DB Exam Project 2020

Group: #TypeScriptCarry

- Mohammad Hariri
- Rasmus Jarnborg Friis
- Andreas Guldborg Heick

Exam Description

In the modern IT world, many businesses provide their products or services on-line, using Internet as a channel of communication with their customers. During economically hard times as Corona isolation for many it remains the only platform for marketing and retailing.

To meet the requirements of the on-line digital transition, modern applications work with purpose-built and decoupled data stores, where different application components can be mapped to different data sources, rather than to a single database. This approach is implemented in microservices architecture, as well as in polyglot persistence applications.

Project Objective

Your objective is to provide, as a proof of concept, database support to an application or related

applications, where multiple diverse database models are built and implemented for different purposes.

Installation Instructions

Requirements:

- Install all required databases (Neo4J, MongoDB, PostgreSQL)
- Node

1. Open and start databases - make sure constants.ts matches your login credentials for all databases
    a. PostgresSQL: CREATE DATABASE db_exam;
2. Clone the project
3. Cd into the project root
4. npm install
5. npm run dev

If you get errors or don't get "[Database name] is connected" message in your console for each database, then you probably didn't start the database services correctly.

Project description

We have chosen to build a polyglot persistence application that implements and uses four different databases. We have chosen Neo4J, MongoDB, Redis and PostgreSQL to build our application.

The idea behind the application is to make a social media using four different kind of database each to their advantages.

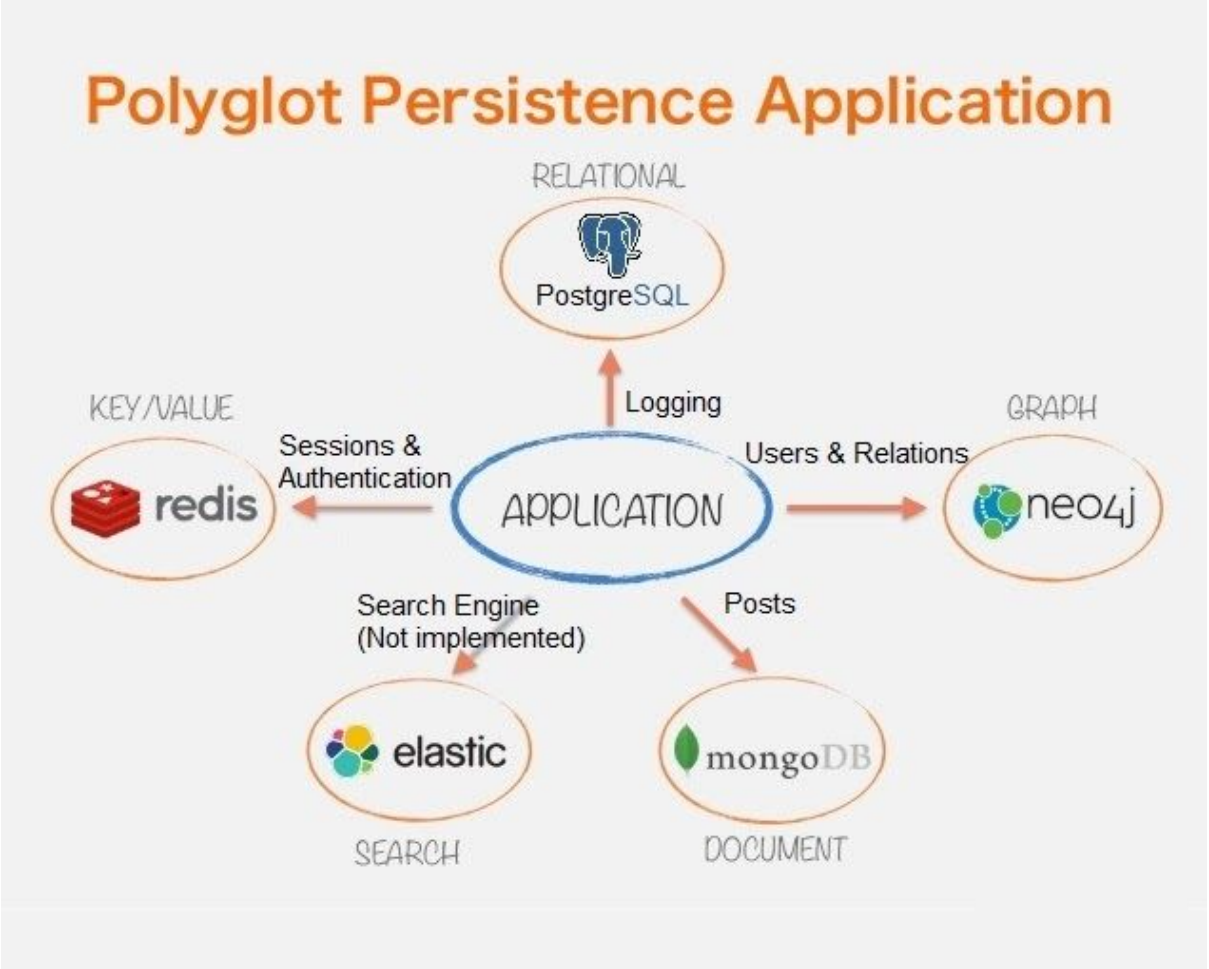
We are primary using Neo4J to store the users data, and the users relation with each other. A relation in our application is following one or multiple users.

MongoDB is used to store posts that the users make, and the idea is that you will only be able to see the posts, from the people you follow.

We also want people to be able to login to the media, this action is handled by Redis, meaning it will control the users sessions and general authentication.

Lastly we are using PostgreSQL to control the logging of the system, because it is an important security aspect, and we can use it to see usage and maybe improve the platform. Logging is an essential part of security, used to spot issues and solve them.

If time was not a problem and this project was a real business case, then we would probably also implement Elastic Search. Elastic Search is an open-source search engine that can store documents, or in our case users or posts. Elastic Search would be a good last database to use if a media had enough data and required fast and good searching on the platform.



Overview of our application plan

Requirements

Use case:

For a blog or social media, where data is the main business model.

With our application a media could analyse relations between users and use this to improve the platform. From these relations we would know who follows who, and then we can look at posts and show the posts of all followed.

Logs would provide us with data about usage so we can improve, and it would help with security. With sessions and authentication a media could make sure who can use and see what.

Functional requirements:

1. Authentication - Users have to be authenticated to use the application
2. User Sessions - Users sessions needs to be stored so we at all times know who is logged in and if they are authenticated to use the application
3. Data Analysis - We need a way of analyzing users and their relations to other users.
4. Storage of data - We need a place to store Users and Posts.
5. Logging - We need a way of logging interactions and use of the application, and a log storage so we can go back and view the usage. We also need logging for security purposes, to be able to catch unauthorized usage.

Non-functional requirements:

1. Has to work in Postman
2. Has to work on Windows 10 & Mac

Technologies

Databases:

- Neo4J - NoSQL graph database
- MongoDB - NoSQL document-oriented database
- Redis - key/value NoSQL database
- PostgreSQL - relational SQL database

Tools & languages:

- Node - JavaScript runtime for serverside scripting
- ExpressJS - Node webapplication framework for building web applications and API’s
- TypeScript - A strict syntactical superset of JavaScript that adds optional static typing. Usually used for larger applications.
- Postman - A tool to test and develop API web requests