

**Assignment Cover Sheet**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student name:** | **Bruna Minhos, Jocimara Melo & Rodrigo Bezerra** | | | | | |
| **Student number:** | **3017504 - 2981500 - 3039721** | | |  | | |
| **Faculty:** | **Computing** | | |  | | |
| **Course:** | **Computer Science** | | | **STAGE** | **3** | |
| **Subject:** | **Web Technologies** | | | | | |
| **Study Mode:** | Full time | **x** |  | Part-time |  |  |
| **Lecturer Name:** |  | | | | | |
| **Assignment Title:** | **Research, Plan, Design and Build Single-Page Web Application** | | | | | |
| **No. of pages:** |  | | |  | | |
| **Disk included?** | Yes |  |  | No | **x** |  |
| **Additional Information:** | (ie. number of pieces submitted, size of assignment, A2, A3 etc) | | | | | |
| **Database, Documentation, Zipfile of source code, and link to Stackblitz** | | | | | |
|  | | | | | |
| **Date due:** | **09/05/2022** | | |  | | |
| **Date submitted:** | **23/05/2022** | | |  | | |
|  | | | | | | |
| **Plagiarism disclaimer:**  *I understand that plagiarism is a serious offence and have read and understood the college policy on plagiarism. I also understand that I may receive a mark of zero if I have not identified and properly attributed sources which have been used, referred to, or have in any way influenced the preparation of this assignment, or if I have knowingly allowed others to plagiarise my work in this way.*  *I hereby certify that this assignment is my own work; based on my personal study and/or research, and that I have acknowledged all material and sources used in its preparation. I also certify that the assignment has not previously been submitted for assessment and that I have not copied in part or whole or otherwise plagiarised the work of anyone else, including other students.*  **Signed & dated: Bruna Minhos, Jocimara Melo & Rodrigo Bezerra – 23/05/2022** | | | | | | |

## **Please note:** Students **MUST** retain a hard / soft copy of **ALL** assignments as well as a receipt issued and signed by a member of Faculty as proof of submission.

Contents

[Overview 3](#_Toc104231139)

[Features 4](#_Toc104231140)

[Architecture 5](#_Toc104231141)

[Wireframes 6](#_Toc104231142)

[Homepage 6](#_Toc104231143)

[Login/Logout Page 6](#_Toc104231144)

[Signup Page 7](#_Toc104231145)

[Tasks page 8](#_Toc104231146)

[Add new Folder 8](#_Toc104231147)

[Add new Task 9](#_Toc104231148)

[Database Structure 10](#_Toc104231149)

[Project timeline 10](#_Toc104231150)

[Project Distribution: 11](#_Toc104231151)

# Overview

This project aims to research, plan, design and build a single-page web application. The Task is to develop a  *“to-do list”* using Node.js, Angular and MongoDB.

The main requirements for this project are to have an app that allows users to sign up, log in, create tasks, set tasks, assign others to tasks, and get notifications of incomplete tasks, and these features we might come up with. To address all those requirements, we developed an application that helps students manage their to-do lists. Students can log in/log out of the application, create folders to organize their to-dos, and also create each Task independently inside those folders. They can also choose a deadline for each Task and assign a student to complete it. Besides, they can delete/edit folders and tasks. The application is based on the following stack:



The project is based on a MEAN stack, which is defined as a Javascript-based framework that makes use of the following technologies:

* M 🡪 MongoDB 🡪 Database
* E 🡪 ExpressJs 🡪 nodeJs web framework
* A 🡪 Angular 🡪 Js client-side
* N 🡪 Node 🡪 Js Web Server

The application also used some libraries that we deemed necessary to complete the Task; one of the libraries is *Bulma*, an open-source CSS framework based on flexbox and built with Sass.

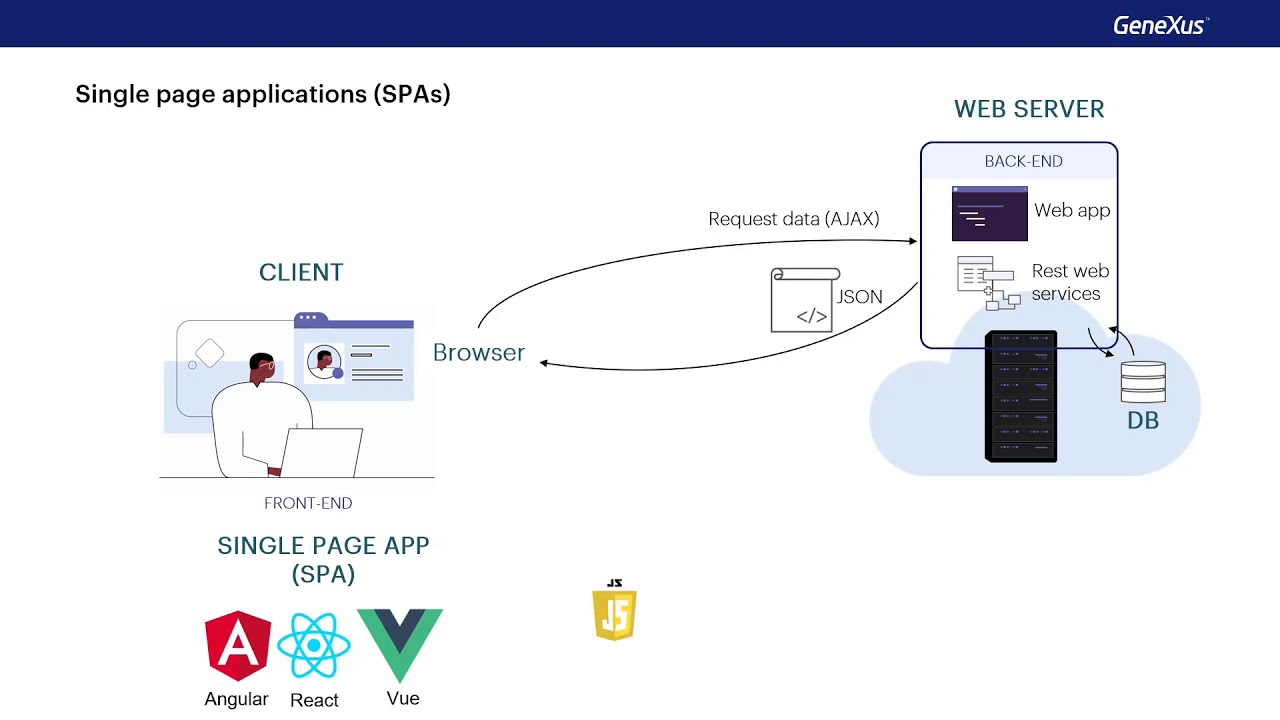
# Features

|  |  |  |
| --- | --- | --- |
| Table of Implemented Features | | |
| # | Feature | Implemented |
| 1 | Homepage | Checkmark outline |
| 2 | Sign up page | Checkmark outline |
| 3 | Login page | Checkmark outline |
| 4 | Create a folder | Checkmark outline |
| 5 | Create a task | Checkmark outline |
| 6 | Edit a task | Checkmark outline |
| 7 | Delete a task | Checkmark outline |
| 8 | Create more folders | Checkmark outline |
| 9 | Delete folders | Checkmark outline |
| 10 | Edit folders name | Checkmark outline |
| 11 | Set a task as complete | Checkmark outline |
| 12 | Assign others to a task | Badge Cross outline |
| 13 | Get alerts about incomplete tasks | Badge Cross outline |
| 14 | Logout | Checkmark outline |
| 15 | Hashing password | Checkmark outline |
| 16 | Sessions | Checkmark outline |
| 17 | Tokens | Checkmark outline |

# Architecture

The application developed demands a high interactively and fast response time, so the architecture chosen was based on single-page applications to perform better. In this type of architecture, the application is sent to the browser, and the page is not reloaded while in use. Therefore, providing a user experience similar to a desktop application.

This implies that the client handles all the functionalities, including HTML code generation since the web application runs on a single page. The resources are dynamically loaded from the web server in response to user actions. To this end, the code on the server includes services, in other words. These programs respond to the request made by the client, like retrieving data from the database. However, the application logic is on the client.

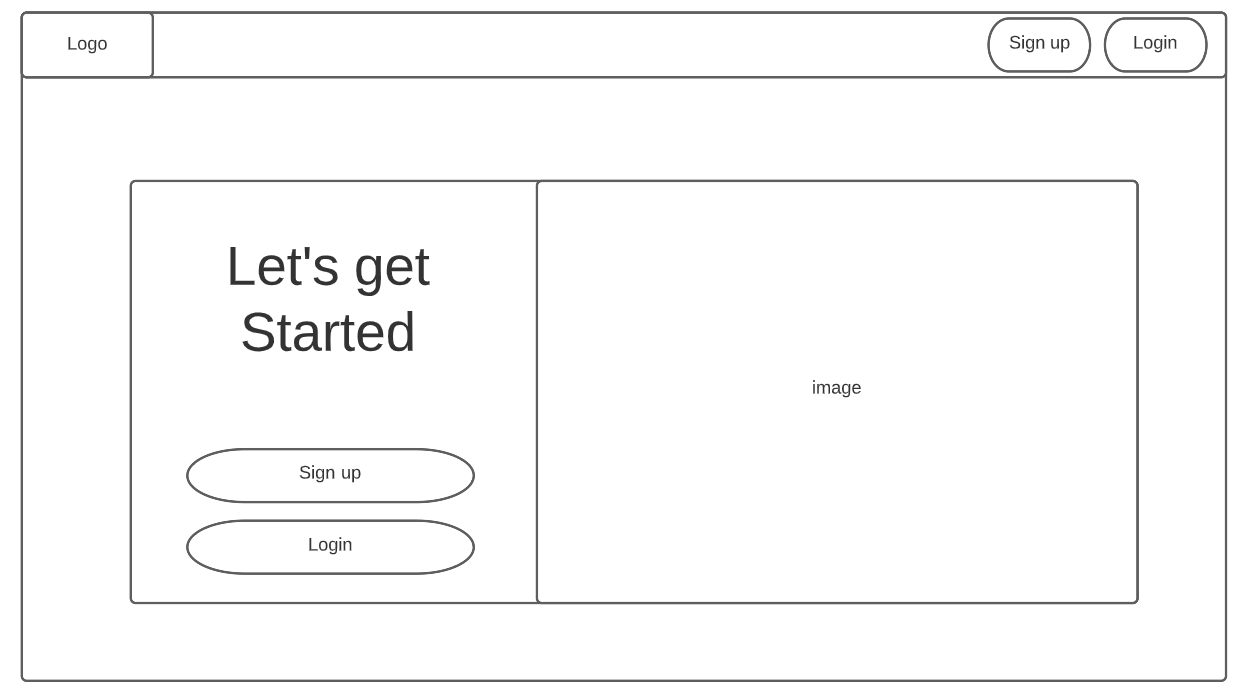


When the browser requires a URL from the server, the server responds with the page in formatted and decorated HTML code. Then the client can use Ajax to request data from the server by invoking the corresponding API, which obtains the data from the database, and returns the information in JSON format to the client. As a result, it prevents the page from reloading, thus achieving better performance and enhancing the user experience. To implement this type of solution, we use javascript frameworks, such as Angular, React or Vue.

# Wireframes

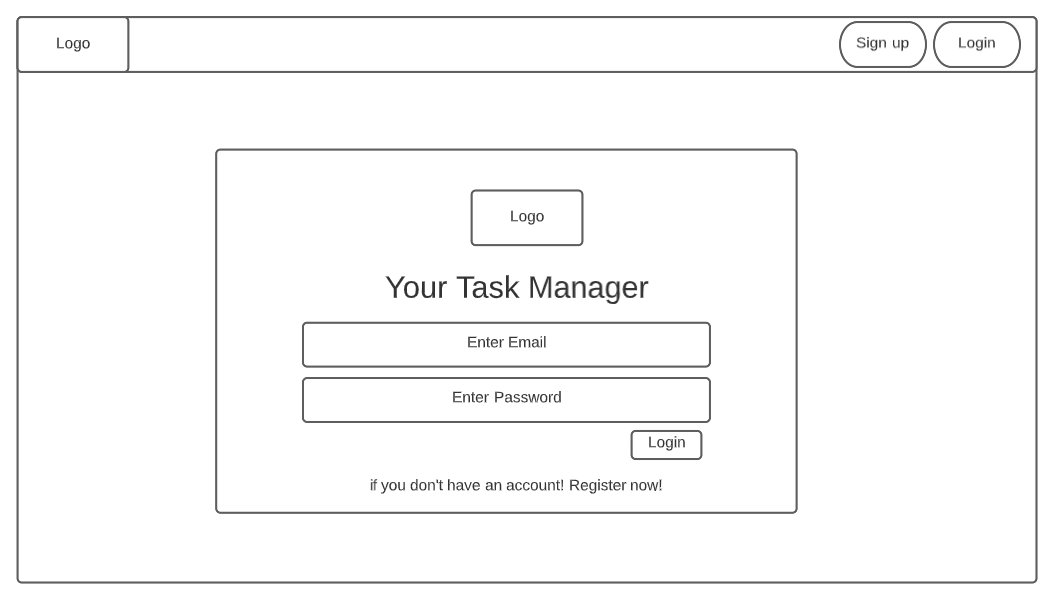
### Homepage

The home page application allows the user to create an account by clicking the signup button and logging in. The ain colour chosen for the app is ***#b30000;*** the colour is based on the original colour used by Griffith College, so that reminds the students that this can be used for college tasks.



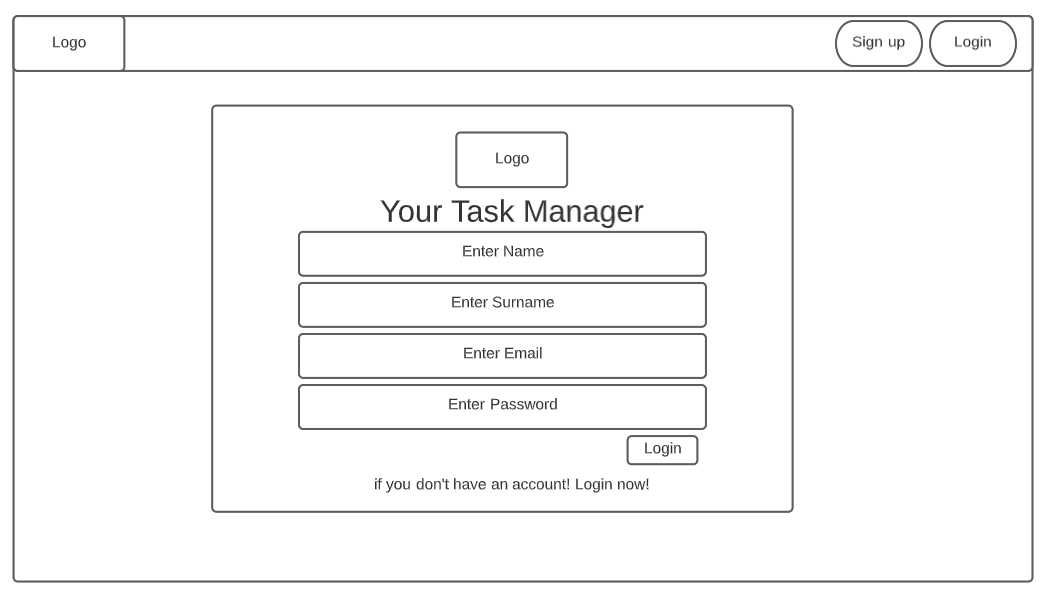
### Login/Logout Page

On the login page, the student must enter their email and password, or if they don’t have an account, they can use the signup button or link, and they will be redirected to the signup page.



### Signup Page

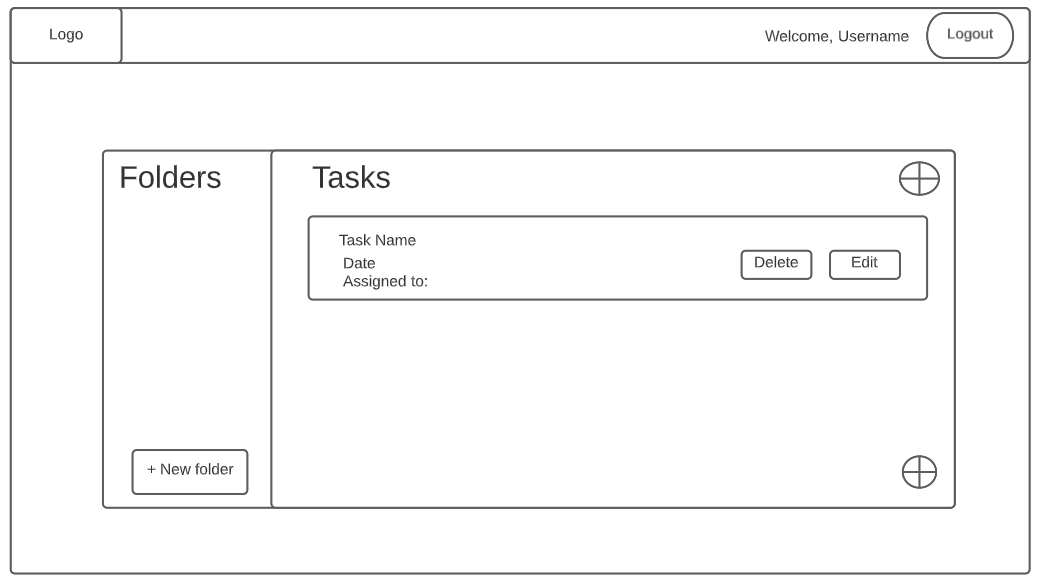
On the signup page, the student must enter their name, surname, email and password, or if they already have an account, they can use the login button or link, and they will be redirected to the login page.



### Tasks page

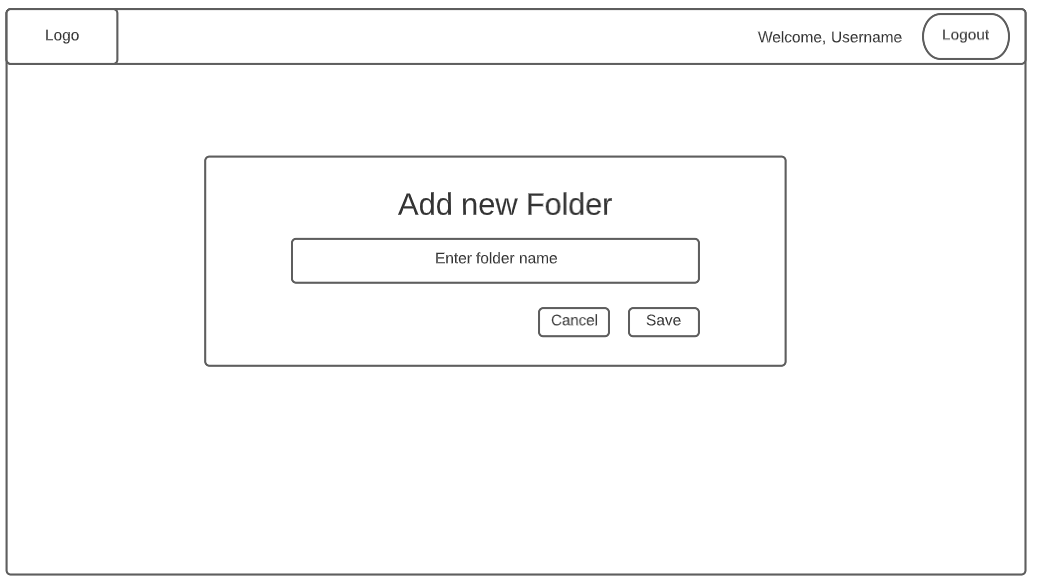
The task page is defined as the main page of the application. The user can create folders to organize their to-do tasks. Besides, the user can log out of the application in the top hand corner. Once the user is logged in, their name appears with a welcoming message.

In the tasks section, students can manage their tasks by adding a new one, editing them or marking as complete, to mark as complete students must click on the Task they wish to mark as completed.



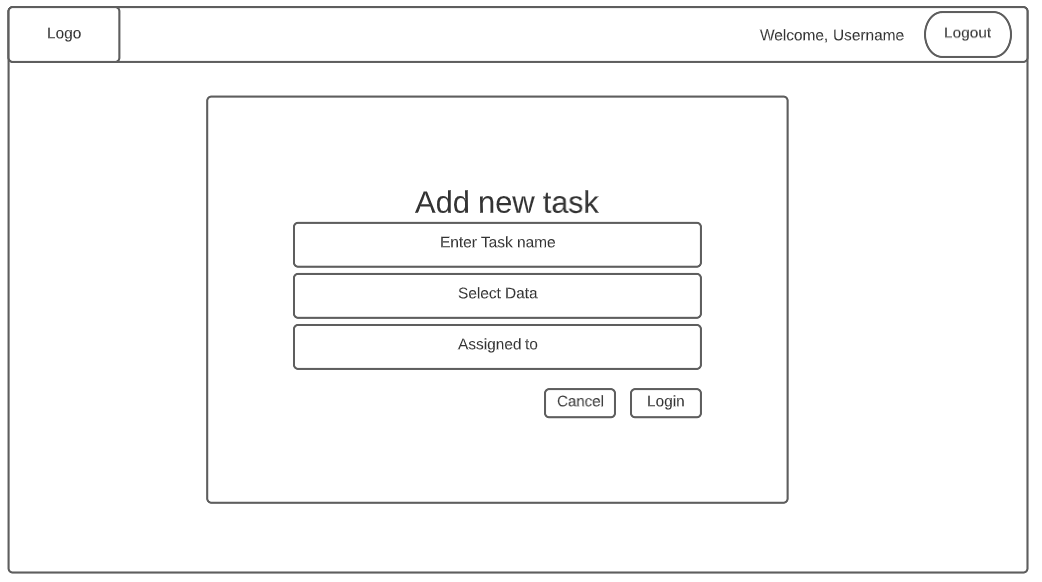
### Add new Folder

Students can create a new folder by clicking on the + New Folder button, and they will be redirected to this page, where they can enter a name for the task folder.

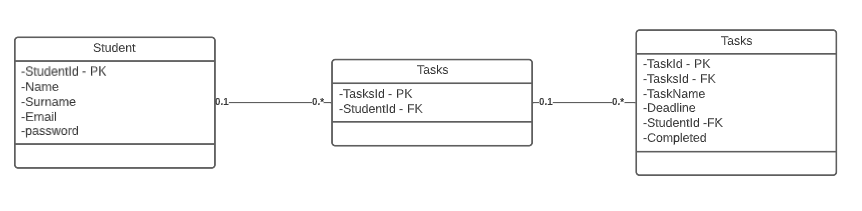


### Add new Task

When creating tasks within the task folders, students can enter a name to the Task, set the deadline and assign it to a student. Unfortuallity, due to the timeframe, we couldn’t implement the notification and a proper assignment to another student, so it must be entered manually.



# Database Structure



The database structure is based on the main tables: Student, Tasks and Task. The student table holds information about the student, and the Tasks table contains all the information about its tasks.

# Project timeline

Day of Title issue: 18/04/2022

Project due date: 09/05/2022

Project submission date: 23/05/2022

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 18/04 | 25/04 | 02/05 | 09/05 | 16/05 | 23/05 |
| Project Title Issue |  |  |  |  |  |  |
| Review Content – Learning Curve |  |  |  |  |  |  |
| Design Wireframes |  |  |  |  |  |  |
| Implement project home page |  |  |  |  |  |  |
| Setup Database schema |  |  |  |  |  |  |
| Implement Signup/Login pages |  |  |  |  |  |  |
| Implement Task main page |  |  |  |  |  |  |
| Implement logout functionality |  |  |  |  |  |  |
| Implement add new Folder |  |  |  |  |  |  |
| Implement add new Task |  |  |  |  |  |  |
| Implement Edit/Delete functions |  |  |  |  |  |  |
| Test all functionalities |  |  |  |  |  |  |
| Complete documentation report |  |  |  |  |  |  |
| Deploy app on stackblitz |  |  |  |  |  |  |
| Deliver all files and documentation |  |  |  |  |  |  |

### Project Distribution:

|  |  |
| --- | --- |
| Colour | Student |
|  | Bruna Minhos |
|  | Jocimara Melo |
|  | Rodrigo Bezerra |
|  | All |