**Project Title:** BANKING MANAGEMENT SYSTEM

**Date:** 09-06-2023

Project Mentor:

**Key Project Team Members:** Shubam Bopche, Manasa kumari Varkala, G. Rajaskekar Reddy

**Objective**: A Bank Management System project in Python is a software application that allows users to perform banking transactions such as creating and managing accounts, depositing and withdrawing funds, checking balances, and viewing transaction history.

**Technologies used:** Core Python, SQL

Banking Project Management System: The main aim of this project is we are using both front end and back-end coding using Python programming language and also a data base MySQL. Firstly, we used Jupiter notebook and we connected it to database MySQL using commands. Then we created Admin account and banking management system as header and added menu options available to user such as creating account, logging in or exiting the system. We have given 2 commant line prompts in which the user to enter their choice. If the user has entered 1, it indicated they want to create an account. We also provided in command prompt so that user to provide various details required for creating an account, such as the account type, name, address, KYC information, mobile number, and email address. If we select choice 2 the it redirects to the web page user details and password. We used modules: random and mysql.connector. These modules are used for generating random numbers and connecting to a MySQL database, respectively.

This code establishes a connection to the MySQL database using the provided host, username, password, and database name. A cursor object **c** is created to execute SQL queries on the database.

We defined a class called **Account**. This class represents a bank account and has attributes such as account number, account type, name, address, KYC information, mobile number, email address, and balance. It also has methods for depositing, withdrawing, checking balance, transferring funds, saving the account to the database, and updating the account balance in the database.

We defined another class called **Bank**. This class represents a bank and has a list of accounts as one of its attributes. It provides methods for creating a new account, retrieving an account by account number, and getting the total number of accounts.

We defined two more classes: User and Admin. Both classes inherit from the User class. The User class represents a regular user of the bank system and has attributes for username and password. The Admin class represents an administrator user and has an additional method for retrieving account statistics.

And we also defined two helper functions: create\_tables and create\_admin\_user. The create\_tables function creates a table named "accounts" in the database if it doesn't already exist. It also checks if an admin user exists and calls the create\_admin\_user function if no admin user is found. The create\_admin\_user function prompts the user to enter a username and password for the admin account and inserts the values into the "users" table in the database.

We also generated code a login function that prompts the user to enter a username and password. It then authenticates the user by checking if the entered credentials match either the admin user or a regular user.

The code defines several other functions for performing various operations suchas opening an account, depositing money, withdrawing money, checking balance, transferring funds, etc.

The main function is the entry point of the program. It first calls the create\_tables function to ensure the necessary database tables exist. It then presents a menu to the user with options to create an account, login, or exit the program. Based on the user's choice, it calls the corresponding function.

Finally, the Bank object, bank, is created along with an admin user and a regular user. The main function is called to start the bank management system.

The database connection is closed at the end of the code.