

# Step 1: Access SAP Business Application Studio

i) Begin by logging into your **SAP BTP Trial Account**.

Navigate to **Instances and Subscriptions** and click on **SAP Business Application Studio (BAS)** to launch the development environment.

The screenshot shows the SAP BTP Cockpit interface. On the left, there is a sidebar with various navigation options: Overview, Services, Service Marketplace, Instances and Subscriptions (which is selected), Cloud Foundry, HTML5 Applications, Connectivity, Security, and Legal Information. The main content area has a breadcrumb trail: Trial Home / d8ae6be0trial / trial. The title is "Subaccount: trial - Instances and Subscriptions". Below the title, there are sections for "Subscriptions (2)", "Instances (2)", and "Environments (1)". The "Instances (2)" section contains a table with two rows:

Instance	Service	Plan	Runtime Environment	Scope	Credentials	Status
Database	SAP HANA Schema...	hdi-shared	Cloud Foundry	dev	1 key	Created
DBADMIN	SAP HANA Cloud	hana	Cloud Foundry	dev		Created

Image 1.1 - Accessing SAP BAS from Instances and Subscriptions in SAP BTP

ii) Once SAP Business Application Studio (BAS) is launched, click on “**Create Dev Space**” to begin setting up your development environment.

The screenshot shows the SAP Business Application Studio interface. At the top, there is a navigation bar with links: My Dev Spaces, Watch Demo, Tutorials, Blogs, Community, Documentation, Privacy, and a power button icon. The main content area has a heading "Welcome to SAP Business Application Studio". Below it, there is a message: "Now you can efficiently develop business applications for the Intelligent Enterprise with a powerful and modern development environment." A note states: "At the heart of SAP Business Application Studio are the dev spaces. A dev space is a pre-configured private environment (your own "virtual machine" on the cloud) where you can develop, build, test, and run using pre-installed runtimes and tools. You can create a dev space for each of your development scenarios." A "Create Dev Space" button is located at the bottom left. On the right, there is a "Dev Spaces" section with a table showing five existing dev spaces:

Name	Type	Status	Created On
Employee_Productivity	SAP Fiori	RUNNING	Created On: 09/12/2020 3:52 PM
Strategic_Buyer	SAP Intranet Web Application	RUNNING	Created On: 03/06/2020 1:43 PM
S4_HANA_UI	Full Stack Cloud Application	RUNNING	Created On: 11/04/2020 2:30 PM
Traveling_Agent	SAP Fiori	RUNNING	Created On: 09/03/2020 7:32 PM
Manager_Approval	SAP Mobile Application	PENDING	Created On: 04/06/2020 12:00 PM

Image 1.2 - Creating a new Dev Space in SAP BAS

## Step 2: Configure Your Dev Space

i) Enter a **Dev Space name** (e.g., dev), select **Full Stack Cloud Application** as the application type, and check the box for **Additional SAP Extensions** to enable relevant tools and features.

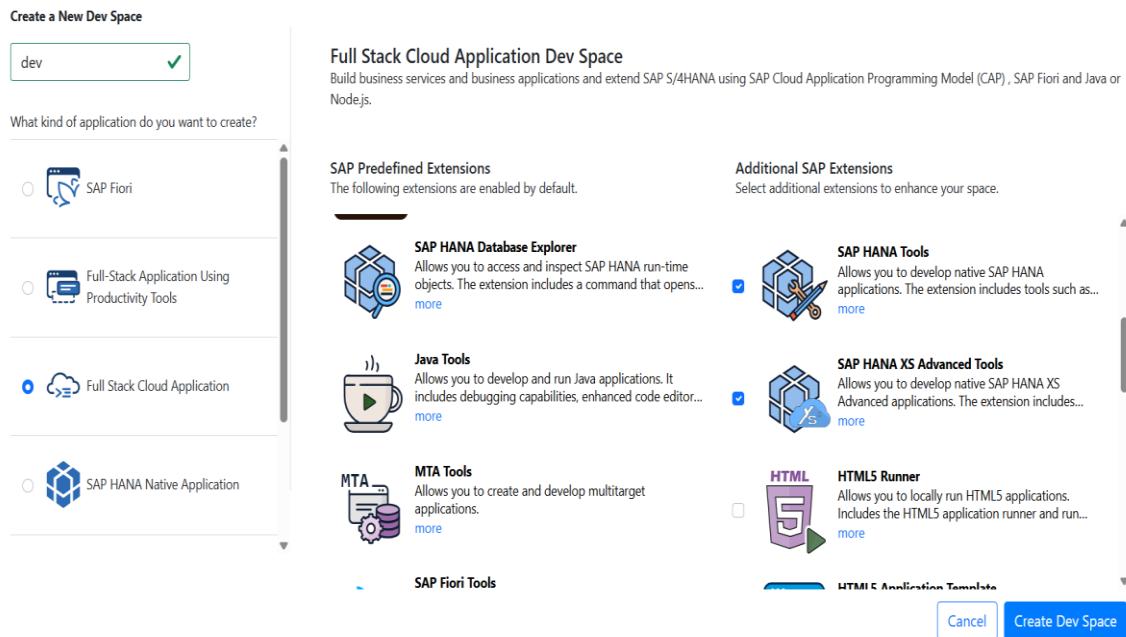
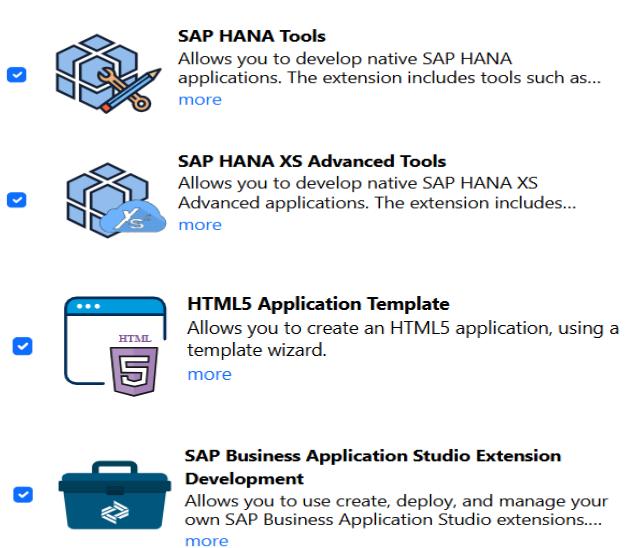


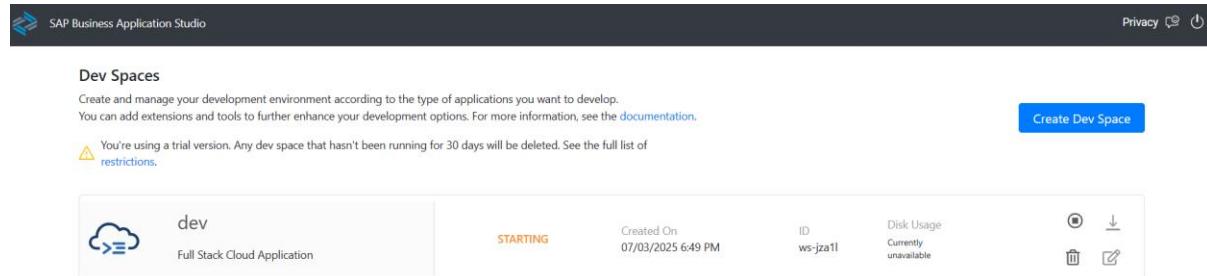
Image 2.1- Naming and configuring the Dev Space in SAP BAS

ii) **Note:** Below are the most essential **Additional SAP Extensions** you should select while creating your Dev Space:



## ◆ Step 3: Dev Space Creation & Status

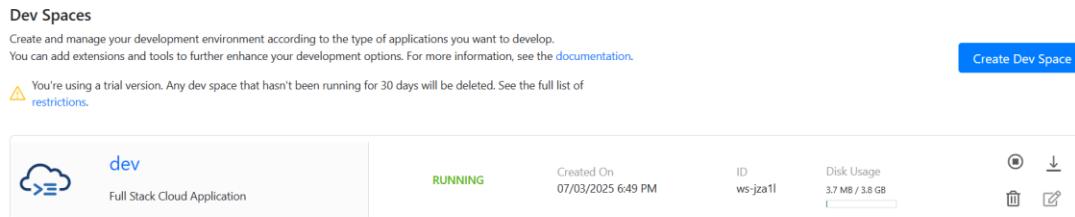
Once the Dev Space is created, it will appear with a status of “**STARTING**”. After a few moments, it will change to “**RUNNING**”, indicating that it’s ready for use.



The screenshot shows the SAP Business Application Studio interface. At the top, there's a navigation bar with the SAP logo, "SAP Business Application Studio", and links for "Privacy", "Help", and "Logout". Below the navigation bar is a section titled "Dev Spaces" with a sub-section header "Create and manage your development environment according to the type of applications you want to develop. You can add extensions and tools to further enhance your development options. For more information, see the [documentation](#). ⚠ You're using a trial version. Any dev space that hasn't been running for 30 days will be deleted. See the full list of [restrictions](#).". On the right side of this section is a blue button labeled "Create Dev Space". The main content area displays a single dev space entry in a table format:

Name	Status	Created On	ID	Disk Usage	Action Buttons
dev Full Stack Cloud Application	STARTING	07/03/2025 6:49 PM	ws-jza1l	Currently unavailable	

Image 3.1- Dev Space status – STARTING



This screenshot shows the same SAP Business Application Studio interface as the previous one, but the dev space named "dev" is now in the "RUNNING" state. The table data has changed to reflect this status:

Name	Status	Created On	ID	Disk Usage	Action Buttons
dev Full Stack Cloud Application	RUNNING	07/03/2025 6:49 PM	ws-jza1l	3.7 MB / 3.8 GB	

Image 3.2- Dev Space status – RUNNING

## ◆ Step 4: Start a New Project from Template

i) Once your development environment is ready, click on “**Start from Template**” to create a new project using the guided wizard provided by SAP Business Application Studio.

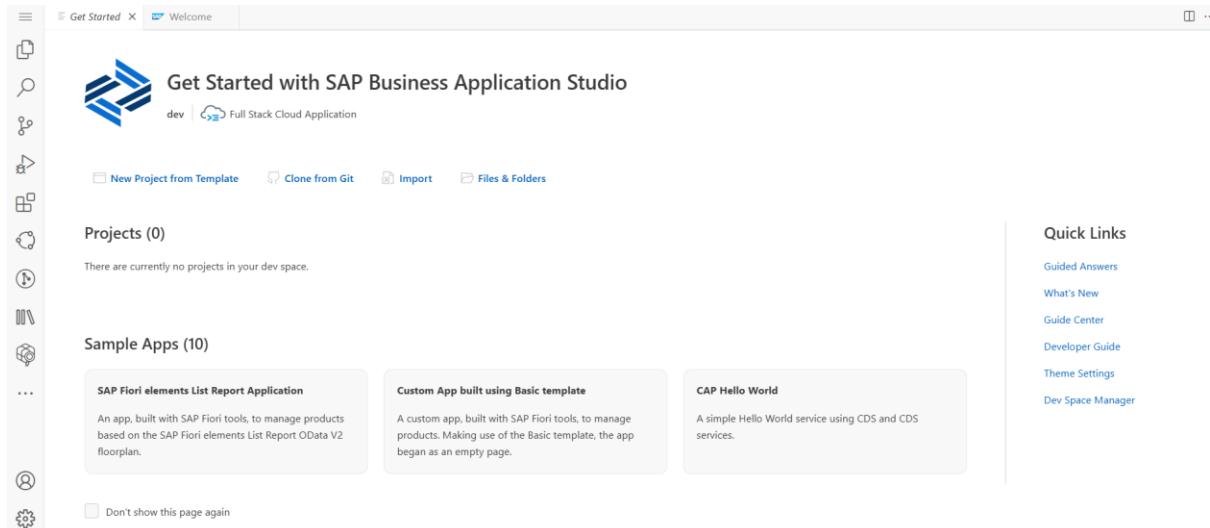


Image 4.1- Launching a new project from template in SAP BAS

ii) From the list of available templates, choose “**CAP Project**” (Cloud Application Programming Model) to start building your full-stack application using the CAP framework.

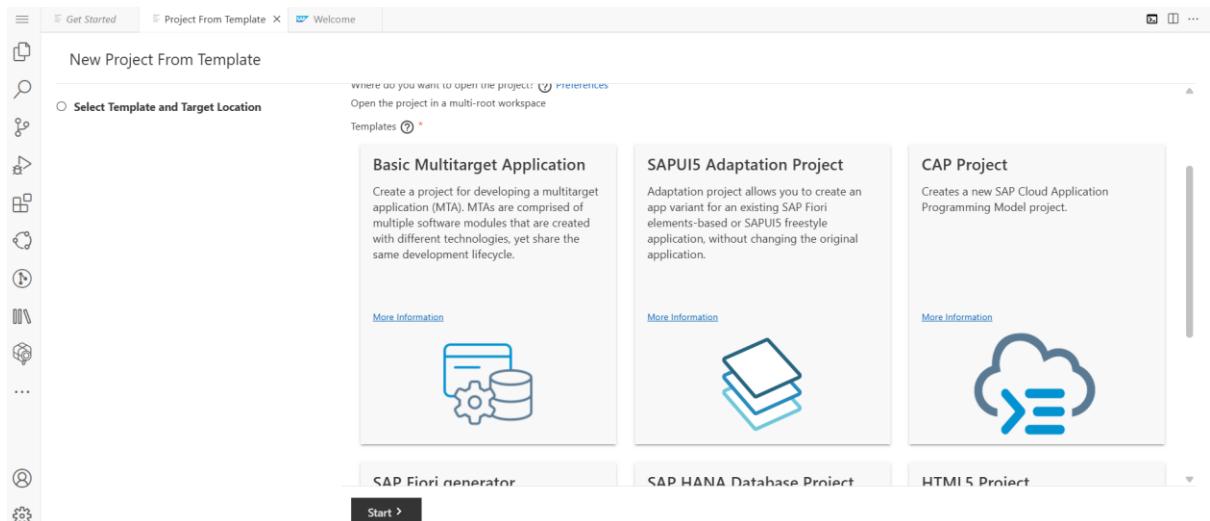


Image 4.2- Selecting the CAP Project template in the template wizard and click on start

## ◆ Step 5: Configure CAP Project Settings

Enter your **Project Name** (e.g., project), select **Java** as the runtime, and choose **SAP HANA Cloud** as the database.

Then, select your preferred **deployment method**, and click “**Finish**” to generate the project.

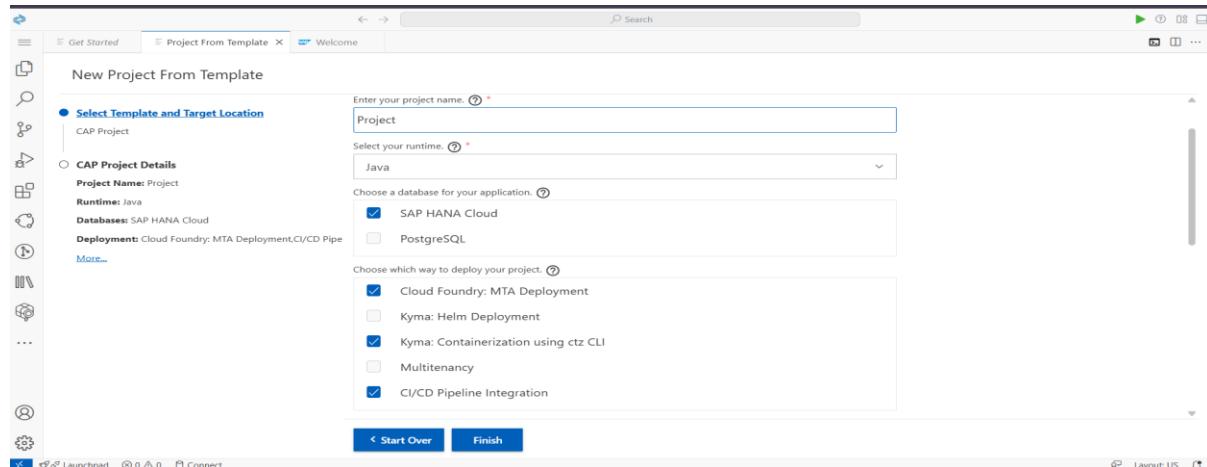
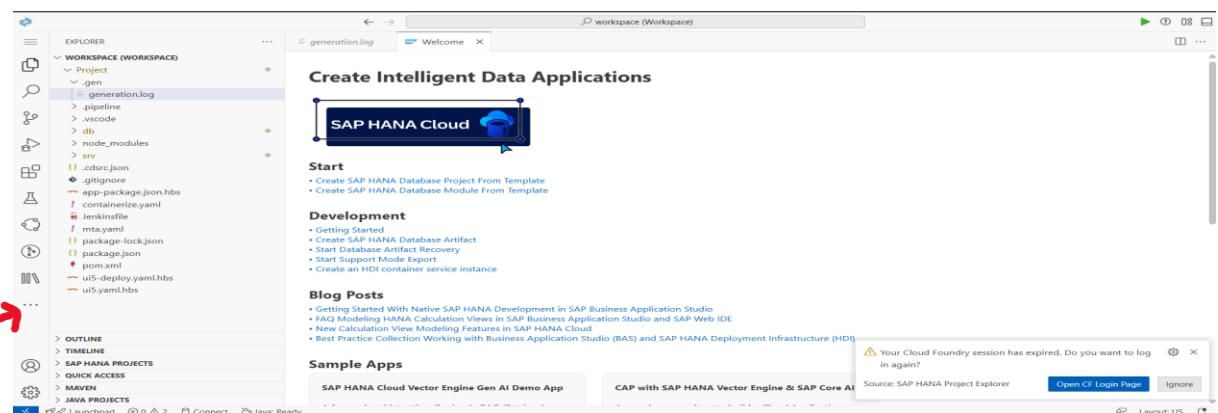


Image 5.1- Configuring CAP project with Java runtime and SAP HANA Cloud

## ◆ Step 6: Sign In to Cloud Foundry

Once your workspace is created, click on the **three dots (...)** in the left panel (as shown by the arrow), and select “**Sign in to Cloud Foundry**” to authenticate your environment with the SAP BTP Cloud Foundry instance.



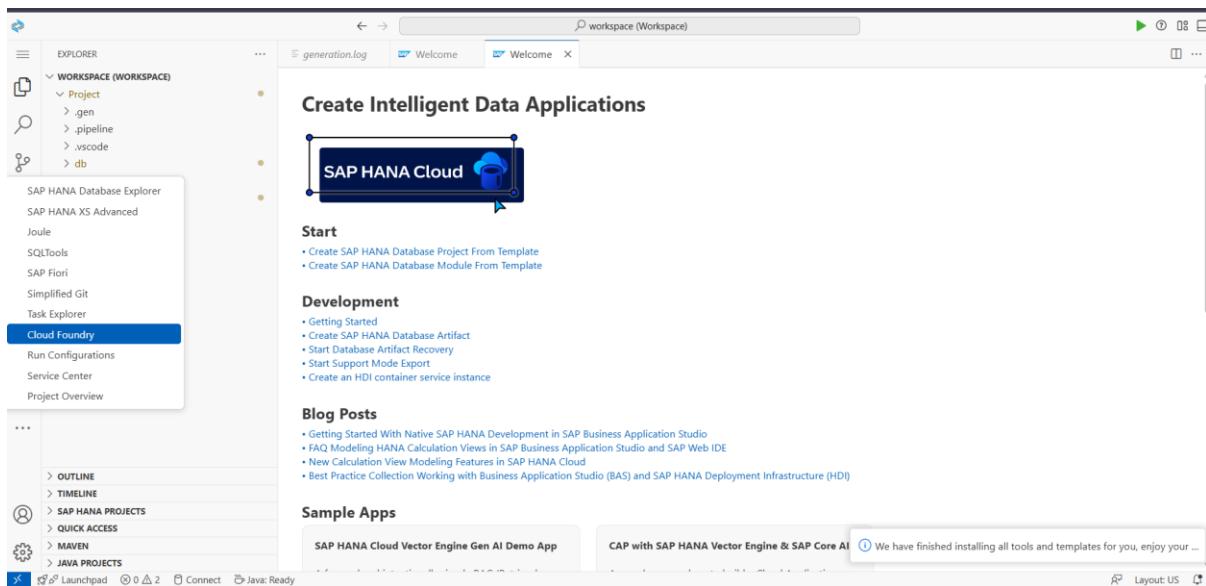


Image 6.2- Click on the three dots and sign in to Cloud Foundry

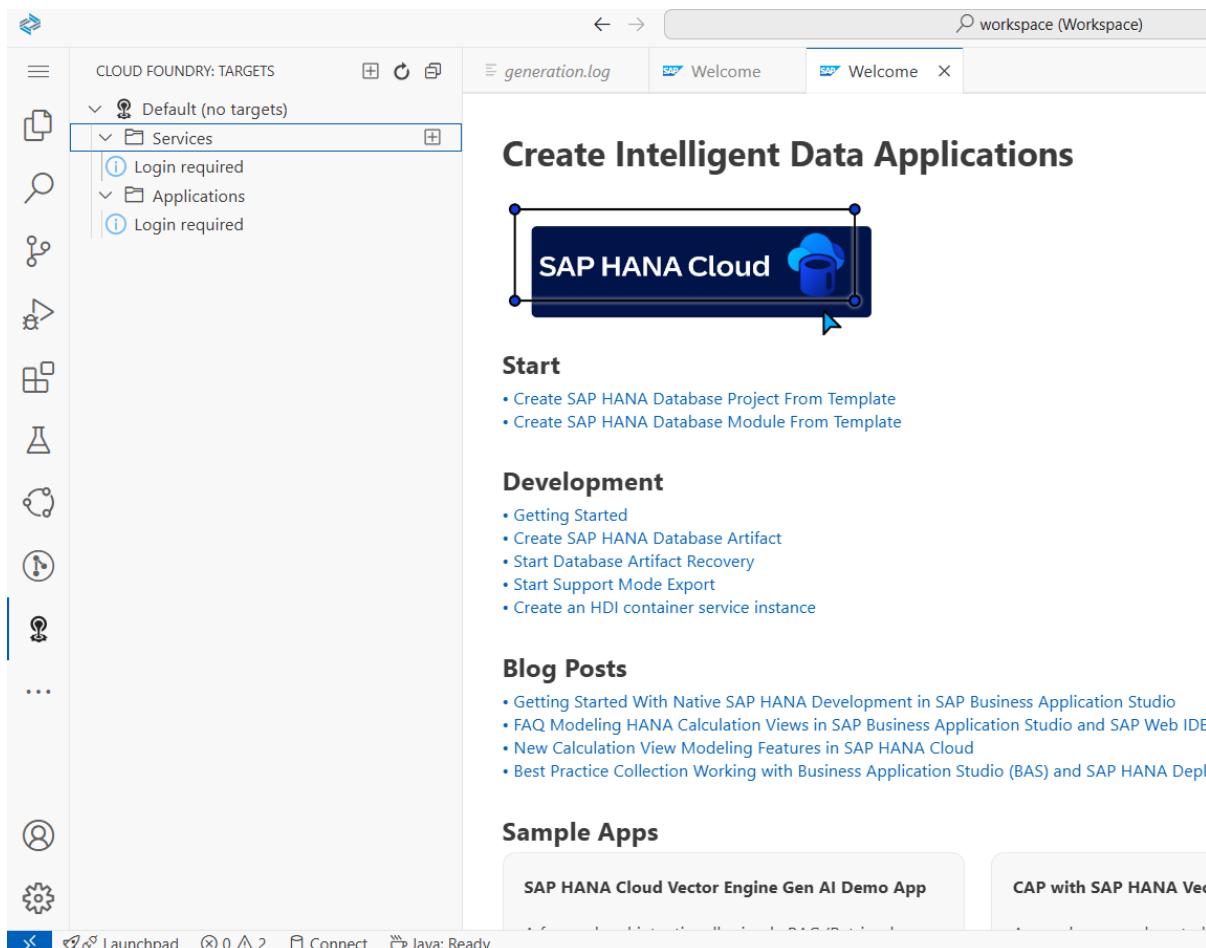


Image 6.3 - Click Login Required

## ii) Authenticate to Cloud Foundry

Choose your **authentication method** (typically username and password), then enter your **SAP BTP Cloud Foundry credentials** to complete the sign-in process.

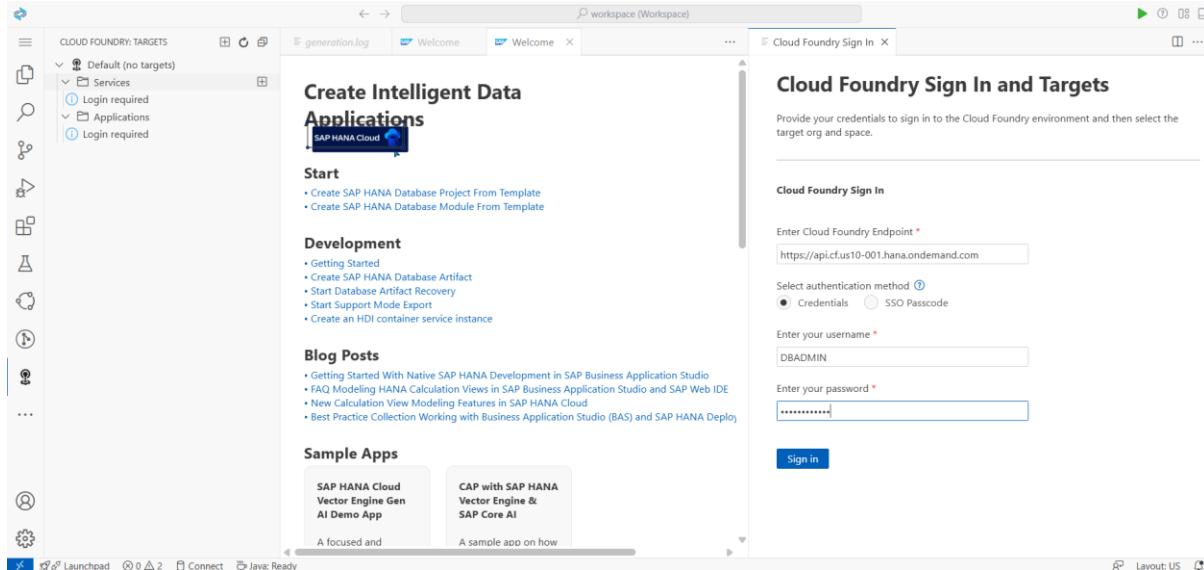


Image 6.4- Entering Cloud Foundry username and password for authentication

## ◆ Step 7: Clean Up pom.xml File

After signing in, navigate to the \*\*pom.xml\*\* file inside the \*\*srv\*\* folder and **remove the existing plugin step** that is not required for your CAP Java project setup.

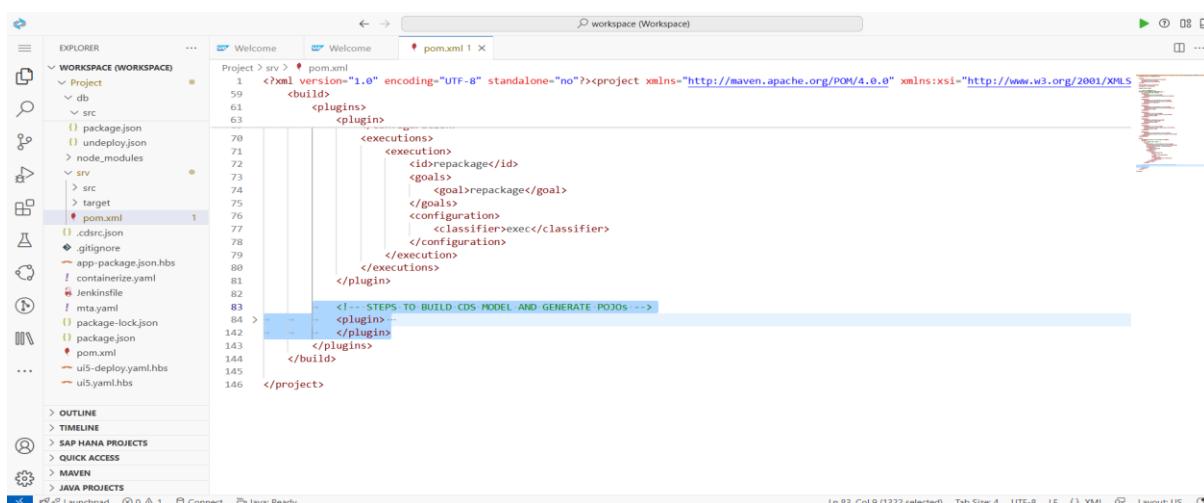


Image 7.1- Removing the unnecessary plugin from pom.xml inside srv

## ◆ Step 8: Temporarily Delete the db Folder

For now, right-click on the \*\*db\*\* folder and select **Delete**. This step is temporary — you can recreate the database module later as needed for your project.

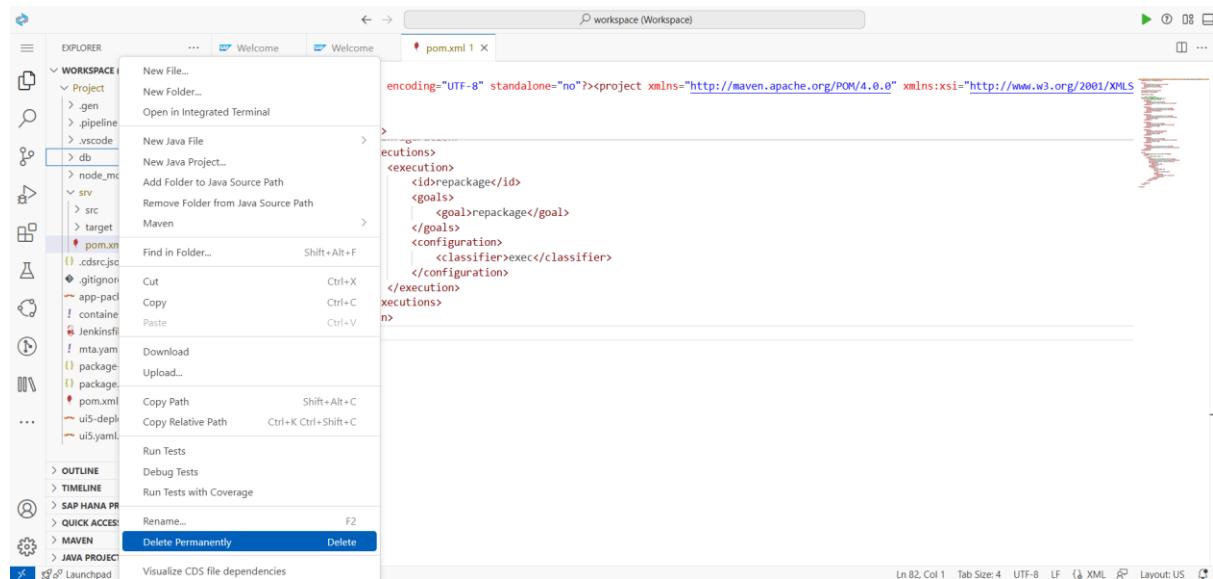


Image 8.1- Deleting the db folder temporarily – will recreate later

### ◆ Clear the application.yaml File

Navigate to the \*\*application.yaml\*\* file and remove its existing content. This will help you start fresh with your own custom configuration later in the project.

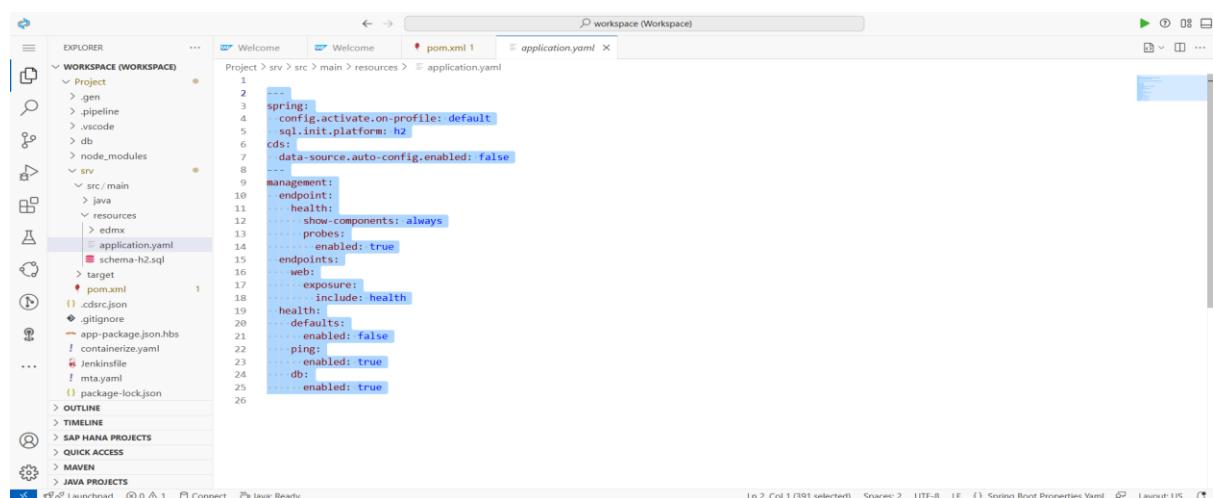


Image 8.2- Clearing contents of application.yaml to start with a fresh configuration

#### ◆ Delete schema-h2.sql File

Since it's not required for the current setup, right-click on the \*\*schema-h2.sql\*\* file and select **Delete** to remove it from the project.

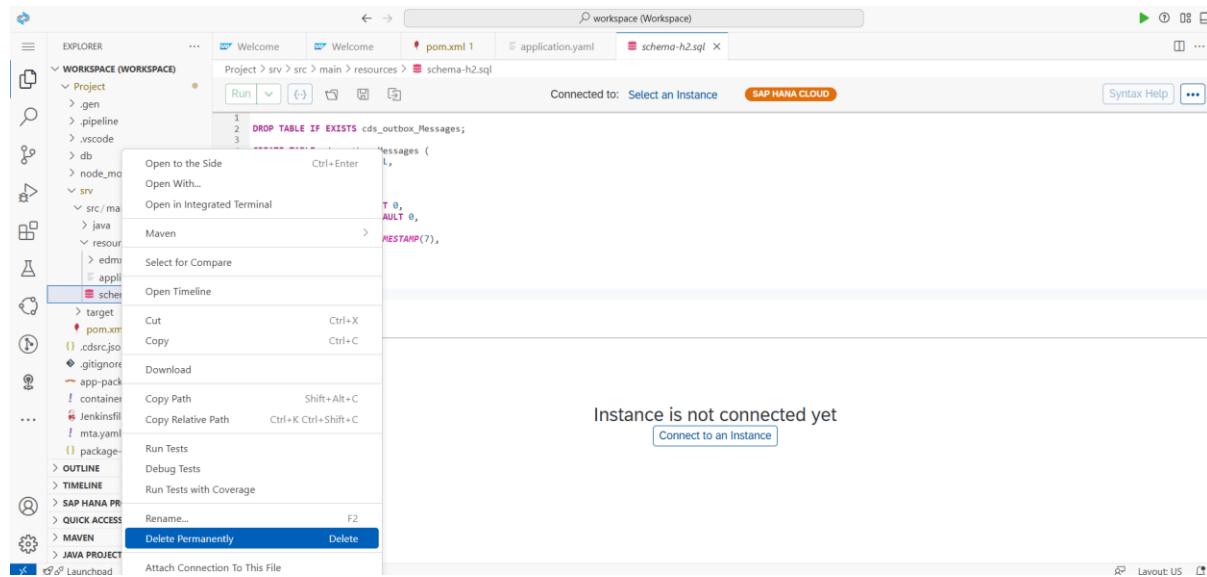


Image 8.3- Deleting the unnecessary schema-h2.sql file

#### ◆ Create MTA Module from Template

Right-click on the \*\*mta.yaml\*\* file and select “**Create MTA Module from Template**” to start adding a new module to your CAP project.

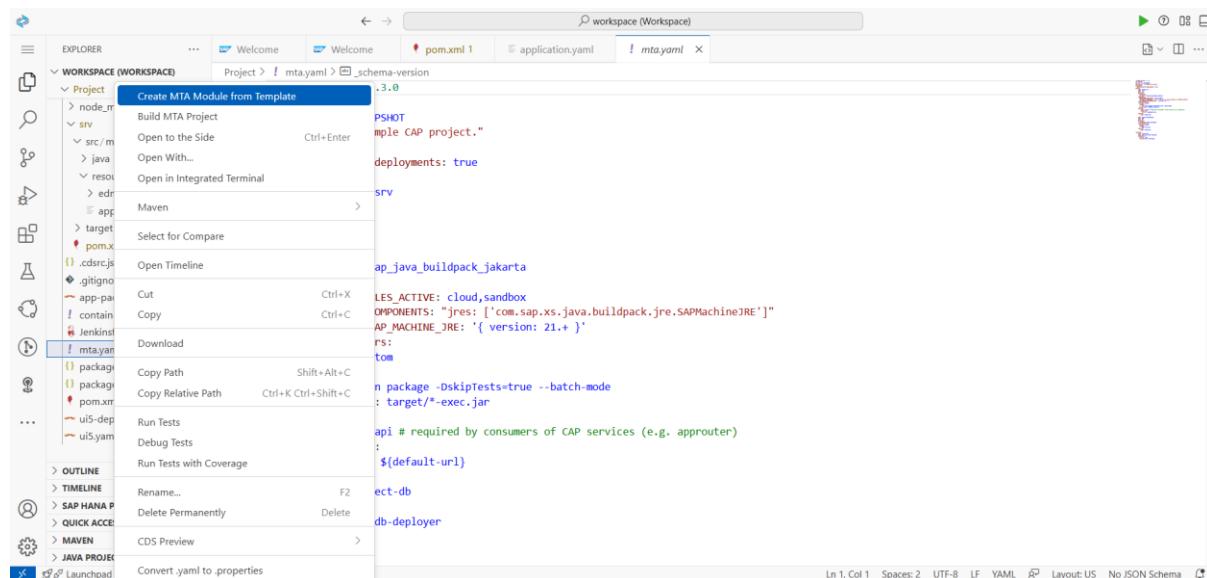


Image 8.4- Creating an MTA module from template by right-clicking on mta.yaml

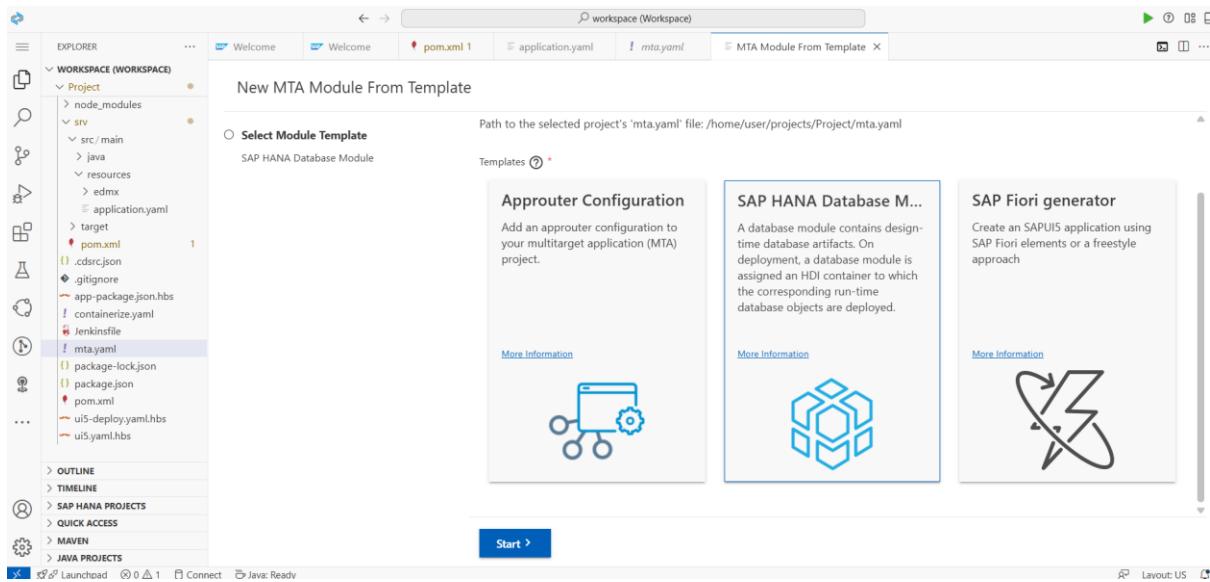


Image 8.5- Select “SAP HANA Database Module” in the templates and click “Start”

#### ◆ Configure HANA Module Properties

Set the basic properties for the new module and provide the **Module Name** as \*\*db\*\* and click next

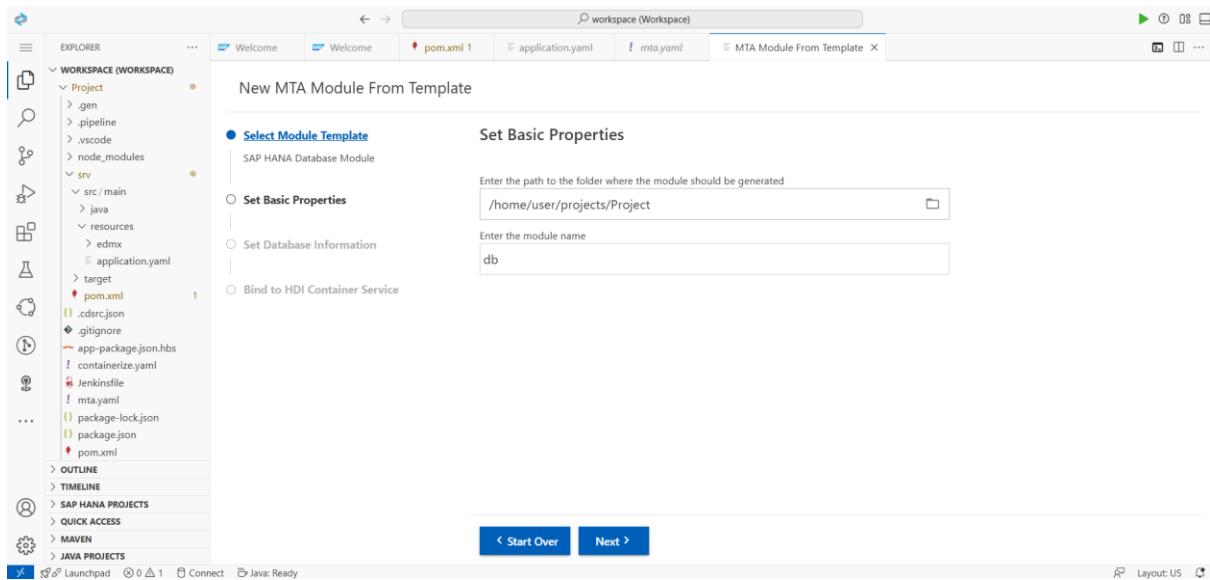


Image 8.6- Setting basic properties and naming the module as db

#### ◆ Set the Database Information

New MTA Module From Template

**● Select Module Template**

SAP HANA Database Module

**● Set Basic Properties**

**○ Set Database Information**

**○ Bind to HDI Container Service**

**Set Database Information**

Define details of your application's database module.

Namespace  
db

Schema Name  
db

SAP HANA Database Version \*  
SAP HANA Cloud

Bind the database module to a run-time environment service instance? [?](#)

Yes  No

Select the type of target container \*  
SAP HANA Cloud HDI container service

Configure additional deployment parameters? [?](#)

Yes  No

[◀ Back](#) [Next ▶](#)

nect Java: Ready

Image 8.7- Setting database information for the SAP HANA module

- ◆ **Bind to HDI Container Service**
- In this step, **bind the SAP HANA Database Module to an HDI Container Service** to manage your database artifacts securely within the SAP BTP environment.

New MTA Module From Template

**● Select Module Template**

SAP HANA Database Module

**● Set Basic Properties**

**● Set Database Information**

**○ Bind to HDI Container Service**

**Bind to HDI Container Service**

You can configure the HDI container service binding if needed.

Create a new HDI service instance?  
 Yes  No

Please enter a unique and non-existing service instance name  
Project-hidb-ws-jza1

Use the default database instance of the selected Cloud Foundry space?  
 Yes  No

[◀ Back](#) [Finish ▶](#)

Java: Ready Layout: US

Image 8.8- Binding the module to an HDI Container Service

## ◆ Step 9: Run the Application

Click on “Run” or select “Run and Debug” from the left panel to start your CAP application locally in SAP Business Application Studio.



Image 9.1- Running the CAP application using “Run and Debug

#### ◆ Select Java and Project

When prompted, choose **Java** as the runtime environment and select your **CAP project folder** to run the application.

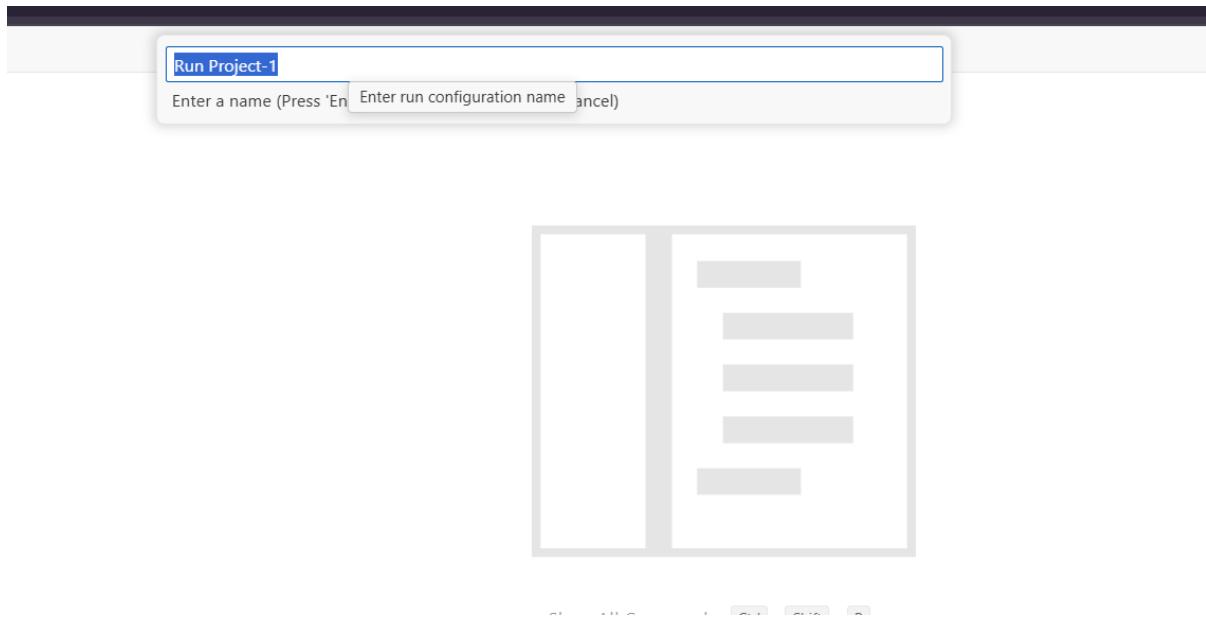


Image 9.2- Selecting Java runtime and the CAP project folder to run the app

#### ◆ Service Started Successfully

Once everything is configured correctly, your **CAP service will start running**, and you'll see confirmation in the terminal that the application is up and running.

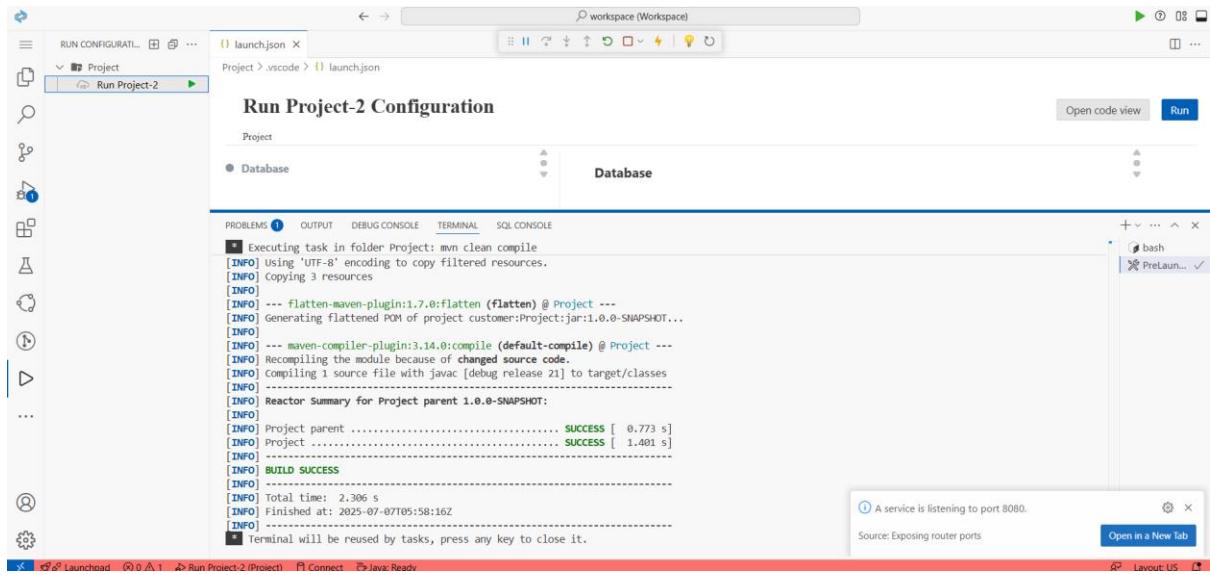


Image 9.3- CAP service started successfully and running in the terminal

## ◊ Part 2: Database Connection Setup

In this section, we'll configure and connect the CAP project to the SAP HANA Cloud database using the HDI container.

## ◆ Step 1: Navigate to SAP HANA Cloud

In your **SAP BTP Trial Account**, select **SAP HANA Cloud** to begin setting up your database instance.

The screenshot shows the SAP BTP Cockpit interface for a trial account. The left sidebar contains a navigation menu with sections like Overview, Services, Service Marketplace, Instances and Subscriptions (which is currently selected), Cloud Foundry, HTML5 Applications, Connectivity, Security, and Legal Information. A red arrow points to the 'SAP HANA Cloud' entry under the Instances and Subscriptions section. The main content area displays the 'Subaccount: trial - Instances and Subscriptions' page. It shows two Subscriptions: 'SAP HANA Cloud' (Subscribed, 3 Jul 2025) and 'SAP Business Application Studio' (Subscribed, 2 Jul 2025). Below that, the 'Instances (2)' section lists two instances: 'Database' (SAP HANA Schema..., hdi-shared, Cloud Foundry, dev, 1 key, Created) and 'DBADMIN' (SAP HANA Cloud, hana, Cloud Foundry, dev, Created).

Image 1.1- Selecting SAP HANA Cloud from the trial account navigation menu

## ◆ Step 2: Ensure the Database is Running

Before proceeding, **make sure your SAP HANA Cloud database instance is in the “Running” state**. This is essential for successful binding and data operations.

The screenshot shows the SAP HANA Cloud Central interface. On the left, there's a sidebar with options like Instances, Migrations, Alerts, SQL Console, Database Objects, and Data Lake Files. The main area is titled 'Instances' and shows a table for 'All Instances'. The table has columns for State, Name, Notifications, Runtime Environment, Memory, Storage, Compute, Nodes, and Actions. One row is listed: 'DBADMIN' is Running, Name is DBADMIN, Runtime Environment is Cloud Foundry, Memory is 16 GB, Storage is 80 GB, Compute is 1 vCPUs, and Nodes is 1 node. There are also buttons for 'Create Instance' and 'Adapt Filters'.

Image 2.1- Verifying that the SAP HANA Cloud database is running

## ◆ Step 3: Open Database Explorer

Navigate to the **Database Explorer** and click on the “+” (plus) symbol to add your HDI container and start exploring or managing database artifacts.

The screenshot shows the SAP HANA Database Explorer. On the left, there's a tree view under 'Filter Instances' showing 'DBADMIN (DBADMIN)' expanded, revealing 'Catalog', 'Database Diagnostic Files', and 'HDI Containers'. A red arrow points to the '+' icon at the top left of the tree view. The main pane shows a table for 'DBADMIN' with a single row for 'Catalog'. The bottom right corner shows a status bar with '1/0'.

Image 3.1- Opening the Database Console and clicking the “+” icon to add HDI container

## ◆ Step 4: Select HDI Container Instance Type

In the popup window, set the **Instance Type** to **HDI Container** to connect your CAP

project's database module to the correct SAP HANA schema and select the database so it will automatically generate key .

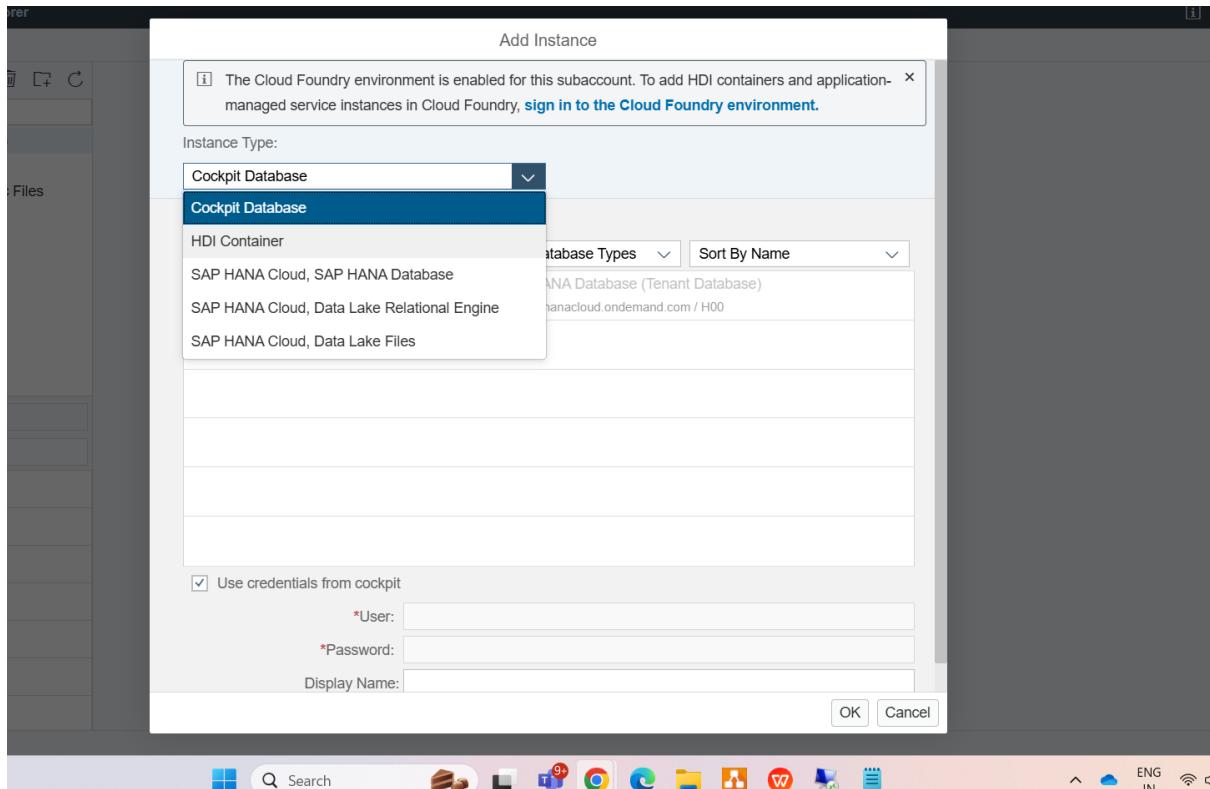


Image 4.1- “HDI Container” as the instance type in the Database Explorer and select the database you created

- ◆ **Step 5: Convert application.yaml to application.properties**

Right-click on the \*\*application.yaml\*\* file inside the resources folder and select the option “**Convert to Properties**”. This will automatically generate the equivalent application.properties file for easier configuration.

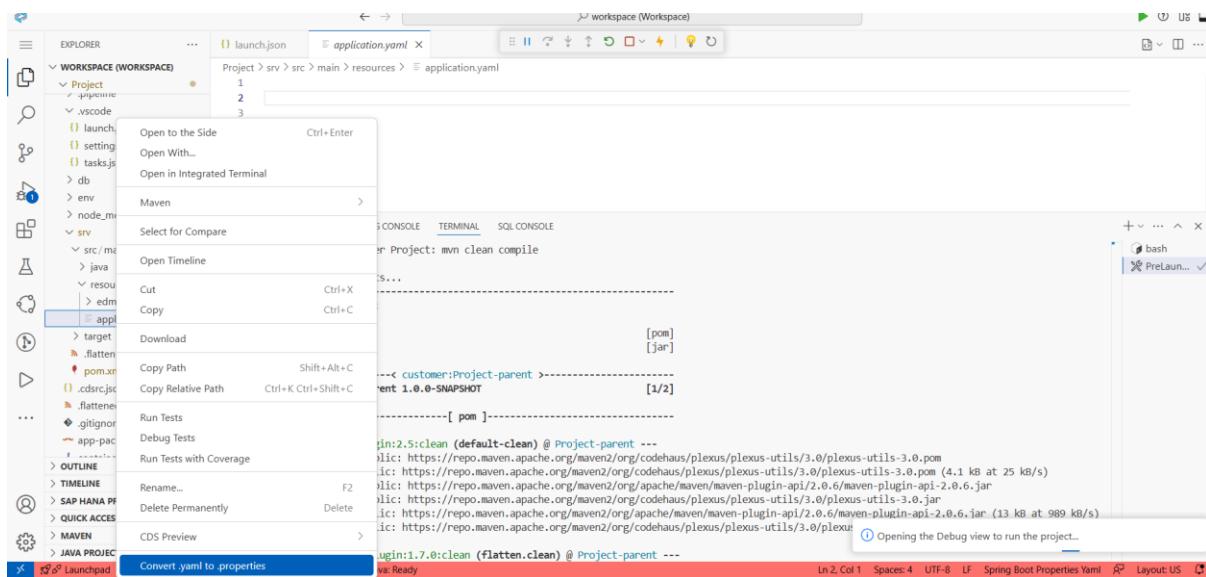


Image 5.1- Right-clicking application.yaml and selecting “Convert to Properties”

## ◆ Step 6: Verify Automatically Created Credentials

Once the **HDI Container** is set in the database, navigate to the **Instance** section. You'll notice that a **Service Key** is automatically generated under **Credentials**, which will be used for secure connectivity between your CAP application and the database.

The screenshot shows the SAP BTP Cockpit's Instances and Subscriptions page. The left sidebar has sections for Overview, Services, Cloud Foundry, HTML5 Applications, Connectivity, Security, and Legal Information. The main area displays a table of service instances. One row for the 'DBADMIN' instance in the 'SAP HANA Cloud' service plan is highlighted, showing a '1 key' credential under the 'Credentials' column. A large red arrow points from the bottom right towards this '1 key' entry.

Instance	Service	Plan	Runtime Environment	Scope	Credentials	Status
Database	SAP HANA Schema...	hdi-shared	Cloud Foundry	dev	<a href="#">1 key</a>	<span>Created</span>
DBADMIN	SAP HANA Cloud	hana	Cloud Foundry	dev	<a href="#">1 key</a>	<span>Created</span>
Project-hdibw-ws-	SAP HANA Schema...	hdi-shared	Cloud Foundry	dev	<a href="#">1 key</a>	<span>Created</span>

Image 6.1- Viewing the auto-generated key under credentials in the HDI container instance

## ◆ Step 7: View HDI Container Credentials

Click on the **Service Key** created under your HDI Container instance. You'll be able to see the detailed **credentials**, including the **username**, **password**, **URL**, and other connection details required to integrate with your CAP application.

The screenshot shows a Fiori application interface for managing HDI Container instances. The main title is 'HDI Container: trial - Instances and Subscriptions'. A modal dialog is open, titled 'Credentials'. Inside the dialog, there is a dropdown menu showing the URL: 'sap.default/930c4f00-bd9b-410d-a5f4-7d3b5a6bdd72#likitha@inflexiontechfze.com:D0@44b8c055-6f8c...'. Below the dropdown are two tabs: 'Form' (selected) and 'JSON'. The JSON content is displayed in a code editor-like area:

```
1  [
2   "url": "jdbc:sap://b182a574-fbc4-44ea-a847-80d29d9c4aa4.hana.trial-us10.hanacloud.ondemand
 .com:443?encrypt=true&validateCertificate=true&currentSchema
 =74595E83C1004A0A89053F90E22B8580",
3   "certificate": "-----BEGIN CERTIFICATE-----\nMIIDrzCCApegAwIBAgIQCDvgVpBCRrGhdWrJWZHHSjANBgkqhkiG9w0BAQUFADBh\nMQswCQYDVQQGEwJ
 VUzEVMBMGA1UEChMMRG1naUN1cnQgSW5jMRkwYDVQQLExB3\\nd3cuZG1naW1ncnQuY29tMSAwHgYDVQDExdEa
 WdpQ2VydCBhbG9iYWlgUm9vdCBD\\nQTAeFw0wNjExMTAwMDAwMDBaFw0zMTExMTAwMDAwMDBaMGExCzAJBgNVBAY
 TA1VT\\nMRUwEwYDVQKExwEaWdpQ2VydCBjbmMxGTAXbgNVBAsTEh3dy5kaWdpY2VydC5j\\nb20xIDAeBgnVBAM
 TF0RpZ21DZXJ0IEdsb2JhbCBSb290IENBMIIBJANBgkqhkiG\\n9w0BAQEFAAOCAQ8AMIIBCgKCAQE4jvhEXLeq
 KTT01eqUKKPC3eQyaK17hL011sB\\nCSDMAZ0nTjC3U/dDxGkAV53jSLdhwZAAIEJzs4bg7
 /fzTtxRulWZscFs3YnFo97\\nnh6Vfe63SKMII2tavegw5BmV
 /S10fvBf4q77uKNd0f3p4mVmFaG5cIzJlV07A6Fpt\\n43C/dxC//AH2hdmoRBByMq11GNXRor5H4idq9Joz
 +EkIYIVuUX7Q6hL+hqkpMFT7P\\nT19sd16gSzeRntwi5m30FBq0asv+zbMUZBfHWymemr/y7vrTC0LUq7dBMoM10
 /4\\ngdW7jVg/tRvoSSiicNoxBN3shbyTApOB6jtSj1etX
 +jkMOvJwIDAQABo2MwYTAO\\nBgnVHQ8BAf8EBAMCAYyDwYDVR0TAQH/BAUwAwEB
 /zAdBgNVHQ8EFgQUA95QNVbR\\nTLtm8KPiGxvD17I90VUwHwYDVR0jBBgwFoAU95QNVbRTLtm8KPiGxvD17I90V
 Uw\\nDQYJKoZIhvclNAQEFBQADggEBAMucN6pIExIK+t1EnE9sPTfrgT1eXkIoyQY/Esr\\nhMatudXH
 /vTBH1jLuG2cenTnmCmrEbXjcKChzUyImZOMXDiqw8cvpOp/2PV5Adg\\n060
 /nVsJ8dW041P0jmP6P6fbtGbFYmbW0W5bjfIttep3Sp+dWOIrWcBAI
 +0+KT1F\\nPnIIkisV\\TRTnDfvrRN7FVRhbn0n07WcRRrAI\\nraII
```

At the bottom of the modal, there are three buttons: 'Copy JSON', 'Download', and 'Close'.

Image 7.1- Viewing detailed credentials from the HDI container service key

**Credentials**

```

sap.default/930c4f00-bd9b-410d-a5f4-7d3b5a6bdd72#likitha@inflexiontechfze.com:D0@44b8c055-6f8c...
  
```

**Form**      **JSON**

```

1-----CERTIFICATE-----
2-----BEGIN CERTIFICATE-----
3-----END CERTIFICATE-----
4   "database_id": "b182a574-fbc4-44ea-a847-80d29d9c4aa4",
5   "driver": "com.sap.db.jdbc.Driver",
6   "hdi_password": "Td7z0xRnocjt6NrDpz03i49M70gKRgctB06TF98Cxjn9hc
                    -186e6umGAMOKk1pZFDeebTAKdjsNo_WfmnJyY0gHRSmCeW2C7Fd3yXCIJXHz018JbrZDNGuY644e58J3",
7   "hdi_user": "74595E83C1004A0A89053F90E22B8580_A18QX38835CQ0768PPTJM58X6_DT",
8   "host": "b182a574-fbc4-44ea-a847-80d29d9c4aa4.hana.trial-us10.hanacloud.ondemand.com",
9   "password": "Zz3ewVmXTATQidEPHL1iYZ5oPUgYnfFe4g3geo0dLex4jwyBxKSsyv5TKyetsbaPDz8_b7YHCuHZxBM3
                    iugzC60fEcEcU3YcciosqaAm5i2GUPq_ofm_Qe8rZtTu2MoS",
10  "port": "443",
11  "schema": "74595E83C1004A0A89053F90E22B8580",
12  "user": "74595E83C1004A0A89053F90E22B8580_A18QX38835CQ0768PPTJM58X6_RT"
13 }
  
```

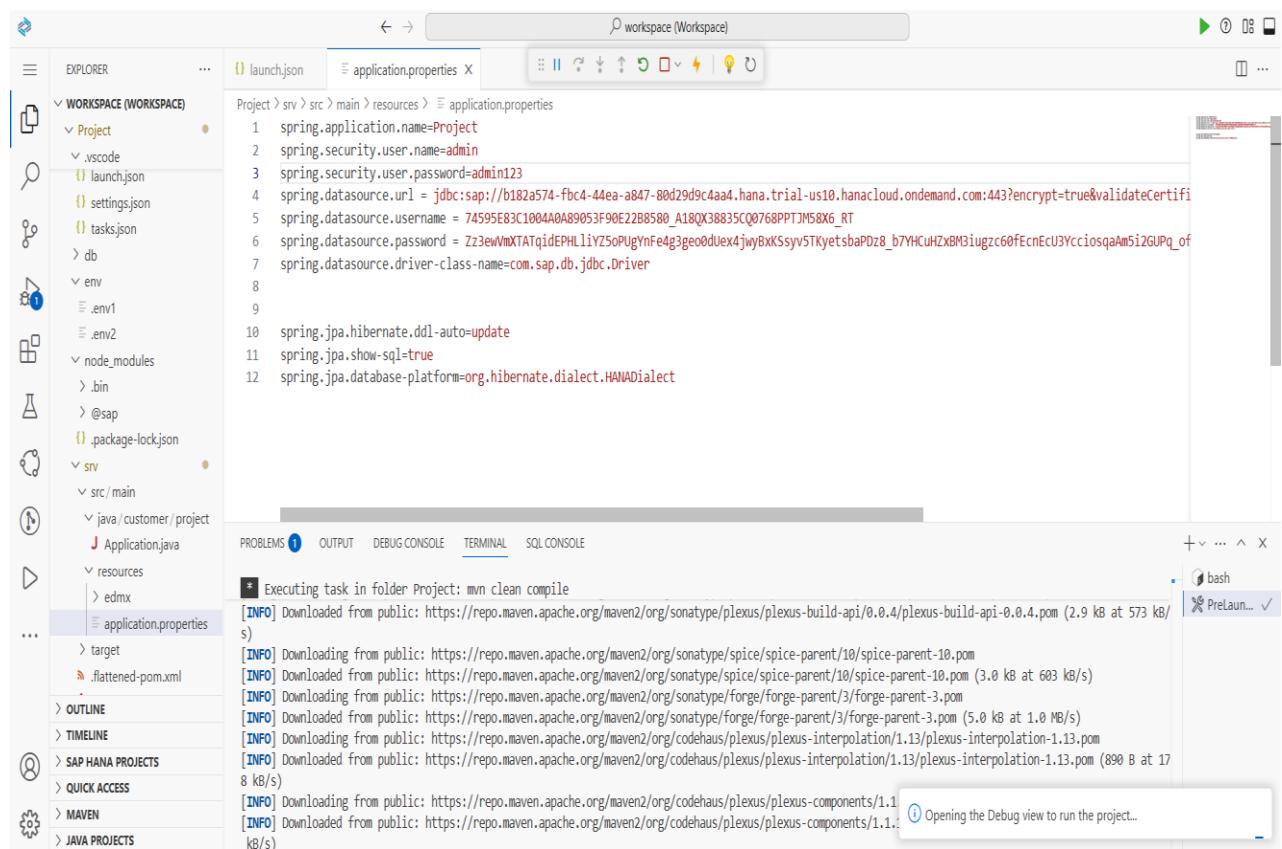
**Copy JSON**   **Download**   **Close**

Image 7.2- Viewing detailed credentials from the HDI container service key

## ◆ Step 8: Configure Credentials in application.properties

Copy the required details (like **URL**, **username**, and **password**) from the HDI container service key and set them in the **application.properties** file using the basic syntax format below:

- 1) spring.datasource.url=your\_hdi\_ur
- 2) spring.datasource.username=your\_username
- 3) spring.datasource.password=your\_password
- 4) spring.datasource.driver-class-name=com.sap.db.jdbc.Driver



```
Project > srv > src > main > resources > application.properties
1 spring.application.name=Project
2 spring.security.user.name=admin
3 spring.security.user.password=admin123
4 spring.datasource.url = jdbc:sap://b182a574-fbc4-44ea-a847-80d29d9c4aa4.hana.trial-us10.hanacloud.ondemand.com:443?encrypt=true&validateCertifi
5 spring.datasource.username = 74595E83C100A40A8905F90E22B8580_A180X38835CQ0768PPTJM58X6_RT
6 spring.datasource.password = Zz3ewmXTATqidePHLliyZ50PUgYnFe4g3ge0duex4jwyBxKssyv5TKyetsbaPDz8_b7YHCuHx8MBiugzc60fEcneCu3YcciosqaAM5i2GUPq_
7 spring.datasource.driver-class-name=com.sap.db.jdbc.Driver
8
9
10 spring.jpa.hibernate.ddl-auto-update
11 spring.jpa.show-sql=true
12 spring.jpa.database-platform=org.hibernate.dialect.HANADialect
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    SQL CONSOLE

Executing task in folder Project: mvn clean compile

```
[INFO] Downloaded from public: https://repo.maven.apache.org/maven2/org/sonatype/plexus/plexus-build-api/0.0.4/plexus-build-api-0.0.4.pom (2.9 kB at 573 kB/s)
[INFO] Downloading from public: https://repo.maven.apache.org/maven2/org/sonatype/spice/spice-parent/10/spice-parent-10.pom
[INFO] Downloaded from public: https://repo.maven.apache.org/maven2/org/sonatype/spice/spice-parent/10/spice-parent-10.pom (3.0 kB at 603 kB/s)
[INFO] Downloading from public: https://repo.maven.apache.org/maven2/org/sonatype/forge/forge-parent/3/forge-parent-3.pom
[INFO] Downloaded from public: https://repo.maven.apache.org/maven2/org/sonatype/forge/forge-parent/3/forge-parent-3.pom (5.0 kB at 1.0 MB/s)
[INFO] Downloading from public: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-interpolation/1.13/plexus-interpolation-1.13.pom
[INFO] Downloaded from public: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-interpolation/1.13/plexus-interpolation-1.13.pom (890 B at 17
8 kB/s)
[INFO] Downloading from public: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-components/1.1.1/plexus-components-1.1.1.pom
[INFO] Downloaded from public: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-components/1.1.1/plexus-components-1.1.1.pom (1.1 kB/s)
```

Opening the Debug view to run the project...

Image 8.1- Setting HDI container credentials in application.properties

## ◆ Step 9: Add Required Dependencies to pom.xml

To ensure smooth development and handle runtime issues, add the following dependencies to your \*\*pom.xml\*\* file:

1) <!-- Enables live reload and developer convenience tools -->

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <optional>true</optional>
</dependency>
```

2)<!-- Resolves HANA JDBC connection errors like:

```
    org.hibernate.exception.GenericJDBCException:
        unable to obtain isolated JDBC connection [SAP DBTech JDBC: [1890]: HANA
Database instance is stopped] -->
```

```
<dependency>
    <groupId>org.hibernate.orm</groupId>
    <artifactId>hibernate-core</artifactId>
</dependency>
```

3)<!-- Useful for testing Spring Security features -->

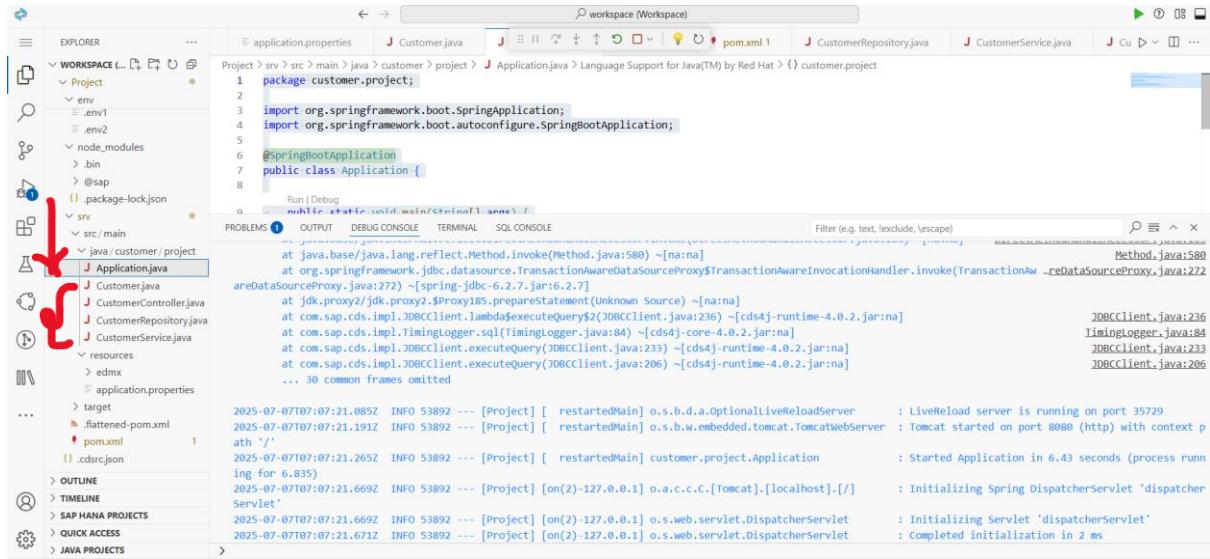
```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-test</artifactId>
    <scope>test</scope>
</dependency>
```

## ◆ Step 10: Add JPA Files – Entity, Repository, Service, and Controller

To structure your CAP Java project following standard Spring Boot practices, create the required **JPA components**:

- **Entity** – Represents your database table.
- **Repository** – Extends JpaRepository for CRUD operations.
- **Service** – Contains business logic.
- **Controller** – Exposes REST APIs.

 These components help in building a clean and maintainable backend using Spring Data JPA.



```

package customer.project;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class Application {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}

```

The screenshot shows the SAP Studio IDE interface. The left sidebar displays the project structure under 'WORKSPACE'. The 'EXPLORER' tab is selected, showing files like Application.java, Customer.java, CustomerController.java, CustomerRepository.java, and CustomerService.java. The right side shows the code editor with the Application.java file open, which contains the provided Java code. Below the code editor is the 'PROBLEMS' view, which lists several warnings related to JDBC proxy statements. At the bottom, the 'OUTPUT' view shows logs from the application's startup.

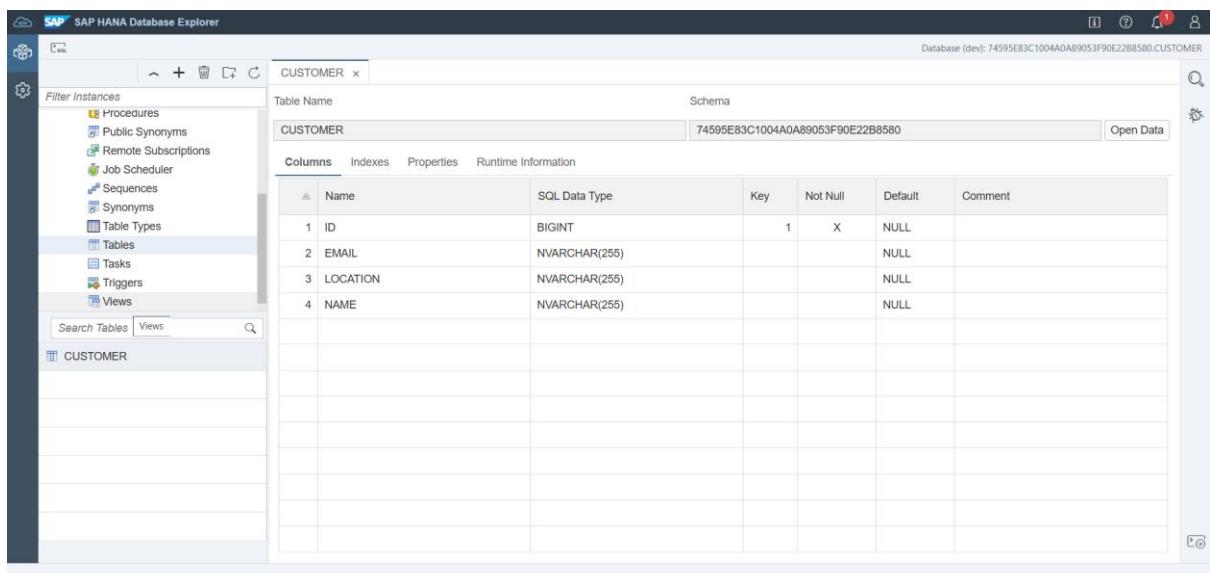
Image 10.1- Adding Entity, Repository, Service, and Controller layers for the CAP project

## ◆ Step 11: Auto-Creation of Tables from Entity Classes

When you run the application, the **JPA Entity classes** will automatically generate corresponding **tables in the SAP HANA Cloud database**.

To enable this behavior, make sure to include the following property in your **\*\*application.properties\*\*** file:

**spring.jpa.hibernate.ddl-auto=update**



Name	SQL Data Type	Key	Not Null	Default	Comment
1 ID	BIGINT	1	X	NULL	
2 EMAIL	NVARCHAR(255)			NULL	
3 LOCATION	NVARCHAR(255)			NULL	
4 NAME	NVARCHAR(255)			NULL	

The screenshot shows the SAP HANA Database Explorer interface. On the left, a tree view shows various database objects like Procedures, Public Synonyms, Remote Subscriptions, Job Scheduler, Sequences, Synonyms, Table Types, Tables, Tasks, Triggers, and Views. The 'Tables' node is selected. In the main pane, the CUSTOMER table is displayed with its columns: ID, EMAIL, LOCATION, and NAME. The table is defined with the schema 74595E83C1004A0A8905F90E22B8580.CUSTOMER.

Image 11.1- Caption: Auto-creating tables in SAP HANA from JPA entity classes using `spring.jpa.hibernate.ddl-auto=update`

## ◆ Part 3: Creation of webapp Folder (UI Layer)

In this part, we'll create the **\*\*webapp\*\*** folder which serves as the **UI layer** of your CAP application, typically used to build SAP Fiori or other frontend modules.

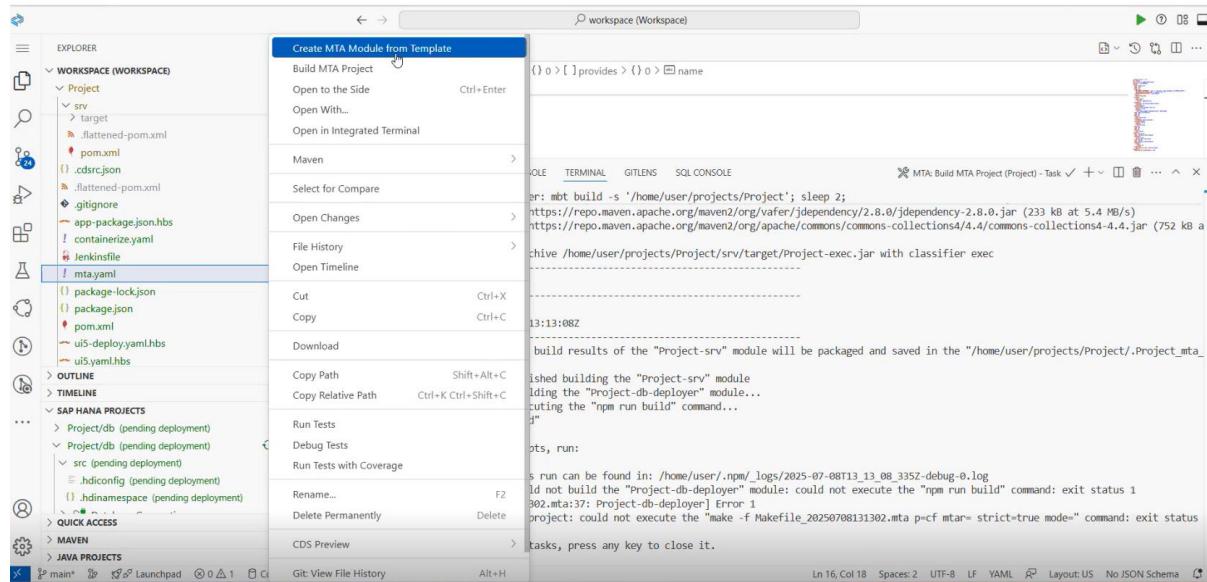


Image 1- click on `mta.yaml` and select “Create MTA Module from Template”

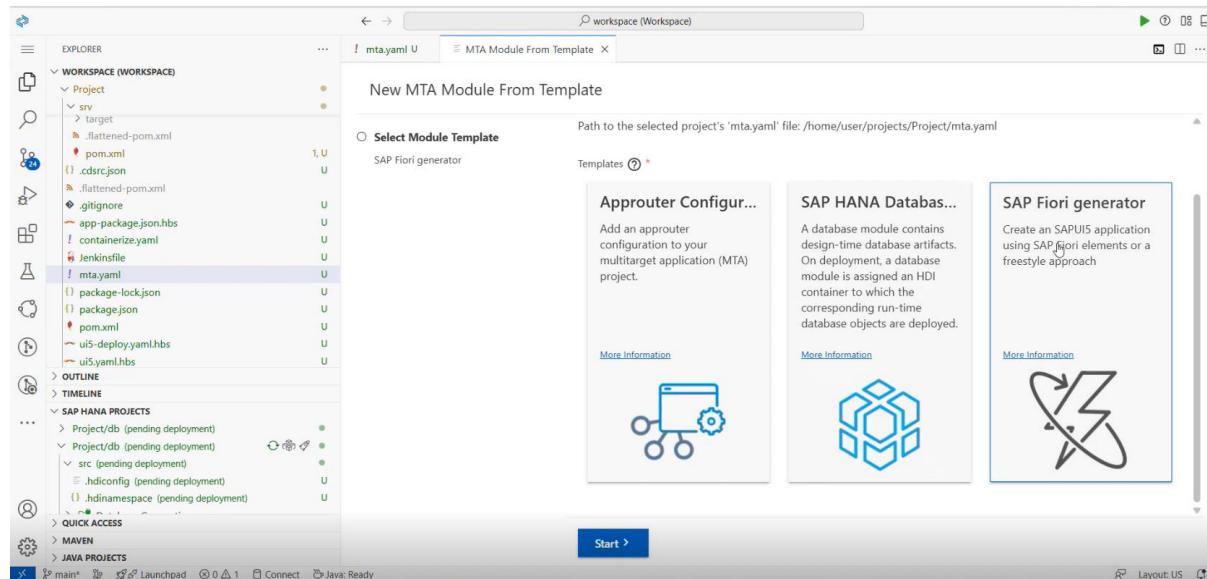


Image 2- Select “SAP Fiori Application” from the list of available templates

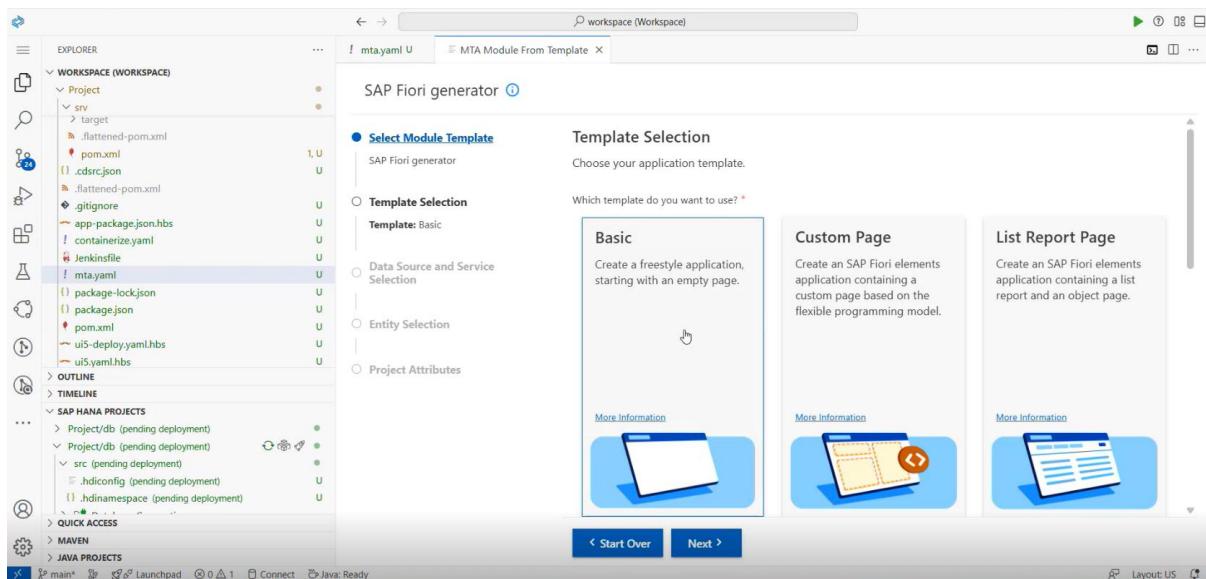


Image 3- Select the “Basic” template for the SAP Fiori application

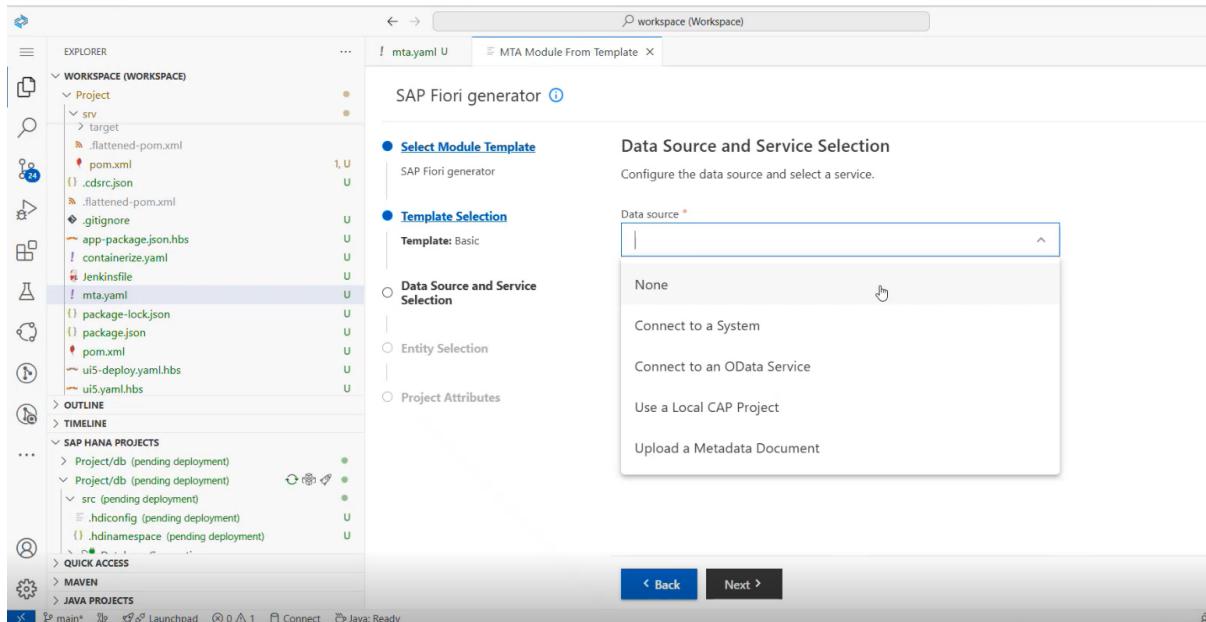


Image 4- Select “None” as the data source and click “Next”

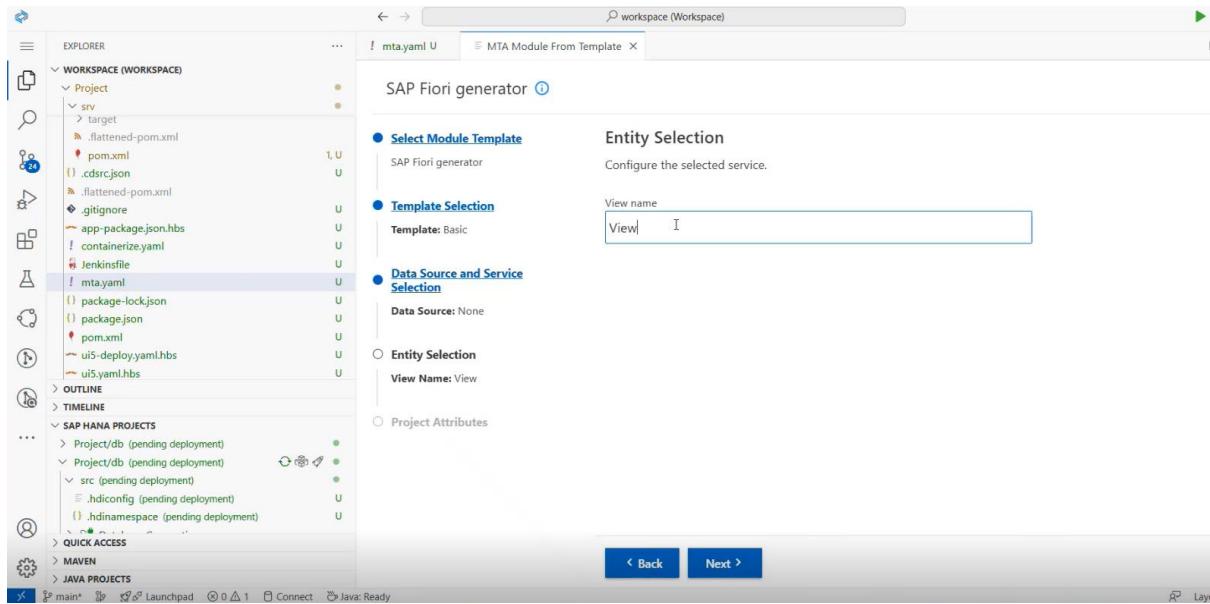


Image 5- Enter the view name as view and proceed

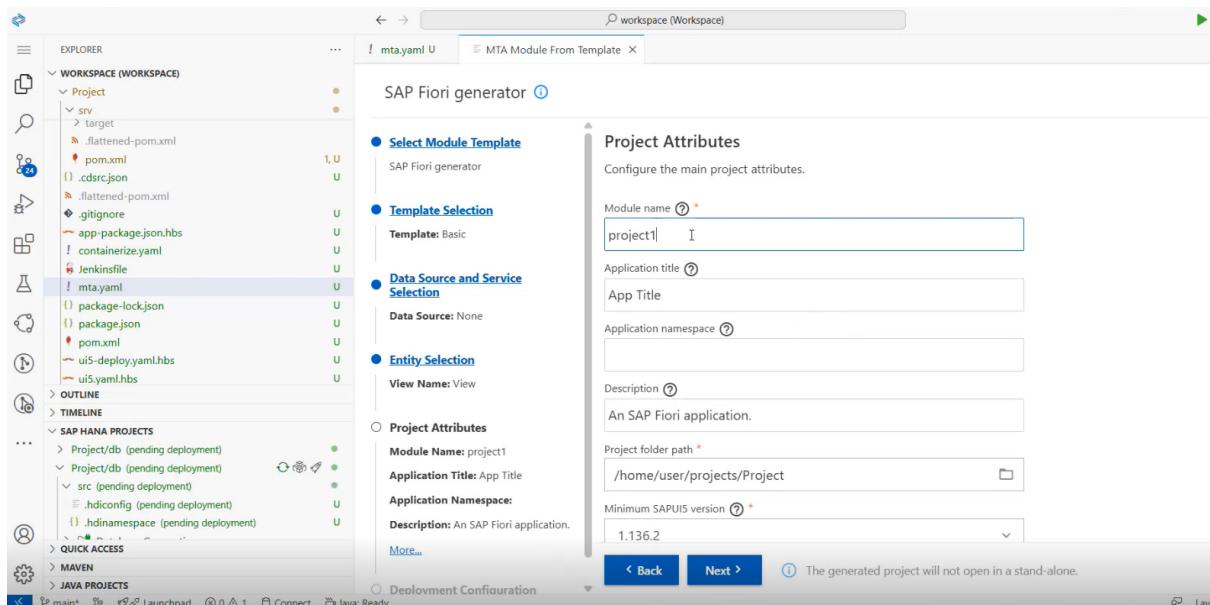


Image 6- Click “Next” to continue with the Fiori module creation

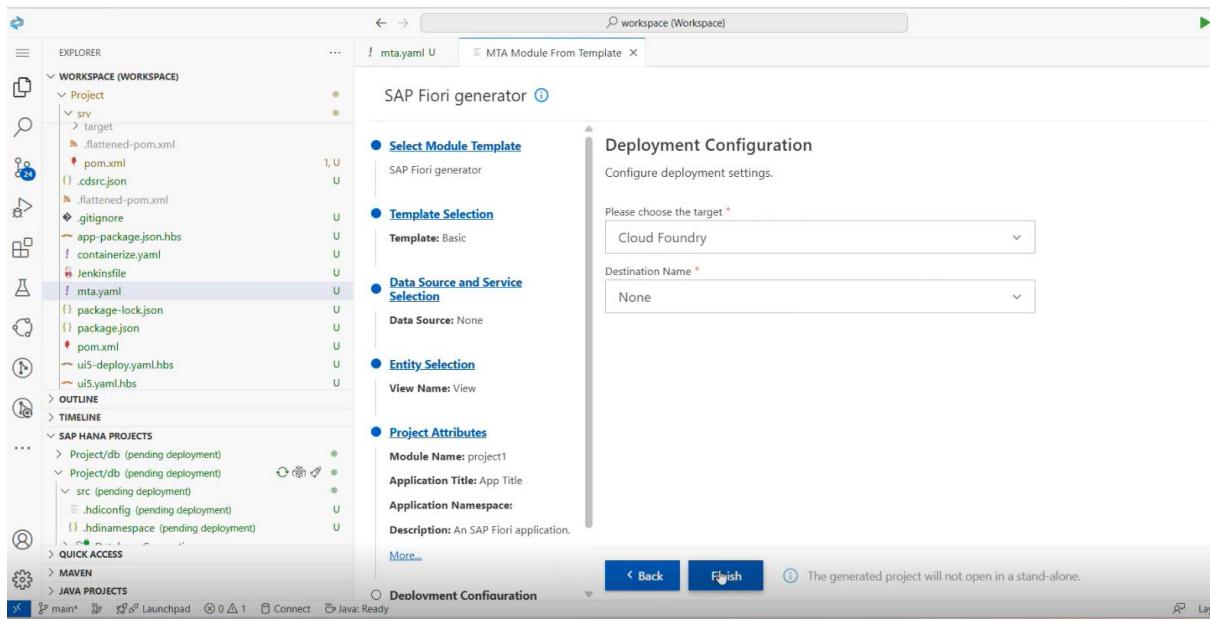


Image 7- Click “Next” to continue with the Fiori module creation

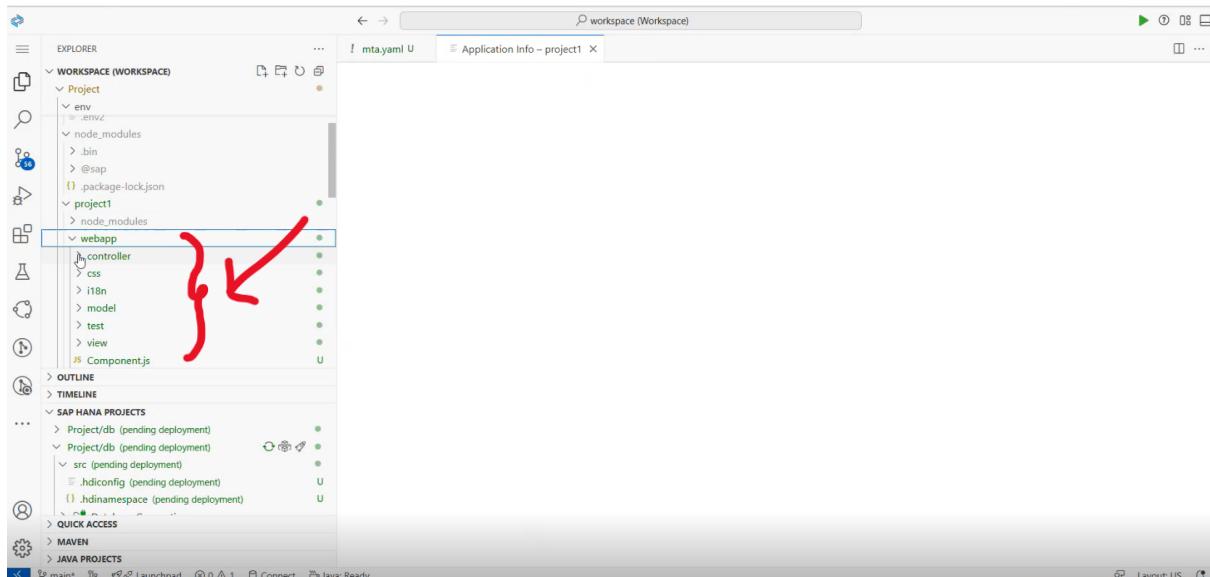


Image 8- webapp folder is successfully created after completing the Fiori module setup

## ◊ Part 5: Deploying the Project to GitHub

To continue development locally or customize the application further, you can now **deploy or open the project in Visual Studio Code**.

To deploy your CAP project from SAP Business Application Studio (BAS) to GitHub, follow these command-line steps:

**1 Initialize Git in Your Project Folder**

```
git init
```

**2 Add Remote GitHub Repository**

```
git remote add origin https://github.com/username/projectname.git
```

**3 Verify Remote Setup**

```
git remote -v
```

**4 Stage Your Changes**

```
git add .
```

**5 Commit Your Changes**

```
git commit -m "Initial commit"
```

**6 Publish Your Branch to GitHub**

Click on "**Publish Branch**" in the BAS Git panel.

👉 A browser window will open prompting you to **copy a verification code**.

**7 Authorize SAP BAS to Access GitHub**

Paste the copied verification code into GitHub and **authorize SAP Business Application Studio**.

**8 Pull Latest Changes from Main Branch (If Needed)**

```
git pull origin main --rebase
```

**9 Or Simply Pull**

```
git pull
```

