Controllers

- getUser()
  - Access UserHandler.getAllUser()
  - Return a response JSON containing information about a user
- getAllUsers()
  - Access UserHandler.getAllUsers()
  - Return a response JSON containing a list of users
- createUser()
  - Access UserHandler.create()
  - Return a response JSON containing newly created user
- deleteUser()
  - Access UserHandler.delete()
  - Return a response JSON containing status
- updateUser()
  - Access UserHandler.delete()
  - Return a response JSON containing status

- getPost()
  - Access PostHandler.getPost()
  - Return a response JSON containing information about a post
- getAllPosts()
  - Access PostHandler.getAllPosts()
  - Return a response JSON containing a list of posts
- createPost()
  - Access PostHandler.create()
  - Return a response JSON containing newly created post
- deletePost()
  - Access PostHandler.delete()
  - Return a response containing status
- updatePost()
  - Access PostHandler.update()
  - Return a response JSON containing status

- getComment()
  - Access CommentHandler.getComment()
  - Return a response JSON containing information about a comment
- getAllComments()

    - Access CommentHandler.getAllComments()
    - Return a response JSON containing information about all comments
- createComment()
    - Access CommentHandler.createComment()
    - Return a response JSON containing newly created content
- deleteComment()
    - Access CommentHandler.deleteComment()
    - Return a response JSON containing status
- updateComment()
    - Access CommentHandler.updateComment()
    - Return a response JSON containing status

- getTag()
    - Access TagHandler.getTag()
    - Return a response JSON containing information about a comment
- getAllTags()
    - Access TagHandler.getAllTags()
    - Return a response JSON containing information about all comments
- createTag()
    - Access TagHandler.createTag()
    - Return a response JSON containing newly created content
- deleteTag()
    - Access TagHandler.deleteTag()
    - Return a response JSON containing status
- updateTag()
    - Access TagHandler.updateTag()
    - Return a response JSON containing status

Handlers

- UserHandler
  - getAllUsers(): Promise<User[]>
    - Return a list of all users
  - getUser(username: string): Promise<User>
    - Return a single user with given username
  - getUserBySearch(searchInput: string): Promise<User[]>
    - Return a list of users with attributes that match searchInput
  - create(username: string, password: string) Promise<User>
    - Create a new user with a username and password
  - updatePassword(username: string, password: string)
    - Set password of a user with given username
  - delete(username: string)
    - Delete a user
- PostHandler
  - getAllPosts(): Promise<Post[]>
    - Return a list of all Posts
  - getPostsByUsername(username: string): Promise<Post[]>
    - Return a list of Posts made by the specified user
  - getPostById(postId: int): Promise<Post>
    - Return a post by ID
  - getPostsByTag(tag: string): Promise<Post[]>
    - Return a list of all posts with the specified tag
  - getPostsBySearch(searchInput: string): Promise<Post[]>
    - Return a list of all posts associated with the search input
  - create(username: string, postId: number, content: string) Promise<Post>
    - Create a new post
  - update(username: string, postId: number, content: string)
    - Update a post
  - delete(username: string, postId: number)
    - Delete a post
- CommentHandler
  - getAllComments(): Promise<Comment[]>
    - Return a list of all comments
  - getCommentsByPost(postId: number): Promise<Comment[]>

        - Return a list of all comments in a post
    - getCommentsByUser(username: string): Promise<Comment[]>
        - Return a list of all comments made by a user
    - getComment(commentId: int): Promise<Comment>
        - Return a comment
    - create(username: string, postId: int,commentId: int, content: string): Promise<Comment>
        - Create a new comment
    - update(commentId: int, content: string)
        - Update a comment
    - delete(commentId: int)
        - Delete a comment
- TagHandler
    - getAllComments(): Promise<Comment[]>
        - Return a list of all comments
    - getCommentsByPost(postId: number): Promise<Comment[]>
        - Return a list of all comments in a post
    - getCommentsByUser(username: string): Promise<Comment[]>
        - Return a list of all comments made by a user
    - getComment(commentId: int): Promise<Comment>
        - Return a comment
    - create(username: string, postId: int,commentId: int, content: string): Promise<Comment>
        - Create a new comment
    - update(commentId: int, content: string)
        - Update a comment
    - delete(commentId: int)
        - Delete a comment

DAO

- **DefaultEntity** {
  id: number;
  deleted: boolean;
  createdAt: Date;
  updatedAt: Date;
  }

- **User** {
  username: string;
  firstName: string;
  lastName: string;
  email: string;
  profilePicture: Image[]
  password: string
  followers: User[];
  following: User[];
  posts: Post[];
  achievements:
  points: number
  upvoted: [];
  playlist: Video[]
  }

- Post {
  userId: number
  content: string
  comments: Comment[]
  upvotes: User[]
  downvotes: User[]
  isPost: boolean
  }

- Tag {
  tagId: number
  tagName: string
  posts: Post[]
  }