Independent Study (Consulted with Riley Knybel)

**Kubera E-Banking Project**

Software Design Document

Christie Kenyon

11/05/2023

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **1.** | **INTRODUCTION** |
|  |  |
| 1.1 | Purpose |
| 1.2 | Scope |
| 1.3 | Overview |
| 1.4 | Reference Material |
| 1.5 | Definitions and Acronyms |
|  |  |
| **2.** | **SYSTEM OVERVIEW** |
|  |  |
| **3.** | **SYSTEM ARCHITECTURE** |
|  |  |
| 3.1 | Architectural Design |
| 3.2 | Decomposition Description |
| 3.3 | Design Rationale |
|  |  |
| **4.** | **DATA DESIGN** |
|  |  |
| 4.1 | Data Description |
| 4.2 | Data Dictionary |
|  |  |
| **5.** | **COMPONENT DESIGN** |
|  |  |
| **6.** | **HUMAN INTERFACE DESIGN** |
|  |  |
| 6.1 | Overview of User Interface |
| 6.2 | Screen Images |
| 6.3 | Screen Objects and Actions |
|  |  |
| **7.** | **REQUIREMENTS MATRIX** |
|  |  |
| **8.** | **APPENDICES** |

**1. INTRODUCTION**

* 1. **Purpose**

The goal of this design document is to provide all shareholders with a detailed description of the Kubera E-Banking System that informs them of which tools and methods will be used to create the system.

* 1. **Scope**

This product is a web-based banking system that eliminates the need for clients to go to a physical location to perform basic banking tasks, such as making deposits, making withdrawals, transferring funds, or paying bills.

* 1. **Overview**

1.3.1 This document outlines the architecture, design, and requirements conformity of the product.

* 1. **Reference Material**

1. “IEEE Standards Association.” IEEE Standards Association, https://standards.ieee.org. Accessed 5 Nov. 2023.
2. Installation — Flask Documentation (3.0.x). https://flask.palletsprojects.com/en/3.0.x/installation/. Accessed 5 Nov. 2023.
3. Quickstart — Flask Documentation (3.0.x). https://flask.palletsprojects.com/en/3.0.x/quickstart/. Accessed 5 Nov. 2023.
4. Flask: Banking System | Introduction | Easy For Beginners. www.youtube.com, https://www.youtube.com/watch?v=c2u9cktRaZU. Accessed 5 Nov. 2023.
5. Flask: Banking System | Episode 1: Setup | Easy For Beginners. www.youtube.com, https://www.youtube.com/watch?v=zPv6mm67r7g. Accessed 5 Nov. 2023.
6. Flask: Banking System | Episode 2: Navbar | Easy For Beginners. www.youtube.com, https://www.youtube.com/watch?v=PKMwhdt0uqo. Accessed 5 Nov. 2023.
7. Flask: Banking System | Episode 3: Registration | Easy For Beginners. www.youtube.com, https://www.youtube.com/watch?v=SFSOCIcH\_vg. Accessed 5 Nov. 2023.
8. Flask: Banking System | Episode 4: Login | Easy For Beginners. www.youtube.com, https://www.youtube.com/watch?v=DqTQc1o3WEE. Accessed 5 Nov. 2023.
9. Flask: Banking System | Episode 5: Creating Account(Object Oriented Programming)| Easy For Beginners. www.youtube.com, https://www.youtube.com/watch?v=0WYxH3JoYUM. Accessed 5 Nov. 2023.
10. Flask: Banking System | Episode 6: Deposit | Easy For Beginners. www.youtube.com, https://www.youtube.com/watch?v=Q3mlSxXatw4. Accessed 5 Nov. 2023.
11. Flask: Banking System | Episode 7: Withdraw | Easy For Beginners. www.youtube.com, https://www.youtube.com/watch?v=2fmKGOC4rOI. Accessed 5 Nov. 2023.
12. contributors, Mark Otto, Jacob Thornton, and Bootstrap. Introduction. https://getbootstrap.com/docs/4.4/getting-started/introduction/. Accessed 5 Nov. 2023.
13. contributors, Mark Otto, Jacob Thornton, and Bootstrap. Navbar. https://getbootstrap.com/docs/5.3/components/navbar/. Accessed 5 Nov. 2023.
14. MVC Framework - Introduction. https://www.tutorialspoint.com/mvc\_framework/mvc\_framework\_introduction.htm. Accessed 5 Nov. 2023.
    1. **Definitions and Acronyms**

* KES. Kubera E-Banking System
* user. The person, or persons, who operate or interact directly with the product, but who may not have established an account on the KES
* account holder. The person, or persons, who operate or interact directly with the product and already has an account on the KES
* DBMS. Database Management System
* HTTPS. Hypertext Transfer Protocol Secure
* FTP. File Transfer Protocol
* SMTP. Simple Mail Transfer Protocol
* TCP/IP (port 443). Transmission Control Protocol/Internet Protocol
* SDD. Software Design Document

**2. SYSTEM OVERVIEW**

2.1 The Kubera E-Banking System (KES) will eliminate the need for clients to go to a physical location to perform common banking tasks such as deposits, withdrawals/transfers, and paying bills. They will not be restricted to banking hours to conduct their business, as, apart from brief maintenance periods in off hours, they will have unfettered access to their account through any device that connects to the internet.

**3. SYSTEM ARCHITECTURE**

* 1. **Architectural Design**
     1. The KES is a web-based system using a Flask framework, so it consists of the client’s network, the user’s network, and the cloud for storage of the supporting database.

<insert diagram here>

* 1. **Decomposition Description**
     1. Flask offers a Model-View-Controller (MVC) architecture, where the *model* component will handle all the logic pertaining to data, the *view* component will handle all the user interface (UI) aspects, and the *controller* component serves as the interface between the *model* and *view* components [14].
     2. The operating system for KES will be Windows.
     3. Python’s SQLite3 package will generate the initial database for storing user data.
     4. Bootstrap templates were used for standardization of webpage design shared across pages and for the navigation bar.
  2. **Design Rationale**

3.3.1 The architecture was designed to be better suited to a system being created by a single developer on a short timeline. Flask’s framework in conjunction with SQLite3 and Bootstrap allows for a simplified approach to generating multiple pages in a condensed timeframe with minimal source files.

**4. SYSTEM ARCHITECTURE**

* 1. **Data Description**

The system will enable customers to create an account, establish a checking and/or savings sub-account, login and be authenticated, view transaction histories, make deposits, transfers, and pay bills online. This necessitates multiple tables in the database:

* users table for storing username, password, and profile information
* checking account transactions table for tracking deposits and withdrawals from the checking account and maintaining that account’s balance
* savings account transactions table for tracking deposits and withdrawals from the savings account and maintaining that account’s balance.
* credit card table for maintaining customer bank card information
* registered payee corporations table for online bill pay functionality
  1. **Data Dictionary**

Users table:

|  |  |  |
| --- | --- | --- |
| **Entity** | **DataType** | **Format/Description** |
| id | INTEGER | Numeric (auto-incremented); Account Number; Primary Key |
| username | TEXT | Any character allowed (required) |
| password | TEXT | Any character allowed (required) |
| fname | TEXT | First Name (required) |
| mname | TEXT | Middle Name (optional) |
| lname | TEXT | Last Name (required) |
| dob | TEXT | Birthdate MM/DD/YYYY (required) |
| ssn | TEXT | SSN ###-##-#### (required) |
| phone | TEXT | Phone ###-###-#### (required only if no email provided) |
| email | TEXT | Client E-mail (required only if no phone provided) |

transactions\_checking and transactions\_savings tables:

|  |  |  |
| --- | --- | --- |
| **Entity** | **DataType** | **Format/Description** |
| id | INTEGER | id from the corresponding users table |
| username | TEXT | username from the corresponding users table |
| date | TEXT | Transaction Date MM/DD/YYYY |
| time | TEXT | Transaction Time HH:MM:SS |
| type | TEXT | DEPOSIT or WITHDRAW |
| amount | FLOAT | Dollar amount deposited or withdrawn |
| balance | FLOAT | Updated balance |
| PRIMARY KEY | COMPOSITE | Primary Key combining id, date, and time |
| FOREIGN KEY | INTEGER | id from corresponding users table |
| FOREIGN KEY | TEXT | username from corresponding users table |

credit\_card table:

|  |  |  |
| --- | --- | --- |
| **Entity** | **DataType** | **Format/Description** |
| id | INTEGER | id from the corresponding users table |
| username | TEXT | username from the corresponding users table |
| date | TEXT | transaction Date MM/DD/YYYY |
| time | TEXT | transaction Time HH:MM:SS |
| vendor | TEXT | purchase information |
| amount | FLOAT | purchase amount |
| balance | FLOAT | Updated balance |
| PRIMARY KEY | COMPOSITE | Primary Key combining id, date, and time |
| FOREIGN KEY | INTEGER | id from corresponding users table |
| FOREIGN KEY | TEXT | username from corresponding users table |

registered\_corporation table:

|  |  |  |
| --- | --- | --- |
| **Entity** | **DataType** | **Format/Description** |
| id | INTEGER | id from the corresponding users table |
| username | TEXT | username from the corresponding users table |
| vendor | TEXT | Payee |
| payment\_date | TEXT | date of scheduled payment |
| account | INTEGER | vendor account payment will be routed to |
| amount | FLOAT | payment amount |
| Total | FLOAT | total amount paid to date |
| PRIMARY KEY | COMPOSITE | Primary Key combining id, date, and time |
| FOREIGN KEY | INTEGER | id from corresponding users table |
| FOREIGN KEY | TEXT | username from corresponding users table |

**5. COMPONENT DESIGN**

5.1 Flask:

5.1.1 Model: The SQLite database will house the following tables and use connection and cursor objects to insert/update/remove data in the database and perform queries:

5.1.1.1 users: store user information

5.1.1.2 transactions\_checking: track transactions in checking account and maintain its balance

5.1.1.3 transactions\_savings: track transactions in savings account and maintain its balance

5.1.1.4 credit\_card: track credit card activity and maintain its balance

5.1.1.5 registered\_corporation table: maintain list of authorized payees to whom funds can be transferred from customer’s account, along with dates of scheduled transactions, payment amounts, and account balance

5.1.2 View: Flask’s routes and view functions will render the following HTML templates, to be built utilizing Bootstrap’s starter template and NavBar template as their foundation and stored in the project’s templates directory, passing data from the backend to be displayed on the frontend:

5.1.2.1 base: The HTML for those elements shared across all pages of the application

5.1.2.2 home.html

5.1.2.3 login.html

5.1.2.4 register.html

5.1.2.5 deposit.html

5.1.2.6 transfer.html

5.1.2.7 transaction.html

5.1.2.8 accountsummary.html

5.1.2.9 billpay.html

5.1.2.10 checking.html

5.1.2.11 savings.html

5.1.3 Controller: The Python functions defined in the source files will control the data flow between the Model and View. They will establish routing of HTTP requests to designated pages and render them; additionally, they will handle tasks related to user authentication, account creation, account viewing, and fund manipulation.

**6. HUMAN INTERFACE DESIGN**

* 1. **Overview of User Interface**
     1. The overall layout of the KES system is clean and simple, with easy navigation, simple-ot-use forms, and smooth transition between pages. There is a friendly welcome on the home page, along with an invite to Log In if they are current users or Create Account if they are new.
     2. The navigation bar ensures that customers can get to any screen from any other

screen, but the natural flow between pages is designed to minimize the need for customer use of the navigation bar.

* 1. **Screen Images**

6.2.1 Home Page

A screenshot of a computer

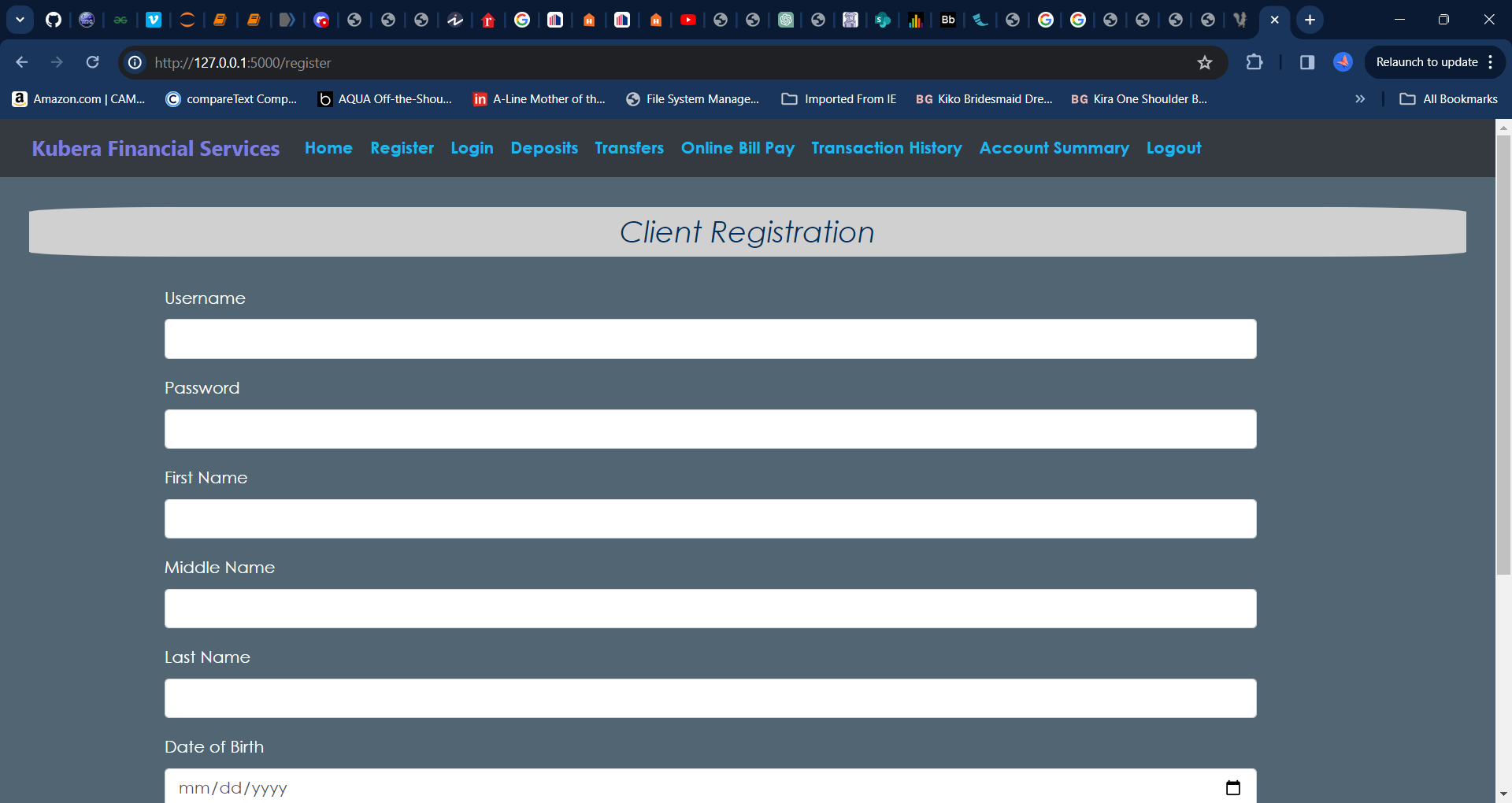
Description automatically generated

6.2.2 Login:

A screenshot of a computer

Description automatically generated

6.2.3 Register (Create Account):



A screenshot of a computer

Description automatically generated

6.2.4 Deposits:

A screenshot of a computer

Description automatically generated

6.2.5 Transfer Funds:

A screenshot of a computer

Description automatically generated

6.2.6 Transaction History:

A screenshot of a computer

Description automatically generated

6.2.7 Account Summary:

A screenshot of a computer

Description automatically generated

6.2.8 Online Bill Pay:

A screenshot of a computer

Description automatically generated

6.2.9 Savings Account Transaction History:

A screenshot of a computer

Description automatically generated

6.2.10 Checking Account Transaction History:

A screenshot of a computer

Description automatically generated

6.2.11 Logout:

A screenshot of a computer

Description automatically generated

**6.3 Screen Objects and Actions**

6.3.1 Home:

6.3.1.1 Welcome message for client

6.3.1.2 Sections for New Users to click a link to Registration page (Create Account) or Existing Users to click a link to Login page (Login)

6.3.1.3 Maintenance schedule displayed

6.3.2 Login:

6.3.2.1 Fields to enter username and password and submit with Login button for authentication

6.3.2.2 Links for users to reset username and/or password

6.3.2.3 Links for users who are new clients to go to registration page if they navigated to login in error by clicking Create Account

6.3.2.4 Successful login navigates user to Account Summary page

6.3.2.5 Failed Login returns user to Login page to try again with error message.

6.3.3 Register:

6.3.3.1 Fields to enter the following information:

6.3.3.1.1 First Name

6.3.3.1.2 Middle Name

6.3.3.1.3 Last Name

6.3.3.1.4 Date of Birth (mm/dd/yyyy)

6.3.3.1.5 Social Security Number (###-##-####)

6.3.3.1.6 Phone (###-###-####)

6.3.3.1.7 E-Mail

6.3.3.2 All Fields Required EXCEPT:

6.3.3.2.1 Middle Name – Optional

6.3.3.2.2 Phone – Conditionally Optional: Only Required if E-Mail field is blank

6.3.3.2.3 E-Mail – Conditionally Optional: Only Required if Phone field is blank

6.3.3.3 Register button to submit information to database

6.3.4 Deposit Funds:

6.3.4.1 Deposit Amount field for entering amount to be deposited

6.3.4.2 Account Type drop-down for selecting destination of either

Checking or Savings for deposit

6.3.4.3 Submit information using Submit button

6.3.5 Transfer Funds:

6.3.5.1 Transfer Amount field for entering amount to be transferred

6.3.5.2 Withdraw From field with drop-down for selecting account type, checking or savings

6.3.5.3 Transfer To field for with drop-down for selecting account type, checking or savings

6.3.5.4 Submit information using Transfer Funds button

6.3.6 Transaction History:

6.3.6.1 Instruction message to user to select either Checking or Savings to view

6.3.6.2 Bulleted link to Checking Account Transaction History Page

6.3.6.3 Bulleted link to Savings Account Transaction History Page

6.3.7 Account Summary:

6.3.7.1 Current Savings Account Balance displayed

6.3.7.2 Current Checking Account Balance displayed

6.3.7.3 Current credit card balance displayed

6.3.8 Online Bill Pay:

6.3.8.1 Payment Amount field for entering dollar amount to pay

6.3.8.2 Payee field for vendor information

6.3.8.3 Pay Now button submit payment

6.3.9 Savings Account Transaction History:

6.3.9.1 Fields

6.3.9.1.1 Date: Date and time in YYYY-MM-DD HH:MM:SS.SSSSSS format

6.3.9.1.2 Type: Indicates DEPOSIT or WITHDRAWAL

6.3.9.1.3 Amount: Dollar amount deposited/withdrawn

6.3.9.1.4 Balance: Updated balance at the time of each transaction

6.3.9.2 All transactions on record displayed in descending order by date/time

6.3.10 Checking Account Transaction History:

6.3.12.1 Fields

6.3.10.1.1 Date: Date and time in YYYY-MM-DD HH:MM:SS.SSSSSS format

6.3.10.1.2 Type: Indicates DEPOSIT or WITHDRAWAL

6.3.10.1.3 Payee: Indicates who received payment from the WITHDRAWAL; is None for DEPOSITS and transfers

6.3.10.1.4 Amount: Dollar amount deposited/withdrawn

6.3.10.1.5 Balance: Updated balance at the time of each transaction

6.3.9.2 All transactions on record displayed in descending order by date/time

**7. REQUIREMENTS MATRIX**

|  |  |  |
| --- | --- | --- |
|  | **Item:** | **Satisfies Requirements:** |
| 7.1 | Flask framework in conjunction with SQLite | 3.4 |
| 7.2 | Flask framework in conjunction with Bootstrap templates and additional HTML code to render pages and supporting Python code | 3.1.1, 3.1.1.1, 3.1.1.1.1 – 3.1.1.1.5, 3.2.1, 3.2.2, 3.2.2.1 – 3.2.2.1.7, 3.2.2.2 – 3.2.2.3, 3.2.4 – 3.2.12 |
| 7.3 | User authentication accomplished through Python and HTML interaction with SQLite database | 3.1.1.2, 3.1.1.2.1 – 3.1.1.2.3, 3.2.3, 3.2.12 |

**8. APPENDICES**

8.1 See Software Requirements Specification, Sections 3.1 through 3.4 for further requirements details.