

Florida International University

School of Computing and Information Sciences

Final Deliverable

Project:

EFORT-EFWE: Empowered FOREX Trader

Team Members: Giuliana Avogadro and Elizabeth Mantilla

Product Owner: Dr. Masoud Sadjadi

Instructor: Dr. Masoud Sadjadi

Abstract

This document explains the work done on the already implemented EFort system to improve and facilitate the exchange of information between the EForT database and Forex's servers and the established VPS servers. The main focus of the project is to test and edit existing code, and implement functionality take care of automatic trading on behalf of the clients who use the EForT product. This document introduces the system and work to do by listing the requirements of the system that pertain to our code, explaining the initial state of the system and finally defining how the current system works. After introductions comes the user story section, this section details the definition and acceptance criteria for each user story we completed in these six sprints. The project plan section is designed to organize these user stories by the sprint they were completed in and how much time they took to complete. The biggest section is titled system design. This section goes into detail about the architecture and framework of the EFort system and presents some diagrams that complement these explanations such as a class and subsystem diagram.

As additional information we included a glossary and appendix. The glossary defines some acronyms that are constantly repeated throughout the document. The appendix is divided into two sections, Appendix-A presents all the UML diagrams that weren't figured in the system design section like sequence and use case diagrams. Appendix-B is a collection of screenshots that illustrate some of the most important aspects of the functionality we implemented.

Table of Contents

Abstract	1
Table of Contents	2
1. INTRODUCTION	3
1.1 Current System	4
1.2 Requirements of the System	4
1.3 Purpose of New System	4
2. USER STORIES	5
Implemented User Stories	5
3. PROJECT PLAN	6
3.1 Hardware and Software Resources	6
3.2 Sprint Plan	6
3.2.1 Gantt Chart	6
4. SYSTEM DESIGN	7
4.1 Architectural Patterns	7
4.2 System and Subsystem Decomposition	7
4.3 Deployment Diagram	7
4.4 Design Patterns	7
4.5 Class Diagram	7
5. GLOSSARY	8
6. APPENDIX	9
Problem	9
Requirements	10
Implementation	10
Verification	11
Summary	12

1. INTRODUCTION

EFort is short for Empowered Forex Traders and is a web application created for fictitious Forex MetaBroker company. The website aims to help individuals with no experience or knowledge about Foreign Exchange Market to trade like professionals by mimicking the actions of historically proven to be successful trading algorithms. It requires the users to pay a flat monthly subscription fee to get the trading benefits the website offers.

Our team, EFort-EFWE, is responsible for improving and facilitating the exchange of information between the EFort database and Forex's servers and our VPS servers which take care of automatic trading on behalf of the clients who use the EFort product.

1.1 Requirements of the System

- The website should be able to start a MetaTrader terminal on any of the three VPS servers (Virtual Private Server) set up with a trader's FOREX account so that automated trading can take place.
- The terminal that is started should be able to be paused, resumed, and stopped at any point via the VPS management page on the website.
- The website should be able to run simultaneously on a VPS server.
- Real data should be generated by an expert advisor running on a MetaTrader terminal and sent to the EFort Web API to be logged in the EFort DB.
- Changes in the value of the equity or balance of a user's FOREX account should be sent to the EFort Web PI and recorded.
- The deals, orders and changes in position a FOREX user does on their account should also be sent and recorded.
- The website Web API and VPS API should have tested functions to get data by the row ID, get all the values, insert data, update data, and delete data for the forex account information and the user account information.

1.2 Previous System

The summer EForT-EFWE team created the framework for the VPS API and created several experts and scripts in MT5. These functions and scripts weren't able to be thoroughly tested and implemented. Our team's job was to check the existing code and making it functional. This entailed fixing some API functions and adding new functionality and fixes to the expert advisors so that it is connected to the EForT API.

1.3 Current System

Our project was divided into three main sections. The first section consisted of testing through Postman API functions in the VPS, API in the web, API that were already implemented to make sure the functionality of these were as expected. Any errors that went against requirements or the desired output were edited and retested.

The second section focused around the feature Start terminal. We implemented changes to the API function corresponding to this feature to achieve the desired result of starting a terminal on MetaTrader on a remote VPS server. We also added a functionality that allowed terminals to open in different sessions on the remote server.

The third section consisted of gathering data for the front end of the trader's accounts. We created functions on an expert advisor that checked if the equity or balance for a FOREX account had changed. When this occurred, the data would be sent and consequently stored in the project's DB. This data would be displayed for the user graphically on their home page. Additional API functions, expert advisor functions and tables were created to keep track of other trading information.

2. USER STORIES

This section summarizes the user stories we focused on and completed during the six two week sprints of this project.

2.1 Implemented User Stories

User Story 0: Set up environment

As a developer I would like to have the EForT and VPS environment to be set up **so that** proper testing can be carried out

Acceptance Criteria

- Have Efort running
- Have VPS running
- Have DB populate pages and be updated correctly
- HAVE Postman set up and be able to post and get
- Get token to authenticate postman

User Story 1: Create tests for StartTerminal function

As a developer I would like to have several tests on the function StartTerminal **so that** its proper functionality in various scenarios is proven to hold.

Acceptance Criteria

- Have variables set on vps side
- Credentials set on setting.txt file
- Written successful config file
- MT4 Terminal that is requested opens

User Story 2: Debugging StartTerminal

As a developer I would like to have several tests pass **so that** there is some assurance of proper functioning

Acceptance Criteria

- Have terminal open

- Have ok test pass
- Have extensive test pass

User Story 3: Make StartTerminal fully functional on staging environment

As a developer I would like to have several tests on the function StartTerminal **so that** its proper end to end functionality on the staging environment is proven to hold.

Acceptance Criteria

- Have MT5 terminal open in the right session, so it can be used by a remote desktop

User Story 4: Test and Debug Discover Terminals

As a developer I would like to have several tests on the function DiscoverTerminals **so that** its proper functionality in various scenarios is proven to hold.

Acceptance Criteria

- Test no unexpected errors or exceptions occur
- Test that the appropriate terminal is reached given its path
- Pass both tests

User Story 5: Write tests and debug for SetEfortCredentials

As a developer I would like to have several tests on the function SetEfortCredentials **so that** its proper functionality in various scenarios is proven to hold.

Acceptance Criteria

- Test that function runs correctly and doesn't produce any unexpected exceptions or errors
- Test the appropriate credentials are successfully written to a text file
- Have both tests pass

User Story 6: Write tests and debug for ForexController

As a developer I would like to test all the functions in the ForexController **so that** its proper functionality in various scenarios is proven to hold.

Acceptance Criteria

- Test that all functions run correctly and doesn't produce any unexpected exceptions or errors
- Test data returned matches the actual data in the DB

User Story 7: Write tests and debug for AccountController

As a developer I would like to test all the functions in the AccountController **so that** its proper functionality in various scenarios is proven to hold.

Acceptance Criteria

- Test that all functions run correctly and doesn't produce any unexpected exceptions or errors
- Test data returned matches the actual data in the DB

User Story 8: Generate real data for EFVT tests

As a developer I would like generate Forex account equity and balance data **so that** the equity graph on a trader's account is populated.

Acceptance Criteria

- Log balance and equity via SyncExpert whenever these variables change
- Check for balance and equity change every tick of the market
- Create a balance table
- Create and call EForT API functions that log data into database
- Have DB equity and balance populate with real time data

User Story 9: Migration to MT5

As a developer I would like generate Forex account order, position and deal data whenever a trade event occurs **so that** the EForT tables are populated with real trading data.

Acceptance Criteria

- Create order, deal and position tables
- Create and call three EForT API functions that log each of these data sets into database
- Edit log deal, order and position functions in SyncExpert to call the corresponding log EForT API functions

- Have DB populate with real time data

User Story 10: Testing expert functionality

As a developer I would like to test that all the functions in SyncExpert **so that** its proper functionality in various scenarios is proven to hold.

Acceptance Criteria

- Test data is logged locally
- Test data is logged on staging server
- Test SyncExpert is running when a terminal is started on the staging server
- Test the equity graph in the trader's account is populated with real data

3. PROJECT PLAN

This section describes the planning we did in order to organize our project. We used agile development techniques like having daily scrums and sprint planning, backlog, review and retrospective meetings. This section also defines the resources that were required for the completion of our work.

3.1 Hardware and Software Resources

The following are the software resources used for this project:

- Visual Studio - IDE for WebAPI and VPS API solutions.
- SQL Server Management Studio - Database Manager
- MetaTrader 5 - Everything to do with MQL5
- Postman - Testing API developed

No specific hardware restrictions or requirements were specified.

3.2 Sprint Plan

Sprint	Issue Key	Summary	Status	Start Date	End Date
1	user_story_0	Set up the environment	Done	09/09/19	09/20/19
1	user_story_1	Create tests for Start terminal	Done	09/09/19	09/20/19
2	user_story_2	Debbuging Start terminal	Done	09/23/19	10/04/19
2	user_story_5	Write tests and debug for SetEfortCredentials	Done	09/23/19	10/04/19
2	user_story_4	Test and Debug Discover Terminals	Done	09/23/19	10/04/19
3-4	user_story_3	Make StartTerminal fully functional on staging environment	Done	10/07/19	11/01/19

3	user_story_7	Write tests and debug for AccountController	Done	10/07/19	10/18/19
3	user_story_6	Write tests and debug for ForexController	Done	10/07/19	10/18/19
4-5	user_story_8	Generate real data for EFVT tests	Done	10/21/19	11/15/19
5	user_story_9	Migration to MT5	Done	11/04/19	11/15/19
6	user_story_10	Testing expert functionality	Done	11/18/19	11/29/19

4. SYSTEM DESIGN

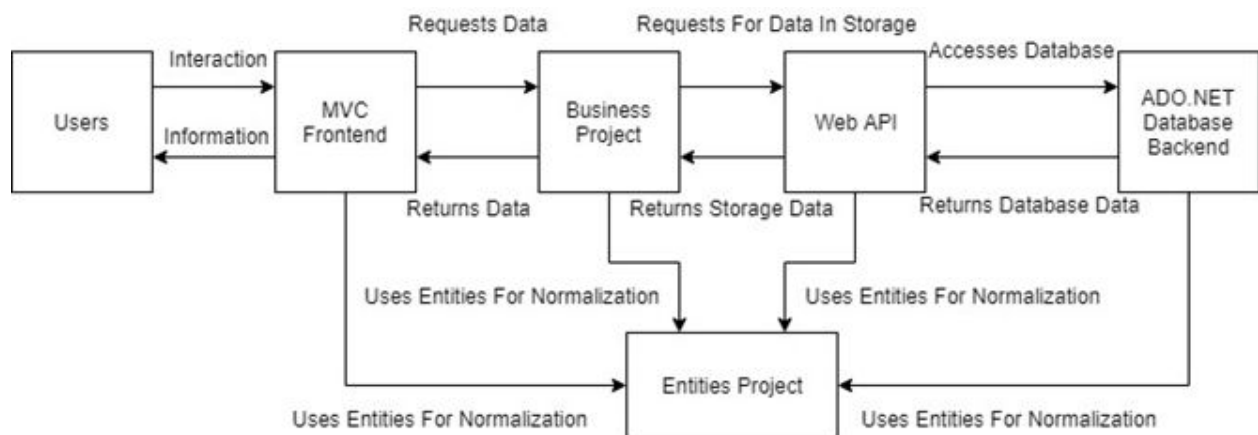
This section will go over the overall design of the system, explaining its design and architectural patterns, showing the different subsystems that make up the system, the logic of deployment and the composition of the classes we modified.

4.1 Architectural Patterns

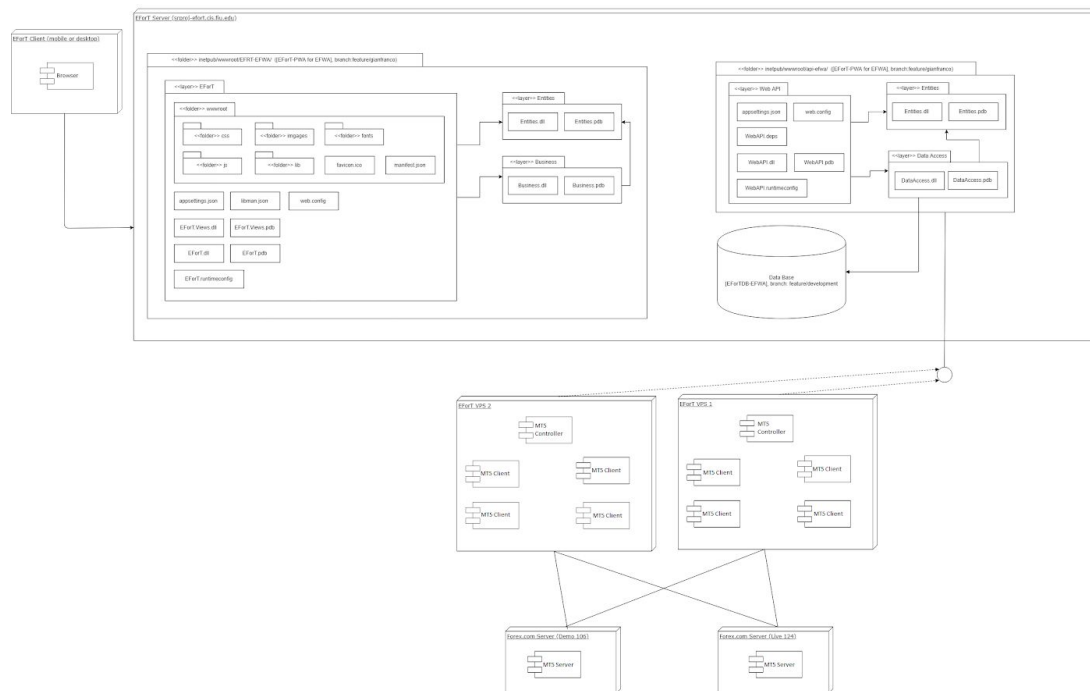
The system is a multi-tier system that uses an MVC design pattern. The tiers are represented as different projects in Visual Studio. The projects used for the system are Business, Data Access, Efort, Web API and Entities.

4.2 System and Subsystem Decomposition

This graph shows the components of the Efort system and their interactions. As the image shows the user interacts with an MVC frontend with its logic organized into models and views. Every layer of the system interacts with the Entities subsystem which represents the tables in the database. The business layer is the communication layer between the frontend and the API functions in Web API. The Web API layer contains controllers that contain various POST and GET functions which purpose is to access information in the DB. The Web API calls a service layer which in turn calls a repository layer which accesses the database.



4.3 Deployment Diagram



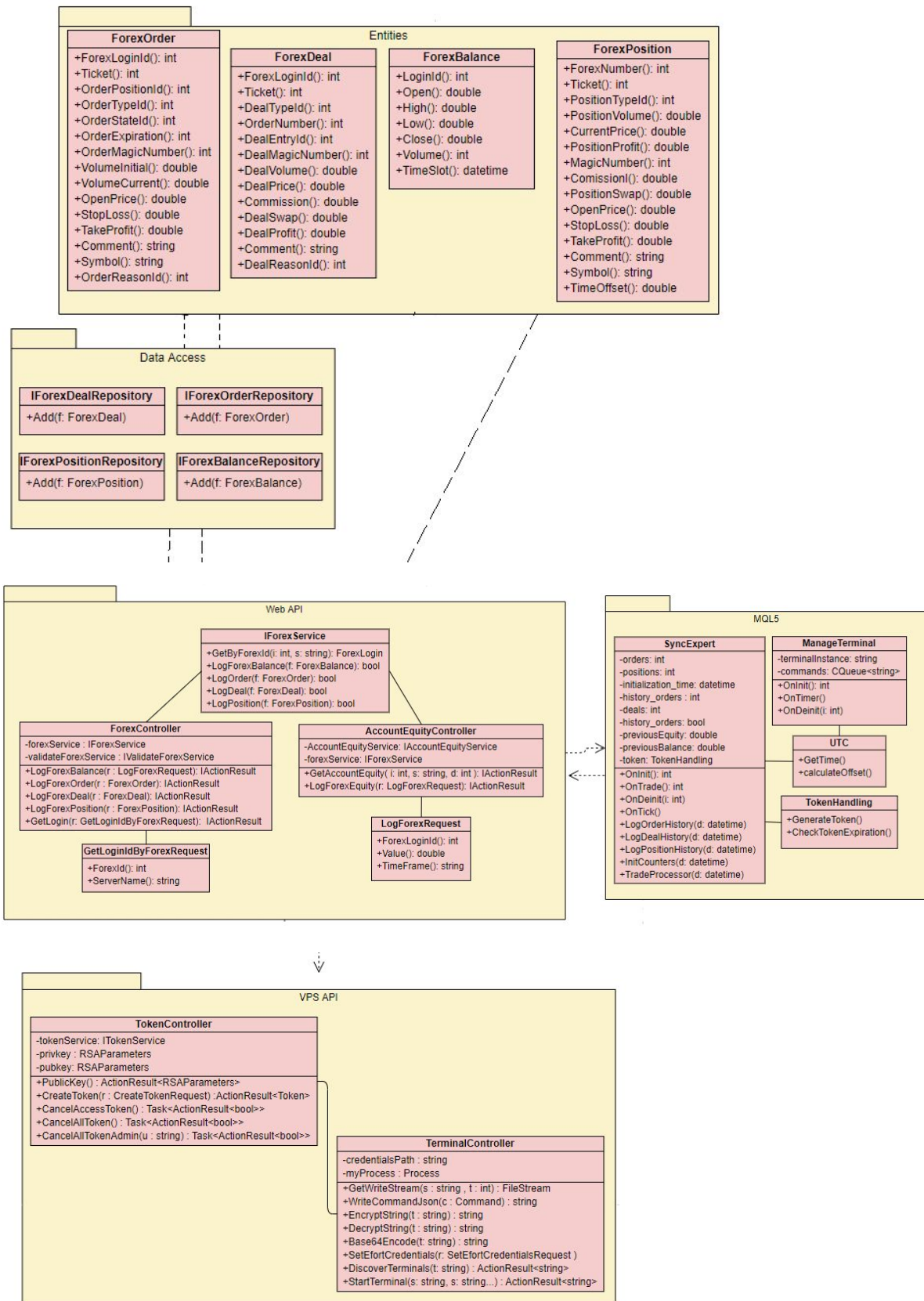
4.4 Design Patterns

The design of the system uses a Model View Controller pattern using the .NET Core and the MVC framework.

The framework of the system works with models, views, services, repositories and controllers. The models and views hold the logic for the frontend of the website. The controllers are the units that have the GET and POST API functions to be called by other functions. The services and repositories are used to connect the controller classes to the DB. So a controller would call a designated service that would then call the designated repository and this layer is in charge of interacting with the tables in the DB.

4.5 Class Diagram

This is a diagram showcasing the classes that were modified by our team



5. GLOSSARY

DB - Database

EFort - Empowered Forex Traders

FOREX - Foreign Currency Exchange

MT5 - MetaTrader 5

MVC - Model View Controller

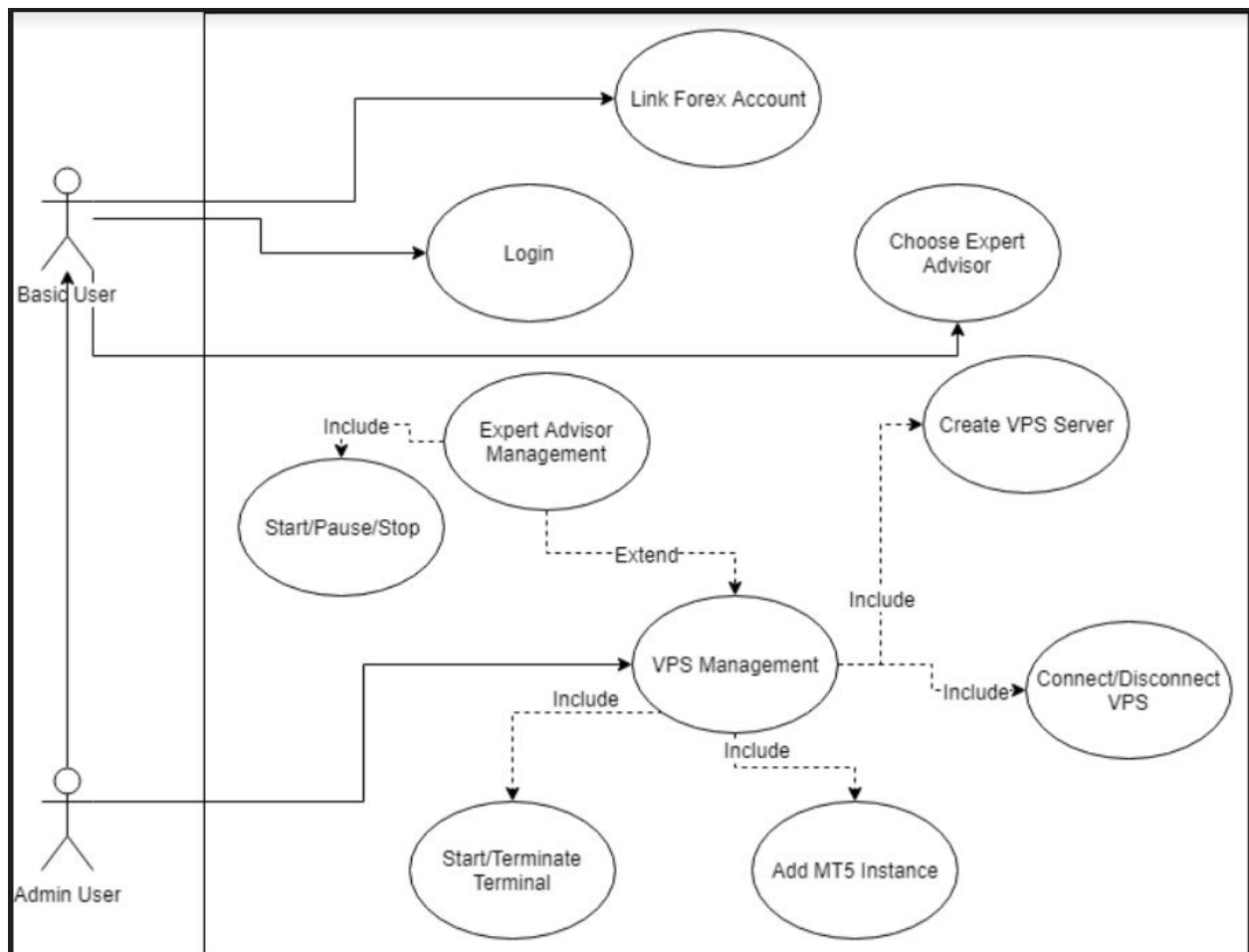
VPS - Virtual Private Server

6. APPENDIX

6.1 Appendix A - UML Diagrams

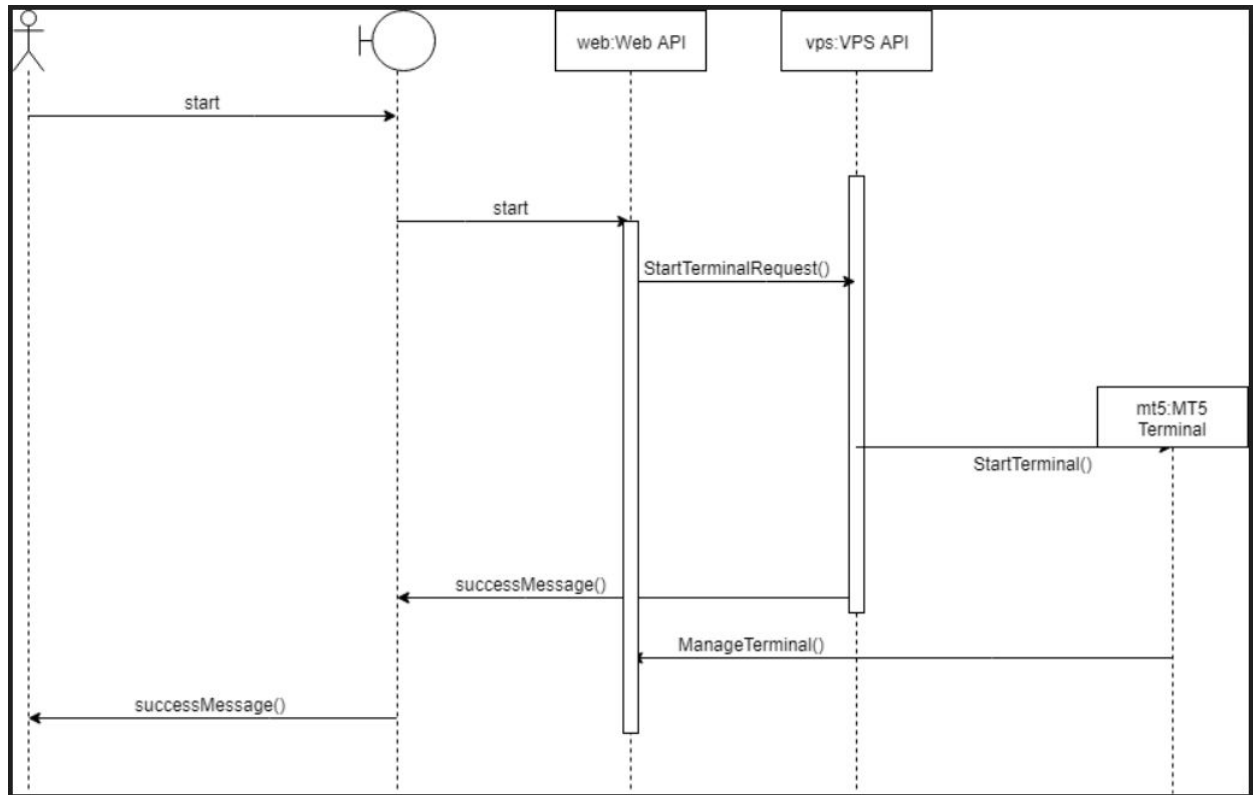
Use Case Diagram

Use case diagram for a basic trader on the website and an administrator account.

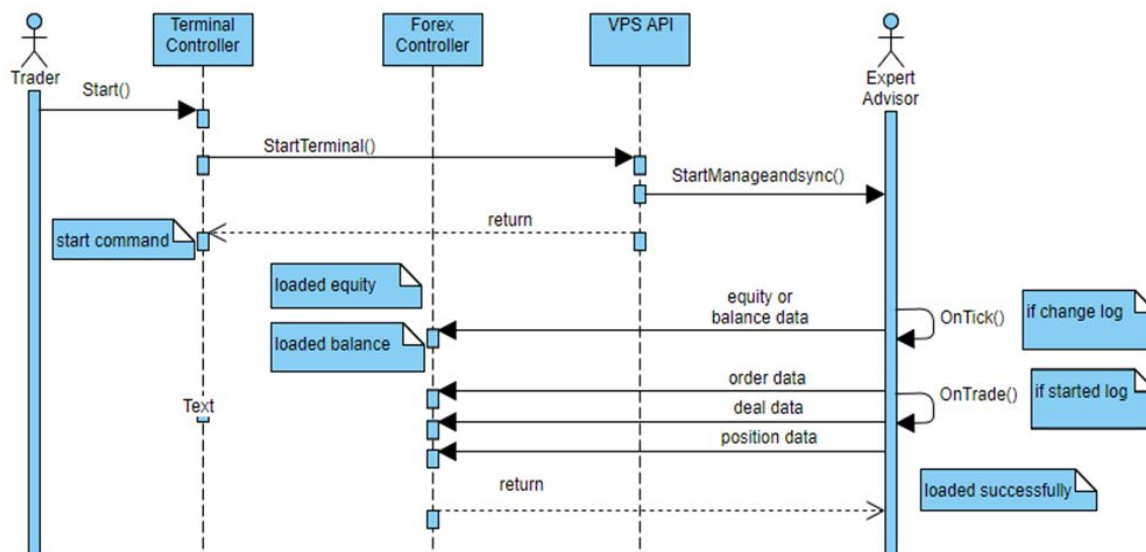


Sequence Diagrams

Sequence diagram for Forex Management



Sequence diagram to show basic trader interaction with the MT5 terminal and its expert advisors.



6.4 Appendix B - User Interface Design

VPS Management interface on admin account with the status of three vps servers. The edit feature allows an admin user to change the name, server address, data folder address and session number.

The screenshot shows the EForT VPS Management interface. The top navigation bar includes links for USER MANAGEMENT, AUTO TRADE MANAGEMENT, REPORT MANAGEMENT, EMAIL TEMPLATE, VPS MANAGEMENT, and SETTINGS. The main content area is titled "VPS Management" and displays a "VPS List" table with the following data:

Server Name	Address	Data Folder Path	Active	Edit	Delete	Status
VPS0	http://srproj-efort.cis.fiu.edu/vps-preprod	C:\Users\EforT\PreProd\AppData\Roaming\MetaQuotes\Terminal\	✓			
VPS1	http://srproj-efort-vps1.cis.fiu.edu/vps-preprod	C:\Users\EforT\PreProd\AppData\Roaming\MetaQuotes\Terminal\	✓			
VPS2	http://srproj-efort-vps2.cis.fiu.edu/vps-preprod	C:\Users\EforT\PreProd\AppData\Roaming\MetaQuotes\Terminal\	✓			

The interface also includes a search bar, a "Show 10 entries" dropdown, and a "Previous 1 Next" pagination control.

The session number should allow different instances of terminal to run at once on the VPS server. In the image below shows two terminal instances running on different sessions, 2 and 4.

```

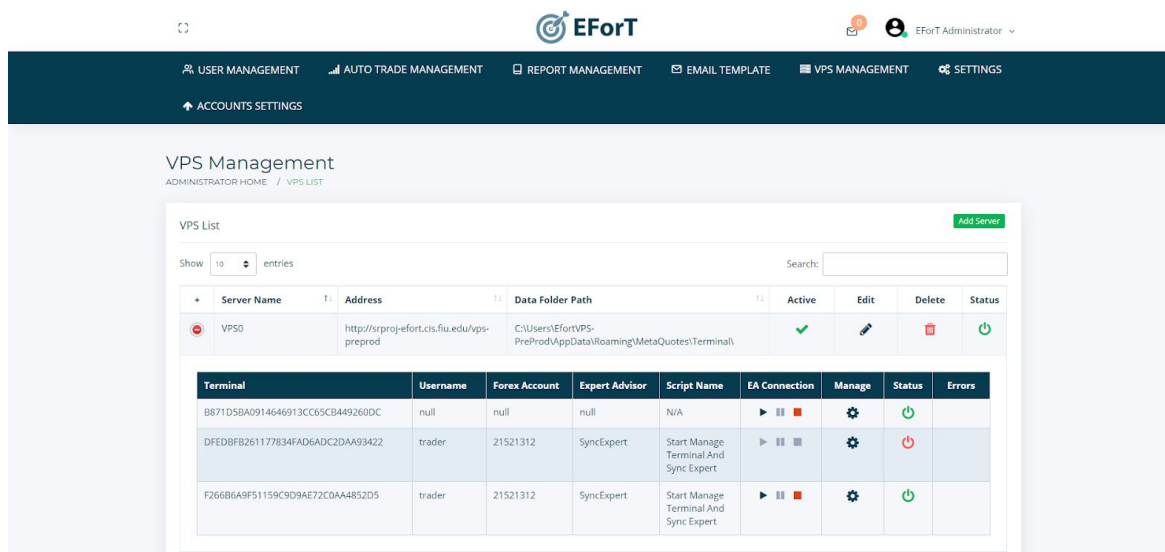
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>tasklist | findstr terminal
terminal64.exe          17500 RDP-Tcp#33          2      98,052 K
terminal64.exe          4736  RDP-Tcp#26          4     201,760 K

C:\Windows\system32>

```

Terminals on the VPS0 server listed. From here a terminal can be set to use a specific user's forex account, run a script upon starting and set it to run with one of the available expert advisors. An admin user can start, pause, resume and stop any terminal that is connected to a valid forex account.



When a terminal is started from VPS Management Metatrader would be started on the server with the SyncExpert and ManageTerminal expert advisors running on them, other trading experts may be running alongside these two staples if the admin specified it in the advisor column. The image below shows a terminal running these two experts and showing messages of success when their intended function is completed.

