C++ Practice Question

For this problem your primary task is to complete the implementation of a class that represents a bank customer. An initial sketch of the BankCustomer object can be found in `BankCustomer.h`

A single customer consists of 2 datafields (`my_customer` and `customer_savings`). These datafields are private and are comprised of 2 distinct classes (`Person` and `SavingsAccount`). You must use these datafields. You will need to implement each class relative to these fields using correct c++ .h and .cpp format.

YOU MAY NOT ADD ANY ADDITIONAL VARIABLES TO THE BankCustomer CLASS!

Implementations of `Person` and `SavingsAccount` classes have been provided you and are contained in their corresponding .h and .cpp files. DO NOT modify these files (**Person.h, Person.cpp, SavingsAccount.h, SavingsAccount.cpp**). At present these classes are defined as follows:

Person: Contains PRIVATE `string` variables for **first_name** and **last_name** along with appropriate constructor, setter, and getter functions.

SavingsAccount: Contains PRIVATE `double` variables for **balance** and **interest_rate** along with appropriate constructor, setter, and getter functions.

TASK: Implement each function within the `BankCustomer` class. Definitions for each function along with a commented description for the expected implementation are provided in the `BankCustomer.h` file. All BankCustomer function implementations should be completed in the `BankCustomer.cpp` file.

You should not need to modify the `BankCustomer.h` file. However, if you feel that you wish to add any additional helper functions, you may add PRIVATE helper functions, which you will need to declare in the space indicated.

Compile your program using the command:

```
g++ main.cpp BankCustomer.cpp SavingsAccount.cpp Person.cpp -o Bank
```

Run problem using command

```
./Bank
```

For your convenience a simple test program has been provided in main.cpp. While the test program does not indicate whether functions are working properly, it provides the framework whereby you can visualize the effects of class member functions. You may modify main.cpp to accommodate any additional tests you feel may be useful to help you verify your program.