# Charter Scope and Requirements Document

## Project: Peer Review Application by Learnification Technologies

University Of British Columbia Okanagan

COSC 499 - Summer 2023

Date: May 29th, 2023


**Group Members:**

Charlotte Zhang (charlottezhang6@gmail.com)

Shila Rahman (shilarahman16@gmail.com)

Prabhmeet Singh Deol (arsh.appleid@gmail.com)

Lance Xu (lance924852785@gmail.com)

Sehajvir Singh Pannu (pannusehajvir@gmail.com)

# TABLE OF CONTENTS

# 1. Project Purpose

The purpose of this peer review application is to foster a collaborative learning environment, empowering students to learn from each other and simplifying the management and evaluation of assignments for instructors. This enables students to provide constructive and helpful feedback to each other.

# 2. Project Objective

The objective of this project is to develop a robust web-based peer review application designed to facilitate students enrolled in a same course to participate in an anonymous feedback exchange. The application will allow students to provide constructive feedback on assignments submitted by their peers, as well as receive feedback on their own assignments anonymously on this application. The application aims to improve the learning experience for students by promoting constructive feedback, collaborative work, and fair assessment within the academic community. Key features of the application include assignment submission for assessment, participation in peer assignment assessments with various question types, and the ability to view and receive feedback on assignments.

# 3. Success Metrics

a. Minimum number of clicks to accomplish a task

b. Users can perform the processes of peer evaluation on the platform .
c. Users have a good experience after using the platform features. (using questionnaires)
d. Stakeholders recognize that the final project product meets their expectations

# 4. High-level Requirements

a. Knowledge of the Node.js, Express.js, React.js and MySQL programming languages and frameworks.
b. Experience using Git and Github version control systems to manage the project.
c. Understanding of how docker containers work with docker networks .
d. API testing through postman.
e. CI/CD pipelines to be built through Gradle CI.
f. Production environment setup through ssh.

# 5. Assumptions And Constraints

## Assumptions:

a. The developers will be able to code in React.js and Node.js
b. The capstone team is able to complete all the deliverables in the given timeframe.
c. Stakeholder and team availability is not an issue.
d. Developers will not run into any computer problems.
e. The Project Manager can arrange the schedule reasonably and monitor the product quality effectively.
f. Minimal change of project requirements to finish on time.

## Constraints

a. Time constraint
b. Developers need to learn new concept and language
c. Clients and teams availability

# 6. High-Level Project Description and Boundaries

This project aims to develop a web-based evaluation and feedback system for instructors to access student performance and provide constructive feedback.

a. The platform focuses on students anonymously evaluating other students' assignments and giving feedback. It will not include features unrelated to evaluations, such as course content delivery or student enrollment management.

b. The system will support a predefined set of questions determined by the instructor for evaluations, including multiple-choice, short answer, and rating scale questions. It will not support additional question types that are not within the defined scope.

c. The system will allow instructors to manage evaluations and view student performance. It will not provide administrative features user management, course scheduling, or grade management.

d. The system will be developed for use on web browsers and will not be optimized for specific mobile platforms or devices.

e. The project will focus on the development of the evaluation system and its core functionalities. It will not include extensive customization options or integration with external systems beyond and defined scope.

f. The project will consider integrations with external systems or APIs as necessary for its functioning. However, the project's boundaries will be set to include only the essential integrations, and any additional integrations may be subject to further assessment and approval from the Sponsor.

# 7. High Level Risks

a. Limited number of hours to finish the project
b. Limited number of weekly client meeting
c. No in-person client meeting as the client is not in Kelowna
d. Developer(s) can't pick up React.js, Node.js, express.js to code functionality for the project.
e. Developer(s) might drop the class
f. Developer(s) might get sick and not be able to code in a timely fashion
g. Weak team communication

# 8. Summary Milestone Schedule

| Milestone No. | Deliverables | Due Date |
|---|---|---|
| 1 | Charter, Scope and Requirement Document | May 29, 2023 |
| 2 | Design Document and Presentation | June 5, 2023 |
| 3 | Final prototype and Presentation | August, 2023 (date TBA) |

# 9. Stakeholders List

a. Parsa Rajabi (Project Sponsor)
b. Shila Rahman (Client Liasion + Frontend Developer)
c. Lance Xu (Full stack Developer)
d. Prabhmeet Singh Deol (Full Stack Developer, Integration Lead)
e. Sehajvir Singh Pannu (Full Stack Developer, Team Lead)
f. Charlotte Zhang (Project Manager)
g. University of British Columbia

# 10. Functional and non-functional requirements

## Functional Requirements:

### a. Functional Requirements for Students:

1. **Account Management:**
   a. Students must be able to register an account by providing their name, ID, and email on the application, ensuring accurate capture and storage of their information.
   a. Students must be able to securely log in to the platform using their registered email and password.
   b. Students must have the ability to reset their passwords in case of forgetting or needing to change them, ensuring a secure and accessible password recovery process.

c. Students must be able to log out of the system, ensuring a secure disconnection and safeguarding their account from unauthorized access.
d. Students must have access to a feature that allows them to manage their account details on the platform, including their student name, email, and password, ensuring control over their personal information and account security.

2. **Assignment Management**:
   a. Students must be able to view a comprehensive list of assignments on the platform, displaying relevant details such as assignment title, due date, review status (reviewed or not reviewed), total marks, and final marks, ensuring easy access to assignment information for monitoring and tracking progress
   b. Students must be able to submit assignments only as links (e.g., Google Drive, Dropbox) on the platform.
   c. Students must have the ability to edit their assignment submission on the platform before the deadline, allowing them to make necessary revisions or updates to their work.
   d. The system must perform link validation to ensure that the submitted links for assignments are valid, ensuring the reliability of the submitted work.

3. **Evaluation:**
   a. All students must have the ability to evaluate their teammates through a comprehensive survey that includes matrix questions, short and long answer questions, multiple answer questions, and multiple-choice questions, allowing for both quantitative and qualitative feedback collection.
   b. Each student within a team must be required to provide feedback to all other members of their team, ensuring comprehensive and inclusive evaluation among team members..
   c. If a student evaluating another student does not give a full mark, they must provide a small comment specifying areas for improvement for the particular question, ensuring constructive feedback and opportunities for growth. The

system should not allow a student to proceed until they have provided a comment feedback.

d. Once evaluations become visible to the students, they must have the ability to view the scores and comments associated with their assignment (without revealing the reviewer), ensuring transparency and providing valuable feedback for self-assessment and improvement.

## b. Functional Requirements for Instructors:

**4. Instructor Account:**

a. 'Super admin' will grant 'Instructor' the privileges to register an account in the platform by providing their name, ID, and email allowing them authorized access and identification within the system.

b. Instructors must be able to log in to the system using their registered email and password, ensuring secure access to their account and associated functionalities.

c. Instructors must have the ability to reset their passwords in case of forgetting or needing to change them, ensuring a secure and accessible password recovery process.

d. Instructors must be able to log out of the system, ensuring a secure disconnection and safeguarding their account from unauthorized access.

e. Instructors must have access to a feature that allows them to manage their account details on the platform, ensuring control over their personal information and account settings.

f. Instructors must be able to assign students to the courses

**5. Evaluation Team:**

a. Instructors must have the ability to create evaluation teams/groups

b. Instructors must have the ability to assign students to teams/groups, either during or after the assignment creation process, ensuring flexibility in organizing students into designated teams as needed.

c. Instructors must have the ability to delete students from teams/groups, either during or after the assignment creation process, ensuring the necessary flexibility to manage team/group membership as required.

**6. Assignment Management :**

a. Instructors must have access to create a new assignment under a specific course, allowing them to efficiently design and manage assessments within the designated course.
b. Instructors must have the ability to set the visibility of an assignment, determining whether it is visible to students or not, during or after the assignment creation process, providing control over when students can access and view the assignment.
c. Instructors must have the ability to set and change the full marks and final marks of assignments, ensuring flexibility in grading and accommodating any adjustments by providing “fudge marks”.

**7. Evaluation Survey Management:**

a. Instructors must have access to create evaluation forms and select the questions to include, allowing them to design customized assessment criteria and collect specific feedback based on their evaluation requirements.
b. Evaluations can ONLY be edited/updated BEFORE a student uses that criteria -- once the evaluation has been "used" by a single student, the instructor should not be able to modify it.
c. Instructors must have the ability to toggle or set options that allow students to view the feedback they have received, providing control over the visibility and accessibility of feedback for student learning and improvement.
d. Instructors have the authority to determine whether students can access their feedback in individual response format or as a summarized version, enabling control over the level of detail provided to students for their assessment feedback.
e. The website must provide a visual analysis or representation of the feedback and marks given by students, clearly indicating the source of feedback and

the recipients, facilitating a comprehensive overview of the evaluation process.

f. The evaluation data must have the capability to be easily exported to Excel format, allowing for convenient analysis, storage, and further manipulation of the data as needed.(

## c. System Functional Requirements:

**8. Evaluation**

a. Calculations must be performed for the quantifiable data derived from the feedback, allowing for the allocation of specific weightings to each question as set by the instructor, ensuring accurate assessment of scores based on predetermined weightings.

b. A notification mechanism, such as email or through the application, must be implemented to alert students when their feedback becomes visible to them, ensuring timely communication and enabling students to access their assessment results promptly.

**9. Visibilty**

a. The system must provide a clear and organized list of courses for students, displaying relevant details such as course name, course number, and course image, ensuring easy navigation and access to course information

b. The system must provide a clear and organized list of assignments for students, displaying relevant details such as assignment title, due date, status (completed or not completed), full marks, and final marks, ensuring easy tracking and monitoring of assignment progress.

c. The system must provide a clear and organized list of evaluations for students, displaying relevant details such as assignment title, due date, status (reviewed or not reviewed), full marks, and final marks, ensuring easy access to evaluation information for monitoring and tracking progress.

d. The system must provide a clear and organized list of grades for students that display relevant details for each assignment, such as assignment title, overall

grades, view feedback (if the student being evaluated did not receive a full mark for some evaluation question).

e. The system must provide a preview of the assignment and a summary of the evaluation in a student view, enabling students to review their work and understand the feedback they have received.
f. When instructors click on "View Feedback," the system must provide a preview of the assignment and a summary of the individual evaluation with the corresponding student names in an instructor view, facilitating the assessment and review process for instructors.
g. When clicked on "Overall Grade," the system must provide a clear and organized rubric of assignments for students and instructors, displaying various evaluation criteria such as overall grade and different sections grade for example Section A (Team Performance), Section B (Respect and Behaviour) etc.

# Non-Functional Requirements:

## a. Security

a. The system must grant access to user accounts when users enter the correct username and password, ensuring secure authentication and authorized access to the platform.
b. The application must not create an account for a user until they create a strong password (At least 12 characters long but 14 or more is better. A combination of uppercase letters, lowercase letters, numbers, and symbols)

## b. Compatibility

a. The system must be designed to run on computers and tablets, considering the impact of display screen size on the user interface (UI) to ensure optimal user experience and usability, while not requiring specific adaptation for mobile devices such as cell phones.

## c. Localization

a. The time of assignments deadline must be based on the Instructors set time zone.

**d. Usability**

a. Users must be able to predict the meanings of icons, such as inferring that tapping a button with a picture of a magnifying glass will open a search bar, ensuring intuitive and user-friendly interface design.
b. The user interface elements, layouts, and interactions must be consistent across different devices and screen dimensions, ensuring a unified and seamless user experience regardless of the device used.

**e. Accessibility**

a. The web application must pass the audit test, such as Google Lighthouse, ensuring adherence to best practices, performance optimization, accessibility, and other relevant criteria for an optimal user experience.

## Technical Requirements:

1. The front end system will be written in ReactJs, Next.js libraries and TailwindCss. These would be the most important frameworks for the front end.
2. Back end will be written using Node.js and Express.js following MVC design choice.
3. The database driver for this application will be My SQL.
4. Gradle CI will be used to run and build the CI/CD pipeline.
5. The project must make use of Git/Github
6. Every piece of code that is written should be associated with a ticket number and the ticket number should be noted in the branch name of Github
7. All github branches must go through pull request (PR) review and be reviewed by other team members AND pass automatic tests BEFORE they're merged into development branch
8. Development branch must be smoke-tests using automatic testing before PR'ed into production branch

# 11. Project Management Methodology and Workflow

## 1. Methodology: Agile

The peer evaluation application project will be implemented using an Agile methodology. Agile methodologies promote iterative and incremental development,

allowing for flexibility, collaboration, and continuous improvement throughout the project lifecycle. The project will follow the Scrum framework, which includes sprints, daily stand-up meetings, and frequent feedback loops.

## 2. Workflow

1. Product Backlog Refinement:
    a. Collaborate with stakeholders to identify and prioritize features and user stories.
    b. Break down user stories into smaller tasks or requirements and estimate their effort.
    c. Continuously refine and update the product backlog.
2. Sprint Planning
    a. Select a set of user stories from the product backlog to be completed in the upcoming sprint.
    b. Define the sprint goal and use the GitHub project board to manage the features and tasks of the project.
    c. Organized user stories and tasks into different columns on the project board, such as “To Do”, “In Progress”, “Testing”, and “Completed.” (Kanban Board)
    d. Create a new branch for each functionality or feature.
    e. Implement the task functionality based on the test cases.
3. Sprint Execution:
    a. Develop the selected user stories in short iterations of 7 days.
    b. Conduct daily 15 minutes stand-up meetings to track progress, discuss challenges, and plan the day’s work.
    c. Regularly review and demonstrate the completed user stories to stakeholders.
4. Sprint Review:
    a. Perform thorough testing of the implemented functionality to verify its behavior and ensure it meets the requirements.
    b. Request code reviews from other team members to get feedback and ensure code quality.
    c. Once the functionality is complete and thoroughly tested, address a pull request.
    d. Merge the branch into the dev branch.
5. Sprint Retrospective:
    a. Discuss improvement, and potential solutions after the end of the 7 day sprint.

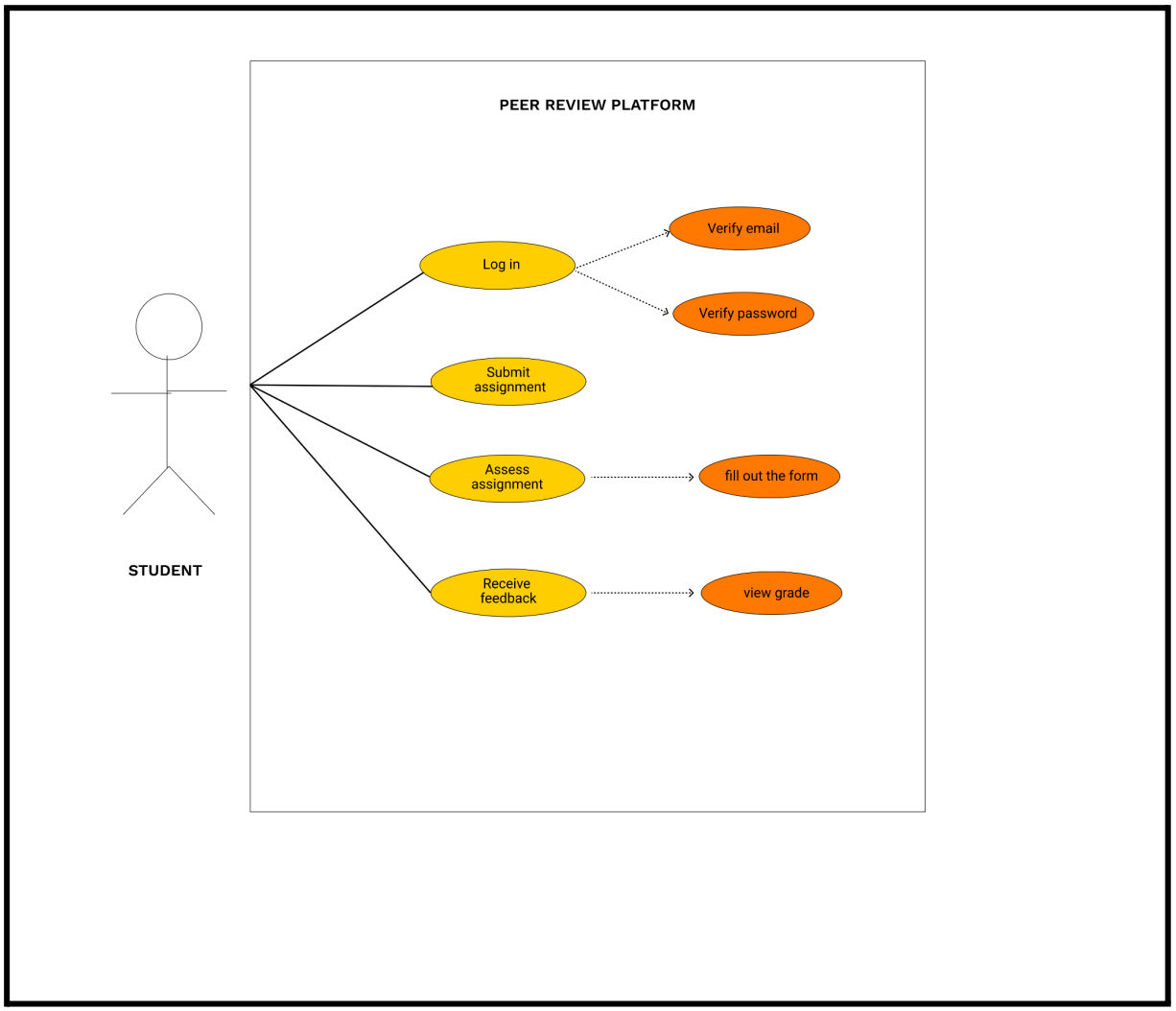
b. Formulating a strategy for refactoring in the upcoming lifecycle if necessary.

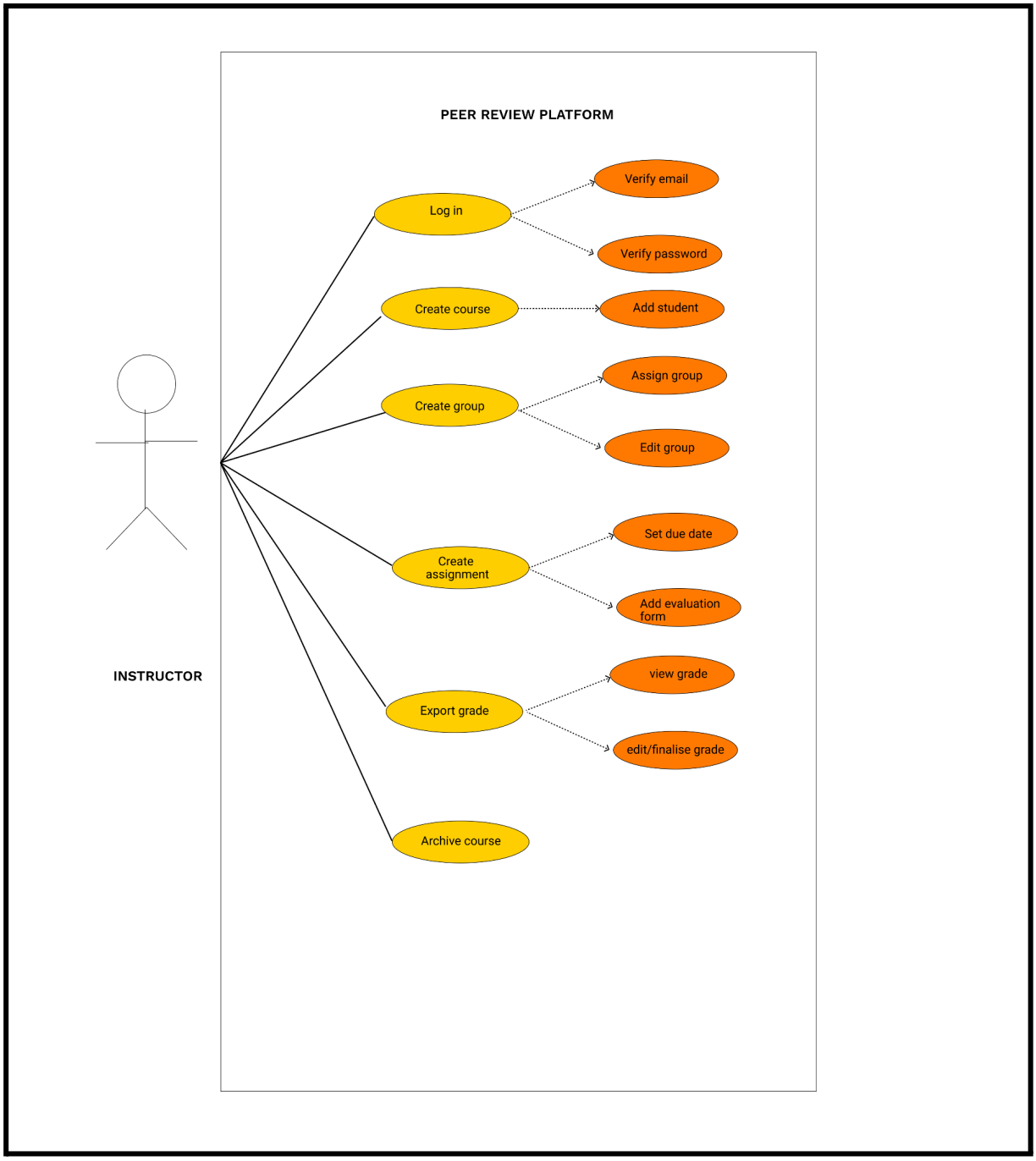
6. Repeat Steps 2-5

# 12. User Group and UML use case

Our platform has two main user groups:

1. **Students:** Students are able to login, submit assignment, assess assignment and receive feedback
2. **Instructors:** Instructors are able to login, create courses, add students, create groups, assign groups, edit grade, export grade and archive courses.




PEER REVIEW PLATFORM
Verify email
Log in
Verify password
Create course
Add student
Assign group
Create group
Edit group
Set due date
Create assignment
Add evaluation form
INSTRUCTOR
view grade
Export grade
edit/finalise grade
Archive course

# 13. Work Breakdown Structure

| **Task List** | **Owner** | **Estimated Hours** | | | | | **Average Estimate** |
|---|---|---|---|---|---|---|---|
| | | Prabhmeet | Sehajvir | Shila | Lance | Charlotte | |
| **Documentation** | | | | | | | |
| Charter and scope | | | 2 | | 1 | | |
| Software description | | 4 | 1 | | 0.5 | | |
| Set of the Criteria | | | | 1 | | | |
| Functional and Non-functional requirements | | | 1 | | | 3 | |
| Technical requirements | | | 1.5 | | | | |
| User requirements | | | | | 1 | | |
| Description of workflow and methodology | | | | | 1 | 2 | |
| UML case diagrams | | | | 2 | | | |
| Work breakdown structure | | | | | 2 | | |
| Monitor project progress and track milestones | | | 3 | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| **Baisc Functionalities** | | | | | | | |
| **All Users** | | | | | | | |
| Login page | | | | | | | |
| Signup page | | | | | | | |
| Retrieve password | | | | | | | |
| View list of courses registered | | | | | | | |
| | | | | | | | |
| **Students** | | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| View list of Courses they are enrolled in | | | | | | | |
| View list of Assignments | | | | | | | |
| View list of evaluation surevys | | | | | | | |
| View list of Assignment grades | | | | | | | |
| Assignment submission | | | | | | | |
| Feedback summary page | | | | | | | |
| | | | | | | | |
| **Instructors** | | | | | | | |
| View list of Assignments | | | | | | | |
| Assignment management | | | | | | | |
| Evaluation survey management | | | | | | | |
| View list of Evaluation teams | | | | | | | |
| Evaluation team management | | | | | | | |
| Feedback management | | | | | | | |
| View of the feedback summary | | | | | | | |
| Exporting Feedback | | | | | | | |
| Notification system | | | | | | | |
| | | | | | | | |
| **Evaluation Form Functionalities** | | | | | | | |
| **Questions** | | | | | | | |
| Matrix Question | | | | | | | |
| Ranking Question | | | | | | | |
| Short and long answer question | | | | | | | |
| Check box questions | | | | | | | |
| Multiple choice questions | | | | | | | |
| | | | | | | | |
| **Others** | | | | | | | |
| Quantifiable data calculation | | | | | | | |
| | | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Database** | | | | | | | |
| General | | | | | | | |
| Setup and Configure the database server | | | | | | | |
| Implement data models and database server | | | | | | | |
| Relationship setup for tables | | | | | | | |
| Docker containor for the database | | | | | | | |
| | | | | | | | |
| **Design and Implementation** | | | | | | | |
| Student database | | | | | | | |
| Instructor database | | | | | | | |
| Course database | | | | | | | |
| Course assignment question database | | | | | | | |
| Course assignment team database | | | | | | | |
| Course assignment question student database | | | | | | | |
| UML diagram | | | | | | | |
| | | | | | | | |
| **User Interface** | | | | | | | |
| General | | | | | | | |
| Login Signup page | | | | | | | |
| Evaluation Form | | | | | | | |
| Account summary page | | | | | | | |
| Icons and clear naviagtion | | | | | | | |
| | | | | | | | |
| **Students** | | | | | | | |
| Dashboard UI for course list | | | | | | | |
| Dashboard UI for assignment list | | | | | | | |
| Dashboard UI for assignment feedback list | | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Instructors** | | | | | | | |
| Assignment management page | | | | | | | |
| Evaluation survey management page | | | | | | | |
| Evaluation team management page | | | | | | | |
| Feedback management page | | | | | | | |
| Feedback summary page | | | | | | | |
| | | | | | | | |
| **Others** | | | | | | | |
| Compatibility with different screen size | | | | | | | |
| | | | | | | | |
| **TESTING** | | | | | | | |
| General | | | | | | | |
| Setup testing framework | | | | | | | |
| Define test cases | | | | | | | |
| Write test scripts | | | | | | | |
| Run tests | | | | | | | |
| Ensure website displays correct data | | | | | | | |
| Vallidate database functionalities | | | | | | | |

# 14. Approvals

| | |
|---|---|
| Client Signature | Date |
| Team Representative | Date |