A Purposeful Walk Down Wallstreet

Exploring Advanced Data Analytics in Financial Markets



Test Plan

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 | 07/23/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/23/2020 | Original document submitted to Professor Seyed |
| 2.0 | William Aman | 07/25/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/25/2020 | Added “Software Versions” and “Figures” sections in response to feedback on Version 1.0 from Professor Seyed |
| 3.0 | Minhajul Abadeen | 07/28/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/28/2020 | Added more functional test cases, improved upon engineered features, linear regression, and buy sell signal test cases, re numbered the test cases, and added a space between fail and pass manually |
| 4.0 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/29/2020 | Frino Jais  Sri Padmini Jayanti  Minhajul Abadeen  William Aman | 07/29/2020 | Added phases and resorted the test cases to reside in those phases |

**TABLE OF CONTENTS**

[1. Introduction 5](#_Toc46953822)

[2. Purpose 5](#_Toc46953823)

[3. Test Case Details 5](#_Toc46953824)

[3.1 ID 5](#_Toc46953825)

[3.2 Items to Test 5](#_Toc46953826)

[3.3 Pre-conditions 6](#_Toc46953827)

[3.4 Test Steps 6](#_Toc46953828)

[3.5 Expected Results 6](#_Toc46953829)

[3.6 Priority 6](#_Toc46953830)

[3.7 Pass/Fail 6](#_Toc46953831)

[5. Software Versions 7](#_Toc46953832)

[5.1 MySQL Workbench 7](#_Toc46953833)

[5.2 PyCharm IDE 7](#_Toc46953834)

[5.3 Power BI Desktop 7](#_Toc46953835)

[6. Figures 7](#_Toc46953836)

[6.1 MySQL Workbench 8](#_Toc46953837)

[6.2 PyCharm IDE 9](#_Toc46953838)

[6.3 Power BI Desktop 9](#_Toc46953839)

[4. Test Cases 10](#_Toc46953840)

[Phase 1: Unit Tests 10](#_Toc46953841)

[Phase 2: Integration Tests 17](#_Toc46953842)

[Phase 3: System Tests 90](#_Toc46953843)

[Nonfunctional Test Cases: 91](#_Toc46953844)

# 1. Introduction

This is the Test Plan document for the “A Purpose Walk Down Wallstreet” project. To get to this point, it means our team has completed all of the requirements as specified and agreed upon with the client. Now, this last step in the development process can be attempted. The Test Plan document is undertaken once the requirements have all been addressed, and the development team feels ready to test the features.

# 2. Purpose

The purpose of this document is to act as a checklist of sorts for verifying the completion and stability of the project as a whole. By detailing dozens of very specific test cases, the client as well as the development team can guarantee the successful completion of the project. The test cases can range from being extremely low level, unit tests, to all encompassing systems tests. This variation in the scale of test cases allows for a well-rounded test of the whole project. This document is very useful to identify which areas are functioning properly, and which may not be. Only once the many test cases all pass their required tests, then the project can be considered complete.

# 3. Test Case Details

Each test case is represented in a table of various components. Each row in the test case table needs to be expressed in great detail. This is important because the test cases need to be easily repeatable by anyone who is testing the project. To be able to understand the test cases, each component of the test case table needs to be clarified. Below, each of these components are explained so the tester or reader has a complete understanding of what the test cases are representing.

## 3.1 ID

The ID of the test case is very simple. It’s sole purpose is to be able to easily identify a specific test case, rather than trying to select it by explaining the test. By using an alphanumeric key, this makes referencing each test case very simple. Every ID must be unique from one another for the ID’s to be useful.

## 3.2 Items to Test

The items to test acts as a summary of what will be tested in the given test case. It’s lengthier than a title, but more concise than explaining the whole test in detail. It is meant to be used as digestible portion of the test case. The portion that anyone can read and know what is going to be tested. The specifics of the test do not reside here, those will come in the rows that follow.

## 3.3 Pre-conditions

The pre-conditions of a test case are the required actions that must be taken before the given test case can even be attempted. If any of these conditions are not met, then the test case will not be valid. The reason for this explicit section is so the tester can isolate each test case in great detail. By providing these pre-conditions, they act as assumptions that can be excluded from the variables for the given test case. This allows the test case to focus on a more specific aspect to test. They are also useful so the tester can see which test cases will also fail if they already know any of these pre-conditions are failing.

## 3.4 Test Steps

The test steps are the actions that need to be taken by the tester to properly attempt the test case. They are listed in numerical order, so the tester knows the proper sequence to execute the test. This section must be very detailed, for this is how each test is to be performed by anyone who would like to test them. If the test steps are too vague, the test might yield incorrect results due to varying interpretations of the steps.

## 3.5 Expected Results

The expected results are what the tester should look for after following the test steps. This section must also be very detailed, to leave interpretation out of the tester’s hands. The expected result is the whole point of the test case. The expected result is an important portion of the application that contributes to a successful project as a whole. Since the test cases can vary from being extremely small in scale, to system wide, this result will also vary in scale.

## 3.6 Priority

The priority of the test case refers to the degree of importance that the expected result is. If the test case is a whole system test, than the priority is high. If the test case is a minor feature of the application that is isolated from the rest of the application, than the test case priority may be lower. The priority of the test cases comes down to the client’s requirements, as well as the integration of the specific test case with the rest of the application.

## 3.7 Pass/Fail

The pass/fail criteria is how each test case can be quantitatively judged. This section is what is used to compare the output of the test case to. A statement is written about what a successful test yields (pass), and what a unsuccessful test yields (fail). Both statements must be made for each test case so the tester can accurately report on each test case. This specificity is important for generating consistent, repeatable results from various parties.

# 5. Software Versions

Software features, interface design, and functionality varies from version to version. That is why this section includes the versions of the various software that was used during the original testing of the project. Below are the specific release versions for the three applications needed to run this project, as well as perform the testing in the way that was intended.

## 5.1 MySQL Workbench

Name: MySQL Workbench 8.0

Version: 8.0.20

Edition: Community

## 5.2 PyCharm IDE

Name: PyCharm Professional 2020.1

Version: 2020.1.1

Edition: Professional

## 5.3 Power BI Desktop

Name: Microsoft Power BI Desktop

Version: 2.83.5894.822

Edition: Desktop

# 6. Figures

This section includes various images from the three applications that will be required to use throughout various phases of testing. They are included in this document to provide further detail and understanding of their interfaces as needed. The test cases are as detailed as possible, but these images can contribute to the better understanding of how to execute the test cases. If one comes across a test case with a reference to either PyCharm IDE, MySQL Workbench, or Power BI Desktop and requires further clarification on the test case, this section will be very useful. Many of these figures are very detailed and taken from a full screen capture, so they might be hard to read. However, that is not their purpose in this document. These figures are included to help the tester navigate the interfaces. They are meant to highlight where important functions reside.

## 6.1 MySQL Workbench

A screenshot of a cell phone

Description automatically generated

Connection

*Figure 1. MySQL Workbench Home Page*

A screenshot of a social media post

Description automatically generated

Schema Navigator

Menu Bar

*Figure 2. MySQL Workbench Connection Page*

## 6.2 PyCharm IDE

A screenshot of a computer

Description automatically generated

Run Tool Window

Project Navigator

Command Section

File Navigator

*Figure 3. PyCharm IDE Project Page*

## 6.3 Power BI Desktop

A screenshot of a computer screen

Description automatically generated

Dashboard Navigator

Dashboard

Menu Bar

*Figure 4. Power BI Desktop Report Page*

# 4. Test Cases

## Phase 1: Unit Tests

|  |  |
| --- | --- |
| **ID** | FTC-1 |
| **Items to Test** | Verify MySQL Connection |
| **Pre-Conditions** | 1. MySQL Workbench connection “localhost” has been created |
| **Test Steps** | 1. Open MySQL Workbench  2. Click the connection named “localhost”  3. Enter password “password” when prompted  4. Click the “Server” menu from the menu bar  5. Click “Server Status” from the dropdown menu |
| **Expected Results** | Server status shows “Running” |
| **Priority** | High |
| **Pass/Fail** | Pass: Server status shows “Running”  Fail: Server status does not show as “Running” |

|  |  |
| --- | --- |
| **ID** | FTC-2 |
| **Items to Test** | Database Schema Creation |
| **Pre-Conditions** | 1.A new connection is made in the MySQL database at port: 3306, where the credentials are:  Username: localhost  Password: password  2. “gmfsp\_db” schema is created with the given schema script under GM-Senior-Capstone-Project-SS2020 -> SourceFolder -> SupplementalMaterial -> SQL -> SQL CREATE TABLE SCRIPTS MYSQL -> CREATE\_DATABASE\_TABLES\_MYSQL.sql file |
| **Test Steps** | 1.Establish an active connection in MySQL database by entering following credentials for port 3306:  Username: localhost  Password: password  2. Open “gmfsp\_db” schema  3. Run “SELECT \* FROM dbo\_instrumentmaster” and “SELECT \* FROM dbo\_macroeconmaster” in MySQL Workbench |
| **Expected Results** | 1.The dbo\_instrumentmaster will show 10 financial instruments namely: GM, PFE, SPY, ZPH, CARZ, ^TYX, FCAU, TM, F, HMC  2. The dbo\_macroeconmaster will show 8 macro-economic variables namely: TYX, CPIUC, COVI, FSI, GDP, IR, MI, UR |
| **Priority** | High |
| **Pass/Fail** | Pass: If the above expected results are attained, I.e., the db\_instrumentmaster and dbo\_macroeconmaster output the pre-defined values in them.  Fail: If the above expected results are not attained, I.e., the db\_instrumentmaster and dbo\_macroeconmaster do not output the pre-defined values in them. |

|  |  |
| --- | --- |
| **ID** | FTC-3 |
| **Items to Test** | Database connection established from Python |
| **Pre-Conditions** | 1.MySQL Workbench is open, and the localhost connection is active  2. Database schema code is run, and the tables are created with required columns  3. The tables: dbo\_instrumentmaster and dbo\_macroeconmaster are populated with the pre-defined data |
| **Test Steps** | 1.Toggle the update\_close\_stats Boolean to True  2. Toggle the update\_macro\_stats Boolean to True  3. Check in database for updated values in dbo\_instrumentstatistics and dbo\_macroeconstatistics specifically |
| **Expected Results** | 1.The database table dbo\_instrumentmaster shows the latest close prices referring to the current date for all the 10 instruments  2. The database table dbo\_macroeconmaster shows the latest quarterly data for all macro-economic variables |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the dbo\_instrumentstatistics and dbo\_macroeconstatistics is up to date  Fail: The data shown in the dbo\_instrumentstatistics and dbo\_macroeconstatistics is not up to date |

|  |  |
| --- | --- |
| **ID** | FTC-4 |
| **Items to Test** | Data properly pulled from Quandl |
| **Pre-Conditions** | 1.MySQL Workbench is open, and the localhost connection is active  2. Database schema code is run, and the tables are created with required columns  3. The table dbo\_macroeconmaster is populated with the pre-defined values |
| **Test Steps** | 1.Toggle the update\_close\_stats Boolean to True  2. Toggle the update\_macro\_stats Boolean to True  3. Include the test case at line number #142 in DataMain.py |
| **Expected Results** | 1.Asserted result specified in the test case is shown |
| **Priority** | Normal |
| **Pass/Fail** | Pass: The data corresponding to Quandl API is present in dbo\_macroeconstatistics is present  Fail: The data corresponding to Quandl API is present in dbo\_macroeconstatistics is not present |

|  |  |
| --- | --- |
| **ID** | FTC-5 |
| **Items to Test** | Data properly pulled from Fred |
| **Pre-Conditions** | 1.MySQL Workbench is open, and the localhost connection is active  2. Database schema code is run, and the tables are created with required columns  3. The table dbo\_macroeconmaster is populated with the pre-defined values |
| **Test Steps** | 1.Toggle the update\_close\_stats Boolean to True  2. Toggle the update\_macro\_stats Boolean to True  3. Include the test case at line number #149 in DataMain.py |
| **Expected Results** | 1.Asserted result specified in the test case is shown |
| **Priority** | Normal |
| **Pass/Fail** | Pass: The data corresponding to Fred API is present in dbo\_macroeconstatistics is present  Fail: The data corresponding to Fred API is present in dbo\_macroeconstatistics is not present |

|  |  |
| --- | --- |
| **ID** | FTC-6 |
| **Items to Test** | Macroeconomic data properly pulled from Yahoo |
| **Pre-Conditions** | 1.MySQL Workbench is open, and the localhost connection is active  2. Database schema code is ran, and the tables are created with required columns  3. The table dbo\_macroeconmaster is populated with the pre-defined values |
| **Test Steps** | 1.Toggle the update\_close\_stats Boolean to True  2. Toggle the update\_macro\_stats Boolean to True  3. Include the test case at line number #157 in DataMain.py |
| **Expected Results** | 1.Asserted result specified in the test case is shown |
| **Priority** | Normal |
| **Pass/Fail** | Pass: The data corresponding to Yahoo access source is present in dbo\_macroeconstatistics is present  Fail: The data corresponding to Yahoo access source is present in dbo\_macroeconstatistics is not present |

|  |  |
| --- | --- |
| **ID** | FTC-7 |
| **Items to Test** | Instrument statistics properly pulled from Yahoo! Finance |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_close\_stats Boolean to True in line 33 of DataMain.py.  3. Open DataFetch.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(data) under line 72 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows the values (date, high, low, open, close, adj close, volume) for each stock in the form of a dataframe |
| **Priority** | Normal |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows the values (date, high, low, open, close, adj close, volume) for each stock in the form of a dataframe  Fail: PyCharm Run Tool Window does not show the values (date, high, low, open, close, adj close, volume) for a given stock in the form of a dataframe |

|  |  |
| --- | --- |
| **ID** | FTC-8 |
| **Items to Test** | Instrument statistics load: Validate Instrument statistics are pulled when instruments are added to the SQL schema |
| **Pre-Conditions** | 1.MySQL Workbench is open, and the localhost connection is active. (FTC - 2)  2. Database schema code is ran, and all the tables are stored in MySQL  3. The table dbo\_instrumentstatistics has all the fields such as “open”, “low”, “high”, “close”, “date” |
| **Test Steps** | 1. Add another instrument in line 44 in the gmfsp SQL Schema, for example the user can copy this syntax for adding an insertion in line 44 (11, 'TSLA', 'Equity', 'YAHOO')  2. Run the Schema with the adding instrument in MySQL with the added instrument (FTC-1)  3. Toggle the update\_close\_stats Boolean to True in line 21 of DataMain.py.  4. Run DataMain.py by clicking the green arrow in the command section of PyCharm,  5. Execute the query SELECT \* from dbo\_instrumentstatistics |
| **Expected Results** | dbo\_instrumentstatistics table can extract instrument statistics such as the “open”, “low”, “high”, “close”, “date” for all instruments, including the new instrument that was added ( in this case TSLA instrument statistics needs to populate since that was an instrument we added) |
| **Priority** | Normal |
| **Pass/Fail** | Pass: dbo\_instrumentstatistics table can extract instrument statistics such as the “open”, “low”, “high”, “close”, “date” for all instruments, including the new instrument that was added ( in this case TSLA instrument statistics needs to populate since that was an instrument we added)  Fail: dbo\_instrumentstatistics table can extract instrument statistics such as the “open”, “low”, “high”, “close”, “date” for all instruments, including the new instrument that was added ( in this case TSLA instrument statistics needs to populate since that was an instrument we added) |

|  |  |
| --- | --- |
| **ID** | FTC-9 |
| **Items to Test** | Algorithmforecast load: Validate all algorithms in algorithmforecast can forecast 30-day instrument prediction when instruments are added to the SQL schema |
| **Pre-Conditions** | 1.MySQL Workbench is open, and the localhost connection is active. (FTC - 2)  2. Database schema code is ran, and all the tables are stored in MySQL  3. The table dbo\_instrumentstatistics has all the fields such as “open”, “low”, “high”, “close”, “date”  4. The columns “prederror”, “forecastdate”,”instrumentid”, “forecastcloseprice”, “algorithncode” exist as columns for dbo\_algorithmforecast in MySQL |
| **Test Steps** | 1. Add another instrument in the gmfsp SQL Schema (11, 'TSLA', 'Equity', 'YAHOO'), and execute the schema in MySQL (NFTC-3)  2.Toggle the update\_remaining\_forecasts Boolean to True in DataMain.py.  3. Run DataMain.py by clicking the green arrow in the command section of PyCharm,  4. Execute the query SELECT \* from dbo\_algorithmforecast order by date desc; |
| **Expected Results** | dbo\_algorithmforecast table can generate forecastcloseprice, prederror, and forecastdate for all instruments including the new instrument that was added (in this case TSLA instrument statistics needs to populate since that was an instrument we added) and insert it into dbo\_forecastcloseprice. |
| **Priority** | Normal |
| **Pass/Fail** | Pass: dbo\_algorithmforecast table can generate forecastcloseprice, prederror, and forecastdate for all instruments including the new instrument that was added (in this case TSLA instrument statistics needs to populate since that was an instrument we added) and insert it into dbo\_forecastcloseprice.  Fail: dbo\_algorithmforecast table cannot generate forecastcloseprice, prederror, and forecastdate for all instruments including the new instrument that was added (in this case TSLA instrument statistics needs to populate since that was an instrument we added) and cannot insert it into dbo\_forecastcloseprice. |

|  |  |
| --- | --- |
| **ID** | FTC-10 |
| **Items to Test** | ARS forecast: The first forecasted date for the instrument close prices matches the same date as the first parameter of calculate\_ars\_forecast |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table dbo\_instrumentstatistics is populated (FTC-5)  4. The python project is open in PyCharm  5. The class DataMain.py from the python project is open in PyCharm  6. The python code has an active connection to the database (FTC-2) |
| **Test Steps** | 1. Set the first parameter of the function call calculate\_ars\_forcast to “2020-07-01”  2. Set the second parameter of the function call calculate\_ars\_forecast to “2020-07-08”  3. Set the third parameter of the function call calculate\_ars\_forecast to 7  4. Set the fourth parameter of the function call calculate\_ars\_forecast to False  5. Set the fifth parameter of the function call calculate\_ars\_forecast to True  6. Set the sixth parameter of the function call calculate\_ars\_forecast to False  7. Set the seventh parameter of the function call calculate\_ars\_forecast to False  8. Toggle the update\_ars\_forecast boolean to True  9. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_algorithmforecast has forecasted close prices with the algorithmcode “ARS” for all instruments starting on 2020-07-01  2. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The table dbo\_algorithmforecast has forecasted close prices with the algorithmcode “ARS” for all instruments starting on 2020-07-01  Fail: The table dbo\_algorithmforecast does not have forecasted close prices with the algorithmcode “ARS” for all instruments starting on 2020-07-01 |

|  |  |
| --- | --- |
| **ID** | FTC-11 |
| **Items to Test** | ARS forecast: The last forecasted date for the instrument close prices matches the same date as the second parameter of calculate\_ars\_forecast |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table dbo\_instrumentstatistics is populated (FTC-5)  4. The python project is open in PyCharm  5. The class DataMain.py from the python project is open in PyCharm  6. The python code has an active connection to the database (FTC-2) |
| **Test Steps** | 1. Set the first parameter of the function call calculate\_ars\_forcast to “2020-07-01”  2. Set the second parameter of the function call calculate\_ars\_forecast to “2020-07-08”  3. Set the third parameter of the function call calculate\_ars\_forecast to 7  4. Set the fourth parameter of the function call calculate\_ars\_forecast to False  5. Set the fifth parameter of the function call calculate\_ars\_forecast to True  6. Set the sixth parameter of the function call calculate\_ars\_forecast to False  7. Set the seventh parameter of the function call calculate\_ars\_forecast to False  8. Toggle the update\_ars\_forecast boolean to True  9. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_algorithmforecast has forecasted close prices with the algorithmcode “ARS” for all instruments starting on 2020-07-08  2. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The table dbo\_algorithmforecast has forecasted close prices with the algorithmcode “ARS” for all instruments starting on 2020-07-08  Fail: The table dbo\_algorithmforecast does not have forecasted close prices with the algorithmcode “ARS” for all instruments starting on 2020-07-08 |

|  |  |
| --- | --- |
| **ID** | FTC-12 |
| **Items to Test** | ARS forecast: The amount of historical close price data used for analysis matches the specified amount of days in the third parameter of calculate\_ars\_forecast |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table dbo\_instrumentstatistics is populated (FTC-5)  4. The python project is open in PyCharm  5. The class DataMain.py from the python project is open in PyCharm  6. The python code has an active connection to the database (FTC-2) |
| **Test Steps** | 1. Set the first parameter of the function call calculate\_ars\_forcast to “2020-07-01”  2. Set the second parameter of the function call calculate\_ars\_forecast to “2020-07-08”  3. Set the third parameter of the function call calculate\_ars\_forecast to 7  4. Set the fourth parameter of the function call calculate\_ars\_forecast to False  5. Set the fifth parameter of the function call calculate\_ars\_forecast to True  6. Set the sixth parameter of the function call calculate\_ars\_forecast to False  7. Set the seventh parameter of the function call calculate\_ars\_forecast to True  8. Toggle the update\_ars\_forecast boolean to True  9. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The output in the Run Tool Window in PyCharm, under the section header “\*\*\*\*\* STARTING HISTORICAL DATA \*\*\*\*\*” has seven rows of data  2. The output in the Run Tool Window in PyCharm, under the section header “\*\*\*\*\* STARTING HISTORICAL DATA \*\*\*\*\*” has dates ranging from “2020-06-22” to “2020-06-30”  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The output in the Run Tool Window in PyCharm, under the section header “\*\*\*\*\* STARTING HISTORICAL DATA \*\*\*\*\*” has seven dates, ranging from “2020-06-22” to “2020-06-30”  Fail: The output in the Run Tool Window in PyCharm, under the section header “\*\*\*\*\* STARTING HISTORICAL DATA \*\*\*\*\*” does not have seven dates, ranging from “2020-06-22” to “2020-06-30” |

## Phase 2: Integration Tests

|  |  |
| --- | --- |
| **ID** | FTC-13 |
| **Items to Test** | ARS forecast: The forecasted close prices are inserted into the database when the fifth parameter of calculate\_ars\_forecast is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table dbo\_instrumentstatistics is populated (FTC-5)  4. The python project is open in PyCharm  5. The class DataMain.py from the python project is open in PyCharm  6. The python code has an active connection to the database (FTC-2) |
| **Test Steps** | 1. Set the first parameter of the function call calculate\_ars\_forcast to “2020-07-01”  2. Set the second parameter of the function call calculate\_ars\_forecast to “2020-07-08”  3. Set the third parameter of the function call calculate\_ars\_forecast to 7  4. Set the fourth parameter of the function call calculate\_ars\_forecast to False  5. Set the fifth parameter of the function call calculate\_ars\_forecast to True  6. Set the sixth parameter of the function call calculate\_ars\_forecast to False  7. Set the seventh parameter of the function call calculate\_ars\_forecast to False  8. Toggle the update\_ars\_forecast boolean to True  9. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The forecasted close prices for the range “2020-07-01” to “2020-07-08” for all instruments are in the database table dbo\_algorithmforecast with the algorithmcode “ARS”  2. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The forecasted close prices for the range “2020-07-01” to “2020-07-08” for all instruments are in the database table dbo\_algorithmforecast with the algorithmcode “ARS”  Fail: The forecasted close prices for the range “2020-07-01” to “2020-07-08” for all instruments are not in the database table dbo\_algorithmforecast with the algorithmcode “ARS” |

|  |  |
| --- | --- |
| **ID** | FTC-14 |
| **Items to Test** | ARS forecast: The function inserts the Mean Absolute Percentage Error values for each forecast into the database when the is\_test parameter is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table dbo\_instrumentstatistics is populated (FTC-5)  4. The python project is open in PyCharm  5. The class DataMain.py from the python project is open in PyCharm  6. The python code has an active connection to the database (FTC-2) |
| **Test Steps** | 1. Set the first parameter of the function call calculate\_ars\_forcast to “2020-07-01”  2. Set the second parameter of the function call calculate\_ars\_forecast to “2020-07-08”  3. Set the third parameter of the function call calculate\_ars\_forecast to 7  4. Set the fourth parameter of the function call calculate\_ars\_forecast to False  5. Set the fifth parameter of the function call calculate\_ars\_forecast to True  6. Set the sixth parameter of the function call calculate\_ars\_forecast to True  7. Set the seventh parameter of the function call calculate\_ars\_forecast to False  8. Toggle the update\_ars\_forecast boolean to True  9. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The forecasted close prices for the range “2020-07-01” to “2020-07-08” for all instruments are in the database table dbo\_algorithmforecast with the algorithmcode “ARS” and a non-zero value in the “prederror” field  2. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The forecasted close prices for the range “2020-07-01” to “2020-07-08” for all instruments are in the database table dbo\_algorithmforecast with the algorithmcode “ARS” and a non-zero value in the “prederror” field  Fail: The forecasted close prices for the range “2020-07-01” to “2020-07-08” for all instruments are in the database table dbo\_algorithmforecast with the algorithmcode “ARS” and a value of zero in the “prederror” field |

|  |  |
| --- | --- |
| **ID** | FTC-15 |
| **Items to Test** | ARS forecast: The function shows console output while generating the forecasted values when the show\_output parameter is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table dbo\_instrumentstatistics is populated (FTC-5)  4. The python project is open in PyCharm  5. The class DataMain.py from the python project is open in PyCharm  6. The python code has an active connection to the database (FTC-2) |
| **Test Steps** | 1. Set the first parameter of the function call calculate\_ars\_forcast to “2020-07-01”  2. Set the second parameter of the function call calculate\_ars\_forecast to “2020-07-08”  3. Set the third parameter of the function call calculate\_ars\_forecast to 7  4. Set the fourth parameter of the function call calculate\_ars\_forecast to False  5. Set the fifth parameter of the function call calculate\_ars\_forecast to False  6. Set the sixth parameter of the function call calculate\_ars\_forecast to False  7. Set the seventh parameter of the function call calculate\_ars\_forecast to True  8. Toggle the update\_ars\_forecast boolean to True  9. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. For each instrument, the Run Tool Window in PyCharm shows eight sections, indicated by ten asterisks and all-capitalized letters, with values printed inside the sections  2. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: For each instrument, the Run Tool Window in PyCharm shows eight sections, indicated by ten asterisks and all-capitalized letters, with values printed inside the sections  Fail: For each instrument, the Run Tool Window in PyCharm does not show eight sections, indicated by ten asterisks and all-capitalized letters, with values printed inside the sections |

|  |  |
| --- | --- |
| **ID** | FTC-16 |
| **Items to Test** | FJF Forecast: “FJF” algorithmcode is added to the dbo\_macroeconmaster table in the gmfsp\_db database in MySQL. |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “FJF” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench. |
| **Priority** | High |
| **Pass/Fail** | Pass: “FJF” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench.  Fail: “FJF” code does not show in the “dbo\_algorithmmaster” table in MySQL Workbench. |

|  |  |
| --- | --- |
| **ID** | FTC-17 |
| **Items to Test** | FJF Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “FJF” code is added to dbo\_algorithmmaster table (FTC-12) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated forecast values are shown in dbo\_algorithmforecast table 30 days from the current date without duplicate dates for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated forecast values are shown in dbo\_algorithmforecast 30 days from the current date without duplicate dates for each instrument.  Fail: Duplicate dates are shown in dbo\_algorithmforcast table for any given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-18 |
| **Items to Test** | FJF Forecast: Data retrieval from “dbo\_instrumentstatistics” table in MySQL Workbench. |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(data) statement on line 87 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from dbo\_instrumentstatistics will be shown in the PyCharm Run Tool Window for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-19 |
| **Items to Test** | FJF Forecast: LSTM Training |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “Learning data for [‘stockname’]” message will be shown to the user with a progress bar for each instrument in the PyCharm Run Tool Window. (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: “Learning data for [‘stockname’]” message will be shown to the user with a progress bar for each instrument.  Fail: No message or progress bar is shown to the user for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-20 |
| **Items to Test** | FJF Forecast: Calculate test forecast values |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | 1. Test forecast prices are shown to the user in the PyCharm Run Tool Window. (Refer to Figure 3 of the Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast prices are shown to the user in the PyCharm Run Tool Window.  Fail: Test forecast prices are not shown to the user in the PyCharm Run Tool Window for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-21 |
| **Items to Test** | FJF Forecast: Add test forecasts to “dbo\_algorithmforecast” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “FJF” code is added to dbo\_algorithmmaster table (FTC-12) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast values are added to the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast values are added to the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument.    Fail: Test forecast values are not added to the “dbo\_algorithmforecast” table along with their respective date until the current date for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-22 |
| **Items to Test** | FJF Forecast: Calculate future forecast values |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-23 |
| **Items to Test** | FJF Forecast: Add future forecast values to ‘dbo\_algorithmforecast” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “FJF” code is added to dbo\_algorithmmaster table (FTC-12)  5. Forecast values are calculated by FJF (FTC-18) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Future forecast values are added to “dbo\_algorithmforecast” table along with their respective date 30 days past the current date for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: Future forecast values are added to “dbo\_algorithmforecast” table along with their respective date 30 days past the current date for each instrument.  Fail: Future forecast values are not added to “dbo\_algorithmforecast” table along with their respective date 30 days past the current date for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-24 |
| **Items to Test** | FJF Forecast: Calculate Mean Absolute Percent Error (MAPE) |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. Test forecast values are calculated for FJF (FTC-16) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows the value for MAPE for each instrument by giving message: “The Mean Absolute Percent Error for the model’s predictions to close values for [stockName] is: [MAPE] (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows the value for MAPE for each instrument by giving message: “The Mean Absolute Percent Error for the model’s predictions to close values for [stockName] is: [MAPE]  Fail: PyCharm Run Tool Window does not the value for MAPE for a given instrument when giving message: “The Mean Absolute Percent Error for the model’s predictions to close values for [stockName] is: [MAPE] |

|  |  |
| --- | --- |
| **ID** | FTC-25 |
| **Items to Test** | FJF Forecast: Add MAPE to “dbo\_algorithmforecast” table in MySQL |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “FJF” code is added to dbo\_algorithmmaster table (FTC-12)  5. MAPE is calculated for all instruments (FTC-21) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_fjf\_forecast Boolean to True in line 30 of DataMain.py.  3. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | MAPE values are shown for each test value given by FJF in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: MAPE values are shown for each test value given by FJF in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench  Fail: MAPE values are not shown for one or more test value(s) given by FJF in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-26 |
| **Items to Test** | LR Forecast: Linear Regression predicts stock price data for all 10 stock instruments |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics has all the fields such as “open”, “low”, “high”, “close”, “date” with historical data dating back to 2010  2. The python code has an active connection to the database (FTC-2)  3. calculate\_lr\_forecast retrieves all instrument data from dbo\_instrumentstatistics in MySQL |
| **Test Steps** | 1. Toggle the update\_regression\_forecast Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3 Go on MySQL and run the query - Select (\*) from dbo\_algorithmforecast where algorithmcode = “lr” |
| **Expected Results** | Predictive instrument stock price data such as “forecastcloseprice” for all 10 stock tickers executed by the calculate\_lr\_forecast algorithm is inserted into the dbo\_algorithmforecast table, under the “algorithmcode” denoted as “lr” in MySQL. |
| **Priority** | High |
| **Pass/Fail** | Pass: Predictive instrument stock price data such as “forecastcloseprice” for all 10 stock tickers executed by the calculate\_lr\_forecast algorithm is inserted into the dbo\_algorithmforecast table, under the “algorithmcode” denoted as “lr” in MySQL  Fail: Predictive instrument stock price data such as “forecastcloseprice”, for all 10 stock tickers executed by the calculate\_lr\_forecast algorithm is not inserted into the dbo\_algorithmforecast table, under the “algorithmcode” denoted as “lr” in MySQL |

|  |  |
| --- | --- |
| **ID** | FTC-27 |
| **Items to Test** | LR Forecast: Linear Regression predicts stock prices into the next 30 days |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics has all the fields such as “open”, “low”, “high”, “close”, “date” with historical data dating back to 2010  2. The python code has an active connection to the database (FTC-2)  3. n\_days in calculate\_lr\_forecast in DataForecast.py is set to 23  4. calculate\_lr\_forecast retrieves all instrument data from dbo\_instrumentstatistics in MySQL |
| **Test Steps** | 1. Toggle the update\_regression\_forecast Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3.set n\_days in the lr\_forecast function in DataForecast.py to 23  Go on MySQL and run the query - Select (\*) from dbo\_algorithmforecast where algorithmcode = “lr” where order by forecastdate desc |
| **Expected Results** | The calculate\_lr\_forecast algorithm predicts stock data for a 30-day time period (30 days from the current day) and inserts “forecastcloseprice” in the instrumentforecast table |
| **Priority** | High |
| **Pass/Fail** | Pass: The calculate\_lr\_forecast algorithm predicts stock data for a 30-day time period (30 days from the current day) and inserts “forecastcloseprice” in the instrumentforecast table  Fail: The calculate\_lr\_forecast algorithm predicts stock data for a 30-day time period (30 days from the current day) and does not insert “forecastcloseprice “ in the instrumentforecast table |

|  |  |
| --- | --- |
| **ID** | FTC-28 |
| **Items to Test** | LR Forecast: Linear Regression outputs 30 day forecast in PyCharm |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics has all the fields such as “open”, “low”, “high”, “close”, “date” with historical data dating back to 2010  2. The python code has an active connection to the database (FTC-2)  3. n\_days in calculate\_lr\_forecast in DataForecast.py is set to 23  4. calculate\_lr\_forecast retrieves all instrument data from dbo\_instrumentstatistics in MySQL  5. the print statement print (forecast) exists in calculate\_lr\_forecast |
| **Test Steps** | 1. Toggle the update\_regression\_forecast Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button |
| **Expected Results** | The calculate\_lr\_forecast algorithm predicts stock data for a 30-day time period (30 days from the current day) and and outputs all the data in a visual representation in the PyCharm Console for all 10 instruments. |
| **Priority** | High |
| **Pass/Fail** | Pass: The calculate\_lr\_forecast algorithm predicts stock data for a 30-day time period (30 days from the current day) and outputs all the data in a visual representation in the PyCharm Console for all 10 instruments.  Fail: The calculate\_lr\_forecast algorithm predicts stock data for a 30-day time period (30 days from the current day) and does not output all the data in a visual representation in the PyCharm Console for all 10 instruments. |

|  |  |
| --- | --- |
| **ID** | FTC-29 |
| **Items to Test** | LR Forecast: Prediction error is accurately calculated for linear regression |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics has all the fields such as “open”, “low”, “high”, “close”, “date” with historical data dating back to 2010  2. The python code has an active connection to the database (FTC-2)  3. “Prederror” is a column in the table dbo\_algorithmforecast  4. The prediction error function ( predError = 100 \* abs(forecastClose - data['Close'][n]) / data['Close'][n]) exists inside the lr\_forecast function in DataForecast.py |
| **Test Steps** | 1. Toggle the update\_regression\_forecast Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Use a calculator to calculate the prediction error by using the forecastclose of the date and subtracted it by the closing price of the date and dividing it by the closing price for the date |
| **Expected Results** | The “prederror” calculated by a calculator matches the “prederror” in dbo\_algorithmforecast for all 10 instruments in dbo\_algorithmforecast. |
| **Priority** | High |
| **Pass/Fail** | Pass: The “prederror” calculated by a calculator matches the “prederror” in dbo\_algorithmforecast for all 10 instruments in dbo\_algorithmforecast.  Fail: The “prederror” calculated by a calculator does not match the “prederror” in dbo\_algorithmforecast for all 10 instruments in dbo\_algorithmforecast. |

|  |  |
| --- | --- |
| **ID** | FTC-30 |
| **Items to Test** | LR Forecast: The “algorithmcode” for linear regression is inserted in the suitable column |
| **Pre-Conditions** | 1. The algorithmcode is a field in the table dbo\_algorithmforecast, and dbo\_algorithmforecast is shown as a table in MySQL  2. The algorithmcode is a field in the table dbo\_algorithmmaster, and dbo\_algorithmmaster is shown as a table in MySQL  3. Algorithmcode is a foreign key of dbo\_algorithmforecast that references to dbo\_algorithmmaster in the MySQL schema  4. The python code has an active connection to the database (FTC-2) |
| **Test Steps** | 1. Toggle the update\_regression\_forecast Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Execute the query select distinct(algorithmcode) from dbo\_algorithmforeacast in MySQL  4. Execute the query select distinct(algorithmcode) from dbo\_algorithmmaster in MySQL |
| **Expected Results** | 1. The algorithm code for linear regression denoted as “lr” is inserted into the table dbo\_algorithmforecast in MySQL  2. The algorithm code for linear regression denoted as “lr” is inserted into the table dbo\_algorithmmaster in MySQL  and dbo\_algorithmmaster under the field “algorithmcode”. |
| **Priority** | High |
| **Pass/Fail** | Pass: 1. The algorithm code for calculate\_lr\_forecast denoted as “lr” is inserted into the table dbo\_algorithmforecast in MySQL  2. The algorithm code for calculate\_lr\_forecast denoted as “lr” is inserted into the table dbo\_algorithmmaster in MySQL  Fail: 1. The algorithm code for calculate\_lr\_forecast denoted as “lr” is not inserted into the table dbo\_algorithmforecast in MySQL  2. The algorithm code for calculate\_lr\_forecast denoted as “lr” is not inserted into the table dbo\_algorithmmaster in MySQL |

|  |  |
| --- | --- |
| **ID** | FTC-31 |
| **Items to Test** | PricePred Forecast: Data retrieval from “dbo\_instrumentstatistics” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(data) statement under line 626 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from the “dbo\_instrumentstatistics” table will be shown in the PyCharm Run Tool Window for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-32 |
| **Items to Test** | PricePred Forecast: Data retrieval from “dbo\_engineeredfeatures” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(data) statement under line 626 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “ltrough”, “lpeak”, “lema”, “lcma”, “highfrllinelong”, “medfrllinelong”, and “lowfrllinelong” from the “dbo\_engineeredfeatures” table will be shown in the PyCharm Run Tool Window for each instrument. (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: A dataframe of the existing rows of “ltrough”, “lpeak”, “lema”, “lcma”, “highfrllinelong”, “medfrllinelong”, and “lowfrllinelong” from the “dbo\_engineeredfeatures” table will be shown in the PyCharm Run Tool Window for each instrument  Fail: The data shown in the PyCharm Run Tool Window does not match with the data stored in dbo\_engineeredfeatures for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-33 |
| **Items to Test** | PricePred Forecast: “PricePred” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “PricePred” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: “PricePred” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench  Fail: “PricePred” code does not show in the “dbo\_algorithmmaster” table in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-34 |
| **Items to Test** | PricePred Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6)  5. “PricePred” code is added to dbo\_algorithmmaster table (FTC-30) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated forecast values are shown in “dbo\_algorithmforecast” table 30 days from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument  Fail: Duplicate dates are shown in “dbo\_algorithmforcast” table for any given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-35 |
| **Items to Test** | PricePred Forecast: Calculate test forecast values |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(forecastClose) under line 661 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast prices are shown to the user in the PyCharm Run Tool Window. (Refer to Figure 3 of the Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast prices are shown to the user in the PyCharm Run Tool Window  Fail: Test forecast prices are not shown to the user in the PyCharm Run Tool Window |

|  |  |
| --- | --- |
| **ID** | FTC-36 |
| **Items to Test** | PricePred Forecast: Add test forecasts to “dbo\_algorithmforecast” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6)  5. “PricePred” code is added to dbo\_algorithmmaster table (FTC-30)  6. Test forecasts are calculated for PricePred (FTC-31) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast values are added to the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast values are shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument  Fail: Test forecast values are not shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-37 |
| **Items to Test** | PricePred Forecast: Calculate Mean Absolute Percent Error (MAPE) |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6)  5. Test forecast values are calculated for PricePred (FTC-16) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(predError) under line 661 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows the value for MAPE for each forecast value in the form of a list |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows the value for MAPE for each forecast value in the form of a list  Fail: PyCharm Run Tool Window shows the value for MAPE for one or more forecast value(s) in the form of a list |

|  |  |
| --- | --- |
| **ID** | FTC-38 |
| **Items to Test** | PricePred Forecast: Add MAPE to dbo\_algorithmforecast table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6)  5. “PricePred” code is added to dbo\_algorithmmaster table (FTC-12)  6. MAPE is calculated for all instruments (FTC-34) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | MAPE values are shown for each test value given by PricePred in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: MAPE values are shown for each test value given by PricePred in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench  Fail: MAPE values are not shown for one or more test value(s) given by PricePred in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-39 |
| **Items to Test** | PricePred Forecast: Calculate tomorrow’s forecast value |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(forecastClose) under line 690 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows the value for tomorrow’s forecast for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows the value for tomorrow’s forecast for each instrument  Fail: PyCharm Run Tool Window does not show one or more value(s) for tomorrow’s forecast for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-40 |
| **Items to Test** | PricePred Forecast: Add tomorrow’s forecast value to dbo\_algorithmmaster table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “PricePred” code is added to dbo\_algorithmmaster table (FTC-12)  5. Forecast values are calculated by PricePred (FTC-36) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Tomorrow’s forecast value is added to “dbo\_algorithmforecast” table along with its respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Tomorrow’s forecast value is added to “dbo\_algorithmforecast” table along with its respective date for each instrument  Fail: Tomorrow’s forecast value is not added to “dbo\_algorithmforecast” table along with its respective date for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-41 |
| **Items to Test** | PricePred Forecast: Calculate forecast values for the 9 days after tomorrow |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(forecast[i])” under line 793 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-42 |
| **Items to Test** | PricePred Forecast: Add forecast values for the 9 days after tomorrow to dbo\_algorithmmaster table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “PricePred” code is added to dbo\_algorithmmaster table (FTC-12)  5. Forecast values are calculated by PricePred (FTC-38) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Forecast values for the future 9 days after tomorrow are added to “dbo\_algorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Forecast values for the future 9 days after tomorrow are added to “dbo\_algorithmforecast” table along with their respective date for each instrument  Fail: Forecast values for the future 9 days after tomorrow are not added to “dbo\_algorithmforecast” table along with their respective date for any given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-43 |
| **Items to Test** | ARIMA Forecast: Data retrieval from “dbo\_instrumentstatistics” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(df) statement under line 840 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from the “dbo\_instrumentstatistics” table will be shown in the PyCharm Run Tool Window for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-44 |
| **Items to Test** | ARIMA Forecast: “ARIMA” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “ARIMA” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: “ARIMA” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench  Fail: “ARIMA” code does not show in the “dbo\_algorithmmaster” table in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-45 |
| **Items to Test** | ARIMA Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_engineeredfeatures” is populated with stock data for all instruments. (FTC-6)  5. “ARIMA” code is added to dbo\_algorithmmaster table (FTC-41) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument  Fail: Duplicate dates are shown in “dbo\_algorithmforcast” table for any given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-46 |
| **Items to Test** | ARIMA Forecast: Calculate test forecast values |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(forecastClose) under line 862 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast prices are shown to the user in the PyCharm Run Tool Window (Refer to Figure 3 of the Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast prices are shown to the user in the PyCharm Run Tool Window  Fail: Test forecast prices are not shown to the user in the PyCharm Run Tool Window |

|  |  |
| --- | --- |
| **ID** | FTC-47 |
| **Items to Test** | ARIMA Forecast: Add test forecasts for dbo\_algorithmforecast table |
| **Items to Test** | ARIMA Forecast: Add test forecasts to “dbo\_algorithmforecast” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “ARIMA” code is added to dbo\_algorithmmaster table (FTC-41)  5. Test forecasts are calculated for ARIMA (FTC-43) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast values are added to the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast values are shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument  Fail: Test forecast values are not shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-48 |
| **Items to Test** | ARIMA Forecast: Calculate Mean Absolute Percent Error (MAPE) |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. Test forecast values are calculated for ARIMA (FTC-43) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(predError) under line 862 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows the value for MAPE for each forecast value |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows the value for MAPE for each forecast value  Fail: PyCharm Run Tool Window shows the value for MAPE for one or more forecast value(s) |

|  |  |
| --- | --- |
| **ID** | FTC-49 |
| **Items to Test** | ARIMA Forecast: Add MAPE to dbo\_algorithmforecast table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “ARIMA” code is added to dbo\_algorithmmaster table (FTC-41)  5. MAPE is calculated for all instruments (FTC-45) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | MAPE values are shown for each test value given by ARIMA in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: MAPE values are shown for each test value given by ARIMA in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench  Fail: MAPE values are not shown for one or more test value(s) given by ARIMA in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-50 |
| **Items to Test** | ARIMA Forecast: Calculate forecast values for 10 days into the future |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(forecastClose)” under line 886 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-51 |
| **Items to Test** | ARIMA Forecast: Add forecast values for next 10 days to dbo\_algorithmmaster table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “ARIMA” code is added to dbo\_algorithmmaster table (FTC-41)  5. Forecast values are calculated by ARIMA (FTC-47) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument  Fail: Forecast values for the future 10 days after the current date are not added to “dbo\_algorithmforecast” table along with their respective date for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-52 |
| **Items to Test** | Random Forest Forecast: Data retrieval from “dbo\_instrumentstatistics” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(df) statement under line 934 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from the “dbo\_instrumentstatistics” table will be shown in the PyCharm Run Tool Window for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-53 |
| **Items to Test** | Random Forest Forecast: “RandomForest” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “RandomForest” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: “RandomForest” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench  Fail: “RandomForest” code does not show in the “dbo\_algorithmmaster” table in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-54 |
| **Items to Test** | Random Forest Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “RandomForest” code is added to dbo\_algorithmmaster table (FTC-41) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument  Fail: Duplicate dates are shown in “dbo\_algorithmforcast” table for any given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-55 |
| **Items to Test** | Random Forecast: Calculate test forecast values |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(forecastClose) under line 965 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast prices are shown to the user in the PyCharm Run Tool Window (Refer to Figure 3 of the Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast prices are shown to the user in the PyCharm Run Tool Window  Fail: Test forecast prices are not shown to the user in the PyCharm Run Tool Window |

|  |  |
| --- | --- |
| **ID** | FTC-56 |
| **Items to Test** | Random Forest Forecast: Add test forecasts to “dbo\_algorithmforecast” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “RandomForest” code is added to dbo\_algorithmmaster table (FTC-50)  5. Test forecasts are calculated for Random Forest (FTC-52) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast values are added to the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast values are shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument  Fail: Test forecast values are not shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-57 |
| **Items to Test** | Random Forest Forecast: Calculate Mean Absolute Percent Error (MAPE) |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. Test forecast values are calculated for Random Forest (FTC-52) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(predError) under line 965 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows the value for MAPE for each forecast value |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows the value for MAPE for each forecast value  Fail: PyCharm Run Tool Window shows the value for MAPE for one or more forecast value(s) |

|  |  |
| --- | --- |
| **ID** | FTC-58 |
| **Items to Test** | Random Forest Forecast: Add MAPE to dbo\_algorithmforecast table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “RandomForest” code is added to dbo\_algorithmmaster table (FTC-50)  5. MAPE is calculated for all instruments (FTC-45) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | MAPE values are shown for each test value given by Random Forest in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: MAPE values are shown for each test value given by Random Forest in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench  Fail: MAPE values are not shown for one or more test value(s) given by Random Forest in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-59 |
| **Items to Test** | Random Forest Forecast: Calculate forecast values for 10 days into the future |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(forecastClose)” under line 988 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-60 |
| **Items to Test** | Random Forest Forecast: Add forecast values for next 10 days to dbo\_algorithmmaster table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “Random Forest” code is added to dbo\_algorithmmaster table (FTC-50)  5. Forecast values are calculated by Random Forest (FTC-56) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument  Fail: Forecast values for the future 10 days after the current date are not added to “dbo\_algorithmforecast” table along with their respective date for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-61 |
| **Items to Test** | Random Forest algorithm performance: Random forest forecast can store and predict data before 2019-07-15 effectively |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics has all the fields such as “open”, “low”, “high”, “close”, “date” populated with data for each instrumentid  2. The python code has an active connection to the database (FTC-2)  3. calculate\_random\_forest\_forecast function exists in DataForecast.py |
| **Test Steps** | 1. Toggle the run\_simulator Boolean to True in DataMain.py.  2.Change the parameter of forecast.calculate\_random\_forest\_forecast ("2019-07-15",) to - forecast.calculate\_random\_forest\_forecast ("2010-01-10") in DataMain.py (line#111)  3. Run DataMain.py in PyCharm by clicking on the green "run" button  4. execute the query SELECT \* from dbo\_algorithmforecast  Where algorithmcode = ‘RandomForest’ order by date asc; |
| **Expected Results** | The forecast results for calculate\_random\_forest\_forecast going back to 2010-01-10 should be inserted into dbo\_algorithmforecast for all 10 instrumentid’s even when the date parameter is changed in datamain |
| **Priority** | High |
| **Pass/Fail** | Pass: The forecast results for calculate\_random\_forest\_forecast going back to 2010-01-10 should be inserted into dbo\_algorithmforecast for all 10 instrumentid’s even when the date parameter is changed in datamain  Fail: The forecast results for calculate\_random\_forest\_forecast going back to 2010-01-10 should not be inserted into dbo\_algorithmforecast for all 10 instrumentid’s even when the date parameter is changed in datamain |

|  |  |
| --- | --- |
| **ID** | FTC-62 |
| **Items to Test** | SVM Forecast: Data retrieval from “dbo\_instrumentstatistics” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(data) statement under line 1123 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from the “dbo\_instrumentstatistics” table will be shown in the PyCharm Run Tool Window for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-63 |
| **Items to Test** | SVM Forecast: “SVM” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “SVM” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: “SVM” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench  Fail: “SVM” code does not show in the “dbo\_algorithmmaster” table in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-64 |
| **Items to Test** | SVM Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “SVM” code is added to dbo\_algorithmmaster table (FTC-59) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument  Fail: Duplicate dates are shown in “dbo\_algorithmforcast” table for any given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-65 |
| **Items to Test** | SVM Forecast: Calculate test forecast values |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(forecastClose) under line 1154 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast prices are shown to the user in the PyCharm Run Tool Window (Refer to Figure 3 of the Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast prices are shown to the user in the PyCharm Run Tool Window  Fail: Test forecast prices are not shown to the user in the PyCharm Run Tool Window |

|  |  |
| --- | --- |
| **ID** | FTC-66 |
| **Items to Test** | SVM Forecast: Add test forecasts to “dbo\_algorithmforecast” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “SVM” code is added to dbo\_algorithmmaster table (FTC-59)  5. Test forecasts are calculated for SVM (FTC-61) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast values are added to the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast values are shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument  Fail: Test forecast values are not shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-67 |
| **Items to Test** | SVM Forecast: Calculate Mean Absolute Percent Error (MAPE) |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. Test forecast values are calculated for SVM (FTC-61) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(predError) under line 1153 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows the value for MAPE for each forecast value |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows the value for MAPE for each forecast value  Fail: PyCharm Run Tool Window shows the value for MAPE for one or more forecast value(s) |

|  |  |
| --- | --- |
| **ID** | FTC-68 |
| **Items to Test** | SVM Forecast: Add MAPE to dbo\_algorithmforecast table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “SVM” code is added to dbo\_algorithmmaster table (FTC-59)  5. MAPE is calculated for all instruments (FTC-63) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | MAPE values are shown for each test value given by SVM in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: MAPE values are shown for each test value given by SVM in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench  Fail: MAPE values are not shown for one or more test value(s) given by SVM in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-69 |
| **Items to Test** | SVM Forecast: Calculate forecast values for 10 days into the future |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(forecastClose)” under line 1185 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-70 |
| **Items to Test** | SVM Forecast: Add forecast values for next 10 days to dbo\_algorithmmaster table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “SVM” code is added to dbo\_algorithmmaster table (FTC-50)  5. Forecast values are calculated by SVM (FTC-65) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument  Fail: Forecast values for the future 10 days after the current date are not added to “dbo\_algorithmforecast” table along with their respective date for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-71 |
| **Items to Test** | XGBoost Forecast: Data retrieval from dbo\_instrumentstatistics table in MySQL |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics is populated  2. The python code has an active connection to the database |
| **Test Steps** | 1. Toggle the update\_remaining\_forecasts Boolean to True.  2. Include “print(data)” statement on line 1230 of DataForecast.py.  3. Run DataMain.py |
| **Expected Results** | 1. A dataframe of the existing rows from dbo\_instrumentstatistics will be shown in the Python output for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-72 |
| **Items to Test** | XGBoost Forecast: “XGB” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “SVM” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: “XGB” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench  Fail: “XGB” code does not show in the “dbo\_algorithmmaster” table in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-73 |
| **Items to Test** | XGBoost Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “XGB” code is added to dbo\_algorithmmaster table (FTC-68) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument  Fail: Duplicate dates are shown in “dbo\_algorithmforcast” table for any given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-74 |
| **Items to Test** | XGBoost Forecast: Calculate test forecast values |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(forecastClose) under line 1262 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast prices are shown to the user in the PyCharm Run Tool Window (Refer to Figure 3 of the Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast prices are shown to the user in the PyCharm Run Tool Window  Fail: Test forecast prices are not shown to the user in the PyCharm Run Tool Window |

|  |  |
| --- | --- |
| **ID** | FTC-75 |
| **Items to Test** | XGBoost Forecast: Add test forecasts to “dbo\_algorithmforecast” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “XGB” code is added to dbo\_algorithmmaster table (FTC-68)  5. Test forecasts are calculated for SVM (FTC-70) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast values are added to the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast values are shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument  Fail: Test forecast values are not shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-76 |
| **Items to Test** | XGBoost Forecast: Calculate Mean Absolute Percent Error (MAPE) |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. Test forecast values are calculated for XGBoost (FTC-70) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(predError) under line 1262 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows the value for MAPE for each forecast value |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows the value for MAPE for each forecast value  Fail: PyCharm Run Tool Window shows the value for MAPE for one or more forecast value(s) |

|  |  |
| --- | --- |
| **ID** | FTC-77 |
| **Items to Test** | XGBoost Forecast: Add MAPE to dbo\_algorithmforecast table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “XGB” code is added to dbo\_algorithmmaster table (FTC-68)  5. MAPE is calculated for all instruments (FTC-72) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | MAPE values are shown for each test value given by XGBoost in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: MAPE values are shown for each test value given by XGBoost in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench  Fail: MAPE values are not shown for one or more test value(s) given by XGBoost in the “dbo\_algorithmforecast” table under the “prederror” column in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-78 |
| **Items to Test** | XGBoost Forecast: Calculate forecast values for 10 days into the future |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(forecastClose)” under line 1299 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-79 |
| **Items to Test** | XGBoost Forecast: Add forecast values for next 10 days to dbo\_algorithmmaster table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “XGB” code is added to dbo\_algorithmmaster table (FTC-68)  5. Forecast values are calculated by SVM (FTC-74) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument  Fail: Forecast values for the future 10 days after the current date are not added to “dbo\_algorithmforecast” table along with their respective date for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-80 |
| **Items to Test** | Regression Forecast: Data retrieval from “dbo\_instrumentstatistics” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(data) statement under line 1346 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from the “dbo\_instrumentstatistics” table will be shown in the PyCharm Run Tool Window for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-81 |
| **Items to Test** | Regression Forecast: “Regression” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “Regression” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench |
| **Priority** | High |
| **Pass/Fail** | Pass: “Regression” code shows in the “dbo\_algorithmmaster” table in MySQL Workbench  Fail: “Regression” code does not show in the “dbo\_algorithmmaster” table in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-82 |
| **Items to Test** | Regression Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “Regression” code is added to dbo\_algorithmmaster table (FTC-77) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated forecast values are shown in “dbo\_algorithmforecast” table 10 days from the current date without duplicate dates for each instrument  Fail: Duplicate dates are shown in “dbo\_algorithmforcast” table for any given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-83 |
| **Items to Test** | Regression Forecast: Calculate test forecast values |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 30 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(forecastClose) under line 1394 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast prices are shown to the user in the PyCharm Run Tool Window (Refer to Figure 3 of the Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast prices are shown to the user in the PyCharm Run Tool Window  Fail: Test forecast prices are not shown to the user in the PyCharm Run Tool Window |

|  |  |
| --- | --- |
| **ID** | FTC-84 |
| **Items to Test** | Regression Forecast: Add test forecasts to “dbo\_algorithmforecast” table in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “Regression” code is added to dbo\_algorithmmaster table (FTC-77)  5. Test forecasts are calculated for Regression (FTC-79) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Test forecast values are added to the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Test forecast values are shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument  Fail: Test forecast values are not shown in the “dbo\_algorithmforecast” table along with their respective date until the current date for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-85 |
| **Items to Test** | Regression Forecast: Calculate forecast values for 10 days into the future |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(forecastClose)” under line 1408 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for each instrument |

|  |  |
| --- | --- |
| **ID** | FTC-86 |
| **Items to Test** | Regression Forecast: Add forecast values for next 10 days to dbo\_algorithmmaster table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. “Regression” code is added to dbo\_algorithmmaster table (FTC-77)  5. Forecast values are calculated by Regression (FTC-80) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 26 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Forecast values for the future 10 days after the current date are added to “dbo\_algorithmforecast” table along with their respective date for each instrument  Fail: Forecast values for the future 10 days after the current date are not added to “dbo\_algorithmforecast” table along with their respective date for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-87 |
| **Items to Test** | Regression performance: Polynomial regression can store and predict data before 2019-07-15 effectively |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics has all the fields such as “open”, “low”, “high”, “close”, “date” populated with data for each instrumentid  2. The python code has an active connection to the database (FTC-2)  3. calculate\_regression function exists in DataForecast.py |
| **Test Steps** | 1. Toggle the run\_simulator Boolean to True in DataMain.py.  2.Change the parameter of forecast.calculate\_regression("2019-07-15", 20, 5) to - forecast.calculate\_regression("2010-01-10", 20, 5) in DataMain.py (line#87)  3. Run DataMain.py in PyCharm by clicking on the green "run" button  4. execute the query SELECT \* from dbo\_algorithmforecast  Where algorithmcode = ‘regression’ order by date asc; |
| **Expected Results** | The forecast results for calculate\_regression going back to 2010-01-10 should be inserted into dbo\_algorithmforecast for all 10 instrumentid’s even when the date parameter is changed in datamain |
| **Priority** | High |
| **Pass/Fail** | Pass: The forecast results for calculate\_regression going back to 2010-01-10 should be inserted into dbo\_algorithmforecast for all 10 instrumentid’s even when the date parameter is changed in datamain  Fail: The forecast results for calculate\_regression going back to 2010-01-10 should not be inserted into dbo\_algorithmforecast for all 10 instrumentid’s even when the date parameter is changed in datamain |

|  |  |
| --- | --- |
| **ID** | FTC-88 |
| **Items to Test** | MSF1 Forecast: Data retrieval from dbo\_macroeconstatistics table in MySQL |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_macroeconstatistics” is populated with macroeconomic data for all instruments. (FTC-3) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(df) statement under line 1515 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “statistics” and “date” from the “dbo\_macroeconstatistics” table will be shown in the PyCharm Run Tool Window for each macroeconomic indicator used by MSF1 (All) |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_macroeconstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_macroeconstatisticss for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-89 |
| **Items to Test** | MSF1 Forecast: Data retrieval from dbo\_instrumentstatistics table in MySQL |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(df2) statement under line 1510 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from the “dbo\_instrumentstatistics” table will be shown in the PyCharm Run Tool Window for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-90 |
| **Items to Test** | MSF1 Forecast: “MSF1” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “MSF1” code shows in the “dbo\_algorithmforecast” table in MySQL Workbench alongside its forecasted values under the column “algorithmcode” |
| **Priority** | High |
| **Pass/Fail** | Pass: “MSF1” code shows in the “dbo\_algorithmforecast” table in MySQL Workbench alongside its forecasted values under the column “algorithmcode”  Fail: “MSF1” code does not in the “dbo\_algorithmforecast” table in MySQL Workbench alongside one or more forecasted value(s) under the column “algorithmcode” |

|  |  |
| --- | --- |
| **ID** | FTC-91 |
| **Items to Test** | MSF1 Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-3)  5. “MSF1” code is added to dbo\_algorithmmaster table (FTC-84) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated quarterly forecast values are shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated quarterly forecast values are shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for each instrument  Fail: Updated quarterly forecast values are not shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-92 |
| **Items to Test** | MSF1 Forecast: Calculate quarterly forecast values for 2 years into the future |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-3) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(forecastClose)” under line 1408 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-93 |
| **Items to Test** | MSF1 Forecast: Add quarterly forecast values for next 2 days to dbo\_macroeconalgorithmforecast table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-3)  4. “MSF1” code is added to dbo\_algorithmmaster table (FTC-84)  5. Forecast values are calculated by MSF1 (FTC-86) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Quarterly forecast values for the future 2 years after the current date are added to “dbo\_macroeconalgorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Quarterly forecast values for next 2 years are added to dbo\_macroeconalgorithmforecast table along with their respective date for each instrument.  Fail: Quarterly forecast values for next 2 years are not added to dbo\_macroeconalgorithmforecast table along with their respective date for any given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-94 |
| **Items to Test** | MSF2 Forecast: Data retrieval from dbo\_instrumentstatistics table in MySQL |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(instrumentStats) statement under line 1723 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from the “dbo\_instrumentstatistics” table will be shown in the PyCharm Run Tool Window for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-95 |
| **Items to Test** | MSF2 Forecast: Data retrieval from dbo\_macroeconstatistics table in MySQL |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_macroeconstatistics” is populated with macroeconomic data for all instruments. (FTC-3) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(df2) statement under line 1693 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “statistics” and “date” from the “dbo\_macroeconstatistics” table will be shown in the PyCharm Run Tool Window for each macroeconomic indicator used by MSF2 (GDP, UR, IR, MI) |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_macroeconstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_macroeconstatisticss for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-96 |
| **Items to Test** | MSF2 Forecast: “MSF2” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “MSF2” code shows in the “dbo\_algorithmforecast” table in MySQL Workbench alongside its forecasted values under the column “algorithmcode” |
| **Priority** | High |
| **Pass/Fail** | Pass: “MSF2” code shows in the “dbo\_algorithmforecast” table in MySQL Workbench alongside its forecasted values under the column “algorithmcode”  Fail: “MSF2” code does not in the “dbo\_algorithmforecast” table in MySQL Workbench alongside one or more forecasted value(s) under the column “algorithmcode” |

|  |  |
| --- | --- |
| **ID** | FTC-97 |
| **Items to Test** | MSF2 Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-5)  5. “MSF2” code is added to dbo\_algorithmmaster table (FTC-84) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated quarterly forecast values are shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated quarterly forecast values are shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for each instrument  Fail: Updated quarterly forecast values are not shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-98 |
| **Items to Test** | MSF2 Forecast: Calculate quarterly forecast values for 2 years into the future |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-6) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(table)” under line 1764 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-99 |
| **Items to Test** | MSF2 Forecast: Add quarterly forecast values for next 2 years to dbo\_macroeconalgorithmforecast table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-6)  4. “MSF2” code is added to dbo\_algorithmmaster table (FTC-90)  5. Forecast values are calculated by MSF2 (FTC-92) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Quarterly forecast values for the future 2 years after the current date are added to “dbo\_macroeconalgorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Quarterly forecast values for next 2 years are added to dbo\_macroeconalgorithmforecast table along with their respective date for each instrument.  Fail: Quarterly forecast values for next 2 years are not added to dbo\_macroeconalgorithmforecast table along with their respective date for any given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-100 |
| **Items to Test** | MSF3 Forecast: Data retrieval from dbo\_instrumentstatistics table in MySQL |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(instrumentStats) statement under line 1907 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “close” and “date” from the “dbo\_instrumentstatistics” table will be shown in the PyCharm Run Tool Window for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-101 |
| **Items to Test** | MSF3 Forecast: Data retrieval from dbo\_macroeconstatistics table in MySQL |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_macroeconstatistics” is populated with macroeconomic data for all instruments. (FTC-6) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include print(data) statement under line 1877 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | A dataframe of the existing rows of “statistics” and “date” from the “dbo\_macroeconstatistics” table will be shown in the PyCharm Run Tool Window for each macroeconomic indicator used by MSF3 (GDP, COVI, CPIUC, FSI) |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_macroeconstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_macroeconstatisticss for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-102 |
| **Items to Test** | MSF3 Forecast: “MSF3” algorithmcode is added to the “dbo\_algorithmmaster” table in the gmfsp\_db database in MySQL Workbench |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | “MSF3” code shows in the “dbo\_algorithmforecast” table in MySQL Workbench alongside its forecasted values under the column “algorithmcode” |
| **Priority** | High |
| **Pass/Fail** | Pass: “MSF3” code shows in the “dbo\_algorithmforecast” table in MySQL Workbench alongside its forecasted values under the column “algorithmcode”  Fail: “MSF3” code does not in the “dbo\_algorithmforecast” table in MySQL Workbench alongside one or more forecasted value(s) under the column “algorithmcode” |

|  |  |
| --- | --- |
| **ID** | FTC-103 |
| **Items to Test** | MSF3 Forecast: Repopulate forecast values to prevent duplicates |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-3)  5. “MSF3” code is added to dbo\_algorithmmaster table (FTC-96) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Updated quarterly forecast values are shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Updated quarterly forecast values are shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for each instrument  Fail: Updated quarterly forecast values are not shown in “dbo\_algorithmforecast” table 2 years from the current date without duplicate dates for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-104 |
| **Items to Test** | MSF3 Forecast: Calculate quarterly forecast values for 2 years into the future |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-3) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_remaining\_forecasts Boolean to True in line 24 of DataMain.py.  3. Open DataForecast.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataForecast.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  4. Include “print(table)” under line 1948 of DataForecast.py.  5. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | PyCharm Run Tool Window shows future forecast prices for each instrument (Refer to Figure 3 of Figures Section) |
| **Priority** | High |
| **Pass/Fail** | Pass: PyCharm Run Tool Window shows future forecast prices for each instrument  Fail: PyCharm Run Tool Window does not show future forecast prices for a given instrument |

|  |  |
| --- | --- |
| **ID** | FTC-105 |
| **Items to Test** | MSF3 Forecast: Add quarterly forecast values for next 2 years to dbo\_macroeconalgorithmforecast table |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The table “dbo\_instrumentstatistics” is populated with stock data for all instruments. (FTC-5)  4. The table “dbo\_macroeconstatistics” is populated with stock data for all instruments. (FTC-3)  4. “MSF3” code is added to dbo\_algorithmmaster table (FTC-96)  5. Forecast values are calculated by MSF1 (FTC-98) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the update\_msf\_forecast Boolean to True in line 24 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Quarterly forecast values for the future 2 years after the current date are added to “dbo\_macroeconalgorithmforecast” table along with their respective date for each instrument |
| **Priority** | High |
| **Pass/Fail** | Pass: Quarterly forecast values for next 2 years are added to dbo\_macroeconalgorithmforecast table along with their respective date for each instrument.  Fail: Quarterly forecast values for next 2 years are not added to dbo\_macroeconalgorithmforecast table along with their respective date for any given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-106 |
| **Items to Test** | MSF\_final algorithm: Data retrieval from dbo\_macroeconstatistics table in MySQL |
| **Pre-Conditions** | 1. The table dbo\_macroeconstatistics is populated  2. The python code has an active connection to the database |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Include the “DataForecast.MSF\_final(db\_engine)” statement at line #66 of DataMain.py  3. Include “print(data)” statement under line 1963 of DataForecast.py.  4. Run DataMain.py |
| **Expected Results** | 1. A dataframe of the existing rows from dbo\_macroeconstatistics will be shown in the Python output for each macroeconomic indicator used by MSF\_final (GDP, UR, IR, MI, COVI). |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_ macroeconstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_macroeconstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-107 |
| **Items to Test** | MSF\_final algorithm: Data retrieval from dbo\_instrumentstatistics table in MySQL |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics is populated  2. The python code has an active connection to the database |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Include the “DataForecast.MSF\_final(db\_engine)” statement at line #66 of DataMain.py  3. Include “print(data1)” statement on line 1954 of DataForecast.py.  4. Run DataMain.py |
| **Expected Results** | 1. A dataframe of the existing rows for date, close, and instrument ID from dbo\_instrumentstatistics will be shown in the Python output for each instrument. |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentstatistics for each instrument.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentstatistics for a given instrument. |

|  |  |
| --- | --- |
| **ID** | FTC-108 |
| **Items to Test** | MSF\_final algorithm: Cross check the instrumnets if pulled correctly |
| **Pre-Conditions** | 1. The table dbo\_instrumentstatistics is populated  2. The python code has an active connection to the database |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Include the “DataForecast.MSF\_final(db\_engine)” statement at line #66 of DataMain.py  3. Include the print statements from line 1940 to 1949 of DataForecast.py.  4. Run DataMain.py |
| **Expected Results** | A list of 10 currently used instruments will be showed in the Python console, which can simultaneously be checked with the actual dbo\_instrumentmaster data |
| **Priority** | Low |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data stored in dbo\_instrumentmaster.  Fail: The data shown in the Python output does not match with the data stored in dbo\_instrumentmaster. |

|  |  |
| --- | --- |
| **ID** | FTC-109 |
| **Items to Test** | MSF\_final algorithm: Retrieving the macro variables data used for calculations |
| **Pre-Conditions** | 1. The python code has an active connection to the database  2. FTC-100 is satisfied |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Include “print(temp)” statement on line 2066 of DataForecast.py.  4. Run DataMain.py |
| **Expected Results** | A dataframe consisting of the most recent (n+1) rows of data from the dbo\_macroeconstatistics is printed in the Python console |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the (n+1) number of rows calculation, which can be (12+1).  Fail: The data shown in the Python output does not match with the (n+1) number of rows calculation. |

|  |  |
| --- | --- |
| **ID** | FTC-110 |
| **Items to Test** | MSF\_final algorithm: Retrieving the macro variables data used for calculations |
| **Pre-Conditions** | 1. The python code has an active connection to the database  2. FTC-100 is satisfied |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Include “print(data)” statement on line 2069 of DataForecast.py.  4. Run DataMain.py |
| **Expected Results** | A dataframe consisting of the most recent (n) rows of data from the dbo\_macroeconstatistics is printed in the Python console |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the (n) number of rows calculation, which can be (12).  Fail: The data shown in the Python output does not match with the (n) number of rows calculation. |

|  |  |
| --- | --- |
| **ID** | FTC-111 |
| **Items to Test** | MSF\_final algorithm: Verifying the quarterly data pulled |
| **Pre-Conditions** | 1. The python code has an active connection to the database  2. FTC-101 is satisfied |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Include “print(instrumentStats)” statement on line 2085 of DataForecast.py.  4. Run DataMain.py |
| **Expected Results** | A dataframe consisting of the quarterly data of instrument statistics from the year 2012 will be printed the Python console. This data is reteived from dbo\_instrumentstatistics |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the Python output matches with the data in dbo\_instrumentstatistics verified in MySQL Workbench  Fail: The data shown in the Python output does not match with the data in dbo\_instrumentstatistics verified in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-112 |
| **Items to Test** | MSF\_final time frame: Check for proper past data time frame used |
| **Pre-Conditions** | 1. The python code has an active connection to the database |
| **Test Steps** | 1.Check the DataForecast.py file for lines # 2153, 2155, and 2157  2. Toggle the update\_msf\_forecast Boolean to True.  3. Run DataMain.py  4. Check output in the database table dbo\_macroeconalgorithmforecast |
| **Expected Results** | The python lines of code mentioned above should be between the proper 5 year time frame. And the data in dbo\_macroeconalgorithmforecast should reflect predictions accordingly |
| **Priority** | High |
| **Pass/Fail** | Pass: The date time frame shown in the Python lines of code matches with the data in dbo\_macroeconalgorithmforecast verified in MySQL Workbench  Fail: The date time frame shown in the Python lines of code does not match with the data in dbo\_macroeconalgorithmforecast verified in MySQL Workbench |

|  |  |
| --- | --- |
| **ID** | FTC-113 |
| **Items to Test** | MSF\_final algorithm: Verify the actual calculated forecast array |
| **Pre-Conditions** | 1. The python code has an active connection to the database  2. FTC-100 is satisfied  3. FTC-101 is satisfied |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Run DataMain.py  3. Include the “print(calculated\_forecast)” statement in line #2213 in DataForecast.py |
| **Expected Results** | The python console output shows an array of calculated forecast results that are given out quarterly for the next 1 year |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the python console matches with the data stored in dbo\_macroeconalgorithmforecast after the entire program is run  Fail: The data shown in the python console does not match with the data stored in dbo\_macroeconalgorithmforecast after the entire program is run |

|  |  |
| --- | --- |
| **ID** | FTC-114 |
| **Items to Test** | MSF\_final error calculation: Verify Average Percent Error |
| **Pre-Conditions** | 1. The python code has an active connection to the database  2. FTC-100 is satisfied  3. FTC-101 is satisfied |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Include the “DataForecast.MSF\_final(db\_engine)” statement at line #66 of DataMain.py  3. Run DataMain.py |
| **Expected Results** | The python console output shows a list of print statements showing error percentages for each of the 10 instruments individually |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the python console matches with the data stored in dbo\_paststatistics after the entire program is run  Fail: The data shown in the python console does not match with the data stored in dbo\_paststatistics after the entire program is run |

|  |  |
| --- | --- |
| **ID** | FTC-115 |
| **Items to Test** | MSF\_final accuracy calculation: Verify Trend accuracy percentages |
| **Pre-Conditions** | 1. The python code has an active connection to the database  2. FTC-100 is satisfied  3. FTC-101 is satisfied |
| **Test Steps** | 1. Toggle the update\_msf\_forecast Boolean to True.  2. Include the “DataForecast.MSF\_final(db\_engine)” statement at line #66 of DataMain.py  3. Include the print statement present in line#2336 of DataForecast.py  3. Run DataMain.py |
| **Expected Results** | The python console output shows a list of trend accuracy alongside each of the 10 instruments separately |
| **Priority** | High |
| **Pass/Fail** | Pass: The data shown in the python console matches with the data stored in dbo\_paststatistics after the entire program is run  Fail: The data shown in the python console does not match with the data stored in dbo\_paststatistics after the entire program is run |

|  |  |
| --- | --- |
| **ID** | FTC-116 |
| **Items to Test** | Buy Sell Signals: Validate Right Buy Sell signals show in the database |
| **Pre-Conditions** | 1. The table dbo\_actionsignals is populated with the “date”, “instrumentid”, strategy code” and “signal  2. BuySell algorithms for cma, frl, , ema , macd, , algo\_signal, are stored in BuySell.py in PyCharm  3. The python code has an active connection to the database (FTC–2) |
| **Test Steps** | 1. Toggle the update\_signal Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Execute the query select signal from dbo\_actionsignals |
| **Expected Results** | The Buy sell signals are denoted as “signal” in dbo\_actionsignals only show the values -1 (buy), 0 (hold), and 1 (sell) for each instrumentid in dbo\_actionsignals in MySQL |
| **Priority** | High |
| **Pass/Fail** | Pass: The Buy sell signals are denoted as “signal” in dbo\_actionsignals only show the values -1 (buy), 0 (hold), and 1 (sell) for each instrumentid in dbo\_actionsignals in MySQL  Fail: The Buy sell signals are denoted as “signal” in dbo\_actionsignals does not show the values -1 (buy), 0 (hold), and 1 (sell) for each instrumentid in dbo\_actionsignals in MySQL |

|  |  |
| --- | --- |
| **ID** | FTC-117 |
| **Items to Test** | Buy Sell Signals: Buy Sell “StrategyCode” is inserted in dbo\_strategymaster |
| **Pre-Conditions** | 1. The table dbo\_actionsignals is populated with the “date”, “instrumentid”, strategy code” and “signal  2. BuySell algorithms for cma, frl, , ema , macd, , algo\_signal, are stored in BuySell.py in PyCharm  3. The python code has an active connection to the database (FTC -2) |
| **Test Steps** | 1. Toggle the update\_signal Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Execute the query select strategycode from dbo\_actionsignals |
| **Expected Results** | The “strategyCode“ such as: FRL, CMA, EMA, MACF. BuySell, that exists for each BuySell algorithm in BuySell.py is inserted in to the dbo\_strategymaster table and denoted as the column “StrategyCode” |
| **Priority** | High |
| **Pass/Fail** | Pass: The “strategyCode“ such as: FRL, CMA, EMA, MACF. BuySell, that exists for each BuySell algorithm in BuySell.py is inserted in to the dbo\_strategymaster table and denoted as the column “StrategyCode”  Fail: The “strategyCode“ such as: FRL, CMA, EMA, MACF. BuySell, that exists for each BuySell algorithm in BuySell.py is inserted in to the dbo\_strategymaster table and denoted as the column “StrategyCode” |

|  |  |
| --- | --- |
| **ID** | FTC-118 |
| **Items to Test** | Buy Sell Signals: Buy Sell does not do any future price predictions |
| **Pre-Conditions** | 1. The table dbo\_actionsignals is populated with the “date”, “instrumentid”, strategy code” and “signal  2. BuySell algorithms for cma, frl, , ema , macd, , algo\_signal, are stored in BuySell.py in PyCharm  3. The python code has an active connection to the database (FTC -2) |
| **Test Steps** | 1. Toggle the update\_signal Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Execute the query select \* from dbo\_actionsignals order by date desc; |
| **Expected Results** | No future dates under the field “date” is shown in dbo\_actionsignals when the above query is executed. The Buy Sell signals does not calculate predictions/insert future stock predictions in dbo\_actionsignals as they are economic indicators not predicitive models |
| **Priority** | High |
| **Pass/Fail** | Pass: No future dates under the field “date” is shown in dbo\_actionsignals when the above query is executed. The Buy Sell signals does not calculate predictions/insert future stock predictions in dbo\_actionsignals as they are economic indicators not predicitive models  Fail: Future dates under the field “date” is shown in dbo\_actionsignals when the above query is executed. The Buy Sell signals does not calculate predictions/insert future stock predictions in dbo\_actionsignals as they are economic indicators not predicitive models |

|  |  |
| --- | --- |
| **ID** | FTC-119 |
| **Items to Test** | Enginreered features: All “EngineeredFeatures” data inserted in to dbo\_engineeredfeatures |
| **Pre-Conditions** | 1. The table dbo\_engineeredfeatures is populated with all the columns when the gmfsp\_db schema is executed (FTC-1)  2. The file EngineeredFeatures.Py exists in PyCharm with all the engineered feature 100% similar to engineeredfeatures the SQL schema  3. The python code has an active connection to the database (FTC -2) |
| **Test Steps** | 1. Toggle the update\_engineered\_forecasts Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Execute the query select \* from dbo\_engineeredfeatures |
| **Expected Results** | All “engineered feature” data in the data.to\_sql function (line #90) in EngineeredFeatures.py is insrterted in to dbo\_engineeredfeatures. |
| **Priority** | High |
| **Pass/Fail** | Pass: All “engineered feature” data in the data.to\_sql function (line #90) in EngineeredFeatures.py is insrterted in to dbo\_engineeredfeatures.  Fail: All “engineered feature” data in the data.to\_sql function (line #90) in EngineeredFeatures.py is not insrterted in to dbo\_engineeredfeatures |

|  |  |
| --- | --- |
| **ID** | FTC-120 |
| **Items to Test** | Statistical Returns: portfoliovalue starting price |
| **Pre-Conditions** | 1. The table dbo\_statisticalteturns is populated with fields such as “date”, “instrumentid” , “ strategycode” , “positionsize” , “cashonhand “, and “porttfoliovalue”  2. the value “cash” in TradingSimulator.py on line 46 is set to 10000  3. The python code has an active connection to the database (FTC -2) |
| **Test Steps** | 1. Toggle the run\_simulator Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Check that the starting value of the “portfoliovalue” field is starting off at 10000 for each instrument in mySQL by using the syntax SELECT\* FROM gmfsp\_db.dbo\_statisticalreturns where instrument id = ({insert each instrument id 1 by 1 in here} order by date asc) |
| **Expected Results** | The “portfoliovalue” in MySQL starts off with 10000 for each instrument id that is in the dbo\_statisticalreturn |
| **Priority** | High |
| **Pass/Fail** | Pass: The “portfoliovalue” in MySQL starts off with 10000 for each instrument id that is in the dbo\_statisticalreturn table in the database  Fail: The “portfoliovalue” in MySQL oes not start off with 10000 for each instrument id that is in the dbo\_statisticalreturn table in the database |

|  |  |
| --- | --- |
| **ID** | FTC-121 |
| **Items to Test** | Statistical Returns: portfoliovalue correct fluctuation |
| **Pre-Conditions** | 1. The table dbo\_statisticalteturns is populated with fields such as “date”, “instrumentid” , “ strategycode” , “positionsize” , “cashonhand “, and “porttfoliovalue”  2. the value “cash” in TradingSimulator.py on line 46 is set to 10000  3. The python code has an active connection to the database (FTC -2) |
| **Test Steps** | 1. Toggle the run\_simulator Boolean to True in DataMain.py  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Check that the starting value of the “portfoliovalue” field is fluctuating correctly in relation to the instrument closing price, for each instrumentid in dbo\_statisticalreturns by executing the query SELECT\* FROM gmfsp\_db.dbo\_statisticalreturns |
| **Expected Results** | The “portfoliovalue” fluctuates\changes correctly in relation to the instrument closing price with the values in dbo\_statisticalreturns, (so if a stock is valued at 25 dollars now and it was valued at 36 dollars 2 years ago, the porfolio value now would be less than it was 2 years ago) |
| **Priority** | High |
| **Pass/Fail** | Pass: The “portfoliovalue” fluctuates correctly in relation to the instrument closing price with the values in dbo\_statisticalreturns, (so if a stock is valued at 25 dollars now and it was valued at 36 dollars 2 years ago, the porfolio value now would be less than it was 2 years ago)  Fail: The “portfoliovalue” does not fluctuate correctly in relation to the instrument closing price with the values in dbo\_statisticalreturns, (so if a stock is valued at 25 dollars now and it was valued at 36 dollars 2 years ago, the porfolio value will fail if the portfoliovalue is higher now than 2 years ago) |

|  |  |
| --- | --- |
| **ID** | FTC-122 |
| **Items to Test** | Statistical Returns: “StrategyCode” insertion |
| **Pre-Conditions** | 1. The table dbo\_statisticalteturns is populated with fields such as “date”, “instrumentid” , “ strategycode” , “positionsize” , “cashonhand “, and “porttfoliovalue”  2. The python code has an active connection to the database (FTC -2)  3. “StrategyCode” is a foreign key in dbo\_statisticalreturns that references dbo\_strategymaster(strategycode) |
| **Test Steps** | 1. Toggle the run\_simulator Boolean to True in DataMain.py.  2. Run DataMain.py in PyCharm by clicking on the green "run" button  3. Execute the query SELECT strategycode, instrumentid from gmfsp.dbo\_statisticalreturns |
| **Expected Results** | The “strategycode” such as “ FRL, COMB, MACD, EMA, CMA, MACD” is inserted in dbo\_statisticalreturns as the field “strategycode” for all 10 instrumentid’s in MySQL |
| **Priority** | High |
| **Pass/Fail** | Pass: 1. The “strategycode” such as “ FRL, COMB, MACD, EMA, CMA, MACD” is inserted in dbo\_statisticalreturns as the field “strategycode” for all 10 instrumentid’s in MySQL  Fail: 1. The “strategycode” such as “ FRL, COMB, MACD, EMA, CMA, MACD” is not inserted in dbo\_statisticalreturns as the field “strategycode” for all 10 instrumentid’s in MySQL |

|  |  |
| --- | --- |
| **ID** | FTC-123 |
| **Items to Test** | The database is populated with a calendar when reset\_date\_dim is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2) |
| **Test Steps** | 1. Toggle the reset\_date\_dim boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_datedim has been populated with records  2. The Run Tool Window in PyCharm shows the statement "Populating The Date Dimension..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The table dbo\_datedim has been populated with records  Fail: The table dbo\_datedim has not been populated with records |

|  |  |
| --- | --- |
| **ID** | FTC-124 |
| **Items to Test** | The most recent closing price data is inserted into the database when the update\_close\_stats is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2)  6. The table dbo\_datedim has been populated with dates (FTC-117) |
| **Test Steps** | 1. Toggle the update\_close\_stats boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_instrumentstatistics has the most recent ten years worth of closing prices for the instruments listed in the table dbo\_instrumentmaster  2. The Run Tool Window in PyCharm shows the statement "Getting Instrument Close Prices..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The table dbo\_instrumentstatistics has the most recent ten years worth of closing prices for the instruments listed in the table dbo\_instrumentmaster  Fail: The table dbo\_instrumentstatistics does not have the most recent ten years worth of closing prices for the instruments listed in the table dbo\_instrumentmaster |

|  |  |
| --- | --- |
| **ID** | FTC-125 |
| **Items to Test** | The most recent macroeconomic indicator data is inserted into the database when update\_close\_stats is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2)  6. The table dbo\_datedim has been populated with dates (FTC-117) |
| **Test Steps** | 1. Toggle the update\_macro\_stats boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_macroeconstatistics has up-to-date macroeconomic indicator data for all of the indicators listed in the table dbo\_macroeconmaster  2. The Run Tool Window in PyCharm shows the statement "Getting Macroeconomic Indicator Statistics..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The table dbo\_macroeconstatistics has up-to-date macroeconomic indicator data for all of the indicators listed in the table dbo\_macroeconmaster  Pass: The table dbo\_macroeconstatistics does not have up-to-date macroeconomic indicator data for all of the indicators listed in the table dbo\_macroeconmaster |

|  |  |
| --- | --- |
| **ID** | FTC-126 |
| **Items to Test** | The MSF forecasts are inserted into the database when update\_msf\_forecast is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2)  6. The table dbo\_instrumentstatistics has been populated (FTC-118)  7. The table dbo\_macroeconstatistics has been populated (FTC-119) |
| **Test Steps** | 1. Toggle the update\_msf\_forecast boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_macroeconalgorithmforecast is populated with forecasts for the first six instruments in dbo\_algorithmmaster  2. The Run Tool Window in PyCharm shows the statement "Generating MSF Forecasts..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | Medium |
| **Pass/Fail** | Pass: The table dbo\_macroeconalgorithmforecast is populated with forecasts for the first six instruments in dbo\_algorithmmaster  Fail: The table dbo\_macroeconalgorithmforecast is not populated with forecasts for the first six instruments in dbo\_algorithmmaster |

|  |  |
| --- | --- |
| **ID** | FTC-127 |
| **Items to Test** | The engineered close price data is inserted into the database when update\_engineered\_features is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2)  6. The table dbo\_instrumentstatistics has been populated (FTC-118) |
| **Test Steps** | 1. Toggle the update\_engineered\_features boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_engineeredfeatures is populated with engineered data for all of the instruments listed in dbo\_instrumentmaster  2. The Run Tool Window in PyCharm shows the statement "Calculating Engineered Features..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | Medium |
| **Pass/Fail** | Pass: The table dbo\_engineeredfeatures is populated with engineered data for all of the instruments listed in dbo\_instrumentmaster  Fail: The table dbo\_engineeredfeatures is not populated with engineered data for all of the instruments listed in dbo\_instrumentmaster |

|  |  |
| --- | --- |
| **ID** | FTC-128 |
| **Items to Test** | The forecasted close prices are inserted into the database when update\_remaining\_forecasts is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2)  6. The table dbo\_instrumentstatistics has been populated (FTC-118) |
| **Test Steps** | 1. Toggle the update\_remaining\_forecasts boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_algorithmforecast is populated with forecasted close prices using all of the algorithms listed in dbo\_algorithmmaster  2. The Run Tool Window in PyCharm shows the statement "Calculating..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | Medium |
| **Pass/Fail** | Pass: The table dbo\_algorithmforecast is populated with forecasted close prices using all of the algorithms listed in dbo\_algorithmmaster  Fail: The table dbo\_algorithmforecast is not populated with forecasted close prices using all of the algorithms listed in dbo\_algorithmmaster |

|  |  |
| --- | --- |
| **ID** | FTC-129 |
| **Items to Test** | The buy/sell signals are inserted into the database when update\_signals is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2)  6. The table dbo\_engineeredfeatures has been populated (FTC-121) |
| **Test Steps** | 1. Toggle the update\_signals boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_actionsignals is populated with buy/sell/hold signals for all of the instruments in dbo\_instrumentmaster  2. The Run Tool Window in PyCharm shows the statement "Generating Buy/Sell Signals..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The table dbo\_actionsignals is populated with buy/sell/hold signals for all of the instruments in dbo\_instrumentmaster  Fail: The table dbo\_actionsignals is not populated with buy/sell/hold signals for all of the instruments in dbo\_instrumentmaster |

|  |  |
| --- | --- |
| **ID** | FTC-130 |
| **Items to Test** | The investment simulation is inserted into the database when run\_simulation is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2)  6. The table dbo\_engineeredfeatures has been populated (FTC-121) |
| **Test Steps** | 1. Toggle the run\_simulator boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_statisticalreturns is populated with portfolio values for all of the instruments in dbo\_instrumentmaster  2. The Run Tool Window in PyCharm shows the statement "Running Investment Simulator..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The table dbo\_statisticalreturns is populated with portfolio values for all of the instruments in dbo\_instrumentmaster  Fail: The table dbo\_statisticalreturns is not populated with portfolio values for all of the instruments in dbo\_instrumentmaster |

|  |  |
| --- | --- |
| **ID** | FTC-131 |
| **Items to Test** | The ARS forecasts are inserted into the database when update\_ars\_forecast is set to True |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. The python project is open in PyCharm  4. The class DataMain.py from the python project is open in PyCharm  5. The python code has an active connection to the database (FTC-2)  6. The table dbo\_instrumentstatistics has been populated (FTC-118) |
| **Test Steps** | 1. Toggle the update\_ars\_forecast boolean to True  2. Click the “Run” icon from the command section of PyCharm |
| **Expected Results** | 1. The table dbo\_algorithmforecast is populated with forecasted close prices for all of the instruments listed in dbo\_instrumentmaster using for the ARS forecasting algorithm  2. The Run Tool Window in PyCharm shows the statement "Calculating ARS..."  3. The Run Tool Window in PyCharm shows the statement “Process finished with exit code 0” |
| **Priority** | High |
| **Pass/Fail** | Pass: The table dbo\_algorithmforecast is populated with forecasted close prices for all of the instruments listed in dbo\_instrumentmaster using for the ARS forecasting algorithm  Fail: The table dbo\_algorithmforecast is not populated with forecasted close prices for all of the instruments listed in dbo\_instrumentmaster using for the ARS forecasting algorithm |

## Phase 3: System Tests

|  |  |
| --- | --- |
| **ID** | FTC-132 |
| **Items to Test** | Power BI connects to the database |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Launch Power BI Desktop  2. Select “Get Data” from the Data group on the ribbon  3. Type “MySQL database” in the search bar  4. Select “MySQL database” from the list  5. Select “Connect”  6. Type in the database credentials  - Server: localhost  - Database: gmfsp\_db  - User name: root  - Password: password |
| **Expected Results** | Power BI produces a dialog box named “Navigator” with all of the tables from the database showing |
| **Priority** | High |
| **Pass/Fail** | Pass: Power BI produces a dialog box named “Navigator” with all of the tables from the database showing  Fail: Power BI produces a dialog box with the message “Unable to connect” as the message |

|  |  |
| --- | --- |
| **ID** | FTC-133 |
| **Items to Test** | A Power BI template file populates the dashboards |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2)  3. Power BI is connected to the database (FTC-126) |
| **Test Steps** | 1. Launch Power BI Desktop  2. Select the File tab  3. Select Import  4. Select Power BI Template  5. Open William\_powerbi\_dashboards\_7-19.pbit |
| **Expected Results** | Power BI produces a report with nine dashboards that have data being visualized with the most recent date showing |
| **Priority** | High |
| **Pass/Fail** | Pass: Power BI produces a report with nine dashboards that have data being visualized with the most recent data showing  Fail: Power BI produces a report, but the nine dashboards of the report do not have visualized data and/or do not have the most recent date showing |

## Nonfunctional Test Cases:

|  |  |
| --- | --- |
| **ID** | NTC-1 |
| **Items to Test** | Date dimension population run time |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the reset\_date\_dim Boolean to True in line 30 of DataMain.py.  3. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | Dbo\_datedim table in MySQL populates within 5 minutes. |
| **Priority** | High |
| **Pass/Fail** | Pass: Dbo\_datedim table in MySQL populates within 5 minutes.  Fail: Dbo\_datedim table in MySQL populates in over 5 minutes. |

|  |  |
| --- | --- |
| **ID** | NTC-2 |
| **Items to Test** | Initial GM Fintech Application run time |
| **Pre-Conditions** | 1. MySQL Workbench is running the “localhost” connection. (FTC-1)  2. “gmfsp\_db” schema is created. (FTC-2) |
| **Test Steps** | 1. Open DataMain.py by double clicking the source file under the path (GM-Senior-Capstone-Project > Source Folder > Python Project Folder > DataMain.py) in the Project Navigator in PyCharm. (Refer to Figure 3 of Figures Section)  2. Toggle the reset\_date\_dim Boolean to False in line 22 of DataMain.py.  3. Toggle all remaining Booleans to True in between lines 33-50.  4. Run DataMain.py by clicking the green arrow in the Command Section of PyCharm. (Refer to Figure 3 of Figures Section) |
| **Expected Results** | All tables in MySQL populate within 2 hours. |
| **Priority** | High |
| **Pass/Fail** | Pass: All tables in MySQL populate within 2 hours.  Fail: All tables in MySQL populate in over 2 hours. |