# Angular Project Assignment

Your task is to **design** and **implement** a web application (Single Page Application) using Angular. Use a service like Kinvey or Firebase for your **back-end** or create your own with Node.js and MongoDB or a framework in another language (ASP.NET, Spring, Symfony). It can be a **discussion forum, blog system, e-commerce site, online gaming site, social network,** or any other web application of your choice.

**Note**: The back-end part of your project must function properly, but it will NOT be considered during the exam. This is a front-end course and only the front-end part will be assessed.

## Submission Deadline

* You **must** submit a **link** to your project **before 23:59h on 30-November-2024** using a survey that will show up on **23-November-2024**.
* You **can continue working** on your project until the **end** of **11-Dec-2024** (**23:59h**).
* A presentation **schedule** will be available on **12-Dec-2024** and will include only the projects **submitted beforehand**. Non-submitted projects will **NOT** be evaluated.

## Application Structure

The application should have:

• Public Part (Accessible without authentication)

• Private Part (Available for Registered Users)

### 1.1 Public Part

The public part of your projects should be visible **without authentication**. This public part could be the application start page, the user login, and user registration forms, as well as the public data of the users, e.g., the blog posts in a blog system, the public offers in a bid system, the products in an e-commerce system, etc.

**1.2 Private Part (User Area)**

Registered users should have a personal area in the web application **accessible after** **successful login**. This area could hold for example the user's profiles management functionality, the user's offers in a bid system, the user's posts in a blog system, the user's photos in a photo-sharing system, the user's contacts in a social network, etc.

**Note**: A logged-in user should stay logged-in after refreshing the page.

## General Requirements

Your Web application should use the following technologies, frameworks, and development techniques:

* At least 3 different **dynamic pages**: pages that render dynamic data (pages like about, contacts, login, register, create, etc. do NOT count towards that figure)
* Must have specific **views**:
  + **Catalog** – list of all created records
  + **Details** – information about a specific record
* At least one collection, different from the User collection, with all CRUD operations (create, read, update, delete)
  + **Logged-in users** should be able to create **records** (by sending **requests** to the REST API) and **interact** with those records (via likes, dislikes, comments, etc.)
  + **Logged-in users** should be able to **edit** and **delete** their own records
  + A **guest user** should have **access** to basic website **information** (catalog, details), but **not** to the **functional activities**
* Use Angular for the **client-side**
* Communicate to a **remote service** (via REST, sockets, GraphQL, or a similar client-server technique)
* Implement **client-side routing to at least 4 pages (at least 1 with parameters)**
* **Meaningful commits** with **descriptive commit messages** to a source control system like GitHub, Bitbucket, etc., **for at least 3 days**
* The **GitHub** repository must be **public**.

**IMPORTANT:** If your project **doesn't cover** these conditions, you will **not** be graded!

## Other requirements

* Apply **error handling** (conditional rendering based on error response) and **data validation** to avoid crashes when invalid data is entered
* Use appropriate folder structure
* Brief **documentation** describing the project (used frameworks and libraries, how to run it, functionality, architecture) **as .md file**
* Demonstrate use of the following programming concepts, **specific to the Angular framework**:
  + TypeScript with specific types (avoid the type "any")
  + at least 2 interfaces
  + observables
  + at least 2 RxJS operators
  + lifecycle hooks (ngOnInit, ngOnDestroy)
  + pipes (date format, shorten, …)
* Component Styling (use **at least some external CSS files**)
* Implement route guards for the private AND the public part: guest users shouldn‘t be able to access private pages, logged-in users shouldn‘t be able to see the login/register pages
* Interact with the records (via likes, dislikes, comments, etc.) by sending requests to the REST API
* Good usability. Good UI and UX (**You can follow the Design Best Practice guide**)

## Bonuses

* Deploy the application in a **cloud environment.**
* Use a **file storage cloud API**, e.g. **Dropbox**, **Google Drive** or other for storing the files.
* Use of features of HTML 5 like Geolocation, SVG, Canvas, etc.
* Use **Angular Animations** somewhere in your application.
* Write **unit tests** for your components.
* Use RxJS powered **state management** for Angular applications, inspired by Redux (ngRx store)
* Anything that is not described in the assignment is a bonus if it has some practical use.

**Note**: You will NOT receive any bonus points for back-end customizations!

## Public Project Defense

Each student will have to deliver a **public defense** of their work in front of the other students, trainers, and assistants. Students will have **20 minutes** for the following:

* **Demonstrate** how the application works (very shortly)
* Show the **source code** and explain how it works
* Show any bonus functionalities they have implemented
* Answer **questions**

Please be **strict in the timing**! On the **10th** minute, your presentation ends. The remaining time will be for a **question-and-answer** session.

Be **well prepared** to present the maximum of your work for the minimum amount of time. Open **the project assets** beforehand to save time.

The project defense will be happening **online** through **Discord.**

## Assessment Criteria

### General Requirements – 30 %

Implementing all the general requirements will grant you a place on the defense schedule. All projects that do not have the general requirements will not be accepted for defense.

### Other Requirements – 45 %

### Functionality Presentation – 5 %

Adequately demonstrate the requested functionality. Know your way around the application and quickly demonstrate the code.

### Answering Questions – 20 %

Answer questions about potential functionality outside the scope of the project.

### Bonuses – up to 10 %

Additional functionality or libraries outside the general requirements, with motivated usage.

## Restrictions

You can use **parts** (some components, routing configurations, form validation, etc...) of the **course workshop**, but you are **NOT** allowed to use the **whole workshop** as your project assignment. You are **NOT** allowed to use **HTML & CSS** structures from any SoftUni course.

## Project Challenge

The **best three projects** will win a discount for the next course or module:

* First place – 80% discount voucher
* Second place – 50% discount voucher
* Third place – 30% discount voucher

The ranking of the projects is done **based only on the submitted project** (it does not include the assessment of the theoretical exam). Please make sure your project works when downloaded from the repository and keep it available at the same link up to 3 weeks after the exam.

The voucher could be used for **one course or one module in the open or the professional program at SoftUni**. It **cannot be divided** into parts or **given to another person**. The voucher is valid for **one year** after the announcement of the winners.