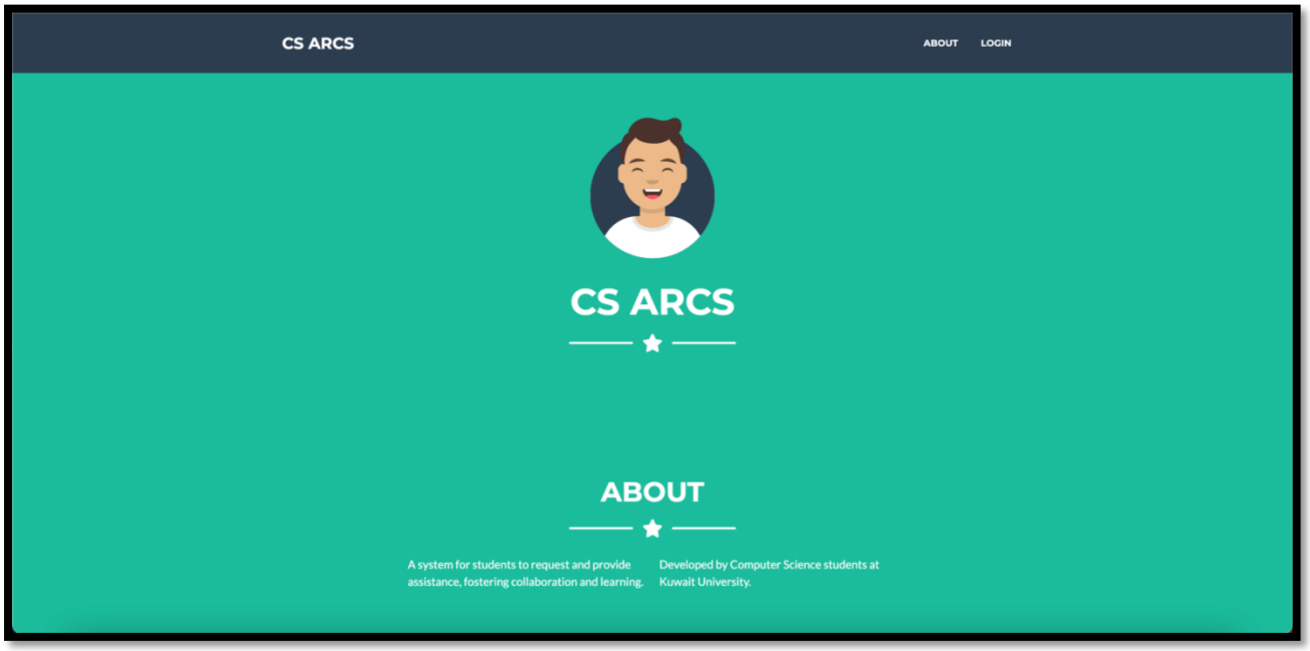


Fig. (9.12): Dashboard for faculty members

9.3.Final Screenshots

During the final phase, we improved the user interface and changed the design of most pages. The final design and screenshots for the pages are listed below.


CS ARCS

ABOUT LOGIN

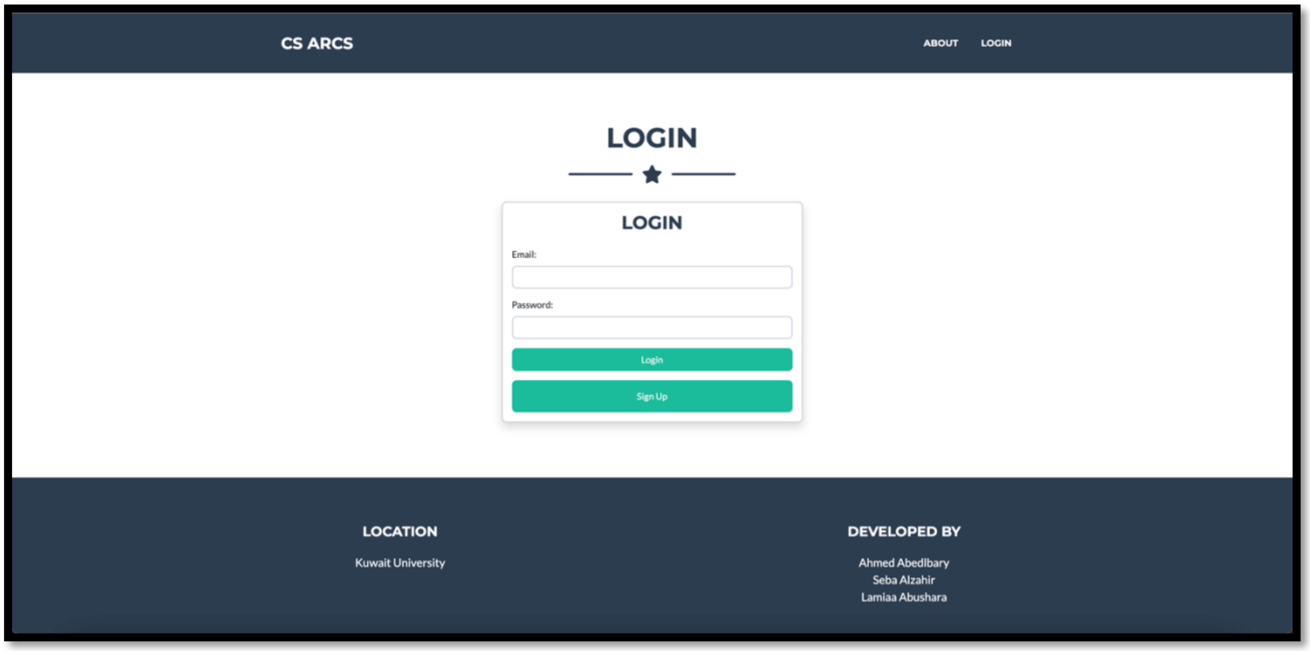
CS ARCS

ABOUT

A system for students to request and provide assistance, fostering collaboration and learning.

Developed by Computer Science students at Kuwait University.

Fig. (9.13): Home page for CS ARCS


CS ARCS

ABOUT LOGIN

LOGIN

LOGIN

Email:

Password:

Login

Sign Up

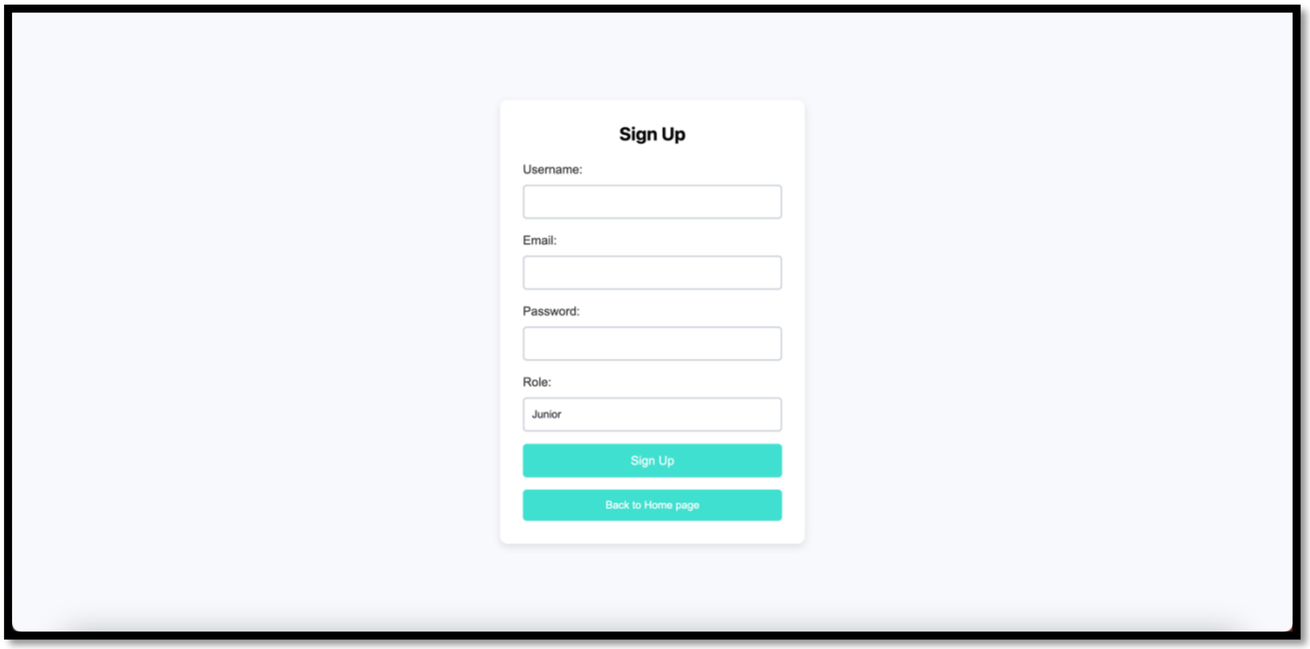
LOCATION

Kuwait University

DEVELOPED BY

Ahmed Abedlbary
Seba Alzahir
Lamiaa Abushara

Fig. (9.14): Home page for CS ARCS (after scrolling down) showing login/signup section



Sign Up

Username:

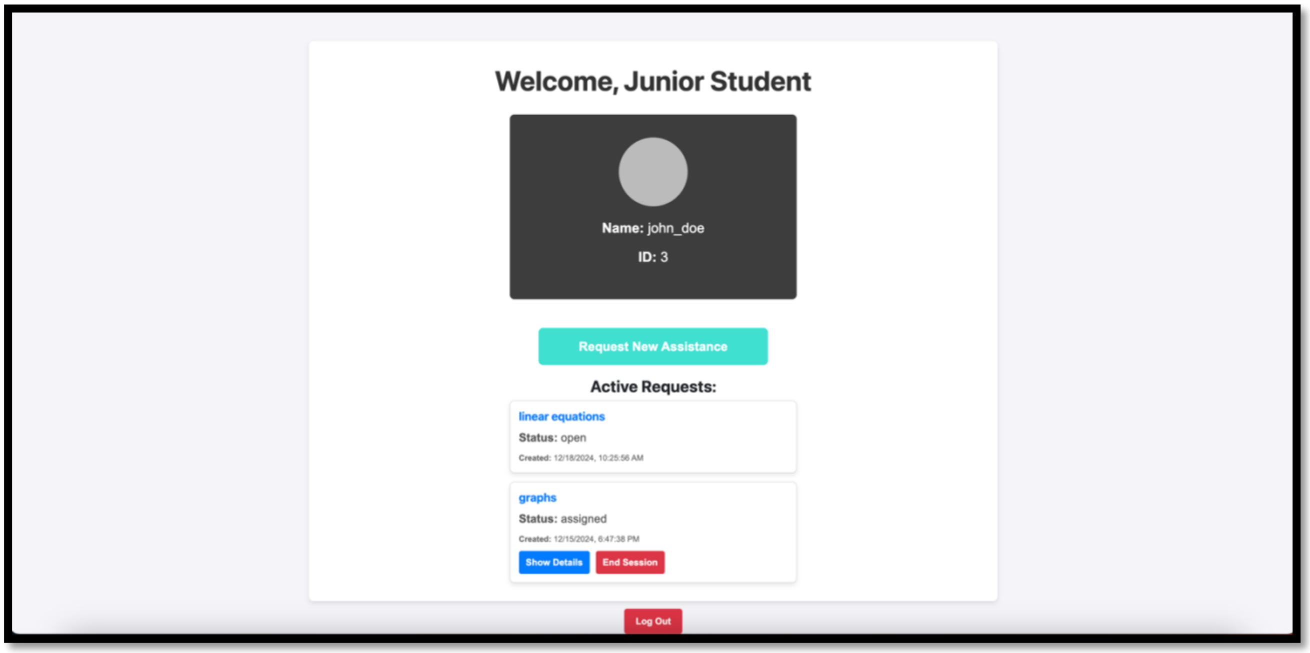
Email:

Password:

Role:

Junior

Fig. (9.15): Sign Up page



Welcome, Junior Student

Name: john_doe

ID: 3

Active Requests:

linear equations

Status: open

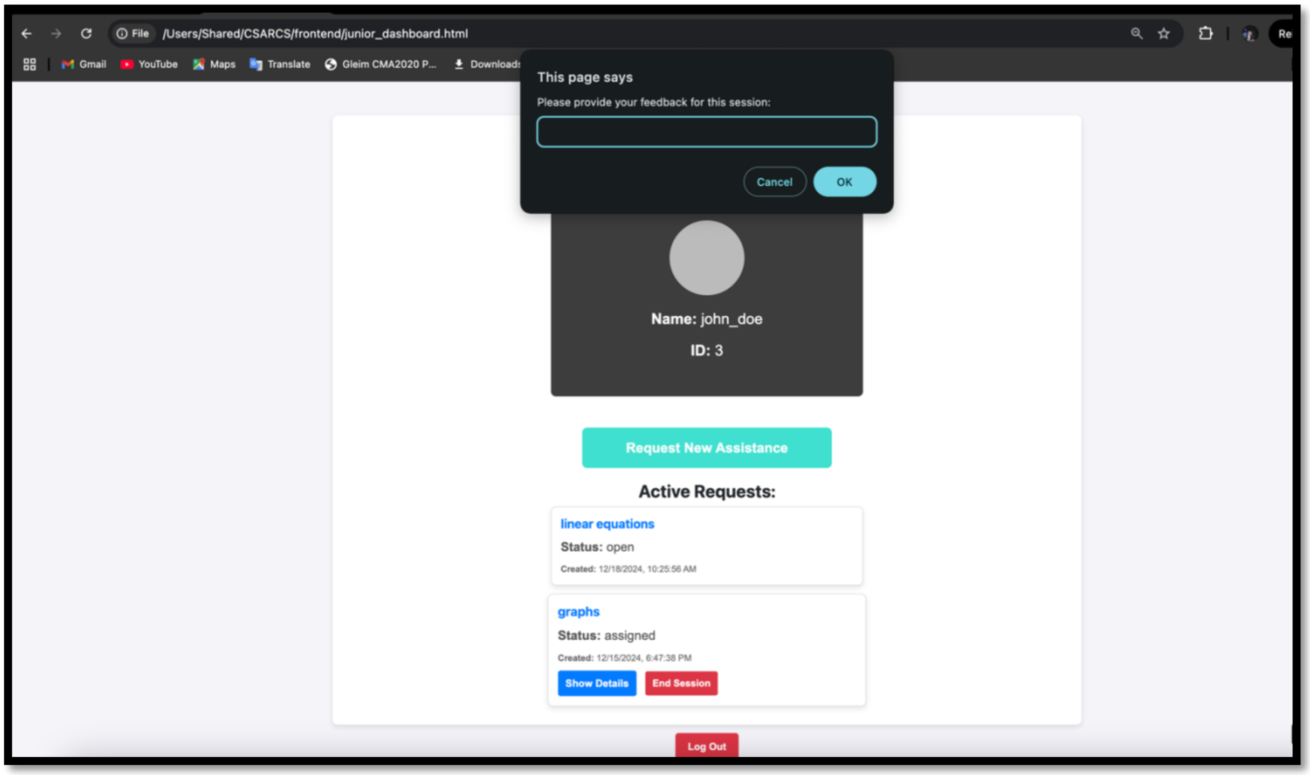
Created: 12/18/2024, 10:25:56 AM

graphs

Status: assigned

Created: 12/15/2024, 6:47:38 PM

Fig. (9.16a): junior student's dashboard. Where the junior student can view their active requests' details, end their session and request a new assistance task



/Users/Shared/CSARCS/frontend/junior_dashboard.html

This page says

Please provide your feedback for this session:

Cancel OK

Name: john_doe

ID: 3

Request New Assistance

Active Requests:

linear equations

Status: open

Created: 12/18/2024, 10:25:56 AM

graphs

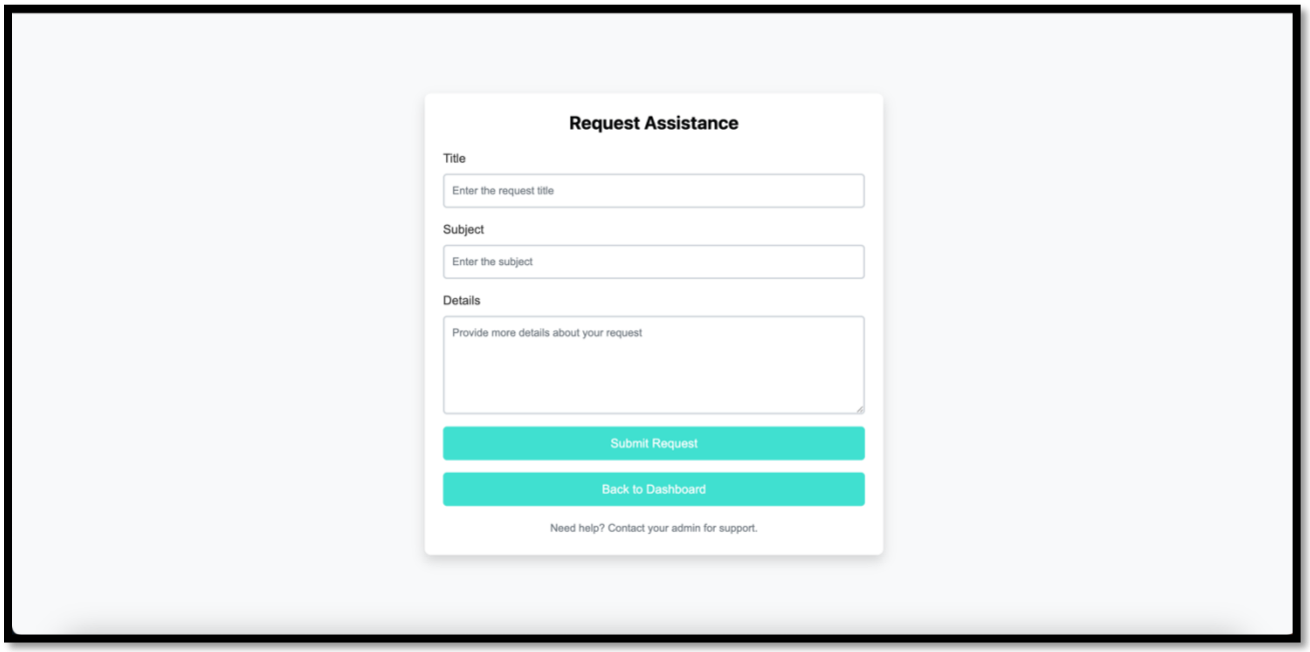
Status: assigned

Created: 12/15/2024, 6:47:38 PM

Show Details End Session

Log Out

Fig. (9.16b): When the junior chooses to end a session, the page prompts them to provide a feedback for the session (a junior cannot end a session if they do not provide a feedback) therefore marking the session as complete.



Request Assistance

Title

Enter the request title

Subject

Enter the subject

Details

Provide more details about your request

Submit Request

Back to Dashboard

Need help? Contact your admin for support.

Fig. (9.17): Request assistance page for juniors

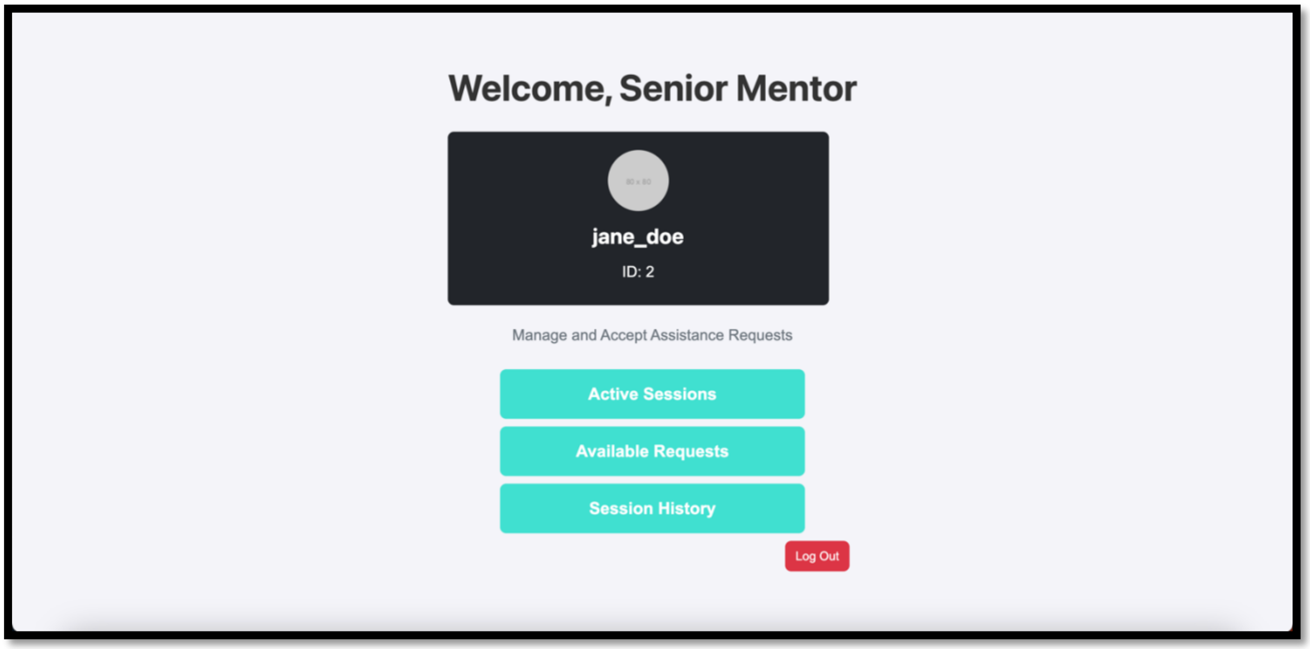
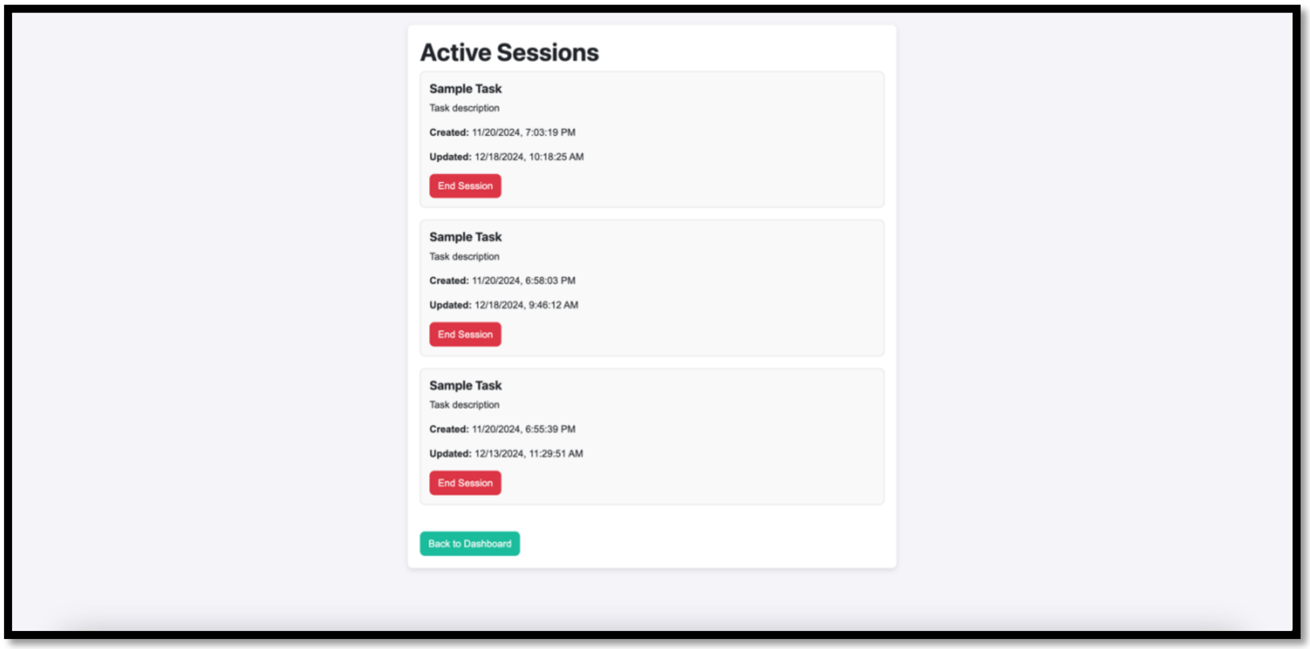


Fig. (9.18): Dashboard for senior students, where seniors can view their active sessions , view available requests or see their session history

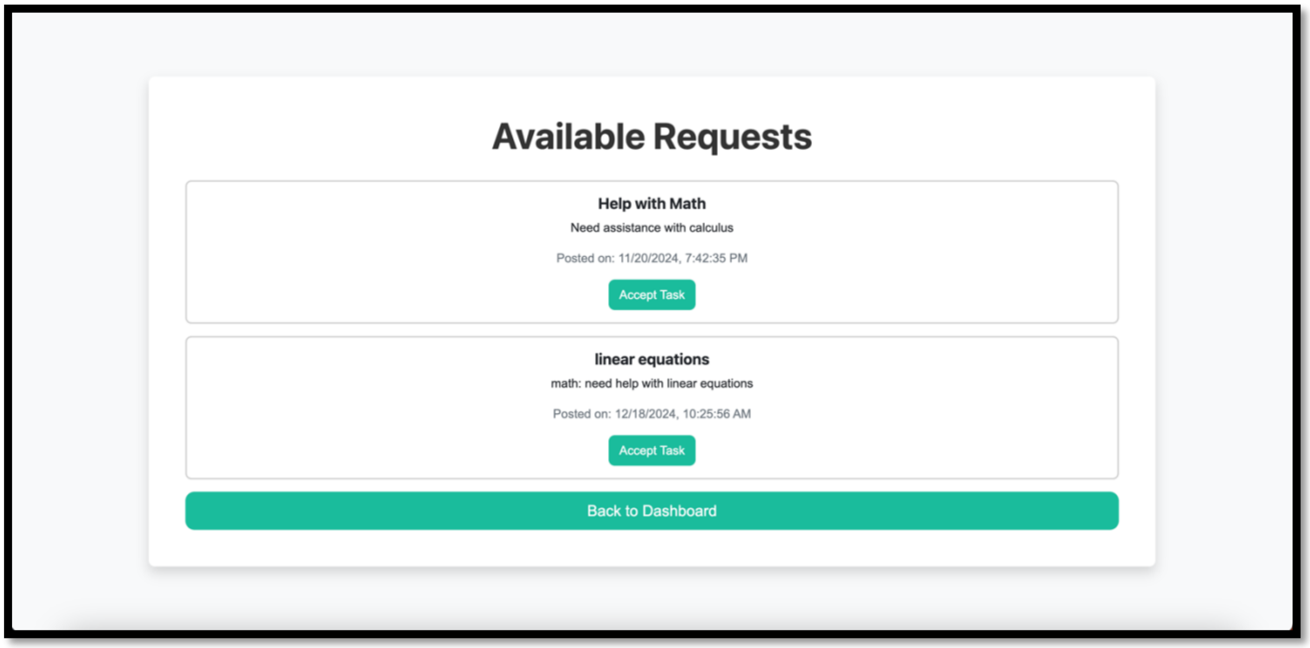

Active Sessions

Sample Task
Task description
Created: 11/20/2024, 7:03:19 PM
Updated: 12/18/2024, 10:18:25 AM

Sample Task
Task description
Created: 11/20/2024, 6:58:03 PM
Updated: 12/18/2024, 9:46:12 AM

Sample Task
Task description
Created: 11/20/2024, 6:55:39 PM
Updated: 12/13/2024, 11:29:51 AM

Fig. (9.19): Active sessions page (seniors)



Available Requests

Help with Math

Need assistance with calculus

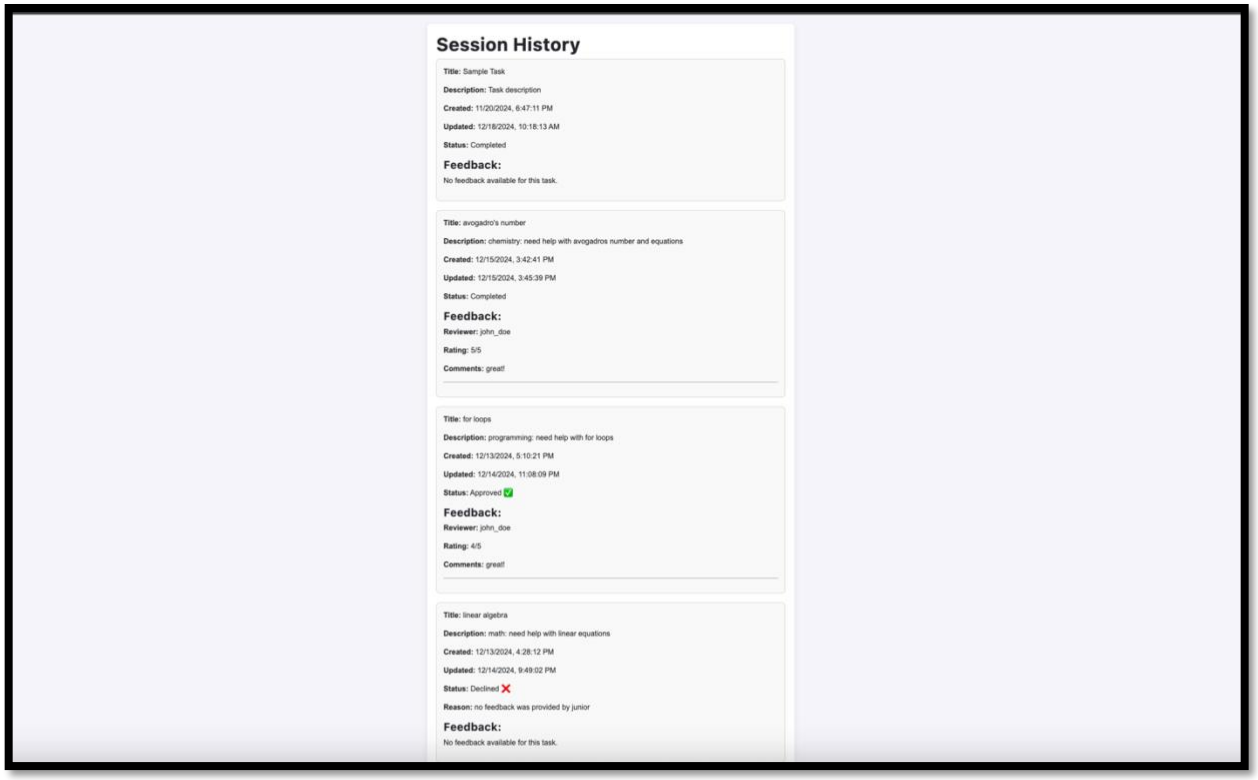
Posted on: 11/20/2024, 7:42:35 PM

linear equations

math: need help with linear equations

Posted on: 12/18/2024, 10:25:56 AM

Fig. (9.20): Available requests page (seniors) where they can accept tasks the juniors requested



Session History

Title: Sample Task

Description: Task description

Created: 11/20/2024, 6:47:11 PM

Updated: 12/18/2024, 10:18:13 AM

Status: Completed

Feedback:

No feedback available for this task.

Title: avogadro's number

Description: chemistry: need help with avogadros number and equations

Created: 12/15/2024, 3:42:41 PM

Updated: 12/15/2024, 3:45:39 PM

Status: Completed

Feedback:

Reviewer: john_doe

Rating: 5/5

Comments: great!

Title: for loops

Description: programming: need help with for loops

Created: 12/13/2024, 5:10:21 PM

Updated: 12/14/2024, 11:08:09 PM

Status: Approved ✅

Feedback:

Reviewer: john_doe

Rating: 4/5

Comments: great!

Title: linear algebra

Description: math: need help with linear equations

Created: 12/13/2024, 4:28:12 PM

Updated: 12/14/2024, 9:49:02 PM

Status: Declined ❌

Reason: no feedback was provided by junior

Feedback:

No feedback available for this task.

Fig. (9.21): Session history page (seniors) where they can see their previous task details, their statuses(approved – declined) and feedbacks from juniors

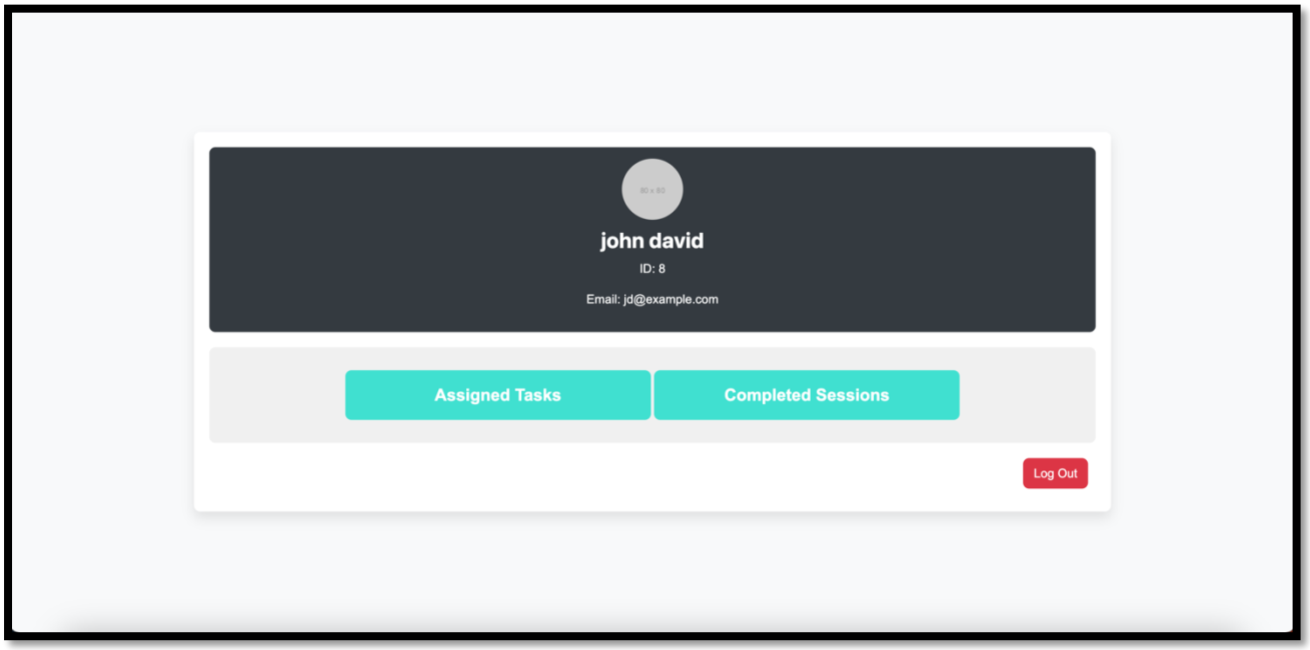
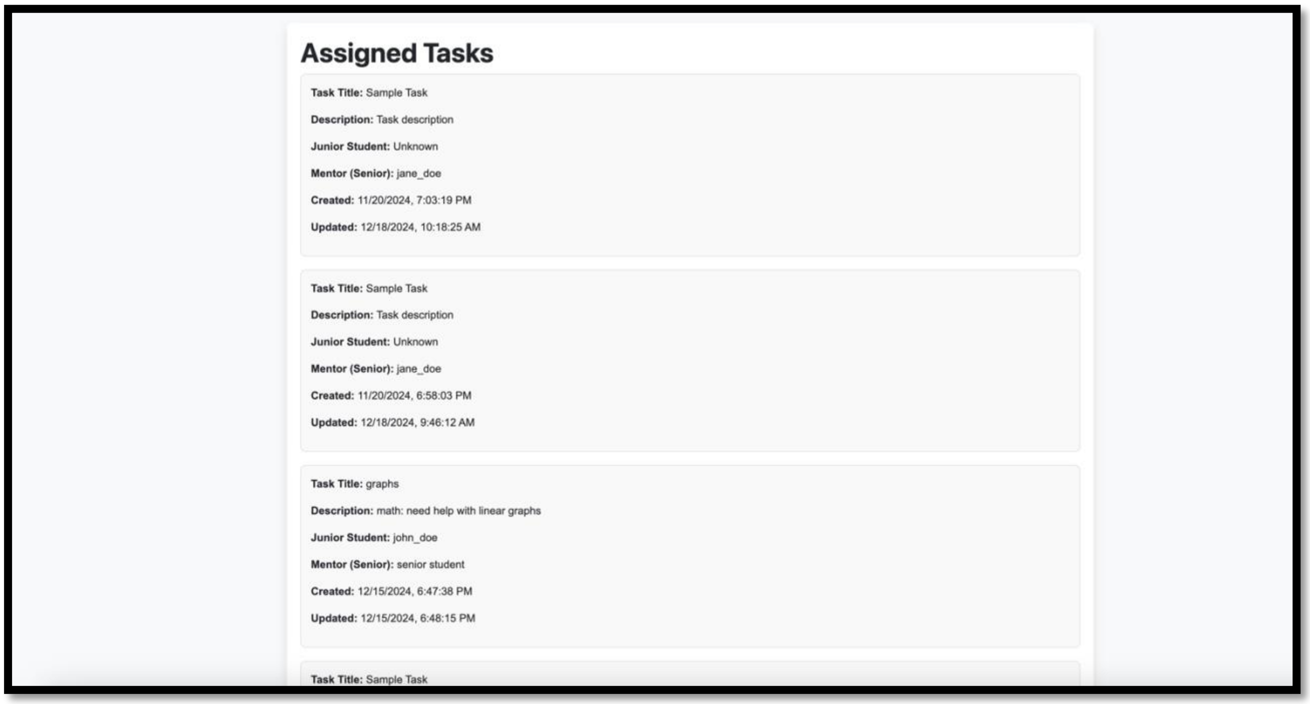


Fig. (9.22): Dashboard for faculty members



Assigned Tasks

Task Title: Sample Task

Description: Task description

Junior Student: Unknown

Mentor (Senior): jane_doe

Created: 11/20/2024, 7:03:19 PM

Updated: 12/18/2024, 10:18:25 AM

Task Title: Sample Task

Description: Task description

Junior Student: Unknown

Mentor (Senior): jane_doe

Created: 11/20/2024, 6:58:03 PM

Updated: 12/18/2024, 9:46:12 AM

Task Title: graphs

Description: math: need help with linear graphs

Junior Student: john_doe

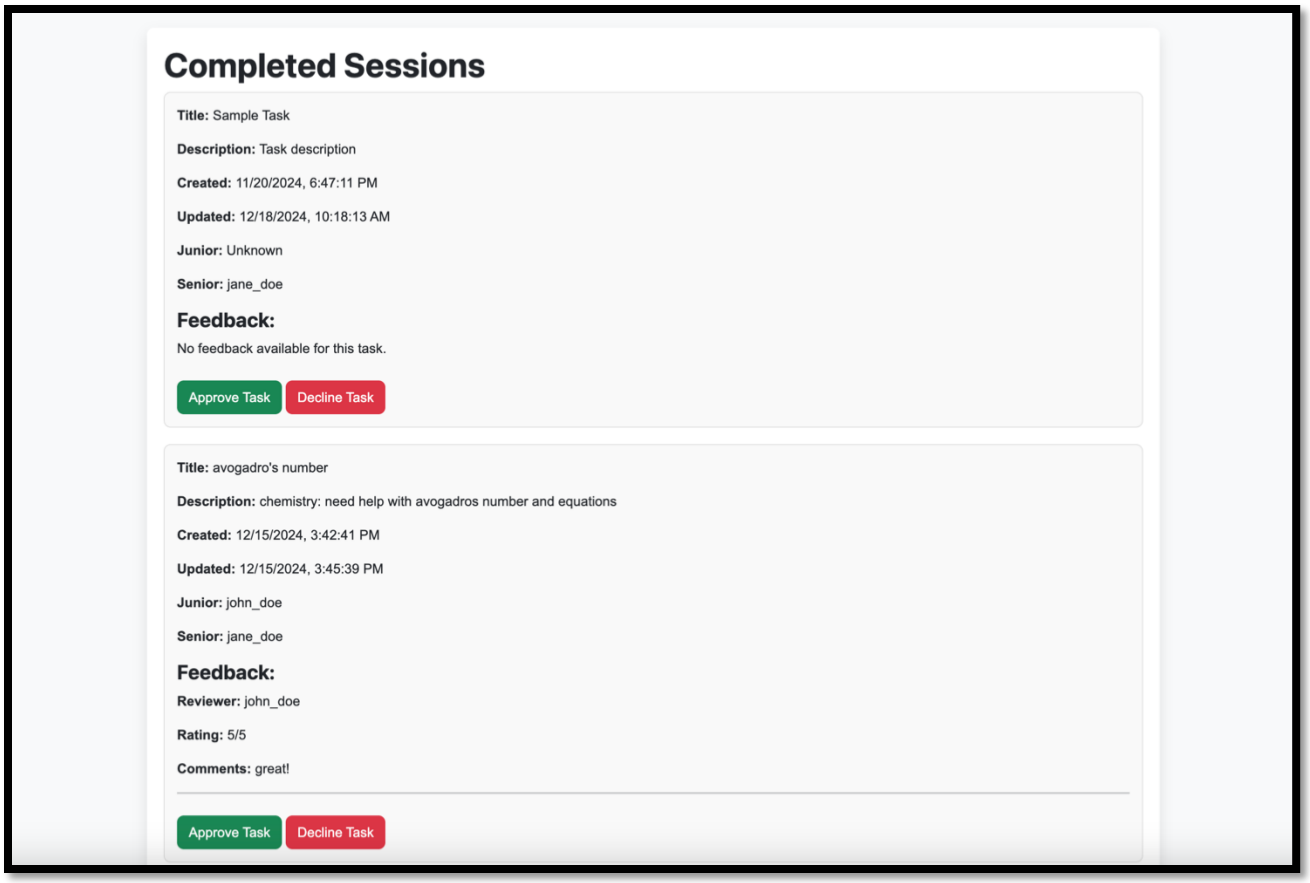
Mentor (Senior): senior student

Created: 12/15/2024, 6:47:38 PM

Updated: 12/15/2024, 6:48:15 PM

Task Title: Sample Task

Fig. (9.23): Assigned tasks page (faculty) where they can see the ongoing tasks and their details


Completed Sessions

Title: Sample Task

Description: Task description

Created: 11/20/2024, 6:47:11 PM

Updated: 12/18/2024, 10:18:13 AM

Junior: Unknown

Senior: jane_doe

Feedback:

No feedback available for this task.

Title: avogadro's number

Description: chemistry: need help with avogadros number and equations

Created: 12/15/2024, 3:42:41 PM

Updated: 12/15/2024, 3:45:39 PM

Junior: john_doe

Senior: jane_doe

Feedback:

Reviewer: john_doe

Rating: 5/5

Comments: great!

Fig. (9.24): Completed sessions page(faculty) where they can view completed tasks, their details and their feedbacks and approve or decline completed tasks, therefore deciding whether to give credit or not

10.Testing (section 2)

First, we will start by addressing our test cases that we used for out program and the category they fall under:

1- Creating Assistance Request

Test ID	Description	Input	Expected Result	Actual results
TC-1	Create request with valid inputs	title="Math Help", subject="Algebra", details="Need guidance"	201 Created. Request created successfully.	pass
TC-2	Create request with missing required fields	title="", subject="Algebra"	400 Bad Request. Error: "All fields are required."	pass
TC-3	Create request without authentication	None	401 Unauthorized. Error: "Unauthorized."	fail

2- Viewing Available Tasks

Test ID	Description	Input	Expected Result	Actual results
TC-4	Fetch tasks relevant to the Senior Student's courses	Valid JWT for Senior role	200 OK. List of filtered tasks returned.	pass
TC-5	Fetch tasks with unauthorized role (e.g., Junior)	Valid JWT for Junior role	403 Forbidden. Error: "Access denied for role: junior."	pass
TC-6	Fetch tasks without providing a token	None	401 Unauthorized. Error: "Unauthorized."	pass

3- Accept Task

Test ID	Description	Input	Expected Result	Actual results
TC-7	Accept a valid task	Valid task_id=1, Senior JWT	200 OK. Task accepted successfully.	Pass
TC-8	Accept a task that doesn’t exist	Invalid task_id=999, Senior JWT	404 Not Found. Error: "Task not found."	pass
TC-9	Accept a task with unauthorized role	Valid task_id=1, Junior JWT	403 Forbidden. Error: "Access denied for role: junior."	pass

4- Approve Task Completion

Test ID	Description	Input	Expected Result	Actual results
TC-10	Approve task with valid input	Valid task_id=1, Faculty JWT	200 OK. Task approved successfully.	pass
TC-11	Approve a task that doesn’t exist	Invalid task_id=999, Faculty JWT	404 Not Found. Error: "Task not found."	fail
TC-12	Approve task with unauthorized role	Valid task_id=1, Junior JWT	403 Forbidden. Error: "Access denied for role: junior."	pass

5-Provide Feedback on Assistance

Test ID	Description	Input	Expected Result	Actual result
TC-16	Fetch notifications for a valid user	Valid JWT	200 OK. Notifications fetched successfully.	Pass
TC-17	Fetch notifications for a non-existent user	Invalid `user_id=999`	404 Not Found. Error: "No notifications found."	Pass

Test ID	Description	Input	Expected Result	Actual result
TC-18	Fetch notifications without authentication	None	401 Unauthorized. Error: "Unauthorized."	Pass

6-Receive notifications

Test ID	Description	Input	Expected Result	Actual result
TC-16	Fetch notifications for a valid user	Valid JWT	200 OK. Notifications fetched successfully.	Pass
TC-17	Fetch notifications for a non-existent user	Invalid `user_id=999`	404 Not Found. Error: "No notifications found."	Pass
TC-18	Fetch notifications without authentication	None	401 Unauthorized. Error: "Unauthorized."	Pass

7-Logging in

Test ID	Description	Input	Expected Result	Actual result
TC-19	Log in with valid credentials	email="user@example.com", password="correctpassword"	200 OK. Token generated successfully.	Pass
TC-20	Log in with invalid credentials	email="user@example.com", password="wrongpassword"	401 Unauthorized. Error: "Invalid credentials."	Pass
TC-21	Log in with missing required fields	email="", password=""	400 Bad Request. Error: "Email and password are required."	Pass

Now let us introduce the traceability matrix to map the use cases to the test cases (note: go back to assignment 1 to see the fully dressed use cases, their IDs and descriptions):

ID	UC-1	UC-2	UC-3	UC-4	UC-5	UC-6	UC-7	UC-8	UC-9	UC-10	UC-11	UC-12
TC-1	X											
TC-2	X											
TC-3	X											
TC-4		X										
TC-5		X										
TC-6		X										
TC-7			X									
TC-8			X									
TC-9			X									
TC-10				X								
TC-11				X								
TC-12				X								
TC-13						X						
TC-14						X						
TC-15						X						
TC-16								X				
TC-17								X				
TC-18								X				
TC-19									X			
TC-20									X			
TC-21									X			

11.Risk Management (section 2)

Due to time constraints, we couldn't implement the Notification and Library parts, but we considered and documented the risks as if they were implemented.

Risk	Probability	Effects	Change in List (deleted/modified/added)
Major sheet modification	1	Some subjects in the old Major sheet could be combined or deleted in the new major sheet resulting in a gap between senior and junior batches.	removed
Library rooms are temporary closed	2	The system's restriction on library rooms as the only permitted spaces has resulted in no sessions taking place.	No modification
Library admin is not available	2	No one is assigning room for sessions	added
Cameras and computers in all library rooms are not working	3	There is no way to verify attendance in the session.	No modification
Outdated database	3	It may take time for recent junior students to be added to the system, and some senior students might face rejection when attempting to accept tasks due to newly enrolled tasks not being visible in the system.	Removed [the university database is always updated]
Sudden disagreement between team members	2	it can cause delays, confusion, and reduce progress	added

Challenge	Probability	Effect	Change in List (deleted/modified/added)
dishonest feedback	4	System goal is unachieved due to manipulation/ faculty doubt its credibility/ unfair advantages for some students, while others may be undervalued.	No modification
library rooms number are small causing late session appointments	3	Late appointments may result in junior students turning to ask other students directly, bypassing the system.	No modification
Senior team members who provide incorrect guidance to juniors	3	leading juniors to receive lower-than-expected grades due to misdirection. As a result, juniors lose trust in the system.	added
Account Misuse or Impersonation	3	Misused accounts could interfere with the proper workflow of the system, such as creating fraudulent task requests, accepting tasks without intention to complete them, or altering session statuses.	added

12.Project Plan – no modification

Week	Milestones	Lamiaa	Ahmed	Seba
1	Research	Research and understand project		
1	Gathering requirements	documenting, and understanding the needs and goals of a project		
2-3	System design	Design UI	Create system architecture and diagrams	Design data flow
4-6	Development phase 1	Develop UI components	Develop task management functionality	Develop feedback system
7-9	Development phase 2	Develop authentication system	Integrate all components	
10	Testing phase 1	Write test case	Implement test cases	Testing and fixing bugs

13.References

Books and Articles:
LucidChart | Diagramming powered by Intelligence. (n.d.). https://www.lucidchart.com/

Fakhroutdinov, K. (n.d.). *Use case diagrams are UML diagrams describing units of useful functionality (use cases) performed by a system in collaboration with external users (actors).* https://www.uml diagrams.org/use-case-diagrams.html

Wikipedia contributors. (2024, June 11). *Sequence diagram*. Wikipedia. https://en.wikipedia.org/wiki/Sequence_diagram

Nitin. (2024, December 2). *10 Top software development Risks and Mitigation Techniques.*https://eluminoustechnologies.com/blog/software-development-risks/

Hipp, D. R. (n.d.). *SQLite documentation*. SQLite. Retrieved from https://www.sqlite.org/docs.ibhtml

PostgreSQL Global Development Group. (n.d.). *PostgreSQL documentation*. PostgreSQL. Retrieved from https://www.postgresql.org/docs/

IBM. (n.d.). *What is an entity relationship diagram (ERD)?* IBM Think. Retrieved from https://www.ibm.com/think/topics/entity-relationship-diagram
SequenceDiagram.org. (n.d.). *Online Sequence Diagram Tool*. Retrieved from https://sequencediagram.org/

Microsoft Support. (n.d.). *Create a UML Sequence Diagram*. Retrieved from https://support.microsoft.com/en-us/office/create-a-uml-sequence-diagram-c61c371b-b150-4958-b128-902000133b26