Project Report

TITLE - *Client-Server Quiz Application for Multi-Participant Engagement*

SUBMITTED TO -

1. DR. PARMEET KAUR SODHI
2. DR. JAGRITI

TEAM MEMBERS -

AIBAD KHAN	(21103218)	B11 Batch
CHIRANSHU AGRAWAL	(21103220)	B11 Batch
VATSAL AGARWAL	(21103221)	B11 Batch
MOHIT GARG	(21103229)	B11 Batch

ACKNOWLEDGEMENT

The completion of this project owes a debt of gratitude to several individuals who have contributed their time, expertise, and support.

First and foremost, we extend our sincere appreciation to our esteemed guides, **Dr. Parmeet Kaur Sodhi and Dr. Jagriti ,** whose invaluable guidance and unwavering encouragement propelled us forward in our journey. Their mentorship, insights, and willingness to address our queries have been instrumental in shaping the direction of this project.

We would also like to express our heartfelt thanks to our friends who generously offered their assistance and support, aiding us in navigating through challenges and ensuring the timely completion of this assignment.

Their support and guidance have been indispensable, and we are deeply grateful for their contributions to the success of this endeavor.

INDEX

Introduction & Problem Statement

In today's digital age, interactive platforms for engaging activities such as quizzes play a crucial role in fostering learning, entertainment, and social interaction. Traditional quiz competitions often face limitations in scalability, interactivity, and efficiency due to manual management and limited participation. Hosting quizzes with multiple participants requires an automated system to distribute questions, collect answers, and calculate scores accurately and efficiently. This project aims to address these challenges by developing a Client-Server Quiz Application. The application will facilitate multi-participant quiz competitions, allowing a server to connect with multiple clients, distribute questions, collect answers, calculate scores, and declare winners. By utilizing a client-server architecture and integrating with Excel sheet data, this application offers an efficient and scalable solution for hosting engaging quiz events, overcoming the limitations of traditional quiz formats.

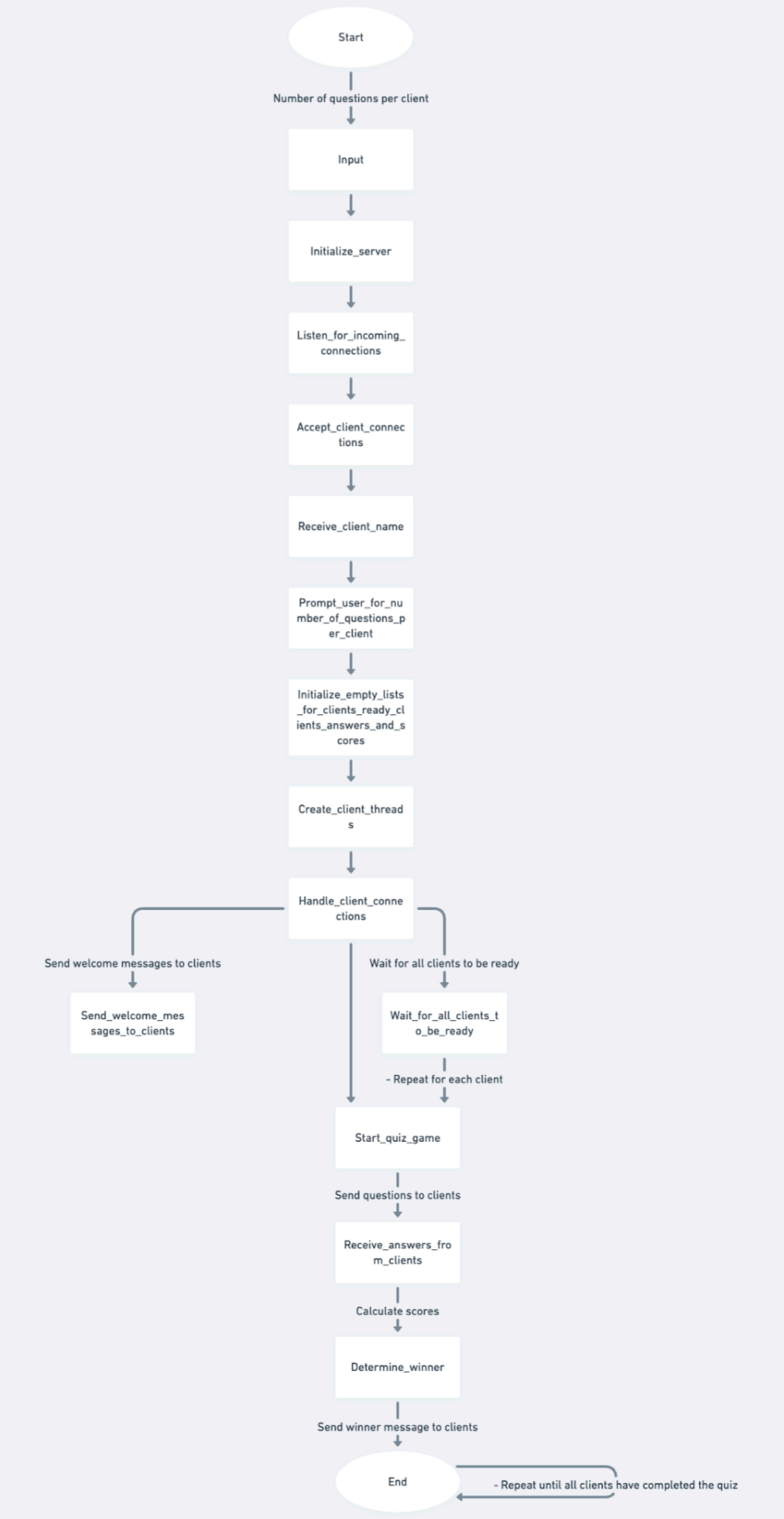
Objective

Develop a client-server architecture for facilitating quiz competitions.
Integrate with an Excel sheet to store and retrieve question data.
Implement a scoring system based on correctness and response time.
Provide real-time updates on scores and leaderboard standings.
Enhance interactivity and engagement for participants through dynamic question distribution and instant feedback.

Technologies Used:

- Python programming language for its simplicity and versatility.
- Socket programming for establishing network connections and communication.
- Pandas library for reading quiz questions from an Excel file.
- Threading module for handling multiple client connections concurrently.

Flow Diagram



System Architecture:

The system comprises a central server and multiple clients connected over a TCP/IP network. Clients establish connections with the server using sockets and communicate via predefined protocols. The server manages the quiz session, generates questions, distributes them to clients, and receives answers for scoring.

Implementation Details:

Server (server.py):

- Loads quiz questions from an Excel file using the Pandas library for flexibility in managing questions.
- Listens for incoming connections from clients and assigns each client a unique identifier.
- Allows the user to specify the number of questions per client for customization.
- Utilizes threading to handle concurrent connections, ensuring responsiveness and scalability.
- Sends questions to clients, receives their answers, calculates scores, and announces the winner.

Client (client.py):

- Connects to the server using sockets and provides a unique identifier for identification.
- Receives initial welcome messages and instructions from the server upon connection.
- Displays questions received from the server and prompts the user for answers.
- Sends user responses back to the server for evaluation and scoring.
- Receives feedback on the correctness of answers and current score from the server.

6. Workflow:

- The server initializes and listens for incoming connections from clients.
- Clients connect to the server, provide their identifiers, and await further instructions.
- Once all clients are connected and ready, the server initiates the quiz by sending questions to each client.
- Clients respond to the questions, and their answers are sent back to the server for evaluation.
- The server computes scores based on the responses received and determines the winner.

Score Calculation Method:

- Scores were calculated based on a predetermined criteria:
- Each correct answer earned the player a certain number of points determined by the weightage of the question.
- Points were adjusted based on the time taken to answer, rewarding faster responses with higher scores.
- The formula used for score calculation:

Score = Weightage × 1/ Time taken

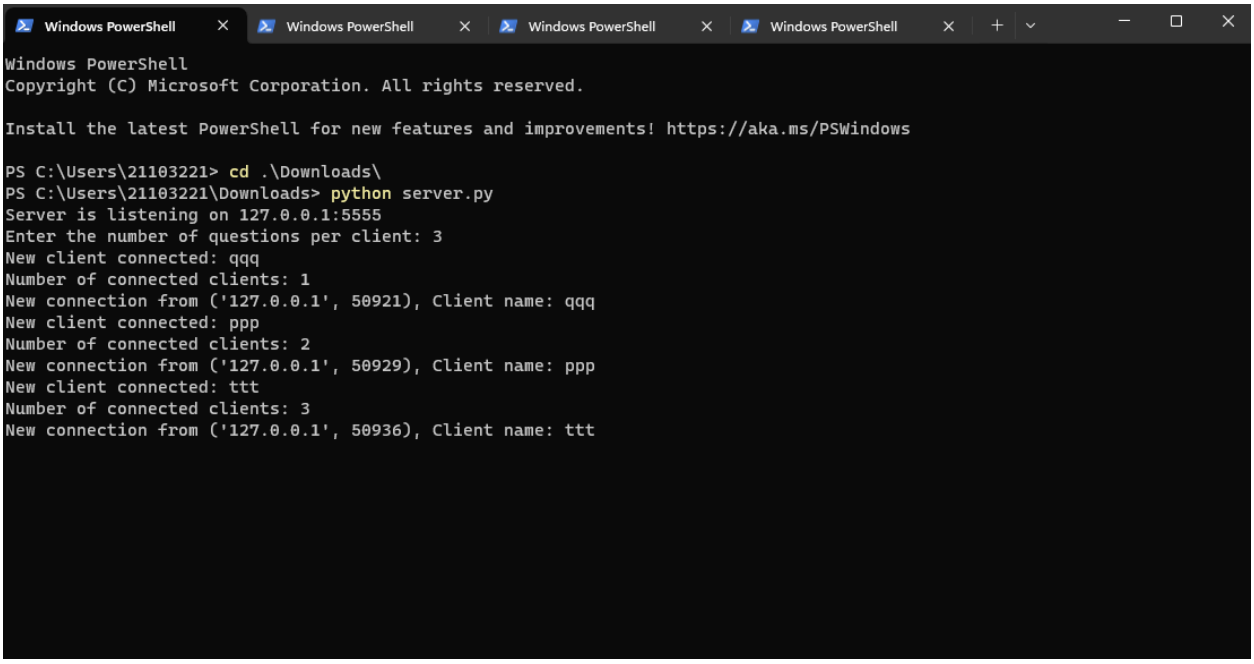
.

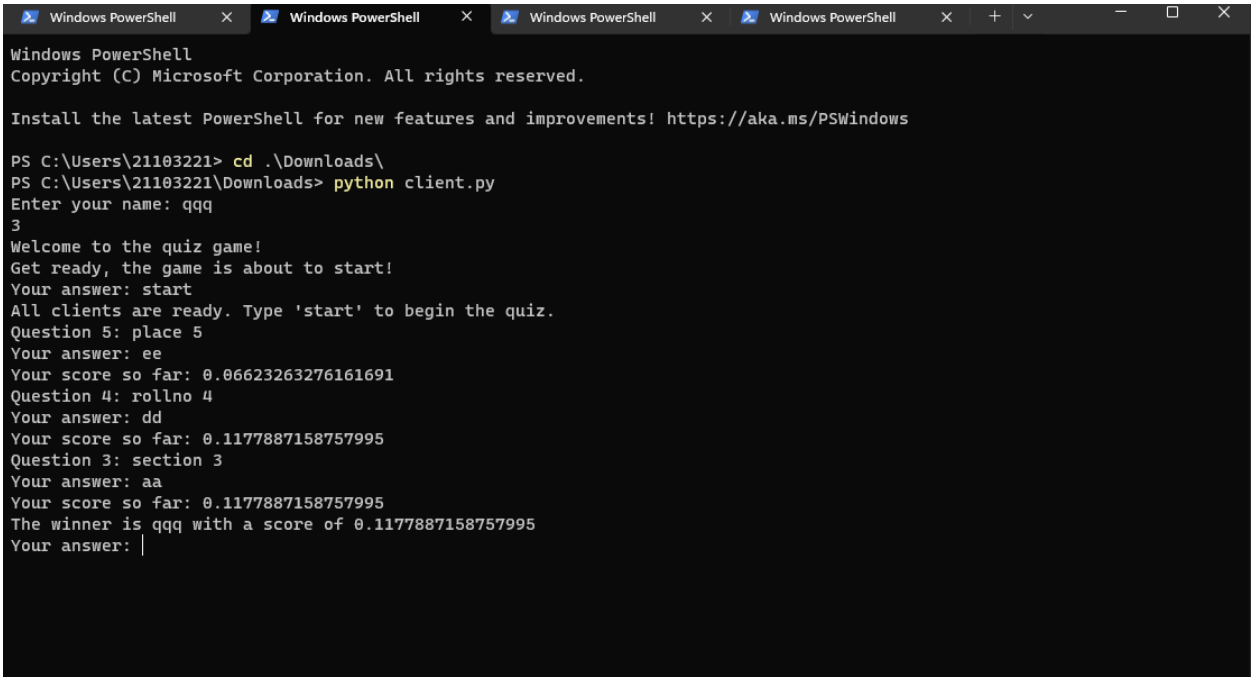
Results & Conclusion

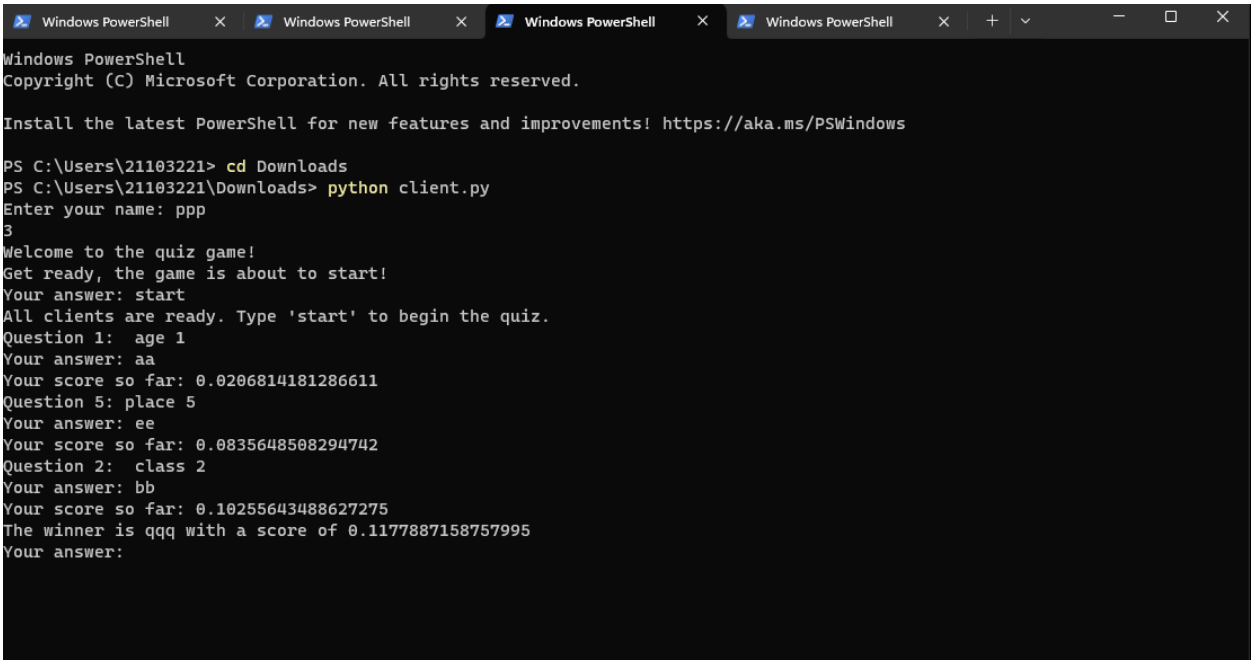
The developed Client-Server Quiz Application successfully facilitates multi-participant quiz competitions with real-time scoring and leaderboard updates. Participants can connect to the server, receive questions, submit answers, and view their scores and rankings on the leaderboard. Integration with an Excel sheet allows for easy management of question data, enabling customization and scalability. The application demonstrates improved interactivity and engagement compared to traditional quiz formats, leading to a more enjoyable and competitive quiz experience.

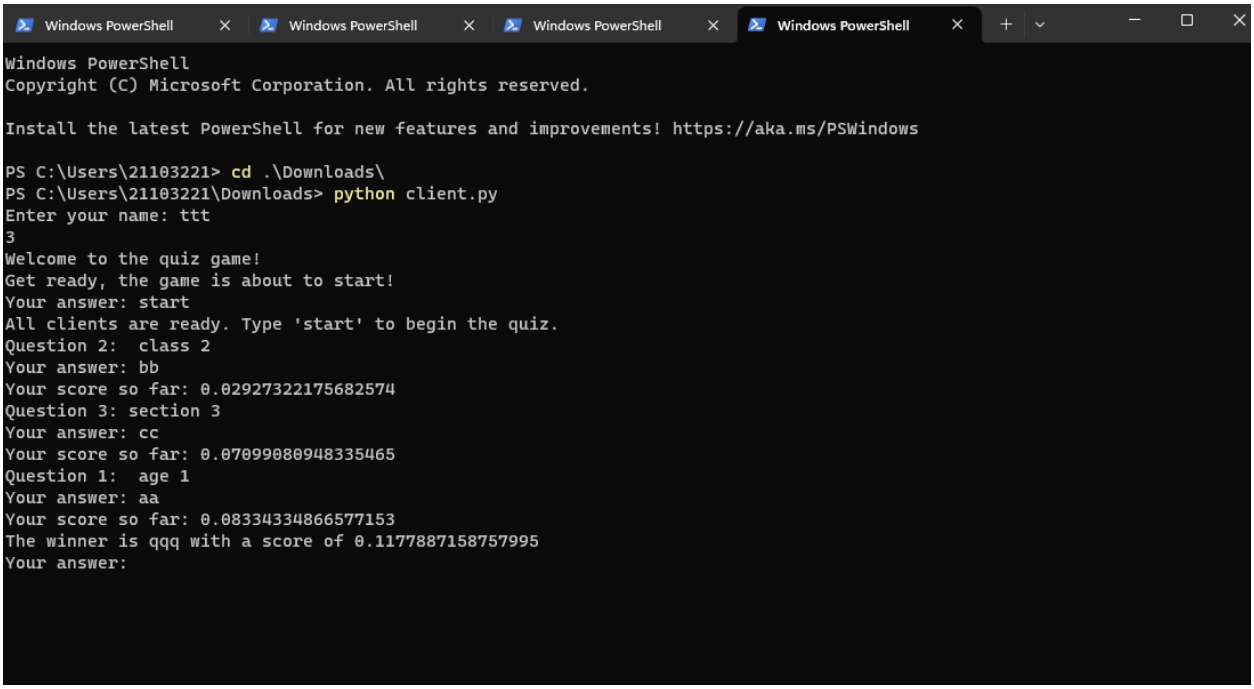
The Client-Server Quiz Application provides an efficient and scalable solution for hosting engaging quiz competitions with multiple participants. By leveraging a client-server architecture and integrating with external data sources, the application enhances interactivity, efficiency, and customization capabilities. With real-time scoring and leaderboard updates, participants can enjoy a dynamic and competitive quiz experience. Overall, the project demonstrates the effectiveness of modern technology in enhancing traditional activities like quizzes, paving the way for innovative solutions in interactive entertainment and education.

Output ScreenShots






Windows PowerShell
Windows PowerShell
Windows PowerShell
Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\21103221> cd Downloads
PS C:\Users\21103221\Downloads> python client.py
Enter your name: ppp
3
Welcome to the quiz game!
Get ready, the game is about to start!
Your answer: start
All clients are ready. Type 'start' to begin the quiz.
Question 1: age 1
Your answer: aa
Your score so far: 0.0206814181286611
Question 5: place 5
Your answer: ee
Your score so far: 0.0835648508294742
Question 2: class 2
Your answer: bb
Your score so far: 0.10255643488627275
The winner is qqq with a score of 0.1177887158757995
Your answer:


Windows PowerShell
Windows PowerShell
Windows PowerShell
Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\21103221> cd .\Downloads\
PS C:\Users\21103221\Downloads> python client.py
Enter your name: ttt
3
Welcome to the quiz game!
Get ready, the game is about to start!
Your answer: start
All clients are ready. Type 'start' to begin the quiz.
Question 2: class 2
Your answer: bb
Your score so far: 0.02927322175682574
Question 3: section 3
Your answer: cc
Your score so far: 0.07099080948335465
Question 1: age 1
Your answer: aa
Your score so far: 0.08334334866577153
The winner is qqq with a score of 0.1177887158757995
Your answer:

References

- [1] Python Software Foundation. (2024). Python 3.10.2 Documentation - Socket. [Online]. Available: https://docs.python.org/3/library/socket.html
- [2] McKinney, W. (2024). Pandas Documentation. [Online]. Available: https://pandas.pydata.org/docs/
- [3] Stevens, W. Richard, Bill Fenner, and Andrew M. Rudoff. "Unix Network Programming: The Sockets Networking API." Addison-Wesley Professional, 2003.
- [4] Forouzan, Behrouz A. "Data Communications and Networking." McGraw-Hill Education, 2017.