FACULTAD DE
INGENIERÍA Y CIENCIAS
APLICADAS

Software Architecture
Assignment 1

2024-20

Alumnos:

Tomás Erdmannsdörffer

Juan Pedro Lira

Domingo González

08/08/2024

Description

Phoenix: web development framework written in Elixir. Focuses on high performance and concurrency. The main characteristics of Phoenix are the following:
-Real-time communication
=Performance
=Productivity
=Ecto
=Templates and views

Elixir: Functional, concurrent and general-purpose programming language. Runs on Earlang VM. It's key features are:
=Concurrency
-Fault-Tolerance
-Immutability
-Pattern Matching

MongoDB: Is an open-source NoSQL database, designed for high performance, availability and ease of scalability. It uses a JSON based storing system called BSON, differentiating it from the other databases.

MongoDB is mainly used for activities like content management, real-time analytics, IoT and E-commerce.

PostgreSQL: Open-source object-relational database, that mainly focuses on robustness, standardization and extensibility. This is one of the most popular and documented databases thanks to its ease of use. Is very robust and friendly with the user, so it is easy to scale, the SQL standards work everywhere and generally functions well on all ambits. A very well-rounded database.

Familiarity

Our group did not have any familiarity or experience with the Phoenix framework, neither the Elixir language nor the MongoDB database. This really caused a lot of confusion on the members of the group, so much that we ended changing databases to PostgreSQL.
Although at the end, Phoenix was particularly similar to the Django framework so the model part of things was very intuitive. The MCV architectural pattern was easy to understand in phoenix due to several projects we have had in the past.

Connectivity

Connectivity wise, it was very easy to implement PostgreSQL. We had to start PostgreSQL, and phoenix automatically created a database. The config.exs file has the database configuration pre-created, we only had to change the password to make it work. To run seeds we applied this command:

-mix run priv/repo/seeds.exs

Challenges

We had a very hard time trying to connect to MongoDB, so we ended up using PostgreSQL. The connection of Phoenix and MongoDB was not implemented for the following reasons:

- **Deprecated versions**: the Mongo-Ecto versions do not work with most of the concurrent libraries that are necessary for a good development environment on Phoenix. The main github had commits log ago, and all the test were failed, meaning that the system was not prepared for the environment.

Also mongo-driver had many failures with libraries like decimal, ecto, phoenix, phoenix_html and telemetry_metrics. Most of the libraries had to be downgraded to a later version for the server to compile correctly.

- **Migration and table system**; Ecto did not allow us to create migrations, so the only way to create a table was to do it manually on the actual database. This came to a lot of problems as it needed a lot of additional configuration files that were causing many issues.

- **Lack of documentation**: The Phoenix framework, works naturally with PostgreSQL database, it was designed to do so, in consequence, MongoDB aside of being deprecated, also did not have a lot of documentation to rely on. All we could find was made for other frameworks or very old information. In many forms we found information that it wasn't recommended to try to connect MongoDB with Phoenix

- **Unfamiliarity with the environment**: We weren't prepared at all to create a proper connection on the time frame we had available and didn't have enough experience too.

Phoenix was confusing at first, but we soon realized that the concept was very similar to Django, and we got the hang of it, the model to table relation and the views were almost the same.

For the models we used a command (ex. for sales mix phx.gen.html Sales Sale sales book_id:references:books year:integer sales:integer) to generate the models, and then we edited them on the schema.

The queries were SQL which we are familiar with, so we had no problem there.