An Efficient Synchronization Model in Random Word Educational Games with Multiplayer Platforms

I Made Sukarsa [1,]*, I Putu Agung Bayupati [1], Gede Ocha Dipa Ananda [1], Putu Wira Buana [1],
and Ni Wayan Wisswani [2]

[1] Department of Information Technology, Udayana University, Jimbaran, Indonesia;
Email: bayupati@unud.ac.id (I.P.A.B.), dipa.ananda@student.unud.ac.id (G.O.D.A.), wbhuana@gmail.com (P.W.B.)
[2] Department of Informatic Management, Bali State Polytechnic, Jimbaran, Bali, Indonesia
*Correspondence: sukarsa@unud.ac.id (I.M.S.)

***Abstract*—The availability of fun learning media is an essential aspect of learning a language. One of the educational methods that can be used is education through multiplayer-based interactive games. Its implementation requires real-time communication to support rapid information changes that require large synchronization activities. Thus, it requires high costs in its implementation, especially server needs and internet capacity on servers, likewise, in the implementation of the multiplayer game with WebSocket. Thus, an efficient communication and data synchronization design model is required so that it does not require high costs in its implementation. The method applied to the design was a special synchronization model between the local database and the server combined with socket technology. The design worked on in this study provided performance savings of 39.97% to 75.47% compared to conventional implementation models and without causing any disturbances related to the need for real-time aspects of data synchronization.**

***Keywords*—real time, efficiency, game mobile, WebSocket, multiplayer**

I. Introduction

Language is one of the important requirements for communicating between one people to another. Language is an arbitrary sound symbol system used by social group members to work together, communicate and identify themselves [1]. The society of every country has its own culture. Indonesia has various cultural diversity and is rich in languages. Indonesia has 668 regional languages verified and up-to-date according to dialectology [2]. English is also the most popular language because it is used in many countries worldwide. The number of languages that must be learned and the vocabulary of each language make people need appropriate, fast, and fun educational media.

Problems regarding the need for a language education learning media that is appropriate, fast, and fun is a problem in the digital world. One of the fun educational methods is through interactive games. Interactive games make students more productive by increasing student interest and motivation in obtaining better results and growing confidence in communicating [3]. Good learning is learning not only from knowledge but also from experience [4]. The implementation gamification is one of the methods that can fulfill this need.

Another problem is the development of multiplayer games, which require high costs in development and cloud service costs. The need for synchronization between players in multiplayer is one of the reasons for the large cost requirements.

Research conducted by developing a WebSocket-based Random Word Multiplayer Mobile Game Application tries to solve this problem. The random word mobile game application is developed on the smartphone platform assisted by the WebSocket protocol. The smartphone platform is the right platform for developing an interactive game because the smartphone platform is the most widely owned by the public. According to the Ministry of Communication and Information Technology, smartphone users in Indonesia reach 167 million people or 89% of the total population [5].

The multiplayer concept requires a suitable synchronization model, so it does not require high costs in its implementation. A design model is needed that enables real-time communication with low communication costs.

Developers of interactive games with the multiplayer concept require real-time communication to deal with rapid changes in information. Each player will need the latest information for each player. WebSocket is a purpose-triggered full-duplex asynchronous communication channel to transmit data [6]. WebSocket makes it possible to exchange information in real-time, so it does not require data synchronization in a fixed time. The synchronization model using WebSocket can be a solution in increasing the efficiency of data synchronization compared to conventional implementation models without causing any disturbances related to the need for real-time aspects of data synchronization.

Manuscript received February 3, 2023; revised April 18, 2023; accepted May 10, 2023; published September 6, 2023.

doi: 10.12720/jait.14.5.897-906

The research results are to develop a mobile-based multiplayer game with WebSocket that raises random case studies of words from four languages: Indonesian, Balinese, English, and Javanese. The word that becomes the knowledge base will be randomized letter-by-letter, and the user must guess the vocabulary. Players will receive the results and meaning of the vocabulary questions that have been answered at the end of the game. In addition to playing the game independently, players can invite other players into their own game rooms. Thus, players can compete with other players in solving scramble word problems

II. Literature Review

Research on the development of web-based multiplayer games with WebSocket with open-source WebSocket, namely Socket.io, has been implemented for the Big Two of card games [7]. The results showed that the game prototype had good quality. The game could set the display of four different players, manage players, and save game data. The drawback of this research was that there was no data synchronization performance test.

Research on the development of online games using Socket.io and NodeJS was implemented by designing and implementing socket programming in the form of a web-based combat aircraft online game application [8]. This research ascertained the value of socket and socket programming in real-time two-way communication (client-server) on computer networks. The drawback of this research was that there was no data synchronization performance test yet.

Research on snakes and ladders multiplayer educational games based on Hash Map and vector data structures has been carried out to build snakes and ladders educational Multiplayer Games with arithmetic learning content [9]. The game was built using Java socket programming and a Local Area Network (LAN) as a medium for data communication between players. The drawback of this research was that it still used a local area network to connect between players. Another drawback was the performance of data synchronization, which is still not optimal.

Research on a Virtual Reality-based multiplayer building construction educational game developed in 2021 was written by Jacobsen *et al.* [10]. This research discussed lean construction simulations that could be used for academic purposes or professional education that still rely on manual data input and analysis. The game engine used was Unity3D with the C# programming language assisted by the XR Interaction Toolkit for the game framework section. PUN 2 from Photon was used as an integration to build a multiplayer experience (server connects clients A, B, and C). The drawback of this research was the expensive cost of renting a PUN 2 server.

Research on the efficiency of data storage models in mobile applications was written in 2022 by Sukarsa *et al.* [11]. This research discussed redesigning a database system into a more efficient version using the SQLite database model and a cheaper version of the NoSQL database. This research aimed to reduce the use of storage space for read, write and delete processes on Cloud Firestore, which is cheaper. The drawback of this research was that it still used Cloud Firestore as its own data storage cloud, so it could still reduce the cost efficiency. The drawback of this research was that it still used Cloud Firestore as its own data storage cloud, so it could still reduce the cost efficiency.

The application used as a reference in this research was Kahoot! [12] and Quizizz [13]. Kahoot! [12] and Quizizz [13] have a similar concept, namely an interactive application with many players. Kahoot! [12] and Quizizz [13] allows players to create their own questions, share, and organize their own game concepts being played. The drawback of the reference application mentioned was that this application does not have a knowledge base in making questions because the players have to make questions first.

Other applications that became references for development were Wordbrain [14] and Wordscapes [15]. Wordbrain [14] and Wordscapes [15] have a similar concept, namely an educational application of random word games in one language, namely English. The drawback of the application was that it could not play games with other players.

Besides applications like Wordbrain [14] and Wordscapes [15], several multiplayer games have been developed. These were “Knights Fantasy Online” [16], Robostrike game [17], two-dimensional multiplayer games using the Unity 3D engine [18], multiplayer matching quiz games on the Moodle learning management system [19], massive multiplayer online real-time strategy games [20], Multiplayer Online Battle Arena (MOBA) games with the theme of Wayang [21]. The last was research on the design of multiplayer game applications for quizzes [22].

Other applications of gamification for learning include interactive programming for teaching JavaScript programming [23] and a conceptional Engagement Framework for gamification learning platform [24], Game for design, development, and evaluation of language assessment tools [25], API-based backend system of the game client Discover Indonesia [26], a mobile educational game to learn Javanese vocabulary [27], a mobile guessing game as training for children with autism [28], a mobile-based Balinese Script Educational Game [29], an Android-based sign language educational game [30], New Normal COVID-19 educational game [31], a rare animal recognition game with Construct 2 [32], research on Bingo games for elderly users [33], two-dimensional top-down puzzle adventure game [34], an educational game about Indonesia [35], a game for collecting experimental data obtained from youths who have joined criminal groups around Los Angeles city [36], and games as learning English vocabulary medium in college [37].

The games developed in the discussion above still have a significant constraint in the form of high server capacity requirements in managing multi-users in real-time. The advantage of using WebSocket compared to HTTP technology has also been implemented and can be used as a reference [38]. The difference between WebSocket performance testing and the HTTP protocol that has been studied by Łasocha [38] with the performance testing under study was the scheme used in the test. Scheme used was taken from the data synchronization scheme between the player and the server. In addition to Web Socket technology, another factor that will be influential in developing the design was the memory usage optimization in non-relational database structures compared to RDBMS systems [11].

III. Materials and Methods

A. Gamification Model

The Gamification model used involved primarily educational Gamification principles. The Gamification model with the Gamification principle was based on pedagogical views that guide game mechanics for eccentric motivation [39]. The Gamification model used contains various mechanisms where players are motivated to complete challenges, such as unlocking the next level, earning points, and competing with other players.

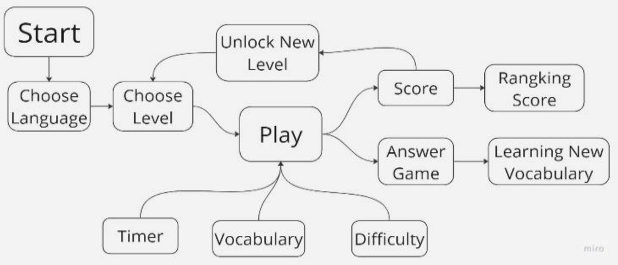


Figure 1. Gamification model on random words.

The Gamification model was designed in the development to improve skills, expand the learning experience, present a learning guide, and present game rules. Fig. 1 illustrates the Gamification mapping in the Random Words application. The Gamification model presented several parts, such as Goals, Challenges, Customization, Progress, Feedback, Competition, Visible Status, Access, and Time Restriction [39]. The Gamification model can be seen in Table I.

TABLE I. The Gamification Model

Design Principle	Game Mechanics
Goals	Result Game, Learning New Vocabulary
Challenges	Level, Language
Customization	Level Customization
Progress	Points
Feedback	Result Answer after Game End
Competition	Ranking Score in Multiplayer Game
Visible Status	Ranking Score in Multiplayer Game
Access/unlocking Content	Level
Time restriction	Time Countdown

B. Software Development Life Cycle

The method used in this research was the Waterfall Method in developing Random Words Mobile Game. The Waterfall method was chosen because it has several advantages, such as analyzing needs at the beginning of the work so that the team already understands the scope of work, utilizing resources that can be done separately, and the project completion time can be projected more precisely [40]. The Waterfall method is one of the Software Development Life Cycle (SDLC), which consists of requirements analysis, design, implementation, and testing.

1) Requirements definition

Requirements definition was carried out at the beginning of application development to determine the initial scope of the application and prepare requirements in application development. Requirements defined included research tools and knowledge databases.

a) Research tools

The hardware used to design and develop WebSocket-based Multiplayer Mobile Game Applications were ASUS TUF with AMD Ryzen 5 4600H Processor with Radeon Graphics 3.00 GHz & NVIDIA GTX1650TI 4GB, 8.00 GB RAM, and 64-bit operating system, x64-based processor. The software used to carry out the design and development of WebSocket-based Multiplayer Mobile Game Application Development Design included the Windows 10 Home Operating System, Visual Studio Code, Android Studio, and Flutter SDK.

b) Knowledge base sources

The data used in this research were obtained from various sources, such as open-source projects and language dictionary websites. The language database was obtained from several open-source projects from the project management cloud service. Indonesian Dictionary was taken from an open-source project repository developed by Umami [41]. English Dictionary was taken from an open-source project repository developed by Amir [42]. Javanese dictionary was taken from an open-source project repository developed by Lutfi [43]. Balinese dictionary was taken from Bali Dictionary website [44].

The method used in this research was the literature study method. The literature study method was carried out by looking for written sources on previous research, open source projects, and language dictionary websites.

2) System and software design

a) System overview

Created system overview can be seen in Fig. 2. Fig. 2 is an overview of the WebSocket-based Scramble word Multiplayer Mobile Game Application design. The system network has two threads: the single-player thread and the multi-player thread. Single-player allowed players to play alone off the network (offline). Players could choose the level and save the game progress. The data source used in offline games was obtained from an encrypted local database, so it does not require a network to process knowledge base requests from the Server. Multiplayer thread is gameplay with many players in one game on the network (online). Multiplayer had two types of entities that interact with the system. The first entity was the first player as well as the game host, and the second was the other players who join the game.

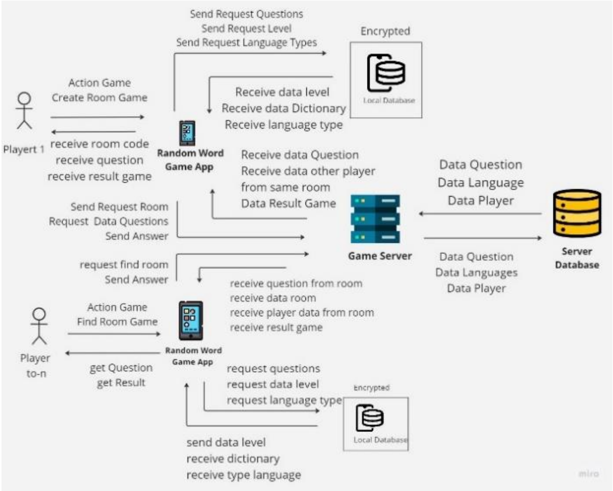


Figure 2. System overview.

b) Gameplay design

Gameplay consisted of the core mechanics and the game interface experience. The core mechanism was the essence of a game [45]. The core mechanism became a game mechanism rule that made players feel entertained or challenged based on that game rule.

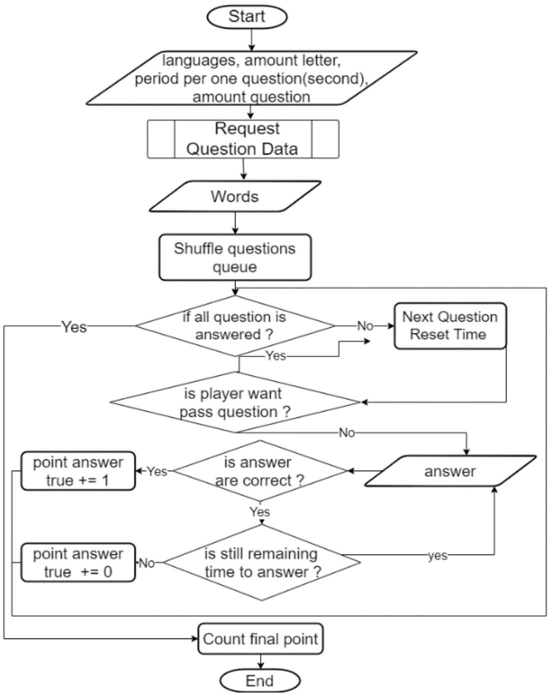


Figure 3. Core gameplay mechanisms flow.

Fig. 3 is the core game mechanism flowchart of the Random Word Mobile Game, which begins with input requests such as the type of language, the length of the letters, the time limit for answering each question, and the number of questions. The first process was to request data in the form of words according to the choice of game type and language. The second process was randomizing the order of questions and words. Then, the third process was the process of players answering questions. The number of questions, random letters, and the time limit for answering questions became the main rules in the core mechanism of the Scramble Word Mobile Game. The process started with the system checking unanswered questions; if it does not yet, the player must answer the questions. Players could skip the previous scramble word question to the next question and could return to answering the missed questions. The player could not answer again if the player had answered correctly and had run out of time to answer a question. The fourth process was to calculate the score obtained by the player. The score results will be displayed on the game results page.

c) Application process flow design

The WebSocket-based Scramble Word Multiplayer Mobile Game Application had three process flows in running game applications. Application process flows included single-player, game room creation, and game room join flows.

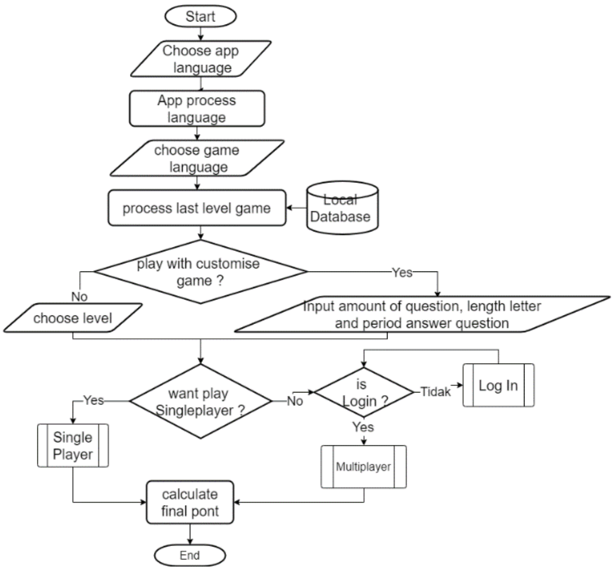


Figure 4. Random word system flowchart.

Fig. 4 is a process flowchart for the WebSocket-based Scramble Word Multiplayer Mobile Game Application system. The process flow started with the user selecting the application language consisting of two languages, Indonesian and English. The second process was selecting the game language. The game language had four choices: Indonesian, Balinese, English, and Javanese. Further, the user must select the game language as the choice of word source for scramble words. Then, the third process was selecting the scramble word game rules. The user selects the game using level rules or game customization. Player-level rules will be given game rules adjusted to the level stage. The higher the level, the more difficult the game. Levels have their own target score to unlock the next level. Game customization allowed users to create their own rules for answering questions. The fourth process was that the user chose a single-player mode or multiplayer. Single-player allows the user to play the game alone without competing with other players. In contrast, multiplayer allows users to play with other people. First, users must log in to be able to play with many players (Multiplayer). Furthermore, the fifth process was that the player accepted the game results. For users who choose a game with a single mode, choose level settings, and have exceeded the level target, players can access the next level.

d) Use case diagram design

Use Case diagram for the WebSocket-based Random Multiplayer Mobile Game Application design diagram can be seen in Fig. 5.

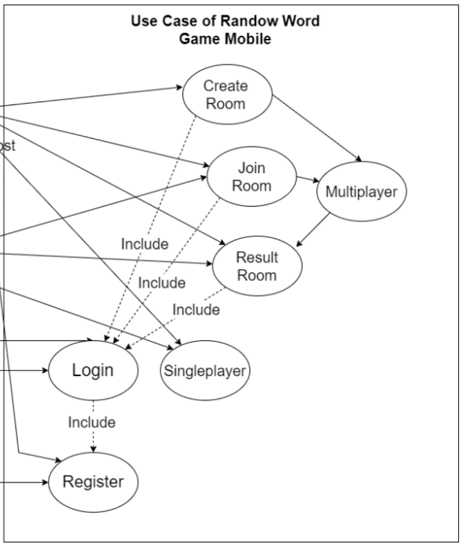


Figure 5. Use case diagram of random word application.

Use Case Diagram in the Scramble Word Multiplayer Mobile Game Application has two actors interacting with the system. The 1st player or Host was an actor who created and managed the game room on the system. The nth player was a player who only needed to join the game room without creating a game room to play online. All actors also get the opportunity to play in multiplayer or singleplayer. Each actor was also required to authenticate users in order to access multiplayer games. Players could play offline in singleplayer game mode.

e) Multiplayer design

Design scheme of Multiplayer game flow can be seen in Fig. 6.

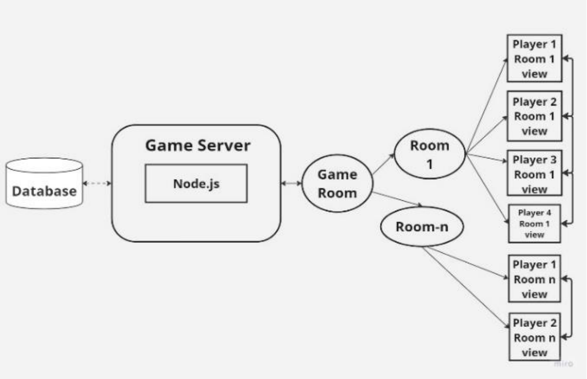


Figure 6. Design scheme of multiplayer game flow.

Design scheme of Multiplayer game is explained as follows:

(1) A player can create a game room where other players can join the game room. When the players connect to the game, the players will directly enter the game room, invited by the game host or creator. Each game room has a limit on the number of players joining according to the game room settings created by the game host;

(2) The game host can start the game at any time. Each player will have the same questions, but it will be randomized again by the system so that the players have a different order of questions from other players. Every player's status that changes will update the player's status data to other players so that the other player's data will always be synchronized;

(3) The server will store the player's data who have joined and the game status of each player. The server can host multiple game rooms simultaneously.

f) Game synchronization architecture

- *Game Preparation Synchronization Architecture*

Game Preparation Synchronization is the data synchronization between all players and the server before the game starts. Game preparation synchronization architecture can be seen in Fig. 7.

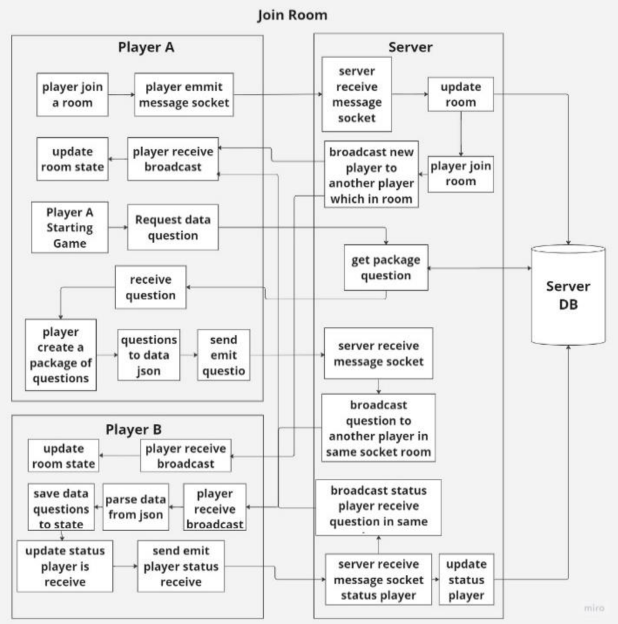

Join Room
Player A
player join a room
player emmit message socket
update room state
player receive broadcast
Player A Starting Game
Request data question
receive question
player create a package of questions
questions to data json
send emit questio
Player B
update room state
player receive broadcast
save data questions to state
parse data from json
player receive broadcast
update status player is receive
send emit player status receive
Server
server receive message socket
update room
broadcast new player to another player which in room
player join room
get package question
server receive message socket
broadcast question to another player in same socket room
broadcast status player receive question in same
server receive message socket status player
update status player
Server DB
miro

Figure 7. Game preparation synchronization architecture.

Game preparation synchronization started with player A who created the game room and joined the game room simultaneously. The server side will create a separate socket room when a player creates a separate game room. Socket room was used to create a limited broadcast room that certain clients could only receive. Other players who have just joined the game room will update the latest data on the server database. Every player who has joined the game room will get player data that have joined so that the data is synchronized with the player data on the server. Player A or the game host sent information that the game was about to start. Player A requested a question from the server according to the request from player A. Player A sent questions again that have been received through the server to each player who has joined the game room. Player B, whose question data will be parsed again and saved back to the local state. Player B, who has received the question, will give the status that player B has received the question to the server section. The server will save the status and pass it on to other players. The game starts when every player in the game room has received a question.

- *Game Result Synchronization Architecture*

Game Results Synchronization is the game data synchronization between all players and the server after a game is over to display game results. Game result synchronization architecture can be seen in Fig. 8.

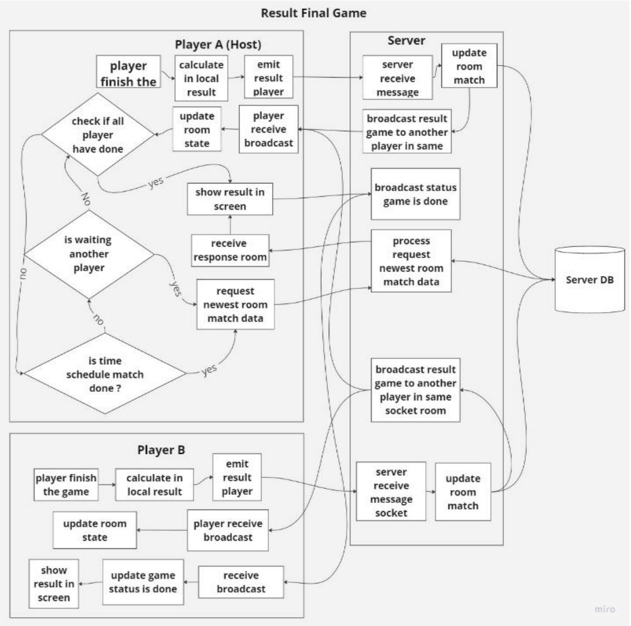

Result Final Game
Player A (Host)
player finish the
calculate in local result
emit result player
check if all player have done
update room state
player receive broadcast
yes
No
show result in screen
receive response room
is waiting another player
yes
request newest room match data
no
no
is time schedule match done ?
yes
Server
server receive message
update room match
broadcast result game to another player in same
broadcast status game is done
process request newest room match data
broadcast result game to another player in same socket room
server receive message socket
update room match
Server DB
Player B
player finish the game
calculate in local result
emit result player
update room state
player receive broadcast
show result in screen
update game status is done
receive broadcast
miro

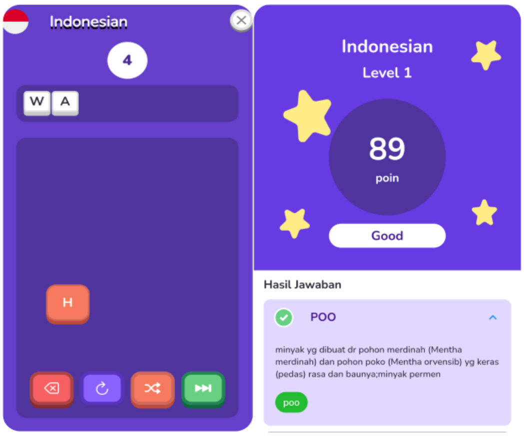
Figure 8. Game result synchronization architecture.

Game Result Synchronization started with the player who had finished the game. Players sent game results information to the server. The server stored the player's game results and sent the game results back to other players. The process of sending game results will be carried out by every player who has finished the game. Thus each player has the same game results as the server. Player A as well as the game room host, wait for other players who have not finished the game. The game room host waits for a specified amount of time until it ends. The game room host sent the game over status when all players had finished and the waiting time had ended. The game status will be sent through the server and broadcast by the server to all players.

3) Implementation

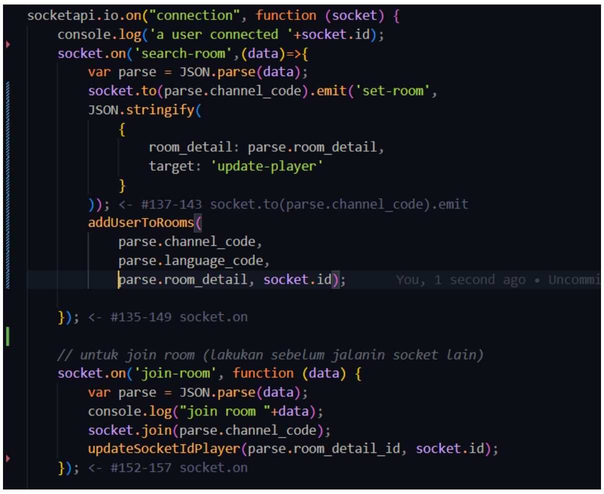
The implementation of the system was built based on a predetermined software design. System implementation was divided into two sides, the application side (Fig. 9) and the server side (Fig. 10).

The implementation of the application side was the main implementation in building the Random Word Game. The application side required the server side in the game process with other players.



minyak yg dibuat dr pohon merdinah (Mentha merdinah) dan pohon poko (Mentha orvensib) yg keras (pedas) rasa dan baunya;minyak permen

Figure 9. Implementation on the application side.



```
socketapi.io.on("connection", function (socket) {
    console.log('a user connected '+socket.id);
    socket.on('search-room',(data)=>{
        var parse = JSON.parse(data);
        socket.to(parse.channel_code).emit('set-room',
        JSON.stringify(
            {
                room_detail: parse.room_detail,
                target: 'update-player'
            }
        )); <- #137-143 socket.to(parse.channel_code).emit
        addUserToRooms(
            parse.channel_code,
            parse.language_code,
            parse.room_detail, socket.id);        You, 1 second ago • Uncommi

    }); <- #135-149 socket.on

    // untuk join room (lakukan sebelum jalanin socket lain)
    socket.on('join-room', function (data) {
        var parse = JSON.parse(data);
        console.log("join room "+data);
        socket.join(parse.channel_code);
        updateSocketIdPlayer(parse.room_detail_id, socket.id);
    }); <- #152-157 socket.on
```

Figure 10. Implementation on the server side.

Server side was the side that managed the request and delivery of data to the client. Server side used Socket.IO to help manage two-way synchronization between the client and server.

4) System testing

Testing in research was divided into two parts, application testing with the black-box method and WebSocket testing. Black-box testing was carried out to know the functionality of the application parts running or not. WebSocket testing to test the performance and efficiency of using WebSocket.

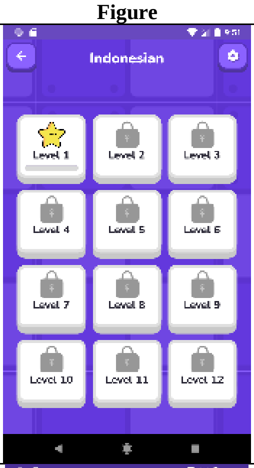
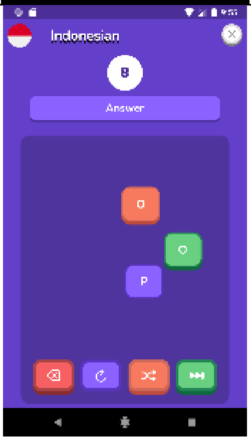
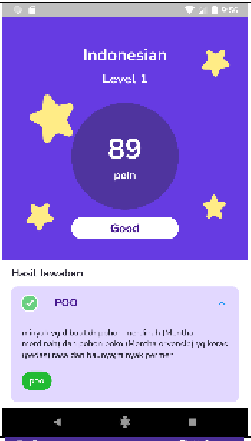
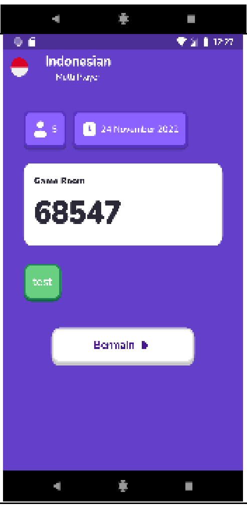
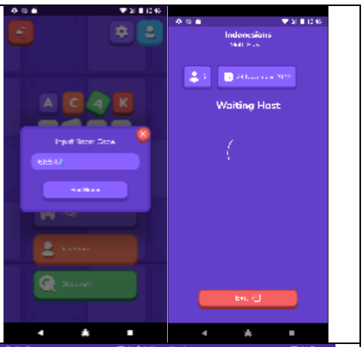
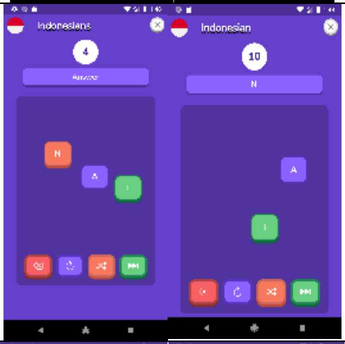
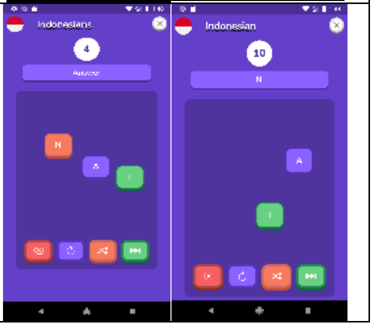
IV. Result and Discussion

Game Scramble Word Multiplayer is a education game multiplayer for new language education, with WebSocket technology in synchronizing data. The development is focused on game mechanics and multiplayer games. Application use four languages, Indonesian, English, Balinese, and Javanese.

A. *Task Scenario Test Results*

Test scenarios are carried out to ensure whether the application runs according to the specified scenario. The test scenario can be seen in Table II.

TABLE II. TASK SCENARIO TESTING PROCESS

Process	Result	Figure
Displays the game level page	OK	 Indonesian Level 1 Level 2 Level 3 Level 4 Level 5 Level 6 Level 7 Level 8 Level 9 Level 10 Level 11 Level 12
Run games offline	OK	
Displays game results offline	OK	 Indonesian Level 1 89 poin Good Hasil Jawaban POO
Game rooms creation and generating multiplayer token codes	OK	 Indonesian Multiplayer 24 November 2022 Game Room 68547 test Bermain
Join a game room for multiplayer game	OK	 A C K Input Token Game Indonesians Multiplayer Waiting Host
Send questions to all players	OK	 Indonesians 4 Answer N A I Indonesian 10 N A I
Send game results when the game is over	OK	 Indonesians 4 Answer N A I Indonesian 10 N A I

B. Test Result Comparison

The testing comparison was carried out to determine the performance and efficiency of the synchronization model with WebSocket and the HTTP synchronization model. Testing was carried out based on two scenarios. The first scenario was game questions synchronization and the second scenario was the game results. Each test scenario will run a test duration of 100 times, with each duration running the same 10 scenarios. Scenario testing will run a scenario replica that occurs in the application according to the protocol being tested. The questions synchronization scenario will run a questions synchronization replica between players so that they have the same questions between players. Game results synchronization scenario runs a game results synchronization replica where players will wait for other players to finish the game. Testing was carried out using the load testing method using the Artillery.io tools [46].

The results of the scenario testing of questions distribution synchronization and results distribution can be seen in Table III. Testing with questions distribution synchronization scenario with HTTP resulted in an average processing time per scenario of 22.20802 ms/scenario with the longest scenario processing time of 28.7608 milliseconds. There were 305 successful scenarioss for question distribution with HTTP and 695 failed scenarios. Testing with questions distribution synchronization scenario with WebSocket resulted in an average processing time per scenario of 5.44735 ms/scenario with the longest scenario of 5.5656 milliseconds. There were 1000 successful scenarios for question distribution with WebSocket and 0 failed scenario.

TABLE III. Test Result of Load Testing Scenarios of Questions Distribution and Result Distribution

Testing Items	**Questions Distribution**		**Results Distribution**	
	HTTP	**WebSocket**	**HTTP**	**WebSocket**
Average Completion Time	22.20802 ms/scenario	5.44735 ms/scenario	5.13909 ms/scenario	3.08463 ms/scenario
Longest Completion Time	28.7608 ms	5.5656 ms	5.223 ms	3.1759 ms
Success Scenario	305 scenarios	1000 scenarios	1000 scenarios	1000 scenarios
Failed Scenario	695 scenarios	0 scenarios	0 scenarios	0 scenario

Testing with results distribution synchronization scenario with HTTP resulted in an average processing time per scenario of 5.13909 ms/scenario with the longest scenario processing time of 5.223 ms. Testing with results distribution synchronization scenario with Websocket resulted in an average processing time per scenario of 3.08463 ms, with the longest scenario of 3.1759 ms. There were 1000 successful scenarios for the game results scenario with Websocket and o failed scenario.

The testing results showed that the process completion for two types of scenarios that used synchronization with WebSocket has a faster process than synchronization with HTTP. The test results on the question distribution scenario showed an increase in performance from the WebSocket version, from the average completion time of percentage increase was 75.47%, the scenario completion time was 81.83%, and increased success in running scenario was 69.5%. A comparison table of synchronization test results can be seen in Table IV, which shows a synchronization comparison. The percentage value is obtained in Eq. (1).

$$NP = (NVH - NN) \div NN \ \times 100\% \qquad (1)$$

NP is the percentage value being searched for, and NPH is the smallest value from the test comparison. NN is the highest value of the test comparison. The results of testing with the question distribution scenario can be seen in Table IV.

TABLE IV. Comparison of Synchronization Test Result

Testing Items	**Questions Distribution**		**Results Distribution**		**Difference (%)**	
	HTTP	**WebSocket**	**HTTP**	**WebSocket**	**Questions Distribution Synchronization**	**Results Distribution Synchronization**
Average Completion Time	22.20802 ms/scenario	5.44735 ms/scenario	5.13909 ms/scenario	3.08463 ms/scenario	75.47 %	39.97%
Longest Completion Time	28.760.8 ms	5.5656 ms	5.223 ms	3.1759 ms	81.83 %	39.21%
Success Scenario	305 scenarios	1000 scenarios	1000 scenarios	1000 scenarios	69.50 %	0%
Failed Scenario	695 scenarios	0 scenario	0 scenario	0 scenario	100%	0%

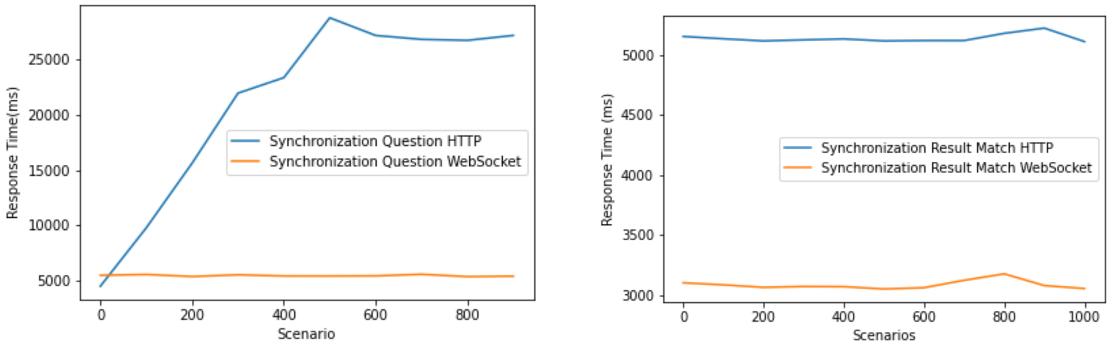


Figure 11. Graph comparison of test results.

Test results on scenarios of question distribution and game results distribution show an increase in performance in the new WebSocket-based synchronization model. The comparison results between the HTTP version and the WebSocket version in questions distribution synchronization based on the average completion time have a difference of 75.47%, the difference between the completion times of the longest process was 81.83%, and the scenario that runs successfully has a difference of 69.5%. In addition, it does not encounter any sync failures in the new model. The comparison results of results distribution synchronization based on the average time have a difference of 39.97%, the difference in the longest processing time was 39.21%, and the difference between successful and failed scenarios was 0%.

The test results graph of the two scenarios can be seen in Fig. 11. The performance of the game result distribution scenario (a) between synchronization with HTTP and the new model synchronization with WebSocket experienced a marked difference. Scenarios with HTTP have a response time between 5.0505 to 5.223 ms, while Scenarios with WebSocket have a response time of 3.0108 to 3.1759 ms. The question distribution scenario performance (b) also experienced the same thing. Questions synchronization scenarios with HTTP from the first scenario experienced an increase in response time until the final scenario process. Question distribution scenarios with HTTP have a response time between 3.5154 to 28.7608 ms, while question distribution scenarios with WebSocket have a response time of 5.221 to 5.565 ms.

V. Conclusion

This research has developed a synchronization model by maximizing the role of the local database in mobile applications combined with the application of WebSocket technology to overcome the high resource requirements in implementing multiplayer-based games. This research has been successfully implemented in a mobile game application with random word case studies with a knowledge base sourced from four languages, namely Indonesian, Balinese, English, and Javanese, on the Flutter programming language platform

Comparison testing of data synchronization between the conventional synchronization design and the new synchronization design was carried out with a test duration of 100 times, with each test duration running the same 10 scenarios. The scenarios tested in this research were the synchronization of question data and game results. The test results on the question distribution scenario showed an increase in performance in several aspects. The average time for task completion has increased by 75.47%, the longest time required for task completion has increased by 81.83%, successful tasks have increased by 69.5%, and repairs to failed tasks have increased to 100%.

The test results on the scenario of game results distribution also showed an increase in performance. The average time for task completion has increased by 39.97%, the longest time required for task completion has increased by 39.21%, no failed task scenarios were found, and all tasks were successfully completed.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Sukarsa and Bayupati supervised the whole research development and took the lead in writing the manuscript. Ocha established the theoretical formalism and transfer it to technological development. Wira and Wisswani helped write manuscript and provided critical feedback. All authors had approved the final version.

Funding

This research is funded by Institute for Research and Community Service (Udayana University) through the 2023 Invention Grant research scheme (PIU) with contract number B/1.670/UN14.4. A/PT.01.03/2023.

References

[1] K. Djoko, *Fundamentals of General Liguistics*, Jakarta: Fakultas Sastra, Universitas Indonesia, 1982.

[2] D. W. Hadi, W. Permanawiyat, N. Sambodo, A. O. Anindyatri, and Mas'ad, *Language Statistics 2019*, Jakarta: Center for Data and Statistics on Education and Culture, Ministry of Education and Culture, 2019.

[3] L. Solano, E. Ulehlova, and V. Espinoza, "Interactive games as educative strategy to motivate students to communicate inside the classroom," *Advances in Intelligent Systems and Computing*, pp. 579–586, 2018.

[4] M. Silalahi, "Improving students' interest in learning English by using games," *International Journal of Theory and Application in Elementary and Secondary School Education*, vol. 1, no. 1, pp. 50–56, May 2019.

[5] A. Hanif, "Online marketing digital learning module by calculating Android-based selling prices," *Profitabilitas Jurnal*, vol. 1, no. 1, pp. 38–45, 2021.

[6] A. Lombardi, *WebSocket*, Sebastopol: O'Reilly, 2015.

[7] Sugiyanto, W. K. Tai, and G. Fernando, "The development and evaluation of web-based multiplayer games with imperfect information using WebSocket," in *Proc. 2019 12th International Conference on Information and Communication Technology and System (ICTS)*, 2019, pp. 252–257.

[8] I. P. A. E. Pratama and I. W. G. Arisna, "Fighter plane online game based on design science research methodology using socket.io and nodejs," *Indonesian Journal of Engineering and Science*, vol. 3, no. 2, pp. 29–37, Feb. 2022.

[9] A. Syauqi and P. Hijjah, "Design of networked multiplayer snake and ladder educational game based on hash map and vector data structure," *JUITA: Informatics Journal*, vol. 8, no. 2, 159, May 2020.

[10] E. L. Jacobsen, N. S. Strange, and J. Teizer, "Lean construction in a serious game using a multiplayer virtual reality environment," in *Proc. 29th Annual Conference of the International Group for Lean Construction (IGLC)*, 2021, pp. 56–64.

[11] M. Sukarsa, I. Antara, P. W. Buana, I. P. A Bayupati, N. Wisswani, and D. Puteri, "Data storage model in low-cost mobile applications," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 28, no. 2, pp. 1128–1138, May 2022.

[12] Kahoot! [Online]. Available: https://kahoot.com/

[13] Quizizz. [Online]. Available: https://quizizz.com/admin

[14] WordBrain—Word puzzle game. [Online]. Available: https://play.google.com/store/apps/details?id=se.maginteractive.wordbrain&hl=en_US

[15] Wordscapes—Apps on google play. [Online]. Available: https://play.google.com/store/apps/details?id=com.peoplefun.wordcross&hl=en_US

[16] D. Haris, V. Mawardi, and D. Pratama, "Developing multiplayer online game 'knight fantasy online'," *Computatio: Journal of Computer Science and Information Systems*, vol. 5, no. 1, 34, May 2017.

[17] C. J. R. Piza, J. R. P. S. I. L. Castro, J. R. P. Tutor, and I. L. Castro, "Robostrike: Analysis and implementation of a multiplayer game with current technologies," *Computer Science and Information Systems*, 2019.

[18] S. B. P. A. Somesh, A. Sharath, and G. Kumaresan, "2D multiplayer game development using unity," *International Journal of Advance Research and Innovative Ideas in Education*, vol. 6, pp. 256–261, 2020.

[19] I. N. D. Kotama, K. O. Saputra, and Linawati, "Proposed model of multiplayer matching game plugins using WebSocket in Moodle," *International Journal of Emerging Technologies in Learning*, vol. 14, no. 11, pp. 194–201, 2019.

[20] J. A. Luís, P. Fazenda, and D. Reis, "Creating a dynamic battle system for a massive multiplayer online real-time strategy game," *Computer Science and Information Systems*, 2017.

[21] E. H. Suliman and Andre, "The adoption of multiplayer online battle arena to introduce Wayang character," *Journal of Game, Game Art and Gamification*, vol. 6, no. 2, pp. 56-66, 2021.

[22] M. A. Wibowo, I. M. A. Suyadnya, and K. O. Saputra, "Design and build a multiplayer game application as a tool for WebSocket-based learning processes," *Journal Spektrum*, vol. 8, no. 3, pp. 120–127, 2021.

[23] R. Maskeliunas, A. Kulikajevas, T. Blazauskas, R. Damaševičius, and J. Swacha, "An interactive serious mobile game for supporting the learning of programming in JavaScript in the context of eco-friendly city management," *Computers*, vol. 9, 102, May 2020.

[24] M. A. Alsubhi, N. Sahari, and T. S. M. T. Wook, "A conceptual engagement framework for gamified e-learning platform activities," *International Journal of Emerging Technologies in Learning*, vol. 15, no. 1, pp. 4–23, 2020.

[25] J. Delcker and D. Ifenthaler, "Mobile game-based language assessment," *International Journal of Emerging Technologies in Learning*, vol. 15, 195, May 2020.

[26] L. J. Tandey, "The design and development of backend system for a game application," *Journal of Game, Game Art and Gamification*, vol. 3, no. 1, pp. 14–19, 2018.

[27] R. Wijayanti, S. Muntomimah, and R. Khoirunnisak, "Android game: Education Javanese vocabulary," *J. Phys. Conf. Ser.*, vol. 1869, 12089, May 2021.

[28] H. Kalantarian, D. P. Wall, P. Washington, and K. Jedoui, "A mobile game for automatic emotion-labeling of images," *IEEE Trans Games*, vol. 12, no. 2, pp. 213–218, 2020.

[29] I. G. Andika, D. Saputra, C. P. Yanti, and G. Indrawan, "Mengenal Aksara Bali: Balinese script game education based on mobile application," *Journal of Games, Game Art, and Gamification*, vol. 4, no. 1, pp. 1–9, 2019.

[30] I. F. M. Rachmat and G. Gazali, "Development of an android-based sign language education game using A scrum-game approach," *Juita*, vol. 9, no. 1, May 2021.

[31] T. F. Prasetyo, A. Bastian, and L. T. Wahyuni, "New normal after education game design pandemic uses approaches agile scrum application of Finite State Machine (FSM) method," *International Journal of Information System and Technology Akreditasi*, vol. 5, no. 2, pp. 111–126, 2021.

[32] D. Damayanti, M. Akbar, and H. Sulistiani, "Android-based educational game for introducing endangered animals using construct 2," *Journal of Information Technology and Computer Science*, vol. 7, 275, May 2020.

[33] Y. J. Chin, W. N. Lim, and C. S. Lee, "Mobile game for the elderly: Bundled bingo game," in *Proc. TENCON 2017—2017 IEEE Region 10 Conference*, 2017, pp. 2262–2267.

[34] A. Jaffray, A. Finn, and J. R. C. Nurse, "SherLOCKED: A detective-themed serious game for cyber security education," in *Proc. Human Aspects of Information Security and Assurance*, 2021, pp. 35–45.

[35] F. Suprapto and F. Irrahali, "Educational game 'Mengenal Indonesia' as a medium to introduce Indonesia to the kids," *International Journal of Higher Education*, vol. 8, 47, May 2019.

[36] S. Cooney, "Mobile game theory with street gangs," in *Learning and Knowledge Discovery in Databases*, Springer International Publishing, pp. 579–589, 2020.

[37] Y. L. Huang, D. F. Chang, and B. Wu, "Mobile game-based learning with a mobile app: Motivational effects and learning performance," *J. Adv. Comput. Intell. Intell. Informatics*, vol. 21, no. 6, pp. 963–970, 2017.

[38] W. P. Łasocha and M. Badurowicz, "Comparison of WebSocket and HTTP protocol performance," *Journal Computer Sciences Institute*, vol. 19, pp. 67–74, 2021.

[39] D. Dicheva, C. Dichev, G. Agre, and G. Angelova, "Gamification in education: A systematic mapping study," *Educational Technology and Society*, vol. 18, pp. 75–88, May 2015.

[40] R. Sherman, *Business Intelligence Guidebook*, Elsevier Inc., 2015.

[41] S. R. Umami. GitHub—Rizaumami/kbbi: Kamus Besar Bahasa Indonesia Edisi IV. [Online]. Available: https://github.com/rizaumami/kbbi

[42] F. Amir. GitHub—Caravancodes/dictionary-box: Dictionary apps English—Indonesia. [Online]. Available: https://github.com/caravancodes/dictionary-box

[43] M. I. Luthfi. GitHub—Milstrike/Kamus-Jawa-Indonesia: Sebuah aplikasi kamus Jawa—Indonesia menggunakan database sqlite. [Online]. Available: https://github.com/milstrike/Kamus-Jawa-Indonesia

[44] Dictionary—Balinese-Indonesian-English Complete Digital Dictionary. [Online]. Available: https://dictionary.basabali.org/Dictionary

[45] E. Adams, *Fundamentals of Game Design*, 2nd ed. Berkeley: New Riders, 2010.

[46] Artillery. Cloud-scale Load Testing. [Online]. Available: https://www.artillery.io/

Copyright © 2023 by the authors. This is an open access article distributed under the Creative Commons Attribution License (CC BY-NC-ND 4.0), which permits use, distribution and reproduction in any medium, provided that the article is properly cited, the use is non-commercial and no modifications or adaptations are made.