ALPHA TRADING API

Reference Manual

Version 1.0

DOCUMENT HISTORY

Date	Version	Author Comments	Document Author
07-16-2019	1.0	New Document	S. Divin Rakesh

APPROVALS

Approval Date	Approved Version	Approver Role	Approver

Table of Contents

1. Introduction

Alpha Trading API is an interface to Alpha Suite thru which a trader can place trades directly using Web sockets. This document provides a detailed description of each protocol to create a sustainable communication between Alpha Fund Account and Trading scripts.

1.1. Audience

This document is intended for developers and business decision makers responsible for developing the product and design performance solution.

The intended audience includes the following business roles:

- Software developers
- Business decision makers
- Consultants, Business Partners, and other technical staffs who work for its development.

1.2. Scope

1.2.1. In Scope

All items listed under section 3 comes under the scope of this document.

1.2.2. Out of Scope

Client systems are out of scope for this document.

2. Solution Overview

2.1. Functional Requirements

- Operating System: Windows 8 and above (32-bit or 64-bit)
- Processor: 1Ghz and above.
- RAM: 1 gigabyte (GB) (32-bit) or 2 GB (64-bit)
- Framework: Microsoft .NET Framework 4.5.2
- Database & Query: MS SQL

2.2. Nonfunctional Requirements

The product is intended to follow with the below nonfunctional parameters:

- Performance
- Reliability
- Usability
- Supportability
- Scalability
- Maintainability

2.3. Data Flow Diagram

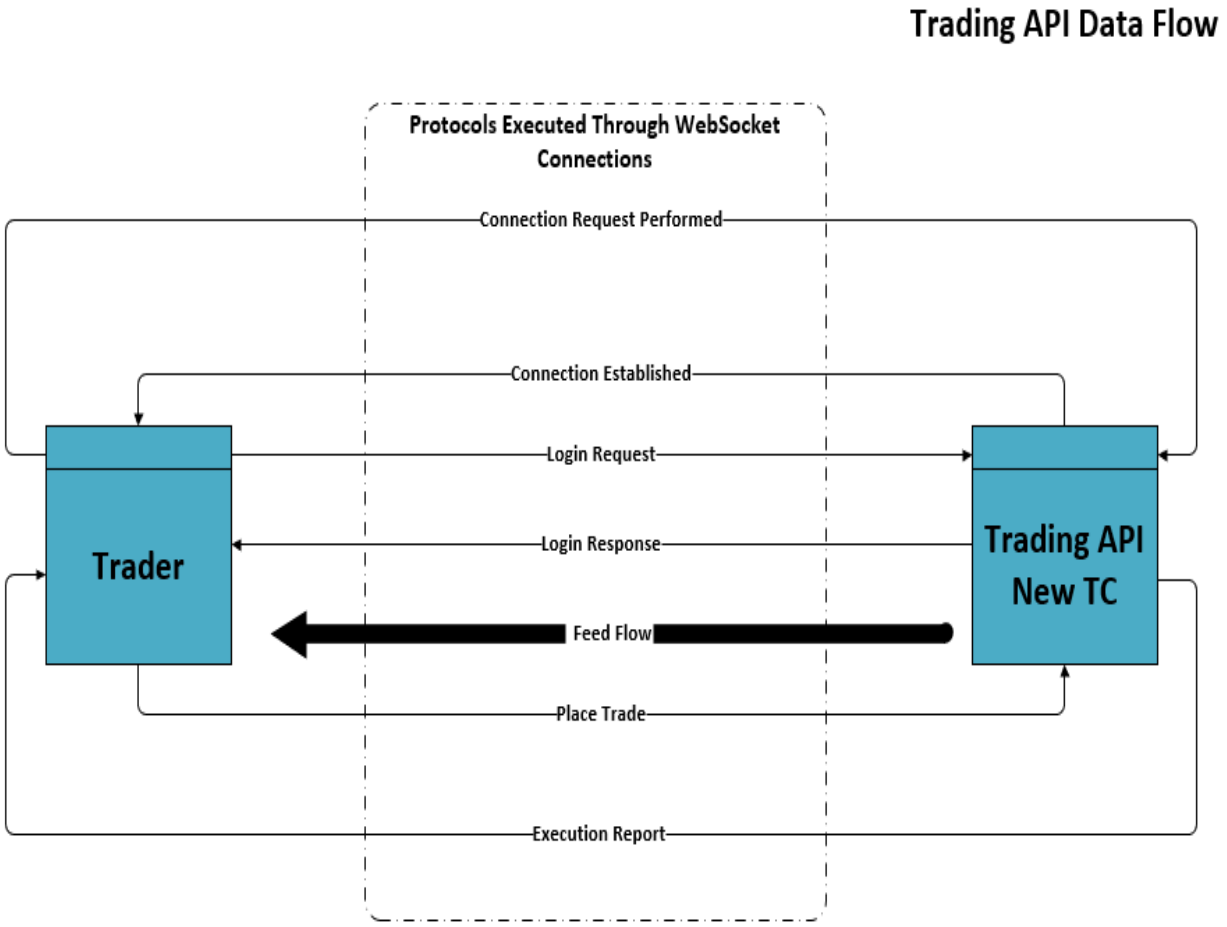

Figure 1: Trading API Data Flow Diagram

3. Specification Overview & Use Cases

The Alpha Trading API forms an integral service for the traders to connect with the Alpha platform.

The Alpha platform will now host both the fund and the investor accounts, and the trades placed on the fund manager account will be clubbed with investors accounts and placed to coverage account for execution.

High-frequency trades (HFT) are not supported by the API at present.

Important: The developer must connect to Trade Server and Feed Server through web socket connection.

3.1. Protocols

3.1.1. Logon Request

The Logon request functionality enables the trader to perform login to the API; thus, enabling them to connect to the Alpha network. This protocol is applicable for both Feed and Trade servers.

The following parameters are required for application into Logon request protocol.

S. No.	Field	Data Type	Description
1	Message Type	Int	41
2	UserID	Char [30]	Username to login
3	Password	Char [30]	Password
4	Version	Double	Version of the protocol being used by the client. Default value is 1.5.
5	RTAccountInfo	Bool	Flag to set if Realtime AccountInfo Broadcast should be subscribed. For Feed Server, this value will always be 0.

JSON Request:

{"UserName":"100327","Password":"admin123","RTAccountInfo":1,"Version":1.5,"msgtype":41}

3.1.2. Logon Response

The Logon response functionality enables API to send a response back to trader corresponding to their request to login. Once the request and response cycle are successful the trader gets logged in to the Alpha network for performing trade using the Trading API. This protocol is applicable for both Feed and Trade servers.

The following parameters are required for application into Logon response protocol.

S. No.	Field	Data Type	Description
1	Message Type	Int	42
2	AccountID	Int	Account ID
3	UserName	Char [30]	Username to login.
4	Status	Int	0, 1, or 2
5	Reason	Char [100]	Reason if Login is unsuccessful: • If Status 0, then Reason ***VALID***. • If Status 1, then Reason ***Invalid Client Version. Please Upgrade***. • If Status 2, then Reason ***Invalid Username/Password***.

JSON Response:

{"AccountId":1028,"Reason":"VALID","Status":0,"UserName":"100327","msgtype":42}

3.1.3. Logout Request

The Logout request functionality enables the trader to perform logout from the Alpha network. This protocol is applicable for both Feed and Trade servers.

The following parameters are required for application into Logout request protocol.

S. No.	Field	Data Type	Description
1	Message Type	Int	43
2	UserID	Char [30]	Username to logout

JSON Request:

{"UserName":"100327","msgtype":43}

3.1.4. Logout Response

The Logout response functionality enables API to send a response back to trader corresponding to their request to logout. Once the request and response cycle are successful the trader gets logged out from the Alpha network. This protocol is applicable for both Feed and Trade servers.

The following parameters are required for application into Logout response protocol.

S. No.	Field	Data Type	Description
1	Message Type	Int	4
2	UserName	Char [30]	Username to logout
3	Status	Int	0, 1, or 2
4	Reason	Char [100]	Reason if Login is unsuccessful: • If Status 0, then Reason ***VALID.*** • If Status 1, then Reason ***Invalid Client Version. Please Upgrade.*** • If Status 2, then Reason ***Invalid Username/Password.***

JSON Response:

{"AccountId":1028,"Reason":"VALID","Status":0,"UserName":"100327","msgtype":44}

3.1.5. New Trade

The New trade functionality enables the API to place a trade on Fund manager Account. The request is for Trade Server.

The following parameters are required for application into New trade protocol.

S. No.	Field	Data Type	Description
1	Message Type	Int	53
2	Account	int	Account number
3	OrderQty	decimal	Order qty for trade
4	ClOrderID	Char [20]	Unique order identifier assigned by client system. Client system must maintain uniqueness of this value throughout the life of the order.
5	Symbol	Char [32]	Symbol for the new trade.
6	OrderType	char	"1" for Market, "2" for Limit, "3" for Stop, "4" for Stop Limit.
7	Price	decimal	Price of the symbol.
8	Side	char	"1" for Buy and "2" for Sell.
9	TIF	char	GTC, DAY. For future use. Set as "1".
10	StopPx	decimal	Stop Price for Stop and Stop Limit Orders. Used only for future use.
11	OrigOrdID	Char [20]	Blank for Open Trade and Order ID assigned from the broker/Exchange for Close trade.
12	TransactTime	double	Transaction time
13	ExpireDate	double	Expiry date for the symbol.
14	PositionEffect	char	"O" for Open Order, "C" for Close Order
15	LinkedOrderID	Char [20]	Order ID linked to this order. This is for identifying related master order.
16	Slippage	decimal	Slippage allowed while placing trade. Reserved for future use.
17	SL	decimal	Stop Loss
18	TP	decimal	Take Profit
19	Comment	Char [32]	Any specific comment while placing new trade.

JSON Request:

{"Account":1028,"OrigOrdID":"","LnkdOrdId":"","ClOrdId":"","Comment":"","Symbol":"EURUSD","PositionEffect":"O","Side":"2","OrderType":"1","Price":1.00,"SL":0,"TP":0,"OrderQty":124.0,"StopPx":0.00,"ExpireDate":0.00,"TransactTime":0.00,"TIF":"1","msgtype":53};

3.1.6. Execution Report

The Execution report functionality enables the API to send the success or failure report back to the trader performing the trade.

The following parameters are required by the Execution report protocol.

S. No.	Field	Data Type	Description
1	Message Type	Int	54
2	Account	int	Account No
3	ClOrdID	Char [20]	Order ID generated by the client
4	OrderID	Char [20]	Order ID generated by the broker/exchange.
5	PositionEffect	char	“O” for Open Order, “C” for Close Order
6	OpenPrice	decimal	Fill price of the Order
7	ClosePrice	decimal	Close Price of the Order
8	OrderStatus	int	Order Status. 50 = FILLED, 56 = REJECTED
9	OrderQty	int	Fill Qty
10	Reason	Char [150]	Reason in case of rejection
11	Symbol	Char [20]	CONTRACT NAME
12	Side	Char	“1” for Buy and “2” for Sell
13	OrderType	int	49 for Market, 50 for Limit. Current version will only support market orders.
14	SL	decimal	Stop loss
15	TP	decimal	Take profit
16	TransactTime	long	Unix timestamp
17	Profit	decimal	Profit value

JSON Response:

{"Account":1028,"ClOrdID":"","ClosePrice":0,"OpenPrice":109.694,"OrderID":"1559209994219M1",
"OrderQty":437500,"OrderStatus":50,"OrderType":49,"PositionEffect":"O","Profit":0,"Reason":"","
SL":0,"Side":"BUY","Symbol":"USDJPY.fx","TP":0,"TransactTime":0,"msgtype":54}

3.1.1. Get Symbols Request

The Get Symbols Request functionality enables the trader to Alpha symbols for performing trade via the Trading API. The request is for Trade Server.

The following parameters are required by the Get Symbols Request protocol.

S. No.	Field	Data Type	Description
1	Msgtype	int	123
2	Account	int	Account number

JSON Request:

{"Account":1028,"msgtype":123};

Following are the parameters of the nested protocol.

S. No.	Field	Data Type	Description
1	Msgtype	int	124
2	Symbol	List <string>	List of symbols for which feed is to be subscribed. Each list item contains one symbol only, i.e., EURUSD.
3	NoOfSymbols	int	Specifies no of symbols requested in this request. **Note:** A maximum of 90 symbols are subscribed in one request.
4	ConnID	int	Connection ID
5	isLast	int	This is used to list the last order. If more order, isLast is 0; if final order, isLast is 1.

JSON Response:

{"ConnID":1000,"NoOfSymbols":3,"Symbols":[{"Name":"EURUSD"},{"Name":GBPUSD"},{"Name":"EURJPY"}],"isLast":1,"msgtype":124}

3.1.2. Subscribe Feed

The Subscribe feed functionality enables the trader to receive the Alpha price feeds via the Trading API. The request is for Feed Server.

The following parameters are required by the Subscribe feed protocol.

S. No.	Field	Data Type	Description
1	Msgtype	int	17
2	Symbol	List <string>	List of symbols for which feed is to be subscribed. Each list item contains one symbol only, i.e., EURUSD.
3	NoOFSymbols	int	Specifies no of symbols requested in this request. **Note:** A maximum of 90 symbols are subscribed in one request.
4	ConnID	int	Connection ID

JSON Request:

{"Symbol":[{"Name":"EURUSD"},{"Name":"GBPUSD"}],"ConnID":1000,"NoOfSymbols":2,"msgtype":17};

3.1.3. Unsubscribe Feed

The Unsubscribe feed functionality enables the trader to stop the Alpha price feeds via the Trading API. The request is for Feed Server.

The following parameters are required by the Unsubscribe feed protocol.

S. No.	Field	Data Type	Description
1	Msgtype	int	18
2	Symbol	List <string>	List of symbols for which feed is to be unsubscribed. Each list item contains one symbol only, i.e., EURUSD.
3	NoOFSymbols	int	Specifies no of symbols requested in this request. **Note:** A maximum of 90 symbols are subscribed in one request.
4	ConnID	int	Connection ID

JSON Request: {"Symbol":[{"Name":"EURUSD"}],"ConnID":1000,"NoOfSymbols":1,"msgtype":18};

3.1.4. Subscribed Symbol Information and Feed Response

For the symbol subscription request, the API sends back two responses. They are:

- Symbol Information
- Feed Response

Symbol Information: Following parameters are required for the Symbol Information response protocol to acquire desired results.

S. No.	Field	Data Type	Description
1	Msgtype	int	110
2	NoOfSymbols	int	Number of symbols ServerSymbolInfo list.
3	IsLast	int	This is used to list the last order. If more order, isLast is 0; if final order, isLast is 1.
4	ServerSymbolInfo	List	This list will contain symbol information.

ServerSymbolInfo List:

S. No.	Field	Data Type	Description
1	Msgtype	int	110
2	NoOfSymbols	int	Number of symbols requested by the trader.
3	BuyPercentage	decimal	Applicable for margin calculation.
4	ConnID	int	This is the gateway ID used to connect to manager.
5	ContractSize	int	Size of the contract.
6	Digit	int	This is the number of digits after the point in the quote.

S. No.	Field	Data Type	Description
7	MarginCurrency	Char [12]	The currency used for margin calculation.
8	Margin_initial	decimal	This is the margin currency required to open an order with the volume of 1 lot.
9	Margin_maintenance	decimal	This is the margin to maintain open orders.
10	Margin_mode	int	This is the margin calculation mode (0 - Forex; 1 - CFD; 2 - Futures; 3 - CFD-Index).
11	Point	decimal	This is the symbol point value.
12	SecurityType	int	Type of securities.
13	SellPercentage	decimal	This is used in the calculation of swap.
14	SymCurrency	Char [12]	Currency of the symbol.
15	Symbol	Char [22]	An instrument.
16	Tick_size	decimal	Tick size.
17	Tick_value	decimal	The price of the tick.
18	Profit_mode	int	This is the mode used for profit calculations.
19	ProfitCurrency	Char [12]	Profit currency of the symbol.

JSON Response:

{"NoOfSymbols":1,"ServerSymbolInfo":[{"Ask_tickvalue":0,"BuyPercentage":100,"ConnID":1000,"
ContractSize":100000,"Digit":5,"MarginCurrency":"EUR","Margin_initial":0,"Margin_maintenance
":0,"Margin_mode":0,"MarkUp":0,"Point":0.00001,"ProfitCurrency":"USD","SecurityType":0,"SellP
ercentage":100,"Source":"EURUSD","SymCurrency":"EUR","Symbol":"EURUSD","Tick_size":0,"Tick
_value":0,"profit_mode":0}],"isLast":1,"msgtype":110}

Feed Response: Following parameters are required for the Feed Response protocol to acquire desired results.

S. No.	Field	Data Type	Description
1	Msgtype	int	20
2	NoOfSymbols	int	Number of symbols in QuotesStream list.
3	ConnID	int	This is the gateway ID used to connect to manager.
4	QuotesStream	List	Feed details

QuotesStream List:

S. No.	Field	Data Type	Description
1	Depth	int	No of market depth for feed. (Max 10)
2	FeedInDepth	List	Feed details per market depth
3	Symbol	Char [32]	Symbol Name
4	Time	Long	Last updated time of feed. This value will be in Unix timestamp.

FeedInDepth List:

S. No.	Field	Data Type	Description
1	AskPrice	decimal	Ask price of feed per market depth.
2	AskQuantity	int	Ask volume of feed per market depth.
3	BidPrice	decimal	Bid price of feed per market depth.
4	BidQuantity	int	Bid volume of feed per market depth.

JSON Response:

{"ConnID":"1000","NoOfSymbols":1,"QuotesStream":[{"Depth":1,"FeedInDepth":[{"AskPrice":1.12069,"AskQuantity":161660936,"BidPrice":1.12064,"BidQuantity":578}],"Symbol":"EURUSD","Time":1561361302}],"msgtype":20}

3.1.5. PNL Statements

The PNL Statements is a functionality which provides trader with the information of their profit and loss for the trade. It is a standalone protocol, which is passed by Alpha (default) and does not need any kind of subscription. The protocol gets initiated when a logon request is received by the Trading API from Trader on Trade Server, and as a reply the PNL Statements is sent correspondingly.

The following parameters are required for the PNL Statements protocol to acquire desired results.

S. No.	Field	Data Type	Description
1	Msgtype	int	115
2	UPNL	decimal	This is required for Manager API Gateway linking.
3	Account	int	This is the account number.
4	Balance	decimal	The amount available with the trader.
5	Credit	decimal	The amount used by a trader (in credit) to perform a trade.
6	Equity	decimal	The amount available with the trader for trade.
7	NoOfOrders	int	Number of orders in TradeDetails list
8	VAMI	decimal	This is the value-added monthly index, which is used only with index funds, else the value it holds is always zero.
9	IsLast	int	This is used to list the last order. If more order, isLast is 0; if final order, isLast is 1.
10	FreeMargin	decimal	This is the free margin available on the account.
11	ReturnPerc	decimal	This is the current return percentage on closed trades.
12	UsedMargin	decimal	The amount blocked by open trades.
13	TradeDetails	List	This list Trade information

TradeDetails List:

S. No.	Field	Data Type	Description
1	CurrentAllocation	decimal	The amount allocated to each order. If master, then 0; if slave, then a value will auto-populate.
2	CurrentPrice	decimal	This is the price for each order.
3	EquityPerMapping	decimal	This is the equity for each order being mapped.
4	TradeID	int	This is the individual ID for each trade.
5	UPNL	decimal	This is the Unrealized Profit and Loss for the trade.

JSON Response:

{"Account":1028,"Balance":107301.44,"Credit":0,"Equity":101394.32000000002,"FreeMargin":100394.32000000002,"NoOfOrders":3,"ReturnPerc":8.3859999999999815,"TradeDetails":[{"CurrentAllocation":0,"CurrentPrice":1.12063,"EquityPerMapping":0,"TradeID":"4942993","UPNL":-4139.0000000000182},{"CurrentAllocation":0,"CurrentPrice":0.68696000000000002,"EquityPerMapping":0,"TradeID":"4946055","UPNL":-365.62000000000478},{"CurrentAllocation":0,"CurrentPrice":1.1206799999999999,"EquityPerMapping":0,"TradeID":"4949191","UPNL":-1402.4999999999623}],"UPNL":-5907.1199999999844,"UsedMargin":1000,"VAMI":0,"isLast":1,"msgtype":115}

3.2. Web Services

The following web service functions are used by the Alpha Trading API, currently:

3.2.1. Validate Account

This method will validate the account and return unique GUID. The user need to pass this GUID in future service calls along with other inputs. This GUID will expire after 30 min, automatically, for idle user.

Method Type: POST

Method Signature: ValidationResponse ValidateAccount(LoginCredential User)

URI Template = "ValidateAccount"

Input:

```
public class LoginCredential
{
    public string UserId;
    public string Password;
    public int ConnectionId; // Primary Key of Gateway Connection table
}
```

Output:

```
public class ValidationResponse
{
    public int ConnectionId;
    public Result result;
    public string message;
    public int AccountId;
    public string UserId;
    public string Password;
    public bool IsAuthenticated;
    public bool Active;
    public string Guid;
}

public enum Result
{
    SUCCESS = 0,
    FAILED
}
```

Service Link: http://185.62.85.23:6006/TS_ReportingAPI.svc/ValidateAccount

3.2.2. Get Account Audit Report

This method will return the audit report of an account.

Note: Only Master/Slave Accounts (not supported Index & Coverage accounts).

Method Type: POST

Method Signature: AccountAuditReportResponse GetAccountAuditReport(AccountAuditReportInput Input)

UriTemplate = "GetAccountAuditReport"

Input:

```
public class AccountAuditReportInput

{
    public int AccountID;
    public string Guid;
    public long FromDate; // Unix Timestamp
```

```
    public long ToDate; // Unix Timestamp
}
```

Output:

```
public class AccountAuditReportResponse
{
    public Result result { get; set; }              // Response Result
    public string message { get; set; }              // Error message
    public List<AccountAuditReport> lstSlaveAccountAuditReport;
}

public class AccountAuditReport

{
    public int OrderID { get; set; }
    public int AccountID { get; set; }
    public string Symbol { get; set; }
    public string Side { get; set; }
    public int Volume { get; set; }
    public Decimal OpenPrice { get; set; }
    public string OpenTime { get; set; }
    public Decimal ClosePrice { get; set; }
    public string CloseTime { get; set; }
    public Decimal Profit { get; set; }
    public Decimal Commission { get; set; }
    public Decimal PSCommission { get; set; }
    public Decimal BSCommission { get; set; }
    public Decimal Swap { get; set; }
    public Decimal Balance { get; set; }
    public decimal ContractSize { get; set; }
}

public enum Result
{
    SUCCESS = 0,
    FAILED
}
```

Service Link: http://185.62.85.23:6006/TS_ReportingAPI.svc/GetAccountAuditReportAbbrevi

4. Abbreviations

Abbreviation	Description

5. References

Title	Description