|  |  |
| --- | --- |
|  | Design Document  **Online Casino djigitBET** |
|  |  |
|  | by Radoslav Radev  ICT & Software Engineering semester 3  9/10/2022  ver. 2.0 |

# Document Versioning

|  |  |
| --- | --- |
| Date | 14/09/2022 |
| Version | V1 |
| State | Defined (Initial Draft) |
| Author | Radoslav Radev |

# 

|  |  |
| --- | --- |
| Date | 22/11/2022 |
| Version | V2 |
| State | Semi Completed |
| Author | Radoslav Radev |

Contents

[**Document Versioning**](#_heading=h.dlf3z4kkul37) **1**

[Introduction](#_heading=h.1fob9te) **3**

[General Information](#_heading=h.3znysh7) **3**

[Architectural Approach](#_heading=h.2et92p0) **3**

[Project Methodology](#_heading=h.tyjcwt) **3**

[Frameworks](#_heading=h.3dy6vkm) **4**

[Feature Prioritization](#_heading=h.1t3h5sf) **4**

[**Quality Assurance**](#_heading=h.qop2dldv9ste) **4**

[User Stories](#_heading=h.4d34og8) **5**

[**Technical Architecture**](#_heading=h.yecd81du28j2) **10**

[API URL tables](#_heading=h.17dp8vu) **10**

[**How is SOLID guaranteed?**](#_heading=h.cnos74vbl0mg) **11**

[**Framework selection:**](#_heading=h.hya7gv6ney1o) **11**

[**C4 Diagram**](#_heading=h.qxx4o2i72w1q) **12**

# Introduction

# General Information

The application's technical and architectural components are the primary emphasis of this Design paper (online Casino).

The document contains things such as:

* *User stories*
* *RESTful API design tables*

# Architectural Approach

Object-oriented design based on SOLID principles will establish a Backend as a Service application, fully separated from the front end

# Project Methodology

The Methodology used in this project is **Agile**.

*Agile methodology:*

*“a software development and project management method that is continuous, and aids teams in delivering value to clients more quickly and with fewer hassles”*

# Frameworks

Frameworks used in this project are as follows:

* ***Java & Spring Boot*** *- RestAPI*
* ***JPA - ORM***
* ***MySQL -*** *database*
* ***React & MUI*** *- front end*

Project is going to be continuously deployed using **CI Pipelines** and **Docker Compose**.

# Feature Prioritization

To understand which features are more important than others, a prioritization system is implemented/included.

*It goes as followed.*

* ***MUST – “Feature must be implemented. It has the highest priority.”***
* ***SHOULD – “Feature should be implemented if everything goes smoothly and by schedule. It has medium priority.”***
* ***COULD – “Feature could be implemented if there is time upon finishing the project. It has lowest priority.”***

# Quality Assurance

Quality of the project is going to be assured through test-driven development as follows:

* *Unit tests*
* *Integration tests*
* *End-to-end tests*

# User Stories

A user story is a comprehensive explanation of a software feature written from the viewpoint of the client or end user.

Difficulty of the user stories will be measured on a scale from 1-10

*User story I*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| Register | **MUST** | 8 |
| As a customer,  I want to Register at the website.  So that I can gamble. | | |
| Acceptance Criteria | | |
| Given provided information,  When the Register button, in the Registration component of the website, is pressed  The system:   * Creates an account * Redirects to the home page in case of success   No/Insufficient information entered:   * error message displaying which data is field incorrectly | | |

*User story II*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| Log-In | **MUST** | 8 |
| As a customer,  I want to Log-In to the website  So that I can gamble. | | |
| Acceptance Criteria | | |
| Given provided information,  When the Log-In button, in the Login component of the website, is pressed  The system, depending on the type of the account:   * Logs in the user and redirects to the home page in case of success   No/Insufficient information entered:   * display and error message | | |

*User story III*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| Profile update | **MUST** | 4 |
| As a customer,  I want to update my Account information,  So that my information stays safe, secure, and up to date. | | |
| Acceptance Criteria | | |
| Given provided information,  When the Update button, in the Profile component of the website, is pressed  The system:   * Updates the information in case of success   Insufficient information entered:   * error message displaying which fields are incorrect | | |

*User story IV*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| Account deletion | **Could** | 2 |
| As a customer,  I want to delete my account,  Because I do not need the services of the online casino anymore. | | |
| Acceptance Criteria | | |
| When the Delete my Account button, in the Profile component of the website, is pressed  The system:   * Deletes the account after a confirmation pop-up | | |

*User story V*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| Add funds via a payment system | **Could** | 8 |
| As a customer,  I want to add funds to my Account  So that I can gamble them on the slot-machine. | | |
| Acceptance Criteria | | |
| When the Amount is specified and confirmed by a button, in the Funds component the system redirects to a payment provider and confirms the transaction  The transaction failed:   * funds are not added to the account | | |

*User story VI*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| Slot Machine | **MUST** | 10 |
| As a customer,  I want to try my luck on the slot-machine | | |
| Acceptance Criteria | | |
| When the spin button is pressed on the slots component of the website:  The algorithm decides on giving a prize to the customer,   * the amount betted by the user is substracted from thair account * if a prize is won the funds will be added to the customer`s account | | |

*User story VII*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| Modify User Account | **MUST** | 3 |
| As an Administrator,  I want to Update Delete and Review all aspects of a user`s account with a usefull search engine | | |
| Acceptance Criteria | | |
| When text is entered in the search box of the User\_anengment Componet, the system:   * Refines the presented data according to the search criteria   When the Edit button next to a user account is pressed in the User-Manengment Componet, the system:   * presents the user with a form in which he can modify the information and confirm it   When the Deletebutton next to a user account is pressed in the User-Manengment Componet, the system:   * the user can confirm the deletion of the account | | |

*User story IX*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| view Statistics | **MUST** | 2 |
| As an Administrator,  I want to view statistics on profit and winCoeficent by user bases, an overall win rate of the slot machine | | |
| Acceptance Criteria | | |
| The system presents the required statistic data with filter options in the Statistics component of the website | | |

*User story x*

|  |  |  |
| --- | --- | --- |
| Tittle | Priority | Difficulty |
| Customer Chat | **MUST** | 5 |
| As a customer i want to communicate to other players via chat | | |
| Acceptance Criteria | | |
| The system provides the user with a realtime chat | | |

# 

# Technical Architecture

# API URL tables

This section of the paper contains API tables per user story.

|  |  |  |  |
| --- | --- | --- | --- |
| URL | Resource | Operations | Description |
| /users/ | User | GET | get a collection of all users |
| /users/$id | User | GET | get the specified user |
| /users/ | User | POST | add a new user account to the system |
| /users/$id | User | PUT | Update the contents of the account object |
| /users/$id | User | DELETE | Delete the selected account |
| /authenticate/login | User | POST | Login,generate JWT |
| /authenticate/register | User | POST | Create a new Player account |
| /unpriviligeduser/ | User | PUT | get the user who is logged in this unprivileged account |
| /payment | string{confirmation of the payment} | POST | generate a payment intent |
| /slots/ | slotCalculationsDTO | POST | place a bet |
| /slots/jackpot | double | GET | get the current jackpot |

# How is SOLID guaranteed?

S– Every class has only one responsibility.

O– Classes are open for extension but closed for modification

L– If class A is a subtype of class B, we should be able to replace B with A

I – Proper use of interfaces and abstract classes

D– Dependency injection and inversion are by the Spring Been system

# Framework selection:

The frameworks used in this project are as followed:

* Java with Spring Boot and JPA, is used for the Rest API,
* JPA ORM is used to manage the database

- MySQL database

- React is used for the presentation layer of the application (FE)

**Why did I use these technologies?**

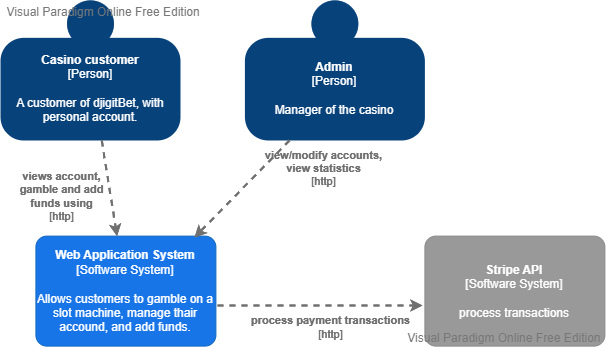
React - is the most popular front-end framework and it is recommended as the first one to learn, although the industry is moving towards Svelte it`s not recommended for beginners

Java and spring boot - it is a main requirement of the curriculum.

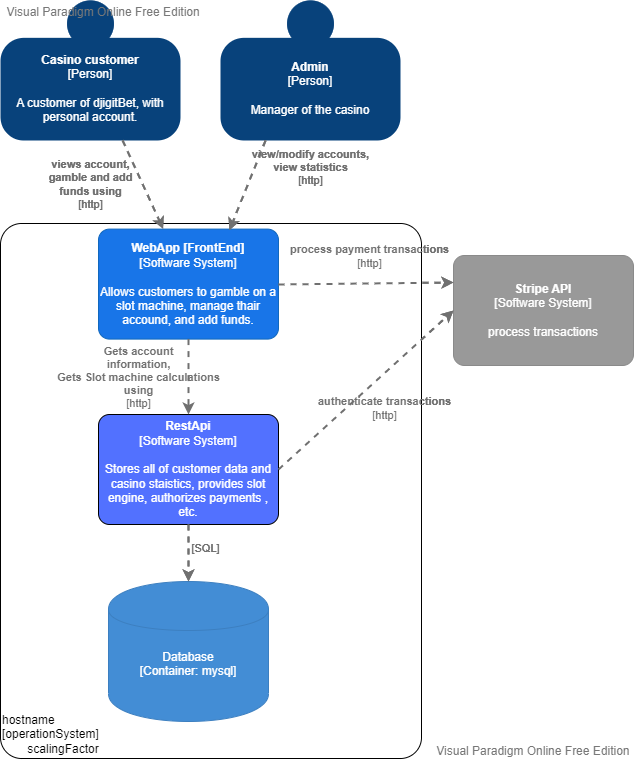
* I would have preferred to do the restAPI service with ASP.NET WebAPIs in the dot.net ecosystem, with the comfort of modern developer features, more stable code, and overall ease of use and support because:
* As one of the main high-level OOD-oriented languages, java is missing a lot of features
  + Disadvantages: poor garbage collector, no null safety, less maintained
  + Main features I miss from dot.net: parallel iteration, Span, LINQ, primitive type integration (e.g String), modern syntax
* Spring-Boot encourages taking the control out of the developer and leaving it up to chance in a lot of cases, but on the opposite side requires a large amount of configuration code,
* documentation, is poor and version mismatched,
* SpringBoot it`s hard to debug and produces a lot of bugs
* JPA-hibernate lacs features in comparison with Entity Framework
* dot.net by default uses Internal-Language compiler which is similar to the java VM, but much more flexible and faster. Furthermore, dot.net introduced Native AOT(ahead-of-time) compilation which uses even less memory and it`s even faster (compiles to native machine code for the selected architecture)

# C4 Diagram

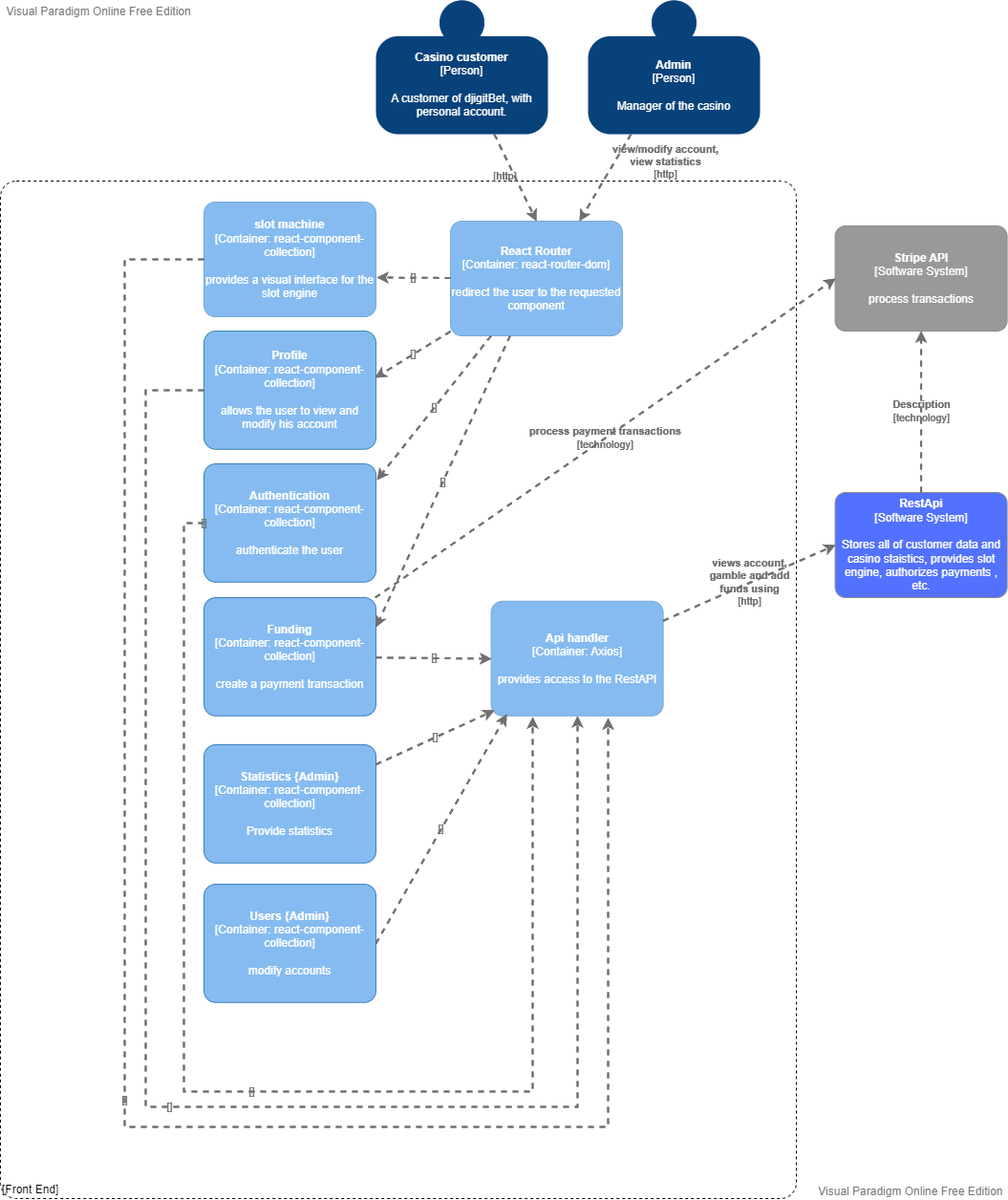
* + 1. **C1-Context**



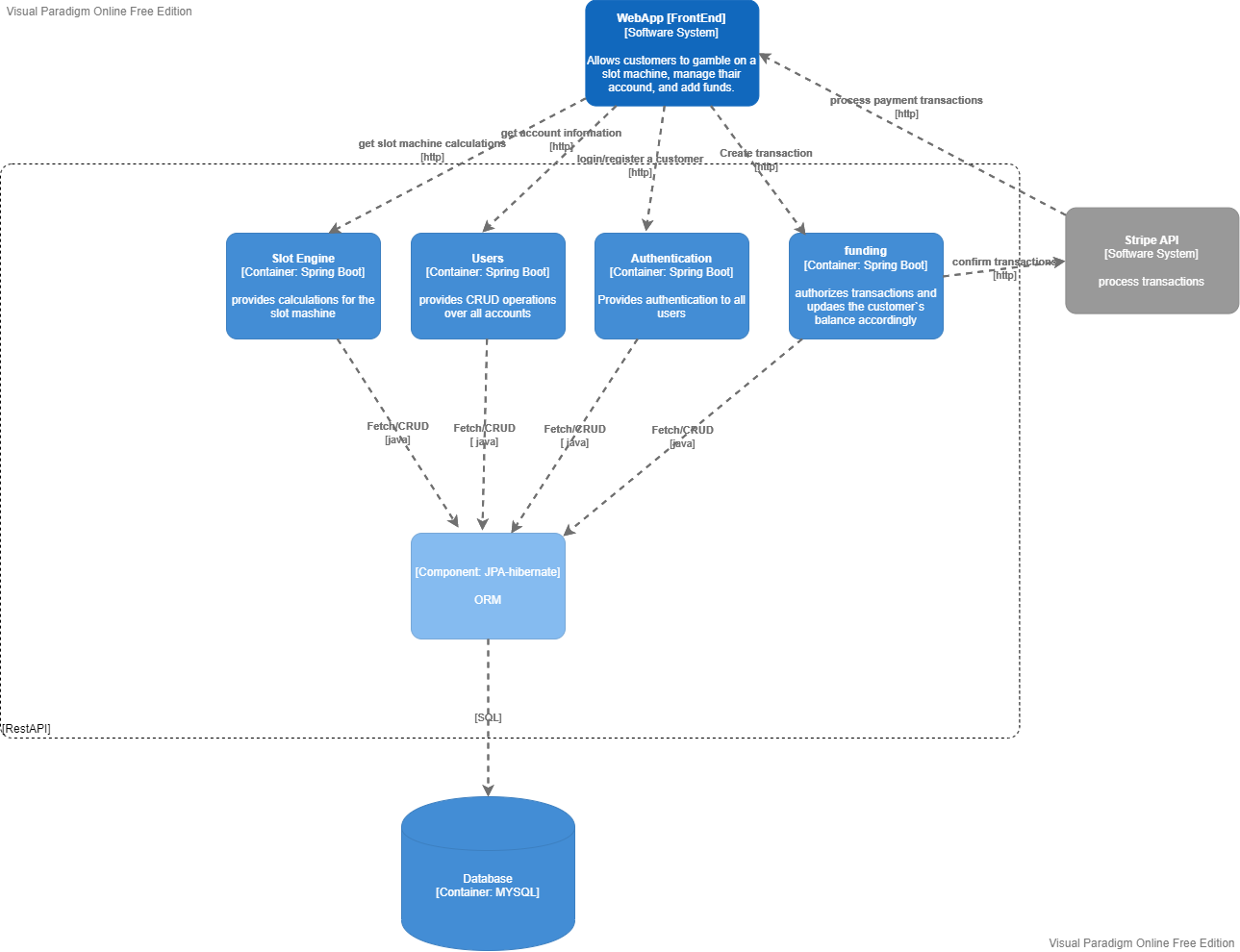
* + 1. **C2-Context**



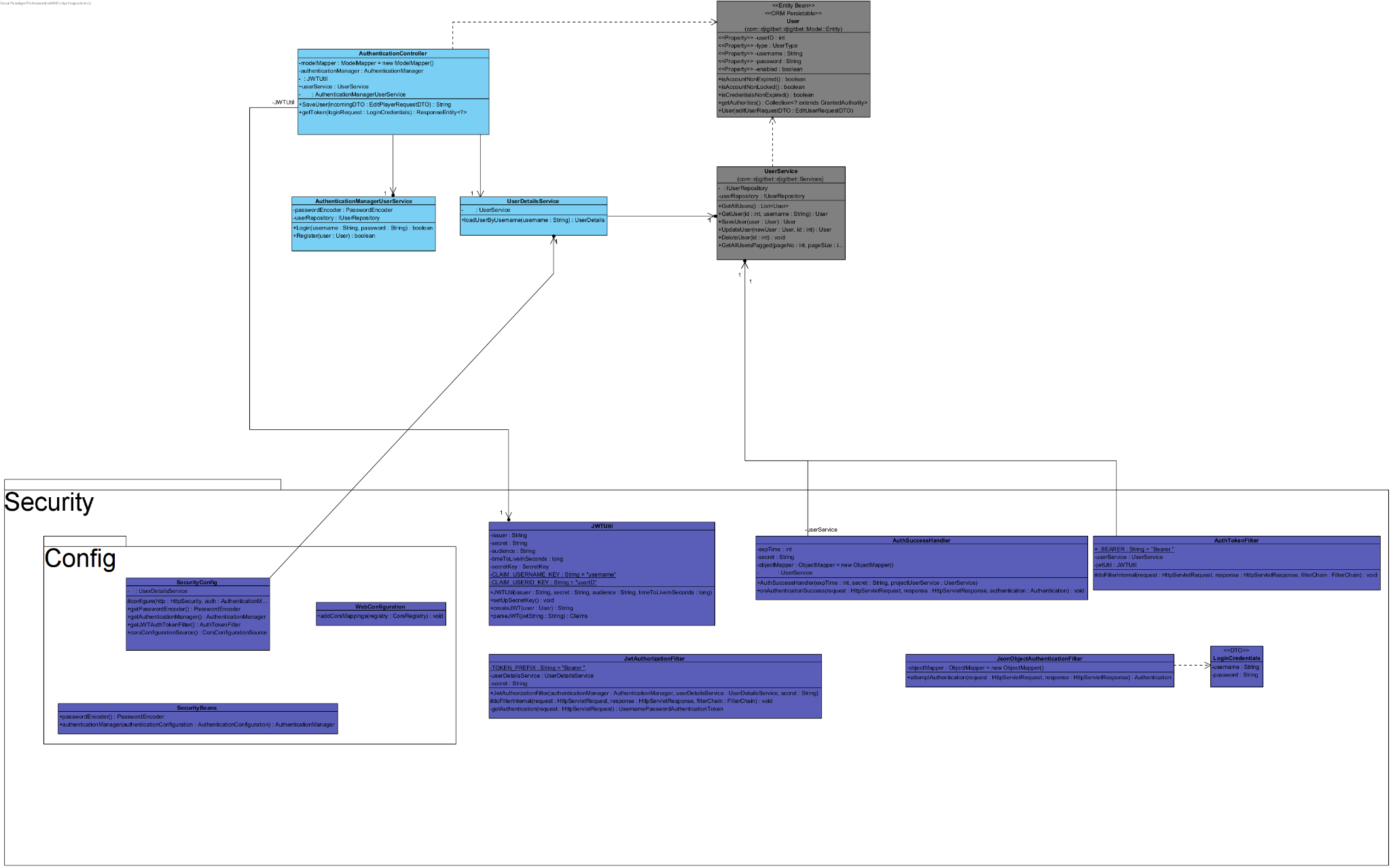
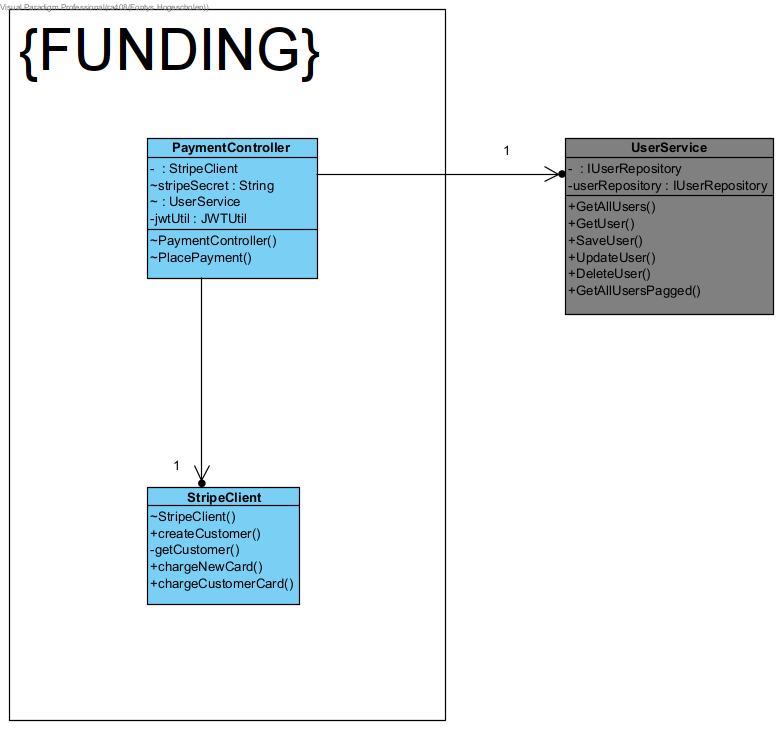
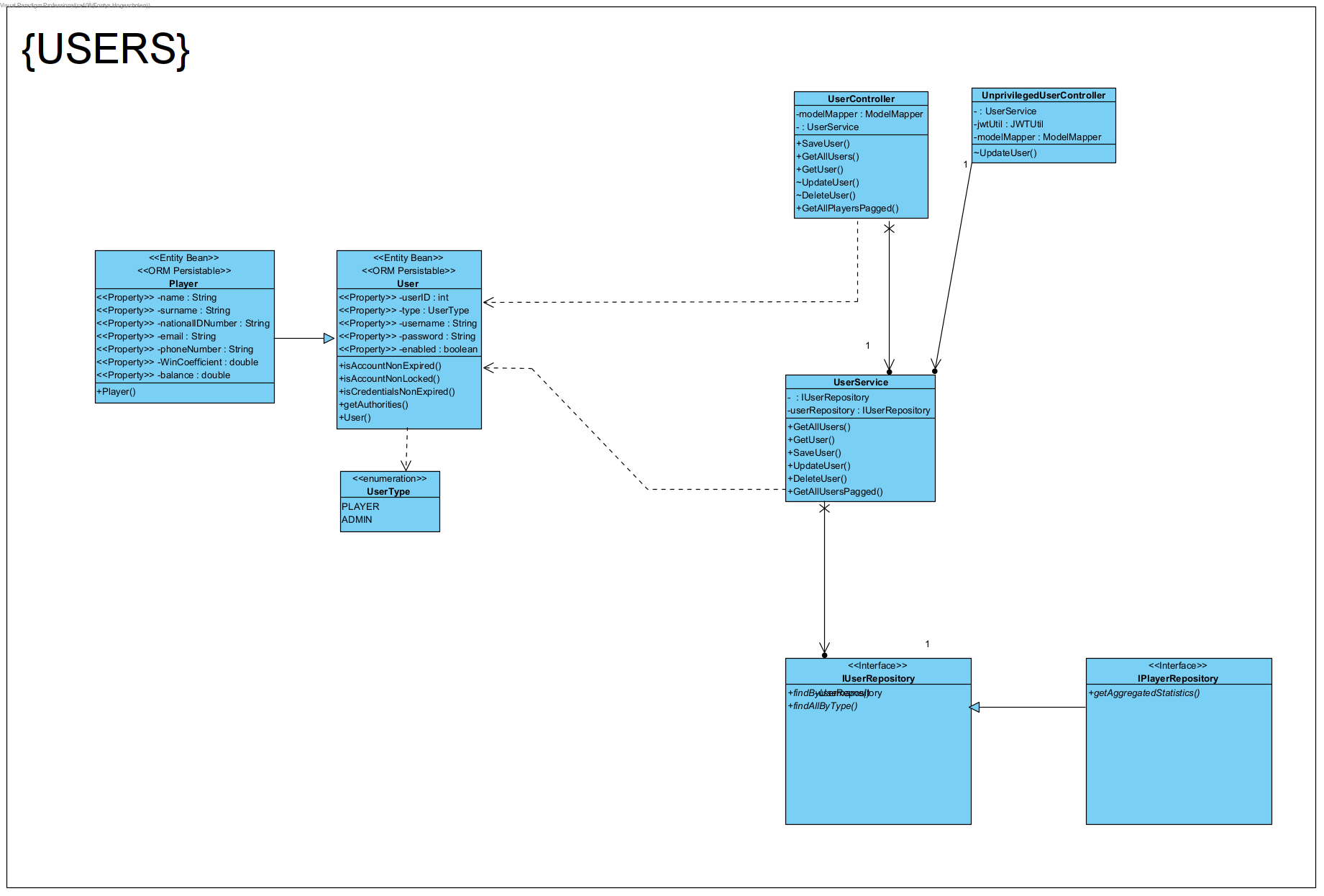
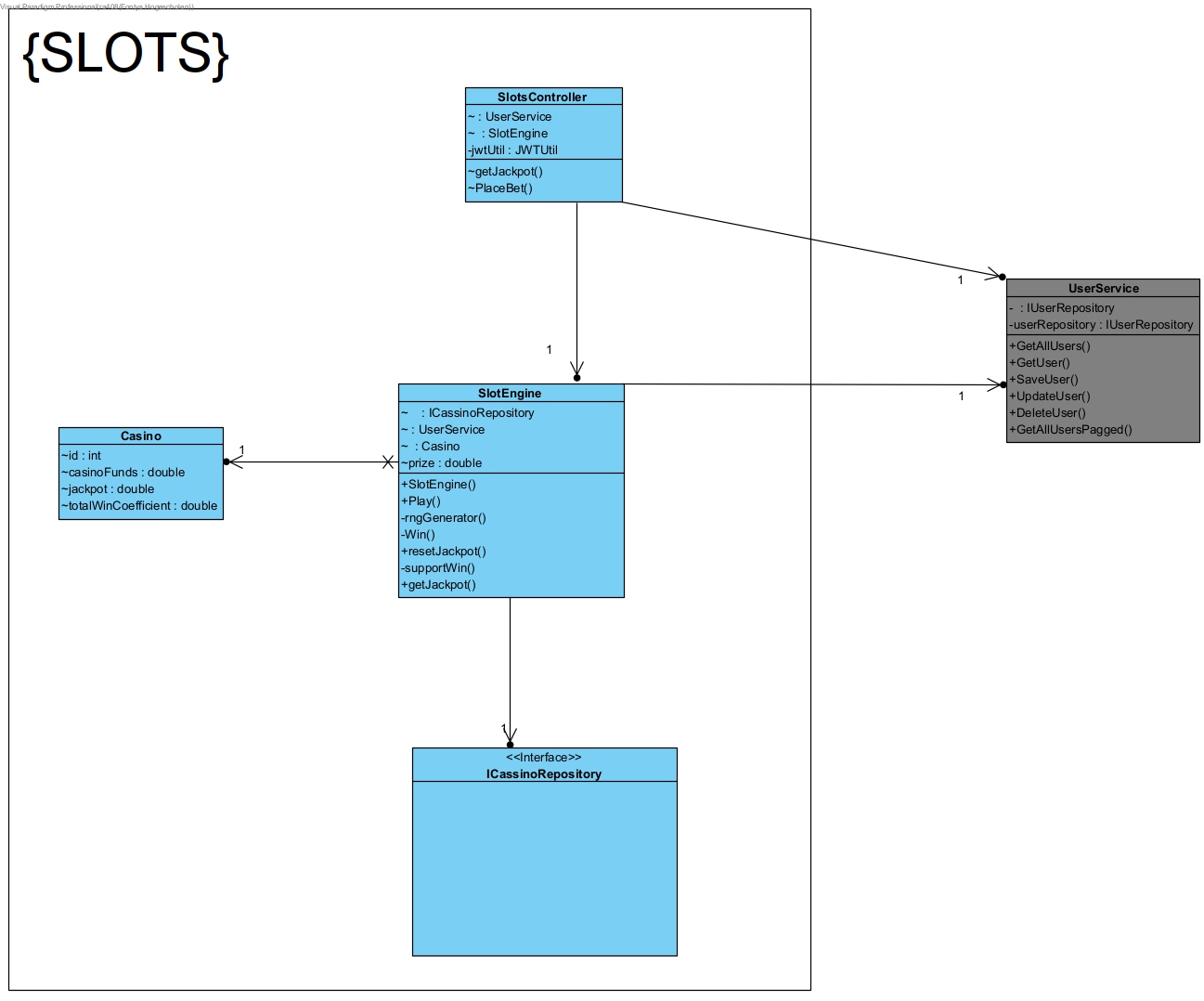
* + 1. **Zoom into the WebApp[Front End] [Software System]**



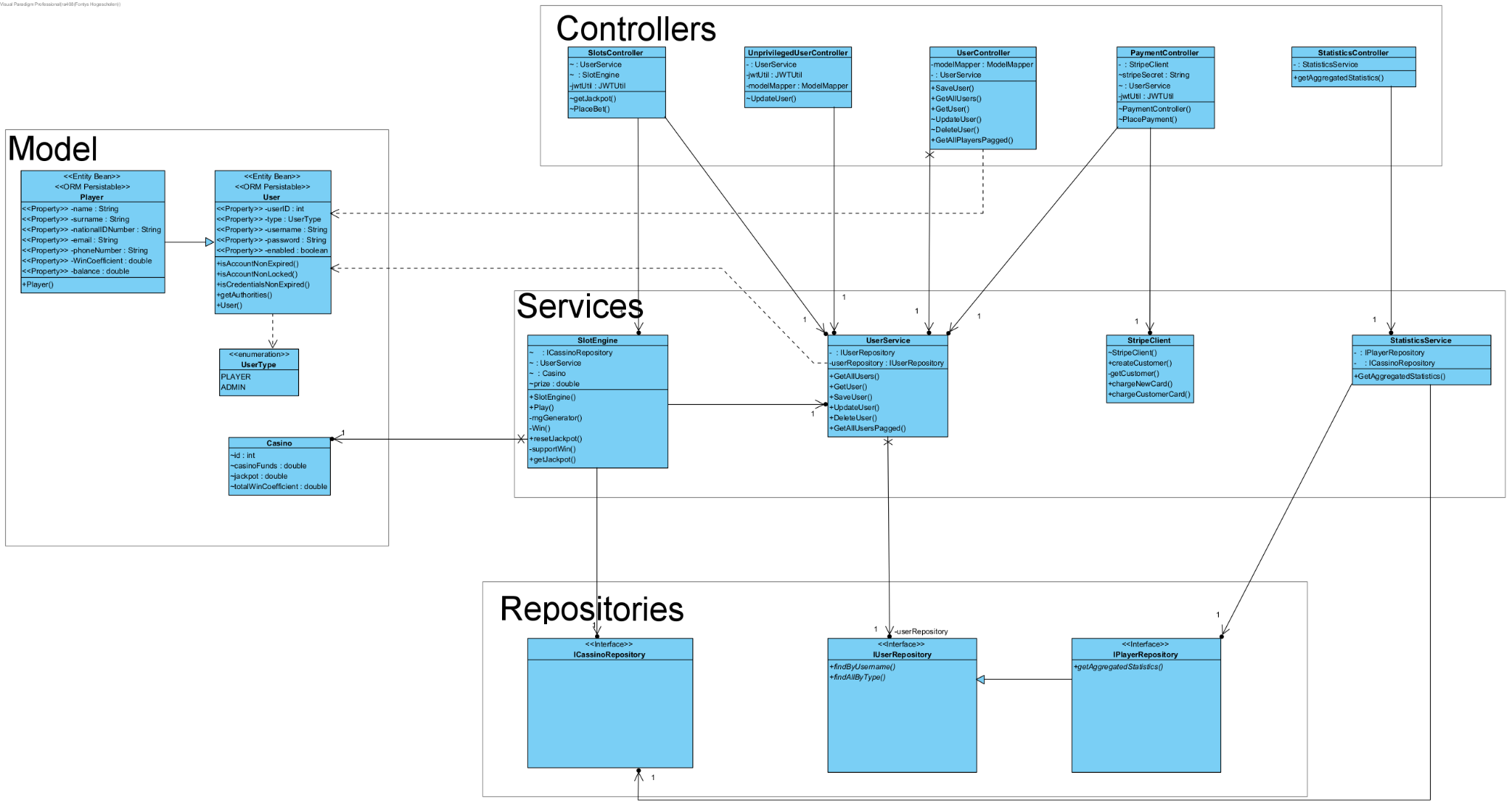
* + 1. **Zoom into the RestAPI[Softwere System]**



* + 1. **Zoom into Components**



* + 1. **Combined Layer ClassDiagram [BackEnd]**



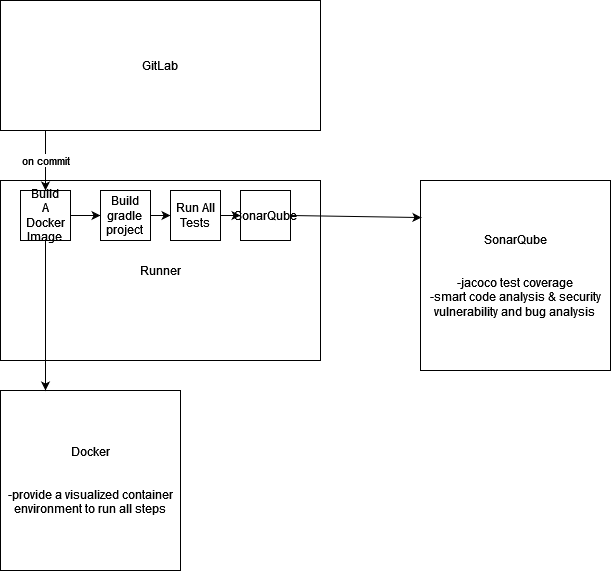
# 

# Continuous Deployment

The continuous deployment goes as follows:

* Features are developed in the order of Backend to Frontend
* Upon finishing a feature, it is pushed to GitLab where test pipelines are setup
* After the features are finished and unit tests are completed the project is deployed with Docker
* After the deployment, the project goes into the staging environment where integration tests and SonarQube analysis are performed
* Finally after the staging environment the project is pushed to production

*A flowchart explaining the process:*

**