Design document for IAS

BluDevil digital marketplace



Members: Alexandru, Emilian E.M.

Tutors: Geurts, Jaap J. ; Guayrin, Brice B.P.B.J.P.

Version: 1.2

# Table of Contents:

[1.0 Entity relation diagram 3](#_Toc69499754)

[1.1 The product section 3](#_Toc69499755)

[1.2 The user section 4](#_Toc69499756)

[1.3 The order section 4](#_Toc69499757)

[2.0 Data persistence justification 5](#_Toc69499758)

[3.0 UML diagram 6](#_Toc69499759)

[2.1 Controllers section 6](#_Toc69499760)

[2.2 Services section 7](#_Toc69499761)

[2.3 Services and repositories section 8](#_Toc69499762)

[2.4 Models section 9](#_Toc69499763)

[4.0 Design decisions 10](#_Toc69499764)

[4.1 Back-end: 10](#_Toc69499765)

[4.1.1 Spring boot: 10](#_Toc69499766)

[4.2 Front-end: 11](#_Toc69499767)

[4.2.1 Frameworks criteria 11](#_Toc69499768)

[4.2.2 Frameworks 11](#_Toc69499769)

[4.2.2.1 Vue 11](#_Toc69499770)

[4.2.2.2 React 11](#_Toc69499771)

[4.2.2.3 Angular 12](#_Toc69499772)

[4.2.3 Framework conclusion: 12](#_Toc69499773)

[4.2.4 Bootstrap 12](#_Toc69499774)

[4.2.5 Font Awesome 13](#_Toc69499775)

[4.0 Wireframes 14](#_Toc69499776)

[4.1 Navigation & Product catalog 14](#_Toc69499777)

[4.2 Product details 15](#_Toc69499778)

[4.0 Quality assurance metrics tool results 15](#_Toc69499779)

[4.1 Initial test results: 16](#_Toc69499780)

[4.2 Last test results: 16](#_Toc69499781)

# 

# Diagram, Word Description automatically generatedEntity relation diagram

The major parts of the ERD diagram can be summarized into three sections:

## 1.1 The product section

The main table for storing the product information is the “Products” table. It contains the common fields that each type of product shares and the primary key as “product\_id”, used in other tables for reference. One of the common fields, the “platform\_id”, stores as a foreign key the primary key of the “ProductPlatform” table, representing the many to one relationship as many products have exactly one platform.

The cdkey table, used to store the cdkey id and sequence, has the field “product\_id” as a reference for the many to one relationship, many products have one cdkey.

Currently, there are only two types of products video games and software products, each type has its table and contains the “product\_id” as their primary key which is also a foreign key.

The video game table has a many to many to relationship to the genres table, many products have many or one genre. The relationship is represented in the “Product\_Genres” table which has as fields the “product\_id” and the “genre\_id” as foreign keys.

The limitations for the products are that once a product has been persisted its type cannot be changed and to persist a product its platform and genres need to firstly exist in their tables. For persisting a cdkey its product must first exist in the “Products” table.

## 1.2 The user section

The main table of the user section is the “Users” table which contains the user information like the first name, last name, email, and password as fields. The users' table has many to many relationship to the Roles table used to store each role a user can have. The relationship is represented in the “Users\_Roles” table that contains the primary key of the users the table and the primary key of the roles table as foreign keys and fields.

The limitations of the users are that to persist a user a role must first exist and that a user role cannot be removed.

## 1.3 The order section

The main table of the order section is the “Orders” table which stores the “order\_id” as the primary key, the creation date, and the total price as fields. The order table has a many to many relationship with the “OrderItems” table, many orders have many orders items, and a many to zero or many relationship with the “Users” table as an order has one user represented by the field “user\_id” in the “Orders” table. The order items relationship is represented in the “Orders\_OrderItem” table.

The “OrderItems” table contains the ordered product quantity and the “product\_id”. As relationships, it has a many to one relationship as many order items have exactly one product and a many to many relationship with the CdKey table as many order items have many cdkeys represented in the “OrderItems\_Cdkeys”.

The limitation of the order section is that to persist an order the user and the product from its order items must exist.

# Data persistence justification

For data persistence, I will be using Spring Data JPA as it will speed up the development time of the application, keep everything simple and remove a lot of boilerplate code. Spring Data JPA is an add-on for JPA, an abstraction over the Data Access Layer that uses JPA(Java Persistence API) and ORM implementation like Hibernate. Because it is based on a JPA specification it uses all its defined features, it enables the entity objects and their metadata mappings and it also enables the entity manager, responsible for persisting and retrieving the entities from the database, while keeping mostly everything under the hood.

Spring Data JPA also removes the need to write native SQL statements by providing a set of Interfaces(Repositories) that define query methods for dealing with data and for which Spring

automatically provides an implementation. The name of the methods declared in the repository will be converted to low-level SQL queries

# UML diagram

The following diagrams are for the back-end architecture. The major parts of the UML diagram can be summarized into four sections:

## 2.1 Controllers section

A picture containing text, sign, screenshot, several

Description automatically generatedThis layer is responsible for the communication with the client thru the means of endpoints and the request types. Each resource has its controller and each controller has at least one reference to a service interface.

## 2.2 Services section

The controllers will call this layer's objects to perform requests. Each service depends on an interface applying the dependency injection required by spring.

A picture containing timeline

Description automatically generatedThe service layer is responsible for retrieving, creating, and updating the models, performing application-specific logic and manipulations.

## 2.3 Services and repositories section

Diagram, timeline

Description automatically generatedThese are the abstractions that the service layer will call to get/update or perform other operations that the data needs. This layer will generally call the database and perform the requested operations. Each entity has a repository. Spring Data JPA is used for managing the entities so each abstraction extends from one of the JPA repositories type (JpaRepository, CrudRepository, PagingAndSortingRepository).

## 2.4 Models section

A picture containing text, sign, white

Description automatically generatedThe model section contains the model classes used by the other layers.

# 4.0 Design decisions

## Spring Boot: The Most Notable Features You Should Know ...4.1 Back-end:

### 4.1.1 Spring boot:

One of the frameworks that I choose for this project is Spring boot, a back-end framework based on Java.

**Some of spring boots disadvantages:**

* From a learning perspective, Spring boot doesn’t cover most of the details of Spring, so if you never worked with Spring before details like proxies, dependency injection and AOP you’ll have a harder time during troubleshooting or modifications
* Due to its automation you can very easily miss many concepts of springs ecosystems like Spring Security, Spring Integration etc..

**I prefer it because:**

* It is flexible as it allows configuring beans in multiple ways like XML, Annotations and JavaConfig. I mostly tends towards annotations as they offer the fastest and easiest solutions but as a second option or by necessity I would rely on JavaConfig.
* It makes developing spring-powered applications better by making Spring or Spring MVC easier to use with its auto configuration
* It provides starters, like the spring boot starter web with all the needed dependencies prepackaged for developing an application to expose RESTful services
* It is open-source and quite popular with developers offering good documentation
* It simplifies integration with JPA/Hibernate ORM
* It very compatible with most technologies as it supports NoSQL databases, Oracle, PostgreSQL, MongoDB and several other processes
* It utilizes dependency injection as Spring at its core it’s a dependency injection container

## 4.2 Front-end:

### 4.2.1 Frameworks criteria

The most relevant criteria’s are for me:

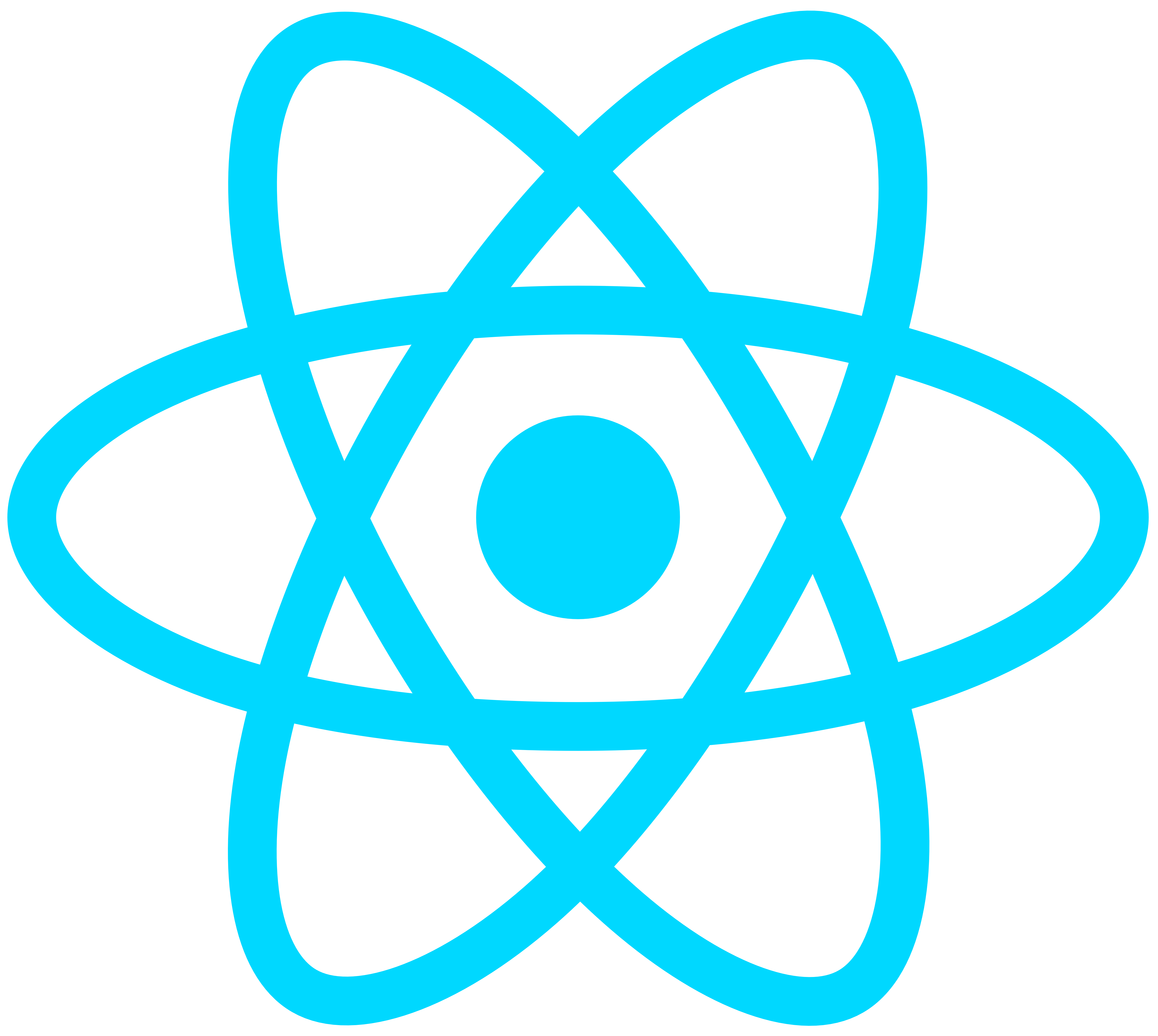
|  |  |  |  |
| --- | --- | --- | --- |
| Criteria | Angular | React | Vue |
| Support community | 9 | 9 | 7 |
| Opinionated | 9 | 6 | 6 |
| Documentation | 9 | 8 | 7 |
| Ease of learning | 6 | 7 | 7 |
| Ease of use | 6 | 7 | 7 |

### 4.2.2 Frameworks

Although there are a lot of frameworks when it comes to the front-end I try to limit myself to one of the three recently popular frameworks: Angular, React and Vue.

#### VUE Logo PNG Transparent & SVG Vector - Freebie Supply4.2.2.1 Vue

|  |  |
| --- | --- |
| Pros | Cons |
| Empowered use of HTML  Circumstantial documentation  Adaptable and flexible  Good integration, can be used for both building single-page applications and more difficult web apps  Tiny size, can weight around 20KB  UI and behavior are part of components  One-way and two-way data-binding | Young framework  Lack of resources  Risk of over flexibility |



#### 4.2.2.2 React

|  |  |
| --- | --- |
| Pros | Cons |
| Use of JSX (an HTML-like syntax) for templating  Detailed documentation  Very fast due to the Virtual DOM implementation and various rendering optimizations  Great support for content-focused applications  Implements Functional Programming  Offers support also for typescript  Good size, around 43k  UI and behavior are part of components | JSX mixes templating with logic which can become confusing as some points  Data-binding is one-way only  React is unopinionated so most of the developing choices are up to the developer  React is moving away from class-based components |

#### Angular (web framework) - Wikipedia4.2.2.3 Angular

|  |  |
| --- | --- |
| Pros | Cons |
| Empowered use of HTML  Mature framework  Exceptional support for typescript  Allows intellisense and autocomplete inside of component external HTML template files  It comes with the Angular CLI  Detailed documentation  Two-way data-binding  Uses dependency injection  Complete setup on startup | Enforces the MVVM pattern  Steep learning curve due to the variety of different structures (Injectables, Components, Pipes, Modules etc.)  Relatively slower performance compared to the other frameworks  Is quite bloated, the zipped file size is around 143k  Is not as flexible and universal |

### Angular (web framework) - Wikipedia4.2.3 Framework conclusion:

In conclusion, Angular is my framework of choice because it exceeds the others regarding detailed documentation, community support, and is highly opinionated. As a framework, it provides an easy and complete startup by making use of the Angular CLI. Being the most “opinionated” Angular provides all the tools needed to make it easy to build my web application, including but not limited to state management, routing, and dependency management. The enforced MVVM pattern is advantageous as it provides a separation of concern. I want to have a better understanding and correct use of the pattern as this will be my first project tackling it.

Due to its variety of structures, Angular scores are lower in ease of use and ease of learning criteria. These structures will make the project more challenging but, after a deeper dive in understanding them, they will surely prove useful in the long run.

### Bootstrap 4 is released - SD Times4.2.4 Bootstrap

An increasingly popular and feature rich front-end design framework. Bootstrap focuses on the design aspect of the application UI enchanting the already existing elements that come with HTML5 but also providing lots of custom additions.

**Some disadvantages:**

* The styles are verbose and can lead to lots of output in HTML
* It increases loading time for the pages

**The main reasons for choosing bootstrap are:**

* It’s extremely easy to set up either by CDN or npm
* The speed of development is drastically increased, as bootstrap provides ready-made components which I can easily customize and build on top using CSS or jQuery scripts
* I am familiar with the framework as I have used it a bit in the past
* It uses jQuery for handling scripts which I am familiar with
* It has very good community support and documentation

### Font Awesome Icons | Drupal.org4.2.5 Font Awesome

Font Awesome is another design framework that I decided to use because of the big options of icons that it provides:

**Some advantages:**

* A very light design framework
* It’s extremely easy to set up and use either by CDN or npm
* It provides a lot of icons to choose from
* It is open source

**Some disadvantages:**

* Not all icons and variations of the icons are available in the free version

# 4.0 Wireframes

## 4.1 Navigation & Product catalog

## 4.2 Product details

Table

Description automatically generated

# Quality assurance metrics tool results

For the inspection of the code quality I’ll be using SonarQube which provides a detailed report of the bugs, code smells, vulnerabilities and code duplications inside the application. The tests are usually runed on each commit and addressed in the next one but sometimes tests might be runed locally during development to ensure a bug free code. Based on the results run so far the applications the remaining main issues are related to the security, as persistence entities are used as arguments for the request mapping, the high number of code-smells, the low code coverage by tests and the little amount of comments. So far I have tackled the bugs, lowered the high number of code-smells and the number of duplications.

## 4.1 Initial test results:

## 4.2 Last test results: