Tests & Quizzes

CS425-SWE-202207-FinalExam

Return to Assessment List

Part 1 of 2 - 10.0 / 10.0 Points

Question 1 of 6 2.0 2.0 Points

Shown below is code for two Entity classes named, Product and Category, with an association relationship between them.

The Association relationship is Bidirectional. True or False? Give a reason for your answer (1 point)

```java
package model;

@Entity
public class Product {
    private Integer productId;
    private String productNumber;
    private String productName;
    private Category category;

    public Product() {
    }

    public Product(Integer productId, String productNumber,
        String productName, Category category) {
        this.productId = productId;
        this.productNumber = productNumber;
        this.productName = productName;
        this.category = category;
    }
    // TODO Getters and Setters
}
```

```java
package model;

@Entity
public class Category {
    private Integer categoryId;
    private String categoryName;
    private Product product;

    public Category() { }

    public Category(Integer categoryId,
                    String categoryName,
                    Product product) {
        this.categoryId = categoryId;
        this.categoryName = categoryName;
        this.product = product;
    }
    // TODO Getters and Setters
}
```

✔ ◯ True
◯ False

Rationale:
because there is the property of each class in the other class

Answer Key: True

Question 2 of 6 2.0 2.0 Points

The association relationship between Product and Category entities, as defined in the code shown below is:

```java
package model;

@Entity
public class Product {
    private Integer productId;
    private String productNumber;
    private String productName;
    private Category category;

    public Product() {
    }

    public Product(Integer productId, String productNumber,
        String productName, Category category) {
        this.productId = productId;
        this.productNumber = productNumber;
        this.productName = productName;
        this.category = category;
    }
    // TODO Getters and Setters
}
```

```java
package model;

@Entity
public class Category {
    private Integer categoryId;
    private String categoryName;
    private Product product;

    public Category() { }

    public Category(Integer categoryId,
                    String categoryName,
                    Product product) {
        this.categoryId = categoryId;
        this.categoryName = categoryName;
        this.product = product;
    }
    // TODO Getters and Setters
}
```

- ✔ ○ A. One-to-one
- ✔ ○ B. Many-to-many
- ✔ ○ C. All of the above
- ✔ ○ D. One-to-Many
- ✔ ○ E. None of the above

Answer Key: A

Question 3 of 6 2.0 2.0 Points

Given below is the code for a Product entity class.

```java
package model;

@Entity
public class Product {
    private Integer productId;
    private String productNumber;
    @NotEmpty
    private String productName;
    private Category category;

    public Product() {
    }

    public Product(Integer productId, String productNumber,
        String productName, Category category) {
        this.productId = productId;
        this.productNumber = productNumber;
        this.productName = productName;
        this.category = category;
    }
    // TODO Getters and Setters
}
```

Based on the @NotEmpty annotation placed on the productName attribute, the value, " ", which is a string of two spaces, will be validated as valid. True or False?

✔ ○ True
○ False

Answer Key: True

Question 4 of 6 2.0 2.0 Points

The class named, GlobalProductUtils, shown below, will be visible to all other classes across the system.

True or False?

Give a reason for your answer (1 point)

GlobalProductUtils.java

```java
package edu.mum.cs.cs425.prodmgmt.global.publicc;

class GlobalPublicUtils {

    public static final long MAX_INVENTORY_COUNT = 10000L;

    // TODO
}

```

✔ ○ True
○ False

Rationale:
because it doesnt has the right to be accessed

Answer Key: False

Question 5 of 6 2.0 2.0 Points

Of the 4 options given below, which one is NOT a correct/valid Spring RestController method annotation for an HTTP url endpoint?

- ✔ ○ A. @GetMapping
- ✔ ○ B. @ReadMapping
- ✔ ○ C. None of the above
- ✔ ○ D. @PostMapping

Answer Key: B

Part 2 of 2 - Part2 0.0 / 70.0 Points

Part2-Coding

Question 6 of 6 0.0 70.0 Points

Part II – Software Engineering Problem-solving, Coding skills: (70 points)

Implementing an end-to-end, full-stack data-driven enterprise web application:

Note 1: You are expected to use an IDE or any Code Editor tool of your choice to implement your solution for this question.

***Note 2**: For the tasks in this question, you are expected to take screenshot(s) of your result(s), save each into a .png or .jpg image file, placed inside a folder named, screenshots and include these in the FinalExam.zip file, you submit.*

Note 3: *For this question, when you complete your own solution, you are required to take each of the set of 6 evidential sample screenshots, which have been included at the end of the question. See below.*

Upon completion, to submit, put your entire project(s), into a single zip file named, say, **FinalExam.zip**, and upload it here as your submission to this question.

Problem Statement:

Assume you have been hired by a bank named, Special Bank of Burlington. And they want you to design and develop a web-based software solution, which they call, "Customer-Account Management system", which they will be using to manage the customer-accounts data.

Especially important to the Bank Manager is, the current Liquidity Position of the bank, which is calculated by summing the total balance of all the customer-accounts. Also important to the Bank Manager, is the **Prime Accounts. A Prime Account** is account whose balance is greater than $10,000.

Here is the simple solution model for the system:

A Customer can own one and only one Account.

And, each Account is owned by only one Customer.

Your solution model should consist of only the following two data entity:

1. Customer
2. Account

Here are the attributes for the entities, including some useful descriptions and/or sample data values:

Customer:

customerId: long, (Primary Key field)

firstName, (required field) (e.g. Bob, Anna, Carlos etc.)

lastName, (required field) (e.g. Jones, Smith etc.)

Account:

accountId: long, (Primary Key field)

accountNumber, (required field; unique) (e.g. AC1001, AC1002, etc.)

accountType, (required field) (e.g. Checking, Savings)

dateOpened, (optional field) (e.g. 2021-12-3, 2022-5-21, etc.)

balance, (required field) – This is the amount of money in the account

Data:

Here is Bank's existing data, which you are expected to input into your database:

Customer data:

Customer Id	First Name	Last Name
1	Bob	Jones
2	Anna	Smith
3	Carlos	Jimenez

Account data:

AccountId	Account Number	Account Type	Date Opened	Balance	Customer Id
1	AC1002	Checking	2022-07-10	10900.50	2
2	AC1001	Savings	2021-11-15	125.95	1
3	AC1003	Savings	2022-07-11	15000	3

For this question, you are required to do the following:

1. Sketch a simple UML Static (class/domain) model for the solution.
2. Using the set of tools, technologies and frameworks which you have learnt about in this CS425-Software Engineering course, including Spring Boot, Spring Web MVC, Spring Data JPA, etc., (or some other Enterprise Web application development platform/tool(s) that you prefer), implement a working web application for the Bank. You may use any database of your choice.

You are expected to implement only the following features and use-cases:

1. Display a homepage which presents a set of menu options. (see sample screenshot below)
2. Display list of all Customer-Accounts (Allows the user to view a list of all the customer-accounts registered in the system). The Bank requires this list to be displayed sorted in descending order of the Account balance amounts.

Also, display at the bottom of this list, the Liquidity Position of the bank, which is the sum of all Account balances.

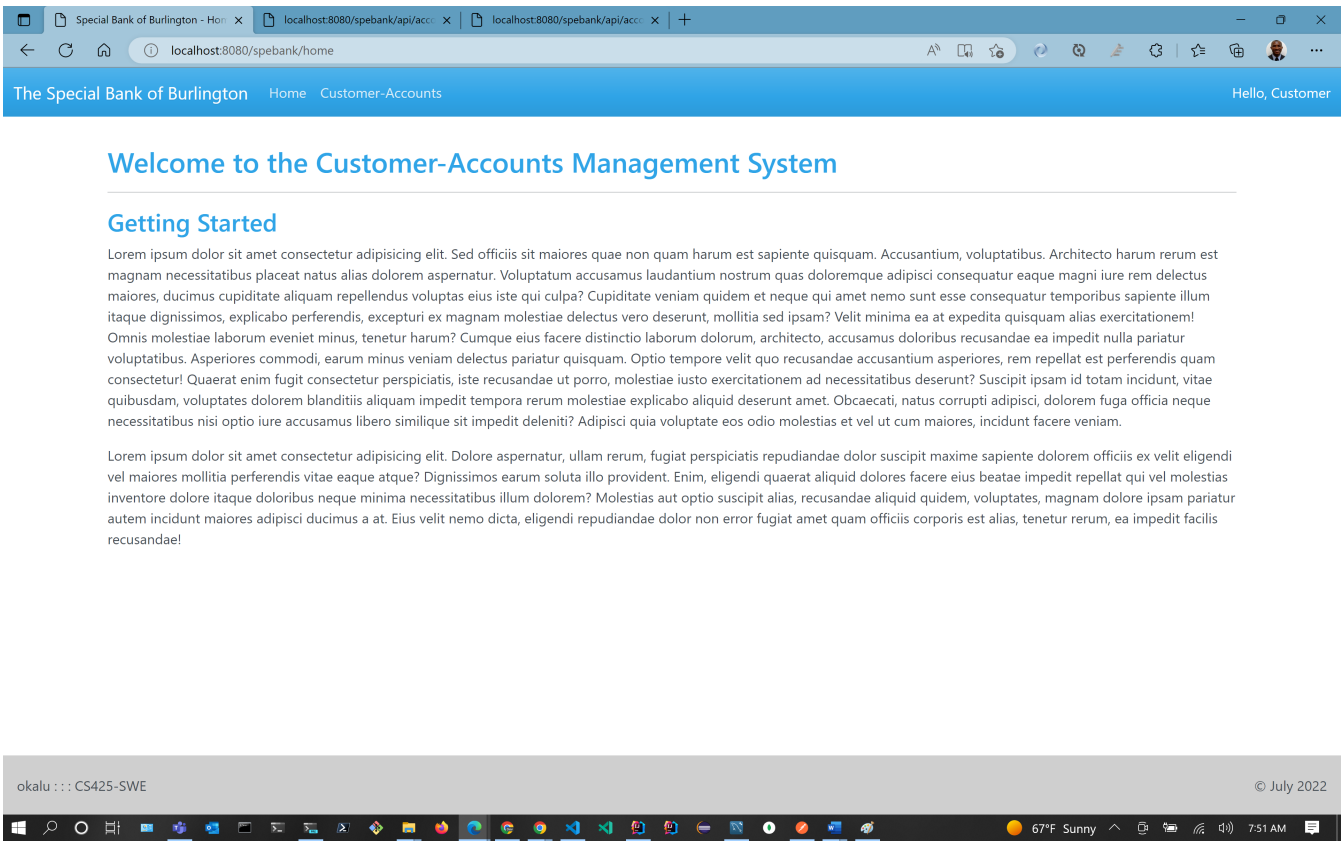
3. Implement a RESTful Web Service Endpoint (i.e. Web API) url which presents the list of all Customer-Accounts in JSON format.

4. Implement a RESTful Web Service Endpoint (i.e. Web API) url which presents the list of only Prime Customer-Accounts in JSON format. Note: Prime Customer-Accounts are accounts that have more than $10,000.00.

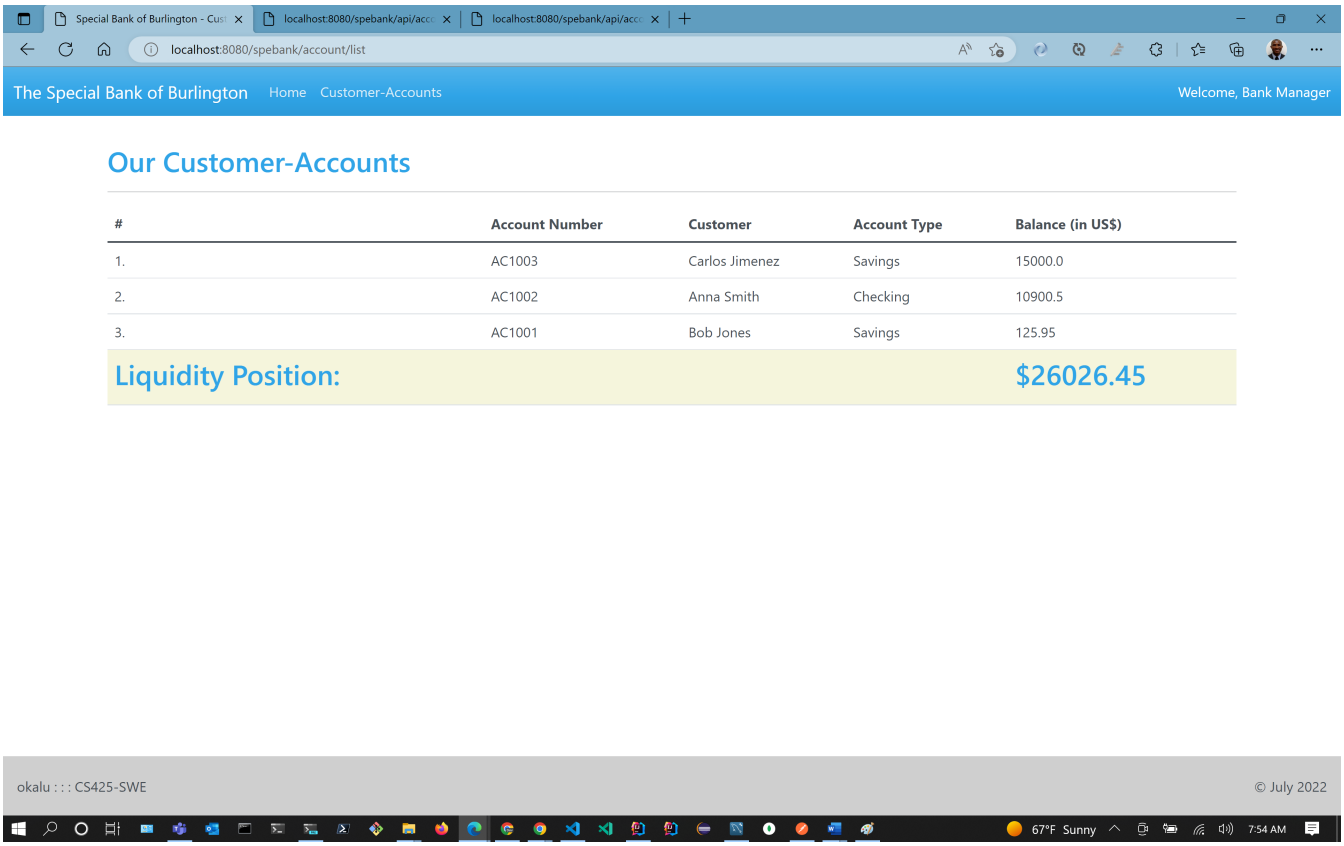
Shown below are sample User Interfaces and data presentation for the above features/use-cases.

Note: Your own UI design does NOT necessarily have to look exactly like these samples. But your UIs should contain/present all the necessary data and data fields, as required.

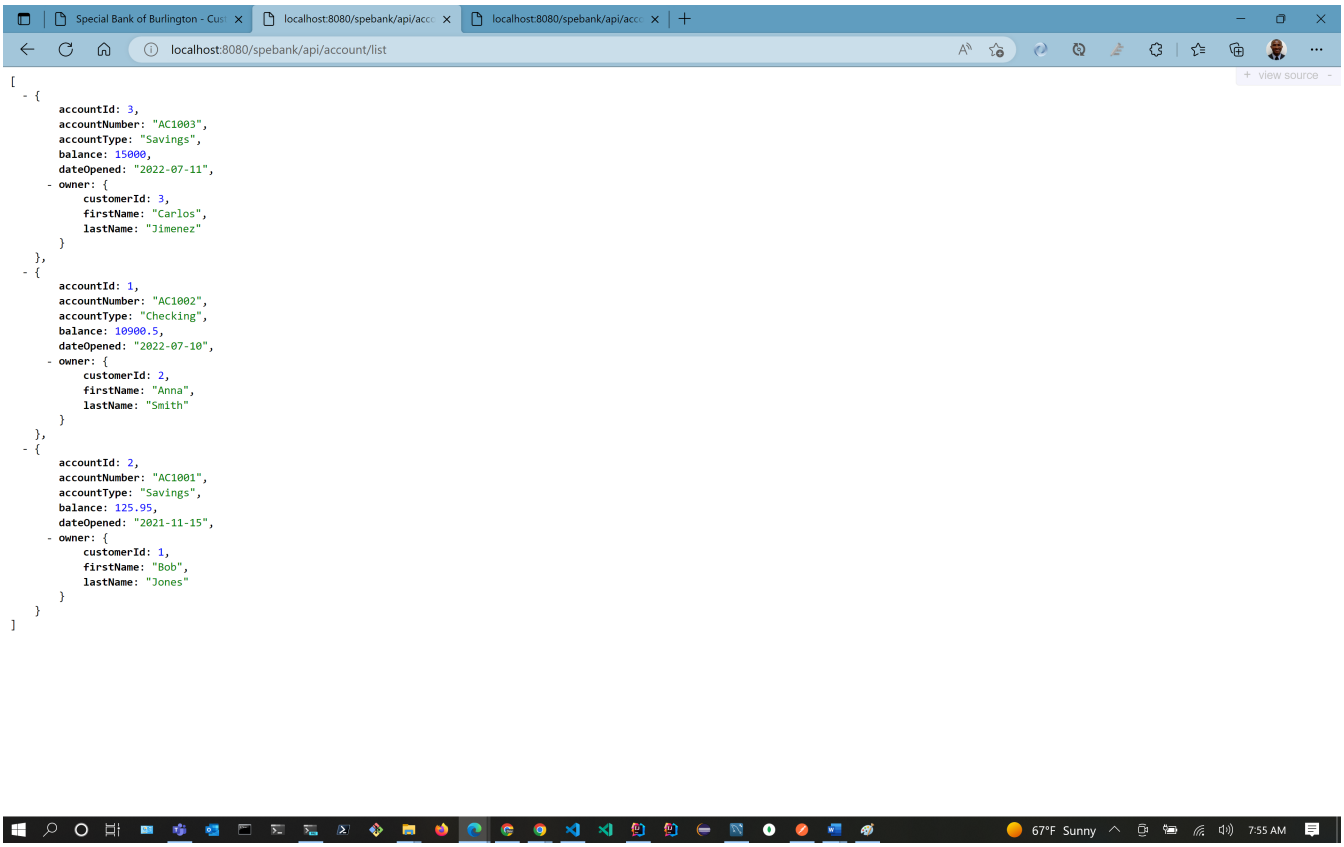
Homepage:



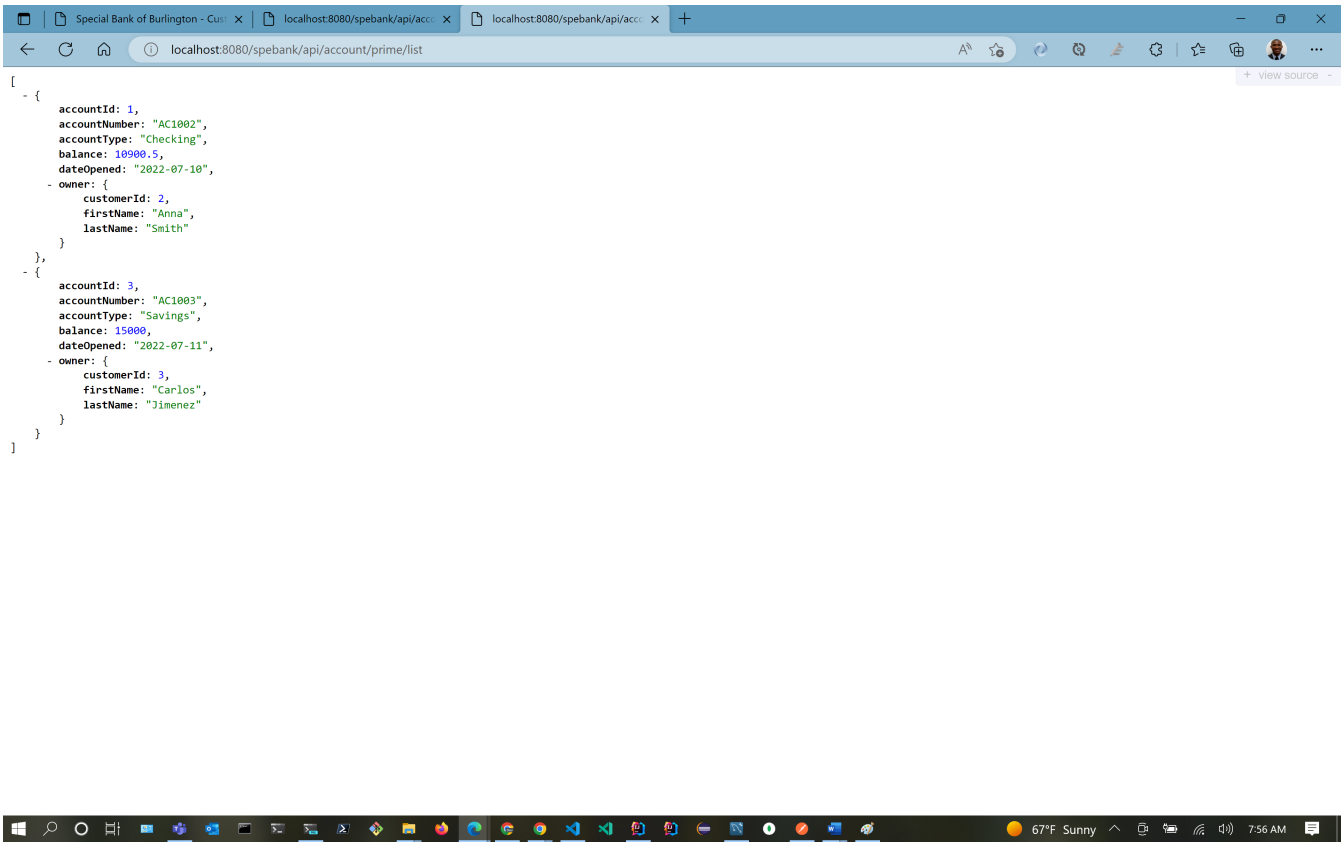
List of all Customer-Accounts (note: Sorted in descending order of their Balance):



RESTful (Web) API endpoint url for List of all Customer-Accounts:

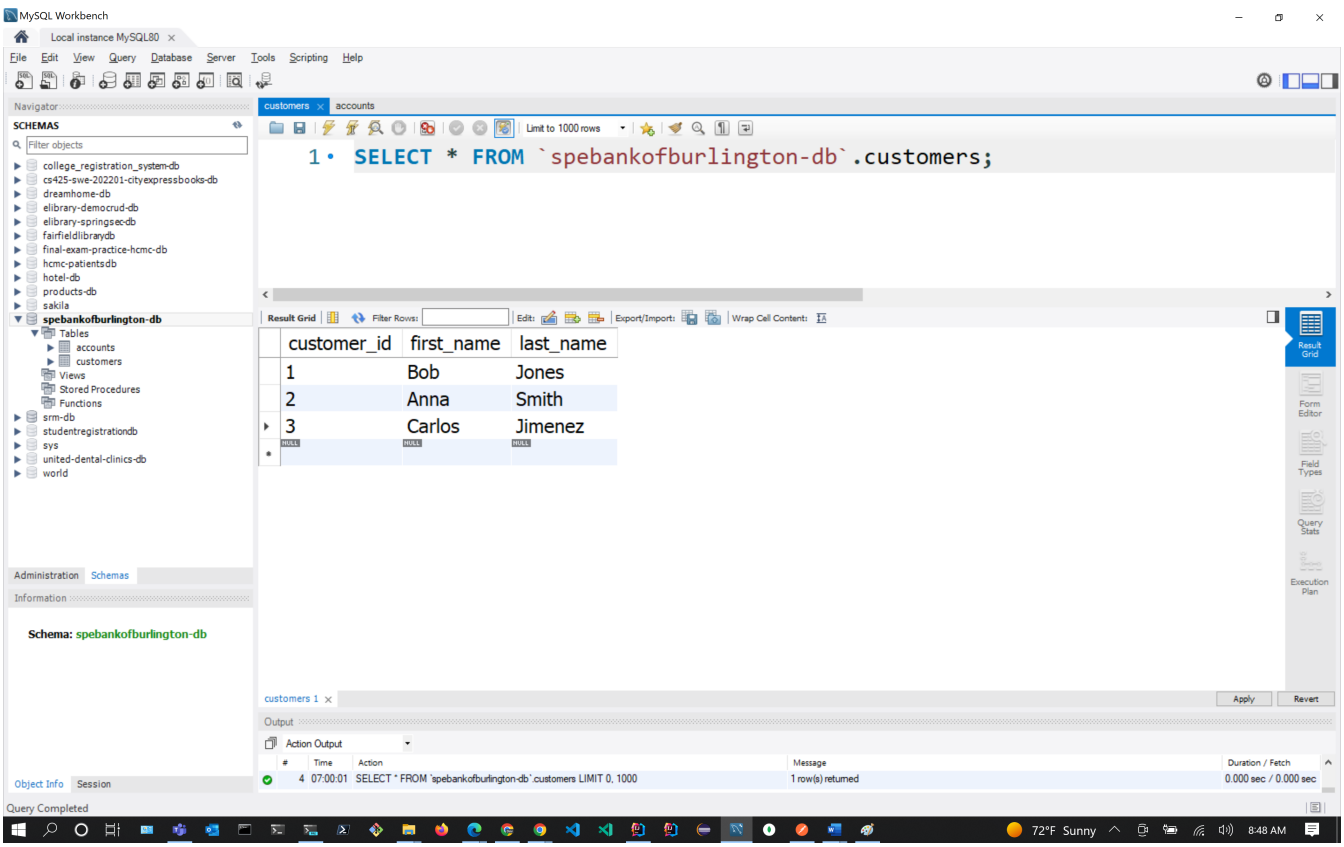


RESTful (Web) API endpoint url for List of Prime Customer-Accounts:

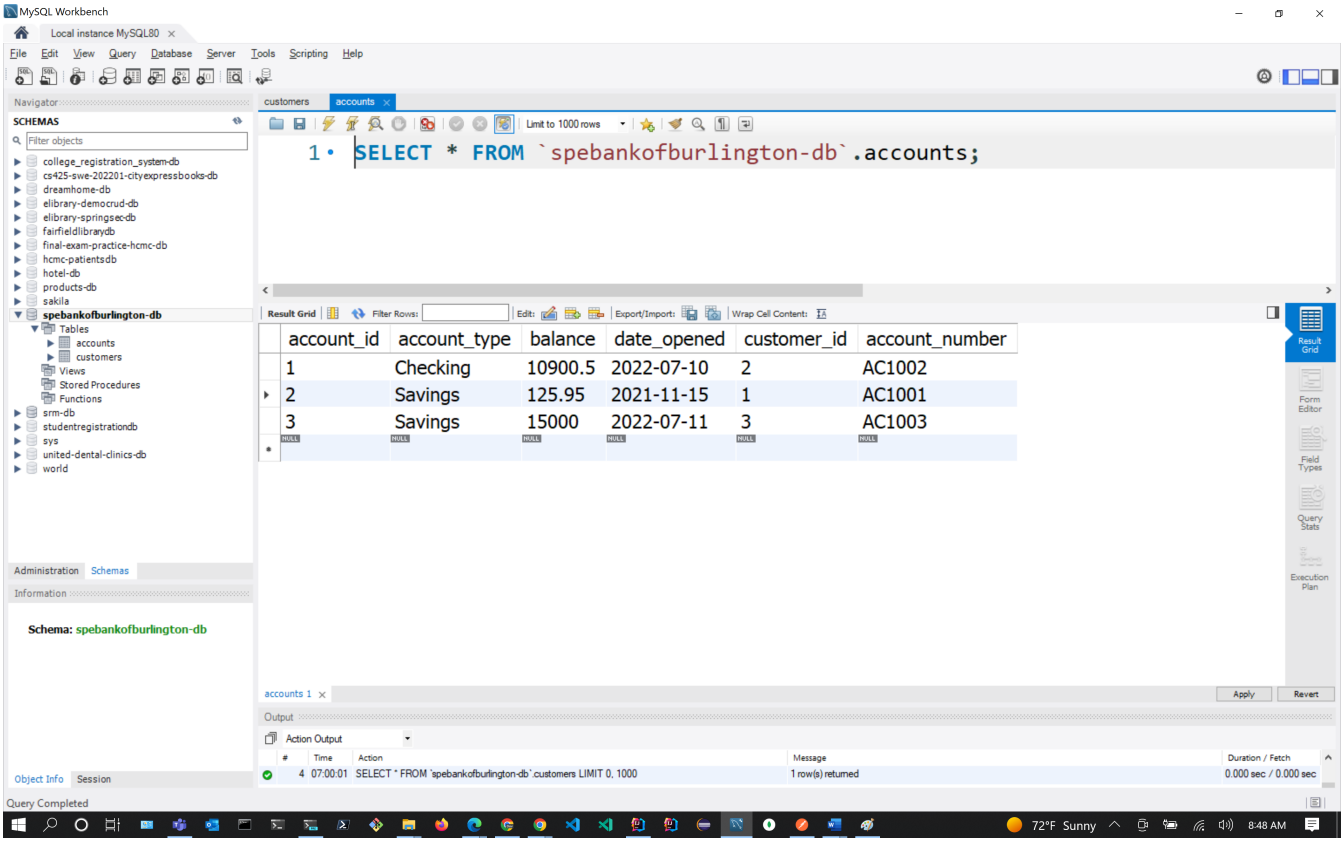


Database Tables screenshot (take a screenshot of your database tables, similar to the two pasted below):

Customers table



Accounts table



//-- The End --//

final-ijlal-machkori.zip (2,501.09 KB)