SAP Cloud Application Programming Model (CAPM or simply CAP) is a framework to build enterprise-grade services and applications efficiently. It allows developers to use different programming models like Node.js or Java to build applications that can run on the SAP Business Technology Platform (BTP).

SAP CAPM is a framework of languages (Java and Node.js), libraries (CDS-Core Data Services, Developments Kits), and tools (Visual Studio, Business Application Studio). It is an opinionated and open framework to build the application efficiently. It is an emerging technology in the SAP cloud for creating scalable multi-target applications which could easily translate and run on different platforms.

Here's a list of commonly used SAP **CAPM** (Cloud Application Programming Model) commands:

```
COMMANDS
  i  init          jump-start cds-based projects
  a  add           add a feature to an existing project
  v  bind          bind application to remote services
  m  import        add models from external sources
  c  compile       compile cds models to different outputs
  p  parse         parses given cds models
  s  serve         run your services in local server
  w  watch         run and restart on file changes
  r  mock          call cds serve with mocked service
  r  repl          read-eval-event loop
  e  env           inspect effective configuration
  b  build         prepare for deployment
  d  deploy        deploy to databases or cloud
  u  subscribe     subscribe a tenant to a multitenant SaaS app
  u  unsubscribe   unsubscribe a tenant from a multitenant SaaS app
  l  login         login to extensible multitenant SaaS app
  l  logout        logout from extensible multitenant SaaS app
  p  pull          pull base model of extensible SaaS app
  p  push          push extension to extensible SaaS app
  t  lint          [beta] run linter for env or model checks
  v  version       get detailed version information
  ?  help          get detailed usage information
```

CAP CLI Commands:

1. Initialize a New CAP Project:
   - cds init <project-name>
2. Add a Service:
   - cds add srv --name <service-name>
3. Add a Database:
   - cds add db
4. Add a UI Module:
   - cds init <project-name>
5. Run the Application:
   - cds watch
6. Deploy to Database (SQLite by default):
   - cds deploy --to sqlite
7. Deploy to SAP HANA:
   - cds deploy --to hana --auto-undeploy
8. Build the Project:
   - cds build
9. Add a CDS Model:
   - cds add model
10. Add an External Service:
    - cds add external <service-name>
11. Create a CDS Schema:
    - cds add schema
12. Install Required Dependencies:
    - npm install
13. Test the Application:
    - npm test

SAP BTP (Cloud Foundry) Commands:

1. Login to BTP CLI:
   - cf login
2. Create a Service Instance:
   - cf create-service <service-name> <plan> <instance-name>
3. Bind a Service Instance to an App:
   - cf bind-service <app-name> <instance-name>
4. Push the App to SAP BTP:
   - cf push <app-name>

NAIDU KARRI
mr_loyal_ml | +91 8247262816
karrinaidunk@gmail.com

CDS Syntax Commands:

1. Define an Entity:
    - entity <entity-name> { ... }
2. Define a Service:
    - service <service-name> { ... }
3. Create Associations:
    - entity <entity-name> {

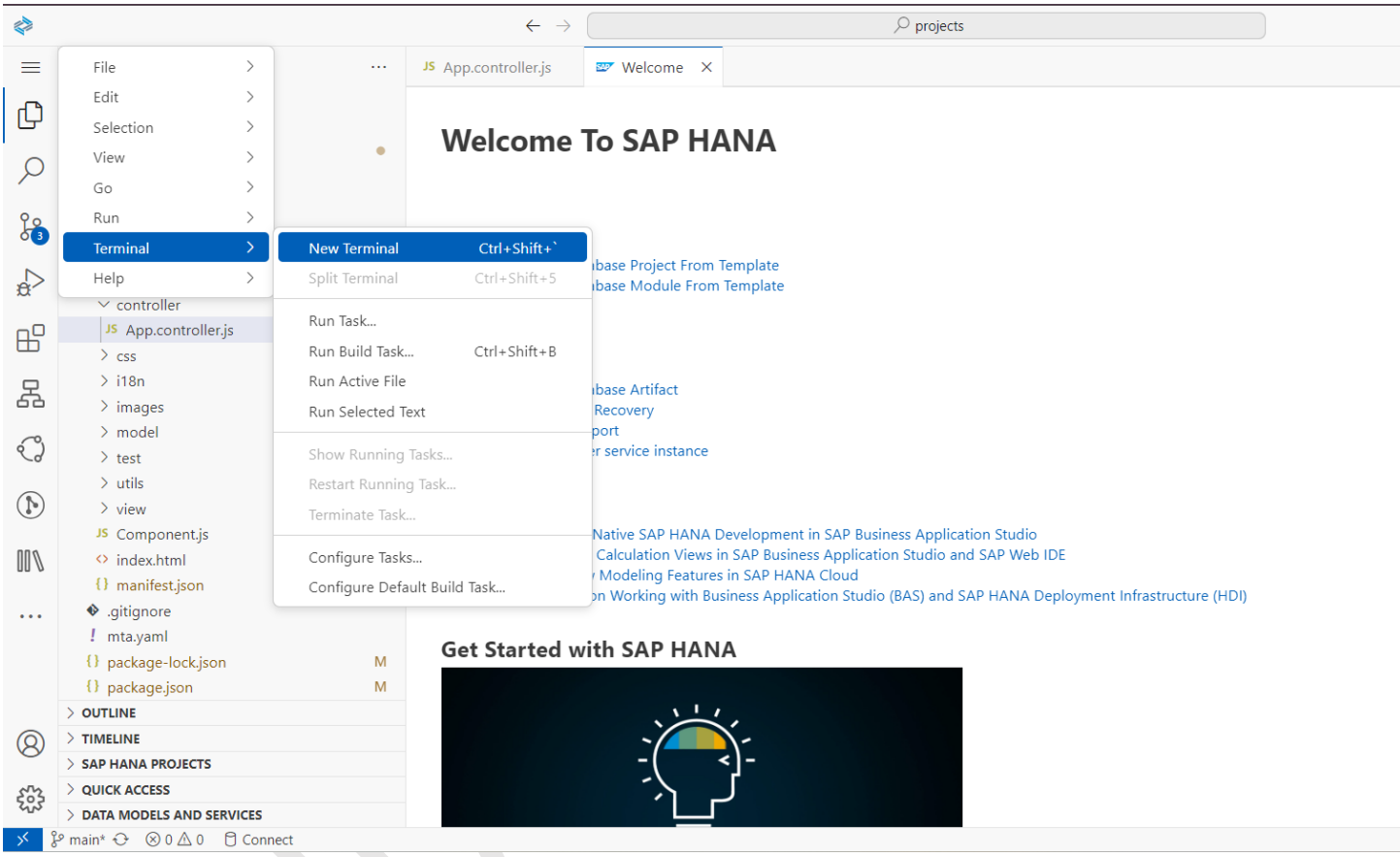
      assocName : Association to <target-entity>;

      }
4. Custom Event Handler:
    - this.on('<event>', '<entity>', async (req) => { ... });

CDS (Core Data Services):

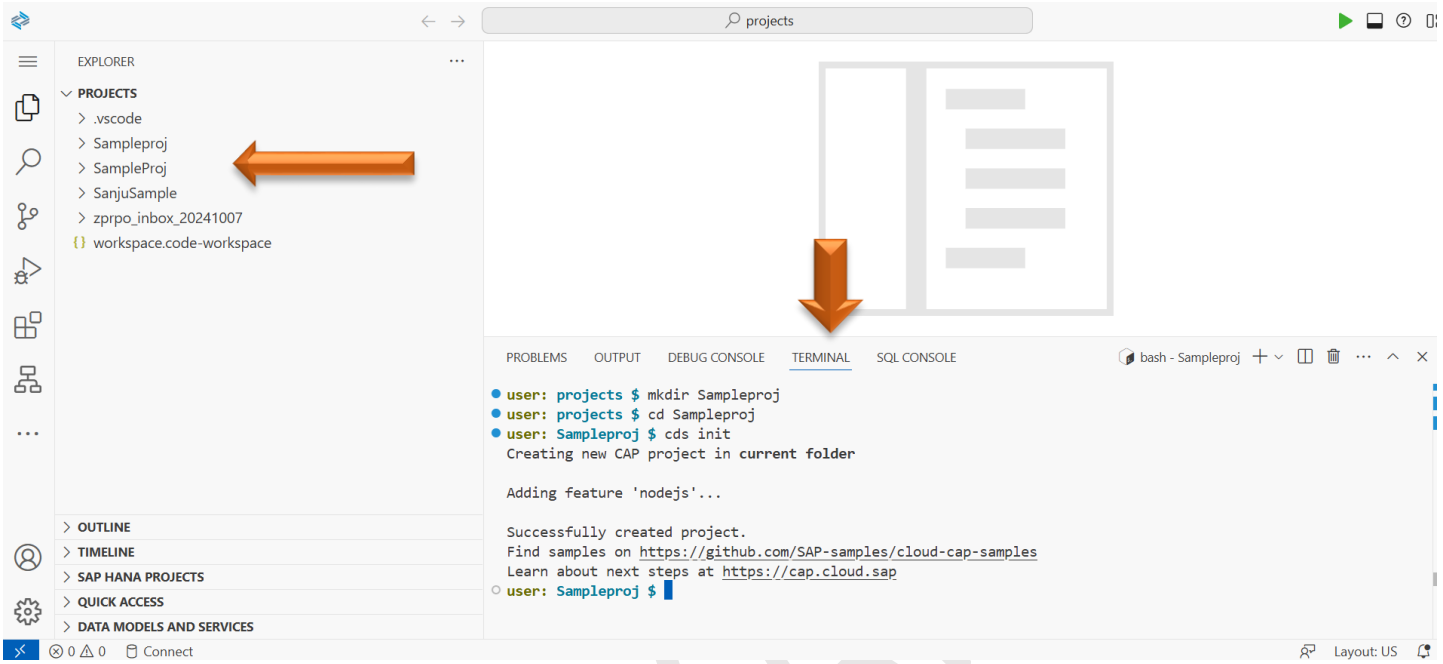
CDS is the backbone of the SAP Cloud Application Programming Model (CAP). It provides the means to declaratively capture service definitions and data models, queries, and expressions in plain (JavaScript) object notations. CDS features to parse from a variety of source languages and to compile them into various target languages.

Open Business Application Studio (BAS), Go to Terminal and click on new Terminal.

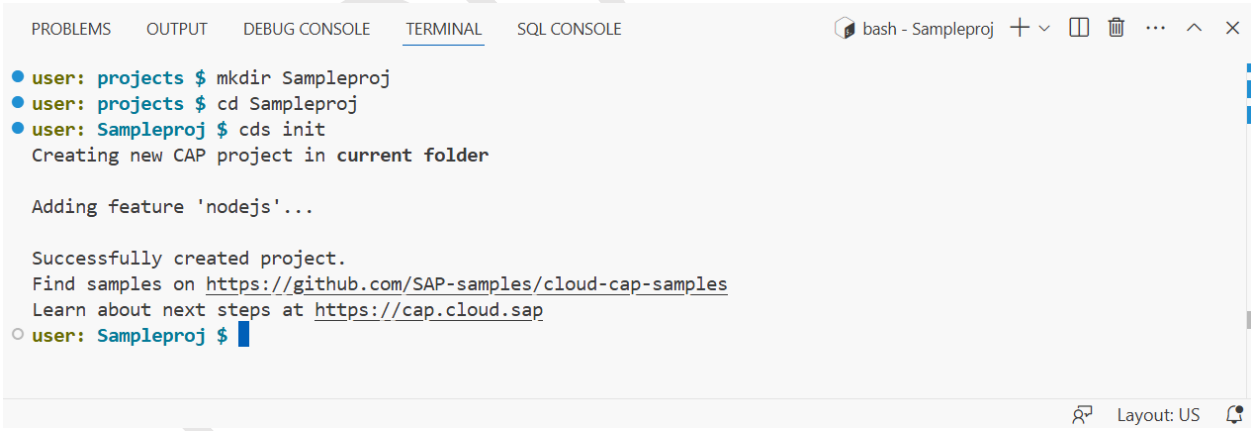


You will get the below terminal screen. Now copy the below command & run it in your terminal. This will create a Project with the name: **CAP-PROJ**.

I used different project name for continuation of the documentation **SampleProj** instead of **CapmProject**.

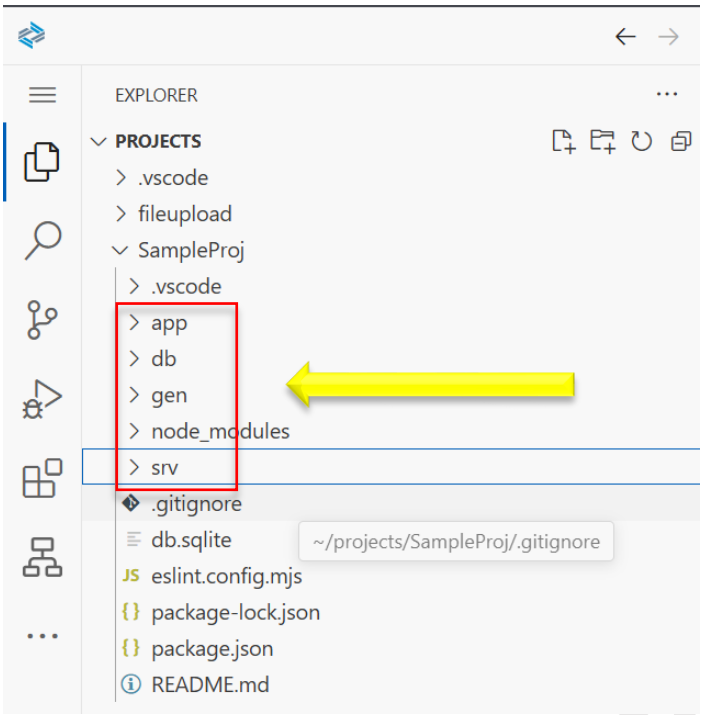


These commands are used to build application.



Now, you have your project ready

NAIDU KARRI
mr_loyal_ml | +91 8247262816
karrinaidunk@gmail.com



As you can see we have 3 major folders

- app
- db
- srv

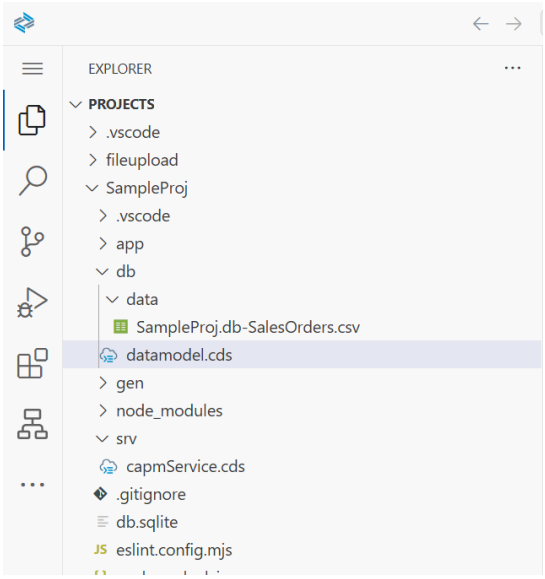
We will work on them one by one & get to know the functionality.

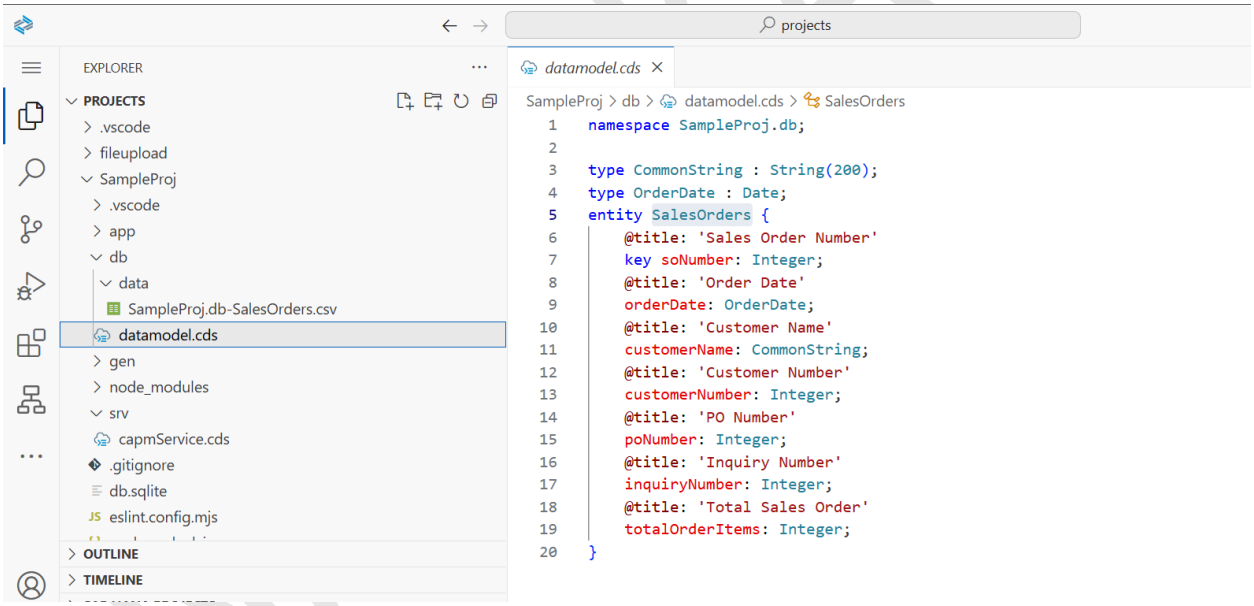
Step 1: Create Database Entity

Let's start with the **DB** Folder: It contains the Database Entity which is based on CDS Model. The design-time artefacts declared in this file will be converted to run-time, physical artefacts in the database. In this example, the entities will become tables.

Right Click on ***db*** folder & create a *New File*.

Name the file as ***datamodel.cds***



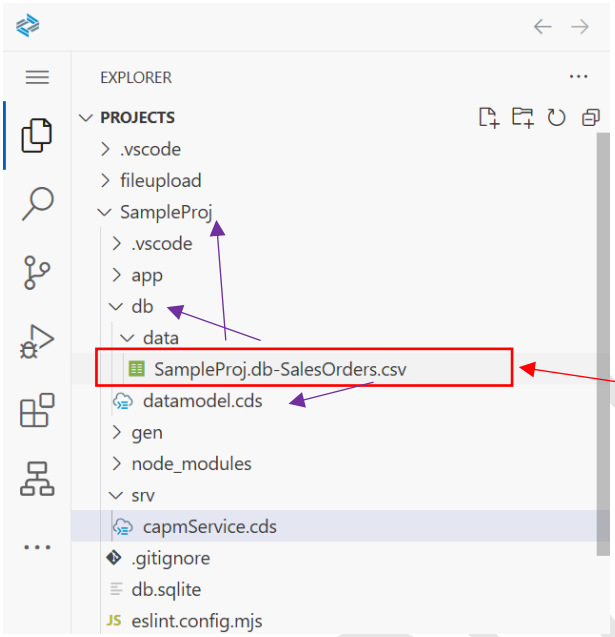


We have created an Entity with the name ***SalesOrders***, which have 7 properties. It will create a table with 7 fields. Here ***key*** denotes the Primary key of the table ***@title*** is an annotation that we are using to define the text for our table properties.

Step 2: Create Service

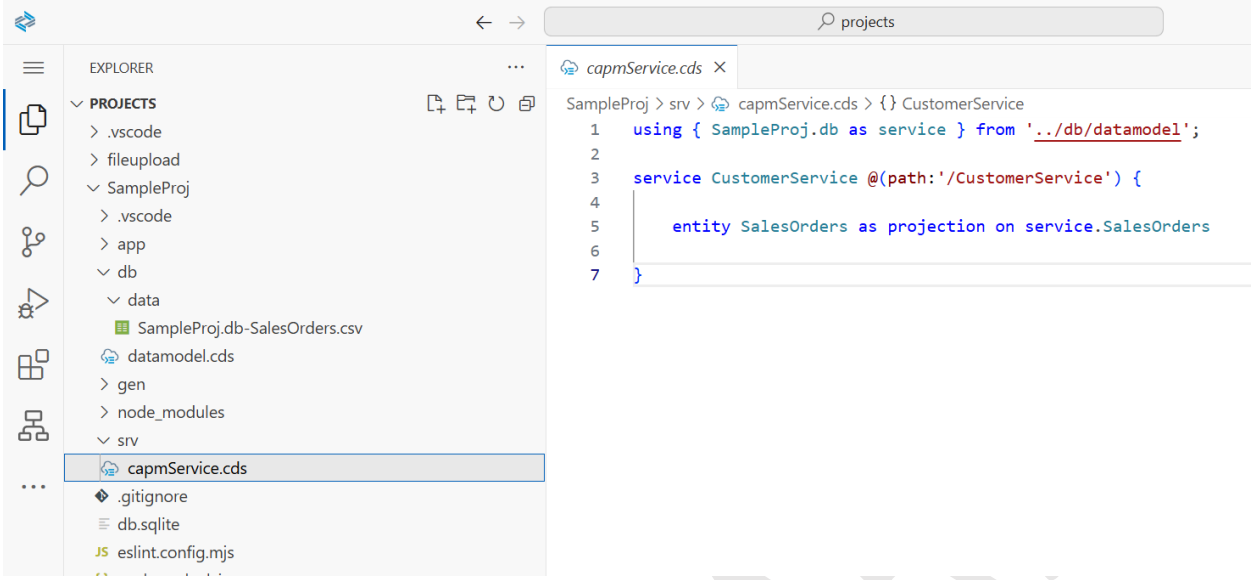
Now go to the ***srv*** folder. We will be declaring services to expose the database entities we declared in the previous step.

Create a new file (**capmService.cds**) in ***srv*** folder by right-clicking on the same.



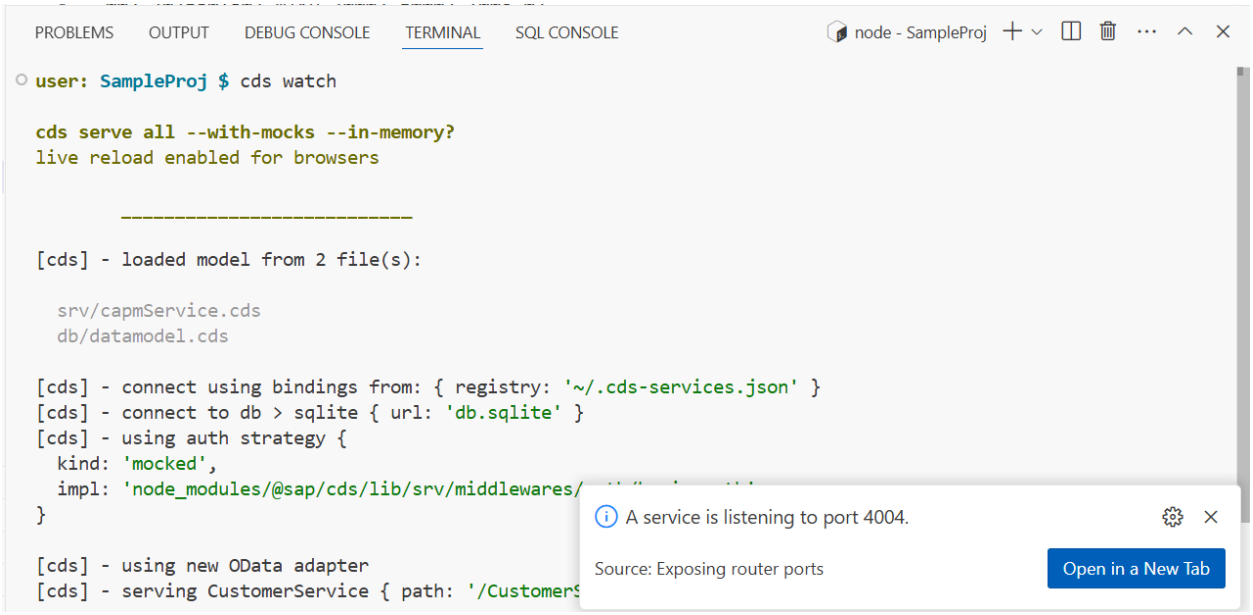
Namespace Important-
ProjectName.db-EntityName.csv

we have created a CAP service that will expose our entity **SalesOrders** and add below in **Package.json**

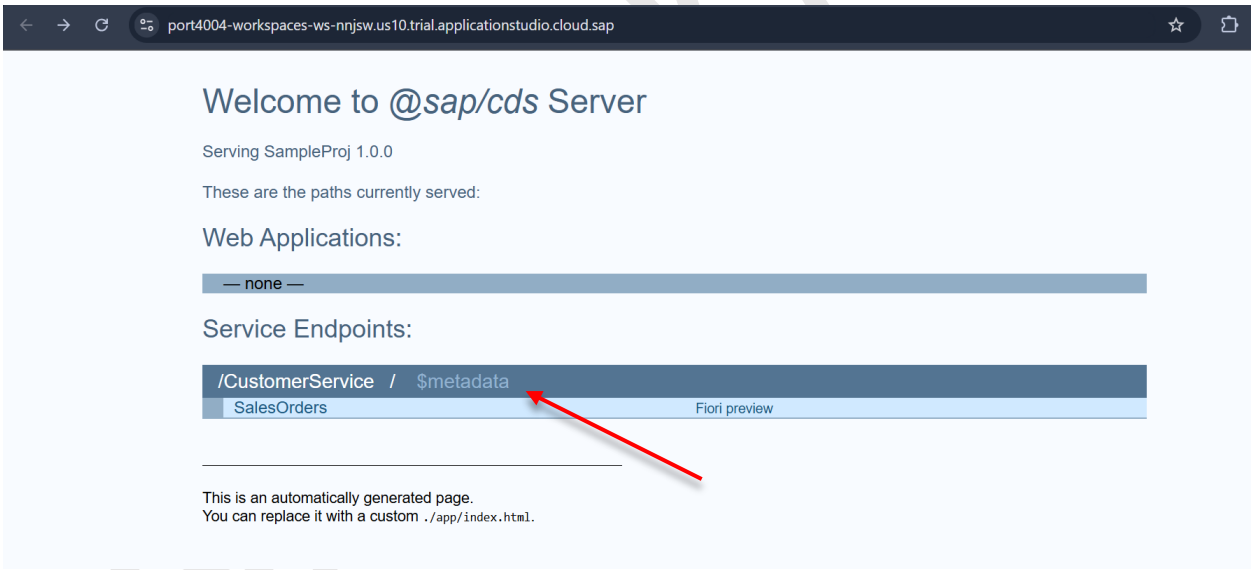


Let's build the project now once. For that do ***cds build***

Now Preview the Project **cds watch**

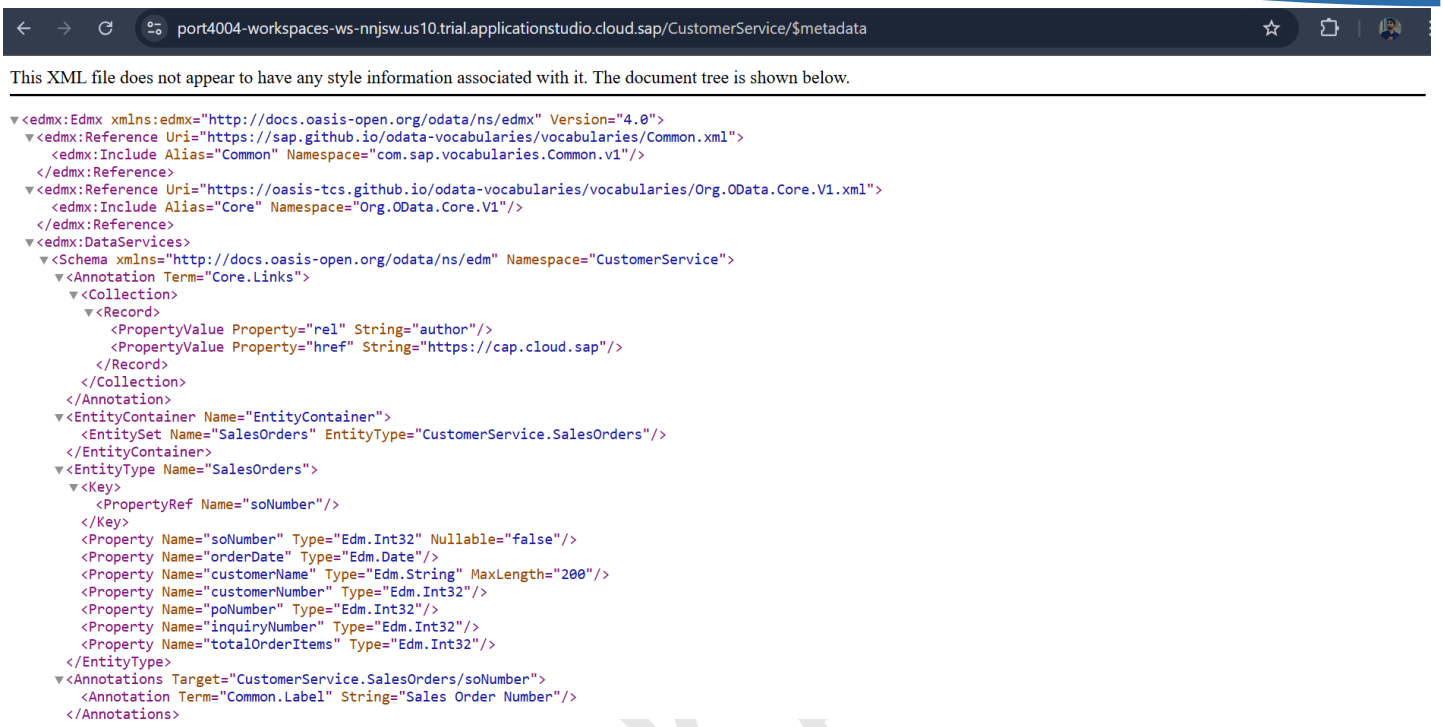


You will get navigated to new Tab and display the screen something like below

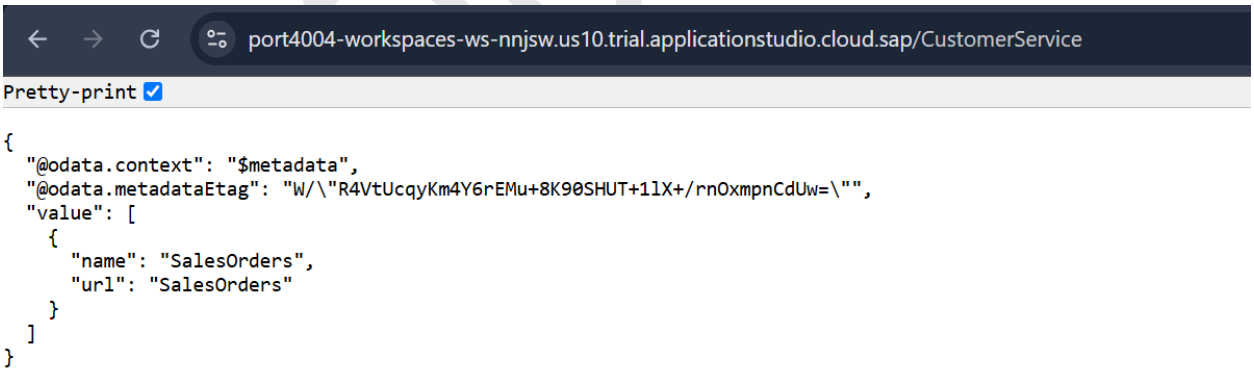


Now, you can click on ***$metadata*** to check the Metadata / Structure of the Exposed OData Service.

NAIDU KARRI
mr_loyal_ml | +91 8247262816
karrinaidunk@gmail.com



Now, let's click on the **SalesOrders Entity**.

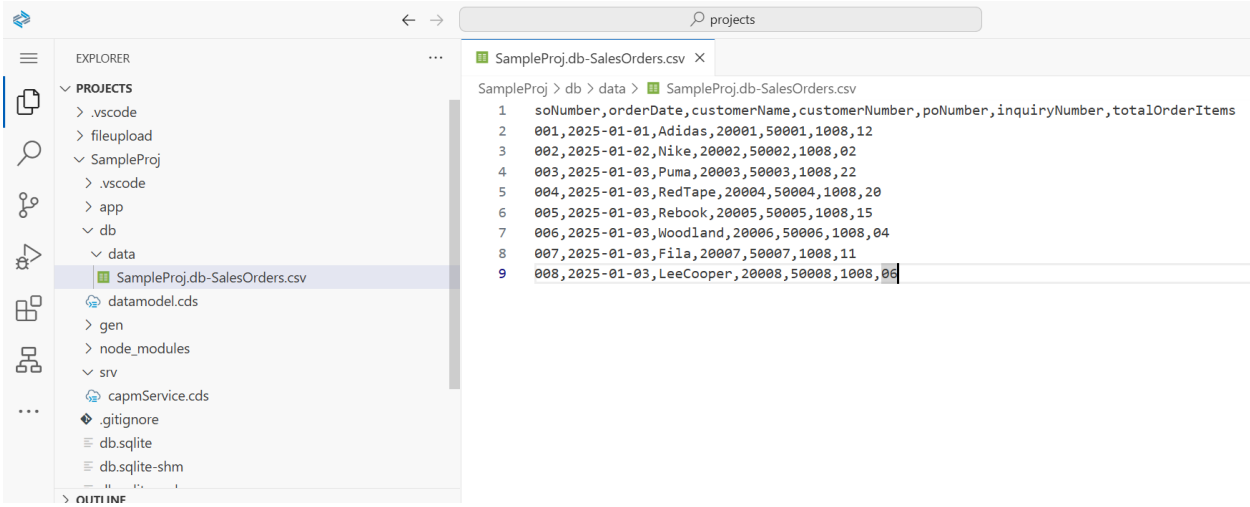


Step 1: Creating Data

So let's create a folder in ***db* folder**. Right-click on ***db*** folder and select ***New Folder***.

Now create a ***new file*** in the created ***data*** folder.

Name your file as ***SampleProj.db-SalesOrders.csv***. Here ***SampleProj*** is the **namespace – *SalesOrders* is our entity**. It's a nomenclature that needs to be followed for mapping the data.



Now, it's time to create actual data. So, in CAPM we use **.csv files** to create local data.

First let's save everything and then we will proceed.

Go to **file** and select the **save**.

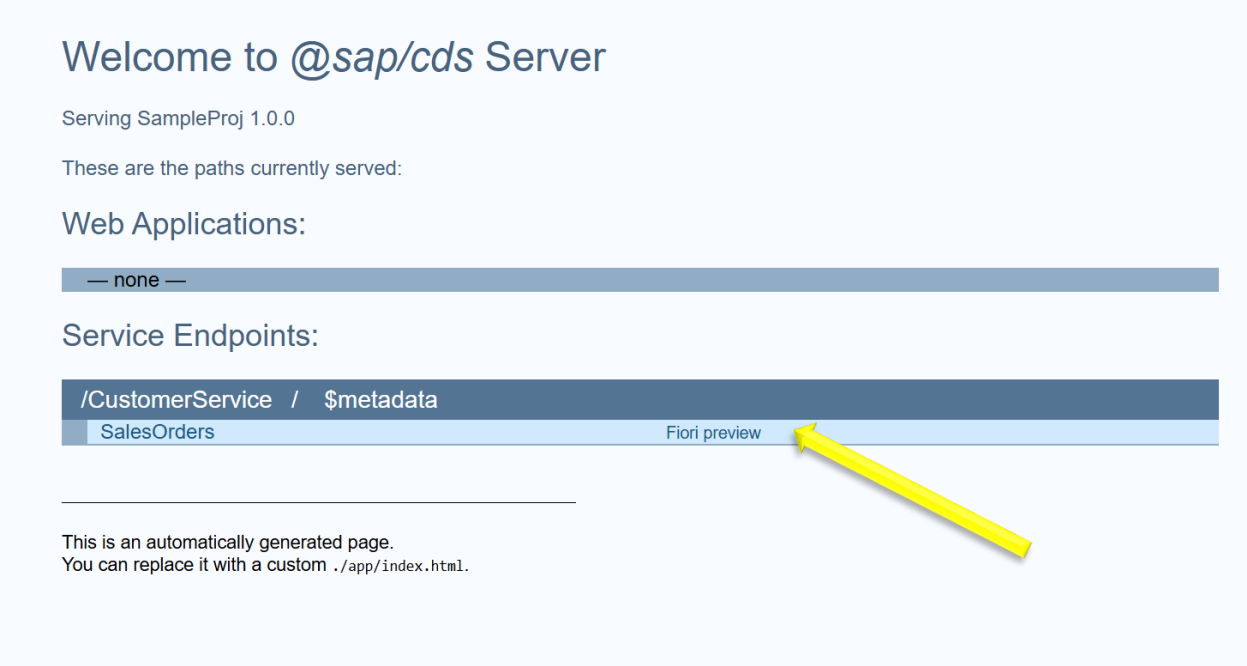
Now to build the application. ***cds build*** and then deploy it to sqlite using the command

cds deploy --to sqlite:db.sqlite

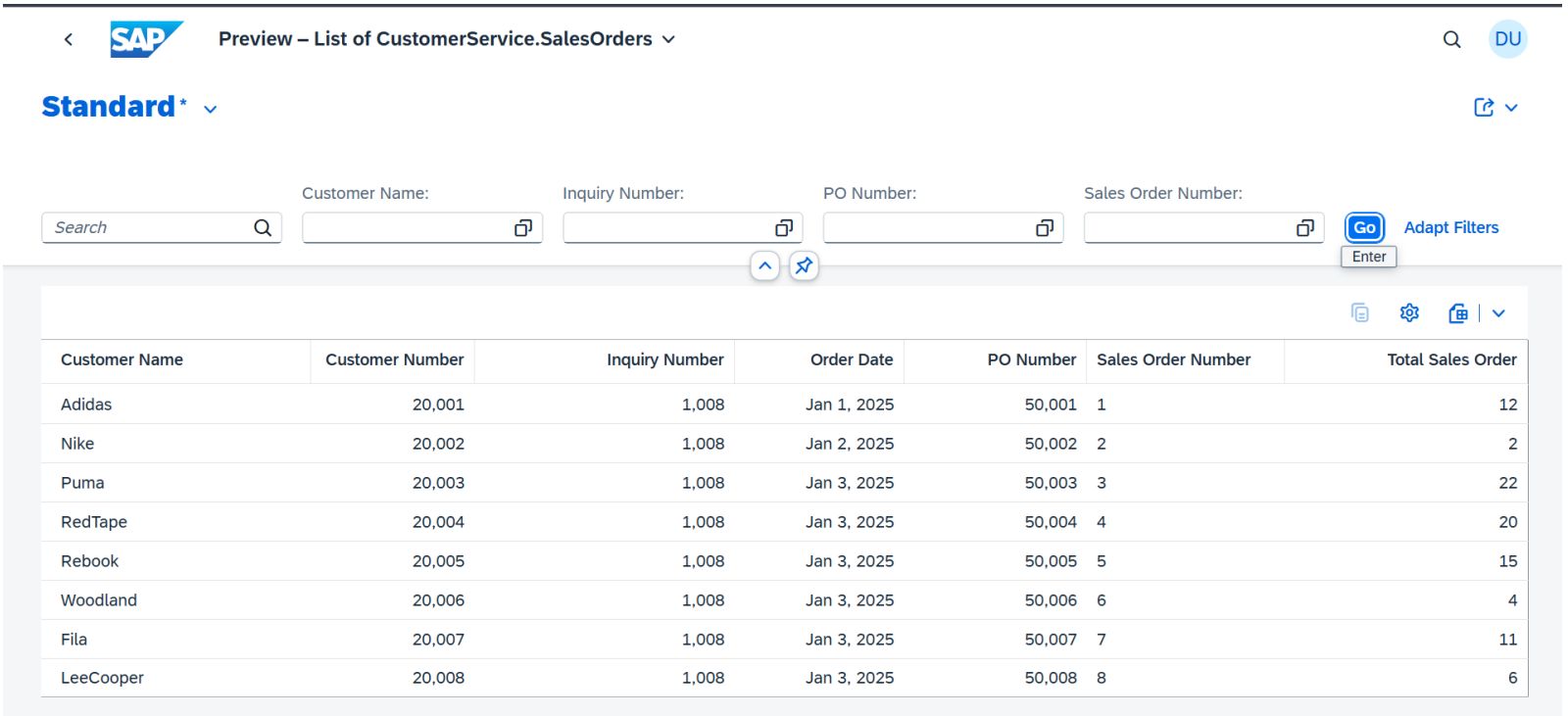
```
user: SampleProj $ cds build
building project [/home/user/projects/SampleProj], clean [true]
cds-dk [8.5.1], cds [8.6.0], compiler [5.6.0], home [/home/user/projects/SampleProj/node_modules/@sap/cds]

{
  build: {
    target: 'gen',
    tasks: [
      { for: 'nodejs', src: 'srv', options: { model: ['db', 'srv', 'app'] }}
    ]
  }
}

done > wrote output to:
```

```
user: SampleProj $ cds deploy --to sqlite:db.sqlite
  > init from db/data/SampleProj.db-SalesOrders.csv
/> successfully deployed to db.sqlite
```

And **cds watch** then click on **SAP FIORI PREVIEW.**



OUTPUT:



Customer Name	Customer Number	Inquiry Number	Order Date	PO Number	Sales Order Number	Total Sales Order
Adidas	20,001	1,008	Jan 1, 2025	50,001	1	12
Nike	20,002	1,008	Jan 2, 2025	50,002	2	2
Puma	20,003	1,008	Jan 3, 2025	50,003	3	22
RedTape	20,004	1,008	Jan 3, 2025	50,004	4	20
Rebook	20,005	1,008	Jan 3, 2025	50,005	5	15
Woodland	20,006	1,008	Jan 3, 2025	50,006	6	4
Fila	20,007	1,008	Jan 3, 2025	50,007	7	11
LeeCooper	20,008	1,008	Jan 3, 2025	50,008	8	6

Explore the SAP topics together

Certified SAP Consultant
NAIDU KARRI
mr_loyal_ml | +91 8247262816
karrinaidunk@gmail.com