INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

todo

Bernardo Rodrigues

Supervisor: Nuno Leite

Project report on Projecto e Seminário
Graduation on Software Engineering and Computer Science

July 2021

Acronyms

JWT Json Web Token.

Chapter 1

Introduction

In today's fast and complex world people try to schedule days or weeks. People create small reminders, either electronically or physically, to do something, whatever that may be. Those little notes we give to ourselves, those "to-do's" are often forgotten and not fully fulfilled. There exists software applications in the market that partially accomplishes this goal. They will often remind the user of the task that needs to be done only using the time frame the user gave. This creates a problem where the individual makes a reminder, does not do it and the application will not keep notifying the user.
As such the proposed solution will attempt to mend that by creating an environment where a reminder is easily created but hardly forgotten or skipped over. The solution consists of a web application divided into three different components, a front-end web app, a back-end web Application Programming Interface (API) and a database. This application will allow the users to create an account and then make reminders. The users will be notified by the application until they mark said reminder as done.
The notification is a necessity to ensure the user fulfills the reminder. Since the user will be notified until the reminder is marked as "done" the person will be reminded every day until the completion of the task.

For this purpose a set of mandatory requirements was outlined:

- Account creation, login, logout and account deletion;
- Reminder creation, update and deletion;
- Notifications;

Some optional requirements were also outline to enrich the application:

- Translations
- Priority notifications

This report will outline the process of the application development, the choices made and the work that remains to be done.

Chapter 2

State of the Art

Chapter 3

Architecture

The architecture for the API was thoroughly researched and studied. At first this project was gonna use a monolithic architecture [?] where the whole API would be implement on a single server as seen in the image below.

This was later changed due to the fact that it wasn't as easily scalable [?]. After this it was decided that the API would be implemented using a *microservice*[?] architecture. This allows for a more select scalability. When needed, more instances of the same services can be launched and improve overall performance.

There will be different services:

1. User

2. To-do

3. Task

There was also the idea of combining both the monolithic and micro-service architecture in a mixture of both patterns. This idea was later scrapped and the micro-service system was then chosen.

Each of these services are independent from one another and self sufficient, each with it's own responsibilities.

3.1 User

The user micro-service will have the operations needed for an user to utilize the application. These operations consist of the sign up, login, logout and, delete. The micro-service will use Json Web Token (JWT) [?] as a way to make sure the users only access their information. The service is split into two different areas, the repository[?] that accesses the database and ,the routing.

3.2 To-do

The to-do micro-service handles all the operations necessary to manage the to-do's. These operations embedded in this service are to-do creation, deletion and update. This micro-service will interact with the user service to validate and extract the JWT information.

3.3 Task

The task micro-service has three main objectives. It offers a way to subscribe and unsubscribe users to receive notifications as well as checking the database periodically to ensure the notifications are sent to the subscribed users. The notifications will be sent according to the push protocol[?].

Chapter 4

Database and Data Modeling

Chapter 5

API

Bibliography