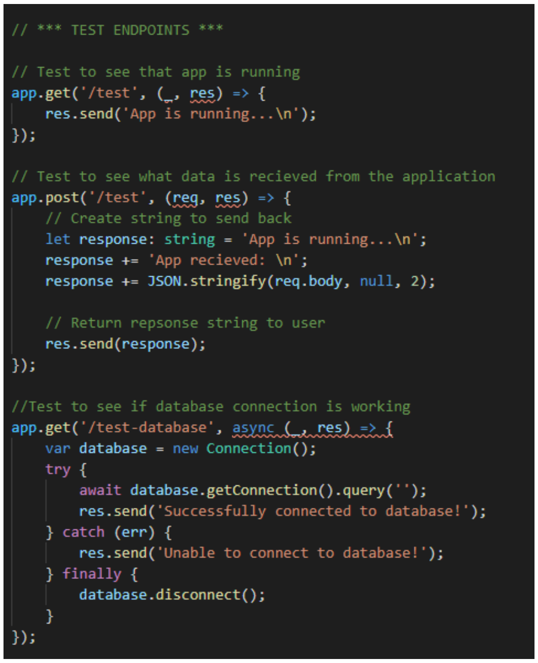
COSC 360 Preliminary Server Document
CoinTalk

The endpoints that are created for the web application are vital to keeping the website organized and up to date. The endpoints are used to access the database using SELECT and also used to update the database such as new accounts, posts, comments etc.

The server.ts file contains all the endpoints and backend functions for retrieving and inserting into the database. The following GET and POST endpoints were created, as long with a few test endpoints:

TEST ENDPOINTS:

We have three test endpoints. The following endpoints are used to test that the app is running, the data is being received from the application, and that database has been connected to successfully.



```
// *** TEST ENDPOINTS ***

// Test to see that app is running
app.get('/test', (_, res) => {
    res.send('App is running...\n');
});

// Test to see what data is recieved from the application
app.post('/test', (req, res) => {
    // Create string to send back
    let response: string = 'App is running...\n';
    response += 'App recieved: \n';
    response += JSON.stringify(req.body, null, 2);

    // Return repsonse string to user
    res.send(response);
});

//Test to see if database connection is working
app.get('/test-database', async (_, res) => {
    var database = new Connection();
    try {
        await database.getConnection().query('');
        res.send('Successfully connected to database!');
    } catch (err) {
        res.send('Unable to connect to database!');
    } finally {
        database.disconnect();
    }
});
```

GET ENDPOINTS:

The next seven endpoints in the server.ts file are GET endpoints. These endpoints are used to retrieve data from the database and send it to the application.

Get Image endpoint:

This endpoint retrieves the following image based on the id that is passed through the URL parameter. This endpoint will send the file where the file is located in: uploads/'filepath'

Get Posts Endpoint:

This endpoint is to display the posts on the front page. This will select all the posts that are related to the category of the coin that the user selected, the title of the post matching the search query. This endpoint also displays the page's limit to allow easier viewability to the user.

Get Comments Endpoint:

This endpoint will select the comments from the parentId. The parentID is the post that the comments are child's of. The comments display the username and content of the comment.

Get Post Endpoint:

This will display the actual post when the user clicks on one of the posts on the homepage. This will display all the information related to the post including it's comments.

Get User Endpoint:

This will retrieve the user based on their username. This will display all the information related to the user such as username, uid, email, etc.

Get User's Post Endpoint:

This endpoint will retrieve all the posts that are posted by the given username. This query will retrieve all the posts, where each post will also hold its children comments.

Get User's Comments Endpoint

This endpoint will retrieve all the comments that the user has posted. Each comment will bring you to its parent post.

Get Logged in User Account

This endpoint will select username, uid, email, admin, and the profile image related to the currently logged in user. This user has it’s session id stored in the browser to allow the user to stay logged in throughout page navigation.

POST ENDPOINTS

Post Login Endpoint:

This endpoint is when a user logs into the web application. This will retrieve all the information related to the user. If the user is banned, the user is unable to log into the website. If the user enters invalid credentials such as a username, or password that is incorrect, the website will display such information.

Post Signup Endpoint:

This endpoint is to allow the user to signup and created an account on the web page. The following inputs must reach the following signup parameters:

- Username is required
- Email is required
- Password is required
- Username length must be greater than 5 and less than 20.
- Email length must be greater than 5 and less than 50
- Password length must be greater than 6 and less than 30.

If all those parameters are met, this post will insert all that information into the database, along with the current time of signup and a default banned value of FALSE.
If the user creates an account that already exists, such as an already existing username or email the website will display an error.

Check Logged-in Endpoint

This is a simple endpoint that checks if the current user is logged in. This is to allow greater site functionality to the users who are logged in.

Log-out endpoint:

This is a simple endpoint that will log the user out if the user clicks the logout button.

Create comment endpoint:

This will allow the user to create a comment to the post. The user must be logged in, creating a comment for a valid post id, and have content to the comment.

If those parameters are met, the comment is added to the comment table in the database with the following parentID which is the post that the comment is created on.

Create Post endpoint.

This will allow the user to create a post. This endpoint will require a title, post type (image or text), content, and which coin the post relates to. If it is an image post it will do a different query than if it was a text post. The user must be logged into to make a post.

Like Post Endpoint

This endpoint will allow the user to like a post. The user must be logged into to use this feature. This first checks to see if the user liked the post or not. If the user already liked the post, liking it again will remove their like from the post. If the post is not currently liked by the user, the post will be liked and added to the corresponding table.

Check-Like Endpoint.

This is a helper endpoint for the Like Post Endpoint. This one checks if the user has already liked a post or not.

Post-Like Endpoint

This endpoint will retrieve the current number of likes that the post has.

Ban-User endpoint

This is an admin only function. The user must be logged in and be an admin of the webpage. If these parameters are met, the admin can ban a user based on their id. The resulting ban will make it so the banned user can no longer login to the website.

Remove Post endpoint:

This endpoint is to remove a post from the page. For this endpoint to be used by the user, they must be logged into the account that created the post. If these conditions are not met it will Much like the previous endpoint, this will remove the comment related to the parent post where the comment was created by the currently logged-in account. If these are met, the user can delete the comment from the post.

Edit Post Endpoint:

This will allow the user to edit an already-created post. This will update the text value in the Post table. To use this endpoint the user must be logged in.

Edit Comment Endpoint:

Much like the previous endpoint, this will allow the user to edit a comment that is made on a post. The user must be logged in and be the creator of that comment to be allowed to edit a comment.

isPostOwner Endpoint:

This is a helper endpoint which checks to see if the user is the owner of a post to make any alterations to it.

isCommentOwner Endpoint:

Much like the previous endpoint, this is a helper endpoint to see if the current signed in user is the owner of the comment to make any changes to the comment.

isAdmin Endpoint:

This is a helper endpoint which tests if the currently signed in user is an admin of the webpage or not. If they are an admin, they will be granted additional privileges.

UpdateLikeCounter Endpoint:

This is a helper endpoint to the Post-Like endpoint. This endpoint will update the like counter on the post when the user clicks on the star.