6.3. Detailed CSC and CSU Descriptions

The Async website is a classroom discussion board platform for classrooms. Using Node.js, React, and Firebase there are multiple Computer Software Components (CSCs) that are made up of Computer Software Units (CSUs).

User Interface CSC

The User Interface handles the interactive page of the website. This is what the users will be interacting with when they are using the website. It uses React and Tailwind CSS to create an intuitive and user friendly design.

User Interface CSUs:

Discussions List: All discussions ordered from the top being most recent to the bottom being the oldest.
Create New Discussion: Allows for creation of a new discussion, with title and text box.
Discussion: Features title and text attached.

Backend CSC

The Backend CSC will be handled by Firebase. It will remember the discussion board text, title, author, timestamp, and other attributes to ensure data consistency and history.

Backend CSUs:

Discussion Object: Defines a *Discussion* object. It will feature attributes like title, author, time, etc.

Authentication CSC

The Authentication CSC is a layer of security for the website to ensure that the people that have access to the discussion boards are allowed to have access.

Authentication CSUs

Authentication CSU: Logins will be handled by Google to ensure classroom security.
Class Codes: Class codes will be provided by the administrator to verify identity.

6.3.1 Detailed Class Descriptions

The following are descriptions of the classes used in the Async application. These classes are in progress and are subject to change.

6.3.1.1 Discussion Class

The Discussion class represents a single discussion on the website
Fields:

- Id: Provides an identifier for the discussion thread.
- Title: Title of the discussion.
- Content: The text of the discussion.
- CreatedAt: Timestamp for discussion creation.
- Comments: Array of comments attached to the discussion.

6.3.1.2 Discussion Component

This component of Discussions displays the list of available Discussions.
Fields:

- selectedDiscussion: Holds the currently selected discussion.
- Discussions: An array containing all discussions available.
- newDiscussionTitle: Creates a new discussion.
- newDiscussionContent: Creates new text for a discussion.
- newComment: Creates a new comment on a discussion.
- showForm: Shows or hides a discussion.

Methods:

- handleDiscussionClick(discussion): Selects discussion.
- handleAddDiscussion(): Allows for discussion to be viewed.
- handleSubmit(e): Processes submission for new discussion.
- handleAddComment(): Adds a comment.

6.3.1.3 Login Component

The login component is what allows users to login through a UI.
Methods:

- Login: Allows users to login using Username and Password.

6.3.2 Detailed Interface Descriptions Section

6.3.2.1 Login Interface

The login interface provides security to the website by authenticating user access through usernames and passwords

Inputs:

- Username: Entered by user. Processed by backend.
- Password: Entered by user. Processed by backend.

Outputs:

- Authentication token: If successful login, show user access has been granted.
- Error Message: In case of invalid access(incorrect username or password, or unauthorized) .

6.3.2.2 Discussions Interface

The Discussions interface is where the users will be able to view discussions.

Inputs:

- User Clicks: Selects discussions that the user wants to be displayed.
- Discussion Creator: Opens a text box where users can create a new discussion.
- Comments Creator: Opens a new text box where users can create a new comment to be attached to a discussion.

Outputs:

- Discussions List: A list of discussions on the left of the board.
- Discussion: The discussion selected by the user to be displayed.
- Discussion Comments: Comments attached by users to the selected discussion.

6.3.3 Detailed Data Structure Descriptions

6.3.3.1 Data Structures

The following are data structures associated with the Async website. There are User, Discussions, Comment, and Authentication data structures.

6.3.3.1.1 User Data Structure:

This contains the necessary information for a user to be stored in data.

Fields:

- userID: Unique id for each user.
- Username: Username for login for user.
- Password: Password for login for user.

6.3.3.1.2 Discussion Data Structure:

This contains the necessary information for discussion to be stored as data.

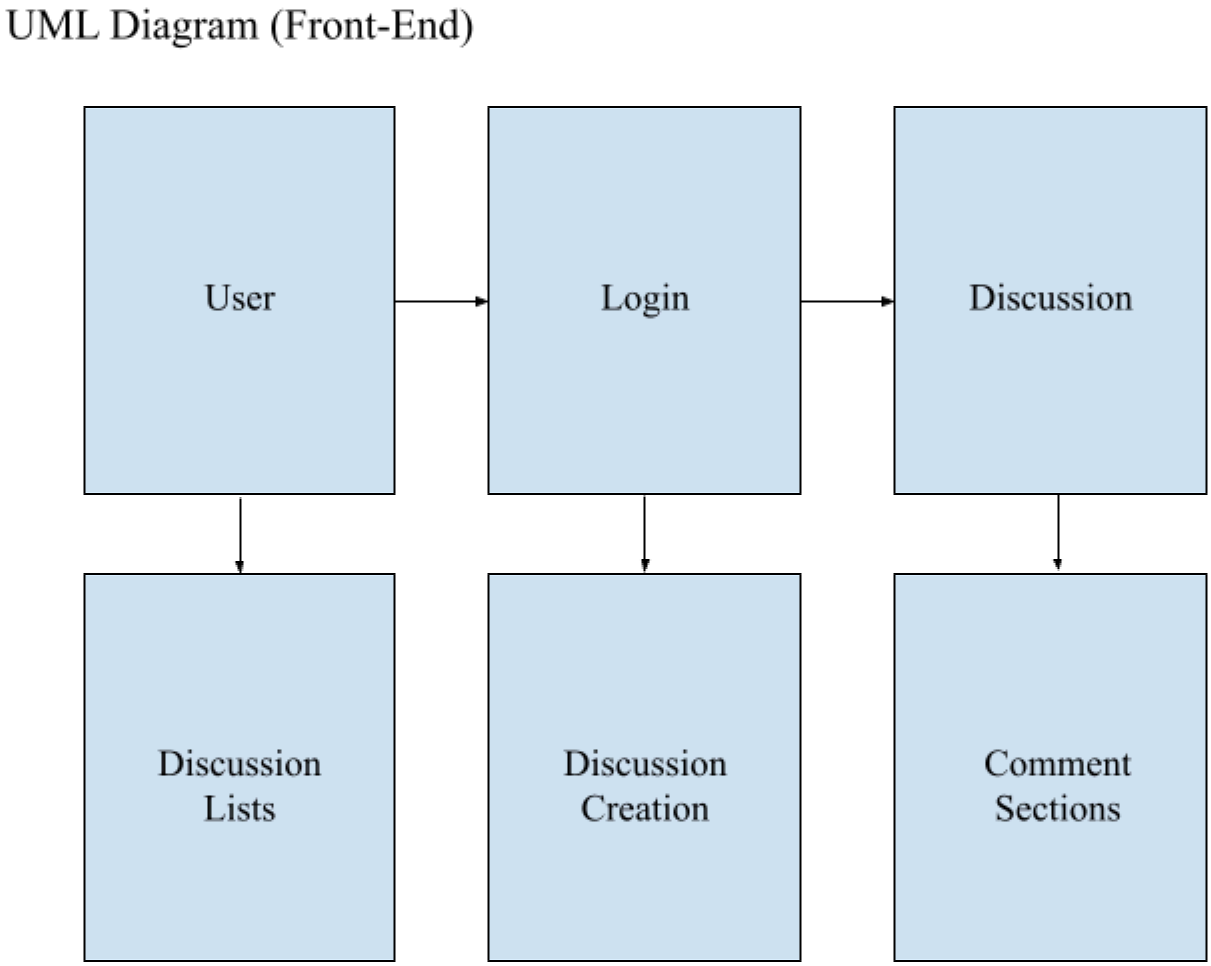
Fields:

- Id: Unique id for each discussion.
- Title: Title for each discussion.
- Content: Content for each discussion.

- Author: The author of each discussion.
- CreatedAt: A timestamp for the creation of the discussion.
- Comments: A list of comments that were attached to the discussion.

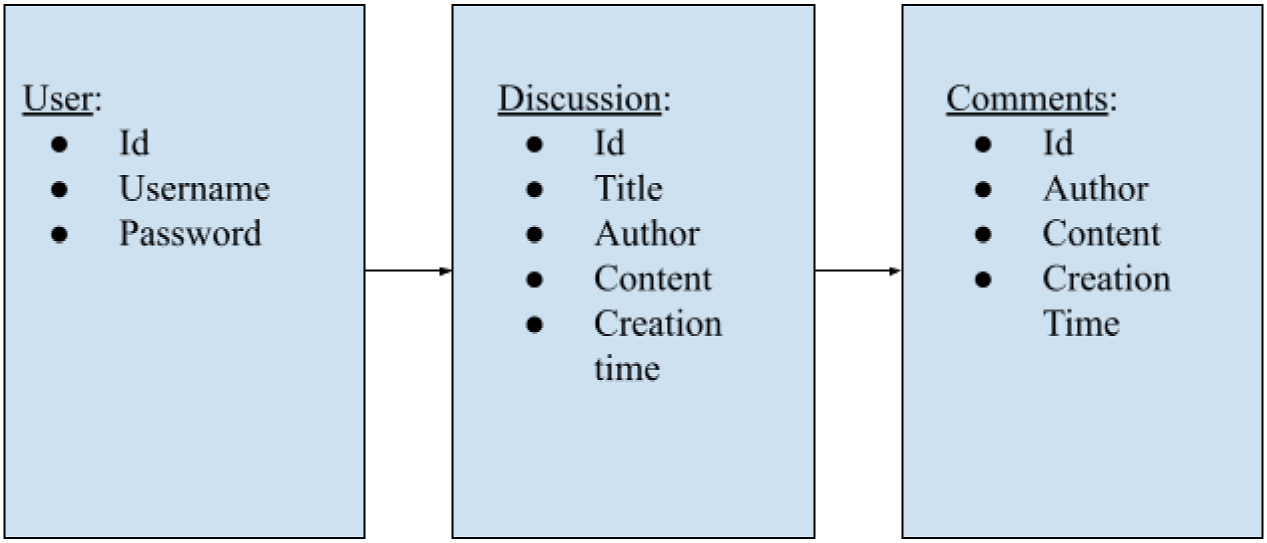
6.3.4 Detailed Design Diagrams Section

6.3.4.1 Front-End UML Diagram



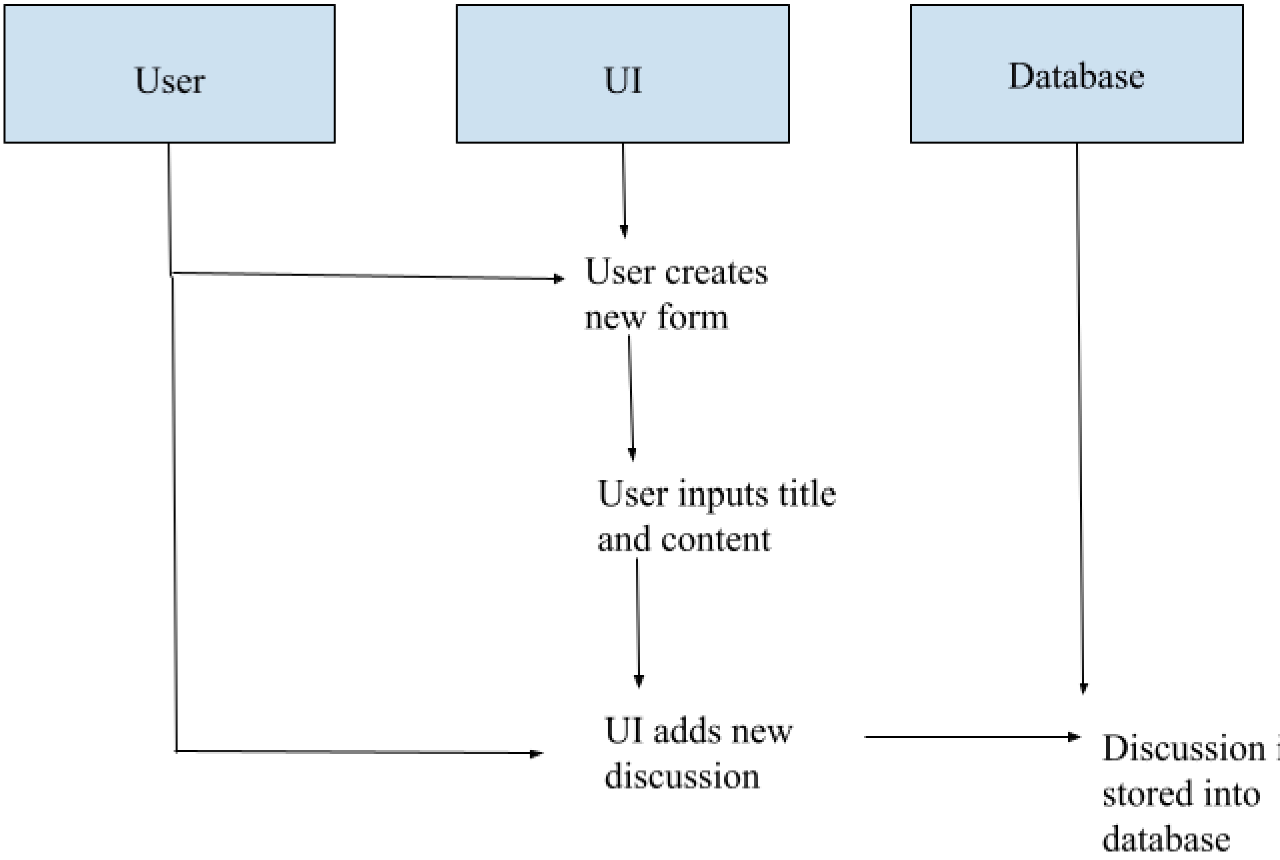
6.3.4.2 Entity Database Interactions

Entity Database Interactions



6.3.4.3 Entity Communication

Entity Communication

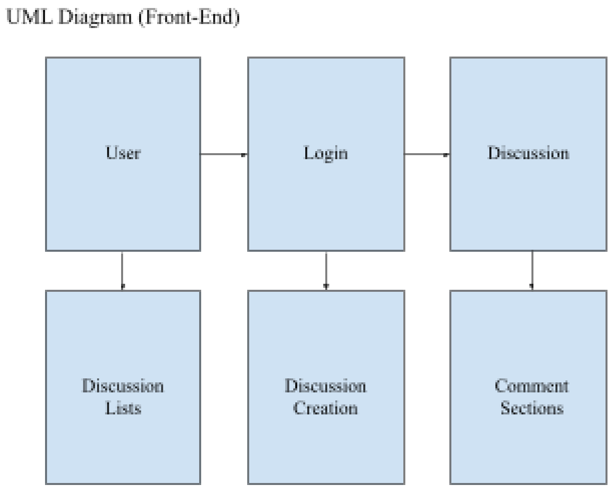


6.4 Database Design and Description Section

The database utilizes Google Firebase to store data.

6.4.1 Database Entity Relationship

The entity-relationship diagram illustrate the relationships between the the user, discussions, and comments.



Users:

- Users can login, see discussion lists(left side of screen), and a discussion.

Login:

- After a user logs in, they can create discussions and view discussions and comment sections.

Comments:

- The final entity viewable are comment sections, which requires a user to login, view the discussion lists, select a discussion, then view the comments added to the discussion.

6.4.2 Database Access

Database access will be very limited since Firebase provides extensive security and authentication. Firebase allows for **role-based access,** which will allow for user-specific roles to be defined within Firebase as to limit who can access and edit and who are read-only.

6.4.3 Database Security

Firebase authentication ensures that real and verified users are the only ones who can access the database. It also comes with security rules and data encryption to create a secure environment.