Initial Ideas and Working Features

The idea to the project given was to implement a system that has a message board. This involved a login screen and message board where we can view all messages and search them.

My idea of this was to create a task application where the user can login and put any tasks he would like to do and if the task has been completed, he can then set the task as published.

As I Started to do the application I realized the app could be done multiple ways with a lot of technologies to use. I tried to stick to the basics to demonstrate my understanding of application development. I believe I still have a lot to do in this application but due to my lack of experience in advanced application implementation I tried to use the basic but powerful options I could learn and demonstrate.

I the following I will explain my approach, ideas and Implementations that I still will be working on and learn from.

Client Server Datastore

Angular

Express

CURD Restful API

MongoDB

Routes

Message. Route

Controllers

Message

Models

Index, message

Mongoose

Guard

Services

Auth.service,

data. service

Components

Add, List, details, Navbar, Auth

My Implementation:

Front-end:

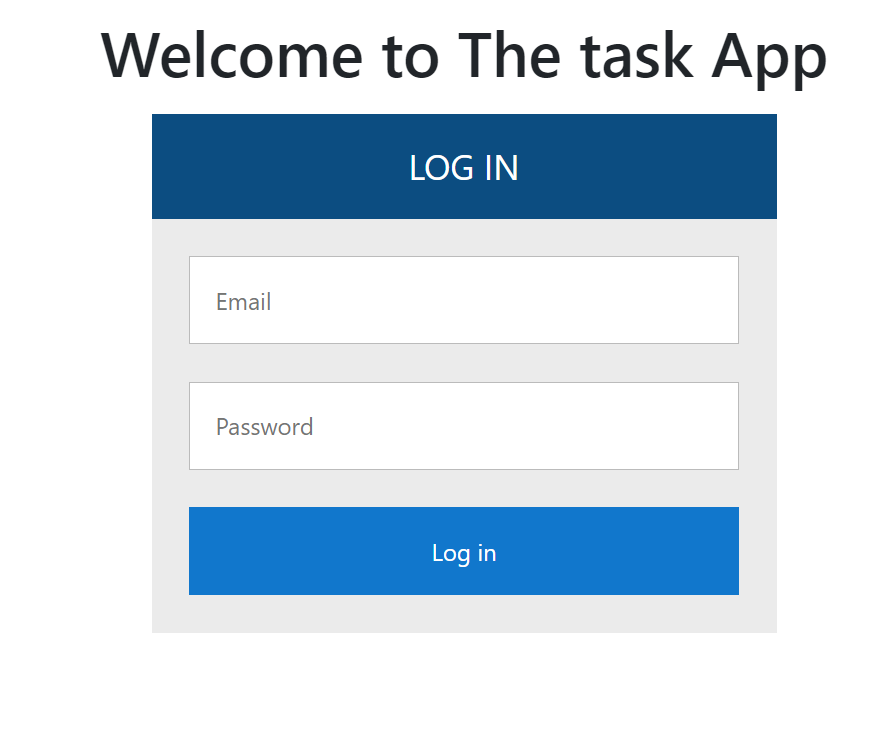
The Application uses Angular for client view such as a login screen, add page to add elements to the list, List page and detail component to view all elements stored. Other features such as a search bar to search by text and delete all. On active select you can view details regarding the task.

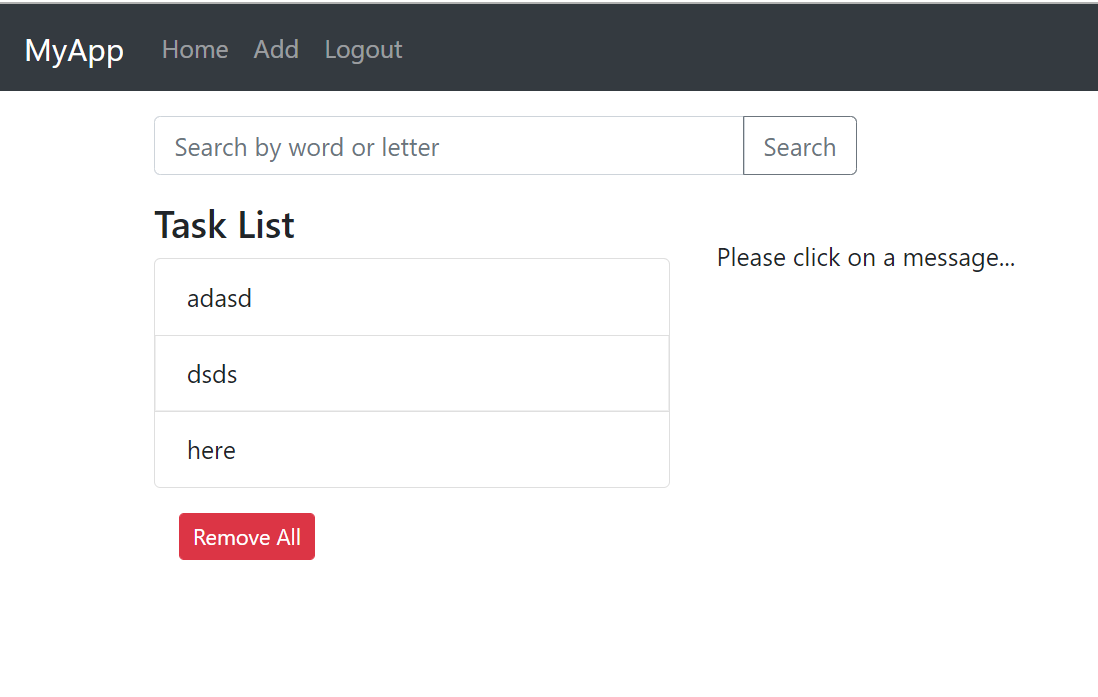
At the login screen you will requested to enter email and password, this component is a work in progress as I would like to add a registration component and a backend to store the details. But the login does have validation such as correct email and password over 4 characters.

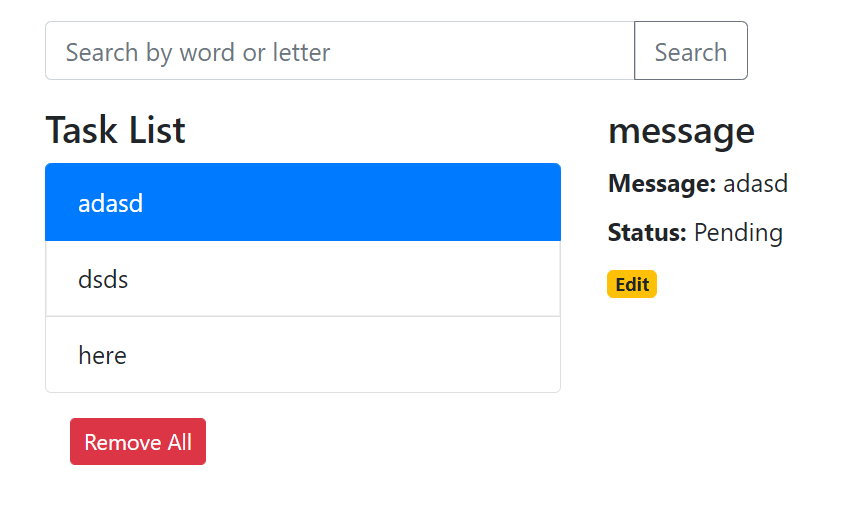
On login you will be show page containing your tasks, where you can use the search bar to find one or use the mouse to select one. On selection you can view the detail such as published and an option to edit. On clicking edit you will go to the edit component where you can change you text or change the published element. You can also delete the component here. The search using tags is still a work InProgress as I have not worked on tag system before, but I am in the process of learning it as we speak. But the element will recognise the tag element and display.

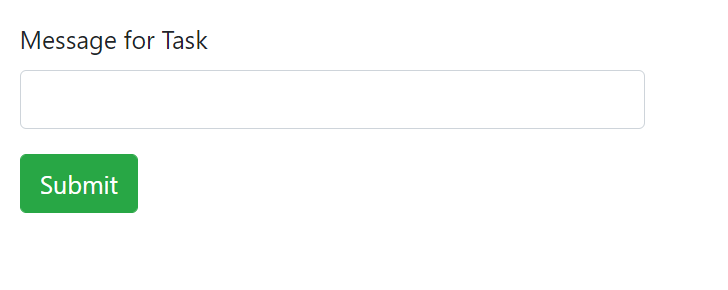
A logout button is provided where it will bring you back to your login-in screen. A routing element was implemented for page traversal along with a guard which makes sure the pages will does not access directly until you use a proper login page.

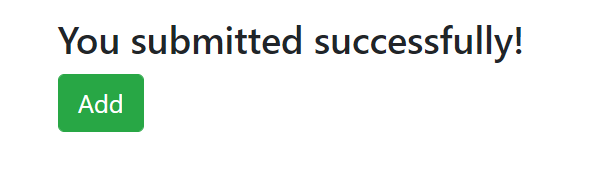
Images:

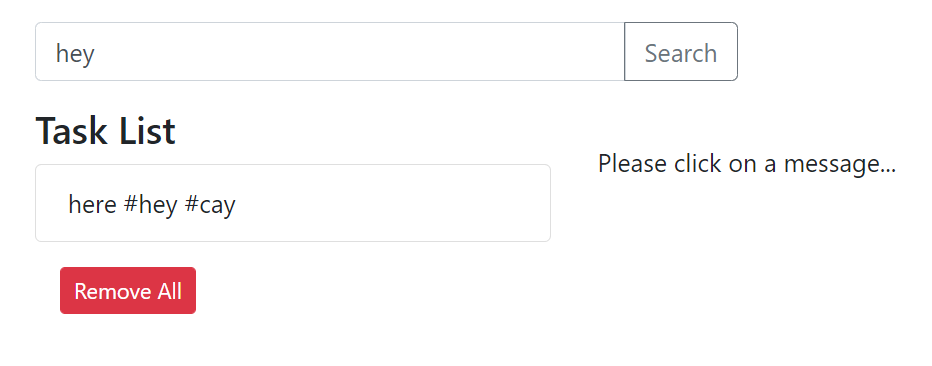












Server/Backend:

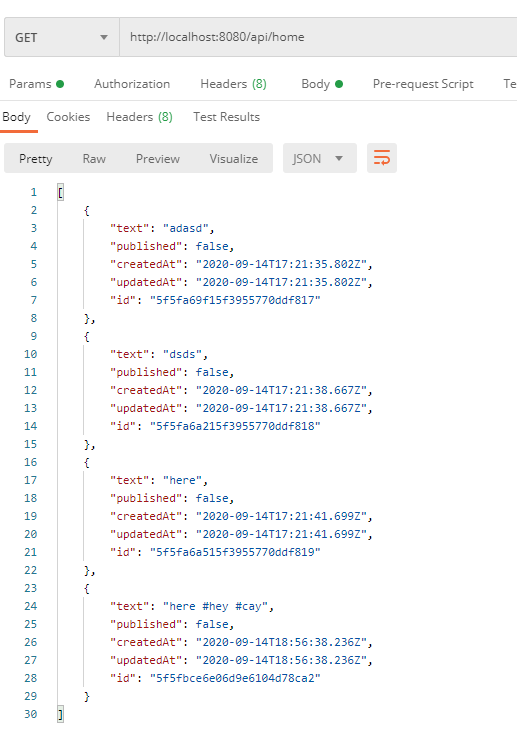
The backend uses a mongoose server with the express, Node.js and MongoDB.

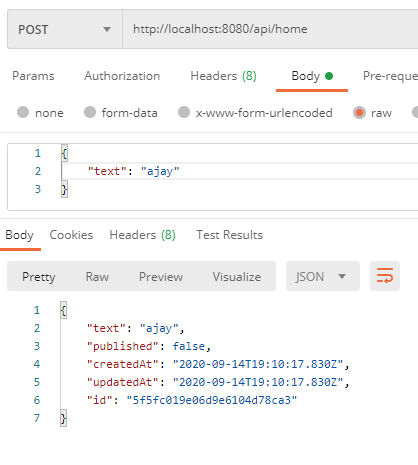
This consist of API where we can create, retrieve, update, delete and find. First, we start with an Express web server. Next, we add configuration for MongoDB database, create message model with Mongoose, write the controller. Then we define routes for handling all CRUD operations (including custom finder).

This server connects the DB with the front end where it takes care of any requests. We create a model and controller to define the calls and structure necessary. The server automatically initializes a DB upon call and decrease and need to maintain it directly.

| **Methods** | **Urls** | **Actions** |
| --- | --- | --- |
| GET | api/messages | get all Messages |
| GET | api/messages/:id | get Message by id |
| POST | api/messages | add new Message |
| PUT | api/messages/:id | update Message by id |
| DELETE | api/messages/:id | remove Message by id |
| DELETE | api/messages | remove all Messages |
| GET | api/messages/published | find all published Messages |
| GET | api/messages?text=[kw] | find all Messages which text contains 'kw' |

Images:





MongoDb:



Tasks Working on, Ideas and reason for using this:

As I was trying to plan the application, I came across multiple methods such as using a firebase, MySQL or MongoDB database. I decided to use NoSQL MongoDB as it is easy maintenance and I can use a controller from a server. For future development the DB can be easily manipulated rather than an Sql based system. Express system provided me with tools that decrease the time to deploy and maintain. I used Mongoose as my server using it alongside MongoDB and express helps me well structure my deployment. As the application is Js based it can well integrate with angular.

I decide to use angular because it is a framework that has SPA/ single page application. It deploy one and run anywhere provide me the right tools for structuring my needs.   
The Application can be greatly improved on elements such as security, my idea for this is using the JWT token system using spring boot or in angular. A registration page can be used alongside a backend for it to store and retrieve the authentication. Due to the time and resources available I prioritized building an application that has working features. I would love to provide features such as pagination to the application.

This has been a great learning experience for me to experiment and would love if the company can help me experience this under the a guidance of a team.