Model, design and develop a banking application

. The banking application allows the user, i.e., the bank employee to create a bank account for a specific client. It allows him to list all the available bank accounts. For each account, it allows him to display the account details, withdraw money and deposit money.

There are two types of bank accounts. The first type is the basic **BankAccount**. It holds the following data:

∙ account ID

∙ balance

The following methods apply to this class:

∙ Constructor There are 2 constructors. The first sets the balance to a given value. The second is a no-argument constructor and it sets the balance to 0.

∙ Setters and getters These methods allow accessing the private data fields.

∙ withdraw withdraws an amount of money from the account if the balance is sufficient.

∙ deposit deposits an amount of money in the account

The second type of accounts extends the basic Bank Account and may have some extra data fields and operations. It is called **SavingsBankAccount**.

This account requires the user to keep a minimum amount of money in the account, which is called the minimum balance, as long as the account is open. It also requires him to make deposits that are not less than 100 a time. So, it has the following additional data field:

∙ minimumBalance This minimum balance takes a default value of 1000 L.E.

It has the following methods plus those inherited from the parent class:

∙ Constructor The constructor sets the value of the initial balance and the minimum balance. Initial balance should be >= min balance.

∙ Setters and getters These methods allow accessing the private data fields.

∙ withdraw overrides the method withdraw to allow withdrawing money but not below the minimum balance.

∙ deposit deposits an amount of money in the account but only if the amount to deposit is 100 or more.

There is also a Client class which holds the basic information of a clientaddress a like his name, nd phone number. It holds a pointer to his bank account. An account also points to its owner.