Logo

Description automatically generated**AIN SHAMS UNIVERSITY**

**FACULTY OF ENGINEERING**

**ICHEP – Senior 1 Level – CESS**

**CSE354: Distributed Computing**

**Spring 2023**

**2D Multiplayer Racing Car Game**

**Submitted to:**

Prof. Dr. Ayman Bahaa

Eng. Mostafa Ashraf

**Submitted by:**

Omar Mohamed Yousef (19P4953)

Omar Helmy Ibrahim Elbanna (19P3904)

Github Repo: [mourra950/Distrubited-System-Racing-Cars (github.com)](https://github.com/mourra950/Distrubited-System-Racing-Cars)

**Table of Contents**

[ABSTRACT 3](#_Toc136465573)

[1.0 SERVER 4](#_Toc136465574)

[2.0 Client 7](#_Toc136465575)

[2.1 Chat Feature 7](#_Toc136465576)

[3.0 Unity Game 10](#_Toc136465577)

[4.0 Deployment 12](#_Toc136465578)

**List of Figures**

[Figure 1 Server Code Snippet 5](#_Toc136465904)

[Figure 2 Client Code Snippet 8](#_Toc136465905)

[Figure 3 Testing Chat using Two Clients 9](#_Toc136465906)

[Figure 4 Game Snapshot 1 11](#_Toc136465907)

[Figure 5 Game Snapshot 2 11](#_Toc136465908)

[Figure 6 Render Screenshot 1 13](#_Toc136465909)

[Figure 7 Render Screenshot 2 13](#_Toc136465910)

# ABSTRACT

This is a documentation of the Distributed Computing initial project submission. This project aims to build a multi-player distributed 2D Car Racing Game along with chatting feature. The system must support multiple, autonomous agents (either human or automated) contending for shared resources and performing real-time updates to some form of shared state. The state of the system should be distributed across multiple client or server nodes. The system should be robust. The system should be able to continue operation even if one of the participant nodes crashes. This is our objective with this project, without delving into the technical details, as they will be thoroughly explained and assessed throughout the final submission.

The methods towards achieving such a goal will be clearly detailed as well as thoroughly described in terms of implementation throughout our development process. That is the purpose of this documentation, to know how to plan such a project and implement it as efficiently and accurately as possible.

# INTRODUCTION

In this exciting project, we delved into the field of distributed systems to create a thrilling and immersive gaming experience. The objective of our project is to design and develop a cutting-edge 2D multiplayer racing car game that leverages the power of distributed systems to provide seamless gameplay, real-time interactions, and exhilarating racing competitions.

With the rapid advancement of technology and the increasing popularity of multiplayer gaming, the demand for immersive and engaging experiences has grown exponentially. Traditional single-player games have given way to multiplayer games that allow players from around the globe to compete, collaborate, and challenge each other. However, designing and implementing a multiplayer game presents a unique set of challenges, especially when it comes to ensuring low latency, synchronization, and scalability across a distributed network.

In our project, we aim to tackle these challenges head-on by harnessing the capabilities of distributed systems. By leveraging the power of multiple interconnected nodes, we can distribute the game logic, handle player interactions, and synchronize game states in several nodes. This not only enhances the overall performance and responsiveness of the game but also allows for a scalable architecture that can accommodate a reasonable number of concurrent players.

The key components of our 2D multiplayer racing car game will include a game server implemented using node.js, python proxy, and unity client game, and a communication protocol that facilitates real-time communication between the players and the proxy, and between proxy and the server. The game server will be responsible for managing player connections and handling game events. The client applications will provide the graphical user interface and render the game elements, while also interacting with the server through our python proxy that acts as a client to the server and a server to unity game to receive game updates and transmit player actions and chat messages.

Throughout the development process, we will focus on key aspects such as minimizing network latency, implementing efficient synchronization mechanisms, and designing a robust fault-tolerant system.

# PROJECT DESCRIPTION

The aim of this project is to develop a distributed system for a 2D car racing game that incorporates interesting features from a systems perspective. The system will support multiple agents contending for shared resources and performing real-time updates to a shared state. By leveraging distributed systems principles, the project will ensure robustness, fault tolerance, and the ability to recover from node crashes.

The system will be designed to meet several important properties. Firstly, it should support multiple, autonomous agents participating in the 2D car racing game. These agents can be human players or automated entities that compete for shared resources and dynamically update the game’s state in real-time.

Robustness is a crucial aspect of the system. It should be designed to handle failures gracefully and continue operation even if one of the participant nodes crashes. This fault tolerance ensures uninterrupted gameplay and prevents any loss of progress or data. Additionally, the system should support the recovery of a crashed player’s state, allowing player to resume playing seamlessly.

The system will offer real-time playing and viewing capabilities, allowing participants to actively race against each other and observe the progress of other players simultaneously. This real-time interaction enhances the multiplayer experience and immerses participants in an engaging environment.

To facilitate communication between participants, chat functionality will be implemented. This feature enables players to exchange messages during, before, and after playing the game. It enriches the overall multiplayer experience.

Caching and copy migration techniques will be employed to optimize application response time. By caching frequently accessed data or game resources, the system can reduce latency and provide faster response times to participants. Additionally, copy migration can dynamically move game resources closer to the participants, reducing network latency and enhancing the overall gameplay experience.

# BENEFICIARIES

Our project is developed for academic purposes with the intention of further enhancements for publication as a real game, can benefit several groups of individuals.

Gaming enthusiasts who have an interest in multiplayer games can benefit from experiencing the 2D car racing system. While not yet ready for commercial release, the project offers an engaging gaming environment where participants can compete against each other in real-time. The chat functionality enables communication between players, fostering social interaction and collaboration. Gaming enthusiasts can enjoy the game for entertainment purposes and gain an understanding of the potential features and challenges involved in developing distributed multiplayer games.

Students and researchers in the field of distributed systems are among the primary beneficiaries. The project provides them with a practical learning experience, allowing them to apply the theoretical knowledge gained in their studies. By actively developing the 2D car racing system with distributed architecture, students can gain insights into the challenges and complexities of building robust and fault-tolerant systems. It serves as a valuable educational tool, enhancing their understanding of distributed systems principles, real-time updates, fault tolerance, and state management.

Although the project requires additional enhancements to be published as a real game, it offers valuable insights for game developers. Developers interested in creating multiplayer racing games with distributed architecture can leverage the project as a foundation. It provides a starting point, demonstrating the fundamental components and functionality required for a multiplayer game. Developers can build upon the project's structure, integrate their own gameplay mechanics, and enhance the user experience to create a fully-fledged, commercially viable game.

Finally, the project can contribute to the open-source community. By sharing the project's source code and documentation, it becomes a valuable resource for developers worldwide. The open-source community can study, modify, and improve upon the project, fostering collaboration and innovation in the fields of distributed systems and game development. Developers can learn from the project's insights and implementations, potentially leading to the creation of more robust and feature-rich distributed game systems.