5/9/2023

Architecture document

Fontys University of Applied Sciences | ICT

Fabiënne Leidekker

S3 SOFTWARE | VERSION 1.3

Version history

Version	Date	Author(s)	Changes
0.1	17-03-2023	Fabiënne Leidekker	Added introduction, C4-model (C1, C2, C3)
0.2	22-03-2023	Fabiënne Leidekker	Added sequence diagram
0.3	23-03-2023	Fabiënne Leidekker	Changed introduction, added architecture choices
1.1	29-03-2023	Fabiënne Leidekker	Added ERD
1.2	12-04-2023	Fabiënne Leidekker	Added C4-model (C4)
1.3	09-05-2023	Fabiënne Leidekker	Added CI flow, updated ERD

Content

Introduction

As an amateur musician, it can be difficult to find people to jam with. The musician's application makes it possible for a musician, such as a guitarist, to find a band to play with. It will also work the other way around, so a band can look for their missing member. In addition, a user can chat with people to ask for tips during their own learning process. And last but not least, a user could also search for a music teacher.

Architectural choices for an application are very important. It is not something that can simply be changed once it's implemented. So, picking the right frameworks and languages and knowing why they would be suitable for my type of project is vital. In this document I am going to make these decisions by comparing what is mostly used. Furthermore, other architecture choices (like the C4-model) will also be described.

Architecture choices

When creating a full stack web application, it is important to know what frameworks and languages your architecture is going to have. Each framework has their strengths and weaknesses.

Front-end

When searching for front-end languages, the one you will encounter most often is JavaScript. It is a programming language that allows programmers to implement (complex) functionalities on their websites. It can also be used for back-end purposes, so it is a very versatile language. In addition, it also has a lot of frameworks, more than eighty that can be used for different purposes (Acharya, 2023). The use of JS frameworks is quite common. It is efficient and can save you a lot of time, because there are pre-made components that you can use, so no need to program everything from scratch. There are three frameworks that have been around for some time, while also being used quite a lot. According to Dhaduk (2023), the top three front-end frameworks for web development in 2023 are:

- React
- Vue
- Angular

React

Pros:

- It uses a virtual DOM. This ensures that changes of data can be loaded on the page quick and efficiently without having to refresh it (Acharya, 2023).
- Reusability of components makes it easy to collaborate and reuse them in other parts of the application (Dhaduk, 2023).
- React dev tools are advanced and super useful (Dhaduk, 2023).
- It supports JavaScript XML (JSX) that combines both JS and HTML. It helps in component rendering with nested elements, attributes, JS expressions, and conditional statements (Acharya, 2023).
- Easy to learn – it’s a very simple framework, especially for those who are familiar with frontend and JS (Sakovich, 2023).
- Large community – developers from all over the world use it, so the community has accumulated an extensive knowledge base (Sakovich, 2023).

Cons:

- Poor documentation - due to the rapid development and frequent updates in the framework, it’s hard to prepare thorough documentation (Sakovich, 2023).
- Developers find it hard to understand the complexities of JSX while beginning with the framework (Dhaduk, 2023).

Vue

Pros:

- Extensive and detailed documentation (Dhaduk, 2023)
- Easy to learn – the framework syntax is comprehensible even to beginners; Prior knowledge of JS, HTML, CSS is sufficient (Sakovich, 2023).
- High performance – due to it's simplicity and small size, Vue is an incredible fast and scalable tool (Sakovich, 2023).
- Typescript support (Dhaduk, 2023).
- Virtual DOM – an abstraction of the traditional DOM is a default feature, and it can help you significantly optimize the app performance (Sakovich, 2023).

Cons:

- Lack of stability in components (Dhaduk, 2023).
- Relatively small community (Dhaduk, 2023).
- Linguistic challenges – many Vue components are written in Chinese, which can be an obstacle for developers from other regions (Sakovich, 2023).

Angular

Pros:

- Reduces the amount of code since most of the prominent features like two-way data binding are provided by default (Dhaduk, 2023).
- A vast community for learning and support (Dhaduk, 2023).
- High performance – the framework ensures faster load time and better security due to the ahead-of-time compiler, which converts HTML and TypeScript code into efficient JS code during the development (Sakovich, 2023).
- Components are reusable and easy to manage using the dependency injection (Dhaduk, 2023).

Cons:

- Hard to learn – developers agree that the Angular framework is more complicated compared to React or Vue, and the documentation may be confusing (Sakovich, 2023).
- Heavy – due to its complexity, Angular is heavier than React, which can negatively impact dynamic apps, and code optimization will be required (Sakovich, 2023).

Conclusion

Now that I have weighed the pros and cons of each framework, I have decided that I'm going to use React. Even though the documentation might not be thorough, there is a wide community that uploads tutorials and articles that could be very useful. And also, since this project is just going to last this semester, it is wiser to pick a framework that has a smaller learning curve.

Back-end

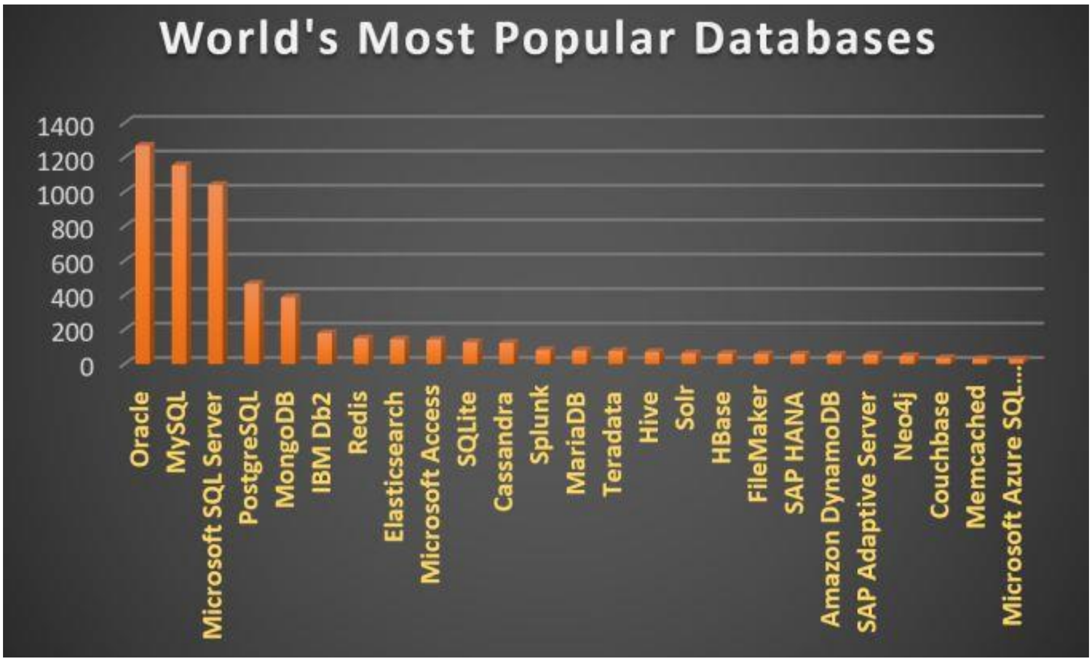
When searching for back-end frameworks and languages each kind could be used for a different purpose. One could be more suitable to manage big data, while the other is mainly used for web development. The table below shows the most popular backend frameworks for 2023 (Fincher & Desmond, n.d.).

Framework	Language
Spring	Java
Django	Python
Laravel	PHP
Flask	Python
Express.js	JavaScript
Ruby on Rails	Ruby
Next.js	JavaScript
ASP.NET Core	.Net

Since this semester hints at choosing between C# and Java, I'm only going to compare those two. What's already notable from the table is that Spring is used way more than ASP.NET Core. Spring also allows you to spend less time configuring the framework, and more time working on your application. It provides components to interact with your databases in a simple and powerful manner. On the other hand, ASP.NET Core is hyper-focused on performance, and runs faster than other web frameworks like Node.js (Fincher & Desmond, n.d.). Because I'm creating a rather small application, performance difference will probably not really be noticeable, so I will choose a language based on preference. And that preference is Java with Spring. I might be less experienced with it, but with all the documentation available, it would be a nice opportunity to expand my knowledge about it.

Database

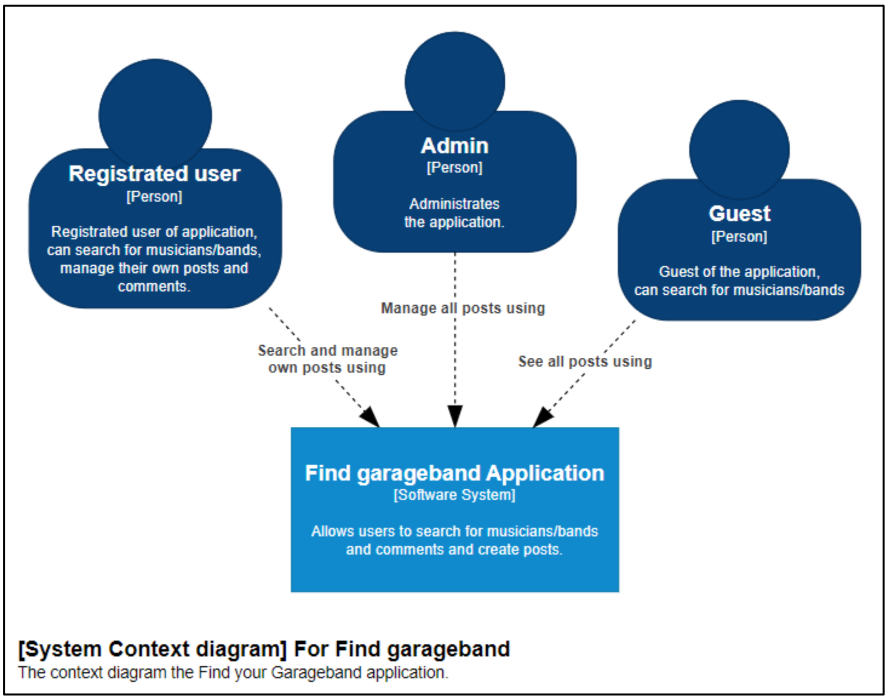
When picking a database, there are several types to choose from. The image below shows a ranking of the most popular databases (Chand, 2022). The most common types of databases are relational and NoSQL. For my personal project I'm looking for a relational database, because certain entities will need to have a relation to one another, so NoSQL isn't an option. If we look at the top three, they're all relational database systems. To choose between the 3, more aspects need to be considered. Oracle, MySQL and MS SQL server are quite similar, but an oracle database is commonly used for large and complex applications, while MySQL is ideal for small to medium-sized applications, web applications and e-commerce sites (Dearmer, 2023). As I'm developing a web application that is rather small, MySQL would be a more suitable fit. I'm also more familiar with MySQL than MS SQL server, so that is another reason why I'm going to choose MySQL.


World's Most Popular Databases
1400
1200
1000
800
600
400
200
0
Oracle
MySQL
Microsoft SQL Server
PostgreSQL
MongoDB
IBM Db2
Redis
Elasticsearch
Microsoft Access
SQLite
Cassandra
Splunk
MariaDB
Teradata
Hive
Solr
HBase
FileMaker
SAP HANA
Amazon DynamoDB
SAP Adaptive Server
Neo4j
Couchbase
Memcached
Microsoft Azure SQL...

C4 model

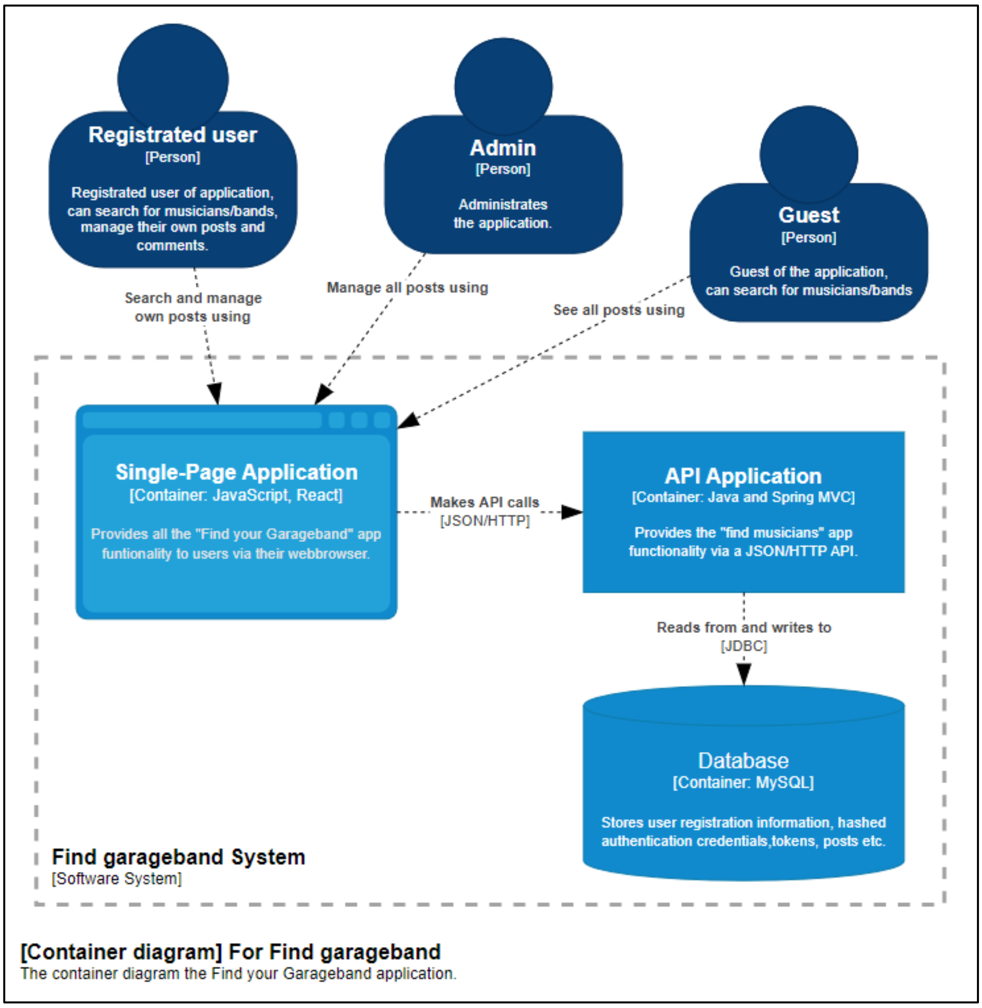
C1: System context diagram

The system context diagram gives a view from a high level. There are 3 types of users that all that use the "Find garageband Application", but they do not all have access to the same functionalities. An admin has different permissions than a normal user.


Registrated user
[Person]
Registrated user of application, can search for musicians/bands, manage their own posts and comments.
Admin
[Person]
Administrates the application.
Guest
[Person]
Guest of the application, can search for musicians/bands
Manage all posts using
Search and manage own posts using
See all posts using
Find garageband Application
[Software System]
Allows users to search for musicians/bands and comments and create posts.
[System Context diagram] For Find garageband
The context diagram the Find your Garageband application.

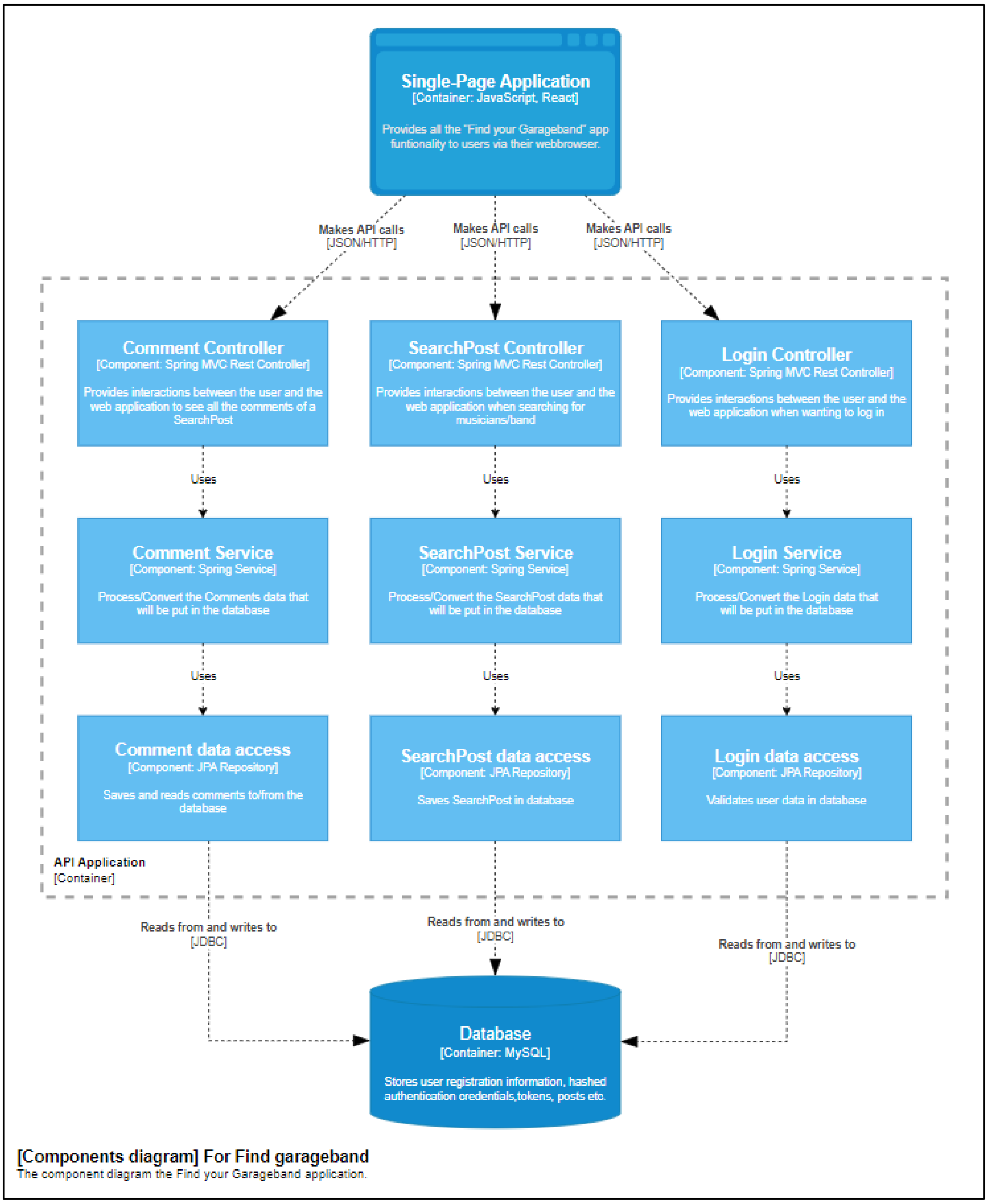
C2: Containers diagram

If we zoom in further on the software system, we get the following diagram. The software system consists of three separate units. The front-end, the back-end and a database. Users can make 'requests' on the front-end, the front-end sends these requests/API calls to the back-end, where it instructs the database to get the information needed in order to send it back to the front-end and display it to the user.


Registrated user
[Person]
Registrated user of application, can search for musicians/bands, manage their own posts and comments.
Admin
[Person]
Administrates the application.
Guest
[Person]
Guest of the application, can search for musicians/bands
Search and manage own posts using
Manage all posts using
See all posts using
Single-Page Application
[Container: JavaScript, React]
Provides all the "Find your Garageband" app funtionality to users via their webbrowser.
Makes API calls
[JSON/HTTP]
API Application
[Container: Java and Spring MVC]
Provides the "find musicians" app functionality via a JSON/HTTP API.
Reads from and writes to
[JDBC]
Database
[Container: MySQL]
Stores user registration information, hashed authentication credentials,tokens, posts etc.
Find garageband System
[Software System]
[Container diagram] For Find garageband
The container diagram the Find your Garageband application.

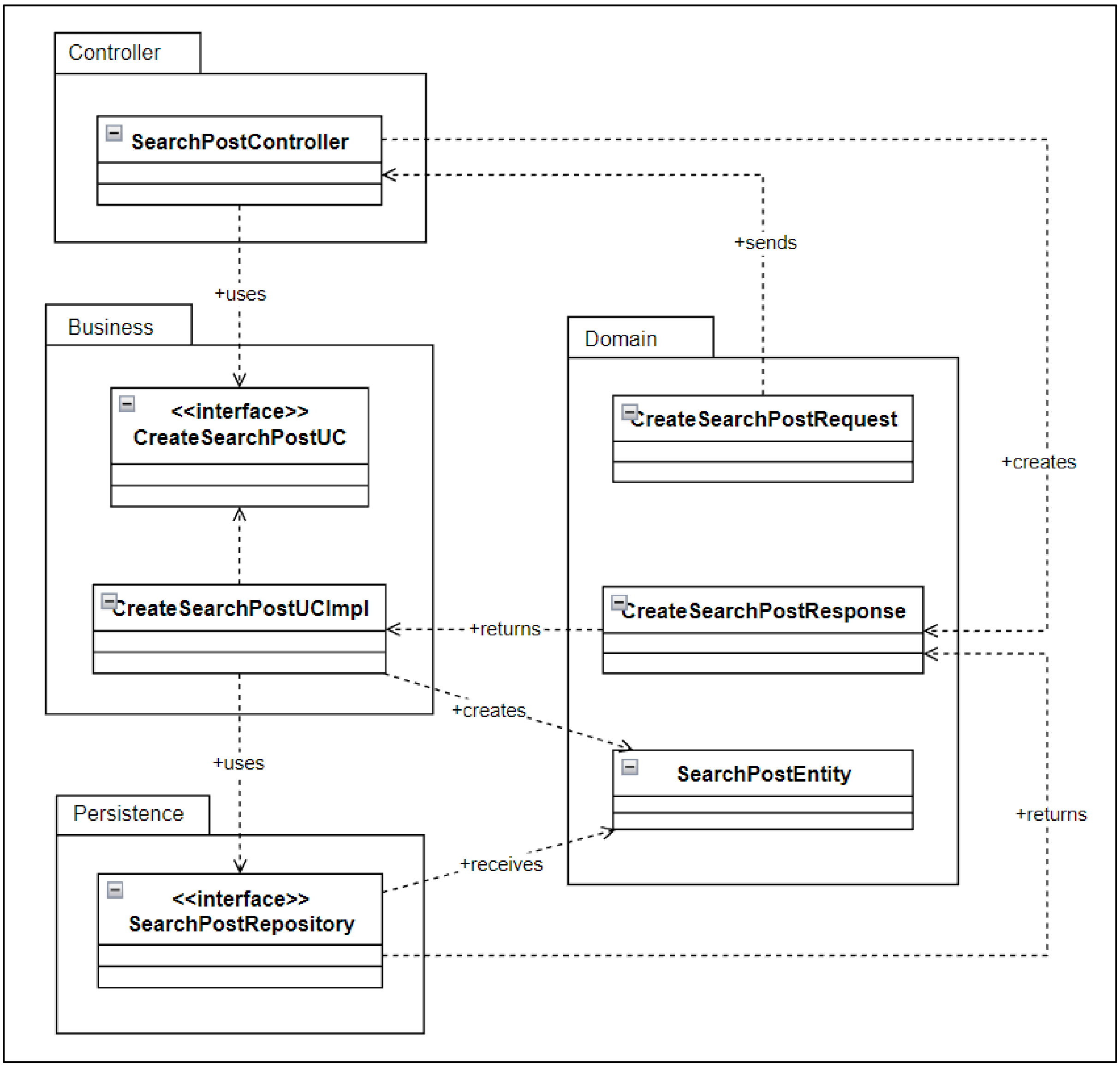
C3: Components diagram

The components diagram shows a few components that can be found within the API application from the C2 diagram. When making an API request from the front-end, all the calls go to their own specific controller in the back-end. This is one way how I implement the SOLID principle. Every entity should have their own flow and deal with their own responsibilities. The controller uses the service to do any logic operations and can use the data access layer/repository to access the data from the database.


Single-Page Application
[Container: JavaScript, React]
Provides all the "Find your Garageband" app funtionality to users via their webbrowser.
Makes API calls
[JSON/HTTP]
Makes API calls
[JSON/HTTP]
Makes API calls
[JSON/HTTP]
Comment Controller
[Component: Spring MVC Rest Controller]
Provides interactions between the user and the web application to see all the comments of a SearchPost
SearchPost Controller
[Component: Spring MVC Rest Controller]
Provides interactions between the user and the web application when searching for musicians/band
Login Controller
[Component: Spring MVC Rest Controller]
Provides interactions between the user and the web application when wanting to log in
Uses
Uses
Uses
Comment Service
[Component: Spring Service]
Process/Convert the Comments data that will be put in the database
SearchPost Service
[Component: Spring Service]
Process/Convert the SearchPost data that will be put in the database
Login Service
[Component: Spring Service]
Process/Convert the Login data that will be put in the database
Uses
Uses
Uses
Comment data access
[Component: JPA Repository]
Saves and reads comments to/from the database
SearchPost data access
[Component: JPA Repository]
Saves SearchPost in database
Login data access
[Component: JPA Repository]
Validates user data in database
API Application
[Container]
Reads from and writes to
[JDBC]
Reads from and writes to
[JDBC]
Reads from and writes to
[JDBC]
Database
[Container: MySQL]
Stores user registration information, hashed authentication credentials,tokens, posts etc.
[Components diagram] For Find garageband
The component diagram the Find your Garageband application.

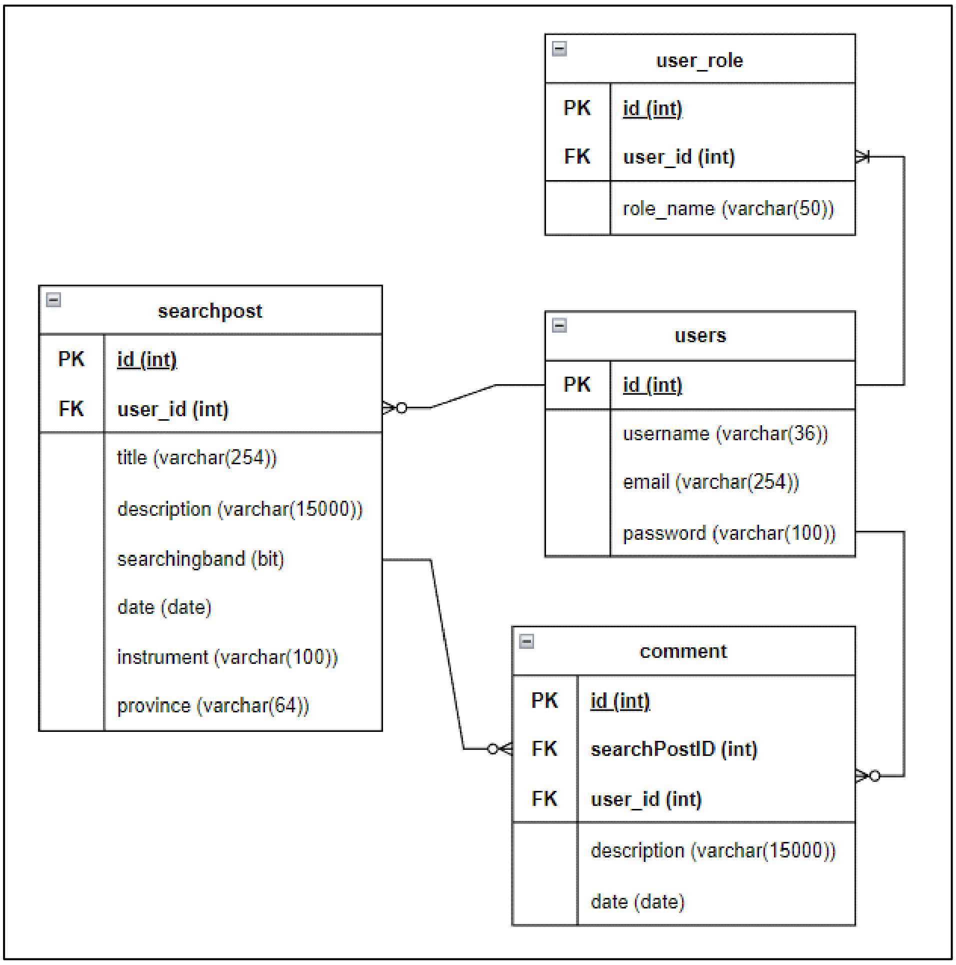
C4: Code diagram

The last diagram is the code diagram. In this diagram we see the flow of creating a post. We also see the SOLID principle being implemented. Each class belongs to their function-corresponding layer. All the object/entities are placed in the domain layer. The controller layer receives and sends all the requests from/to the front-end. The business layer handles all the logic calculations. It can for example get information from the repository, or convert entities to different type of models.



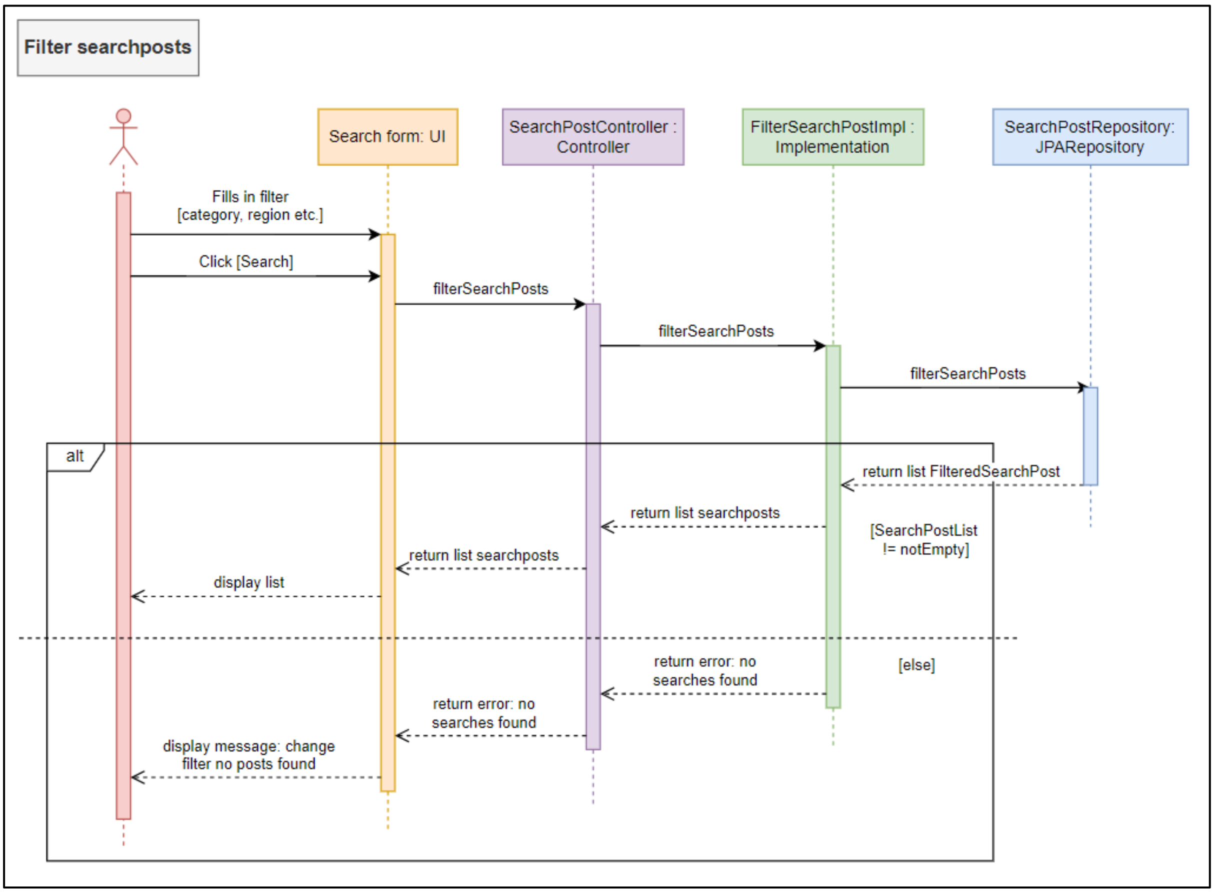
Entity relationship diagram

The users table is linked to the user_role table, this is because are different type of users. Someone could be a normal user or an admin that has access to more functionalities. But a user could also have multiple roles. A user could also make multiple search posts and comments. Between search post and comment, there is a zero-to-many optional relationship. This means that a search post could have zero or many comments, but a specific comment belongs only to one search post.



Sequence diagram

Getting filtered search posts


Filter searchposts
Search form: UI
SearchPostController : Controller
FilterSearchPostImpl : Implementation
SearchPostRepository: JPARepository
Fills in filter [category, region etc.]
Click [Search]
filterSearchPosts
filterSearchPosts
filterSearchPosts
alt
return list FilteredSearchPost
return list searchposts
[SearchPostList != notEmpty]
return list searchposts
display list
return error: no searches found
[else]
return error: no searches found
display message: change filter no posts found

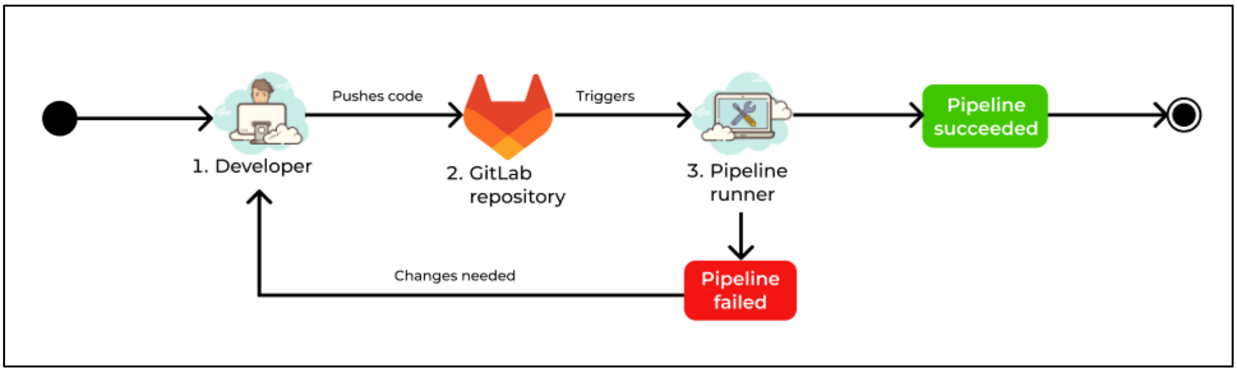
CI flow diagram

Steps:

1. The developer writes code and pushes it to their Gitlab repository.
2. The repository is stored/hosted on the Gitlab server that triggers the pipeline.
3. The pipeline runs on my local machine and executes 3 tasks:
   - **Build**: build the app from the source code
   - **Test**: runs the tests (unit & integration*)
   - **SonarQube**: writes a code quality report about the source code**

If not all 3 tasks succeed, the code needs to be changed, otherwise the flow is complete.

*Docker needs to be up and running for integration tests to successfully be executed
**SonarQube needs to be operational to create a report and successfully execute



References

Acharya, D. P. (2023, March 1). *The 40 Best JavaScript Libraries and Frameworks for 2023*. Kinsta.

https://kinsta.com/blog/javascript-libraries/#:~:text=There%20are%20around%2083%20of,their%20usability%20in%20this%20section

Chand, M. (2022, December 18). *Most Popular Databases In The World (2023)*. C# Corner.

https://www.c-sharpcorner.com/article/what-is-the-most-popular-database-in-the-world/#:~:text=The%20most%20popular%20database%20in%20the%20world%20is%20Oracle%20according,and%20MongoDB%20in%20the%20ranking.

Dearmer, A. (2023, March 14). *Oracle vs. MySQL: Detailed Comparison of Syntax & Uses for These Databases*. Integrate.io.

https://www.integrate.io/blog/oracle-vs-mysql/#:~:text=MySQL%20supports%20full%2Dtext%20and,%2C%20Partitioned%2C%20and%20so%20forth.&text=Oracle%20is%20better%20for%20enterprise,Oracle%20requires%20a%20licensing%20fee

Dhaduk, H. (2023, March 22). *Best Frontend Frameworks for Web Development in 2023*. Simform.

https://www.simform.com/blog/best-frontend-frameworks/#:~:text=The%20State%20of%20Best%20Frontend,js%2C%20and%20Angular

Fincher, J., & Desmond, K. (n.d.). *The 11 Best Backend Frameworks – 2023*. Coding Nomads. Retrieved March 3, 2023, from

https://codingnomads.co/blog/best-backend-frameworks/

Sakovich, N. (2023, March 23). *Top Most Popular Frontend Frameworks 2023*. Sam Solutions.

https://www.sam-solutions.com/blog/best-frontend-framework/