# Description

This is a simple web API that retrieves information about selected publicly traded stocks from a third party, stores that information in its database, and makes it available to users.

# Supported actions

**index.php?page=CreateRecord&symbol=IBM&timeSeries=daily&adminKey=admin**

Method: POST

Description: Creates historical data for a stock of your choosing.

Parameters:

* symbol: Required: The name of the equity of your choice. For example: symbol=IBM
* timeSeries: Required: The selected time frame for historical data. Accepts three different inputs: daily, weekly and monthly. For example: timeSeries=daily
* adminKey: Required: This function is only available to authorised users with an adminKey. For example: adminKey=admin

**index.php?page=ReadRecord&symbol=IBM&timeSeries=daily**

Method: GET

Description: Returns tracked historical data for a stock of your choosing.

Parameters:

* symbol: The name of the equity of your choice. Leave out the symbol in order to generate the latest data of every stock in the time series. For example: symbol=IBM
* timeSeries: Required: The selected time frame for historical data. Accepts three different inputs: daily, weekly and monthly. For example: timeSeries=daily

**index.php?page=UpdateRecord&symbol=IBM&timeSeries=daily&adminKey=admin**

Method: PUT

Description: Updates tracked historical data for a stock of your choosing.

Parameters:

* symbol: Required: The name of the equity of your choice. For example: symbol=IBM
* timeSeries: Required: The selected time frame for historical data. Accepts three different inputs: daily, weekly and monthly. For example: timeSeries=daily
* adminKey: Required: This function is only available to authorised users with an adminKey. For example: adminKey=admin

**index.php?page=DeleteRecord&symbol=IBM&timeSeries=daily&adminKey=admin**

Method: DELETE

Description: Deletes tracked historical data for a stock of your choosing.

Parameters:

* symbol: The name of the equity of your choice. If empty, the database is wiped for the selected timeSeries. For example: symbol=IBM
* timeSeries: Required: The selected time frame for historical data. Accepts three different inputs: daily, weekly and monthly. For example: timeSeries=daily
* adminKey: Required: This function is only available to authorised users with an adminKey. For example: adminKey=admin

Example:

<http://localhost/BurzaAPIProjekt/index.php?page=ReadRecord&symbol=IBM&timeSeries=daily>

<http://localhost/BurzaAPIProjekt/index.php?page=UpdateRecord&symbol=IBM&timeSeries=daily&adminKey=admin>

# Architecture

Our API is based on the MVC (Model View Controller) pattern.

### Model

Our API uses a simple database which consists of an arbitrary number of tables, each corresponding to a single stock. The columns in a table are:

* Date – the date at which the information was retrieved
* Open – the stock's value, in USD, at market open
* High – the stock's highest value during the trading day
* Low – the stock's lowest value during the trading day
* Close – the stock's value at market close
* Volume – the trading volume during that day

Once the app is started, it checks if the database exists, and if it doesn't, it creates it.

### View

The view consists of an index page, which documents supported functions, and a default page, which simply prints whatever data it is fed, i.e. JSON output and error/success messages.

### Controller

The API supports the four CRUD operations, each of which accepts certain parameters. If the parameters are passed correctly, the desired information is shown or database operation performed. In the latter case, a success message is displayed.  
If the call was somehow made incorectly, an error message is displayed.

If the user accesses the index page, all the supported operations are listed, alongside their descriptions.

### Miscallaneous

The API also contains various configuration files, a router, tests, and PHPUnit, a testing software.

Like most modern APIs, our API was built with REST principles in mind.

# Hosting the API

Hosting this API is simple. The authors recommend using XAMPP, as it has both the web server and the database software built-in. Make sure both the Apache HTTP Server and phpMyAdmin are running, and from that point, running the API is no different from running any other web site.

# Tests

The API comes packaged with two unit tests:

* A test of the model, where a query is sent to the database
* A test of the *create* function, where a table alongside two entries is created

Both tests use fake data.