SOFTWARE ENGINEERING CAPSTONE

COMMERCE BANK

Architecture Document

Team VIT

Team Members:

Bollimuntha Jahnavi Sri Kavya

Jayadeep Reddy Mannengi

Akhil Palagati

Yeshwanth Varma

Contents

1 Introduction:

E-banking (Internet Banking) is an e-commerce program that enables clients to conduct virtual banking and financial transactions online in a safe and secure manner. It entails the distribution of banking goods and services through the internet that which makes the easy and secure transactions. Hence we are aiming to develop Commerce bank transactions application. This is mainly implemented to follow up the transactions done through the banks and provide the notifications related to the transactions.

Initially customer will register with a particular bank with their particulars (name, email id, password and phone number) and with the initial amount to open the account.

Once after registration, customer can login into their account and can check the transactions history and also make the transactions. Customer will be notified with the email during their transactions. The notifications are based on the purpose of transaction, time and location (Local and non-local). If the customer forgets password, they can request for change of the password.

2 Design Goals:

The Primary goals in the design are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

3 System Behavior:

3.1 System Requirements Specification:

Functional and Non-Functional requirements:

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

Functional requirements:

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:
1) Authentication of customer whenever he/she logs into the system.
2) System shutdown in case of a cyber-attack.
3) A verification email is sent to customer whenever a transaction is happened.

Non-Functional requirements:

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:
1. Portability
2. Security
3. Maintainability
4. Reliability
5. Scalability
6. Performance
7. Reusability
8. Flexibility

Examples of non-functional requirements:
1) Emails should be sent with a latency of no greater than 12 hours from such an activity.

2) The processing of each request should be done within 10 seconds.
3) The site should load in 3 seconds whenever of simultaneous users are greater than 10000.

3.2 System Specifications:

Hardware Specifications:

Processor: I3/Intel Processor
RAM:8GB (min)
Hard Disk: 128 GB

Software Specifications:

Operating System : Windows 7/8/10
Technology : Python 3.6+
Server –side script : HTML, CSS, JS
IDE : PyCharm
Database : MySQL
Server : Xampp
Libraries Used : Numpy, IO, OS, Flask.

3.3 System Design:

Input Design:

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well-designed input forms and screens have following properties:

1. It should serve specific purpose effectively such as storing, recording, and retrieving the information.
2. It ensures proper completion with accuracy.
3. It should be easy to fill and straightforward.
4. It should focus on user's attention, consistency, and simplicity.

5. All these objectives are obtained using the knowledge of basic design principles regarding:
   (a) What are the inputs needed for the system?
   (b) How end users respond to different elements of forms and screens?

Objectives for Input Design:

The objectives of input design are:

1. To design data entry and input procedures.
2. To reduce input volume.
3. To design source documents for data capture or devise other data capture methods.
4. To design input data records, data entry screens, user interface screens, etc.
5. To use validation checks and develop effective input controls.

Output Design:

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design:

The objectives of output design are:

1. To develop output design that serves the intended purpose and eliminates the production of unwanted output.
2. To develop the output design that meets the end user's requirements.
3. To deliver the appropriate quantity of output.
4. To form the output in appropriate format and direct it to the right person.
5. To make the output available on time for making good decisions.

3.4 Modules:

1. System
2. Customer

System:

1. Here the system acts as the database that which will stores the customer information during their registration process and about the transaction details.

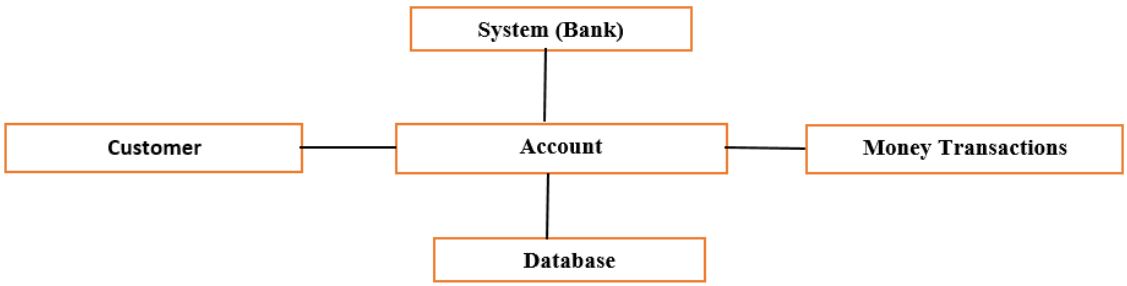
2. System gives access to customer during the login, if the entered details are same as registered details from database.
3. System will generate the transaction histories.
4. The transaction details will be sent to customer as an email notification.

Customer:

1. Here the customer will register with their details into the account and can login.
2. After login into the account, they can check the transaction details and can make their transactions.
3. Customer will get the email notifications about the transaction that which includes the purpose, time and location (local or non-local).
4. Customer can request for the change of the password if he/she forgets password.

4 Logical View:

4.1 High Level Design:



1. Here the customer will register into the account, where they can check their transaction history and make transactions.

2. Account holds a main part from where the transactions can be made.

3. System (Bank) is the one where a customer can register for the account.

4. Once after the registration with bank and login into the account and then money transactions can be held by customer.

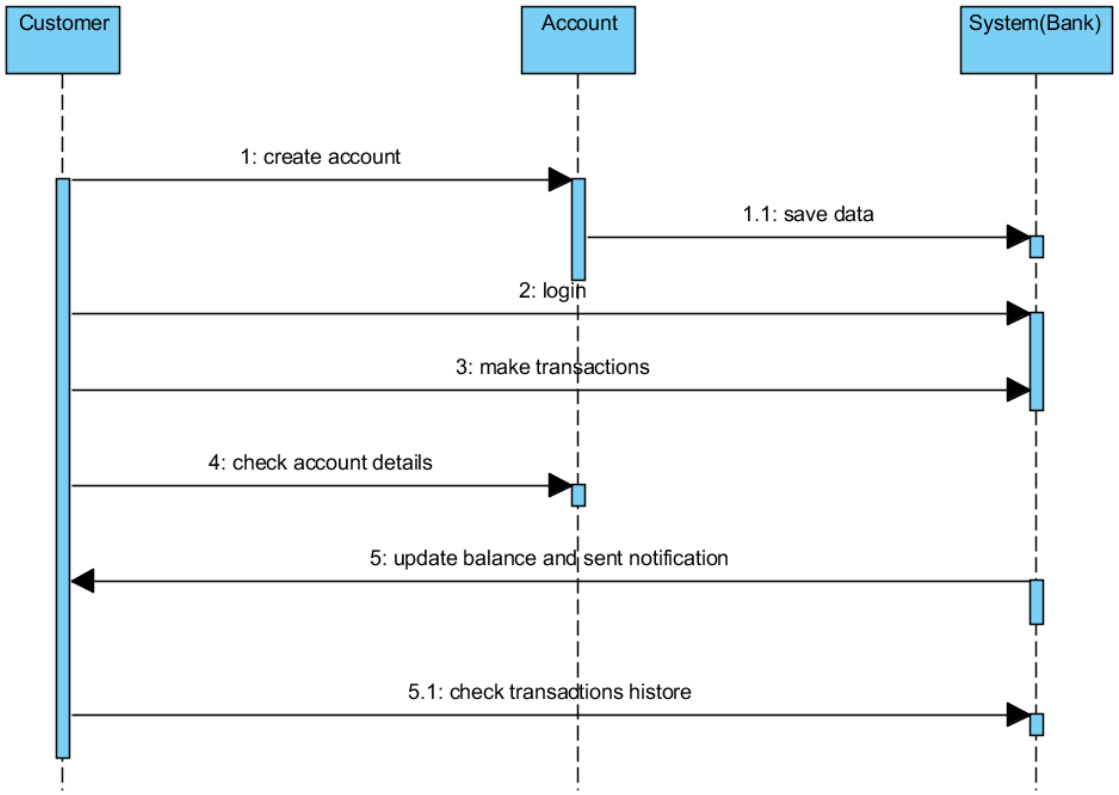
5. The registration details of customer and all the transaction details are stored in

the database.

4.2 Mid-Level Design:

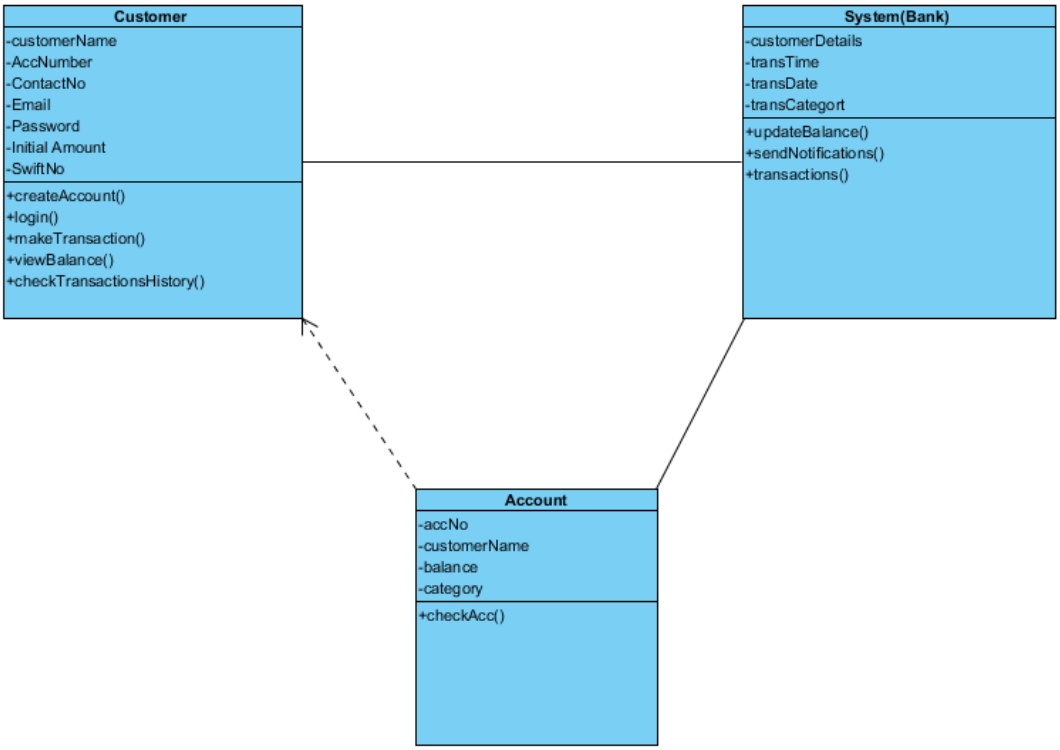
1. A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.

2. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



4.3 Detailed class design:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



5 Process View:

1. We are using PyCharm IDE to implement a project where at first we will create project and set python environment to the project.
2. At first step we will install all the necessary packages required for the project.
3. After package installations, HTML files will be placed in the templates directory and all static files are placed in static directory.
4. We are using the flask framework and render template function that which combine the python along with the script language.
5. Using this python and the scripting language we are creating a web page.
6. We can use the terminal to run the program.
7. Once after executing the program we can find a link from which we will be redirected to created web page in which customer can use the web page based upon his requirements.

6 Physical View:

The main physical components are considered as Account and Email. The customer needs to register into account in order to make any transactions or to get the information

about the transaction history.

Email is another main component where the details about any transaction will be notified to the email. The notification is mainly consists of purpose of the transaction, time and about the place, either it is from the own station or out of station.

7 Use case View:

1. A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.
2. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.
3. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.