**EDUCATIONAL PLATAFORM**

[INTRODUCTION 2](#_Toc152801647)

[FRONTEND 3](#_Toc152801648)

[UI / UX 3](#_Toc152801649)

[HOME PAGE 3](#_Toc152801650)

[AUTHENTICATION AND LOGIN 4](#_Toc152801651)

[USER PROFILE 6](#_Toc152801652)

[COURSE CATALOG 8](#_Toc152801653)

[APPLICATION PROGRAMMING INTERFACE 10](#_Toc152801654)

[DATABASE 10](#_Toc152801655)

[FRAMEWORKS, TOOLS AND VERSIONS 10](#_Toc152801656)

[PROGRAMMING LANGUAGES 10](#_Toc152801657)

[FRAMEWORKS 10](#_Toc152801658)

[PYTHON MODULES 11](#_Toc152801659)

[OTHER UTILITIES 11](#_Toc152801660)

[GIT BRANCH STRATEGY 11](#_Toc152801661)

[REFERENCES 12](#_Toc152801662)

# INTRODUCTION

This project is full application with microfrontends, microservices, an API developed on Python and a database provided by MongoDB

The fronend is developed using a micro frontend architecture and the NextJS framework. These consist of the next modules, each one of them is routed and enables navigation between them

* Catalog of available courses on the platform
* Individual page for each course
* Platform administration page
* Authentication and login
* User profile page

Functionally, the system aims to serve as an educational platform where people can create their own profile and register for different courses available on the platform. All the code for the API to connect this functionallity with the database where data is retrieved, was created using Python.

Finally, the database itself was designed as a No-SQL database, therefore it implemnts MongoDB as a Docker container configured to connect and work seamlessly with the platform

Following in this document, you Will find more information baout each one of the components, tolos and ujses of everything implemented in this project

# FRONTEND

All the code for the views in this Project was developed using the following frameworks: NodeJS, React.js and Next.js, all with the porpousse to create a fast, responsive and manteinable web application. Those frameworks also ensure to be easier to learn and so to deploy the application in no time.

## UI / UX

We used an independant framework for stylin and graphic design: MaterialUI, specifically the following modules: emotion/react, emotion/styled, mui/icons-material

## HOME PAGE

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza mediaUpon launching the application and after successfully logging in using any of the methods, this page appears.

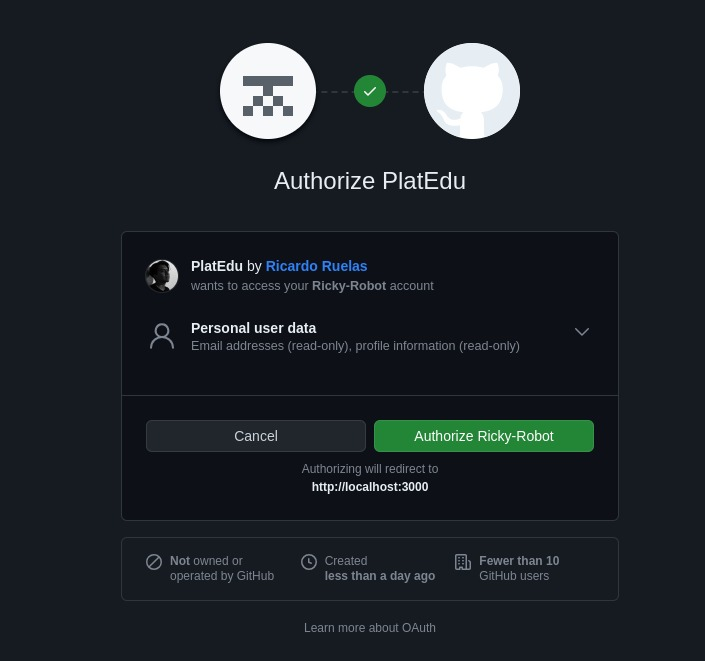
## AUTHENTICATION AND LOGIN

Authentication is implemented using two methods: a form requesting email and password, and through the GitHub API. Without successfully logging, the other functionalities of the application cannot be accessed. The respective code can be found in '**main/login/logingit/app/api/auth/[...nextauth]/route.js**', utilizing NextAuth.js and the mentioned providers as per the documentation provided on the page.

A local file named '**/.env.local**' is created to contain the API keys. This file must be included in the '.gitignore' file. It is necessary to configure the button in '**/components/signin-btn.jsx**' to enable the login functionality.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente



Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

The functionality for logging in via GitHub is achieved with the following lines of code:

Texto

Descripción generada automáticamente

## USER PROFILE

To access this view, it is necessary to click on the user's avatar, on the upper right cornero n the screen and then select 'Profile'.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

In addition to basic information (name, email, and password), this view includes the functionality to modify the profile image. Users can upload a new image using the operating system's file system interface and there's a button to save the information.

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamenteThe default avatar is retrieved from the session generated by OAuth, along with the name and email.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto

Descripción generada automáticamenteThe email field already implements a validator

## COURSE CATALOG

In this view, the available courses for each user are displayed. There's a section to enter the keyword for the subject. In this platform, the keywords are: Mathematics, SoftwareEngineering, OOP and Math. The image below shows that all subjects have been added.

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Texto

Descripción generada automáticamenteThe part of the code that enables this functionality is:

Where the dependencies of React, useState, Next.js utilities, and Material UI are exported, local state is utilized to store the list of courses (courses) and the current course key (courseKey). Additionally, the Next.js router object is obtained.

An addCourse function is defined to add a course to the list of courses.

A viewCourseDetails function is defined to navigate to the details of a specific course.Texto

Descripción generada automáticamente

# APPLICATION PROGRAMMING INTERFACE

Each API is developed usign Python and its module Flask.

We developed four API, one for each module that the frontend has, thus there is an API for the catalogo courses, another for each one of the specific course, one for the login and last one for the profile of each user.

# DATABASE

Due to the initial requirement of creating a distributed system, it is necessary to implement a NoSQL database, for which MongoDB was chosen.

In our database, we store information about the users registered on the platform, which is retrieved specifically for each user during their session. Additionally, we store information regarding the available courses, which is referenced when displayed in the catalog as well as in a particular view on a page when a specific course is queried.

# FRAMEWORKS, TOOLS AND VERSIONS

Throughout this project, various technologies have been utilized, spanning multiple programming languages and smaller frameworks. Below is a detailed, though not exhaustive, list of the main tools implemented

## PROGRAMMING LANGUAGES

* Python 3.10: used for the development of the API and connection trough the database
* JavaScript ECMAScript 2021: not used in raw code, but in the framework NodeJS and thus trough all the frontend

## FRAMEWORKS

* NodeJS v.20.10.0
* React v.18.1
* Next.js v.14.0.1
* MaterialUI v.5.0.6
* NextAuith v.4.24.5

## PYTHON MODULES

* Flask v.3.0
* MarkupSafe v.2.1.3
* Pymongo v.4.6.1

## OTHER UTILITIES

This Project was developed by a team of 7 members, each one of them was using a Fedora 38 Workstation, working on both local and remote repository provided by GitHub; andthe Git versión is 2.43.0

Docker, for the attempt to create the several conteiners for the microfrontends, as well for the database MongoDB, all the developers installed the versión 24.0.6. And, using this technology, we pull de container “mongo”, versión 3.1 and for the easiest creation and manipulation of the data, Mondo Compass.

# GIT BRANCH STRATEGY

Diagrama, Esquemático

Descripción generada automáticamenteThe Feature-Based branch strategy involves creating separate branches for each new feature or functionality, allowing independent tracking and development of each part of the project before merging it into the main branch. This facilitates collaboration between teams and version control, maintaining a modular and organized approach to development.

# REFERENCES

*Deploying*, Next.JS. Available in: [https://nextjs.org/docs/app/building-your-application/ deploying], October, 29, 2023.

*GitHub*, NextAuth.js. Available in: [ https://next-auth.js.org/providers/github], October, 29,2023

*Google*, NextAuth.js. Available in: [https://next-auth.js.org/providers/google], October, 29, 2023

*Mongo*, DockerHub. Available in [https://hub.docker.com/\_/mongo], November, 12, 2023.