MDRest4i User Manual

MDRest4i REST on IBM i Simplified

Midrange Dynamics

1. MDRest4i v14

1.1 MDRest4i Documentation

About This Guide

This manual is comprised of these sections:

- Installation
- Product Overview
- SDK Web UI Help
- Coding Guide
- MDRest4i Commands

You can download a **PDF version of this guide here**.

Getting Started

Here are a few steps to get you started and up and running with MDRest4i:

- Download the Product
- Read the Product Overviews
- Install the Product Library on IBM i.
- Install the SDK Web UI
- Login to the SDK - http://youribmiserver:yourSDKport/cons/
- Create a Provider or consumer program

Useful Shortcuts

SDK Web UI

Coding

Provider-Overview

Consumer-Overview

Quick Install

Troubleshooting

Downloads

Previous Versions

1.2 Installation

1.2.1 MDRest4i 14.0 Installation Instructions

This installation guide will take you through the installation of the V14 MDRest4i MDRFRAME framework component, and the MDRest4i SDK web application on the IBM i.

> **Please Note**
>
> Please get license keys from Midrange Dynamics before continuing!

BRIEF DESCRIPTION OF INSTALLATION

The installation consists of installing the MDRST library using command from the MDRST.savf save file on the IBM i.

Apply the license key using command MDRST/MRLICKEY.

The MDRST library contains the **MDRDK** save file, which contains the MDRest4i SDK web UI web application.

If the SDK is required on that LPAR, the next step is to run the MDRST/MDRSDKINS command. This restores, configures, and adds an HTTP server instance for the MDRest4i SDK web UI, creates a default Provider and Consumer program library, and creates a default HTTP server instance to expose Providers(API's)

If additional REST API servers are required on this LPAR or to setup an LPAR for running REST Providers and Consumers only(not developing), create an API server instance using command **MDRHTTPAPI**

> **Please Note**
>
> This installation is for V14 upwards, and cannot be used to automatically upgrade from V12 and earlier.

> **Which Product to Install?**
>
> MDRest4i **MDRFRAME** is required on **all** LPAR's where any MDRest4i API or consumer runs, including the development LPAR where the SDK will be installed.
> MDRest4i **SDK** is only required on the LPAR where API or Consumer development takes place.

> **Updating Products**
>
> Command MDRST/MDREST4INS is used for both **installing** AND **updating** MDRest4i. Command MDRST/MDRSDKUPD is used for **updating** the SDK (after MDRST/MDREST4INS has completed)

SIMPLIFIED INSTALLATION STEPS

The following is a simplified version of the installation steps, assumes you have already installed the MDREST4iNS command, and you have your MDRest4i License keys.

Step	Comments
Login to your IBM i with a user that has QSECOFR Authority	
Download zip and upload MDRST.SAVF onto IBM i	MDREST4INS command should still be in QGPL so you can skip that
Stop the MDRST SDK HTTP server and any HTTP API server you have created :	`ENDTCPSVR SERVER(*HTTP) HTTPSVR(MDRST) ENDTCPSVR` `ENDTCPSVR SERVER(*HTTP) HTTPSVR(MDRSTAPI) ENDTCPSVR`
Make sure lib MDRST not locked and in use	`WRKOBJLCK OBJ(MDRST) OBJTYPE(*LIB)`
Prompt Command `MDREST4INS`	Prompt command **MDREST4INS** to specify a different backup lib or different instance name. **MDREST4INS** is also used to *update* the server applications.
`ADDLIBLE MDRST`	This lib will now be at the lasted build
Enter command `MRLICKEY`	Use command **MRLICKEY**Review the license status and the correct build and version number at the bottom. Press enter when complete
Prompt command `MDRSDKINS`	**MDRSDKINS** is only required if SDK is being installed. This will install the SDK web application server **MDRSDKUPD** is only required if SDK is being updated. This will update the SDK web application server code and copy back the existing settings.
Restart your SDK and API servers	`STRTCPSVR SERVER(*HTTP) HTTPSVR(MDRST)`
Test the server is running	`http://[your serverip or host]:[yourport]/mdrapi/mdrhello?firstName=Mike&lastName=Smith&Title=Mr`

DETAILED STEP-BY-STEP INSTALLATION

For first time installations, kindly select the following installation pages in sequence:

1. Pre-Requisites
2. Objects Created by Installation
3. MDRest4i MDRFRAME Installation
4. MDRest4i SDK Installation
5. Create API Server Instance

1.2.2 Step by Step Installation

Pre-Requisites

Operating System

IBM i (AS/400) System with OS/400 Release V7R2M0 or higher

The current version can be checked using command WRKLICINF

PTFs

- Latest cumulative PTF's for TCPIP/IP
- Latest cumulative PTF's for IBM HTTP Server for i

MRLICKEY License key command

In order to use the products, a valid License Key is required for the core product MDRest4i, which is based on the Serial Number of the system, and the version of the Product.

MDREST4i uses a license key system independent from MDCMS. So anylicense keys or installations for MDREST4i prior to May 21, 2020 will need to be reentered into MDRST4i using prompted command `MDRST/MRLICKEY LTYP(\*INT)` after the new version is installed.

Please provide Midrange Dynamics or a reseller with the serial number (obtained with command WRKLICINF or DSPSYSVAL QSRLNBR) and product version and continue with the installation once the key or keys have been provided.

The following screen will appear when using `MDRST/MRLICKEY LTYP(\*INT)`:

```
                        Modify MDRest4i License Keys


Enter the new License Key(s) and press Enter.

 Product       License Key          Status
 MDRest4i      ____ ____ ____ ____

                           System Serial Number: 78656EX
                                    Partition#: 007
                                    Version ID: 14

F3=Exit
```

Paste in the key provided in the email received from Midrange Dynamics support, and press enter

If the keys are provided in file MDLICENSE.savf, include this file in the same IFS path or library as the product save files. When MDLICENSE.savf is included, the keys for the given serial number will be automatically applied to the product instance, and the keys for all other included systems will be stored in the product instance so that if a switch is made to a backup system, the backup keys will automatically be applied to the product, thus avoiding any delays.

Note: License keys obtained as a save file (SAVF) from the mdlicense.mdcms.ch website will sometimes have additional data in the save file name. for example: “MDLICENSE_20220829_160955_411724.savf” Before running MRLICKEY to install this save file, rename it so the name of the save file is “MDLICENSE.savf”

User Authorization

The user performing the installation must have *SECADM and *ALLOBJ authority on the system.

Access to the Digital Certificate Manager (DCM) is required for using SSL for REST APIs or Consumers and SOAP Services or Consumers

Adopted Authority

The system value **QALWOBJRST** must include the choice of ***ALL** or ***ALWPGMADP** so that the product programs with the adopt attribute can be restored.

Disk Space

The product initially requires about 205 MB of space at installation time.

MDREST4i history typically requires an additional 5 MB per year of use. This is subject to the logging details selected in each program generated.

Product Library Locks when Upgrading

When upgrading an existing version of the MD Products to a new build, the existing library instances for MDRST, may not be in use.

Object locks can be checked by using command:

```
WRKOBJLCK OBJ(MDRSTxxxx) OBJTYPE(*LIB).
```

Where “xxx” is the instance extension – blank if the default MDRST was used.

Please also check any http instances that use MDRST instance libraries in the ScriptAlias, ScriptAliasMatch, **SetEnv QIBM_CGI_LIBRARY_LIST** or job descriptions (mdrst/mdrst) setting in the HTTP server instance configuration. These server instances should be stopped until after the installation.

If locks exist, you can cleanly end the jobs ahead of time, or specify parameter END(***YES**) on the MDREST4INS command to automatically end all jobs locking the product libraries.

Installed Objects

OBJECTS CREATED BY INSTALLATION PROCESS

QGPL Objects

The following commands are placed by default into library QGPL:

MDREST4INS – The MDREST4i Product Installer

For each command, a corresponding program and panel group are also placed in QGPL.

Libraries

Some or all of the following libraries will be created, depending on the product:

Library	Description	Initial Size
MDRST	MDRest4i objects	205 MB

MDRST contain some site-specific data, so they should be regularly backed up.

By default, the libraries are named as stated in the table. At installation time,
a 1-4 character Instance ID (ENV) can be defined which will be used as the suffix for the library names. For example, suffix X would mean that product MDRST would be stored in library MDRSTX.

This way, several instances of the MDREST4i products can reside on the same system.

User Profiles

A user profile is created to own the objects in the product libraries. The profile is created without the ability for users to sign on or otherwise make use of the profile. By default, the name of this user profile is MDOWNER, but can be changed at installation time by specifying a different value for MDREST4INS parameter OWN.

If the product owner profile already exists, it is left as is.

MDRST4i programs use adopted authority from the owner profile, which has *ALLOBJ authority, so that the actual users authorized to perform functions in MDREST4i do not need to have any special system authorities to accomplish the task of making changes to your business applications.

Any programs providing access to a command line do not have adopted (*OWNER) authority.

None of the programs have parameter “Use adopted authority” set to *YES, ensuring that authority won’t be inherited from your internal calling programs.

MDRFRAME Installation

> **Tip: Determine your installed version of MDRest4i**
>
> To determine the version you have installed use the following command: `DSPDTAARA MDRSTxxxx/MRVERSION` where *xxxx* is the instance name. Leave as *MDRST* if default instance is installed

> **Warning**
>
> This installation is for V14 upwards, and cannot be used to automatically upgrade from V12 and earlier.

> **Please Note**
>
> The value of MRVERSION will be displayed as 8.2.8 for v11 of MDREST4i.

THE INSTALLATION STEPS

Downloading and extracting the Save Files

1. Sign into the Midrange Dynamics Service Desk portal at https://support.mdcms.ch/plugins/servlet/desk

   You will need to be registered to use the portal. If not yet registered, request registration from: https://www.midrangedynamics.com/request-access-to-md-service-desk/
2. Proceed to the Downloads section. From here click on MDRest4i V14
3. Select a mirror and download the zip file
4. Save the zip file to a local directory on your PC.
5. Extract the save files to a local directory on your PC.

Option 1 - Installing from Save Files in IFS

1. Copy the save files to an IFS folder on your IBM i system.

> **Tip: Copying files to IBM i IFS**
>
> One of the easiest ways to copy the save files to an IFS folder on your IBM i is to drag and drop them using IBM's ***System i Navigator***.
>
> You can also use FTP. A recommended FTP client is **FileZilla**, which is available for free from the internet. Or you can run FTP in a command or PowerShell/Terminal window on PC or MAC

1. If command **MDREST4INS** already exists in an IBM i library on your system, and it was created since version 8.2.5, build May 20, 2020, skip to section Invoke the MDREST4INS Command
2. Enter command:

   ```
   CRTSAVF QTEMP/MDREST4INS
   ```
3. Enter command:

   ```
   CPYFRMSTMF FROMSTMF('/x/mdrest4ins.savf') TOMBR('/qsys.lib/qtemp.lib/ mdrest4ins.file') MBROPT(\*REPLACE)
   ```

   Where **x** is the name of the IFS folder containing the save files
4. Enter command:

   ```
   RSTOBJ OBJ(\*ALL) SAVLIB(QGPL) DEV(\*SAVF) SAVF(QTEMP/MDREST4INS)
   ```

   The objects may be restored to a different library than QGPL, if desired. However, the CHGCMD command will need to be used on the **MDREST4INS** command to change the library for the program and panel group.

Option 2 - Installing from Save Files in a Library

1. Copy the save files to a Library on your IBM i system.
2. If command MDREST4INS already exists in an IBM i library on your system, and it was created since version 8.2.5, build May 20, 2020, skip to section Invoke the MDREST4INS Command
3. Enter command:

```
RSTOBJ OBJ(\*ALL) SAVLIB(QGPL) DEV(\*SAVF) SAVF(x/MDREST4INS)
```

Where **x** is the name of the library containing the save files

The objects may be restored to a different library than QGPL, if desired.

Invoke the MDREST4INS Command

The MDREST4INS command is used for both first time installs, and upgrades of MDRest4i. it also installs/updates the MDRDK save file into the MDRST library which contains the MDRest4i SDK software.

Enter command **MDREST4INS** and press F4 to review command parameters and make any necessary changes (press F1 or use the parameter table on next page for more information).

> **Note**
>
> Installations may occur in an interactive or batch job. Interactive is recommended for new installations for improved monitoring and prompting of the process.

MDREST4INS Command Screen:

```
                      Install MDRest4i (MDREST4INS)

Type choices, press Enter.

Save File Location Type  . . . .   *IFS          *IFS, *LIB
IFS-Path containing Save Files     '/MDREST4i'

Library containing Save Files  .                 Character value
Product Instance . . . . . . . .   *DFT          *DFT, Product Library Instanc
Copy Data from Instance  . . . .   *SAME         *SAME, *DFT, *NONE, Instance
Backup Library Suffix  . . . . .   '_BU'         Suffix for Backup Libs
Replace existing Backup Libs . .   *YES          *YES, *NO
Install in ASP Device  . . . . .   *SYSBAS       *SYSBAS, Device Name
End Jobs locking Product Lib . .   *NO           *NO, *YES
Product Object Owner . . . . . .   MDOWNER       User Profile

                                                                      Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

The default values typically used to install MDREST4i standalone are as follows:

MDREST4INS Parameter Table | Parameter | Label | Description | | :--- | :--- | :--- | | LTYP | Save File Location Type | Specifies if the product save files are in an IFS folder or in a library.
***IFS** – The save files are located in an IFS folder. Specify the full path of the folder in parameter PATH
***LIB** – The save files are located in a library. Specify the name of the library in parameter LIB | | PATH | IFS-Path containing Save Files | Specifies the IFS path containing the save files used to install the Midrange Dynamics products. The contents of the path must contain the MDRST.savf save file. | | LIB | Library containing Save Files | Specifies the Library containing the save files used to install the Midrange Dynamics products. The contents of the library must contain the MDRST.savf save file. | | ENV | Product Instance | Specifies the instance ID of the product. The instance value is appended to the MDRST product library in order to allow multiple instances of MDRest4i on the same partition.

***DFT** – A suffix is not appended to the library name. The product library will be named **MDRST**.

The suffix to be used. Up to 4 characters are allowed and each character must be accepted as part of a library name. Example ***ENV=T11*** so the name of the installed product library becomes **MDRSTT11** | | FENV | Copy Data from Instance | Specifies the location of an instance of the products, if it
exists, that should be used for the copy of the data to the new

version.

***SAME** – The same instance, or library suffix, that was defined in parameter *ENV*.

***DFT** – Copy the data from the default instance, which doesn't contain a suffix.

***NONE** – Do not copy any data to the new version. The new version will be a clean installation.

Specify the product instance, or library suffix, containing the data to be copied to the new version. | | BSFX | Backup Library Suffix | Specifies the suffix to be appended to the libraries containing the version to be replaced by this installation, if a prior version exists for the instance defined in parameter ENV. This way, the prior version isn't lost and can be reactivated by renaming the libraries from the backup suffix to the instance suffix. | | BREP | Replace existing Backup Libs | Specifies if existing backup libraries using the same suffix as defined in parameter BSFX should be automatically replaced.

***YES** – Any existing libraries with the same name as the new Backup libraries will be automatically replaced.

***NO** – Libraries with the same names as the new Backup libraries will not be replaced, and the installation will not occur if any exist. | | ASPD | Install in ASP Device | Specifies the auxiliary storage pool (ASP) device to which the product libraries should be restored.

***SYSBAS** – The product libraries will be restored to the base system iASP.

character-value – the product libraries will be restored to the indicated iASP device. | | END | End Jobs locking Product Libs | Specifies if all jobs that have a lock on one of the product libraries should automatically be ended immediately.

***NO** – If a lock exists for one of the libraries, the installation process will be cancelled.

***YES** – Each job that has a lock on one of the product libraries will be ended immediately. If ending fails for a job, the installation process will be cancelled. | | OWN | Product Object Owner | Specifies the user profile to be used to own the objects in the MD product libraries and IFS folders. If the profile doesn't exist yet, the installer will create it. The profile should have ***ALLOBJ**, ***JOBCTL**, and ***SPLCTL** special authorities to ensure that deployments function correctly. |

Using MDCMS to Install MDRest4i

If using MDREST4INS to upgrade an existing product to a newer version, the distribution and installation can occur as part of an RFP using MDCMS. In this case ensure the following in MDCMS:

- MDREST4INS should be a Post-Installation (3) attribute command attached to the *IFS or *FILE attribute defined with the IFS path or save file library as the target object library.
- Run for Modifications = Y
- Keep MD Libs in Libl = N
- Frequency = R
- MDREST4INS command should begin with SBMJOB so that it runs separately from the RFP
- END(*YES) to terminate any locks
- DLY(30) to allow RFP time to finish before starting with installation
- USER for SBMJOB should be a profile with *SECADM authorit00
- INLLIBL for SBMJOB should include only QTEMP and QGPL (or library where MDREST4INS command exists)

Otherwise continue as per the following example command definition:

```
Appl.....: MD Run for Modifications: Y Y/N
Lvl......: 50 Recompiles...: N Y/N
Attribute: INSTALL Attribute, \*RFP Deletes......: N Y/N
Updates......: N Y/N
Type.....: 3 Post-Installation Ignore Errors........: Y Y/N
```

```
Sequence.: 1 Keep MD Libs in Libl.: N Y/N
Frequency: R O=Object, R=RFP
```

Command:

```
SBMJOB CMD(MDREST4INS PATH('##OBJLIB##') END(\*YES) DLY(30)) JOB(MRINSTALL) USER(QSO) INLLIBL(QTEMP QGPL)
```

Default API and Consumer logs folder

The examples provided by MDREST4i and the default generator options, store logs and uploaded files in folders on the IFS. These folders is created automatically by the installer. Each instance installed will add instance sub-folders to these folders.

The folders created are:

/MDREST4i/**logs**/<instances>
/MDREST4i/**attachments**/<instances>
/MDREST4i/**json**/<instances>
/MDREST4i/**sdk**/<instances>
/MDREST4i/**sdklogs**/<instances>

Installing or Updating the MDRest4i SDK

The MDRest4i SDK web interface runs as a web application served by an HTTP server on IBM i. The MDREST4i SDK web application uses REST APIs (Built using MDREST4i) which are exposed via the **same instance** of this Apache HTTP server.

To **install** SDK web UI application, run the MDRSDKINS command

To **update** an existing installation, first stop the MDRST server instance and any server instance created over MDRST for running your API's, and then run the MDRSDKUPD command.

Invoking the MDRSDKINS command

- `ADDLIBLE MDRST`
- Prompt `MDRSDKINS`

The following screen will appear:

```
                     Install MDRest4i SDK (MDRSDKINS)

Type choices, press Enter.

SDK HTTP PORT  . . . . . . . . . PORT           4545
Start SDK HTTP Instance  . . . . STRSRV         Y
Default SDK Administrator  . . . ADMPRF       > DFTPROFILE
Default Generator Library  . . . DFTLIB         MDRAPIDFT
Default Hostname or IP . . . . . DFTHOST        'ibmi.yourdomain.com'

Create Default API Server  . . . CRTSRVA        Y
Default API Server . . . . . . . APISRV         MDRAPISRV
Default API Server PORT  . . . . PORTA          4546


                                                                        Bottom
 F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel
```

MDRSDKINS Parameter Table

PORT	HTTP Instance PORT Number	Port number the HTTP server instance for the SDK web UI runs under. 4545 is a default value and should be set as per your own port preferences on the IBM i server
STRSRV	Start HTTP Instance	Select **Y** to start the SDK HTTP server instance.
ADMPRF	Default SDK Administrator	MDRest4i SDK uses hierarchal authority classes for users. An administrator user must be created when installing MDRest4i SDK, so additional users can be authorized. The value supplied in this parameter MUST be a valid, existing IBM i user, that can login to the IBM i. No special IBM i authority is required for this user. This default Admin user has authority to Manage SDK Users, including authorizing new users.
DFTLIB	Default Generator Library	A default library to contain generated Provider and Consumer programs and their source. MDRSDKINS creates this library if it does not exist. *DFTLIB* is used when creating the **Default API Server** below as one of the libraries in the JOBD for this default API Server. *DFTLIB* is also used for setting default values, for default the SDK user profile. See Default User MDRDFTUSR below for more details.
DFTHOST	Default Hostname or IP	This is the host name or IP address of the IBM i server, where MDRest4i Providers will run from. It is used to to add the DFT Server value for new users. See Default User MDRDFTUSR below for more details.
CRTSRVA	Create Default API Server	Specify **Y** to create an HTTP instance to host REST Providers created by MDRest4i SDK. MDRSDKINS calls the MDRHTTPAPI command to create this HTTP server instance. MDRSDKINS will start the Default API Server after installation completes.
APISRV	Default API Server	The name of the HTTP instance created as a default Provider(API) server. A job description of the same name will also be created in MDRST, and used by this HTTP instance.
PORTA	Default API Server PORT	Port number that the Default API Server created above will listen on.

Default User MDRDFTUSR

A dummy user: **MDRDFTUSR** is created by MDRSDKINS.

When a new SDK user signs up using the Signup link in the SDK Login page, this MDRDFTUSR record is copied to create the new user profile record, in file MDRST/MDRDUSER.

Default values are set in this dummy profile by MDRSDKINS. Individual users default values can be edited later from the Edit User screen in the SDK UI. The default values set by MDRSDKINS are:

- The default value for Object Library is set with the **DFTLIB** value.
- The default Source File for code generation is set as *MDRDFTSRC* using the library specified in **DFTLIB** above. For example: DFTLIB = MYAPILIB. The default source file value will be set as `/QSYS.LIB/MYAPILIB.LIB/MDRDFTSRC.FILE`
- The default Server URL value added to the OAPI/SWAGGER definition when a new Provider or Consumer is created in the SDK UI.

See the SDK Edit Profile for editing the default values set for a user during Signup.

The default values for *MDRDFTUSR* can be edited at a later date, using the MDRSDKUSR command.

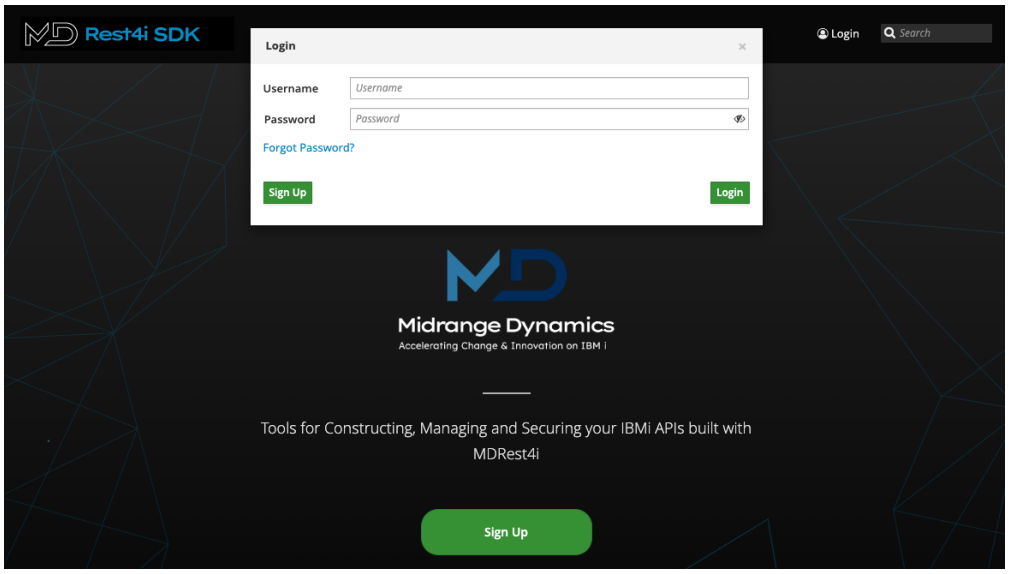
> **Note**
>
> The Owner of the SDK web UI objects in the IFS is set the same owner used to install the product via the MDREST4INS command.

Running MDRest4i SDK Web UI

Use the following url to connect to the web UI

HTTP://youribmiserver:yourport/cons/

This will bring up the following interface:



Login with *user profile provided in the MDRSDKIN.ADMPRF command parameter above*

Once logged in, click on any “?” in the UI to bring up the online help

Updating the SDK Web Application

The **MDRSDKUPD** command has no parameters. It only executes if the SDK has already been installed. It backs up the existing config files for the MDRest4i SDK web UI, installs the new code base, and copies back the SDK web UI config files.

> **Warning**
>
> If unsure if the SDK has already been installed, check the value of data area **MDRST/MDRSDKROOT**. If this is blank, then command MDRSDKINS *MUST* be run, not MDRSDKUPD

MDRSDKUPD backs up the SDK web UI application to IFS folder: */mdrest4i/sdkbu-mdrst*

> **Note**
>
> The backup path is based uon the instance name for MDRest4i. If the MDRest4i instance is 'V14' for example, the backup folder for te SDK update will be: */mdrest4i/sdkbu-mdrst*

To run command MDRSDKUPD:

- `ADDLIBLE MDRST`
- `MDRSDKUPD` command and enter

API Server Setup

CREATING AN HTTP SERVER FOR YOUR API'S

REST APIs are exposed via an IBM i Apache server instance. By default command MDRSDKINS installs a default server for your Provider and Consumer programs.

> **Note**
>
> The default API server created by the SDK installation can be used as necessary. Addiotnal API servers can easily be installed for production, or other testing environments. There is virtually no limit to the number of API servers that can be added in the same MDRest4i instance. The job description and configuration required in these instances, can be created on the IBM i using the command **MDRHTTPAPI**

1.2.3 Troubleshooting

Troubleshooting the HTTP Servers

Please Note: In the sections below, if you are using any instance name other than default for MDRest4i, MDRST should be replaced with MDRSTxxxx where, xxxx is the instance name

If the SDK web UI wont load, or an API wont respond, it may be caused by a server configuration problem.

Please check the following areas.

IS THE SERVER RUNNING

Use the following command to see if the server jobs are started:

```
WRKACTJOB SBS(QHTTPSVR) JOB(MDRST)
```

where "*MDRST*" is the library where MDrest4i MDRFRAME is installed.

If it isn't running, use the following command to start the server

```
STRTCPSVR SERVER(*HTTP) HTTPSVR(MDRST)
```

Then check to see if the jobs are active. If they are test the web ui again

SDK SERVER CONFIG - HTTPD.CONF

If the server wont start of you cannot connect it may be the server configuration.

View the HTTP config file here: /www/mdrst/conf/httpd.conf file (either from the HTTPAdmin console or the IFS in 5250 screens/ VSCode). It should look like this:

```
# MDRest4i: Created by MDRHTTPSDK on 2024-02-28-16.43.16
Listen *:2519
DocumentRoot /www/mdrst/htdocs
TraceEnable Off
LogFormat "%h %T %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%{Cookie}n \"%r\" %t" cookie
LogFormat "%{User-agent}i" agent
LogFormat "%{Referer}i -> %U" referer
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access_log combined
LogMaint logs/access_log 30 0
LogMaint logs/error_log 30 0
SetEnvIf "User-Agent" "Mozilla/2" nokeepalive
SetEnvIf "User-Agent" "JDK/1\.0" force-response-1.0
SetEnvIf "User-Agent" "Java/1\.0" force-response-1.0
SetEnvIf "User-Agent" "RealPlayer 4\.0" force-response-1.0
SetEnvIf "User-Agent" "MSIE 4\.0b2;" nokeepalive
SetEnvIf "User-Agent" "MSIE 4\.0b2;" force-response-1.0
<Directory />
 Require all denied
</Directory>
<Directory /www/mdrst/htdocs>
  Require all denied
</Directory>

### MDRest4i: Config Start
HTTPStartJobDesc mdrst/mdrst
SetEnv QIBM_CGI_CHANGE_CURDIR N
SetEnv QIBM_CGI_CHANGE_CURLIB N
## MDREST4i: The line below allows authorization tokens to be passed through to the CGI program
SetEnvIf Authorization "(.*)" HTTP_AUTHORIZATION=$1

## MDRest4i: CORS header settings
Header Set Access-Control-Allow-Origin "*"
Header Set Access-Control-Allow-Headers "AUTHORIZATION,ORIGIN,CONTENT-TYPE,ACCEPT"
Header set Access-Control-Allow-Methods "POST,GET,OPTIONS,DELETE,PUT"

## MDRest4i: API Script Alias Settings
ScriptAliasMatch mdrsdk /QSYS.LIB/mdrst.LIB/MDRSDK.PGM
ScriptAliasMatch mdrapi /QSYS.LIB/mdrst.LIB/MDRAPI.PGM

## MDRest4i: SDK WEB UI Alias Settings
AliasMatch /wiki/(.*) /www/mdrst/wiki/$1
AliasMatch /cons/(.*) /www/mdrst/cons/$1
AliasMatch /docu/(.*) /www/mdrst/docu/$1

## MDRest4i: REST API Library
<Directory /QSYS.LIB/mdrst.LIB>
```

```
 Require all granted
 SetEnv MDREST4I_RESOURCE_COMPONENT 3
 SetEnv MDREST4I_SDK_CONFIG_PATH "/www/mdrst/cons/mdrsdk_Templates.json"
 SetEnv MDREST4I_SDK_INITIALIZE_DEFAULTS *NO
 DefaultNetCCSID 1208
 CgiConvMode binary
 ServerUserID mdowner
</Directory>

## MDRest4i: Console Config
<Directory /www/mdrst/cons>
 <FilesMatch "index.html|startup.json|build.json">
  FileEtag None
  Header Unset ETag
  Header Set Cache-Control "max-age=0, no-store, no-cache, must-revalidate"
  Header Set Pragma "no-cache"
  Header Set Expires "Thu, 1 Jan 1970 00:00:00 GMT"
 </FilesMatch>
 Header set Access-Control-Allow-Origin "*"
 Options +FollowSymLinks -MultiViews
 DirectoryIndex index.html
 RewriteEngine On
 RewriteRule mdrconsoleapp_(.*)\.js mdrconsoleapp.js
 RewriteCond %{REQUEST_FILENAME} !-f
 RewriteRule artefacts/TokenManager/tokenapp_(.*)\.js artefacts/TokenManager/tokenapp.js
 RewriteCond %{REQUEST_FILENAME} !-f
 RewriteRule ^ index.html [QSA,L]
 Require all granted
 ServerUserID mdowner
</Directory>

## MDRest4i: Documenter Config
<Directory /www/mdrst/docu>
 <FilesMatch "index.html|startup.json|build.json">
  FileEtag None
  Header Unset ETag
  Header Set Cache-Control "max-age=0, no-store, no-cache, must-revalidate"
  Header Set Pragma "no-cache"
  Header Set Expires "Thu, 1 Jan 1970 00:00:00 GMT"
 </FilesMatch>
 Header set Access-Control-Allow-Origin "*"
 Options FollowSymLinks MultiViews
 DirectoryIndex index.html
 RewriteEngine On
 RewriteRule mdrdocumenterapp_(.*)\.js mdrdocumenterapp.js
 RewriteCond %{REQUEST_FILENAME} !-f
 RewriteRule ^ index.html [QSA,L]
 Require all granted
 ServerUserID mdowner
</Directory>

## MDRest4i: Wiki Docs
<Directory /www/mdrst/wiki>
 Require all granted
 ServerUserID mdowner
</Directory>
### MDRest4i: Config End
```

HTTP SERVER LOGS

Navigate to /www/mdrst/logs and view the contents of the error and/or access logs.

SPOOL FILES FOR THE SERVER JOB

If there is a crash or server startup problem, it may generate a spool file which can be found here:

`WRKSPLF SELECT(QTMHHTTP *ALL *ALL MDRSTT14)`

HTTP SERVER JOB DESCRIPTION

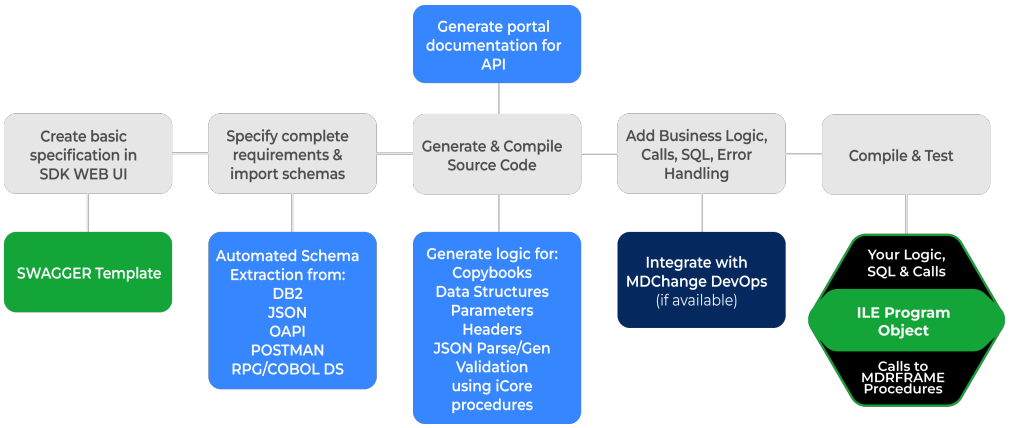
In the httpd.conf configuration file there is a directive: `HTTPStartJobDesc mdrst/mdrst`

Look at the library list part of this job description, to ensure that **MDRST** (or the library name **MDRSTxxx** consistent with the instance installed) is in the library list of this JOBD. `DSPJOBD JOBD(MDRST/MDRST)`

1.3 Concepts & Overviews

1.3.1 MDRest4i 14 Development Overview

THE MDREST4I DEVELOPMENT PROCESS



The MDRest4i SDK uses templates to create a basic OAPI (SWAGGER) specification.

The user then adds details according to requirements in the form based SDK GUI.

The SDK extracts OAPI schemas for payloads from many different types of source information(DDS).

The user then assigns these schemas to request or response payloads, and generates the RPG or COBOL code.

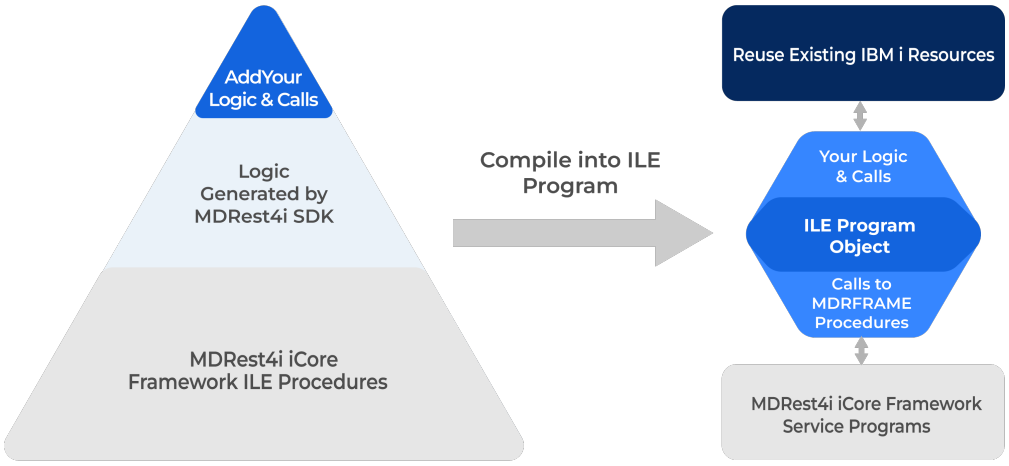
MDChange DevOPs requests can be triggered at this stage if available.

The user then manually adds their own logic to the generated programs to handle getting data to and from payloads, calls to existing assets, business logic and any additional messages. The MDRFRAME ILE framework functions can be called to customize, or add to the data and flow of the generated code.

Compile and test as normal.

THE MDREST4I CONCEPT

MDRest4i is a combination of ILE Framework and automation tools, that simplify REST API and REST client development on IBM i.

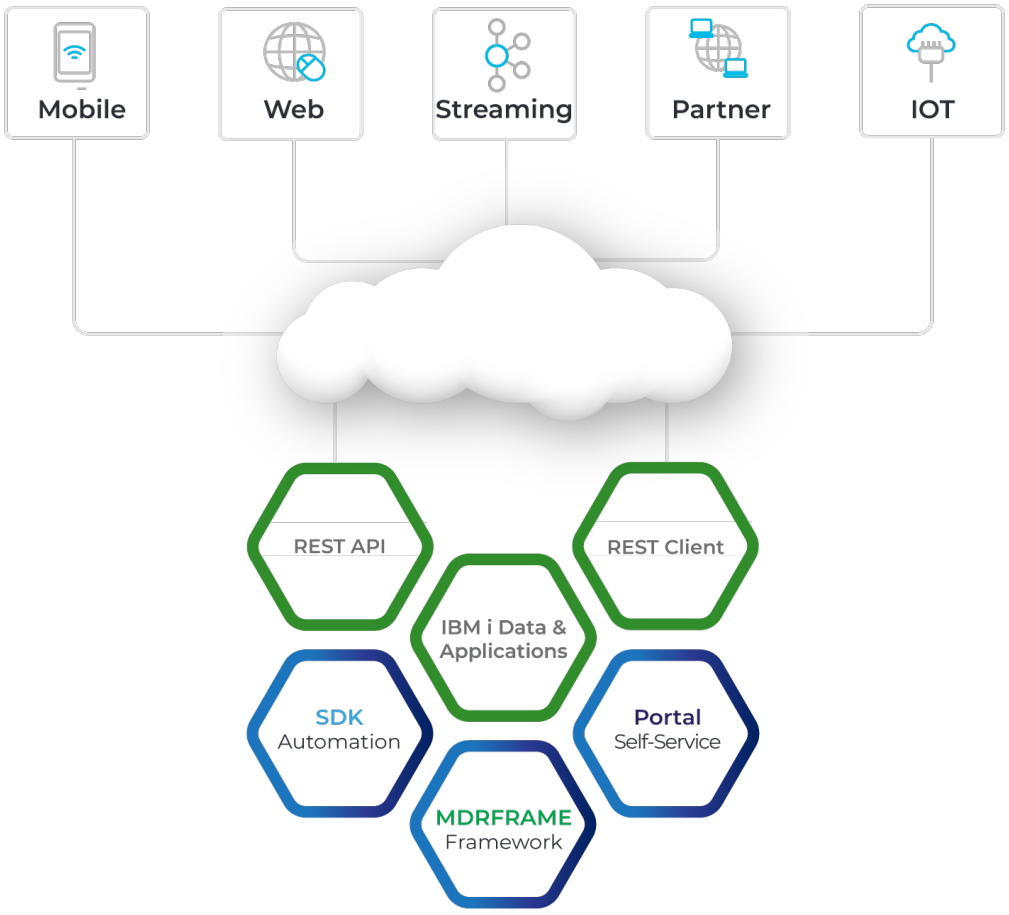


MDREST4I COMPONENTS

MDRFRAME Framework - a native IBM i ILE framework for building REST API's and REST Clients in any ILE language

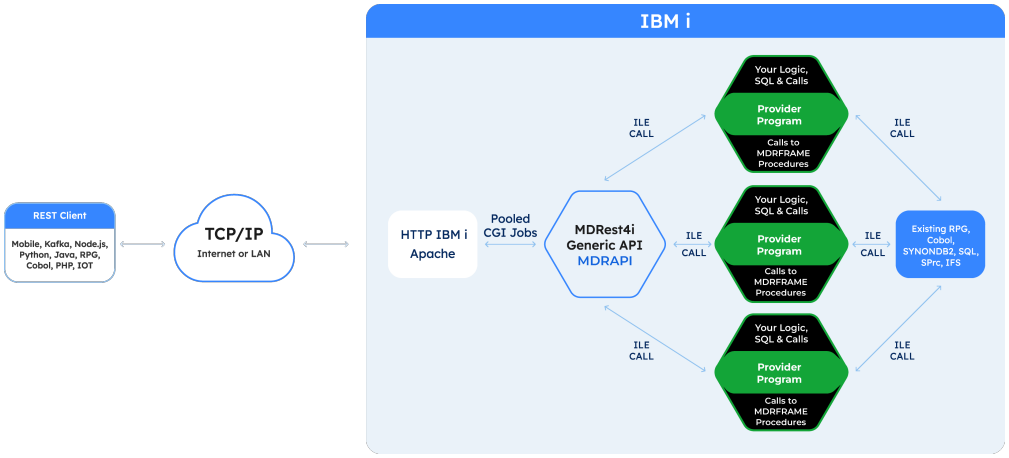
SDK - a Software Development Kit, that automates the generation of REST API and Rest Client Programs that use the MDRFRAME framework

Portal - a generator, editor and web portal for REST API documentation



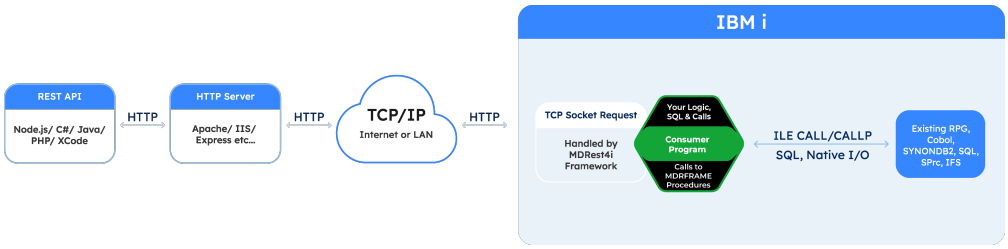
REST API RUNTIME ARCHITECTURE

REST API's are also referred to as *Providers*. More details available in the REST API-Provider section.



REST CLIENT RUNTIME ARCHITECTURE

REST Clients are also referred to as *Consumers*. More details available in the REST CLIENT-Consumer section.



USEFUL SHORTCUTS:

Provider Overview

Consumer Overview

Copybooks

Coding in COBOL

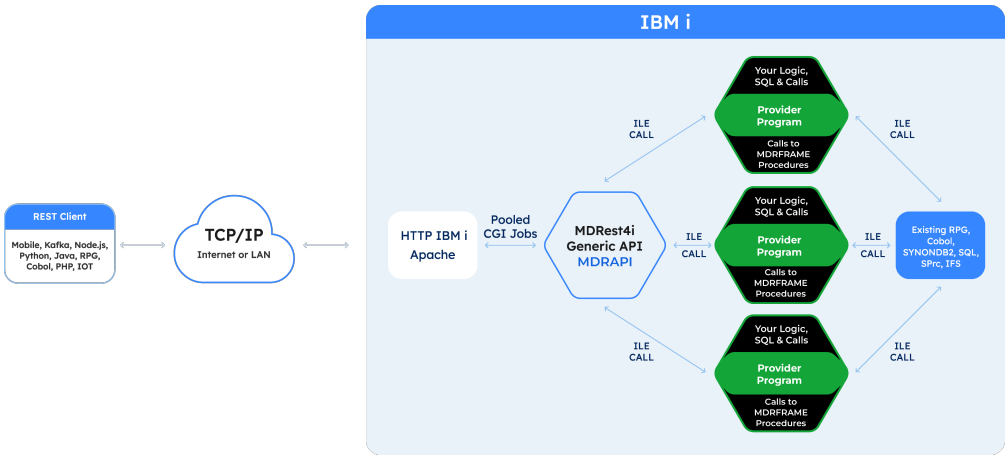
Create HTTP API Server

Troubleshooting

1.3.2 REST API - Provider

The common used term for a REST API is a *Provider*. A REST Client is referred to as a *Consumer*.

RUNTIME ARCHITECTURE



The runtime architecture a REST API/Provider is self contained on the IBM i using native components.

An HTTP server is created using the MDRST/MDRHTTPAPI command. This server instance is configured to receive requests from a REST client and call the MDRAPI controller program.

The Provider program is the program generated by the SDK and is called by MDRAPI.

MDRAPI

MDRAPI is an ILE program. It is called as a CGi program by the HTTP server. Its roll is to control the flow of data between the HTTP server and the REST API program generated by the SDK.

MDRAPI allocates a unique "handle", gathers all the inbound message data from a request, and sets this up in memory using pointers keyed by the "handle" for each request.

It then calls the REST API program(written by the developer), passing some of the inbound data(handle, method, body) in the initial call. The REST API program, then manipulates the data in memory using calls to the MDRFRAME service program procedures, passing the "handle" passed in by MDRAPI each time.

The REST API program then passes control back to MDRAPI, and MDRAPI sends the response back to the REST Client request, via the HTTP server.

MDRAPI release any resource allocation created with the initial "handle".

What API Program to call, and what, and hwo to log each request is controlled by MDRAPI, as defined in the MDRDCFG database described in MDRST/MDRSTCFG section.

Info

It is imprtant therefore to either explicitly specify "mdrapi" in the uri to invoke a REST API in V14. In both cases an httpd.conf server directive must be set ensure MDRAPI is called. For example:

```
ScriptAliasMatch mdrapi /QSYS.LIB/MDRSTV14T.LIB/MDRAPI.PGM

ScriptAliasMatch ^/MDCMST86/AZURE/RELEASE/CALLBACK /QSYS.LIB/MDCMST86.LIB/MDRAPI.PGM
```

PROVIDER PROGRAM

MDRAPI calls the provider program passing three parameters:

Parameter	Description
handle	unique threadsafe memory key used by MDRFRAME to allocate and deallocate memory structures
method	the HTTP method of the request
body	the HTTP message request body (if applicable)

The general flow of the provider program is:

- Handle the extraction of inbound of REST headers, parameters, attachments, parsing payloads etc.
- Call or run your business logic, DB/IFS IO, and calls to IBM i applications/Stored procedures etc.
- Set any error messages.
- Build REST headers and payloads.
- Set the HTTP status.
- Return control to MDRAPI which sends the response via the HTTP server to the REST Client.

This data extraction of the request, and writing of the response is executed by calling the appropriate MDRFRAME procedures (found in the MDRFRAME service program).

EXAMPLES

Below is an example of a simple REST Provider program that uses a POST method. The source of these examples can be found in MDRST/EXAMPLES.

Info

In the COBOL example below the MDR-DATAGEN SECTION is used to generate JSON. This function is unique to MDRFRAME, and overcomes the limitation that IBM's DATA-GEN procedure is only available in RPGLE or from V7R2 upwards. The Copybook "*MAPIC001C*" below shows how the MDR_DATAGEN and MDR_DATAINTO functions use a special '*schema*' to parse or generate JSON via the MDRFRAME framework. The MDR_DATAINTO-MDR_DATAGEN section has the details about how to use this.

RPGLE **COBOL** **COBOL COPY MAPIC001C**

```
**free
// CRTBNDRPG PGM(&O/&ON) SRCFILE(&L/&F) OPTION(*EVENTF) DBGVIEW(*ALL) -
// USRPRF(*OWNER) TGTRLS(V7R2M0)

// RPGLE REST API Template Using DATA-GEN and DATA-INTO for JSON.
ctl-opt dftactgrp(*no) actgrp(*new);
ctl-opt BNDDIR('MDRFRAME');

/copy mdrframe

dcl-ds output qualified;
message char(50); // example only : change as required
end-ds;

dcl-ds input qualified;
message char(50); // example only : change as required
end-ds;

dcl-pi *n;
handle    like(MDR_Handle_t);
method    char(32) const;
body      varchar(500000); // MAxSize: 16000000
end-pi;

dcl-s result varchar(1000);

// Logic to process request body
MDR_genParseOptions(handle: 'document_name=input');
data-into input %data('':'')
               %parser('MDRFRAME(PARSER)':handle);

eval-corr output = input;
// Logic to process response
MDR_genParseOptions(handle: 'document_name=output');
data-gen output %data(result: '')
               %gen('MDRFRAME(GENERATOR)':handle);

*Inlr = *On;

PROCESS varchar
        apost
        nomonoprc
        nosync
        nostdtrunc

IDENTIFICATION DIVISION.
PROGRAM-ID.   MAPIC001.
AUTHOR.       Midrange Dynamics.

*>  CRTCBLMOD MODULE(&O/&ON) SRCFILE(&L/&F) DBGVIEW(&DV) -
*>            OPTION(*SOURCE *EVENTF *IMBEDERR)
*>  CRTPGM PGM(&O/&ON) MODULE(&O/&ON) BNDDIR(MDRFRAME) -
*>         ACTGRP(*NEW)
*****************************************************************
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-I.
OBJECT-COMPUTER. IBM-I.

SPECIAL-NAMES.
* MDRest4i Special Names for MDRFRAME Procedures
* Please insert any program specific SEPCIAL NAMES BEFORE this
* COPYBOOK, and the COPYBOOK delimits the divsion with a period
COPY MDRCBLSPC OF QLBLSRC.

*****************************************************************
DATA DIVISION.
*****************************************************************
WORKING-STORAGE SECTION.
* MDRest4i REQ/RSP Payload Schemas
COPY MAPIC001C OF QLBLSRC.
* MDRest4i Framework Variables
COPY MDRCBLWSC OF QLBLSRC.
* MDRest4i Large Framework Variables
COPY MDRCBLWSVS OF QLBLSRC.

* ===============================================================
LINKAGE SECTION.
COPY MDRCBLLSC OF QLBLSRC.

* MDR-HTTP-METHOD - To recive supplied HTTP method
*****************************************************************
01  MDR-HTTP-METHOD      PIC X(10).

* MDR-INPUT-DATA - To recive input JSON request.
*****************************************************************
01  MDR-INPUT-DATA.
    10 MDR-INPUT-DATA-LEN      PIC 9(9) USAGE BINARY.
    10 MDR-INPUT-DATA-BUFFER   PIC X(5000000).

* MDR-BODY-TYPE - To recive Body type ( *CAR or *VARCHAR)
*****************************************************************
01  MDR-BODY-TYPE              PIC X(10).

PROCEDURE DIVISION USING MDR-HANDLE
                         MDR-HTTP-METHOD
                         MDR-INPUT-DATA
                         MDR-BODY-TYPE.

0000-MAIN-CONTROL SECTION.

BEG.
```