Advanced Cloud

T-CLO-901

LINKEDIN

Software Architecture Specifications

EPITECH.
L'ECOLE DE L'INNOVATION ET DE L'EXPERTISE INFORMATIQUE

Table of contents

Introduction

The aim of this software architecture specification (SAS) is to present the technical elements necessary for the Linkedin project.

1. Project context

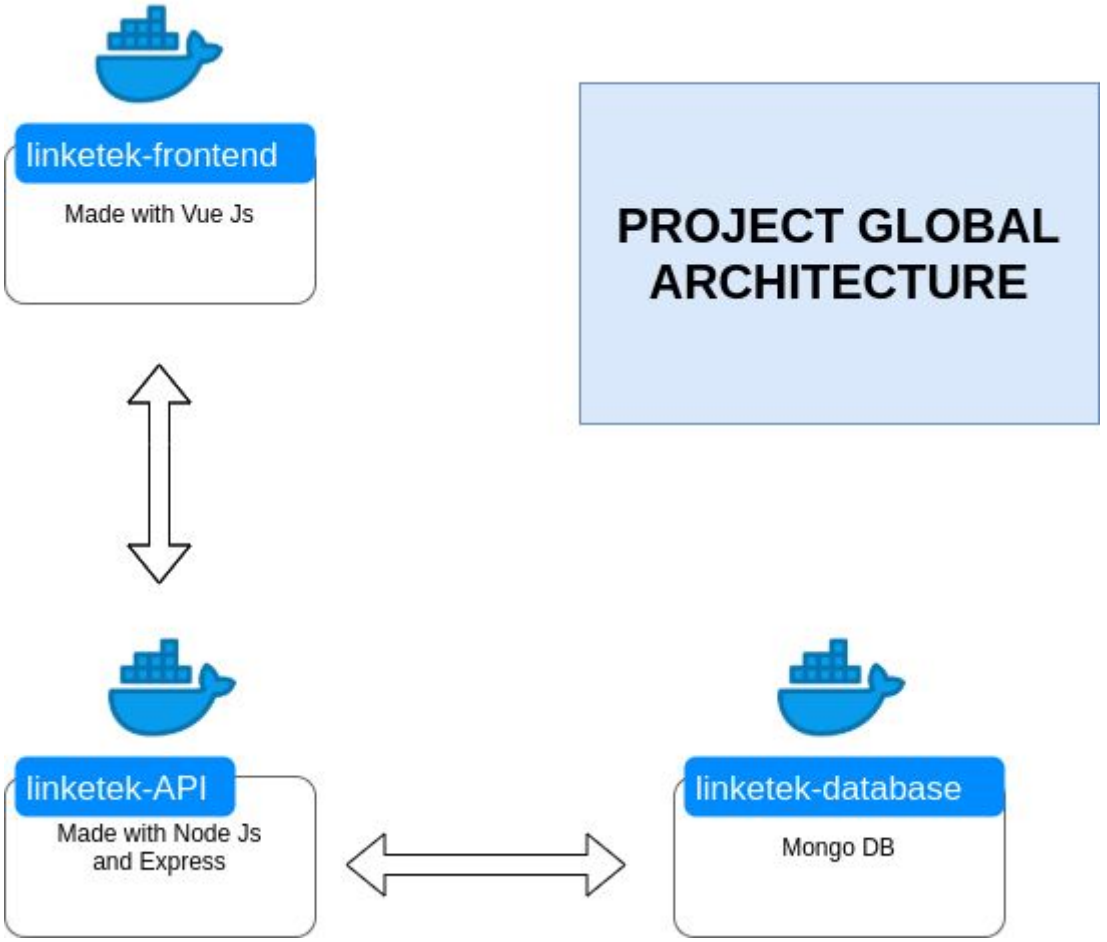
The project consists of a clone of the famous social-network Linkedin

2. Global architecture

The project is composed of three main parts. A Node JS API, a Vue JS front end and a mongo DB database.

Each port is dockerized and the whole project is mountable in a single ***docker-compose up*** command line.

Below is a simple visualization of the global architecture and how each component communicates or does not communicate with the others.



3. Component description

a. Mongodb

i. Why Mongodb

We have chosen Mongodb as database for two main reasons :

- It is a Nosql oriented database
- It's easy to implement in the back-end with Node JS and Express.

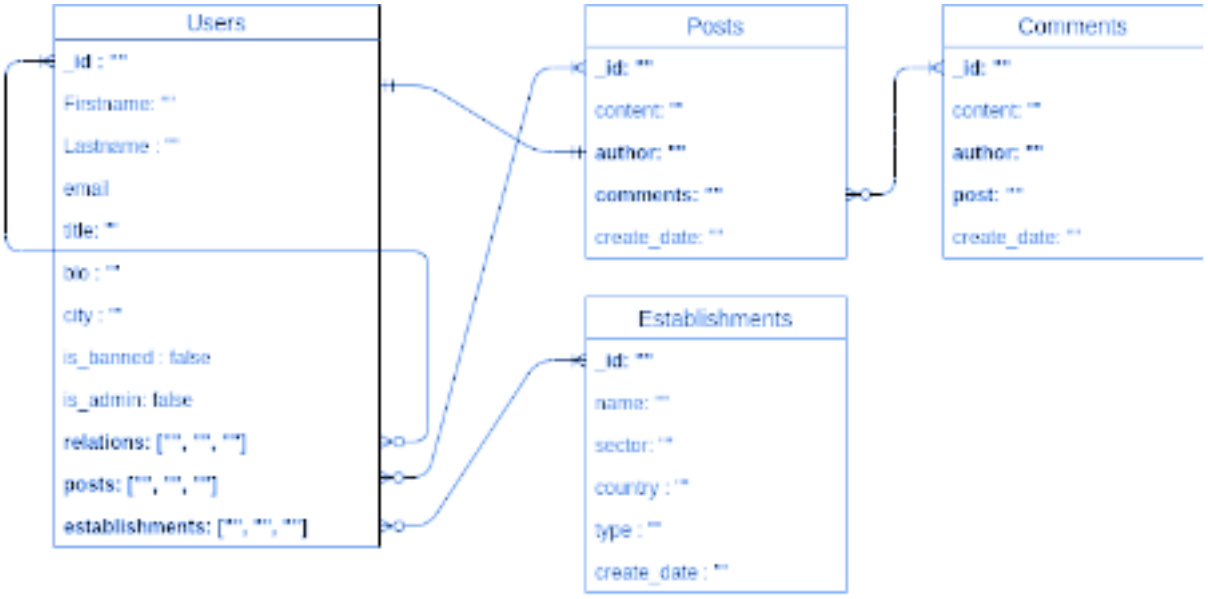
The fact that it's a No SQL is important because we store and read data as JSON documents. What is consistent with the front end where Posts requests payloads are JSON documents and where we read data as JSON Documents.

ii. Collections

We use a single database with four collections.

- users
- posts
- comments
- establishments

Below, a visualization of the collections and the links between them.



b. API

The API is made in Node JS.

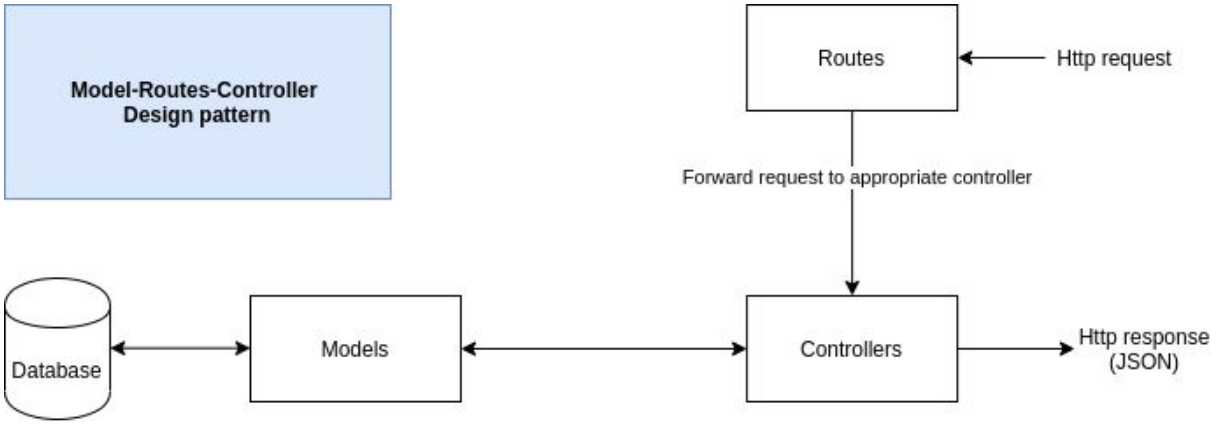
This language was chosen for its ease of use and its ability to easily and cleanly implement the design pattern described above.

Plus, it's Javascript and it's match with the front-end.

The API main role is to establish a link between the front end and the database without giving too much information about the environment in the client.

The API is designed by the Model-Routes-Controller design pattern.

Below, a visualization of this pattern.



A full documentation of the endpoints is available here :
https://documenter.getpostman.com/view/9726978/SWLiZ6BH?version=latest#91e9cebe-c70a-4bd0-b166-d876edd5d348

c. Client

The client is made with Vue JS.

Additionals frameworks are used, Here is a list of them and their usefulness.

- **Vuetify :** Used for the design of the site.
  It allows to make material design without touching the css, thanks to native View components.
- **Router :** Used for handle differentes routes by extension the permissions.
  This framework determine which component is rendered for an url and if a user is allowed to access it or not
- **VueX :** Handle the **"store"** of the application.
  This allows data of the user logged-in accessible anywhere in the app without passing it through every component

4. Traceability matrix

A traceability matrix is available in the project repository.

This matrix makes the correspondence between components, classes, functions and requirements developed in the request for proposal.