A Purposeful Walk Down Wallstreet

Exploring Advanced Data Analytics in Financial Markets



Design Specification

Version 3.0

July 30, 2020

Frino Jais Sri Padmini Jayanti William Aman Minhajul Abadeen

**VERSION HISTORY**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented By** | **Revision Date** | **Approved By** | **Approval Date** | **Reason** |
| 1.0 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 06/18/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 06/18/2020 | Original document submitted to Professor Seyed |
| 2.0 | William Aman | 06/23/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 06/23/2020 | Corrected a bolding inconsistency and unnecessary amount of white space after Appendix A |
| 3.0 | Frino Jais | 07/01/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/01/2020 | -Fixed Data Flow Diagram to only show the flow of data  - Fixed Use Case Diagram removed unnecessary parts |
| 4.0 | Frino Jais | 07/02/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/02/2020 | -Added Communication Architecture Diagram |
| 5.0 | Frino Jais | 07/03/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/02/2020 | -Added Communication Architecture Description |
| 6.0 | Sri Padmini Jayanti | 07/05/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/05/2020 | - Added new use cases and edited the previous ones, more detailed descriptions and necessary columns added |
| 7.0 | Frino Jais | 07/11/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/11/2020 | - Updated Use Case Diagram per Sri’s edits |
| 8.0 | Minhajul Abadeen | 07/11/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/11/2020 | - Updated Design Constraints and Assumptions |
| 9.0 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/29/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/29/2020 | - Fixed Design Constraints & formatting |

**TABLE OF CONTENTS**

[1 Introduction 4](#_Toc46940031)

[1.1 Purpose Of The Product Design Specification Document 4](#_Toc46940032)

[2 General Overview And Design Guidelines 4](#_Toc46940033)

[2.1 Assumptions / Constraints / Standards 4](#_Toc46940034)

[2.2 Technical Constraints 4](#_Toc46940035)

[2.3 Business Constraints 5](#_Toc46940036)

[2.4 Design Constraints 5](#_Toc46940037)

[3 Architecture Design 5](#_Toc46940038)

[3.1 Hardware Architecture 5](#_Toc46940039)

[3.2 Software Architecture 5](#_Toc46940040)

[3.3 Security Architecture 7](#_Toc46940041)

[3.4 Communication Architecture 7](#_Toc46940042)

[3.5 Performance 8](#_Toc46940043)

[4 System Design 9](#_Toc46940044)

[4.1 Use-Cases 9](#_Toc46940045)

[4.2 Sequence Diagram 15](#_Toc46940046)

[4.3 Data Flow Diagram 16](#_Toc46940047)

[4.4 Database Design 17](#_Toc46940048)

[4.5 Class Diagram 18](#_Toc46940049)

[4.6 Application Program Interfaces 19](#_Toc46940050)

[4.7 User Interface Design 19](#_Toc46940051)

[5 Product Design Specification Approval 24](#_Toc46940052)

[Appendix A: References 25](#_Toc46940053)

[Appendix B: Key Terms 25](#_Toc46940054)

# 

# 1 Introduction

## 1.1 Purpose Of The Product Design Specification Document

The purpose of this Product Design Specification document is to provide an overview for the design and architecture of the GM Fintech Application. This document will establish the necessary information required to define the construction of the overall application. The target audience of this documentation is aimed towards the business analyst, data analyst, product owners, and developers. The document provides a baseline of the overall specification of the software design. The design document should outline the applications functionality end to end, from the front-end, middle layer, and the back end. This document is used to define a set of architectural guidelines for developers such as interface designs, software architectural layers, and use cases.

# 2 General Overview And Design Guidelines

## 2.1 Assumptions / Constraints / Standards

* + 1. Power BI must be installed to view all the visualizations along with a connection from the database to Power BI
    2. All packages must be separately installed once the user downloads PyCharm 2019.3
    3. MySQL Workbench with the right schema and tables
    4. A proper connection is established between the database, the application, and the front end

## 2.2 Technical Constraints

1. Programming language: Python code used as the back-end structure
2. Data Visualization: Power BI used as the front-end graphical software
3. pandas-datareader: Reads python data
4. Database Management System: MySQL Workbench 8.0.20
5. Statistical Python Functions: FRL, MSF, ARIMA, Cross Moving Averages, Polynomial Regression, etc
6. NumPy: Statistical library used for operations on multidimensional arrays and matrices
7. SQL packages: PyMySQL must be used as the primary MySQL API connection, and SQLAlchemy must be installed to abstract SQL data
8. Testing: PyTest used for unit testing
9. Data Extraction: Yahoo! Finance packages must be installed to extract instrument statistic data as well as Quandl to extract macroeconomic variables

## 2.3 Business Constraints

1. The main server (gmfsp\_db) does not connect to the python code or Power BI is slow
2. Issues connecting to the internet
3. Outdated or inaccurate data extracted
4. Inaccurate predictions from python compilation

## 2.4 Design Constraints

1. API for online sources are functional so that data can be pulled by the code.
2. The free version of Power BI Desktop does not allow merging of workbooks.

# 3 Architecture Design

## 3.1 Hardware Architecture

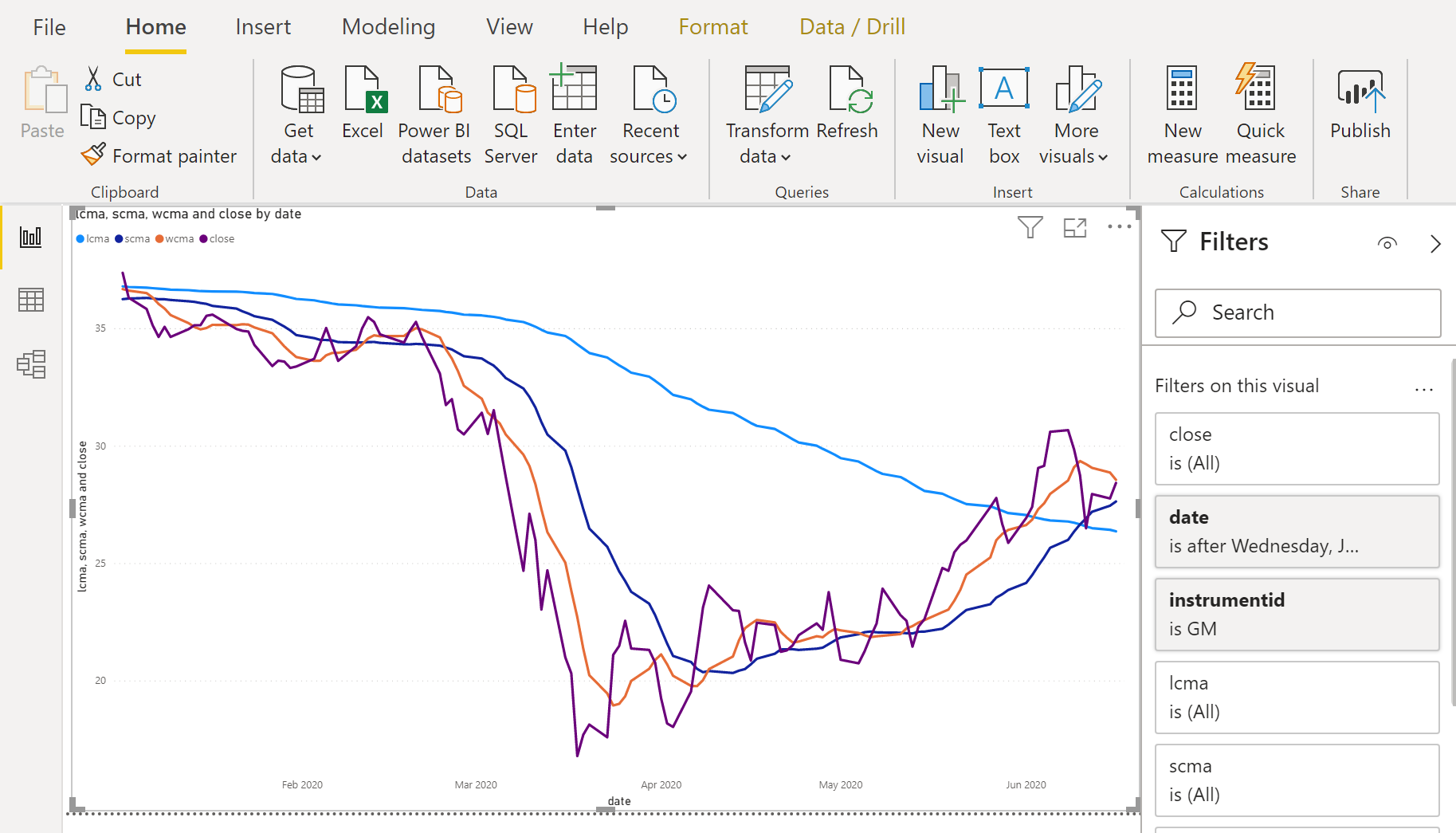
The GM Fintech Application does not have any specific hardware architecture that needs to be used to run the program. It is a standalone application which can run on any personal computer with the required software architecture setup.

## 3.2 Software Architecture

The software architecture for the GM Fintech Application follows a layered architecture pattern. Details about each level of abstraction are as follows:

Presentation Layer:

The Presentation Layer, also known as the User Interface Layer, for this application is Power BI. This layer will be responsible for the delivery of information to the user in an understandable graphical manner. For the GM Fintech Application, the platform that will be used for this layer will be Power BI. Power BI will act as the User Interface for this application to give visualizations of analysis performed by the algorithms and strategies implemented in the Domain Layer by retrieving the stored data from the Database Layer. Figure 1 shows a view from Power BI, depicting the cross moving averages and close values for General Motors.



***Figure 1. CMA view for GM.***

Domain Layer:

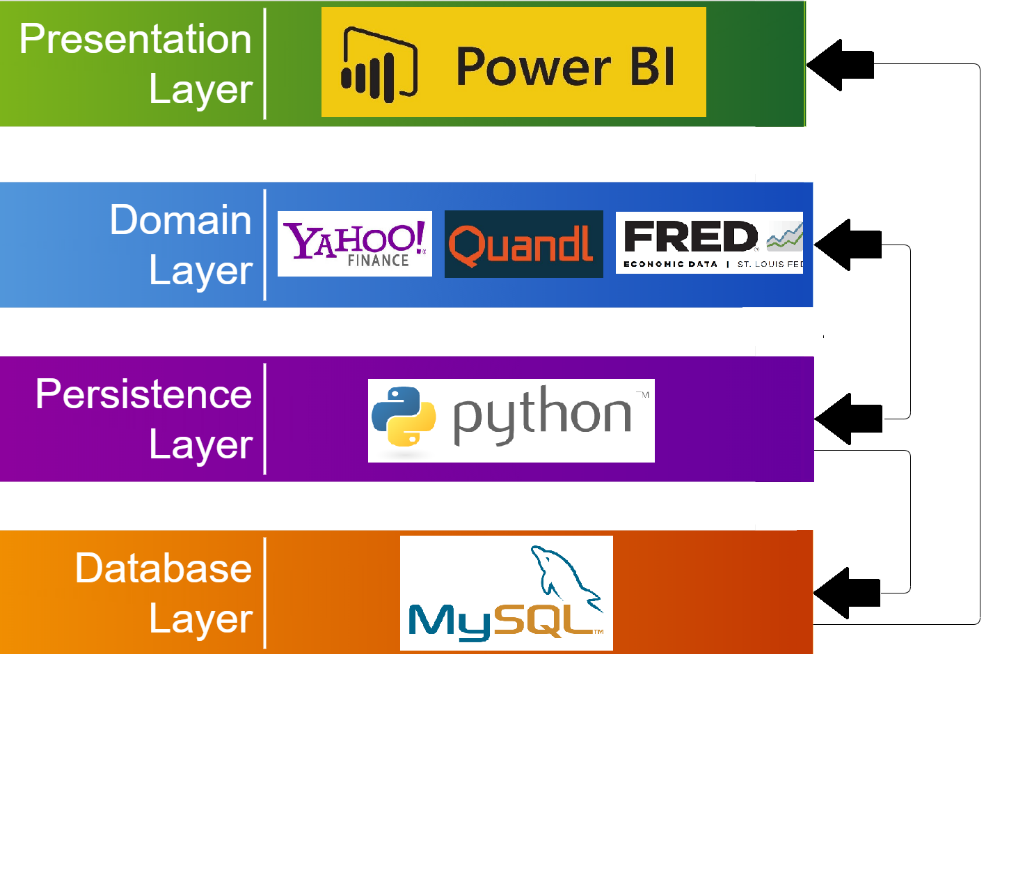
The Domain Layer, also referred to as the Business Layer, encapsulates the rules and logic of the of the application. For the GM Fintech Application, the components for this layer will be the sources used to retrieve the data needed for analysis. These sources include Yahoo! Finance, Quandl and FRED. Details and more information about these sources can be found in Section 4.6, Application Development Interfaces.

Persistence Layer:

The Persistence Layer, also known as the Data Access Layer, is responsible for retrieving the data from the predetermined sources as well as accessing the data in the database layer. Additionally, any operations performed on this data takes place in this layer. Therefore, the components of the Persistence Layer relative to the GM Fintech Application include the Python classes and their respective functions used to carry out the strategies for this application.

Database Layer:

The Database Layer is the layer of the application that provides simplified access to the stored data in the application. The main component of this layer is the Database Management System used for the application which is MySQL.



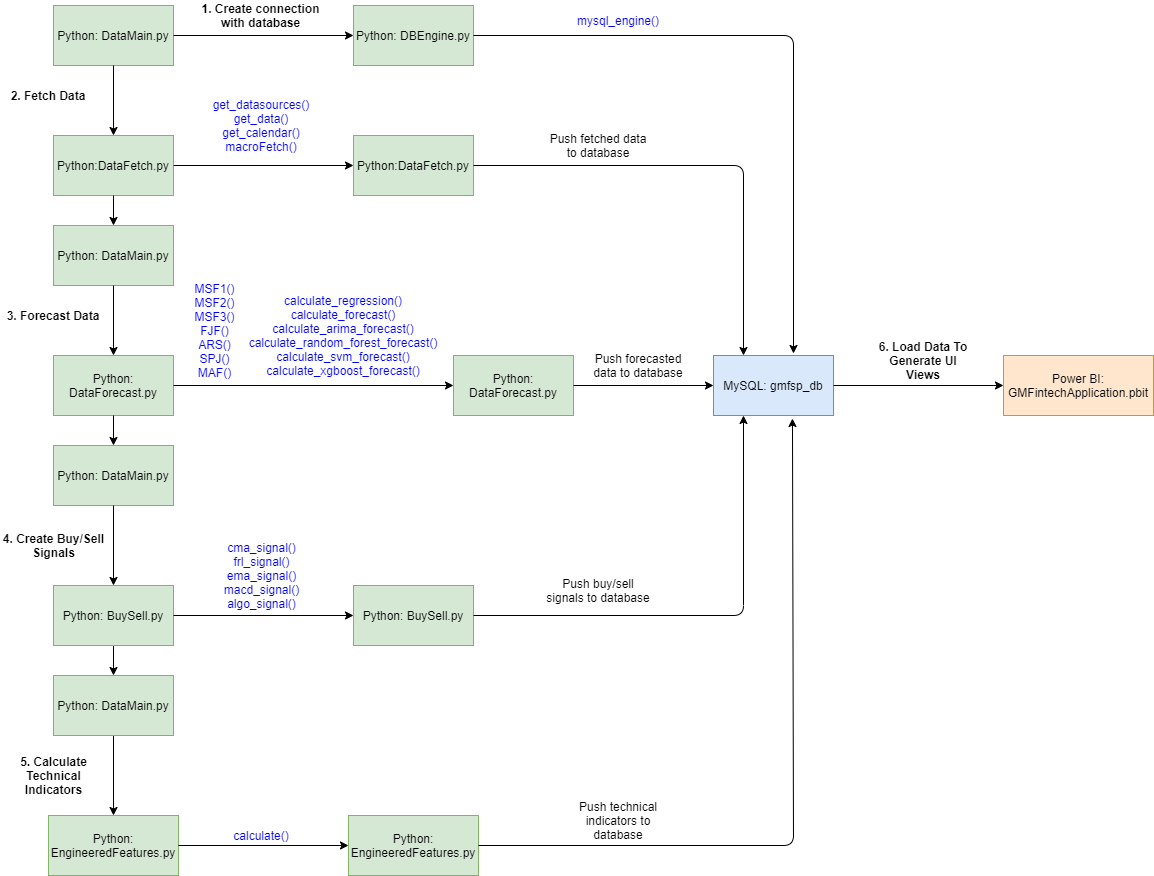
***Figure 2. Software Architecture Diagram.***

## 3.3 Security Architecture

There is no security architecture that needs to be implemented for the GM Fintech Application since any stored data is only accessible from the user’s local machine. If a new user would like to access the application, a full setup must be done as any of the tools, platforms, or data cannot be easily transferred over a server. In the current stage of the application, security is not part of the scope for this project.

## 3.4 Communication Architecture

The communication architecture for the GM Fintech Application deals with three main components: the Python code, MySQL, and Power BI. When the application runs, the Python code will be storing data from online data sources into MySQL as well as perform analysis on this data to generate the forecasted values and signals to buy or sell. Visualizations of these outputs will be given when the stored data is loaded into Power BI. An overview of the communication architecture can be seen in the figure below.



## 3.5 Performance

The performance of the GM Fintech Application relies heavily on the user’s internet connection as explained in the communication architecture for this program. The performance time frame will be from when the user initially runs the Python code to when the database is populated with new data. If the user satisfies all requirements for the installation of the application, a complete execution can take less than 30 seconds.

# 4 System Design

## 4.1 Use-Cases

|  |  |
| --- | --- |
| **UC-1.0** | **Update Financial Instruments Data Everyday** |
| Actor(s) | User |
| Description | This use case will describe the necessary operations a user has to perform to ensure up-to-date data is pulled by the application program. This data is being extracted from Yahoo! Finance. |
| Trigger | There will be two triggers for this use case:  1.MySQL “localhost” server connection established by the user  2.User opens DataMain.py |
| Preconditions | The database shows data from previous date |
| Postconditions | The dbo\_instrumentstatistics table in the database is updated with present day information for all the 10 financial instruments used in the application. |
| Normal Flow | 1. Open MySQL “localhost” server and establish a successful connection by entering the following credentials:  Username: root  Password: password  2. Run DataMain.py located in the W2020 tab.  3.Run the following SQL query in MySQL Workbench:  *SELECT \* FROM gmfsp\_db.dbo\_instrumentstatistics;* |
| Alternate Flow | NONE |
| Frequency of use | Once every working day |
| Assumptions | 1. User has all the necessary software installed 2. MySQL “localhost” server connection has been successfully established by the user using the above specified credentials 3. “gmfsp\_db” schema is created using the README guidelines 4. User has all the necessary Python dependencies installed |

|  |  |
| --- | --- |
| **UC-2.0** | **Update Macroeconomic Data Every Month** |
| Actor(s) | User |
| Description | This use case will describe the necessary operations a user has to perform to refresh data that is being extracted from Quandl. |
| Trigger | There will be two triggers for this use case:  1.MySQL “localhost” server connection established by the user  2.User opens DataMain.py |
| Preconditions | The database shows data from last date of the previous month |
| Postconditions | The dbo\_macroeconstatistics table in the database is updated with macroeconomic data for all the macro economic variables. |
| Normal Flow | 1. Open MySQL “localhost” server and establish a successful connection by entering the following credentials:  Username: root  Password: password  2. Run DataMain.py located in the W2020 tab.  3.Run the following SQL query in MySQL Workbench:  *SELECT \* FROM gmfsp\_db.dbo\_macroeconstatistics;* |
| Alternate Flow | NONE |
| Frequency of use | Once at the end of every month |
| Exceptions | The macro economic data updates on itself every day when DataMain.py is run. However, this is one use case that the user needs to keep an eye for. |
| Assumptions | 1.User has all the necessary software installed  2.MySQL “localhost” server connection has been successfully established by the user using the above specified credentials  3.“gmfsp\_db” schema is created using the README guidelines  4.User has all the necessary Python dependencies installed |

|  |  |
| --- | --- |
| **UC-3.0** | **Update Data in Power BI Views** |
| Actor(s) | User |
| Description | This use case describes how the user can view refreshed data in the Power BI views |
| Trigger | User opens the “GMFintechApplication.pbit” Power BI Dashboard |
| Preconditions | 1.The graphs show previous date’s values  2.U.C-1.0 is satisfied/completed |
| Postconditions | Dashboards in Power BI are updated |
| Normal Flow | 1. Open GMFintechApplication.pbit  2. Click “Refresh” from the “Home” tab  3.Check the last refresh information by hovering over to the respective table in the “Fields” column in the Power BI interface |
| Alternate Flow | 1. Open GMFintechApplication.pbit 2. Right click on the respective table present in the “Fields” column in the Power BI interface 3. Click on “Refresh data” option 4. Check the updated date by hovering over on the visual to see the latest date |
| Frequency of use | Data needs to only be refreshed after each business day |
| Assumptions | 1. User has all the necessary software installed 2. MySQL “localhost” server connection has been successfully established by the user using the above specified credentials 3. “gmfsp\_db” schema is created using the README guidelines 4. User has Power BI installed on their computer 5. User has all the necessary Python dependencies installed 6. User has already run the DataMain.py, so the MySQL database is updated |

|  |  |
| --- | --- |
| **UC-3.1** | **Power BI Shows CMA/Close View For All Instruments** |
| Actor(s) | User |
| Description | Power BI will show a graphical representation of the CMA/close values for all financial instruments under review for this project (GM, CARZ, XPH, PFE, SPY, ^TYZ, FCAU, TM, F, HMC) |
| Trigger | User clicks on the dashboard named “Closing Price & Cross Moving Average” |
| Preconditions | UC-1.0 |
| Postconditions | “Closing Price & Cross Moving Average” view will display an updated graph with the data generated by the program |
| Normal Flow | 1. Open GMFintechApplication.pbit  2. Click “Closing Price & Cross Moving Average” dashboard  3. “Closing Price & Cross Moving Average” view will display |
| Alternate Flow | 1. Open GMFintechApplication.pbit  2. Click “Closing Price & Cross Moving Average” dashboard  3. Click “Refresh” from the “Home” tab  4. “Closing Price & Cross Moving Average” view will display |
| Exceptions | N/A |
| Assumptions | All necessary python dependencies are installed  User has Power BI installed  gmfsp\_db is set as the data source for Power BI |

|  |  |
| --- | --- |
| **UC-3.2** | **Power BI Shows Macroeconomic View For All Indicators** |
| Actor(s) | User |
| Description | Power BI will show a graphical representation of the macroeconomic statistics for all economic indicators under review for this project (Unemployment Rate, GDP, Interest Rate, Misery Index, COVI, CPIUC, FSI) |
| Trigger | User clicks on dashboard named “Macroeconomic Variables” |
| Preconditions | UC-1.0, UC-2.0 |
| Postconditions | “Macroeconomic Variables” view will display an updated graph with the data generated by the program |
| Normal Flow | 1. Open GMFintechApplication.pbit  2. Click “Macroeconomic Variables” dashboard  3. “Macroeconomic Variables” view will display |
| Alternate Flow | 1. Open GMFintechApplication.pbit  2. Click “Macroeconomic Variables” dashboard  3. Click “Refresh” from the “Home” tab  4. “Macroeconomic Variables” view will display |
| Exceptions | N/A |
| Assumptions | All necessary python dependencies are installed  User has Power BI installed  gmfsp\_db is set as the data source for Power BI |

|  |  |
| --- | --- |
| **UC-3.3** | **Power BI Shows Stock Performance View For All Instruments** |
| Actor(s) | User |
| Description | Power BI will show a graphical representation of the stock statistics for all financial instruments under review for this project (GM, CARZ, XPH, PFE, SPY, ^TYZ, FCAU, TM, F, HMC) |
| Trigger | User clicks on the dashboard named “Stock Performance” |
| Preconditions | UC-1.0 |
| Postconditions | “Stock Performance” view will display an updated graph with the data generated by the program |
| Normal Flow | 1. Open GMFintechApplication.pbit  2. Click “Stock Performance” dashboard  3. “Stock Performance” view will display |
| Alternate Flow | 1. Open GMFintechApplication.pbit  2. Click “Stock Performance” dashboard  3. Click “Refresh” from the “Home” tab  4. “Stock Performance” view will display |
| Exceptions | N/A |
| Assumptions | All necessary python dependencies are installed  User has Power BI installed  gmfsp\_db is set as the data source for Power BI |

|  |  |
| --- | --- |
| **UC-3.4** | **Power BI Shows Data Forecast View For All Instruments** |
| Actor(s) | User |
| Description | Power BI will show a graphical representation of the forecast values for all financial instruments under review for this project (GM, CARZ, XPH, PFE, SPY, ^TYZ, FCAU, TM, F, HMC) |
| Trigger | User clicks on the dashboard named “Data Forecast” |
| Preconditions | UC-1.0 |
| Postconditions | “Data Forecast” view will display an updated graph with the data generated by the program |
| Normal Flow | 1. Open GMFintechApplication.pbit  2. Click “Data Forecast”” dashboard  3. “Data Forecast” view will display |
| Alternate Flow | 1. Open GMFintechApplication.pbit  2. Click “Data Forecast” dashboard  3. Click “Refresh” from the “Home” tab  4. “Data Forecast” view will display |
| Exceptions | N/A |
| Assumptions | All necessary python dependencies are installed  User has Power BI installed  gmfsp\_db is set as the data source for Power BI |

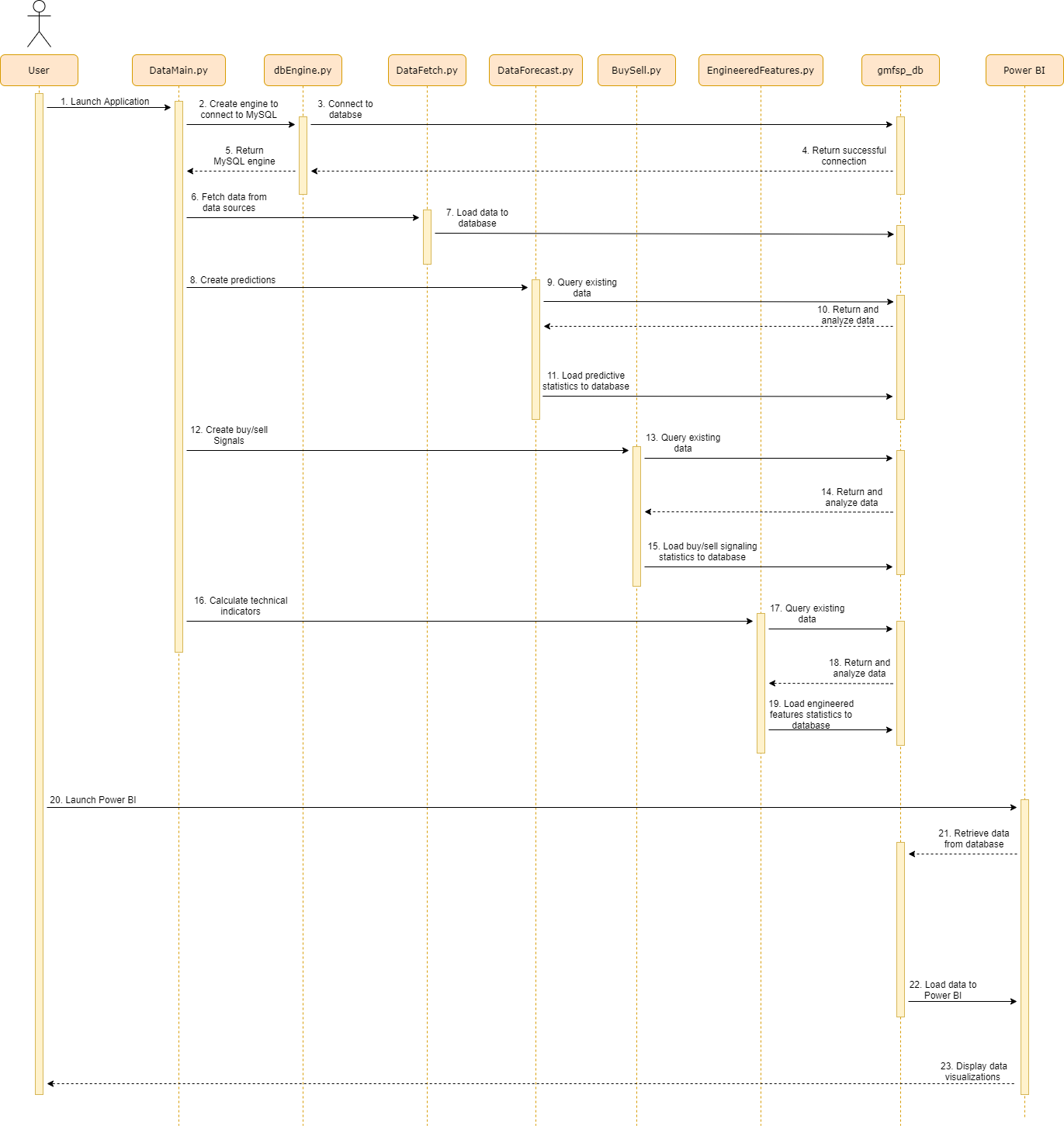
|  |  |
| --- | --- |
| **UC-3.5** | **Power BI Shows Buy/Sell Signals View For All Instruments** |
| Actor(s) | User |
| Description | Power BI will show a graphical representation of the buy/sell signals for all financial instruments under review for this project (GM, CARZ, XPH, PFE, SPY, ^TYZ, FCAU, TM, F, HMU) |
| Trigger | User clicks on the dashboard named “Buy/Sell Signals” |
| Preconditions | UC-1.0 |
| Postconditions | “Buy/Sell Signals” view will display an updated graph with the data generated by the program |
| Normal Flow | 1. Open GMFintechApplication.pbit  2. Click “Buy/Sell Signals” dashboard  3. “Buy/Sell Signals” view will display |
| Alternate Flow | 1. Open GMFintechApplication.pbit  2. Click “Buy/Sell Signals” dashboard  3. Click “Refresh” from the “Home” tab  4. “Buy/Sell Signals” view will display |
| Exceptions | N/A |
| Assumptions | All necessary python dependencies are installed  User has Power BI installed  gmfsp\_db is set as the data source for Power BI |

A close up of a map

Description automatically generated

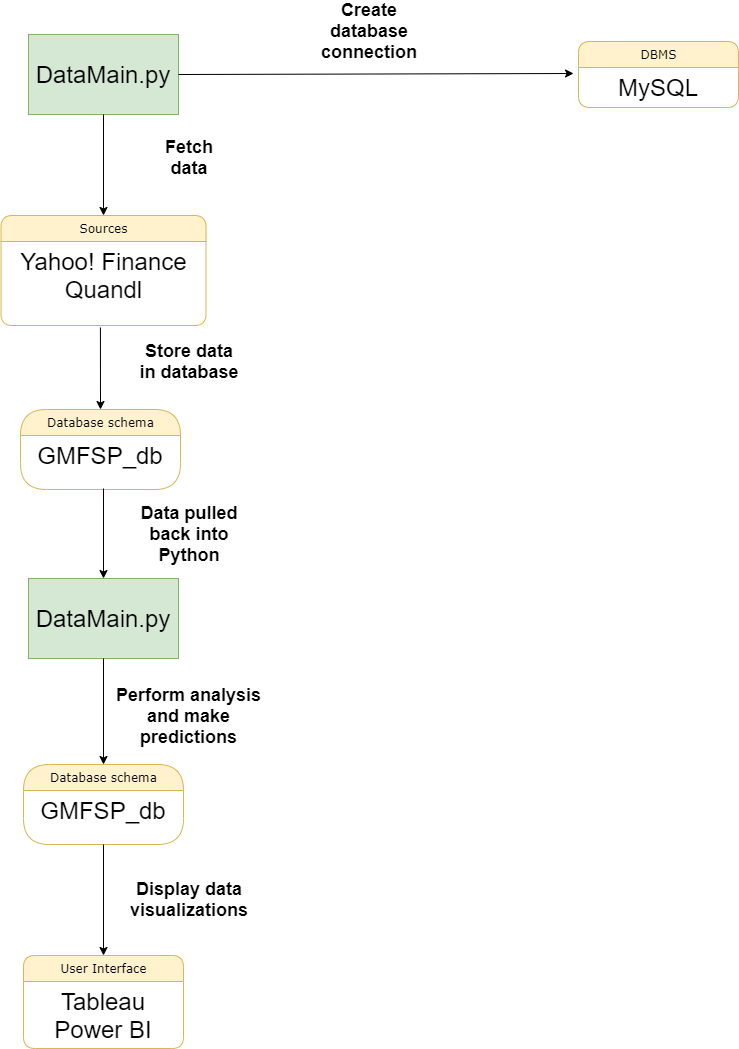
***Figure 3. Use Case Diagram.***

## 4.2 Sequence Diagram



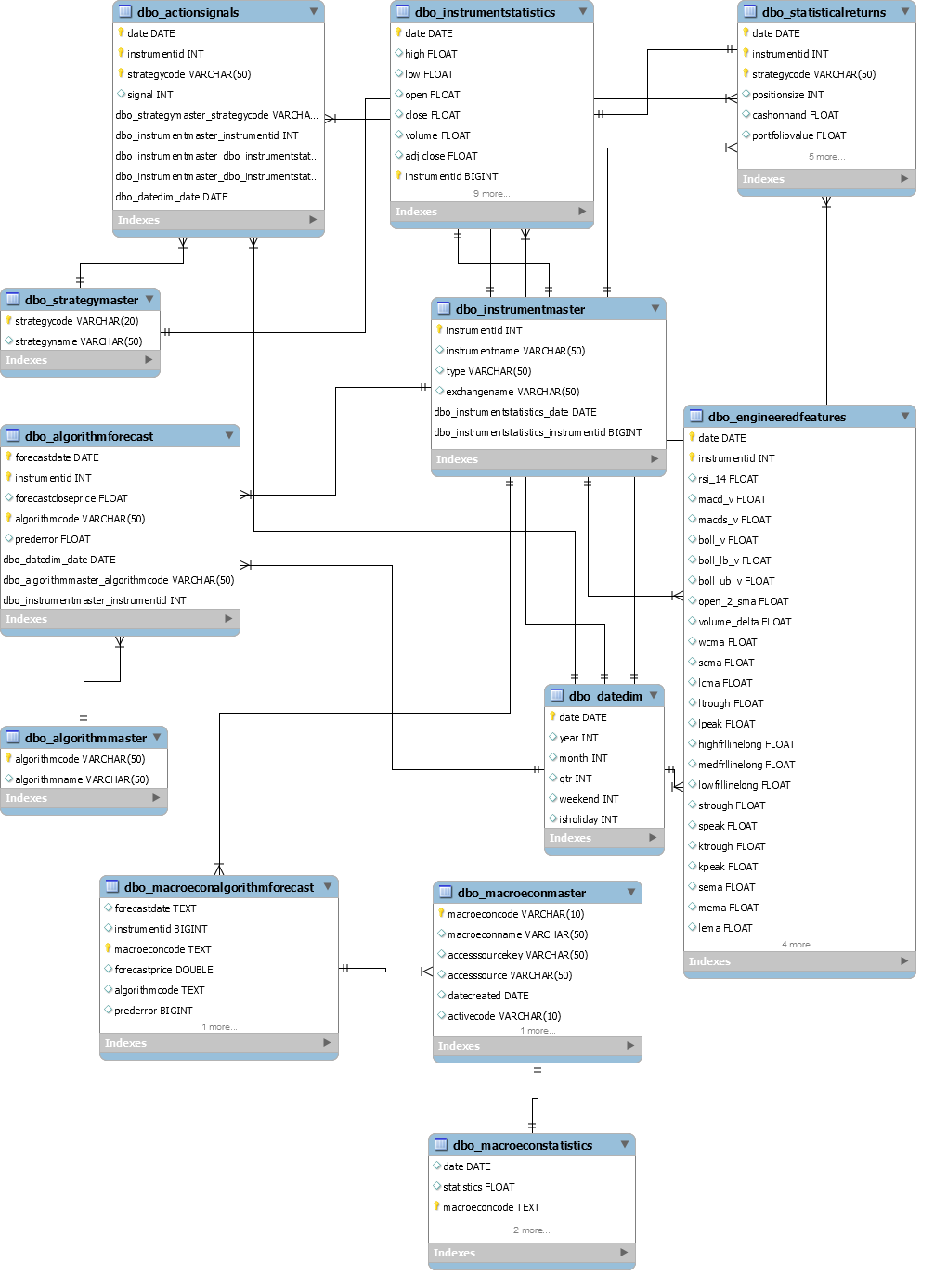
***Figure 4. Sequence Diagram.***

## 4.3 Data Flow Diagram



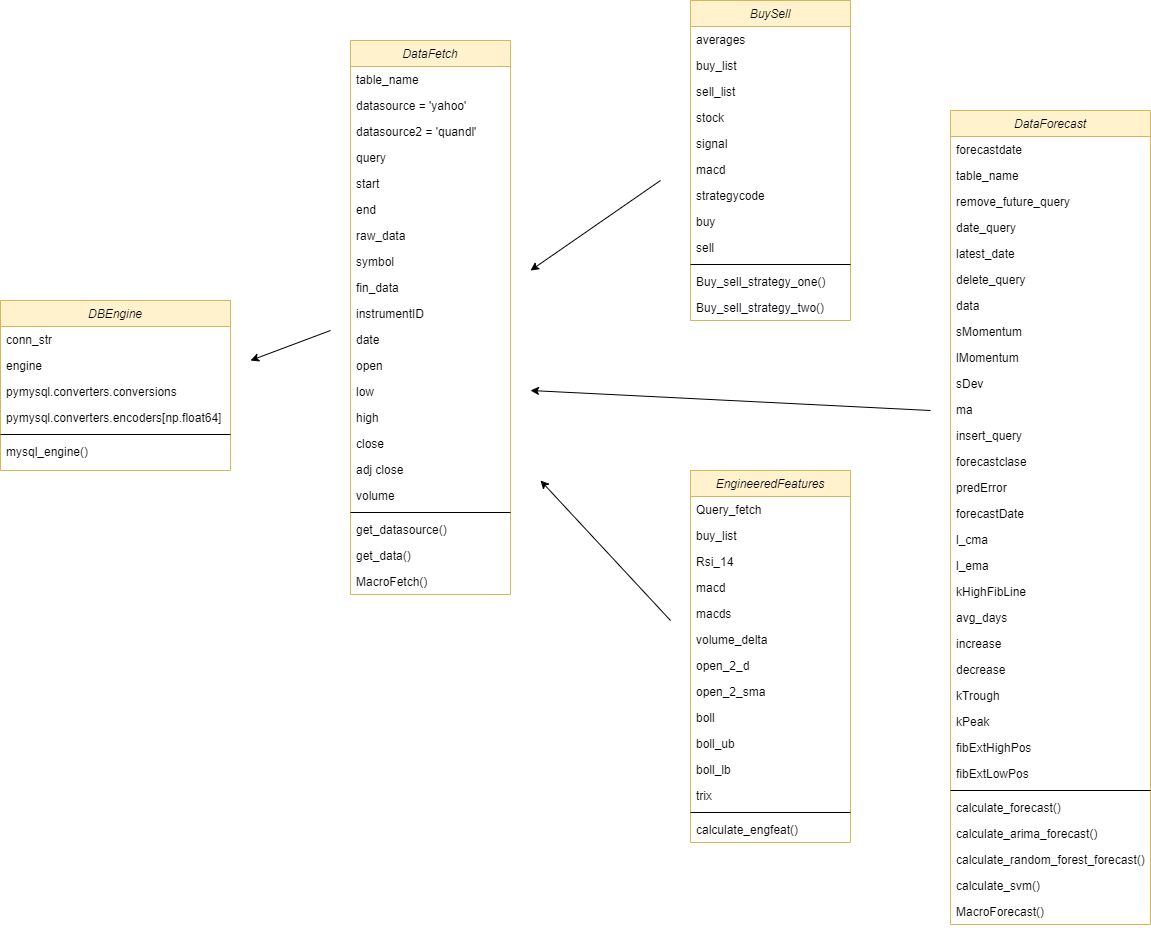
***Figure 5. Data Flow Diagram.***

## 4.4 Database Design



***Figure 6. Database Design.***

## 4.5 Class Diagram



***Figure 7. Class Diagram.***

## 4.6 Application Program Interfaces

1. Yahoo! Finance API

The GM Fintech Application retrieves the necessary stock market data from Yahoo! Finance using an imported dependency called pandas-datareader. This package allows users to pull information and create data frames from various internet data sources. Yahoo! Finance is among the supported data sources for this package. The application will use this data as the foundation for movement predictions.

2. Quandl API

Quandl is an online platform for financial, economic, and alternative data delivered in modern formats for ease of analysis. Quandl offers a free financial data API that makes accessing its data extremely convenient. The GM Fintech Application will leverage this API to include economic variables and factors when determining predictions in the financial instrument movements.

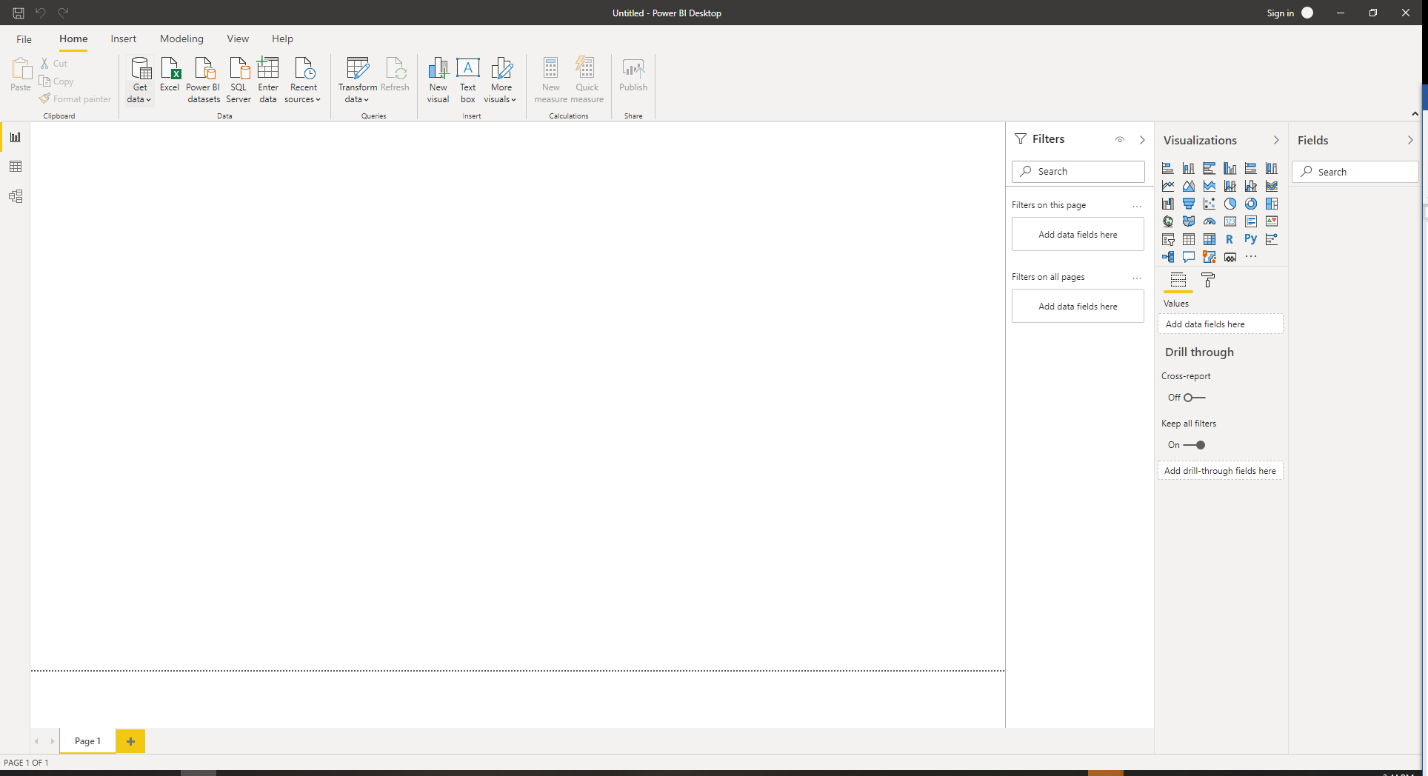
3. FRED API

The Python API “fredapi” is the API for FRED (Federal Reserve Economic Data). This data is provided by the Federal Reserve Bank of St. Louis. This fredapi provides a wrapper in Python to the FRED web service and also provides various methods for parsing and analyzing time series data. The GM Fintech Application will use this API to further expand on economic factors that play a role in stock market performance.

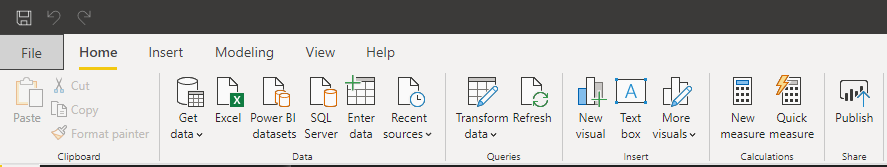
## 4.7 User Interface Design

The user interface of this project will be presented through the use of Power BI. Power BI will be used to provide the user with charts and graphs relating to various components and information relevant to stock data. These include macroeconomic charts, closing prices of various stocks, buy/sell prediction charts, and many more. Below are a few sample images of what the user interface looks like, accompanied by a caption explaining what is being shown.

Below is what Power BI looks like from a fresh launch of the application. Seeing as this program was created by Microsoft, it resembles the Microsoft Office suite in many ways. Even though the user may have never used Power BI before, this familiar look to the interface makes the software very approachable.

***Figure 8. Default Power BI Launch Window.***

Below is the ribbon of Power BI. This section of the interface resides at the top of the application window. The ribbon contains most of the commands that the user will be needing for this application. Selecting a database, transforming data, refreshing the data source, importing/exporting reports, and much more all start at the ribbon.



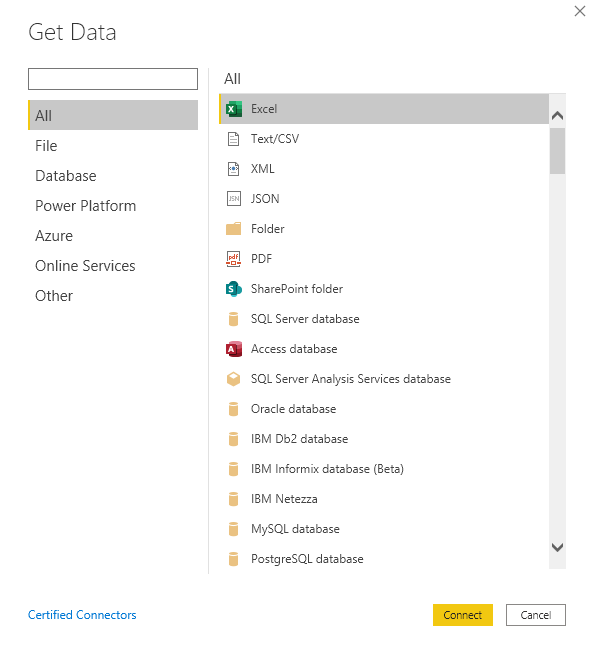
***Figure 9. Tabs, Groups, and Commands.***

Below is the page navigation section of Power BI. This section resides at the bottom of the application window. Power BI is capable of housing hundreds of dashboards in the same report. This navigator allows for changing which dashboard is being displayed at any given moment.

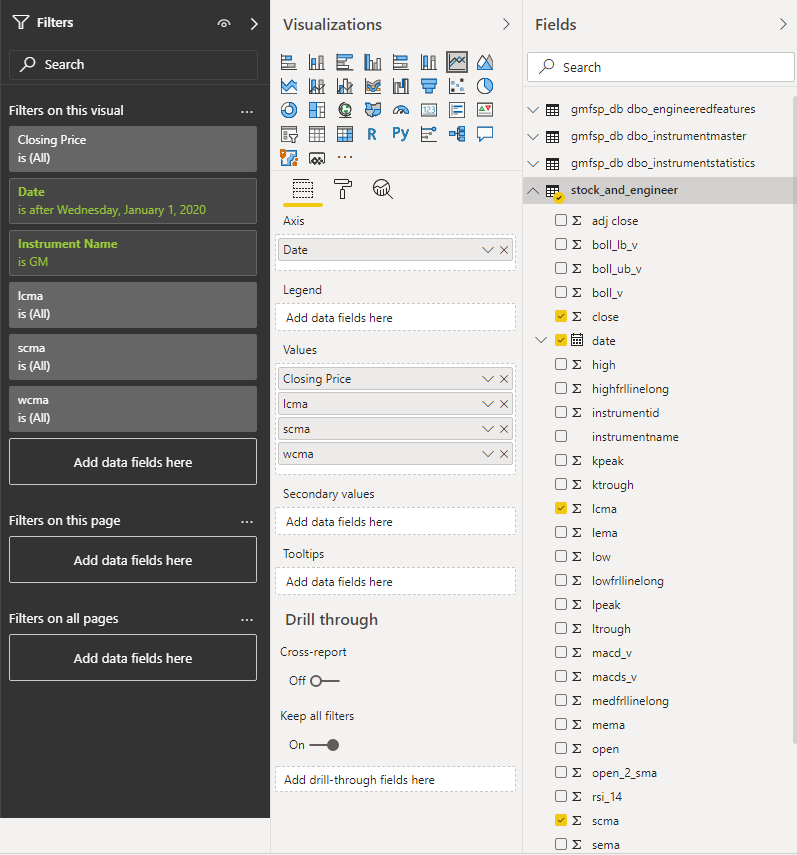


***Figure 10. Dashboard Navigator.***

Below is the dataset selection window. Power BI is oriented around creating visualizations for various types of datasets. This window allows the user to select the source of their data.

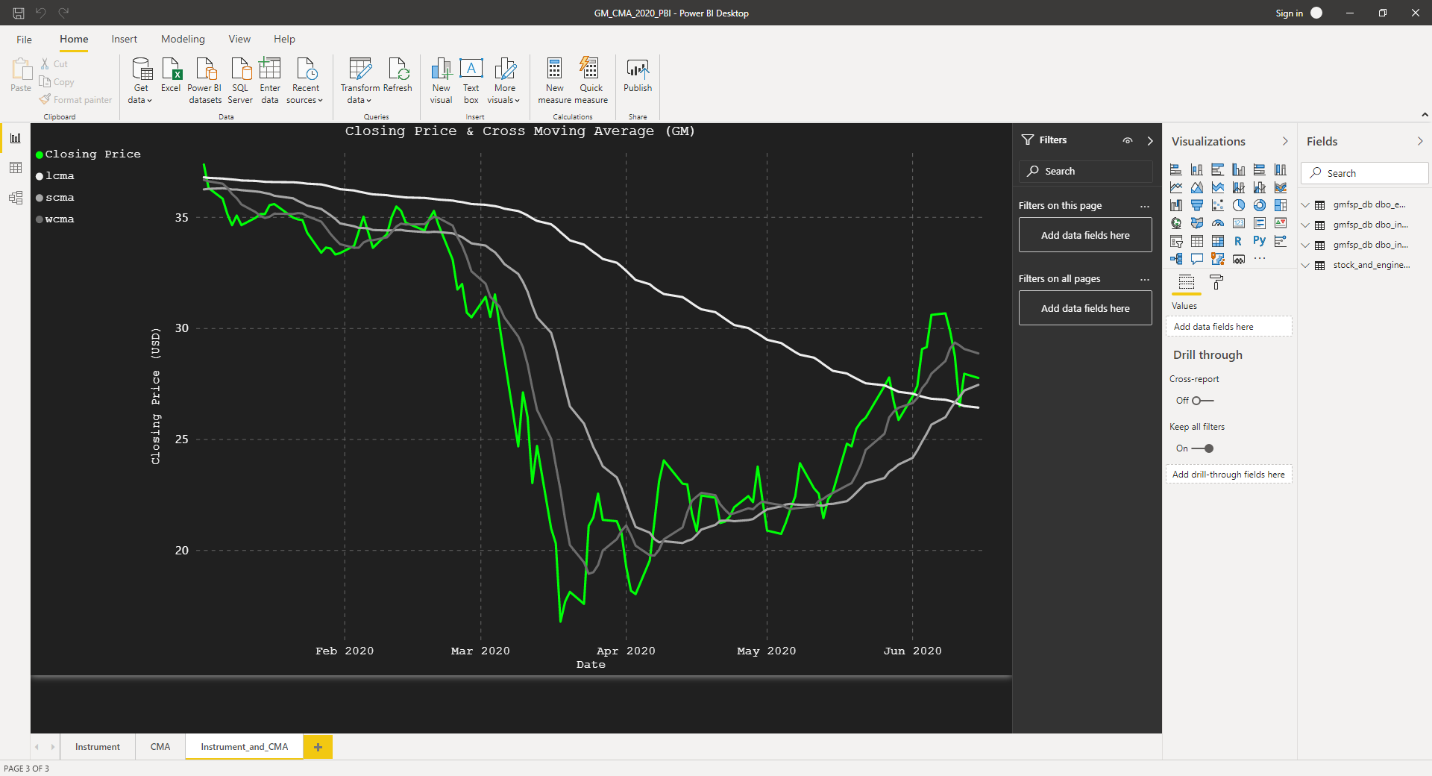
 ***Figure 11. Dataset Selection Window.***

Below is the visualization customization section. This portion of Power BI resides to the right of the visualization window. As the ribbon contains commands that can be used across the whole application, this section contains commands specific to the current visualization.



***Figure 12. Power BI Chart Development Section (Filters, Visualizations, Fields).***

Below is a sample dashboard. This is just one dashboard in the current report. There are two other dashboards in this specific report, as you can see in the dashboard navigation section of the application. Multiple visualizations can be assembled in the dashboard if needed.



***Figure 13. Sample Dashboard.***

# 5 Product Design Specification Approval

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Date: | \_\_\_\_\_\_\_\_\_\_\_\_ |
| Print Name: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Title: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Role: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Date: | \_\_\_\_\_\_\_\_\_\_\_\_ |
| Print Name: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Title: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Role: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Date: | \_\_\_\_\_\_\_\_\_\_\_\_ |
| Print Name: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Title: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Role: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Date: | \_\_\_\_\_\_\_\_\_\_\_\_ |
| Print Name: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Title: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Role: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |

# Appendix A: References

|  |  |  |
| --- | --- | --- |
| **Document Name and Version** | **Description** | **Location** |
| Glossary for Power BI service | A terminology directory for various terms related to Power BI | [Link](https://docs.microsoft.com/en-us/power-bi/consumer/end-user-glossary) |

# Appendix B: Key Terms

|  |  |
| --- | --- |
| **Term** | **Definition** |
| .pbit | An extension of Power BI that keeps the report’s structure, but not any of the data associated with the report |
| .pbix | An extension of Power BI that stores all the imported data from the data sources and the report structure |
| Power BI | A business analytics program made by Microsoft meant for creating data visualizations |
| API | Short for Application Program Interface. Defines interactions between multiple software programs. |
| ARIMA | Short for Auto-Regressive Integrated Moving Averages. It is a data science model that forecasts future values using its own past values |
| CMA | Short for Cross-Moving Average. It is an algorithm that detects when a shorter period moving average intersects a longer period moving average |
| Dashboard | A single screen of interactive visualizations and text |
| Data frame | A two-dimensional labeled data structure with columns of potentially different types |
| Dataset | A container for data |
| fredapi | A Python API for the FRED data provided by the Federal Reserve Bank of St. Louis. FRED stands for Federal Reserve Economic Data |
| FRL | Short for Fibonacci Retracement Lines. They are horizontal lines overlaid on a stock chart to indicate support and resistance to a price |
| gmfintechapplication.pbit | The name of the Power BI template file that contains all of the dashboards for this project |
| gmfsp\_db | The name of the database for the GM Fintech Application stored in MySQL |
| MSF | Short for Michael Shields Function. This is a collection of custom predictive algorithms |
| NumPy | A python library that is used for working with arrays for scientific computation |
| pandas-datareader | A python library that allows to create data frames from various internet data sources |
| Polynomial Regression | A form of regression analysis in which the relationship between the independent variable x and the dependent variable y is modelled as an nth degree polynomial in x |
| PyMySQL | A python focused MYSQL client library for connection |
| pytest | A framework in Python used to write code tests |
| Report | A collection of dashboards focused on the same dataset |
| SQLAlchemy | A Python toolkit that allows for the communication between Python programs and databases |
| Visualization | A visual display of data from a dataset that makes interpretation of the data easier |
| Yahoo! Finance | A source of financial data from which the application will source the stock data from |