Project Part 3

You will build the functionality of your app in this part of the project. Your team should discuss what you want to build for your project. You must choose between one of the app categories below, though there is much room for customization and design within each of these categories. You can also go beyond the minimum requirements if your team would like to see their idea fully realized.

Game

Make a multiplayer game with lobby support. You have full freedom in the design of the game and the game itself can be very simple (In the past, teams have made simple 2-player games) or very complex. The game can run entirely in the browser via JavaScript while sending/receiving updates for the game state via WebSockets.

Auction House

Make an auction house where users can put items up for auction and other users can bid on those items. At the end of the auction, the highest bidder wins the item.

Quiz App (TopHat)

Make a question site where users can post questions for all other users to answer. Users will have a fixed time limit to answer each question and their scores will be tracked in a gradebook.

Objective 1: Custom Posts (With Images)

Customize your posts to be specific to your app. In all cases, the username of the poster must still be displayed on the post. You may modify the like feature to become one of the new required features (ie. The like feature is no longer required). The requirements are different for each app category:

Game

Each authenticated user can create a game lobby that includes:

- A title for the lobby
- A description of the lobby
- An optional (for the user) image for the lobby (Lobbies should still work if the user chooses not to upload an image) [To be clear, your app **must** support image uploads.

The option is for the end user to decide whether or not they want to upload an image for their lobby]

Auction House

Each authenticated user can create an auction with:

- The title which is the name of the item being auctioned
- The description of the item
- An image of the item being auctioned
- The starting price of the auction
- The duration of the auction, or the auction end time

Quiz App (TopHat)

Each post is a quiz question. You have much freedom in how you design the question format (eg. Multiple choice, numeric answer, etc). You only need to support 1 question format.

Each authenticated user can create a quiz question with:

- The title of the question
- The description of the question (The question itself)
- An optional (for the user) image for the question (Questions should still work if the user chooses not to upload an image) [To be clear, your app **must** support image uploads. The option is for the end user to decide whether or not they want to upload an image for their question]
- The answer method (eg. If it's multiple choice, this would be the choices with radio buttons; If it's an open-ended question, this would be a text box)
- When creating the question, the user should be able to specify the correct answer(s) depending on your grading method

Security: You must escape any HTML for all user-submitted text that will be displayed to other users
Security: You must protect against a user accessing arbitrary files from your server
Security: If you are using a SQL database, you must protect against SQL injection attacks

Testing Procedure

1. Start your server with docker compose up
2. Open a browser two browsers (Chrome and Firefox) and navigate to http://localhost:8080/
3. If a post for is visible, attempt to create a post

    a. Verify that you are unable to create a post (unauthenticated)
4. Register and login with both browsers
5. In the first browser, create a post containing all possible information (Including an image)
    a. Navigate back to the home page (If you were navigated away) and verify that you can see the post with all expected content, including your username
    b. Refresh the page in the second browser and verify that you can see the post as expected
6. Create a post in the second browser with all available information
    a. Navigate to/refresh the page in both browsers and verify that the post appears for both users as expected
7. In either browser, submit a third post. If the image is optional, do not include an image
    a. Navigate to/refresh the page in both browsers and verify that all three posts appear as expected for both users
8. Restart the server with docker compose restart
9. Refresh the page on both browsers and verify that all 3 posts remain after the restart
10. Check for relevant **security** vulnerabilities

Objective 2: WebSocket Interactions

Note: You are required to use WebSockets for these features! Resorting to polling or long-polling is not allowed (Please note that the SocketIO library often resorts to long-polling. You must disable this and force it to use WebSockets if you are using this library)

Add WebSocket interactions to your app. The requirements vary depending on the category of your app. All of the following features are required to use WebSocket interactions to create a live feel for your app:

Game

- Users are able to join existing lobbies
- Users can only join 1 lobby at a time
- Each lobby displays the current number of players in the lobby
- When a lobby is full, the game starts for all users in that lobby (The game itself is built in the next objective)
- Users cannot join full lobbies
- The user who created the lobby should be automatically added to the lobby (You should decide how you want to handle the case where the creator is already in another lobby - Should they be removed from that lobby or not be allowed to create their own)

Auction House

- The current bid and time remaining are displayed on every auction (The time must be accurate and sent from the server)
- Users can enter a bid for any item up for auction as long as there is time remaining

- Bid that are <= the current bid have no effect
- At the end of the auction, no more bids are accepted and the user with the highest bid wins the item (Winning is handled in the next objective)
- The creator of an auction cannot bid on their own auction

Quiz App (TopHat)

- The time remaining on each question is displayed (The time must be accurate and sent from the server)
- Users can submit their answer for each question
- Users cannot answer a question more than once
- When the timer for a question ends, the question is graded (Grading is handled in the next objective)
- Users cannot submit answers after the time for a question ends
- The creator of a question cannot answer their own question

Note: You will have to authenticate the WebSocket connections. This can take some work depending on the library you use as some of them make it difficult to access the HTTP upgrade request that contains your auth token. Once a connection is authenticated, you can treat all WS frames over that connection as authenticated.

Testing Procedure

1. Start your server using docker compose up
2. Open the app with at least 3 tabs including both Firefox and Chrome and navigate to http://localhost:8080/
3. Register and login on all three tabs
4. Create at least 3 posts with at least 2 posts made by the same account
5. Interact with the posts with all three accounts and check for all expected behavior
    a. Include cases where the creator interacts with their own post (Or creates a lobby while already in a lobby for the game)
    b. Include cases where a user interacts with a post multiple times (joining 2 lobbies or answering the same question twice)
6. Ensure that posts "end" as expected:
    a. **Game**: *Something* should happen and users can no longer join the lobby. If the game is built, it will start. If it isn't, this objective is still complete as long as anything happens for all users in the lobby to indicate that this is when the game would start if it existed. This can be as simple as a JS alert
    b. **Auction**: Users can no longer place bids after time expires
    c. **Quiz**: Users can no longer answer questions after time expires
7. Check for relevant **security** vulnerabilities

Objective 3: App Specific Features

Finish the core features of your app.

Game

Build the actual game. When a lobby is full, all players in the lobby will start an instance of your game.

- Multiple games must be able to played at the same time by different users
- Interactions and updates in the game must be sent over WebSockets so all players can see the current/live state of the game
- You have full freedom to build whatever game you'd like (As long as your team builds the game!)

Auction House

Track the winners of each auction.

- When an auction ends, your server will track the winner of that auction
- When viewing the home page, each auction that has ended must specify the winner of the auction and the winning bid
- Users have a way to see all the auctions they've **won** without scrolling through every auction (This can be a separate page where only the auctions they've won are displayed)
- Users have a way to see all the auctions they've **created** without scrolling through every auction (This can be a separate page where only the auctions they've created are displayed)

Quiz App (TopHat)

When a questions ends, it must be autograded and the grades must be tracked by your server

- When a question ends, each submission is graded based on the correct answer provided when the question was created
- Grades are recorded and securely stored on your server
- Each user has a way to view their grades for all questions they've answered (eg. Their own grades)
- Each user has a way to view all the grades for questions they've created (eg. The gradebook for their questions)
- No users can view grades, by any means, that are not their own unless it's for a question they created

Testing Procedure

1. Start your server using docker compose up
2. Open the app with at least 3 tabs including both Firefox and Chrome and navigate to http://localhost:8080/
3. Register and login on all three tabs
4. Create posts and interact with them until several posts "end"
5. Verify that the expected behavior occurs
    a. **Game**: The game begins and all players from the lobby can interact with each other in the game in real time. If the game is turn-based, the other player(s) can immediately see when an action is taken by the active player
    b. **Auction**: Winners and winning bids are displayed for all users on the home page. The home page has a clear way to find only the auctions you've won and a clear way to see only the auctions you've posted
    c. **Quiz**: The home page has a clear way to view your grades for all questions you've answers and a clear way to view the gradebook for all questions you've posted
6. Check for relevant **security** vulnerabilities

"A clear way" can mean a link that takes you to a separate page or any other means that provides the information in a way that a typical user can clearly understand.

Submission

All of your project files must be in your GitHub repo at the time of the deadline. Please remember to include:

- A docker-compose file in the root directory that exposes your app on port 8080
- All of the static files you need to serve in the public directory (HTML/CSS/JavaScript/images)

It is **strongly** recommended that you download and test your submission. To do this, clone your repo, enter the directory where the project was cloned, run docker compose up, then navigate to localhost:8080 in your browser. This simulates exactly what the TAs will do during grading.

If you have any Docker or docker compose issues during grading, your grade for each objective may be limited to a 1/3.

Team Scoring

Each objective will be scored on a 0-3 scale as follows:

3	Clearly correct. Following the testing procedure results in all expected behavior
2	Mostly correct, but with some minor issues. Following the testing procedure does not give the exact expected results
1	Clearly incorrect, but an honest attempt was made to complete the objective. Following the testing procedure gives clearly incorrect results, or no results at all. This includes issues running Docker or docker-compose even if the code for the objective is correct
0	No attempt to complete the objective or violation of the assignment (Ex. Not using a framework) -or- a security risk was found while testing the objective

3	2 Application Objectives
2	1 Application Objective
1	0 Application Objectives
0	0 Application Objectives

Please note that there is only one chance to earn these application objectives. There will not be a second deadline for project parts 2-4.

Individual Grading

The grading above will be used to determine your team score which is based on the functionality of your project. Your actual grade may be adjusted based on your individual contributions to the project. These decisions will be made on a case-by-case basis at the discretion of the course staff. Factors used to determine these adjustments include:

- Your meeting form submissions: team meeting and eval form
    - You must fill out this form after every meeting. Failure to do so is an easy way to earn a negative individual grade adjustment
    - The quality of your submissions will be taken into account as well (eg. Saying "I'll do stuff" before the next meeting is a low quality submission)
    - You may submit more meeting forms than are required even if there was no meeting (eg. If you want to adjust your evals after a deadline without waiting for the next meeting, or if you attempted to schedule a meeting and your teammates refused)

- Your evaluations from the meeting form
    - If you teammates rated you poorly/excellently, your grade may be adjusted down/up respectively (pending a manual review by the course staff to verify the claims)
- Your commits in the team repo
    - Your commits may be checked to see if you did in fact complete the work you mentioned in the meeting form, as well as compare the amount of work you completed to that of your teammates
    - You **MUST** commit your own code! It is not acceptable for your teammate to commit your code for you. You should have a clear separation between your tasks and commit the code for your task. If a commit is not in your name, you effectively did not write that code. If a teammate is making this difficult, let the course staff know in the meeting form