Project Due on Tuesday May 24, 2016, end of the day, no late submissions

Version 1

Books Online Store Web application with a REST web services back end

Contents

Overview

In this Web Services course project you will have to build a web-based application with a web-services back-end.
The application you will be building is an online book store that allows the customer to **create / modify the account**, search for products (books) and create an order (buy a book).
You application should have 3 screens:

1. Create a new account screen (5%)
2. My account screen to display account info and customer's orders (5%)
3. Products screen that allows the users to search for products (books) and put in a new order into your back end system (10%)

Your HTML/JavaScript front end will be using the REST web-services that you design and develop as well as 3rd parties' web services. You will be using 3rd parties' web-services to retrieve a list of countries. And you will be using Google API to search for the books (your products).

You can use any java structures as your data store to keep track customers and orders – a HashMap should be fine – similar to our exercises in class. Your domain model should have at least 2 classes Customer and Order.

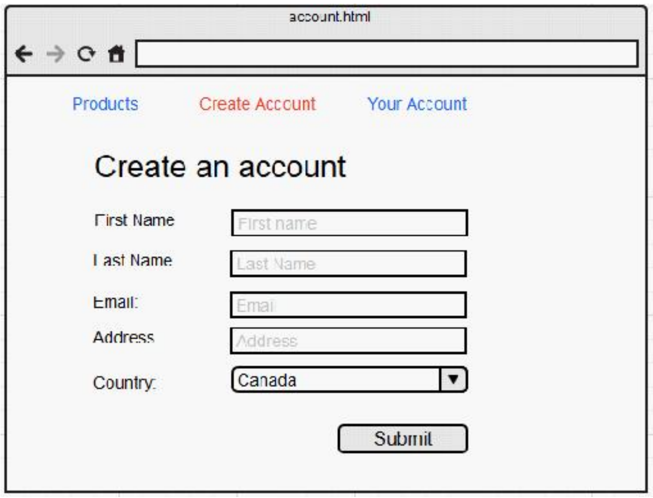
You have to be able to test your web services alone – using curl tool. I won't grade the usage of the JUnit but it is highly recommended to have the test unit coverage for your service class.

You have to **demo** your project in class, on May 24. If the demo is not completed you lose 20% of the grade for the project.

Task 1. Create New Customer (new account) screen

The screen shall have 4 fields (First name, Last Name, Email, Address) and one selection drop down list with country list The Country list drop down should use a 3rd party Web Service. A RESTful GET request and a simple response is below
http://services.groupkt.com/country/get/all

```
{
  "name" : "Canada",
  "alpha2_code" : "CA",
  "alpha3_code" : "CAN"
},
```



Create an account

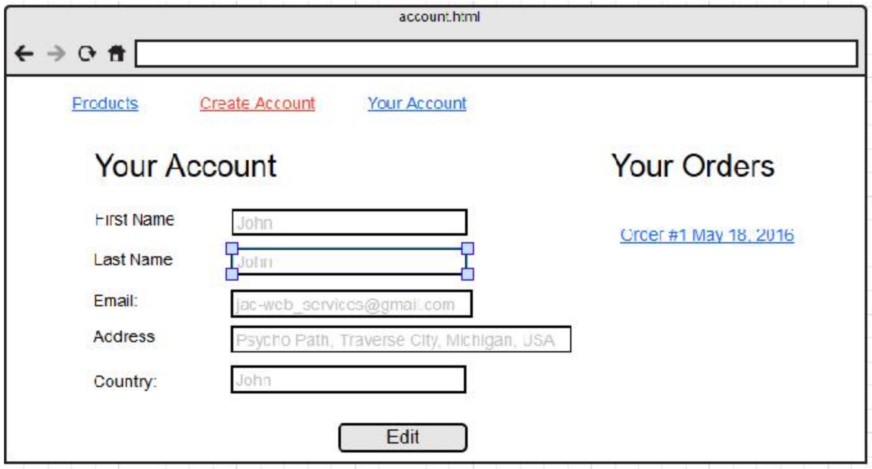
First Name	First name
Last Name	Last Name
Email:	Email
Address	Address
Country:	Canada

The 'Submit' button should send a REST request to your JAX-RS WS to create a new customer.
User input validation:

- Duplicate emails are not allowed
- all fields from Create Account screen are mandatory (names, email, address, country code)

The web service should return a response and your screen should display a confirmation (if the response code was HTTP 200) or an error message

Task 2. Your Account screen



Your Account

First Name	John
Last Name	John
Email:	jac-web_services@gmail.com
Address	Psycho Path, Traverse City, Michigan, USA
Country:	John

Your Orders

Order #1 May 18, 2016

Your account screen should retrieve the Customer's data (using the customer ID) and display the data on the screen. Use editable controls (inputs) so that the customer's data can be updated.

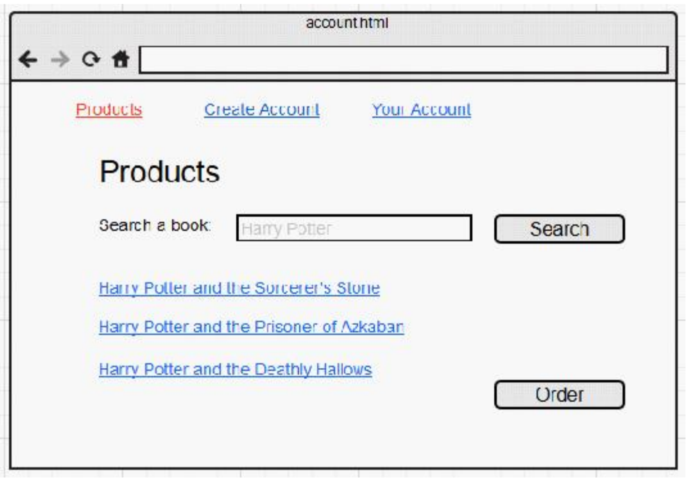
'Edit' button shall update the customer info.

The screen shall also display all orders that were created by the customer. The order should display the id as a link.

Task 3. Products screen

The **Products** screen should have a "Search for a book" field. The customer should be able to search for books. When the customer hits the 'Search' button the screen should send an Ajax request using [Google Books API] that returns a list of books.
A simple REST GET query to Google Books is below.

The response has a complex JSON structure but we ignore all the fields except the title and ISBN (a unique ID) and may be a price and an author. Your Java script should loop over a list of search results and build a piece of an html to display the list of books.


account.html
Products
Create Account
Your Account
Products
Search a book
Harry Potter
Search
Harry Potter and the Sorcerer's Stone
Harry Potter and the Prisoner of Azkaban
Harry Potter and the Deathly Hallows
Order

When user selects a single item your screen should generate a REST web service request to **create an order** (for simplicity your order will have just one item – the selected book). Your order data should have a customer id, product id (ISBN), product name, price

Simple Google Books API: `https://www.googleapis.com/books/v1/volumes?q=harry+potter`

Hints: Google Books API JavaScript example

```
<script type="text/javascript">
        function searchInBook() {
                var googleAPI = "https://www.googleapis.com/books/v1/volumes?q=harry+potter";

                // Make a ajax call to get the json data as response.
                $.getJSON(googleAPI, function(response) {

                        // Loop through all the items one-by-one
                        for (var i = 0; i < response.items.length; i++) {
                                alert("JSON Data: " + response.items[i]);

                                // set the item from the response object
                                var item = response.items[i];

                                // Set the book title in the div
                                document.getElementById("content").innerHTML += "<br>"
                                                + item.volumeInfo.title;
                        }
                });
        }
</script>
```