CAPM Binding Data to a Table in a Basic SAP UI5 Application (Chapter 1 - READ)

Authored by:
Mahesh L
SAP UI5 Fiori Associate Consultant
Email: lokkidimahesh6@gmail.com

This document provides a beginner-friendly guide to binding data to a table in a basic SAP UI5 application, focusing on integration with an SAP HANA database and deployment within the Cloud Foundry runtime environment. It will walk you through setting up your development environment, creating data models, and connecting them to your UI5 application.

What You Will Learn:

- Setting up the development environment for binding data to a UI5 table.
- Connecting an SAP HANA database with your SAP UI5 application.
- Creating an XML view control for the UI5 table.
- Binding data with OData V4 to populate the table in the UI.

Chapter 1: Implementing CRUD Operations – Read (Chapter 1)

This guide currently covers the **Read** operation (data retrieval) in the SAP UI5 table. The **Create**, **Delete**, and **Update** operations will be covered in the upcoming chapters of this series.

Prerequisites:

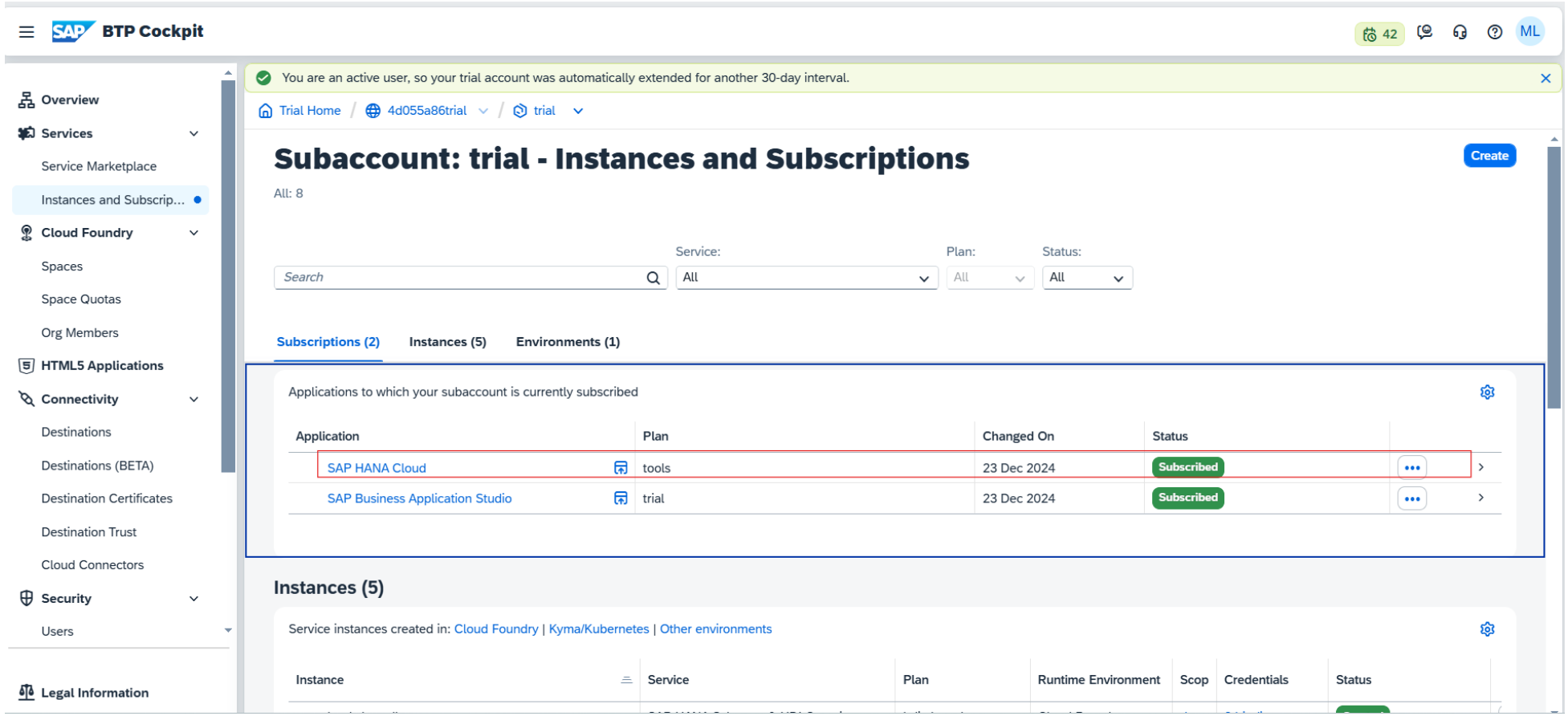
- Basic knowledge of JavaScript and Node.js.
- Familiarity with SAP BTP.
- Installed tools: Node.js, npm, and SAP Business Application Studio or Visual Studio Code.

For detailed steps on database and instance creation in CAPM, please refer to the following document:
https://www.linkedin.com/posts/mahesh-lokkidi-8664b9135_sap-capm-steps-pdf-guide-activity-7278389861781315585-nKxN?utm_source=share&utm_medium=member_desktop

1. **Open SAP HANA Cloud**

If SAP HANA Cloud is not visible in your environment, please subscribe to SAP HANA Cloud first. Once subscribed, proceed with creating a Cloud Foundry instance.

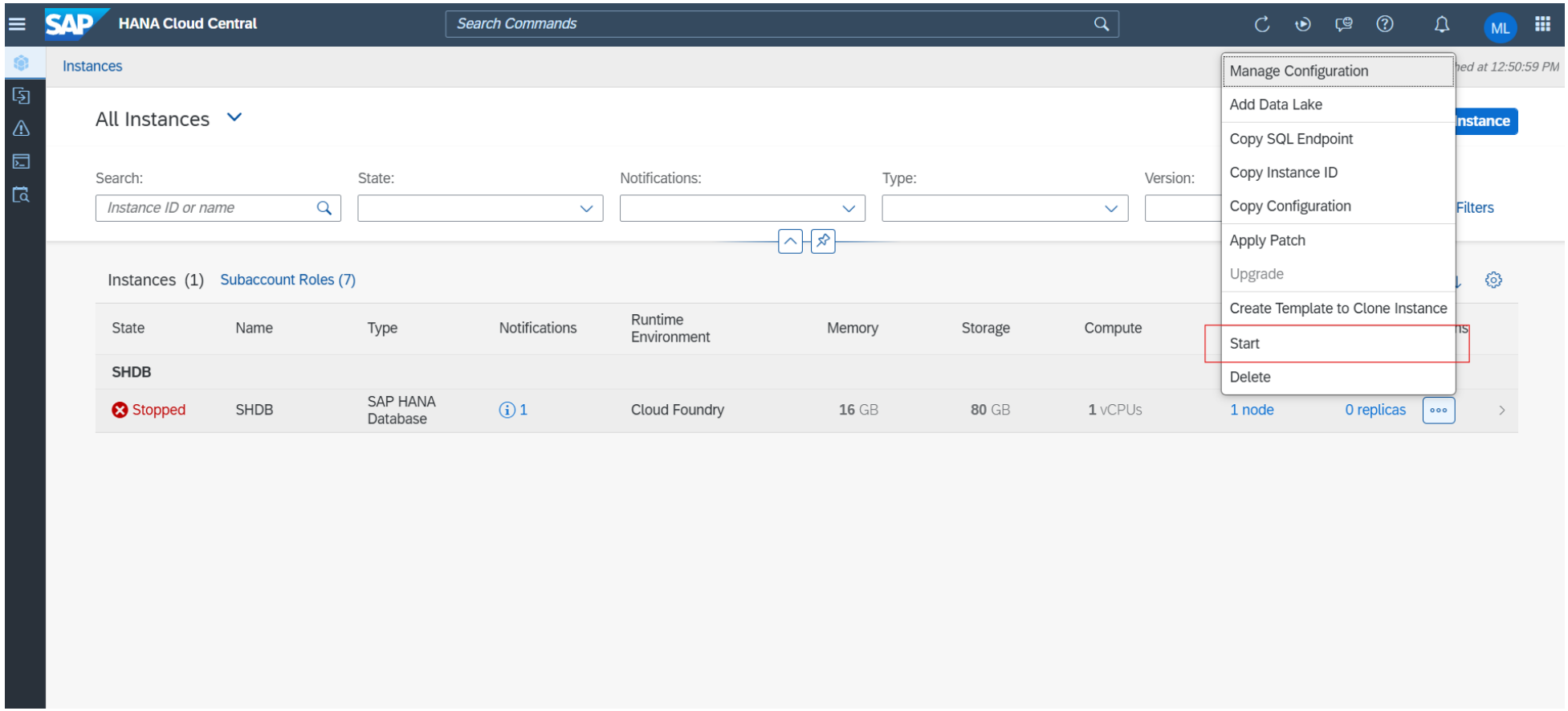
For detailed instructions and reference, kindly refer to the document linked above.



2.Start SAP HANA Cloud

Click on **Start** to initiate the SAP HANA Cloud service.

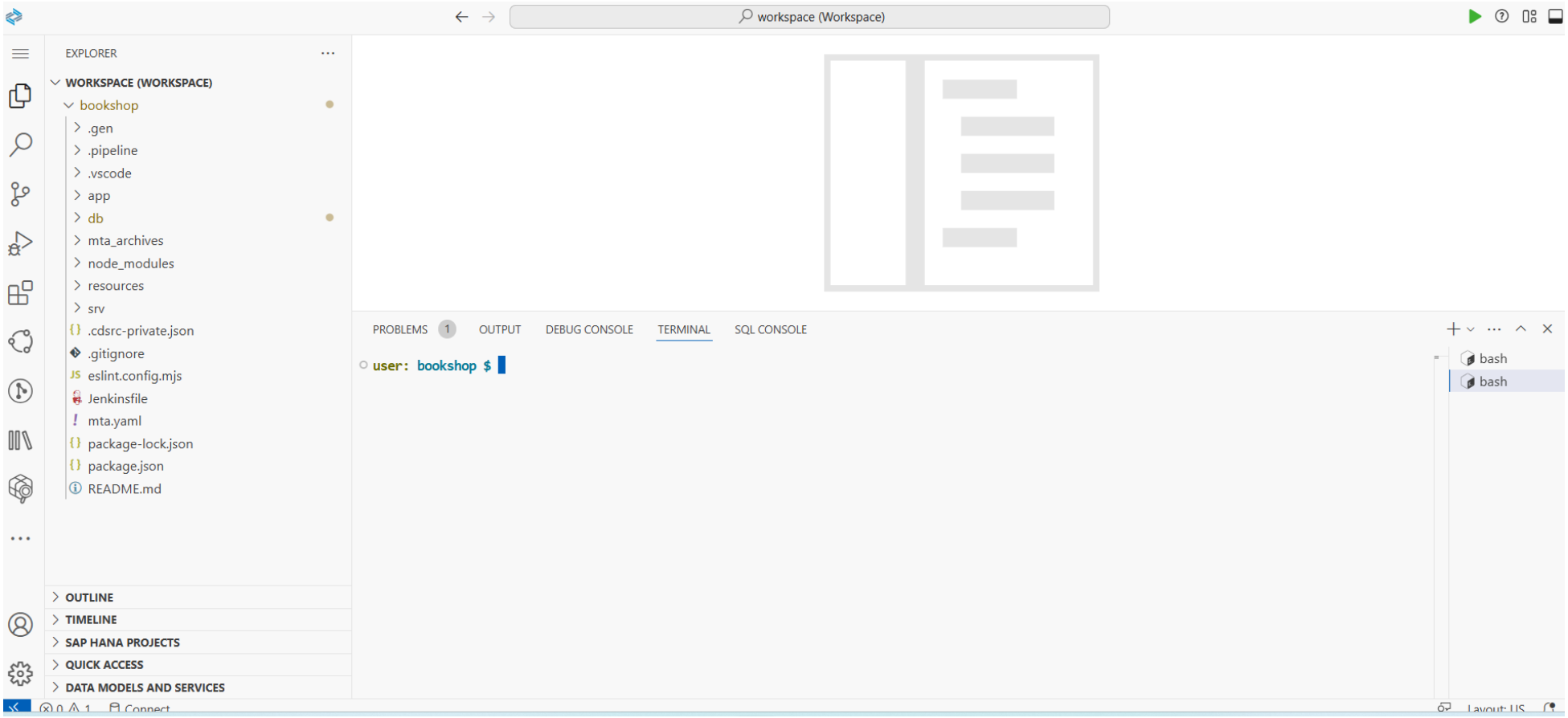
Note: For trial accounts, it is required to start the service daily. However, this step is not necessary for real-time (paid) accounts, as the service remains active.



3.Open SAP Business Application Studio

Open the **SAP Business Application Studio**.

In this example, I have already created a **CAPM Bookshop** project. For your reference, I have provided the document link on the first page. The project I created is named **Supermarket**, which is similar to the Bookshop project.

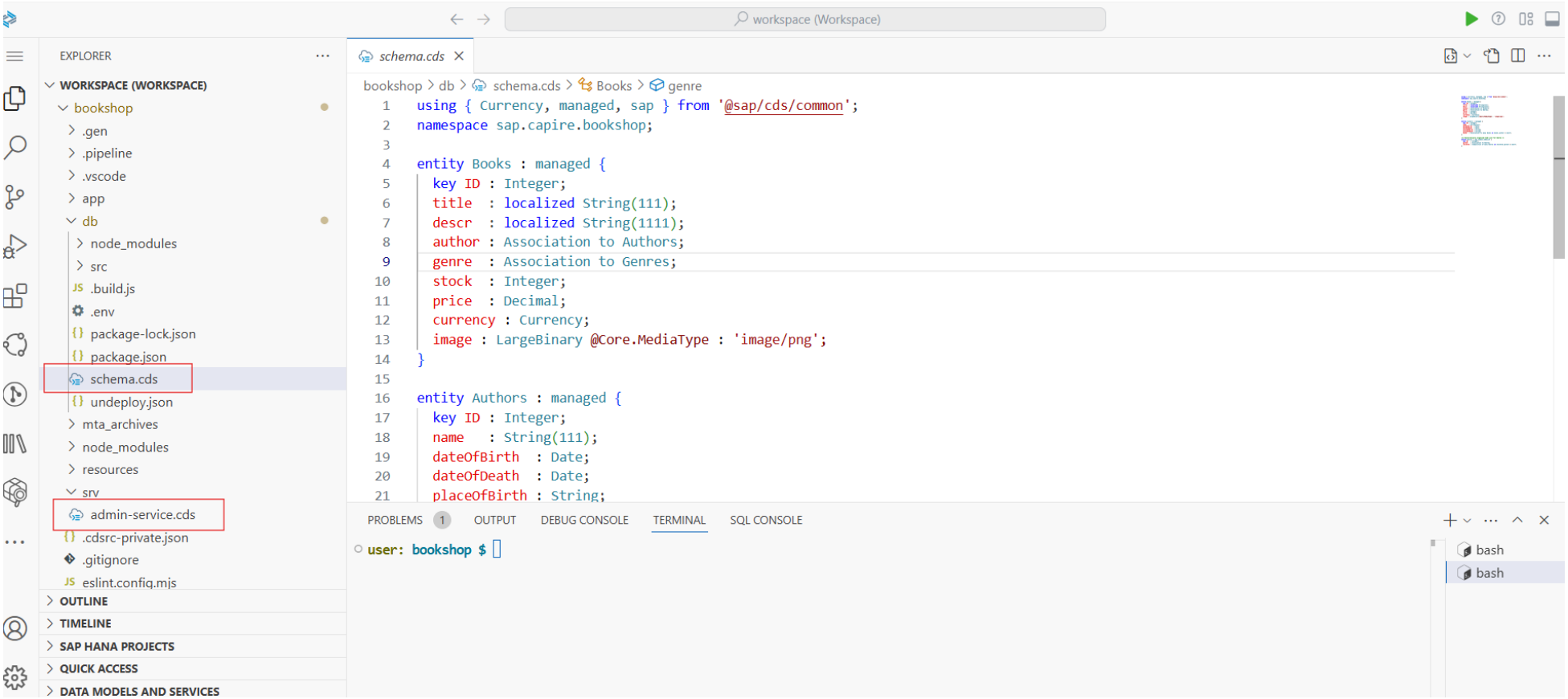


4.Create CDS Files for Database and Service

I have created the following CDS files:

1. **schema.cds** – This file defines the database schema and tables for the application.
2. **admin-service.cds** – This service file defines the OData services for interacting with the database.

These files are essential for setting up the database structure and exposing the necessary APIs for CRUD



schema.cds

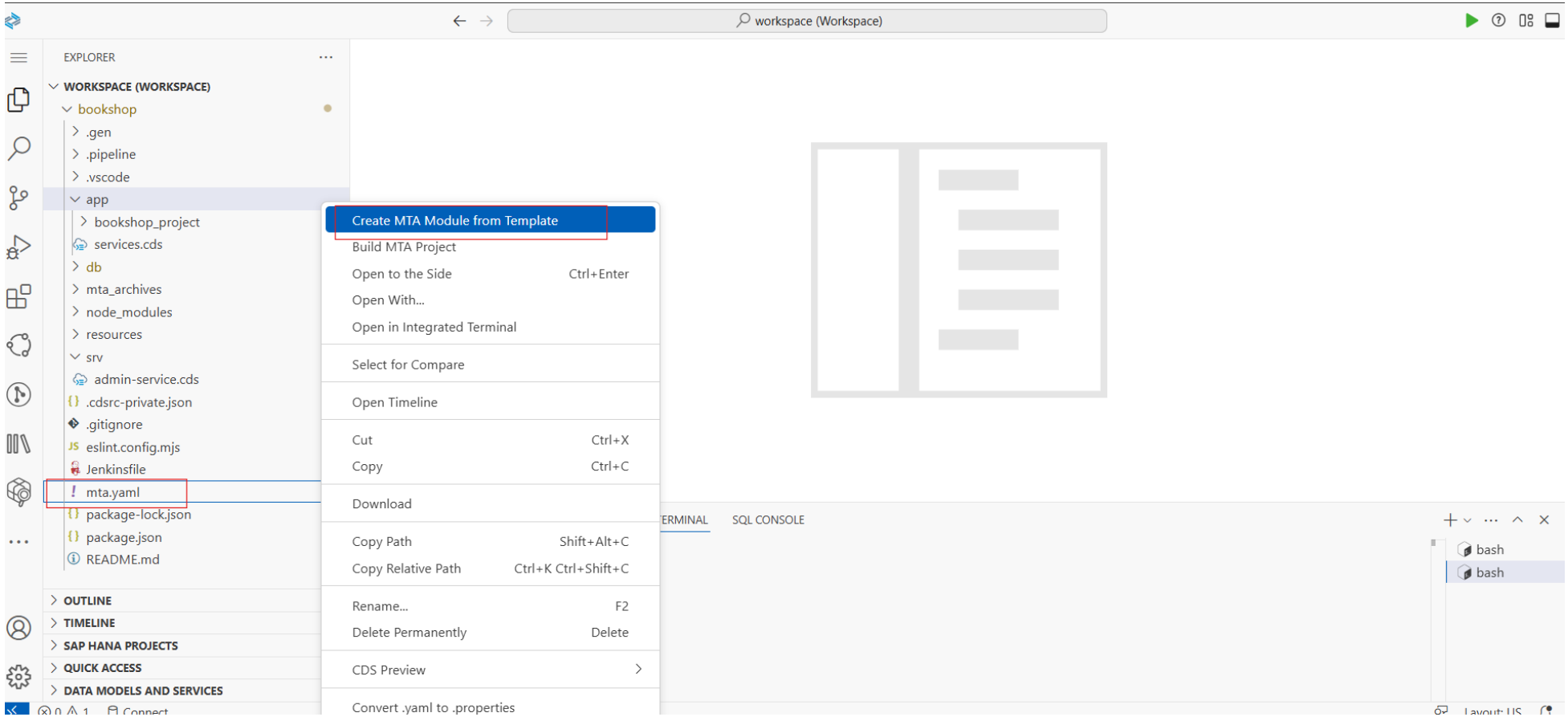
```
using { Currency, managed, sap } from '@sap/cds/common';
namespace sap.capire.bookshop;

entity Books : managed {
  key ID : Integer;
  title  : localized String(111);
  descr  : localized String(1111);
  author : Association to Authors;
  genre  : Association to Genres;
  stock  : Integer;
  price  : Decimal;
  currency : Currency;
  image : LargeBinary @Core.MediaType : 'image/png';
}

entity Authors : managed {
  key ID : Integer;
  name   : String(111);
  dateOfBirth  : Date;
  dateOfDeath  : Date;
  placeOfBirth : String;
  placeOfDeath : String;
  books  : Association to many Books on books.author = $self;
}

/** Hierarchically organized Code List for Genres */
entity Genres : sap.common.CodeList {
  key ID   : Integer;
  parent   : Association to Genres;
  children : Composition of many Genres on children.parent = $self;
}
```

admin-service.cds

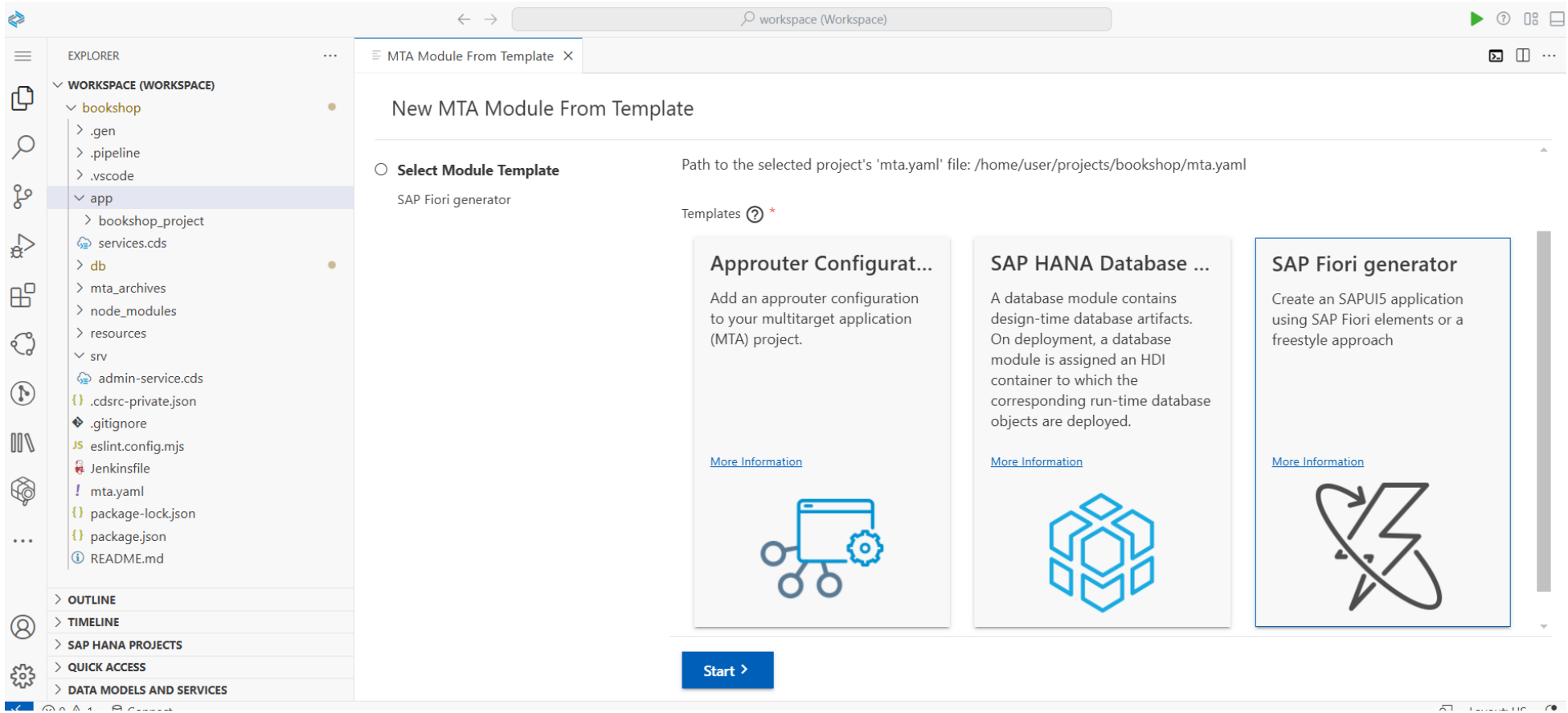
```
using { sap.capire.bookshop as my } from '../db/schema';
service AdminService @(requires:'any') {
  entity Books as projection on my.Books;
  entity Authors as projection on my.Authors;
}
```

5.Create MTA Module from Template

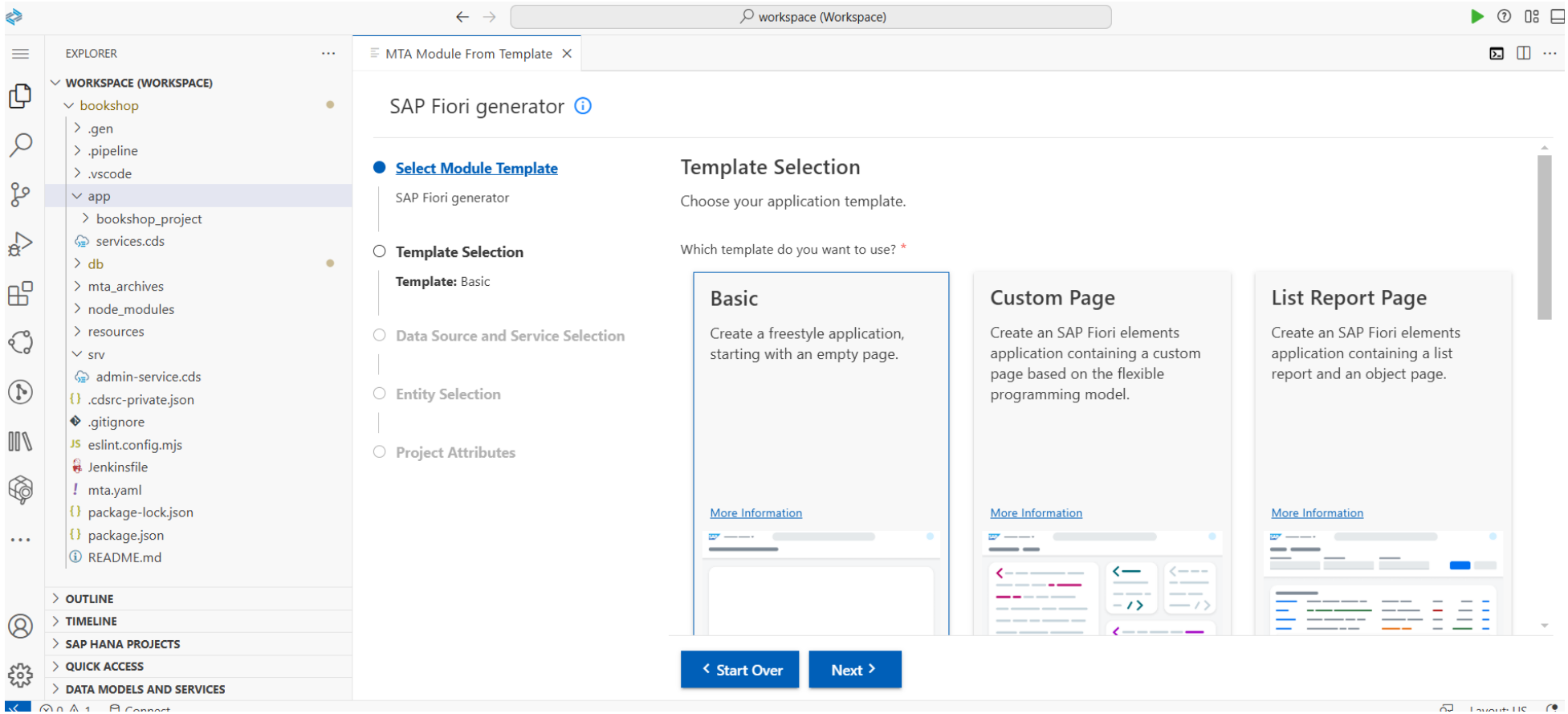
In the **MTA (Multi-Target Application)** file, right-click and select **Create MTA Module from Template**.



6.Create a Freestyle UI5 Application with SAP Fiori Generator

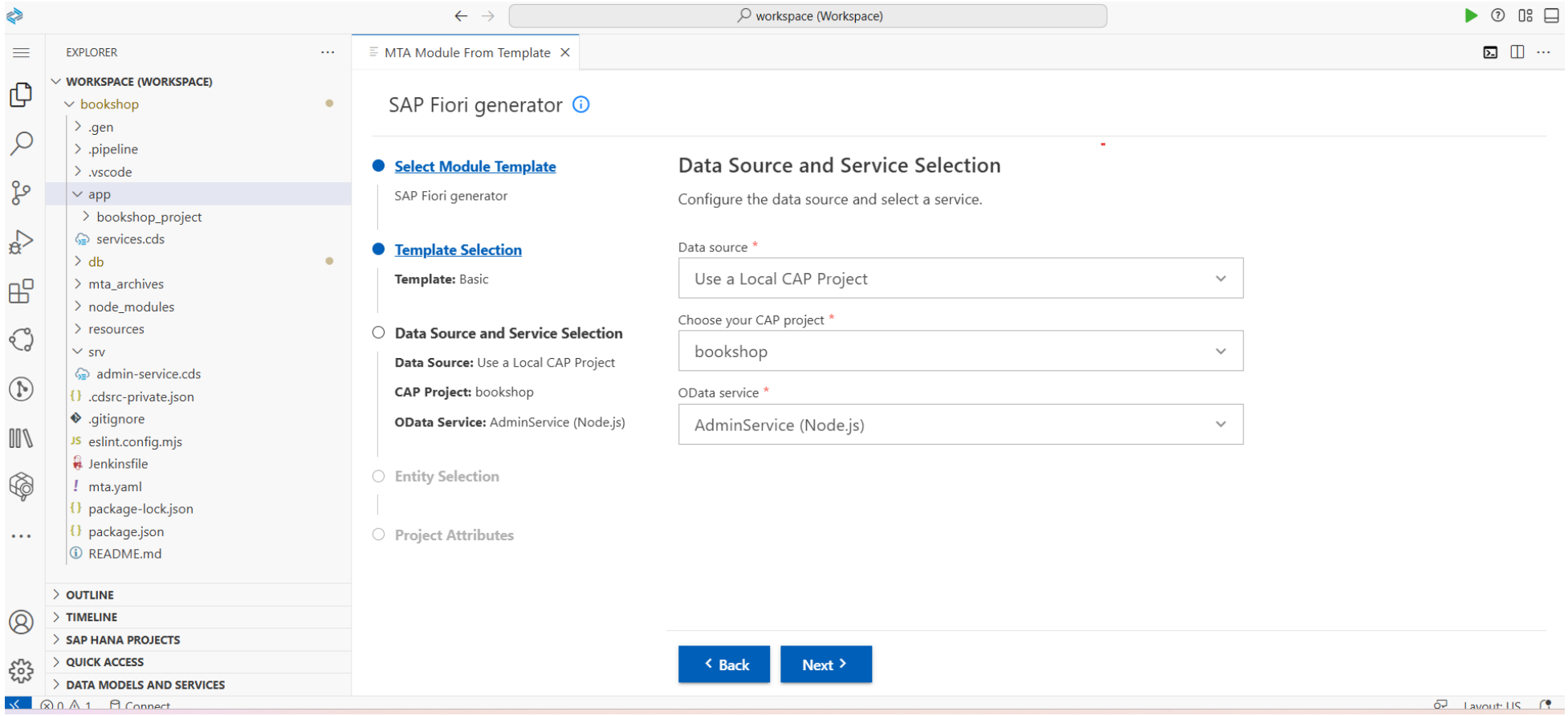


7.Select Basic and Next

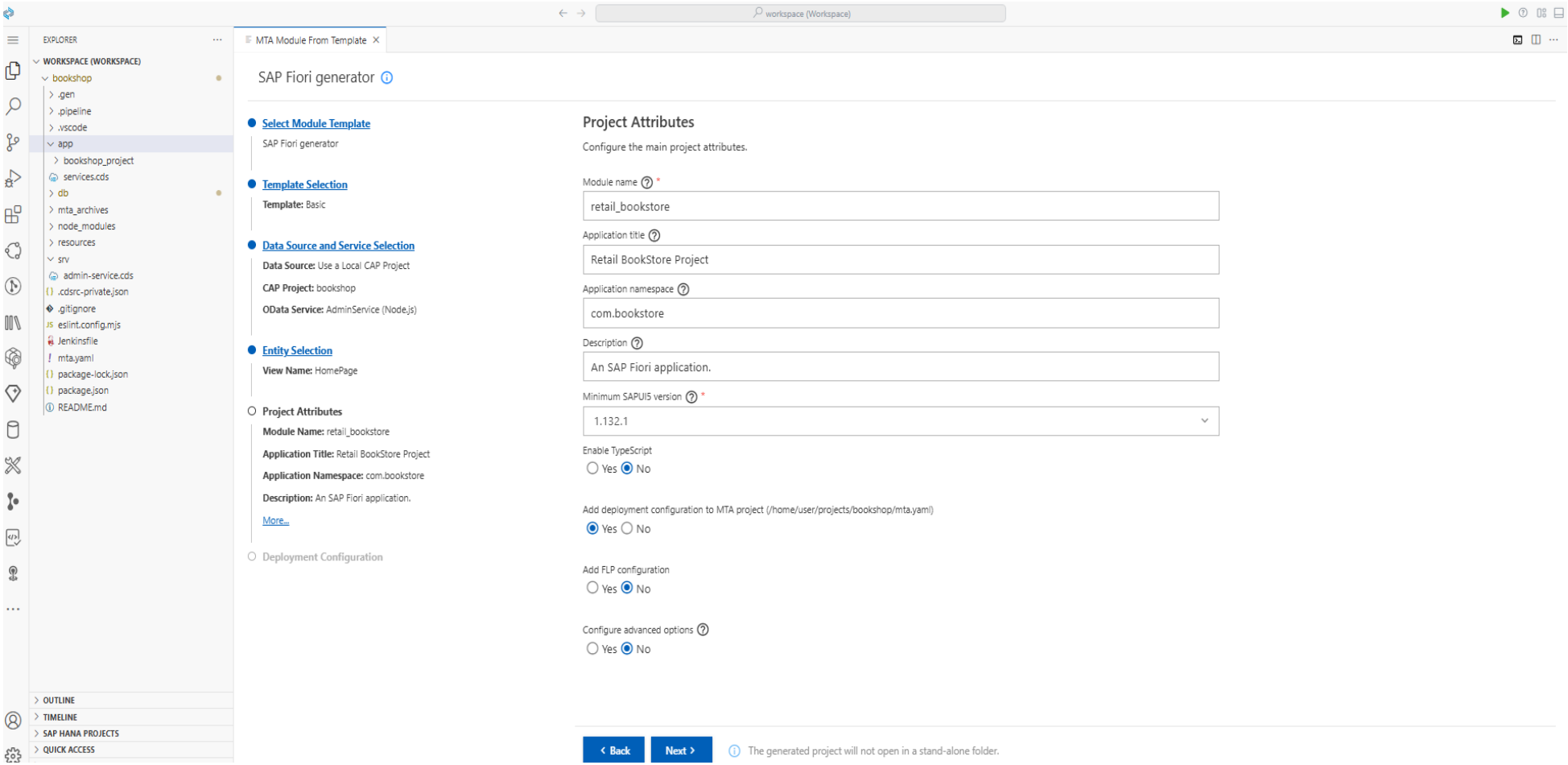


8.Select Local CAP Project and Service

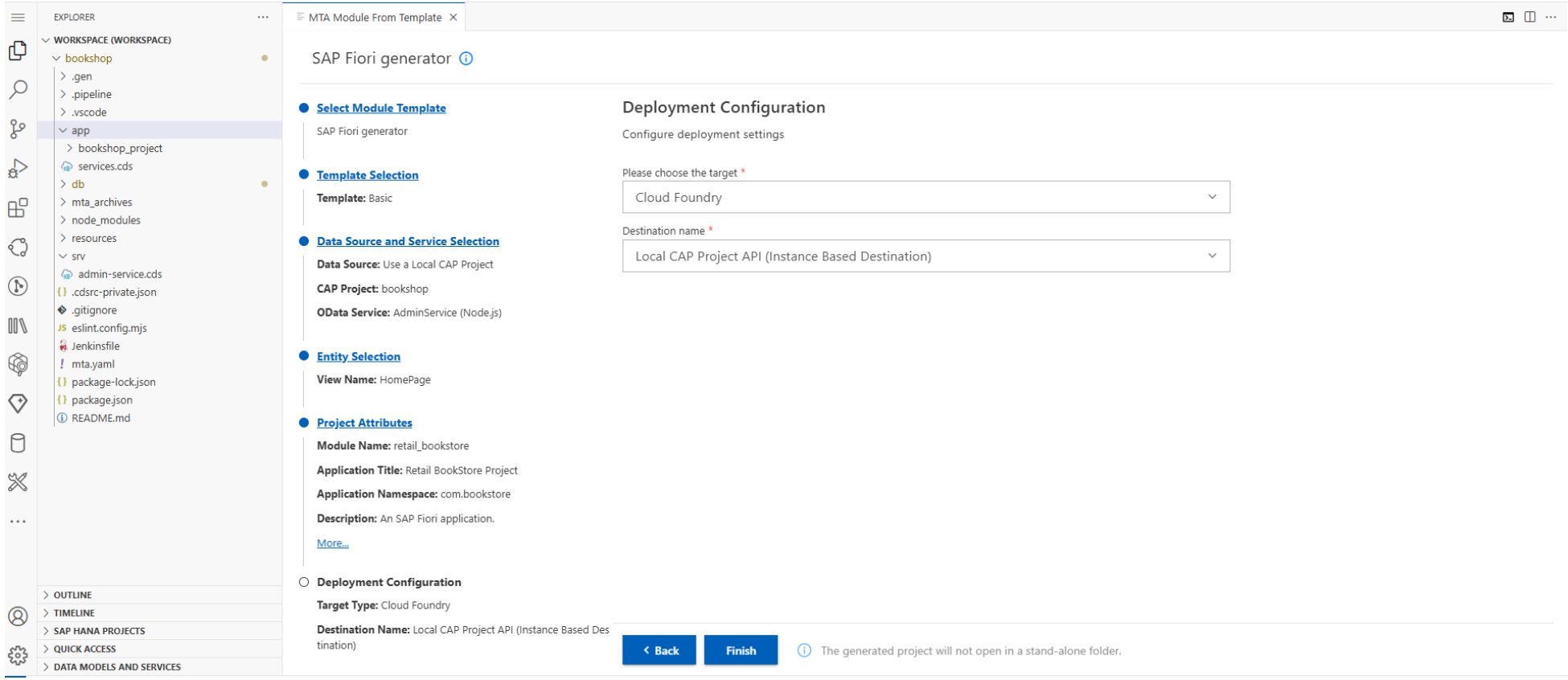
1. Choose the **service** (such as admin-service.cds) that you defined within the CAP project.



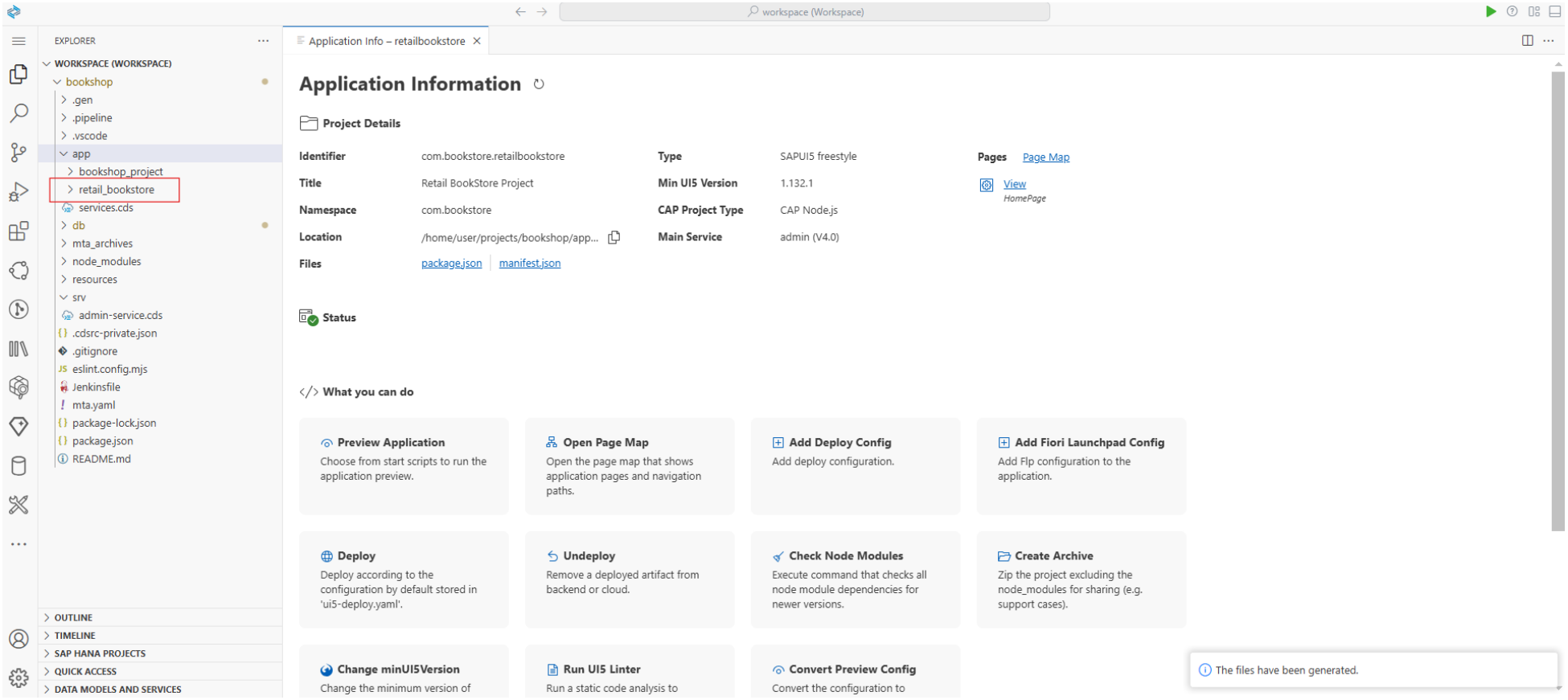
9.Enter Project Attributes



10.Select Cloud Foundry (earlier which is used in CAP creation)



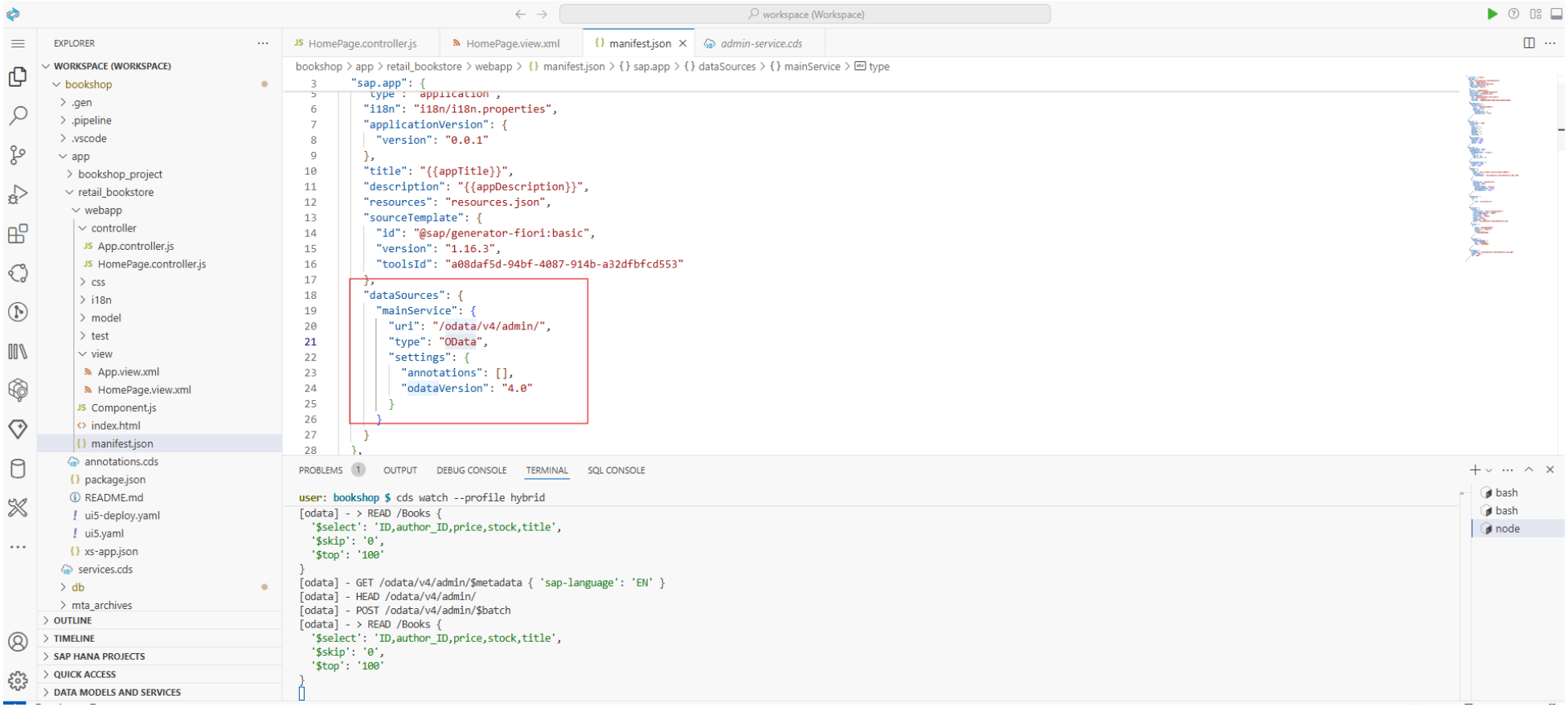
11.Application is Ready



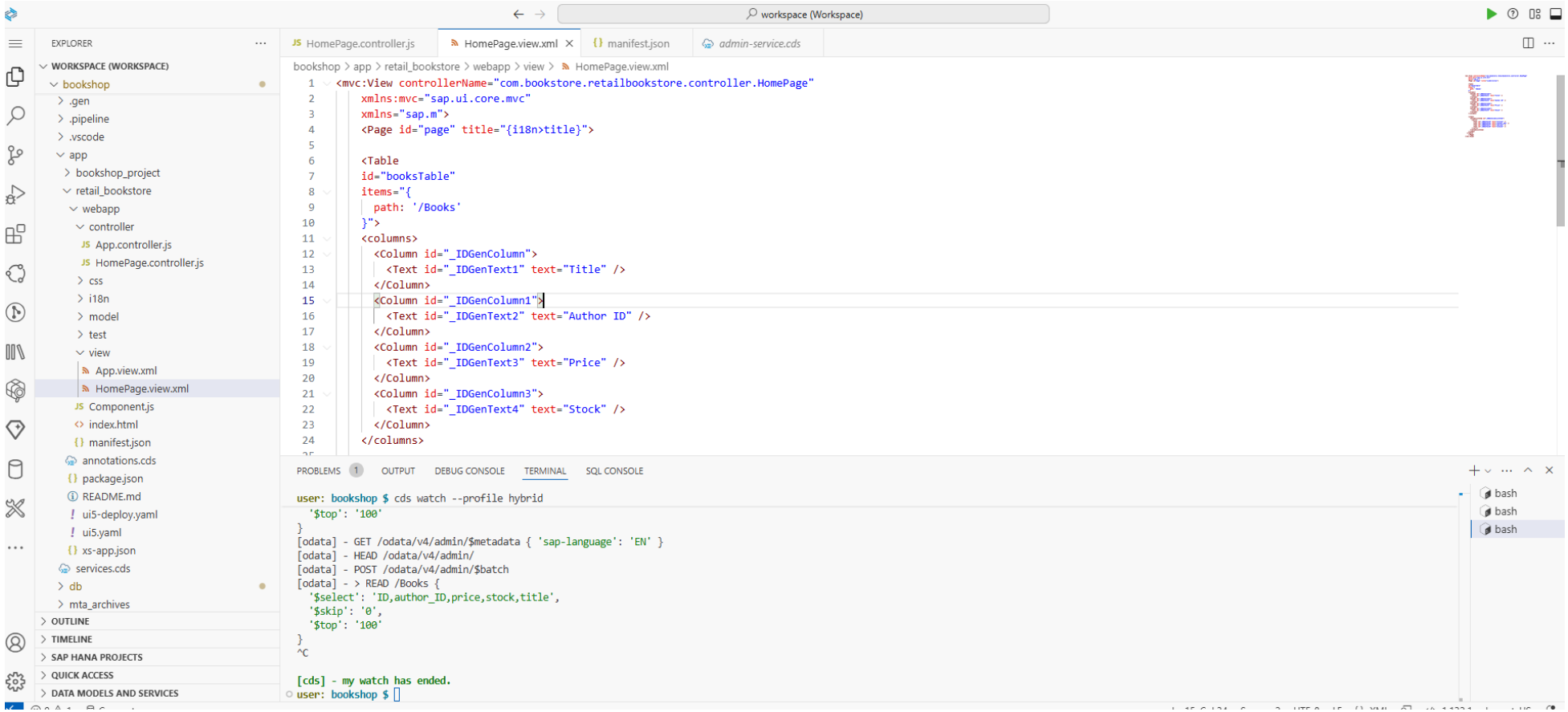
12.Adding the dataSources in manifest.json:

Adding the dataSources section in manifest.json binds your SAP UI5 application to the OData service exposed by your CAP project. It defines the service URL, specifies the OData version (4.0), and sets up the necessary configurations for communication between frontend and backend. This is crucial for making CRUD operations and retrieving data.

```
"dataSources": { "mainService": {"uri": "/odata/v4/admin/","type": "OData","settings": { "annotations": [], "odataVersion": "4.0 } }}
```



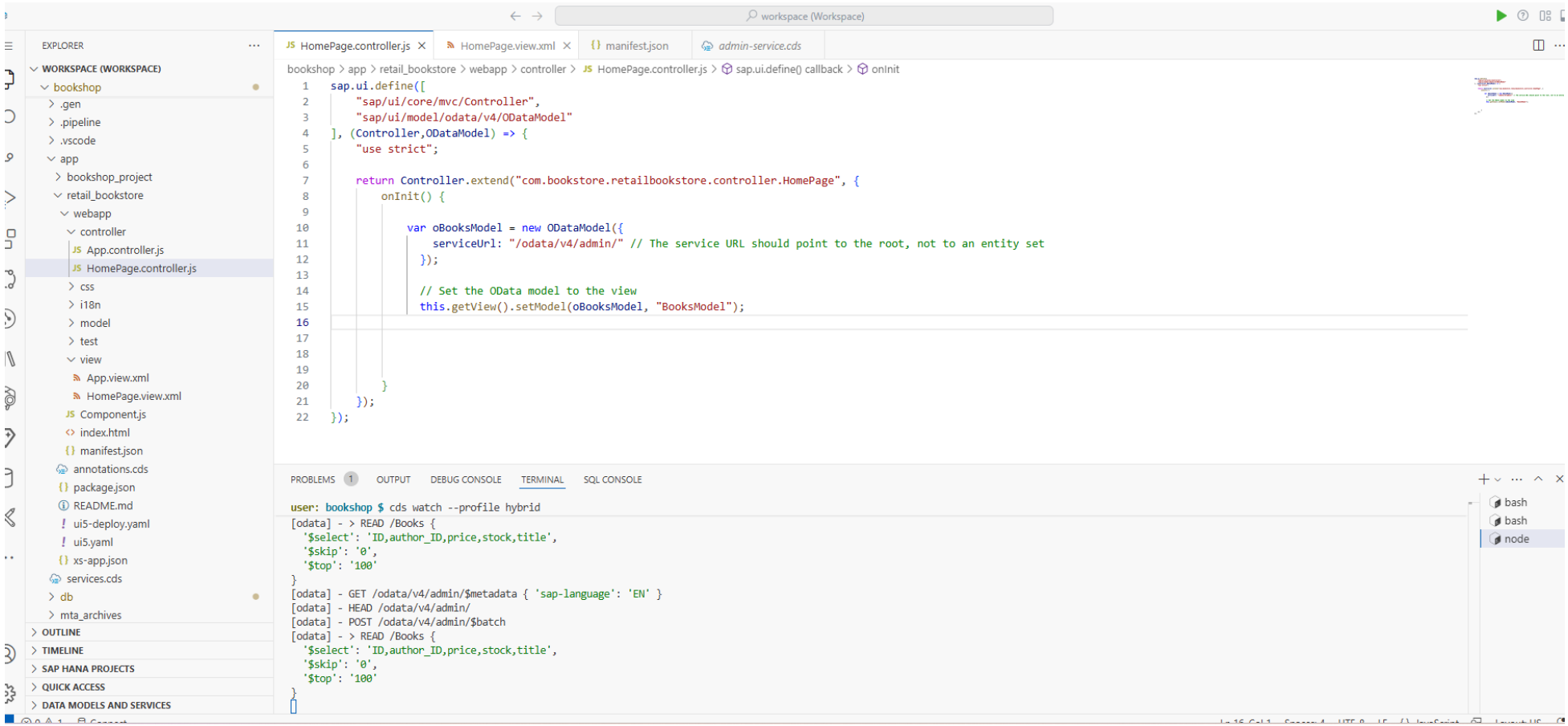
13.Screen Shots and code for xml view and controller



To connect the OData service to your UI5 application, you can use the following code:

```
var oBooksModel = new ODataModel({
   serviceUrl: "/odata/v4/admin/" // The service URL should point to the root, not to an entity set
});

// Set the OData model to the view
this.getView().setModel(oBooksModel, "BooksModel");
```

This code creates a new **ODataModel** with the service URL pointing to the root of the OData service and binds it to the view with the model name BooksModel. This allows your UI to interact with the data from the backend OData service.



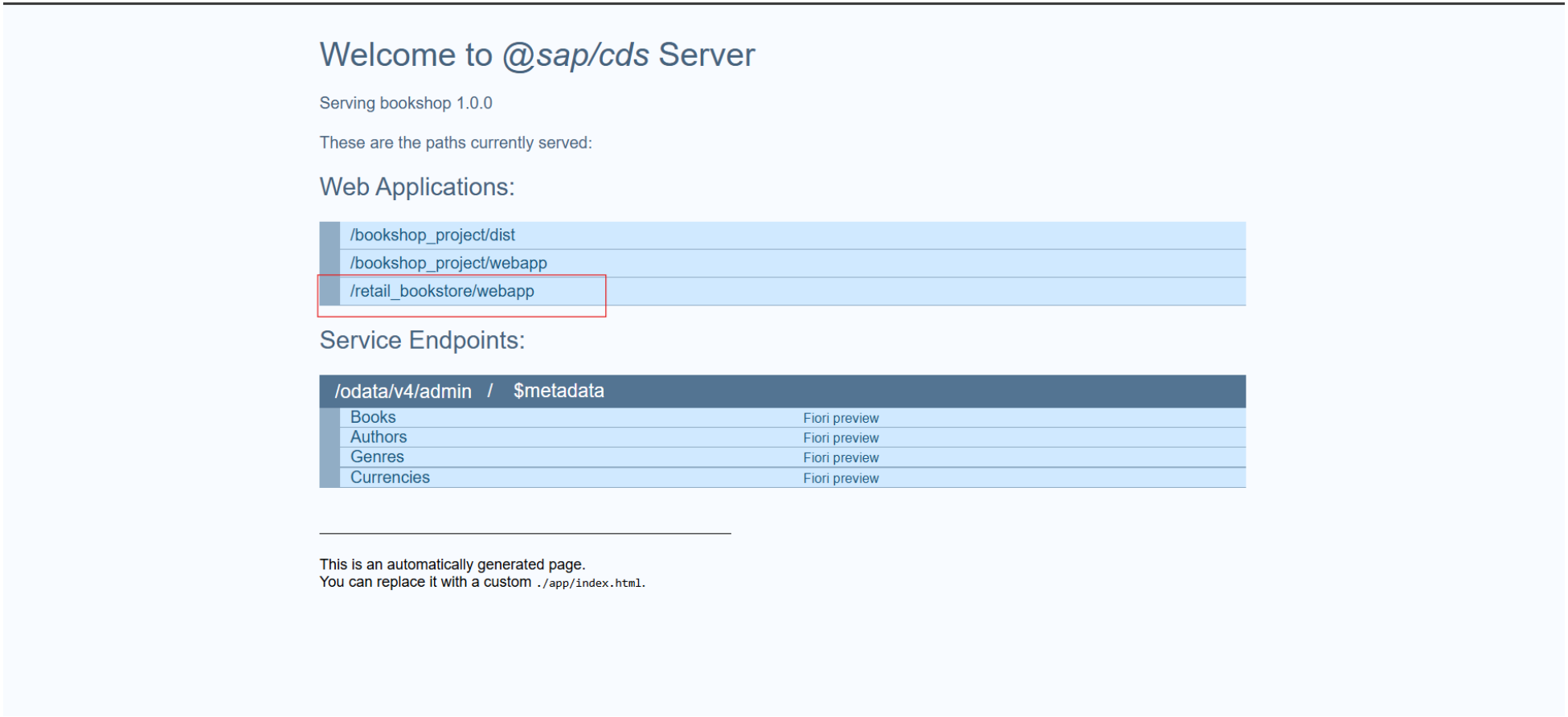
View.xml

```
<mvc:View controllerName="com.bookstore.retailbookstore.controller.HomePage"
   xmlns:mvc="sap.ui.core.mvc"
   xmlns="sap.m">
   <Page id="page" title="{i18n>title}">

   <Table
   id="booksTable"
   items="{
    path: '/Books'
   }">
   <columns>
    <Column id="_IDGenColumn">
     <Text id="_IDGenText1" text="Title" />
    </Column>
    <Column id="_IDGenColumn1">
     <Text id="_IDGenText2" text="Author ID" />
    </Column>
    <Column id="_IDGenColumn2">
     <Text id="_IDGenText3" text="Price" />
    </Column>
    <Column id="_IDGenColumn3">
     <Text id="_IDGenText4" text="Stock" />
    </Column>
   </columns>

   <items>
    <ColumnListItem id="_IDGenColumnListItem">
     <cells>
      <Text id="_IDGenText5" text="{title}" />
```

```
        <Text id="_IDGenText6" text="{author_ID}" />
        <Text id="_IDGenText7" text="{price}" />
        <Text id="_IDGenText8" text="{stock}" />
       </cells>
     </ColumnListItem>
   </items>
  </Table>
   </Page>
</mvc:View>

Controller.js

sap.ui.define([
   "sap/ui/core/mvc/Controller",
   "sap/ui/model/odata/v4/ODataModel"
], (Controller,ODataModel) => {
   "use strict";

   return Controller.extend("com.bookstore.retailbookstore.controller.HomePage", {
      onInit() {

         var oBooksModel = new ODataModel({
            serviceUrl: "/odata/v4/admin/" // The service URL should point to the root, not to an entity set
          });

          // Set the OData model to the view
          this.getView().setModel(oBooksModel, "BooksModel");



      }
   });
});
```

14.Application Ready – Click on Retail Bookstore

Once the application is fully set up and deployed, click on **Retail Bookstore** to access and interact with the application. This will launch the UI5 interface connected to your backend OData service.



15.Application Final: Table View with All Data

Once the application is ready, the **Retail Bookstore** will display a table view populated with all the data from the backend OData service. This table will show a list of items, such as books, retrieved from the database, providing users with an interactive and dynamic view of the data.

Retail BookStore Project

Title	Author ID	Price	Stock
The Art of Coding	101	29.99	50
Database Essentials	102	19.99	30
Cloud Computing	103	39.99	20

Thank you for reading **Chapter 1: Implementing CRUD Operations – Read**.
The next chapter, **Create**, will be available soon. Stay tuned for more!

Thank You