[AssetClass\_IR]

r\_type\_1 = IR1

r\_type\_2 = IR2

[AssetClass\_Equity]

r\_type\_1 = EQ1

r\_type\_2 = EQ2

[AssetClass\_FX]

r\_type\_1 = FX1

r\_type\_2 = FX2

[General]

username = your\_username

password = your\_password

[URLs]

Auth\_url\_prod = https://auth-prod-url.com

Query\_url\_prod = https://query-prod-url.com

Auth\_url\_ppe = https://auth-ppe-url.com

Query\_url\_ppe = <https://query-ppe-url.com>

Fstring

import sys

import configparser

from PyQt5.QtWidgets import QApplication, QMainWindow, QWidget, QVBoxLayout, QLabel, QComboBox, QPushButton

# Function to run the backend script (replace with your actual code)

def run\_backend(assetclass, r\_type, username, password, Auth\_url\_prod, Query\_url\_prod, Auth\_url\_ppe, Query\_url\_ppe):

# Call your backend script here with the provided inputs

# This is a placeholder, replace with your actual logic

print(f"Asset Class: {assetclass}")

print(f"R Type: {r\_type}")

print(f"Username: {username}")

print(f"Password: {password}")

print(f"Auth URL (Prod): {Auth\_url\_prod}")

print(f"Query URL (Prod): {Query\_url\_prod}")

print(f"Auth URL (PPE): {Auth\_url\_ppe}")

print(f"Query URL (PPE): {Query\_url\_ppe}")

class AppWindow(QMainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setWindowTitle("Backend Integration")

self.setGeometry(100, 100, 600, 400)

self.central\_widget = QWidget()

self.setCentralWidget(self.central\_widget)

self.layout = QVBoxLayout()

# Create a ConfigParser object and read the configuration file

config = configparser.ConfigParser()

config.read('config.ini')

# Get the list of asset classes from the configuration

asset\_classes = [section for section in config.sections() if section.startswith("AssetClass\_")]

# Create a dropdown (QComboBox) for asset classes

self.assetclass\_label = QLabel("Asset Class:")

self.assetclass\_combo = QComboBox()

self.assetclass\_combo.addItems(asset\_classes)

self.layout.addWidget(self.assetclass\_label)

self.layout.addWidget(self.assetclass\_combo)

# Create a dropdown (QComboBox) for r-types

self.r\_type\_label = QLabel("R Type:")

self.r\_type\_combo = QComboBox()

self.layout.addWidget(self.r\_type\_label)

self.layout.addWidget(self.r\_type\_combo)

# Load initial r-type options based on the selected asset class

selected\_asset\_class = self.assetclass\_combo.currentText()

self.load\_r\_types(selected\_asset\_class)

# Connect the asset class selection to the r-type options

self.assetclass\_combo.currentTextChanged.connect(self.load\_r\_types)

# Create input fields for common configuration options

self.username\_label = QLabel("Username:")

self.username\_input = QLabel(config.get("General", "username"))

self.layout.addWidget(self.username\_label)

self.layout.addWidget(self.username\_input)

self.password\_label = QLabel("Password:")

self.password\_input = QLabel(config.get("General", "password"))

self.layout.addWidget(self.password\_label)

self.layout.addWidget(self.password\_input)

self.auth\_url\_prod\_label = QLabel("Auth URL (Prod):")

self.auth\_url\_prod\_input = QLabel(config.get("URLs", "Auth\_url\_prod"))

self.layout.addWidget(self.auth\_url\_prod\_label)

self.layout.addWidget(self.auth\_url\_prod\_input)

self.query\_url\_prod\_label = QLabel("Query URL (Prod):")

self.query\_url\_prod\_input = QLabel(config.get("URLs", "Query\_url\_prod"))

self.layout.addWidget(self.query\_url\_prod\_label)

self.layout.addWidget(self.query\_url\_prod\_input)

self.auth\_url\_ppe\_label = QLabel("Auth URL (PPE):")

self.auth\_url\_ppe\_input = QLabel(config.get("URLs", "Auth\_url\_ppe"))

self.layout.addWidget(self.auth\_url\_ppe\_label)

self.layout.addWidget(self.auth\_url\_ppe\_input)

self.query\_url\_ppe\_label = QLabel("Query URL (PPE):")

self.query\_url\_ppe\_input = QLabel(config.get("URLs", "Query\_url\_ppe"))

self.layout.addWidget(self.query\_url\_ppe\_label)

self.layout.addWidget(self.query\_url\_ppe\_input)

# Create a "Run Backend" button

self.run\_button = QPushButton("Run Backend")

self.run\_button.clicked.connect(self.on\_run\_backend)

self.layout.addWidget(self.run\_button)

self.central\_widget.setLayout(self.layout)

def load\_r\_types(self, asset\_class):

# Clear existing r-type options

self.r\_type\_combo.clear()

# Create a ConfigParser object and read the configuration file

config = configparser.ConfigParser()

config.read('config.ini')

# Get the list of r-types for the selected asset class

r\_types = [config.get(asset\_class, key) for key in config[asset\_class]]

self.r\_type\_combo.addItems(r\_types)

def on\_run\_backend(self):

assetclass = self.assetclass\_combo.currentText()

r\_type = self.r\_type\_combo.currentText()

username = self.username\_input.text()

password = self.password\_input.text()

Auth\_url\_prod = self.auth\_url\_prod\_input.text()

Query\_url\_prod = self.query\_url\_prod\_input.text()

Auth\_url\_ppe = self.auth\_url\_ppe\_input.text()

Query\_url\_ppe = self.query\_url\_ppe\_input.text()

# Call the backend function with user inputs

run\_backend(assetclass, r\_type, username, password, Auth\_url\_prod, Query\_url\_prod, Auth\_url\_ppe, Query\_url\_ppe)

if \_\_name\_\_ == "\_\_main\_\_":

app = QApplication(sys.argv)

window = AppWindow()

window.show()

sys.exit(app.exec\_())

scriot2

import sys

import configparser

from PyQt5.QtWidgets import QApplication, QMainWindow, QWidget, QVBoxLayout, QLabel, QLineEdit, QComboBox, QPushButton

# Import your backend functions and modules here

# Replace 'backend\_functions' with the actual name of your backend script/module

import backend\_functions

# Function to run the backend script with user inputs

def run\_backend():

# Get the selected configuration options from the frontend

assetclass = assetclass\_combo.currentText()

r\_type = r\_type\_combo.currentText()

username = username\_input.text()

password = password\_input.text()

environment = environment\_combo.currentText()

# Call your backend functions with the provided inputs

backend\_functions.Get\_moontoken(username, password, environment)

backend\_functions.Get\_moonquery(assetclass, r\_type)

backend\_functions.Get\_moon\_data()

backend\_functions.Process\_moon\_data()

backend\_functions.Write\_output()

backend\_functions.compare\_market\_pricing\_data()

backend\_functions.compare\_highlighting\_data()

class AppWindow(QMainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setWindowTitle("Backend Integration")

self.setGeometry(100, 100, 600, 400)

self.central\_widget = QWidget()

self.setCentralWidget(self.central\_widget)

self.layout = QVBoxLayout()

# Create a ConfigParser object and read the configuration file

config = configparser.ConfigParser()

config.read('config.ini')

# Create input fields for login and password

self.username\_label = QLabel("Username:")

self.username\_input = QLineEdit()

self.layout.addWidget(self.username\_label)

self.layout.addWidget(self.username\_input)

self.password\_label = QLabel("Password:")

self.password\_input = QLineEdit()

self.password\_input.setEchoMode(QLineEdit.Password) # To hide the password

self.layout.addWidget(self.password\_label)

self.layout.addWidget(self.password\_input)

# Create a dropdown for environment selection

self.environment\_label = QLabel("Environment:")

self.environment\_combo = QComboBox()

self.environment\_combo.addItems(["prod", "ppe"]) # Add your environment options

self.layout.addWidget(self.environment\_label)

self.layout.addWidget(self.environment\_combo)

# Create a dropdown for asset class selection

self.assetclass\_label = QLabel("Asset Class:")

self.assetclass\_combo = QComboBox()

self.assetclass\_combo.addItems(config.sections())

self.layout.addWidget(self.assetclass\_label)

self.layout.addWidget(self.assetclass\_combo)

# Create a dropdown for r-type selection (initially empty)

self.r\_type\_label = QLabel("R Type:")

self.r\_type\_combo = QComboBox()

self.layout.addWidget(self.r\_type\_label)

self.layout.addWidget(self.r\_type\_combo)

# Connect asset class selection to r-type options

self.assetclass\_combo.currentTextChanged.connect(self.load\_r\_types)

# Create a "Run Backend" button

self.run\_button = QPushButton("Run Backend")

self.run\_button.clicked.connect(run\_backend)

self.layout.addWidget(self.run\_button)

self.central\_widget.setLayout(self.layout)

def load\_r\_types(self):

# Clear existing r-type options

self.r\_type\_combo.clear()

# Create a ConfigParser object and read the configuration file

config = configparser.ConfigParser()

config.read('config.ini')

# Get the selected asset class

selected\_asset\_class = self.assetclass\_combo.currentText()

# Get the r-types for the selected asset class

if selected\_asset\_class in config:

self.r\_type\_combo.addItems(config[selected\_asset\_class].keys())

if \_\_name\_\_ == "\_\_main\_\_":

app = QApplication(sys.argv)

window = AppWindow()

window.show()

sys.exit(app.exec\_())

script3