Siddaganga Institute of Technology, Tumakuru

Autonomous institution affiliated to Visvesvaraya Technological University, Belagavi, Approved by AICTE, New Delhi, Accredited by NAAC and ISO 9001:2015 certified)

Major Project Synopsis

on

An Online Scalable Low Latency Multiplayer Game Using Docker

submitted

for the partial fulfilment of the requirements of

Bachelor of Engineering

In

Computer Science and Engineering

by

1	Subramanya G	1SI18CS115
2	Swaroop K	1SI18CS121
3	Vishnu Teja S	1SI18CS133
4	Vishwak Vemuru	1SI18CS135

Under the Guidance of

Prof. Kavitha M

Assistant Professor

Department of Computer Science & Engineering

(Program Accredited by NBA)

Siddaganga Institute of Technology

B.H Road, Tumakuru-572 103, Karnataka, India.

Web: www.sit.ac.in

Table of Contents

Sl. No.	Title	Page No.
1	Introduction	3
2	Objectives	5
3	Literature Survey	6
4	Motivation	9
5	Methodology	12
6	Tools and Technologies	13
7	Expected Outcome	14
8	Conclusion	15

Introduction

A multiplayer game is a game in which more than one person can play in the same game environment at the same time, either locally and on the same computing system, locally and on different computing systems via a local area network, or via a wide area network, most commonly the Internet.

We aim to build a 2D online multiplayer game that has the following features

- Orthogonal Top-down view
- Co-op Combat
- Character customization
- Public and private rooms for game sessions

There are two types of multiplayer games:

1. Non-networked Games: These are predominantly 2-player games played on a single device with multiple controllers.
2. Networked Games: These allows users from different devices to connect to each other and play the games. It has following two categories:
    a. Local Multiplayer: They are always played on the same local network.
    b. Online Multiplayer: They Can be played through any device across the Internet.

The game would be an online multiplayer game where players are not restricted to the same network. There are two ways of implementing this:

1. Peer-to-Peer: In these types of games, one of the players become the authority and all other players has to send the data to him. He updates the game states and sends the updated states to all the other players. Peer-to-Peer type of game has the following disadvantages:
    - The peer who becomes the authority, has to have a large bandwidth.
    - If the authoritative peer exits, the game automatically ends for all players.
    - Peer server can change the states, which affects the playing experience for all players.

2. Client Server: In these types of games, a dedicated server is set up. All clients must connect to this server and the server is the authority. This successfully overcomes all the disadvantages of the Peer-to-Peer types of online multiplayer games.

However, in a client-server type of online games, there are issues of latency and performance. Solution to the above problems is to use docker containers. Docker enables easy management of

game servers and allows for deployment on the cloud. This allows an instance of the game to run on the nearest cloud server to the user solving the issues of latency. As there are multiple docker containers running the server, the performance is boosted. Further, using docker containers enables support for rolling deployment, autoscaling, monitoring and maintenance.

Objectives

- To enable cross-platform support by building a web application.
- To build a fast-paced combat game.
- To establish real time two-way interactive communication session between the user's browser and a server.
- To enable users to create private and public rooms for game sessions.
- To containerize the game server and reduce the game server overhead.

Literature Survey

Many works and papers were referred and the following things were noted.

Title: Cloud Gaming System in Docker Container Image

Authors: Arun Pugalendhi

Description: This paper discusses about how cloud gaming is the future technology for the gaming industry and that it allows users to play the game anywhere, anytime and from any device. The author says the main challenges in the cloud gaming are resource allocation and ensuring quality of user experience. This paper explains how to improve the performance of cloud gaming by initializing the cloud gaming package inside Docker containers which will allow the application to be more reliable by secularizing its resource allocation and increasing the overall performance of the cloud gaming system and makes it more reliable while utilizing less resources.

Title: Multi-platform Version of StarCraft: Brood War in a Docker Container: Technical Report

Authors: Michal Šustr, Jan Malý, Michal Čertický

Description: The paper presents a dockerized version of a real-time strategy game StarCraft: Brood War, commonly used as a domain for AI research, with a pre-installed collection of AI development tools supporting all the major types of StarCraft bots. This provides a convenient way to deploy StarCraft AIs on numerous hosts at once and across multiple platforms despite limited OS support of StarCraft. In this paper, the authors describe the design of their Docker images and present a few use cases.

Title: Characterizing Docker Overhead in Mobile Edge Computing Scenarios

Authors: G. Avino, M. Malinverno, F. Malandrino, C. Casetti, C. F. Chiasserini

Description: This paper discusses that Mobile Edge Computing (MEC) is an emerging network paradigm that provides cloud and IT services at the point of access of the network. Such proximity to the end user translates into ultra-low latency and high bandwidth, while, at the same time, it alleviates traffic congestion in the network core. Due to the need to run servers on edge nodes, key element of MEC architectures is to ensure server portability and low overhead. A possible tool that can be used for this purpose is Docker, a framework that allows easy, fast deployment of Linux containers. This paper addresses the suitability of Docker in MEC scenarios by quantifying the CPU consumed by

Docker when running two different containerized services: multiplayer gaming and video streaming. The authors' tests, run with varying numbers of clients and servers, yield different results for the two case studies: for the gaming service, the overhead logged by Docker increases only with the number of servers; conversely, for the video streaming case, the overhead is not affected by the number of either clients or servers.

Title: Enhancing Cloud Gaming User Experience through Docker Containers in Fog Nodes

Authors: Manoj Kannan

Description: This paper discusses about cloud gaming and the developments in the field of cloud computing that will allow cloud Gaming to provide high-end quality of gamer experience. The author says that a fundamental requirement of gaming is to provide maximum quality of gamer experience, however cloud gaming suffers in terms of providing Quality of Experience, because the network transmission of game scenes from cloud game server to gamer device is distant. The author endeavours to minimize latency and increase performance in cloud gaming through this paper. The paper proposes moving the cloud game server to the fog nodes present at the edge network of the player based on Node selection algorithm. The paper also suggests that, to increase the performance of cloud gaming, traditional virtual machine should be replaced by light-weight containers.

Title: The Future of Web and Mobile Game Development

Authors: Kevin Curran, Ciaran George

Description: The paper discusses that as HTML5 has become open to the public, many developers have been experimenting with the new possibilities for web development. The authors' aim, through this research paper, to give an overview of what this means to the game development community. This paper evaluated new HTML5 elements and JavaScript features. The paper highlights WebGL, Canvas and WebSockets, which the authors opine have given developers the opportunity to flaunt their creativity by manipulating images, creating 3D environments and providing real-time interaction.

Title: The Development and Evaluation of Web-based Multiplayer Games with Imperfect Information using WebSocket

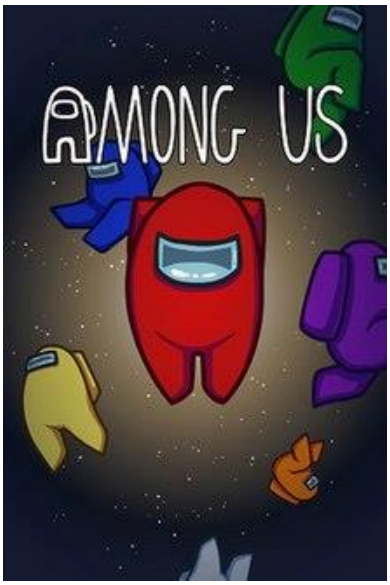
Authors: Sugiyanto, Wen-Kai Tai, Gerry Fernando

Description: This paper considers an example of multiplayer games with imperfect information: Big Two. In this game, each player plays without knowing the opponent's confidential information. The authors say that web-based multiplayer games with imperfect information require real-time communication for handling rapid information changes and the condition of the current state at any time. This paper proposes a new framework for web-based multiplayer games with imperfect information. The authors utilized an open-source WebSocket, namely Socket.IO, and implemented this framework in Big Two as a case study.

Motivation

The following existing works related to online multiplayer games and docker has motivated us to come up with creative ideas for the implementation of the project.

1. **Among Us:** Among Us is a 2018 online multiplayer game developed and published by American game studio Innersloth.



2. **Pokémon Go:** Pokémon Go is a 2016 augmented reality (AR) mobile game developed and published by Niantic in collaboration with Nintendo and The Pokémon Company for iOS and Android devices.

3. **Steam:** Steam is a video game digital distribution service by Valve. It was launched as a standalone software client in September 2003 as a way for Valve to provide automatic updates for their games, and expanded to include games from third-party publishers.

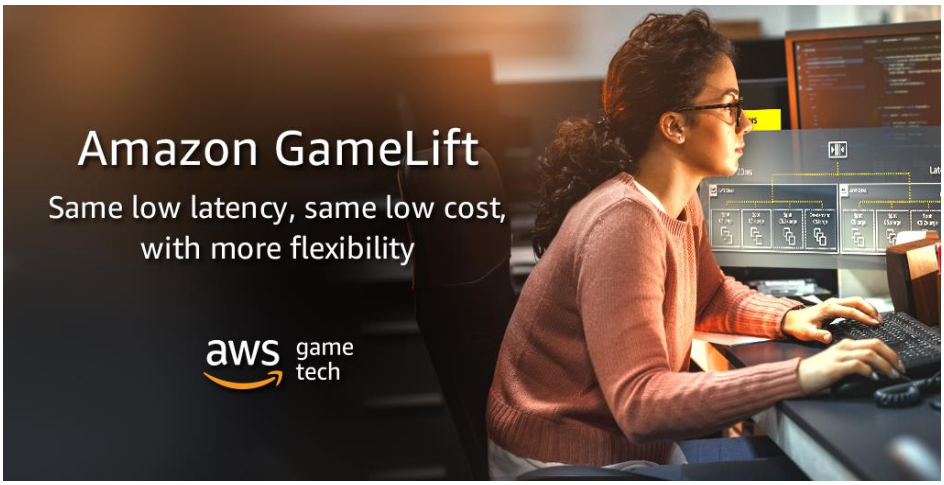
4. **Unity:** Unity is a cross-platform game engine developed by Unity Technologies, first announced and released in June 2005 at Apple Inc.'s Worldwide Developers Conference as a Mac OS X-exclusive game engine.

5. **OVHCloud:** OVH, legally OVH Groupe SAS, is a French cloud computing company which offers VPS, dedicated servers and other web services. As of 2016 OVH owned the world's largest data centre in surface area. As of 2019, it was the largest hosting provider in Europe, and the third largest in the world based on physical servers.

6. **Google Cloud:** Google Cloud Platform, offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search, Gmail, Google Drive, and YouTube.

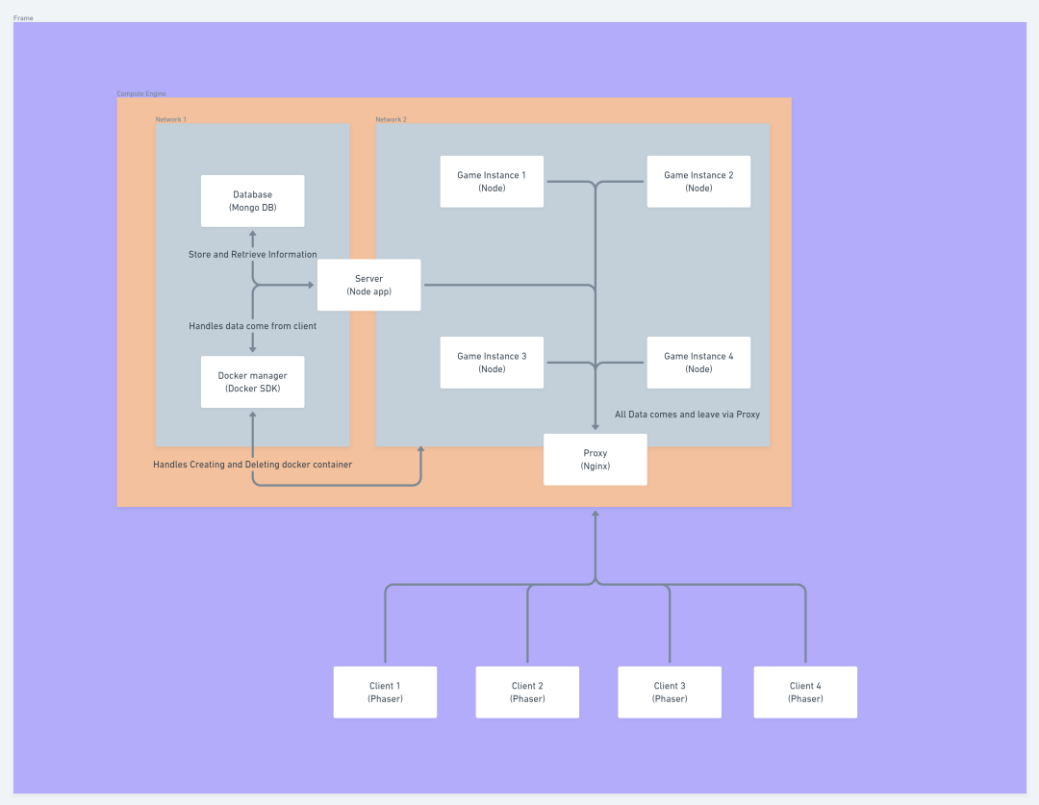
7. **Heroic Labs:** Heroic Labs offers Nakama which is a real-time social competitive game server. It is open-source, scalable, low latency real-time engine. It offers many features like server authoritative multiplayer, Match listing and lobby rooms, sophisticated in-memory matchmaking and many more for the development of online multiplayer games.

8. **Amazon GameLift:** Amazon GameLift is a dedicated game server hosting solution that deploys, operates, and scales cloud servers for multiplayer games. GameLift leverages the power of AWS to deliver the best latency possible, low player wait times, and maximum cost savings.

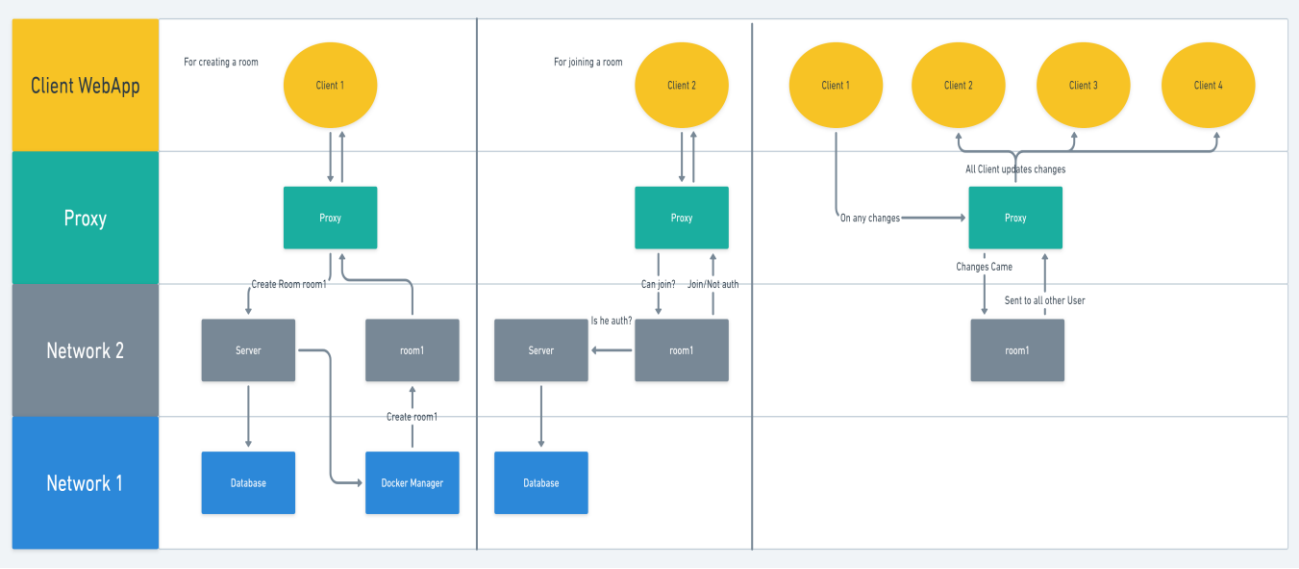


Methodology

The following Fig. shows the architecture of the game. The top part of the architecture diagram depicts the server. Any number of clients can join the server and each client has their own game instance in the server, grouped under rooms. A cluster of docker containers are used to run the server. We also make use of phaser game framework and MongoDB for the database. We make use of AWS EC2 for the hosting of our game.



The following Fig. shows the swimlane diagram of the project. It depicts the three possible interactions between the client and the server.



Tools and technologies

We shall make use of the following tools and technologies for the design and implementation of the project.

Server Side:

- Docker – For containerization
- Docker SDK – For managing the container
- Node – For server scripting
- Express – For server web application framework
- Express-ws – For handling the WebSocket
- Nginx – For reverse proxy
- MongoDB – Database

Client Side:

- React – Client-side rendering
- Phaser – Game engine

Development:

- Docker desktop
- VS Code

Expected Outcomes

- The game will be able to provide cross-platform support.
- The game will be a fast-paced combat game.
- The game will be able to establish real time communication between the user's browser and a server.
- The game will be able to allow users to create private and public rooms for game sessions.
- The game server will run on docker containers.

Conclusion

Thus, we propose to build an online multiplayer game that allow users from across the world to connect and play together. The game would be a 2D multiplayer game that supports features like orthogonal Top-down view, co-op combat, Character customization and public and private rooms for game sessions.

We use docker containers for the game server which allows us to, easily manage, monitor and maintain the game server, allows for deployment on the cloud, reduces latency by running an instance of the game on the cloud server nearest to the user, boost server performance by using multiple docker containers to run the server, it also enables support for rolling deployment and autoscaling.