Champlain College - Lennoxville
Project: Implementation

PROGRAM:	420.B0 Computer Science Technology	
COURSE:	Transactional Web Applications 2	
COURSE CODE:	420-530-LE	
WEIGHT:	10% of the final score (10% out of 15% of the project, part 1)	
SEMESTER:	Fall 2023	
INSTRUCTOR:	Francis Gauthier fgauthier@crcmail.net	Office C-239

Multiplayer Quiz App

The goal of this project is to create a multiplayer quiz app that users can play simultaneously. The project is a web-based application, that must be written with JavaScript frameworks and use WebSocket to provide real-time features.

Design Requirements

Teacher point-of-view

Creation of a new room:

- Must provide the ability to create new quizzes of different categories
- Must be able to specify the timer setting for each quiz (reflection time allowed for each question). **Each question will have the same time.**
- Must be able to select the difficulty level
- Must be able to select the number of questions
- A unique alphanumeric code must be provided to share with users on quiz creation

Monitoring the quiz:

- Must be able to see the quiz progress, but cannot interfere once the quiz is started

After the quiz:

- Must be able to see the scores of all participants and the winner

Player point-of-view

Joining a new room:

- Players must join by entering the quiz code

- Players must enter their name

While the quiz is running:

- Multiple users must be able to simultaneous participate of in a single quiz session
- Real-time synchronization of questions and answers across all players' screens
- Interactive countdown timer for each question
- Intuitive user interface for answering questions
- Results are shown after each question:
    - Correct answer
    - Current player score

After the quiz:

- Must be able to see the scores of all participants and the winner

Scope

Here are features that are **not required:**

- Persistent leaderboard. The quiz results are shown once and are lost after each quiz is finished.
- Ability to interrupt a launched quiz.

Evaluation breakdown

Criteria	Weight
Use cases	(70% total)
1. Host can create a new quiz	10%
2. Host can wait for incoming players and start a quiz	10%
3. Host can monitor a quiz session and see the results	10%
4. Player can join a quiz waiting room	10%
5. Player can play a quiz and answer questions	20%
6. Player can see the results and leaderboard	10%
User interface and user experience	(30% total)
1. Navigation	10%
a. Easy and intuitive navigation between different views	
b. Users can navigate efficiently using buttons and links	
2. User interface and user experience	20%
a. Efficient use of input controls	
b. Intuitive design of the application	
c. Users are well guided in the actions that they must take with labels and instructions when needed	
d. Consistent design and color theme across views	

Use case detailed

Host can create a new quiz:

- Host can navigate to a page where they must set the parameters of the quiz:
    - Category
    - Number of questions
    - Time for each question
    - Difficulty (easy, medium, hard)

Host can wait for incoming players

- Host receives a unique alphanumeric code for the room
- Host has a waiting page and can monitor the number of participants joining the room
- Host can start the quiz by pressing a button

Host can monitor a quiz session and see the results:

- Host can see the active question during the quiz
- Host can see how many players have answered
- When the quiz in finished, the host can see the final player scores (leaderboard)

Player can join a quiz session:

- Players can navigate to the “join quiz” view
- Players join by entering a username and room code
- Players are notified if the room code is not valid
- Players are shown the waiting lobby when they join successfully
- Players must wait for the game to start

Players can play a quiz and answer questions

- Players can see the active question during the quiz. Questions arrive simultaneously across connected clients.
- Players can see the possible answers during the quiz
- Players can click on a button to answer the question
- Players have an indication of time left to answer
- Players can see the correct answer after the time runs out. Answers arrive simultaneously across connected clients.

Players can see the results and leaderboard

- Players are redirected to the final view after the last question
- Players can see their final score
- Players can see the top scores

Submission

When your work is done, push your code to a GitHub repository and share that repository with me.

Github username: **frangauthier**

1. Submission and last commit should be made before **Wednesday September 20th, End of Day** to get no grade penalty.
2. A second submission deadline is available with a 10% penalty if made before **Sunday September 24th End of Day**.

Use case grading

For each use case, this marking grid will be used to determine the overall success of the use case.

Flawless (100%) The use case can be completed easily and without issue The design is intuitive, and the user is provided with instructions when needed The quiz interface provides much information about the current situation of the game (overall progress of questions, timer, name of players, player in the lead etc.) Websockets events are used properly
Excellent (90%) The use case can be completed with no issue or only a minor issue The design is efficient, and users are provided with instructions when needed The quiz interface provides much information about the current situation of the game (overall progress of questions, timer, name of players, player in the lead etc.) Minor aesthetic issue can be found, but nothing major Websockets events are used properly
Great (80%) The use case can be completed with only a minor issue The design is efficient The quiz interface provides some information about the current situation of the game (overall progress of questions, timer, name of players, player in the lead etc.) Minor issue impedes the completion of the use case, but can be fixed with few lines of code Websockets events are used properly most of the time
Good (70%) The use case can be completed with a minor or major issue The design is sufficient The quiz interface provides some information about the current situation of the game (overall progress of questions, timer, name of players, player in the lead etc.) Major issue impedes the completion of the use case, but can be fixed with few lines of code Websockets events are used properly most of the time
Decent (60%) The use case can be completed with difficulty or cannot be completed

The design is sufficient The quiz interface provides few or no information about the current situation of the game (overall progress of questions, timer, name of players, player in the lead etc.) Major issue impedes the completion of the use case, but can be fixed with few lines of code Some websockets events are used properly
Problematic (50% -) The use case cannot be completed The design is insufficient, unintuitive, or difficult to figure out The quiz interface provides few or no information about the current situation of the game (overall progress of questions, timer, name of players, player in the lead etc.) Major issue impedes the completion of the use case and cannot be fixed with few lines of code Websockets events are not used properly