**Banking System**

Summary Description

A banking system will be developed to allow clients who hold accounts (checking, savings, Etc.) in the bank to use it to do banking transactions such as withdrawal, money transfer, balance, inquiry, Etc. The software system is used also by the tellers at the banking centers to do regular banking transactions for the clients who go to the banking centers. The bank has several centers all over the country but their capabilities (e.g., availability of ATM machines, international money transfer services, currency exchange services, etc.) may differ. The system has capabilities to identify its clients as well as the bank employees. Customers can associate their monthly bills to their checking accounts and the bills are automatically deducted from the account. Clients can also associate their monthly payments to any of their accounts so that the payments are automatically deposited into the accounts.

**Use Cases**

The system you design should support the following use cases:

1. Login

* The first screen a user of the system sees is the login screen, which requests ID and password. When the Submit button is clicked, the ID is looked up in the data store. If this ID can be found, and if the password for this ID matches the password submitted, the authorization level is returned. Authorization levels are BUSINESS, ADMIN, and BOTH. If login is successful, UI features are made available according to the authorization level of the user.

2. Create saving account

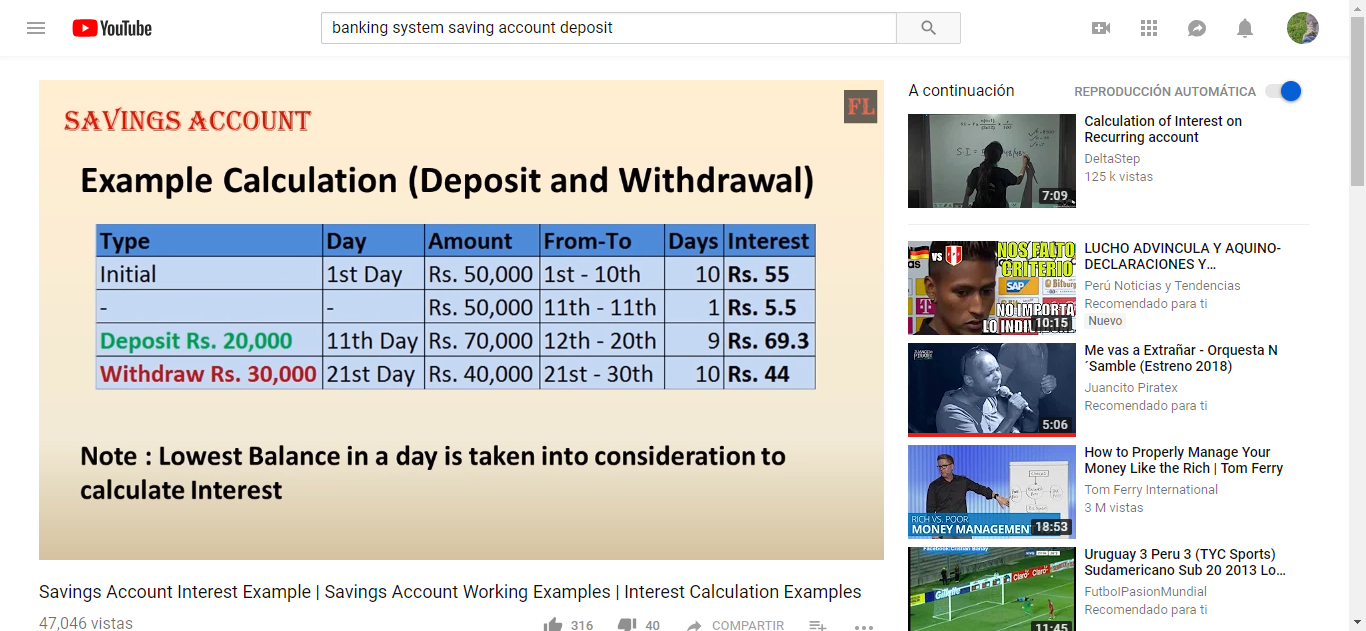
* The method computeUpdatedBalance in SavingsAccount performs the following computation to obtain the return value:

balance + (interestRate \* balance)

* Example calculation (Fixed amount)

Jhon saving account : $50,000

Daily interest=



3. Create checking account

* The method computeUpdatedBalance in CheckingAccount does the following computation to obtain the return value:

balance – monthlyFee.