**Design Document: Python Script for MySQL Database Connection and Data Processing**

**1. Introduction**

This document details the design and functionality of a Python script used for connecting to a MySQL database, processing data from Excel files, and inserting cleaned data into a MySQL table. The script uses mysql.connector for database interactions, pandas for data manipulation, and streamlit for user interface error handling.

**2. Code Explanation**

**2.1 Import Statements**

import mysql.connector

from mysql.connector import Error

import streamlit as st

import pandas as pd

import os

* **import mysql.connector**: Imports the MySQL connector library to enable Python to interact with MySQL databases.
* **from mysql.connector import Error**: Imports the Error class to handle exceptions related to MySQL operations.
* **import streamlit as st**: Imports the Streamlit library for creating web applications and displaying error messages.
* **import pandas as pd**: Imports the pandas library for data manipulation and analysis.
* **import os**: Imports the OS module to interact with the operating system (e.g., file and directory operations).

**2.2 Database Connection Function**

def create\_connection():

try:

mydb = mysql.connector.connect(

host="localhost",

user="root",

password="", # use your own password

database="redbus",

)

if mydb.is\_connected():

return mydb

except Error as e:

st.error(f"Error connecting to MySQL: {e}")

return None

* **Function Definition**: create\_connection() establishes a connection to the MySQL database.
* **Parameters**: None.
* **Functionality**:
  + Attempts to connect to the MySQL database with specified credentials.
  + Returns the connection object if successful.
  + If connection fails, displays an error message using Streamlit and returns None.

**2.3 Data Cleaning and Insertion Function**

def datacleandbinsert(statename):

directory = f'D:/Project/' # Update the path here

if not os.path.exists(directory):

raise FileNotFoundError(f"The directory '{directory}' does not exist.")

excel\_files = [f for f in os.listdir(directory) if statename in f and f.endswith('.xlsx')]

if not excel\_files:

raise FileNotFoundError(f"No Excel files found for {statename} in '{directory}'.")

excel\_files\_full\_path = [os.path.join(directory, f) for f in excel\_files]

latest\_file = max(excel\_files\_full\_path, key=os.path.getmtime)

df = pd.read\_excel(latest\_file)

print(f"Imported file: {latest\_file}")

df['star\_rating'] = df['star\_rating'].astype(str).str[:3]

df['price'] = df['price'].astype(str).apply(lambda x: x.split()[-1])

prefixes = ('KAAC', 'RSRTC', 'Bihar', 'Kadamba', 'HRTC', 'JKRTC', 'PEPSU', 'SBSTC', 'WBTC', 'NBSTC', 'WBSTC', 'West bengal', 'NBSRTC', 'RSRTC')

df['operator'] = df['busname'].apply(lambda x: 'Government' if isinstance(x, str) and x.startswith(prefixes) else 'Private')

required\_columns = ['busname', 'bustype', 'departing\_time', 'duration', 'reaching\_time', 'star\_rating', 'price', 'seat\_availability', 'route\_name']

df = df.dropna(subset=required\_columns)

print(df.head())

try:

mydb = create\_connection()

cursor = mydb.cursor()

state\_to\_clean = df['state'].iloc[0]

delete\_query = """

DELETE FROM bus\_routes WHERE state = %s

"""

cursor.execute(delete\_query, (state\_to\_clean,))

print(f"Existing records for state '{state\_to\_clean}' removed from bus\_routes table.")

insert\_query = """

INSERT INTO bus\_routes (

busname, bustype, departing\_time, duration, reaching\_time, star\_rating,

price, seats\_available, route\_name, route\_link, state, operator

) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)

"""

for index, row in df.iterrows():

cursor.execute(insert\_query, (

row['busname'], row['bustype'], row['departing\_time'], row['duration'],

row['reaching\_time'], row['star\_rating'], row['price'],

row['seat\_availability'], row['route\_name'], row['route\_link'],

row['state'], row['operator']

))

mydb.commit()

except mysql.connector.Error as err:

print(f"Error: {err}")

finally:

cursor.close()

mydb.close()

* **Function Definition**: datacleandbinsert(statename) processes and inserts data into the MySQL database.
* **Parameters**: statename – a string representing the state name to filter files.
* **Functionality**:
  + Checks if the directory exists; raises an error if not.
  + Lists and identifies the latest Excel file containing the statename.
  + Reads the latest file into a pandas DataFrame.
  + Cleans and processes the DataFrame:
    - Trims star\_rating to the first three characters.
    - Extracts the final price value from the price column.
    - Determines whether the busname starts with specific prefixes to classify as 'Government' or 'Private'.
    - Drops rows with missing values in required columns.
  + Connects to the MySQL database and executes:
    - A delete query to remove existing records for the state.
    - An insert query to add new records from the cleaned DataFrame.
  + Commits the transaction and handles exceptions.
  + Closes the database connection.