Project Document

peer8: Peer Review Web Application

Version 1.0

04/22/2023

CPSC 4900 Spring 2023

Prepared by:

peer8

Project Manager / Scrum Master: Cole Cagle

Application Architect: Francisco Ayala

Security Manager: Jannat Saeed

Development Manager: Noah Taylor

Database Administrator: Yhara Mendoza

Quality Assurance Manager: Cuinn Owens

peer8: Peer Review Web Application

SECTION 1 - INTRODUCTION

The peer8: Peer Review Web Application was designed to allow faculty members at universities and other educational institutions the ability to provide peer reviews to their classes for students. The peer reviews serve as a means for students to grade other students they worked with on projects. The feedback is automatically compiled by the application and can be used for grading purposes for faculty members. The peer reviews can be customized as needed, based on the projects that students are working on. Accounts will need to be registered to use the application. The two options are student and instructor, with the instructor account needing to be verified by email at their respective institution. The instructor accounts will have control over the student accounts that are registered with their classes. The interface was designed to be user-friendly for all users and provide a dashboard that is simple to navigate.

1.1 PURPOSE

The purpose of the application is to automate the gathering of peer review feedback and compile the data for grading purposes. The process of using Microsoft Excel or Google Sheets can be laborious and time-consuming for faculty members. The peer8 application erases the need for using these applications and can allow faculty to provide grades efficiently to students. The application gives the ability of customization to faculty members and alerts students through email to complete peer reviews.

1.2 SCOPE

The scope of this application is for faculty/instructors at educational institutions that use student feedback as a means for grading. The faculty/instructors will have to have their accounts verified before using the application. Students are also within the scope and will register for accounts. They will automatically be added to the classes upon registering.

1.3 TERMINOLOGY

Cascading Style Sheets (CSS): A standardized system for designing web pages that are available on the web; designing includes components such as colors, fonts, layouts, etc.

Cyber Security: The implementation of protective features during software development to protect user data from malicious actors and actions.

Back-End: "Behind the scenes" processes for an application, such as creation, deletion, and retrieval of data; usually these processes are not seen by the user.

Dashboard: Visual representations of information in one place so that users can see the goals that need to be achieved.

Database: Stores user data in an organized and structured manner for application processing.

Database Management System (DBMS): A tool that is used to quickly and efficiently retrieve data that is stored in a database.

Front-End: The graphical interface of an application that is designed to be user-friendly.

Functional Requirement: A statement that describes how a system should respond when given user input.

Gantt Chart: A visual representation tool that is used by a project manager to create and assign tasks to complete a project to meet a deadline; also called a project plan or project schedule.

Hypertext Markup Language (HTML): A standardized system for formatting web pages that are available on the Web.

Input: Information that is given to a system by a user, usually through a form, that is then processed by a system.

Nonfunctional Requirement: A statement that is concerned with meeting the criteria of a system.

Output: Information that is returned to a user after being processed; based on the information given to the system by the user at a previous time.

Peer Review: Serves as a means to allow peers to rate or give feedback on one another for the goal of improving teamwork.

PostgreSQL: An open-source relational DBMS that uses the SQL language for processing and storing data.

Security Plan: Outlines the policies and procedures implemented to secure user data from unauthorized users.

Structured Query Language (SQL): A standardized language for manipulating and managing data.

Template: A created document that can be retrieved at later dates to be used again.

Test Plan: Implemented during software development to check the application for any bugs/errors when the application is processing data.

Unified Modeling Language (UML): A standardized system for creating models for a system during the design phase so that models can be implemented during development.

User Interface (UI): Graphical illustrations/designs that allow users to interact with information systems; the goal is to be user-friendly.

User Stories: Obtained from conversations with stakeholders; usually these are issues that will serve as the basis for solving the problems through software development.

Virtual Machine: A computer system created using software on one physical computer in order to emulate the functionality of another separate physical computer.

Web Application: A software that is executed through the use of a web browser, such as Google Chrome or Microsoft Edge.

Web Browser: An application that allows users to visit websites, such as Google Chrome or Microsoft Edge.

Web Server: A computer that is responsible for delivering web pages to users of a web browser.

SECTION 2 - MANAGEMENT

Project Manager / Scrum Master: Cole Cagle

Responsible for the management and planning of the project. Scheduled meetings with the stakeholder and the development team at peer8. Guided the discussion of meetings and documented the meeting minutes. Created and assigned tasks to the development team to meet the project deadline.

Application Architect: Francisco Ayala

Responsible for the designing of the application by using tools such as UML diagrams and Figma. Provided updates to the project team about the current state of the design and any issues that were encountered. Would provide any questions to the project manager to note for meetings with the stakeholder about the design.

Security Manager: Jannat Saeed

Responsible for the implementation of security into the design and development of the project. Created a security plan that covers the appropriate policies and procedures that were discussed as concerns from the stakeholder and development team. Provided updates to the project team about the creation of the security plan during the development process.

Development Manager: Noah Taylor

Responsible for managing the development of the project. Communicated closely with the application architect to adhere to the processes designed in UML diagrams and implemented these processes into the code. Provided updates to the project team about the development process and any issues that needed to be addressed.

Database Administrator: Yhara Mendoza

Responsible for the creation of the database for the project. Designed the structure of the database for the purposes of the management of data. Provided updates to the project team during the development process. Closely communicated with the development manager and security manager to adhere to the security principles created to protect user data.

Quality Assurance Manager: Cuinn Owens

Responsible for creating and implementing a test plan for the project. Used created test cases throughout the development process to make sure the project was addressing issues. Provided updates to the project team about any issues that appeared during the testing phases. Communicated closely with the project manager and development manager so that solutions could be created and implemented.

2.1 GENERAL DESCRIPTION

The designing and development of the peer8: Peer Review Web Application was completed using methods that adhered to Agile/Scrum project management and planning. The tasks were created and assigned to team members based on their roles and responsibilities that were defined by the project manager. The stakeholder and manager were provided updates during the designing and development phases of the project. The updates were provided in-person and remotely through the use of the application, Zoom. The project manager created a communication plan for the project team to follow so that the project was not held up for any reasons. Discussions were held in-person at the University of Tennessee-Chattanooga library and remotely through the use of the application, Discord. If any urgent issues needed to be communicated, those messages were sent on the application, GroupMe, to alert the project team. When tasks were in-progress or completed, these were communicated to the project manager so that the Gantt chart could be updated for the project team to view. Documents and files were created and shared through Google Drive so that the project team could make any modifications as needed. The final product was presented and demonstrated to the stakeholder in-person and met the project deadline.

2.2 SOFTWARE VALIDATION

The main issue that is addressed by the peer8: Peer Review Web Application is the automation of gathering peer review feedback and compiling this data for instructors. The instructors can use this feedback for grading purposes, or to critique students that are working in teams on assignments/projects. Instructors will have the ability to use customization in multiple aspects through the application. Instructors can create their own templates and scales as needed. The templates can be saved and used at later dates as needed. They can also create groups and place students within their respective groups. Instructors can also remove students in the event that they drop the course so that unresponded feedback does not affect student grades. The feedback is secure and will remain confidential between the instructor and their students. The implementation of both instructor and student accounts for using the application allows for the differentiation of the described features. The user interface is user-friendly and implements a dashboard that differs depending on the account type.

2.3 CONFIGURATION MANAGEMENT PLAN

The project manager was responsible for understanding the various components and requirements of the project and providing updates to the stakeholder. The initial requirements were gathered in the initial meeting with the stakeholder over Zoom. The project manager and team then discussed the requirements. The scope of the project was created and the baseline goals were identified in order to meet the criteria from the stakeholder. Other meetings were held with the stakeholder to provide updates and demonstrations of functionality of the application. Any issues that the stakeholder identified in these meetings, would lead to changes. The changes were immediately discussed with the team at the conclusion of stakeholder meetings to create solutions for addressing the issues. These changes would then be shown and communicated to the stakeholder for feedback. The documentation of these configuration processes from discussions allows for review of the application for auditing purposes for any future issues that may arise.

2.4 PROJECT SCHEDULE

The planning of the project was conducted by the project manager. A Gantt chart was created to document the major tasks of the project that needed to be completed in order for the project to be successful and meet the deadline set by the stakeholder. Each major task had predecessors that were identified and needed to be completed before the next task could be started. Progress for each task was measured on a percentage scale from 0-100. The time that was spent on each major task was estimated in days and the assigned resources were allocated to members of the project team. Start and end dates were also implemented and tracked using a dated schedule within the Gantt chart.

A product backlog was created by the project team. This identified each task in the project that was created based on the user stories from the stakeholder. The tasks were assigned priority levels: L (Low), M (Medium), and H (High). Each task was given a start date and estimated completion time scaled to hours. The status of the tasks were tracked using: NS (Not Started), IP (In-Progress), and C (Completed).

The communication plan was implemented by the project manager to identify key communication events. These events would involve the project team and the stakeholder or just the project team. The audience of the events, communication method, and dates were identified. Any important notes would also be included with the associated communication event.

SECTION 3 - REQUIREMENTS

This section will cover the system requirements for the peer8: Peer Review Web Application, which are crucial for ensuring that the system is dependable, efficient, and effective.

3.1 FUNCTIONAL REQUIREMENTS

Functional requirements describe the specific functions that the application must be able to perform to satisfy the needs of its users. They are essential to ensure that the software meets the expectations of the stakeholders and can provide the necessary features and capabilities to support the educational objectives of the instructors and students using the application.

3.1.1 Functional Capabilities

Class creation and management: Instructors can create and manage classes within the peer8 web application. Instructors can create a new class, add students to the class, and set up peer review assignments for the class. With the ability to create classes, instructors can organize their course materials and students in a structured manner. This is a critical feature for instructors as it enables them to organize and manage their students efficiently, and it provides a centralized location for all class-related activities, such as peer review assignments and student submissions.

Display dashboards: The dashboards of the peer8 application provide both instructors and students with a location to view important information and updates related to their classes and assignments. Instructors can view the progress of their students, track assignment submissions, and assess their performance with calculated scores. Students can view their own progress, upcoming assignments, and receive feedback from their peers and instructors. The dashboard is designed to be user-friendly, with easy navigation and intuitive display of data. By providing a clear and organized view of class information, the dashboard feature helps to enhance communication and engagement between instructors and students.

Group creation and management: Group creation and management allows for instructors to easily organize their classes and assign peer review assignments to specific groups of students within a class. Instructors can create new groups and add students to them. Once groups are created, instructors can assign peer review assignments to specific groups or to the entire class. The application also allows instructors to manage their groups by adding or removing students, renaming groups, or deleting groups altogether. This feature streamlines the process of managing classes and ensures that peer review assignments are assigned to the correct group of students, leading to a more efficient and effective learning experience for both instructors and students.

Login: The login functionality provides secure access to the user accounts. The login process begins when a user enters their login information, which is then validated to ensure that it is in the correct format. If the user's account exists in the database, they are granted access to their respective dashboard, where they can access and manage their peer reviews and submissions. The login functionality is critical in ensuring the confidentiality and integrity of user data, as it ensures that only authorized users have access to their account and the information contained within it.

Peer review assignment creation and management: Instructors use the application to create and manage peer review assignments for their students. This functionality enables students to evaluate each other's work, which helps them develop critical thinking skills and provides valuable feedback for their assignments. With the peer review assignment creation and management feature, instructors can set assignment guidelines, select the students who will participate in the review, and provide specific instructions for the review process. The application ensures that the peer reviews are anonymous, which promotes honest and constructive feedback. Once the peer review assignment is created, instructors can track the progress of their students' submissions and review the feedback provided by their peers. This feature also allows instructors to manage the grading process of peer review assignments, ensuring that the grades are fair and consistent.

Peer review assignment completion: Students are able to submit their peer review assignments and for their peers to evaluate the submissions. The system is designed to ensure that each student receives feedback on their assignment from their peers, which helps improve their skills and encourages collaboration among classmates. Students are required to provide feedback on the assignments of their peers based on a set of predetermined criteria. The system collects and aggregates this feedback, and generates a score for each student's submission. Once the evaluation process is complete, the scores are made available to the instructor for review. This process allows instructors to monitor the progress of their students and evaluate their work in a more comprehensive manner.

Registration: Instructors and students can create accounts and gain access to the peer8 platform's features. During registration, users are required to provide personal information, such as their name and email address, which is stored in the application's database. This information is used to verify the user's identity and to enable communication between the user and the platform. Once a user has completed registration, they can log in to the platform and access its features depending on their account type. The secure registration process ensures that users can quickly and easily access the application and that their personal information is protected.

Score aggregation: Score aggregation involves the calculation and presentation of student scores based on their performance in peer reviews. The system collects scores from all the student reviewers and aggregates them into a final score for each student. The system then displays the scores for each student in a clear and concise format that is easily understandable. This information allows the instructor to evaluate student performance and make informed decisions on grading, feedback, and further actions. This functionality is essential for the effective implementation of the peer review process and enables instructors to make accurate and unbiased assessments of student work.

Student enrollment: The student enrollment process in the peer8 application is designed to be simple and automated. When an instructor creates a class, they have the option to add students to the class. Once the students are added to the class, they are automatically registered in the system and are able to access it once they log in. This automated process ensures that all enrolled students have access to the necessary course materials and assignments, and that their progress can be tracked within the application. It also helps to minimize errors and inconsistencies that can occur when enrollment is done manually.

User account management: Managing authorized user accounts is done using a directory file. This directory file contains all the necessary information about each user, such as their name and email. The file is accessible only by administrators who have the necessary privileges to access and modify user accounts.

User authentication: User authentication is a crucial process that involves verifying the identity of a user who is trying to access the application. This process involves the use of a username and password, which is cross-checked against the user database stored in the application's backend. If the verification is successful, the user is granted access to their account and associated functionalities. To ensure an additional layer of security, the application also incorporates measures like two-factor authentication during registration to prevent unauthorized access.

3.1.2 System Inputs

Account creation information: This refers to the details required to create a new user account in the peer8 application. It includes the user's first and last name, email address, a secure password, and verification code. By collecting and verifying this information during the account creation process, the application can establish trust between the user and the system, allowing for secure and reliable use of the application.

Class information: The class information refers to the details of the courses offered by UTC, including the course name and course number. The system requires this information to assign students to the correct class, create and manage assignments, and calculate grades. It is also used to track the progress of each student in their respective course, allowing instructors to identify areas of improvement or additional support needed for their students. The class information is stored in a database and can be updated by authorized personnel, such as the institution's administrators. Accurate and up-to-date class information is critical for the successful operation of the peer8 web application and ensuring an efficient and effective educational experience for all users.

Group information: Instructors have the ability to create and manage groups of students within the application, allowing for efficient organization and management of peer review assignments. When creating a group, the instructor can define the group name and add or remove students. This information is stored in the database and is accessible to the instructor throughout the peer review process. The ability to manage groups of students allows for targeted and specific assignments to be created, ensuring that students are able to receive and provide feedback to their peers in a meaningful and effective way.

Instructor and student data (directory file): The directory file is a crucial component of the application, as it contains the list of users (both instructors and students) who are authorized to sign up for and access the application. The directory file acts as a gatekeeper, ensuring that only authorized users can create accounts and log in to the application. In addition to containing user account information such as usernames, passwords, and contact information, the directory file also contains permissions for each user. An instructor may have permission to create peer review assignments and view submitted reviews, while a student may only have permission to complete and submit assignments. The directory file is securely stored on the server and can only be accessed by authorized personnel with proper clearance and authentication.

Login information: Login information is entered by both instructors and students and includes an email and password. This information is used to authenticate users and ensure that only authorized users can access the system. It is important that users keep their login information confidential to prevent unauthorized access to the system.

Peer review template: A peer review template is created by the instructor and includes the questions assigned as well as the grading parameters that will be used by students to assess their peers' work. The instructor may create a new template or modify an existing one, depending on the assignment requirements. The peer review template serves as a guide for students during the review process and ensures that the evaluation is consistent and fair. It also helps instructors to monitor the quality of the feedback given by students and identify any areas where students may be struggling with the assignment.

Peer review assignment details: When an instructor creates a peer review assignment, they provide details such as the title, the deadline for submission, and the students it should be assigned to. The students are then notified of the assignment and can access the details of the task, including the submission deadline and the criteria for assessment. As students complete their reviews, the instructor can track their progress and view the scores and comments submitted by each student.

Peer review submission details: This refers to the information that a student submits as part of a peer review assignment. The student is asked to provide an overall score or rating for the work they reviewed. The scores are used to calculate a grade for each student based on the feedback received from their peers. This may also include comments regarding a review of the content, organization, clarity, and presentation of the work, as well as any suggestions for improvement.

3.1.3 System Outputs

Two-factor authentication email and code: Two-factor authentication (2FA) is an important security feature in the peer8 web application that prevents unauthorized users from signing up for an account. When a user signs up for an account, they are sent an email containing a unique code that they must enter to verify their identity. This code is generated randomly and is only valid for a short period of time. Once the code is entered correctly, the user is granted access to the account.

Aggregated scores: These scores are calculated by the system by averaging out the scores given by students. Once the scores are aggregated, the system presents them to the instructor by displaying them on the dashboard. This allows the instructor to quickly identify students who are excelling, struggling, or performing at an average level.

Dashboards: The dashboards provide a centralized location for users to view and manage their tasks and assignments. The Instructor Dashboard provides an overview of the courses and classes that they are teaching, as well as the students enrolled in those classes. From the dashboard, instructors can create and manage peer review assignments, view and grade student submissions, and monitor student progress. The Student Dashboard, on the other hand, provides students with an overview of the courses they are enrolled in and the assignments they need to complete. Students can access peer review assignments and submit their reviews directly from the dashboard. Both dashboards provide an intuitive and user-friendly interface that allows users to quickly access the information they need and complete their tasks efficiently.

Notifications: Notifications provide timely updates to users about the status of their assignments, scores, and other important events. For instance, when an instructor creates a peer review assignment, the website notifies all students in the class about the new assignment. When a student submits their peer review assignment, they receive a notification confirming the submission. The instructor is also notified when all students in the class have submitted their peer review assignments, and the calculated scores are available for review. These notifications ensure that users are kept informed and can take necessary actions as needed.

Peer review reports: The Peer Review Reports, which are generated in the form of an Excel spreadsheet, contain the peer review submissions from students. These reports provide a comprehensive view of the feedback received from the students. The reports typically include the name of the students and a detailed breakdown of the scores and who they were submitted by. The Peer Review Reports can be used to identify trends and patterns in the feedback provided by students, allowing instructors to identify common areas of concern or issues that need to be addressed in future assignments.

3.2 NONFUNCTIONAL REQUIREMENTS

Non-functional requirements refer to the characteristics that describe the overall behavior of a software system, rather than its specific features or functions. These requirements are not directly related to what the software does, but instead focus on how well it performs..

3.2.1 Performance Considerations

Two-factor authentication code generation: A 2FA code is generated and sent via email is required to sign up for peer8 is an essential security feature that adds an additional layer of protection to the user account. Generating and sending an email with a unique code for each user can take some time and resources, which may affect the user experience. To mitigate this impact, the system should use a reliable email delivery service that can handle a high volume of emails efficiently. While 2FA is an important security measure, it should be implemented in a way that does not significantly degrade the performance of the application.

API design: API design is the process of defining the interface and behavior of an application programming interface (API). A well-designed API can make it easier for developers to use an application or service by providing clear and consistent methods for accessing and manipulating data. Another important aspect of API design is security, which involves implementing authentication and authorization mechanisms to ensure that only authorized users can access sensitive data. As the needs of the system and its users evolve, the API may need to be updated to add new functionality or fix bugs. It is important to ensure backward compatibility with older versions to avoid breaking existing applications or services that rely on the API.

Database design and optimization: A well-designed database structure can significantly enhance the application's speed and scalability, while poorly designed databases can lead to slow queries and system crashes. To ensure database performance, it is important to monitor and tune the system regularly. This includes identifying and fixing slow queries, optimizing database settings, and regularly performing database maintenance tasks such as indexing, backups, and updates.

Login authentication: The authentication process typically involves verifying the user's identity through a combination of a username and password. User passwords should be stored securely using techniques such as password hashing to protect against security attacks. Login authentication is a critical component of any application that requires users to access sensitive information. By balancing security and performance considerations, developers can create a login authentication process that is both secure and efficient.

Performance testing: Performance testing is an essential part of the development process to ensure that the application can handle a high volume of requests and provide a fast and responsive user experience. By identifying and addressing performance bottlenecks early on in the development process, developers can ensure that the application performs well under different scenarios and meets the performance expectations of its users.

Server infrastructure: Server infrastructure refers to the underlying hardware and software components that support an application or website. A well-designed server infrastructure is critical to ensuring that an application can handle a high volume of traffic, maintain high availability, and provide good performance. A well-designed server infrastructure can help to ensure that an application is available and responsive to users, even under heavy load or adverse conditions.

Security measures and considerations: Implementing security measures is essential to protect user data and prevent unauthorized access to the system. The application should have a robust authentication mechanism that verifies the user's identity before granting access to sensitive data. It is important to regularly update the application and its dependencies to ensure that any security vulnerabilities are addressed promptly. The security of the application should be a top priority, and all security measures should be implemented in a way that does not compromise the performance or usability of the application.

3.2.2 User Interface (UI)

The user interface (UI) is the part of a software application or website that the user interacts with. It encompasses all of the visual and interactive elements, such as buttons, menus, forms, and icons, that allow the user to navigate and perform tasks within the application. A well-designed user interface can greatly improve the user experience by making it intuitive, efficient, and enjoyable to use. It should be easy to understand and navigate, with clear labels, logical organization, and consistent design. The peer8 user interface includes the login and registration windows, dashboards for instructors and students, screens for creating and submitting peer reviews, and screens for viewing and managing students and assignments.

3.2.3 Language

The application is currently only supported by the English language. It is intended through further development and updates to include other languages so that other countries can use the application.

3.2.4 Operating Environment

The application is meant to be used by educational institutions for the purpose of evaluating students on a customizable criteria. Only licensed users should be able to use the application as appropriate. It currently is supported on Windows operating systems as well as MacOS. A mobile version of the application is currently being developed so that users can access the application with a mobile device. This allows for peer reviews to be conducted anywhere and at any time. Instructors can also check the scores at any given time. If future operating systems are created that become popular, then the development team will look into adding compatibility with those new systems.

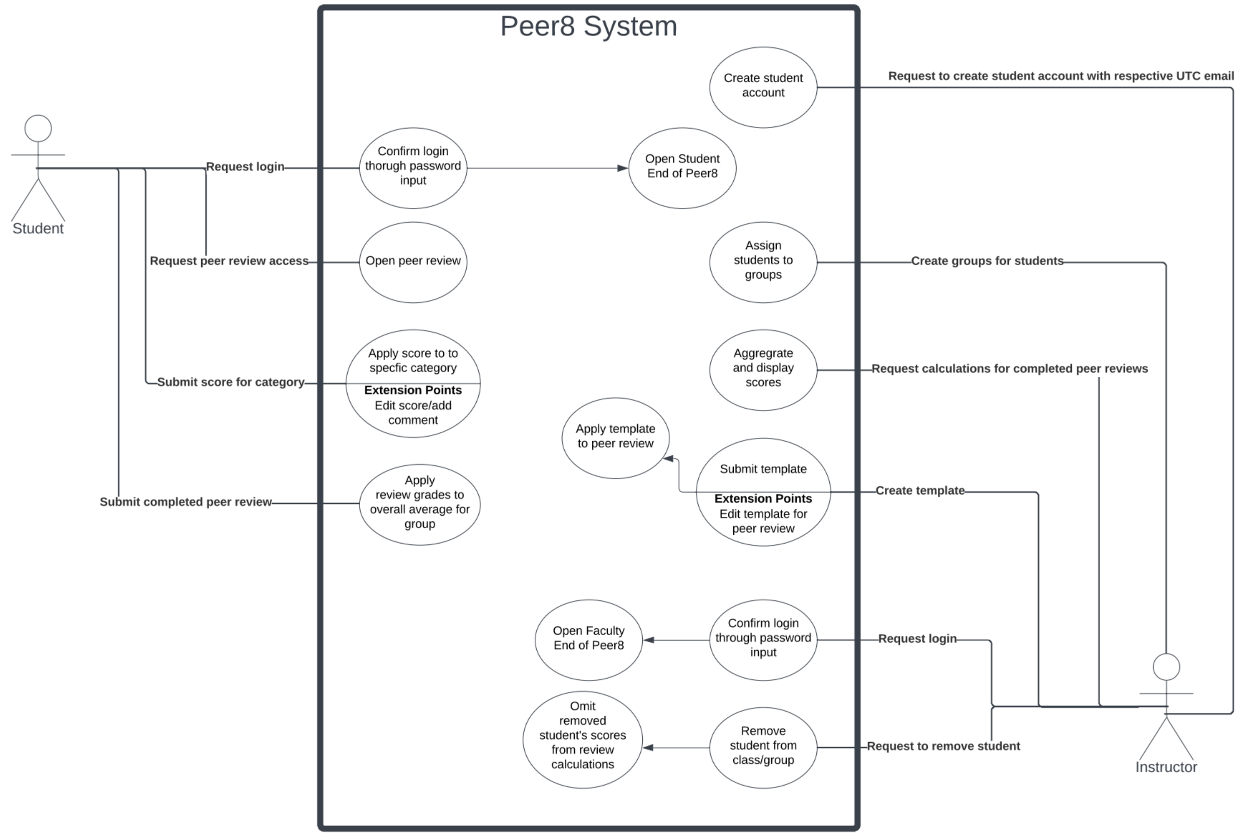
SECTION 4 - DESIGN

This section contains the UML diagrams, ER Diagrams, and other technical diagrams and documentation used by the development team to create and document the peer8 Web Application System.

4.1 UML DIAGRAMS

This section contains all the UML diagrams used by the development team to create and document the peer8 Web Application System.

4.1.1 Use Case UML Diagram


Peer8 System
Create student account
Request to create student account with respective UTC email
Student
Request login
Confirm login thorugh password input
Open Student End of Peer8
Request peer review access
Open peer review
Assign students to groups
Create groups for students
Aggregrate and display scores
Request calculations for completed peer reviews
Submit score for category
Apply score to to specfic category
Extension Points
Edit score/add comment
Apply template to peer review
Submit template
Extension Points
Edit template for peer review
Create template
Submit completed peer review
Apply review grades to overall average for group
Open Faculty End of Peer8
Confirm login through password input
Request login
Omit removed student's scores from review calculations
Remove student from class/group
Request to remove student
Instructor

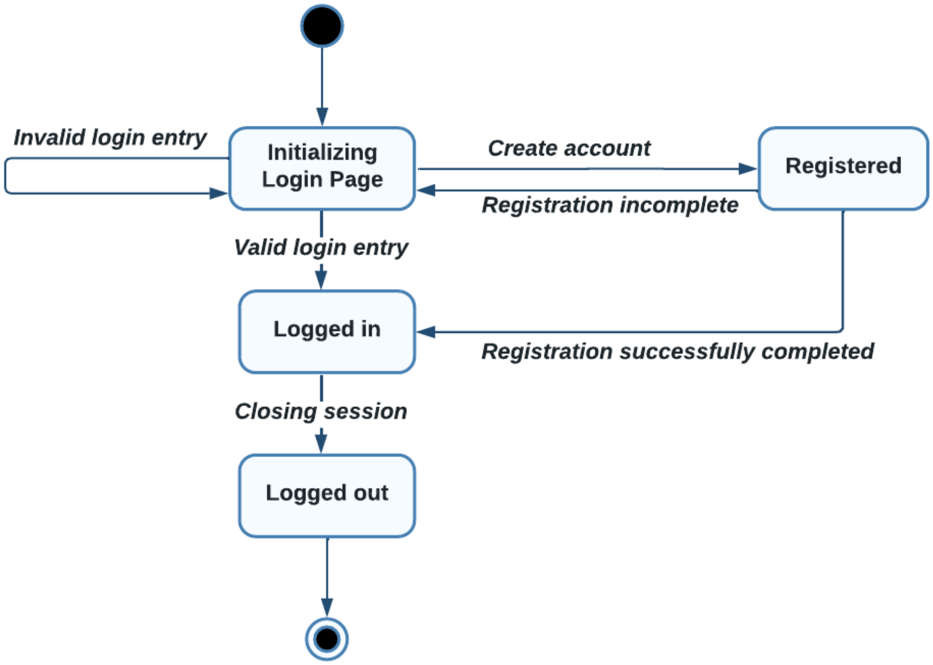
This diagram illustrates the two actors in the peer8: Peer Review Web Application System. There are two external actors, the “Student” actor, and the “Instructor” actor. The first thing that both the Student and Instructor actors have to do before completing any other task in the peer8 system is to complete the “Request Login" process. The rest of the explanation for this diagram will assume that either of the actors has already completed this process.

There is typically only one Instructor actor per class. The Instructor is the one that creates peer review assignments for the Student actors to complete. Instructors are also able to create and edit templates for peer reviews, add or remove Students from classes, request grade scores from the peer reviews submitted by Students, assign students to groups within a class, and omit certain Student submissions.

There are typically several Student actors per class. The Student actors are the ones to fill out the peer review assignments and submit them to the Instructor. That is the only responsibility of the Students, however, they are allowed to go back and edit their submissions and resubmit them before the deadline.

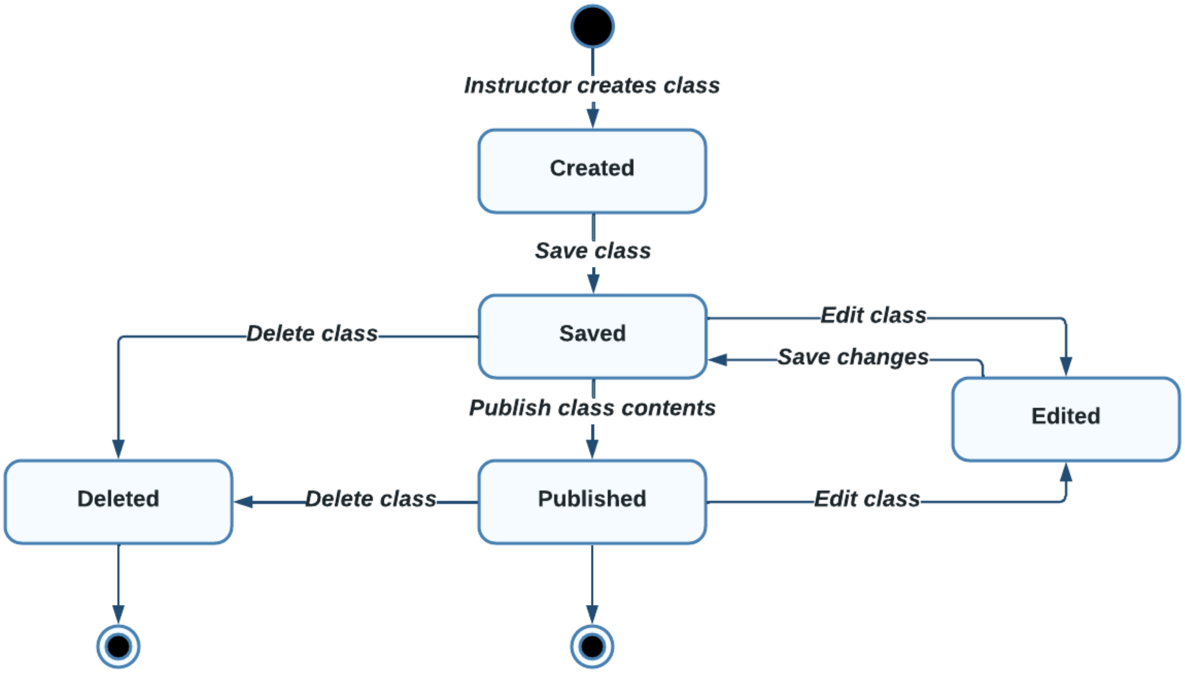
4.1.2 State UML Diagrams

4.1.2.1 Account Login and Creation Status



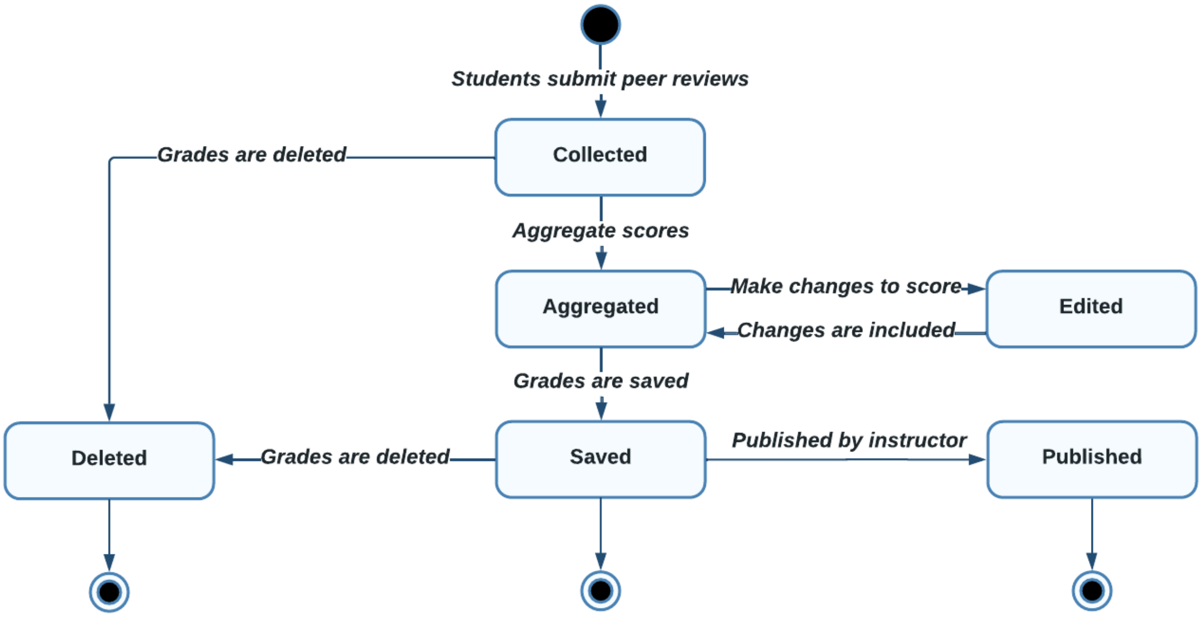
This diagram illustrates the change in the states of account creation, logging in, and logging out. This starts with the initialization of the login page. On this page, the user has two options, either log in with a preexisting account, or create one. If the user chooses to log in with a preexisting account, it will ask for your login credentials. If the user inputs valid login credentials, the account state will be changed to “Logged in." If the user instead chooses to create an account, the page will be redirected to a create-account page. After the user has successfully created an account, the account state will be set to “Logged in” as well. From there, the user does whatever task they are trying to complete. When the user closes the session, the account is changed to the “Logged out” state.

4.1.2.2 Classes Status



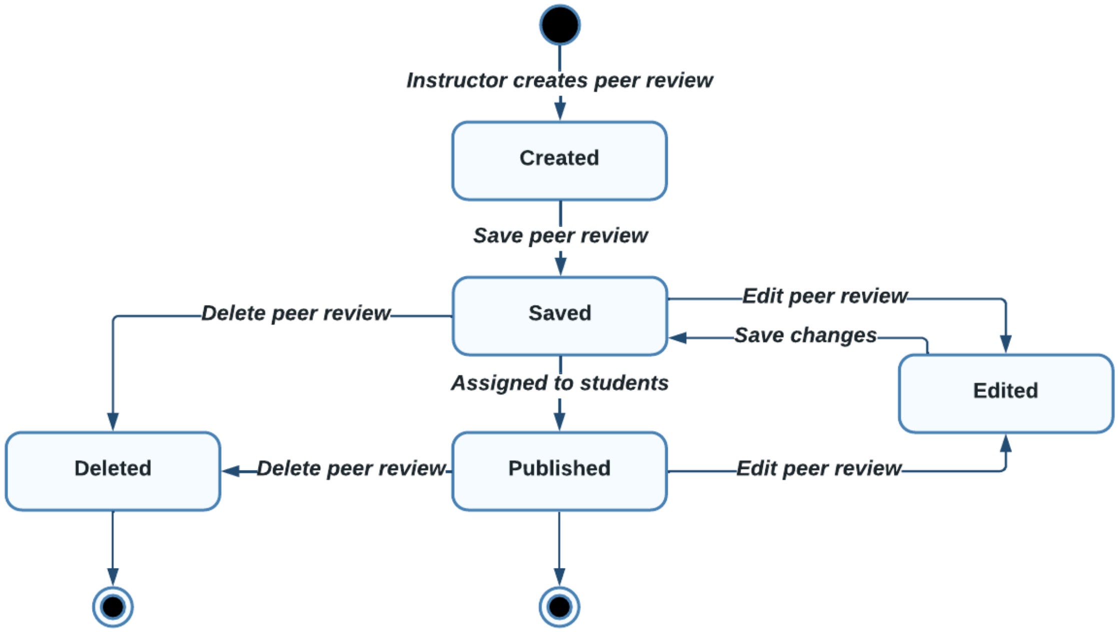
This diagram illustrates the change in the state of classes by the instructor. First, a class must be created by an instructor account. Once that is done, the class status will be set to “Saved." From there, the instructor can edit the class by adding or removing students, publishing or removing peer review submissions, or grouping students into groups. As the instructor is editing the class, it changes states to “Edited." When the instructor decides to save the changes made to the class, the state changes back to “Saved." From there, the instructor also has the ability to publish, delete, or edit the class again. If the instructor chooses to delete, the state changes to “Deleted." If the instructor decides to publish the class, students are able to see the newly edited class, and the state is changed to “Published." From the “Published” state, the instructor has the ability to edit the class or delete it.

4.1.2.3 Grades Status



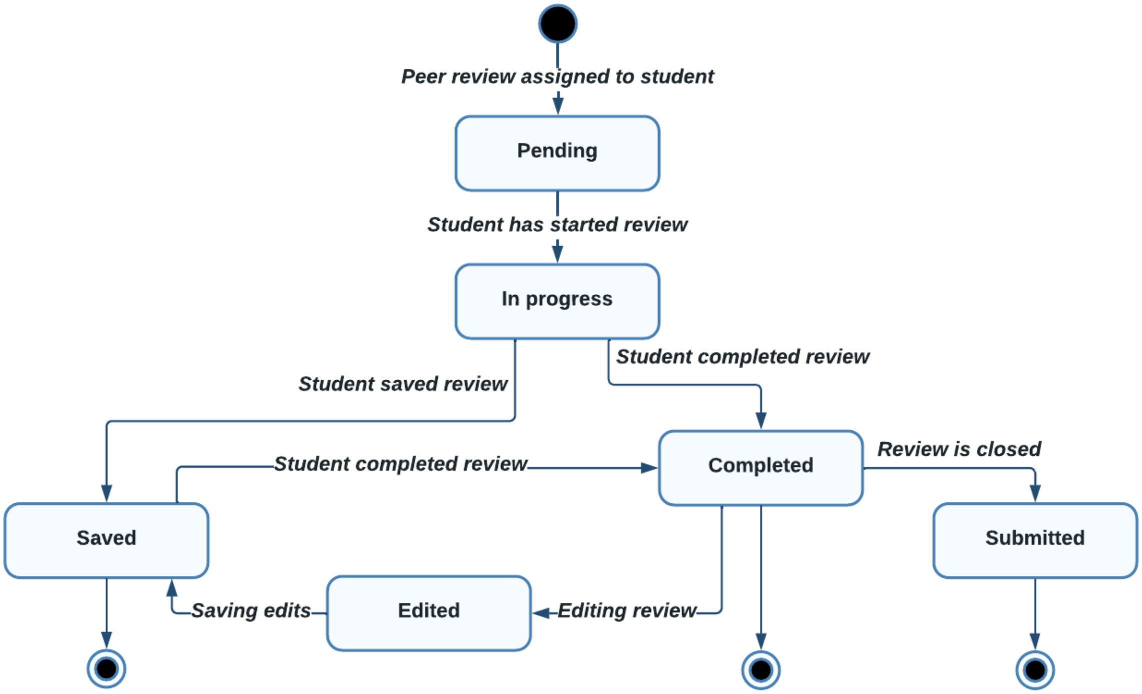
This diagram illustrates the change in the state of grades. First, the students must submit their completed peer review assignment. After that is completed, the grades will have the state "Collected." Then, the system calculates the aggregate scores for each student. This changes the state to "Aggregated." The instructor can then choose to save the scores or make changes to them. If the instructor decides to make changes, while the instructor is editing them, the state is set to "Edited." After the instructor is satisfied with the changes, the changes are then adjusted and the scores are set back to the "Aggregated" state. From there, the grades are saved, and the "Saved" state is applied. From there, the instructor can choose to either publish or delete. If the instructor chooses to delete, the scores are set to the "Deleted" state. If the instructor instead decides to publish the scores, they are published, and the "Published" state is applied.

4.1.2.4 Peer Review Assignments Status by the Instructor



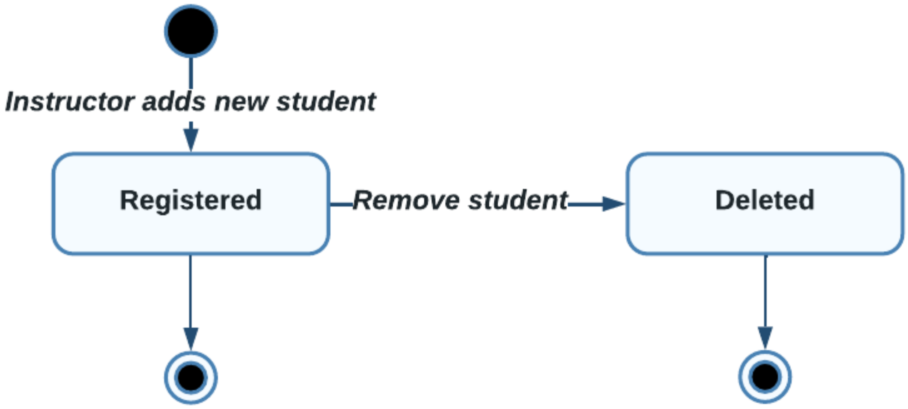
This diagram illustrates the change in the state of the peer review assignments by the instructor. First, a peer review assignment must be created by an instructor account. Once that is done, the assignment's status will be set to “Saved." From there, the instructor can edit the assignment by editing, adding, or removing questions, as well as editing the allocated weighted points for each question and the total amount of points possible for that assignment. As the instructor is editing the assignment, it changes states to “Edited." When the instructor decides to save the changes made to the assignment, the state changes back to “Saved." From there, the instructor also has the ability to delete it, assign it to the students, or edit it once again. If the instructor chooses to delete, the state changes to “Deleted." If the instructor decides to assign it to the students, the students are able to see the newly edited assignment, and the state is changed to “Published." From the “Published” state, the instructor has the ability to edit the assignment or delete it.

4.1.2.5 Peer Reviews Assignments Status by the Student



This diagram illustrates the change in the state of the peer review assignments by the students. First, the peer reviews are assigned to the students. This starts them in the “Pending” state while the student has yet to fill them out before the deadline. Once the student has started the review, the assignments state is changed to “In progress." From there, the student can save the amount filled out. That would change the state of the assignment to “Saved." The student can also choose to complete the assignment. This would change the state from “In progress” to “Completed." Immediately after the student closes the assignment, the state is changed from “Completed” to “Submitted." However, if the student wishes to edit their submission, the student can reassess the assignment. When editing, the state of the assignment will be “Edited." From there, the student saves it again and submits it.

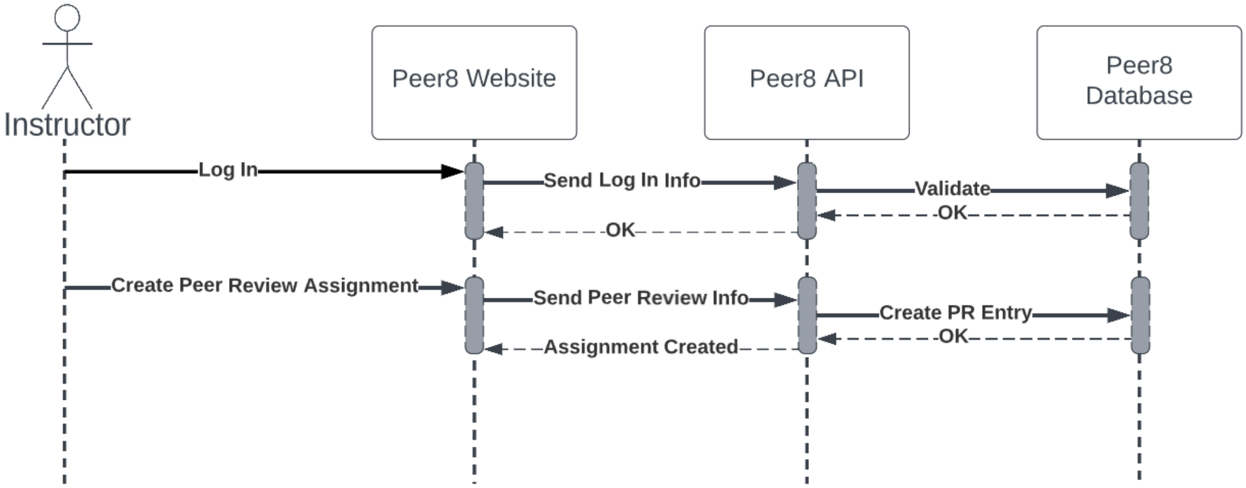
4.1.2.6 Student-in-Class Status



This diagram illustrates the state of a student account being either in a class or not, from the point of view of an instructor. First, the student is added to the class by an instructor. The state of the student is “Registered.” If the instructor wishes to remove the student, the state of the student is changed to “Deleted.”

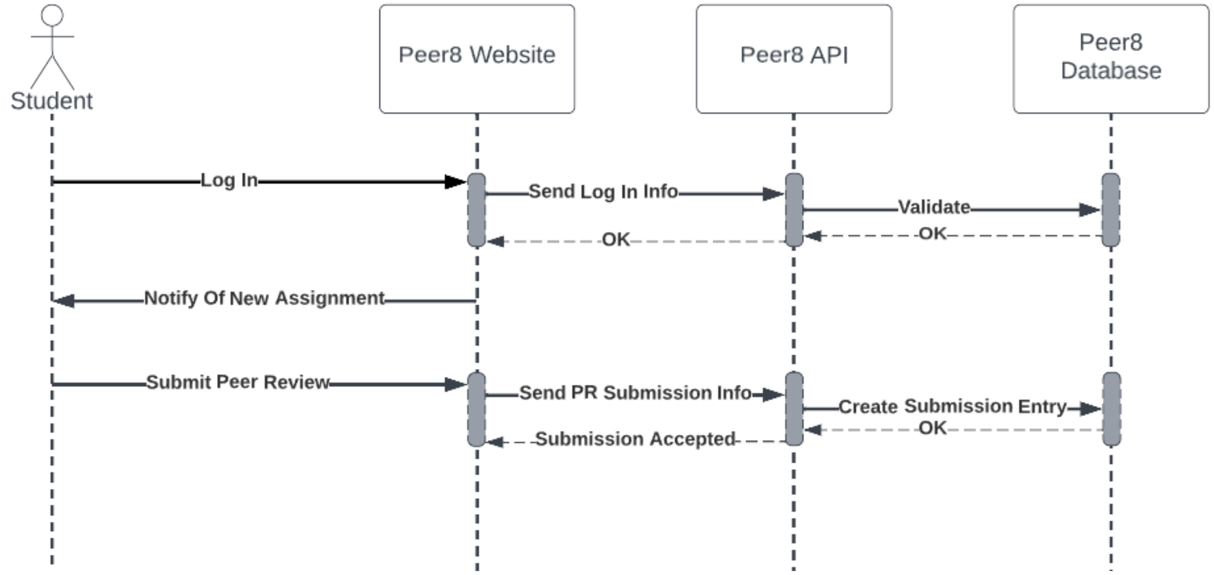
4.1.3 Sequence UML Diagrams

4.1.3.1 Creation of Peer Review Assignment by Instructor



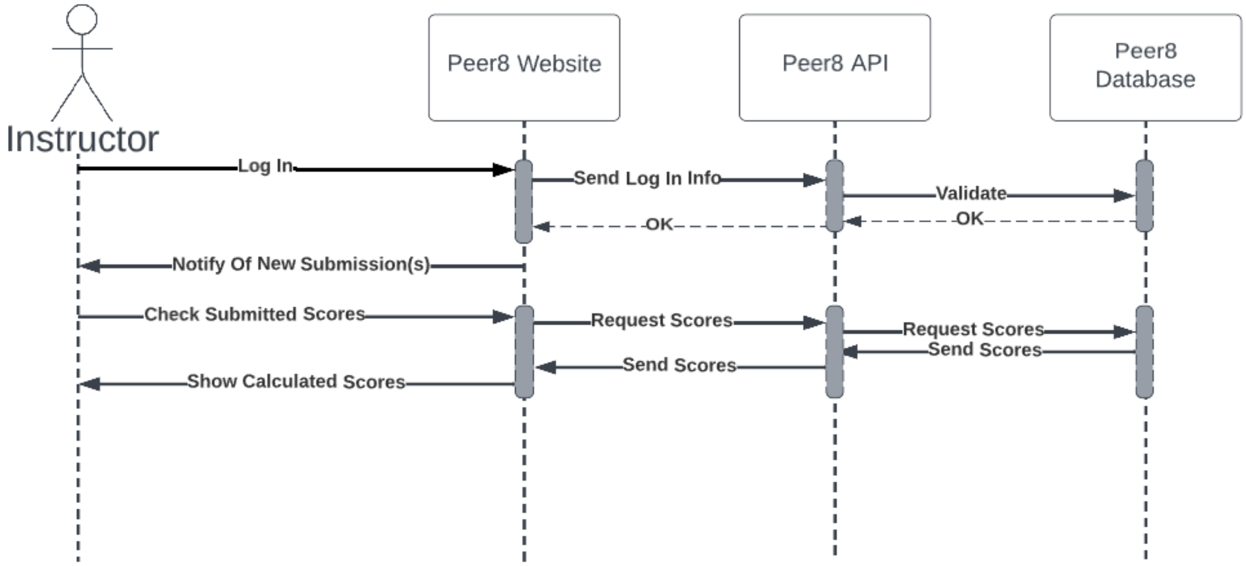
This diagram illustrates the sequence of events from the Instructor logging in, to a peer review assignment being created. This diagram assumes that the Instructor starts at the login screen of the website. First, the Instructor enters their login information into the corresponding input boxes. Then, the API validates that they are in the correct format. Then the API checks to see if there is a corresponding entry in the accounts table in the database. If it is a valid account, the database sends an OK through the API, and the Instructor is logged into the website. After that, the Instructor chooses to create a peer review. The instructor creates the peer review, the API makes sure the inputs are valid, then the database stores it. The website then notifies the Instructor that the assignment has been created.

4.1.3.2 Fulfillment of Peer Review Assignment by Student



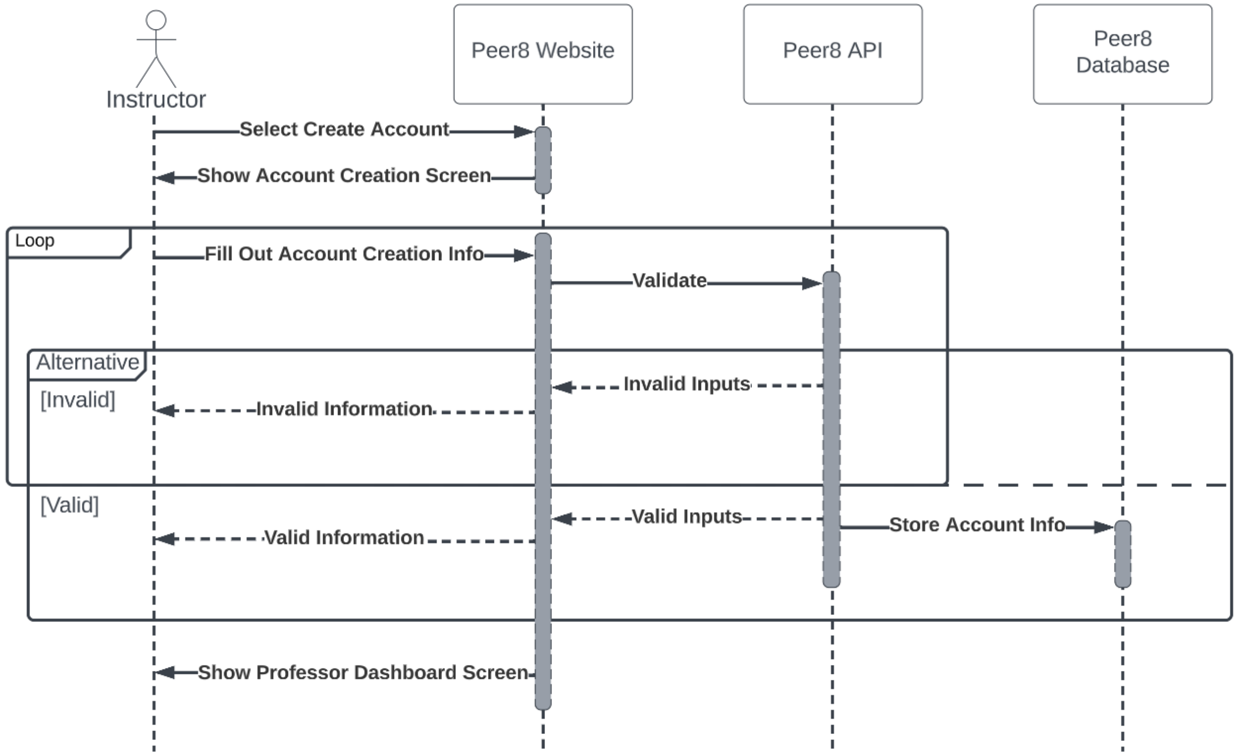
This diagram illustrates the sequence of events from the Student logging in, to the Student submitting a peer review assignment. This diagram assumes the Student starts at the login screen of the website. First, the Student enters their login information into the corresponding input boxes. Then, the API validates that they are in the correct format. Then the API checks to see if there is a corresponding entry in the accounts table in the database. If it is a valid account, the database sends an OK through the API, and the Student is logged into the website. After that, the website notifies the student that there is a new peer review assignment to be completed. The student chooses to fill out a peer review assignment and submit it. The website sends the assignment information to the API, and the API makes sure that the inputs are in the correct format. After that, the submission is entered into the database. Then, the website notifies the student that the peer review has been submitted.

4.1.3.3 Retrieval of Graded Scores by Instructor



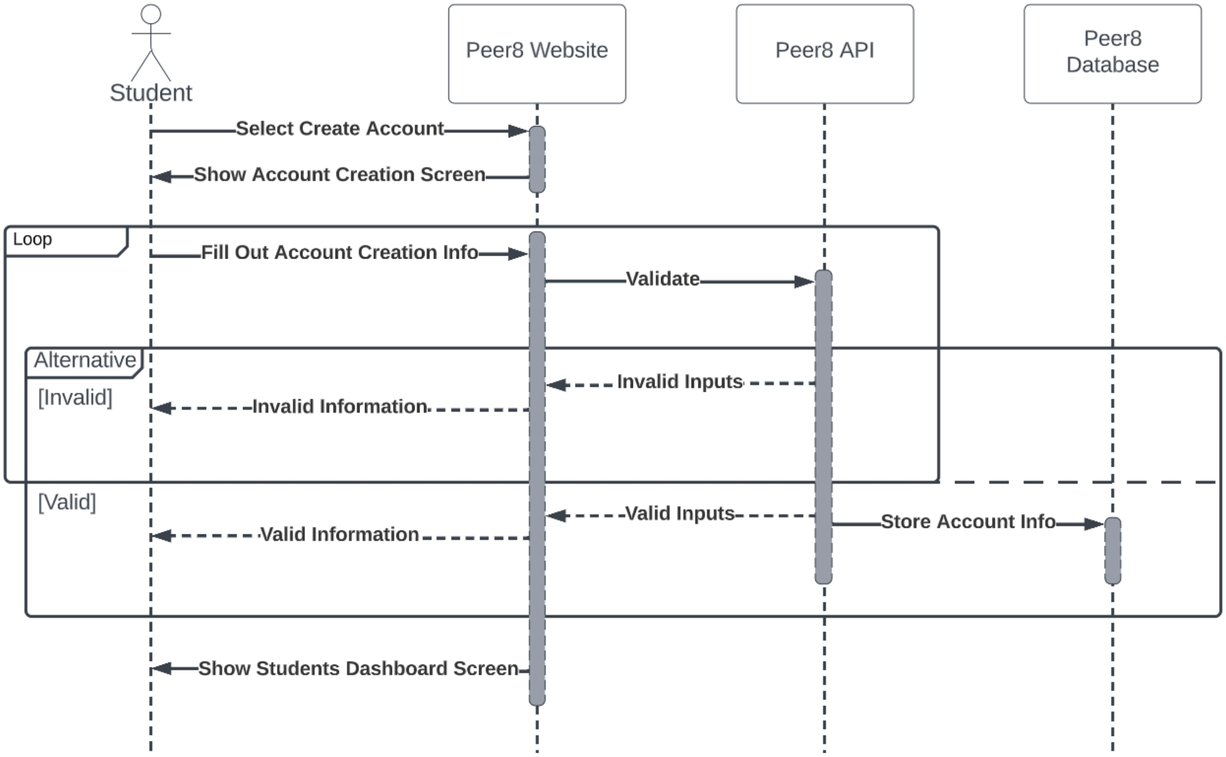
This diagram illustrates the sequence of events from the Instructor logging in, to the calculated scores of each student being shown to the Instructor through the website. This diagram assumes that the Instructor starts at the login screen of the website. First, the Instructor enters their login information into the corresponding input boxes. Then, the API validates that they are in the correct format. Then, the API checks to see if there is a corresponding entry in the accounts table in the database. If it is a valid account, the database sends an OK through the API, and the Instructor is logged into the website. After that, the website notifies the Instructor of the new submissions. The Instructor then chooses to check the submitted scores. The website requests to see the scores through the API. The API accesses the scores in the database and sends them to the website to be viewed by the Instructor.

4.1.3.4 Instructor Account Creation



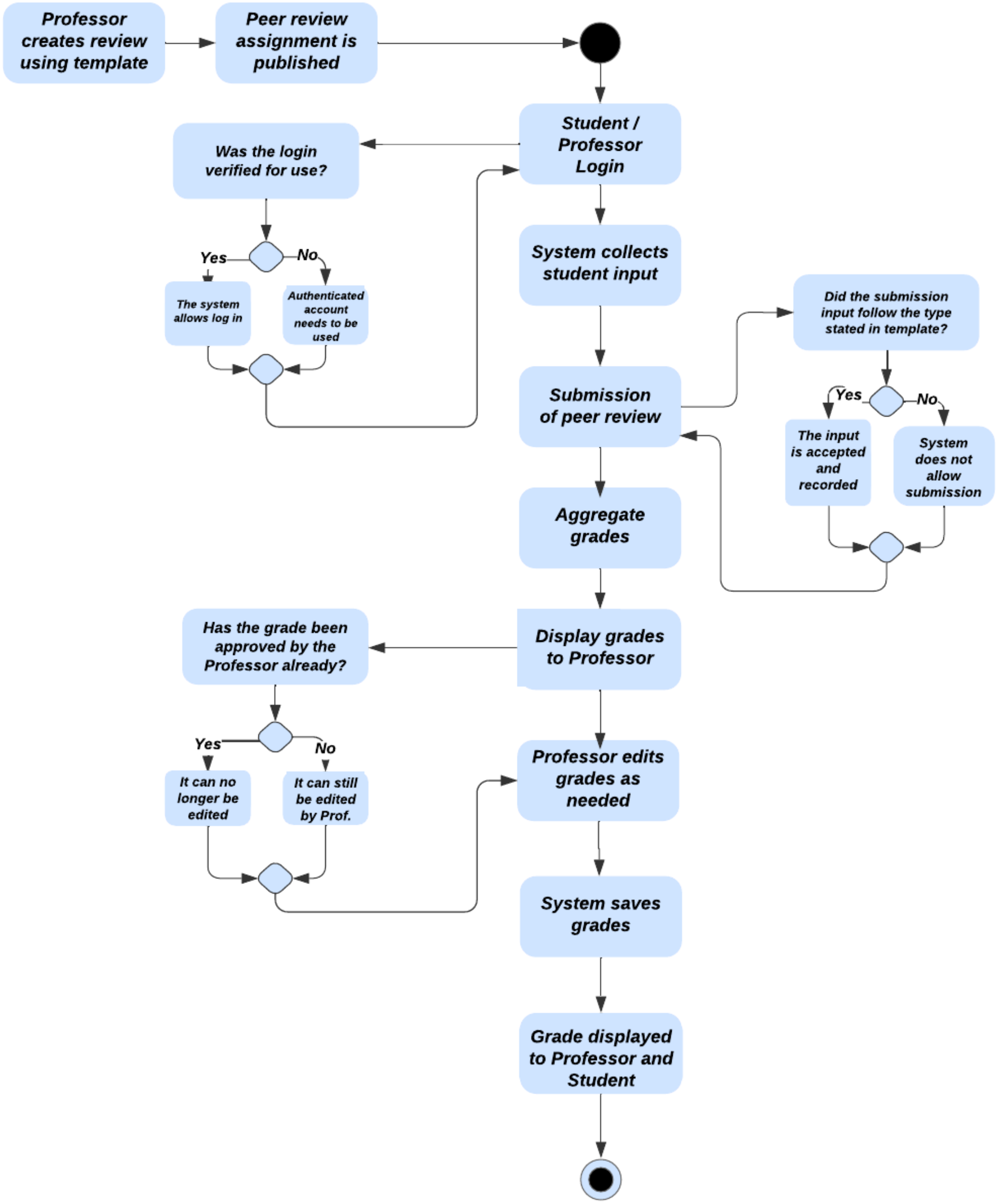
This diagram illustrates the sequence of events of creating an Instructor account. This diagram assumes that the Instructor starts at the login screen of the website. First, the Instructor chooses to create an account. The website then shows the create account screen. The Instructor then fills out the account creation information and submits it. The website sends the information to the API, and the API validates the information is in the correct format. If it is not, the API sends an “Invalid Inputs” message to the website. The website then displays an invalid information message and requests the Instructor to input valid information. If valid information is submitted, the API stores a new Instructor account in the database with the information submitted by the Instructor. The website displays the account created, and the Instructor is logged into the new account. Lastly, the Instructor Dashboard screen is displayed.

4.1.3.5 Student Account Creation



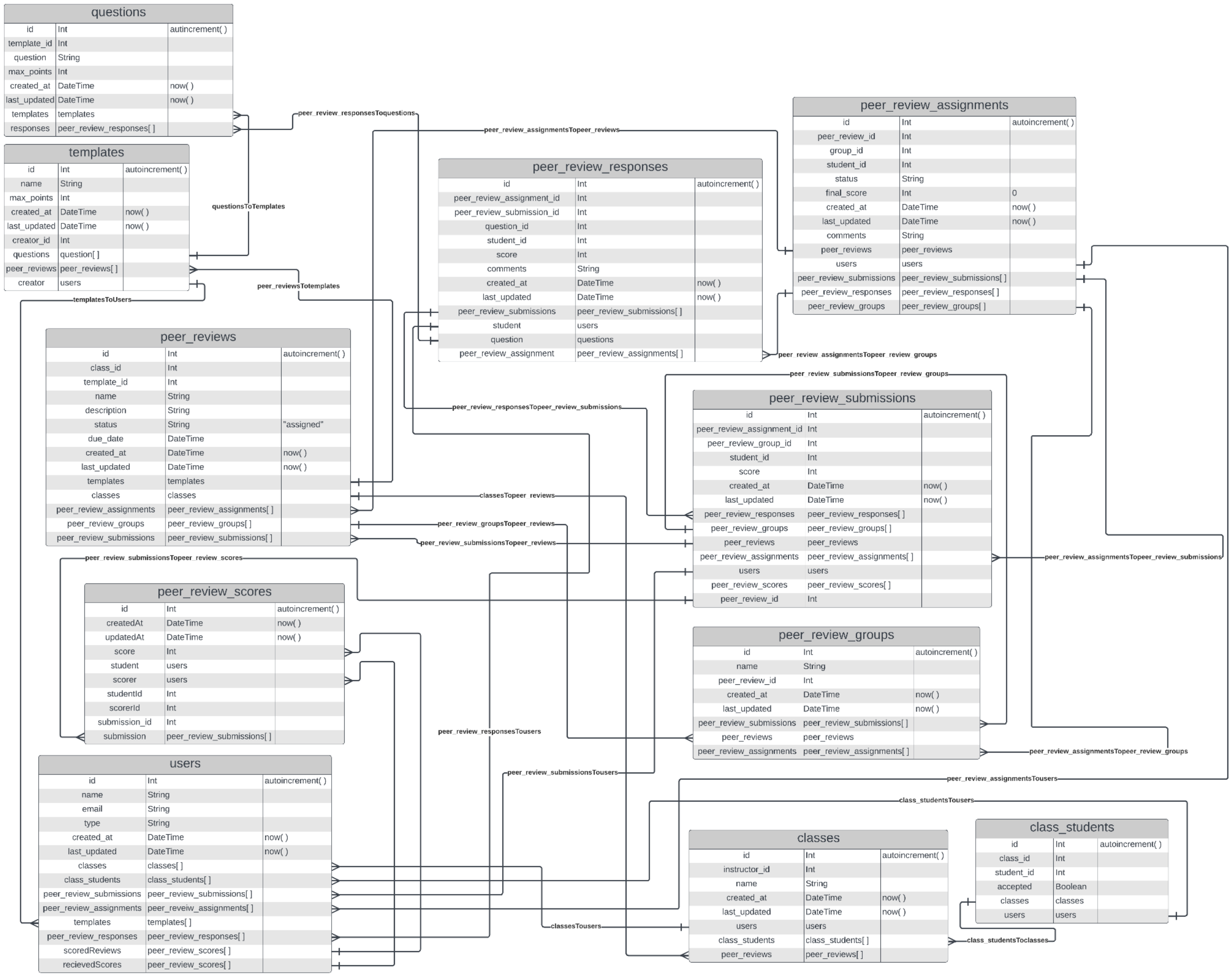
This diagram illustrates the sequence of events in creating a Student account. This diagram assumes that the Student starts at the login screen of the website. First, the Student chooses to create an account. The website then shows the create account screen. The Student then fills out the account creation information and submits it. The website sends the information to the API, and the API validates the information is in the correct format. If it is not, the API sends an “Invalid Inputs” message to the website. The website then displays an invalid information message and requests the Student to input valid information. If valid information is submitted, the API stores a new Student account in the database with the information submitted by the Student. The website displays the account created, and the Student is logged into the new account. Lastly, the Student Dashboard screen is displayed.

4.1.4 Activity UML Diagram


Professor creates review using template
Peer review assignment is published
Student / Professor Login
Was the login verified for use?
Yes
No
The system allows log in
Authenticated account needs to be used
System collects student input
Submission of peer review
Did the submission input follow the type stated in template?
Yes
No
The input is accepted and recorded
System does not allow submission
Aggregate grades
Display grades to Professor
Has the grade been approved by the Professor already?
Yes
No
It can no longer be edited
It can still be edited by Prof.
Professor edits grades as needed
System saves grades
Grade displayed to Professor and Student

This diagram illustrates the flow of the function of the web application system.

4.2 DATABASE ER DIAGRAM



questions

id	Int	autincrement( )
template_id	Int	
question	String	
max_points	Int	
created_at	DateTime	now( )
last_updated	DateTime	now( )
templates	templates	
responses	peer_review_responses[ ]	

templates

id	Int	autoincrement( )
name	String	
max_points	Int	
created_at	DateTime	now( )
last_updated	DateTime	now( )
creator_id	Int	
questions	question[ ]	
peer_reviews	peer_reviews[ ]	
creator	users	

peer_reviews

id	Int	autoincrement( )
class_id	Int	
template_id	Int	
name	String	
description	String	
status	String	"assigned"
due_date	DateTime	
created_at	DateTime	now( )
last_updated	DateTime	now( )
templates	templates	
classes	classes	
peer_review_assignments	peer_review_assignments[ ]	
peer_review_groups	peer_review_groups[ ]	
peer_review_submissions	peer_review_submissions[ ]	

peer_review_scores

id	Int	autoincrement( )
createdAt	DateTime	now( )
updatedAt	DateTime	now( )
score	Int	
student	users	
scorer	users	
studentId	Int	
scorerId	Int	
submission_id	Int	
submission	peer_review_submissions[ ]	

users

id	Int	autoincrement( )
name	String	
email	String	
type	String	
created_at	DateTime	now( )
last_updated	DateTime	now( )
classes	classes[ ]	
class_students	class_students[ ]	
peer_review_submissions	peer_review_submissions[ ]	
peer_review_assignments	peer_reveiw_assignments[ ]	
templates	templates[ ]	
peer_review_responses	peer_review_responses[ ]	
scoredReviews	peer_review_scores[ ]	
recievedScores	peer_review_scores[ ]	

peer_review_responses

id	Int	autoincrement( )
peer_review_assignment_id	Int	
peer_review_submission_id	Int	
question_id	Int	
student_id	Int	
score	Int	
comments	String	
created_at	DateTime	now( )
last_updated	DateTime	now( )
peer_review_submissions	peer_review_submissions[ ]	
student	users	
question	questions	
peer_review_assignment	peer_review_assignments[ ]	

peer_review_assignments

id	Int	autoincrement( )
peer_review_id	Int	
group_id	Int	
student_id	Int	
status	String	
final_score	Int	0
created_at	DateTime	now( )
last_updated	DateTime	now( )
comments	String	
peer_reviews	peer_reviews	
users	users	
peer_review_submissions	peer_review_submissions[ ]	
peer_review_responses	peer_review_responses[ ]	
peer_review_groups	peer_review_groups[ ]	

peer_review_submissions

id	Int	autoincrement( )
peer_review_assignment_id	Int	
peer_review_group_id	Int	
student_id	Int	
score	Int	
created_at	DateTime	now( )
last_updated	DateTime	now( )
peer_review_responses	peer_review_responses[ ]	
peer_review_groups	peer_review_groups[ ]	
peer_reviews	peer_reviews	
peer_review_assignments	peer_review_assignments[ ]	
users	users	
peer_review_scores	peer_review_scores[ ]	
peer_review_id	Int	

peer_review_groups

id	Int	autoincrement( )
name	String	
peer_review_id	Int	
created_at	DateTime	now( )
last_updated	DateTime	now( )
peer_review_submissions	peer_review_submissions[ ]	
peer_reviews	peer_reviews	
peer_review_assignments	peer_review_assignments[ ]	

classes

id	Int	autoincrement( )
instructor_id	Int	
name	String	
created_at	DateTime	now( )
last_updated	DateTime	now( )
users	users	
class_students	class_students[ ]	
peer_reviews	peer_reviews[ ]	

class_students

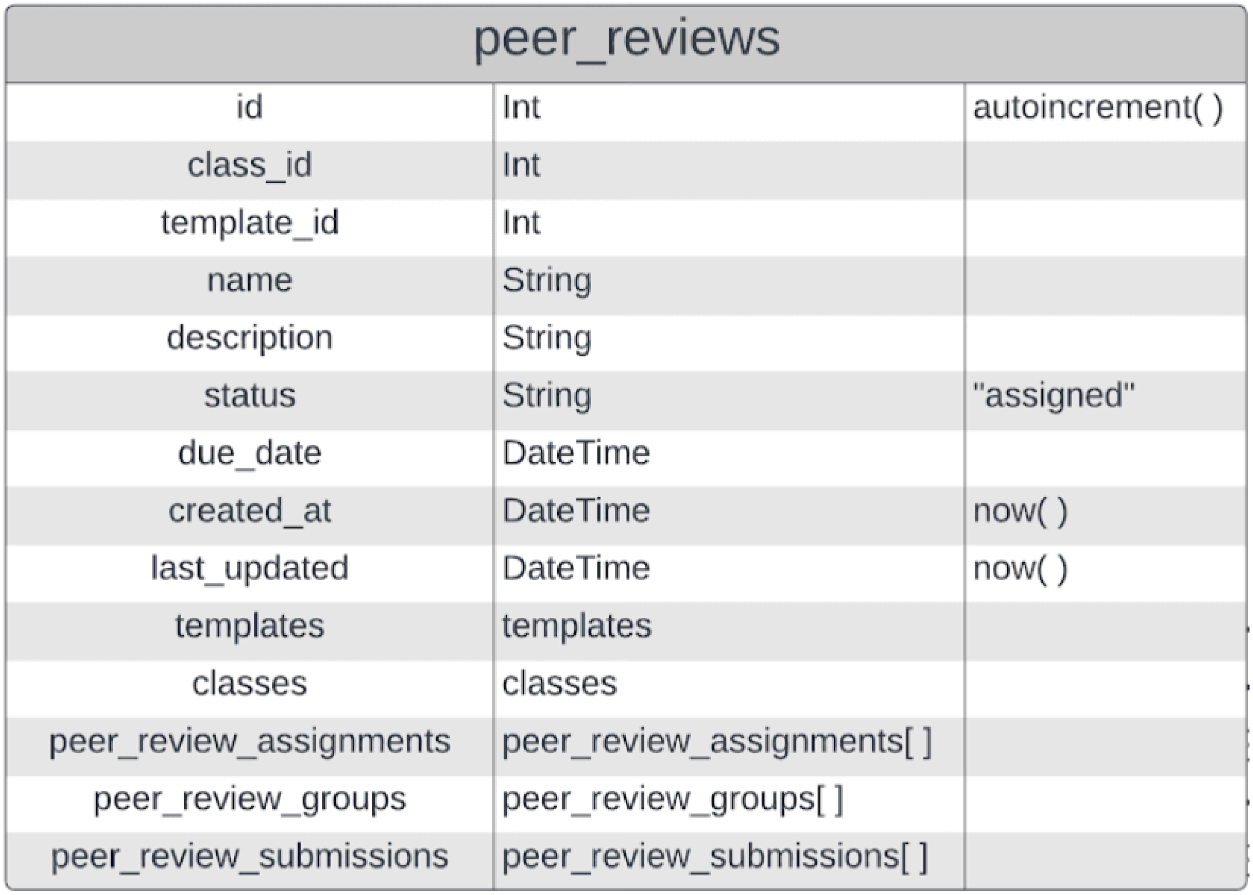
id	Int	autoincrement( )
class_id	Int	
student_id	Int	
accepted	Boolean	
classes	classes	
users	users	

peer_review_responsesToquestions
peer_review_assignmentsTopeer_reviews
questionsToTemplates
peer_reviewsTotemplates
templatesToUsers
peer_review_assignmentsTopeer_review_groups
peer_review_submissionsTopeer_review_groups
peer_review_responsesTopeer_review_submissions
classesTopeer_reviews
peer_review_groupsTopeer_reviews
peer_review_submissionsTopeer_reviews
peer_review_submissionsTopeer_review_scores
peer_review_assignmentsTopeer_review_submissions
peer_review_responsesTousers
peer_review_submissionsTousers
peer_review_assignmentsTopeer_review_groups
peer_review_assignmentsTousers
class_studentsTousers
classesTousers
class_studentsToclasses

This is an overview of the Entity Relationship Diagram for the database of the peer8 Web Application System.

Below is the zoomed-in view of each table in the database.

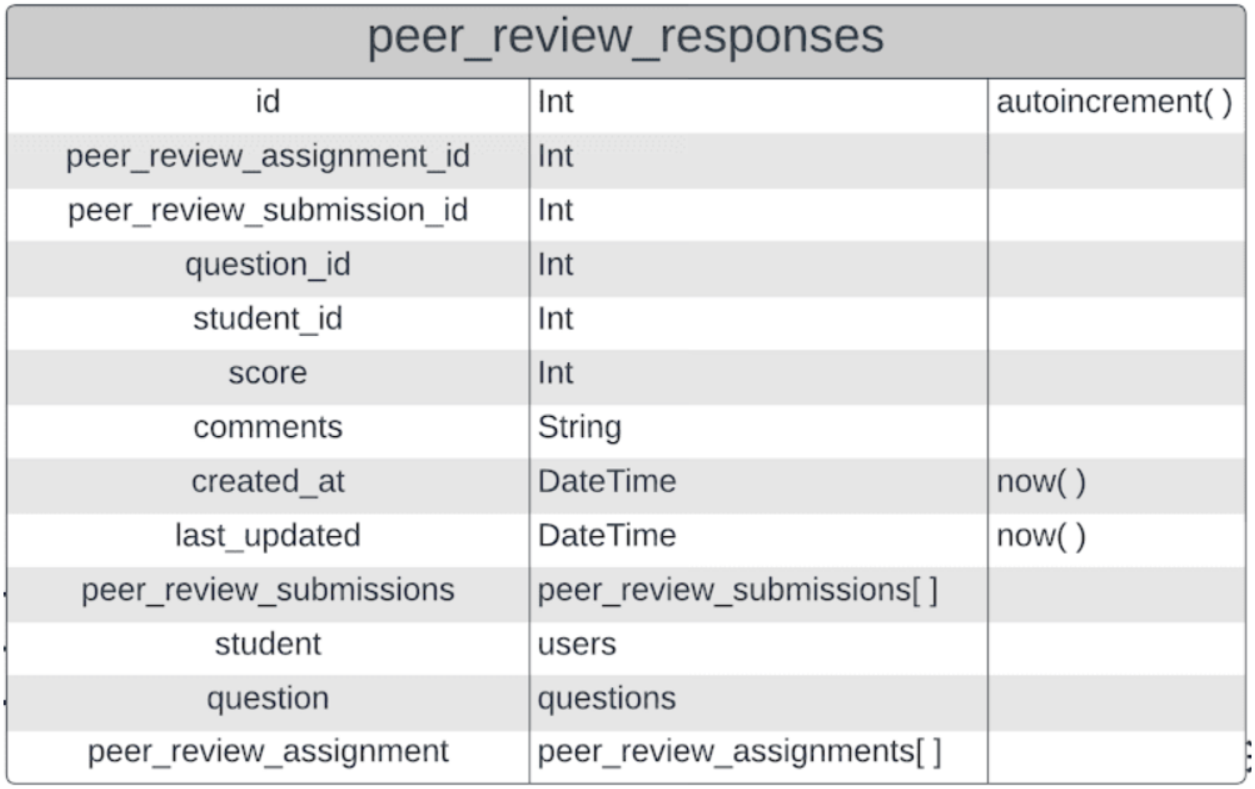
4.2.1 Peer_reviews Table



peer_reviews

id	Int	autoincrement( )
class_id	Int	
template_id	Int	
name	String	
description	String	
status	String	"assigned"
due_date	DateTime	
created_at	DateTime	now( )
last_updated	DateTime	now( )
templates	templates	
classes	classes	
peer_review_assignments	peer_review_assignments[ ]	
peer_review_groups	peer_review_groups[ ]	
peer_review_submissions	peer_review_submissions[ ]	

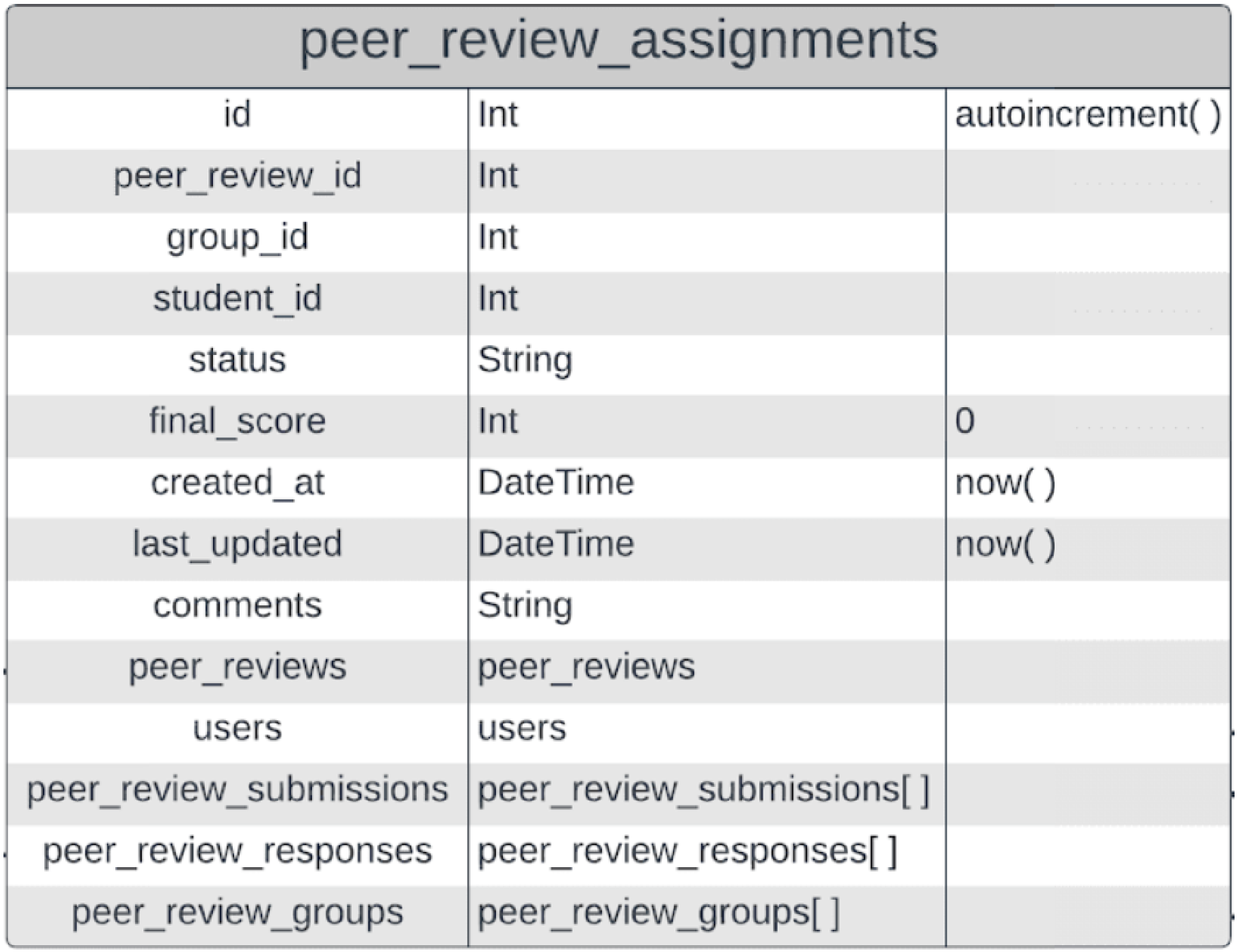
4.2.2 Peer_review_responses Table



peer_review_responses

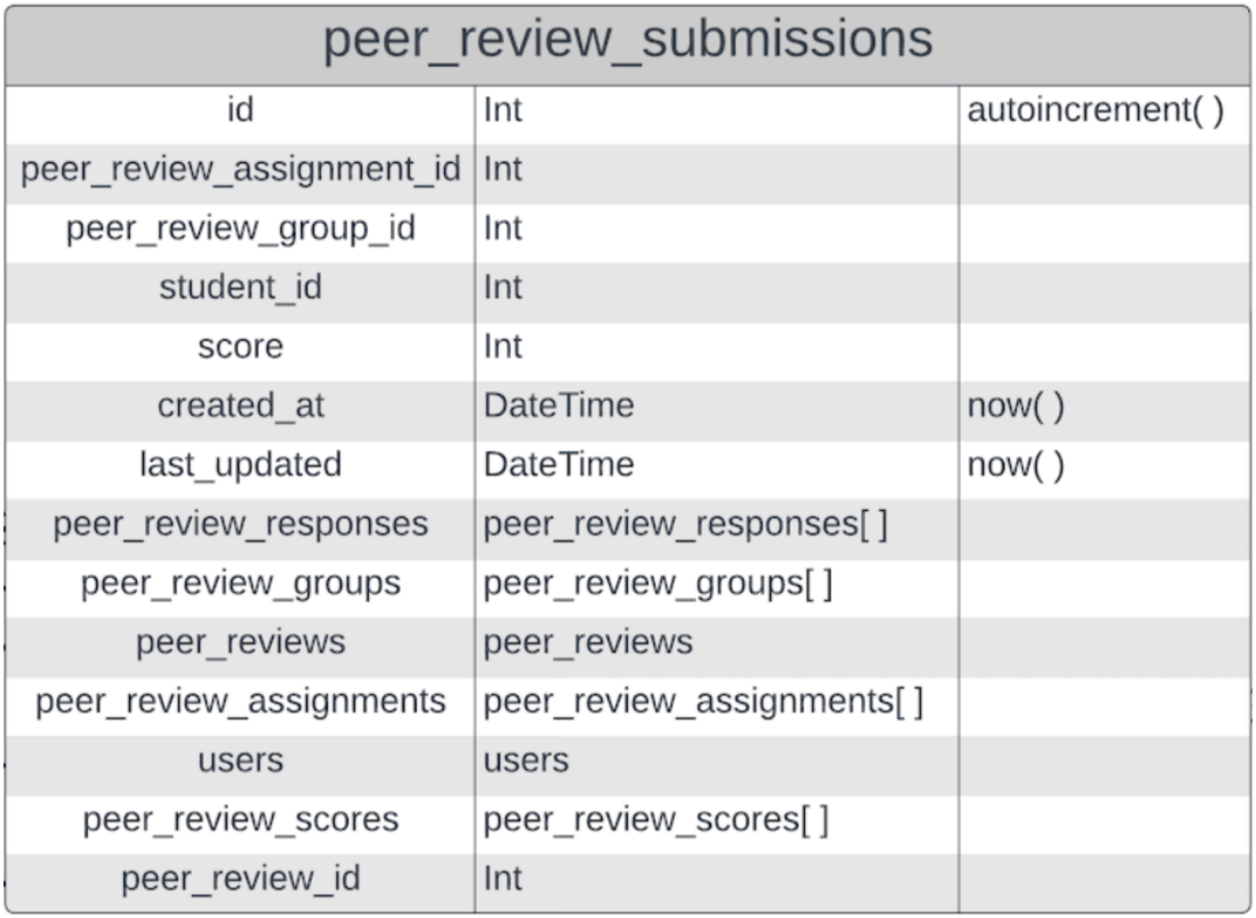
id	Int	autoincrement( )
peer_review_assignment_id	Int	
peer_review_submission_id	Int	
question_id	Int	
student_id	Int	
score	Int	
comments	String	
created_at	DateTime	now( )
last_updated	DateTime	now( )
peer_review_submissions	peer_review_submissions[ ]	
student	users	
question	questions	
peer_review_assignment	peer_review_assignments[ ]	

4.2.3 Peer_review_assignments Table



peer_review_assignments		
id	Int	autoincrement( )
peer_review_id	Int	
group_id	Int	
student_id	Int	
status	String	
final_score	Int	0
created_at	DateTime	now( )
last_updated	DateTime	now( )
comments	String	
peer_reviews	peer_reviews	
users	users	
peer_review_submissions	peer_review_submissions[ ]	
peer_review_responses	peer_review_responses[ ]	
peer_review_groups	peer_review_groups[ ]	

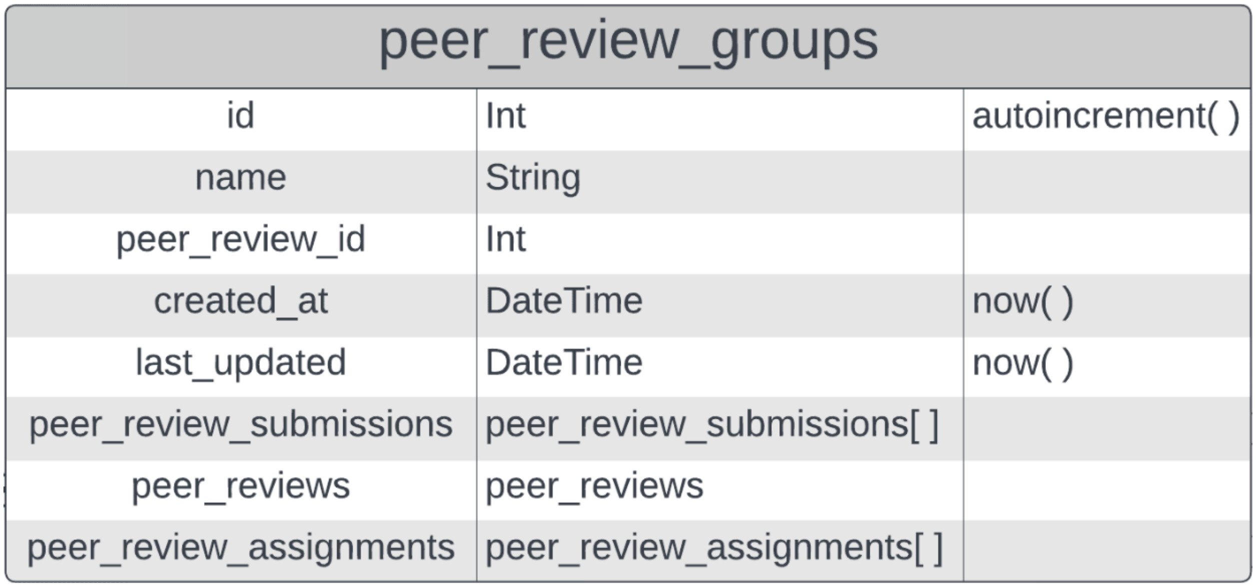
4.2.4 Peer_review_submissions Table



peer_review_submissions

id	Int	autoincrement( )
peer_review_assignment_id	Int	
peer_review_group_id	Int	
student_id	Int	
score	Int	
created_at	DateTime	now( )
last_updated	DateTime	now( )
peer_review_responses	peer_review_responses[ ]	
peer_review_groups	peer_review_groups[ ]	
peer_reviews	peer_reviews	
peer_review_assignments	peer_review_assignments[ ]	
users	users	
peer_review_scores	peer_review_scores[ ]	
peer_review_id	Int	

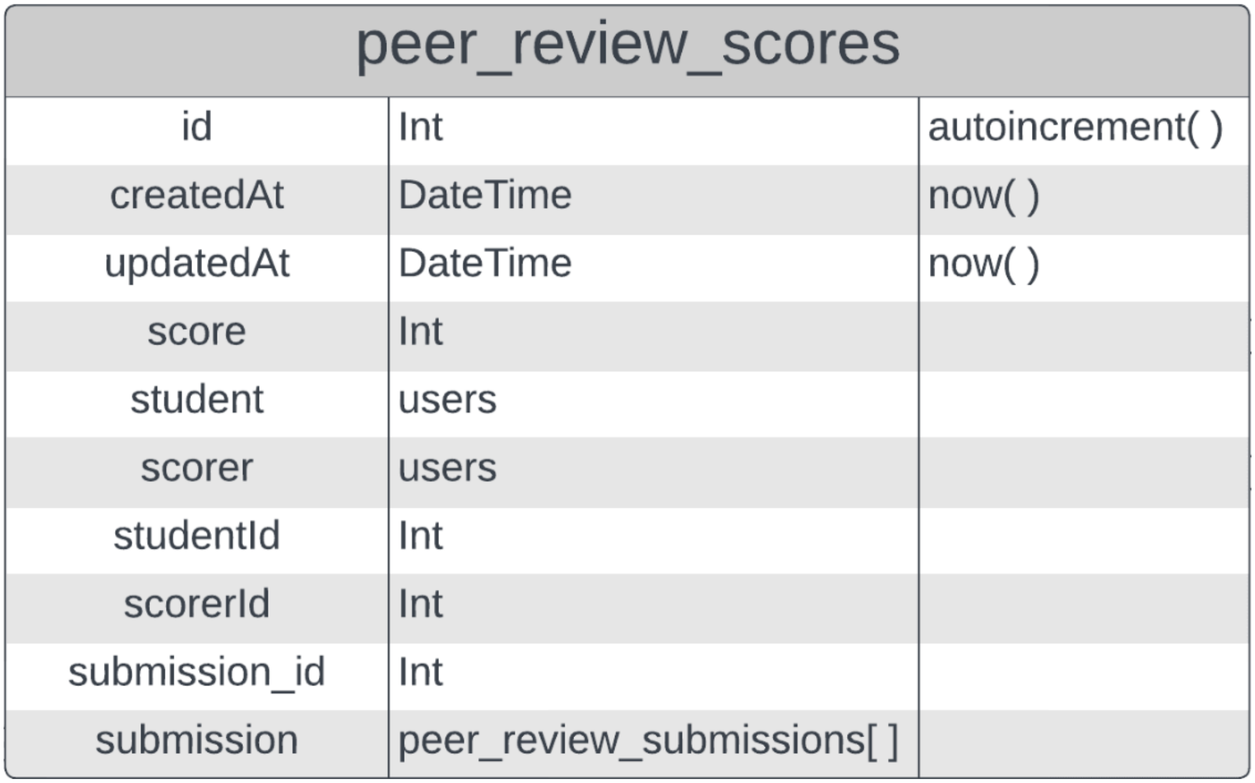
4.2.5 Peer_review_groups Table



peer_review_groups

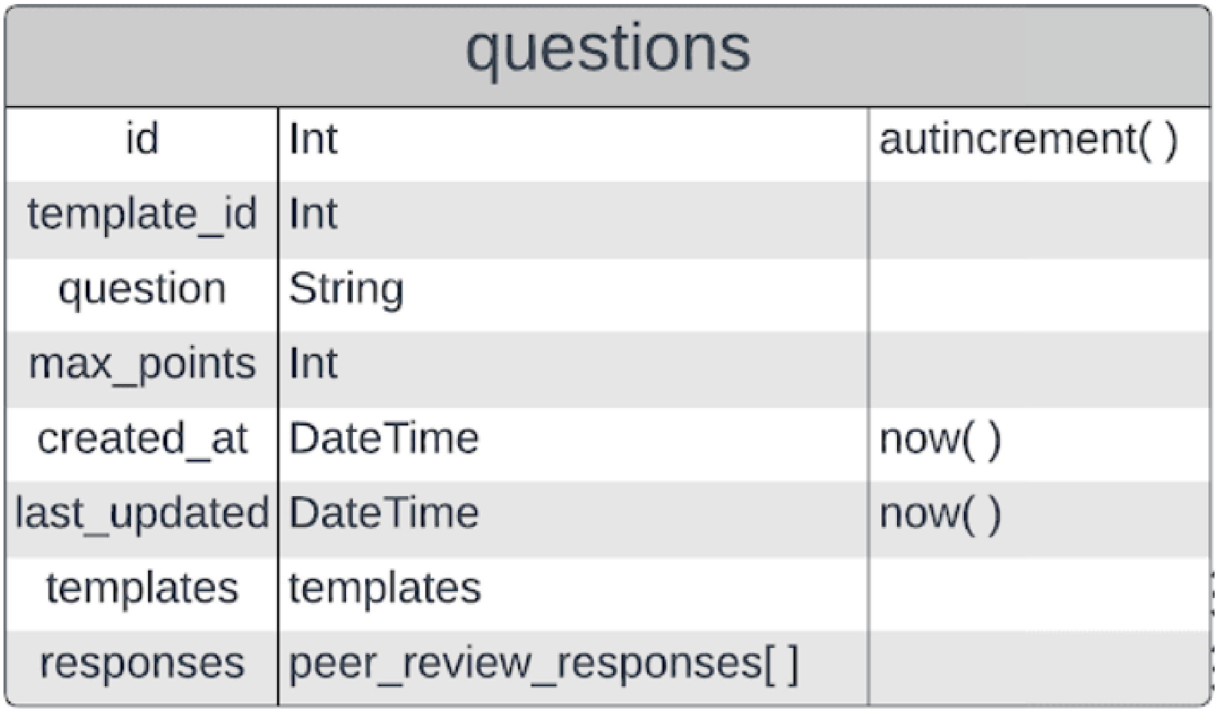
id	Int	autoincrement( )
name	String	
peer_review_id	Int	
created_at	DateTime	now( )
last_updated	DateTime	now( )
peer_review_submissions	peer_review_submissions[ ]	
peer_reviews	peer_reviews	
peer_review_assignments	peer_review_assignments[ ]	

4.2.6 Peer_review_scores Table



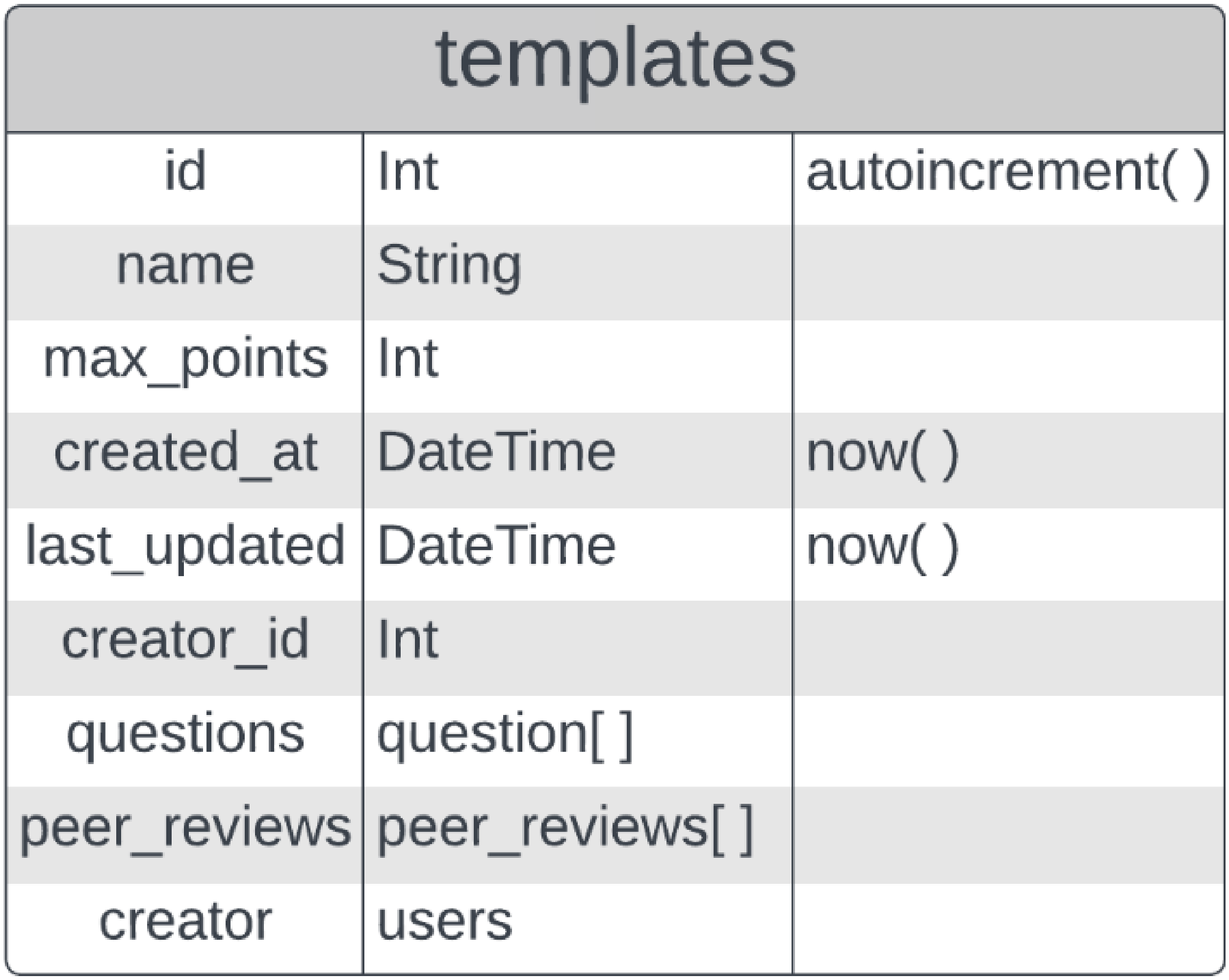
peer_review_scores		
id	Int	autoincrement( )
createdAt	DateTime	now( )
updatedAt	DateTime	now( )
score	Int	
student	users	
scorer	users	
studentId	Int	
scorerId	Int	
submission_id	Int	
submission	peer_review_submissions[ ]	

4.2.7 Questions Table



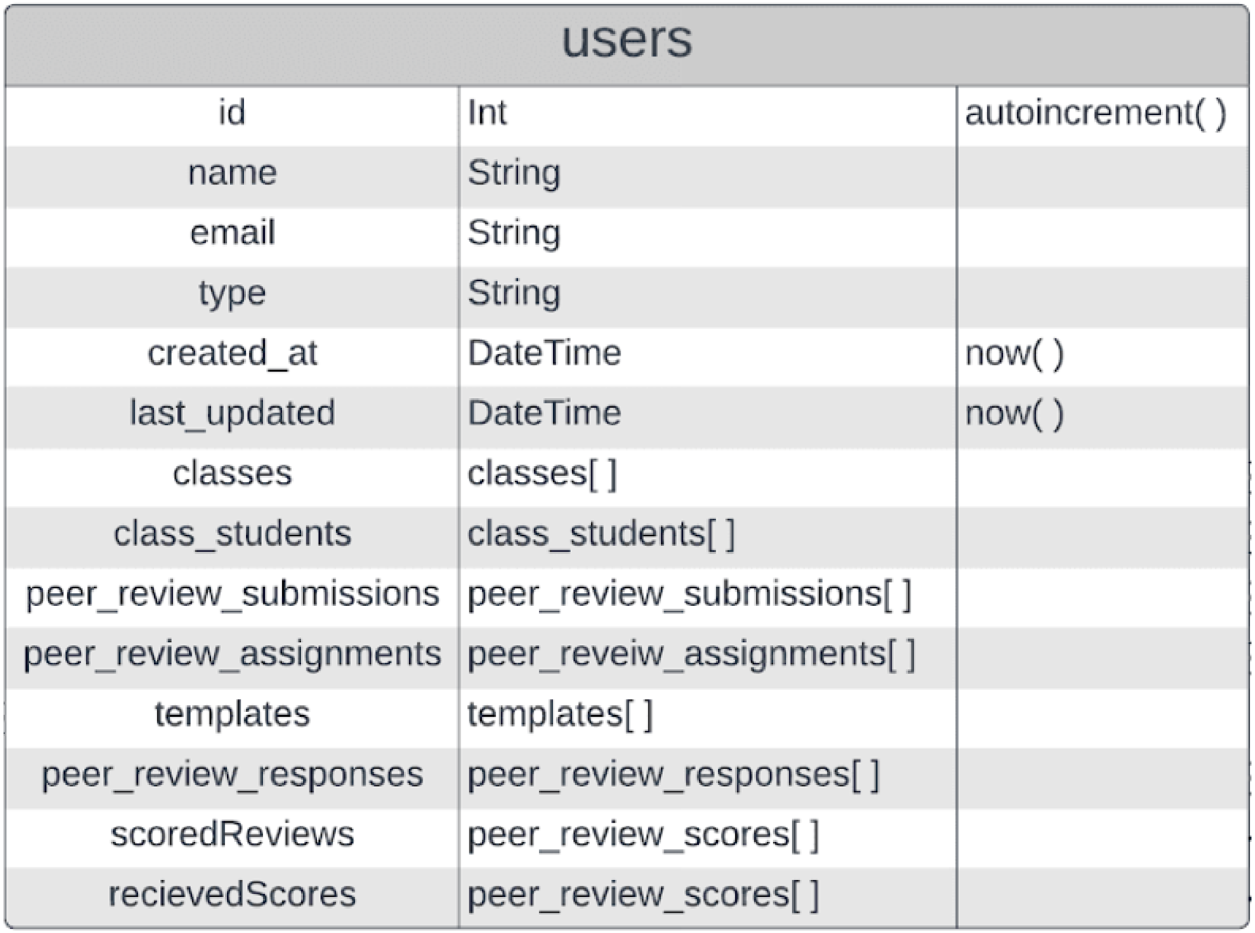
questions		
id	Int	autincrement( )
template_id	Int	
question	String	
max_points	Int	
created_at	DateTime	now( )
last_updated	DateTime	now( )
templates	templates	
responses	peer_review_responses[ ]	

4.2.8 Templates Table



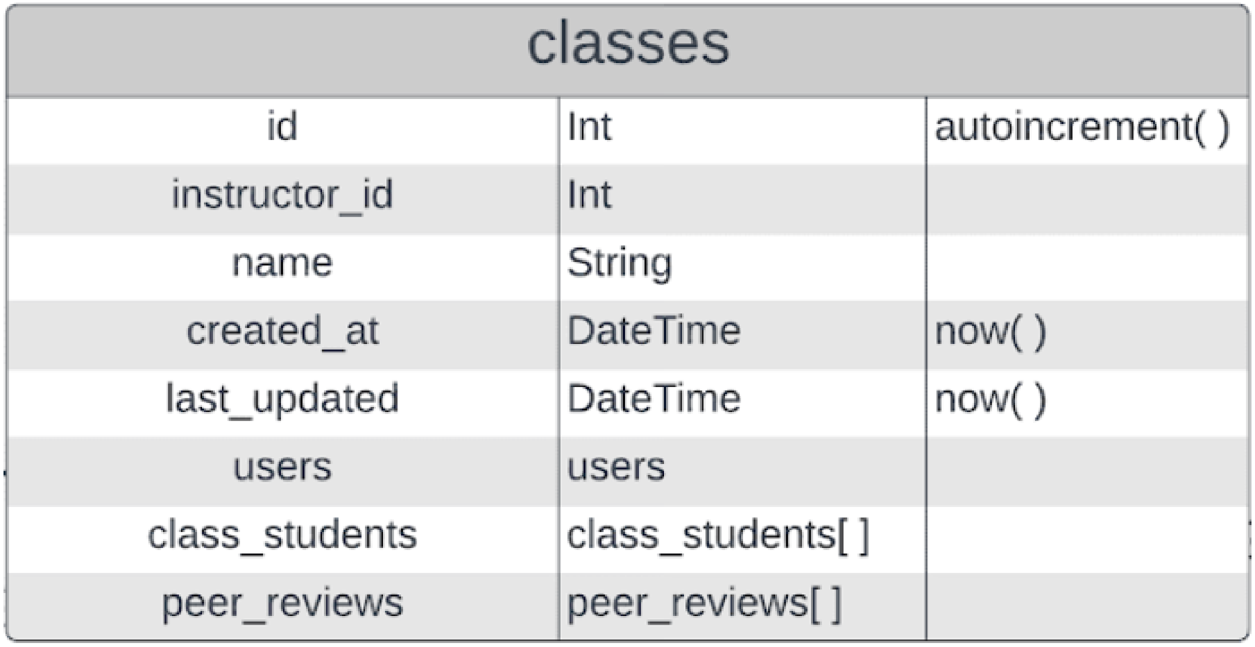
templates		
id	Int	autoincrement( )
name	String	
max_points	Int	
created_at	DateTime	now( )
last_updated	DateTime	now( )
creator_id	Int	
questions	question[ ]	
peer_reviews	peer_reviews[ ]	
creator	users	

4.2.9 Users Table



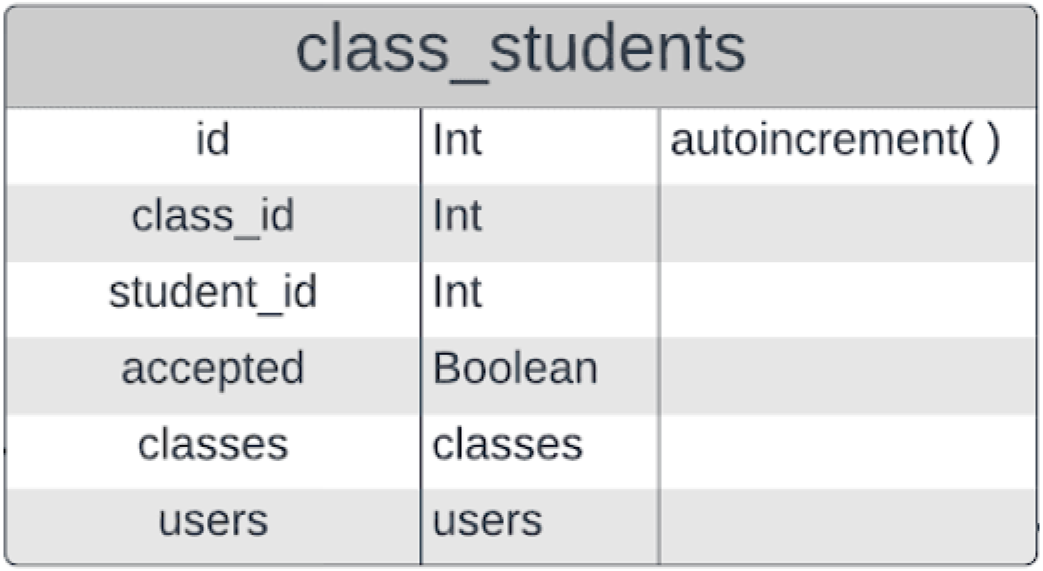
users		
id	Int	autoincrement( )
name	String	
email	String	
type	String	
created_at	DateTime	now( )
last_updated	DateTime	now( )
classes	classes[ ]	
class_students	class_students[ ]	
peer_review_submissions	peer_review_submissions[ ]	
peer_review_assignments	peer_reveiw_assignments[ ]	
templates	templates[ ]	
peer_review_responses	peer_review_responses[ ]	
scoredReviews	peer_review_scores[ ]	
recievedScores	peer_review_scores[ ]	

4.2.10 Classes Table



classes		
id	Int	autoincrement( )
instructor_id	Int	
name	String	
created_at	DateTime	now( )
last_updated	DateTime	now( )
users	users	
class_students	class_students[ ]	
peer_reviews	peer_reviews[ ]	

4.2.11 Class_students Table



class_students		
id	Int	autoincrement( )
class_id	Int	
student_id	Int	
accepted	Boolean	
classes	classes	
users	users	

SECTION 5 - IMPLEMENTATION STRATEGY

This section covers development methodology, development tools and technologies, risks and mitigation strategies, and deployment and maintenance plans.

5.1 DEVELOPMENT METHODOLOGY

The Scrum Agile development methodology was used to implement the peer8 Web Application System.
The roles and responsibilities of the development team were as follows:

Scrum Master/Developer: Cole Cagle
Application Architect/Developer: Francisco Ayala
Security Manager/Developer: Jannat Saeed
Development Manager/Developer: Noah Taylor
Database Administrator/Developer: Yhara Mendoza
Quality Assurance Manager/Developer: Cuinn Owens

During the development of this project, the team met one to three times per week for Stand-up and Sprint-planning.

The chosen development methodology benefitted the development team by keeping constant contact with the stakeholder at the forefront of the process. Additionally, it allowed the team to take on a more flexible approach to the development process.

5.2 DEVELOPMENT TOOLS AND TECHNOLOGIES

- The peer8 Web Application System was developed with the combination of the free and open-source front-end framework, Svelte, and the programming languages JavaScript and TypeScript.
- Discord was used as the main source of communication when meeting online for Stand-up and Sprint-planning meetings.
- GroupMe was also used as a means of communication for the development team, but only as a source of quick, text-based communication.
- GitHub was used as the version control.
- Google Docs was used to create the Project Document, Security Plan, Meeting Agenda and Minutes, and Stakeholder Meeting notes.
- Google Sheets was used in creating the Communication Plan, Gantt Chart, Product Backlog, and Test Plan.
- Google Slides was used in creating the power points used with oral stakeholder status reports and the End of Sprint 1 Presentation.
- LucidChart was used in creating the UML diagrams and the database ER diagram.
- Prisma Studio was used in designing and visualizing the database.
- Regexr.com was used in creating the regular expressions for user input validations.
- Figma was used in creating mockups for the original design of the user interface system.

5.3 RISKS AND MITIGATION STRATEGIES

Potential high-level risks that could delay, discontinue, or change the scope of the project are as follows:

- Possibility of team members being pulled to other projects or leaving the company.
- Possibility of not meeting deadlines.
- Academic privacy laws including but not limited to FERPA laws being changed in any way.
- Stakeholder expectation misalignment.
- Changing requirements.
- Budget and resource constraints.

Mitigation plans for these high-level risks are as follows:

- Lost development team member:
  - Cross training between developers, encouraging knowledge sharing, encouraging accurate documentation, and fostering a positive and supportive work environment and team culture.
- Missing Deadlines:
  - Development of a realistic and achievable project plan.
  - Identify dependencies between project activities and plan for contingencies and potential delays.
  - Regularly monitor and review progress made by the development team.
- Changing Academic Privacy Laws:
  - Monitor any potential new laws and regulations that could be passed during or shortly after the development of the system. Keep close communication with the stakeholder and plan accordingly with them regarding the new laws or regulations.
- Stakeholder Expectation Misalignment:
  - Conduct regular meetings with the stakeholders to ensure alignment on project goals and requirements. During these meetings, ensure that both the development team and stakeholders are on the same page about what is expected of them.
- Changing Requirements:
  - Conduct regular meetings with stakeholders to ensure proper communication of changing requirements.
  - Use Scrum Agile methodologies to enable flexibility during the development process.
- Budget and Resource Constraints:
  - Conduct a thorough project planning and estimation process. Prioritize work based on value and risk. Consider alternative sourcing options like outsourcing or partnering with other organizations to mitigate resource constraints.

5.4 DEPLOYMENT AND MAINTENANCE PLAN

The Deployment and Maintenance Plan outlines the steps that will be taken to install and maintain the peer8 web application. The plan stipulates that the University of Tennessee at Chattanooga's Information Technology (IT) team will be responsible for deploying the program and integrating it into its own system. This involves providing the directory that the program will use to verify emails when users create new accounts.

The IT team will begin by identifying the hardware and software requirements necessary to run the peer8 application. They will then install the application on the servers and configure it according to the specifications provided by the development team. After the initial deployment, the IT team will be responsible for maintaining the application and ensuring that it is functioning properly. This includes performing regular system checks and updates, as well as monitoring the performance of the application to ensure that it is meeting the needs of the users. In addition to maintenance, the IT team will also be responsible for providing technical support to users of the application. This includes troubleshooting any issues that users may encounter while using the application, as well as providing guidance on how to use the application effectively.

To ensure that the application is always available to users, the IT team will also implement a backup and disaster recovery plan. This plan will include regular backups of the application and data, as well as procedures for restoring the application in the event of a system failure or other disaster. The IT team's responsibility is to ensure the system is up and running with all necessary updates and security features to provide a seamless user experience.

The peer8 team is responsible for providing these updates and hotfixes periodically to ensure that the application remains up-to-date and secure. The team monitors the application regularly for any issues or vulnerabilities and takes steps to address them promptly. In addition, the team is committed to keeping users informed about any updates or changes that are made to the application, and they may provide documentation or training to ensure that users are able to make the most of any new features or functionalities that are introduced.

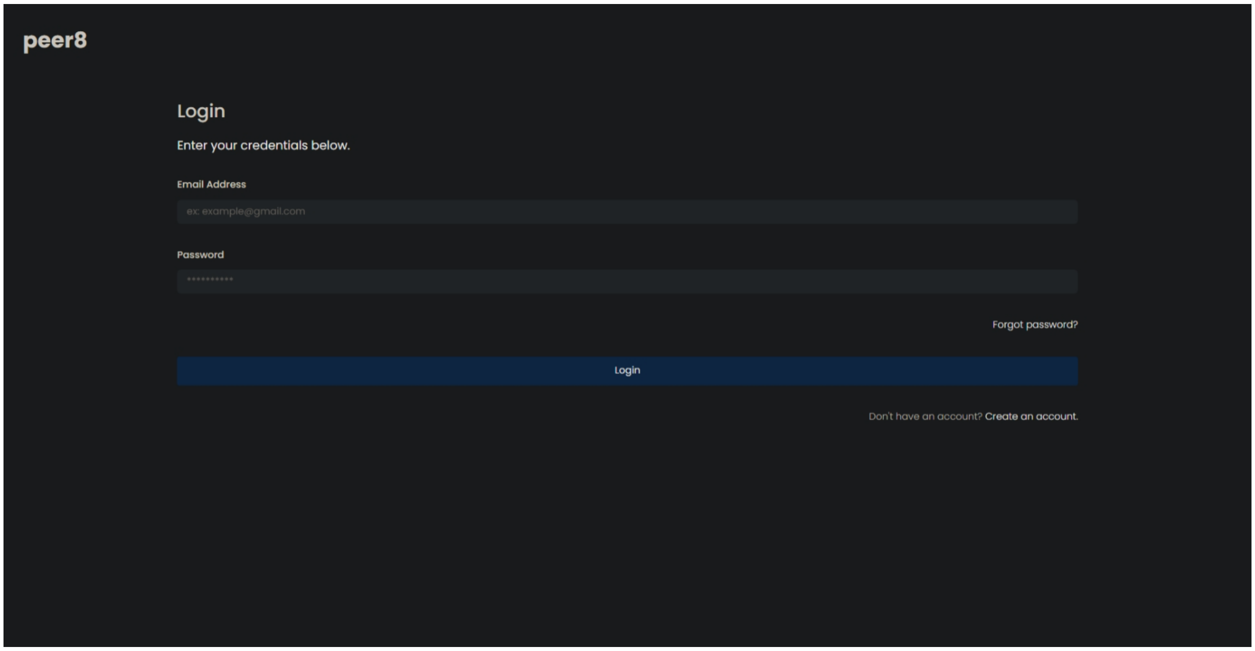
SECTION 6 - USER DOCUMENTATION

6.1 INSTALLATION GUIDE

See setup instructions and installation on GitHub:
https://github.com/fmayala/peer8-svelte/blob/master/README.md

6.2 USER'S GUIDE

Login



Login

Enter your credentials below.

Email Address

ex: example@gmail.com

Password

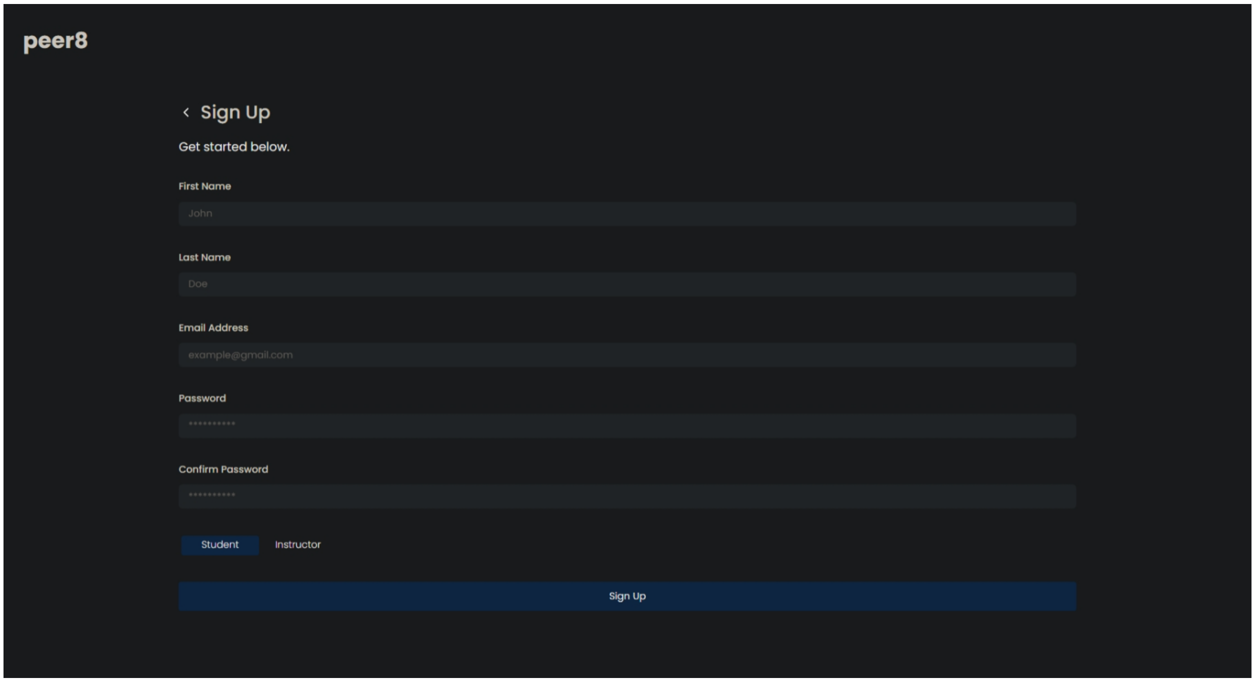
****

Don't have an account? Create an account.

Login Page

After going through the installation guide and running the web application you should see this screen (on http://localhost:5173 or whatever port the command line displays)

From here you can now **Create an Account**.



peer8

‹ Sign Up

Get started below.

First Name

John

Last Name

Doe

Email Address

example@gmail.com

Password

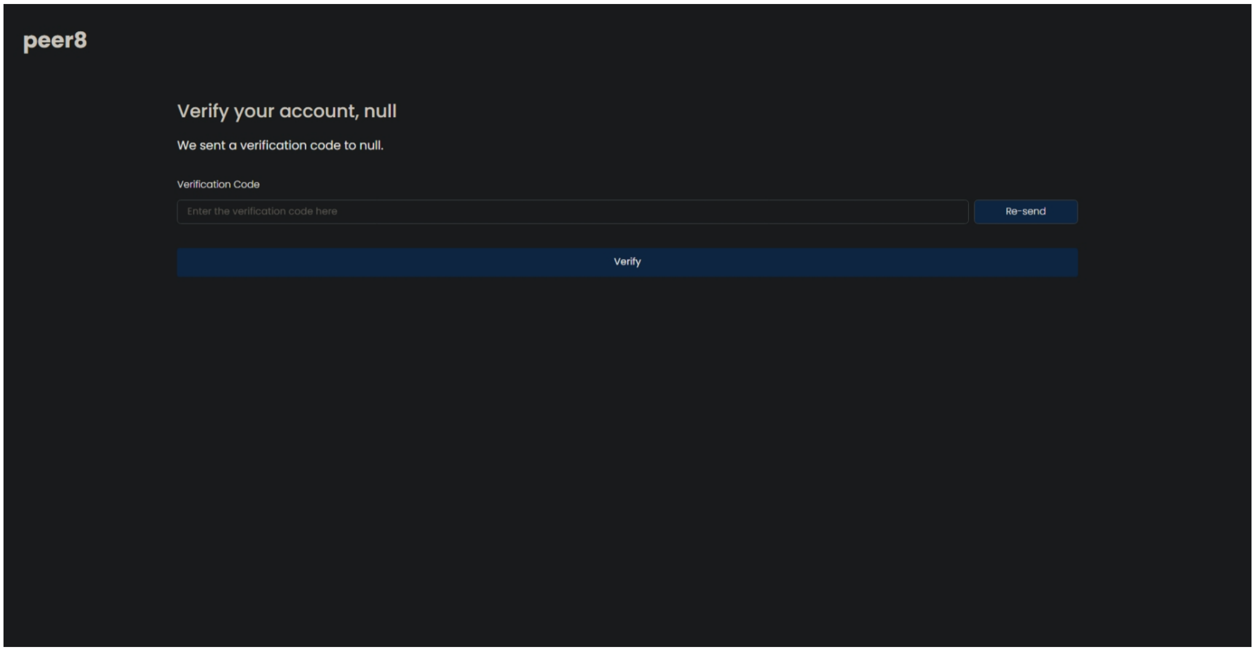
••••••••••

Confirm Password

••••••••••

If you have configured your directory.json file correctly, you should be able to sign up as an instructor using the email addresses you have allowed.

After entering valid inputs, you should be sent to a verification screen:



peer8

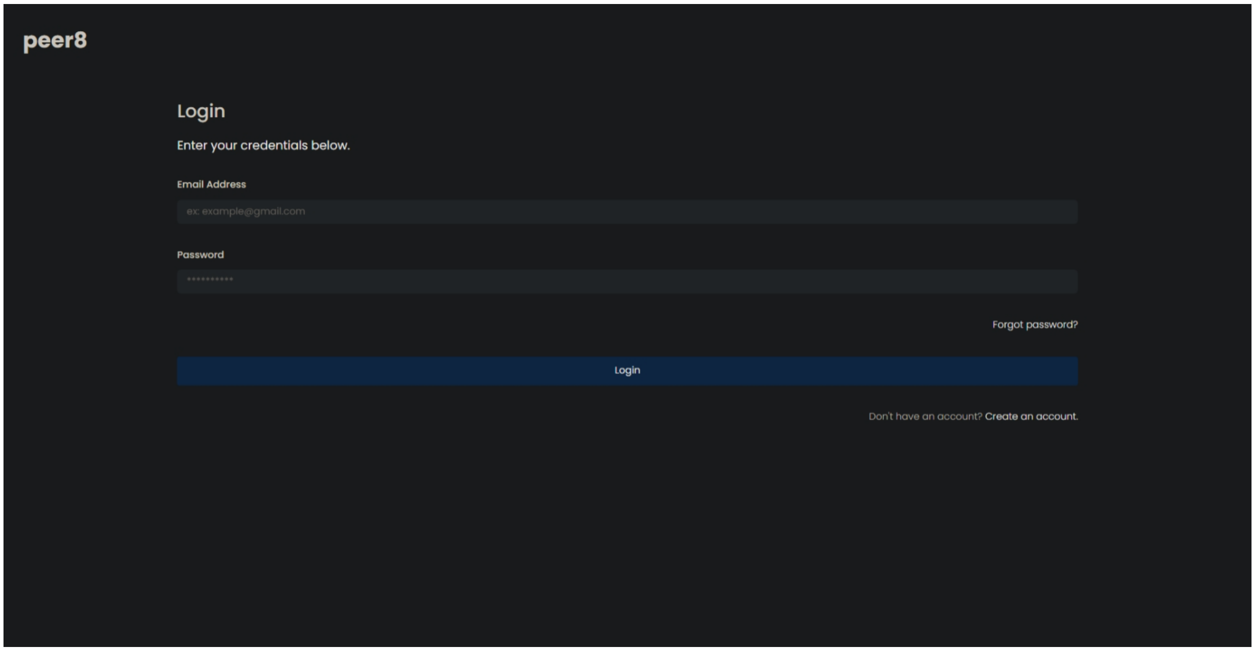
Verify your account, null

We sent a verification code to null.

Verification Code

Enter the verification code here

A code was automatically sent to your email at this point, and you can enter the code in the field to verify your account. If the code failed to send, you can try the '**Re-send**' button. After success, you will be taken back to the login screen once again.



peer8

Login

Enter your credentials below.

Email Address

ex: example@gmail.com

Password

••••••••••

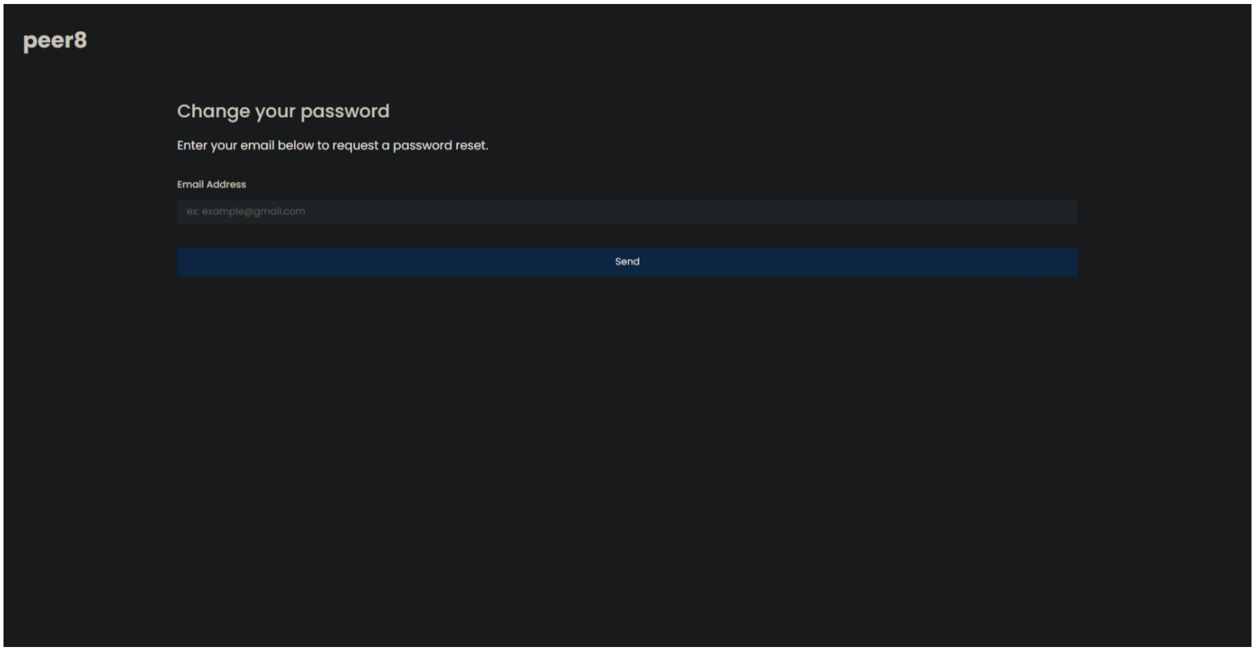
Forgot password?

Login

Don't have an account? Create an account.

You can now enter your newly created account credentials and attempt to log in.

If you realize that you can not log in because your password was input incorrectly, you can select the '**Forgot Password**' button. It will send you to a screen like below:



peer8

Change your password

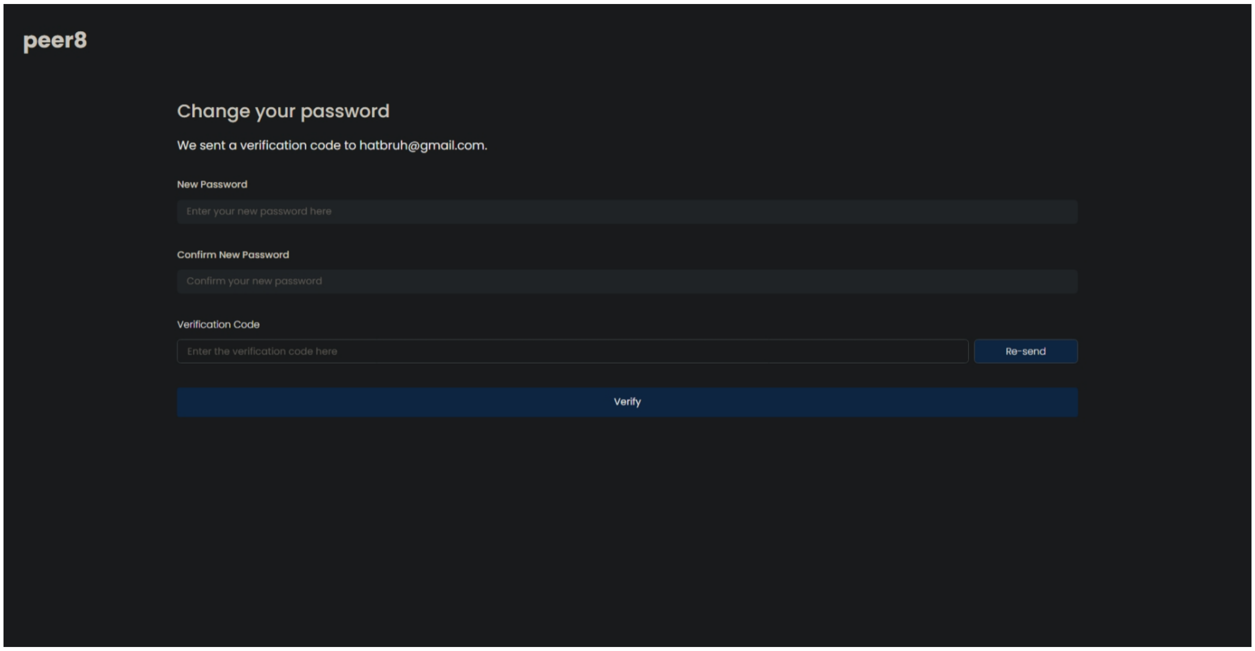
Enter your email below to request a password reset.

Email Address

ex: example@gmail.com

Send

From here you can enter the email address associated with your account and if the code was sent successfully, it will redirect you to this page:



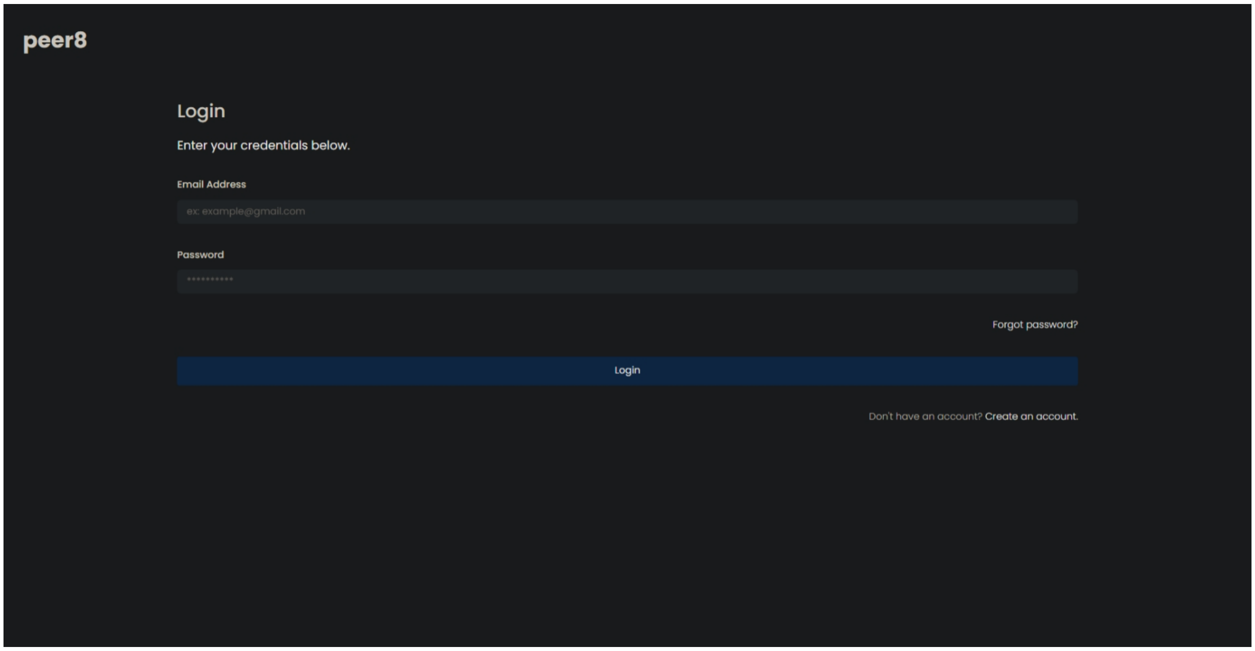
Change your password

We sent a verification code to hatbruh@gmail.com.

peer8
New Password
Enter your new password here
Confirm New Password
Confirm your new password
Verification Code
Enter the verification code here
Re-send
Verify

You can fill out the details necessary to change your password and enter the verification code from your email in order to verify and change your password.

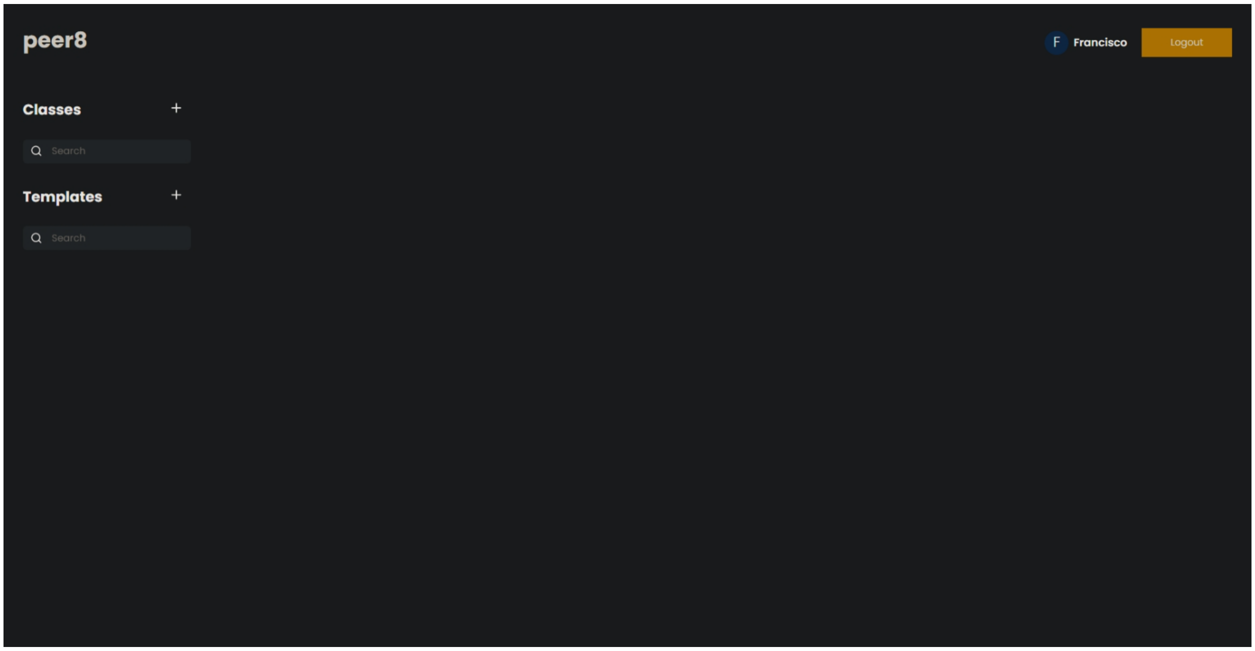
After succeeding, you will again be sent back to the login screen where you can log in.



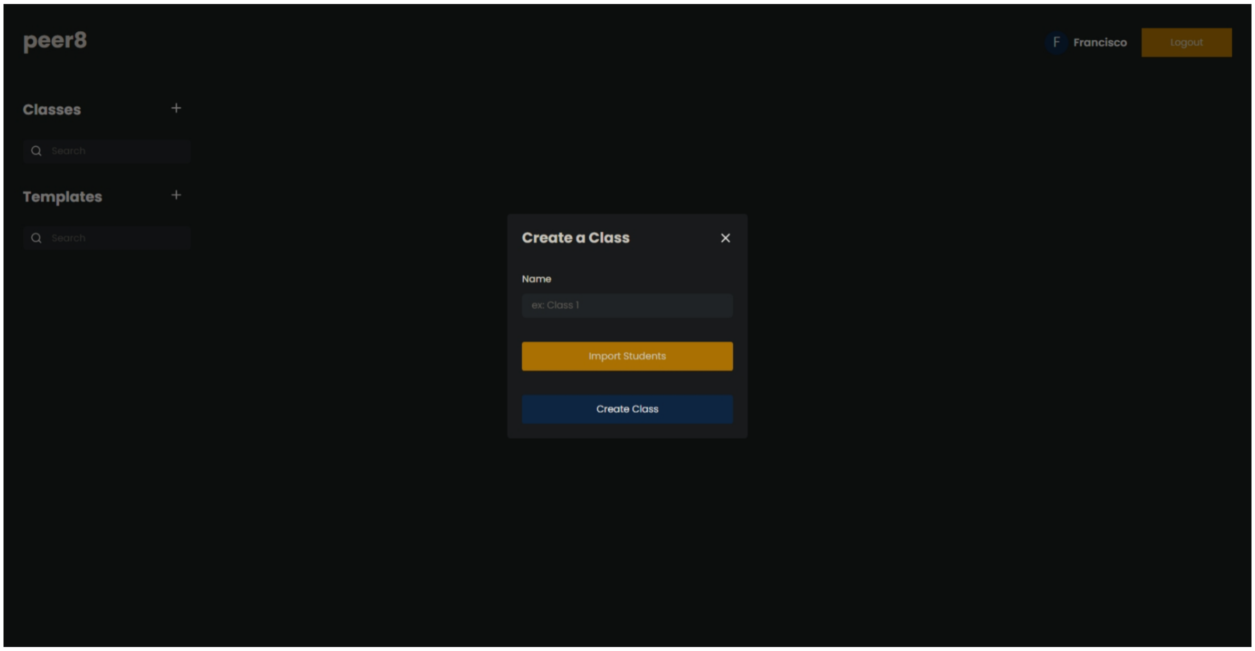
Login

Enter your credentials below.

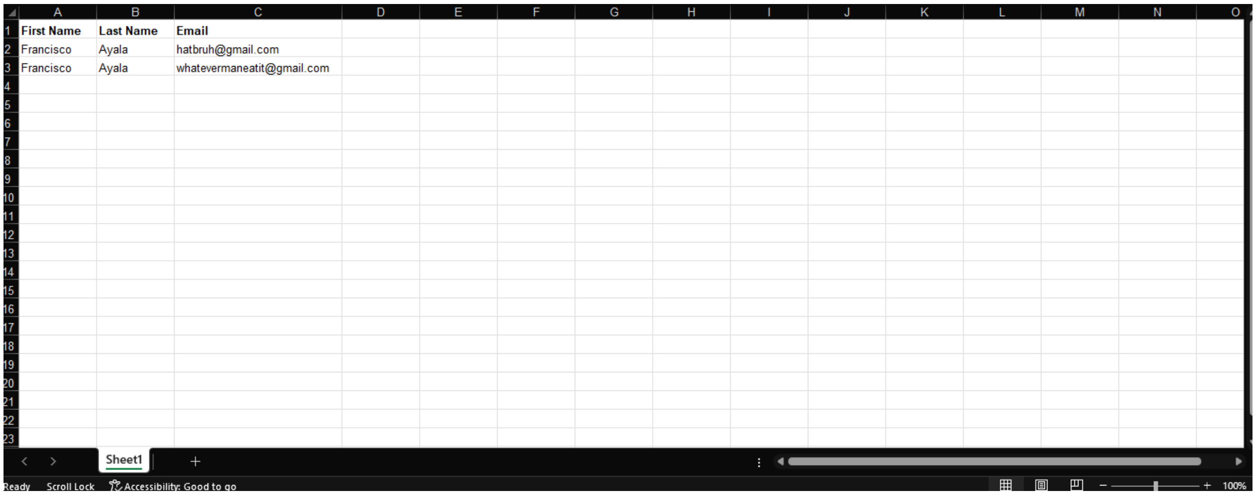
Enter your changed credentials and attempt a login, upon success, you should see a dashboard (as an instructor):



From this screen, you can now see a sidebar where you can add a class. Clicking the plus icon on the sidebar for a class shows the following panel:

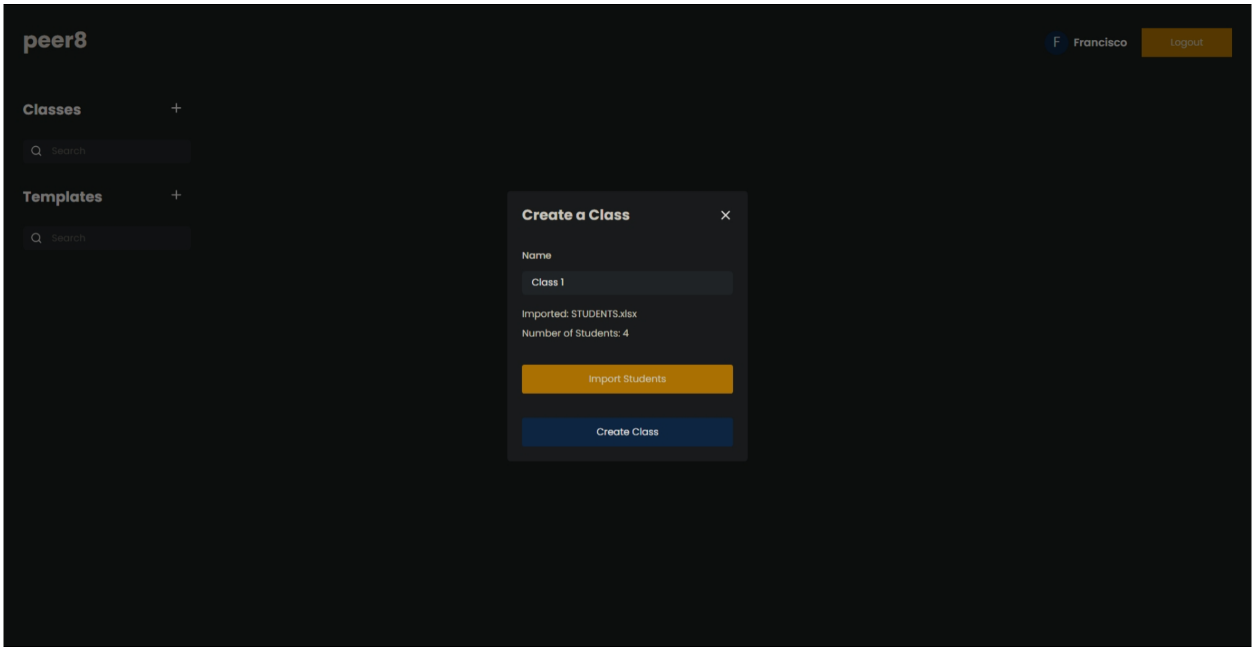


You will need to provide a class name as well as an Excel sheet with students. The format is shown below:



First Name	Last Name	Email
Francisco	Ayala	hatbruh@gmail.com
Francisco	Ayala	whatevermaneatit@gmail.com

Once you have uploaded your Excel sheet using the '**Import Student**' button you should see the panel update with the number of students that are going to be added to the class:



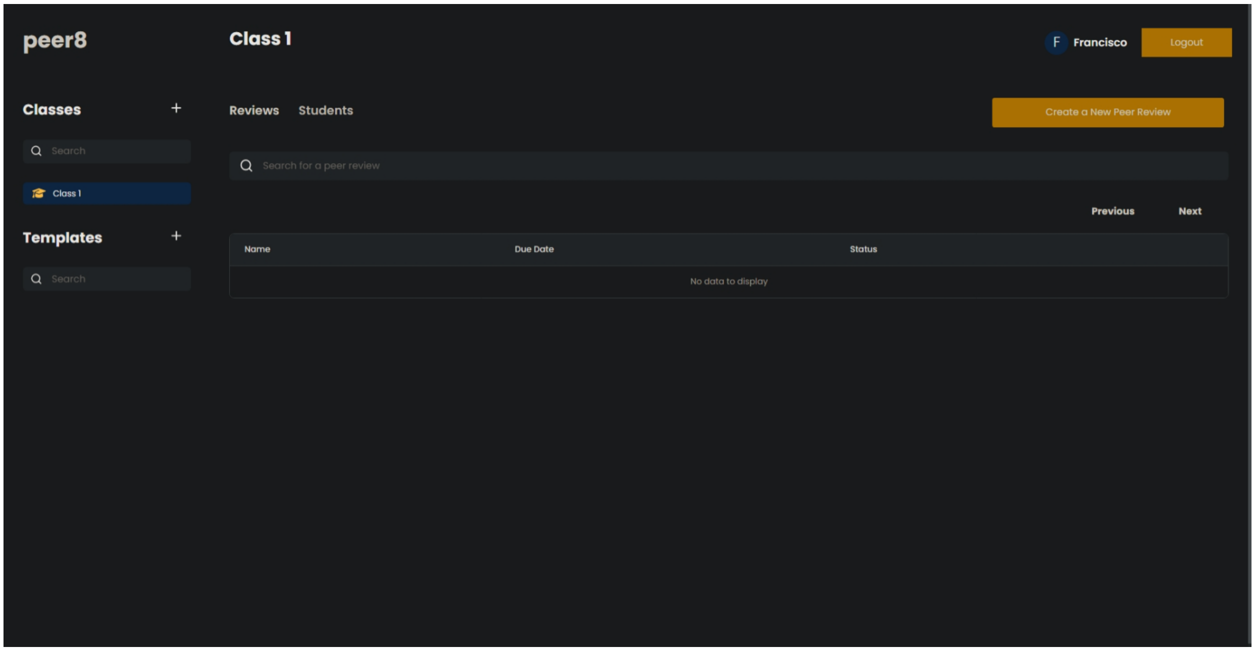
Create a Class

Name: Class 1

Imported: STUDENTS.xlsx

Number of Students: 4

Clicking ‘**Create Class**’ will create the class with the students’ emails and name details. It will create temporary student accounts for them since they have not yet signed up themselves. You may need to refresh the page after the panel has closed to allow the sidebar to populate.



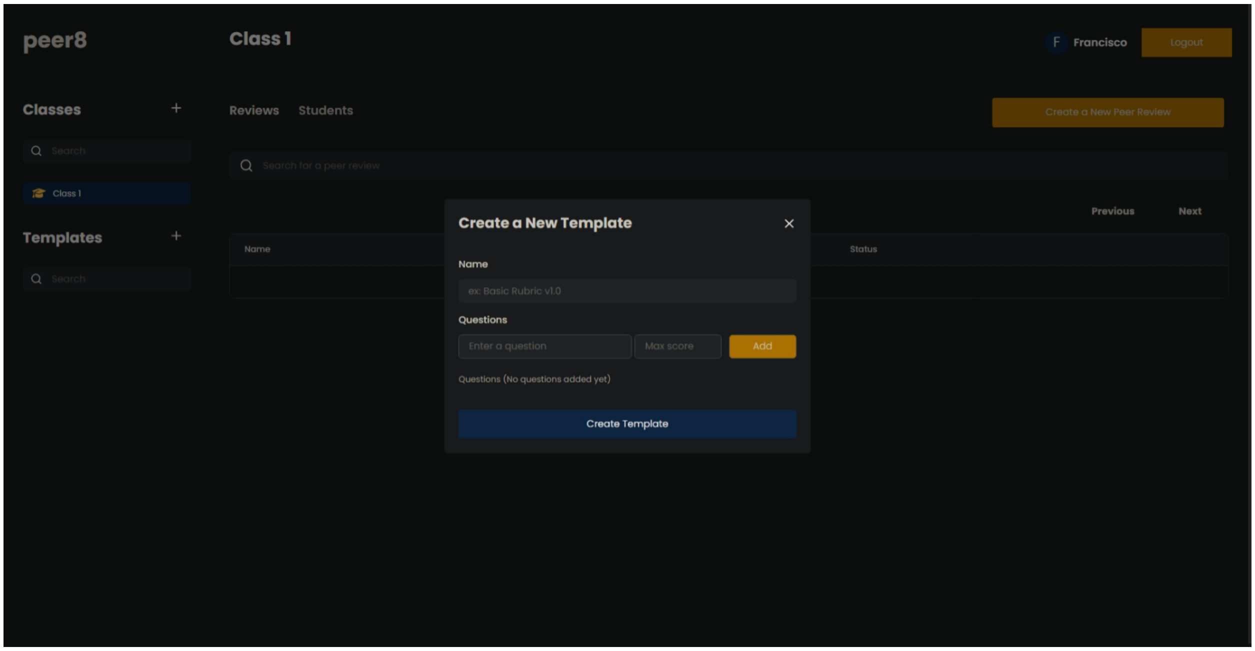
Name	Due Date	Status
No data to display		

Above is a dashboard screen of our newly created class, with the option to create a peer review. Additionally, you can see a table of peer reviews and search for a particular review if needed.

There are two tabs above and we are currently on the '**Reviews**' tab, we will get to the '**Students**' tab later.

For now, we are going to walk through the creation of a Peer Review:

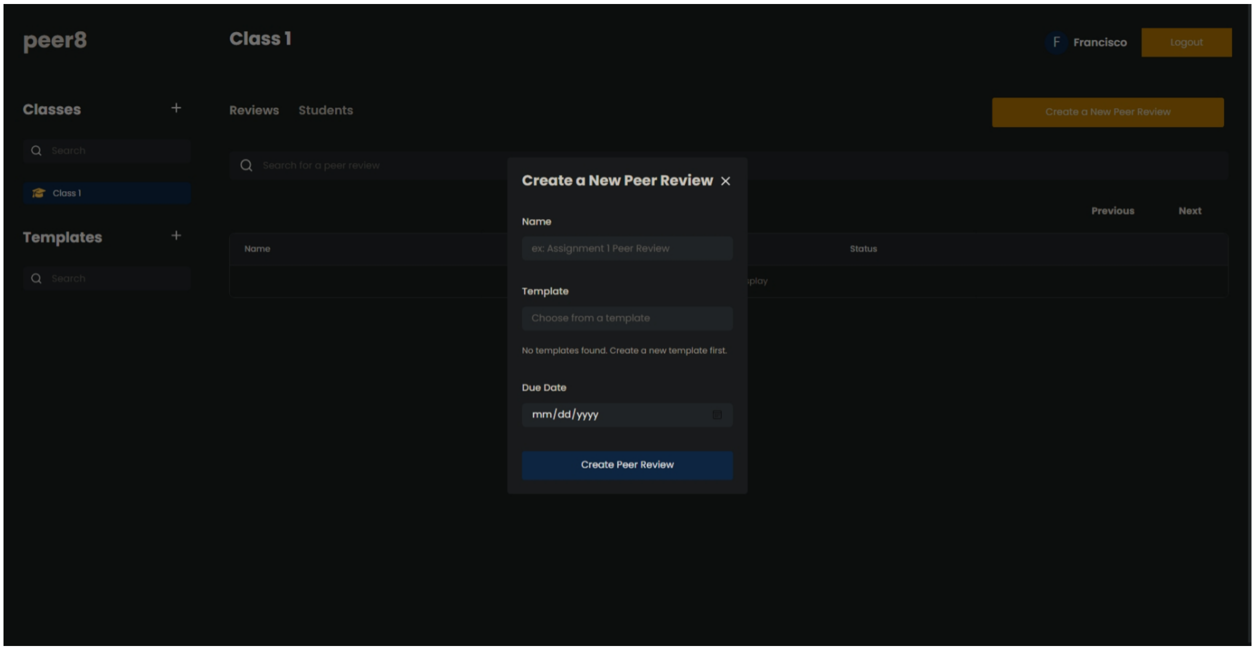
First, we will have to create a Peer Review Template (click the Plus icon on the sidebar for Templates):



You can now provide a Template name and multiple questions with varying maximum scores. Hitting the '**Add**' button for a question will add to the list of questions that will be associated with our new template.

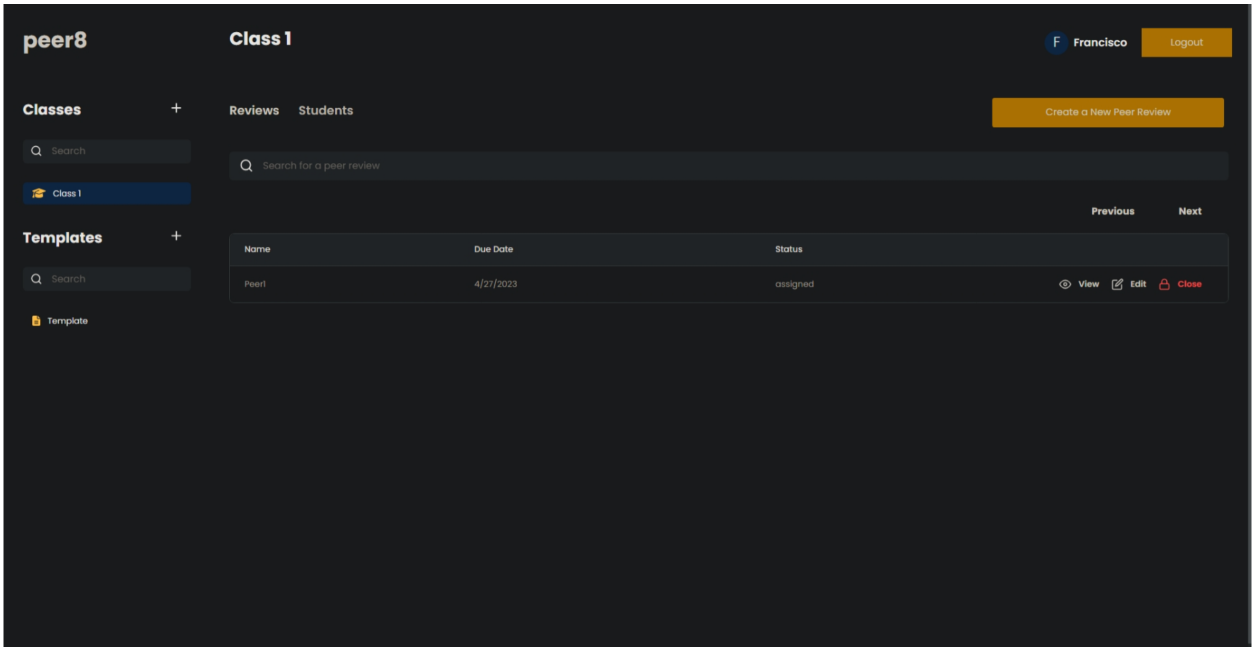
Once you are satisfied with the Template, hit '**Create Template**' and refresh the page if needed. It should now show on the sidebar.

Now we can actually create a Peer Review. Head back over to the class you have created by clicking the populated class item on the sidebar, you should once again see the class dashboard for the class. Click '**Create a New Peer Review**.'



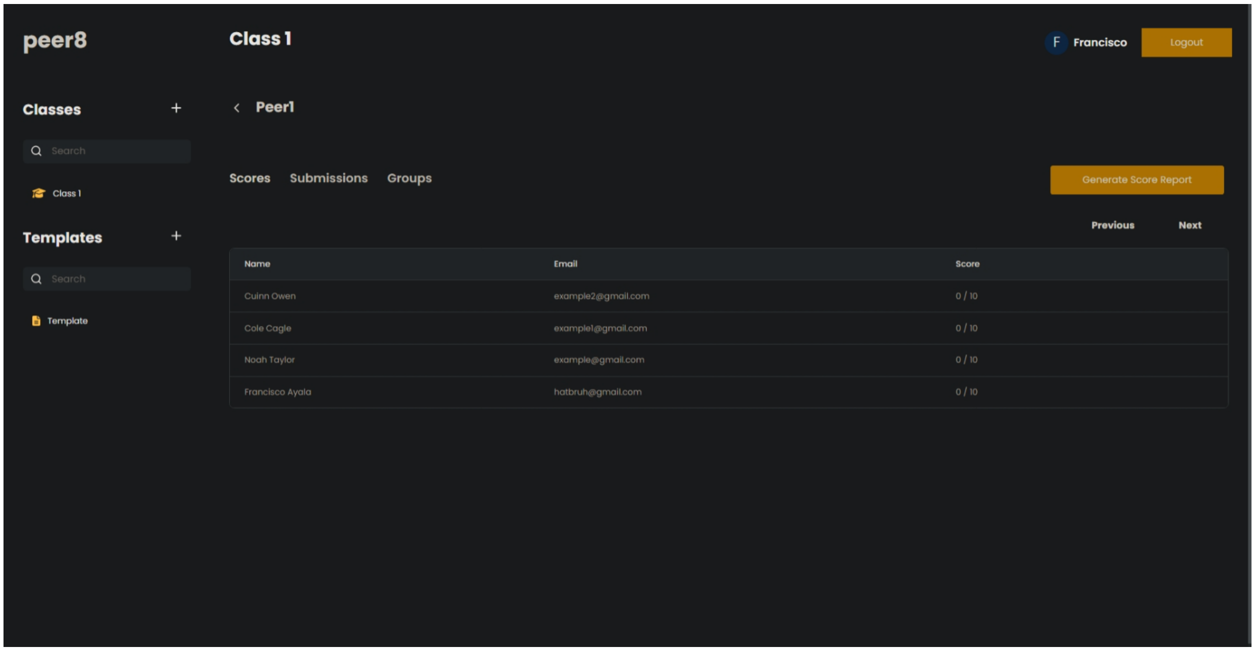
You can provide details for the peer review name, template, and due date. A template must be chosen from a dropdown in order to create the peer review. Make sure the due date is in the future as well and then click ‘**Create Peer Review**.’

Congratulations. You have created your first peer review. The table should look like below:



Name	Due Date	Status	
Peer1	4/27/2023	assigned	View Edit Close

You should have multiple options for a particular peer review. Closing a peer review will close the peer review immediately and disallow any future submissions from students, it will also aggregate scores immediately. For now, we will view the Peer Review by clicking ‘**View**.’

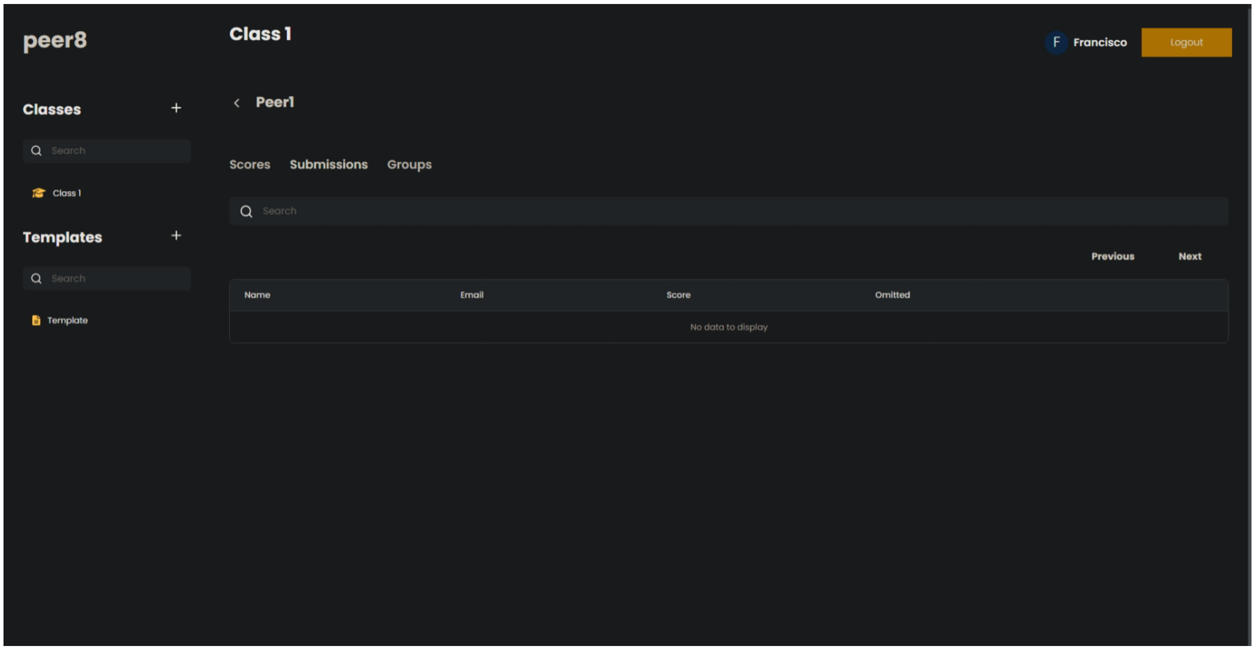


Name	Email	Score
Cuinn Owen	example2@gmail.com	0 / 10
Cole Cagle	example1@gmail.com	0 / 10
Noah Taylor	example@gmail.com	0 / 10
Francisco Ayala	hatbruh@gmail.com	0 / 10

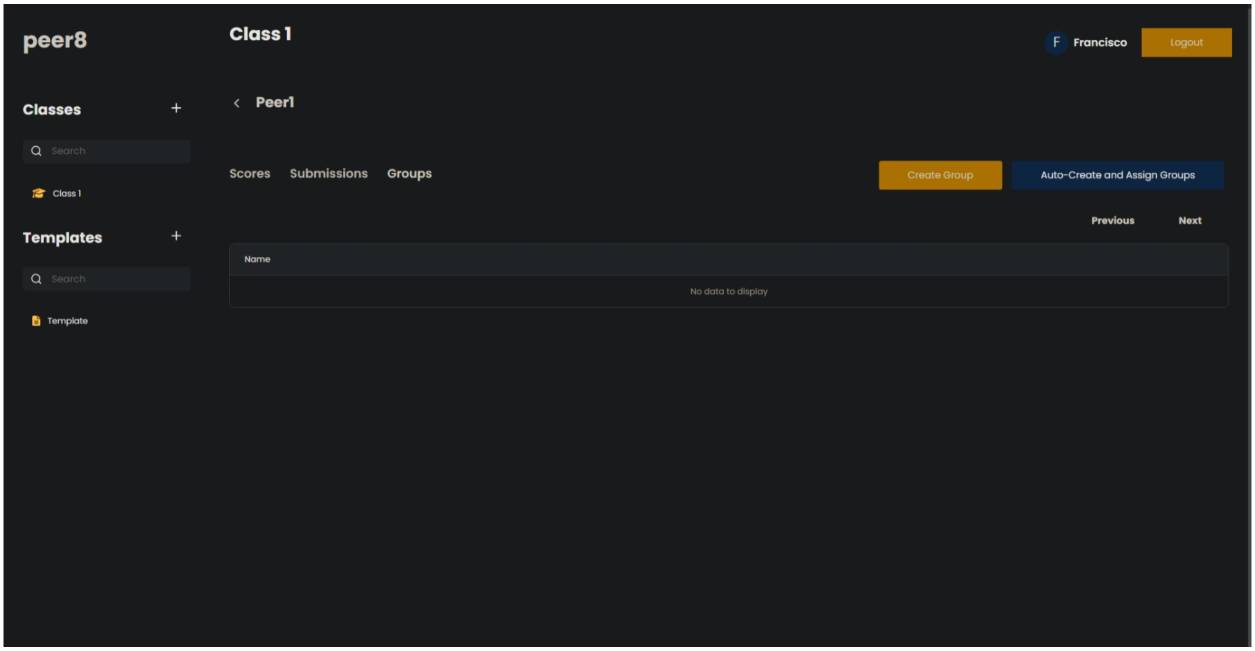
The above is the dedicated page for the Peer Review we have created. The ‘**Scores**’ tab is to see finalized scores from every student in the peer review. You will see a recurring theme in the top right corner of most tables which is **Pagination**. The data is retrieved on a page-by-page basis. To view the scores for all students, you will have to ‘**Generate Score Report**’.

Generating a score report will download an Excel sheet to your computer with all the scores for all the students in the class.

The **‘Submissions’** tab is where you will be able to see all student submissions and omit/un-omit a submission if needed.

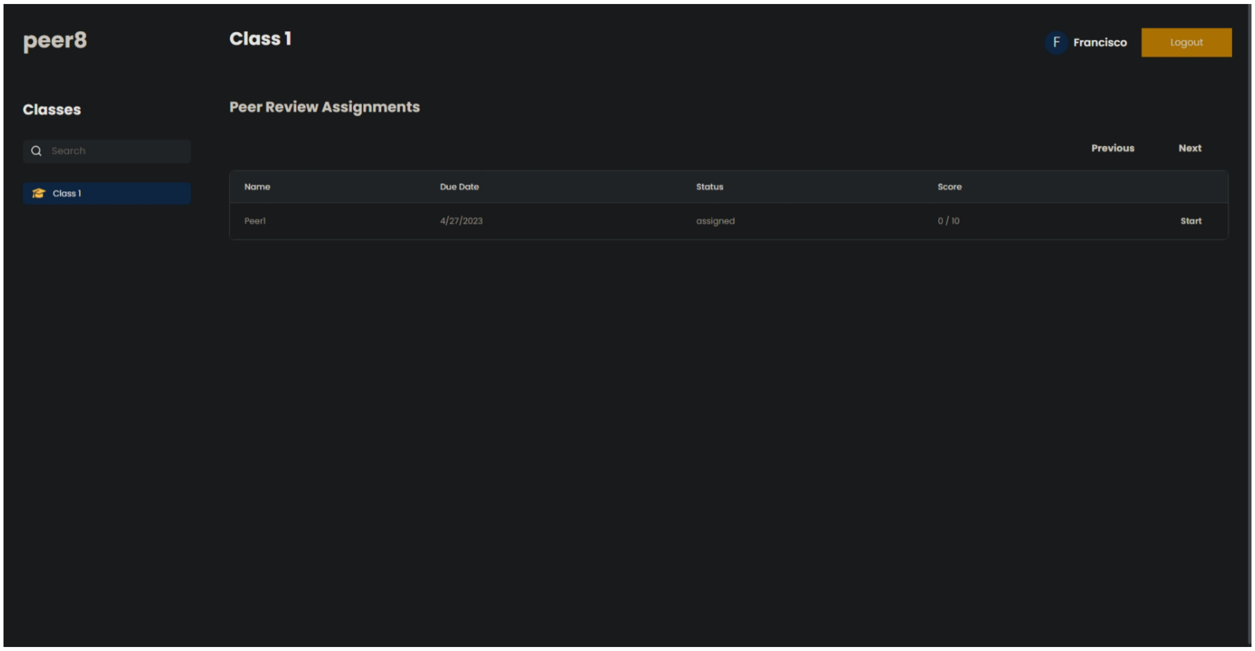


The ‘**Groups**’ page is integral to making sure a student can begin their peer review.



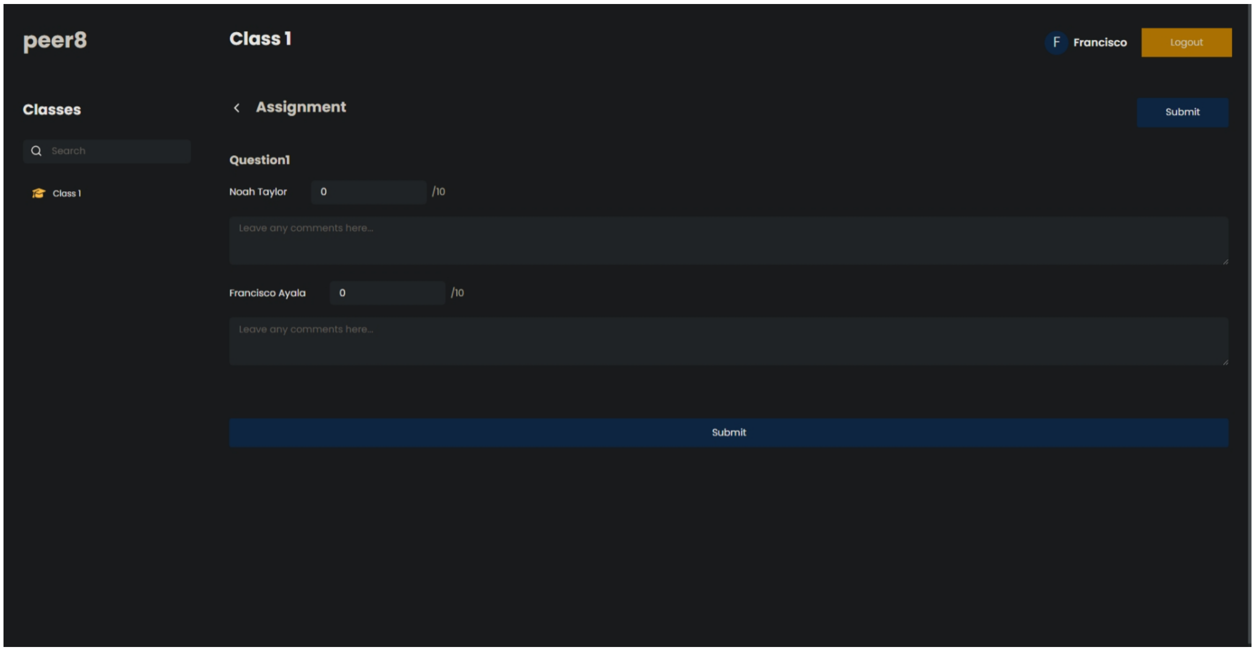
You can click ‘**Create Group**’ or ‘**Auto-Create and Assign Groups**.’ Create Group will prompt you with inputs for the group name and selection of students. Auto-create and Assign Groups will create multiple groups based on the number of students in the peer review. If the number of students is odd, there will be a group with a larger number of students than every other group.

Now we can get to the student view, assuming the student has created their account with the same associated email the instructor used to create the class.



Name	Due Date	Status	Score	
Peer1	4/27/2023	assigned	0 / 10	Start

The student can now click ‘**Start**’ to fill out the peer review or edit their submission. The assignment page looks like so:



Question1

Noah Taylor 0 /10

Leave any comments here...

Francisco Ayala 0 /10

Leave any comments here...

From here the student can submit, as can other students in order to fulfill the peer review and so scores can be aggregated. You will be able to run through the same walkthrough above to see updates to scoring and student submissions.

You can find a usable version of the web application (hosted on provided VM and on UTC's network) at: http://10.199.17.172:5173/

SECTION 7 - TEST AND VALIDATION

7.1 TEST PLAN

The test plan was created by the quality assurance manager and implemented with the assistance of the development team, and was developed after the initial planning phase of the project, and test cases were crafted in regard to the features stemming from the stakeholder's user stories. A total of eleven base test cases were created, each with various inputs to test functionality and work to validate user creation, secure logins, review assigning, score aggregation, and revision/commenting features. User authentication and security were considered at the very start of the creation of the test plan to ensure confidentiality and integrity for the peer8 system.

The test plan is as follows:

Authentication Test: Tests to ensure that the authentication login feature is working properly by testing each input for validity, confirming valid inputs are accepted, and confirming invalid inputs are rejected.

Template Creation Test: Tests to ensure templates made by teachers can be created and submitted by requesting to create template, designing template, requesting to submit template, and confirming template is submitted and accurate to what was previously created.

Template Edit Test: Tests to ensure templates made by teachers can be edited by confirming a pre-existing template is available, requesting to edit template, changing one question's wording, changing a separate question's scoring, changing a separate question's wording and scoring, submitting updated template, and confirming template changes have been applied.

Student Access Test: Tests to ensure student users have access to assigned peer reviews and can see them in their pending assignments by assigning peer review to test student account, logging into test student account, and confirming peer review is visible in pending assignments and can be accessed.

Review Deadline Test: Tests to ensure deadline for submissions are working properly by assigning peer review to test student account, logging into test student account after deadline for peer review has expired, and confirming student can't access expired review assignment.

Review Question Submission Test: Tests to ensure answers submitted on peer reviews are properly submitted and stored by assigning peer review to test student account, logging into test student account, accessing assigned peer review, applying ratings to questions on the review, submitting peer reviews, and logging into faculty account and confirm scores are accurate.

Review Comment Test: Tests to ensure comment system in peer reviews for students is working properly by assigning peer review to test student account, logging into test student account, accessing assigned peer review, applying ratings to questions on the review and comment on every question, submitting peer review, and logging into faculty account and confirm comments are visible on the peer review.

Review Submission Test: Tests to ensure completed reviews can be submitted by students by assigning peer review to test student account, logging into test student account, accessing assigned peer review, applying ratings to questions on the review, submitting peer review, and logging into faculty account and confirming the completed review is submitted.

Submission Revision Test: Tests to ensure completed reviews can be resubmitted by students by assigning peer review to test student account, logging into test student account, accessing assigned peer review, applying ratings to questions on the review, submitting peer review, logging into faculty account and confirm the completed review is submitted, relogging into test student account, changoing scores for every question by +1 or -1, resubmitting peer review, and relogging into the faculty account and confirm the edited review is submitted and scores are accurate.

Data Consolidation Test: Tests to ensure computations for all peer reviews post deadline are accurate by assigning peer review to test student account, logging into test student account, accessing assigned peer review, applying ratings to questions on the review, submitting peer review, repeating steps 2-5 on three different test student accounts in the same student group, and logging into faculty account and confirm scores are accurate and calculations accurately reflect individual and group scores.

Submission Omission Test: Tests to ensure instructors can choose to omit student review submissions by assigning peer review to test student account, logging into test student account, accessing assigned peer review, applying ratings to questions on the review, submitting peer review, repeating steps 2-5 on a different test student account, logging into faculty account and confirm scores are accurate, and omitting the first test student's scores and confirm new scores are accurate.

7.2 TESTING PROCESS

The testing of this application was done during and after the completion of the software to ensure the program meets stakeholder requirements, functions properly, and maintains a high level of security. Using the test plan and its respective test cases, the Quality Assurance Manager and developers accurately completed each test, documenting the results on the test plan. Out of the eleven base test cases, a total of nine test cases were successfully completed, with the two failures resulting in minute changes to implement the necessary features.

SECTION 8 - SECURITY PLAN

The security plan was created by the security manager and implemented with the assistance of the development team. The project manager provided oversight of the security plan. The overarching goal of the security plan is to protect user data from malicious actors or unauthorized users. The data that is presented by users of the application is processed in a manner to keep this data confidential between the instructor and their students. The data should not be manipulated by any unauthorized actions to keep integrity. The data should always be available to the instructors for their discretional usage. It is important that all users of the peer8: Peer Review Web Application read and understand the standards and procedures that were implemented. Only licensed users can use this application. A licensed user is described as someone who has purchased legal ownership of the application for its services. Any violations of this security plan can result in consequences from the educational institutions, state laws, and federal laws that govern the usage of this application.

8.1 SCOPE

The standards and procedures that are implemented by peer8 and the legal owners of the product are intended to be applied to any devices owned by the educational institutions that are capable of using the application, the servers provided by the University of Tennessee-Chattanooga to host the database, and any data that is stored on the servers. If any personal devices are used to access the application, they are subject to the standards and procedures that are outlined in the security plan.

8.2 STANDARDS

1. Data is owned by the University of Tennessee-Chattanooga due to it being stored on servers that are housed within their data center.
2. Devices are owned by educational institutions, such as desktops, laptops, etc. that can be found across campuses.
3. Personal devices are owned by the individual who purchased the device.
4. Data is meant to remain confidential to the student and the faculty member responsible for obtaining this data.
5. Data is meant to maintain integrity by not being accessed and modified by unauthorized users.
6. Data is meant to remain available and be accessed by the students for viewing and faculty members for grading.
7. Devices that have the application installed are subject to the AUP agreement that is signed by the students each semester and collected by the faculty members.
8. The servers provided by the University of Tennessee-Chattanooga are closely monitored by the IT department on campus.
9. The network provided by the University of Tennessee-Chattanooga is closely monitored by the IT department on campus.
10. Peer8's application is a licensed product that must be legally owned for its services.

11. The application's security plan is readily available for users to read and understand and must be posted digitally and physically for viewing.
12. The application will be used in settings that are overlooked by a security team to protect users from malicious actions.

8.3 POLICIES

1. Only licensed users should be able to use the application.
2. An account, whether it is a student or instructor, should only be used by that authorized user.
3. Email verification should be used to prevent needless accounts from being created.
4. Users should be able to reset passwords in the event they are forgotten or hacked.
5. Instructors have a level of privileges over students and this is distinguished when creating an account.
6. Email addresses that are used for registration must be associated with an appropriate institution.
7. Any accounts created with email addresses that do not meet these requirements should be deleted.
8. Only instructors should have the ability for customization while students do not.
9. Any user troubles should be reported to the development team.
10. If the development team implements maintenance times, all licensed users should be notified in advance.