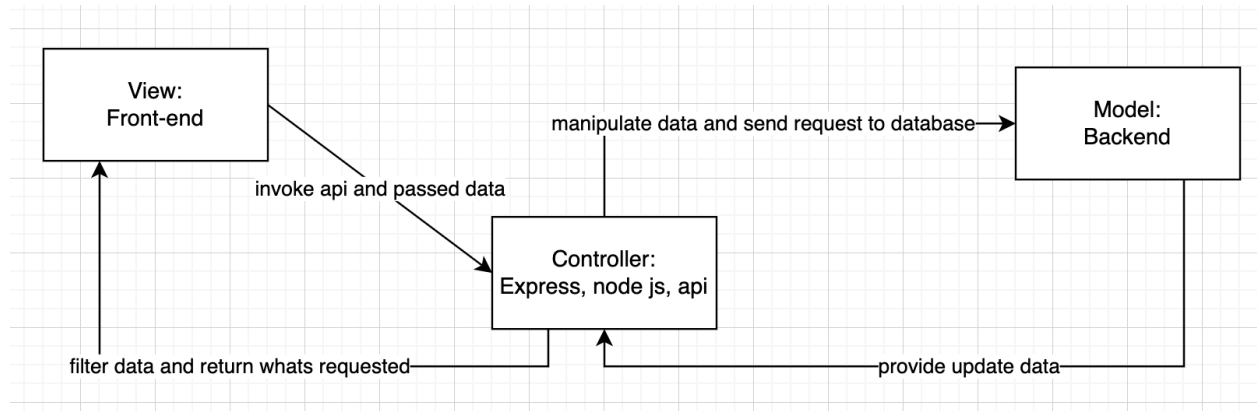


# System Design Document

# Table Of Content

<b>Software Architecture Diagram</b>	<b>3</b>
<b>CRC Cards</b>	<b>4</b>
Backend CRC cards	4
Frontend CRC Cards	9

## Software Architecture Diagram



Our architecture design follows mvc pattern such that the front-end will invoke api and pass data to the controller which will manipulate data and send it to our back end database. The backend would then return updated data that the controller requested. The controller would then update the front-end.

<https://www.geeksforgeeks.org/mvc-design-pattern/>

# CRC Cards

## Backend CRC cards

API name: question Url: GET <a href="http://127.0.0.1:8000/api/questions">http://127.0.0.1:8000/api/questions</a>	
Required field: None	
Responsibilities: <ul style="list-style-type: none"> <li>- Returns all the data related to each questions</li> <li>- Keeps tracks on the status of the question, which user are associated with it</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>- userID</li> </ul>

API name: question Url: POST <a href="http://127.0.0.1:8000/api/questions">http://127.0.0.1:8000/api/questions</a>	
Required field: Title, status, talD, userID, body, bounty, date	
Responsibilities: <ul style="list-style-type: none"> <li>- Create and track all the given data under a new question</li> </ul>	Collaborators:

API name: question Url: PUT <a href="http://127.0.0.1:8000/api/questions">http://127.0.0.1:8000/api/questions</a>	
Required field: Question ID, and fields that needs to be change	
Responsibilities: <ul style="list-style-type: none"> <li>- Find the questions with the given ID and update the data</li> </ul>	Collaborators:

API name: question Url: DELETE <a href="http://127.0.0.1:8000/api/questions/:id">http://127.0.0.1:8000/api/questions/:id</a>	
Required field: Question ID	
Responsibilities: <ul style="list-style-type: none"> <li>- Delete the question associated with this ID</li> <li>- Return the question ID</li> </ul>	Collaborators:

API name: setProfile Url: POST <a href="http://localhost:8000/api/profile">http://localhost:8000/api/profile</a>	
Required field: Name:str ID:num	
Responsibilities: <ul style="list-style-type: none"> <li>- Create a user with name and id</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>- userID</li> </ul>

API name: getProfile Url: GET <a href="http://localhost:8000/api/profile/id">http://localhost:8000/api/profile/id</a>	
Required field: ID:num	
Responsibilities: <ul style="list-style-type: none"> <li>- Get stored user information from id</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>- userID</li> </ul>

API name: deleteProfile Url: DELETE <a href="http://localhost:8000/api/profile/id">http://localhost:8000/api/profile/id</a> :	
Required field: ID:num	
Responsibilities: - Delete stored user information from id	Collaborators: - userID

API name: updateProfile Url: PUT <a href="http://localhost:8000/api/profile/id/name">http://localhost:8000/api/profile/id/name</a> :	
Required field: Name:str ID:num	
Responsibilities: - Delete stored user information from id - Change user name in this case	Collaborators: - userID

API name: currency Url: POST <a href="http://127.0.0.1:8000/api/currency">http://127.0.0.1:8000/api/currency</a>	
Required field: None	
Responsibilities: - Create an api related to this user - Keeps tracks on the amount the user have	Collaborators:

API name: currency Url: GET <a href="http://127.0.0.1:8000/api/:userID">http://127.0.0.1:8000/api/:userID</a>	
Required field: UserID	
Responsibilities: - Return the amount this user has	Collaborators:

API name: currency Url: PATCH <a href="http://127.0.0.1:8000/api/add/:userID">http://127.0.0.1:8000/api/add/:userID</a>	
Required field: UserID	
Responsibilities: - Add to the amount this user has	Collaborators:

API name: currency Url: PATCH <a href="http://127.0.0.1:8000/api/subtract/:userID">http://127.0.0.1:8000/api/subtract/:userID</a>	
Required field: UserID	
Responsibilities: - Subtract the amount of this user - Return error if not enough amount	Collaborators: Question, user

## User-account\_feature CRC

Api name: registerUser url: POST http://localhost:8000/api/user	
Required field: <pre>{   "username": ???,   "email": ???,   "password": ??? }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>Users are able to create an account based on their email.</li> <li>Users can create their own password and username</li> <li>Users don't have to login again since information is stored in cookie with JWT Token</li> <li>Users' password are hashed in database</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>JWT</li> <li>Cookie</li> <li>bcrypt</li> <li>Users (schema)</li> </ul>

Api name: loginUser url: POST http://localhost:8000/api/user/login	
Required field: <pre>{   "username": ???,   "password": ??? }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>Users with correct username and password should be able to login with this API, and the cookie should store the information.</li> <li>If a user fail to login because he provided wrong password and username the frontend should receive the correct response</li> <li>this API will compare the password that user provided with the one in the database</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>bcrypt</li> <li>cookie</li> <li>User (schema)</li> <li>JWT</li> </ul>

Api name: logout url: POST http://localhost:8000/api/user/logout	
Required field: <pre>{ }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>clean the cookie</li> <li>send notification to frontend</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>Cookie</li> </ul>

Api name: changePassword url: PATCH http://localhost:8000/api/user/changePassword	
Required field: <pre>{   "password": ??? }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>Change the password of the current user (from cookie)</li> <li>Hash the new password</li> <li>Store the password into backend</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>bcrypt</li> <li>Cookie</li> </ul>

Api name: getUser url: GET http://localhost:8000/api/user/:userId	
Required field: <pre>{ }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>Return the user's email and username based on userId provided in URL</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>User (schema)</li> </ul>

Api name: getAllUser url: GET http://localhost:8000/api/all	
Required field: <pre>{ }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>Return all user in the database with their email and username</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>User (schema)</li> </ul>



## Frontend CRC Cards

Class Name: Register page	
Parent Class: None Sub Class: None	
Responsibilities: <ul style="list-style-type: none"> <li>- A form with name, email, password, re-enter password and submit button input for the registration process.</li> <li>- Send the form data to the backend by calling the backend API.</li> <li>- A navigation bar to toggle from register page to login page.</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>- registerUser</li> <li>- React</li> <li>- Tailwindcss</li> <li>- React-redux</li> </ul>

Class Name: Login page	
Parent Class: None Sub Class: None	
Responsibilities: <ul style="list-style-type: none"> <li>- A form with name, password and submit button input for the login process.</li> <li>- Send the form data to the backend by calling the backend API.</li> <li>- A navigation bar to toggle from register page to login page.</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>- loginUser</li> <li>- React</li> <li>- Tailwindcss</li> <li>- React-redux</li> </ul>

API name: getQuestion Url: GET <a href="http://127.0.0.1:8000/api/questions/:id/:category?">http://127.0.0.1:8000/api/questions/:id/:category?</a>	
Required field: title	
Responsibilities: <ul style="list-style-type: none"> <li>- Return the matching question for corresponding category/title, whichever one is included</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>• questionSchema</li> <li>• MongoDB</li> <li>• Javascript</li> </ul>

API name: getQuestionById Url: GET http://127.0.0.1:8000/api/questions/:id/q_id/:question_id	
Required field: None	
Responsibilities: <ul style="list-style-type: none"> <li>- Return the matching question for corresponding question_id.</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>• questionSchema</li> <li>• MongoDB</li> <li>• Javascript</li> </ul>

## Frontend CRC Cards

Class Name: Dashboard	
Parent Class: App Sub Class: Sidebar, NewQuestion, Profile	
Responsibilities: <ul style="list-style-type: none"> <li>- A main stage to display different pages when it is called.</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>- React</li> <li>- Tailwindcss</li> <li>- React-redux</li> <li>- react-router-dom</li> </ul>

Class Name: NewQuestion	
Parent Class: Dashboard Sub Class: None	
Responsibilities: <ul style="list-style-type: none"> <li>- A form with title, body and bounty with a button “publish” for posting a new question process.</li> </ul>	Collaborators: <ul style="list-style-type: none"> <li>- React</li> <li>- Tailwindcss</li> <li>- React-redux</li> <li>- axios</li> </ul>

Class Name: Profile
---------------------

Parent Class: Dashboard Sub Class: None	
Responsibilities: <ul style="list-style-type: none"><li>- A form with current username, email, new password and confirm new password with a button "save chage" for updating a user's profile process.</li></ul>	Collaborators: <ul style="list-style-type: none"><li>- React</li><li>- Tailwindcss</li><li>- React-redux</li><li>- axios</li></ul>

Api name: <i>changeUserInfo</i> url: <i>PATCH</i> <i>http://localhost:8000/api/user/changeUserInfo</i>	
Required field: <pre>{   "id": ???,   "password": ???,   "username": ???,   "email": ??? }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>Based on the given field change the user informations</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>bcrypt</li> <li>Cookie</li> </ul>

Api name: <i>google</i> url: <i>GET</i> <i>http://localhost:8000/auth/google</i>	
Required field: <pre>{ }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>use passport to authenticate with scope of profile and email</li> <li>Get a google user's information from google provider</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>passport</li> </ul>

Api name: <i>google</i> url: <i>GET</i> <i>http://localhost:8000/auth/google/callback</i>	
Required field: <pre>{ }</pre>	
Responsibilities: <ul style="list-style-type: none"> <li>callback function of google passport</li> <li>Whenever google OAuth is invoked handle it's action if succeed, save user id to cookie</li> </ul>	Collaborator: <ul style="list-style-type: none"> <li>passport</li> <li>cookie</li> </ul>