

JAVA TECHNOLOGY

(503111)

LAB 9 - 10

OBJECTIVES

- Create Restful Web Service using Spring Boot.
- Learn the concept of Cross-Origin Resource Sharing (CORS).
- Use Spring Data JPA to store data in a database system.
- Learn Spring Security, Json web tokens and apply them to set up authentication and authorization for web services.

PROBLEM DESCRIPTION

Build a web service to manage objects including user accounts, products, and orders. The details of each object are as follows:

- **User accounts** include information: email, password, first and last name.
- **Products** have information: code, product name, price, illustration, description.
- The **orders** have the following information: order number, total selling price, and product list.

This web service exposes two endpoints `/products` and `/orders` that support various HTTP methods to perform functions such as add, update, delete, get list and get details of a product/order. There are also `/account/register` and `/account/login` endpoints to manage account registration and login. Some api endpoints require users to login before accessing. Input parameters for all endpoints are in **JSON** format. The detailed description of the api endpoints and supported methods are as follows:

- `http://localhost/api/account/register`:
 - POST: register a new account
- `http://localhost/api/account/login`:
 - POST: user login

- <http://localhost/api/products>:
 - GET: returns a list of all products in the system
 - POST: Add a new product
- <http://localhost/api/products/{id}>
 - GET: Returns detailed information of a product based on id.
 - PUT: Replace the entire product with with new data based on its id.
 - PATCH: Update some information of a product based on its id.
 - DELETE: Delete products based on id.
- <http://localhost/api/orders>:
 - GET: Returns a list of all orders.
 - POST: Add a new order.
- <http://localhost/api/orders/{id}>:
 - GET: Returns the details of an order based on id.
 - PUT: Update an order's information by id.
 - DELETE: Delete orders based on id.

For api endpoints with methods highlighted in red, users need to be logged in to access them.

OTHER REQUIREMENTS

- All data about accounts, products, orders need to be stored in a database connected from SpringBoot through [Spring Data JPA](#).
- Set up [Cross-Origin Resource Sharing](#) to allow any web client with different domain name to still access and interact with the web service.
- Use [bcrypt](#) library to hash passwords, use [JWT](#) to authenticate users.
- There should be a mechanism to handle errors and return appropriate error messages: for example, missing information, malformed information, file upload is too large, invalid endpoint error, method (http method)) unsupported...