Software Design Document

Playvera

Ivan Ovcharov, 4090993

This page has been left blank intentionally

Table of content

0. Versioning table

Version 1 (Sprint 1) - September 16, 2021

Version 2 (Sprint 2) - October 6, 2021

Version 3 (Sprint 3) - November 5, 2021

Version 4 (Sprint 4) - November 26, 2021

Version 5 (Sprint 5) - December 16, 2021

1. Introduction

1.1 Purpose

This software design document describes the architecture and system design of Playvera game content webshop and its functionality.

1.2 Scope

Playvera is designed to target game content creators and game dev enthusiasts, providing a webshop where users may share questions, content and games amongst each other. The software is planned to have a favourite games section, most downloaded, greenlighting and forums.

1.3 Overview

This document serves as a description of the functionality and software design for the Playvera webshop.

1.4 Reference Material

https://www.digitalocean.com/community/conceptual_articles/s-o-l-i-d-the-first-five principles-of-object-oriented-design - *SOLID principles*

https://en.wikipedia.org/wiki/Interface_(computing) - *Interface explanation*

https://axios-http.com/docs/intro - *Axios explanation*

https://condor.depaul.edu/sjost/it231/documents/resttable.htm#:~:text=Representational%20state%20transfer%20(REST)%20means,of%20a%20client%2Dserver%20architecture
– *REST AND CRUD OPERATIONS*

https://en.wikipedia.org/wiki/Object-oriented_programming - *OOP programming*

https://en.wikipedia.org/wiki/HTTPS - *HTTP security*

https://en.wikipedia.org/wiki/Class_(computer_programming)

1.5 Definitions and Acronyms

SDDW: Software Design Document

SOLID principles: S - Single-responsiblity Principle O - Open-closed Principle L - Liskov Substitution Principle I - Interface Segregation Principle D - Dependency Inversion Principle

Interface: In computing, an interface is a shared boundary across which two or more separate components of a computer system exchange information. The exchange can be between software, computer hardware, peripheral devices, humans, and combinations of these

Axios: Axios is a promise-based HTTP Client for node.js and the browser. It is isomorphic (= it can run in the browser and nodejs with the same codebase). On the server-side it uses the native node.js http module, while on the client (browser) it uses XMLHttpRequests.

REST/CRUD: Representational state transfer (REST) means using the HTTP GET, POST, PUT, and DELETE operations to implement the CRUD operations: REST also assumes. use of a client-server architecture

Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods).

HTTP security: Hypertext Transfer Protocol Secure (HTTPS) is an extension of the Hypertext Transfer Protocol (HTTP). It is used for secure communication over a computer network, and is widely used on the Internet.In HTTPS, the communication protocol is encrypted using Transport Layer Security (TLS) or, formerly, Secure Sockets Layer (SSL). The protocol is therefore also referred to as HTTP over TLS, or HTTP over SSL.

Class: In object-oriented programming, a class is an extensible program-code-template for creating objects, providing initial values for state (member variables) and implementations of behavior (member functions or methods).

JSON Web Token is a proposed Internet standard for creating data with optional signature and/or optional encryption whose payload holds JSON that asserts some number of claims. The tokens are signed either using a private secret or a public/private key.

React - React (also known as React.js or ReactJS) is a free and open-source frontend JavaScript library for building user interfaces or UI components.

2.0 SYSTEM OVERVIEW

The functionality of Playvera consists of a backend (developed using Java Spring Boot) and React as the front-end UI. The backend has functionality for retrieving games, creating new content, personal information and email management. Admin functionality includes managing application users and game content that is uploaded on the website.

3.0 SYSTEM ARCHITECTURE

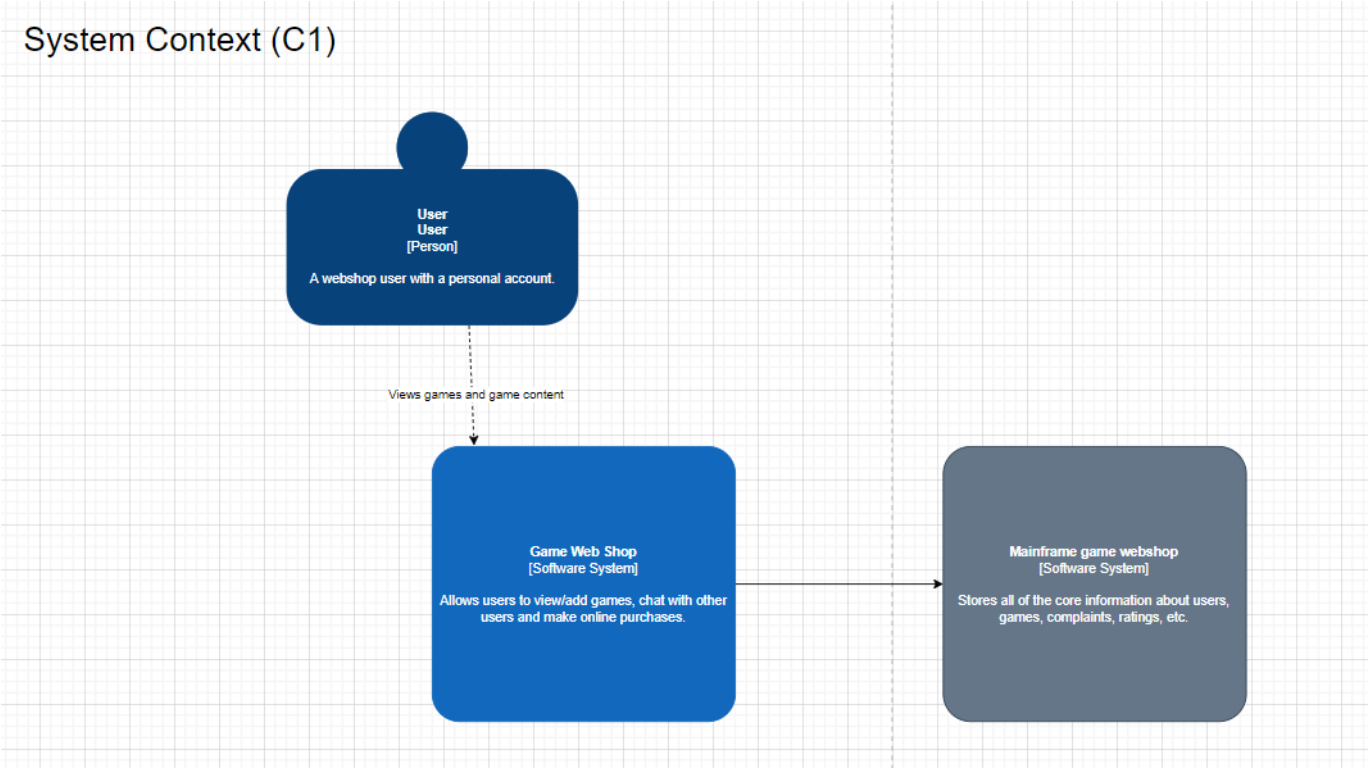
A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organised in a way that supports reasoning about the structures and behaviors of the system.

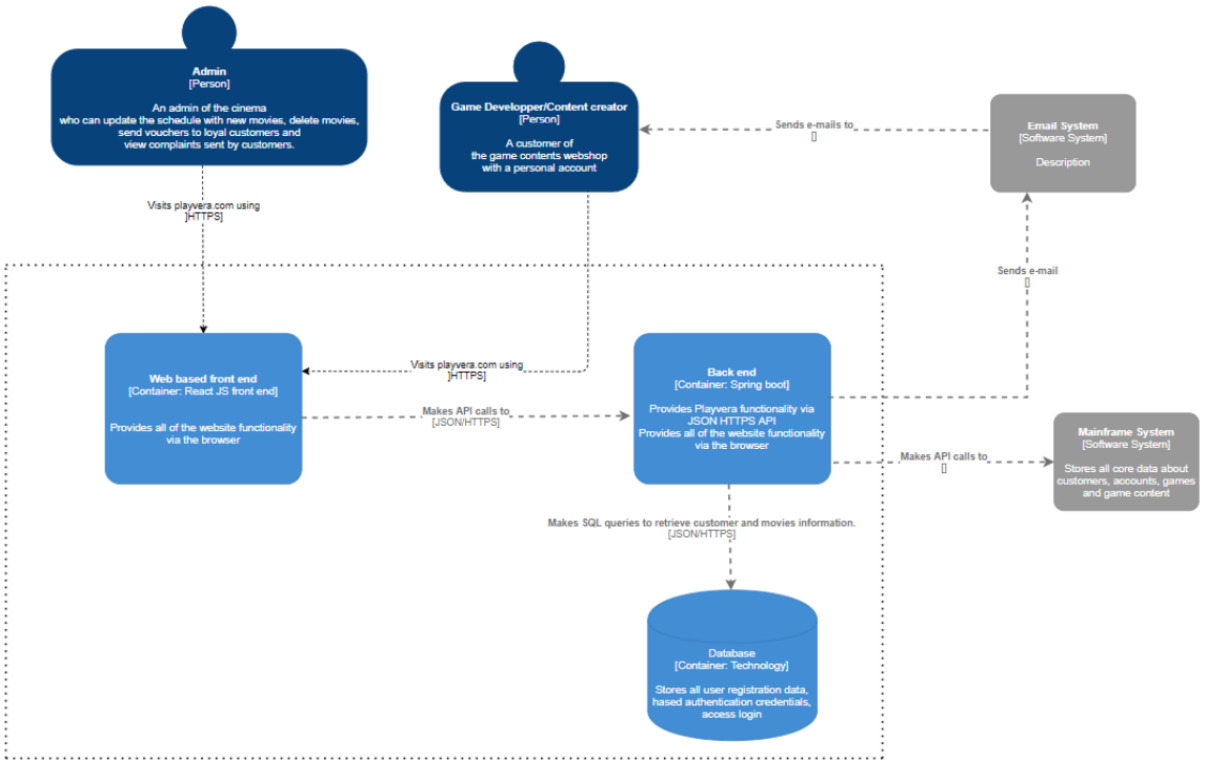
3.1 Architectural Design

Playvera is developed by strictly following SOLID principles and ensuring that security is provided to the users. It uses interfaces to connect to the database layer, ensuring that there is no redundancy and repetition in the implementation. The backend of the application is connected with the front-end using HTTP requests with Axios to connect to REST endpoints, allowing the performance of CRUD operations.

3.2 CI-CD SETUP

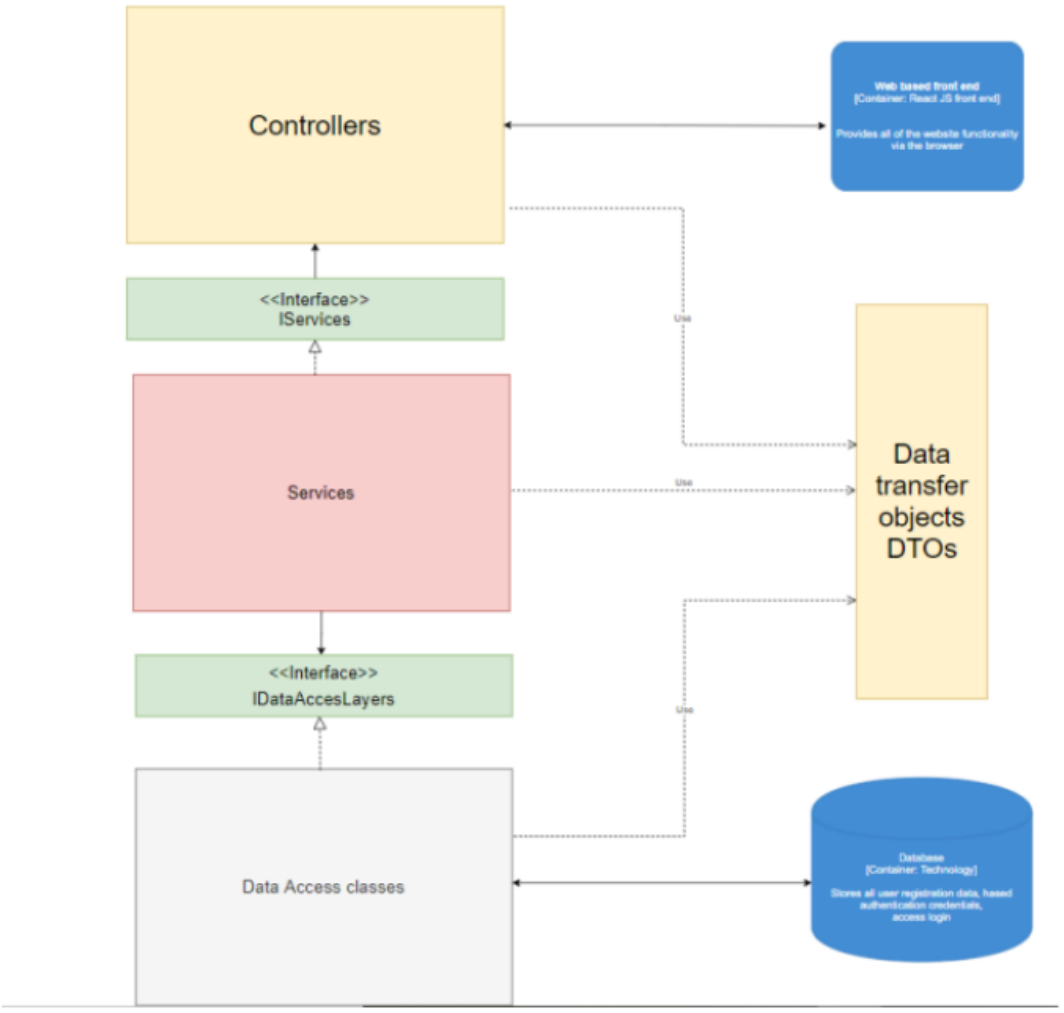
Level 1





Level 2 Component breakdown: Backend and front end connection between users and administrators: - REACT front end - it provides the UI and all of the website functionality within a browser environment. Users and administrators use the interface to interact with the back-end of the application - Back end - it provides all of the data and logical functionality of the website using springboot and Json https api. It makes the front-end functionality possible. - Database layer - it is used to store all user/content information and directly interacts with the back-end when a request for specific information is created (retrieving/pushing/updating/deleting information).

Level 3:



Level 3 Component breakdown: Controllers: Controllers are used to connect with the front-end (UI) of the application. The also function with the help of interfaces, allowing them for a flexible connection with the service components. Service component: Service components provide the logic for the controllers to use. They are directly connected to data access layers and can CRUD different data. Database: it is used to store all user/content information and directly interacts

with the back-end when a request for specific information is created (retrieving/pushing/updating/deleting information

3.3 Software design choices

- Security: One of the features that the website has inherited are Json web tokens for authentication. Currently, they are stored in the local storage of the user's browser and that could potentially lead to a security breach. Further measures applied are unique usernames and emails, encoded passwords and refresh tokens that expire after a certain amount and require the user to re-log in. ' The website runs on a role system, where each user has a unique identifier that allows him into different parts of the application. Such roles can only be given upon creation of an account or by the admin.

- Web Sockets: In order to bring in interactivity in web application, a chat system has been implemented that runs with the help of a websocket, which closes once users have left the room. It works responsively and with the help of some asynchronicity, making it so the messages aren't instant, but instead delayed by 0.3 seconds when sent.

- JPA: The application uses Springboot in parallel with JPA to do all of the backend with database connectivity. The users and other "objects" are presented to JPA as entities, which then stores them accordingly in pre-configured tables. Some custom queries are also present in places where JPA can't be used to do some of the functionality.

- Data display: In order to present the data in the back-end to the users, different methods are and will be used to improve the performance and interface of the front-end part of the application. Pagination is something that can be seen in the user tables of the admin site, however it is only being managed by the front-end, instead of the back-end. This is to be changed in the last release. Data can be seen displayed as tables and cards, each being implemented in places where it is more suitable to adapt to the needs of the user.

- Object Identification: UUID's are not used in the application, but instead –" Long" numbers. This is because after a short research on the differences between the two and the pros and cons, the conclusion was made that UUID's, while non-objectively being more secure, just slow down the process by a tiny bit if there were to be an attack that guesses different id's. Furthermore, if the configuration is made in a way such as unknown users can't access different pages and their sub-content, it breaks the point of having them. Another con is that testing UUID's is extremely difficult and hard to work around.

- Testing: Testing the application is a crucial part of its development and ensuring the functionality. That being said, 3 different variations of testing have been applied: - Integration testing with postman - Backend unit testing - Front-end cypress testing

- Data validation: To prevent data duplication and manipulation, unique username and email is required upon registering into the application. If the user tries to use the same username as someone else, it sends out an error message that is given by JPA from the back-end. Furthermore, passwords are required to have a specific length. If a password is forgotten, the user should be able to request a new one with an email with a reset link being sent to them. This is to be implemented in the future.

- **Usage of DTO's:** The application uses DTO's for two of the main components: the users and the game objects in the system. There are other objects (such as TypeGame) that do not need the application of DTO's. The reason behind this is that they do not contain any sensitive data that could be breached/modified. They are simple collections of enum values that may be exposed to the public. The reason behind using DTO's is to ensure that any sensitive data is not shown to the front end, preventing anyone from potentially using it in a malicious way.

4.0 DATA DESIGN

4.1 Data Description

All of Playvera's data is stored in a database that contains the following data tables:
-AppUser: Containing all of the user's information(email, password, username, id)
-Game: Containing all of the necessary information of a user created game
-TypeOfGame: Containing all of the types of games in the web application
-Role: Containing all of the roles a user may have in the application (admin, user)
-GameRating: This is used to connect a game and its rating. Once a game is deleted, the game rating linked using its' id is also automatically removed.

5.0 COMPONENT DESIGN

In this section, we take a closer look at what each component does in a more systematic way. Playvera's functionality and way of development is done using OOP.

The following are objects that reside in the backend functionality of the webshop:

AppUser – An app user consists of an id, username, email, password and a collection of roles(admin rights, game developer rights and user rights)
Game – A game consists of id, game name, game type, size, rating and price.
Role – The roles are given to different users, providing them access to the application using HTTP security.
TypeOfGame – The type of game object is assigned to a game. A game may have multiple types assigned to it.

The security component of the application is done using the HTTP security class in Java. It consists of two sections: ***1. A custom authentication filter*** *that ensures a proper JWT token is given to the right user, creating a security layer.*
2. A custom authorization filter *that makes sure the proper token is handled correctly and a user may login using his credentials securely.*

6.0 HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

The user interface of the application is fully developed using React, using JavaScript with HTML and CSS For the styling. It consists of custom built components that represent the following functionalities: Navigation bar: Used for website navigation and connection to all of the other visual components in the application Admin panel: Used for retrieving user information, deletion of users and correcting information/general statistics Game component: Represents a game where a picture may be uploading, pricing, size and a short description. A user may also view the rating of a game. Comments component: Represents user-written comments that may be filtered and deleted by the administrators of the application.

Playvera About us All Games Add your own game Chat My Games All users Your account Sign out

A place for indie game developers

Find indie titles or create your own

Get Started

At Playvera, we aim to create a community where developers can create and share indie games, upload different assets and help each other with the path of creating a game.

Playvera About us All Games Add your own game Chat My Games All users Your account Sign out

Available games

Search

Mario

A true classic, enjoy the beauty of jumping on goombas and saving the princess!

Adventure

Like the game?

Rust

Rust - naked, scared..and a rock in your hands?

Survival

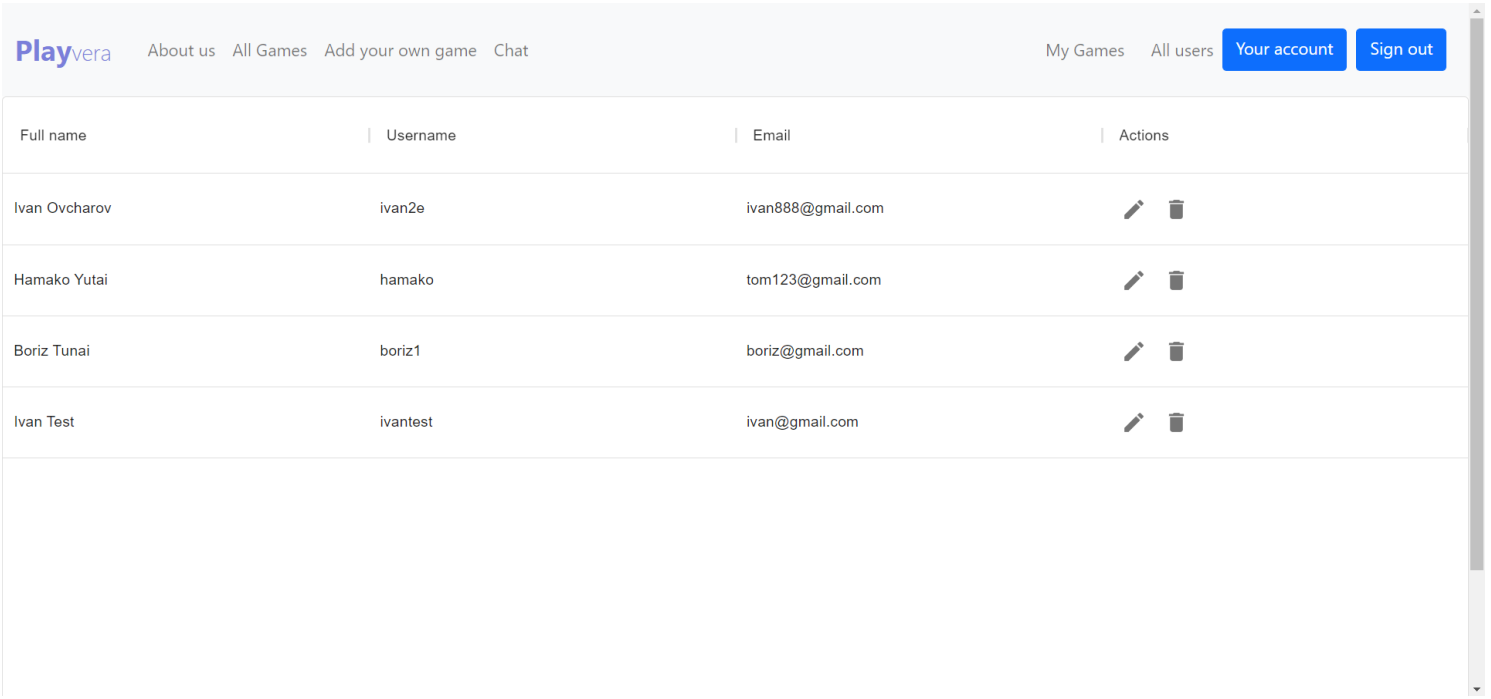
Like the game?

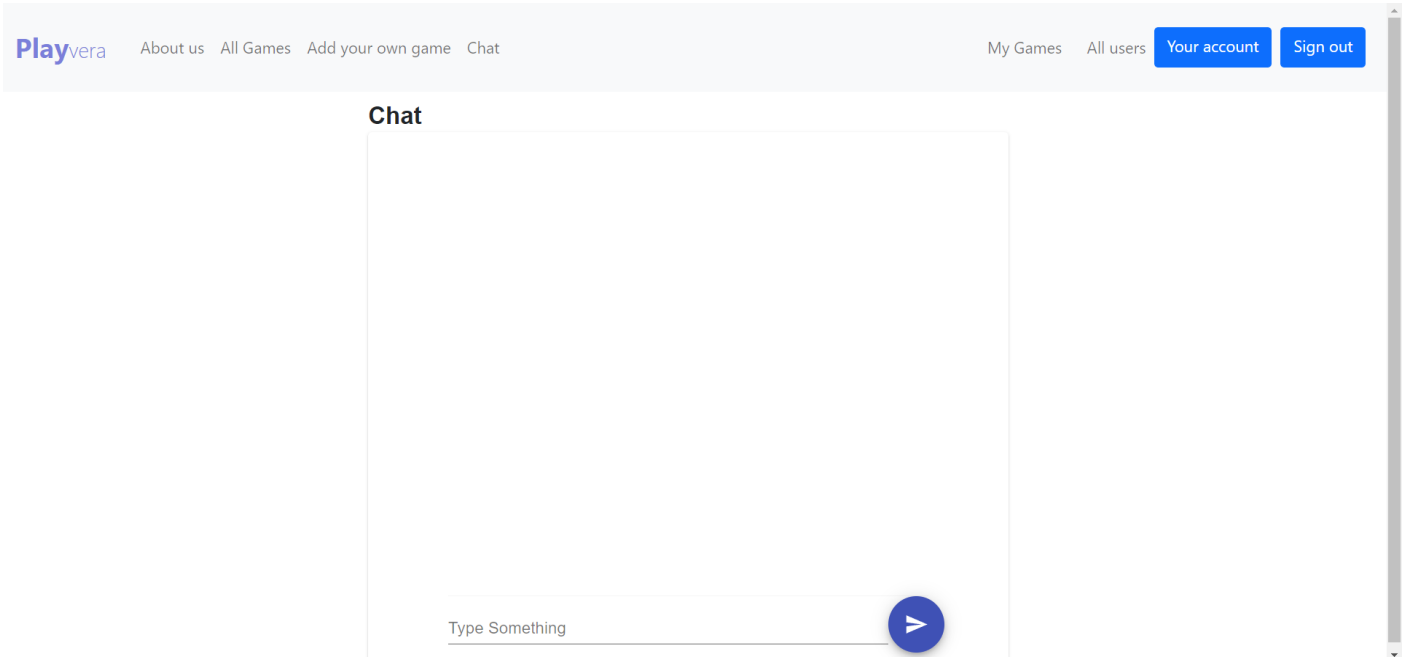
Zelda

For sword lovers, this game will let you embark on epic quests in the land of Zelda!

Adventure

Like the game?


Playvera
About us
All Games
Add your own game
Chat
My Games
All users
Your account
Sign out
Full name
Username
Email
Actions
Ivan Ovcharov
ivan2e
ivan888@gmail.com
Hamako Yutai
hamako
tom123@gmail.com
Boriz Tunai
boriz1
boriz@gmail.com
Ivan Test
ivantest
ivan@gmail.com

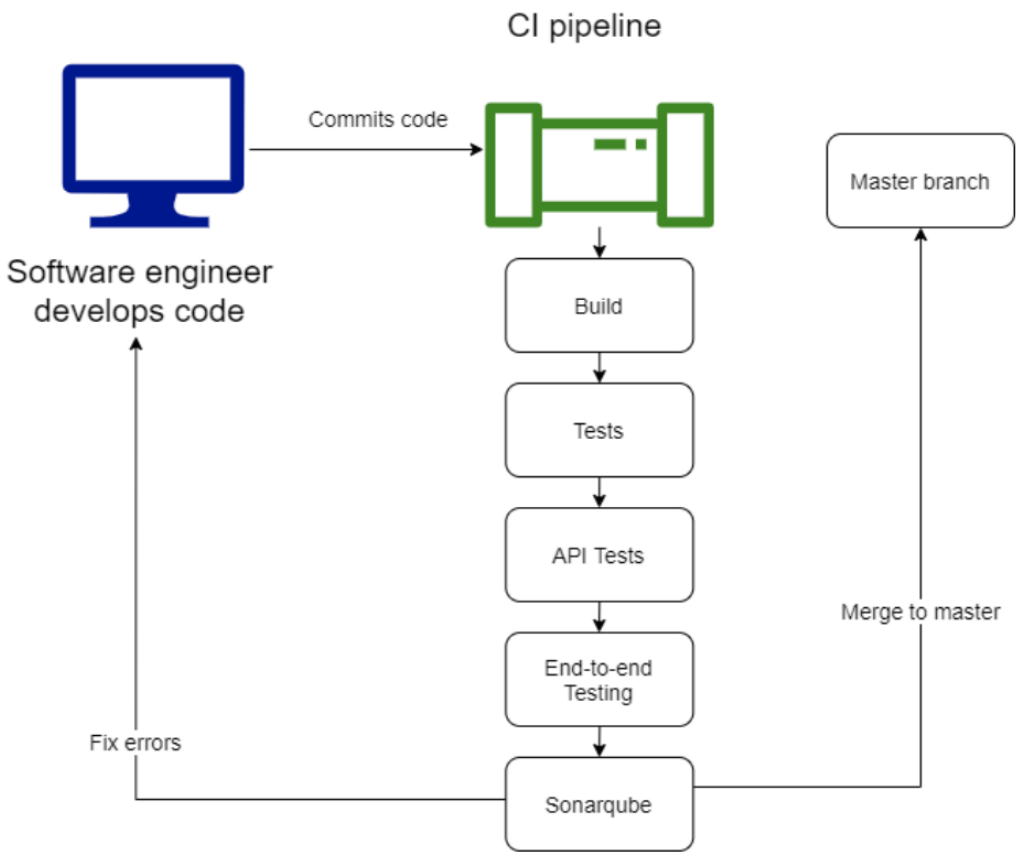

Playvera
About us
All Games
Add your own game
Chat
My Games
All users
Your account
Sign out
Chat
Type Something

7.0 CI-CD Pipeline Setup

7.1 What is CI-CD?

Continuous integration (CI) and continuous delivery (CD) embody a culture, set of operating principles, and collection of practices that enable application development teams to deliver code changes more frequently and reliably. The implementation is also known as the CI/CD pipeline.

7.2 CI Pipeline Diagram



7.3 Project implementation

CI-CD Pipeline is integrated with sonarqube in order to have a clearer view of how testing works and manage to fix errors with greater ease. It also allows for running different statistics and tests on your application/code to find possible data/security leakage and prevent further mistakes Furthermore, cypress has been added to the frontend part of the application to test end to end components.The entire diagram is also wrapped in a Docker container that finalizes the whole CI/CD process.