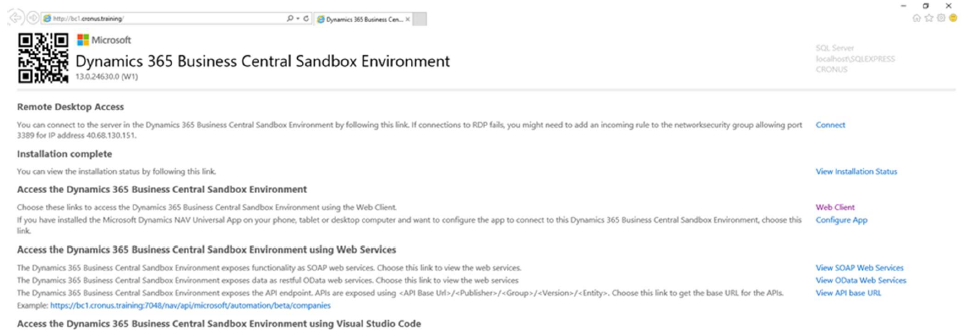
**Data exchange with AL language**

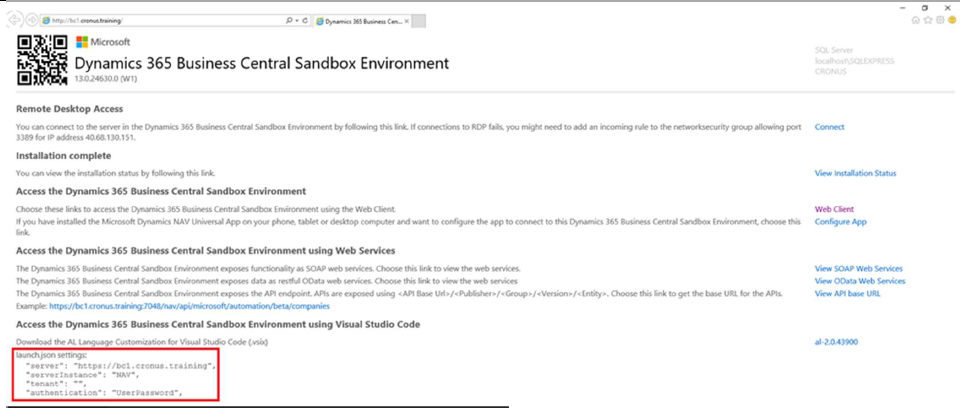
1. **Setting up environment:**

* Open your landing page
* Click “Connect”
* Login with the credentials you received with your boarding pass



1. **Creating new App:**

* Start VS Code
* Press “F1” (“Command Palette”)
* Enter “AL: Go!”
* Enter path (e.g. “…\Documents\Task1”)
* Select the latest target platform (4.0)
* In the dropdown list, pick “Your own server”
* When it asks you for credentials, don’t log in, just press ESC
* Set up the “launch.json”
* Make sure the settings are the same as the ones from your own landing page
* You can find those settings when you click “landing page” on the desktop of the VM.





Set up the “app.json”:

* Change “name” to “Task <Number of Task>”
* Change “publisher” to “<<your Name and Surname>>”
* Add feature “TranslationFile”



Download the Symbols:

* Use the Command Palette (F1) to search for “AL: Download symbols”

If necessary:

* Provide username and password from boarding pass



Delete “Hello world.al” file.

1. **Task 1.**

Parsing simple json request. Importing Share rates to Customer’s card.

**Sources:**

1. Rates source: <https://www.alphavantage.co>
2. Getting API key: <https://www.alphavantage.co/support/#api-key>

For tests you can use MSFT symbols and apikey = demo

1. API URL: https://www.alphavantage.co/query
2. GitHub Repository with source code: github.com/abaludin/SharesRate
3. Request sample:

https://www.alphavantage.co/query?function=GLOBAL\_QUOTE&symbol=MSFT&apikey=demo

**Step 1**

Run Request sample <https://www.alphavantage.co/query?function=GLOBAL_QUOTE&symbol=MSFT&apikey=demo>

in Postman or browser. You send company Symbols (MSFT) to API and get back last rate of company’s shares. In Task 1 we will send request, parse json response from API and save share rate to customer card.

So, let’s begin:

Create new files for Integration Setup table and Page (card type) with next fields:

* Entry no. (As you prefer) \*key
* API URL (Text) = https://www.alphavantage.co/query
* API Token (Text)
* Last Rate Function (Text) = GLOBAL\_QUOTE
* Daily Rate Function (Text) = TIME\_SERIES\_DAILY

Full Table code: <https://github.com/ABaludin/SharesRates/blob/master/src/table/Tab50110.AB_IntegrationSetup.al>

Full Page code:

<https://github.com/ABaludin/SharesRates/blob/master/src/page/Pag50110.AB_IntegrationSetup.al>

**Step 2**

Create “Customer” Table extension with next fields:

* Ticker Symbols (code[10])
* Rate (Decimal)

Create Page Extension for Customer Card (or list) to add these fields

Full Table extension code:

<https://github.com/ABaludin/SharesRates/blob/master/src/tableextension/Tab-Ext50110.AB_CustomerExtension.al>

Full Page extension code:

<https://github.com/ABaludin/SharesRates/blob/master/src/pageextension/Pag-Ext50110.AB_CustomerCard.al>

**Step 3**

* Create management codeunit
* Create GetLastRate function which gets Symbols value from Customer record as parameter and returns Share rate (decimal back).
* Each time you need to send GET request to some API – you could use next universal code (pay attention at new http type variables if you never met them before):

## if not Client.Get(URL, Response) then

## Error(Text001\_Err);

## Response.Content().ReadAs(ResponseText);

## if not Response.IsSuccessStatusCode() then

## Error(Text002\_Err, Response.HttpStatusCode(), ResponseText);

Variables:

URL: Text;

Client: HttpClient;

Response: HttpResponseMessage;

ResponseText: Text;

Text001\_Err: Label 'Service inaccessible';

Text002\_Err: Label 'The web service returned an error message:\ Status code: %1\ Description: %2';

To construct URL – use fields of “Integration Setup” table and Symbols variable which you pass to function as a parameter:

## URL := IntegrationSetup."API URL" +

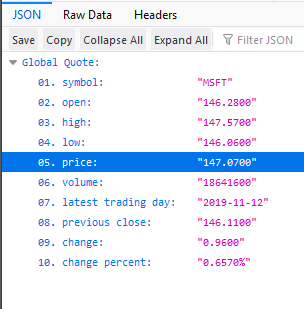
## '?function=' + IntegrationSetup."Last Rate function" +

## '&symbol=' + Symbols +

## '&apikey=' + IntegrationSetup."API Token";

Create ParseResponse function which accepts ResponseText and returns decimal value.

Look at response:



We need value '05. price' as final result.

As I mentioned at the beginning – it’s a simple json response with only token, so code extracting the needed value will be like this (pay attention at new json type variables if you never met them before):

## local procedure ParseResponse(ResponseText: Text): Decimal

## var

## ContentToken: JsonToken;

## QuoteToken: JsonToken;

## PriceToken: JsonToken;

## begin

## ContentToken.ReadFrom(ResponseText);

## ContentToken.AsObject().Get('Global Quote', QuoteToken);

## QuoteToken.AsObject().Get('05. price', PriceToken);

## Exit(PriceToken.AsValue().AsDecimal());

## end;

Full Codeunit code:

<https://github.com/ABaludin/SharesRates/blob/master/src/codeunit/Cod50110.AB_AlphaVantagemanagement.al>

**Step 4**

Create Get Last Rate action on Customer Card Page extension. Pass Symbols from customer and update Rate field with response:

## action(AB\_GetRate)

## {

## Image = Web;

## Caption = 'Get Rate';

## ApplicationArea = All;

## trigger OnAction()

## var

## AlphaMgt: Codeunit "AB\_Alpha Vantage management";

## begin

## TestField(Symbols);

## Rate := AlphaMgt.GetLastRate(Symbols);

## Modify();

## end;

## }

Full Page extension code:

<https://github.com/ABaludin/SharesRates/blob/master/src/pageextension/Pag-Ext50110.AB_CustomerCard.al>

**Step 5**

**Now let’s make our task more complex and try to get and save rates history for customer:**

Run Request sample <https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol=MSFT&apikey=demo>

in Postman or browser. Looks bigger, isn’t it? It contains information about daily rates for last 100 days – so we are able to save rates history to Business Central in one click.

Create Daily Rates Table and list page with next fields:

* Symbols (code[10]) \*key
* Rate Date (Date) \*key
* Open (Decimal)
* High (Decimal)
* Low (Decimal)
* Close (Decimal)

Full table code:

<https://github.com/ABaludin/SharesRates/blob/master/src/table/Tab50111.AB_DailyRates.al>

Full page code:

<https://github.com/ABaludin/SharesRates/blob/master/src/page/Pag50111.AB_DailyRates.al>

**Step 6**

* Create GetDailyRate function with Symbols parameter in management codeunit (same way as GetLastRate function before)
* Get response from Daily Rate API using universal code from **Step 3.** Url will change a little bit:

## URL := IntegrationSetup."API URL" +

## '?function=' + IntegrationSetup."Daily Rate function" +

## '&symbol=' + Symbols +

## '&apikey=' + IntegrationSetup."API Token";

* Create ParseDailyResponse function. A little bit tricky – because token name equals rate date. By default response contains last 100 dates. Remember that weekend and holiday dates does not exists. You could create date loop and select any quantity of days.

Get “'Time Series (Daily)” token from response – we need to parse only it:

ContentToken.ReadFrom(ResponseText);

ContentToken.AsObject().Get('Time Series (Daily)', SeriesToken);

If you already have imported rates – use Last rate date as final for loop. Else – take -100 days from today:

## DailyRate.SetRange(Symbols, Symbols);

## if DailyRate.FindLast() then

## EndTokenDate := CalcDate('<1D>', DailyRate."Rate Date")

## else

## EndTokenDate := CalcDate('<-100D>', Today());

After that you can take token by token from response from final day to current, creating Rate record for each token:

## for StartTokenDate := CalcDate('<-1D>', Today()) Downto EndTokenDate do

## if SeriesToken.AsObject().Get(Format(StartTokenDate, 0, 9), PriceToken) then begin

## DailyRate.Init();

## DailyRate.Symbols := Symbols;

## DailyRate."Rate Date" := StartTokenDate;

## PriceToken.AsObject().Get('1. open', ValueToken);

## DailyRate.Open := ValueToken.AsValue().AsDecimal();

## PriceToken.AsObject().Get('4. close', ValueToken);

## DailyRate.Close := ValueToken.AsValue().AsDecimal();

## PriceToken.AsObject().Get('2. high', ValueToken);

## DailyRate.High := ValueToken.AsValue().AsDecimal();

## PriceToken.AsObject().Get('3. low', ValueToken);

## DailyRate.low := ValueToken.AsValue().AsDecimal();

## DailyRate.Insert();

## end;

Full codeunit code:

<https://github.com/ABaludin/SharesRates/blob/master/src/codeunit/Cod50110.AB_AlphaVantagemanagement.al>

**Step 7**

Create action for Customer card calling “Daily rates” page:

## action(AB\_GetDailyRate)

## {

## Image = Web;

## Caption = 'Get DailyRate';

## ApplicationArea = All;

## trigger OnAction()

## var

## DailyRates: Record “AB\_DailyRates”;

## AlphaMgt: Codeunit "AB\_Alpha Vantage management";

## begin

## TestField(Symbols);

## AlphaMgt.GetDailyRates(Symbols);

## DailyRates.Setrange(Symbols, Symbols);

## Page.Run(Page::"AB\_Daily Rates", DailyRates);

## end;

## }

Full page extension code:

<https://github.com/ABaludin/SharesRates/blob/master/src/pageextension/Pag-Ext50110.AB_CustomerCard.al>

**Final Result:**

