Architecture

Text

Description automatically generated with medium confidence

24/03/2022 Eindhoven

Version: 0.1

Members:

Lars Kluijtmans: 4220269

Intro

Version controlee

index

How is SOLID guaranteed

* Single responsibility
* Every class is responsible for only one type of object, in this application the only typed of objects are users and products, although these are connected different classes are in charge of managing them.
* Open/closed principle
* By using interfaces, I can, if needed just extend from that interface to another class without having to modify any of the already existing code.
* Liskov substitution
* In this application only the user object implements this principle, it has 2 subclasses, NormalUser and Admin.
* Interface segregation
* At the moment this principle is not used in the application.
* Dependency inversion

-In between every layer of this application (controller, service and repository) there is a interface separating them.

Important design decisions

* Why user spring boot
* Spring Boot helps developers to start coding right away without wasting time on preparing and configuring the environment. In contrast to other Java frameworks
* Front end library for building user interfaces
* Most of the class wanted to use react so I just went with it. Still I believe that Angular would have been the better option, just because the client of our group project said he would prefer for us to use it and we have to use the same front end for both the individual and group projects .
* Database to use
* I think I’ll use a mongodb database for this project mainly because be a learning experience. Of course there where other reasons for this too like: I think it fits well with my project, I can make a free online database, it’s faster then a relational database like Mysql… But I’ll also use this as my research questions to get into the details of what would be the differences and the most advantages database to use.

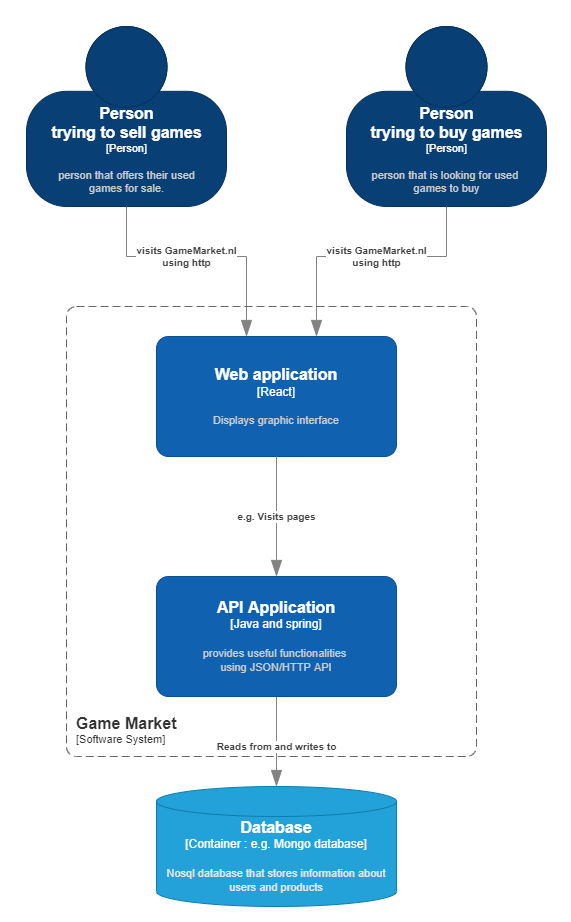
C4

C1

* Diagram

  Description automatically generatedThis system has only 1 user but these can preform 2 very different actions which is why they are here represented as different people One using the system to sell old and or used games and the other to buy these same games.

C2

* This system is divided into 2 separate containers the Web applications and API. The Web application is the front end made using react that the users interact with and the API is responsible for sending and receiving data from the Web application.

C3

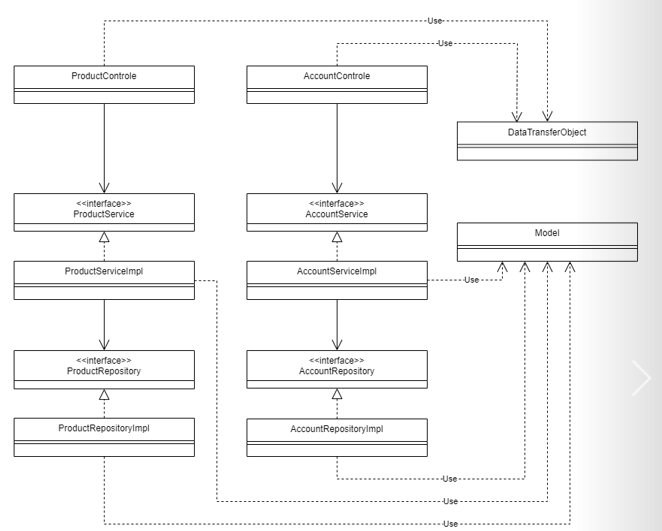
* The API as display here is divided into 2 sections (Accounts and Applications) and 3 layers(Controller, service, repository).

Diagram

Description automatically generated

C4

* A simple UML, the controller layer uses a DataTransferObject in contract to the Service and Repository layers that user the a Model. I think that this naming makes more sense then the other way around because the DataTranserObject is the object that will be transferred to the other systems that use the API.



Research Questions:

The sending of a model is unsafe and slow what options are there to fix that

* When to convert a model to a DTO, when to call a DTO

Are there any ways to identify different products without using a integer"