BSc (Honours) in Computing - Web Services & API Development - Project (50% of Module)

This is a group project, if you are not in a group contact the lecturer.

Problem Domain

Online Banking is a mainstream service offered by most banks today. A typical consumer online banking application requires the following:

A Customer Customers are individuals with a name, correspondence address, email and security credentials. A customer can hold one or more accounts.

An Account An account has a sort code (identifying the branch), an account number and a current balance, The account has a list of transactions.

A Transaction Each transaction is either a debit or credit on a particular date, with a description and post-transaction balance.

Customers will be able to do the following:

Create Customers should be able to create an account with the bank, and a customer who has an account should be able to add additional accounts. For example a typical customer may have a current account and a savings account.

Lodgement For the lodgement, a bank customer can specify the amount to lodge with the credit card that will be debited.

Transfer For the transfer, the bank customer can specify the amount to transfer and an account to transfer to.

Withdrawal For withdrawal, the bank customer can specify the amount to withdraw and the card that will be credited.

Balance The customer can request a balance on any account at any time.

Project Requirements

You are required to design, document, and develop, an API for the problem domain above. The document will have the following sections:

Introduction (5%)

The introduction will expand the problem statement to describe how the API fits together. For instance:

- Textual description of the problem and the proposed solution. (1 mark)
- An Entity-Relationship diagram describing the problem area, reflecting your database structure. (4 Marks)

Introduction is to be approximately 1.5 pages.

The RESTful API (30%)

The API describes all the entry points for the above problem domain. Each entry point (at least 15 entry points - 2 Marks for each) should be documented under the following headings:

API Name e.g. Users Resource
Description e.g. This allows the retrieval of all user resources
URI e.g. /users
HTTP verb e.g. GET
Parameters e.g. user_id (Integer, Required), name (String, Optional), for a POST this would be a JSON object
Resource contents e.g an example of the returned resource
Pre-Conditions e.g. no record for the user with the specified user_id must exist.
Post-Conditions e.g. a new record for the user with the specified user_id exists.

The design of the API will be assessed here in terms of its adherence to REST principles, readability, and the quality of the documentation. The documentation should allow anyone with knowledge of Web APIs to use your API with no issue. You will receive 0% for this section if you do not include this documentation despite having implemented it.

Prototype (65%)

The API prototype, must be implemented according to the following requirements:

- An HTML+JavaScript or Mobile or Desktop client calling ALL portions of the API. E.g., the client should check the account balance allowing a transfer or withdrawal. (10 Marks)

- A server developed in Java which implements ALL portions of the API using in-memory objects. Constraints should be implemented, balances should be updated and transactions should be remembered as the API is called. (20 Marks)
- OR A server developed in Java which implements ALL portions of the API using Java Persistence API (JPA) objects . Constraints should be implemented, balances should be updated and transactions should be remembered as the API is called. This should be achieved with a database layer (either memory or disk). (35 Marks)
- A server implementing ALL of the entry points returning resource representations in XML and JSON. (10 Marks)
- The API must be well secured with access points which are only available to specific users (by access level, role, and their identity), and protection from web attack vectors such as SQL injection (10 marks)

You must provide a video (or an accessible link to one) and screen shots to demonstrate the operation of the Web Service. This should step through the functionality of the project. There are no specific marks for this but if you don't submit it we will not be in a position to grade the project.

The prototype source code should also be included with the submission in a ZIP file.