# System Design

Table of Contents

[System Design 1](#_Toc485680237)

[CRC CARD 3](#_Toc485680238)

[Architecture Diagram 6](#_Toc485680239)

[Directory Structure 7](#_Toc485680240)

## CRC CARD

|  |  |
| --- | --- |
| ***User*** | |
| * Username * Major * School * Current Year * Resume * Profile Pic * Phone number * Register as a user * Log in * Search Chatroom by course code, semester, year * Set as a professor enrolled in course * Set as a TA enrolled in course * (Default – student enrolled in course) * Inside chatroom, set as chatroom manager (by other managers) * Enroll into chatroom to communicate with fellow classmates * Check chatting histories * Chat in chatroom [casually with classmates] * Post questions * Answer questions * Click on other registered users * Send private messages to other users * Adopt an answer to resolve post * Has the ability to adopt an answer to be able to resolve post * Search for posts in chatrooms through keywords * Send messages anonymously * Filters unwanted posts [or not as important posts] in chatrooms * File complaints on inappropriate posts * Access all functionality in app * Cannot hack in using CSS Injection | Server  Login  SignUp  Profile  Chatroom  Post  CrowdSourcing |
| ***Chatroom*** | |
| * Course (for that chatroom) * All [enrolled] Users (inside chatroom) * Chatroom admins (mediators/managers) * Keep track of historic posts * Notify all users about alerts/important posts * Keep track of posts that are alerts/important posts * Private Messaging (PM) with other users in the chatroom * Collect tags from collection of posts | User  Server |

|  |  |
| --- | --- |
| ***Post*** | |
| * Post Content * Date posted * User association of post * Save post in database * Resolved post with utilization and adoption of an answer | User  Chatroom  Server |

|  |  |
| --- | --- |
| ***CrowdSourcing*** | |
| * Upload file to server * Download file from server * Sharing file with other user | User  Controller[s] |

|  |  |
| --- | --- |
| ***Sign Up*** | |
| * Obtain username * Obtain first name * Obtain last name * Obtain password * Obtain UTORid * Create new user using this info | User  Server  Controller[s] |

|  |  |
| --- | --- |
| ***Login*** | |
| * Obtain username * Obtain password * Authenticate within databases using username and password | User  Server  Controller[s] |

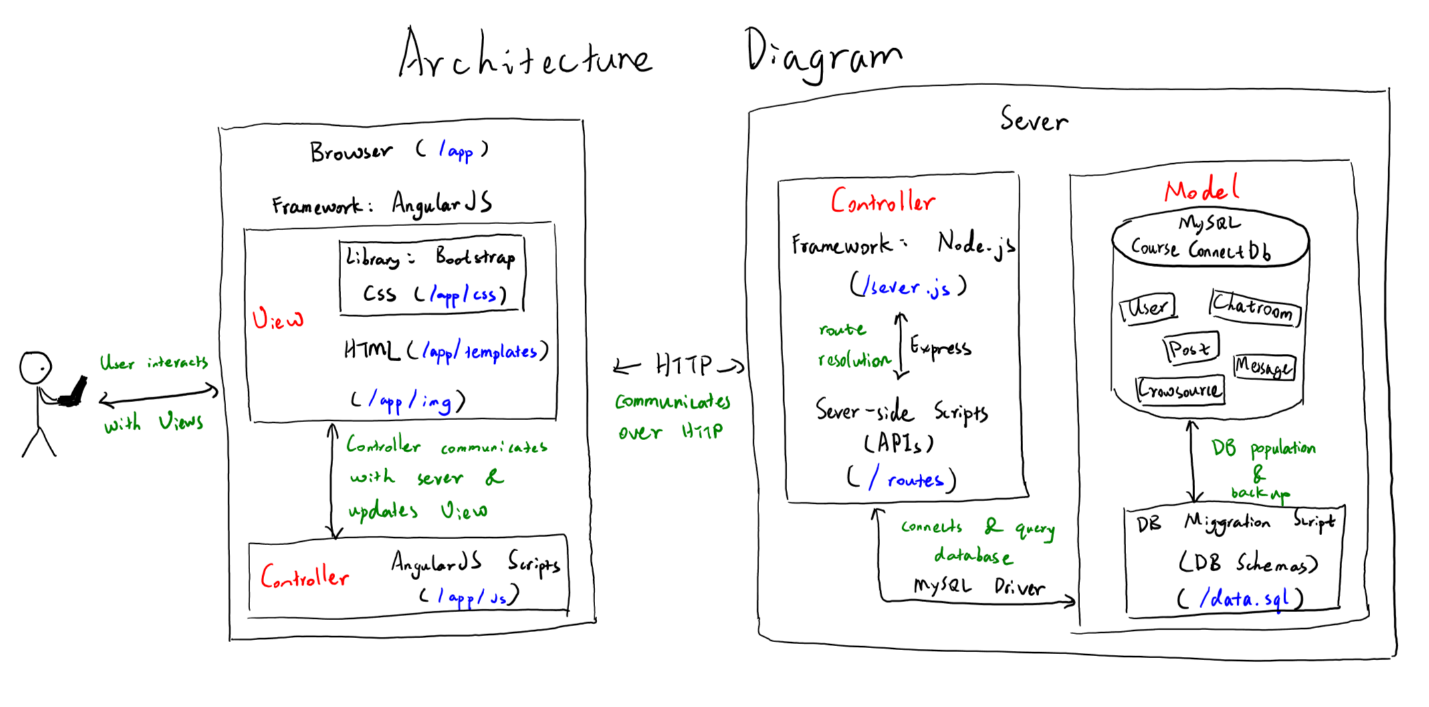
|  |  |
| --- | --- |
| ***ProfilePage*** | |
| * Upload and set profile picture * Set first name and last name * Change first name and last name * Upload resume * Set major * Change password | User  Controller[s] |

|  |  |
| --- | --- |
| ***LandingPage*** | |
| * Navigate to another portion of the application (gateway to user profile and chatroom[s]) | User  Controller[s] |

|  |  |
| --- | --- |
| ***Server*** | |
| * Connects to database * Connects to frontend * Authenticates HTTP requests * Returns HTTP response * Sends errors when needed | User  Controller[s] |

|  |  |
| --- | --- |
| ***Controller*** | |
| * Response to user interaction * Communicate with server | Server |

## Architecture Diagram



## Directory Structure

