Summary of Completed Features for Minimum Core Functionality

Roarke DeCrewe | S#:32026403

All elements of project will be graded based on how well they are completed

Green - Completely implemented

Yellow - Partially implemented, either only partial implementation or full implementation with bugs

Red - Yet to be implemented

Baseline Objectives:

- **Browse Site without registering:** Can view all pages for other users (profiles and lessons) and you check the main page
- **Search for items without registering:** Any type of user can search via the main page or on the search page itself to find posts or creators that have any of the search text in the title or display name
- **Register on the site:** A user can register via the register tab, found on any page when a user is not currently logged in. Must input display name, user name, email, password, and profile picture. After registering, user is instantly logged in
- **Login with username and password:** Can be accessed on any page when not logged in, and requires username and password, and if input correctly will be redirected to main page and be logged in.
- **Create and comment when logged in:** While not exactly “comments”, users can send a message to any profile on the site, where it will then be delivered to that user’s inbox.
- **Users can view/edit their profile:** Users can access and view their profile at any time via the profile button in the top left hamburger menu. From their they can edit their profile, and will be redirected back to their profile
- **Admin search by name, email, or post:** You can search with the search tab and find by both name or username. Email search is not used anywhere after being registered.
- **Admin Enable/disable users:** You can’t disable users, but admins can delete any account they want.
- **Admin Edit/remove posts:** Admins can remove posts, but no post editing has been implemented for admins or users

Minimum Functionality Requirements:

- **Hand-styled layout with contextual menus:** The website was designed with a specific theme in mind, with a large amount of CSS to make it look unique for its “Wizard” theme. Menus that appear are generally static page to page, but login and signup only show the other option rather than both.
- **2 or 3 column layout using appropriate design principles, responsive design:** The website uses a rough 2 column layout, depending on the page, with the page having responsive elements in the multiple carousels, moving menus, and buttons/images which change color when hovered.

- **Form validation with JavaScript:** Form inputs are validated all via javascript, on Registration, Login, Create Lesson, and Edit Profile. We check both formatting of inputs, and making sure their are actual inputs for required fields
- **Server-side scripting with PHP:** The majority of the site is set up via php, with almost every page having inline php for formatting our UX for registered vs non-registered vs administrator users, while non-display php is used to interface with the image storage and SQL database.
- **Data storage in MySQL:** All information on the server, including user, lesson, message, and image data with a hierarchy of Foreign Key Relations such that they are all related to one other.
- **Appropriate security for data:** All passwords are md5 hashed on insert and compared to the md5 of the input for comparison. Input data is sanitized via mysqli_real_escape_chars to avoid SQL Injection attacks, and all important information is stored off site away from the ability of users to access.
- **Site must maintain state:** A username session var stores the name of a logged in user, and is used to determine whether the user is logged in and whether they are an administrator. The logout function or any error in login or registration wipes the session variables to avoid getting a half-logged in state.
- **Responsive design philosophy (minimum requirements for different non-mobile display sizes):** The CSS has been designed with % as to allow the website to work (while not looking fantastic) at variable sizes on a monitor, though it was not designed with mobile in mind.
- **AJAX (or similar) utilization for asynchronous updates:** The inbox system is done via a JS fetch loop which decodes message data from the SQL server into JSON packets constructed in a php file. This system allows your messages to live update every 5 seconds without refreshing the page.
- **User images (thumbnail) and profile stored in database:** The images are not directly store via blobs (as large images could lag the server immensely', rather the randomly generated name of the image is stored, and it is routed to via an image.php file which is used to access all images within pages.
- **Simple discussion (topics) grouping and display:** Lessons and profiles are displayed on multiple different pages in different ways. The main page orders them by popular lessons, soon to happen lessons, and popular creators. The search page sorts by lessons and creators according to whether their name or displayname has the text searched for. Lessons show lessons that you have either created or enrolled in. Thus the "topics" on display or accessed in multiple ways, all leading to the central profile or lesson page which has information about the respective topic and ways to interact with it.
- **Navigation breadcrumb strategy (i.e. user can determine where they are in threads):** While "threads" are not a relevant thing to this project, you can generally tell where you are by the page heading, or what profile/lesson you are on by the title or display name shown.
- **Error handling (bad navigation):** Through testing, we have removed all bad page navigation, such that if you go to pages that are for logged in users only, or

navigate to profile page without a profileid, they will all reroute you to the main page.

Project Deliverable

- **Client-side security**: Clients data is not being exposed on client side, as all privileged (password) data is encrypted prior to it being sent anywhere.
- **Posted on cosc360.ok.ubc.ca:** The website is posted and usable on cosc360.ok.ubc.ca, and has been filled with users and posts by volunteers
- **Server-side security:** Data being sent to server is sanitized prior to it being sent, anything which adds, updates, or removes data is done via transaction as to avoid partial changes to the database, and cascading rules have been added to avoid hanging data in the system.
- **Discussion thread storage in database:** All profile, lesson, and message data is stored remotely on the server, and is updated and added by users in real time.
- **Asynchronous updates:** The inbox is updated via AJAX (see above).
- **Database functionality complete:** The DB scheme is complete given what is expected of the website, and will need no further additions.
- **Core functional components operational (see baseline objectives):** Successful (See above)
- **Preliminary summary document, indicating implemented functionality:** You're reading it!
- **Submit a link to your repo and a PDF document that describes what you have done regarding the points listed above.:** Done!

NOTE: There is one unimplemented page, modpage.html, which is what is to be used for the additional features in the final submit, and had been left in for testing purposes and feature completeness on all pages leading to the modpage!