My Personal Commerce Account-Banking Application

Architecture/Design Document

Team 7

Mariam Habib, Scott Howard, Maryam Albakhiet, Grant Jenkins, Irem Atik

**Table of Contents**

1 Introduction……………………………………………………………....5

2 Design Goals..............................................................................................6

3 System Behavior…………………………………………………………6

4 Logical View…………………………………………………………….6

5 Process View………………………………………………………….....7

6 Physical View........................................................................................... 8

7 Case View……………………………………………………………..... 9

# Change History

**Version:** <6>

**Modifier:** <Mariam Habib>

**Date:** 04/4/2020

**Description of Change:** <Made final changes to document, fixed any spelling or grammar errors>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Version:** <5>

**Modifier:** <Scott Howard>

**Date:** 04/04/2020

**Description of Change:** <Edited the logical view>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Version:** <4>

**Modifier:** < Maryam Albakhiet>

**Date:** 04/02/2020

**Description of Change:** <Edited the process view>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Version:** <3>

**Modifier:** <Grant Jenkins >

**Date:** 03/27/2020

**Description of Change:** <Edited the physical view>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Version:** <2>

**Modifier:** <Irem Atik>

**Date:** 03/25/2020

**Description of Change:** <Edited user case view>

­­­­­­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Version:** <1>

**Modifier:** <Mariam Habib>

**Date:** 03/23/2020

**Description of Change:** <Initial changes made to document, edited introduction, design goals, and system behavior>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 1 Introduction

**Architecture and Design**

The purpose of this document is to describe the architecture and design of the My Personal Commerce Account-Banking Application in a way that addresses the interests and concerns of all major stakeholders. For this application the major stakeholders are:

* Users and the customer – The customer for this product is Commerce Bank and the users will be Commerce Bank customers. Commerce Bank expects a working application that meets the requirements, goals, and constraints. Commerce Bank customers expect an easy to use, aesthetically pleasing application that does what it needs them to do. The architecture will provide for correct system functionality and non-functional quality requirements.
* Developers – they want an architecture that will minimize complexity and development effort. They will use Django framework, SQL server, HTML, CSS, Java, and C#.
* Project Manager – the project manager is responsible for planning, organizing, staffing, directing, controlling, and managing risks. She wants an architecture that divides the system into components of roughly equal size and complexity that can be developed simultaneously with minimal dependencies. Each team member has knowledge on particular areas, so she assigns each person to help code a section of the product using their skills.

Here the architecture of the My Personal Commerce Account-Banking Application is described from 4 different perspectives [1995 Krutchen]:

1. Logical View – major components, their attributes and operations. This view also includes relationships between components and their interactions. When doing OO design, class diagrams and sequence diagrams are often used to express the logical view.
2. Process View – the threads of control and processes used to execute the operations identified in the logical view.
3. Development View – how system modules map to development organization.
4. Use Case View – the use case view is used to both motivate and validate design activity. At the start of design the requirements define the functional objectives for the design. Use cases are also used to validate suggested designs. It should be possible to walk through a use case scenario and follow the interaction between high-level components. The components should have all the necessary behavior to conceptually execute a use case.

# 2 Design Goals

The design priorities for the My Personal Commerce Account-Banking Application are:

* The design should minimize complexity and development effort.
* The design should be simple, but high-level.
* The design should include information hiding and security.
* The design should be responsive and aesthetically pleasing
* The design should be user friendly and easy to understand
* The design should be professional and use Commerce Bank color scheme in styling
* The design should make it easier for customers to recall and recognize information
* The design should satisfy the requirements, priorities, goals, and constraints
* The design should be intuitive and straightforward

# 3 System Behavior

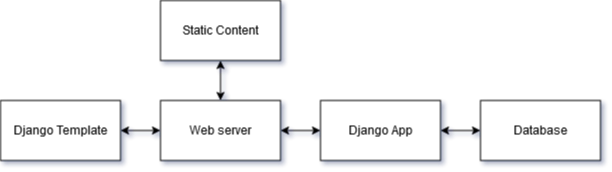
The My Personal Commerce Account-Banking Application should be able to allow users to log into their accounts using the Login page, view their transactions in detail in the Dashboard page, and set triggers for notification rules using the Triggers tool. It should also allow users to receive notifications around the triggers. The application should be able to remember users if they close their browser and log in again. The system should also save data to a database so recurring reports can be created.

# 4 Logical View

The logical view describes the main functional components of the system. This includes modules, the static relationships between modules, and their dynamic patterns of interaction.

In this section the modules of the system are first expressed in terms of high level components (architecture) and progressively refined into more detailed components and eventually classes with specific attributes and operations.

## *4.1 High-Level Design (Architecture)*



**System Architecture**

* **Django Template** facilitates client/server communication. It is part of the framework that exists in the client’s browser to send and receive http objects from the server.
* **Static Content** resides on the application server’s filesystem for security reasons. Its purpose is to hold the static CSS, JavaScript, and HTML pre-generated by the Django Framework.
* The **Database** holds data on for user accounts, bank accounts, transactions, triggers, notifications.
* The **Django App** is the business logic of the application. It retrieves information and displays it to the user and responds to various triggers and events
* The **Web Server** routes requests to the Django App and Static Content.

# 5 Process View

5.1 Process View Description

The Process View is essential in understanding how the separate components and subcomponents communicate with each other in a concurrent application. By better understanding the necessary paths of communication between the components, it may be possible to optimize the data flow and storage of the application, as well as ensuring thread-safety.

5.2 Application Thread

This is the main thread created upon starting the web server. This thread is not created by the user. This thread handles program flow from one page to another, retrieving and sending information to the database and retrieving user input from the web pages.

5.3 Presentation Thread

This thread is created when the user accesses the web site and is created by the user. It is responsible for displaying content sent from the web server and providing an interface for entering information to send to the server.

5.4 Database Thread

This thread is created when the database server begins and is not created or directly connected to the user. This thread is responsible for retrieving data requested from the web server and storing data sent from the web server in addition to compiling periodic reports as well as generating alerts in response to new transactions.

# 6 Physical View

6.1 Physical View Description

The Physical View, also known sometimes as the Deployment View, describes how the application software will map onto hardware and what devices will be responsible for running certain software. All software for this application will be ran on the host computer.

6.2 Web Server

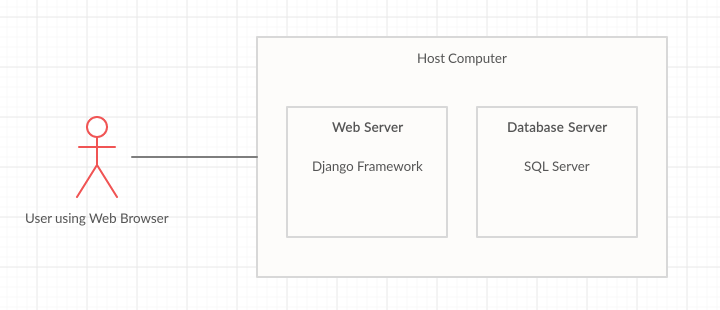
The application web server is used to serve HTTP requests and will be ran using the Django Python web framework on the host computer.

6.3 Database Server

The database server will be used to store end users’ information and will be run using an SQL server on the host computer.

6.4 Deployment Diagram

The below image is a diagram to visualize the Physical View



# 7 Use Case View

1. User Login
   1. Login Page
      1. Valid input
         1. User is taken to the dashboard
      2. Invalid input
         1. Click “forget password” to change user’s password
2. User Register
   1. Valid input
      1. Register user
      2. User is taken to the dashboard
   2. Errors
      1. Error message, enter correct input
3. User Dashboard
   1. Daily screen
   2. Navigation bar
      1. Logout
      2. Dashboard button
      3. Notification button
      4. Transaction button
4. Transaction Page
   1. Export all data to spreadsheet
   2. Click to add a new transaction
      1. Valid input
         1. Create a new transaction
      2. Invalid input
         1. Error message
5. Notification Page
   1. All notifications
   2. Notification rule button
      1. User can add/edit/delete notification rules