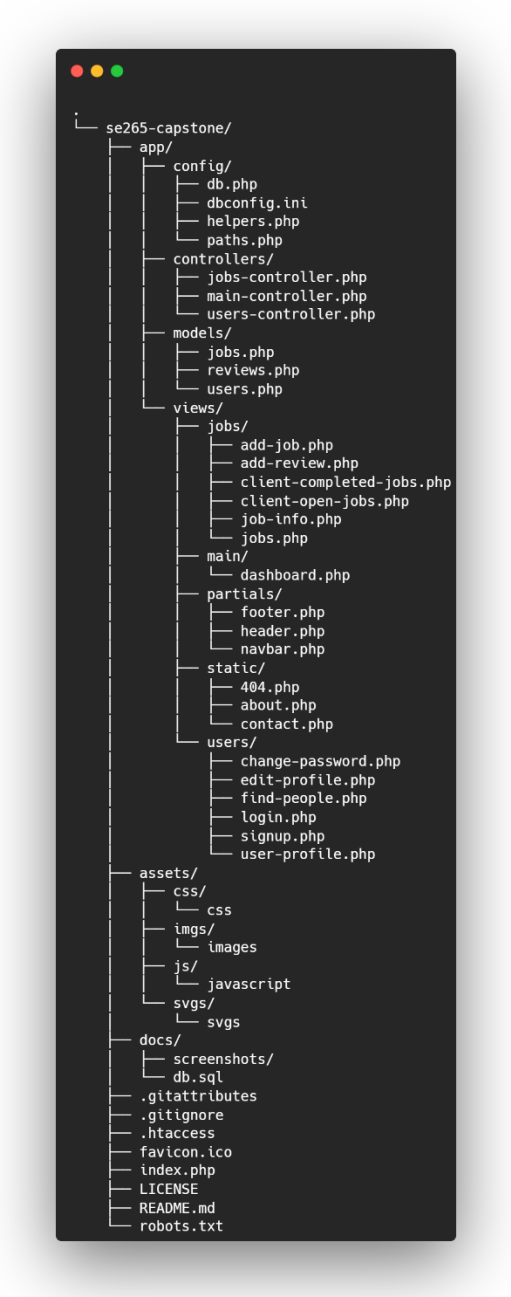
Technical Design

A.J. Saporito and Tristen Jussaume

PHP Procedural MVC

Directory Tree:



Router:

This is the site's index and handles all requests and responses. For every request, the URI is checked and evaluated against an array of routes. If the route exists, the correct resources are loaded; otherwise, a 404 HTTP error is thrown. This file ‘index.php’ is the brain of the website and is the only file ever rendered straight to the browser.

Controllers:

Three files store all the functions responsible for rendering content. Functions can access the model functions, along with loading in views and AJAX calls. All functions here are named render {whatever action}, e.g., renderLogin().

Views

Views contain HTML-like markup and are responsible for how each page looks.

Models:

Functions that connect to the MySQL database are located here. They are documented below:

~ users.php

getAllUsers()

- **Description**: Fetches all users from the Users table.
- **Parameters**: None.
- **Returns**: An array of all users if successful, otherwise an empty array.

getUserId($username)

- **Description**: Fetches the user ID based on the provided username.
- **Parameters**: $username (string): The username of the user.
- **Returns**: The user ID if found, otherwise null.

getUserById($id)

- **Description**: Retrieves user information based on the user ID.
- **Parameters**: $id (int): The user ID.
- **Returns**: An associative array with user information if found, otherwise an empty array.

getUserRecord($id)

- **Description**: Fetches a user record based on the user ID.
- **Parameters**: $id (int): The user ID.
- **Returns**: An associative array with the user record if found, otherwise an empty array.

userExistsById($id)

- **Description**: Checks if a user exists based on the user ID.
- **Parameters**: $id (int): The user ID.
- **Returns**: true if the user exists, otherwise false.

userExistsByUsername($username)

- **Description**: Checks if a username is already taken.
- **Parameters**: $username (string): The username to check.
- **Returns**: An array with success and message indicating whether the username is taken.

userExistsByEmail($email)

- **Description**: Checks if an email address is already taken.
- **Parameters**: $email (string): The email to check.
- **Returns**: An array with success and message indicating whether the email is taken.

signUp($firstName, $lastName, $username, $email, $password)

- **Description**: Creates a new user account.
- **Parameters**:
    - $firstName (string): User's first name.
    - $lastName (string): User's last name.
    - $username (string): Desired username.
    - $email (string): User's email address.
    - $password (string): User's password.
- **Returns**: true if the user was successfully created, otherwise false.

logIn($username, $password)

- **Description**: Authenticates a user based on username and password.
- **Parameters**: $username (string): User's username, $password (string): User's password.
- **Returns**: An array with success and message, indicating whether the login was successful.

updateProfile($id, $firstName, $lastName, $username, $email)

- **Description**: Updates a user's profile information.
- **Parameters**:
    - $id (int): User ID.
    - $firstName (string): New first name.
    - $lastName (string): New last name.
    - $username (string): New username.
    - $email (string): New email address.
- **Returns**: true if the update was successful, otherwise false.

userExistsByUsernameEdit($username, $currentId)

- **Description**: Checks if a username is taken by someone other than the current user.
- **Parameters**: $username (string): The username to check, $currentId (int): Current user's ID.
- **Returns**: An array with success and message indicating whether the username is taken.

userExistsByEmailEdit($email, $currentId)

- **Description**: Checks if an email is taken by someone other than the current user.
- **Parameters**: $email (string): The email to check, $currentId (int): Current user's ID.
- **Returns**: An array with success and message indicating whether the email is taken.

deleteUser($id)

- **Description**: Deletes a user based on the user ID.
- **Parameters**: $id (int): The user ID.
- **Returns**: true if the user was successfully deleted, otherwise false.

getCompletedJobsByUserId($user_id)

- **Description**: Retrieves all completed jobs for a specific user.
- **Parameters**: $user_id (int): The user ID.
- **Returns**: An array of completed jobs if found, otherwise an empty array.

getReviewsByJobId($job_id)

- **Description**: Fetches all reviews for a specific job.
- **Parameters**: $job_id (int): The job ID.
- **Returns**: An array of reviews if found, otherwise an empty array.

~ reviews.php

getJobId($id)

- **Description**: Fetches job IDs for jobs posted by a specific user.
- **Parameters**: $id (int): The user ID of the job poster.
- **Returns**: An array of job IDs posted by the user.

getReviewsByJobId($job_id, $user_id = null)

- **Description**: Retrieves reviews for a specific job. Optionally checks if the job was posted by or contracted to the given user.
- **Parameters**: $job_id (int): The job ID, $user_id (int|null): Optional user ID for additional access check.
- **Returns**: An array of review data, including comments, ratings, and reviewer/contractor usernames, or an empty array if no reviews are found.

addJobReview($job_id, $reviewer_id, $contractor_id, $communication, $time_management, $quality, $professionalism, $comments)

- **Description**: Adds a new review for a job.
- **Parameters**:
    - $job_id (int): The job ID being reviewed.
    - $reviewer_id (int): The user ID of the reviewer.
    - $contractor_id (int): The user ID of the contractor being reviewed.
    - $communication (int): Rating for communication.
    - $time_management (int): Rating for time management.
    - $quality (int): Rating for quality of work.
    - $professionalism (int): Rating for professionalism.
    - $comments (string): Review comments.
- **Returns**: true if the review was successfully added, otherwise false.

getReviewById($review_id)

- **Description**: Retrieves a specific review by its ID, including details about the reviewer and contractor.
- **Parameters**: $review_id (int): The review ID.
- **Returns**: An associative array containing review details, reviewer name, and contractor name if found, otherwise false.

updateReview($review_id, $communication, $time_management, $quality, $professionalism, $comments)

- **Description**: Updates the specified review's ratings and comments.
- **Parameters**:

- $review_id (int): The review ID.
- $communication (int): Updated communication rating.
- $time_management (int): Updated time management rating.
- $quality (int): Updated quality rating.
- $professionalism (int): Updated professionalism rating.
- $comments (string): Updated review comments.

- **Returns**: true if the review was successfully updated, otherwise false.

deleteReview($review_id)

- **Description**: Deletes a review based on its ID.
- **Parameters**: $review_id (int): The review ID to delete.
- **Returns**: true if the review was successfully deleted, otherwise false.

~ jobs.php

getAllJobs()

- **Description:** Retrieves all jobs from the Jobs table.
- **Parameters:** None
- **Returns:** An array of job records, each containing details about a job.

getJobsByStatus($user_id, $status)

- **Description:** Retrieves jobs posted by a specific user with a specified status.
- **Parameters:** $user_id (int): The user ID of the job poster, $status (string): The status of the jobs to be fetched (e.g., "complete", "in-progress").
- **Returns:** An array of jobs matching the criteria, including job details and the contractor's name.

getJobById($job_id)

- **Description:** Retrieves a job by its ID, including associated skills.
- **Parameters:** $job_id (int): The job ID.
- **Returns:** An associative array containing job details and associated skills if the job is found, otherwise false.

saveJobWithSkills($jobData, $skills)

- **Description:** Saves a job and its associated skills to the database.
- **Parameters:** $jobData (array): An associative array of job data (e.g., posted_by, title, location), $skills (array): An array of skill IDs associated with the job.
- **Returns:** None, but throws an exception if an error occurs.

getAllSkills()

- **Description:** Retrieves all distinct skills available in the Skills table.
- **Parameters:** None
- **Returns:** An array of skills, each containing skill name and skill ID.

hasUserRequestedJob($user_id, $job_id)

- **Description:** Checks if a user has requested a specific job.
- **Parameters:** $user_id (int): The user ID, $job_id (int): The job ID.
- **Returns:** true if the user has requested the job, otherwise false.

addJobRequest($job_id, $user_id)

- **Description:** Adds a job request for a user, setting the status to "pending."
- **Parameters:** $job_id (int): The job ID, $user_id (int): The user ID of the requester.
- **Returns:** true if the request is successfully added, otherwise false.

getJobRequestsByJobId($job_id)

- **Description:** Retrieves all job requests associated with a specific job, including requester details.
- **Parameters:** $job_id (int): The job ID.
- **Returns:** An array of job requests with details about each request and the requesting user.