CLD200

SAP Extension Suite

PARTICIPANT HANDBOOK
INSTRUCTOR-LED TRAINING

Course Version: 20
Course Duration: 3 Day(s)
Material Number: 50157754

SAP Copyrights, Trademarks and Disclaimers

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors.

National product specifications may vary.

These materials may have been machine translated and may contain grammatical errors or inaccuracies.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP SE or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP SE or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE’s or its affiliated companies’ strategy and possible future developments, products, and/or platform directions and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, which speak only as of their dates, and they should not be relied upon in making purchasing decisions.

Typographic Conventions

American English is the standard used in this handbook.

The following typographic conventions are also used.

This information is displayed in the instructor's presentation	🖥
Demonstration	➤
Procedure	1 2 3
Warning or Caution	⚠
Hint	💡
Related or Additional Information	»
Facilitated Discussion	💬
User interface control	*Example text*
Window title	*Example text*

Contents

Course Overview

TARGET AUDIENCE

This course is intended for the following audiences:

- Application Consultant
- Development Consultant
- Super / Key / Power User
- Business Process Architect
- Business Process Owner/Team Lead/Power User
- Developer
- Enterprise Architect
- Solution Architect

UNIT 1 SAP Business Technology Platform (BTP)

UNIT OBJECTIVES

- Explain the SAP Business Technology Platform

Unit 1
Lesson 1

Introducing SAP Business Technology Platform (BTP)

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Explain the SAP Business Technology Platform

The Intelligent Enterprise

As an intelligent enterprise, you can keep your workforce engaged and increase organizational agility. You can deliver the products and services your customers need and deliver personal, trusted, and connected customer experiences. You can control every source and category of spend. You can increase efficiency and gain insight to guide your business. And you can make decisions confidently and drive continuous innovation.

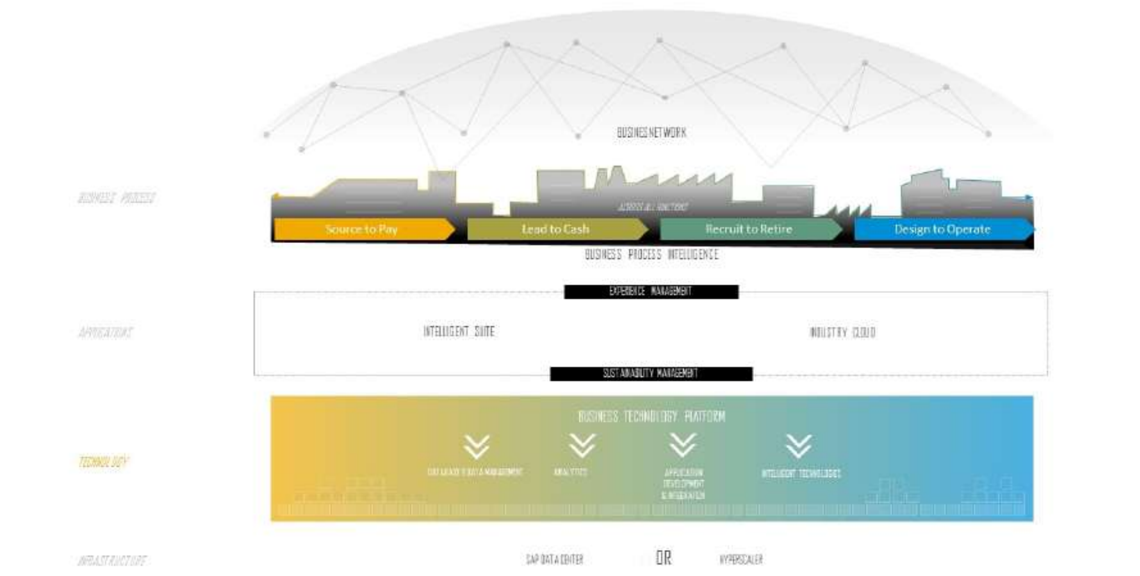


Figure 1: The Intelligent Enterprise

The figure The Intelligent Enterprise provides an overview of the Intelligent Enterprise.

Key dimensions of intelligent enterprises include:

- Assisted and automated business process execution.
- Actionable business insights.
- Innovative business models and practises.
- Collaborative networks.

- Experience management.

The intelligent enterprise comprises of:

- Business Network

  The SAP Business Network will help you to digitalize cross-company business processes. The network builds on current procurement, travel, and contingent workforce solutions to help intelligent enterprises work together to create flexible value chains.

- Experience Management

  Understanding what people want, and how they feel is critical to making the right decisions. Experience management solutions give insight on the sentiments and feelings of customers, employees, and other business stakeholders.

- Intelligent Suite

  SAP offers an integrated suite of applications that support your end-to-end business processes. The suite helps manage every part of your organization – employees, customers, products, spend, finance, and IT. With embedded analytics, we provide a 360-degree view of your business.

- Industry Cloud

  SAP’s industry cloud will enable you to discover and deploy vertical solutions from SAP and partners. These help you to apply leading-edge industry best practices and extend your current business processes.

- Sustainability Management

  Being best-run means running sustainably. SAP solutions for sustainability will help customers understand and manage their impact on people and the environment.

- SAP Business Technology Platform

  The platform provides data management and analytics, supports application development and integration, and allows you to use intelligent technologies – such as artificial intelligence, machine learning, and the Internet of Things – to drive innovation.

SAP Business Technology Platform

SAP Business Technology Platform (SAP BTP) brings together intelligent enterprise applications with database and data management, analytics, integration and extension capabilities into one platform for both cloud and hybrid environments, including hundreds of pre-built integrations for SAP and third-party applications.

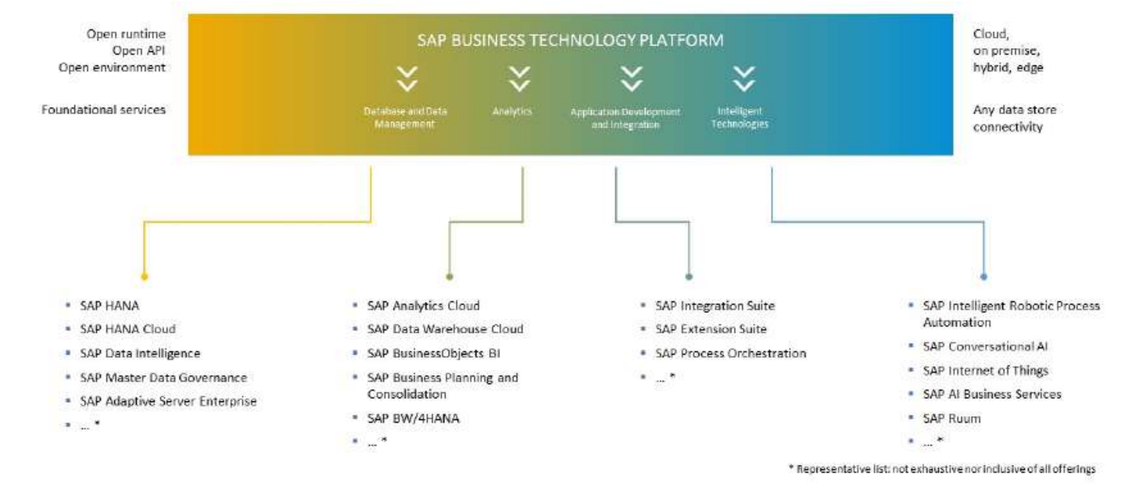

Open runtime
Open API
Open environment
Foundational services
SAP BUSINESS TECHNOLOGY PLATFORM
Database and Data Management
Analytics
Application Development and Integration
Intelligent Technologies
Cloud, on premise, hybrid, edge
Any data store connectivity
SAP HANA
SAP HANA Cloud
SAP Data Intelligence
SAP Master Data Governance
SAP Adaptive Server Enterprise
... *
SAP Analytics Cloud
SAP Data Warehouse Cloud
SAP BusinessObjects BI
SAP Business Planning and Consolidation
SAP BW/4HANA
... *
SAP Integration Suite
SAP Extension Suite
SAP Process Orchestration
... *
SAP Intelligent Robotic Process Automation
SAP Conversational AI
SAP Internet of Things
SAP AI Business Services
SAP Ruum
... *
* Representative list: not exhaustive nor inclusive of all offerings

Figure 2: SAP Business Technology Platform

The figure SAP Business Technology Platform provides an overview of the four key areas and the corresponding solutions for each area.

SAP BTP solution areas

- Database and Data Management:

  Capture, manage, and govern your data to drive better business outcomes.

- Analytics:

  Analyze all your data to accelerate insights and transform the data you have into the answers you need.

- Application Development and Integration

  Integrate and extend applications - build new ways to access and interact with your data.

- Intelligent Technologies:

  Data is the fuel propelling intelligent technologies forward - optimizing processes, and igniting innovation.

SAP BTP will be the unified, business-centric and open data and development platform for the entire SAP ecosystem. With unrivaled development efficiency, it will empower SAP developers, customers and partners alike to integrate, create value from data and extend in SAP landscapes. To achieve this, we bring our existing technology products (e.g. SAP HANA, SAP Integration Suite, or SAP Analytics Cloud) together with new capabilities (e.g. low-code/no code, marketplace) in one place for consumption. All built on a stable multi-cloud, multi-runtime foundation.

LESSON SUMMARY

You should now be able to:

- Explain the SAP Business Technology Platform

UNIT 2 Development Environment and CAP-Project

UNIT OBJECTIVES

- Get ready to build your extension project
- Explain the SAP BTP account model
- Identify the most suitable SAP BTP environment for a new cloud application project
- Describe key capabilities of SAP Business Application Studio (BAS)

- Create a CAP-based service
- Describe the OData standard for web-based applications
- Use APIs to exchange information in a standardized way
- Explain JSON and YAML

Get Started

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Get ready to build your extension project

SAP Extension Suite: Introduction

Business Case

Animation: SAP Extension Suite: Introduction

For more information on *SAP Extension Suite: Introduction*, please view the animation in the lesson *Get Started*, in your online course.

Your manufacturing company exports goods to foreign countries. Along this supply chain, day-to-day as well as exceptional risks that can significantly influence the export business are continuously identified and monitored. These risks are to be managed in a new application.

Your company manages all business processes via an SAP S/4HANA Cloud system. Customer extensions within the system itself, so-called in-app extensions, only meet individual requirements to a certain extent. For the required risk management application, the company decided to implement a side-by-side extension. Side-by-side extensions are developed using the SAP Extension Suite on the SAP Business Technology Platform (BTP) and offer more flexibility compared to in-app extensions.

The new risk-management application should be developed using the SAP Business Application Studio (BAS) on SAP BTP. There are several options and runtime environments available:

- ABAP Environment
- Cloud Foundry Runtime
- Kyma Runtime

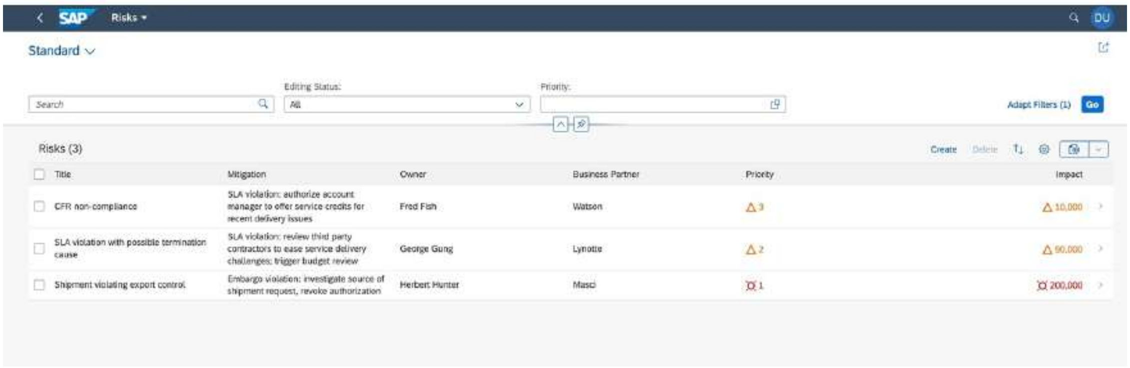
Before choosing a runtime, it is essential to familiarize yourself with the level of knowledge in your development department. If the developers only have ABAP knowledge, the decision for an extension written in ABAP, which in the end runs on the ABAP environment in the cloud, is quite obvious.

Your development team consists mainly of developers with Node.js knowledge. Therefore, the decision was made to develop an extension with SAP's Node.js-based Cloud Application Programming Model (CAP). The deployment will then take place on the SAP BTP, Cloud Foundry Environment - SAP Help Portal.

As part of our development team, you will develop a CAP-based application for risk management.

Course Preview

First, you will deploy your application manually into the SAP BTP, Cloud Foundry environment. After that, you will add authorizations and make your application ready for continuous integration and delivery.



Risks (3)

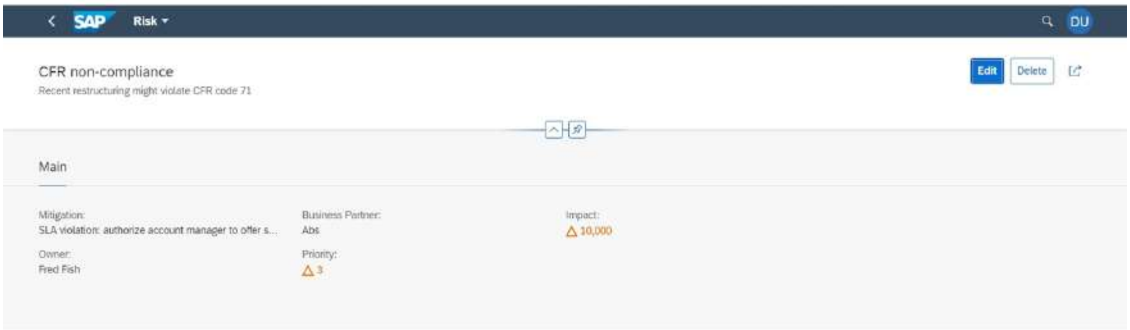
Title	Mitigation	Owner	Business Partner	Priority	Impact
CFR non-compliance	SLA violation: authorize account manager to offer service credits for recent delivery issues	Fred Fish	Watson	△ 3	△ 10,000
SLA violation with possible termination cause	SLA violation: review third party contractors to ease service delivery challenges; trigger budget review	George Gung	Lynotte	△ 2	△ 90,000
Shipment violating export control	Embargo violation: investigate source of shipment request, revoke authorization	Herbert Hunter	Masci	⊠ 1	⊠ 200,000

Figure 4: Risks List Page

Users can create, edit, and delete risks from this list page. Each risk has the properties of `impact` and `priority` based on each risk's potential.

The risk manager can assign mitigations to the identified risks. Both, risks and mitigations, are stored in the extension's own database. Details like the name of the individual business partner that is authorized to make decisions will be retrieved from our S/4HANA Cloud system.

To view and edit a single risk, the user can select an item. This opens the object page:



CFR non-compliance
Recent restructuring might violate CFR code 71

Mitigation: SLA violation: authorize account manager to offer s...
Business Partner: Abs
Impact: △ 10,000
Owner: Fred Fish
Priority: △ 3

Figure 5: Object Page

Caveat

Kindly note that we cannot provide a complete S/4HANA Cloud system for this course. Therefore, we have chosen to use the SAP API Business Hub S/4HANA Sandbox, which provides the same OData API for business partners as a real S/4HANA Cloud system.

Troubleshooting

In case you run into issues during your development process, we suggest re-reading the corresponding steps carefully.

What's Next?

The following units will guide you through an end-to-end development process from development to automated deployment.

Get Started

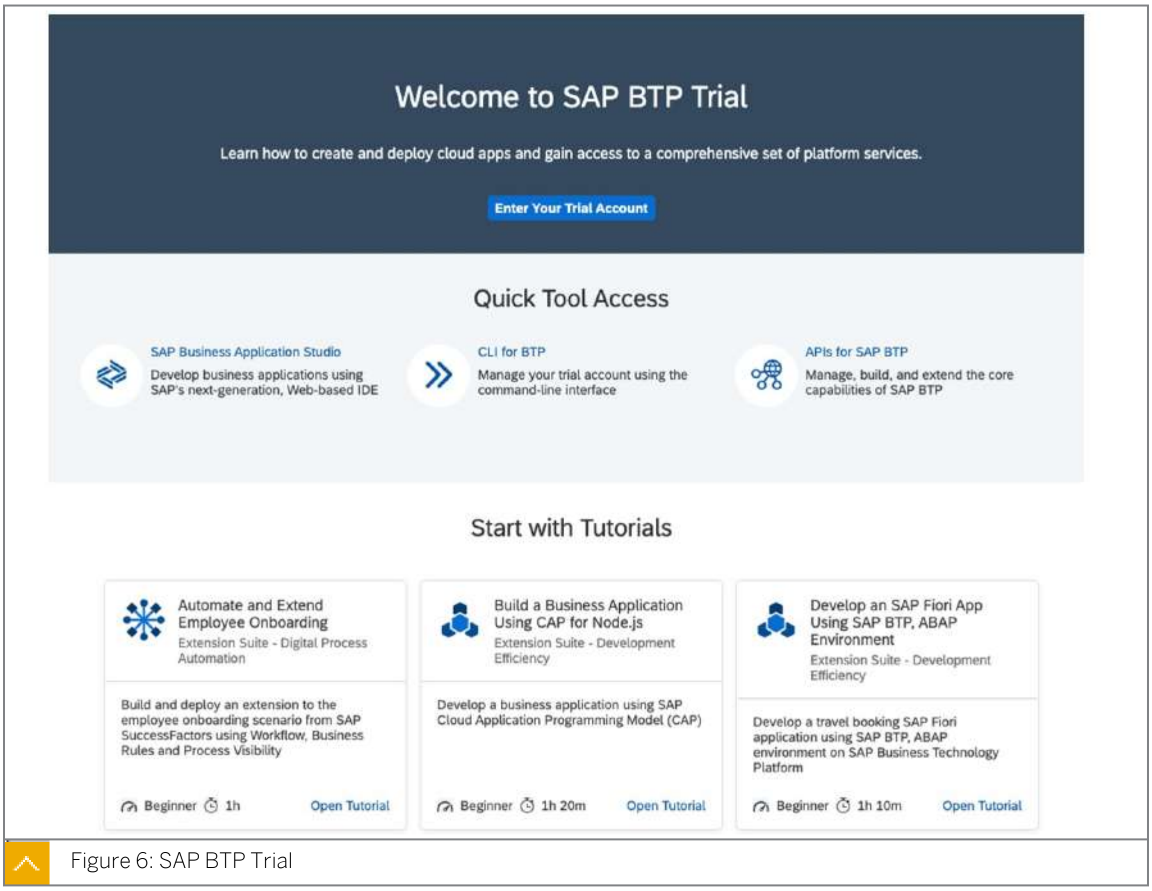
To build your extension project, there are a few things you need to do first:

- Request an SAP BTP Trial account.
- Create a Dev Space for Business Applications in SAP Business Application Studio.

SAP Business Technology Platform Trial

Access the tutorial Get a Free Account on SAP BTP Trial to learn how to request an SAP BTP trial account.

The SAP BTP Trial offering contains many of the most important services and tools for development on the platform.



Welcome to SAP BTP Trial

Learn how to create and deploy cloud apps and gain access to a comprehensive set of platform services.

Enter Your Trial Account

Quick Tool Access

SAP Business Application Studio
Develop business applications using SAP's next-generation, Web-based IDE

CLI for BTP
Manage your trial account using the command-line interface

APIs for SAP BTP
Manage, build, and extend the core capabilities of SAP BTP

Start with Tutorials

Automate and Extend Employee Onboarding
Extension Suite - Digital Process Automation
Build and deploy an extension to the employee onboarding scenario from SAP SuccessFactors using Workflow, Business Rules and Process Visibility
Beginner 1h Open Tutorial

Build a Business Application Using CAP for Node.js
Extension Suite - Development Efficiency
Develop a business application using SAP Cloud Application Programming Model (CAP)
Beginner 1h 20m Open Tutorial

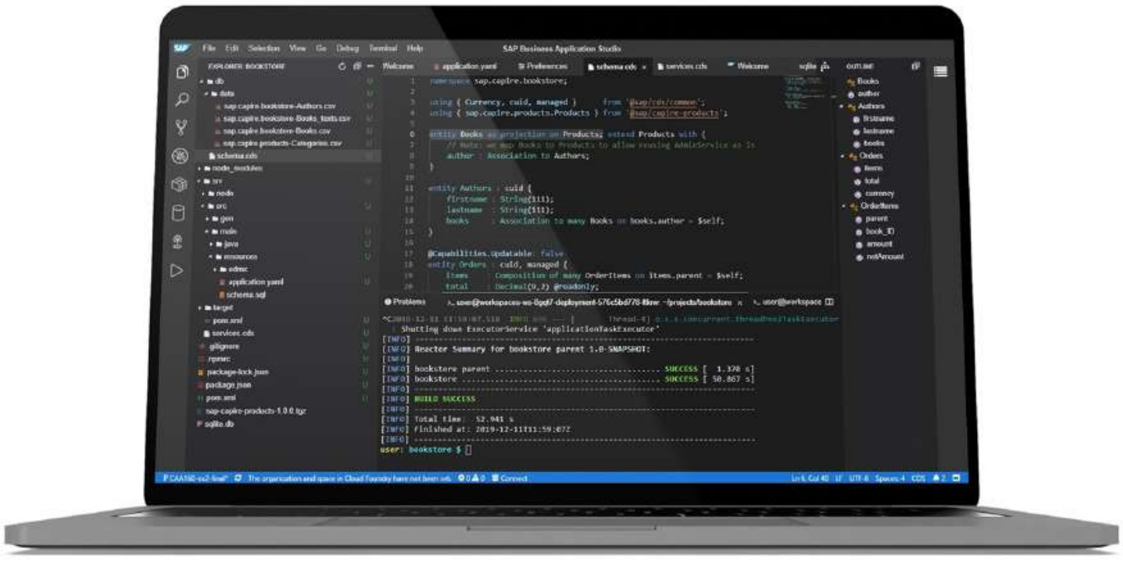
Develop an SAP Fiori App Using SAP BTP, ABAP Environment
Extension Suite - Development Efficiency
Develop a travel booking SAP Fiori application using SAP BTP, ABAP environment on SAP Business Technology Platform
Beginner 1h 10m Open Tutorial

Figure 6: SAP BTP Trial

Dev Space for Business Applications

Refer to the tutorial Create a Dev Space for Business Applications to learn how to create a development space (dev space) for business applications.

SAP BAS is the cloud-based Integrated Development Environment (IDE) for SAP’s multi-cloud environment. It supports you in developing and extending SAP solutions, and with integrating SAP services, technologies, and solutions seamlessly.



```
namespace sap.capire.bookstore;

using { Currency, cuid, managed }      from '@sap/cds/common';
using { sap.capire.products.Products } from '@sap/capire-products';

entity Books as projection on Products; extend Products with {
    // Note: we map Books to Products to allow reusing AdminService as is
    author : Association to Authors;
}

entity Authors : cuid {
    firstname : String(111);
    lastname  : String(111);
    books     : Association to many Books on books.author = $self;
}

@Capabilities.Updatable: false
entity Orders : cuid, managed {
    Items     : Composition of many OrderItems on Items.parent = $self;
    total     : Decimal(9,2) @readonly;
```

```
^C2019-12-11 11:59:07.518  INFO ... --- [       Thread-4] o.s.s.concurrent.ThreadPoolTaskExecutor
  : Shutting down ExecutorService 'applicationTaskExecutor'
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for bookstore parent 1.0-SNAPSHOT:
[INFO]
[INFO] bookstore parent ................................... SUCCESS [  1.370 s]
[INFO] bookstore .......................................... SUCCESS [ 50.867 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  52.941 s
[INFO] Finished at: 2019-12-11T11:59:07Z
[INFO] ------------------------------------------------------------------------
user: bookstore $
```

Figure 7: SAP BAS IDE

SAP BAS provides turn-key solutions based on dev spaces. A *dev space*:https://help.sap.com/products/SAP%20Business%20Application%20Studio/9d1db9835307451daa8c930fbd9ab264/6053df8bca3946f098bc9f89e49d7317.html?locale=en-US[3] "is a development environment with the tools, capabilities, and resources needed for developing your application". SAP BAS offers the following *dev space types:* https://help.sap.com/products/SAP%20Business%20Application%20Studio/9d1db9835307451daa8c930fbd9ab264/4142f786f3d345699c3d5fbebda5ded6.html?locale=en-US[4]

- SAP Fiori
- Full Stack Cloud Application
- SAP HANA Native Application
- SAP Mobile Application
- Basic

Reference links

For your convenience this section contains the external references of this lesson in the following format:

- Reference number
- Section heading
- Context text fragment to identify the location in the section
- Brief description of the linked content
- Link to the content as link and in clear text

If links are used multiple times in a text, only the first location is mentioned in the reference table.

Table 1: Reference Links

Ref#	Section	Context text fragment	Brief description	Link
1	SAP Business Technology Platform Trial	Tutorial: Get a Free Account on SAP BTP Trial	SAP BTP trial account	https://developers.sap.com/tutorials/hcp-create-trial-account.html
2	Dev Space for Business Applications	Tutorial: Create a Dev Space for Business Applications	Create a dev space	https://developers.sap.com/tutorials/appstudio-devspace-create.html
3	Dev Space for Business Applications	A dev space "is a development........	Dev Spaces in SAP Business Application Studio	https://help.sap.com/products/SAP%20Business%20Application%20Studio/9d1db9835307451daa8c930fbd9ab264/6053df8bca3946f098bc9f89e49d7317.html
4	Dev Space for Business Applications	SAP BAS offers the following dev space types:	Dev spaces	https://help.sap.com/products/SAP%20Business%20Application%20Studio/9d1db9835307451daa8c930fbd9ab264/4142f786f3d345699c3d5fbebda5ded6.html

LESSON SUMMARY

You should now be able to:

- Get ready to build your extension project

Unit 2
Lesson 2

SAP BTP Account Model

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Explain the SAP BTP account model

SAP Business Technology Platform Account Model

Usage Scenario

Your company has ordered and received a global account in SAP Business Technology Platform (SAP BTP) to use services from there. To understand the basics, you need to familiarize yourself with the SAP BTP account model.

Learning objectives

- Explain the difference between global accounts and subaccounts.
- Explain the meaning of regions and environments in the SAP BTP.
- Explain the basic platform concepts of SAP BTP.

Overview

SAP BTP is an integrated offering comprising four technology portfolios:

- Database and data management
- Application development and integration
- Analytics
- Intelligent technologies

The platform offers users the ability to turn data into business value, compose end-to-end business processes, and build and extend SAP applications quickly.

The central point of entry to the cloud platform is the SAP BTP cockpit, where you can access your accounts and applications and manage all activities associated with them.

Global Accounts

A global account is the realization of a contract you made with SAP. A global account is used to manage subaccounts, members, entitlements and quotas. You receive entitlements and quotas to use platform resources per global account and then distribute the entitlements and quotas to the subaccount for actual consumption.

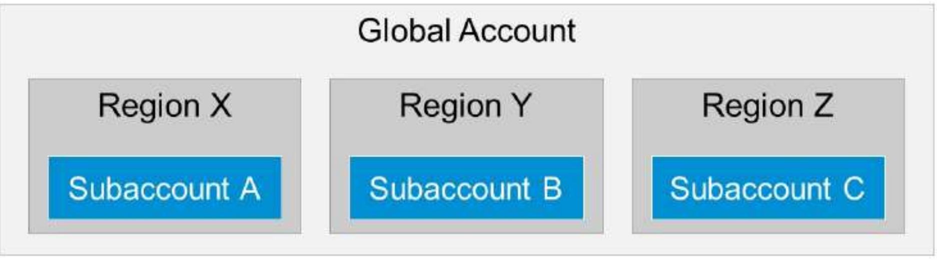


Figure 8: Global Account

Subaccounts

Subaccounts let you structure a global account according to your organization's and project's requirements with regard to members, authorizations, and entitlements. A global account can contain one or more subaccounts in which you deploy applications, use services, and manage your subscriptions. Subaccounts in a global account are independent of each other. This is important to consider with respect to security, member management, data management, data migration, integration, and so on, when you plan your landscape and overall architecture.

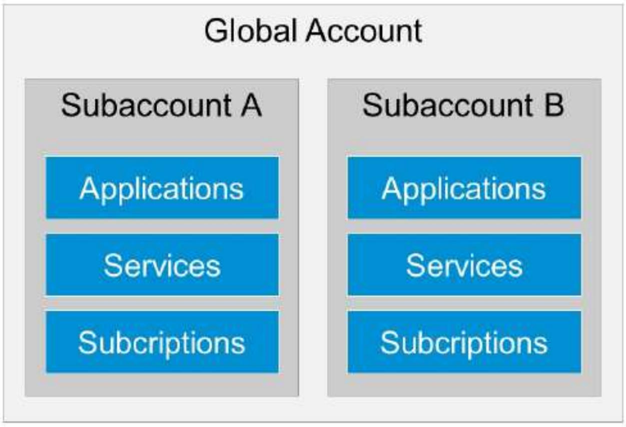


Figure 9: Subaccounts

Regions and Environments

You can deploy applications in different regions. Each region represents a geographical location (for example, Europe, US East) where applications, data, or services are hosted.

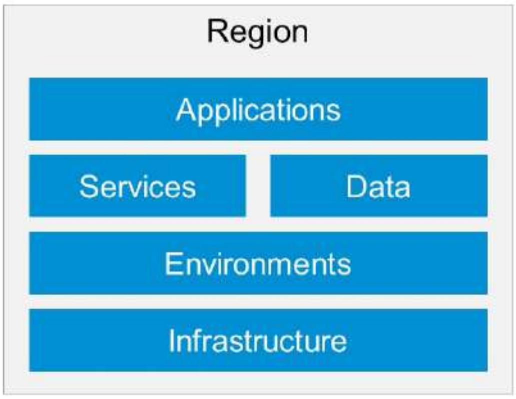


Figure 10: Regions and Environments

Animation:

For more information on , please view the animation in the lesson *SAP BTP Account Model*, in your online course.

Infrastructure: The infrastructure layer of a region is either provided by SAP or by one of SAP's Infrastructure as a Service (IaaS) partners Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), and Alibaba Cloud.

Environments: Environments constitute the actual platform-as-a-service offering of SAP BTP that allows for the development and administration of business applications.

Services: Services enable, facilitate, or accelerate the development of business applications and other platform services on SAP BTP. You find all available services in the SAP Discovery Center.

Data: Your business and application data, managed through services like the SAP HANA Cloud service or the SAP Data Warehouse Cloud service.

Each environment comes equipped with the tools, technologies, and runtimes that you need to build applications. The availability of different environments allows for greater flexibility in your development process.

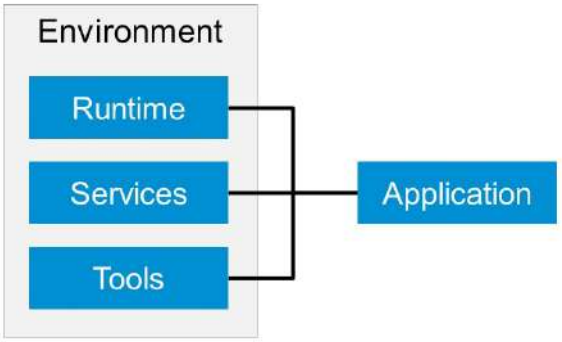


Figure 11: Environment

The following environments are available: Cloud Foundry Environment, ABAP Environment, Neo Environment, and KYMA Environment.

Basic Platform Concepts

If your global account is on cloud management tools, feature set B, the new hierarchical element called directory is introduced, which is essentially a grouping of subaccounts. Furthermore, subaccounts can have multiple environments.

The figure below depicts the relationship between a global account, its directories, subaccounts, environments, regions, entitlements, and quotas for feature set B.

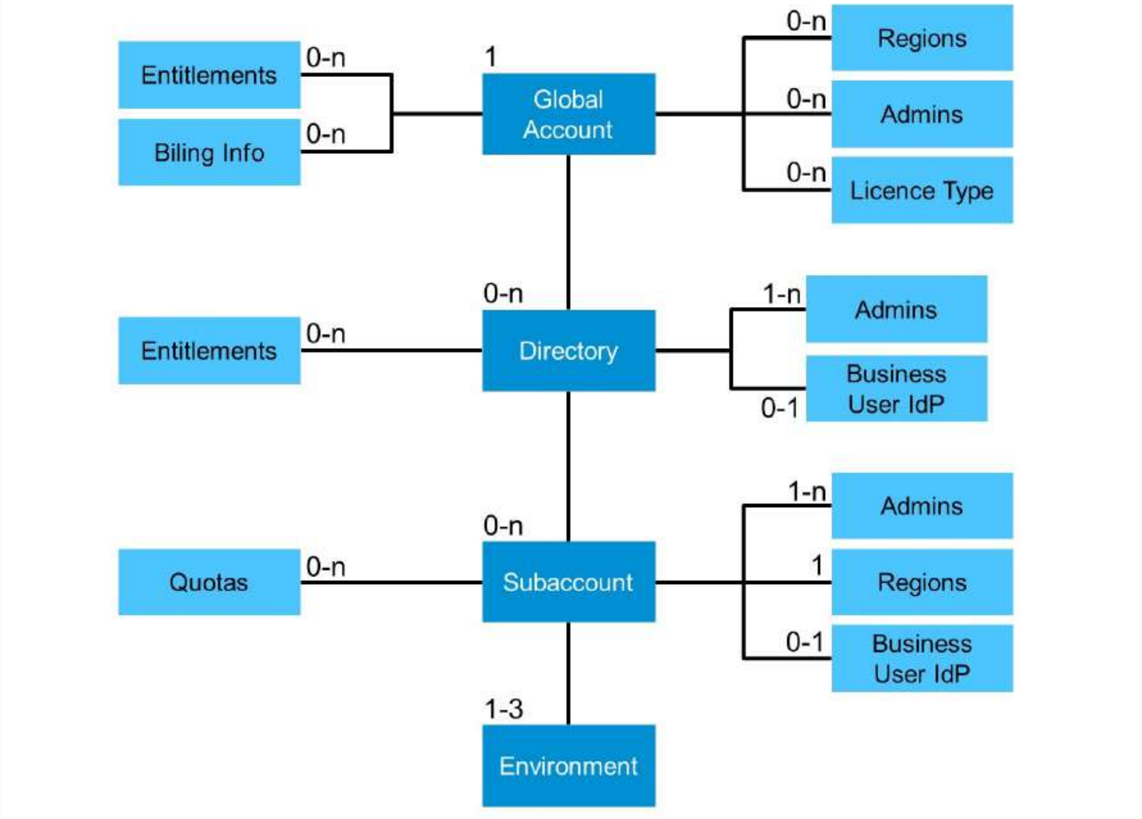


Figure 12: Feature Set B

A subaccount is specified for the environment Neo or for the environment Multi-Environment (KYMA, ABAP and Cloud Foundry environment).

SAP BTP: Different Account Model Examples

To create a staged development environment, your account model can be built on functional areas.

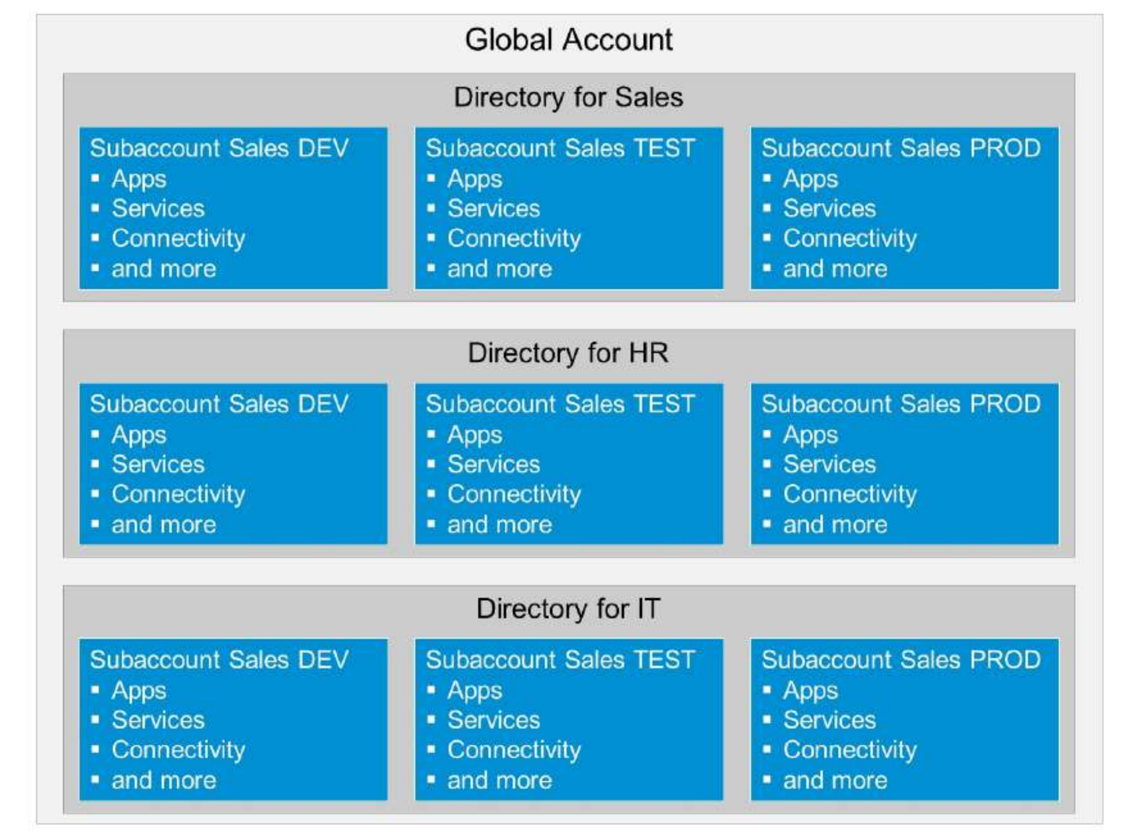


Figure 13: SAP BTP: Different Account Model Examples

Using this account model, you can distribute the subaccount administration to several teams, which allows for easy scaling as the number of cloud projects grows while still having a manageable amount of maintenance and governance efforts. If possible, consider assigning responsible colleagues to each group of three subaccounts, that is, to each account landscape.

Create directories per location or subsidiary.

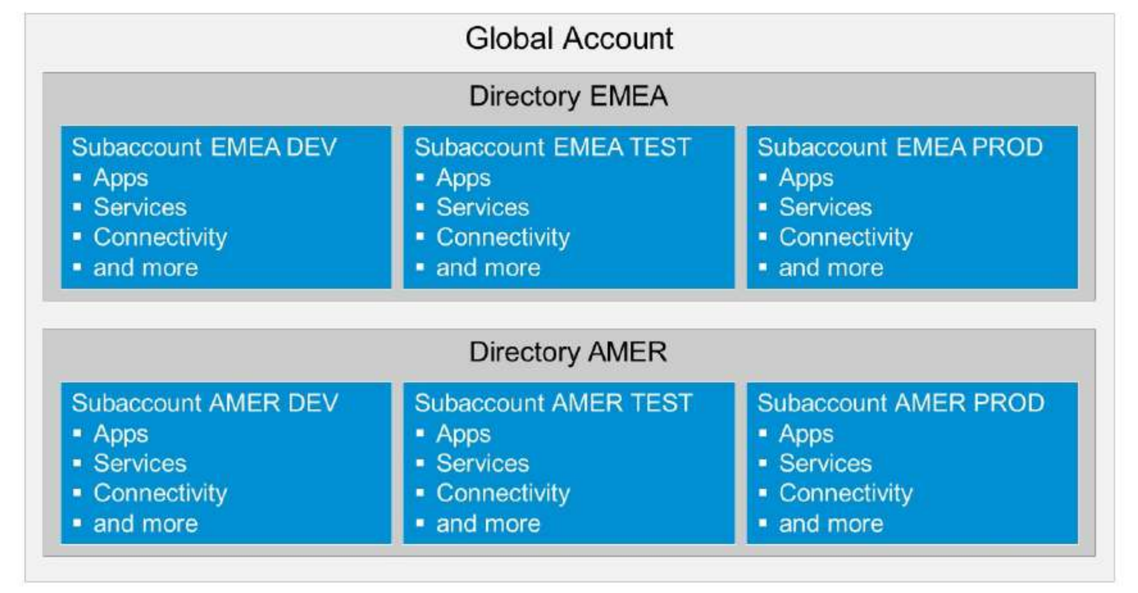


Figure 14: Directories

In this account model, you create different directories for geographical areas. Additionally, for example, you can create custom properties for subaccounts that belong to the same departments in those locations.

Summary

You have now become familiar with the SAP BTP account model and its basic concepts and know how the terms global accounts, subaccounts, regions, and environments are related.

Further Reading

- SAP Business Technology Platform
- SAP BTP Best Practices

LESSON SUMMARY

You should now be able to:

- Explain the SAP BTP account model

Unit 2
Lesson 3

SAP BTP Environments

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Identify the most suitable SAP BTP environment for a new cloud application project

SAP Business Technology Platform Environments

Usage Scenario

Your company is planning to develop a set of applications on the SAP Business Technology Platform (SAP BTP). You have a diverse team of developers with different skill sets. Some team members have worked in the SAP ecosystem for many years and are especially proficient in ABAP development and the SAP Application Server ABAP. Other team members joined your company recently and have a skill set around web development. They are used to programming languages like Java and JavaScript and some even have experience in cloud application platforms like Cloud Foundry or Kubernetes.

The applications that you are going to develop must meet a wide variety of requirements. In some cases, you just want to build a custom UI for some data object, that is already provided by your backend SAP S/4HANA system. In other cases, you want to extend the existing data model from the backend system with custom entities and data, that should be managed independently of your core system. And then you also want to develop more complex and highly scalable applications, that follow a microservice architecture pattern and integrate with a multitude of other services and applications.

The available environments on the SAP BTP meet these complex requirements.

Animation: Usage Scenario

For more information on *Usage Scenario*, please view the animation in the lesson *SAP BTP Environments*, in your online course.

Learning objectives

- Describe the runtime environments available in the SAP BTP
- Describe key characteristics of each environment

Key Characteristics of SAP BTP Environments

Environments constitute the actual platform-as-a-service offering of SAP BTP that enables the development and administration of business applications. Each environment is equipped with the tools, technologies, and runtimes that you need to build applications.

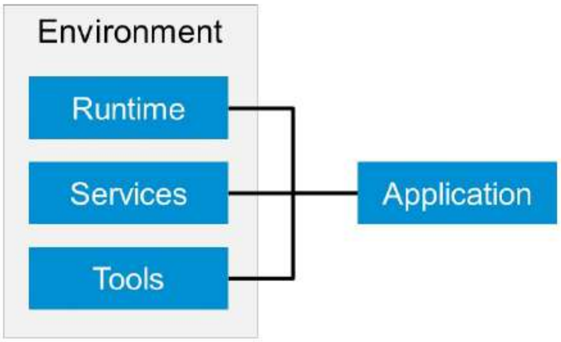


Figure 16: Environment

- A runtime provides your applications with the environment and infrastructure to be executed.
- A tool is something that lets you interact with the runtime environment, for example a command line interface or a software development kit (SDK).
- A service is something that your application depends on, like a SAP HANA Cloud database.

The availability of different environments allows for greater flexibility in your development process.

The following environments are available: Cloud Foundry Environment, ABAP Environment, Kyma Environment, and Neo Environment.

Cloud Foundry Environment

The Cloud Foundry environment allows you to create polyglot cloud applications. Cloud Foundry (CF) is an open-source platform as a service (PaaS), which comprises a set of specifications and tools around the whole application development lifecycle. This includes application deployment, application lifecycle management, service dependency management, security, and more. For interaction, Cloud Foundry offers a command line interface (CF CLI), which allows for high automation through scripting. You can build your application on open standards with Java, Node.js, and Python build packs provided by SAP. Alternatively, bring other languages with community build packs for PHP, Ruby, or Go.

ABAP Environment

The ABAP environment is mainly targeted at creating extension applications for ABAP-based products, such as SAP S/4HANA Cloud, but also lets you develop independent cloud applications. Technically, the ABAP environment lives within the Cloud Foundry environment. The ABAP environment is based on the latest ABAP platform cloud release that is also used for SAP S/4HANA Cloud. The software stack contains standard technology components that are familiar from the standalone Application Server ABAP. The ABAP environment supports the ABAP RESTful Application Programming Model (RAP) including SAP Fiori and Core Data Services (CDS).

Kyma Environment

The Kyma environment provides a fully managed Kubernetes runtime based on the open-source project "Kyma". Kubernetes, similar to Cloud Foundry, is an open-source platform for managing the application lifecycle. In contrast to CF it has a larger focus on containers, container orchestration, and scalability.

Kyma allows developers to extend SAP solutions with serverless Functions and combine them with containerized microservices.

SAP systems connected to a Kyma environment expose APIs and events collected under the Service Catalog. To extend the existing logic of these SAP services, you can build serverless applications called Functions, and trigger them to react to particular events or calls to your application's API. You can also use the Kyma environment to deploy microservices or even build full-stack applications.

Animation:

For more information on , please view the animation in the lesson *SAP BTP Environments*, in your online course.

Neo Environment

The Neo environment, historically, was the first environment available on the SAP BTP - though at the time of its emergence the platform was not called SAP BTP yet. Also, while the other environments are supposed to run on Infrastructure as a Service (IaaS) providers like Amazon Web Services, Google Cloud Platform, Alibaba Cloud, and Microsoft Azure (also known as the multi-cloud foundation), the Neo environment runs in SAP data centers only.

The Neo environment lets you develop HTML5, Java, and SAP HANA extended application services (SAP HANA XS) applications. You can also use the UI Development Toolkit for HTML5 (SAPUI5) to develop rich user interfaces for modern web-based business applications.

SAP recommends to migrate your existing scenarios from the Neo environment to the multi-cloud foundation.

Which Environment to choose?

If you need further information on the benefits of the different environments and which environment to choose, have a look at this overview of runtimes, environments and programming models.

Summary

You now know about the available environments on the SAP BTP and can describe their key characteristics. When starting a new cloud application project, you can give an initial assessment on which environment to choose and you know where to find further information.

Further Reading

- Runtimes from SAP
- Official Cloud Foundry Documentation
- Official Kyma Documentation

LESSON SUMMARY

You should now be able to:

- Identify the most suitable SAP BTP environment for a new cloud application project

Unit 2
Lesson 4

SAP Business Application Studio (BAS)

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Describe key capabilities of SAP Business Application Studio (BAS)

SAP Business Application Studio

Usage Scenario

Your company is planning to develop extension applications on the SAP Business Technology Platform (BTP). Your software developers have different devices and operating systems, but they need a common development environment for their work, that runs independently of their technical system setup. The development environment should provide all the tools for modern cloud development, especially specific tools required for business application development in the SAP ecosystem.

Animation: Usage Scenario

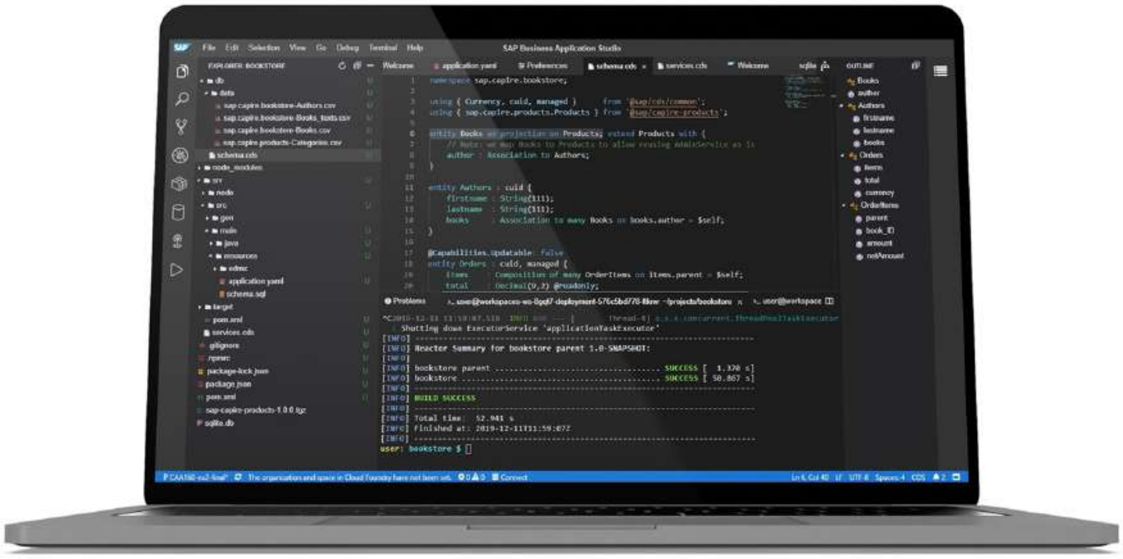
For more information on *Usage Scenario*, please view the animation in the lesson *SAP Business Application Studio (BAS)*, in your online course.

Learning objectives

- Explain the advantages of using SAP Business Application Studio.
- Explain the concept of dev spaces.

Key Capabilities of SAP Business Application Studio

SAP Business Application Studio, see: https://help.sap.com/products/SAP%20Business%20Application%20Studio/9d1db9835307451daa8c930fbd9ab264/8f46c6e6f86641cc900871c903761fd4.html?locale=en-US is an SAP BTP service that offers a modern development environment tailored for business application development. You access the development environment using your internet browser, nevertheless, it provides a desktop-like development experience.



SAP Business Application Studio
File Edit Selection View Go Debug Terminal Help
EXPLORER: BOOKSTORE
Welcome
application.yaml
Preferences
schema.cds
services.cds
OUTLINE
Problems
user@workspaces-ws-8gqt7-deployment-576c5bd778-8kw ~/projects/bookstore

```
namespace sap.capire.bookstore;

using { Currency, cuid, managed }      from '@sap/cds/common';
using { sap.capire.products.Products } from '@sap/capire-products';

entity Books as projection on Products; extend Products with {
    // Note: we map Books to Products to allow reusing AdminService as is
    author : Association to Authors;
}

entity Authors : cuid {
    firstname : String(111);
    lastname  : String(111);
    books     : Association to many Books on books.author = $self;
}

@Capabilities.Updatable: false
entity Orders : cuid, managed {
    Items    : Composition of many OrderItems on Items.parent = $self;
    total    : Decimal(9,2) @readonly;
```

```
^C2019-12-11 11:59:07.518  INFO 888 --- [       Thread-4] o.s.s.concurrent.ThreadPoolTaskExecutor
  : Shutting down ExecutorService 'applicationTaskExecutor'
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for bookstore parent 1.0-SNAPSHOT:
[INFO]
[INFO] bookstore parent ................................... SUCCESS [  1.378 s]
[INFO] bookstore .......................................... SUCCESS [ 50.867 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  52.941 s
[INFO] Finished at: 2019-12-11T11:59:07Z
[INFO] ------------------------------------------------------------------------
user: bookstore $ 
```

Figure 18: SAP BAS IDE

If you already know Microsoft Visual Studio Code and you access SAP Business Application Studio for the first time, you might notice many similarities. This is because SAP Business Application Studio is based on an open source IDE, Eclipse Theia, which embraces the Microsoft Visual Studio Code experience.

The development environment offers you all the tools required for your business application development, including a command line, and editors optimized for different use cases.

Dev Spaces

At the heart of SAP Business Application Studio are the dev spaces. The dev spaces are isolated environments in the cloud containing tailored tools and preinstalled runtimes per business scenario, such as SAP Fiori, SAP S/4HANA extensions, Workflow, Mobile, and more. Technically, a dev space is implemented as a Kubernetes Pod with multiple containers providing the tools and extensions. Check out this blog post for more technical information about dev spaces.

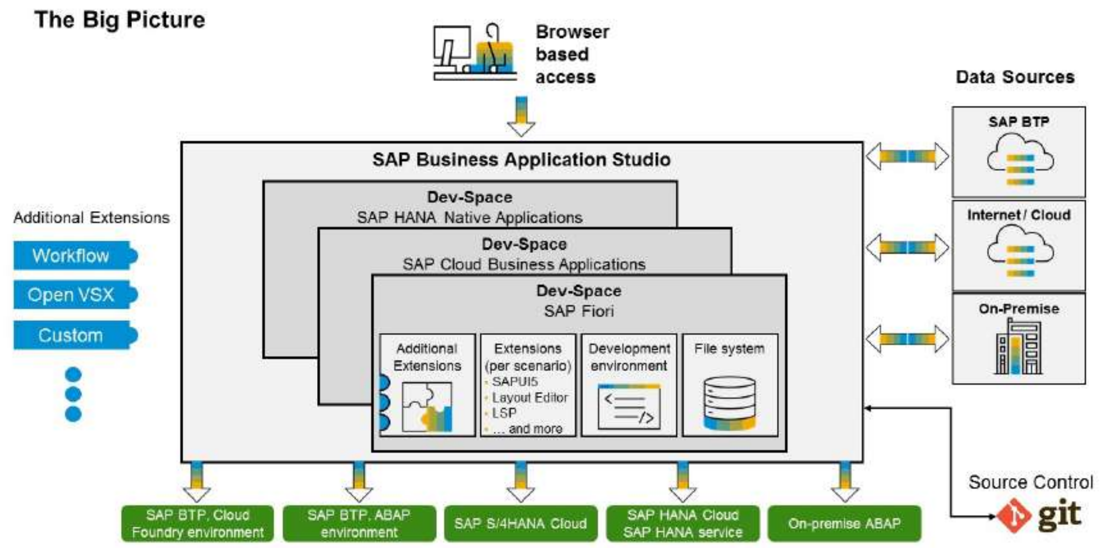
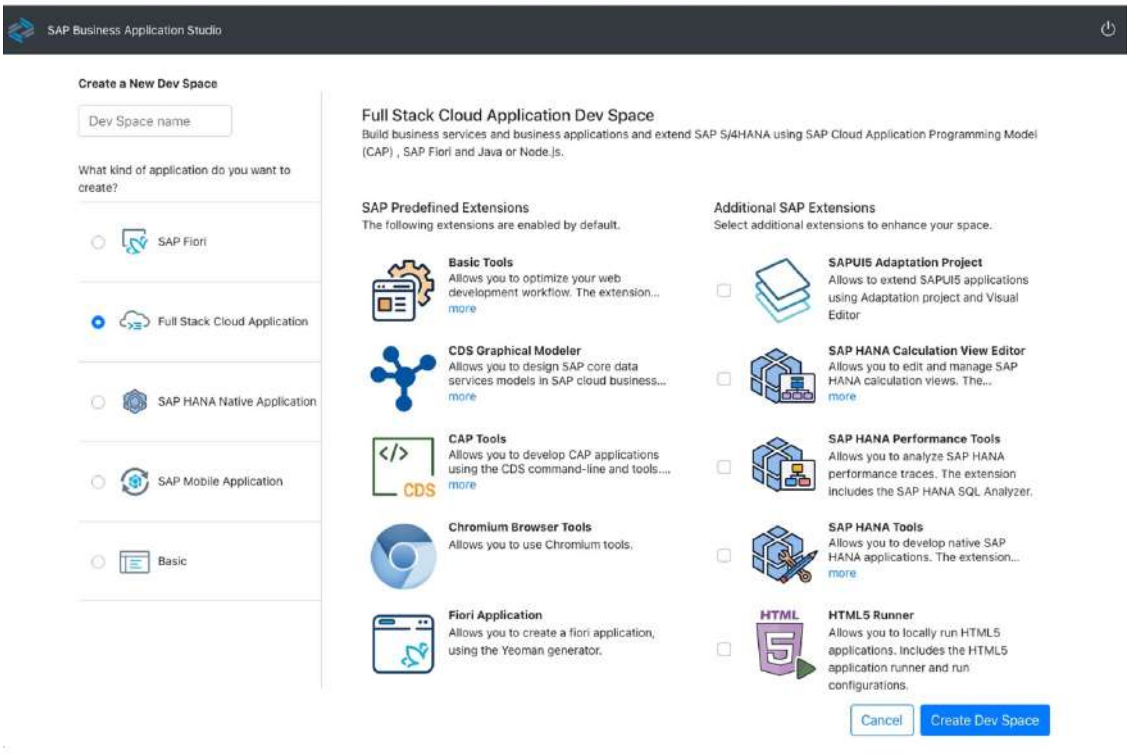


Figure 19: The Big Picture

When creating a new dev space, you choose a pre-configured development environment, based on the type of application that you want to build.

The following **dev space types**, see: https://help.sap.com/products/SAP%20Business%20Application%20Studio/9d1db9835307451daa8c930fbd9ab264/4142f786f3d345699c3d5fbebda5ded6.html?locale=en-US are available:

- SAP Fiori
- Full Stack Cloud Application
- SAP HANA Native Application
- SAP Mobile Application
- Basic



SAP Business Application Studio

Create a New Dev Space

Dev Space name

What kind of application do you want to create?

○ SAP Fiori

● Full Stack Cloud Application

○ SAP HANA Native Application

○ SAP Mobile Application

○ Basic

Full Stack Cloud Application Dev Space

Build business services and business applications and extend SAP S/4HANA using SAP Cloud Application Programming Model (CAP) , SAP Fiori and Java or Node.js.

SAP Predefined Extensions

The following extensions are enabled by default.

Basic Tools
Allows you to optimize your web development workflow. The extension... more

CDS Graphical Modeler
Allows you to design SAP core data services models in SAP cloud business... more

CAP Tools
Allows you to develop CAP applications using the CDS command-line and tools.... more

Chromium Browser Tools
Allows you to use Chromium tools.

Fiori Application
Allows you to create a fiori application, using the Yeoman generator.

Additional SAP Extensions

Select additional extensions to enhance your space.

□ **SAPUI5 Adaptation Project**
Allows to extend SAPUI5 applications using Adaptation project and Visual Editor

□ **SAP HANA Calculation View Editor**
Allows you to edit and manage SAP HANA calculation views. The... more

□ **SAP HANA Performance Tools**
Allows you to analyze SAP HANA performance traces. The extension includes the SAP HANA SQL Analyzer.

□ **SAP HANA Tools**
Allows you to develop native SAP HANA applications. The extension... more

□ **HTML5 Runner**
Allows you to locally run HTML5 applications. Includes the HTML5 application runner and run configurations.

Figure 20: SAP Business Application Studio

Each dev space type comes with a set of SAP predefined extensions, that are required for the respective application scenario. You can select additional extensions to enhance your space. Due to its open source nature and the close resemblance to Microsoft Visual Studio Code, SAP Business Application Studio also lets you integrate Visual Studio Code-compatible extensions from the Open VSX Registry. It is possible to create multiple dev spaces per user, depending on the chosen *application plan*, see https://help.sap.com/products/SAP%20Business%20Application%20Studio/9d1db9835307451daa8c930fbd9ab264/2c72917df87e47c290e061a556d92398.html?locale=en-US.

Benefits of SAP Business Application Studio at a glance:

- Provides a managed, pre-configured, hosted environment, optimized for SAP application development.
- Can be centrally administered with tools repositories, systems access, and company policies.
- Integrates with existing SAP solutions, systems, and services.
- Provides easy access to Visual Studio Code-compatible extensions from open source Open VSX Registry.

Summary

You can now describe the key capabilities of SAP Business Application Studio and basic advantages over similar offline development environments like Microsoft Visual Studio Code. Also, you know what a dev space is and the benefits it brings in regard to application development in the SAP ecosystem.

Further Reading

- SAP Business Application Studio Info Blog
- SAP Business Application Studio Environment Overview YouTube Video
- SAP Business Application Studio Community Page

LESSON SUMMARY

You should now be able to:

- Describe key capabilities of SAP Business Application Studio (BAS)

Create a CAP-Based Service

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Create a CAP-based service

Create a CAP-Based Service: Exercise Overview

Scenario

You will create a new SAP Cloud Application Programming Model (CAP) project. This project provides the basis for your risk-management extension that comprises:

- A database that stores application data.
- A service that exposes the application data from the database via application programming interfaces (APIs).
- A user interface that consumes the application data from the service and presents it to the end users.

In this part of the course, you create:

- A data model
- A service definition

The **data model** defines what kind of data your application contains, how it is structured, and how it will be stored in the database.

The **service definition** defines which part of your data should be exposed as a service via APIs.

Task Flow

In this exercise, you will perform the following tasks:

- Create an SAP CAP project from a template.
- Add a data model to a project.
- Create a service definition.
- Add an OData service definition to a project.
- Add data from a CSV file to the project.

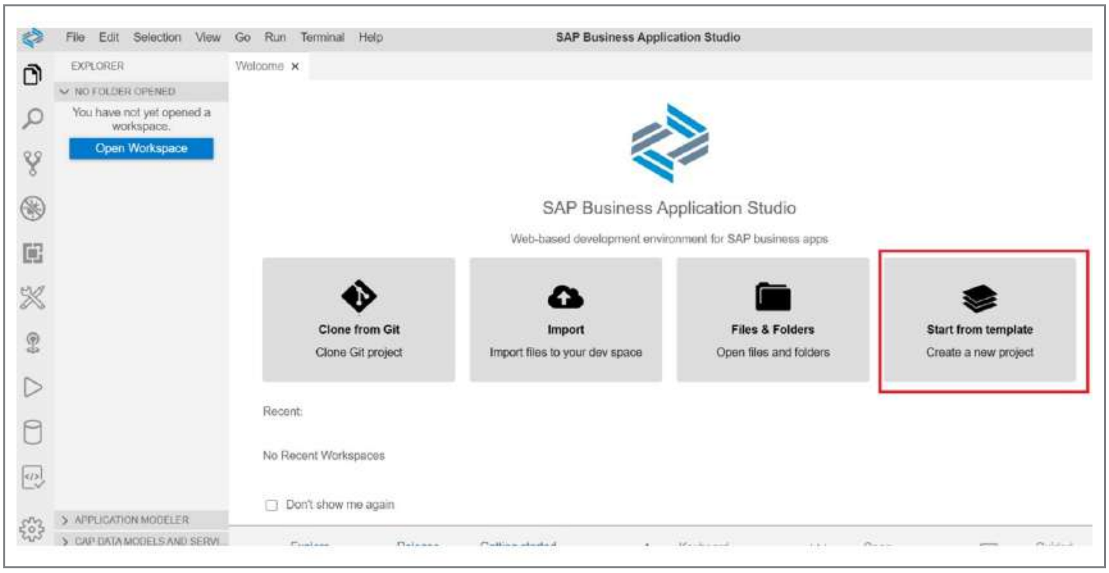
Prerequisites

You have successfully created an SAP BTP Trial account and a dev space.

You have added your Dev Space for Business Applications in SAP Business Application Studio.

Create and Initialize the Project

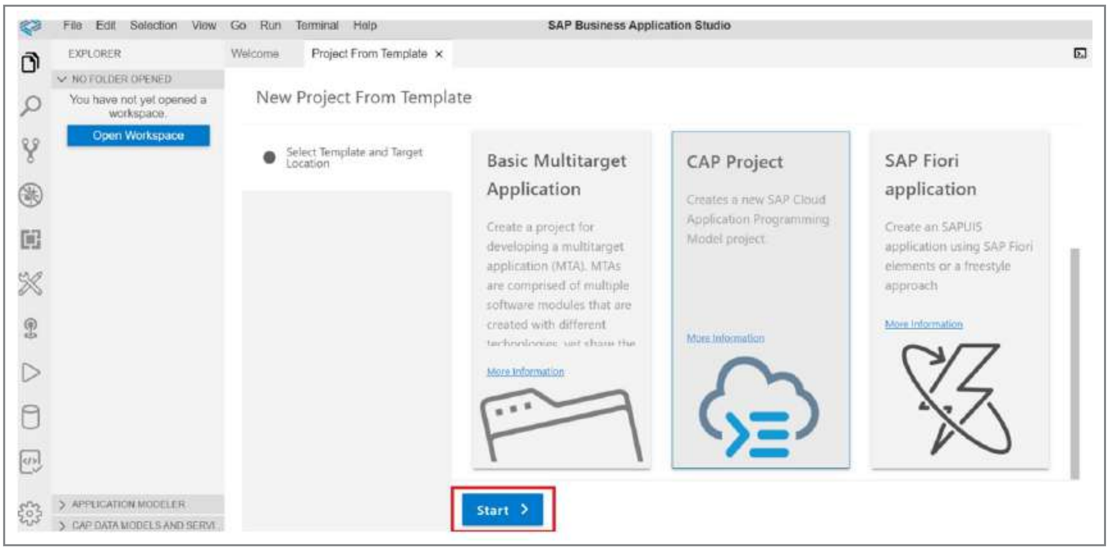
1. In the SAP Business Application Studio, choose *Start from template*.



You have not yet opened a workspace.

Web-based development environment for SAP business apps

2. Select **`CAP Project`** and select *Start*.


File Edit Selection View Go Run Terminal Help
SAP Business Application Studio
EXPLORER
NO FOLDER OPENED
Open Workspace
Welcome
Project From Template ×
New Project From Template
● Select Template and Target Location
Basic Multitarget Application
More Information
CAP Project
More Information
SAP Fiori application
More Information
Start >
APPLICATION MODELER
CAP DATA MODELS AND SERVI...

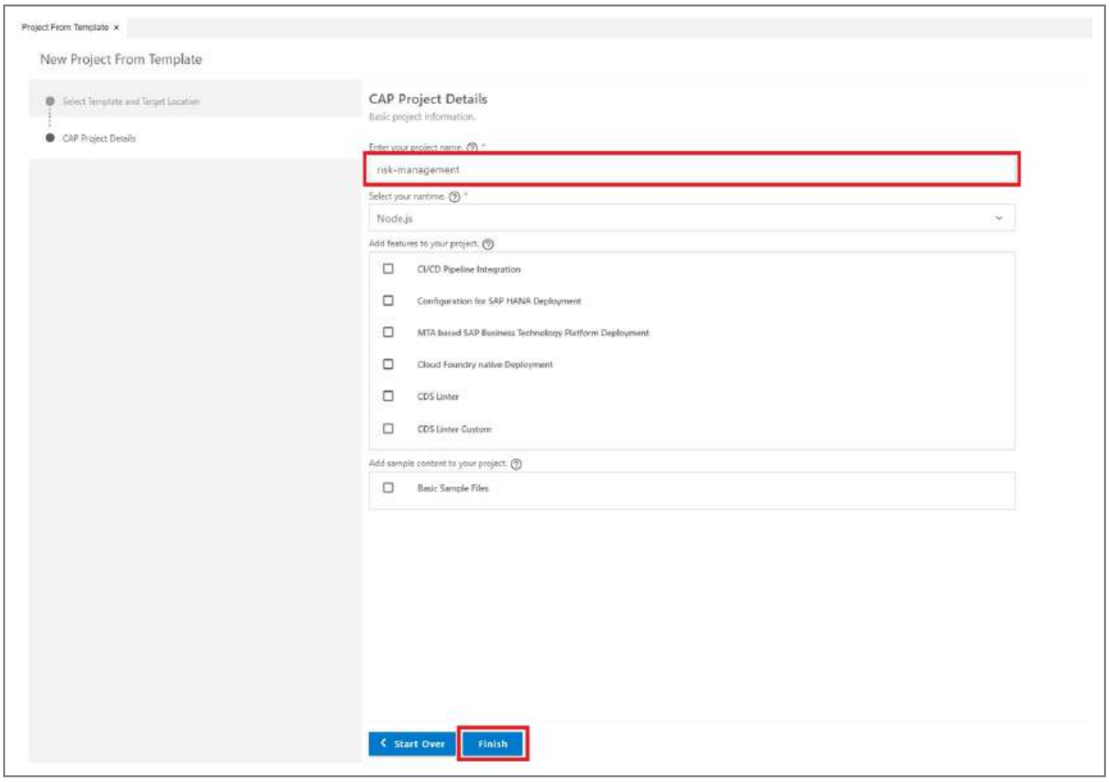
You have not yet opened a workspace.

Create a project for developing a multitarget application (MTA). MTAs are comprised of multiple software modules that are created with different technologies, yet share the

Creates a new SAP Cloud Application Programming Model project.

Create an SAPUI5 application using SAP Fiori elements or a freestyle approach

3. Enter **`risk-management`** as a *project name*. Don't tick any of the checkboxes below. Make sure, that *Node.js* is the selected runtime. Select *Finish*.



Project From Template ×

New Project From Template

● Select Template and Target Location

● CAP Project Details

CAP Project Details

Basic project information.

Enter your project name. ⓘ *

risk-management

Select your runtime. ⓘ *

Node.js

Add features to your project. ⓘ

□ CI/CD Pipeline Integration

□ Configuration for SAP HANA Deployment

□ MTA based SAP Business Technology Platform Deployment

□ Cloud Foundry native Deployment

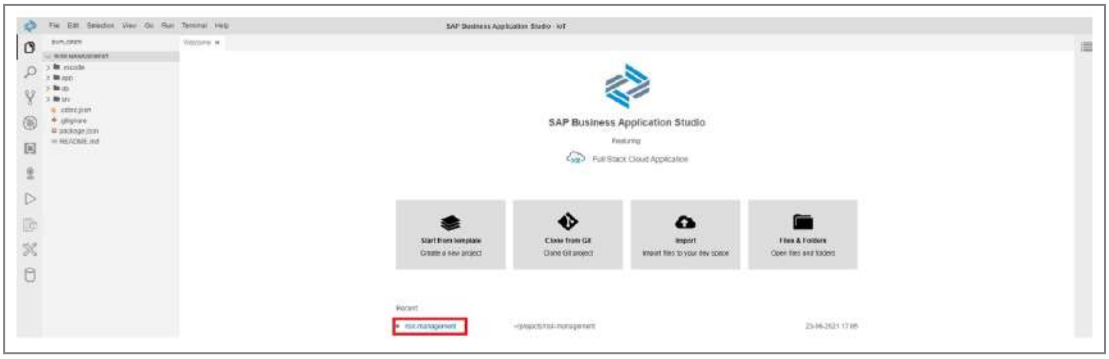
□ CDS Linter

□ CDS Linter Custom

Add sample content to your project. ⓘ

□ Basic Sample Files

4. After the project has been generated, the new workspace will open and display the generated *Risk-Management* project like this:



5. In Business Application Studio choose *Terminal* → *New Terminal* from its menu.

   A new terminal opens in the lower right part of the *SAP Business Application Studio* screen.

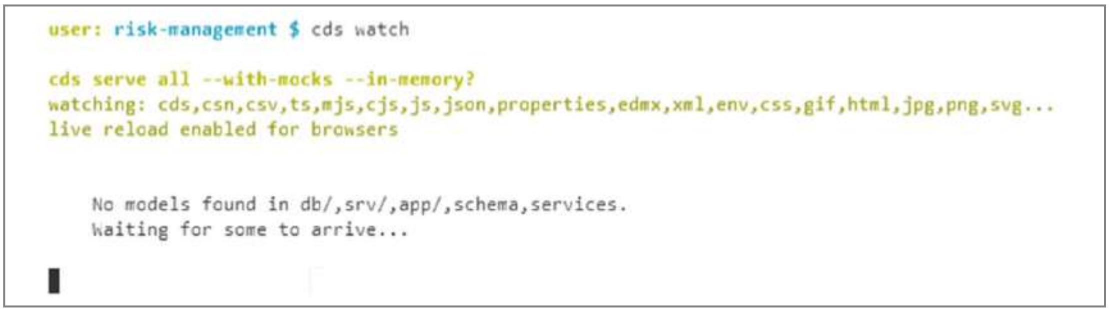
6. In the terminal, run command `cds watch`.

   This might take a while. It installs all the dependencies of your project into a new folder `node_modules` within the `root` of your project. If you encounter messages about `Deprecated` dependencies ignore them for now, as long as there are no errors in the log.

7. In the terminal, start a CAP server by typing: `cds watch`.

   The CAP server serves all the CAP sources from your project. It also “watches” all the files in your projects and conveniently restarts the server whenever you save a file. Changes you've made will immediately be served without you having to do anything.

   The terminal now looks like this:



```
user: risk-management $ cds watch

cds serve all --with-mocks --in-memory?
watching: cds,csn,csv,ts,mjs,cjs,js,json,properties,edmx,xml,env,css,gif,html,jpg,png,svg...
live reload enabled for browsers


    No models found in db/,srv/,app/,schema,services.
    Waiting for some to arrive...

█
```

`cds watch` tells you that there’s no model yet that it can serve. You add one in the following steps.

In general, you can keep `cds watch` running in a terminal for the whole exercise. There is no need to restart it or try to add a second instance of it (in fact, if you do this, you get an error, described here). In the follow-up chapters, you will open a new browser tab to see the results of `cds watch`. You can just keep this open and refresh it each time there is a change. `cds watch` notices any file change and makes sure you get the new results after a refresh in the browser.

Add a Data Model to the Project

Now, you create a data model using the Core Data Services (CDS) format from CAP.

1. In the project, go to folder `db`, representing the data model on the database. In the context menu, select *New File*.

2. Enter **`schema.cds`** as a *name*.

3. Select the new file in the explorer, an editor opens.

4. Enter the following lines into the editor

```
namespace riskmanagement;
 using { managed } from '@sap/cds/common';

 entity Risks : managed {
    key ID : UUID @(Core.Computed : true);
    title : String(100);
    owner : String;
    prio : String(5);
    descr : String;
    miti : Association to Mitigations;
    impact : Integer;
    //bp : Association to BusinessPartners;
    // You will need this definition in a later step
    criticality : Integer;
 }

 entity Mitigations : managed {
    key ID : UUID @(Core.Computed : true);
    descr : String;
    owner : String;
    timeline : String;
    risks : Association to many Risks on risks.miti = $self;
 }
```

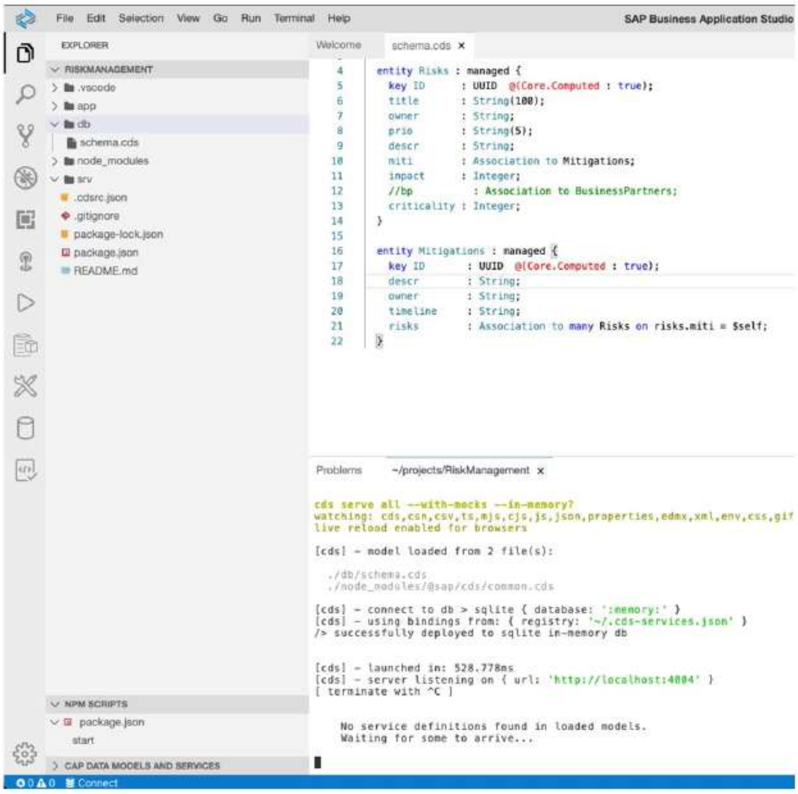
5. Save the file.

   You are using the namespace [1] `riskmanagement`. A namespace is optional. It is helpful to use short unique names without bloating the code with fully qualified names. Essentially, namespaces are just prefixes that are applied to all relevant names in a file automatically.

The code creates 2 entities in the namespace `riskmanagement`, `Risks`, and `Mitigations`. Each of them has a key called `ID` and several other properties. A `Risk` has a mitigation and therefore, the property `miti` has an association to exactly one `Mitigation`. A `Mitigation` in turn can be used for many `Risks`, so it has a “to many” association. The key is automatically filled by CAP, which is exposed to the user of the service with the annotation `@(Core.Computed : true)`.

At this point, you can ignore the commented property `bp` (as well as the other commented lines further down in the file and in subsequent files and chapters). You will use the commented lines at a later stage of the project.

The screen now looks like this:



```
entity Risks : managed {
  key ID      : UUID  @(Core.Computed : true);
  title       : String(100);
  owner       : String;
  prio        : String(5);
  descr       : String;
  miti        : Association to Mitigations;
  impact      : Integer;
  //bp          : Association to BusinessPartners;
  criticality : Integer;
}

entity Mitigations : managed {
  key ID       : UUID  @(Core.Computed : true);
  descr        : String;
  owner        : String;
  timeline     : String;
  risks        : Association to many Risks on risks.miti = $self;
}
```

```
cds serve all --with-mocks --in-memory?
watching: cds,csn,csv,ts,mjs,cjs,js,json,properties,edmx,xml,env,css,gif
live reload enabled for browsers

[cds] - model loaded from 2 file(s):

  ./db/schema.cds
  ./node_modules/@sap/cds/common.cds

[cds] - connect to db > sqlite { database: ':memory:' }
[cds] - using bindings from: { registry: '~/.cds-services.json' }
/> successfully deployed to sqlite in-memory db


[cds] - launched in: 528.778ms
[cds] - server listening on { url: 'http://localhost:4004' }
[ terminate with ^C ]


    No service definitions found in loaded models.
    Waiting for some to arrive...
```

Notice how `cds watch` reacted to dropping the file. It now tells you that it has a model but there are no service definitions yet, as a result, it still can’t serve anything. So, you will add a service definition.

Add a Service to the Project

In this part you will create a new service with two entities. Both are projections of the data models that we created in the chapter before. A service provides an interface to the application. It enables other applications and users outside of the application to interact with it. In this case, the service let's us interact with the `Risks` and `Mitigations` entities, the data within the application.

1. In the project, go to folder `srv/`, representing the service. In the context menu, select *New File*.

2. Enter **`risk-service.cds`** as a *name*.

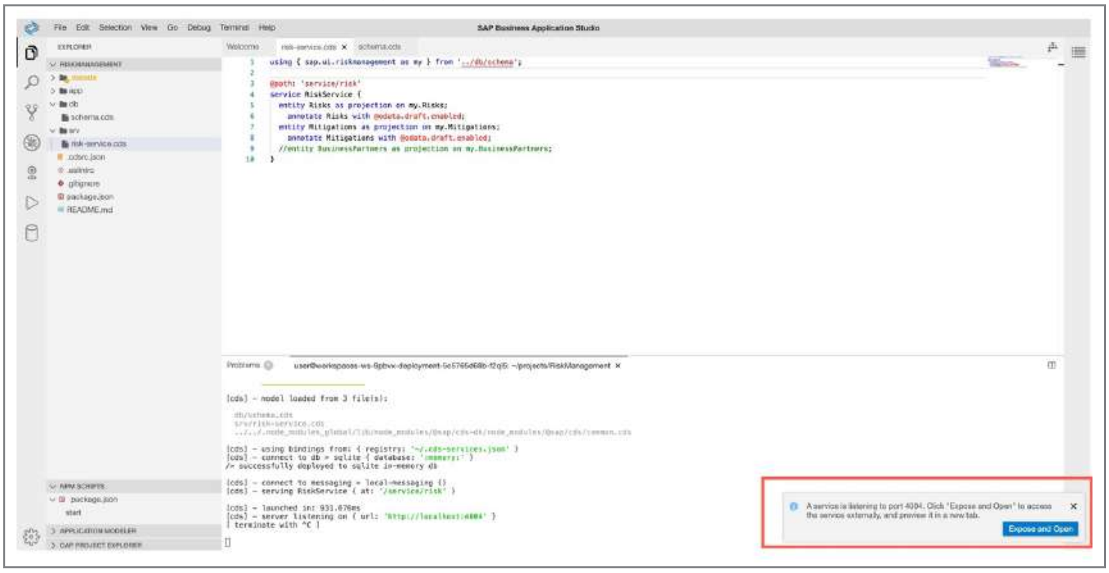
3. Select the new file in the explorer, an editor opens.

4. Enter the following lines into the editor

```
using { riskmanagement as rm } from '../db/schema';

 @path: 'service/risk'
 service RiskService {
 entity Risks as projection on rm.Risks;
    annotate Risks with @odata.draft.enabled;
 entity Mitigations as projection on rm.Mitigations;
    annotate Mitigations with @odata.draft.enabled;
 //@readonly entity BusinessPartners as projection on
rm.BusinessPartners;
 }
```

This creates a new service `RiskService` in the namespace `riskmanagement`. The annotation `@path: 'service/risk'` tells the CAP framework to expose the `RiskService` at URL path `service/risk`. The service exposes two entities (again, just ignore the commented part for the business partner), `Risks` and `Mitigations`, which are both just exposing the entities of the database schema you've created in the step before. The `@odata.draft.enabled` annotation enables edit sessions with draft states [2] stored on the server for the UI, that will be added in a later step.

If you again look at the terminal, you see that `cds watch` has noticed the new file and now tells us that it serves something:



```
using { sap.ui.riskmanagement as my } from '../db/schema';

@path: 'service/risk'
service RiskService {
  entity Risks as projection on my.Risks;
    annotate Risks with @odata.draft.enabled;
  entity Mitigations as projection on my.Mitigations;
    annotate Mitigations with @odata.draft.enabled;
  //entity BusinessPartners as projection on my.BusinessPartners;
}
```

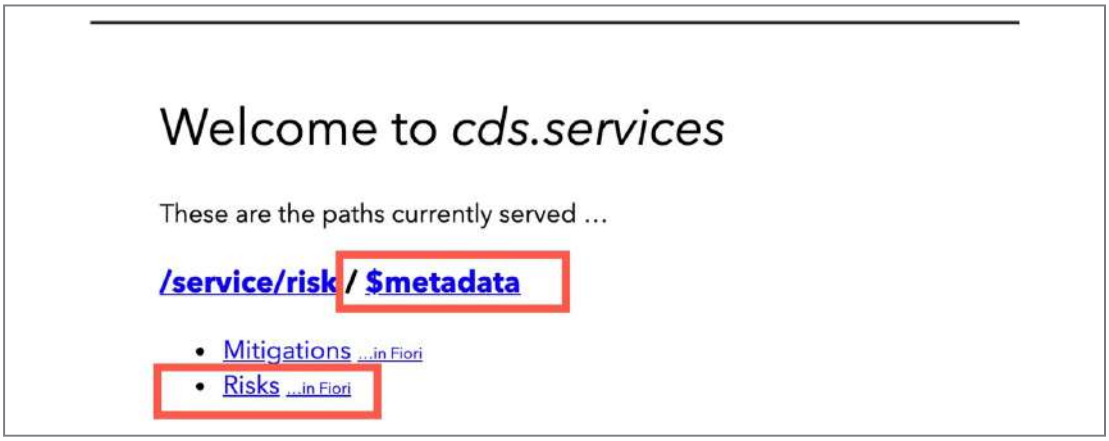
```
[cds] - model loaded from 3 file(s):

  db/schema.cds
  srv/risk-service.cds
  ../../.node_modules_global/lib/node_modules/@sap/cds-dk/node_modules/@sap/cds/common.cds

[cds] - using bindings from: { registry: '~/.cds-services.json' }
[cds] - connect to db > sqlite { database: ':memory:' }
/> successfully deployed to sqlite in-memory db

[cds] - connect to messaging > local-messaging {}
[cds] - serving RiskService { at: '/service/risk' }

[cds] - launched in: 931.076ms
[cds] - server listening on { url: 'http://localhost:4004' }
[ terminate with ^C ]
```

A service is listening to port 4004. Click "Expose and Open" to access the service externally, and preview it in a new tab.

5. Select the *Expose and Open* button.

6. If you are asked to enter a name - just press return.

   You now see this screen:



Welcome to *cds.services*

These are the paths currently served ...

/service/risk / $metadata

- Mitigations ...in Fiori
- Risks ...in Fiori

7. Select the *$metadata* link.

   The service already exposes a full-blown OData metadata document.

8. Go back and choose the *Risks* link.

   This exposes the data for the `Risks` entity. As there is no data yet, you only see this:

```
{
    @odata.context: "$metadata#Risks",
    value: [ ]
 }
```

   Don't close the window, you will need it again.

Add Data for the Service

In this part, we add data to the service locally. The data is stored in a local database called SQLite that CAP invokes behind the scenes. CAP makes it easy to add test data to a service. All it needs is a Comma Separated Values (CSV) file that contains the entities' elements as column headers. Note that the separation of values in this case has to be done using semicolons (`;`) by convention.

1. In the project, go to folder `db/`. In the context menu, select *New Folder*.
2. Enter **`data`** as a name.
3. On the `data/` folder, in the context menu, select *New File*.
4. Enter **`riskmanagement-Risks.csv`** as a *name*.
5. Choose the new file in the explorer, an editor opens.
6. Enter the following lines into the editor

```
ID;createdAt;createdBy;title;owner;prio;descr;miti_id;impact;

20466922-7d57-4e76-b14c-e53fd97dcb11;2019-10-24;SYSTEM;CFR non-
compliance;Fred Fish;3;Recent restructuring might violate CFR code
71;20466921-7d57-4e76-b14c-e53fd97dcb11;10000;
20466922-7d57-4e76-b14c-e53fd97dcb12;2019-10-24;SYSTEM;SLA violation
with possible termination cause;George Gung;2;Repeated SAL violation
on service delivery for two successive quarters;20466921-7d57-4e76-
b14c-e53fd97dcb12;90000;
20466922-7d57-4e76-b14c-e53fd97dcb13;2019-10-24;SYSTEM;Shipment
violating export control;Herbert Hunter;1;Violation of export and
trade control with unauthorized downloads;20466921-7d57-4e76-b14c-
e53fd97dcb13;200000;
```

ID	createdAt	createdBy	title	owner	prio	descr	miti_id	impact
20466922-7d57-4e76-b14c-e53fd97dcb11	2019-10-24	SYSTEM	CFR non-compliance	Fred Fish	3	Recent restructuring might violate CFR code 71	20466921-7d57-4e76-b14c-e53fd97dcb11	10000

7. Save the file.
8. On the `data/` folder, in the context menu, select *New File*.
9. Enter **`riskmanagement-Mitigations.csv`** as a *name*.
10. Choose the new file in the explorer, an editor opens.
11. Enter the following lines into the editor:

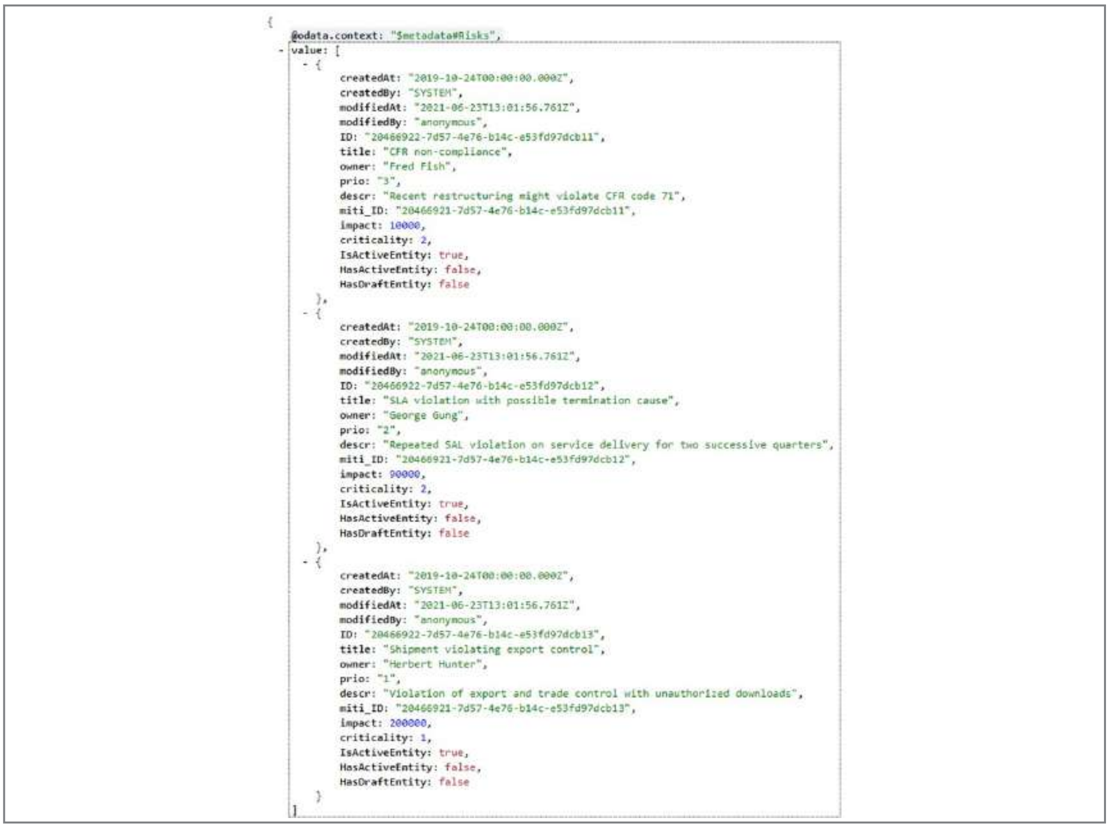
```
ID;createdAt;createdBy;descr;owner;timeline
20466921-7d57-4e76-b14c-e53fd97dcb11;2019-10-24;SYSTEM;SLA violation:
authorize account manager to offer service credits for recent delivery
issues;suitable BuPa;Q2 2020
20466921-7d57-4e76-b14c-e53fd97dcb12;2019-10-24;SYSTEM;"SLA violation:
review third party contractors to ease service delivery challenges;
trigger budget review";suitable BuPa;Q3 2020
20466921-7d57-4e76-b14c-e53fd97dcb13;2019-10-24;SYSTEM;Embargo
violation: investigate source of shipment request, revoke
authorization;SFSF Employee with link possible?;29.03.2020
20466921-7d57-4e76-b14c-e53fd97dcb14;2019-10-24;SYSTEM;Embargo
violation: review shipment proceedure and stop delivery until further
notice;SFSF Employee with link possible?;01.03.2020
```

12. Save the file.

    The files have the name of the namespace of the entities in the data model (for example `riskmanagement`), followed by a '-' and the name of the entity (for example `Risks`). When adhering to this naming convention, CAP recognizes the file as data for the data model and automatically adds it to the built-in SQLite database. Looking at the contents of the file `riskmanagement-Risks.csv`, the first line contains all the properties from your `Risks` entity. While the other ones are straightforward, consider the `miti_id` property. In your entity, you only have a `miti` property, so where does this come from? `miti` is an association to `Mitigations`, as `Mitigations` could have several key properties, the association on the database needs to point to all of these. Therefore, CAP creates a property `<AssociationProperty>_<AssociatedEntityKey>` for each key.

    As always, `cds watch` has noticed the change.

13. Return to the browser window where the service is still shown and press refresh in the browser. It will now show values for `Risks`



```
{
   @odata.context: "$metadata#Risks",
 - value: [
   - {
         createdAt: "2019-10-24T00:00:00.000Z",
         createdBy: "SYSTEM",
         modifiedAt: "2021-06-23T13:01:56.761Z",
         modifiedBy: "anonymous",
         ID: "20466922-7d57-4e76-b14c-e53fd97dcb11",
         title: "CFR non-compliance",
         owner: "Fred Fish",
         prio: "3",
         descr: "Recent restructuring might violate CFR code 71",
         miti_ID: "20466921-7d57-4e76-b14c-e53fd97dcb11",
         impact: 10000,
         criticality: 2,
         IsActiveEntity: true,
         HasActiveEntity: false,
         HasDraftEntity: false
     },
   - {
         createdAt: "2019-10-24T00:00:00.000Z",
         createdBy: "SYSTEM",
         modifiedAt: "2021-06-23T13:01:56.761Z",
         modifiedBy: "anonymous",
         ID: "20466922-7d57-4e76-b14c-e53fd97dcb12",
         title: "SLA violation with possible termination cause",
         owner: "George Gung",
         prio: "2",
         descr: "Repeated SAL violation on service delivery for two successive quarters",
         miti_ID: "20466921-7d57-4e76-b14c-e53fd97dcb12",
         impact: 90000,
         criticality: 2,
         IsActiveEntity: true,
         HasActiveEntity: false,
         HasDraftEntity: false
     },
   - {
         createdAt: "2019-10-24T00:00:00.000Z",
         createdBy: "SYSTEM",
         modifiedAt: "2021-06-23T13:01:56.761Z",
         modifiedBy: "anonymous",
         ID: "20466922-7d57-4e76-b14c-e53fd97dcb13",
         title: "Shipment violating export control",
         owner: "Herbert Hunter",
         prio: "1",
         descr: "Violation of export and trade control with unauthorized downloads",
         miti_ID: "20466921-7d57-4e76-b14c-e53fd97dcb13",
         impact: 200000,
         criticality: 1,
         IsActiveEntity: true,
         HasActiveEntity: false,
         HasDraftEntity: false
     }
 ]
```

Note:
To see the `JSON` data in a structured way as shown in the screenshot, you require a browser extension. Without such an extension, the data will just be displayed as plain text without indentations, and so on.

Note:
There are a few additional fields like `createdAt` or `modifiedBy` which you did not actively define in the entity itself. These were added by CAP, because you added the `managed` aspect in the entity definition:

```
entity Risks : managed { ... }
```

Other fields like `HasDraftEntity` were added because you added the `@odata.draft.enabled` annotation in the service definition (`risk-service.cds`).

You've now got a full-blown OData service, which complies to the OData standard and supports the respective queries without having to code anything but the data model and exposing the service itself.

Note:
The service is completely exposed without any authentication or authorization check. You will extend the service later with such checks.

Summary

You have added a data model, data, and a service definition to your project. Next, you will create a UI for your application.

Unit 2
Exercise 1

Create a CAP-Based Service

Simulation: Create a CAP-Based Service

For more information on *Create a CAP-Based Service*, please view the simulation in the lesson *Create a CAP-Based Service* in your online course.

1.

Unit 2
Solution 1

Create a CAP-Based Service

Simulation: Create a CAP-Based Service

For more information on *Create a CAP-Based Service*, please view the simulation in the lesson *Create a CAP-Based Service* in your online course.

1.

Reference Links

For your convenience, this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 2: Reference Links CAP-Based Service

Ref #	Section	Context text fragment	Brief description	Link
1	Add a Data Model to the Project	You are using the namespace riskmanagement.	Using namespaces	https://cap.cloud.sap/docs/guides/domain-models#using-namespaces
2	Add a Service to the Project	annotation enables edit sessions with draft states	Draft-based editing	https://cap.cloud.sap/docs/advanced/fiori#draft-support

LESSON SUMMARY

You should now be able to:

- Create a CAP-based service

Unit 2
Lesson 6

What is OData?

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Describe the OData standard for web-based applications

What is OData?

Usage Scenario

Your company is planning to develop a set of web-based applications. Open Data Protocol (OData) is intended to be used for data access as a standardized access protocol that meets the requirements of modern Web development.

Learning objectives

- Explain the OData protocol.
- Explain the purpose of an OData service.
- Explain the CRUD Operations used by OData.

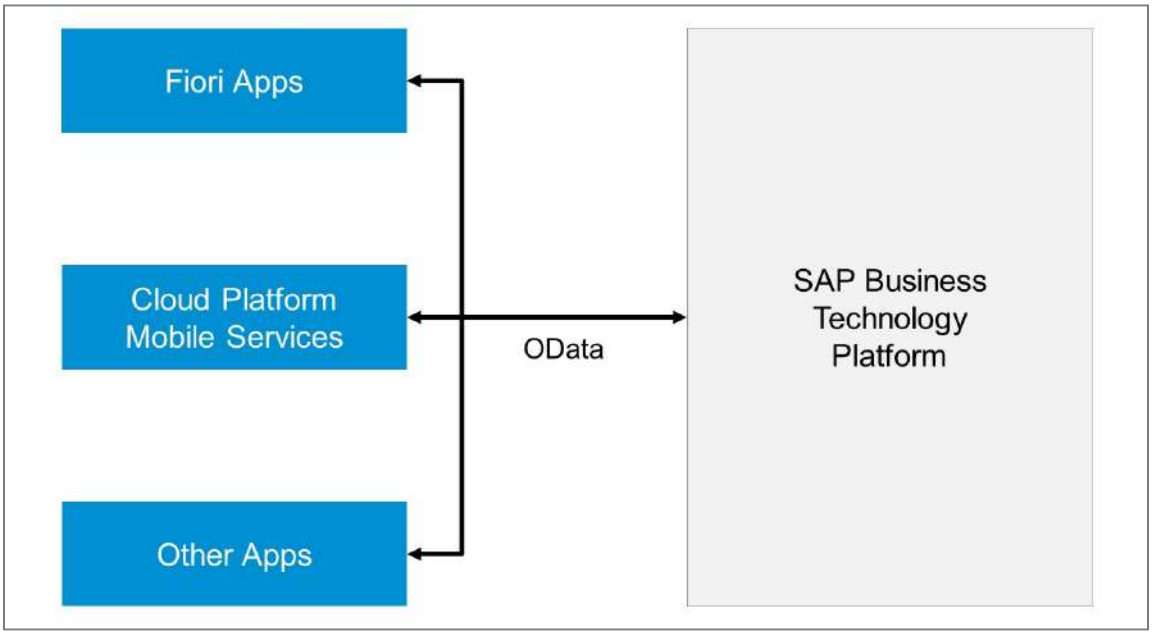
Overview

The OData is a data access protocol built on core protocols like HTTP and commonly accepted methodologies like REST (Representational State Transfer) for the web. Thus, as common practices of REST, OData builds on HTTP, AtomPub, and JSON using URIs to address and access data feed resources.

Key Challenges in Enterprise Computing for Business Consumers

The problem in the past was that for different consumers, a developer had to build a specific integration scenario that used its own protocol. However, one-off applications specific to each device or platform were not transferable. For example, a solution built for browser-based applications could not be used for an integration of enterprise software.

The solution is to use one protocol, the OData protocol.


Fiori Apps
Cloud Platform Mobile Services
Other Apps
OData
SAP Business Technology Platform

Watch this video to discover the key challenges in enterprise computing for business consumers and the solution.

Animation: Key Challenges in Enterprise Computing for Business Consumers

For more information on *Key Challenges in Enterprise Computing for Business Consumers*, please view the animation in the lesson *What is OData?*, in your online course.

OData Service

An **OData service** is a logical data model; it describes entities (resources) using associations and operations. The most important point is that the OData service forms a kind of contract between the UI and the backend system side, helping to bring together developers on both sides.

OData currently supports two formats for representing the resources it exposes - the XML-based AtomPub and the JSON formats. JSON has significantly less protocol overhead than the Atom Publishing protocol. JSON can easily be consumed with Java Script and by SAPUI5.

Each OData service is represented by a URI, called the service root URI. A URI is a uniform resource identifier, which is a string of characters used to identify a resource. More precisely, each resource can be accessed using a URL, a uniform resource locator, describing how to access the resource. This type of identification enables interaction with representations of the resource across a network using specific protocols like OData.

There are two types of document associated with each OData service:

- The service document
- The service metadata document

The service document lists entity sets, functions, and singletons that can be retrieved. Clients can use the service document to navigate the model in a hypermedia-driven fashion. The service document is available at `http://<host>:<port>/<service>/`.

The metadata document describes the types, sets, functions, and actions understood by the OData service. Clients can use the metadata document to understand how to query and interact with entities in the service. The service metadata document is available at `http://<host>:<port>/<service>/$metadata`. The URL will return XML metadata of the service (Entity data model). The response of a service metadata document only supports XML.

CRUD Operations

One of the main features of OData is that it uses the existing HTTP verbs GET, PUT, POST, and DELETE against addressable resources identified in the URI. Conceptually, OData is a way of performing database-style create, read, update, and delete operations on resources by using HTTP verbs:

- GET: Get the resource (a collection of entities, a single entity, a structural property, and so on).
- POST: Create a new resource.
- PUT: Update an existing resource by replacing it with a complete instance.
- PATCH: Update an existing resource by replacing part of its properties with a partial instance.
- DELETE: Remove the resource.

> Animation:
>
> For more information on , please view the animation in the lesson *What is OData?*, in your online course.

OData API

An Application Programming Interface (API) allows you to access data, for example, monitoring data.

The OData API is implemented as a REST API and the technical protocol is OData. This means that you can use standard HTTP methods (for example, the GET method) to call the API.

Summary

You now have a more profound understanding of the OData protocol and why it is so useful in modern web development.

Further Reading

- SAP Gateway – Building OData Services
- OData - the Best Way to REST
- Understand OData in 6 steps

LESSON SUMMARY

You should now be able to:

- Describe the OData standard for web-based applications

Unit 2
Lesson 7

What are APIs?

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Use APIs to exchange information in a standardized way

What are APIs?

Usage Scenario

Your company is planning to develop an application that will use standard functionality, such as creating and updating entities, that are provided by another company as Application Programming Interfaces (APIs). So, what are APIs and how can you use them?

Learning objectives

- Explain the meaning of APIs.
- Explain why APIs are important to make UI development easy.
- List rules when dealing with APIs.

Benefits of API

An API is a way to communicate with other software programs.

APIs specify how software programs are able to exchange information with each other, even if they are designed and run by different organizations. APIs facilitate interaction by selectively exposing functionality that allow different apps, websites, and devices to communicate effectively with each other. More importantly, APIs allow businesses to reach beyond regular business channels and share data, content, and services directly with both business to business (B2B) and business to consumer (B2C)) clients, making UI development easy.

Let us consider an example.

Your company creates an application for customers to create and display maps. You create an API to enable customers to display your company's maps in their application,

The customer creates an application, such as a customer relationship management (CRM) application. Within this CRM application, the customer calls your API to add a map next to their customer’s profile.

Your application receives the call and does something, such as access your proprietary map database. What your application does can be as complicated as it needs to be, but the customer doesn't need to know exactly what you are doing behind the scenes.

When your application is ready, it sends a response. For example, a map for an address, entered by the customer.

To summarize, you create an API and customers call your API.

When your application is ready, it sends a response — either a confirmation that you did something, or some information, requested by the customer.

You do something and you send back a response that gives your customers the information they need for their own application.

Rules to Develop and Deploy APIs

Since the developer of the API and the consumers usually do not know each other, it is useful to follow some rules:

- API documentation is a pillar of the developer experience. Good API documentation ensures that the developers fully understand the API, can use it, and don't get stuck.
- Customers and partners find it easier to adopt APIs with names that are meaningful, clear, and self-explanatory.
- Developer or service guides complement API references by describing how to use an API and related services, any SDKs, or the development platform in general.
- Rules dealing with the deprecation of APIs and the decommissioning of their deliverables.

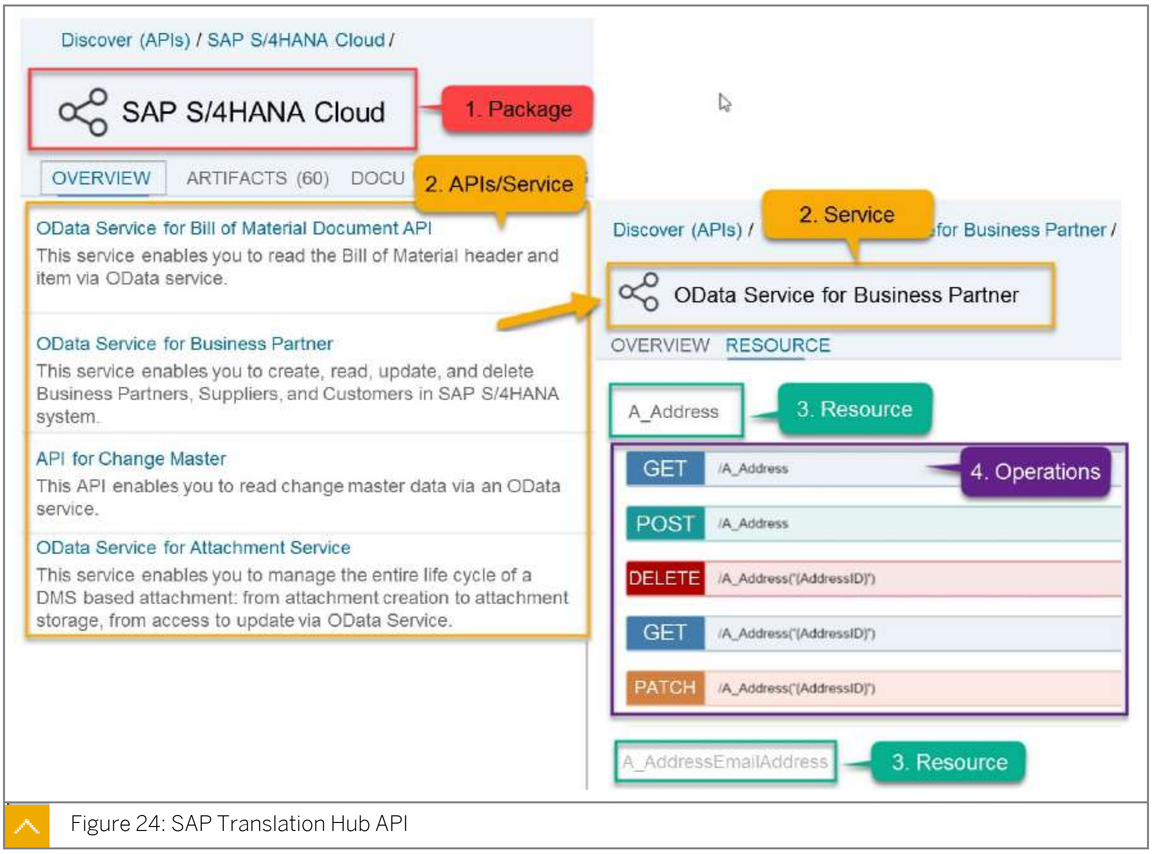
APIs in the SAP API Business Hub

Watch this video to learn about the SAP API Business Hub.

Animation: APIs in the SAP API Business Hub

For more information on *APIs in the SAP API Business Hub*, please view the animation in the lesson *What are APIs?*, in your online course.

The SAP API Business Hub is a web application hosted by SAP to discover, explore, and test SAP and partner APIs that are required to build extensions or process integrations. In the SAP API Business Hub, the information about APIs is organized according to the following hierarchy as shown in the SAP Translation Hub API:



Discover (APIs) / SAP S/4HANA Cloud /

SAP S/4HANA Cloud

OVERVIEW ARTIFACTS (60) DOCU

OData Service for Bill of Material Document API
This service enables you to read the Bill of Material header and item via OData service.

OData Service for Business Partner
This service enables you to create, read, update, and delete Business Partners, Suppliers, and Customers in SAP S/4HANA system.

API for Change Master
This API enables you to read change master data via an OData service.

OData Service for Attachment Service
This service enables you to manage the entire life cycle of a DMS based attachment: from attachment creation to attachment storage, from access to update via OData Service.

Discover (APIs) / ...for Business Partner /

OData Service for Business Partner

OVERVIEW RESOURCE

A_Address

GET	/A_Address
POST	/A_Address
DELETE	/A_Address('{AddressID}')
GET	/A_Address('{AddressID}')
PATCH	/A_Address('{AddressID}')

A_AddressEmailAddress

Figure 24: SAP Translation Hub API

- API Package: An API Package is a collection of related APIs or services, belonging to one product or product area, packaged and delivered together.
- API/service: An API or service is a collection of related resources and operations available for each resource.
- Resource: A resource is a remote data entity, identified by a URI, on which operations are performed.
- Operation: An operation is a data operation, such as GET, POST, PUT, UPDATE, or DELETE, performed on a resource.

Summary

When software programs want to exchange information in a standardized way, such as creating or modifying business partners (entities), these standard functions can be defined as APIs. This also allows that APIs may be developed and operated by different organizations. It is then important to follow some rules when developing and deploying APIs.

Further Reading

- What is an API?

LESSON SUMMARY

You should now be able to:

- Use APIs to exchange information in a standardized way

Unit 2
Lesson 8

JSON/YAML

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Explain JSON and YAML

JSON and YAML

Usage scenario

You use standardized techniques for modern web development. This includes the use of JavaScript Object Notation (JSON) and YAML (originally means Yet Another Markup Language) for data exchange and configuration files, for example. You need to become more familiar with these open standard formats.

Learning objectives

- Explain JSON
- Explain YAML
- Explain YAML in relation to JSON

What are JSON and YAML?

JSON

JSON is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute:value pairs and arrays. It is a common data format with a diverse range of functionality in data interchange including communication of web applications with servers.

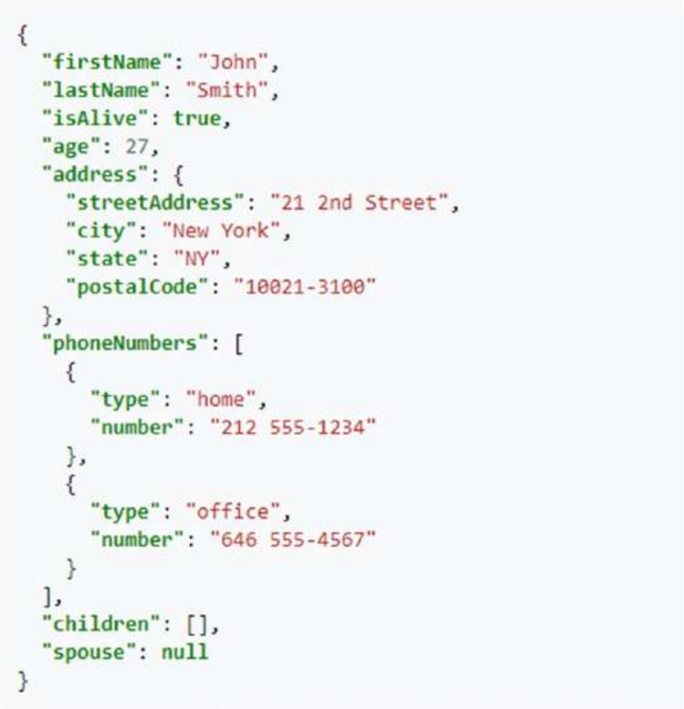
JSON, is a lighter plain-text alternative to XML and based on JavaScript syntax, but is independent of JavaScript and supported in other programming languages as well. JSON filenames use the extension `.json`.

JSON is built on two structures:

- A collection of name : value pairs. In various languages, this is realized as an object.
- An ordered list of values. In most languages, this is realized as an array.

The following example shows a possible JSON representation describing a person.

```
{
   "firstName": "John",
   "lastName": "Smith",
   "isAlive": true,
   "age": 27,
   "address": {
     "streetAddress": "21 2nd Street",
     "city": "New York",
     "state": "NY", "postalCode": "10021-3100"
   },
 "phoneNumbers": [
   {
     "type": "home",
     "number": "212 55-1234"
   },
   {
     "type": "office",
     "number": "646 555-4567"
   }
  ],
  "children": [],
  "spouse": null
 }
```



```
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 27,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [],
  "spouse": null
}
```

Figure 25: JSON Representation Describing a Person

YAML

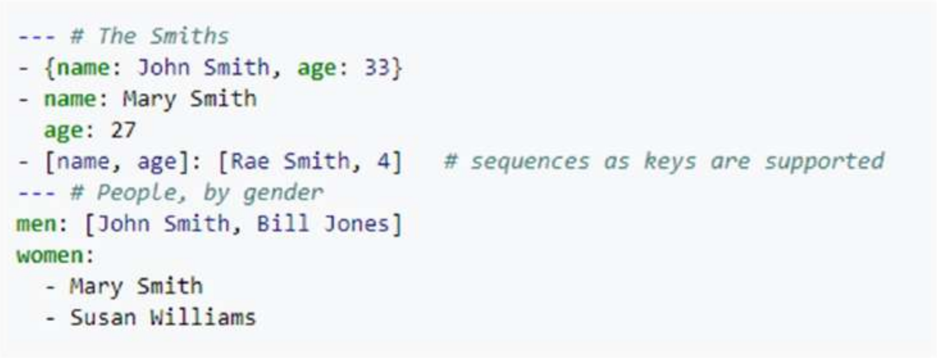
YAML is a human-friendly, cross language, Unicode based data serialization language designed around the common native data types of agile programming languages. It is broadly useful for programming needs ranging from configuration files to internet messaging to object persistence to data auditing.

YAML was specifically created to work well for common use cases such as: configuration files, log files, interprocess messaging, cross-language data sharing, object persistence, and debugging of complex data structures. When data is easy to view and understand, programming becomes a simpler task. YAML filenames use the extension `.yaml` or `.yml`.

YAML, is a strict JSON superset and includes additional features such as the notion of tagging data types, support for non-hierarchical data structures, the option to structure data with indentation, and multiple forms of scalar data quoting. YAML is an open format.

The following example shows a possible YAML representation describing a family.

```
--- # The Smiths
 - { name: John Smith, age: 33 }
 - name: Mary Smith
   age: 27
 - [name, age]: [Rae Smith, 4] # sequences as keys are supported
 --- # People, by gender
 men: [John Smith, Bill Jones]
 women:
   - Mary Smith
   - Susan Williams
```



```
--- # The Smiths
- {name: John Smith, age: 33}
- name: Mary Smith
  age: 27
- [name, age]: [Rae Smith, 4]   # sequences as keys are supported
--- # People, by gender
men: [John Smith, Bill Jones]
women:
  - Mary Smith
  - Susan Williams
```

Figure 26: YAML Representation Describing a Family

Objects and lists are important components in YAML and can be mixed. The first example is a list of key-value objects, all people from the Smith family. The second lists them by gender; it is a key-value object containing two lists.

YAML in Relation to JSON

Both JSON and YAML aim to be human readable data interchange formats. However, JSON and YAML have different priorities.

Animation:

For more information on , please view the animation in the lesson *JSON/YAML*, in your online course.

JSON's foremost design goal is simplicity and universality. Therefore, JSON is trivial to generate and parse, at the cost of reduced human readability. It also uses a lowest common denominator information model, ensuring any JSON data can be easily processed by every modern programming environment.

In contrast, YAML's foremost design goals are human readability and support for serializing arbitrary native data structures. Thus, YAML allows for extremely readable files, but is more complex to generate and parse. In addition, YAML ventures beyond the lowest common denominator data types, requiring more complex processing when crossing between different programming environments.

YAML can therefore be viewed as a natural superset of JSON, offering improved human readability and a more complete information model. This is also the case in practice; every JSON file is also a valid YAML file. This makes it easy to migrate from JSON to YAML if or when the additional features are required.

Summary

You now have a more profound understanding of JSON and YAML.

Further Reading

- JSON - Wikipedia
- YAML - Wikipedia

LESSON SUMMARY

You should now be able to:

- Explain JSON and YAML

Learning Assessment

1. What is the default IDE for SAP's multi-cloud environment?

   Choose the correct answer.

   □ A Eclipse

   □ B Oxygen

   □ C SAP Business Application Studio

   □ D Microsoft Visual Code

2. Which dev space type should you select to extend SAP S/4HANA with a CAP project?

   Choose the correct answer.

   □ A SAP Fiori

   □ B SAP HANA Native Application

   □ C Full Stack Cloud Application

3. Which account types does SAP BTP offer?

   Choose the correct answers.

   □ A Global accounts

   □ B Regional accounts

   □ C Subaccounts

   □ D Local accounts

4. What do you use to access and manage accounts?

   Choose the correct answer.

   □ A SAP Business Application Studio

   □ B SAP BTP Cockpit

   □ C SAP Integration Suite

5. Commercial models (license types) are connected directly to which account type?

   Choose the correct answer.

   □ A Subaccount

   □ B Global account

6. How many global accounts are associated with one license type?

   Choose the correct answer.

   □ A Many

   □ B 2

   □ C 1

7. You can you purchase entitlements and share them across multiple global accounts.

   Determine whether this statement is true or false.

   □ True

   □ False

8. In how many regions and environments can one subaccount run?

   Choose the correct answer.

   □ A Multiple regions and one environment.

   □ B Multiple regions and multiple environments.

   □ C Exactly one region and one environment.

   □ D Exactly one region and multiple environments.

9. What are directories?

   Choose the correct answer.

   □ A Groups of subaccounts

   □ B Groups of environments

   □ C Groups of applications

10. Which of the following is provided by an environment?

Choose the correct answers.

□ A Runtime

□ B Subaccounts

□ C Services

□ D Tools

□ E Regions

11. Which environment would you use to write robust, transactional cloud applications?

Choose the correct answer.

□ A Kyma

□ B ABAP

□ C Cloud Foundry

12. Which environment would you use to develop small to medium extensions?

Choose the correct answer.

□ A Kyma

□ B ABAP

□ C Cloud Foundry

13. Which environment would you use for an open build-on approach?

Choose the correct answer.

□ A Kyma

□ B ABAP

□ C Cloud Foundry

14. What is SAP Business Application Studio?

    Choose the correct answer.

    □ A A Microsoft Visual Studio Code extension

    □ B A web browser-based development environment

    □ C An SAP Mobile Application

    □ D A source code management system

15. In SAP Business Application Studio, how many dev space types are available?

    Choose the correct answer.

    □ A 3

    □ B 4

    □ C 5

    □ D 6

16. Which command do you use to install dependencies in your project?

    Choose the correct answer.

    □ A cds

    □ B yum

    □ C bash

    □ D npm

17. In data models, are namespaces optional or mandatory?

    Choose the correct answer.

    □ A mandatory

    □ B optional

18. What is the difference between entities and types?

    Choose the correct answer.

    □ A Types represent data elements, entities describe aspects of types.

    □ B Entities represent data, types describe properties of entity elements.

19. What is OData?

Choose the correct answer.

□ A A standard to access data through RESTful APIs

□ B A standard to create user interfaces (UI) for applications

□ C A standard to share data under a creative commons license

20. Which of the following does OData use to address and access data feed resources?

Choose the correct answer.

□ A URL

□ B URN

□ C URI

21. Which document types are associated with an OData service?

Choose the correct answers.

□ A Service manifest document

□ B Service document

□ C Service description document

□ D Service metadata document

22. What is an Application Programming Interface (API)?

Choose the correct answer.

□ A A software development kit (SDK) for mobile applications

□ B A way for applications to interact with other applications

□ C A way for applications to remotely modify other applications

23. Where can you find APIs provided by SAP and SAP partners?

Choose the correct answer.

□ A SAP Cloud Connector

□ B SAP Business Application Studio

□ C SAP API Business Hub

24. JSON is based on which programming language?

Choose the correct answer.

□ A Java

□ B Julia

□ C JCL

□ D JavaScript

25. What are the two structures JSON is built on?

Choose the correct answers.

□ A Collections of name/value pairs

□ B Collections of value/value pairs

□ C Unordered list of strings

□ D Ordered list of values

26. What is the relationship between YAML and JSON?

Choose the correct answer.

□ A YAML and JSON are unrelated.

□ B JSON is a superset of YAML.

□ C YAML is a superset of JSON.

27. Which of the following supports non-hierarchical data?

Choose the correct answer.

□ A YAML

□ B JSON

□ C Neither YAML nor JSON

28. Which of the following statements are correct?

    Choose the correct answers.

    □ A Each JSON file is a valid YAML file.

    □ B Each YAML file is a valid JSON file.

    □ C JSON’s foremost design goal is support for serializing arbitrary native data structures.

    □ D YAML’s foremost design goal is support for serializing arbitrary native data structures.

UNIT 3 User Interface and Business Logic

UNIT OBJECTIVES

- Generate a User Interface (UI) using SAP Fiori Elements
- List the design principles and key benefits of SAP Fiori
- Evaluate the difference between SAP Fiori and SAP Fiori Elements
- Add custom business logic
- Evaluate when to use event handlers
- Explain error handling

Unit 3
Lesson 1

Generate a User interface

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Generate a User Interface (UI) using SAP Fiori Elements

Generate the User Interface Using SAP Fiori Elements: Exercise Overview

Scenario

An SAP Fiori Elements (FE) app is an application that uses SAPUI5, its controls, and its Model View Controller (MVC) concepts. Most code of an SAP FE app is outside the project, managed centrally by the SAP FE team. The code inside a project only references these central components, which take care of creating the UI according to the latest SAP Fiori design guidelines and covers all the controller logic for you out of the box. The UI can be influenced by OData annotations. They determine, for example, which properties of an OData service make up the columns of a table that displays the content of the service.

Task Flow

In this exercise, you will perform the following tasks:

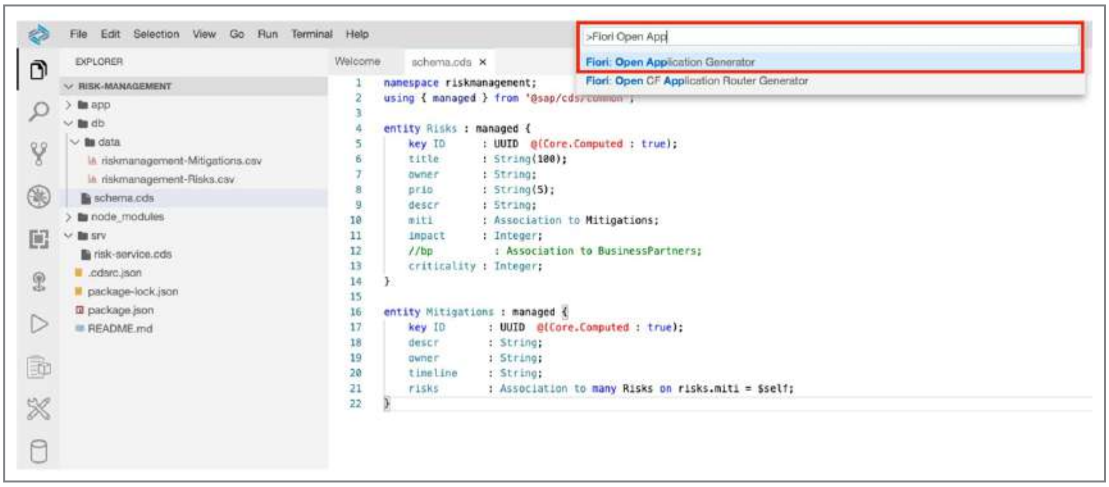
- Build a user interface with SAP Fiori elements [1].
- Explain the advantages of using SAP Fiori elements.
- Modify the user interface with OData annotations.

Prerequisites

You have successfully implemented an SAP CAP-based service.

Generate the UI with an SAP Fiori Elements Template

1. In SAP Business Application Studio, invoke the Command Palette (*View → Find Command* or `Cmd+Shift+P`) and choose *Fiori: Open Application Generator*



>Fiori Open App

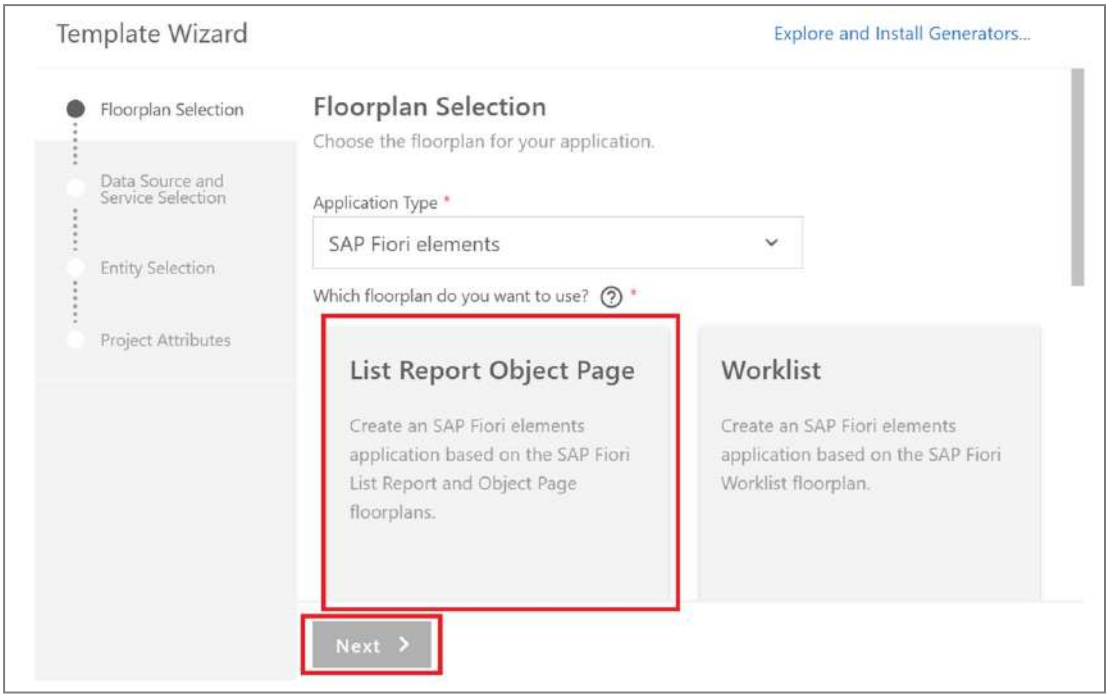
Fiori: Open Application Generator

Fiori: Open CF Application Router Generator

```
namespace riskmanagement;
using { managed } from '@sap/cds/common';

entity Risks : managed {
    key ID      : UUID  @(Core.Computed : true);
    title       : String(100);
    owner       : String;
    prio        : String(5);
    descr       : String;
    miti        : Association to Mitigations;
    impact      : Integer;
    //bp          : Association to BusinessPartners;
    criticality : Integer;
}

entity Mitigations : managed {
    key ID       : UUID  @(Core.Computed : true);
    descr        : String;
    owner        : String;
    timeline     : String;
    risks        : Association to many Risks on risks.miti = $self;
}
```

.

2. Choose *SAP Fiori Elements* and *List Report Object Page* and select *Next*.



Floorplan Selection

Choose the floorplan for your application.

Application Type *: SAP Fiori elements

Which floorplan do you want to use? *

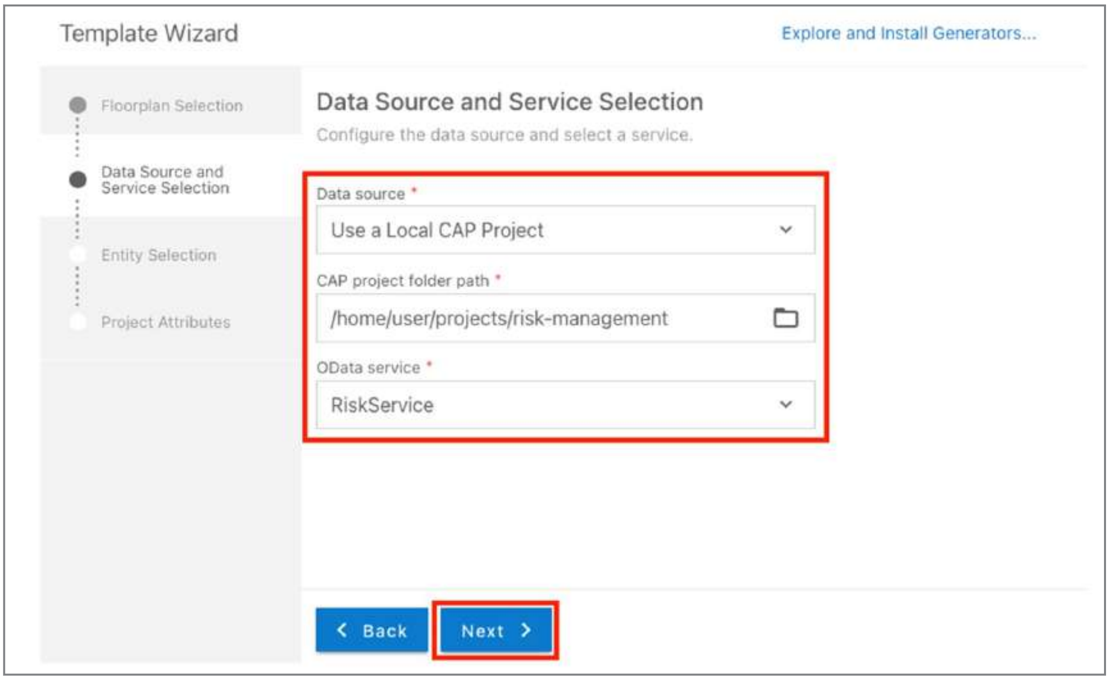
List Report Object Page

Create an SAP Fiori elements application based on the SAP Fiori List Report and Object Page floorplans.

Worklist

Create an SAP Fiori elements application based on the SAP Fiori Worklist floorplan.

3. In the next dialog, choose *Use a Local CAP Node.js Project* and point to the folder of your current `RiskManagement` project. Select `RiskService` as the OData service and select *Next*.



Data Source and Service Selection

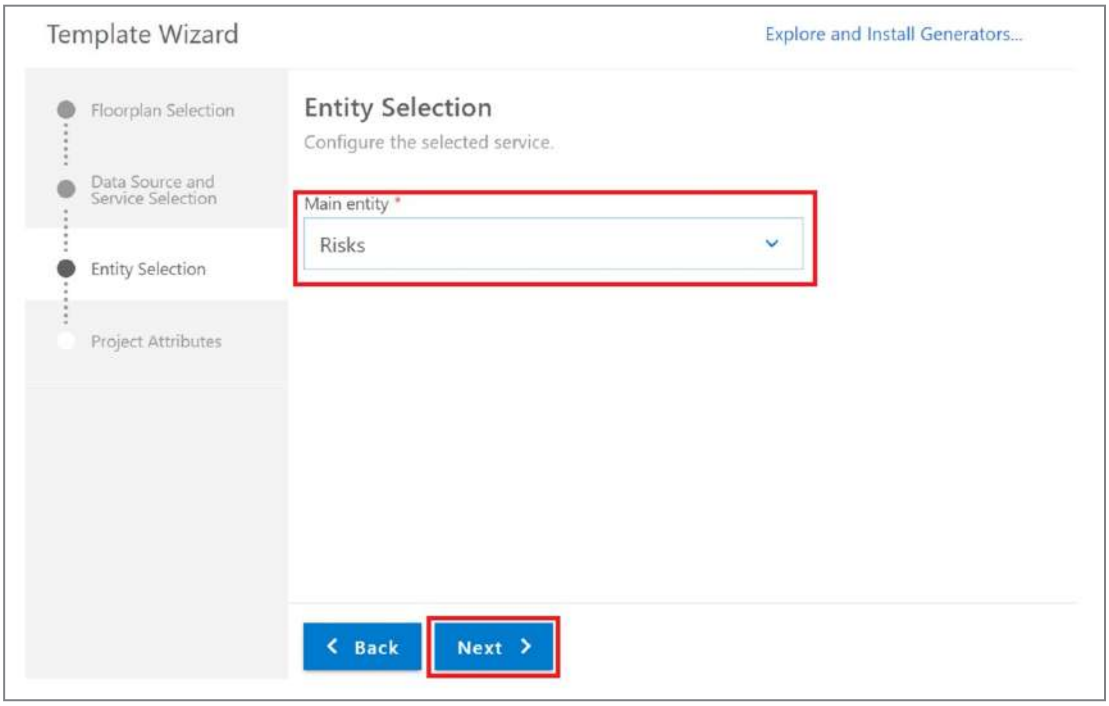
Configure the data source and select a service.

Data source *: Use a Local CAP Project

CAP project folder path *: /home/user/projects/risk-management

OData service *: RiskService

4. Choose `Risks` as the *main entity* and choose *Next*.

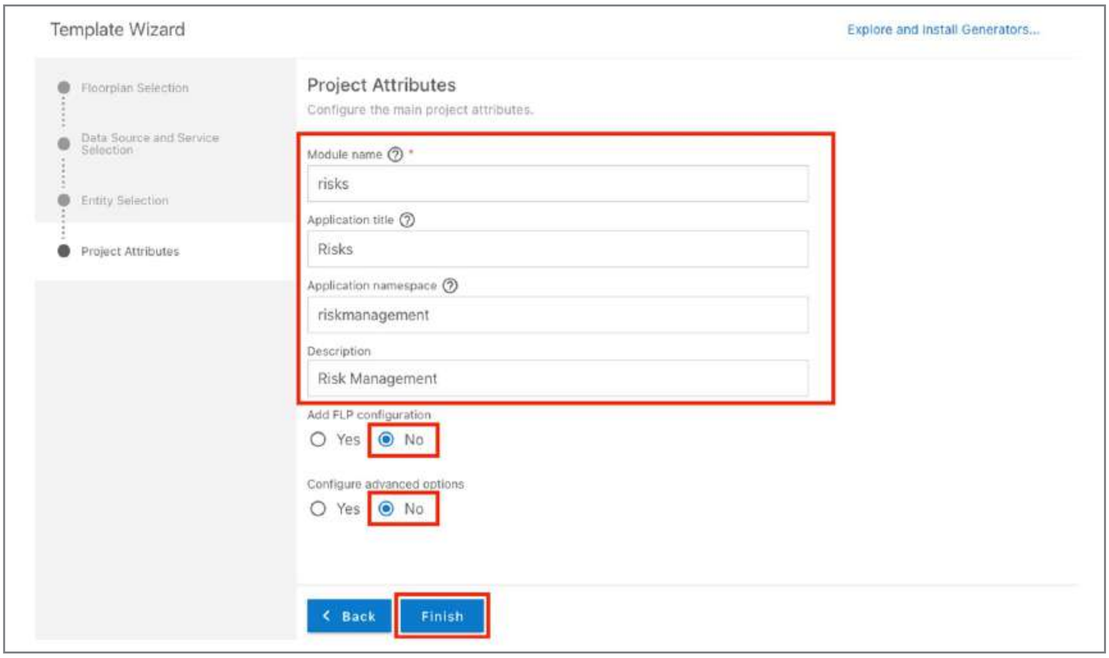


Entity Selection

Configure the selected service.

Main entity *: Risks

5. Enter **`risks`** as the *module name*. Enter **`Risks`** as the *application title* and the description for the application, as well as **`riskmanagement`** as the *namespace*. Choose **`No`** for all additional settings. Choose *Finish*.



Project Attributes

Configure the main project attributes.

Module name *: risks

Application title: Risks

Application namespace: riskmanagement

Description: Risk Management

Add FLP configuration: ○ Yes ● No

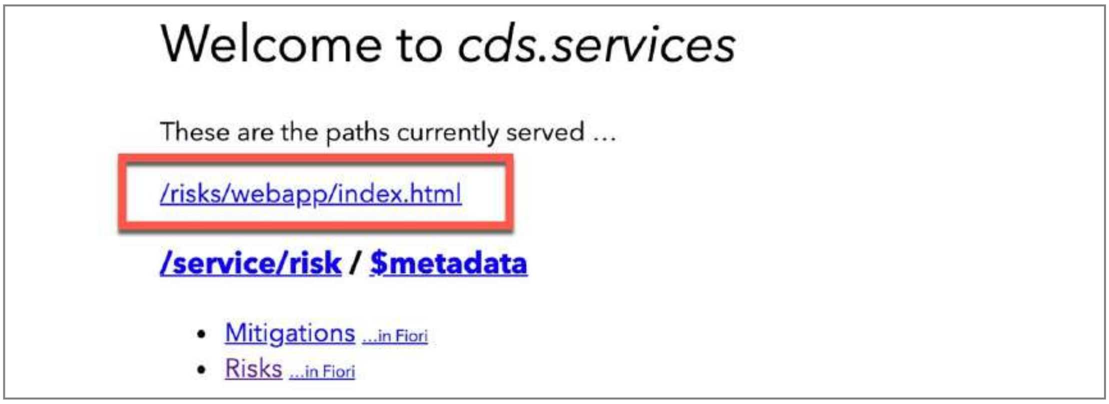
Configure advanced options: ○ Yes ● No

(If you get a pop-up that says "**`A project has been generated. What would you like to do with it?`**", you can ignore it and just close the pop-up).

The application is now generated and after a couple of seconds you can see it in the `app` folder of your project. It contains a `risks`and a `webapp` folder with a `Component.js` file, which is characteristic for a UI5 app. However, the code there is minimal and it basically inherits its logic from the `sap/fe/core/AppComponent`.

6. If `cds watch` isn't still running from the previous chapter, execute it in a terminal and press on the *Open in New Tab* button in the right lower corner. If it is still running from the last chapter, it is enough to refresh the browser page where it is running.

   You can now see that `cds watch` has discovered an HTML page in your app folder:



Welcome to *cds.services*

These are the paths currently served …

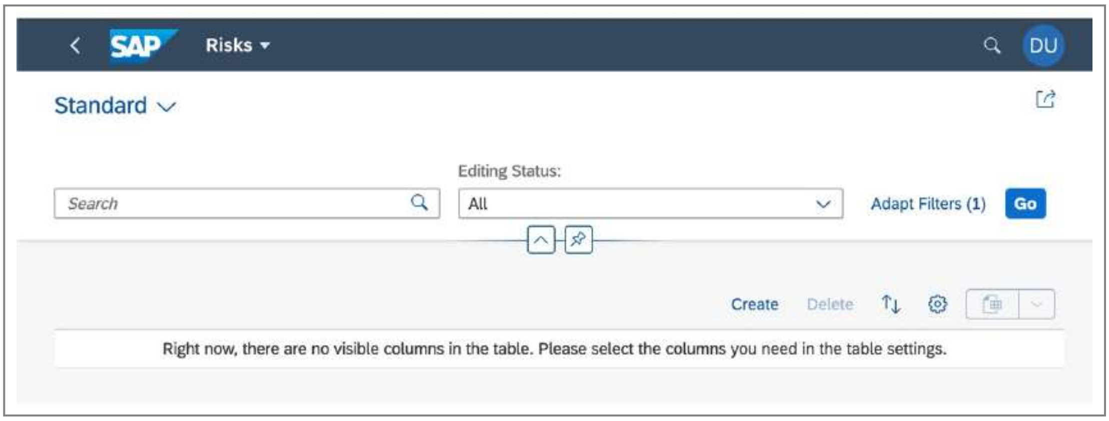
/risks/webapp/index.html

/service/risk / $metadata

- Mitigations ...in Fiori
- Risks ...in Fiori

7. Select this link. On the launch page that now comes up, Choose the *Risks* tile.

   You can now see the list page, it looks like this:



Right now, there are no visible columns in the table. Please select the columns you need in the table settings.

Unfortunately, the app looks rather empty, for example, the list has no columns yet. This is because we miss an essential part of a SAP Fiori elements application that tells it about columns, form fields and many other things: It is missing UI annotations.

Modify the UI with OData Annotations

Now we are going to modify the UI with annotations.

Since, our entire UI content goes into the `app/` folder, we create our annotation files inside this folder.

Project Folder/file	Content
`app/`	UI content
`db/`	Domain Models and db-related content
`srv/`	Service definitions and implementations
`package.json`	Your project descriptor

The SAP Fiori Generator automatically generates the `services.cds` file inside the app folder. This file ensures all annotation files are loaded: `usingfrom'./risks/annotations';`

In case you are splitting up your annotations into multiple files, make sure that you also include the files in the `services.cds`.

Usually, you can also have multiple SAP Fiori / frontend projects in your CAP project. To avoid writing redundant annotations for the same entities or schemas, you can create - based on the guidelines - the file `common.cds` inside the `app/` folder. All annotations inside this file will then apply to all UI5-Applications.

1. To add the OData annotations, in the project, go to folder *app* representing the service and select *New File* in the context menu.
2. Enter **`common.cds`** as a *name*.
3. Select the new file in the explorer, an editor opens.
4. Enter the following lines into the editor:

```
using riskmanagement as rm from '../db/schema';

 // Annotate Risk elements
 annotate rm.Risks with {
 ID @title : 'Risk';
 title @title : 'Title';
 owner @title : 'Owner';
 prio @title : 'Priority';
 descr @title : 'Description';
 miti @title : 'Mitigation';
 impact @title : 'Impact';
 }

// Annotate Miti elements
 annotate rm.Mitigations with {
 ID @(
      UI.Hidden,
      Commong : {Text : descr}
 );
owner @title : 'Owner';
descr @title : 'Description';
 }
annotate rm.Risks with {
   miti @(Common : {
    //show text, not id for mitigation in the context of risks
    Text : miti.descr,
    TextArrangement : #TextOnly,
    ValueList : {
    Label : 'Mitigations',
    CollectionPath : 'Mitigations',
    Parameters : [
    {
       $Type : 'Common.ValueListParameterInOut',
       LocalDataProperty : miti_ID,
       ValueListProperty : 'ID'
       },
       {
       $Type : 'Common.ValueListParameterDisplayOnly',
       ValueListProperty : 'descr'
       }
    ]
    }
  });
 }
```

5. Now we have to include the `common.cds` file into the `services.cds` file:

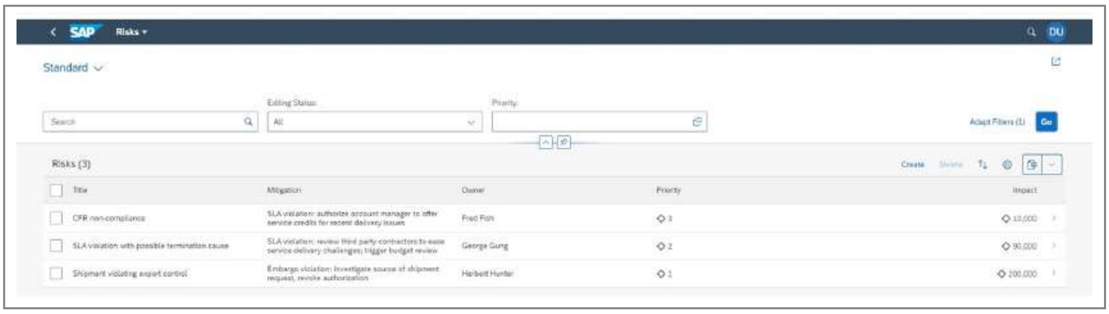
```
using from './risks/annotations';
 using from './common';
```

6. Open `app/risks/annotations.cds`. Here we are going to use UI Annotations to tell SAP Fiori Elements how the *List* and *Object* page should look. These annotations will only apply to the `risks` app.

```
using RiskService from '../../srv/risk-service';
 // Risk List Report Page
 annotate RiskService.Risks with @(UI : {
    HeaderInfo : {
       TypeName : 'Risk',
       TypeNamePlural : 'Risks',
       Title : {
          $Type : 'UI.DataField',
          Value : title
       },
       Description : {
          $Type : 'UI.DataField',
            Value : descr
       }
    },
    SelectionFields : [prio],
    Identification : [{Value : title}],
    // Define the table columns
    LineItem : [
       {Value : title},
       {Value : miti_ID},
       {Value : owner},
       {
          Value : prio,
          Criticality : criticality
       },
       {
          Value : impact,
          Criticality : criticality
       },
    ],
 });
 // Risk Object Page
 annotate RiskService.Risks with @(UI : {
     Facets : [{
        $Type : 'UI.ReferenceFacet',
        Label : 'Main',
        Target : '@UI.FieldGroup#Main',
     }],
     FieldGroup #Main : {Data : [
       {Value : miti_ID},
       {Value : owner},
       {
           Value : prio,
           Criticality : criticality
       },
       {
           Value : impact,
           Criticality : criticality
       }
    ]},
 });
```

7. Save all files. As in the steps before, `cds watch` has noticed the new file and compiled the service again, so now it contains the additional annotations.

8. In the browser, reload the test page, which shows the service and the index page. Select the index page link `/risks/webapp/index.html`. On the launch page that now comes up, choose the *Risks* tile. Select *Go*. It now shows a work list with some columns and the data from the service.



Risks (3)

Title	Mitigation	Owner	Priority	Impact
CFR non-compliance	SLA violation: authorize account manager to offer service credits for recent delivery issues	Fred Fish	◇ 3	◇ 10,000
SLA violation with possible termination cause	SLA violation: review third party contractors to ease service delivery challenges; trigger budget review	George Gung	◇ 2	◇ 90,000
Shipment violating export control	Embargo violation: Investigate source of shipment request, revoke authorization	Herbert Hunter	◇ 1	◇ 200,000

You’ve now already finished a full-blown service with a full-blown UI application on top, running locally.

Check the Annotation Files

Let's have a look at the `common.cds` file and the annotations in there. At the beginning, we see:

```
using riskmanagement as rm from '../db/schema';

 // Annotate Risk elements
 annotate rm.Risks with {
    ID @title : 'Risk';
    title @title : 'Title';
    owner @title : 'Owner';
    prio @title : 'Priority';
    descr @title : 'Description';
    miti @title : 'Mitigation';
    impact @title : 'Impact';
 }
```

It's referring to the definitions of the `schema.cds` file that defines the `Risks` and `Mitigations` entities. Then it annotates the `Risk`entity with numerous texts. These should be in a translatable file normally, but for now, we will keep them here. These texts are used as labels in form fields and column headers by SAP Fiori elements.

Next, take a look at the `app/risks/annotations.cds` file:

```
using RiskService from '../../srv/risk-service';

 // Risk List Report Page
 annotate RiskService.Risks with @(UI : {
    HeaderInfo : {
       TypeName : 'Risk',
       TypeNamePlural : 'Risks',
       Title : {
          $Type : 'UI.DataField',
          Value : title
},
Description : {
 $Type : 'UI.DataField',
 Value : descr
 }
 },
 SelectionFields : [prio],
 Identification : [{Value : title}],
 // Define the table columns
     LineItem : [
        {Value : title},
        {Value : miti_ID},
        {Value : owner},
        {
            Value : prio,
            Criticality : criticality
        },
        {
            Value : impact,
            Criticality : criticality
        },
    ],
 });
```

This file defines the content of the work list page and the object page, to which you are navigated, when you select a line in the work list.

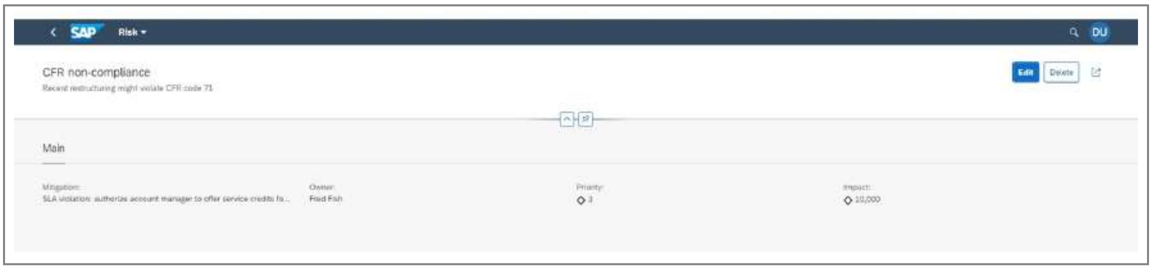
The `SelectionFields` section defines which of the properties are exposed as search fields in the header bar above the list, in this case `prio` is the only explicit search field.

From the `LineItem` section all the columns and their order of the work list are derived. While in most cases the columns are defined by `Value:` followed by the property name of the entity, in the case of `prio` and `impact` there's also `Criticality`, which for now you can disregard, but keep it in mind in case you go to the later modules. It currently adds a diamond icon right left of the fields. You can just ignore it.

The next section defines the content of the object page:

```
// Risk Object Page
 annotate RiskService.Risks with @(UI : {
    Facets : [{
       $Type : 'UI.ReferenceFacet',
       Label : 'Main',
       Target : '@UI.FieldGroup#Main',
    }],
    FieldGroup #Main : {Data : [
      {Value : miti_ID},
      {Value : owner},
      {
          Value : prio,
          Criticality : criticality
      },
      {
          Value : impact,
          Criticality : criticality
      }
   ]},
 });
```

This section defines a single facet, a `ReferenceFacet`, of the field group `FieldGroup#Main`. This field group just shows up as a form. The properties of the `Data` array within `FieldGroup#Main` determine the fields in the form:



CFR non-compliance

Recent restructuring might violate CFR code 71

Mitigation: SLA violation: authorize account manager to offer service credits fo...

Owner: Fred Fish

Priority: ◇ 3

Impact: ◇ 10,000

The last part of the `common.cds` file is the most complicated one:

```
annotate rm.Risks with {
    miti @(Common : {
    //show text, not id for mitigation in the context of risks
    Text : miti.descr,
    TextArrangement : #TextOnly,
    ValueList : {
       Label : 'Mitigations',
       CollectionPath : 'Mitigations',
       Parameters : [
         {
            $Type : 'Common.ValueListParameterInOut',
            LocalDataProperty : miti_ID,
            ValueListProperty : 'ID'
         },
         {
            $Type : 'Common.ValueListParameterDisplayOnly',
            ValueListProperty : 'descr'
         }
      ]
    }
  });
}
```

Without these lines, you would see the id of the mitigations from the `miti` field, in both the list and the object page:

By introducing the annotations for the `miti` property, instead of just displaying the original value of `miti`, that is, the ID, the UI shows its `description` property. The subsequent part `ValueList` introduces a value help for `miti` that you can see on the object page in its edit mode. The value help takes the id as an input parameter and again displays the `description` parameter.

Summary

You have created a SAP Fiori elements UI for your application. Next, you will add custom business logic to your application to add highlighting to some fields.

Unit 3
Exercise 2

Generate a User Interface using SAP Fiori Elements

Simulation: Generate the User Interface using SAP Fiori Elements

For more information on *Generate the User Interface using SAP Fiori Elements*, please view the simulation in the lesson *Generate a User interface* in your online course.

1.

Unit 3
Solution 2

Generate a User Interface using SAP Fiori Elements

Simulation: Generate the User Interface using SAP Fiori Elements

For more information on *Generate the User Interface using SAP Fiori Elements*, please view the simulation in the lesson *Generate a User interface* in your online course.

1.

Reference Links

For your convenience, this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 3: Reference Links: User Interface

Ref#	Section	Context text fragment	Brief description	Link
1	Task Flow	Build a user interface with SAP Fiori elements	SAP Fiori elements	https://experience.sap.com/fiori-design-web/smart-templates/

LESSON SUMMARY

You should now be able to:

- Generate a User Interface (UI) using SAP Fiori Elements

Unit 3
Lesson 2

UI: SAP Fiori

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- List the design principles and key benefits of SAP Fiori

SAP Fiori

Usage Scenario

Your company is planning to develop a set of cloud-based applications. These applications are to be used on different types of devices such as mobile phones, tablets, and laptops. The appearance of these applications should be similar on these devices to reduce the learning time for the end-users. Therefore, you need an efficient method for designing and developing the user interface of these applications. SAP Fiori provides an efficient and consistent way to serve frontends to your business applications.

Animation: Usage Scenario

For more information on *Usage Scenario*, please view the animation in the lesson *UI: SAP Fiori*, in your online course.

Learning objectives

- Explain the SAP Fiori user experience (UX).
- Explain the SAP Fiori design principles.
- List the key benefits of SAP Fiori.

Overview

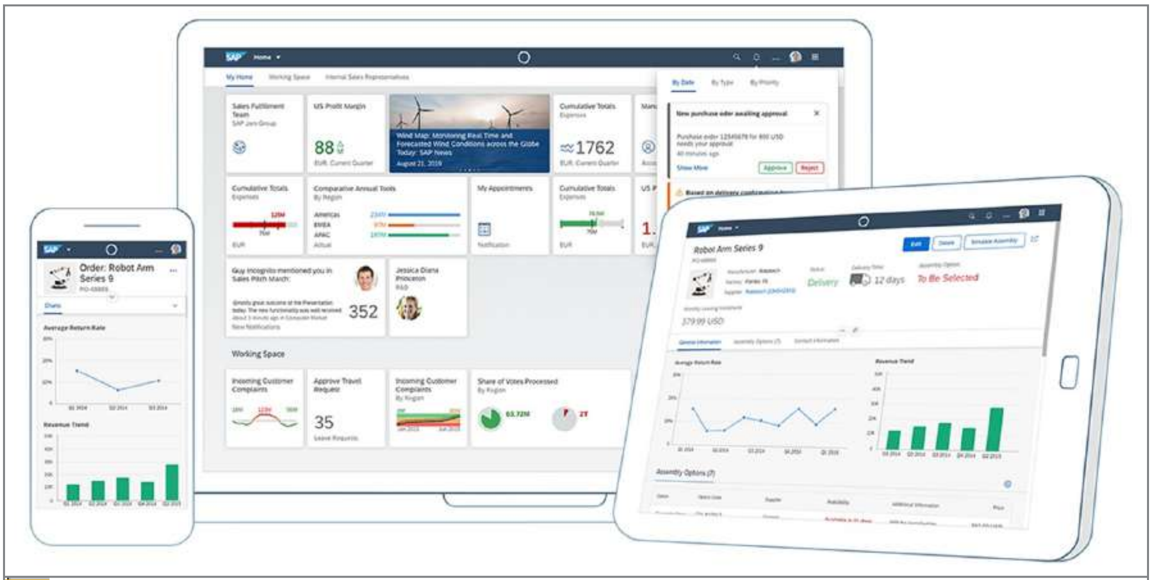


Figure 29: SAP Fiori

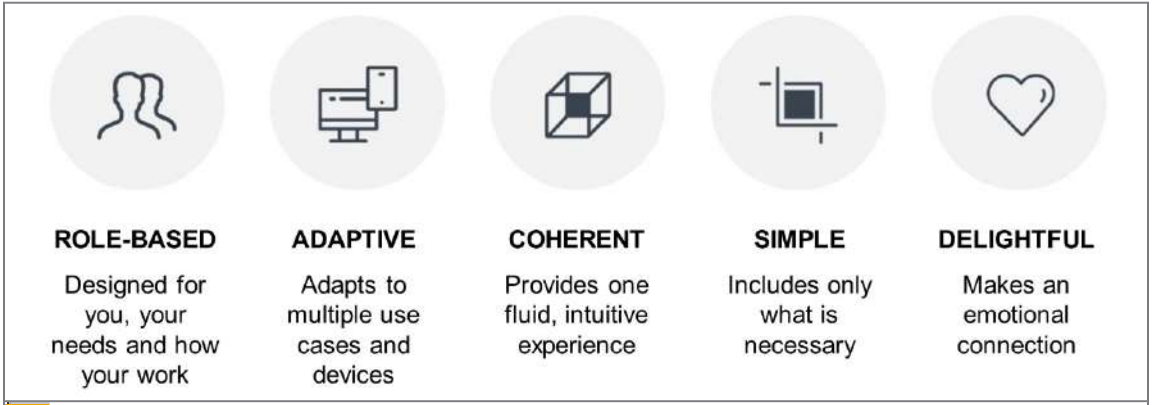
SAP Fiori is the design language that enables user experiences in enterprise applications. At the time of SAPPHIRE 2013, the first SAP Fiori apps were released for managers and employees with request and approval functions. Since then, the number of apps has grown massively. With the 2016 release of SAP S/4HANA, SAP Fiori 2.0 was introduced. Today, SAP Fiori 3 represents the current target design, which further develops the SAP Fiori design language for all SAP products to fully support the Intelligent Suite.

In a nutshell, SAP Fiori defines a simple and role-based user experience (UX) that simplifies the way people work with SAP products.

The underlying technology, which is required for the design language, is SAPUI5 as UI technology.

Design Principles

The underlying design philosophy of SAP Fiori is based on five core principles:



ROLE-BASED
Designed for you, your needs and how your work

ADAPTIVE
Adapts to multiple use cases and devices

COHERENT
Provides one fluid, intuitive experience

SIMPLE
Includes only what is necessary

DELIGHTFUL
Makes an emotional connection

Figure 30: Design Principles

- ROLE-BASED

SAP Fiori is designed for your business, your needs, and the way you work. It is based on SAP's insights into the multiple roles of today's workforce. Providing the right information at the right time, SAP Fiori reflects the way people actually work.

- ADAPTIVE

  With SAP Fiori, you can work how and where you want, regardless of the device you're using. And it delivers relevant information that provides instant insight.

- COHERENT

  No matter whether you're running a sales order, checking your latest KPIs, or managing leave requests, SAP Fiori follows a consistent interaction and visual design language. Across the enterprise, you'll enjoy the same intuitive and consistent experience.

- SIMPLE

  SAP Fiori enables you to get your work done intuitively and quickly. With SAP Fiori, you can focus on what matters — essential functions are easy to use, and you can personalize the experience to focus on your relevant tasks and activities.

- DELIGHTFUL

  Beyond enabling you to work smarter, SAP Fiori also enriches your work experience by allowing you to easily get your work done.

Animation:

For more information on , please view the animation in the lesson *UI: SAP Fiori*, in your online course.

Benefits of SAP Fiori

For businesses, the key benefits of using SAP Fiori are as follows:

- Simple and user-friendly operability
- Increased user satisfaction through modern user interfaces
- Reduced costs for in-house developments to optimize the user interface
- Optimized display on the desktop and most popular mobile devices
- Role-based permissions so that each user only sees the applications relevant to their area of responsibility

Summary

You are now familiar with the key principles and benefits of SAP Fiori. Using SAP Fiori reduces up to 80% of costs in the front end development process and ensures that all products and applications have a simple and consistent end-to-end user experience for their respective business users.

Further Reading

These resources might be helpful if you want to dive deeper into SAP Fiori.

- SAP Fiori Product Page
- SAP Fiori @ SAP

- SAP Fiori Guidelines
- SAP Fiori Foundations ILT-Course

LESSON SUMMARY

You should now be able to:

- List the design principles and key benefits of SAP Fiori

Unit 3
Lesson 3

UI: Elements versus SAP Fiori

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Evaluate the difference between SAP Fiori and SAP Fiori Elements

SAP Fiori Elements

Usage Scenario

Your company is planning to develop a set of cloud-based applications. These applications should be accessible on all device types such as mobile phones, tablets, and laptops. The look and feel of these applications should be similar on these devices to reduce learning time for end-users. Therefore, you need an efficient way to design the UI of these applications. SAP Fiori Elements provides multiple generic templates to do so.

Learning objectives

- Explain the advantages of using SAP Fiori Elements.
- Evaluate when to use SAP Fiori Elements instead of SAPUI5 Freestyle.
- Describe different page types.

Advantages of Using SAP Fiori Elements

SAP Fiori itself represents the design philosophy behind SAP applications. This philosophy defines the core principles that should drive modern user interfaces. Based on these principles, SAP Fiori Elements provides a set of several common page types — comparable to several templates — that give developers a head start in developing applications that connect to data in SAP back-end systems. SAP Fiori builds on SAPUI5, SAP's HTML5 development toolkit.

The main idea behind SAP Fiori elements is to generate an SAP Fiori app at runtime from an existing OData service. OData is the industry standard for exchanging business data via RESTful HTTPS APIs.

Each OData service comes with a metadata document that describes the service. The metadata document is also extendable by developers. SAP entities exposing the OData services can be enriched with additional metadata, called “annotations” to define new attributes and relationships of the data. Based on these annotations, SAP Fiori elements will dynamically generate the app at runtime.

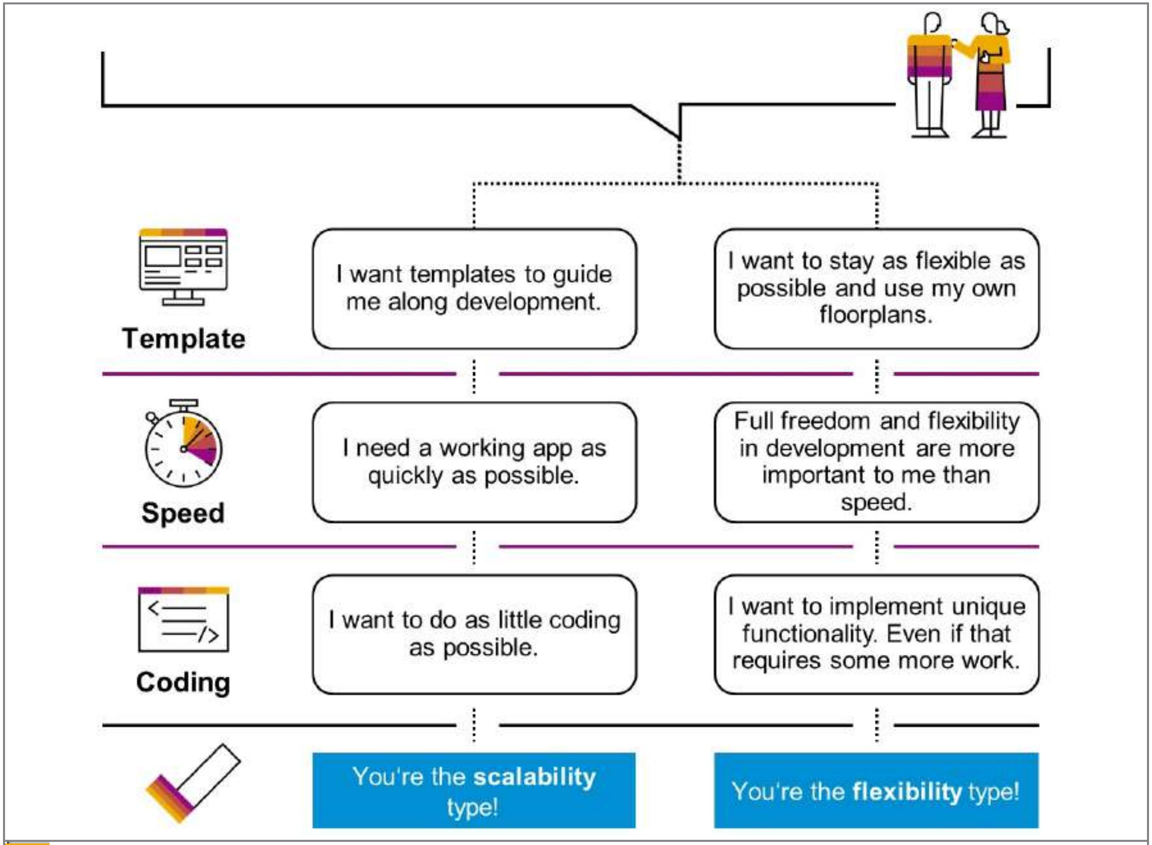
In addition to the content of the pages, SAP Fiori elements manages the logic and behavior of the application. For example, there is no longer the need to write UI code to manage navigation between pages or apps, to apply a filter to the content of a table, or to edit and save an object. This means that standard apps require very little UI development, and occasionally no additional UI code at all.

When your app differs slightly from the standard page specification, it is possible to use annotations to achieve exactly the look or functionality you want. If you want a fully custom app, sometimes called a freestyle app, using your own designs, layouts or workflows, you can do this with SAPUI5, but not with SAP Fiori elements.

When to Use SAP Fiori Elements

When it comes to decision-making, you or your team can choose between two options:

1. Develop the application with SAP Fiori Elements.
2. Start with SAP UI5 freestyle development.



Template

I want templates to guide me along development.

I want to stay as flexible as possible and use my own floorplans.

Speed

I need a working app as quickly as possible.

Full freedom and flexibility in development are more important to me than speed.

Coding

I want to do as little coding as possible.

I want to implement unique functionality. Even if that requires some more work.

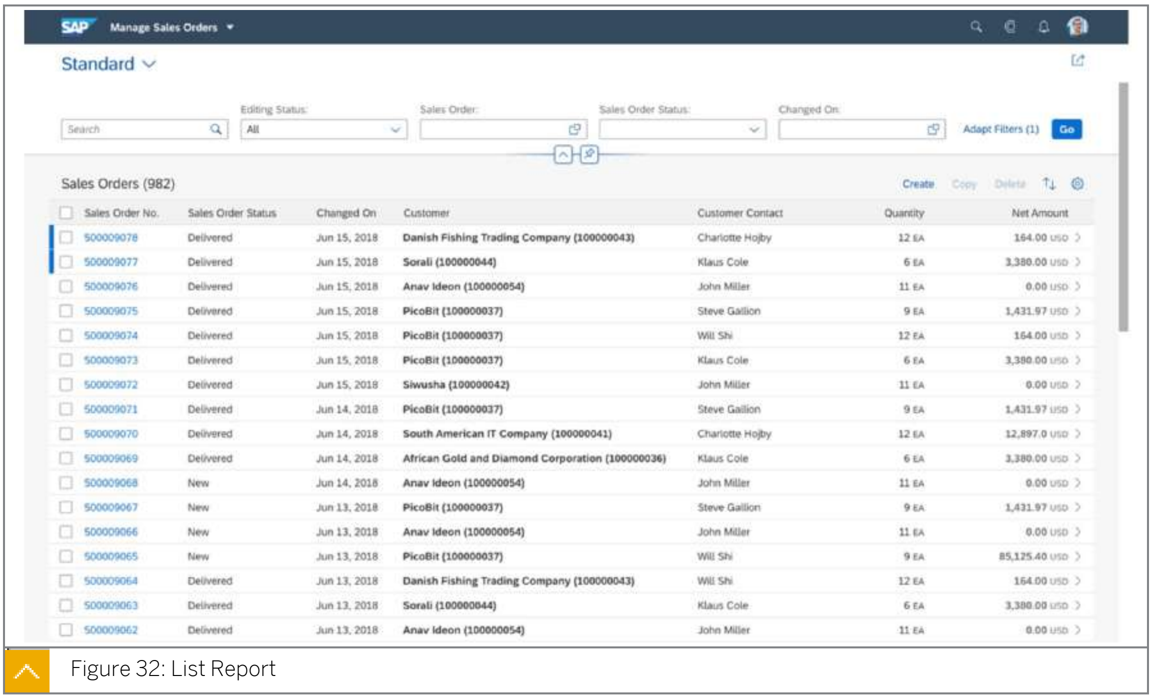
You're the **scalability** type!

You're the **flexibility** type!

Figure 31: When to Use SAP Fiori Elements

In case your business requires a working app as quickly as possible, and you want to do as little coding as possible, SAP Fiori elements might be the right choice for you. The different template wizards will guide you through the creation of the app. During the creation, you have to decide with which page type you want to work with. SAP Fiori elements offers the following page types or floorplans:

List Report: to work with a large set of items.



Editing Status: All

Sales Order:

Sales Order Status:

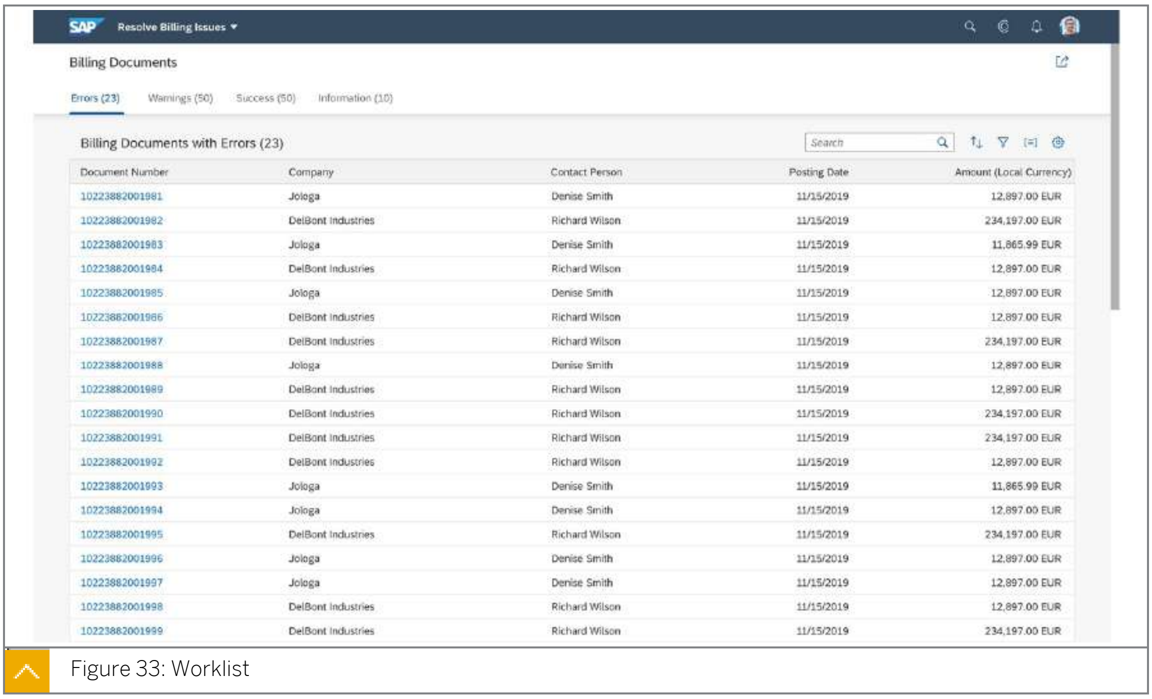
Changed On:

Sales Orders (982)

□	Sales Order No.	Sales Order Status	Changed On	Customer	Customer Contact	Quantity	Net Amount
□	500009078	Delivered	Jun 15, 2018	**Danish Fishing Trading Company (100000043)**	Charlotte Hojby	12 EA	164.00 USD
□	500009077	Delivered	Jun 15, 2018	**Sorali (100000044)**	Klaus Cole	6 EA	3,380.00 USD
□	500009076	Delivered	Jun 15, 2018	**Anav Ideon (100000054)**	John Miller	11 EA	0.00 USD
□	500009075	Delivered	Jun 15, 2018	**PicoBit (100000037)**	Steve Gallion	9 EA	1,431.97 USD
□	500009074	Delivered	Jun 15, 2018	**PicoBit (100000037)**	Will Shi	12 EA	164.00 USD
□	500009073	Delivered	Jun 15, 2018	**PicoBit (100000037)**	Klaus Cole	6 EA	3,380.00 USD
□	500009072	Delivered	Jun 15, 2018	**Siwusha (100000042)**	John Miller	11 EA	0.00 USD
□	500009071	Delivered	Jun 14, 2018	**PicoBit (100000037)**	Steve Gallion	9 EA	1,431.97 USD
□	500009070	Delivered	Jun 14, 2018	**South American IT Company (100000041)**	Charlotte Hojby	12 EA	12,897.0 USD
□	500009069	Delivered	Jun 14, 2018	**African Gold and Diamond Corporation (100000036)**	Klaus Cole	6 EA	3,380.00 USD
□	500009068	New	Jun 14, 2018	**Anav Ideon (100000054)**	John Miller	11 EA	0.00 USD
□	500009067	New	Jun 13, 2018	**PicoBit (100000037)**	Steve Gallion	9 EA	1,431.97 USD
□	500009066	New	Jun 13, 2018	**Anav Ideon (100000054)**	John Miller	11 EA	0.00 USD
□	500009065	New	Jun 13, 2018	**PicoBit (100000037)**	Will Shi	9 EA	85,125.40 USD
□	500009064	Delivered	Jun 13, 2018	**Danish Fishing Trading Company (100000043)**	Will Shi	12 EA	164.00 USD
□	500009063	Delivered	Jun 13, 2018	**Sorali (100000044)**	Klaus Cole	6 EA	3,380.00 USD
□	500009062	Delivered	Jun 13, 2018	**Anav Ideon (100000054)**	John Miller	11 EA	0.00 USD

Figure 32: List Report

Worklist: to take action on work items.



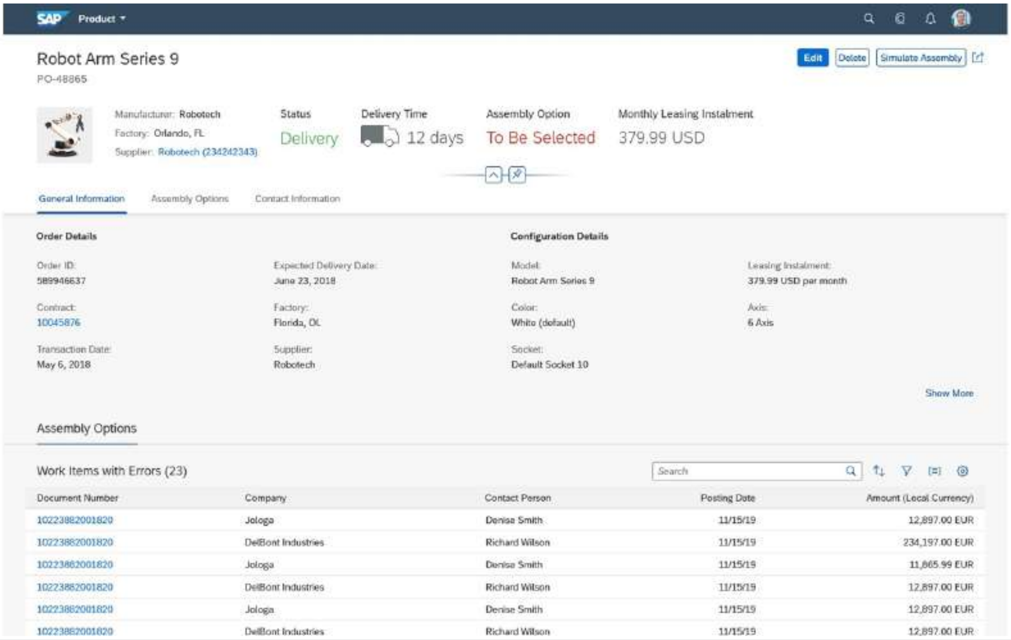
Billing Documents

Billing Documents with Errors (23)

Document Number	Company	Contact Person	Posting Date	Amount (Local Currency)
10223882001981	Jologa	Denise Smith	11/15/2019	12,897.00 EUR
10223882001982	DelBont Industries	Richard Wilson	11/15/2019	234,197.00 EUR
10223882001983	Jologa	Denise Smith	11/15/2019	11,865.99 EUR
10223882001984	DelBont Industries	Richard Wilson	11/15/2019	12,897.00 EUR
10223882001985	Jologa	Denise Smith	11/15/2019	12,897.00 EUR
10223882001986	DelBont Industries	Richard Wilson	11/15/2019	12,897.00 EUR
10223882001987	DelBont Industries	Richard Wilson	11/15/2019	234,197.00 EUR
10223882001988	Jologa	Denise Smith	11/15/2019	12,897.00 EUR
10223882001989	DelBont Industries	Richard Wilson	11/15/2019	12,897.00 EUR
10223882001990	DelBont Industries	Richard Wilson	11/15/2019	234,197.00 EUR
10223882001991	DelBont Industries	Richard Wilson	11/15/2019	234,197.00 EUR
10223882001992	DelBont Industries	Richard Wilson	11/15/2019	12,897.00 EUR
10223882001993	Jologa	Denise Smith	11/15/2019	11,865.99 EUR
10223882001994	Jologa	Denise Smith	11/15/2019	12,897.00 EUR
10223882001995	DelBont Industries	Richard Wilson	11/15/2019	234,197.00 EUR
10223882001996	Jologa	Denise Smith	11/15/2019	12,897.00 EUR
10223882001997	Jologa	Denise Smith	11/15/2019	12,897.00 EUR
10223882001998	DelBont Industries	Richard Wilson	11/15/2019	12,897.00 EUR
10223882001999	DelBont Industries	Richard Wilson	11/15/2019	234,197.00 EUR

Figure 33: Worklist

Object page: to display details about an object.



Robot Arm Series 9

PO-48865

Manufacturer: Robotech
Factory: Orlando, FL
Supplier: Robotech (234242343)

Status: Delivery

Delivery Time: 12 days

Assembly Option: To Be Selected

Monthly Leasing Instalment: 379.99 USD

Order Details

Order ID: 589946637

Expected Delivery Date: June 23, 2018

Contract: 10045876

Factory: Florida, OL

Transaction Date: May 6, 2018

Supplier: Robotech

Configuration Details

Model: Robot Arm Series 9

Leasing Instalment: 379.99 USD per month

Color: White (default)

Axis: 6 Axis

Socket: Default Socket 10

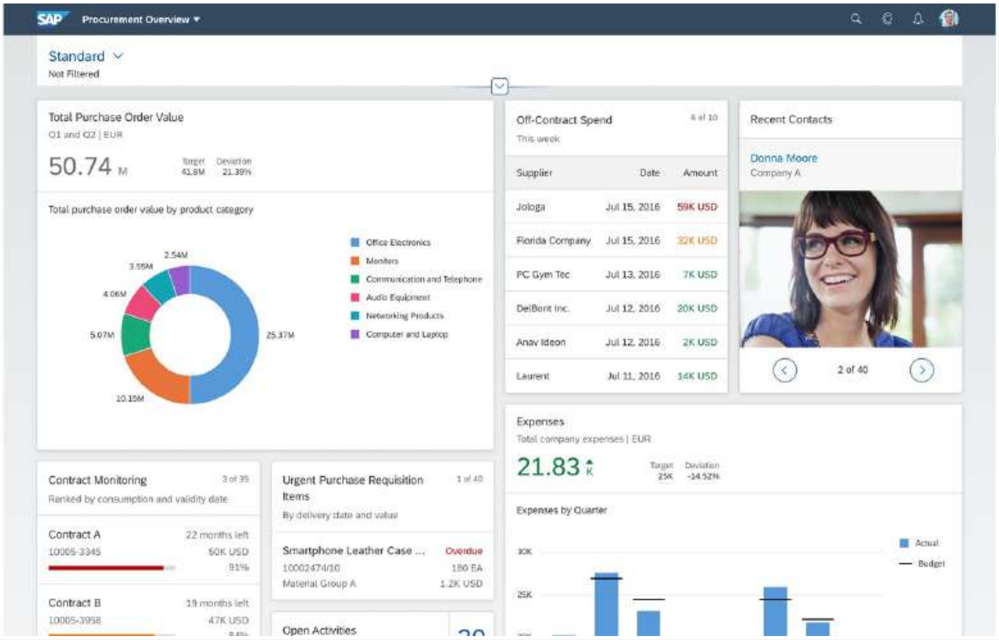
Assembly Options

Work Items with Errors (23)

Document Number	Company	Contact Person	Posting Date	Amount (Local Currency)
10223882001820	Jologa	Denise Smith	11/15/19	12,897.00 EUR
10223882001820	DelBont Industries	Richard Wilson	11/15/19	234,197.00 EUR
10223882001820	Jologa	Denise Smith	11/15/19	11,865.99 EUR
10223882001820	DelBont Industries	Richard Wilson	11/15/19	12,897.00 EUR
10223882001820	Jologa	Denise Smith	11/15/19	12,897.00 EUR
10223882001820	DelBont Industries	Richard Wilson	11/15/19	12,897.00 EUR

Figure 34: Object Page

Overview page: to provide an entry-level view of content.



Total Purchase Order Value
Q1 and Q2 | EUR
50.74 M
Target 41.8M
Deviation 21.39%

Off-Contract Spend
6 of 10
This week

Supplier	Date	Amount
Jologa	Jul 15, 2016	59K USD
Florida Company	Jul 15, 2016	32K USD
PC Gym Tec	Jul 13, 2016	7K USD
DelBont Inc.	Jul 12, 2016	20K USD
Anav Ideon	Jul 12, 2016	2K USD
Laurent	Jul 11, 2016	14K USD

Recent Contacts
Donna Moore
Company A

Expenses
Total company expenses | EUR
21.83 K
Target 25K
Deviation -14.52%

Contract Monitoring
3 of 35
Ranked by consumption and validity date

Contract A
10005-3345
22 months left
50K USD
91%

Contract B
10005-3958
19 months left
47K USD

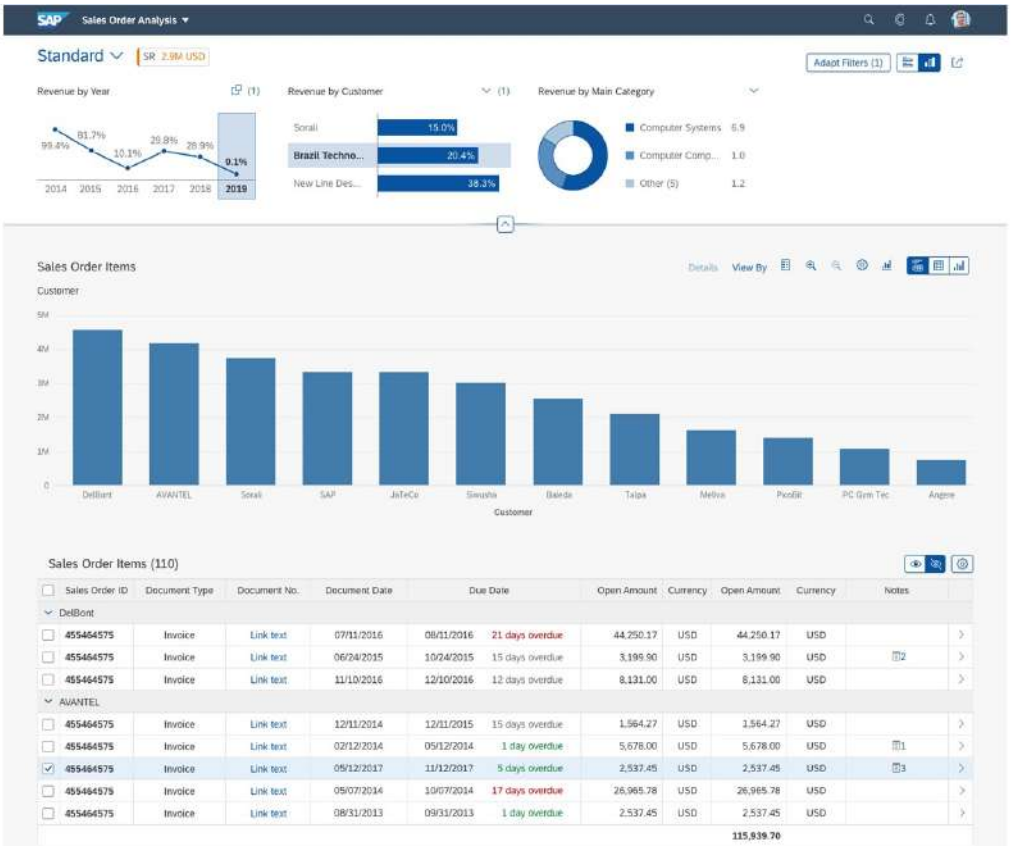
Urgent Purchase Requisition Items
1 of 40
By delivery date and value

Smartphone Leather Case ...
Overdue
10002474/10
180 EA
Material Group A
1.2K USD

Open Activities

Figure 35: Overview Page

Analytical list page: provide multiple angles of data for analyses.



Sales Order Items (110)

□	Sales Order ID	Document Type	Document No.	Document Date	Due Date		Open Amount	Currency	Open Amount	Currency	Notes	
DelBont												
□	455464575	Invoice	Link text	07/11/2016	08/11/2016	21 days overdue	44,250.17	USD	44,250.17	USD		>
□	455464575	Invoice	Link text	06/24/2015	10/24/2015	15 days overdue	3,199.90	USD	3,199.90	USD	2	>
□	455464575	Invoice	Link text	11/10/2016	12/10/2016	12 days overdue	8,131.00	USD	8,131.00	USD		>
AVANTEL												
□	455464575	Invoice	Link text	12/11/2014	12/11/2015	15 days overdue	1,564.27	USD	1,564.27	USD		>
□	455464575	Invoice	Link text	02/12/2014	05/12/2014	1 day overdue	5,678.00	USD	5,678.00	USD	1	>
☑	455464575	Invoice	Link text	05/12/2017	11/12/2017	5 days overdue	2,537.45	USD	2,537.45	USD	3	>
□	455464575	Invoice	Link text	05/07/2014	10/07/2014	17 days overdue	26,965.78	USD	26,965.78	USD		>
□	455464575	Invoice	Link text	08/31/2013	09/31/2013	1 day overdue	2,537.45	USD	2,537.45	USD		>
									115,939.70			

Figure 36: Analytical List Page

Summary

You are now familiar with the fundamentals of SAP Fiori elements. You understand when to use SAP Fiori elements and when to start a SAPUI5 freestyle development project. If you want to dive deeper into this topic, please consider the further reading section.

Further Reading

- SAP Fiori Product Page
- SAP Fiori elements usage guide
- SAP Fiori @ SAP
- SAP Fiori Guidelines
- SAP Fiori Elements ILT-Course

LESSON SUMMARY

You should now be able to:

- Evaluate the difference between SAP Fiori and SAP Fiori Elements

Unit 3
Lesson 4

Add Custom Business Logic

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Add custom business logic

Add Custom Business Logic to Your Application: Exercise Overview

Scenario

In this exercise, you will add custom code to the CAP service to implement conditional formatting of certain cells of your work list. Depending on the value of the properties `impact` and `criticality` respectively, cells in the table of our work list page will change color.

Task Flow

In this exercise, you will perform the following task:

- Create a JavaScript file to implement dynamic color coding

Prerequisites

- Generation of a UI channel
- Familiarity with JavaScript coding

Add Custom Code

1. In the project, go to folder *srv*, representing the service, and select *New File* in the context menu.

2. Enter **`risk-service.js`** as a name.

3. Select the new file in the explorer, an editor opens.

4. Enter the following lines into the editor:

```
// Imports
 const cds = require("@sap/cds");

 /**
    * The service implementation with all service handlers
    */
 module.exports = cds.service.impl(async function () {
    // Define constants for the Risk and BusinessPartners entities
from the risk-service.cds file
    const { Risks, BusinessPartners } = this.entities;

    /**
    * Set criticality after a READ operation on /risks
```

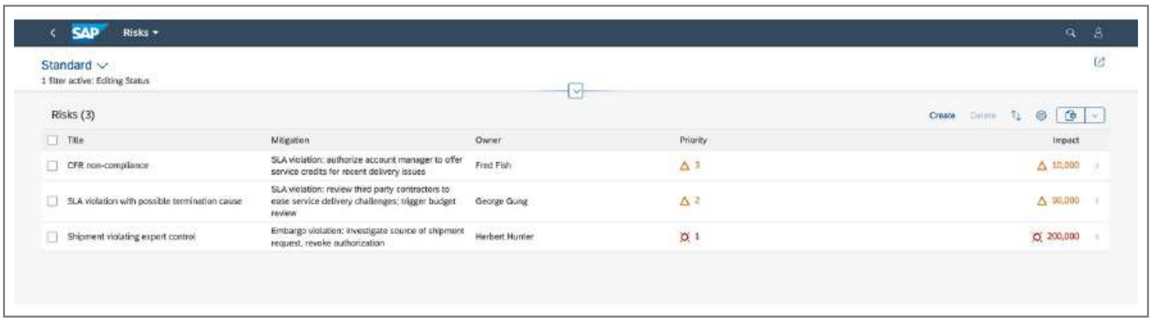
```
   */
    this.after("READ", Risks, (data) => {
       const risks = Array.isArray(data) ? data : [data];

       risks.forEach((risk) => {
         if (risk.impact >= 100000) {
           risk.criticality = 1;
         } else {
           risk.criticality = 2;
         }
      });
   });
});
```

> Note:
> The defined constant for the *BusinessPartners* will be used in a later step.

5. Save the file.

6. In the browser, reload the page of the SAP Fiori Elements app.

   It now shows our work list with the columns `Priority` and `Impact` with color and an icon, depending on the amount in `impact`.



Standard
1 filter active: Editing Status

Risks (3)

Title	Mitigation	Owner	Priority	Impact
CFR non-compliance	SLA violation: authorize account manager to offer service credits for recent delivery issues	Fred Fish	△ 3	△ 10,000
SLA violation with possible termination cause	SLA violation: review third party contractors to ease service delivery challenges; trigger budget review	George Gung	△ 2	△ 90,000
Shipment violating export control	Embargo violation: investigate source of shipment request, revoke authorization	Herbert Hunter	☒ 1	☒ 200,000

Explanation of the Custom Code

Because your file is called `risks-service.js` and shares the same name as your service definition file `risks-service.cds`, CAP automatically treats it as a handler file for the service defined in there. CAP exposes several events [1] and you can easily write handlers like the one above. If you want to use a different name for the JavaScript file, you have to use @(impl:...) notation [2].

The event `after` is triggered after a `READ` is carried out for our `Risks` entity. In your custom handler, you get all the data, in this case all the risks that were read according to the query. You can loop over each of them and, if needed, adjust the data of the response. In this case, the code changes `criticality` value when the `impact` is larger than 100000. The new values for `criticality` are then part of the response to the read request.

How is this change reflected in the UI? To achieve this, you have to go back to the annotations you created in Generate the UI Channel , where you find your `app/risks/annotations.cds` file. There you had the two columns `prio` and `impact` annotated with an additional `Criticality` annotation. This annotation points to the `criticality` property of your service.

Note:
Criticality with an upper case C is the annotation, while the property name criticality could also be called different opposed to the annotation.

As you now set different values in your custom handler for criticality, the SAP Fiori elements application translates these into icons and colors, which you can see in the UI.

```
annotate RiskService.Risks with @(UI : {
    ...
    ...
    LineItem : [
       ...
       ...
       {
          Value : prio,
          Criticality : criticality
       },
       {
          Value : impact,
          Criticality : criticality
       },
    ],
 });
```

You can find out more about the possible values of the Criticality annotation here[3]. This, however, is just one of the many sections of the OData Annotation vocabularies for UI[4] and Common[5] usage.

Summary

You have added custom business logic to your application. Next you will add an external service to finally deploy your application manually. Keep your application running in your web browser, you will need it.

Unit 3
Exercise 3

Add Custom Business Logic to your Application

Simulation: Add Custom Business Logic to your Application

For more information on *Add Custom Business Logic to your Application*, please view the simulation in the lesson *Add Custom Business Logic* in your online course.

1.

Unit 3
Solution 3

Add Custom Business Logic to your Application

Simulation: Add Custom Business Logic to your Application

For more information on *Add Custom Business Logic to your Application*, please view the simulation in the lesson *Add Custom Business Logic* in your online course.

1.

Reference Links

For your convenience, this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 4: Reference Links Business Logic

Ref#	Section	Context text fragment	Brief description	Link
1	Explanation of the Custom Code	CAP exposes several events	Handler registration API	https://cap.cloud.sap/docs/node.js/services#event-handlers
2	Explanation of the Custom Code	you have to use @(impl:...) notation.	Providing Custom Implementations	https://cap.cloud.sap/docs/guides/providing-services#service-impls
3	Explanation of the Custom Code	possible values of the Criticality annotation	Criticality types	https://github.com/SAP/odata-vocabularies/blob/main/vocabularies/UI.md#CriticalityType
4	Explanation of the Custom Code	OData Annotation vocabularies for UI and Common	UI vocabulary	https://github.com/SAP/odata-vocabularies/blob/main/vocabularies/UI.md
5	Explanation of the Custom Code	OData Annotation vocabularies for UI and Common	Common vocabulary	https://github.com/SAP/odata-vocabularies/blob/main/vocabularies/Common.md

LESSON SUMMARY

You should now be able to:

- Add custom business logic

Unit 3
Lesson 5

Event Handling

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Evaluate when to use event handlers

Event Handling - CAP Service SDK for Node.js

Usage Scenario

Your company is planning to build an extension application using the SAP Cloud Application Programming Model (CAP). The generic service handlers that the framework provides for standard CRUD operations (`CREATE`, `READ`, `UPDATE`, `DELETE`) do not fully satisfy the application's requirements. You want to implement custom business logic on top of the standard functionality. For that, you need to understand the concept of event handlers in CAP.

Learning objectives

- Explain the concept of event handlers in CAP.
- Explain when to use event handlers.

Event Handlers in CAP

Watch this video to learn about event handlers in CAP.

Animation: Event Handlers in CAP

For more information on *Event Handlers in CAP*, please view the animation in the lesson *Event Handling*, in your online course.

In CAP everything that happens at runtime is an event that is sent to a service. They are a powerful means to extend CAP. An event handler is simply a method, that is executed when something happens in the application.

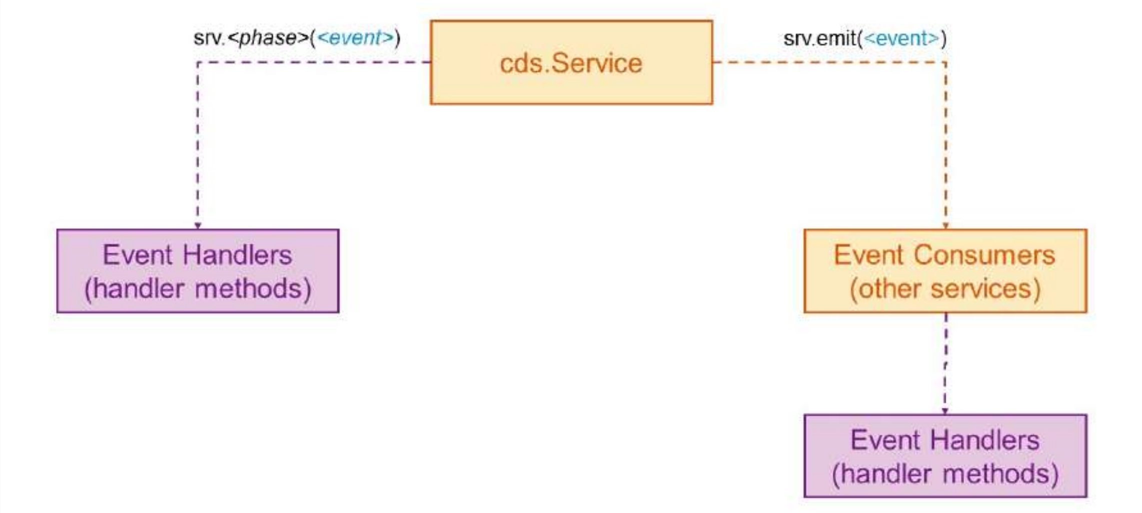

srv.<phase>(<event>)
cds.Service
srv.emit(<event>)
Event Handlers (handler methods)
Event Consumers (other services)
Event Handlers (handler methods)

Figure 39: Event Handlers in CAP

If you need a specific service to react to a specific event, you register an event handler using `srv.<phase>(<event>)`, where

- `srv` is the instance of the service that you are extending,
- `<phase>` is one of `on`, `before`, or `after` (see section Event Phases) and
- `<event>` is any kind of named event as a string (for example `'READ'`).

Once a service has an event handler for a specific event, it becomes a consumer for that event. Using `srv.emit(<event>)`, a service can send arbitrary events. These events then get consumed by other services that have event handlers registered for the respective event.

An Example

Think of a simple application, that manages your company's IT inventory. For each asset category, there is an entity, for example `Notebooks`, `Phones`, `Tablets`. The entities are exposed via an `inventory` service. The service provides an OData API that enables you to interact with the entities. Let's say you want to see the current inventory of notebooks. You perform a `GET` request to the `inventory/Notebooks` service. Within CAP, a `READ` event is triggered for the `Notebooks` entity. There is a built-in event handler (also known as *Generic Provider*) that retrieves the requested data from the database `on` a `READ` event for the `Notebooks` entity.

Making use of Event Handlers

CAP handles all CRUD events (`CREATE`, `READ`, `UPDATE`, `DELETE`) out-of-the-box. You do not need to take any further steps after defining your entities and services. But often times, the standard functionality does not fulfill all of your requirements. You want to implement custom logic. In these cases, you can make use of the handler registration API of the CAP Service SDK for Node.js.

Extending the Example

You are building a UI on top of your inventory service. It should display when a device is eligible for replacement. Whether a device is eligible for replacement might differ depending on device type, date of acquisition, or country - thus there is no easy answer. Your service needs to uncover it. Within your entities there is a boolean field `eligible_for_replacement`, which is set to `false` by default. Now whenever there is a `READ` event for any of the entities in your `inventory` service, `after` the entities have been read, you want to have a custom event handler finding out, whether the individual devices are eligible for replacement or not. It could look like the following:

```
cds.serve('inventory-service') .with (function(){
  this.after('READ', '*', (devices)=>{
    for (let each of devices) {
      var deviceAge = calculateDeviceAgeYears(each)
      if (deviceAge >= 4) {
        each.eligible_for_replacement = true
      } else {
        each.eligible_for_replacement = false
      }
    }
  })
})
```

The code defines that `after` each `READ` of any (*) entity in your `inventory` service, the eligibility for replacement should be calculated. The method loops through each line of the data that was fetched by the generic service handler. Note that instead of registering the handler method for any entity (*), you could also register the event handler for a specific entity, for example `Notebooks`.

Now in this simple example, eligibility for replacement only depends on the age of the device, but you can grasp that almost anything is possible here.

Event Phases

Events are processed in three phases that are executed consecutively: `Before`, `On`, and `After`. When registering an event handler, the phase in which the event handler should be called, needs to be specified. In the previous example the handler method was specified for the `After` phase. It is possible to register multiple event handlers for each event phase.

Summary

You can now explain the concept of event handlers in CAP. You are able to evaluate, whether a custom event handler is required and you know the basics for implementation.

Further Reading

- CAP Service SDK for Node.js

LESSON SUMMARY

You should now be able to:

- Evaluate when to use event handlers

Unit 3
Lesson 6

Error Handling

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Explain error handling

Error Handling - CAP Service SDK for Node.js

Usage Scenario

Having good error handling is key to ensuring the robustness, correctness, and performance of the given application. Building robust applications requires you to know how to throw and handle exceptions which occur during the runtime of the application. Thus, you will be introduced to the basic concepts of exception handling in Node.js as well as specific techniques for the CAP Service SDK for Node.js.

Learning objectives

- Explain why error handling is important.
- Describe how to raise new exceptions.
- Explain how to catch exceptions.
- Register event handlers for error handling.

Error Handling

Error Types

Proper error handling is crucial for today's business applications. Before going into more detail, it is necessary to distinguish between two types of errors:

Programmer Errors

These occur as a result of programming errors (for example, `foo` cannot be read by undefined). They must be corrected.

Operational Errors

These occur during runtime (for example, when sending a request to a faulty remote system). They must be corrected.

> Animation:
>
> For more information on , please view the animation in the lesson *Error Handling*, in your online course.

Guidelines

“Let it crash” is a philosophy taken from the Erlang programming language (Joe Armstrong), which is also (partially) applicable to Node.js.

The key takeaways for programming errors are:

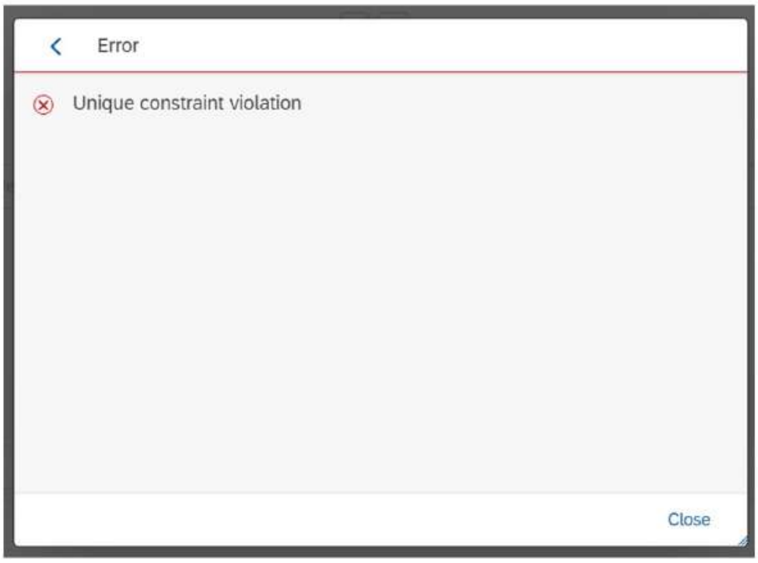
- Fail loudly: Do not hide errors and continue silently. Ensure to log unexpected errors correctly. Don't catch errors you can't handle.
- Don't develop in a defensive fashion. Focus on your business logic and only handle errors when you know they will occur. Use `try/catch` blocks only when necessary.

Never try to catch and handle unexpected errors, rejections of promises, and so on. If it is unexpected, you cannot handle it correctly. If you could, it would be expected (and should already be handled). Even if your apps should be stateless, you can never be 100% sure that a shared resource was not affected by the unexpected error. Therefore, you should never allow an app to continue running after such an event, especially for multi-tenant apps where there is a risk of information disclosure.

Following these guidelines will make your code shorter, clearer and simpler.

Never Hide the Causes of Errors

When an error occurs, it should be possible to know the root cause. The CAP SDK for Node.js also throws exceptions, for example when a CRUD operation violates the foreign key constraints. In this case, the framework throws the exception UNIQUE_CONSTRAINT_VIOLATION. The problem in this case is that the end user will only see a more or less cryptic error message:



Unique constraint violation

Figure 40: An Exception

It is therefore useful to provide a meaningful error message.

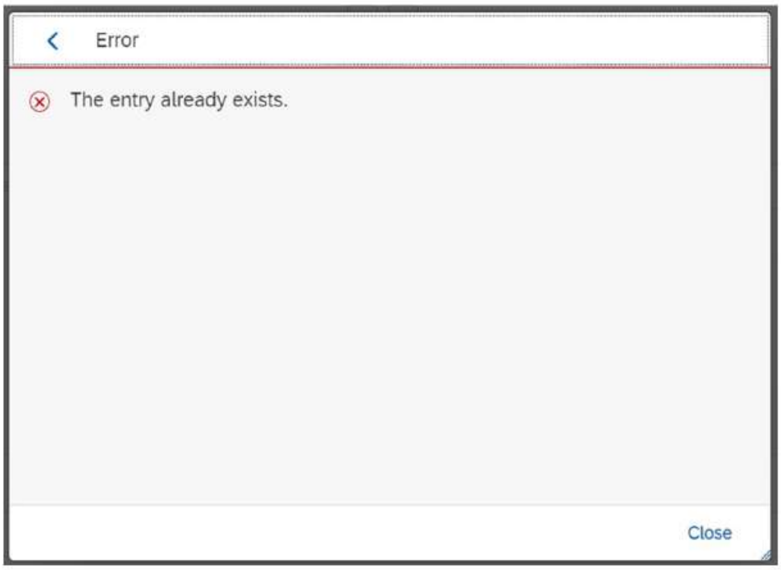
For this purpose, you can register an error handler in your service implementation:

Example:

```
// Imports
constcds=require("@sap/cds");

/**
   * The service implementation with all service handlers
   */
```

```
module.exports=cds.service.impl(asyncfunction(){
  /**
   * Custom error handler
   *
   * throw a new error with: throw new Error('something bad happened');
   *
   **/
  this.on("error",(err,req)=>{
    switch(err.message){
      case"UNIQUE_CONSTRAINT_VIOLATION":
        err.message="The entry already exists.";
        break;

      default:
        err.message=
          "An error occured. Please retry. Technical error message: "+
          err.message;
      break;
    }
  });
});
```

This handler now steps in whenever this exception gets triggered and overrides it with an alternative error message:



The entry already exists.

Figure 41: Meaningful Error Message

Raising and Catching Exceptions

You will certainly add your implementations to your services. It is very likely, that you want to interrupt some operations before something crashes. In this case, you can throw a Node.js exception. Our recommendation is to look at the Node.js documentation for error handling.

Request Response

You can also use the `req.error()` method to collect messages or errors and return them to the caller in the request-response. Read more here .

```
this.on("submitOrder",async(req)=>{
  const{ book, amount }=req.data;
  let{ stock }=awaitdb.read(Books,book,(b)=>b.stock);
  if(stock >= amount){
    awaitdb.update(Books,book).with({stock: (stock-=amount)});
    awaitthis.emit("OrderedBook",{ book, amount,buyer: req.user.id});
    returnreq.reply({ stock });// <-- Normal reply
  }else{
    // Reply with error code 409 and a custom error message
    returnreq.error(409,`${amount} exceeds stock for book #${book}`);
  }
});
```

Summary

The core error handling concepts in the CAP SDK for Node.js are now familiar to you. We strongly recommend incorporating these concepts to ensure the overall robustness of your CAP application.

LESSON SUMMARY

You should now be able to:

- Explain error handling

Learning Assessment

1. You are asked to add a User Interface to your CAP project as quickly as possible and without any unique functionalities. Which option do you choose?

   Choose the correct answer.

   □ A SAPUI5 Freestyle

   □ B SAP Fiori Elements

2. What is a reason to add UI annotations to your project?

   Choose the correct answer.

   □ A To add tooltips to the header fields of tables.

   □ B To allow users to annotate table cells.

   □ C To display columns and form fields.

3. Which of the following are standard floorplans of SAP Fiori Elements?

   Choose the correct answers.

   □ A Blueprint

   □ B List report

   □ C Dashboard

   □ D Worklist

4. What is SAP Fiori?

   Choose the correct answer.

   □ A A collection of design guidelines.

   □ B An SDK to develop multi-target applications.

   □ C A design language.

5. What are some SAP Fiori technology components?

   Choose the correct answers.

   □ A HTML5

   □ B SAPUI5

   □ C Analytics

   □ D React

6. What is the main idea behind SAP Fiori elements?

   Choose the correct answer.

   □ A Provide a showcase for the core principles of modern user interfaces (UI).

   □ B Generate SAP Fiori apps at runtime from an existing OData service.

   □ C Define a role-based user experience (UX).

   □ D Provide a framework and development tool kit for HTML 5.

7. What is SAP Fiori elements designed to do?

   Choose the correct answers.

   □ A Speed up development.

   □ B Drive UX consistency.

   □ C Drive full flexibility of development.

   □ D Implement unique functionality.

8. Which OData versions support SAP Fiori elements?

   Choose the correct answers.

   □ A V1

   □ B V2

   □ C V3

   □ D V4

9. Why does CAP set the file `risks-service.js` automatically as a handler file for the respective service?

Choose the correct answer.

□ A Because the filename and the service definition filename are the same.

□ B Because SAP Business Application Studio added a respective annotation.

10. Which criticality value is assigned to `Negative` criticality?

Choose the correct answer.

□ A 1

□ B 2

□ C 3

11. Which pattern do you use to register an event handler?

Choose the correct answer.

□ A event.()

□ B phase.()

□ C srv.()

12. In CAP, which keyword is used to send events?

Choose the correct answer.

□ A throw

□ B emit

□ C actions

□ D stream

13. How many event handlers can you register for one event phase?

Choose the correct answer.

□ A Multiple

□ B Exactly one

□ C Exactly three

14. What can you do to provide meaningful error messages to users in your CAP application?

    Choose the correct answer.

    □ A Catch exceptions.

    □ B Register an error handler.

    □ C Hide the cause of errors.

    □ D Register an event handler.

15. In *Node.js*, which statement do you use to create an exception?

    Choose the correct answer.

    □ A throw

    □ B try

    □ C catch

UNIT 4 External Services

Lesson 1

Lesson 2

Lesson 3

UNIT OBJECTIVES

- Add and consume an external service
- Describe how SAP BTP applications can securely access remote services
- Manage connections between backend systems and SAP BTP

Unit 4
Lesson 1

Add an External Service

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Add and consume an external service

Add an External Service to Your CAP Service: Exercise Overview

Scenario

In this exercise, you will extend your CAP service with the consumption of an external Business Partner service.

Task Flow

In this exercise, you will perform the following tasks:

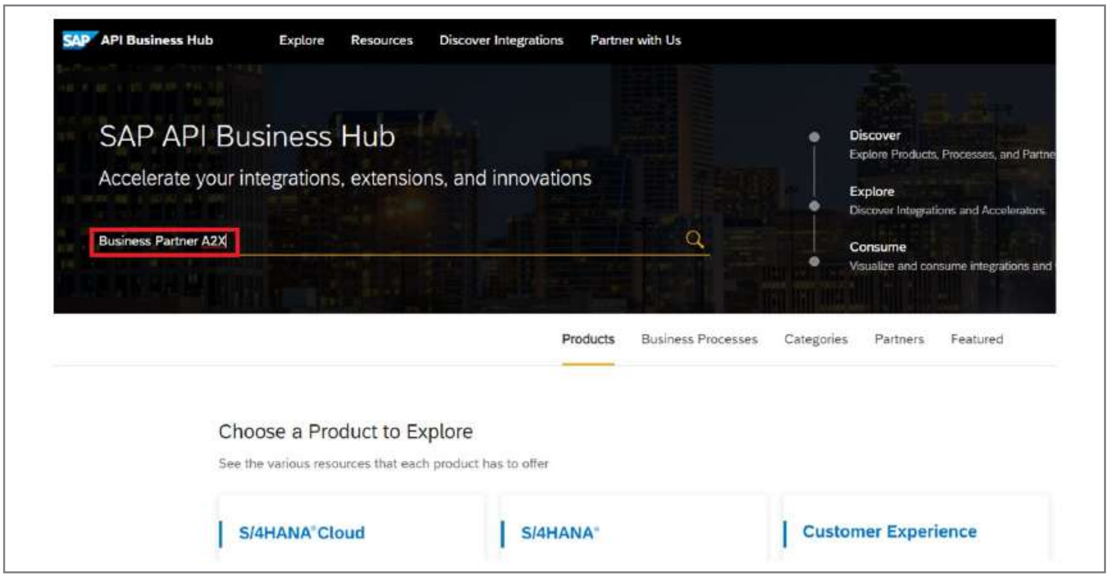
- Search for a service on the SAP API Business Hub.
- Download an Entity Data Model XML (EDMX) file of the external service definition from the SAP API Business Hub.
- Add an EDMX [1] file to your project.
- Consume an External Service in a UI Application.

Prerequisites

You have added custom business logic to your extension.

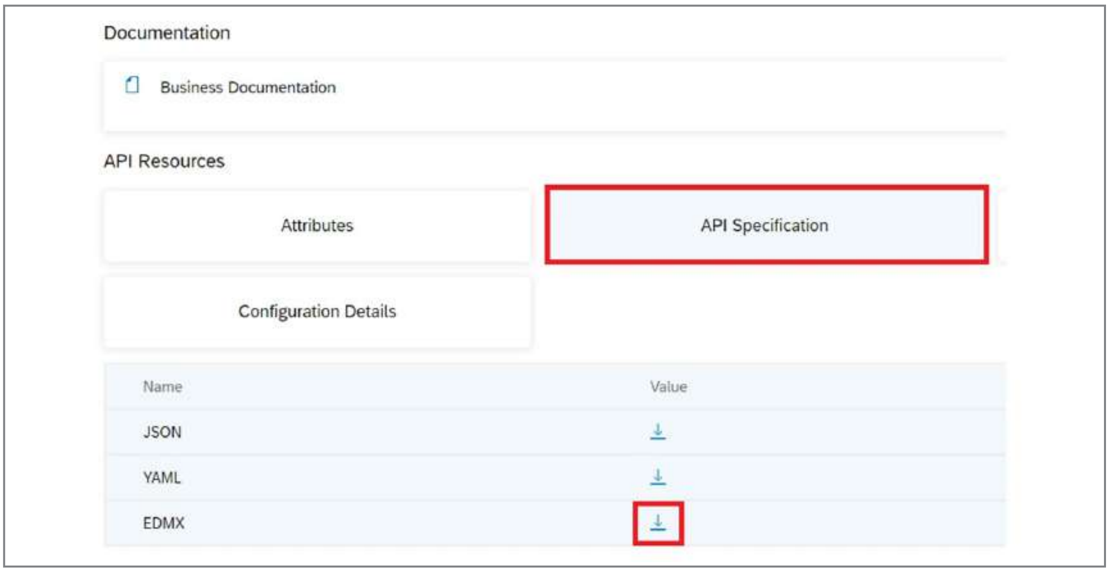
Download the Business Partner EDMX File

1. Open the SAP API Business Hub [2] page in your browser.
2. Type **`Business Partner A2X`** into the page's search field and carry out the search.
3. In the result list, choose `Business Partner (A2X)`.
4. Choose the first API with the `Found in: SAP S/4HANA Sandbox` information on the right upper corner of this API.


SAP API Business Hub
Explore
Resources
Discover Integrations
Partner with Us
SAP API Business Hub
Accelerate your integrations, extensions, and innovations
Business Partner A2X
Discover
Explore Products, Processes, and Partne
Explore
Discover Integrations and Accelerators
Consume
Visualize and consume integrations and
Products
Business Processes
Categories
Partners
Featured
Choose a Product to Explore
See the various resources that each product has to offer
S/4HANA Cloud
S/4HANA
Customer Experience

5. Under *Document*, choose the *API Specification* button.

6. Choose the `EDMX` option from the list and select the download button (if you're asked to log on, log on using your SAP user).



Name	Value
JSON	⤓
YAML	⤓
EDMX	⤓

Add the EDMX File to the Project

1. Ensure *cds watch* is still running in your terminal.

2. Drag the `API_BUSINESS_PARTNER.edmx` file from your browser's download area/folder onto your Business Application Studio workplace and drop it into the `srv` folder of your `risk-management` app.

   CAP has noticed the new file and automatically created a new `external` folder under `srv` and added a new `API_BUSINESS_PARTNER.csn` file to it. (capire [3] is a compact representation of CDS).

3. In your project, open the `db/schema.cds` file and enter the code listed below between `//### BEGIN OF INSERT` and `//### END OF OF INSERT`.

```
namespace riskmanagement;
```

```
using {managed} from '@sap/cds/common';

entity Risks : managed {
   key ID : UUID @(Core.Computed : true);
       title : String(100);
       owner : String;
       prio : String(5);
       descr : String;
       miti : Association to Mitigations;
       impact : Integer;
       //bp : Association to BusinessPartners;
       criticality : Integer;
}

entity Mitigations : managed {
  key ID : UUID @(Core.Computed : true);
      descr : String;
      owner : String;
      timeline : String;
      risks : Association to many Risks
               on risks.miti = $self;
}

//### BEGIN OF INSERT

// using an external service from S/4
using { API_BUSINESS_PARTNER as external } from '../srv/external/
API_BUSINESS_PARTNER.csn';

entity BusinessPartners as projection on external.A_BusinessPartner {
   key BusinessPartner,
   LastName,
   FirstName
}

//### END OF OF INSERT
```

With this code you create a so-called projection for your new service. Of the many entities and properties in these entities, that are defined in the `API_BUSINESS_PARTNER` service, you just look at one of the entities (`A_BusinessPartner`) and just three of its properties: `BusinessPartner`, `LastName`, and `FirstName`, so your projection is using a subset of everything the original service has to offer.

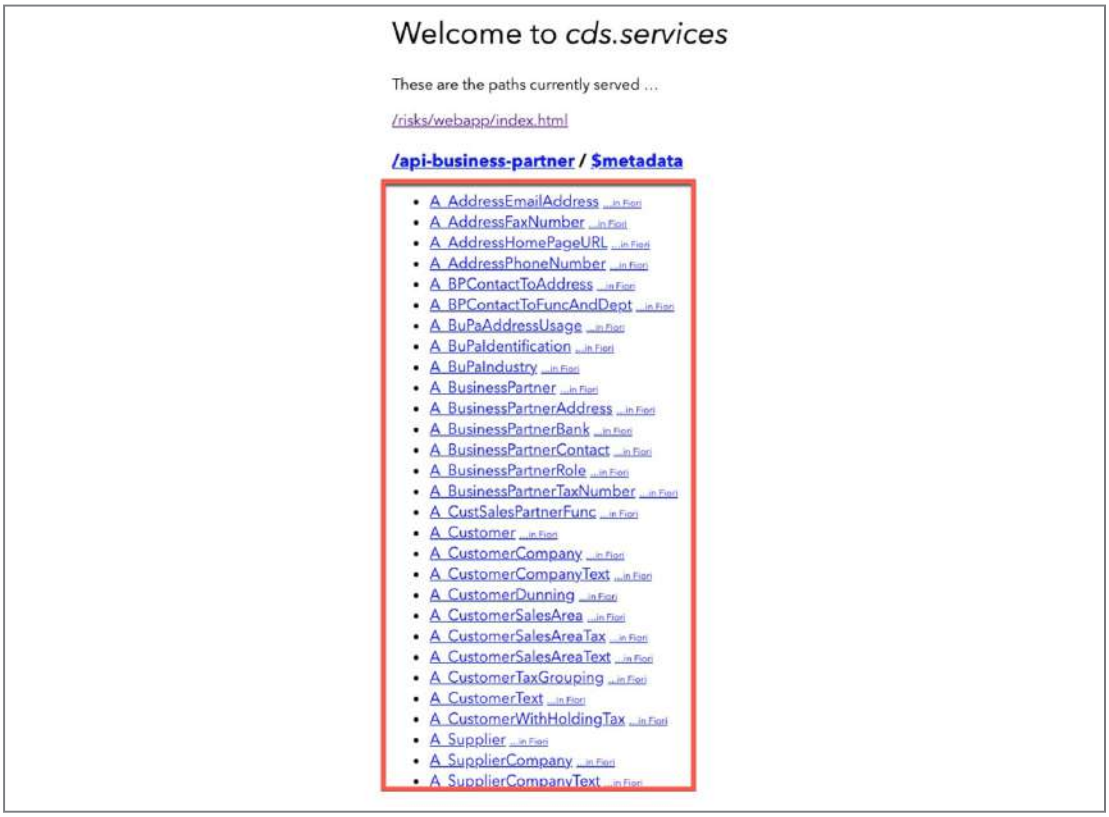
4. Open the `srv/risk-service.cds` file.

5. Uncomment the `entity BusinessPartners` line.

```
using {riskmanagement as rm} from '../db/schema';

/**
 * For serving end users
 */
service RiskService @(path : 'service/risk') {
   entity Risks as projection on rm.Risks;
   annotate Risks with @odata.draft.enabled;
   entity Mitigations as projection on rm.Mitigations;
   annotate Mitigations with @odata.draft.enabled;
   entity BusinessPartners as projection on rm.BusinessPartners;
 }
```

6. Your SAP Fiori elements app should still be running in your web browser. Select the SAP icon on the left upper corner to navigate back to the index page. Hit refresh in your browser. Now select the *Risks* tile and in the application press *Go*.

   The browser now shows a `BusinessPartner` service next to the `Mitigations` and `Risks`



Welcome to *cds.services*

These are the paths currently served ...

/risks/webapp/index.html

/api-business-partner / $metadata

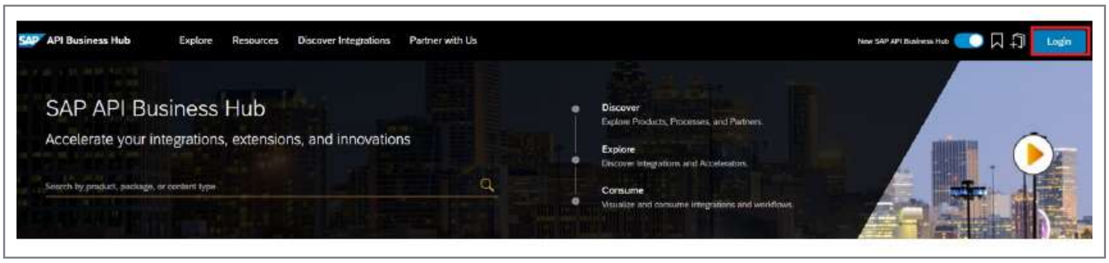
- A_AddressEmailAddress ...in Fiori
- A_AddressFaxNumber ...in Fiori
- A_AddressHomePageURL ...in Fiori
- A_AddressPhoneNumber ...in Fiori
- A_BPContactToAddress ...in Fiori
- A_BPContactToFuncAndDept ...in Fiori
- A_BuPaAddressUsage ...in Fiori
- A_BuPaIdentification ...in Fiori
- A_BuPaIndustry ...in Fiori
- A_BusinessPartner ...in Fiori
- A_BusinessPartnerAddress ...in Fiori
- A_BusinessPartnerBank ...in Fiori
- A_BusinessPartnerContact ...in Fiori
- A_BusinessPartnerRole ...in Fiori
- A_BusinessPartnerTaxNumber ...in Fiori
- A_CustSalesPartnerFunc ...in Fiori
- A_Customer ...in Fiori
- A_CustomerCompany ...in Fiori
- A_CustomerCompanyText ...in Fiori
- A_CustomerDunning ...in Fiori
- A_CustomerSalesArea ...in Fiori
- A_CustomerSalesAreaTax ...in Fiori
- A_CustomerSalesAreaText ...in Fiori
- A_CustomerTaxGrouping ...in Fiori
- A_CustomerText ...in Fiori
- A_CustomerWithHoldingTax ...in Fiori
- A_Supplier ...in Fiori
- A_SupplierCompany ...in Fiori
- A_SupplierCompanyText ...in Fiori

Connect your App with the Business Partner API Sandbox Environment of the SAP API Business Hub

At this point, you have a new service exposed with a definition based on the original `edmx` file. However, it doesn't have any connectivity to a back end, so, there's no data yet. In this case, you do not create local data as with your `risks` and `mitigations` entities, but you connect your service to the Sandbox environment of the SAP API Business Hub for the Business Partner API that you want to use. To use the API Business Hub Sandbox APIs, you require an API key.

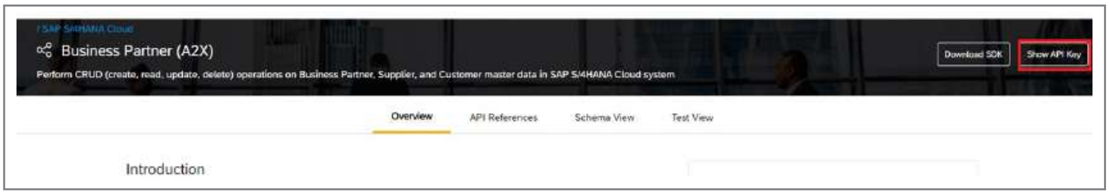
1. Go back to the SAP API Business Hub page in your browser.

2. Make sure you are logged in. If not, press the *Log On* button on the upper right (use the SAP user that you also used to create your BTP trial account for the Log On)


SAP API Business Hub
Explore
Resources
Discover Integrations
Partner with Us
New SAP API Business Hub
Login
SAP API Business Hub
Accelerate your integrations, extensions, and innovations
Search by product, package, or content type
Discover
Explore Products, Processes, and Partners
Explore
Discover Integrations and Accelerators
Consume
Visualize and consume integrations and workflows

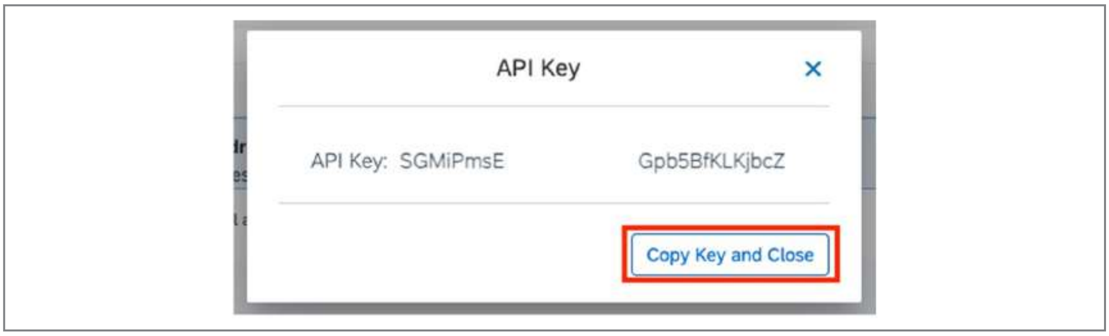
3. Again, navigate to the Business Partner API (*SAP S/4HANA Cloud → Business Partner (A2X)*).

4. In the right upper corner, choose *Show API Key* to see your API key.



Perform CRUD (create, read, update, delete) operations on Business Partner, Supplier, and Customer master data in SAP S/4HANA Cloud system

5. Copy the API key.

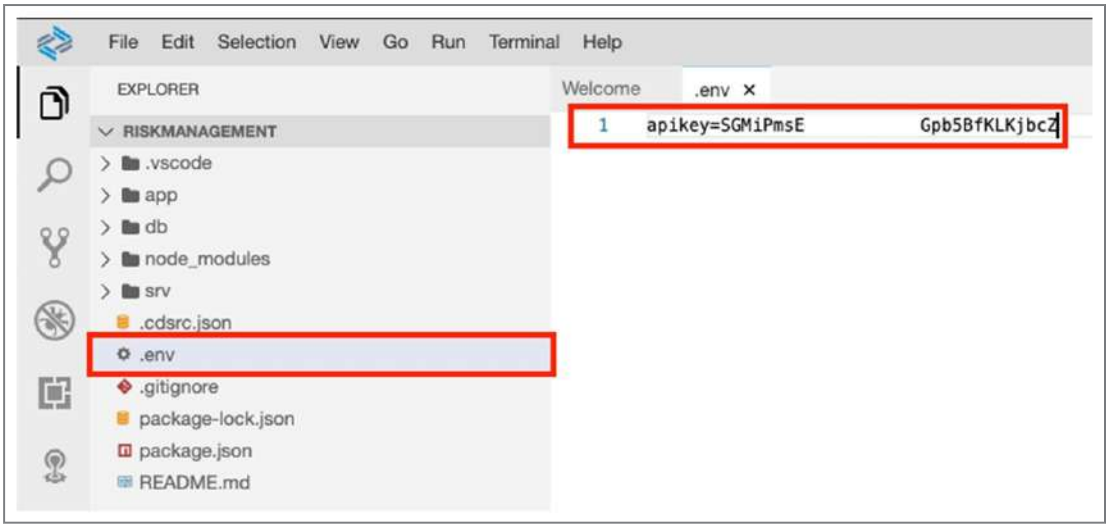


API Key: SGMiPmsE GpbSBfKLKjbcZ

6. In your project in Business Application Studio, create the file `.env` in the `root` of the project (next to files `package.json`, `README.md`, and so on). Copy the following line into the file and replace `<YOUR-API-KEY>` with the API key that you copied in the previous step.

```
apikey=<YOUR-API-KEY>
```

The result should look like the following:



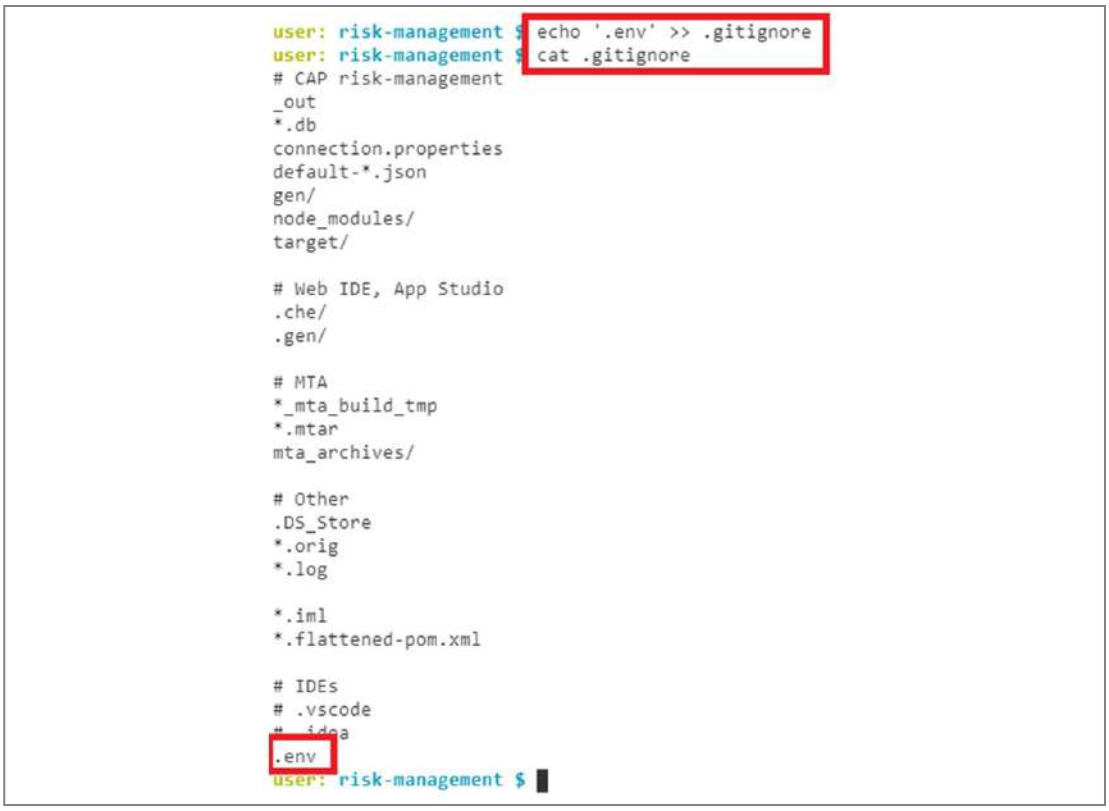
```
apikey=SGMiPmsE          Gpb5BfKLKjbcZ
```

The `.env` file is an environment file providing values into the runtime environment of your CAP service. You are going to use the API key to call the Business Partner API in the API Business Hub Sandbox environment.

7. By mentioning the `.env` file in the `.gitignore` file you make sure, that when you are using git as a version-management-system for your project, no credentials get accidentally leaked into your potentially public git repository.

   To add `.env` to the `.gitignore` file, execute the following command in a new terminal:`echo'.env'>> .gitignore`

You can verify that the `.env` has been added with the command:`cat .gitignore`



```
user: risk-management $ echo '.env' >> .gitignore
user: risk-management $ cat .gitignore
# CAP risk-management
_out
*.db
connection.properties
default-*.json
gen/
node_modules/
target/

# Web IDE, App Studio
.che/
.gen/

# MTA
*_mta_build_tmp
*.mtar
mta_archives/

# Other
.DS_Store
*.orig
*.log

*.iml
*.flattened-pom.xml

# IDEs
# .vscode
# .idea
.env
user: risk-management $
```

8. Open the `package.json` file and add the following lines between `//### BEGIN OF INSERT` and `//### END OF OF INSERT`:

```
"cds": {
   "requires": {
     "API_BUSINESS_PARTNER": {
       "kind": "odata",
       "model": "srv/external/API_BUSINESS_PARTNER",
       //### BEGIN OF INSERT
       "credentials": {
          "url": "https://sandbox.api.sap.com/s4hanacloud/sap/opu/
odata/sap/API_BUSINESS_PARTNER/"
       }
       //### End OF INSERT
    }
  }
}
```

Now that you have set all the configurations for the external call, you will implement a custom service handler for the external *BusinessPartner* service in the next step.

9. Open the `risk-service.js` file and insert the following lines between `//### BEGIN OF INSERT` and `//### END OF OF INSERT`:

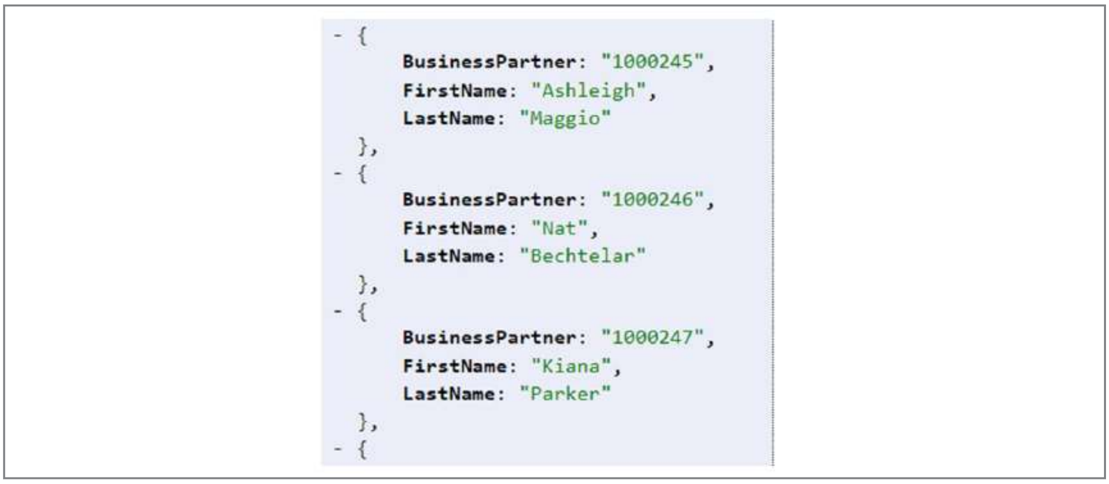
```
// Imports
const cds = require("@sap/cds");

/**
  * The service implementation with all service handlers
  */
module.exports = cds.service.impl(async function () {
  // Define constants for the Risk and BusinessPartners entities from
the risk-service.cds file
  const { Risks, BusinessPartners } = this.entities;

  /**
    * Set criticality after a READ operation on /risks
    */
  this.after("READ", Risks, (data) => {
    const risks = Array.isArray(data) ? data : [data];

    risks.forEach((risk) => {
      if (risk.impact >= 100000) {
        risk.criticality = 1;
      } else {
        risk.criticality = 2;
      }
   });
 });

 //### BEGIN OF INSERT

 // connect to remote service
 const BPsrv = await cds.connect.to("API_BUSINESS_PARTNER");

 /**
  * Event-handler for read-events on the BusinessPartners entity.
  * Each request to the API Business Hub requires the apikey in the
header.
  */
 this.on("READ", BusinessPartners, async (req) => {
   // The API Sandbox returns alot of business partners with empty
names.
   // We don't want them in our application
   req.query.where("LastName <> '' and FirstName <> '' ");

   return await BPsrv.transaction(req).send({
     query: req.query,
     headers: {
       apikey: process.env.apikey,
     },
   });
 });
 //### END OF INSERT
});
```

You've now created a custom handler for your service. This time it called `on` for the `READ` event.

The handler is invoked when your `BusinessPartner` service is called for a read, so whenever there's a request for business partner data, this handler is called. It ensures the request for the business partner is directed to the external business partner service. Furthermore, you have added a `where` clause to the request, which selects only business partners where the first and last name is set.

10. Save the file.

11. In your browser, open the `BusinessPartners` link to see the data.



```
- {
      BusinessPartner: "1000245",
      FirstName: "Ashleigh",
      LastName: "Maggio"
  },
- {
      BusinessPartner: "1000246",
      FirstName: "Nat",
      LastName: "Bechtelar"
  },
- {
      BusinessPartner: "1000247",
      FirstName: "Kiana",
      LastName: "Parker"
  },
- {
```

Consume the External Service in Your UI Application

In this chapter, you incorporate the external service into the UI application.

1. Open the `db/data/schema.cds` file.

2. Uncomment the `bp` property.

```
namespace riskmanagement;

using {managed} from '@sap/cds/common';

entity Risks : managed {
   key ID : UUID @(Core.Computed : true);
    title : String(100);
    owner : String;
    prio : String(5);
    descr : String;
    miti : Association to Mitigations;
    impact : Integer;
     bp : Association to BusinessPartners; // <-- uncomment this
    criticality : Integer;
}

entity Mitigations : managed {
  key ID : UUID @(Core.Computed : true);
    descr : String;
    owner : String;
     timeline : String;
    risks : Association to many Risks
             on risks.miti = $self;
}

// using an external service from S/4
using {API_BUSINESS_PARTNER as external} from '../srv/external/
API_BUSINESS_PARTNER.csn';

entity BusinessPartners as projection on external.A_BusinessPartner {
  key BusinessPartner, FirstName, LastName,
 }
```

   As you got a new property in your entity, you need to add data for this property in the local data file that you've created before for the `risk` entity.

3. Open the file `riskmanagement-Risks.csv` in your `db/data` folder.

4. Replace the content with the new content below which additionally includes the BP data.

```
ID;createdAt;createdBy;title;owner;prio;descr;miti_id;impact;bp_Busines
sPartner
20466922-7d57-4e76-b14c-e53fd97dcb11;2019-10-24;SYSTEM;CFR non-
compliance;Fred Fish;3;Recent restructuring might violate CFR code
71;20466921-7d57-4e76-b14c-e53fd97dcb11;10000;9980000448
20466922-7d57-4e76-b14c-e53fd97dcb12;2019-10-24;SYSTEM;SLA violation
with possible termination cause;George Gung;2;Repeated SAL violation
on service delivery for two successive quarters;20466921-7d57-4e76-
b14c-e53fd97dcb12;90000;9980002245
20466922-7d57-4e76-b14c-e53fd97dcb13;2019-10-24;SYSTEM;Shipment
violating export control;Herbert Hunter;1;Violation of export and
trade control with unauthorized downloads;20466921-7d57-4e76-b14c-
e53fd97dcb13;200000;9980000230
```

5. Save the file.

   If you check the content of the file, you see numbers like `9980000230` at the end of the lines, representing business partners.

Add the Business Partner Field to the UI

You need to introduce the business partner field in the UI:

- Add a label for the columns in the result list table as well as in the object page by adding a title annotation.
- Add the business partner as a line item to include it as a column in the result list.
- Add the business partner as a field to a field group, which makes it appear in a form on the object page.

All this happens in the cds file that has all the UI annotations. Enter the code between `//### BEGIN OF INSERT` and `//### END OF OF INSERT`.

1. Open the `app/common.cds` file.
2. Insert the following parts:

```
using riskmanagement as rm from '../db/schema';

 // Annotate Risk elements
 annotate rm.Risks with {
   ID @title : 'Risk';
   title @title : 'Title';
   owner @title : 'Owner';
   prio @title : 'Priority';
   descr @title : 'Description';
   miti @title : 'Mitigation';
   impact @title : 'Impact';
   //### BEGIN OF INSERT
   bp @title : 'Business Partner';
   //### END OF INSERT
   criticality @title : 'Criticality';
 }

 // Annotate Miti elements
 annotate rm.Mitigations with {
   ID @(
     UI.Hidden,
     Commong : {Text : descr}
   );
   owner @title : 'Owner';
   descr @title : 'Description';
 }

 //### BEGIN OF INSERT
 annotate rm.BusinessPartners with {
   BusinessPartner @(
     UI.Hidden,
     Common : {Text : LastName}
   );
   LastName @title : 'Last Name';
   FirstName @title : 'First Name';
 }
 //### END OF INSERT

annotate rm.Risks with {
   miti @(Common : {
     //show text, not id for mitigation in the context of risks
     Text : miti.descr,
     TextArrangement : #TextOnly,
     ValueList : {
     Label : 'Mitigations',
     CollectionPath : 'Mitigations',
     Parameters : [
       {
         $Type : 'Common.ValueListParameterInOut',
         LocalDataProperty : miti_ID,
         ValueListProperty : 'ID'
       },
       {
         $Type : 'Common.ValueListParameterDisplayOnly',
         ValueListProperty : 'descr'
       }
     ]
   }
 });
 //### BEGIN OF INSERT
 bp @(Common : {
   Text : bp.LastName,
   TextArrangement : #TextOnly,
   ValueList : {
      Label : 'Business Partners',
      CollectionPath : 'BusinessPartners',
      Parameters : [
         {
           $Type : 'Common.ValueListParameterInOut',
           LocalDataProperty : bp_BusinessPartner,
           ValueListProperty : 'BusinessPartner'
         },
         {
            $Type : 'Common.ValueListParameterDisplayOnly',
            ValueListProperty : 'LastName'
          },
          {
             $Type : 'Common.ValueListParameterDisplayOnly',
             ValueListProperty : 'FirstName'
          }
       ]
     }
   })
//### END OF INSERT }
```

3. Open the `app/risk/annotations.cds` file and insert the following lines between `//### BEGIN OF INSERT` and `//### END OF INSERT`:

```
using RiskService from '../../srv/risk-service';

 // Risk List Report Page
 annotate RiskService.Risks with @(UI : {
     HeaderInfo : {
       TypeName : 'Risk',
       TypeNamePlural : 'Risks',
       Title : {
         $Type : 'UI.DataField',
         Value : title
       },
       Description : {
         $Type : 'UI.DataField',
         Value : descr
       }
   },
   SelectionFields : [prio],
   Identification : [{Value : title}],
   // Define the table columns
   LineItem : [
   {Value : title},
     {Value : miti_ID},
     {Value : owner},
     //### BEGIN OF INSERT
     {Value : bp_BusinessPartner},
     //### END OF INSERT
     {
       Value : prio,
       Criticality : criticality
     },
     {
       Value : impact,
       Criticality : criticality
     },
   ],
 });

 // Risk Object Page
 annotate RiskService.Risks with @(UI : {
   Facets : [{
     $Type : 'UI.ReferenceFacet',
     Label : 'Main',
     Target : '@UI.FieldGroup#Main',
   }],
   FieldGroup #Main : {Data : [
     {Value : miti_ID},,
     {Value : owner},
     //### BEGIN OF INSERT
     {Value : bp_BusinessPartner},
     //### END OF INSERT
     {
       Value : prio,
       Criticality : criticality
     },
     {
       Value : impact,
       Criticality : criticality
     }
   ]},
 });
```

What does the code do? The first part enables the title and adds the business partner first as a column to the list and then as a field to the object page, just like other columns and fields were added before.

The larger part of new annotations activates the same qualities for the `bp` field as it happened before in Create a CAP-Based Service [4] for the `miti` field: Instead of showing the ID of the business partner, its `LastName` property is displayed. The `ValueList` part introduces a value list for the business partner and shows it last and first name in it.

4. Save the file.

5. Open the `srv/risk-service.js` file.

6. Add the following lines between `//### BEGIN OF INSERT` and `//### END OF OF INSERT` to the file:

```
// Imports
 const cds = require("@sap/cds");

 /**
   * The service implementation with all service handlers
   */
 module.exports = cds.service.impl(async function () {
   // Define constants for the Risk and BusinessPartners entities from
the risk-service.cds file
     const { Risks, BusinessPartners } = this.entities;
 /**
   * Set criticality after a READ operation on /risks
   */
 this.after("READ", Risks, (data) => {
   const risks = Array.isArray(data) ? data : [data];

   risks.forEach((risk) => {
     if (risk.impact >= 100000) {
       risk.criticality = 1;
     } else {
       risk.criticality = 2;
     }
   });
 });

// connect to remote service
 const BPsrv = await cds.connect.to("API_BUSINESS_PARTNER");

/**
  * Event-handler for read-events on the BusinessPartners entity.
  * Each request to the API Business Hub requires the apikey in the
header.
  */
 this.on("READ", BusinessPartners, async (req) => {
   // The API Sandbox returns alot of business partners with empty
names.
   // We don't want them in our application
   req.query.where("LastName <> '' and FirstName <> '' ");

   return await BPsrv.transaction(req).send({
     query: req.query,
     headers: {
       apikey: process.env.apikey,
     },
   });
 });
//### BEGIN OF INSERT

/**
  * Event-handler on risks.
  * Retrieve BusinessPartner data from the external API
  */
 this.on("READ", Risks, async (req, next) => {
  /*
    Check whether the request wants an "expand" of the business partner
    As this is not possible, the risk entity and the business partner
entity are in different systems (SAP BTP and S/4 HANA Cloud),
    if there is such an expand, remove it
  */
 const expandIndex = req.query.SELECT.columns.findIndex(
   ({ expand, ref }) => expand && ref[0] === "bp"
 );
 console.log(req.query.SELECT.columns);
 if (expandIndex < 0) return next();

 req.query.SELECT.columns.splice(expandIndex, 1);
 if (
    !req.query.SELECT.columns.find((column) =>
    column.ref.find((ref) => ref == "bp_BusinessPartner")
  )
 ) {
   req.query.SELECT.columns.push({ ref: ["bp_BusinessPartner"] });
 }

 /*
   Instead of carrying out the expand, issue a separate request for
each business partner
   This code could be optimized, instead of having n requests for n
business partners, just one bulk request could be created
 */
 try {
   const res = await next();
   await Promise.all(
     res.map(async (risk) => {
     const bp = await BPsrv.transaction(req).send({
       query: SELECT.one(this.entities.BusinessPartners)
        .where({ BusinessPartner: risk.bp_BusinessPartner })
        .columns(["BusinessPartner", "LastName", "FirstName"]),
       headers: {
         apikey: process.env.apikey,
       },
     });
     risk.bp = bp;
    })
   );
  } catch (error) {}
 });
 //### END OF INSERT
});
```

You have added another custom handler, this one is called `on` a `READ` of the `Risks` service. It checks whether the request includes a so-called expand for business partners. This is a request that is issued by the UI when the list should be filled. While it mostly contains columns that directly belong to the `Risks` entity, it also contains the business partner. As we have seen in the annotation file, instead of showing the ID of the business partner, the last name of the business partner will be shown. This data is in the business partner and not in the risks entity. Therefore, the UI wants to expand, that is, for each risk the corresponding business partner is also read.

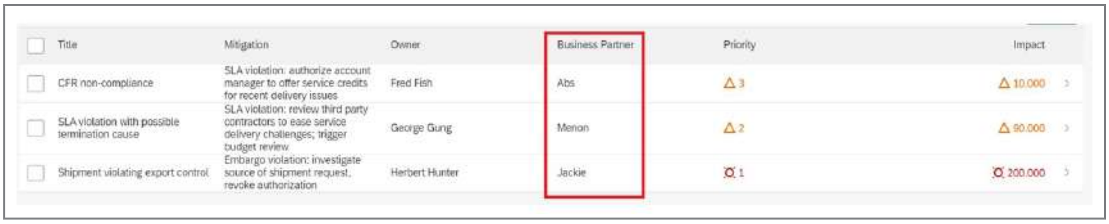
As the business partner does not reside in the CAP database but in a remote system instead, a direct expand is not possible. The data needs to be retrieved from the S/4HANA Cloud system.

7. Save the file.

8. In your tab with the application, go back to the *index.html* page and press refresh.

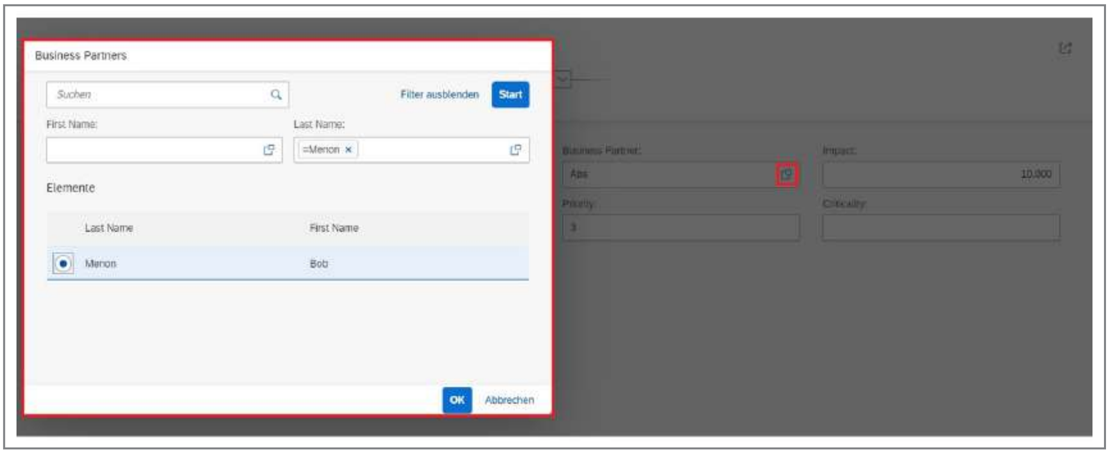
9. On the launch page that now comes up, choose the *Risks* tile and then select *Go*.

   You now see the `Risks` application with the business partner data in both the result list and the object page, which is loaded when you select one of the rows in the table:



□	Title	Mitigation	Owner	Business Partner	Priority	Impact
□	CFR non-compliance	SLA violation: authorize account manager to offer service credits for recent delivery issues	Fred Fish	Abs	△ 3	△ 10.000
□	SLA violation with possible termination cause	SLA violation: review third party contractors to ease service delivery challenges; trigger budget review	George Gung	Menon	△ 2	△ 90.000
□	Shipment violating export control	Embargo violation: investigate source of shipment request, revoke authorization	Herbert Hunter	Jackie	☼ 1	☼ 200.000

   When you are on the object page, select the *Edit* button on the top right of the screen. Now you can use the value help for the Business Partner field and search for other Business Partners, which are provided via the Business Partner API.



Business Partners

Suchen

First Name:

Last Name: =Menon

Elemente

	Last Name	First Name
●	Menon	Bob

Business Partner: Abs

Impact: 10.000

Priority: 3

Criticality:

Summary

You have added an external business partner service to your application. The last step is to deploy your application manually.

Unit 4
Exercise 4

Add an External Service

Simulation: Add an External Service

For more information on *Add an External Service*, please view the simulation in the lesson *Add an External Service* in your online course.

1.

Unit 4
Solution 4

Add an External Service

Simulation: Add an External Service

For more information on *Add an External Service*, please view the simulation in the lesson *Add an External Service* in your online course.

1.

Reference Links

For your convenience, this section contains the external references in this lesson.

If links are used multiple times in the text, only the first location is mentioned in the reference table.

Table 5: Reference Links: External Service

Ref#	Section	Context text fragment	Brief description	Link
1	Task Flow	Add an EDMX file to your project.	EDMX OData standard	http://docs.oasis-open.org/odata/odata/v4.0/odata-v4.0-part3-csdl.html
2	Download the Business Partner EDMX File	Open the SAP API Business Hub page	SAP API Business Hub	https://api.sap.com/
3	Add the EDMX File to the Project	as it happened before in Create a CAP-Based Service	SAP Cloud Application Programming Model	https://cap.cloud.sap/docs/cds/csn
4	Add the Business Partner Field to the UI	capire is a compact representation of CDS	SAP Cloud Application Programming Model	https://cap.cloud.sap/docs/cds/csn

LESSON SUMMARY

You should now be able to:

- Add and consume an external service

Unit 4
Lesson 2

SAP BTP Connectivity

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe how SAP BTP applications can securely access remote services

SAP BTP Connectivity

Usage Scenario

You want to use data from your SAP backend in SAP BTP. You want to use a secure and robust connection.

Learning objectives

- Learn that SAP BTP Connectivity consists of two services, the SAP Connectivity service, which is only required for on-premise connections, and the SAP Destination service.
- Be able to reproduce two scenarios considered, name the associated configuration steps, and assign them to the landscape.

SAP BTP Connectivity Services

SAP BTP Connectivity provides two services for the Cloud Foundry environment:

- The SAP Connectivity service.
- The SAP Destination service.

The Connectivity service provides a connectivity proxy that can be used to access on-premise resources. Using the Destination service, you can retrieve and store the technical information about the target resource (destination) that you need to connect your application to a remote service or system.

In the following, we will only cover the Cloud Foundry Runtime. In principle, there are two types of connections to and from the SAP BTP:

- Internet Connectivity uses Destination service.
- Cloud to On-Premise Connectivity uses Destination service and Connectivity service.

Connectivity to and from the SAP BTP

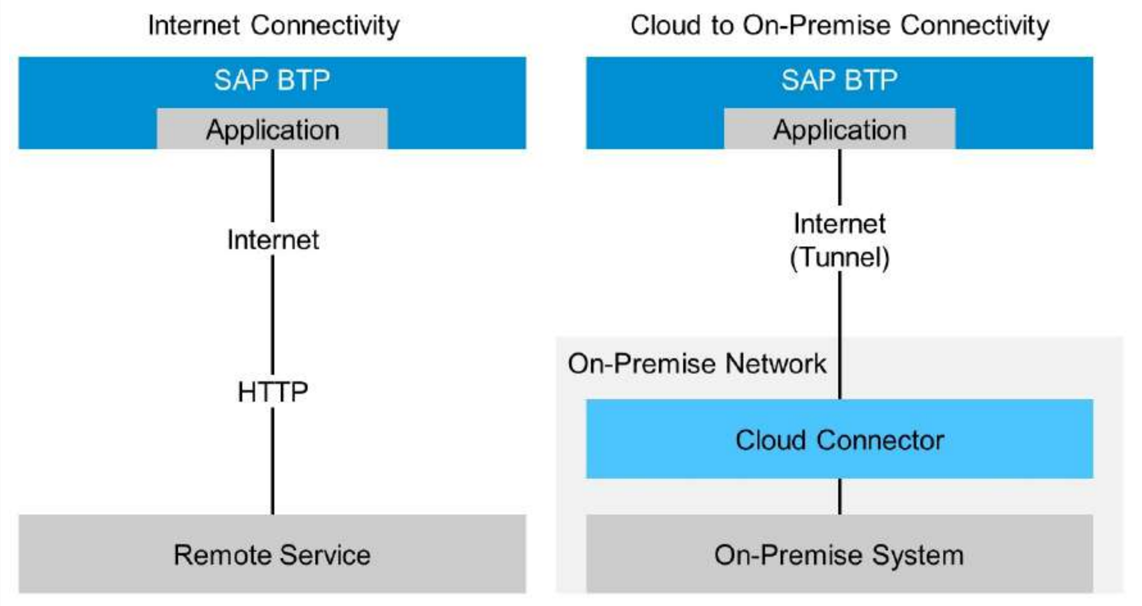


Figure 43: Connectivity to and from the SAP BTP

The upper part depicts the SAP BTP, the lower part the remote systems and services to be connected. The Internet Connectivity can be reached via HTTPs, while the connection to an on-premise system requires the Cloud Connector. The communication is bi-directional.

Connectivity Service Types and Communication Types

Connectivity Service Type	Communication Type
Connectivity Service	on-premise communication
Destination Service	on-premise communication
Destination Service	Internet Connectivity

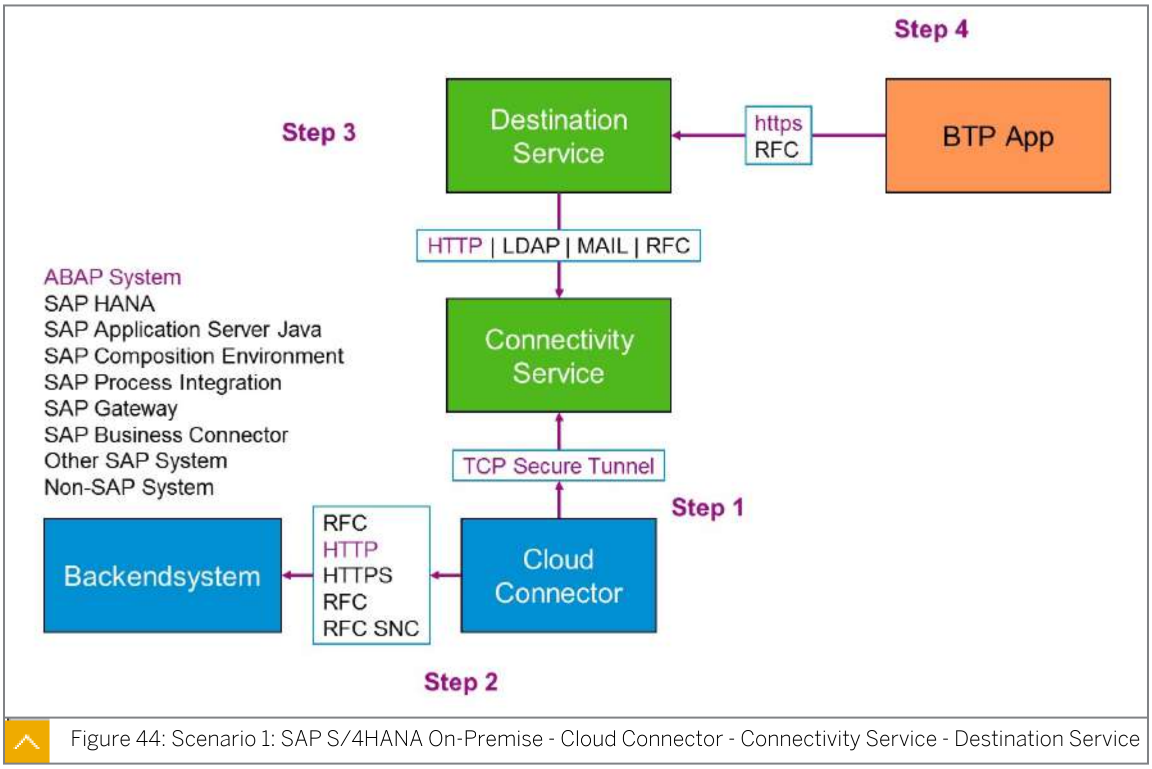
The Connectivity Service is only used in conjunction with the Cloud Connector.

We limit ourselves to the two most used scenarios:

- SAP S/4HANA On-Premise - Cloud Connector - Connectivity Service - Destination Service.
- SAP S/4HANA Cloud - Destination Service.

Scenario 1: SAP S/4HANA On-Premise - Cloud Connector - Connectivity Service - Destination Service

This scenario is used to connect any backends to SAP BTP. The optional connection parameters are located between the connections. The parameters used in the example are colored bold and purple.



Step 4

Step 3

Destination Service

https
RFC

BTP App

HTTP | LDAP | MAIL | RFC

ABAP System
SAP HANA
SAP Application Server Java
SAP Composition Environment
SAP Process Integration
SAP Gateway
SAP Business Connector
Other SAP System
Non-SAP System

Connectivity Service

TCP Secure Tunnel

Step 1

Backendsystem

RFC
HTTP
HTTPS
RFC
RFC SNC

Cloud Connector

Step 2

Figure 44: Scenario 1: SAP S/4HANA On-Premise - Cloud Connector - Connectivity Service - Destination Service

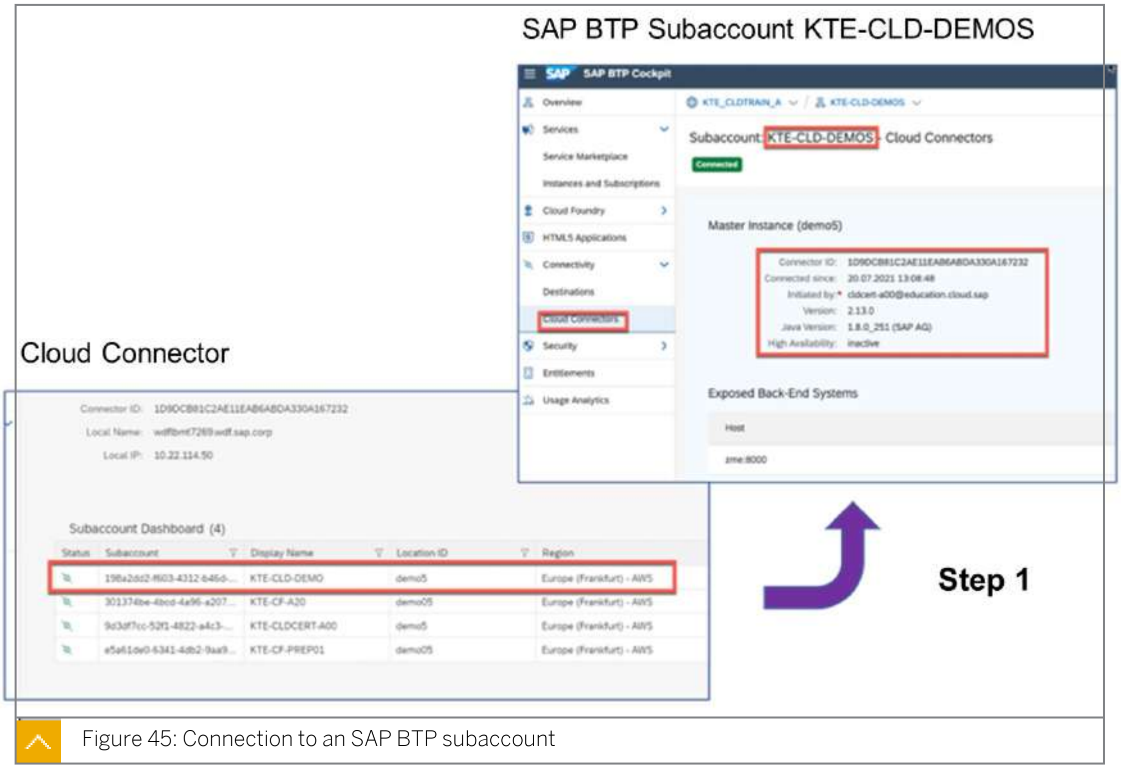
The following four steps describe the structure of the scenario:

Start

The SAP backend system provides an OData interface via the SAP Gateway.

Step 1:

In the Cloud Connector, implemented in the backend, a secure connection to an SAP BTP subaccount is configured.



SAP BTP Subaccount KTE-CLD-DEMOS

SAP BTP Cockpit

Subaccount: KTE-CLD-DEMOS - Cloud Connectors

Connected

Master Instance (demo5)

Connector ID: 1D9DCB81C2AE11EAB6ABDA330A167232
Connected since: 20.07.2021 13:08:48
Initiated by:* cldcert-a00@education.cloud.sap
Version: 2.13.0
Java Version: 1.8.0_251 (SAP AG)
High Availability: inactive

Exposed Back-End Systems

Host
zme:8000

Cloud Connector

Connector ID: 1D9DCB81C2AE11EAB6ABDA330A167232
Local Name: wdfbmt7269.wdf.sap.corp
Local IP: 10.22.114.50

Subaccount Dashboard (4)

Status	Subaccount	Display Name	Location ID	Region
	198a2dd2-f603-4312-b46d-...	KTE-CLD-DEMO	demo5	Europe (Frankfurt) - AWS
	301374be-4bcd-4a96-a207...	KTE-CF-A20	demo05	Europe (Frankfurt) - AWS
	9d3df7cc-52f1-4822-a4c3-...	KTE-CLDCERT-A00	demo5	Europe (Frankfurt) - AWS
	e5a61de0-6341-4db2-9aa9...	KTE-CF-PREP01	demo05	Europe (Frankfurt) - AWS

Step 1

Figure 45: Connection to an SAP BTP subaccount

Step 2:

The Cloud Connector is configured against the backend system via HTTP and all services can be found under the HTTP service tree /sap (see transaction /nSICF in the backend system).

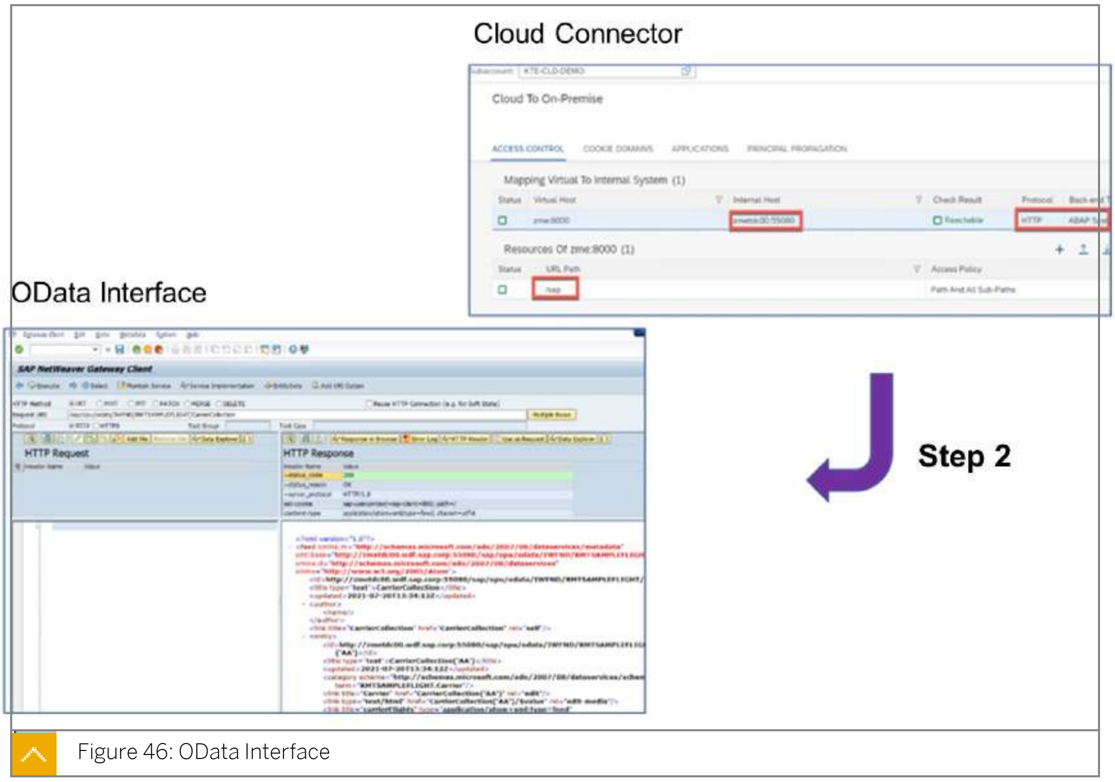
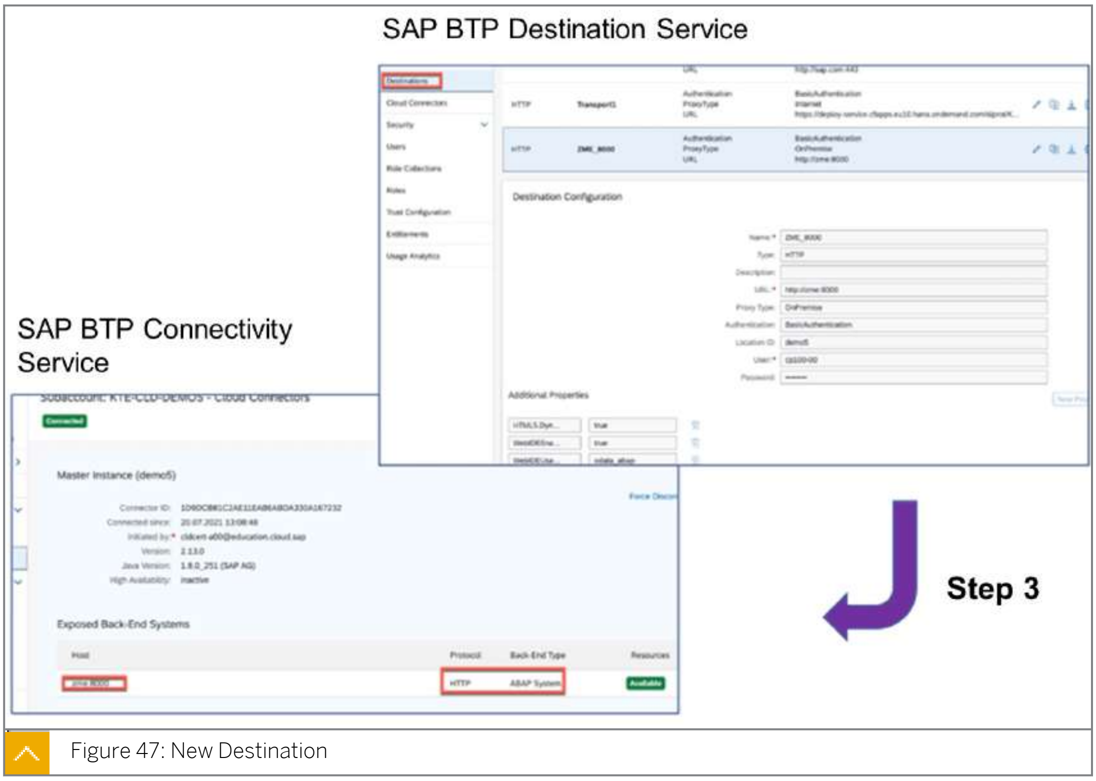


Figure 46: OData Interface

On the backend side, all connections are now created to consume the data from the OData interface.

Step 3:

A new destination is created and configured against the Cloud Connector via the Connectivity Service. The *Proxy type* is **`OnPremise`**.



HTTP	Transport1	Authentication ProxyType URL	BasicAuthentication Internet https://deploy-service.cfapps.eu10.hana.ondemand.com/slprotK...
HTTP	ZME_8000	Authentication ProxyType URL	BasicAuthentication OnPremise http://zme:8000

Destination Configuration

Name*: ZME_8000
Type: HTTP
Description:
URL*: http://zme:8000
Proxy Type: OnPremise
Authentication: BasicAuthentication
Location ID: demo5
User*: cp100-00
Password: •••••••

HTML5.Dyn...	true
WebIDEEna...	true
WebIDEUsa...	odata_abap

Subaccount: KTE-CLD-DEMOS - Cloud Connectors

Connected

Master Instance (demo5)

Connector ID: 1D90CB81C2AE11EAB6ABDA330A167232
Connected since: 20.07.2021 13:08:48
Initiated by:* cldcert-a00@education.cloud.sap
Version: 2.13.0
Java Version: 1.8.0_251 (SAP AG)
High Availability: inactive

Exposed Back-End Systems

Host	Protocol	Back-End Type	Resources
zme:8000	HTTP	ABAP System	Available

Figure 47: New Destination

Step 4:

The destination can now be reached and used from within the App via:

- API
- Programmatically via ABAP, Java, Node.js

Scenario 2: SAP S/4HANA Cloud - Destination Service

In this scenario, only the Destination Service is used. This is configured with a remote service.

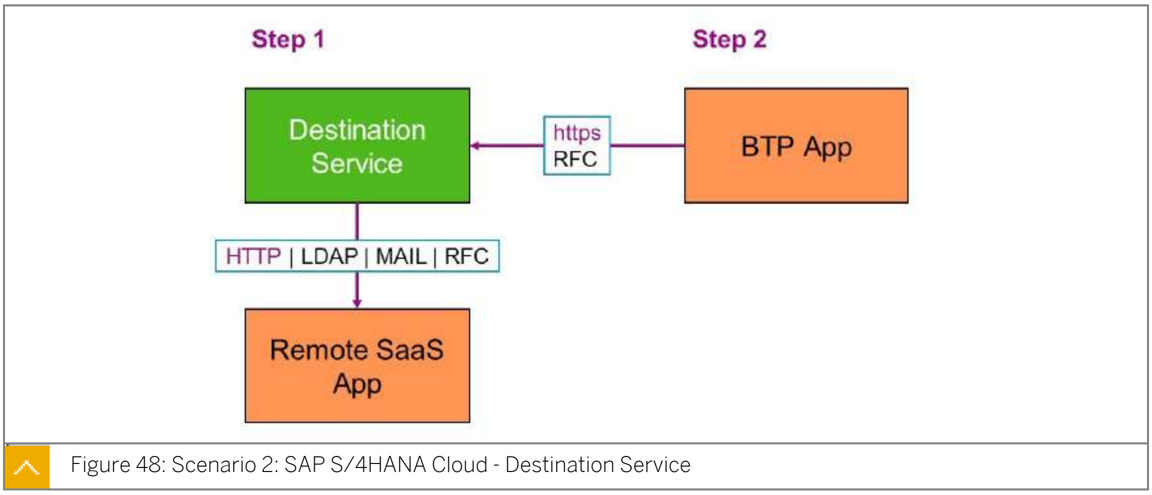


Figure 48: Scenario 2: SAP S/4HANA Cloud - Destination Service

Again, you find the optional connection parameters within the connections.

Summary

SAP BTP Connectivity consists of two services, the SAP Connectivity service and the SAP Destination service. The Destination service encapsulates the concrete connections. If a backend is accessed, the Cloud Connector is used in combination with the SAP Connectivity service.

Further Reading

- SAP BTP Connectivity Help Page For SAP Connectivity tutorials, see the Cloud Connector section.

LESSON SUMMARY

You should now be able to:

- Describe how SAP BTP applications can securely access remote services

Unit 4
Lesson 3

SAP Cloud Connector

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Manage connections between backend systems and SAP BTP

SAP Cloud Connector

Usage Scenario

You want to establish a secure connection between your backend and SAP Business Technology Platform (SAP BTP). You need monitoring, high availability, and full control over the connection.

Learning objectives

- Describe features of the SAP Cloud Connector.
- Describe which systems and landscapes can be connected using the SAP Cloud Connector.
- Describe the bi-directionality of SAP Cloud Connector connections.

SAP Cloud Connector - Basic Concept:

- Serves as a link between SAP BTP applications and on-premise systems.
- Runs as on-premise agent in a secured network.
- Provides fine-grained control over the connectivity.
- Lets you use the features that are required for business-critical enterprise scenarios.

> Note:
> The Cloud Connector must not be used for connectivity other than to the SAP BTP.

More Details about the SAP Cloud Connector, it:

- Serves as a link between SAP BTP applications and on-premise systems.
  - Combines an easy setup with a clear configuration of the systems that are exposed to the SAP BTP.
  - Lets you use existing on-premise assets without exposing the entire internal landscape.
- Runs as on-premise agent in a secured network.

  - Acts as a reverse invoke proxy between the on-premise network and SAP BTP.
- Provides fine-grained control over:
  - On-premise systems and resources that can be accessed by cloud applications.
  - Cloud applications using the Cloud Connector.
- Lets you use the features that are required for business-critical enterprise scenarios.
  - Recovers broken connections automatically.
  - Provides audit logging of inbound traffic and configuration changes.
  - Can be run in a high-availability setup.

Table 6: SAP Cloud Connector Usage Examples

Environments	Examples
Customer on-premise backend systems	SAP ERP, SAP S/4HANA
SAP Hosting	SAP HANA Enterprise Cloud (HEC)
Third-party IaaS providers	AWS, Azure, GCP

Table 7: Additional Usage Examples

Environments	Examples
SAP SaaS solutions	SAP SuccessFactors, SAP Concur, SAP Ariba and more
SAP cloud-based enterprise solutions	SAP S/4HANA Cloud, SAP C/4HANA
Third-party PaaS providers	AWS, Azure, GCP
Third-party SaaS providers	Salesforce

SAP Cloud Connector Scenarios

Watch this video to know more about the SAP Cloud Connector scenarios.

Animation: SAP Cloud Connector Scenarios

For more information on *SAP Cloud Connector Scenarios*, please view the animation in the lesson *SAP Cloud Connector*, in your online course.

Scenario 1: Connecting Cloud Applications to On-Premise Systems

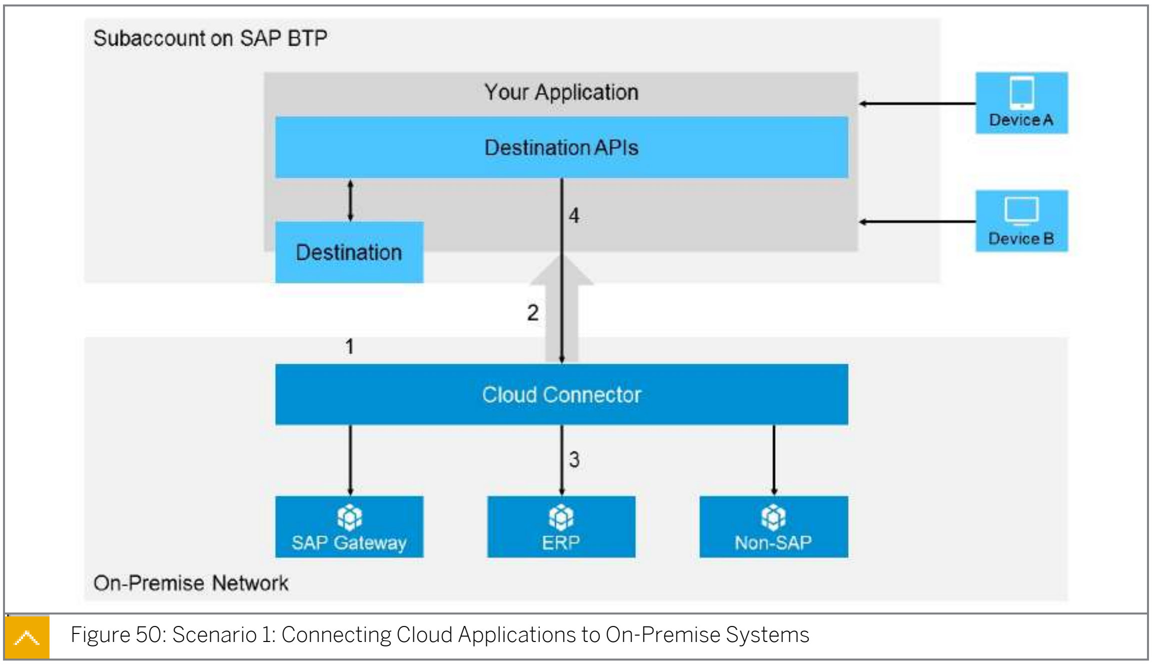


Figure 50: Scenario 1: Connecting Cloud Applications to On-Premise Systems

Data access is based on the SAP BTP - subaccount. Here is the example of the Destination APIs. The tunnel is TSL encrypted. The Cloud Connector then distributes the calls to various connected backend systems. The idea is to provide data from the on-premise network to the SAP BTP.

Scenario 2: Connecting On-Premise Database Tools to SAP HANA Databases

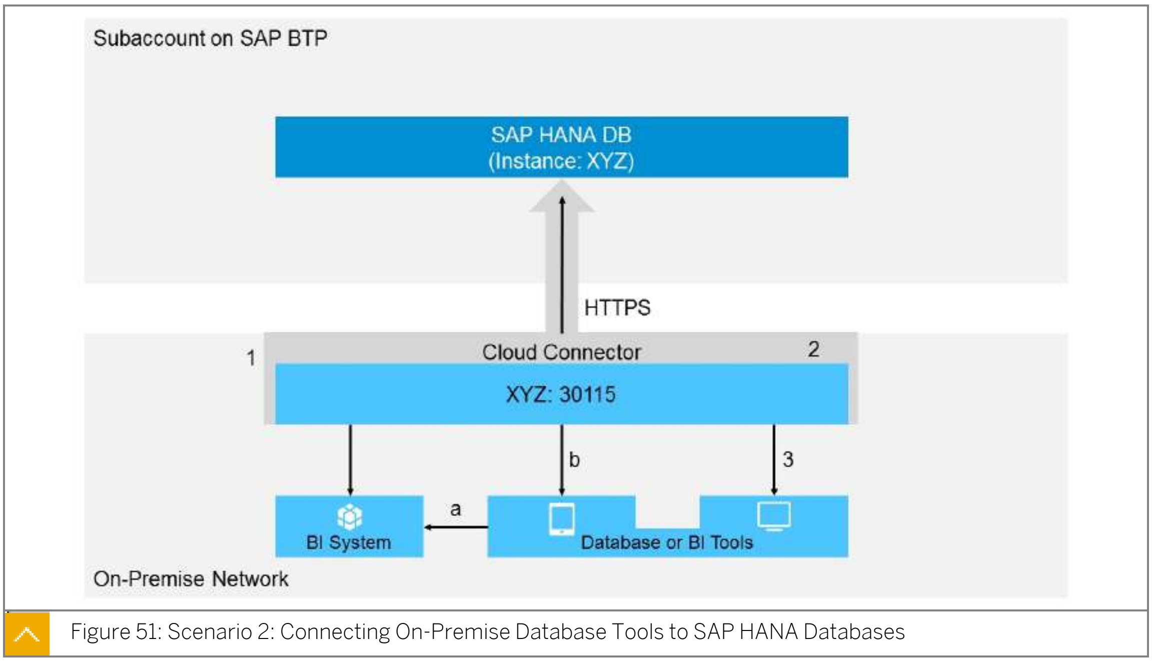


Figure 51: Scenario 2: Connecting On-Premise Database Tools to SAP HANA Databases

In this case, the data access starts from the on-premise database. On the SAP BTP - subaccount you need a SAP HANA service instance. The idea is to make data in the cloud available to the on-premise network.

Features

In addition to the pure connection, other features are available:

- High Availability Setup.
- The Cloud Connector lets you install a redundant (shadow) instance, which monitors the main (master) instance.
- Secure the Activation of Traffic Traces.

  Tracing of network traffic data may contain business critical information or security sensitive data. You can implement a "four-eyes" (double check) principle to protect your traces.
- Monitoring

  Use various views to monitor the activities and state of the Cloud Connector.
- Alerting

  Configure the Cloud Connector to send email alerts whenever critical situations occur that may prevent it from operating.
- Audit Logging

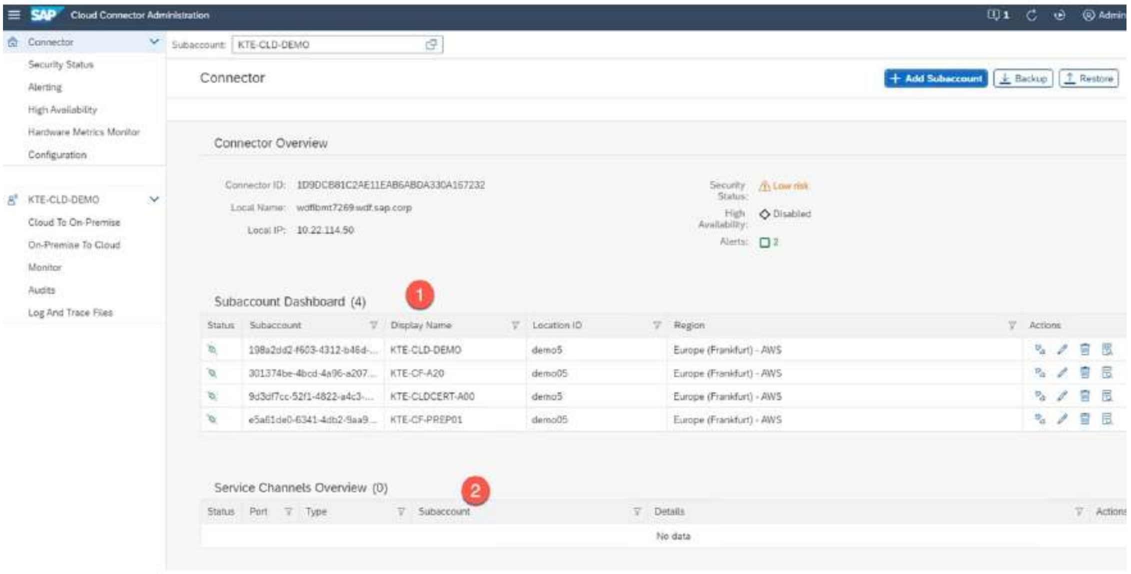
  Use the auditor tool to view and manage audit log information.

Animation:

For more information on , please view the animation in the lesson *SAP Cloud Connector*, in your online course.

Sample

Connections to SAP BTP subaccounts can be created in the administration view (Scenario1) . Under point 2 you can see the opposite direction - Service Channels, if available.



Subaccount: KTE-CLD-DEMO

Connector

Connector Overview

Connector ID: 1D9DCB81C2AE11EAB6ABDA330A157232

Local Name: wdflbmt7269.wdf.sap.corp

Local IP: 10.22.114.50

Security Status: Low risk

High Availability: Disabled

Alerts: 2

Subaccount Dashboard (4)

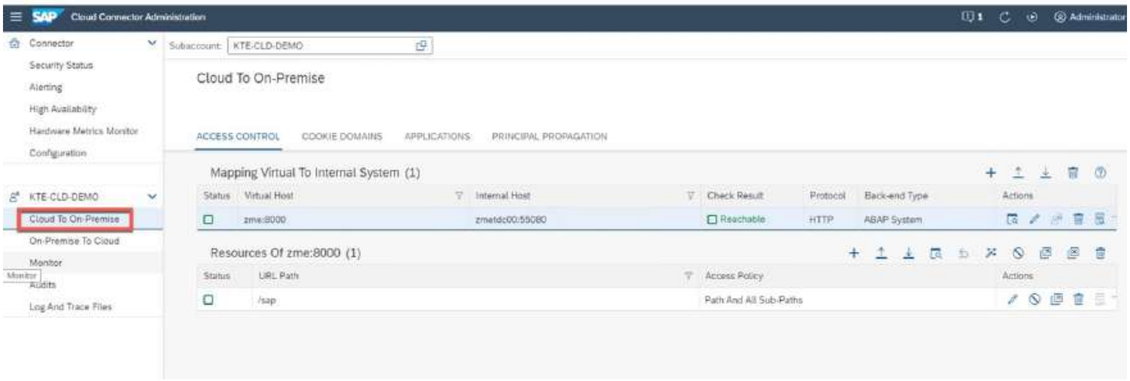
Status	Subaccount	Display Name	Location ID	Region	Actions
	198a2dd2-f603-4312-b46d-...	KTE-CLD-DEMO	demo5	Europe (Frankfurt) - AWS	
	301374be-4bcd-4a96-a207...	KTE-CF-A20	demo05	Europe (Frankfurt) - AWS	
	9d3df7cc-52f1-4822-a4c3-...	KTE-CLDCERT-A00	demo5	Europe (Frankfurt) - AWS	
	e5a61de0-6341-4db2-9aa9...	KTE-CF-PREP01	demo05	Europe (Frankfurt) - AWS	

Service Channels Overview (0)

Status	Port	Type	Subaccount	Details	Actions
				No data	

Figure 52: Administration View

Under the Cloud To On-Premise link you can see the configured backend paths that you can use in a connection.



Subaccount: KTE-CLD-DEMO

Cloud To On-Premise

Mapping Virtual To Internal System (1)

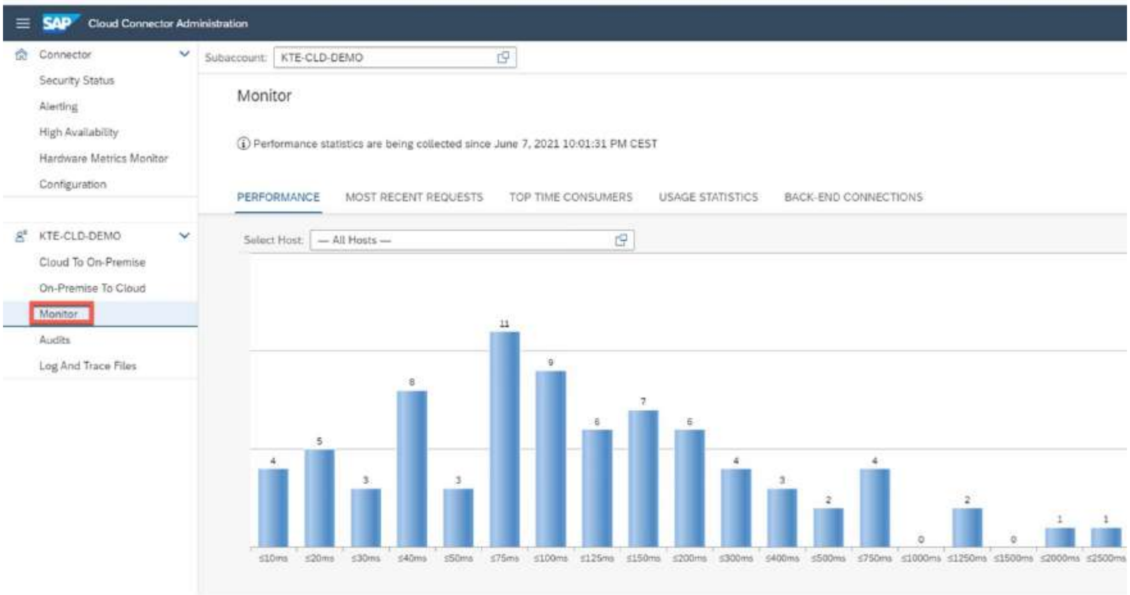
Status	Virtual Host	Internal Host	Check Result	Protocol	Back-end Type	Actions
□	zme:8000	zmetdc00:55080	□ Reachable	HTTP	ABAP System	

Resources Of zme:8000 (1)

Status	URL Path	Access Policy	Actions
□	/sap	Path And All Sub-Paths	

Figure 53: Cloud To On-Premise Link

In the *Monitor* tab you see the amount, size and quantity of data going through the Cloud Connector.



Subaccount: KTE-CLD-DEMO

Monitor

ⓘ Performance statistics are being collected since June 7, 2021 10:01:31 PM CEST

Select Host: — All Hosts —

Figure 54: Monitor Tab

Other functionalities such as High Availability, Alerting, Hardware Metrics Monitor, and more can be found on the left side navigation.

Summary

The Cloud Connector connects SAP backends to SAP BTP subaccounts via an encrypted tunnel. The connection is bidirectional. From an architectural point of view, the Cloud Connector is a possible single point of failure. For this purpose, there is also the possibility of high availability, monitoring, alerting, and more.

Further Reading

Have a look at the [Install the SAP Connectivity Service Cloud Connector in your System Landscape]() tutorial to find out about:

- The available SAP Connectivity Service Cloud Connector versions.
- How to install the SAP Connectivity Service Cloud Connector.
- How to start and stop the SAP Connectivity Service Cloud Connector.
- How to log on to the SAP Connectivity Service Cloud Connector administration UI.
- First steps to secure your installed SAP Connectivity Service Cloud Connector.

LESSON SUMMARY

You should now be able to:

- Manage connections between backend systems and SAP BTP

Learning Assessment

1. What do you need to connect a service to the Sandbox environment of the SAP API Business Hub for the Business Partner API that you want to use?

   Choose the correct answer.

   □ A An API key

   □ B A personal token

2. What does the .env file provide?

   Choose the correct answer.

   □ A Values into the runtime environment of a CAP service

   □ B Values for your version-management-system

3. Which services does SAP BTP Connectivity consist of?

   Choose the correct answers.

   □ A SAP Connectivity service

   □ B OData Connector for SAP Solutions

   □ C SAP Destination service

   □ D XSUAA Connector for SAP Business Technology Platform

4. Which of the following components is required to connect SAP BTP to an on-premise system?

   Choose the correct answer.

   □ A Remote SaaS app

   □ B SAP Gateway

   □ C SAP Cloud Connector

5. What do you use to access the SAP Cloud Connector?

   Choose the correct answer.

   □ A Global account

   □ B Subaccount

   □ C Directory

   □ D Entitlement

6. What encryption type does the SAP Cloud Connector use?

   Choose the correct answer.

   □ A SSL

   □ B SSH

   □ C TLS

   □ D XMPP

7. What are some of the capabilities of the SAP Cloud Connector?

   Choose the correct answers.

   □ A Serves as a link between SAP BTP applications and on-premise systems.

   □ B Runs as on-premise agent in a secured network.

   □ C Manages dev spaces.

   □ D Provides access to the SAP Cloud Identity Services.

   □ E Provides fine-grained control over the connectivity.

UNIT 5 Manual Deployment

Lesson 1

Lesson 2

Lesson 3

UNIT OBJECTIVES

- Perform a manual deployment
- Describe the Cloud Foundry environment
- Use the Cloud Foundry Command Line Interface (CF CLI)

Unit 5
Lesson 1

Deploy Manually

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Perform a manual deployment

Deploy SAP BTP Cloud Foundry Applications Manually: Exercise Overview

Scenario

You will deploy your CAP application into the Cloud Foundry environment of the SAP Business Technology Platform manually using:

- The Cloud Foundry Command Line Interface (CLI)
- A Multi-target application (MTA) file. You will also replace the in-memory SQLite database with an SAP HANA Cloud Trial instance to store the application data.

Task Flow

In this exercise, you will perform the following tasks:

- Prepare a project/extension for SAP HANA Cloud deployment.
- Create an SAP HANA database instance.
- Set up an HDI container.
- Deploy manually using CF CLI tools.
- Deploy manually using a Multi-target application (MTA) file.

Preqrequisites

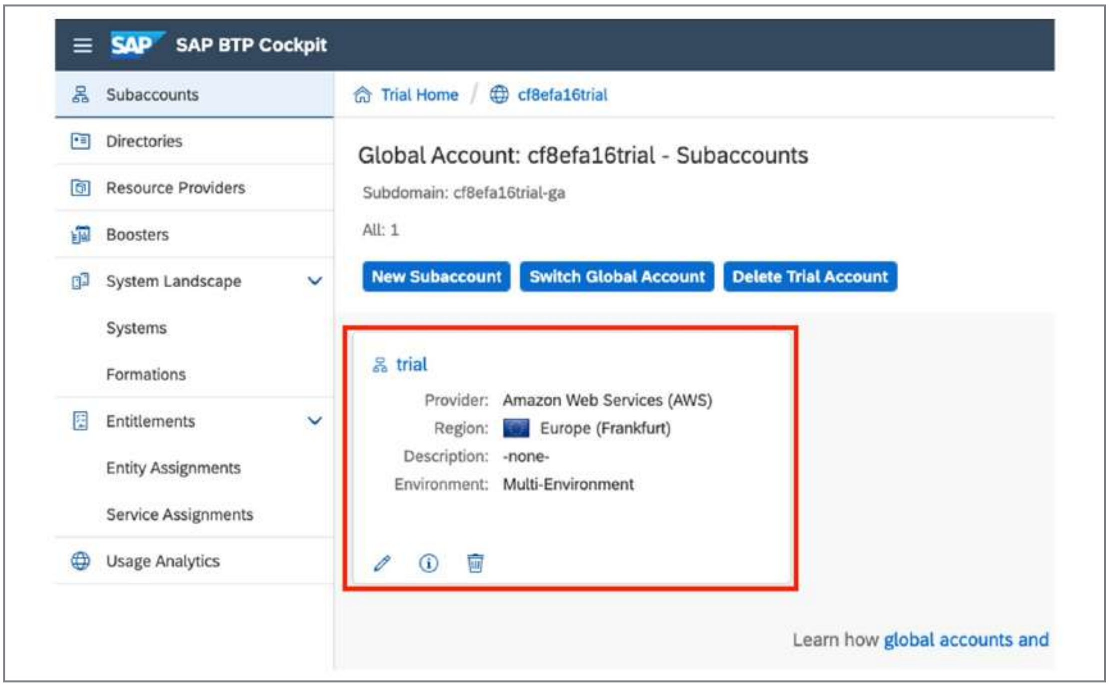
You have added an external service.

Prepare Project for SAP HANA Cloud Deployment

First, you need to set up the SAP HANA Cloud Trial instance in your BTP account.

Set Up SAP HANA Cloud Trial Instance

1. Open your SAP BTP Trial Account Cockpit [1].
2. Enter the subaccount that you are using for this course (for example, the `trial` subaccount).



Global Account: cf8efa16trial - Subaccounts

Subdomain: cf8efa16trial-ga

All: 1

Provider: Amazon Web Services (AWS)

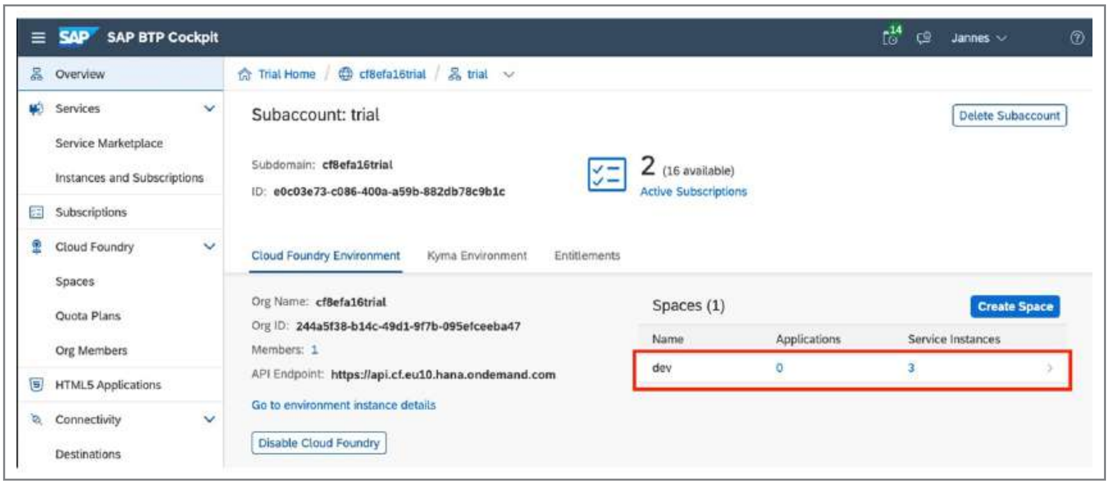
Region: Europe (Frankfurt)

Description: -none-

Environment: Multi-Environment

Learn how global accounts and

3. Enter the space that you are using for this course (for example, the `dev` space).



Subaccount: trial

Subdomain: cf8efa16trial

ID: e0c03e73-c086-400a-a59b-882db78c9b1c

2 (16 available) Active Subscriptions

Org Name: cf8efa16trial

Org ID: 244a5f38-b14c-49d1-9f7b-095efceeba47

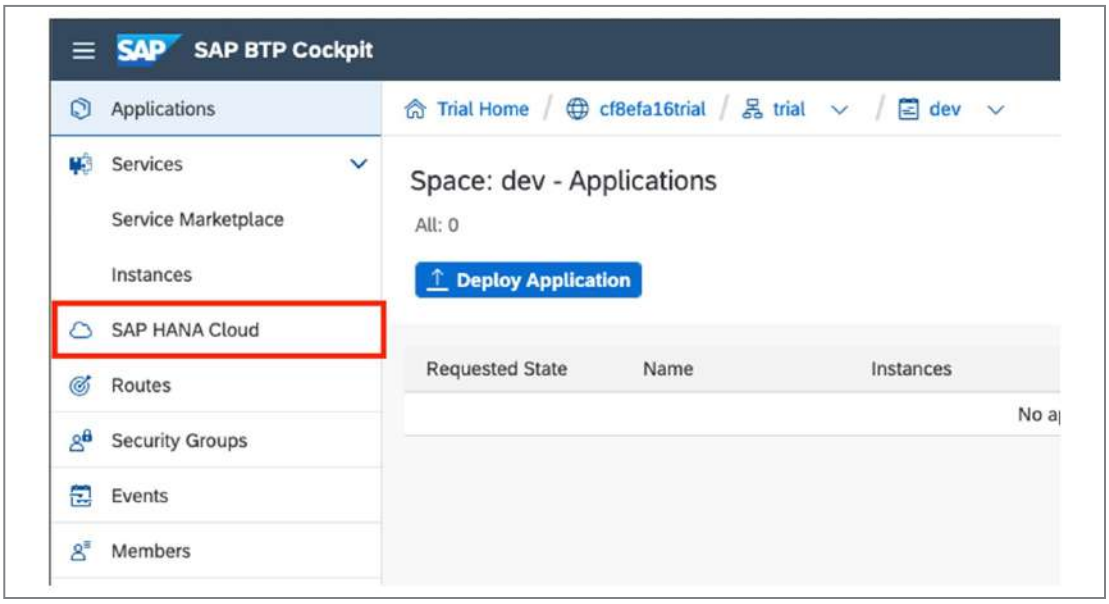
Members: 1

API Endpoint: https://api.cf.eu10.hana.ondemand.com

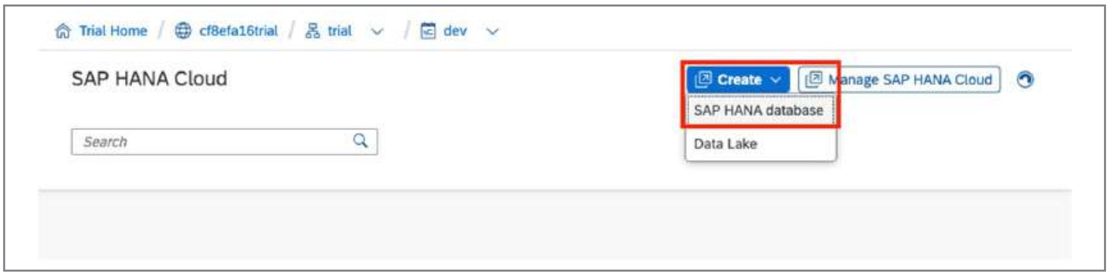
Spaces (1)

Name	Applications	Service Instances
dev	0	3

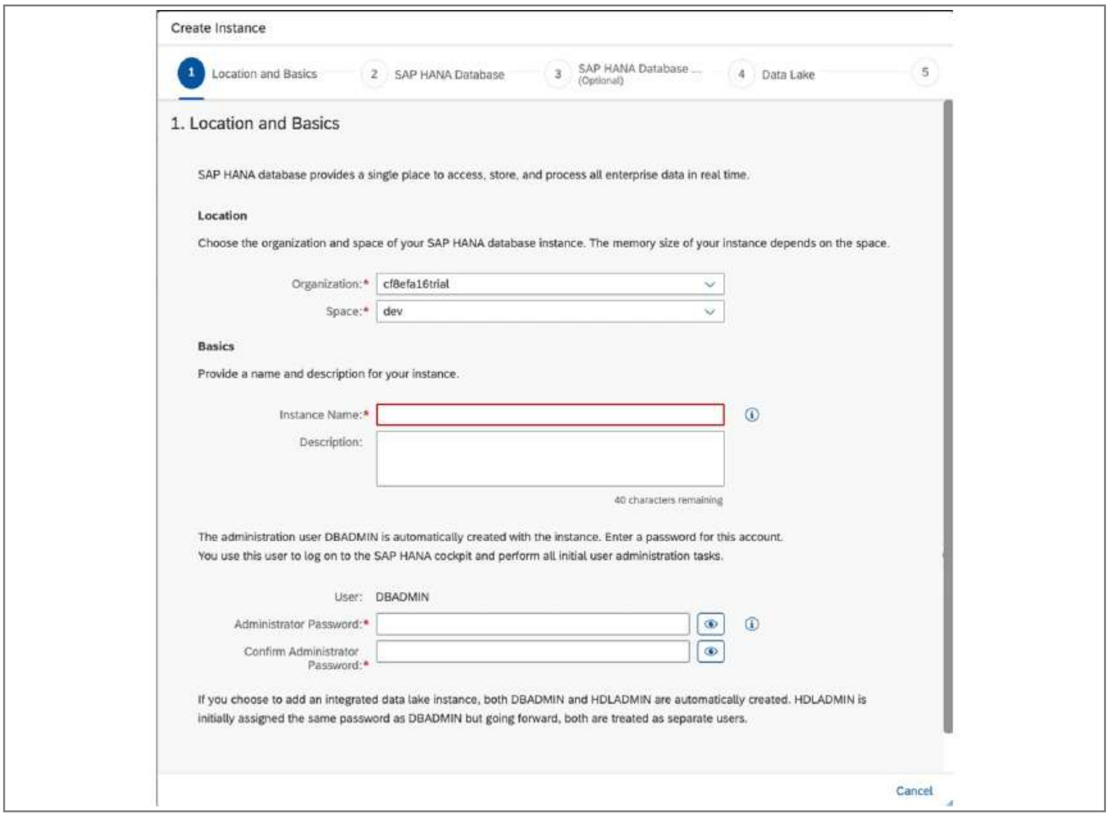
4. In the navigation on the left, choose *SAP HANA Cloud*.



5. Select *Create* and choose *SAP HANA database*.



You will be redirected into SAP HANA Cloud Central for the instance creation. When being asked to authenticate, use your SAP Account user credentials that you used to create your SAP BTP Trial account. You should then see the following dialog:



Create Instance

1. Location and Basics

SAP HANA database provides a single place to access, store, and process all enterprise data in real time.

Location

Choose the organization and space of your SAP HANA database instance. The memory size of your instance depends on the space.

Organization:* cf8efa16trial

Space:* dev

Basics

Provide a name and description for your instance.

Instance Name:*

Description:

40 characters remaining

The administration user DBADMIN is automatically created with the instance. Enter a password for this account. You use this user to log on to the SAP HANA cockpit and perform all initial user administration tasks.

User: DBADMIN

Administrator Password:*

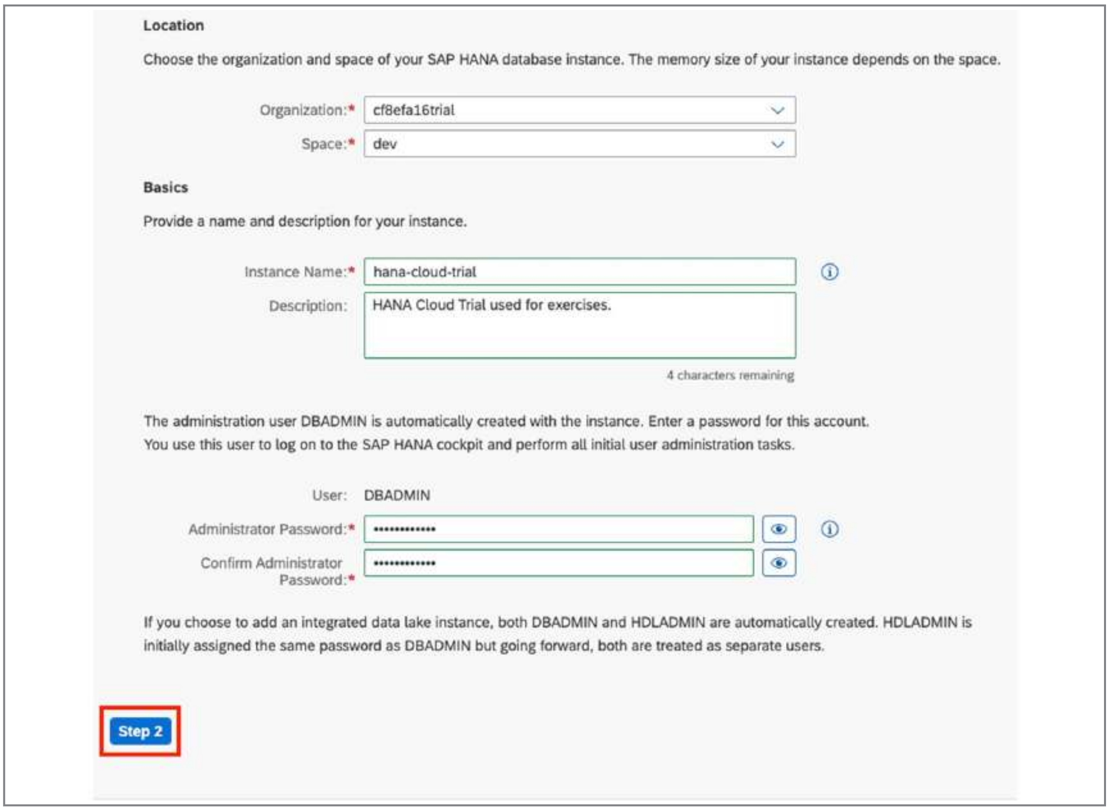
Confirm Administrator Password:*

If you choose to add an integrated data lake instance, both DBADMIN and HDLADMIN are automatically created. HDLADMIN is initially assigned the same password as DBADMIN but going forward, both are treated as separate users.

6. The Location settings should point to your `trial` org and `dev` space. Use the following settings for the remaining fields.

Key	Value
Instance Name	hana-cloud-trial
Description	HANA Cloud Trial used for exercises
Administrator Password	create a password according to the password policy

7. Select the *Step 2* button.



Location

Choose the organization and space of your SAP HANA database instance. The memory size of your instance depends on the space.

Organization:* cf8efa16trial

Space:* dev

Basics

Provide a name and description for your instance.

Instance Name:* hana-cloud-trial

Description: HANA Cloud Trial used for exercises.

4 characters remaining

The administration user DBADMIN is automatically created with the instance. Enter a password for this account.
You use this user to log on to the SAP HANA cockpit and perform all initial user administration tasks.

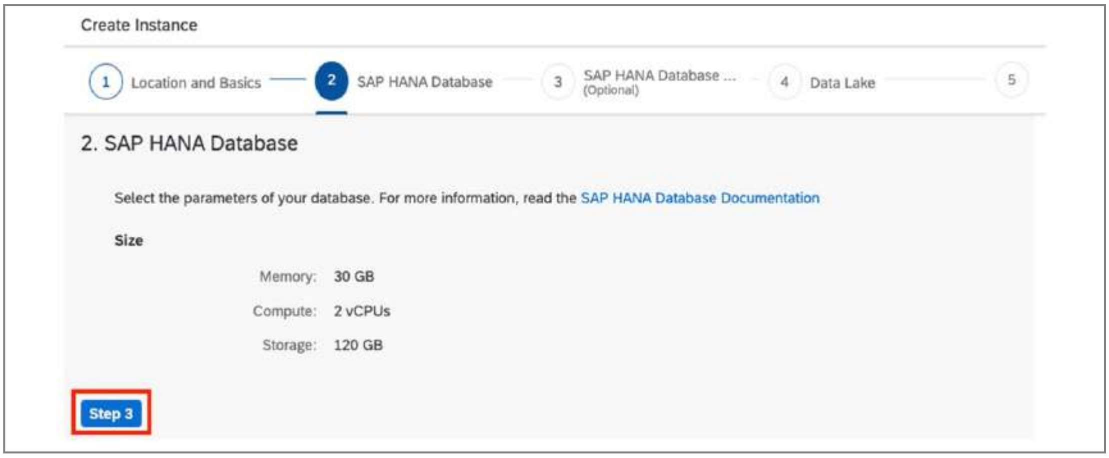
User: DBADMIN

Administrator Password:* ••••••••••••

Confirm Administrator Password:* ••••••••••••

If you choose to add an integrated data lake instance, both DBADMIN and HDLADMIN are automatically created. HDLADMIN is initially assigned the same password as DBADMIN but going forward, both are treated as separate users.

8. The default parameters for your trial SAP HANA Cloud database are displayed. Select the *Step 3* button.



Create Instance

2. SAP HANA Database

Select the parameters of your database. For more information, read the SAP HANA Database Documentation

Size

Memory: 30 GB

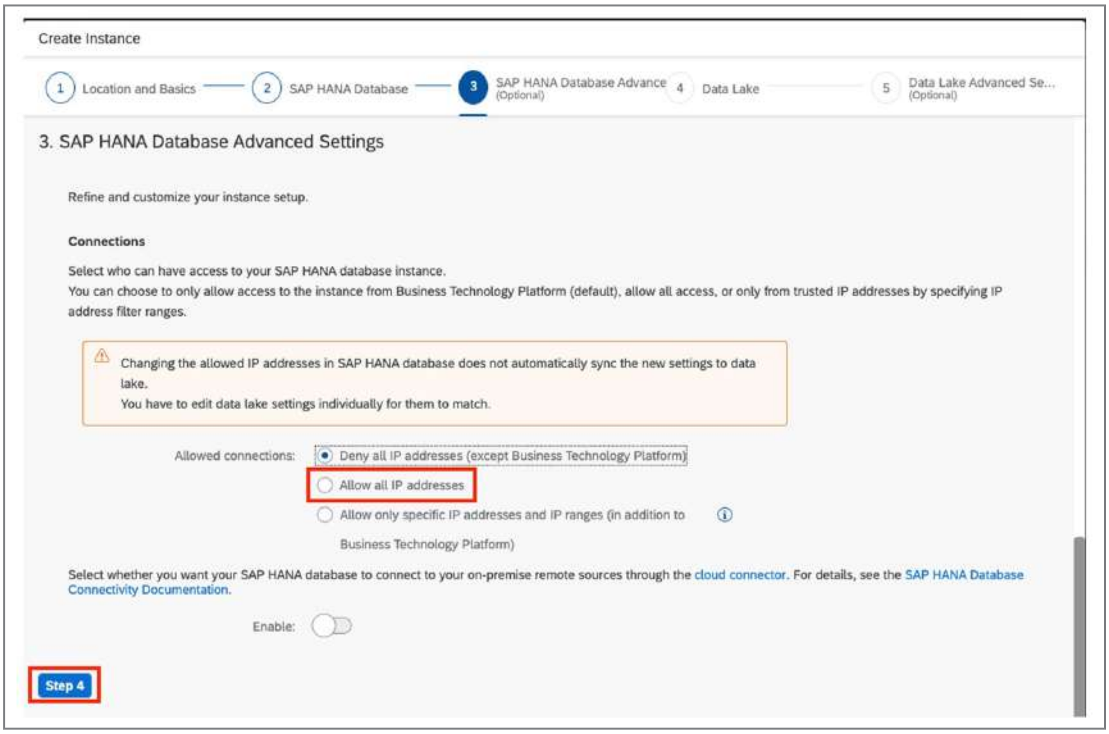
Compute: 2 vCPUs

Storage: 120 GB

9. You will see the *SAP HANA Database Advanced Settings* dialog. Here, you can allow or deny certain IP addresses for connection to your database instance.

   You also find a setting to connect the database to remote sources via the Cloud Connector. This is especially useful in replication use cases, where you need to replicate data from on-premise sources to your SAP HANA Cloud instance. In our use case, this setting is not required.

10. Select the *Allow all IP addresses* option, then select the *Step 4* button.



Create Instance

1 Location and Basics
2 SAP HANA Database
3 SAP HANA Database Advance (Optional)
4 Data Lake
5 Data Lake Advanced Se... (Optional)

3. SAP HANA Database Advanced Settings

Refine and customize your instance setup.

Connections

Select who can have access to your SAP HANA database instance.
You can choose to only allow access to the instance from Business Technology Platform (default), allow all access, or only from trusted IP addresses by specifying IP address filter ranges.

⚠ Changing the allowed IP addresses in SAP HANA database does not automatically sync the new settings to data lake.
You have to edit data lake settings individually for them to match.

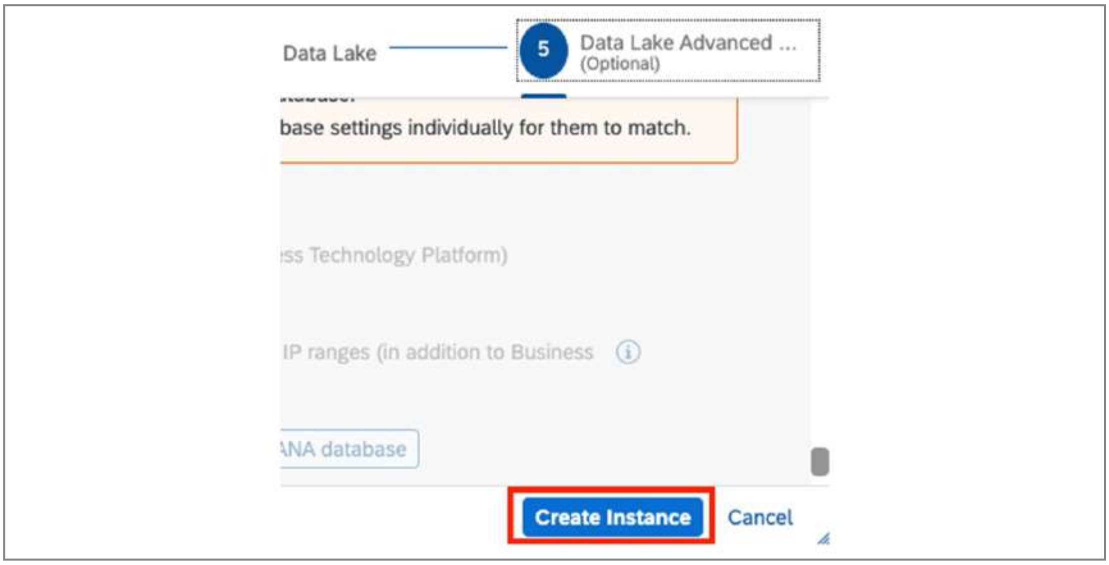
Allowed connections: ● Deny all IP addresses (except Business Technology Platform)
○ Allow all IP addresses
○ Allow only specific IP addresses and IP ranges (in addition to Business Technology Platform) ⓘ

Select whether you want your SAP HANA database to connect to your on-premise remote sources through the cloud connector. For details, see the SAP HANA Database Connectivity Documentation.

Enable: ○

11. The *Data Lake* page appears. Accept the default values and select the *Step 5* button.

12. The *Data Lake Advanced Settings* page appears. Select the *Create Instance* button.



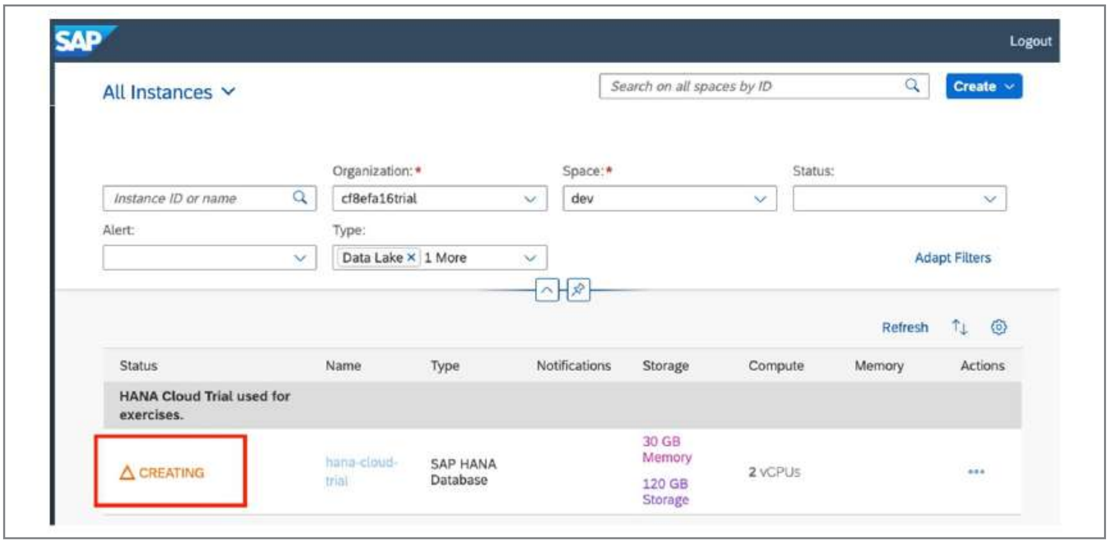
base settings individually for them to match.

ss Technology Platform)

IP ranges (in addition to Business ⓘ

NA database

13. The overview of your database instances appears. There should be a new entry for your hana-cloud-trial instance with *Status* `CREATING`.



Organization: * cf8efa16trial

Space: * dev

Status:

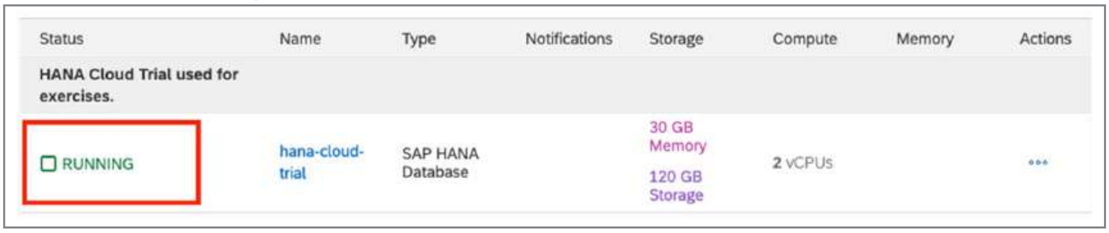
Instance ID or name

Alert:

Type: Data Lake × 1 More

Status	Name	Type	Notifications	Storage	Compute	Memory	Actions
HANA Cloud Trial used for exercises.							
△ CREATING	hana-cloud-trial	SAP HANA Database		30 GB Memory 120 GB Storage	2 vCPUs		•••

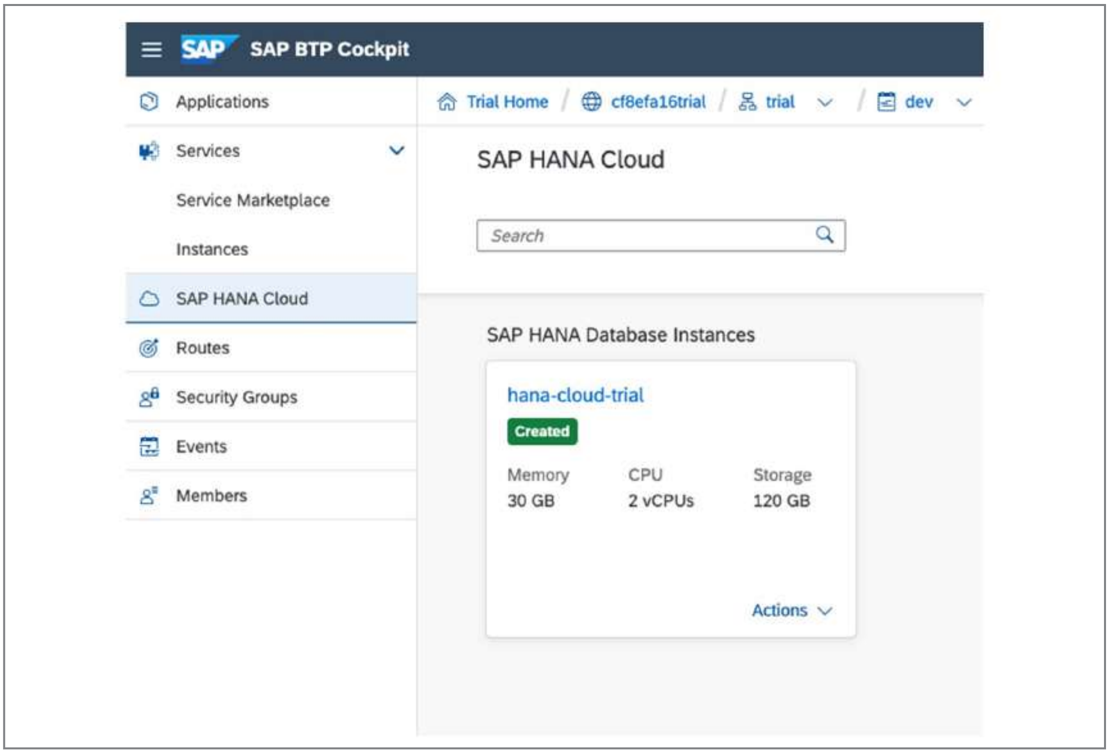
The process will take several minutes to complete. Once done, the *Status* should switch to `RUNNING`. Use the *Refresh* button to update the status. Your SAP HANA Cloud Trial instance is now ready to be used.



Status	Name	Type	Notifications	Storage	Compute	Memory	Actions
HANA Cloud Trial used for exercises.							
□ RUNNING	hana-cloud-trial	SAP HANA Database		30 GB Memory 120 GB Storage	2 vCPUs		•••

14. Return to your BTP trial account and select SAP HANA Cloud.

15. Your hana-cloud-trial instance will appear in the SAP HANA Database instances overview.



SAP HANA Database Instances

hana-cloud-trial

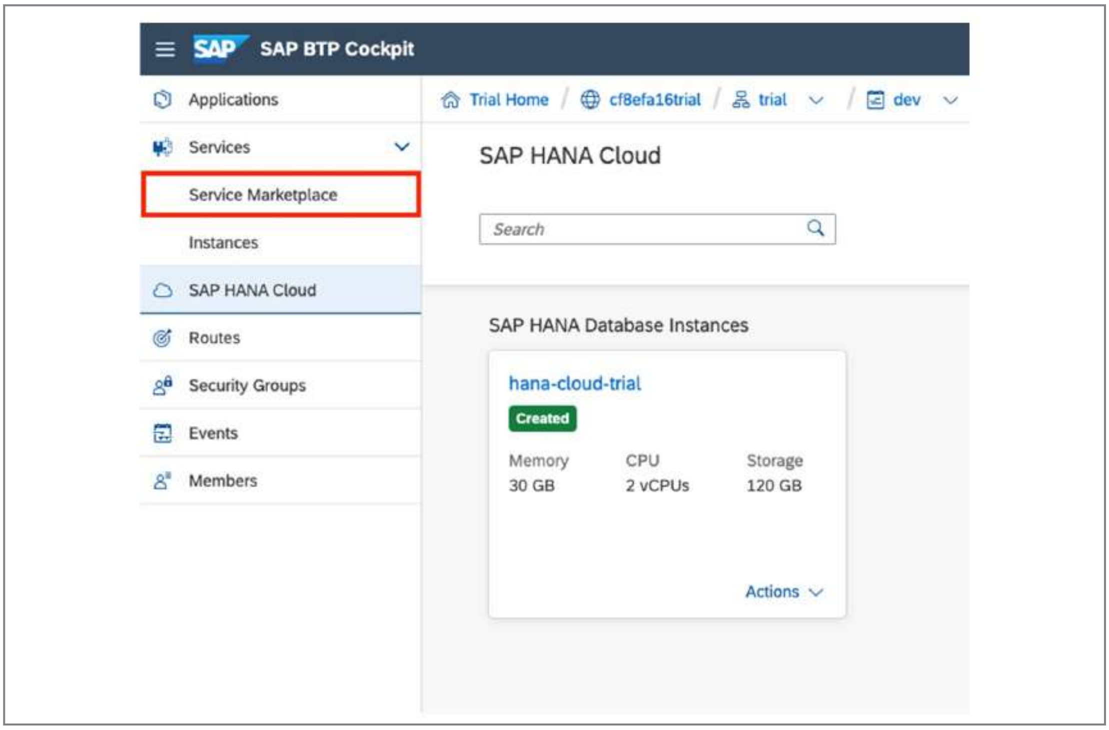
Created

Memory	CPU	Storage
30 GB	2 vCPUs	120 GB

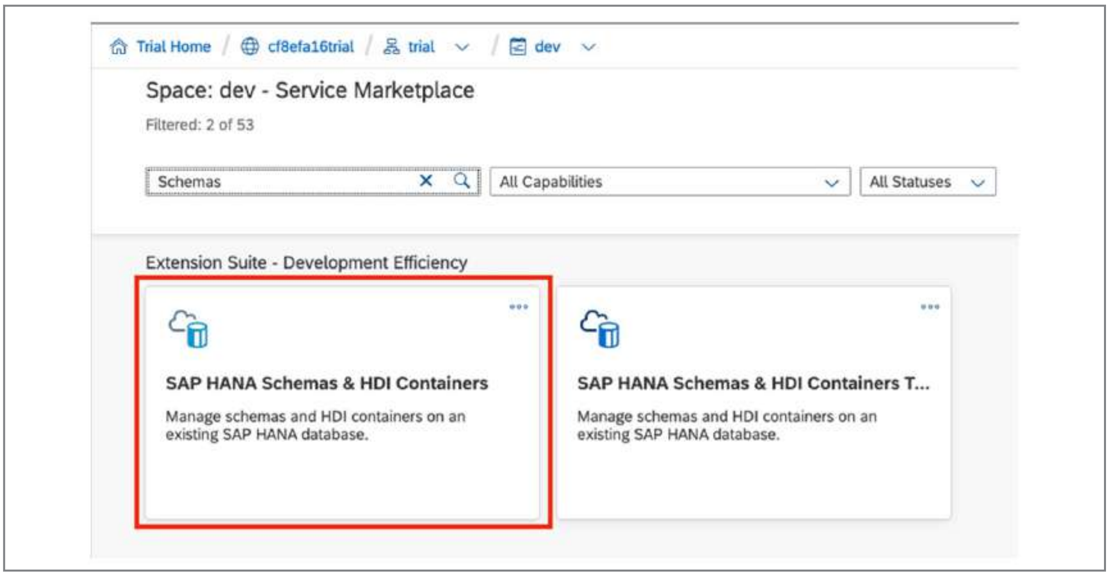
Set Up HDI Container on SAP HANA Cloud Instance

The SAP HANA Cloud instance serves as your cloud database. You can now create schemas or HDI containers on the database. You can bind a schema or HDI container to an application to create the data schema and store the application data.

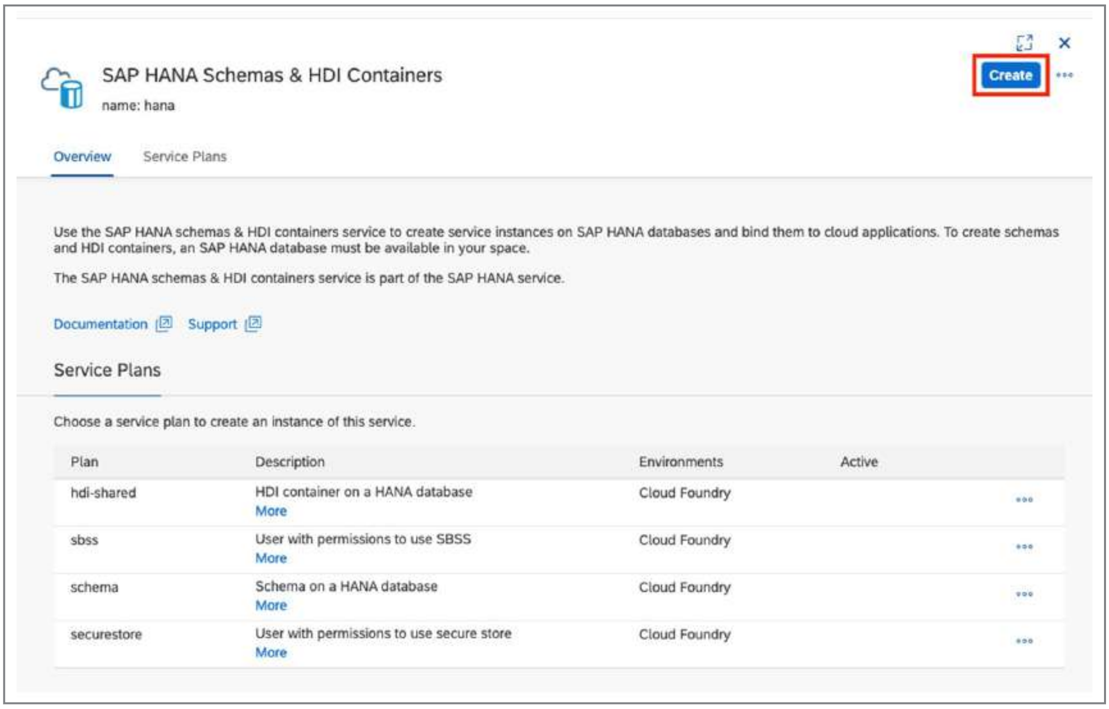
1. In your BTP cockpit, select Service Marketplace on the left.



2. The Service Marketplace displays. Type **`schemas`** in the *Search* field and then select the *SAP HANA Schemas & HDI Containers* tile.


Trial Home / cf8efa16trial / trial / dev
Space: dev - Service Marketplace
Filtered: 2 of 53
Schemas
All Capabilities
All Statuses
Extension Suite - Development Efficiency
SAP HANA Schemas & HDI Containers
Manage schemas and HDI containers on an existing SAP HANA database.
SAP HANA Schemas & HDI Containers T...
Manage schemas and HDI containers on an existing SAP HANA database.

3. Select the *Create* button.



SAP HANA Schemas & HDI Containers

name: hana

Use the SAP HANA schemas & HDI containers service to create service instances on SAP HANA databases and bind them to cloud applications. To create schemas and HDI containers, an SAP HANA database must be available in your space.

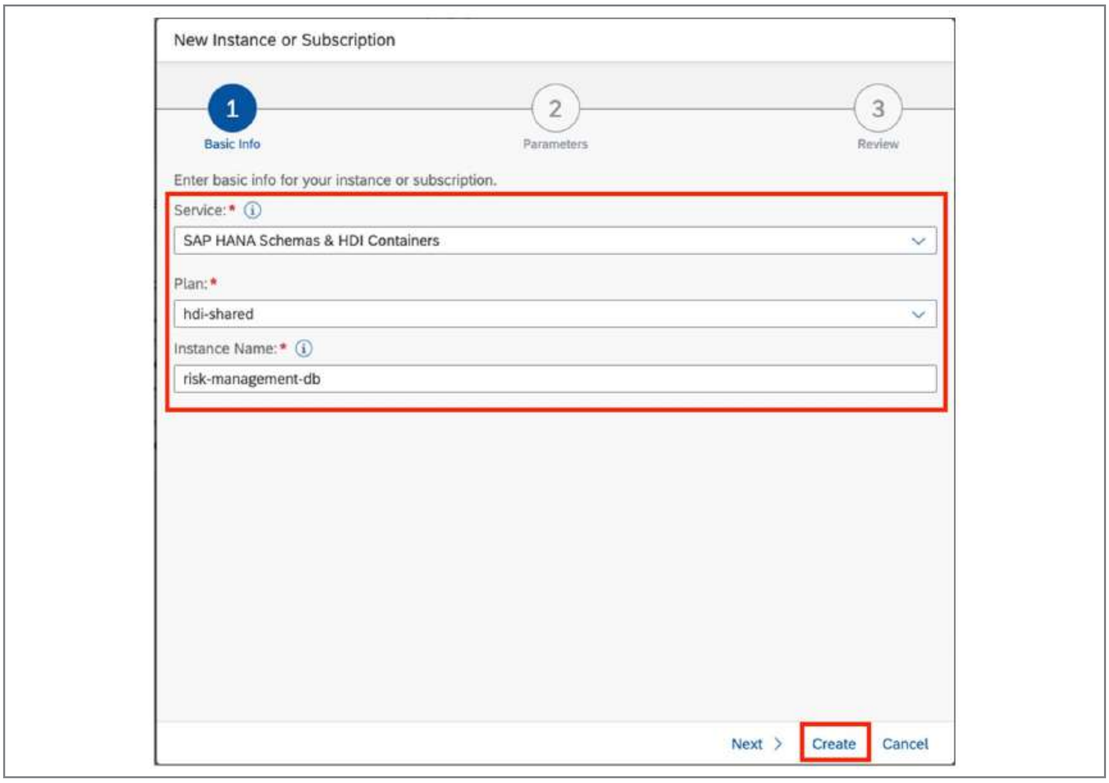
The SAP HANA schemas & HDI containers service is part of the SAP HANA service.

Service Plans

Choose a service plan to create an instance of this service.

Plan	Description	Environments	Active
hdi-shared	HDI container on a HANA database More	Cloud Foundry	
sbss	User with permissions to use SBSS More	Cloud Foundry	
schema	Schema on a HANA database More	Cloud Foundry	
securestore	User with permissions to use secure store More	Cloud Foundry	

4. The *New Instance or Subscription* page appears. Set the *Plan* to **`hdi-shared`**, enter **`risk-management-db`** in the *Instance Name* field, and then select *Create*.



New Instance or Subscription

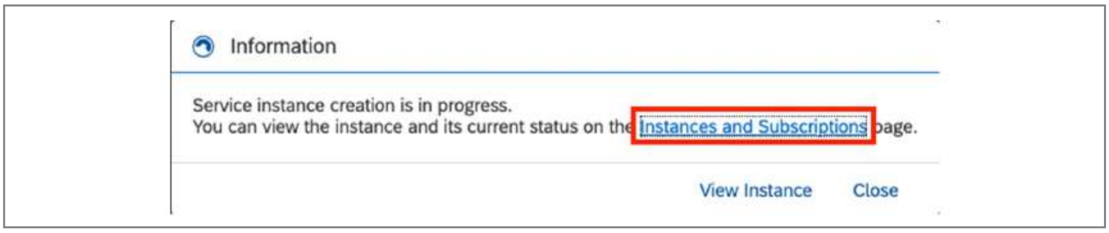
Enter basic info for your instance or subscription.

Service:* SAP HANA Schemas & HDI Containers

Plan:* hdi-shared

Instance Name:* risk-management-db

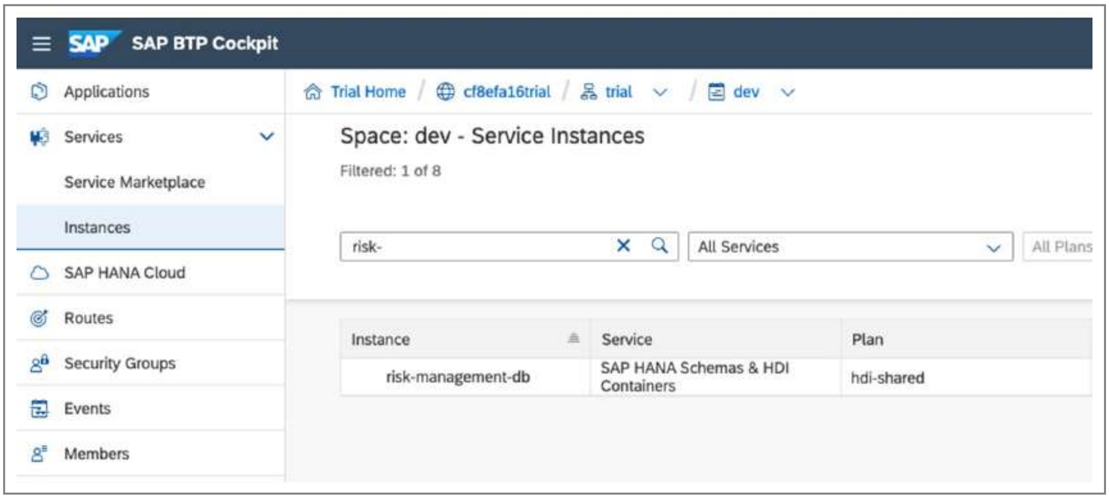
5. In the *Information* dialog that appears, select *Instances and Subscriptions*.



Information

Service instance creation is in progress.
You can view the instance and its current status on the Instances and Subscriptions page.

6. Your `risk-management-db` instance of the *SAP HANA Schemas & HDI Containers* service appears in the list. This service instance is the database container that the CAP data schema will be deployed to.



Space: dev - Service Instances

Filtered: 1 of 8

Instance	Service	Plan
risk-management-db	SAP HANA Schemas & HDI Containers	hdi-shared

Update project to use SAP HANA

As you want your CAP service to use the newly created SAP HANA Cloud HDI container, you need to update the project.

1. Run the command `cds add hana`.

2. The projects `package.json` will be updated with the required configurations.

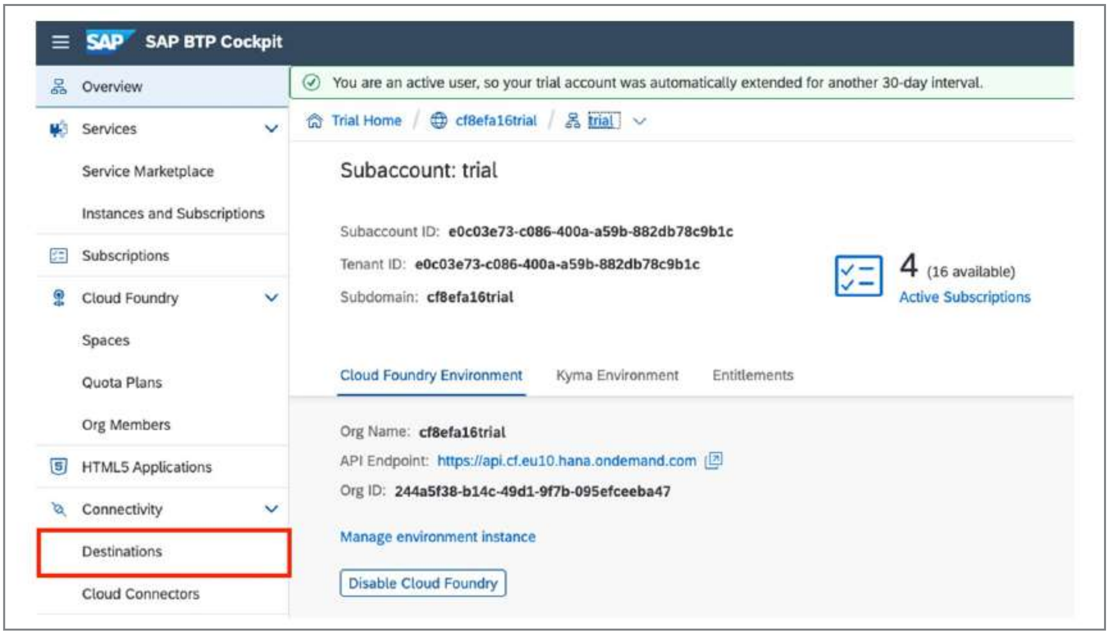
   The following parts are added to the `cds` section of the file:

```
{
    ...
    ...
    "cds": {
       "requires": {
          ...
          ...
          "db": {
             "kind": "sql"
          }
     },
     "hana": {
         "deploy-format": "hdbtable"
     }
   }
}
```

   This configures deployment for SAP HANA to use the *hdbtable* and *hdbview* formats. They are special file formats that are generated during the build process from the `.cds.` definition files and can be deployed to SAP HANA Cloud, in order to create the data schema.

Further Deployment Preparations

1. Go to the SAP BTP Cockpit, subaccount level (`trial`) and select *Destinations* in the *Connectivity* section of the navigational menu.



You are an active user, so your trial account was automatically extended for another 30-day interval.

Subaccount: trial

Subaccount ID: e0c03e73-c086-400a-a59b-882db78c9b1c

Tenant ID: e0c03e73-c086-400a-a59b-882db78c9b1c

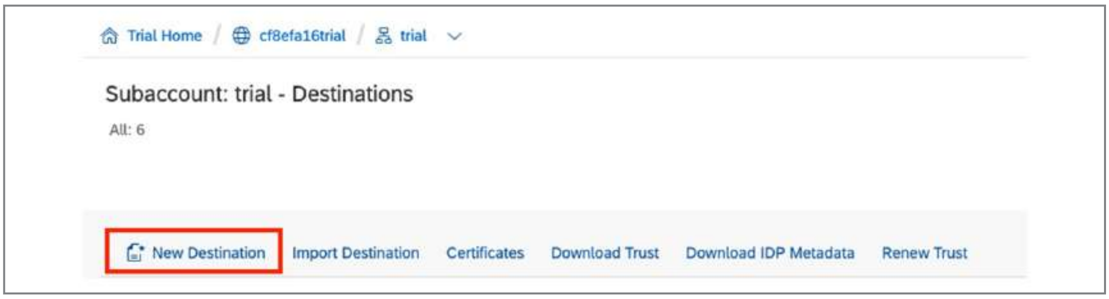
Subdomain: cf8efa16trial

Org Name: cf8efa16trial

API Endpoint: https://api.cf.eu10.hana.ondemand.com

Org ID: 244a5f38-b14c-49d1-9f7b-095efceeba47

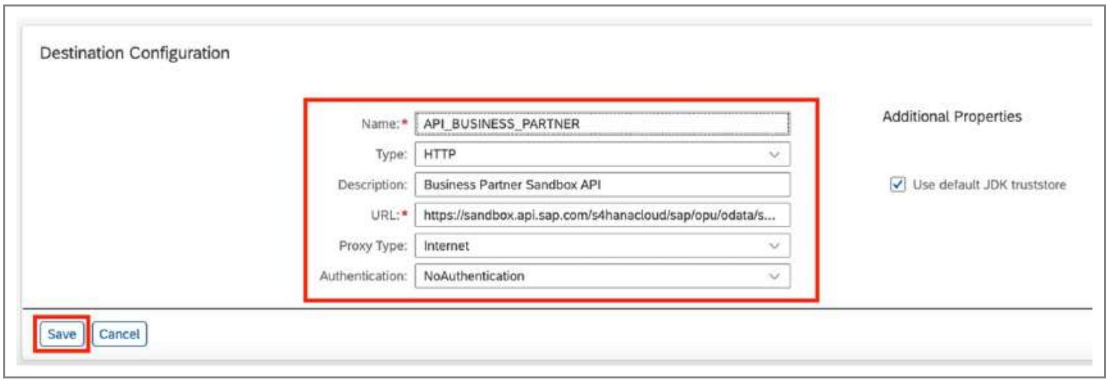
2. Choose *New Destination*.



3. Enter the following values into the form:

Key	Value
Name	API_BUSINESS_PARTNER
Type	HTTP
Description	Business Partner Sandbox API
URL	https://sandbox.api.sap.com/s4hanacloud/sap/opu/odata/sap/API_BUSINESS_PARTNER/
Proxy Type	Internet
Authentication	No Authentication

4. When finished, press *Save*.



Destination Configuration

Name:* API_BUSINESS_PARTNER

Type: HTTP

Description: Business Partner Sandbox API

URL:* https://sandbox.api.sap.com/s4hanacloud/sap/opu/odata/s...

Proxy Type: Internet

Authentication: NoAuthentication

Additional Properties

☑ Use default JDK truststore

5. The destination needs to be accessed from your CAP service. For that, you need to create a service instance of the destination service and adjust the configuration of your `package.json` file.

6. Select *Instances and Subscriptions* and then select *Create*.

7. In the dialog, use the following settings:

Key	Value
Service	Destination
Plan	Lite
Runtime Environment	Cloud Foundry
Space	your space, for example, `dev`
Instance Name	risk-management-destination-service

8. Select *Create*. The destination service instance will be created.

9. Still being in the *Instances and Subscriptions* section, select *Create* again.

10. In the dialog, use the following settings:

Key	Value
Service	Authorization & Trust Management (xsuaa)
Plan	application
Runtime Environment	Cloud Foundry
Space	your space, for example, `dev`
Instance Name	risk-management-xsuaa

11. Select *Next*. You are required to provide a configuration.

12. Go to your BAS project and run the following command: `cds compile srv/ --to xsuaa > xs-security.json`

    This will generate the file `xs-security.json` in your project `root` folder.

13. Open the file and change the xsappname-value to `risk-management-<YOURDEVSPACE>`:

    Make sure to replace `<YOURDEVSPACE>` with your dev-space name. It should then look like this for instance:

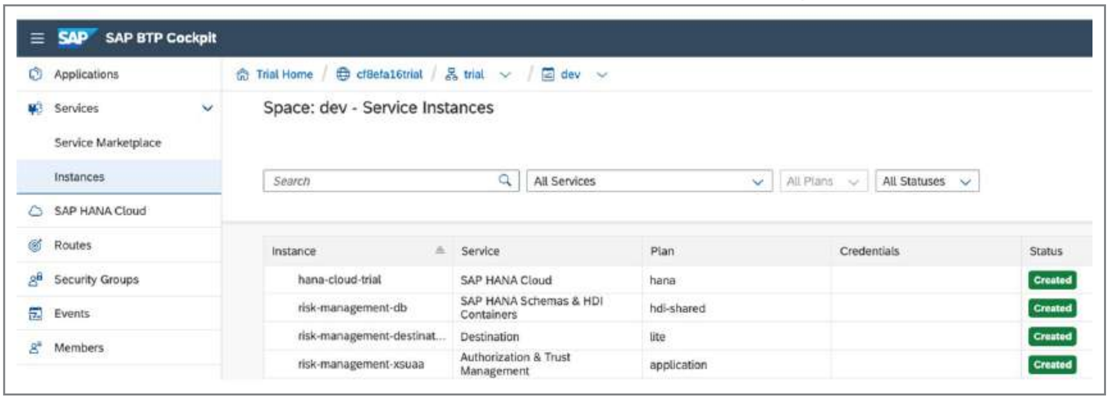
```
"xsappname": "risk-management-dev",
```

14. Copy the `JSON`.

15. Go back to the *SAP BTP Cockpit* dialog for the xsuaa instance creation and paste the copied `JSON` structure.

16. Select *Create*. An instance of the xsuaa service will be created.

17. Your Service Instances overview should now look like the following:



Space: dev - Service Instances

Instance	Service	Plan	Credentials	Status
hana-cloud-trial	SAP HANA Cloud	hana		Created
risk-management-db	SAP HANA Schemas & HDI Containers	hdi-shared		Created
risk-management-destinat...	Destination	lite		Created
risk-management-xsuaa	Authorization & Trust Management	application		Created

Manual Deployment using CF CLI Tools

1. In the `package.json` file, in the `requires` section, add the following code:

```
"xsuaa": {
    "kind": "xsuaa"
}
```

The result should look like the following:

```
{
    ......"cds": {
        "requires": {
            "API_BUSINESS_PARTNER": {
                "kind": "odata",
                "model": "srv/external/API_BUSINESS_PARTNER",
                "credentials": {
                    "url": "https://sandbox.api.sap.com/
s4hanacloud/sap/opu/odata/sap/API_BUSINESS_PARTNER/"
                }
            },
            "db": {
                "kind": "sql"
            },
            //##BEGINOFINSERT"xsuaa": {
                "kind": "xsuaa"
            }
            //##ENDOFINSERT
        },
        "hana": {
            "deploy-format": "hdbtable"
        }
    }
}
```

2. Run the command `cds add cf-manifest` to create the deployment manifest files.

This command creates the files `manifest.yml` and `services-manifest.yml` in your project `root`. You are going to adjust the `manifest.yml` file in the next steps. The `services-manifest.yml` file contains the Cloud Foundry services, that are required for your project and are derived from the service bindings in `package.json` using the `cds.requires` configuration.

3. Take a look at the new `manifest.yml` file in your project `root`. It shows your application components and their services dependencies (`services` sections).

4. For the `risk-management-srv` application, add the following line into the `services` section: `- risk-management-destination-service`

   The resulting `risk-management-srv` application section should look like the following:

```
- name: risk-management-srv
  random-route: true # for development only
  path: gen/srv
  memory: 256M
  buildpack: nodejs_buildpack
  services:
    - risk-management-db
    - risk-management-xsuaa
    - risk-management-destination-service
```

   You have now defined that your application requires the destination service instance that you created earlier.

5. In the `package.json` file in your project `root`, change the following lines by deleting or inserting code as indicated:

```
{
    ......"cds": {
        "requires": {
            "API_BUSINESS_PARTNER": {
                "kind": "odata",
                "model": "srv/external/API_BUSINESS_PARTNER",
                //###BEGINOFDELETE"credentials": {
                    "url": "https://sandbox.api.sap.com/
s4hanacloud/sap/opu/odata/sap/API_BUSINESS_PARTNER/"
                },
                //###ENDOFDELETE//###BEGINOFINSERT"[development]": {
                    "credentials": {
                        "url": "https://sandbox.api.sap.com/
s4hanacloud/sap/opu/odata/sap/API_BUSINESS_PARTNER/"
                    }
                },
                "[production]": {
                    "credentials": {
                        "destination": "API_BUSINESS_PARTNER"
                    }
                }
                //###ENDOFINSERT
            },
            "db": {
                "kind": "sql"
            },
            "xsuaa": {
                "kind": "xsuaa"
            }
        },
        "hana": {
            "deploy-format": "hdbtable"
            }
      }
}
```

Here you tell the service to distinguish between the `development` and the `production` scenarios. When in development, the API URL is retrieved directly from the `package.json`. In a production scenario, the URL should be retrieved from a destination called `API_BUSINESS_PARTNER`.

6. Run the command `cds build --production` to generate the source files required for production deployment.

7. Run the command `cf login` to login to the CF environment.

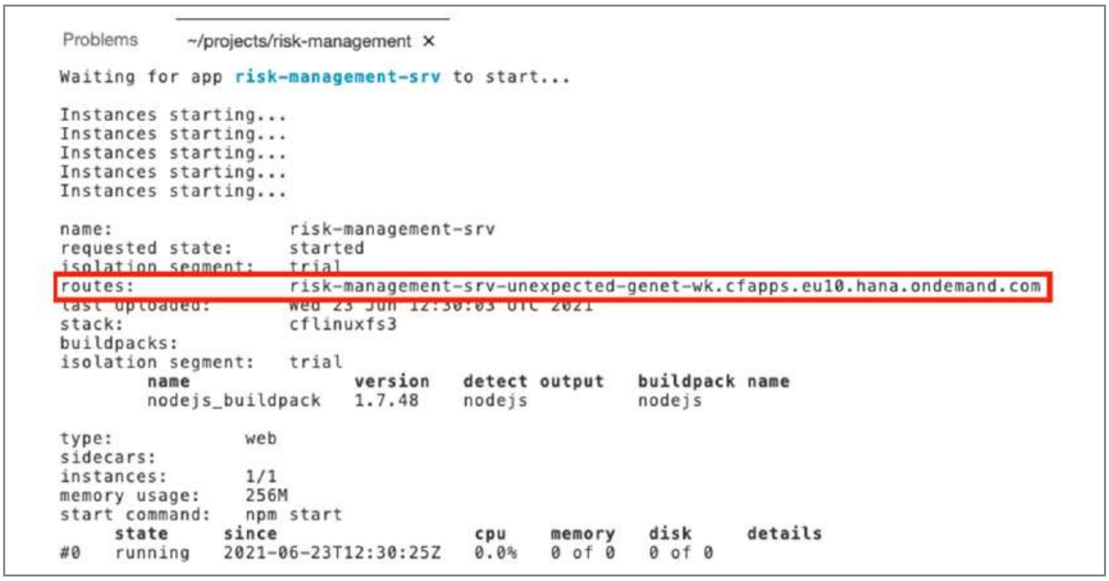
   Use the `API-endpoint` from the BTP cockpit. Use your trial user and password and select the org and space where you want to deploy the application to.

8. Run the command `cf push` within the `root` folder of your BAS project. The database module and the service module defined in the `manifest.yml` will be deployed into your CF account.

   The deployment takes a while. You can follow the deployment activity in the terminal log.

> Note:
> The command `cf create-service-push` also exists, which in addition to deploying the application modules in `manifest.yml` creates or updates the service instances of the services defined in the `services-manifest.yml` file. In this case, as you have created the required service instances manually before, you do not need to use the command.

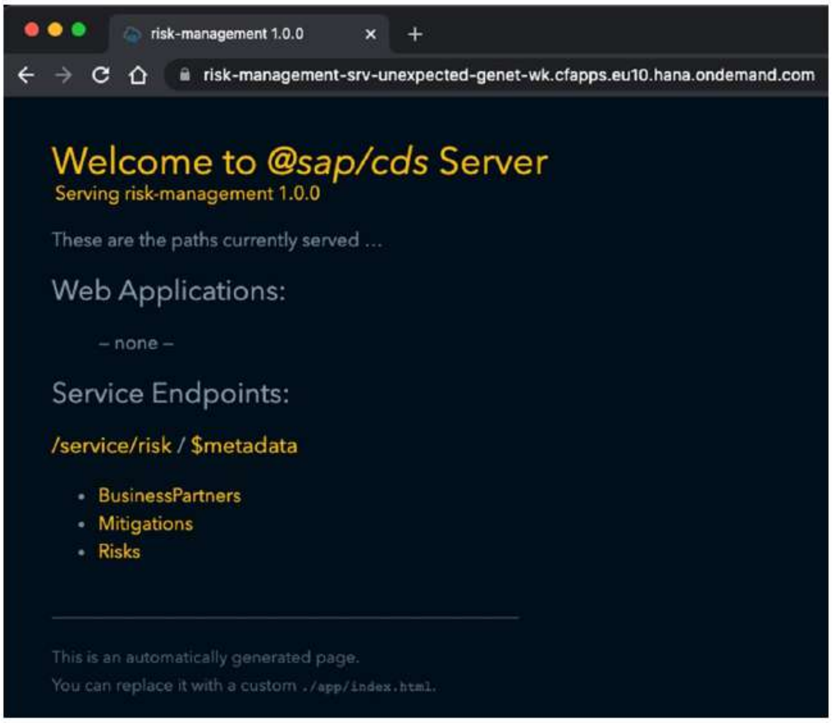
9. When the deployment is complete, check the logs for a section where module `risk-management-srv` is being started. Within that section, you find a route, which represents the service URL:



```
Waiting for app risk-management-srv to start...

Instances starting...
Instances starting...
Instances starting...
Instances starting...
Instances starting...

name:                 risk-management-srv
requested state:      started
isolation segment:    trial
routes:               risk-management-srv-unexpected-genet-wk.cfapps.eu10.hana.ondemand.com
last uploaded:        Wed 23 Jun 12:30:03 UTC 2021
stack:                cflinuxfs3
buildpacks:
isolation segment:    trial
        name                version   detect output   buildpack name
        nodejs_buildpack    1.7.48    nodejs          nodejs

type:            web
sidecars:
instances:       1/1
memory usage:    256M
start command:   npm start
     state     since                  cpu    memory   disk     details
#0   running   2021-06-23T12:30:25Z   0.0%   0 of 0   0 of 0
```

This is the URL for your CAP service that was generated during the deployment. Your service is running on the SAP Business Technology Platform and is publicly available via the Internet using this URL.

10. Copy the URL from your deployment log and open it in a new browser tab.


risk-management 1.0.0
risk-management-srv-unexpected-genet-wk.cfapps.eu10.hana.ondemand.com

Welcome to @sap/cds Server

Serving risk-management 1.0.0

These are the paths currently served ...

Web Applications:

– none –

Service Endpoints:

/service/risk / $metadata

- BusinessPartners
- Mitigations
- Risks

This is an automatically generated page.
You can replace it with a custom ./app/index.html.

Try to open the BusinessPartners endpoint. It should result in an error. The reason is that your service currently cannot access the SAP API Business Hub sandbox environment because it is missing the API key. Previously, during local testing, you provided it via the .env file. This is inaccessible in the BTP environment.

11. You must set the API key as an environment variable apikey for the productive Cloud Foundry environment. You need to do this manually and just once, by running the following command in your terminal:`cf set-env risk-management-srv apikey <your-api-key>`.

    Make sure, that you replace `<your-api-key>` with your API key from the API Business Hub.

    With the following command:`cf env risk-management-srv` you can also verify that the key has been set as *user-provided variable*.

12. Run the following command: `cf restart risk-management-srv`

    This ensures that your environment variable change takes effect in the application.

13. Open the browser tab with the running service. The BusinessPartners endpoint should now provide the data from the external service, the SAP API Business Hub sandbox environment.

    Try to open the other service endpoints. The corresponding data should appear. Just now, it is not stored in an SQLite database, but in your SAP HANA Cloud trial HDI Container.

    You will notice, that the web application (the UI part) is missing. This was left out on purpose. You are going to add it to your BTP deployment in a later step, because it requires an additional component, the so-called application router.

Manual Deployment Using a "Multi Target Application" (MTA) File

In this section, we will create a "Multi Target Application" (MTA) file for deployment. (See the description [2]). MTA is a way to create deployments consisting of multiple modules that can be implemented in different technologies. Some advantages of this technology are:

- It provides a build tool.
- It creates service instances, service keys, and destinations automatically.
- It deploys content (HTML5, workflow, ...).
- It supports blue-green deployment [3].

Compared to the deployment procedure you performed in the previous steps, this will save you a lot of manual work in the long run.

Generate MTA Deployment Descriptor (mta.yaml)

The MTA deployment is described in the MTA Deployment Descriptor [4], a file called `mta.yaml`. In the first step, let CAP generate an initial `mta.yaml` file.

1. Open a new terminal in BAS (*Terminal → New Terminal*).
2. Run the following command `CDS ADD MTA` in the terminal.

   The `cds` Command Line Interface has generated the `mta.yaml` file based on your previously created settings in the `package.json`file.

3. Open the new `mta.yaml` file.

   The `mta.yaml` file consists of different `modules`, that are Cloud Foundry apps, and `resources`, that are Cloud Foundry services.

   Modules:

   - `risk-management-srv` - Your CAP OData service.
   - `risk-management-db-deployer` - Deploys CAP schema and data (CSV files) to database.

   Resources:

   The resources are generated from the `requires` section of `cds` in the `package.json`.

   - `risk-management-db` - HANA DB HDI container service instance
   - `risk-management-xsuaa` - Authorization & Trust Management (xsuaa) service instance

   The resources are Cloud Foundry service instances, that are automatically created and updated during the MTA deployment.

4. The `risk-management-xsuaa` service instance should make use of the `xs-security.json` configuration file you generated in an earlier step. Add the following line to the `parameters` section of the `risk-management-xsuaa` resource in the `mta.yaml` file:

   ```
   path: ./xs-security.json
   ```

Also remove the `config` section, as you are providing the configuration via the `xs-security.json` file.

The result should look like the following:

```
# ------------------------------------------------------------
- name: risk-management-xsuaa
  # ------------------------------------------------------------
  type: org.cloudfoundry.managed-service
  parameters:
    service: xsuaa
    service-plan: application
    path: ./xs-security.json # this line was added
```

The line `type: org.cloudfoundry.managed-service` tells the MTA deployment to automatically create the service instance if it does not yet exist. The `service` and `service-plan`parameters tell the MTA deployment what kind of service instance to create. As you have created the service instance manually before, it will just be updated with the provided configurations during MTA deployment.

5. The dependency to the destination service instance is still missing in the `mta.yaml`. We need to add it manually. As the `risk-management-srv` module requires the destination service, add the following line into its `requires` section:

   ```
   - name: risk-management-destination-service
   ```

   The result should look like the following:

   ```
   # --------------------- SERVER MODULE ------------------------
   - name: risk-management-srv
     # ------------------------------------------------------------
     type: nodejs
     path: gen/srv
     requires:
       # Resources extracted from CAP configuration
       - name: risk-management-db
       - name: risk-management-xsuaa
       - name: risk-management-destination-service # this line was
   added
     provides:
       - name: srv-api # required by consumers of CAP services (e.g.
   approuter)
         properties:
           srv-url: ${default-url}
   ```

   When adding a requirement, this also needs to be defined in the `resources` section of the document. Add the following lines in the resources section:

```
# ------------------------------------------------------------
- name: risk-management-destination-service
  # ------------------------------------------------------------
  type: org.cloudfoundry.managed-service
  parameters:
    service: destination
    service-plan: lite
```

6. Save the file.

7. Run the following command of the MTA build tool in order to build your project: `mbt build -t ./`

The build takes a while. It results in a multitarget application archive file, ending with the `.mtar` file extension.

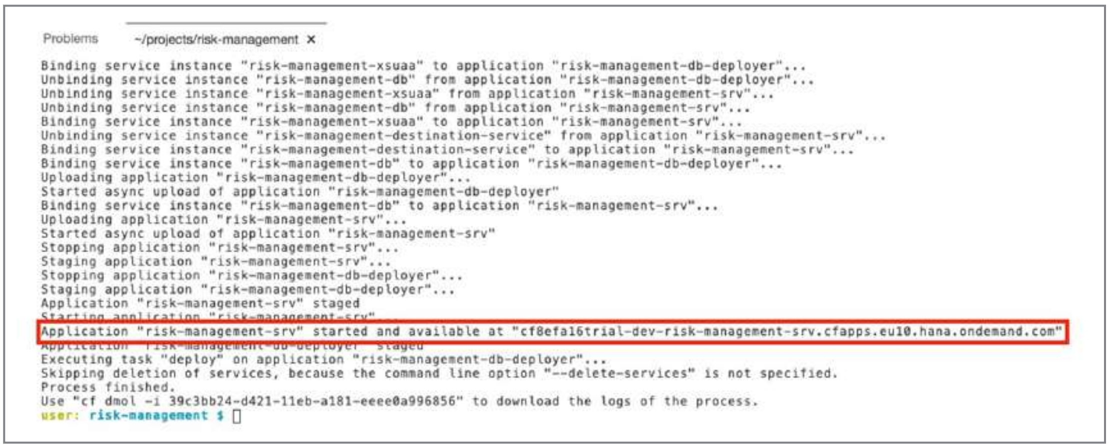
8. Use `cf deploy <.mtar file>` before running the following command to ensure that the SAP HANA database is still running.

   In this case enter `cf deploy risk-management_1.0.0.mtar` to replace `<.mtar file>` as the file name of the generated archive.

   The deployment takes a while. You can follow the deployment activity in the terminal log.

9. At the end of the deployment log, you will find a line saying: `Application "risk-management-srv" started and available at "<some_URL>"`.

   Like in this screenshot:



```
Binding service instance "risk-management-xsuaa" to application "risk-management-db-deployer"...
Unbinding service instance "risk-management-db" from application "risk-management-db-deployer"...
Unbinding service instance "risk-management-xsuaa" from application "risk-management-srv"...
Unbinding service instance "risk-management-db" from application "risk-management-srv"...
Binding service instance "risk-management-xsuaa" to application "risk-management-srv"...
Unbinding service instance "risk-management-destination-service" from application "risk-management-srv"...
Binding service instance "risk-management-destination-service" to application "risk-management-srv"...
Binding service instance "risk-management-db" to application "risk-management-db-deployer"...
Uploading application "risk-management-db-deployer"...
Started async upload of application "risk-management-db-deployer"
Binding service instance "risk-management-db" to application "risk-management-srv"...
Uploading application "risk-management-srv"...
Started async upload of application "risk-management-srv"
Stopping application "risk-management-srv"...
Staging application "risk-management-srv"...
Stopping application "risk-management-db-deployer"...
Staging application "risk-management-db-deployer"...
Application "risk-management-srv" staged
Starting application "risk-management-srv"...
Application "risk-management-srv" started and available at "cf8efa16trial-dev-risk-management-srv.cfapps.eu10.hana.ondemand.com"
Application "risk-management-db-deployer" staged
Executing task "deploy" on application "risk-management-db-deployer"...
Skipping deletion of services, because the command line option "--delete-services" is not specified.
Process finished.
Use "cf dmol -i 39c3bb24-d421-11eb-a181-eeee0a996856" to download the logs of the process.
user: risk-management $ 
```

   This is the application URL for your CAP service, which is publicly available via the Internet. Now this might not seem like a big win, because we are achieving the same deployment result as with the previous manual deployment. But the MTA deployment offers some more advantages:

   - A build tool
   - Automatically created service instances
   - Service keys
   - Destinations
   - Content deployment (HTML5, workflow, ...)
   - Blue-green deployment

Summary

You have:

- Created an application.
- Created a UI with SAP Fiori elements.
- Added custom business logic.
- Added an external service.
- Deployed your application manually.

You still have to deploy the UI part of your application. For this deployment, you will need an application router (approuter). To manage the access to your application, you need to implement restrictions and roles. We will cover all this in the next part.

Unit 5
Exercise 5

Deploy SAP BTP Cloud Foundry Applications Manually

Simulation: Deploy SAP BTP Cloud Foundry Applications Manually

For more information on *Deploy SAP BTP Cloud Foundry Applications Manually*, please view the simulation in the lesson *Deploy Manually* in your online course.

1.

Unit 5
Solution 5

Deploy SAP BTP Cloud Foundry Applications Manually

Simulation: Deploy SAP BTP Cloud Foundry Applications Manually

For more information on *Deploy SAP BTP Cloud Foundry Applications Manually*, please view the simulation in the lesson *Deploy Manually* in your online course.

1.

Reference Links

For your convenience this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 8: Reference Links: Manual Deployment

Ref#	Section	Context text fragment	Brief description	Link
1	Set Up SAP HANA Cloud Trial Instance	Open your SAP BTP Trial Account Cockpit	Open SAP BTP cockpit	https://cockpit.hanatrial.ondemand.com/trial/#/globalaccount
2	Manual Deployment Using a "Multi Target Application" (MTA) File	(MTA) file for deployment	SAP Help Portal, Create the MTA...	https://help.sap.com/viewer/4505d0bdaf4948449b7f7379d24d0f0d/2.0.05/en-US/ebb42efc880c4276a5f2294063fae0c3.html
3	Manual Deployment Using a "Multi Target Application" (MTA) File	it supports blue-green deployment.	SAP Help Portal, Blue-Green Deployment Strategy	https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/7c83810c31d842938cbc39c135a2d99f.html
4	Generate MTA Deployment Descriptor (mta.yaml)	MTA deployment is described in the MTA Deployment Descriptor	SAP Help Portal, MTA deployment	https://help.sap.com/viewer/4505d0bdaf4948449b7f7379d24d0f0d/2.0.03/en-US/33548a721e6548688605049792d55295.html

LESSON SUMMARY

You should now be able to:

- Perform a manual deployment

Unit 5
Lesson 2

Cloud Foundry Overview

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Describe the Cloud Foundry environment

Cloud Foundry Overview

Usage Scenario

Your company plans to use and develop a set of cloud-based applications. They have decided to use the SAP BTP Cloud Foundry environment for this purpose. As a developer in the Cloud Foundry environment, you need to get an overview of the basic terms.

Learning objectives

- Describe the basic terms in the Cloud Foundry environment.
- Explain the relationship between orgs (organizations) and spaces in the Cloud Foundry environment.

Cloud Foundry Environment

The Cloud Foundry environment allows you to create polyglot cloud applications in Cloud Foundry. It contains the SAP BTP, Cloud Foundry runtime service, which is based on the open-source application platform managed by the Cloud Foundry Foundation.

The Cloud Foundry environment enables you to develop new business applications and business services, supporting multiple runtimes, programming languages, libraries, and services. You can leverage a multitude of build packs, including community innovations and self-developed build packs.

Special Considerations for the Cloud Foundry Environment

In addition to global accounts and subaccounts, the Cloud Foundry environment includes another hierarchical level represented by orgs and spaces.

When you enable the Cloud Foundry environment in one of your subaccounts, the system automatically creates a Cloud Foundry org for you. The subaccount and the org have a 1:1 relationship and the same navigation level in the cockpit (even though they may have different names). You can create multiple spaces within that Cloud Foundry org. Spaces let you further break down your account model and use services and functions in the Cloud Foundry environment. There is no limit to how many spaces you can have within one org.

You can use both subaccounts and spaces to develop applications and to use services. You must therefore decide, for example, whether to create different subaccounts or spaces within one subaccount to set up a staged development environment.

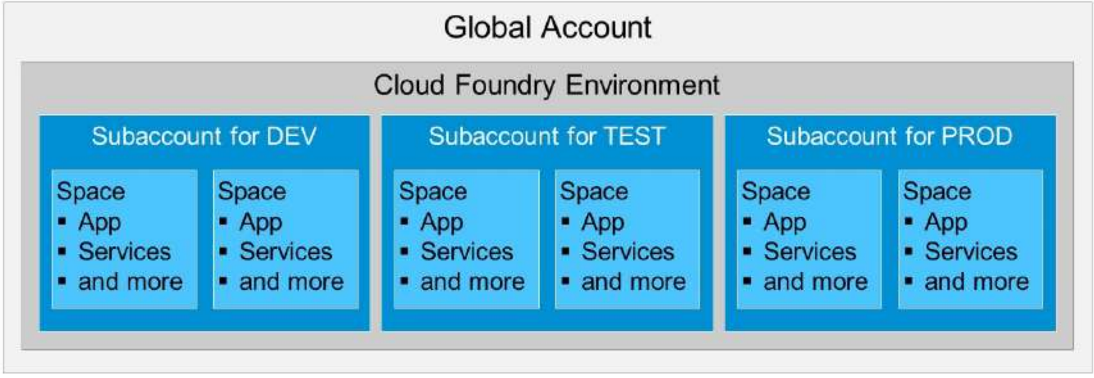


Figure 56: Cloud Foundry Subaccounts

In the Cloud Foundry environment, you deploy applications and consume services at space level. Similar to subaccounts, spaces enable you to once again structure and sub-divide quota if you want to. You can do this by managing space quota plans.

Summary

You now understand the key terms of the Cloud Foundry environment and the relationship between subaccount, org, and spaces in the Cloud Foundry environment. When you enable Cloud Foundry in a subaccount, you create a Cloud Foundry org in which you can then create multiple spaces. Each subaccount can contain only one Cloud Foundry org. There is no limit to how many spaces you can have within one org.

Further Reading

- Org Administration Using the Cockpit
- Cloud Foundry Environment

LESSON SUMMARY

You should now be able to:

- Describe the Cloud Foundry environment

Unit 5
Lesson 3

BTP Management Tool: CF CLI

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Use the Cloud Foundry Command Line Interface (CF CLI)

The Cloud Foundry Command Line Interface

Usage Scenario

You have an account in the SAP Business Technology Platform and you enabled the Cloud Foundry environment in your subaccount. To manage the environment using a command line interface, you need to get familiar with the Cloud Foundry Command Line Interface (CF CLI).

Learning objective

- Explain the use of the Cloud Foundry command line interface.

Manage the Cloud Foundry Environment

The CF CLI enables you to work with the Cloud Foundry environment to deploy and manage your applications. To manage the Cloud Foundry environment you can use both the BTP cockpit and the CF CLI. Many actions can only be performed via the CF CLI, for example renaming a Cloud Foundry org.

To install the CLI, you can either grab the latest release on the official release page or use https://tools.hana.ondemand.com/#cloud.

In order to manage the SAP Cloud Foundry environment you need to provide the CF CLI with an API endpoint. The API endpoint depends on the region you chose for your account:

- For EU: https://api.cf.eu10.hana.ondemand.com
- For US EAST: https://api.cf.us10.hana.ondemand.com
- For US CENTRAL: https://api.cf.us20.hana.ondemand.com

To try this out, you can open a command prompt and enter the following commands (in this case for the EU region):

```
cf api https://api.cf.eu10.hana.ondemand.com
cf login
```

Enter your credentials and try it out.

If you want to have an overview of all commands, enter the following command:

```
cf help -a
```

If you want to know the use of a special command, for example `rename-org` to rename an org, enter the following command:

```
cf help rename-org
```

CF CLI: Plug-ins

CF CLI: Plug-ins offers a list of additional commands that have been implemented as plug-ins to extend the base CF CLI client.

You can find examples of a CF CLI plug-in for performing operations on multi-target applications (MTAs) in Cloud Foundry, such as deploying, removing, viewing, and so on:

- MTA-Plug-in for Cloud Application Programming
- Multi-Apps CF CLI Plug-in

Summary

You now have a more profound understanding of the CF CLI to manage the Cloud Foundry environment.

LESSON SUMMARY

You should now be able to:

- Use the Cloud Foundry Command Line Interface (CF CLI)

Learning Assessment

1. After you run the command `cds add hana`, which file is updated with the required configuration?

   Choose the correct answer.

   □ A package.js

   □ B package.cds

   □ C package.json

   □ D package.mta

2. What are advantages of using an MTA file for deployment? (Choose 2)

   Choose the correct answers.

   □ A It supports red - green deployment.

   □ B It supports blue-green deployment.

   □ C It provides workflows.

   □ D It provides a build tool.

3. What are yaml files used for?

   Choose the correct answer.

   □ A To describe documents

   □ B To describe data

4. Which statements about YAML files are correct? (Choose 2)

   Choose the correct answers.

   □ A YAML uses whitespace indentation for structuring purposes.

   □ B YAML uses tab indentation for structuring purposes.

   □ C YAML uses hyphens: - for comments.

   □ D YAML uses hashes: # for comments.

5. Which concept describes Cloud Foundry applications?

   Choose the correct answer.

   □ A Monoglot

   □ B Polyglot

   □ C Proglot

   □ D Epiglot

6. What kind of relationship does a subaccount and a Cloud Foundry org have?

   Choose the correct answer.

   □ A n:n

   □ B n:1

   □ C 1:n

   □ D 1:1

7. What is the limitation of spaces within one Cloud Foundry org?

   Choose the correct answer.

   □ A Exactly 1 space per org

   □ B Maximum 2 spaces per org

   □ C Unlimited spaces per org

8. Which tools can you use to manage the SAP BTP, Cloud Foundry environment?

   Choose the correct answers.

   □ A SAP Business Application Studio

   □ B Eclipse

   □ C CF CLI

   □ D SAP BTP cockpit

UNIT 6 Authorization and Trust Management

Lesson 1

Lesson 2

Lesson 3

Lesson 4

Lesson 5

Lesson 6

UNIT OBJECTIVES

- Define CDS Restrictions and Roles
- Set Up SAP Authorization and Trust Management
- Describe the SAP Authorization and Trust Management Service
- Create and configure an approuter
- Add the UI and approuter module to the MTA
- Assign role collections

Unit 6
Lesson 1

Define CDS Restrictions and Roles

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Define CDS Restrictions and Roles

Define Restrictions and Roles in CDS: Exercise Overview

Scenario

Before deploying to our productive SAP BTP environment, you want to ensure, that only permitted users can access your app to view and edit data. Therefore, you will first add authorizations to your CAP service and then add two mock users to further test your app locally.

Task Flow

In this exercise, you will perform the following tasks:

- Implement authentication support - roles and restrictions - for an application.
- Add local users to test the authentication implementation.

Prerequisites

You have successfully deployed your application manually.

Enable Authentication Support

To enable authentication support in CAP, a `node.js` module called `passport` needs to be installed.

1. Navigate to your `risk-management` folder in a terminal in the Business Application Studio. With your `cds watch` still running in one terminal, it is the easiest to open another second terminal next to it, by invoking Terminal and the New Terminal in the menu. Alternatively, you can also suspend `cds watch` in your existing terminal by pressing CTRL +C. In both cases, you should already be in the `risk-management` folder.

2. Install the `passport` module. (the --save part makes sure it's also added as a dependency to your project's `package.json`) `npm install --save passport`.

Adding CAP Role Restrictions to Entities

In this step, you will add authorizations to the `Risks` service. You will add two different roles `RiskManager` and `RiskViewer` with different access scope.

1. Open the file `srv/risk-service.cds`.

2. Change the code as shown below and add the restrictions (`@(...)`) to block to your `Risks` and `Mitigations` entities. You have to delete code - anything between `//### BEGIN OF DELETE` and `//### END OF DELETE` - and add code - anything between `//### BEGIN OF INSERT` and `//### End OF INSERT`.

```
using {riskmanagement as rm} from '../db/schema';

 /**
   * For serving end users
   */
 service RiskService @(path : 'service/risk') {
 //### BEGIN OF DELETE
   entity Risks as projection on rm.Risks;
 //### END OF DELETE
 //### BEGIN OF INSERT
   entity Risks @(restrict : [
       {
          grant : [ 'READ' ],
          to : [ 'RiskViewer' ]
       },
       {
           grant : [ '*' ],
           to : [ 'RiskManager' ]
       }
   ]) as projection on rm.Risks;
 //### END OF INSERT
   annotate Risks with @odata.draft.enabled;
 //### BEGIN OF DELETE
   entity Mitigations as projection on rm.Mitigations;
 //### END OF DELETE
 //### BEGIN OF INSERT
   entity Mitigations @(restrict : [
       {
         grant : [ 'READ' ],
         to : [ 'RiskViewer' ]
       },
       {
         grant : [ '*' ],
         to : [ 'RiskManager' ]
       }
   ]) as projection on rm.Mitigations;
 //### END OF INSERT
     annotate Mitigations with @odata.draft.enabled;
   entity BusinessPartners as projection on rm.BusinessPartners;
 }
```

3. Save the file.

   With this change, users who are assigned the role `RiskViewer` can view ("READ") risks and mitigations. Users who are assigned the role `RiskManager` can view and change risks and mitigations ("*").

Add Users for Local Testing

Since the authorization checks have been added to the CAP model, they apply not only when deployed to the cloud but also for local testing. Therefore, you will need a way to log in to the application locally.

CAP allows you to add local users for testing as part of the `cds` configuration. In this tutorial, we use the `.cdsrc.json` file to add the users.

The `.cdsrc.json` file can be used to store project configurations like in the `package.json` file. Learn more here[1].

1. In the project, go to the file *.cdsrc.json* and open it for editing.
2. In the editor, replace its content with the following lines:

```
{
 "[development]": {
   "auth": {
    "passport": {
     "strategy": "mock",
     "users": {
       "risk.viewer@tester.sap.com": {
        "password": "initial",
        "ID": "riskviewer",
        "userAttributes": {
          "email": "risk.viewer@tester.sap.com"
        },
        "roles": ["RiskViewer"]
      },
       "risk.manager@tester.sap.com": {
        "password": "initial",
        "ID": "riskmanager",
        "userAttributes": {
          "email": "risk.manager@tester.sap.com"
        },
        "roles": ["RiskManager"]
      }
     }
    }
   }
  }
 }
```

3. Save the file.

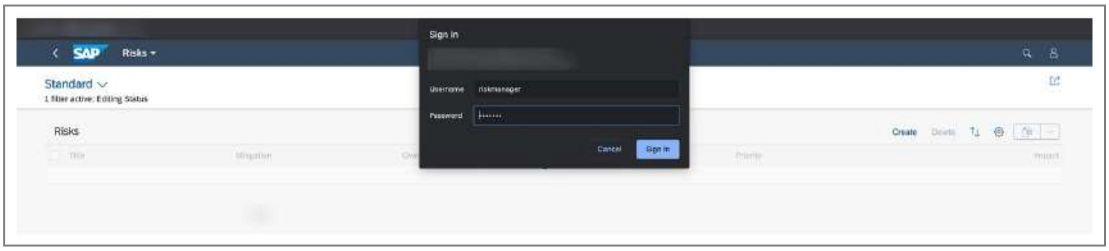
   The file defines two users, `riskviewer` and `riskmanager`. Let's take a look at the `riskmanager` example.

   The user is defined by an `ID`, which can be any identifier for a user. The user has an `email`, a `password` parameter, and a `roles`parameter.

Access the Risk Application with a User and Password

When accessing the `Risks` or the `Mitigations` service in the browser, you get a basic authorization pop-up window, asking for your user and password. You can use both users that you defined in the previous step to log in and see how this works.

1. In the tab with the running application, navigate back to the launch page, press *refresh* in the browser.
2. Choose the *Risks* tile and in the app press *Go*.
3. In the pop-up, enter the *Username* **`riskmanager`**.
4. Enter the *Password* **`initial`**.



Sign in

Username riskmanager

Password •••••••

5. You can now access the *Risks* application.

Caveat

There’s no log out functionality yet. To clear the basic authentication login data from the browser cache, you can either clear the browser cache or simply close all browser windows.

Summary

You enabled authentication using `passport.js`[2]. You also added roles and restrictions to control access to your application. Next you will set up SAP Authorization and Trust management.

Unit 6
Exercise 6

Define Restrictions and Roles in CDS

Simulation: Define Restrictions and Roles in CDS

For more information on *Define Restrictions and Roles in CDS*, please view the simulation in the lesson *Define CDS Restrictions and Roles* in your online course.

1.

Unit 6
Solution 6

Define Restrictions and Roles in CDS

Simulation: Define Restrictions and Roles in CDS

For more information on *Define Restrictions and Roles in CDS*, please view the simulation in the lesson *Define CDS Restrictions and Roles* in your online course.

1.

Reference Links

For your convenience this section contains the external references of this lesson.

If links are used multiple times in a text, only the first location is mentioned in the reference table.

Table 9: Reference Links: Restrictions and Roles in CDS

Ref#	Section	Context text fragment	Brief description	Link
1	Add Users for Local Testing	Learn more here.	CAP project configuration	https://cap.cloud.sap/docs/node.js/cds-env#project-settings
2	Summary	You enabled authentication using passport.js	Passport.js	http://www.passportjs.org/

LESSON SUMMARY

You should now be able to:

- Define CDS Restrictions and Roles

Unit 6
Lesson 2

Set Up SAP Authorization and Trust Management

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Set Up SAP Authorization and Trust Management

Set Up SAP Authorization and Trust Management

Scenario

You will set up SAP Authorization and Trust Management [1] to secure your application.

Task Flow

In this exercise, you will perform the following tasks:

- Enable authentication support.
- Add XS advanced User Account and Authentication (XSUAA) security configuration.
- Modify authorization settings in an MTA.yaml file.

Prerequisite

You have added restrictions and roles to your application.

Enable Authentication Support

To enable authentication support in CAP for SAP BTP, the `xssec` and `xsenv` modules need to be installed.

1. Open a new terminal in BAS (*Terminal* → *New Terminal*).
2. In the terminal, run the following command to install the hdb module and automatically add it as a dependency into the `package.json` file of your project: `npm i --save @sap/xssec @sap/xsenv`

Roles and Scopes

In the SAP BTP, Cloud Foundry environment, a single authorization is called `scope`. For example, there could be a scope "Read" and a scope "Write" that allow users to read or write respectively a certain business object. Scopes cannot be assigned to users directly. They are packaged into roles. For example, the role "Editor" could have "Read" and "Write" scopes, while the role "Viewer" could have the "Read" scope only.

However, CAP recommends to use roles only and do a one-to-one mapping. In Lesson 1: Defining Restrictions and Roles in CDS we defined two roles.

Set Up Application Security with XSUAA Security Configuration

First you need to configure the XSUAA service [2]. Create the file `xs-security.json` in your `RiskManagement` project by executing the following in a terminal in BAS:

```
cds compile srv --to xsuaa >xs-security.json
```

The generated file contains the configuration of the XSUAA. Behind the scenes, CAP has taken the authorization parts `@(restrict ... )` from your service definition and created scopes and role templates from it.

For example, it found the roles `RiskViewer` and `RiskManager` in the `srv/risk-service.cds` file:

```
entity Risks @(restrict : [
    {
      grant : ['READ'],
      to : ['RiskViewer']
    },
    {
      grant : ['*'],
      to : ['RiskManager']
    }
 ]) as projection on rm.Risks;
```

Then it created scopes and roles for both in the `xs-security.json` file in your project:

```
{
   "xsappname": "risk-management",
   "tenant-mode": "dedicated",
   "scopes": [
     {
       "name": "$XSAPPNAME.RiskViewer",
       "description": "RiskViewer"
     },
     {
       "name": "$XSAPPNAME.RiskManager",
       "description": "RiskManager"
     }
   ],
   "attributes": [],
   "role-templates": [
     {
       "name": "RiskViewer",
       "description": "generated",
       "scope-references": ["$XSAPPNAME.RiskViewer"],
       "attribute-references": []
     },
     {
       "name": "RiskManager",
       "description": "generated",
       "scope-references": ["$XSAPPNAME.RiskManager"],
       "attribute-references": []
     }
   ]
 }
```

Adjust Authorization and Trust Management Service (XSUAA) in MTA

The next step is to adjust the configuration of the Authorization and Trust Management Service in the `mta.yaml` to allow user login, authorization, and authentication checks.

1. In your `mta.yaml` file, change the following:

```
_schema-version: '3.1'
 ...
 resources:
 ...
 # ------------------------------------------------------------
   - name: risk-management-xsuaa
 # ------------------------------------------------------------
   type: org.cloudfoundry.managed-service
   parameters:
     service: xsuaa
     service-plan: application
     path: ./xs-security.json
 //### BEGIN OF INSERT
     config:
       xsappname: 'risk-management-${space}'
       role-collections:
       - name: 'RiskManager-${space}'
         description: Manage Risks
         role-template-references:
         - $XSAPPNAME.RiskManager
       - name: 'RiskViewer-${space}'
         description: View Risks
         role-template-references:
         - $XSAPPNAME.RiskViewer
 //### END OF INSERT
```

2. Save the file.

   The configuration for XSUAA is read from the `xs-security.json` file that was updated in the previous step.

   However, in the `config` element of the YAML file, values can be added and overwritten.

   The value `xsappname` gets overwritten with a Cloud Foundry space-dependent value `${space}`. The name has to be unique within an SAP BTP subaccount.

   This allows multiple deployments of this application in different spaces of the same subaccount. This is useful when different members of a team want to try out the application and don't want to create a new subaccount for each team member.

   For a productive application, the `xsappname` should be explicitly set to the desired value.

   Further, you can add role collections using the `xs-security.json` file. Since role collections need to be unique in a subaccount like the `xsappname`, you can add it here and use the `${space}` variable to make them unique like for the `xsappname`.

   Alternatively, role collections [3] can be manually added in the SAP BTP cockpit.

Summary

You added XSUAA security settings to your application. Now you need to add an application router (approuter) to route your application's requests from the web browser to either the service or the UI.

Unit 6
Exercise 7

Set Up SAP Authorization and Trust Management

Simulation: Set Up SAP Authorization and Trust Management

For more information on *Set Up SAP Authorization and Trust Management*, please view the simulation in the lesson *Set Up SAP Authorization and Trust Management* in your online course.

1.

Unit 6
Solution 7

Set Up SAP Authorization and Trust Management

Simulation: Set Up SAP Authorization and Trust Management

For more information on *Set Up SAP Authorization and Trust Management*, please view the simulation in the lesson *Set Up SAP Authorization and Trust Management* in your online course.

1.

Reference Links

For your convenience this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 10: Reference Links: SAP Authorization and Trust Management

Ref#	Section	Context text fragment	Brief description	Link
1	Set Up SAP Authorization and Trust Management	You will set up SAP Authorization and Trust Management	SAP Authorization and Trust Management	https://help.sap.com/viewer/df50977d8bfa4c9a8a063ddb37113c43/LATEST/en-US/aaaad9424e7442eab5d44b20f0ecbfd7.html
2	Set Up Application Security with XSUAA Security Configuration	configure the XSUAA XSUAA service	XSUAA service	https://help.sap.com/viewer/4505d0bdaf4948449b7f7379d24d0f0d/LATEST/en-US/35d910ee7c7a445a950b6aad989a5a26.html
3	Adjust Authorization and Trust Management Service (XSUAA) in MTA	Alternatively, role collections can be manually added	Role collections	https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/LATEST/en-US/0039cf082d3d43eba9200fe15647922a.html

LESSON SUMMARY

You should now be able to:

- Set Up SAP Authorization and Trust Management

Unit 6
Lesson 3

Authorization and Trust Management

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Describe the SAP Authorization and Trust Management Service

SAP Authorization and Trust Management Service

Usage Scenario

Your company is developing a set of applications on the SAP Business Technology Platform, Cloud Foundry environment. As the applications are running in the cloud, you want to protect them from unauthorized access. The SAP Authorization and Trust Management service in conjunction with the Extended Services - User Account and Authentication (XSUAA) service provide all the required features.

Learning objectives

- Describe the SAP Authorization and Trust Management service.
- Describe the Extended Services - User Account and Authentication (XSUAA) service.
- Describe the difference between platform and business users.
- Describe the dependencies between role-collections, roles, and scopes.
- Describe the application security descriptor (xs-security.json) file.

Manage User Authorizations

The SAP Authorization and Trust Management service lets you manage user authorizations and trust to identity providers. Identity providers are the user base for applications. You can use an identity authentication tenant, an SAP on-premise system, or a custom corporate identity provider. User authorizations are managed using technical roles at the application level, which can be aggregated into business-level groups and role collections for large-scale cloud scenarios.

Platform Users vs. Business Users

Platform Users

Platform users are usually developers, administrators, or operators who deploy, administer, and troubleshoot applications and services on SAP BTP.

For platform users, the default identity provider is SAP ID Service.

Business Users

Business users use the applications that are deployed to SAP BTP. For example, the end users of your deployed application or users of subscribed apps or services, such as SAP Business Application Studio or SAP Web IDE, are business users.

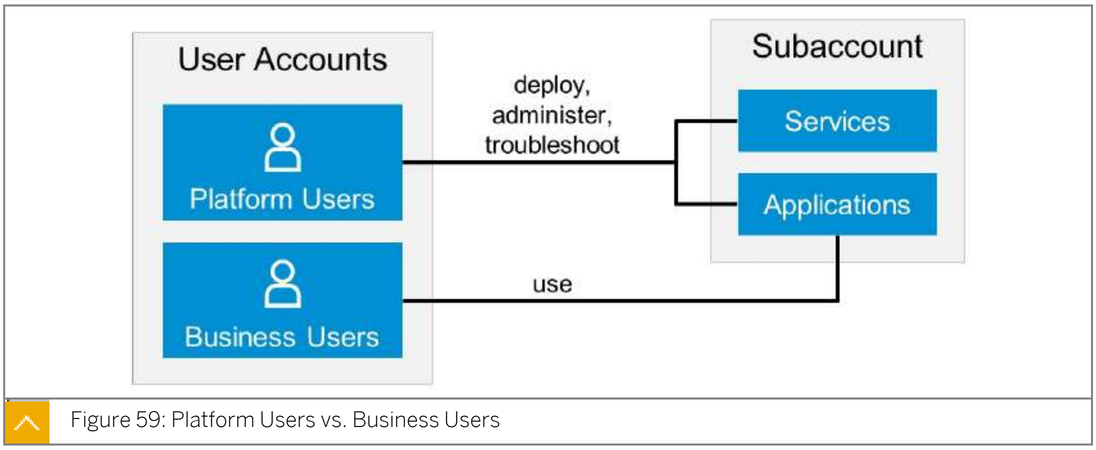


Figure 59: Platform Users vs. Business Users

Animation:

For more information on , please view the animation in the lesson *Authorization and Trust Management*, in your online course.

Extended Services - User Account and Authentication (XSUAA) service

The Extended Services - User Account and Authentication (XSUAA) service is one of the most important components to deal with when developing your own applications on Cloud Foundry. It authenticates and authorizes your users and assigns the right principals to your user's session so your application can:

- Identify the user by Email, UserId, First and Lastname
- Check its roles (scopes) to decide if a user is allowed to do something or prohibit its action

The XSUAA is an internal development of SAP. SAP took the base of the open source UAA OAuth2 Provider of Cloud Foundry and extended it with SAP specific features to be used in SAP applications. One important thing is that the XSUAA does NOT store "real" users. This is why the XSUAA needs to trust an external Identity Provider (IdP).

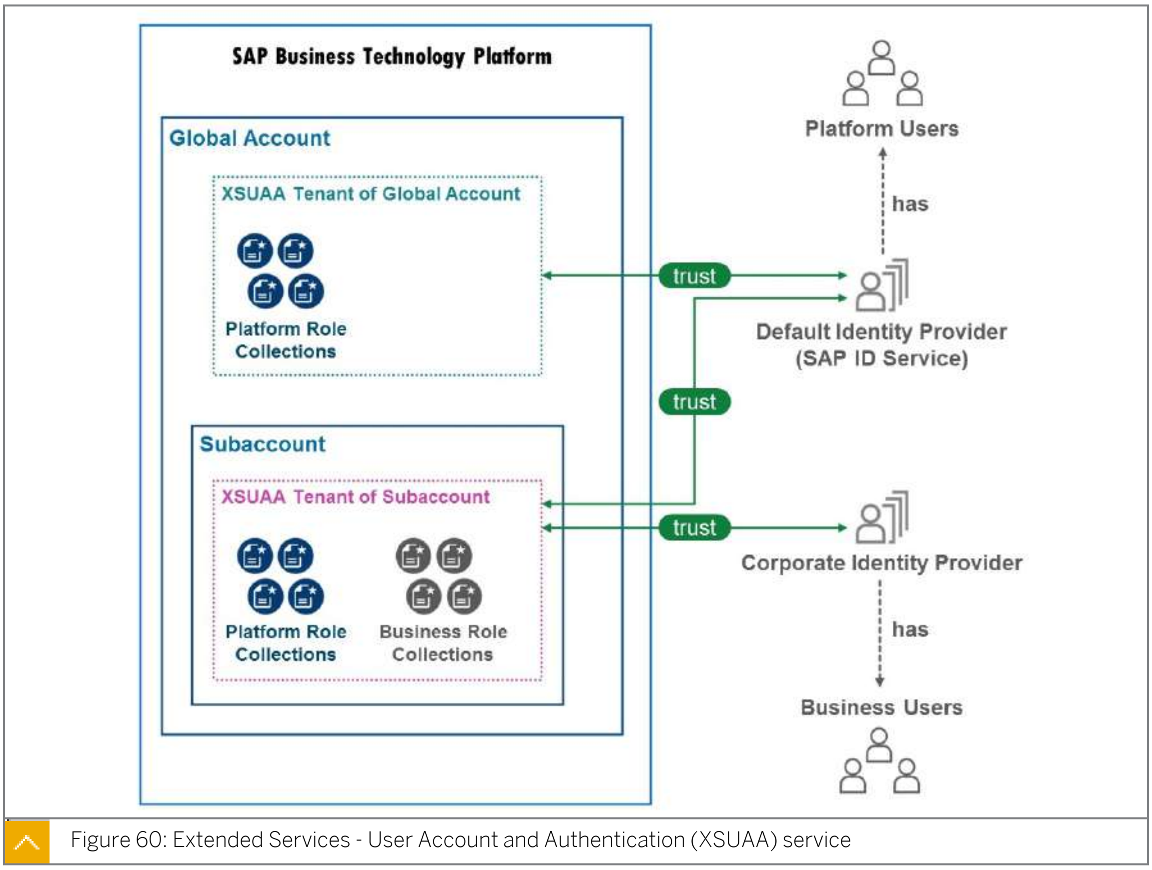

SAP Business Technology Platform
Global Account
XSUAA Tenant of Global Account
Platform Role Collections
Subaccount
XSUAA Tenant of Subaccount
Platform Role Collections
Business Role Collections
trust
trust
trust
Platform Users
has
Default Identity Provider (SAP ID Service)
Corporate Identity Provider
has
Business Users

Figure 60: Extended Services - User Account and Authentication (XSUAA) service

Note:
This graphic only applies to SAP Business Technology Platform cloud management tools Feature Set B. Check out User and Member Management in the SAP Help Portal pages to better understand the differences between Feature Set A and Feature Set B in regard to user management.

In Feature Set B, your SAP BTP global account has its own XSUAA tenant. This XSUAA tenant by default has a trust relationship to the SAP ID Service. The SAP ID Service manages a large base of users, that have created a user account with SAP. You can add users, that exist in the SAP ID Service, as members to your global and subaccount. In order for these users to be able to perform administrative tasks, they need to be assigned with corresponding role-collections. There is a set of default platform role-collections, like `Global Account Administrator`, `Global Account Viewer`, `Subaccount Administrator`, or `Cloud Connector Administrator` that you can use for the purpose of assigning SAP BTP account management authorizations. See this SAP Help Portal page for further information on the default role collections for account management.

Contrary to the platform users from the SAP ID Service, your business users can also be provided via your own corporate identity provider. These are the users you want to provide access to your business applications. These might be SaaS applications provided by SAP, like SAP Business Application Studio or SAP Workflow Management, or your own applications that you develop on the SAP BTP. The business applications have their own role-collections, like -for example- `Business_Application_Studio_Developer`, `WorkflowManagementAdmin`, or any custom role-collection that you create for your application.

It is planned by the end of 2021 to also support corporate identity providers for platform users.

The following sections shed a bit more light on the terms role-collection, role, and scope, and their relationship to each other.

Role-Collections, Roles, and Scopes

The following image shows the relationships between role-collections, roles, and scopes.

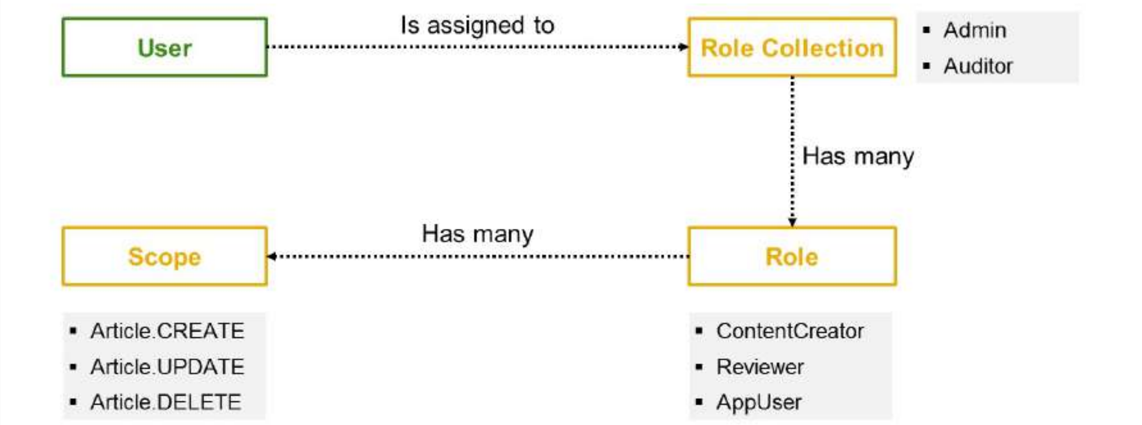


Figure 61: Role-Collections, Roles and Scopes

Scopes

Scopes are arbitrary values that express authorizations / access rights in an application or service. Scopes need to be prefixed with an `xsappname` to make them uniquely identifiable.

Roles

Roles are entities that hold several scopes. Scopes can be put in multiple roles, so you are not limited to have scopes sitting in just one role.

Role-Collections

Role-collections contain one or more roles. A role can be used in multiple role-collections. But it is totally fine to have, for example, a role-collection called `Admin` that only has an `admin` role.

Role-collections are stored as an assignment in the XSUAA and are THE entity that can be assigned to a certain business user.

Animation:

For more information on , please view the animation in the lesson *Authorization and Trust Management*, in your online course.

How Does it Work in Practice?

The following section is a short summary of the official help page.

In the diagram you can see that there are different personas. One is the developer working within a project and space. The other persona is an admin taking care of the CF account as a security admin.

Watch this video to know more about scope, role-collections, and roles. In the video you can see three different personas. One is the developer working within a project and space. The other is an admin taking care of the CF account as a security admin.

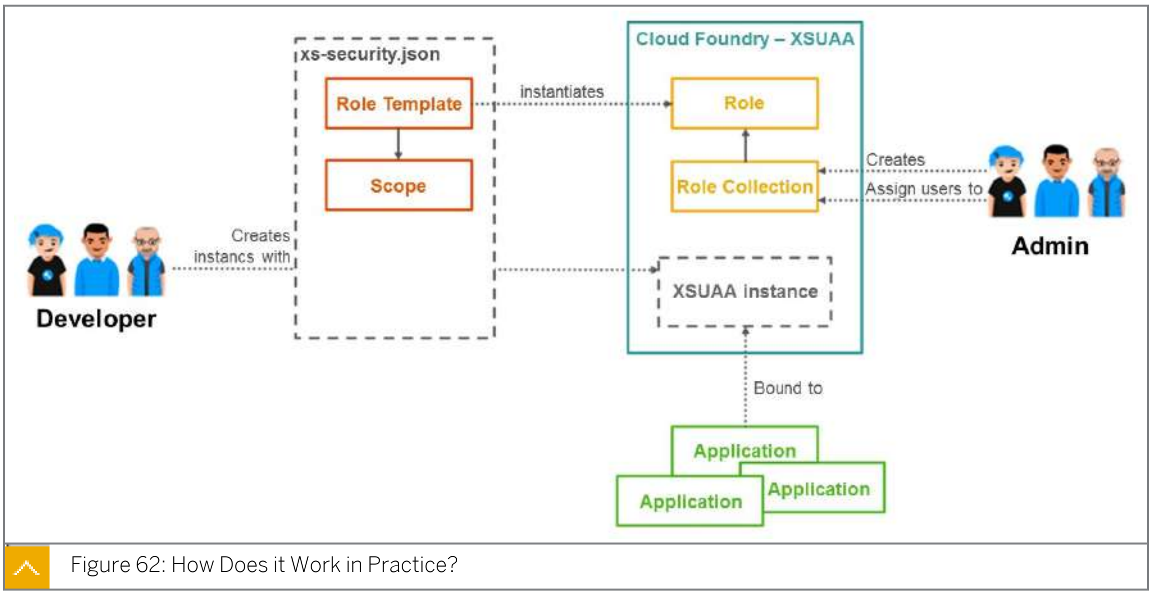


Figure 62: How Does it Work in Practice?

Animation: How Does it Work in Practice?

For more information on *How Does it Work in Practice?*, please view the animation in the lesson *Authorization and Trust Management*, in your online course.

When you as a developer build a new business application, you define scopes and pre-bundle them in role-templates. You perform these definitions in the so called application security descriptor (`xs-security.json`) file. You use the `xs-security.json` file to create an instance of the XSUAA service, which is bound to the corresponding business application(s). The role-template definitions translate into roles. You as an administrator assemble these roles into role-collections and assign them to the business users of your application.

What is xs-security.json?

To simplify things, let’s just call the `xs-security.json` the “declaration of your app’s security”. The following `xs-security.json` is an excerpt of the `risk-management` application being built in the SAP Extension Suite fundamentals course.

```
{
   "xsappname": "risk-management",
   "tenant-mode": "dedicated",
   "scopes": [
     {
       "name": "$XSAPPNAME.RiskViewer",
       "description": "RiskViewer"
     },
     {
       "name": "$XSAPPNAME.RiskManager",
       "description": "RiskManager"
     }
   ],
   "attributes": [],
   "role-templates": [
    {
      "name": "RiskViewer",
      "description": "generated",
      "scope-references": [
        "$XSAPPNAME.RiskViewer"
      ],
      "attribute-references": []
    },
    {
      "name": "RiskManager",
      "description": "generated",
      "scope-references": [
        "$XSAPPNAME.RiskManager"
       ],
      "attribute-references": []
    }
  ]
}
```

You have to tell the XSUAA service how to call your application (`xsappname`) and further define your scopes and role-templates. The scopes are being used within the application to check concrete permissions whenever a user tries to perform a certain action.

Summary

You have now gained an overview over the SAP Authorization and Trust Management service, the relevance of the Extended Services - User Account and Authentication (XSUAA) service and the basic concepts of user management and role-collection assignment. You know what the `xs-security.json` application security descriptor file is used for and how it relates to scopes, roles, and the XSUAA service. You also know where to look for additional information.

Further Reading

- Parts of the text and graphics were taken from the blog post Demystifying XSUAA in SAP Cloud Foundry. Take a look at the blog for even more information. The explanations in regard to user management in the blog post refer to the SAP BTP cloud management tools, Feature Set A.
- The Application Security Descriptor

LESSON SUMMARY

You should now be able to:

- Describe the SAP Authorization and Trust Management Service

Unit 6
Lesson 4

Create an Approuter

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Create and configure an approuter

Create an Application Router: Exercise Overview

Scenario

You will create an application router (approuter) to route requests originating in the web browser to either the CAP service or to the UI.

Using a Standalone Application Router

For your application to run on the cloud, you need another component on top of the service and the UI. It will route the request from the web browser either to the CAP service or to the provider of the UI sources. This component also ensures that authenticated and authorized users get a token from the XSUAA service, which is also routed to the CAP service.

This component is called an “application router” (approuter). "The application router is the single point-of-entry for an application running in the Cloud Foundry environment on SAP BTP. The application router is used to serve static content, authenticate users, rewrite URLs, and forward or proxy requests to other microservices while propagating user information." [1].

You can either have a standalone approuter or a managed approuter [2]. In this module, you are going to use the standalone approuter, because it does not lead to other service dependencies. The managed approuter is part of other SAP BTP services like the SAP Launchpad Service or the SAP Portal Service. Checkout this blog [3] for more detailed information on the managed approuter vs. the standalone approuter.

Task Flow

In this exercise, you will perform the following tasks:

- Create a standalone approuter.
- Configure an approuter using a JSON file.

Prerequisite

You have set up SAP authorization and trust management (XSUAA).

Create the Approuter Module

1. Using a terminal in BAS, create a folder `approuter`, where you store all approuter artifacts, and switch to the new folder:

```
mkdir approuter
cd approuter
```

> Note:
> You could have completed this step in the BAS file explorer instead of the terminal. However, since we will need a command line interface in the folder in the next step, we decided to use the terminal for creating the folder.

2. In the terminal of the new `approuter` folder, initialize `npm` and install the latest version of approuter NPM module[4]:

```
npm init --yes
npm install @sap/approuter
```

The functionality of the approuter is provided by the `@sap/approuter` NPM module.

3. Check the required Node.js version for approuter

This is declared in the `package.json` file of the approuter. You can check it with this script:

```
cat node_modules/@sap/approuter/package.json | grep '"node"'
```

It outputs something like:

```
"node": "^12.0.0 || ^14.0.0"
```

In this example, the approuter supports Node.js 12.x.x and 14.x.x versions.

4. Add the required Node.js version to the `approuter/package.json` file. This depends on the supported versions of the approuter, ^14.0.0 in this example. Also add the start script for the approuter.

```
{
 "name": "approuter",
 ...
 "scripts": {
 //### BEGIN OF DELETE
   "test": "echo \"Error: no test specified\" && exit 1"
 //### END OF DELETE
 //### BEGIN OF INSERT
   "start": "node node_modules/@sap/approuter/approuter.js"
 //### END OF INSERT
 },
 ...
 "dependencies": {
   "@sap/approuter": "^8.5.5"
 //### BEGIN OF DELETE
 }
 //### END OF DELETE
 //### BEGIN OF INSERT
 },
 "engines": {
    "node": "^14.0.0"
 }
 //### END OF INSERT
 }
```

5. Save the file.

Approuter Configuration

Configure the approuter by creating a file `xs-app.json` in the `approuter` folder with the following content:

```
{
   "welcomeFile": "/app/risks/webapp/index.html",
   "authenticationMethod": "route",
   "sessionTimeout": 30,
   "logout": {
     "logoutEndpoint": "/do/logout",
     "logoutPage": "/"
   },
   "routes": [
     {
       "source": "^/app/(.*)$",
       "target": "$1",
       "localDir": "resources",
       "authenticationType": "xsuaa"
     },
     {
       "source": "^/service/(.*)$",
       "destination": "srv-binding",
       "authenticationType": "xsuaa"
     }
   ]
 }
```

The configuration in the `routes` array tells the approuter how to respond to requests.

- The files in the `resources` folder will be served for all requests to `/app`. Later, there's an explanation on how you get the "app" files into this resource folder.
- All requests starting with `/service` will be forwarded to the CAP service based on the URL we configured in the MTA using the destination `srv_app`. Remember, the risk service is reachable via `/service/risk`. Further services are automatically routed as long as they start with `/service/` as well.

Further, the approuter will automatically redirect to the `/app/launchpage.html` when accessed without a path, which will then serve the file `resources/launchpage.html`.

Summary

You added an approuter to your application. In the next step, you will add the UI application to your project.

Unit 6
Exercise 8

Create an Application Router

Simulation: Create an Application Router

For more information on *Create an Application Router*, please view the simulation in the lesson *Create an Approuter* in your online course.

Unit 6
Solution 8

Create an Application Router

Simulation: Create an Application Router

For more information on *Create an Application Router*, please view the simulation in the lesson *Create an Approuter* in your online course.

Reference Links

For your convenience this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 11: Reference Links: Application Router

Ref#	Section	Context text fragment	Brief description	Link
1	Using Standalone Application Router	This component is called “application router” (approuter)	Approuter	https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/LATEST/en-US/01c5f9ba7d6847aaaf069d153b981b51.html
2	Using Standalone Application Router	standalone approuter or a managed approuter	Managed approuter	https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/LATEST/en-US/589a2395df2d481393acb1ba2f17eeef.html
3	Using Standalone Application Router	Checkout this blog	Managed approuter	https://blogs.sap.com/2021/05/17/sap-tech-bytes-faq-managed-approuter-vs.-standalone-approuter/
4	Create the Approuter Module	install the latest version of approuter NPM module:	approuter NPM module	https://www.npmjs.com/package/@sap/approuter#overview

LESSON SUMMARY

You should now be able to:

- Create and configure an approuter

Unit 6 Lesson 5

Add the UI and Approuter Module to the MTA

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Add the UI and approuter module to the MTA

Add the UI and Approuter Module to the MTA: Exercise Overview

Scenario

You will add the UI application and the approuter to the `mta.yaml` file.

Task Flow

In this exercise, you will perform the following task:

- Add the UI and the approuter to the `mta.yaml` file.

Prerequisite

You have set up a standalone approuter.

The automatic creation of the `mta.yaml` file added everything that is needed from the CAP side to the `mta.yaml` file:

- The service.
- The database deployer
- The dependency to the XSUAA and SAP HANA Cloud service

Your SAP Fiori elements-based UI application, however, is still missing. You will have to add this module manually since, currently, there is no automation support to do this.

The approuter is also an application, just like your CAP service and the UI. To deploy the approuter, you will need to add a configuration to the MTA file that you created earlier.

Add Modules to the mta File

1. Add the `risk-management-approuter` and the `risk-management-app` module for the Approuter to the `mta.yaml` file:

```
# -------------------- SIDECAR MODULE ------------------------
 - name: risk-management-db-deployer
   # ------------------------------------------------------------
   type: hdb
   path: gen/db
   parameters:
     buildpack: nodejs_buildpack
   requires:
     # 'hana' and 'xsuaa' resources extracted from CAP configuration
```

```
    - name: risk-management-db
    - name: risk-management-xsuaa

#//### BEGIN OF INSERT

# -------------------- APPROUTER -----------------------------
- name: risk-management-approuter
  # ------------------------------------------------------------
  type: nodejs
  path: approuter
  requires:
    - name: risk-management-xsuaa
    - name: srv-api
    group: destinations
    properties:
      forwardAuthToken: true
      strictSSL: true
      name: srv-binding
      url: "~{srv-url}"
  build-parameters:
    requires:
      - name: risk-management-app
        artifacts:
        - ./*
        target-path: resources
# -------------------- UI ----------------------------------
- name: risk-management-app
  # ------------------------------------------------------------
  type: html5
  path: app
  build-parameters:
    supported-platforms: []
#//### END OF INSERT
```

2. Save the file.

The approuter takes the UI resources of the `risk-management-app` and puts it in the `resources` directory. This is where the `xs-app.json` looks for the files requested for `/app/....`

The `risk-management-xsuaa` binding adds your already existing XSUAA service instance to the approuter. This enables login and log out. The approuter forwards requests with the authentication token (`Authorization: Bearer <jwt-token>`) to the CAP service. The CAP service then uses the token for authentication and authorization checks.

The `srv-binding` creates an environment variable `destinations` that contains a JSON array with one object containing the destination of the CAP service. This is required to forward requests to the CAP service.

The generated environment variable looks like this:

```
destinations='[{ "name": "srv-binding", "forwardAuthToken": true,
"strictSSL": true, url: "https://..." }]
```

The URL is taken from the `risk-management-srv` module that needs to be enhanced to export this information.

Re-Build and Re-Deploy the .mtar File

1. Build your project with the MTA Build Tool (MBT):

```
mbt build -t ./
```

Make sure you're in the `root` folder of your project.

2. Deploy your project to SAP BTP:

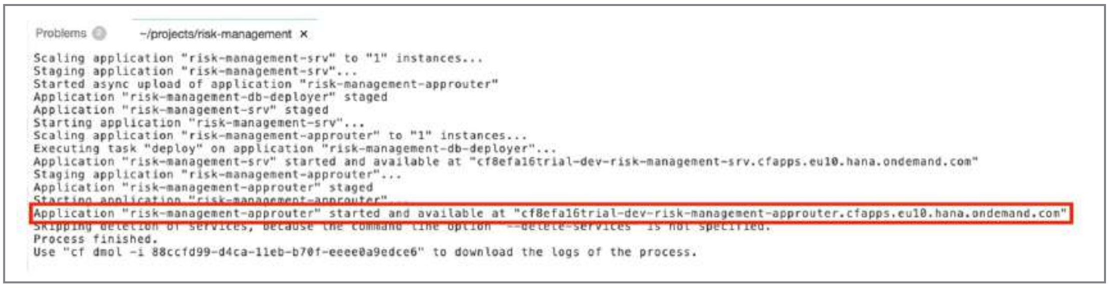
```
cf deploy risk-management_1.0.0.mtar
```

Additional Documentation: How to build an MTA archive from the project sources [1].

3. When the deployment is finished, check the deployment log for a line saying:

```
Application "risk-management-approuter" started and available at
"<some_URL>"
```

Like in the following screenshot:

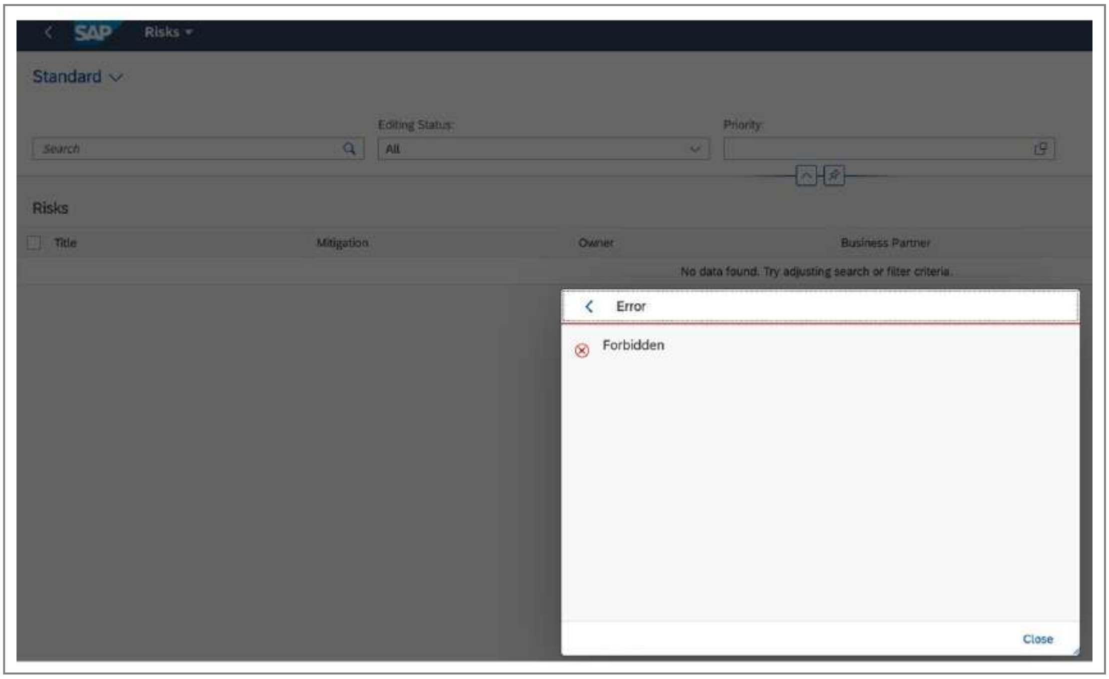


```
Scaling application "risk-management-srv" to "1" instances...
Staging application "risk-management-srv"...
Started async upload of application "risk-management-approuter"
Application "risk-management-db-deployer" staged
Application "risk-management-srv" staged
Starting application "risk-management-srv"...
Scaling application "risk-management-approuter" to "1" instances...
Executing task "deploy" on application "risk-management-db-deployer"...
Application "risk-management-srv" started and available at "cf8efa16trial-dev-risk-management-srv.cfapps.eu10.hana.ondemand.com"
Staging application "risk-management-approuter"...
Application "risk-management-approuter" staged
Starting application "risk-management-approuter"...
Application "risk-management-approuter" started and available at "cf8efa16trial-dev-risk-management-approuter.cfapps.eu10.hana.ondemand.com"
Skipping deletion of services, because the command line option "--delete-services" is not specified.
Process finished.
Use "cf dmol -i 88ccfd99-d4ca-11eb-b70f-eeee0a9edce6" to download the logs of the process.
```

This is the URL of the Approuter, that, from now on, will take care of all the requests to your application.

4. Copy the URL and open it in a new browser tab. In the background, the approuter authenticates your user and provides it with the authorizations that have been assigned to the user. Open the *Risks* application tile and try to retrieve data by selecting the *Go* button.

As your user does not have the required application roles yet, the result should look like the following:



No data found. Try adjusting search or filter criteria.

Forbidden

Summary

You added the UI and the approuter to the MTA file. Next, you will assign the required role collections to your user and check if your user regains access to the application data.

Unit 6
Exercise 9

Add Approuter to MTA

Simulation: Add Approuter to MTA

For more information on *Add Approuter to MTA*, please view the simulation in the lesson *Add the UI and Approuter Module to the MTA* in your online course.

1.

Unit 6
Solution 9

Add Approuter to MTA

Simulation: Add Approuter to MTA

For more information on *Add Approuter to MTA*, please view the simulation in the lesson *Add the UI and Approuter Module to the MTA* in your online course.

1.

Reference Links

For your convenience this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 12: Reference Links: Addition of the Approuter and UI Module to the MTA

Ref#	Section			
1	Re-Build and Re-Deploy the .mtar File	Additional Documentation: How to build an MTA archive from the project sources	Build an MTA archive	https://sap.github.io/cloud-mta-build-tool/usage/#how-to-build-an-mta-archive-from-the-project-sources

LESSON SUMMARY

You should now be able to:

- Add the UI and approuter module to the MTA

Unit 6
Lesson 6

Assign Role Collections

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Assign role collections

Assign Role Collections: Exercise Overview

Scenario

You will add a role collection to access your deployed application through the SAP BTP cockpit with an authenticated and authorized user.

As the name implies, role collections "group together authorizations for resources and services. Role collections consist of individual roles. Role collections are account-specific. Role collections that exist in the global account don't exist in the subaccounts. Likewise, role collections in subaccounts aren't available in the global account." [1].

The way this application is designed, the deployed service can only be accessed when a user has a corresponding role collection assigned. If users tried to open the application without this authorization, they would get a Forbidden message. In this exercise, you will see how to assign the role collection to a user.

Task Flow

In this exercise, you will perform the following tasks:

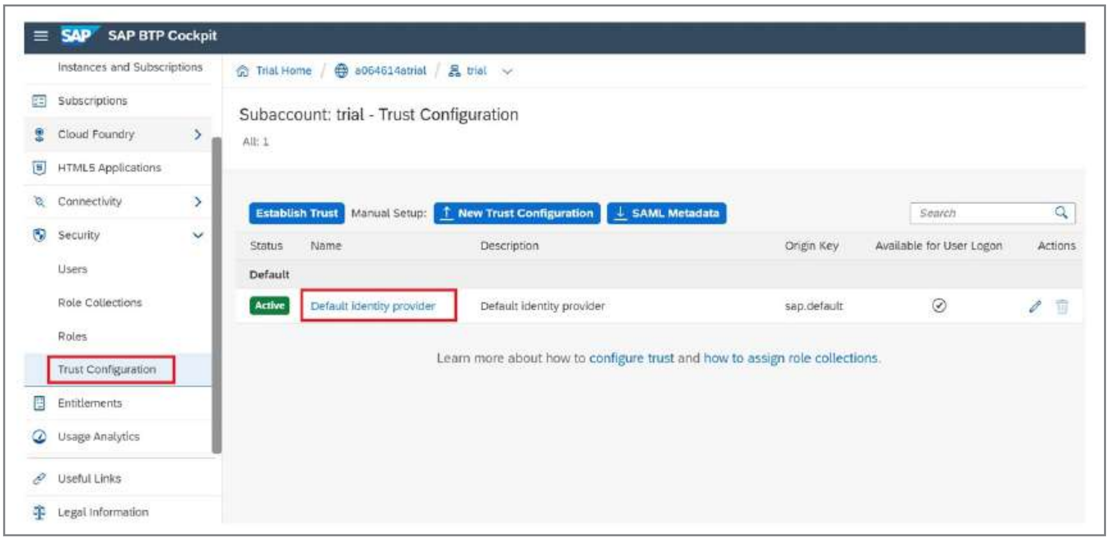
- Assign a role collection.
- Explain role collections.

Prerequisite

You added the UI and approuter to the `mta.yaml` file.

Assign Role Collections

1. In your trial account in the SAP BTP cockpit, navigate to the *Security* tab and choose *Trust Configuration*. Choose the *Default identity provider* link.



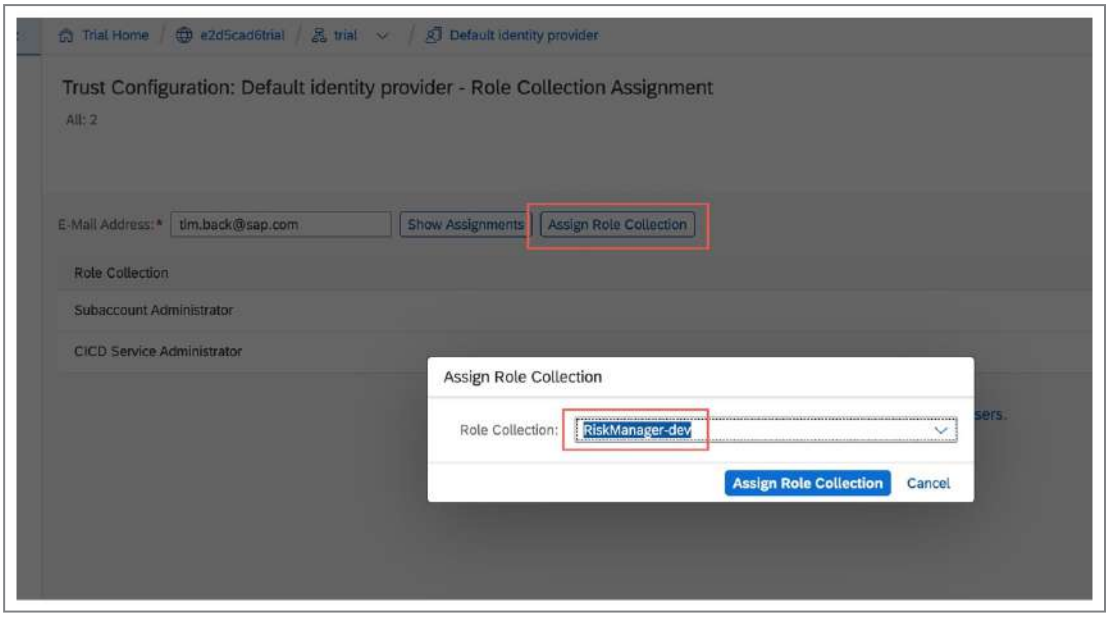
Subaccount: trial - Trust Configuration

All: 1

Status	Name	Description	Origin Key	Available for User Logon	Actions
Default					
Active	Default identity provider	Default identity provider	sap.default	✓	

Learn more about how to configure trust and how to assign role collections.

2. Enter the email address for your SAP BTP account (the one you used to log on) and choose the *Show Assignments* button. Then select the *Assign Role Collection* button and choose *RiskManager-dev* in the dropdown box. Select the *Assign Role Collection* button.



Trust Configuration: Default identity provider - Role Collection Assignment

All: 2

E-Mail Address:* tim.back@sap.com

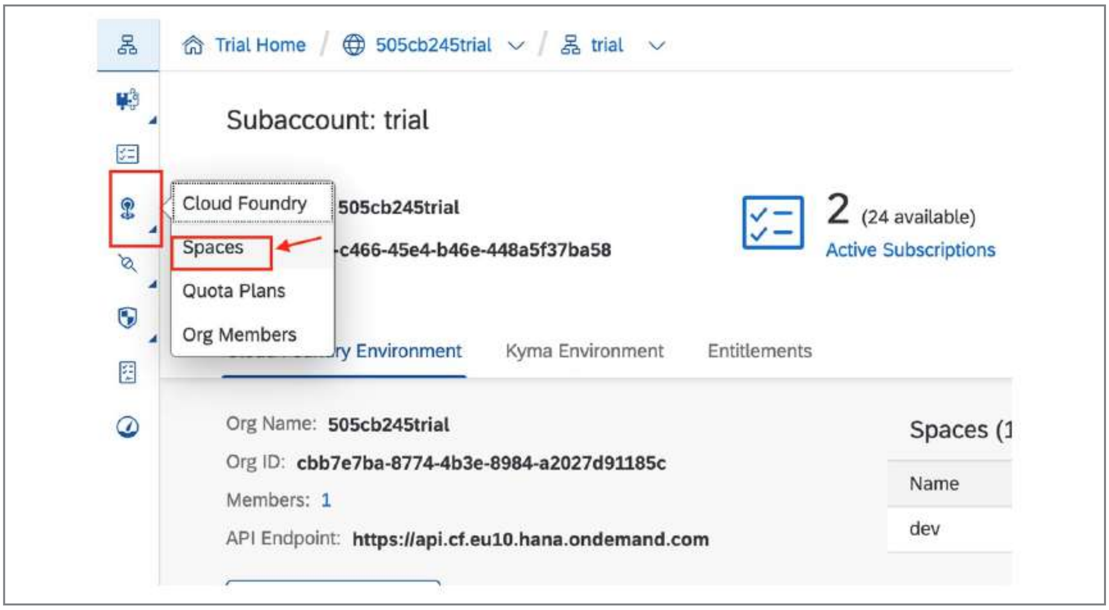
Role Collection
Subaccount Administrator
CICD Service Administrator

Assign Role Collection

Role Collection: RiskManager-dev

Now you are ready to access the application!

3. In the left pane of your trial account, navigate to the *Cloud Foundry* tab and choose *Spaces*.



Subaccount: trial

505cb245trial

-c466-45e4-b46e-448a5f37ba58

Org Name: 505cb245trial

Org ID: cbb7e7ba-8774-4b3e-8984-a2027d91185c

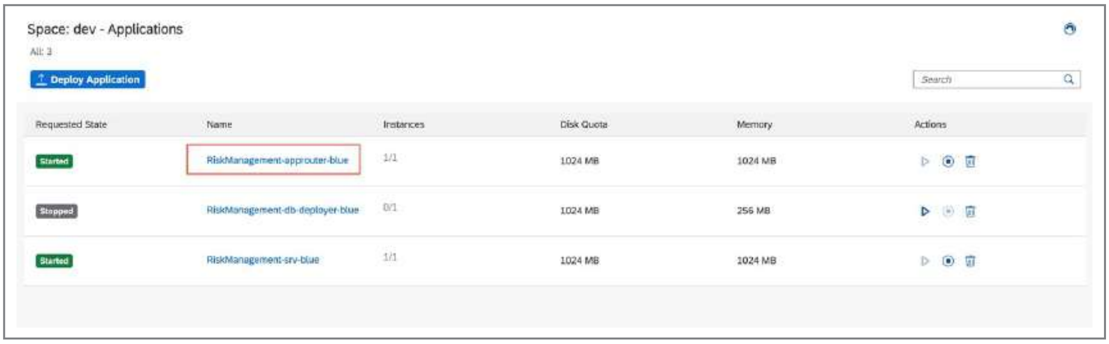
Members: 1

API Endpoint: https://api.cf.eu10.hana.ondemand.com

Name
dev

4. Select your space.

5. Verify that the risk-management application has been deployed and that the *risk-management-approuter* is running.



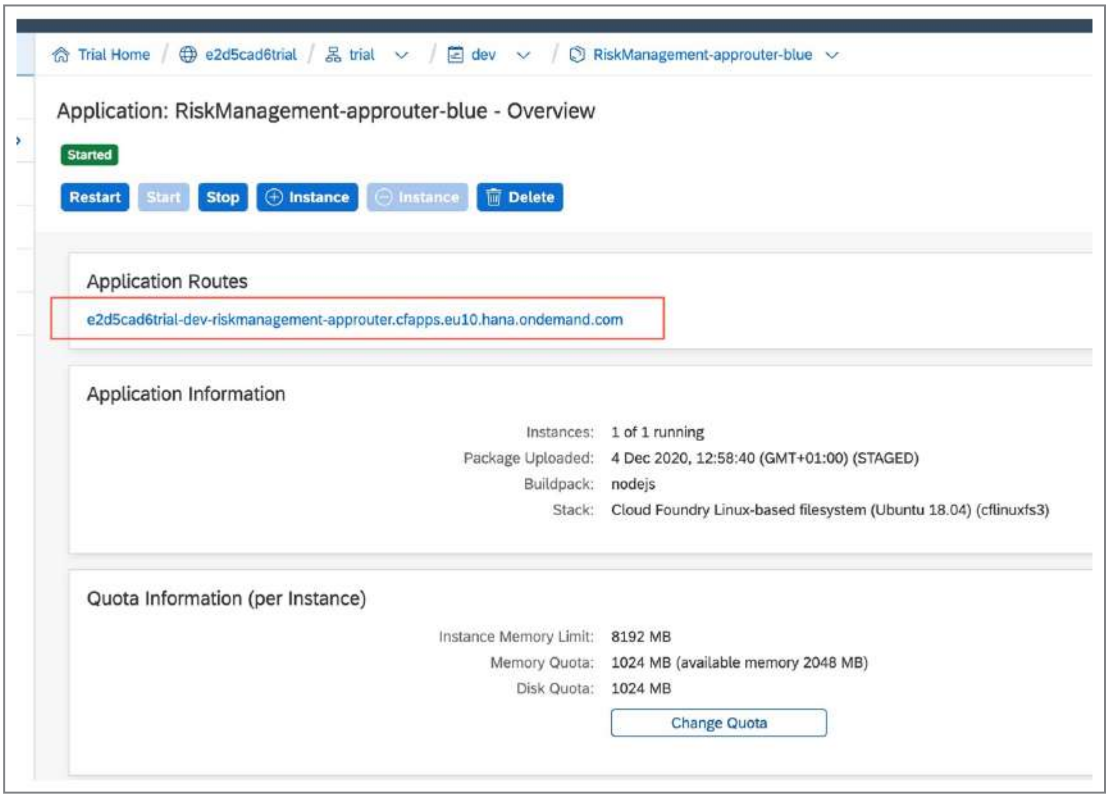
Space: dev - Applications

All: 3

Requested State	Name	Instances	Disk Quota	Memory	Actions
Started	RiskManagement-approuter-blue	1/1	1024 MB	1024 MB	▷ ⦿ 🗑
Stopped	RiskManagement-db-deployer-blue	0/1	1024 MB	256 MB	▷ ⦿ 🗑
Started	RiskManagement-srv-blue	1/1	1024 MB	1024 MB	▷ ⦿ 🗑

6. Choose the `risk-management-approuter`.

7. Choose the link under *Application Routes*.


Trial Home / e2d5cad6trial / trial / dev / RiskManagement-approuter-blue
Started
Restart
Start
Stop
Instance
Instance
Delete
Change Quota

Application: RiskManagement-approuter-blue - Overview

Application Routes

e2d5cad6trial-dev-riskmanagement-approuter.cfapps.eu10.hana.ondemand.com

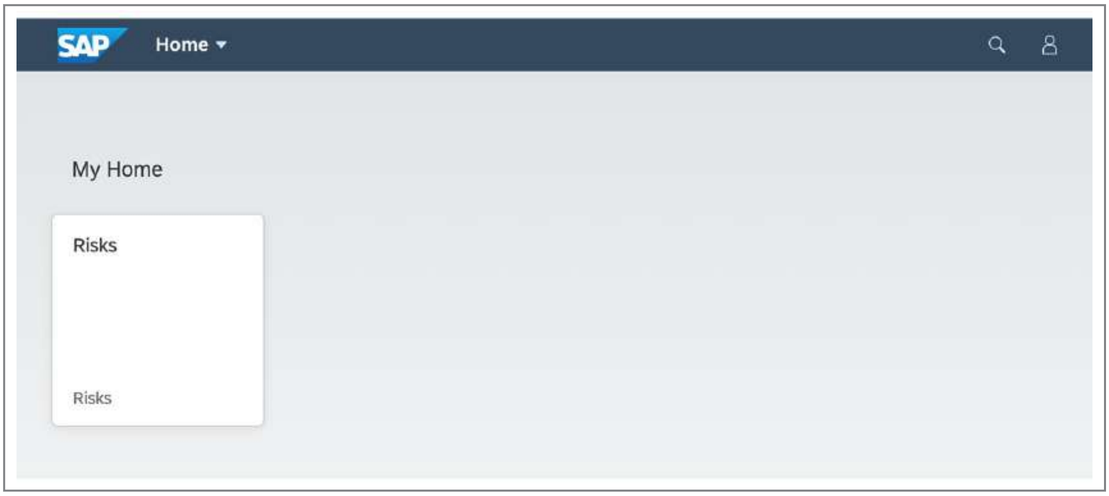
Application Information

Instances: 1 of 1 running
Package Uploaded: 4 Dec 2020, 12:58:40 (GMT+01:00) (STAGED)
Buildpack: nodejs
Stack: Cloud Foundry Linux-based filesystem (Ubuntu 18.04) (cflinuxfs3)

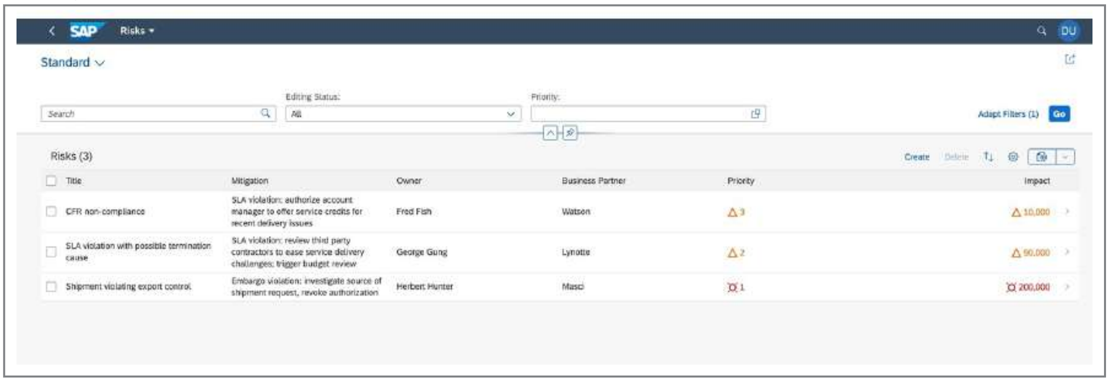
Quota Information (per Instance)

Instance Memory Limit: 8192 MB
Memory Quota: 1024 MB (available memory 2048 MB)
Disk Quota: 1024 MB

8. Verify that the deployed application is running and showing its home screen.



9. Select the *Risks* tile and then in the application the *Go* button, and you should see the final deployed application with the data:



Risks (3)

Title	Mitigation	Owner	Business Partner	Priority	Impact
CFR non-compliance	SLA violation: authorize account manager to offer service credits for recent delivery issues	Fred Fish	Watson	△ 3	△ 10,000
SLA violation with possible termination cause	SLA violation: review third party contractors to ease service delivery challenges; trigger budget review	George Gung	Lynotte	△ 2	△ 90,000
Shipment violating export control	Embargo violation: investigate source of shipment request, revoke authorization	Herbert Hunter	Masci	⊗ 1	⊗ 200,000

Note:
In some cases, your browser might still have cached the authorization information from the previous app call. When you still receive the `Forbidden` message, try to either delete your browser cache and cookies or to close and reopen your browser.

Summary

You have assigned a role collection to access the application through the SAP BTP Cockpit.

Unit 6
Exercise 10

Assign Role Collections

Simulation: Assign Role Collections

For more information on *Assign Role Collections*, please view the simulation in the lesson *Assign Role Collections* in your online course.

1.

Unit 6
Solution 10

Assign Role Collections

Simulation: Assign Role Collections

For more information on *Assign Role Collections*, please view the simulation in the lesson *Assign Role Collections* in your online course.

1.

Reference Links

For your convenience this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 13: Reference Links: Role Collections

Ref#	Section	Context text fragment	Brief description	Link
1	Assign Role Collections	As the name implies, role collections	Role collections	https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/LATEST/en-US/0039cf082d3d43eba9200fe15647922a.html

LESSON SUMMARY

You should now be able to:

- Assign role collections

Learning Assessment

1. What can you install to add authentication to your CAP application?

   Choose the correct answer.

   □ A Node.js

   □ B RiskManager

   □ C Passport.js

2. Which files do you modify to store project configurations?

   Choose the correct answers.

   □ A .cdsrc.json

   □ B passport.js

   □ C package.json

3. Which modules must you install to enable authentication support in CAP for BTP?

   Choose the correct answers.

   □ A xsjs

   □ B xssecure

   □ C xssec

   □ D xsenv

4. What is the name of a single authorization?

   Choose the correct answer.

   □ A aspect

   □ B role

   □ C scope

   □ D privilege

5. What do XS advanced applications use to perform the authentication process?

Choose the correct answer.

□ A OData 4.0

□ B OAuth 2.0

□ C SAML2

□ D UMA 2.0

6. Which user types will work on and with SAP BTP?

Choose the correct answers.

□ A Professional users

□ B Platform users

□ C Trial users

□ D Business users

7. What is the default identity provider for SAP BTP platform users?

Choose the correct answer.

□ A SAP Cloud Identity Services

□ B SAP ID service

□ C SAML 2.0

□ D XSUAA

8. What does the Extended Services - User Account and Authentication (XSUAA) service enable your app to do?

Choose the correct answers.

□ A Store "real" users.

□ B Identify users by address and social security ID.

□ C Identify users by e-mail, userId, first and last name.

□ D Check users' roles to allow or prohibit actions.

9. Which file contains an app's "declaration of security"?

   Choose the correct answer.

   □ A xs-sec.json

   □ B xs-app.json

   □ C xs-security.json

10. Why do you need an approuter for your application?

    Choose the correct answers.

    □ A To connect requests from the web browser to a service

    □ B To route microservices to the web browser that runs your extension

    □ C To connect requests from the web browser to the provider of UI sources

    □ D To replace static URLs with dynamic content

11. The managed approuter is a service in SAP BTP.

    Determine whether this statement is true or false.

    □ True

    □ False

12. What are responsibilities of the approuter?

    Choose the correct answers.

    □ A Hide backend microservices.

    □ B Dispatch requests to backend microservices (reverse proxy).

    □ C Authenticate users.

    □ D Serve static content.

    □ E Provide network isolation.

13. What does the .env file provide?

    Choose the correct answer.

    □ A Values for your version-management-system

    □ B Values into the runtime environment of a CAP service

14. Role collections of a subaccount are available in the global account.

Determine whether this statement is true or false.

□ True

□ False

15. Role collections of a global account are available in a subaccount of this global account?

Determine whether this statement is true or false.

□ True

□ False

16. In the SAP BTP Cockpit, where can you assign role collections?

Choose the correct answer.

□ A *Security → Roles*

□ B *Security → Trust Configuration*

UNIT 7 Automated Deployment

Lesson 1

Lesson 2

Lesson 3

Lesson 4

Lesson 5

Lesson 6

UNIT OBJECTIVES

- Create and connect a GitHub repository
- Describe the principles and benefits of continuous integration and delivery
- Enable SAP Continuous Integration and Delivery
- Configure a job in the SAP Continous Integration and Delivery service
- Configure stages of the CI/CD pipeline
- Start monitoring a job in SAP Continuous Integration and Delivery

Unit 7
Lesson 1

Create and Connect a GitHub Repository

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Create and connect a GitHub repository

Create and Connect a GitHub Repository: Exercise Overview

Scenario

In this exercise, you will create a public GitHub[1] repository for the source code of your application.

Task Flow

In this exercise, you will perform the following tasks:

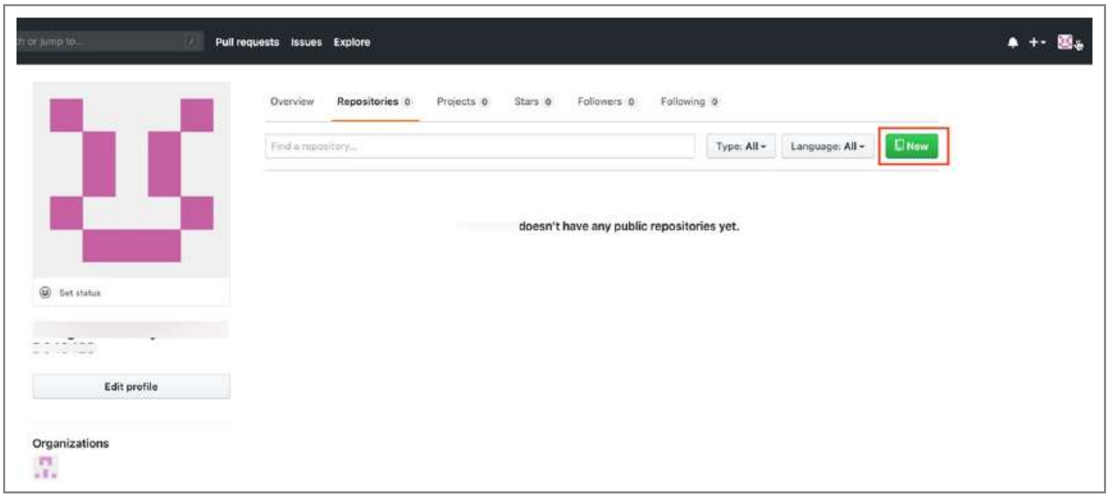
- Create a public GitHub repository to store the source code of a project.
- Create a personal access token for GitHub.
- Connect a CAP project with a GitHub repository using git commands.

Prerequisite

To create a GitHub repository, you need a CAP project and a GitHub user. If you do not have a GitHub user, sign up for GitHub first.

Create a Public GitHub Repository

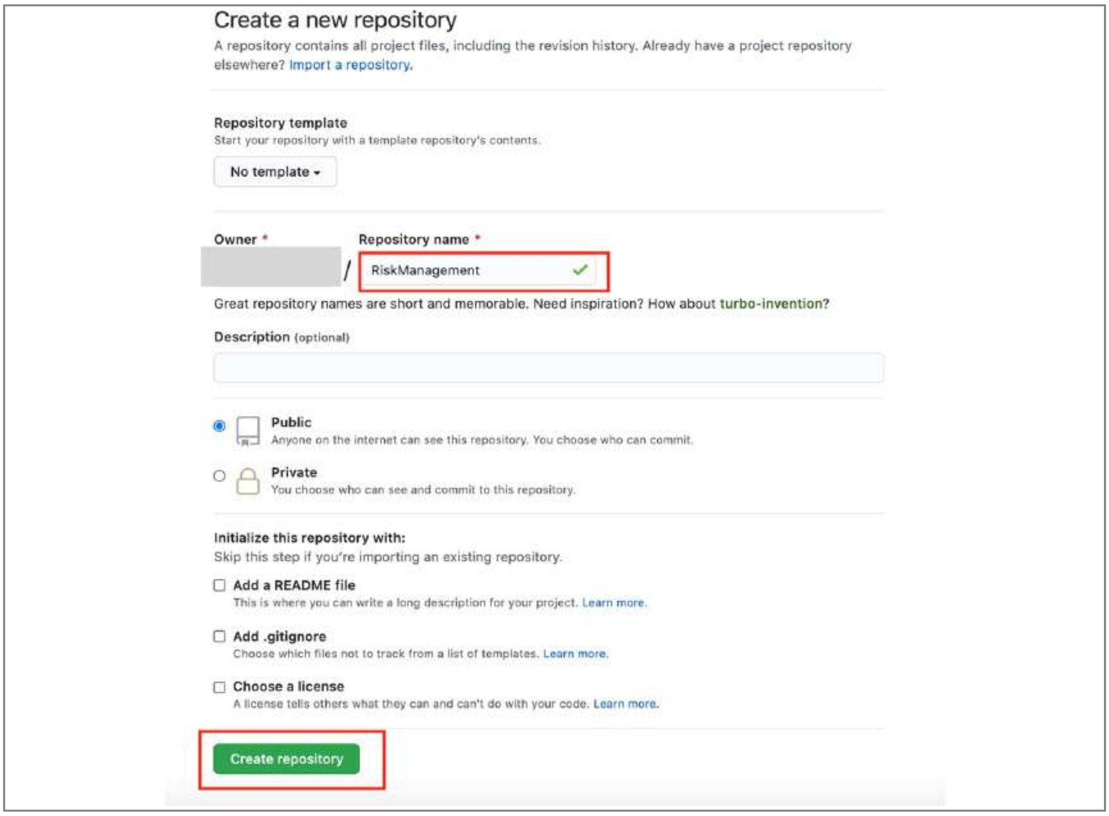
1. Open and sign in to https://github.com/.
2. In the *Repositories* tab, choose *New* to create a new repository.



doesn't have any public repositories yet.

3. As the *Repository name*, enter **`RiskManagement`**. Don't tick any of the *Initialize this repository with* checkboxes.

4. Choose *Create repository*



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner * [redacted] / Repository name * RiskManagement ✓

Great repository names are short and memorable. Need inspiration? How about turbo-invention?

Description (optional)

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

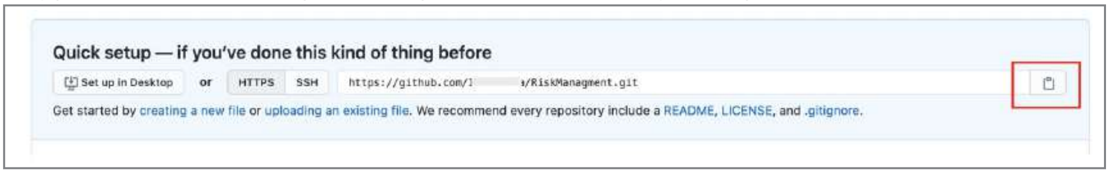
□ Add a README file
This is where you can write a long description for your project. Learn more.

□ Add .gitignore
Choose which files not to track from a list of templates. Learn more.

□ Choose a license
A license tells others what they can and can't do with your code. Learn more.

.

5. Copy the HTTPS URL of your newly created GitHub repository.



Quick setup — if you've done this kind of thing before

https://github.com/l[redacted]a/RiskManagment.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

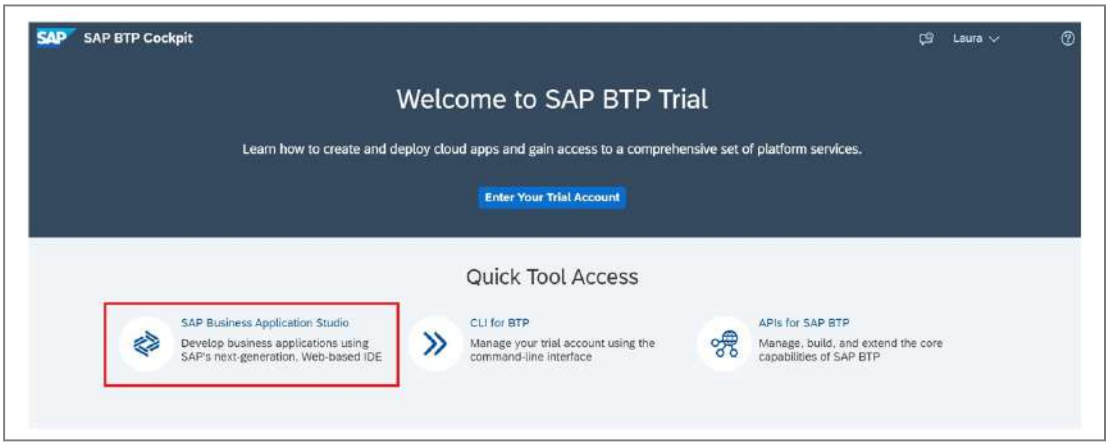
Create a Personal Access Token for GitHub

To create a personal access token, which you can use instead of a password, follow the steps described in Creating a personal access token [2].

Connect Your GitHub Repository with Your CAP Project

You have created a new, empty GitHub repository. To be able to use it as a repository for your risk management application source code, you need to connect it with your CAP project. Until now, the only place where your project's source code resided was your personal dev space in SAP Business Application Studio.

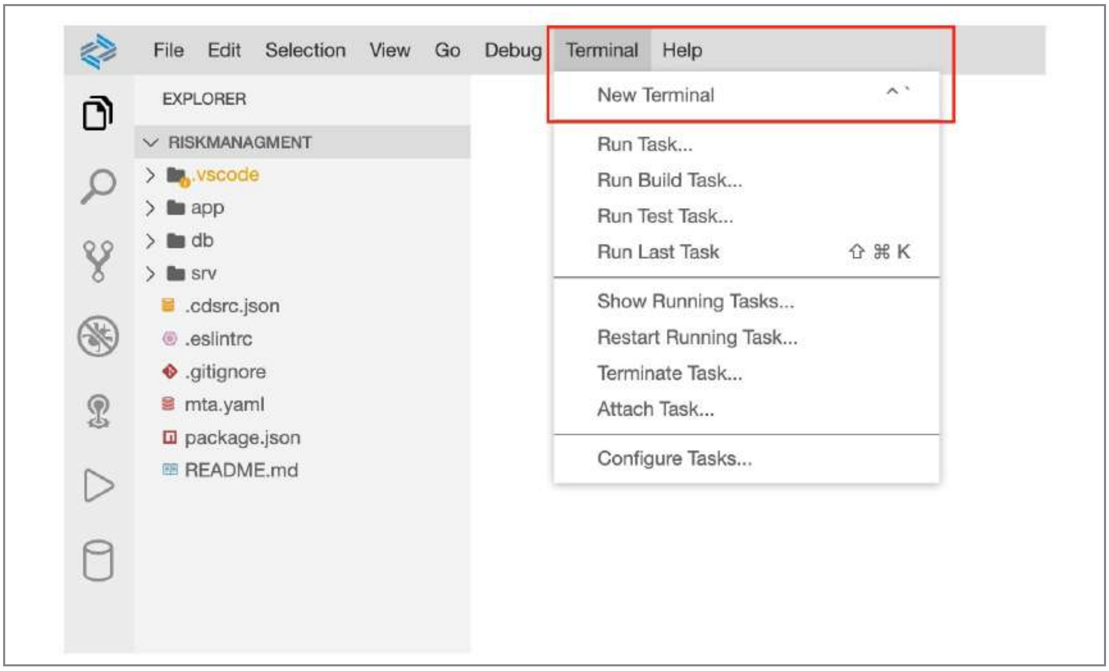
1. If you have SAP Business Application Studio still open from the former exercises, return to it. If you haven't, open your SAP BTP Trial [3] and choose the *Quick Tool Access* icon *SAP Business Application Studio*.


SAP BTP Cockpit
Laura
Welcome to SAP BTP Trial
Learn how to create and deploy cloud apps and gain access to a comprehensive set of platform services.
Enter Your Trial Account
Quick Tool Access
SAP Business Application Studio
Develop business applications using SAP's next-generation, Web-based IDE
CLI for BTP
Manage your trial account using the command-line interface
APIs for SAP BTP
Manage, build, and extend the core capabilities of SAP BTP

.

2. Open the previously created workspace with your CAP application.

3. Open a new terminal and navigate to your project root folder.



4. Enter your email address and username. You can use the email address that you've used to register your GitHub account:

```
git config --global user.email "you@example.com"
git config --global user.name "Your Name"
```

   This configures the git command line tool locally in your Business Application Studio dev space. Now when you do file changes and commit them to a git history, git will use the configured name and email in these commits.

5. To initialize a local git repository and add the project sources to it, execute the following commands within the `root` directory of your CAP project:

   a. Initialize a new local git repository.

   ```
   git init
   ```

   b. Add all directories and files to the git staging area.

```
git add .
```

c. Create the first commit in your git repository.

```
git commit -m "Push project content to GitHub"
```

d. Rename the current branch to main

```
git branch -M main
```

6. Add your copied GitHub repository URL from the previous part of the exercise as remote repository (without the angle brackets '<' and '>'):

   ```
   git remote add origin <copied Git repository url.git>
   ```

   This tells your local git repository in the Business Application Studio dev space, that it has a remote counterpart on GitHub. The remote counterpart should act as the *origin*, thus it is the repository that you and your colleagues use as the central repository of your project.

7. Push the commit with your project content to this GitHub repository:

   ```
   git push -u origin main
   ```

   This tells your local git to push the *main* branch to the remote repository. The `-u` option is used because the branch *main* did not yet exist on the remote repository.

8. When prompted, enter your GitHub username and the personal access token, that you have created previously.

> Note:
> The prompt appears in the upper middle of the SAP Business Application Studio, not in the terminal tab that you have used until now.

Summary

You have connected your CAP project with your public GitHub repository using git commands[4].

Unit 7
Exercise 11

Create and Connect a GitHub Repository

Simulation: Create and Connect a GitHub Repository

For more information on *Create and Connect a GitHub Repository*, please view the simulation in the lesson *Create and Connect a GitHub Repository* in your online course.

1.

Unit 7
Solution 11

Create and Connect a GitHub Repository

Simulation: Create and Connect a GitHub Repository

For more information on *Create and Connect a GitHub Repository*, please view the simulation in the lesson *Create and Connect a GitHub Repository* in your online course.

1.

Reference Links

For your convenience, this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 14: Reference Links: GitHub

Ref#	Section	Context text fragment	Breif description	Link
1	Create and Connect a GitHub Repository	the steps described in Creating a personal access token	GitHub token	https://github.com/
2	Create a Personal Access Token for GitHub	you will create a public GitHub repository	Project GitHub	https://docs.github.com/en/github/authenticating-to-github/keeping-your-account-and-data-secure/creating-a-personal-access-token
3	Connect Your GitHub Repository with Your CAP Project	open your SAP BTP Trial	SAP BTP trial	https://cockpit.hanatrial.ondemand.com/
4	Summary	public GitHub repository using git commands	Git commands	https://git-scm.com/docs

LESSON SUMMARY

You should now be able to:

- Create and connect a GitHub repository

Unit 7
Lesson 2

Continuous Integration and Delivery

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Describe the principles and benefits of continuous integration and delivery

Continuous Integration and Continuous Delivery (CI/CD)

Usage Scenario

Your company develops a set of cloud applications in a dedicated development environment. Usually, this environment is the developers' computer or even a cloud-based Integrated Development Environment (IDE), such as the SAP Business Application Studio (BAS). Locally, you can run and test your application at any time. The manual build and deployment is often very tedious and not convenient at all. To automate these repetitive build and deployment steps, you can set up an automation. CI/CD pipelines are well suited for this.

Animation: Usage Scenario

For more information on *Usage Scenario*, please view the animation in the lesson *Continuous Integration and Delivery*, in your online course.

Learning objectives

- Explain the philosophy and concepts of CI/CD pipelines.
- Describe the benefits of integrating a pipeline into your project.

What is a CI/CD Pipeline?

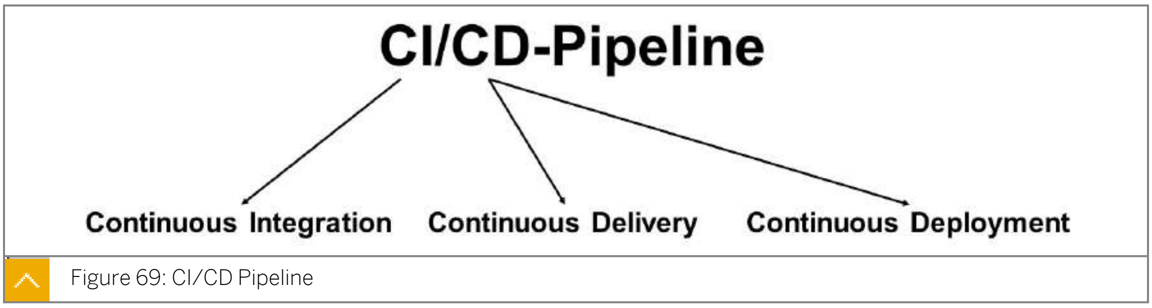


Figure 69: CI/CD Pipeline

The goal of CI/CD is to automate as many steps of software development as possible in order to minimize manual effort. There are a variety of ways to achieve this.

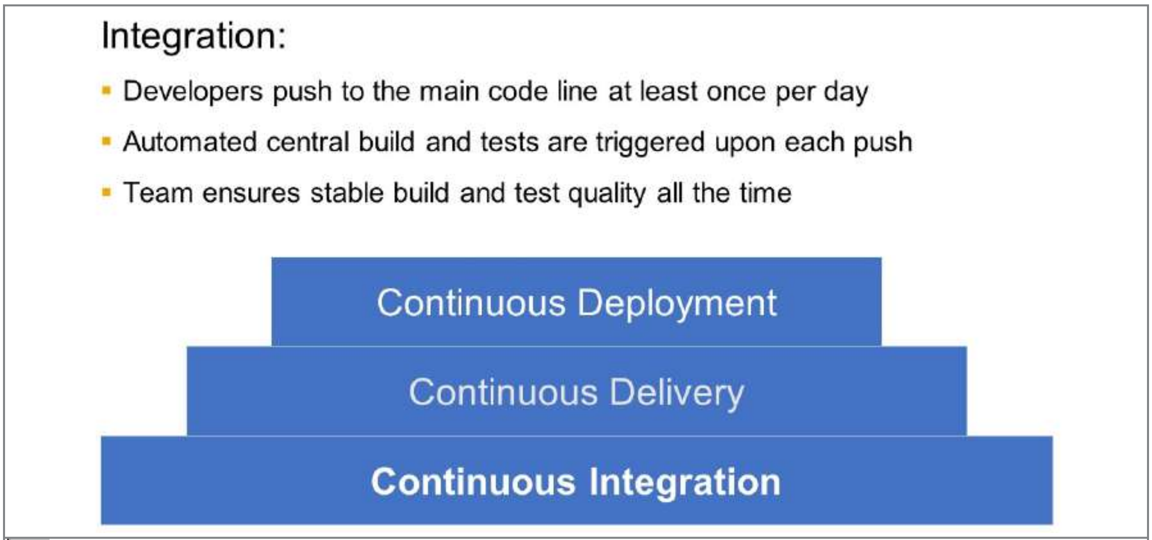
But before diving deeper, let's clarify the question: what is a pipeline?

Well, at its simplest, it is a series of activities that are carried out in a predefined order.

A CI/CD pipeline's goal is to automate as many steps of software development as possible to reduce manual effort.

Now, there is some confusion around the acronym CI/CD. To clarify: "CI" stands for Continuous Integration, while the "CD" can stand for either Continuous Delivery or Continuous Deployment.

Continuous Integration



Integration:

- Developers push to the main code line at least once per day
- Automated central build and tests are triggered upon each push
- Team ensures stable build and test quality all the time

Continuous Deployment

Continuous Delivery

Continuous Integration

Figure 70: Integration

Each of the different Continuous X types does not stand isolated or side by side, instead they build upon each other. Continuous Integration is the foundation, which includes several principles. Ultimately, there is always a stable build available.

Continuous Integration (CI): Basic Flow

Watch this video to learn about the basic flow of continuous integration.

Animation: Continuous Integration (CI): Basic Flow

For more information on *Continuous Integration (CI): Basic Flow*, please view the animation in the lesson *Continuous Integration and Delivery*, in your online course.

The following picture states the basic flow of a continuous integration. The developer writes code and pushes its code changes to a centralized and remote source code management system (SCM) like GitHub. The SCM then triggers the CI server. Usually, this is done via so-called webhooks.

After sending the event to the CI server, the server itself then automatically starts the pre-configured build and unit tests. Finally, the CI server sends the feedback to the developer.

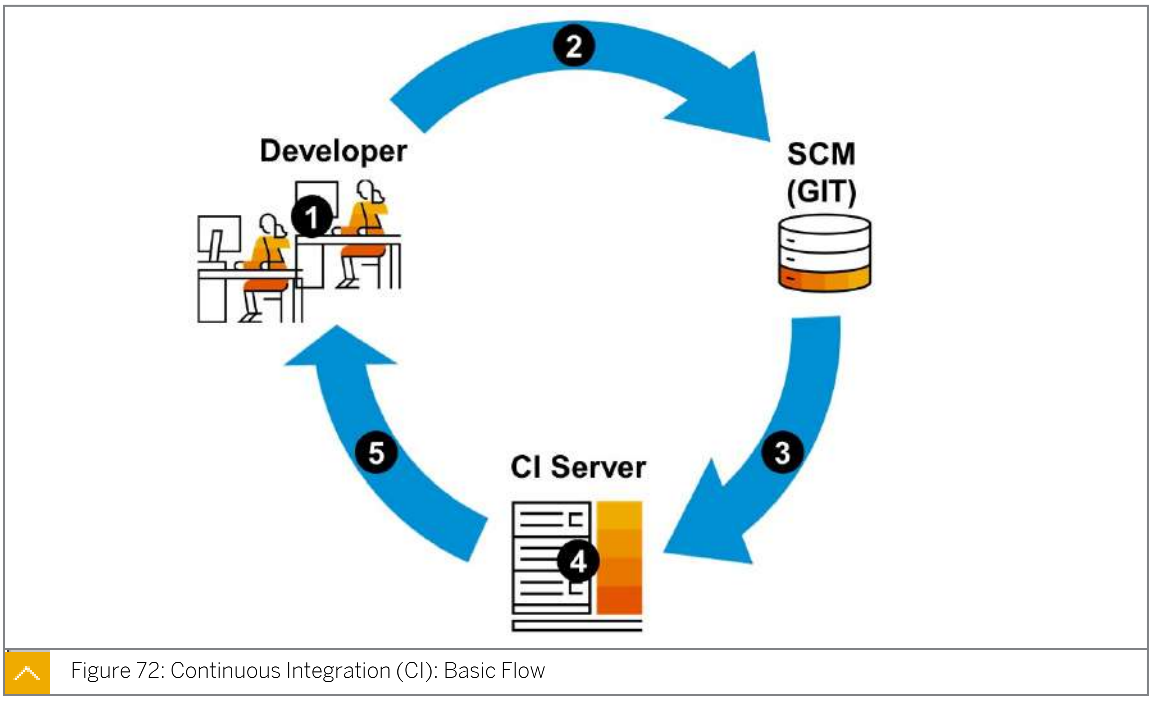


Figure 72: Continuous Integration (CI): Basic Flow

Continuous Delivery

On top of Continuous Integration is Continuous Delivery. While a stable build is always available with CI, Continuous Delivery defines the software in such a way that it is ready for deployment on the production system, whereby the trigger for the deployment is a human decision. Deployment must therefore be triggered manually, or automatically, for example just by pressing a button.

Facts about Continuous Delivery:

- Software is ready for deployment to a productive system all the time.
- Deployment to a productive system is triggered manually.
- Feedback from a productive system gets quickly integrated into teams' backlog.

Continuous Deployment

On top of the Continuous Delivery, Continuous Deployment means, that the deployment to the productive system is triggered with each commit. It is important to be aware that Continuous Delivery and Continuous Deployment are sometimes not clearly separated. Which means that some sources (like blogs, books,...) talk about Continuous Deployment while they mean Continuous Delivery! To avoid misunderstandings, you should always clarify these definitions when talking about CI/CD.

Facts about Continuous Deployment:

Deployment to a productive system is triggered automatically (instead of manual deployment as in Continuous Delivery).

The Pipeline

Putting all these pieces together, you can create a fully automated pipeline to build, test, and deploy your application.

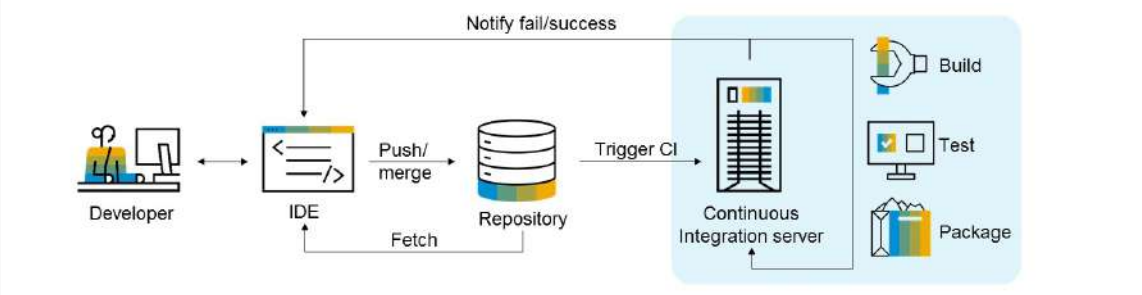


Figure 73: Continuous Integration (CI): Pipeline

Summary

By now, you have a more profound understanding of the core principles and benefits of a CI/CD pipeline.

In short, Continuous Integration (CI) is the adoption of agile principles while Continuous Delivery/Deployment (CD) is a combination of agile methodology techniques and a high-quality delivery process. The goal is to validate each change (commit), preferably in an automated way, so that it can potentially be delivered in a reliable manner.

Further Reading

These resources might be helpful if you want to dive deeper into CI/CD.

- SAP Solutions for CI/CD

LESSON SUMMARY

You should now be able to:

- Describe the principles and benefits of continuous integration and delivery

Unit 7
Lesson 3

Enable SAP Continuous Integration and Delivery

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Enable SAP Continuous Integration and Delivery

Enable SAP Continuous Integration and Delivery: Exercise Overview

Scenario

In this exercise, you will enable the SAP Continuous Integration and Delivery (CI/CD) service for your project.

"Continuous integration (CI) describes a software development process, in which various team members integrate their contributions frequently into a single main line. Before each integration, the changes are verified through builds and automated testing. Thereby, you can detect errors as quickly as possible and prevent integration problems before completing the development."

"Continuous delivery (CD) adds the aspect that any change that has successfully passed the tests is immediately ready to be deployed to production, both from a technical and a qualitative point of view."

For more information about CI/CD such as CI principles and process flows, see:

- Continuous Integration Principles[1]
- Continuous Integration and Continuous Delivery Guide[2]

.

Task Flow

In this exercise, you will perform the following tasks:

- Subscribe to SAP Continuous Integration and Delivery.
- Assign the Administrator role to your user.

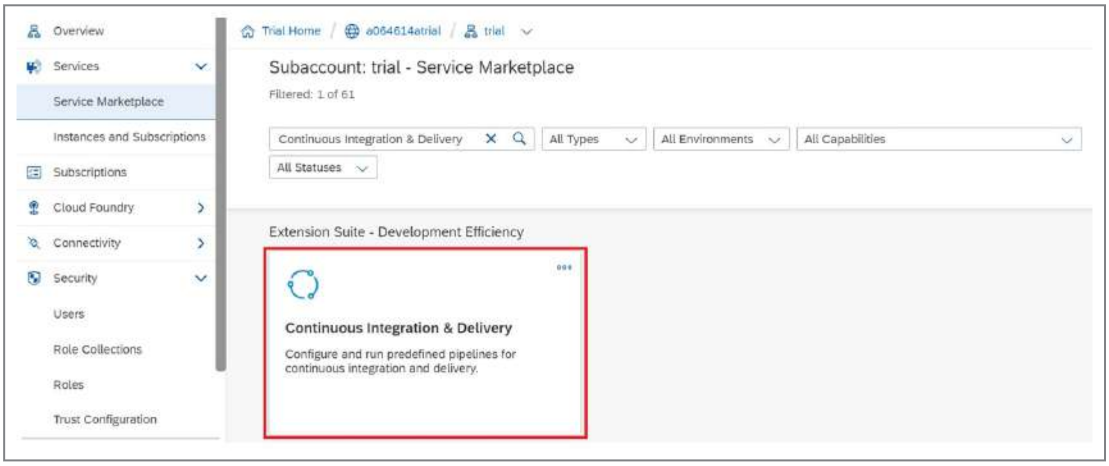
Prerequisite

You have connected your CAP project to your GitHub repository.

Create and Administer Continuous Integration and Delivery Service

1. Enter your SAP BTP trial account[3], choose *Enter Your Trial Account*, choose the *trial* tile and navigate to the *Services* tab. Go to *Service Marketplace*.

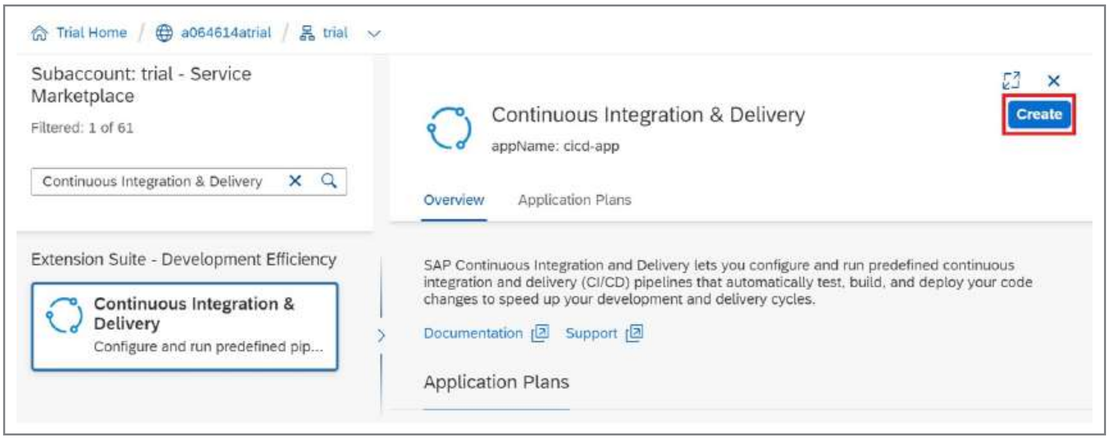
2. Enter *Continuous Integration & Delivery* in the search box. If the tile is displayed, select it and proceed to step 3.


Overview
Services
Service Marketplace
Instances and Subscriptions
Subscriptions
Cloud Foundry
Connectivity
Security
Users
Role Collections
Roles
Trust Configuration
Trial Home / a064614atrial / trial
Subaccount: trial - Service Marketplace
Filtered: 1 of 61
Continuous Integration & Delivery
All Types
All Environments
All Capabilities
All Statuses
Extension Suite - Development Efficiency
Continuous Integration & Delivery

Configure and run predefined pipelines for continuous integration and delivery.

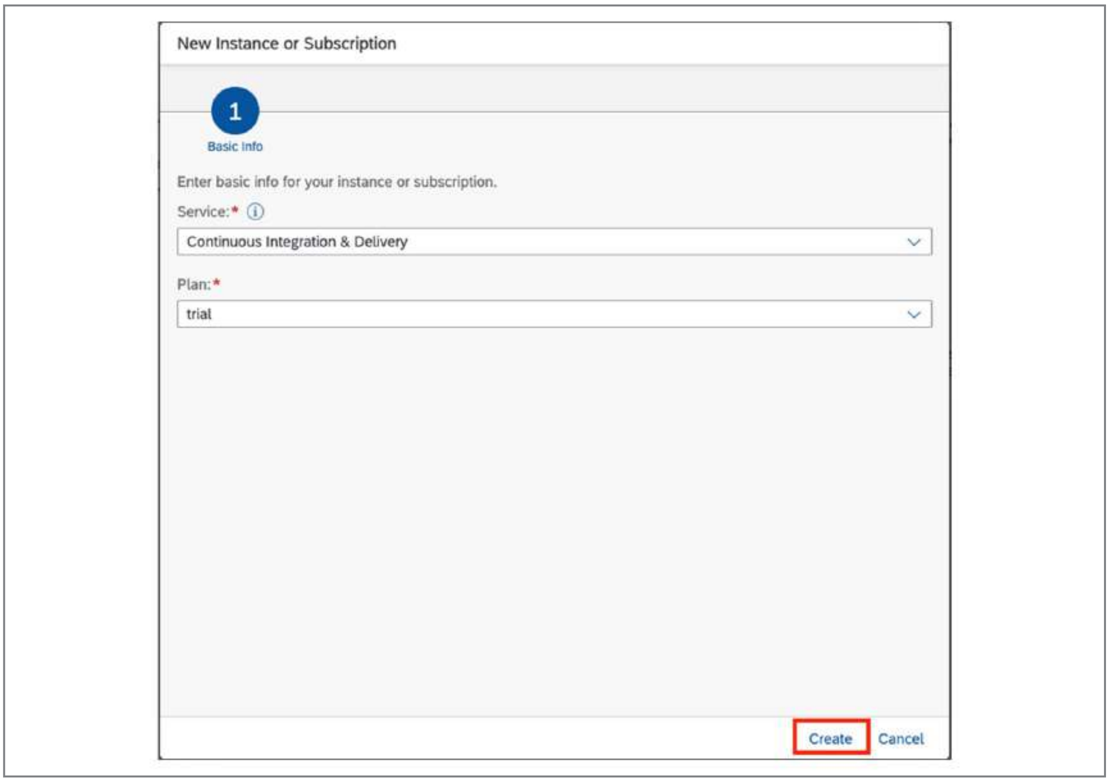
If the tile is not displayed, it must be entitled. To achieve this, select *Entitlements* in the left bar of your trial subaccount, then on the upper-right side, select *Configure Entitlements*, then *Add Service Plans*. A pop-up displays, on the left side of the pop-up select the *Continuous Integration & Delivery* entitlement and on the right side select the *trial* flag. Confirm the pop-up (select *Add 1 Service Plan*). Navigate back to the *Service Marketplace* of your trial subaccount and try again. The tile should now display.

3. Choose *Create*.


Trial Home / a064614atrial / trial
Subaccount: trial - Service Marketplace
Filtered: 1 of 61
Continuous Integration & Delivery
Extension Suite - Development Efficiency
Continuous Integration & Delivery
Configure and run predefined pip...
Continuous Integration & Delivery
appName: cicd-app
Create
Overview
Application Plans
Documentation
Support
Application Plans

SAP Continuous Integration and Delivery lets you configure and run predefined continuous integration and delivery (CI/CD) pipelines that automatically test, build, and deploy your code changes to speed up your development and delivery cycles.

4. Accept the default settings and choose *Create*.



New Instance or Subscription

Enter basic info for your instance or subscription.

Service:* Continuous Integration & Delivery

Plan:* trial

5. In your SAP BTP subaccount, choose *Security* → *Trust Configuration*.

6. Choose the name of your identity provider (For example, 'Default identity provider' in this case).

7. Enter your email address.

8. Choose *Show Assignments*. (If your user is new to your subaccount, choose *Add User* in the confirmation dialog.)

9. Choose *Assign Role Collection*.

10. From the list, select *CICD Service Administrator* and choose *Assign Role Collection*.

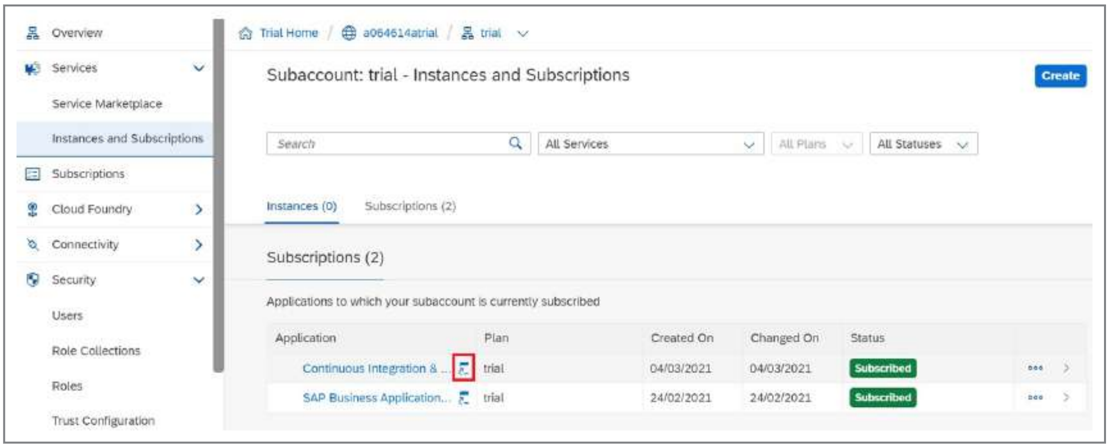
    Your user is now authorized to use and administer the Continuous Integration and Delivery service.

Configure Credentials in SAP Continuous Integration and Delivery

The SAP Continuous Integration and Delivery service will act as an automation tool. It will retrieve the latest state of your GitHub repository, build your project and deploy it to your SAP BTP, Cloud Foundry environment. You need to configure GitHub and SAP BTP credentials for the service to be able to connect to these environments.

1. In your SAP BTP subaccount navigate to *Services* and then to *Instances and Subscriptions*.

2. Choose the *Go to Application* icon located next to the *Continuous Integration & Delivery* subscription.



Subaccount: trial - Instances and Subscriptions

Subscriptions (2)

Applications to which your subaccount is currently subscribed

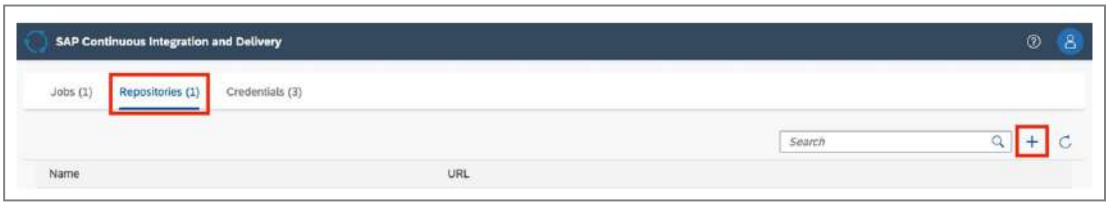
Application	Plan	Created On	Changed On	Status
Continuous Integration & ...	trial	04/03/2021	04/03/2021	Subscribed
SAP Business Application...	trial	24/02/2021	24/02/2021	Subscribed

3. Use your credentials to log in to the application.

4. In the *Credentials* tab in *SAP Continuous Integration and Delivery*, choose + *(Create Credentials)*.

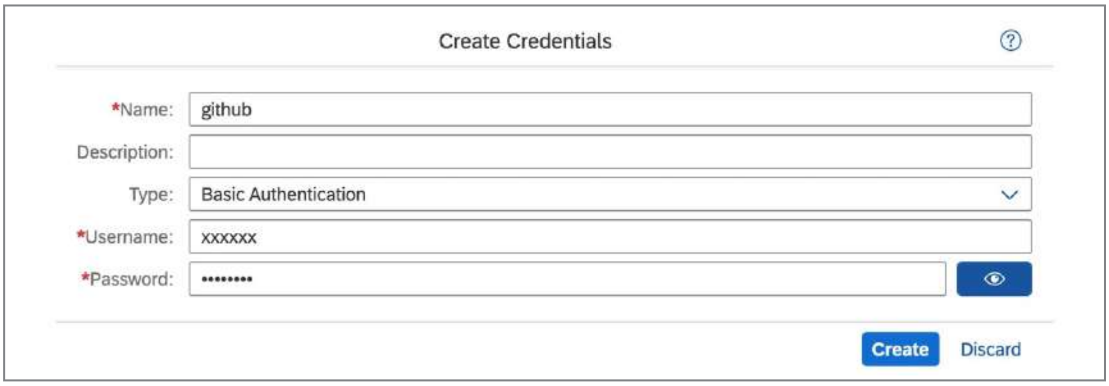
   If your GitHub repository is private, configure credentials for it, so that the *Continuous Integration & Delivery* service can connect to it.

   > Note:
   > If your GitHub repository isn't private, you can skip this step.



5. Enter the following data:

Field	Value
Name	<a freely chosen name for your credential, which is unique in your SAP BTP subaccount>. In this example, the name of the credential is **`github`**.
Type	<select> Basic Authentication
Username	<your GitHub username>
Password	<use the personal access token, which you've created in GitHub in a previous exercise>



*Name: github

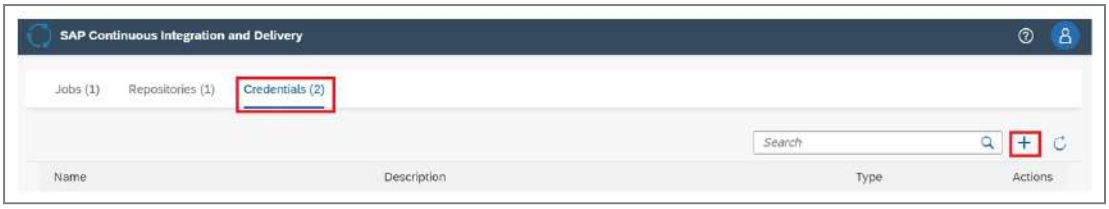
Description:

Type: Basic Authentication

*Username: xxxxxx

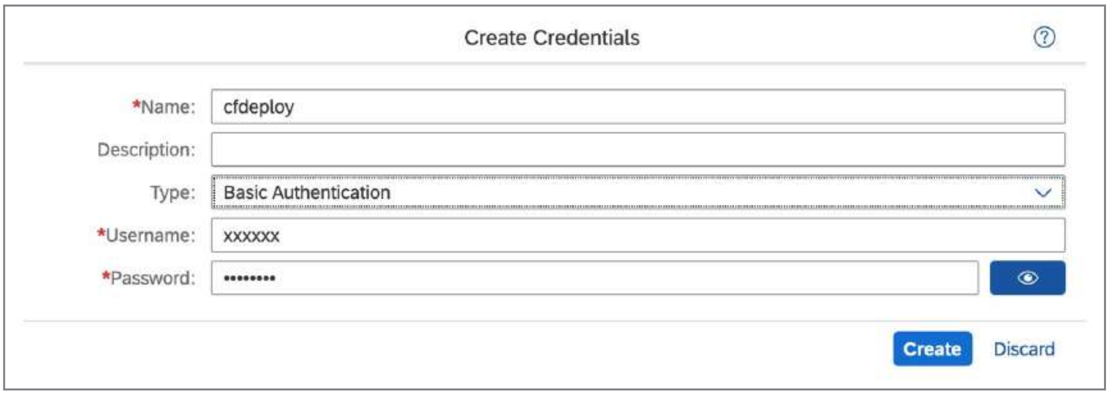
*Password: ••••••••

6. To create credentials for deploying to the SAP BTP, Cloud Foundry environment, go to the *Credentials* tab and choose + *(Create Credentials)*.



7. Enter the following data:

Field	Value
Name	<enter a freely chosen name for your credentials, which is unique in your SAP BTP subaccount.> In this example, the name of the credentials is **`cfdeploy`**
Type	<select> Basic Authentication
Username	<your username for the SAP BTP cockpit>
Password	<use your password for the SAP BTP cockpit>



*Name: cfdeploy

Description:

Type: Basic Authentication

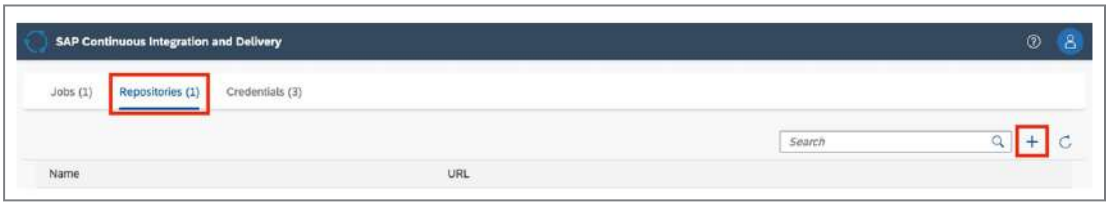
*Username: xxxxxx

*Password: ••••••••

Configure GitHub Repository in SAP Continuous Integration and Delivery

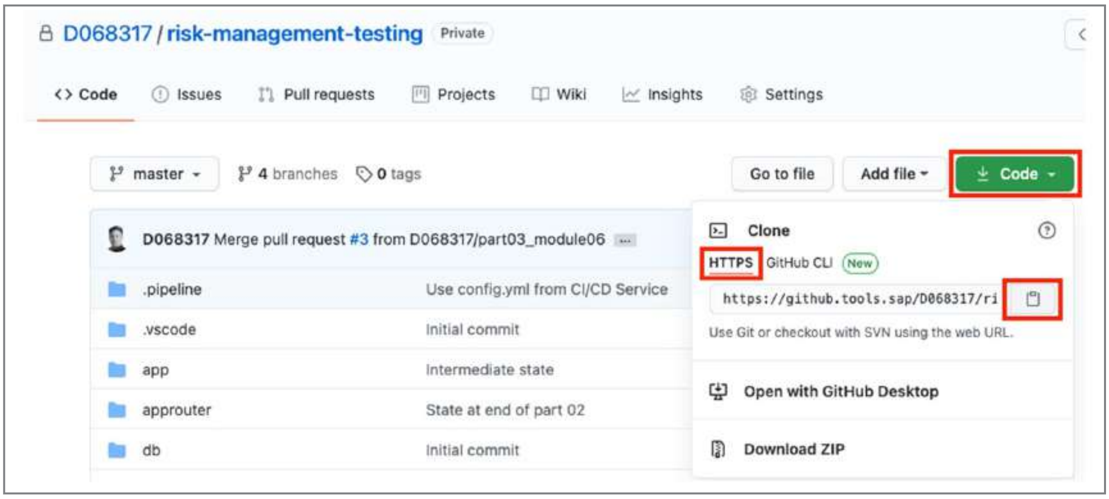
The SAP Continuous Integration and Delivery service can manage multiple repositories. You need to configure your GitHub repository so that the service is able to retrieve sources from it.

1. In the *Repositories* tab in SAP Continuous Integration and Delivery, choose + *(Add repository)*.



2. For *Name*, enter a freely chosen name for your repository, which is unique in your SAP BTP subaccount. In this example, the name of the repository is **`risk-management-repo`**.

3. As *Clone URL* enter the HTTPS Clone URL of your GitHub repository. Find it by selecting the green *Code* button in your GitHub repository. Choose the *Clipboard* icon to copy the URL.



D068317 Merge pull request #3 from D068317/part03_module06

.pipeline	Use config.yml from CI/CD Service
.vscode	Initial commit
app	Intermediate state
approuter	State at end of part 02
db	Initial commit

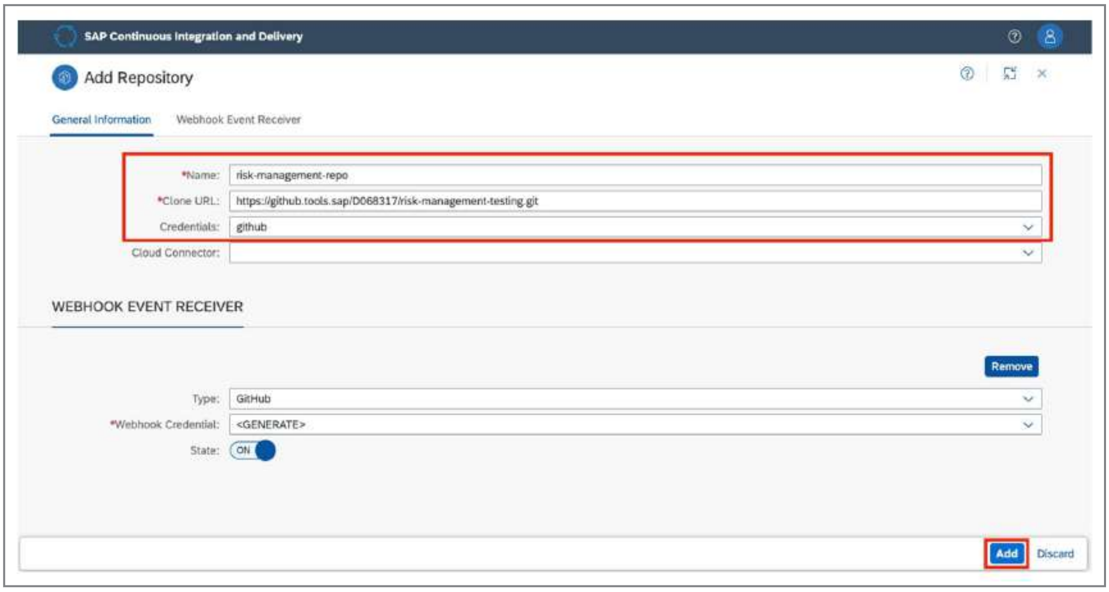
https://github.tools.sap/D068317/ri

Use Git or checkout with SVN using the web URL.

4. As *Credentials*, select the **`github credentials`** that you created in the previous steps.

5. Leave the default values in the *Webhook Event Receiver* section. Make sure that *Type* is set to **`GitHub`**.

6. Select *Add* to add the repository to your SAP Continuous Integration and Delivery service.



*Name: risk-management-repo

*Clone URL: https://github.tools.sap/D068317/risk-management-testing.git

Credentials: github

Cloud Connector:

Type: GitHub

*Webhook Credential: <GENERATE>

State: ON

Summary

You have enabled SAP Continuous Integration and Delivery for your project.

Unit 7
Exercise 12

Enable SAP Continuous Integration and Delivery

Simulation: Enable SAP Continuous Integration and Delivery

For more information on *Enable SAP Continuous Integration and Delivery*, please view the simulation in the lesson *Enable SAP Continuous Integration and Delivery* in your online course.

1.

Enable SAP Continuous Integration and Delivery

Simulation: Enable SAP Continuous Integration and Delivery

For more information on *Enable SAP Continuous Integration and Delivery*, please view the simulation in the lesson *Enable SAP Continuous Integration and Delivery* in your online course.

1.

Reference Links

For your convenience, this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 15: Reference Links: Enable SAP Continuous Integration and Delivery

Ref#	Section	Context text fragment	Brief description	Link
1	Scenario	Continuous Integration Principles	Continuous Integration principles	Continuous Integration Principles
2	Scenario	Continuous Integration and Continuous Delivery Guide	Overview of the continuous integration and delivery concepts	Continuous Integration and Continuous Delivery Guide
3	Continuous Delivery	SAP BTP Trial Account	SAP BTP Cockpit	SAP BTP trial account

LESSON SUMMARY

You should now be able to:

- Enable SAP Continuous Integration and Delivery

Configure a CI/CD Job

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Configure a job in the SAP Continous Integration and Delivery service

Configure a CI/CD Job: Exercise Overview

Scenario

You have enabled SAP Continuous Integration and Delivery for your GitHub repository and for deployment into SAP BTP, Cloud Foundry environment. But until now, you have not told the service what exactly it is supposed to do for you. After completing the next steps, you will have configured a job in SAP Continuous Integration and Delivery. The job is the actual automation part of this course.

Task Flow

In this exercise, you will perform the following task:

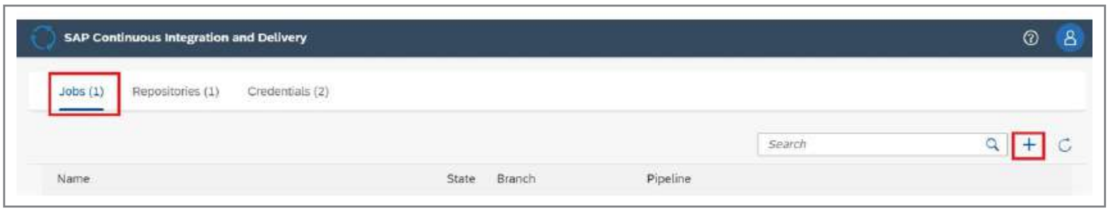
- Configure a job in the SAP Continuous Integration and Delivery service.

Prerequisite

You have enabled the SAP Continous Integration and Delivery service for your GitHub repository and for deployment into SAP BTP, Cloud Foundry environment.

Create Job

1. In the *Jobs* tab in SAP Continuous Integration and Delivery, choose + *(Create Job)*.



2. Enter the following data:

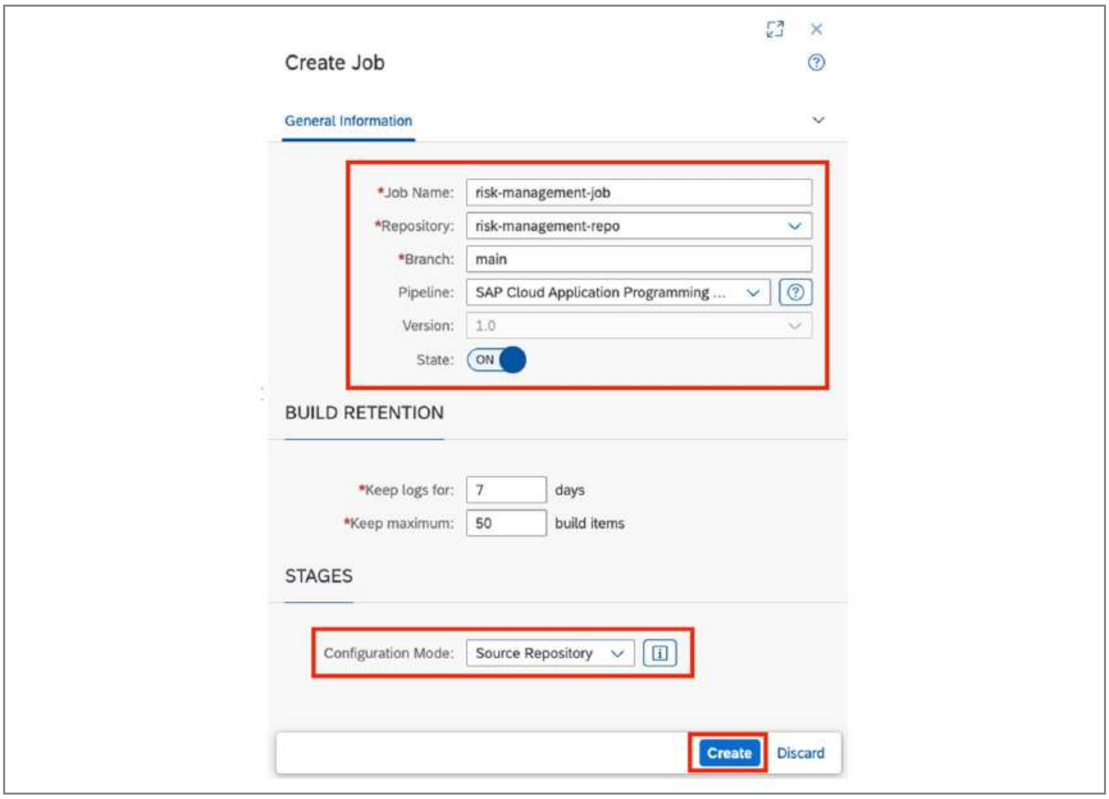
Key	Value
Job Name	A freely chosen name for your job, which is unique in your SAP BTP subaccount. In this example, the name of the job is *risk-management-job*.
Repository	<Choose the repository you created earlier from the dropdown> (for example, *risk-management-repo*).

Key	Value
Branch	<Enter the GitHub branch from which you want to receive push events. In this example, the name of the branch is *main*>
Pipeline	<Choose> SAP Cloud Application Programming Model

3. Scroll down to *Stages* and select **`Source Repository`** as the *Configuration Mode*.

   This tells your job, that the actual stages, the work that is to be performed, will be defined within the source repository itself. You are going to configure the stages in a following part.

4. Choose *Create*.



Create Job

General Information

*Job Name: risk-management-job

*Repository: risk-management-repo

*Branch: main

Pipeline: SAP Cloud Application Programming ...

Version: 1.0

State: ON

BUILD RETENTION

*Keep logs for: 7 days

*Keep maximum: 50 build items

STAGES

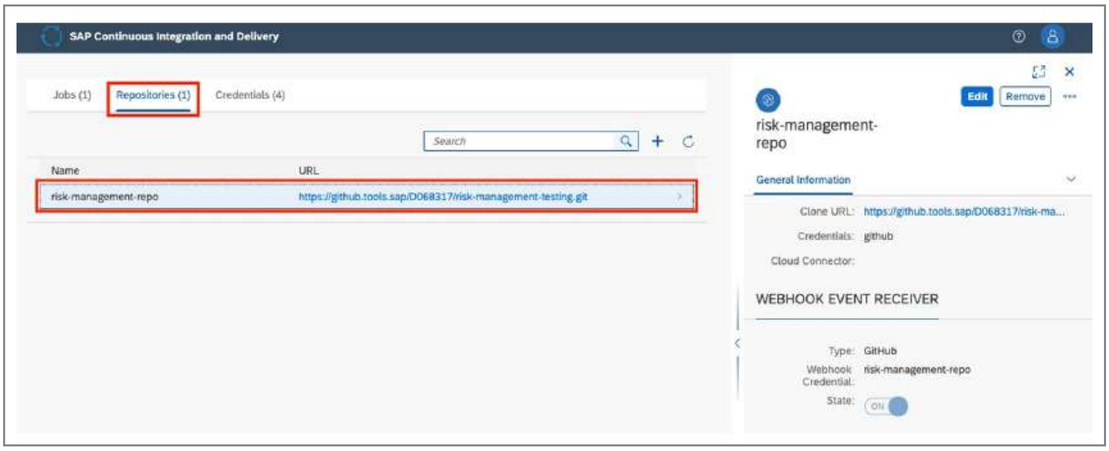
Configuration Mode: Source Repository

Configuring a Webhook

The goal is for your SAP Continuous Integration and Delivery service job to run whenever there is a change on the main branch of your GitHub repository. To retrieve the information about a change on the repository, you make use of a webhook. There is a Webhook Event Receiver for your repository in the SAP Continuous Integration and Delivery service, and you need to configure the webhook itself in your GitHub repository.

1. Open the *Repositories* tab.

2. Select your repository, in this case *risk-management-repo*.



SAP Continuous Integration and Delivery

Jobs (1) | Repositories (1) | Credentials (4)

Name	URL
risk-management-repo	https://github.tools.sap/D068317/risk-management-testing.git

risk-management-repo

Edit | Remove | •••

General Information

Clone URL: https://github.tools.sap/D068317/risk-ma...

Credentials: github

Cloud Connector:

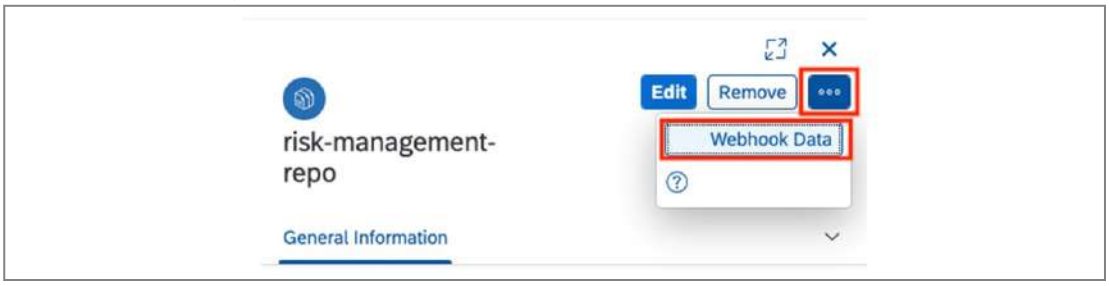
WEBHOOK EVENT RECEIVER

Type: GitHub

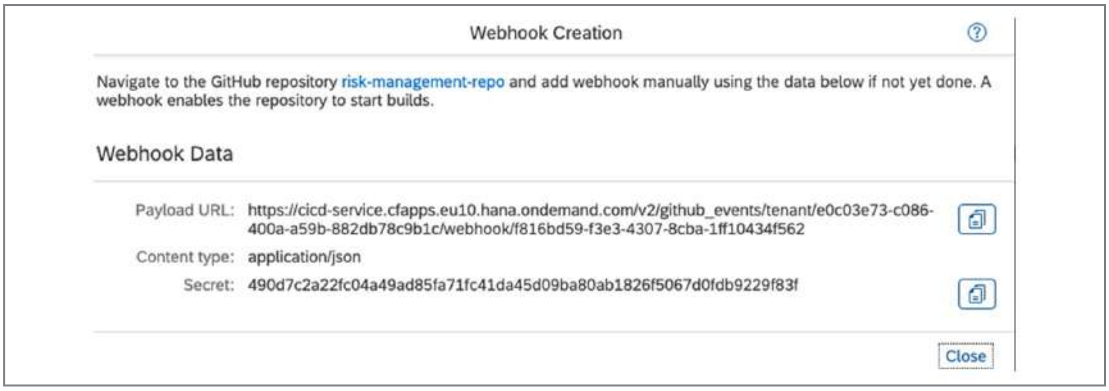
Webhook Credential: risk-management-repo

State: ON

3. Select the ... *(More)* button and choose *Webhook Data*.



The following pop-up window appears:



Webhook Creation

Navigate to the GitHub repository risk-management-repo and add webhook manually using the data below if not yet done. A webhook enables the repository to start builds.

Webhook Data

Payload URL: https://cicd-service.cfapps.eu10.hana.ondemand.com/v2/github_events/tenant/e0c03e73-c086-400a-a59b-882db78c9b1c/webhook/f816bd59-f3e3-4307-8cba-1ff10434f562

Content type: application/json

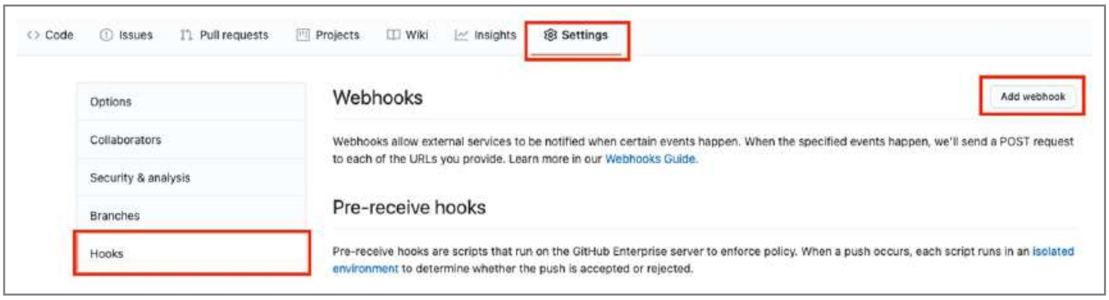
Secret: 490d7c2a22fc04a49ad85fa71fc41da45d09ba80ab1826f5067d0fdb9229f83f

Close

4. Leave it open and, in your project in GitHub, go to the *Settings* tab.

5. From the navigation pane, choose *Webhooks* or *Hooks*.

6. Choose *Add webhook*.



Webhooks

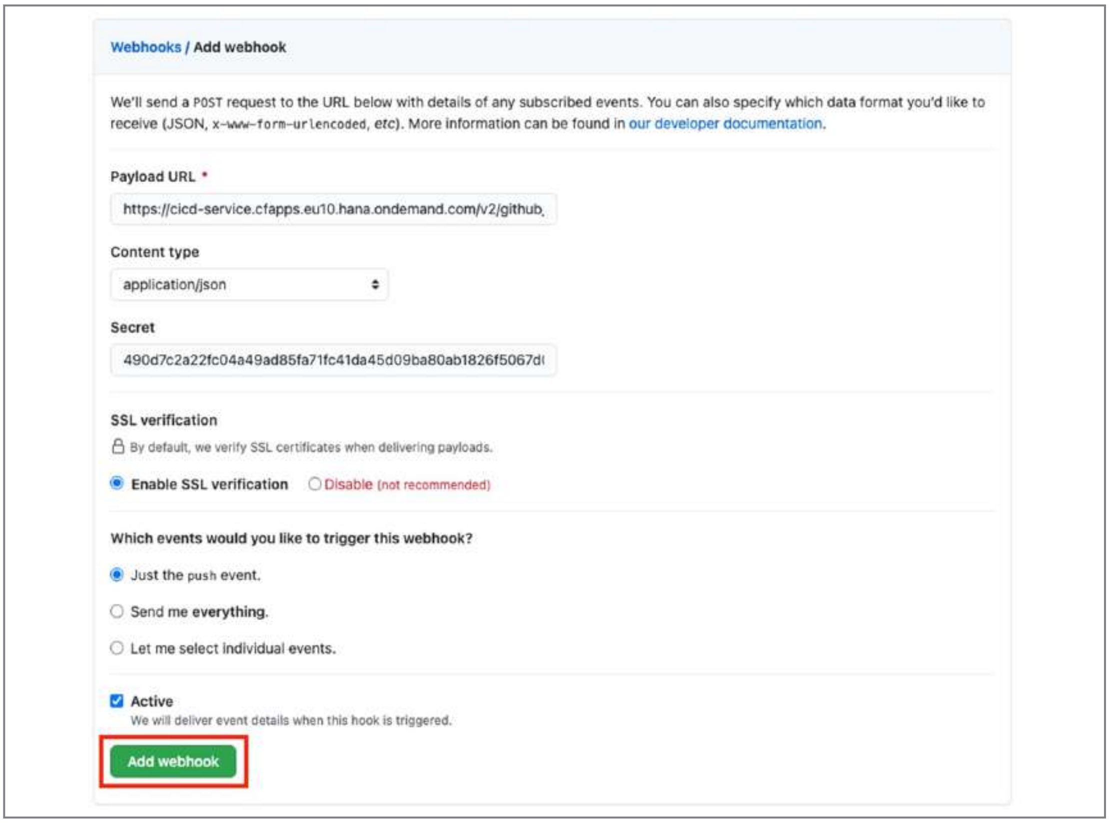
Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

Pre-receive hooks

Pre-receive hooks are scripts that run on the GitHub Enterprise server to enforce policy. When a push occurs, each script runs in an isolated environment to determine whether the push is accepted or rejected.

7. Enter the *Payload URL, Content type*, and *Secret* from the *Webhook Creation* pop-up in SAP Continuous Integration and Delivery. For all other settings, leave the default values.

8. Choose *Add webhook*.



Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in our developer documentation.

Payload URL *

https://cicd-service.cfapps.eu10.hana.ondemand.com/v2/github_

Content type

application/json

Secret

490d7c2a22fc04a49ad85fa71fc41da45d09ba80ab1826f5067d

SSL verification

By default, we verify SSL certificates when delivering payloads.

● Enable SSL verification ○ Disable (not recommended)

Which events would you like to trigger this webhook?

● Just the push event.

○ Send me **everything**.

○ Let me select individual events.

☑ Active

We will deliver event details when this hook is triggered.

Now, whenever there is a push event to any branch of your repository, the webhook will be triggered. The webhook then notifies your SAP Continuous Integration and Delivery service, using the specified *Payload URL* and *Secret*.

Summary

You have configured a SAP Continuous Integration and Delivery job, which automates the build and deployment process of your project.

Unit 7
Exercise 13

Configure a CI/CD Job

Simulation: Configuring a CI/CD Job

For more information on *Configuring a CI/CD Job*, please view the simulation in the lesson *Configure a CI/CD Job* in your online course.

1.

Unit 7
Solution 13

Configure a CI/CD Job

Simulation: Configuring a CI/CD Job

For more information on *Configuring a CI/CD Job*, please view the simulation in the lesson *Configure a CI/CD Job* in your online course.

1.

LESSON SUMMARY

You should now be able to:

- Configure a job in the SAP Continous Integration and Delivery service

Unit 7
Lesson 5

Configure Stages of the CI/CD Pipeline

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Configure stages of the CI/CD pipeline

Configure the Stages of a CI/CD Pipeline: Exercise Overview

Scenario

You have connected your SAP Continuous Integration and Delivery service to your GitHub repository and have created a job that is going to take care of automated build and deployment. Now the job needs to know what steps it needs to perform. The definition of steps that need to be performed is done using a so-called pipeline. The configuration syntax of the pipeline that you are going to configure is based on SAP's Open-Source Project Piper [1]. The pipeline configuration is done within your project.

Task Flow

In this exercise, you will configure stages of a project "Piper" pipeline.

Prerequisite

You have enabled your application for CI/CD.

Configure Stages of a CI/CD Pipeline

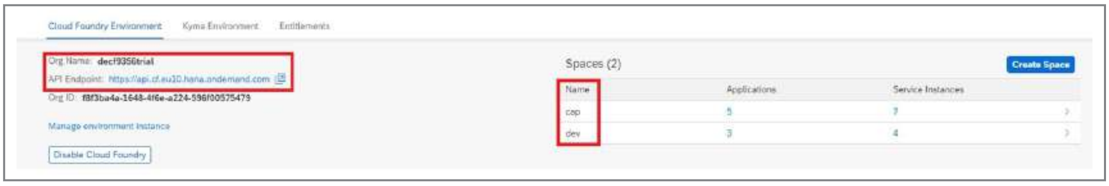
1. In the GitHub repository of your project (or in SAP Business Application Studio), execute the following command in a terminal session: `cds add pipeline`

2. Navigate into the *.pipeline* folder and open the *config.yml* file, and replace everything with the following configuration:

```
###
 # This file configures the project "Piper" pipeline of your project.
 # For a reference of the configuration concept and available options,
please have a look into its documentation.
#
# The documentation for the most recent pipeline version can always be
found at:
# https://sap.github.io/jenkins-library/
#
# This is a YAML-file. YAML is an indentation-sensitive file format.
Please make sure to properly indent changes to it.
###

### General project setup
---
 general:
    pipeline: "sap-cloud-sdk"
    buildTool: "mta"
 stages:
    Build:
       mavenExecuteStaticCodeChecks: false
       npmExecuteLint: false
    Additional Unit Tests:
       npmExecuteScripts: false
       karmaExecuteTests: false
    Release:
       cloudFoundryDeploy: true
       tmsUpload: false
    steps:
      cloudFoundryDeploy:
        cloudFoundry:
          apiEndpoint: "https://api.cf.eu10.hana.ondemand.com"
          org: "myOrg"
          space: "mySpace"
          credentialsId: "myDeploymentCredentialsId" # set this to
'cfdeploy'
          appName: ""
        mtaDeployParameters: "-f --version-rule ALL"
     artifactPrepareVersion:
        versioningType: "cloud_noTag"
```

3. Replace the placeholders with the values of the space in the SAP BTP, Cloud Foundry environment to which you want to deploy.`credentialsId` is the name of the credential you have created in the CI/CD Service for the SAP BTP access. In previous parts, you named it `cfdeploy`.

   You can get the values for `*myOrg*`, `*mySpace*`, and `*apiEndpoint*` from your subaccount overview in the SAP BTP cockpit:



Org Name: dec9356trial
API Endpoint: https://api.cf.eu10.hana.ondemand.com
Org ID: f8f3ba4a-1648-4f6e-a224-596f00575479

Spaces (2)

Name	Applications	Service Instances
cap	5	7
dev	3	4

   Or just run command `CF TARGET` in the Terminal in SAP Business Application Studio.

4. Commit your changes to GitHub. If you have added the new folder and file in the Business Application Studio, you can do so by opening a new terminal from the new `.pipeline` folder and then by performing the following steps:

   a. Navigate into the `.pipeline` directory:

   ```
   cd .pipeline
   ```

   b. Add all recent changes into the git staging area:

   ```
   git add -A
   ```

   c. Create a new commit with the staged changes and push it to the remote repository:

```
git commit -m "new pipeline"
git push
```

Summary

You have configured your CI/CD pipeline. Finally, you will have to verify the success of your build.

Unit 7
Exercise 14

Configure the Stages of a CI/CD Pipeline

Simulation: Configure the Stages of a CI/CD Pipeline

For more information on *Configure the Stages of a CI/CD Pipeline*, please view the simulation in the lesson *Configure Stages of the CI/CD Pipeline* in your online course.

1.

Unit 7
Solution 14

Configure the Stages of a CI/CD Pipeline

Simulation: Configure the Stages of a CI/CD Pipeline

For more information on *Configure the Stages of a CI/CD Pipeline*, please view the simulation in the lesson *Configure Stages of the CI/CD Pipeline* in your online course.

1.

Reference Links

For your convenience, this section contains the external references in this lesson.

If links are used multiple times within the text, only the first location is mentioned in the reference table.

Table 16: Reference Links: CI/CD Pipeline

Ref#	Section	Context text fragment	Brief description	Link
1	Configure the Stages of a CI/CD Pipeline	is based on SAP's Open-Source Project Piper.	Project Piper	https://www.project-piper.io/

LESSON SUMMARY

You should now be able to:

- Configure stages of the CI/CD pipeline

Unit 7
Lesson 6

Verify the Build Success

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Start monitoring a job in SAP Continuous Integration and Delivery

Verify Build Success: Exercise Overview

Scenario

After configuring a job and a pipeline for Continuous Integration and Delivery, you want to ensure, that the automated build and deployment of your project succeeds.

Task Flow

In this exercise, you will perform the following task:

- Start monitoring a job in SAP Continuous Integration and Delivery.

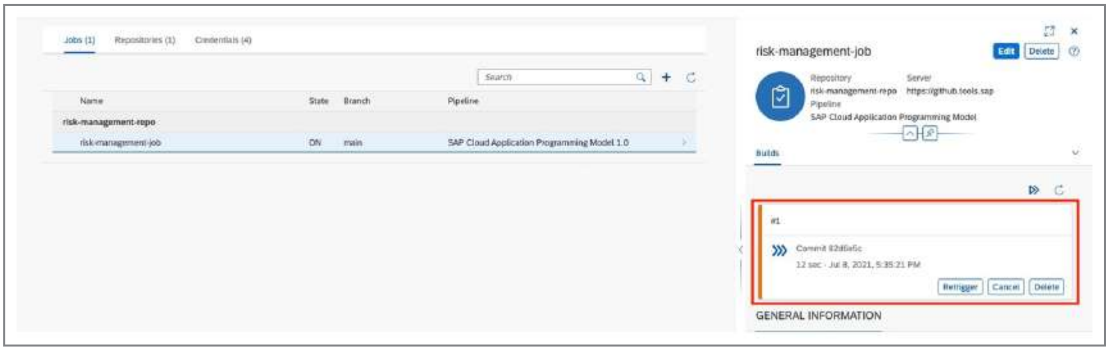
Prerequisite

You have set up an SAP Continuous Integration and Delivery pipeline.

Monitor the Outcome

After completing these steps, you will have monitored the outcome of your job in SAP Continuous Integration and Delivery.

1. In the *Jobs* tab in SAP Continuous Integration and Delivery, select your job and verify that a new tile appears in the *Builds* view. This tile should be marked as running.



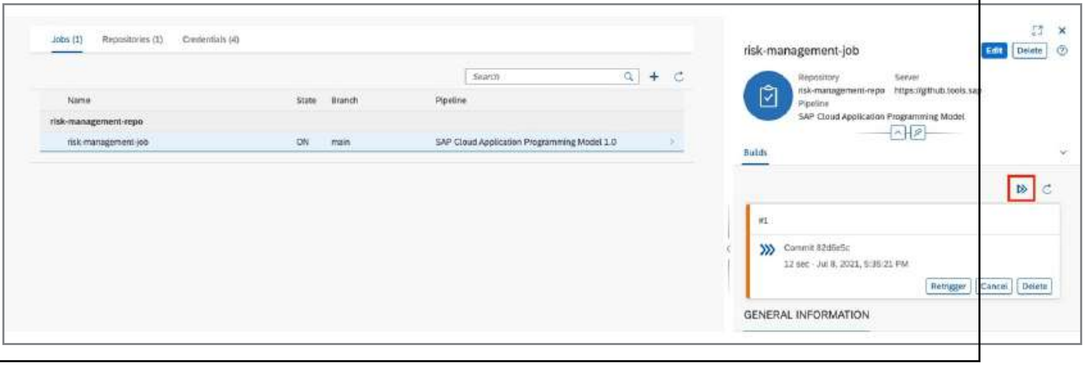
Name	State	Branch	Pipeline
risk-management-repo			
risk-management-job	ON	main	SAP Cloud Application Programming Model 1.0

risk-management-job

Repository: risk-management-repo
Server: https://github.tools.sap
Pipeline: SAP Cloud Application Programming Model

#1
Commit 82d6a5c
12 sec · Jul 8, 2021, 5:35:21 PM

Note:
If no new tile appears, trigger the job manually by choosing the *Trigger Build* button.



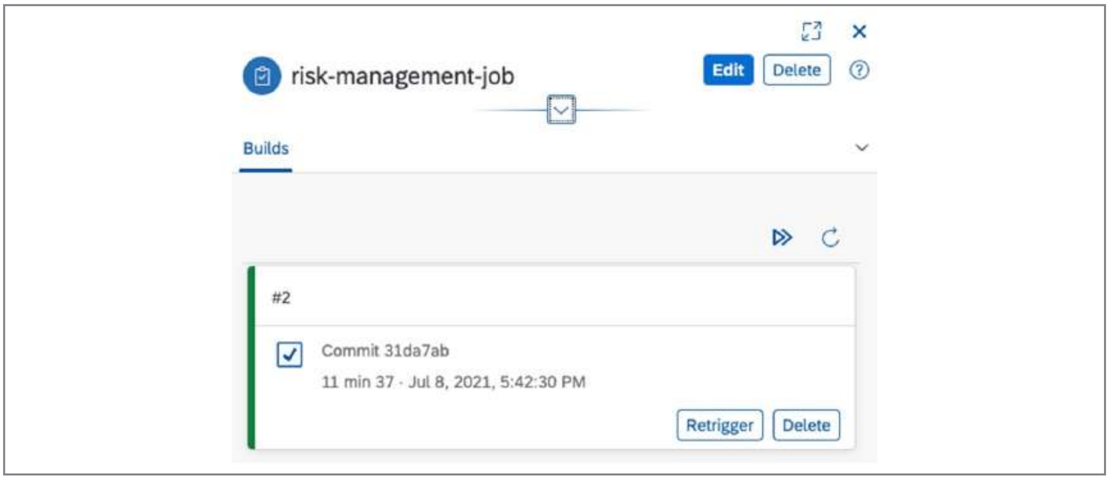
Name	State	Branch	Pipeline
risk-management-repo			
risk-management-job	ON	main	SAP Cloud Application Programming Model 1.0

risk-management-job

Repository: risk-management-repo
Server: https://github.tools.sap
Pipeline: SAP Cloud Application Programming Model

Commit 82d6e5c
12 sec · Jul 8, 2021, 5:36:21 PM

2. Wait until the job has finished and verify that the build tile is marked as successful.



Commit 31da7ab
11 min 37 · Jul 8, 2021, 5:42:30 PM

Summary

You did it!

Unit 7
Exercise 15

Verify Build Success

Simulation: Verify Build Success

For more information on *Verify Build Success*, please view the simulation in the lesson *Verify the Build Success* in your online course.

1.

Unit 7
Solution 15

Verify Build Success

Simulation: Verify Build Success

For more information on *Verify Build Success*, please view the simulation in the lesson *Verify the Build Success* in your online course.

1.

LESSON SUMMARY

You should now be able to:

- Start monitoring a job in SAP Continuous Integration and Delivery

Unit 7

Learning Assessment

1. Which of the following statements about a GitHub Repository are correct?

   Choose the correct answers.

   □ A Anyone on the internet can see a public repository.

   □ B Anyone on the internet can commit into a public repository.

   □ C You choose who can see your private repository.

   □ D You choose who can commit into your private repository.

2. After what period of time does GitHub remove unused personal access tokens?

   Choose the correct answer.

   □ A 28 days

   □ B 100 days

   □ C 1 month

   □ D 1 year

3. What is the next step after you initialize a new local git repository (git init) and add files to it (git add)?

   Choose the correct answer.

   □ A Commit

   □ B Fetch

   □ C Pull

4. What does the source code management system use to trigger the CI server?

   Choose the correct answer.

   □ A Webhooks

   □ B Web services

   □ C HTTP PUT requests

5. Which of the following statements apply to Continuous Delivery?

   Choose the correct answers.

   □ A Software is ready for deployment to a productive system all the time.

   □ B The trigger for deployment to a productive system is a human decision.

   □ C Deployment to a productive system is triggered automatically.

   □ D Code changes are pushed to a remote source code management system.

   □ E Feedback from a productive system gets quickly integrated into teams' backlog.

6. Which of the following statements applies to Continuous Deployment?

   Choose the correct answer.

   □ A Software is ready for deployment to a productive system all the time.

   □ B The trigger for deployment to a productive system is a human decision.

   □ C Feedback from a productive system gets quickly integrated into teams' backlog.

   □ D Deployment to a productive system is triggered automatically.

   □ E Code changes are pushed to a remote source code management system.

7. Which of the following statements about SAP Continuous Integration and Delivery (CI/CD) is correct?

   Choose the correct answer.

   □ A If you want to use SAP CI/CD in SAP BTP, you need to enable it with a service instance.

   □ B If you want to use SAP CI/CD in SAP BTP, you need to subscribe to a service plan of SAP CI/CD.

8. You can use the credentials created in the SAP CI/CD service to:

   Choose the correct answers.

   □ A Connect to a private GitHub repository.

   □ B Access the service itself.

   □ C Deploy applications to the SAP BTP, Cloud Foundry environment.

9. What are some of the Continuous Integration principles?

   Choose the correct answers.

   □ A Use version control.

   □ B Fix errors only when users complain.

   □ C Run tests in the build.

   □ D Run tests only in production.

   □ E Fix errors immediately.

10. What is a "main line" in a source control management system used for?

    Choose the correct answer.

    □ A To automate deployment.

    □ B To enable a reporting line for the project manager.

    □ C To make developers' contribution transparent and avoid clashes.

11. A main line in a source control management system can contain feature branches.

    Determine whether this statement is true or false.

    □ True

    □ False

12. What are the differences between continuous integration (CI) and continuous delivery (CD)?

    Choose the correct answers.

    □ A CI allows team members to add their changes to a main line.

    □ B CD adds an aspect that changes have been tested successfully.

    □ C CI allows developers to integrate their contributions any time.

    □ D CD adds an aspect that changes are immediately ready for deployment.

13. What do you use to retrieve the information about a change on the repository?

    Choose the correct answer.

    □ A A change document

    □ B A webhook

    □ C A PUT request to GitHub

14. What is the actual automation part of SAP CI/CD?

Choose the correct answer.

□ A Configure a job.

□ B Configure a branch in the GitHub repository.

15. What kind of request does the webhook send?

Choose the correct answer.

□ A GET

□ B PUT

□ C POST

16. How can you get the information about org name and API endpoint of your Cloud Foundry environment?

Choose the correct answers.

□ A Run command cf target in the terminal of SAP Business Application Studio.

□ B Run command cf help -a in the terminal of SAP Business Application Studio.

□ C Access the values from the subaccount overview in the SAP BTP cockpit.

17. What do you use to tell a job which steps it needs to perform?

Choose the correct answer.

□ A Create a webhook.

□ B Set up a pipeline.

□ C Set up an OData service.

18. Which command adds a file called Jenkinsfile to your CAP project?

Choose the correct answer.

□ A cds add pipeline

□ B git add Jenkins

□ C cd .pipeline

19. What are some of the artifacts project "Piper" offers?

    Choose the correct answers.

    □ A Feedback loop

    □ B Abaplint

    □ C Docker images

    □ D ABAP Environment pipeline

20. Within SAP CI/CD jobs you can:

    Choose the correct answer.

    □ A Delete deployed applications directly from your Cloud Foundry environment space.

    □ B Monitor the successful creation of your builds.

21. In the Builds view of SAP Continuous Integration and Delivery there is no new tile for your job. Which command do you run to trigger the job manually?

    Choose the correct answer.

    □ A Trigger run

    □ B Trigger title

    □ C Trigger build

    □ D Trigger refresh

UNIT 8

Connection of an SAP S/4HANA Cloud System as an External Service for CAP

Lesson 1

UNIT OBJECTIVES

- Connect your app to the productive Business Partner API of your SAP S/4HANA Cloud system

Unit 8
Lesson 1

Connect an SAP S/4HANA Cloud System as an External Service for CAP

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Connect your app to the productive Business Partner API of your SAP S/4HANA Cloud system

Business Partner API of an SAP S/4HANA Cloud system

In the exercise Add an External Service you added an external Business Partner service to your application, and then connected it to the Sandbox environment of the API Business Hub. As a result, the Business Partner data in your application was provided by the Sandbox environment.

Now, you want to connect your app to the productive Business Partner API of your SAP S/4HANA Cloud system. You can do this using the steps shown in the next four demos.

Task flow

In these demos, you will perform the following tasks:

- Connect SAP S/4HANA Cloud system with an SAP BTP account.
- Create an SAP S/4HANA Cloud Extensibility service instance for the Business Partner API.
- Use SAP S/4HANA Cloud destination in CAP project.
- Deploy CAP project using SAP Continuous Integration and Delivery service.

Prerequisites

The following prerequisites are needed to follow the steps shown in these demos:

- You have an SAP S/4HANA Cloud system with administrator access.
- You have an SAP BTP Global account with administrator access (that is, a Trial account).
- You have successfully finished all the exercises of this course.

LESSON SUMMARY

You should now be able to:

- Connect your app to the productive Business Partner API of your SAP S/4HANA Cloud system

Learning Assessment

1. What is required when registering an SAP S/4HANA Cloud system in your SAP BTP global account?

   Choose the correct answer.

   □ A A token

   □ B A coin

   □ C A voucher

   □ D A developer

2. In which of the following apps can you manage connected SAP BTP accounts in an SAP S/4HANA system?

   Choose the correct answer.

   □ A Maintain Integrations on SAP BTP

   □ B Maintain Extensions on SAP BTP

   □ C Manage Add-ons on SAP BTP

3. Which service plan of the SAP BTP S/4HANA Cloud Extensibility service do you choose when you want to use the Business Partner API of your SAP S/4HANA Cloud system?

   Choose the correct answer.

   □ A Cloud Foundry

   □ B SAP_COM_0008

   □ C api-access

4. Which of the following identifiers of the communication scenario is used for accessing the Business Partner API in an SAP S/4HANA Cloud system?

   Choose the correct answer.

   □ A SAP_COM_0009

   □ B SAP_COM_0213

   □ C SAP_COM_0008

   □ D SAP_COM_0109