

UECS3563 Industry Professional Domain Skill Development

June 2025

Assignment

Full stack app development with Angular and Spring Boot

Assignment deadline: **Thursday Oct 9**

1 Objective and evaluation criteria

The primary objective of this assignment is to evaluate and validate students' understanding and ability to effectively implement the specific full stack libraries / frameworks (**Angular** and **Spring Boot**) taught in the micro-credential training program that forms the core training portion of WBL of this programme.

Key competencies that will be evaluated

- a) Understanding the high-level architecture of a full stack app and how the various key parts are composed and interact together.
- b) Development of a RESTful web service exposing API endpoints that adhere to API best practices
- c) Implementing standard CRUD operations and more complex queries within the web service to interact with a typical RDBMS (MySQL, Oracle, MS SQL Server, etc) database to persist and manipulate business domain data
- d) Implementing core dynamic and reactive features in the front-end web application
- e) Implementation of data entry and validation in appropriate form structures
- f) Implementing appropriate HTTP requests from the front-end to the API endpoints of the REST web service in proper asynchronous fashion
- g) Ability to document, analyse and explain code structure, functionality in a clear and effective manner which facilitates collaborative team effort in a software project

2 Full stack app implementation

Implement a full stack app using the **Angular framework for the front-end** and **Spring Boot framework for the back-end**. The overall design direction of this app should be based as far as possible on the **expected deliverables outlined in your FYP project requirements**, but the majority or all of this business logic can be mocked or simulated (i.e. basic code that produces dummy results). The actual business logic will be implemented in your final application deliverable for FYP2. The primary focus here is on implementing the key features of both frameworks in order to develop a full stack web application.

3 Implementing the Angular front end

Implement at least 10 of these features listed below (you can implement more if you wish). Ensure you include comments to indicate accurately which features you have implemented.

| No. | Feature |
|-----|--|
| 1. | Dynamic page changes via interpolation and property binding with appropriate template expressions |
| 2. | Dynamic page changes via class and style binding and/or NgClass and NgStyle together with basic CSS style rules |
| 3. | Event binding to respond to user events on web page |
| 4. | App should be structured as a component hierarchy consisting of root, child, grand-child components |
| 5. | Data should be transferred between components in the hierarchy using @Input and @Output |
| 6. | Conditional dynamic page changes using @if and @else or @switch and @case directives |
| 7. | Rendering of multiple HTML elements and/or child components using @for. Include appropriate use of loop contextual variables \$index, \$count, \$first. |
| 8. | Create reactive forms with multiple grouped form control elements to accept user input |
| 9. | Implement validation of form controls as well as appropriate display of validation error messages |
| 10 | Use HttpClient library to send standard HTTP requests (GET, POST, PUT, DELETE) to backend REST API service endpoints with correct use of the various relevant classes (Observables, Subscriptions, etc) of RxJS to achieve asynchronous network interaction. |
| 11 | Registering and matching routes to components with appropriate URL path mappings, including route redirecting and wildcard routes. |
| 12 | Passing route parameters and query parameters between routed components |
| 13 | Creating and navigating to nested / child routes where appropriate |
| 14 | Implementing programmatic navigation to routes where appropriate |
| 15 | Provide short comments in your codebase to explain all the features that you have implemented above. You do not need to provide comments for other common code functionality such as importing libraries, modules, declaring functions and properties, initialization with constructors, etc |
| | |

4 Implementing the Spring Boot Backend

Implement at least 4 of these features listed below (you can implement more if you wish). Ensure you include comments to indicate accurately which features you have implemented.

| No. | Feature |
|-----|---|
| 1. | Design a list of suitable API endpoints to be implemented by your Spring REST Web service which should align with existing REST API design best practices as far as possible . These API endpoints should conceptually map to business logic functionality that is expected to be performed by the backend app in accordance to the expected deliverables outlined in your FYP project requirements. Some of your API endpoints should include both query and/or path parameters as appropriate |

| | |
|----|---|
| | |
| 2. | <p>Using the appropriate Spring Boot starter dependencies (Spring Web, Spring Data JPA, Project Lombok, Spring Boot DevTools, etc), implement these API endpoints in appropriate classes and methods with the correct annotations (for e.g. @RestController, @RequestMapping, @GetMapping, @PostMapping, @QueryParam, @Path Variable, etc)</p> <p>The body of your handler methods that handle incoming HTTP requests can provide simple functionality to return dummy data, rather than actual implementation of the required business logic functionality (as this will only be completed in FYP2).</p> |
| 3. | Design one or more suitable domain classes to model the business entities that you expect to be used for your business logic. Initialize the backend database that you intend to use in conjunction with your webservice with one or more tables that contain initial records that correspond to these business domain classes with accords with Spring Data JPA underlying Hibernate ORM mapping. You can optionally initialize these tables with direct SQL commands or by configuring a basic Spring Boot app with existing schema / table files |
| 4. | Implement basic CRUD operations (which can include sorting) in appropriate handler methods for the domain data on the backend database table content using Spring Data JPA various xxxRepository classes (CrudRepository, PagingAndSortingRepository, etc) |
| 5. | Implement one or more derived queries as well as one or more native SQL or Jakarta Query Persistence Language (JPQL) queries in appropriate handler methods for more complex queries on the backend database table content |
| 6. | Implement appropriate exception handling code in your web service to return custom error messages that clearly identify issues with problematic HTTP requests from the client (incorrect JSON content structure, malformed JSON, invalid API endpoint URL path portions, etc). This should also include customization of HTTP response status codes and headers if appropriate as well. Exception handling code should also handle potential server side errors and return a suitable custom error message in this situation |
| 7. | Provide short comments in your codebase to explain all the features that you have implemented above. You do not need to provide comments for other common code functionality such as importing libraries, modules, declaring functions and properties, initialization with constructors, etc |

5 Evaluation and analysis

Write a short report where you analyze and evaluate how the specific features that you implemented for your Angular and Spring Boot portions contribute to the overall implementation of your final FYP2 app and why you feel they are suitable for that purpose.

Examples of such analysis / evaluation might involve addressing questions such as below:

- a) Why the specific choice of API end point names and/or parameters for your backend RESTful service? How do this help map to the specific business logic that your backend RESTful service is expected to implement ?
- b) Why the specific choice of components and routes for navigation in your front-end ?
- c) Why are specific bindings or directives being used in any particular component?
- d) Why perform HTTP client calls in this specific component rather than another?
- e) Why are reactive forms designed and validated in that particular manner and fashion that you have done?
- f) How does all of the above relate to the end-user UI perspective of interacting with the web app to facilitate the business logic of the backend ?

These are not the ONLY issues you can consider; exercise your judgment to determine other similar appropriate ones as well.

6 Submission

Upload your completed codebase for both the Angular front end and Java Spring backend as well as report to a GitHub repo and document this with an [appropriate README](#). An example of an appropriately [documented Angular app](#).

Some links (all similar content) on how to push your codebase to a GitHub Repo:

<https://www.youtube.com/watch?v=vpRkAoCqX3o>

<https://www.youtube.com/watch?v=ueQs5pQ8ZMM>

<https://www.youtube.com/watch?v=g2XjJhrGGg4>

<https://dev.to/brianhough/how-to-connect-your-local-projects-codebase-to-a-github-repository-fast-2l7h>

<https://www.geeksforgeeks.org/git/how-to-add-code-on-github-repository/>

You can then email me (victor.tan.33@gmail.com) with the URL of your GitHub repo.

7 Mark allocation and Course Outcomes (CO) assessed

Marks breakdown is as follows:

Angular features implemented: 40 marks

Spring Boot features implemented: 40 marks

Analysis / evaluation report: 20 marks

Total: 100 marks