Here's a Python project to meet your requirements. We'll use jaydebeapi for querying the Oracle database and Streamlit for creating the interactive dashboard. Below is a complete solution structured as a Python project:

**Project Structure**

bash

Copy code

oracle\_dashboard/

├── app.py # Main script to run the dashboard

├── config.ini # Configuration file for database credentials

├── query\_utils.py # Utility functions for querying the database

└── requirements.txt # Python dependencies

**1. config.ini**

This file stores database credentials securely.

ini

Copy code

[oracle\_db]

host = YOUR\_ORACLE\_DB\_HOST

port = YOUR\_ORACLE\_DB\_PORT

sid = YOUR\_ORACLE\_DB\_SID

username = YOUR\_ORACLE\_DB\_USERNAME

password = YOUR\_ORACLE\_DB\_PASSWORD

**2. query\_utils.py**

Utility functions to handle database connections and queries using jaydebeapi.

python

Copy code

import jaydebeapi

import configparser

def get\_db\_connection():

"""Establish a connection to the Oracle database using credentials from config."""

config = configparser.ConfigParser()

config.read("config.ini")

db\_config = config["oracle\_db"]

jdbc\_url = f"jdbc:oracle:thin:@{db\_config['host']}:{db\_config['port']}:{db\_config['sid']}"

driver = "oracle.jdbc.driver.OracleDriver"

jar\_file = "/path/to/ojdbc8.jar" # Update with the path to the Oracle JDBC driver.

conn = jaydebeapi.connect(driver, jdbc\_url, [db\_config["username"], db\_config["password"]], jar\_file)

return conn

def fetch\_data(query, params=None):

"""Execute a query and fetch data."""

conn = get\_db\_connection()

cursor = conn.cursor()

try:

cursor.execute(query, params or [])

rows = cursor.fetchall()

columns = [desc[0] for desc in cursor.description]

return rows, columns

finally:

cursor.close()

conn.close()

**3. app.py**

Main Streamlit application for the dashboard.

python

Copy code

import streamlit as st

from query\_utils import fetch\_data

st.title("Oracle Table Dashboard")

# Define query options

QUERY\_OPTIONS = {

"By Business Date": "SELECT \* FROM your\_table WHERE business\_date = ?",

"By Business Date and ID": "SELECT \* FROM your\_table WHERE business\_date = ? AND id = ?",

"By Business Date Range": "SELECT \* FROM your\_table WHERE business\_date BETWEEN ? AND ?"

}

# Sidebar menu

query\_type = st.sidebar.radio("Select Query Type", list(QUERY\_OPTIONS.keys()))

# Form inputs

if query\_type == "By Business Date":

business\_date = st.sidebar.date\_input("Business Date")

if st.sidebar.button("Fetch Data"):

query = QUERY\_OPTIONS[query\_type]

data, columns = fetch\_data(query, [str(business\_date)])

st.dataframe(data, columns=columns)

elif query\_type == "By Business Date and ID":

business\_date = st.sidebar.date\_input("Business Date")

record\_id = st.sidebar.text\_input("ID")

if st.sidebar.button("Fetch Data"):

query = QUERY\_OPTIONS[query\_type]

data, columns = fetch\_data(query, [str(business\_date), record\_id])

st.dataframe(data, columns=columns)

elif query\_type == "By Business Date Range":

from\_date = st.sidebar.date\_input("From Date")

to\_date = st.sidebar.date\_input("To Date")

if st.sidebar.button("Fetch Data"):

query = QUERY\_OPTIONS[query\_type]

data, columns = fetch\_data(query, [str(from\_date), str(to\_date)])

st.dataframe(data, columns=columns)

# Dropdown filter for specific IDs

if data:

ids = [row[0] for row in data]

selected\_id = st.selectbox("Filter by ID", options=ids)

filtered\_data = [row for row in data if row[0] == selected\_id]

st.dataframe(filtered\_data, columns=columns)

# Add pagination and error handling

st.sidebar.text("Pagination feature coming soon!")

**4. requirements.txt**

Dependencies for the project.

Copy code

streamlit

jaydebeapi

**Key Features**

1. **Dynamic Query Inputs**: Accepts business\_date, id, and date ranges as user inputs.
2. **Secure Credentials**: Credentials are passed via config.ini.
3. **Interactive Dashboard**: Built using Streamlit with dropdown filters and input fields.
4. **Error Handling**: Ensures safe database connections and query execution.
5. **Pagination (Placeholder)**: Streamlit's built-in capabilities can be extended for paginated views.

**Instructions to Run the Project**

1. Place the Oracle JDBC driver (ojdbc8.jar) in the appropriate path.
2. Install dependencies:

bash

Copy code

pip install -r requirements.txt

1. Run the Streamlit app:

bash

Copy code

streamlit run app.py

This setup provides an intuitive dashboard to query and visualize the Oracle table's execution statuses effectively. Let me know if you need additional features!

4o

Top of Form

Bottom of Form