Fiori App using RAP

1. Create Database Table and Generate UI Service
2. Create Trail BTP account, run booster to create ABAP environment In your sub-account, use ABAP env service key🡪 go to Eclipse 🡪 create new ABAP cloud project 🡪 use service key to complete setup.
3. Create new package ZRAP100\_KS with proper description 🡪 add it as your favourite package
4. Now create database table with below code🡪 ZRAP100\_ATRAV\_KS with description as “Travel Data” 🡪 activate it

@EndUserText.label : 'Travel data'

@AbapCatalog.enhancement.category : #NOT\_EXTENSIBLE

@AbapCatalog.tableCategory : #TRANSPARENT

@AbapCatalog.deliveryClass : #A

@AbapCatalog.dataMaintenance : #RESTRICTED

**define** **table** zrap100\_atrav\_ks **{**

**key** client **:** abap**.**clnt **not** **null;**

**key** travel\_id **:** /dmo/travel\_id **not** **null;**

agency\_id **:** /dmo/agency\_id**;**

customer\_id **:** /dmo/customer\_id**;**

begin\_date **:** /dmo/begin\_date**;**

end\_date **:** /dmo/end\_date**;**

@Semantics.amount.currencyCode : 'zrap100\_atrav\_ks.currency\_code'

booking\_fee **:** /dmo/booking\_fee**;**

@Semantics.amount.currencyCode : 'zrap100\_atrav\_ks.currency\_code'

total\_price **:** /dmo/total\_price**;**

currency\_code **:** /dmo/currency\_code**;**

description **:** /dmo/description**;**

overall\_status **:** /dmo/overall\_status**;**

attachment **:** /dmo/attachment**;**

mime\_type **:** /dmo/mime\_type**;**

file\_name **:** /dmo/filename**;**

created\_by **:** abp\_creation\_user**;**

created\_at **:** abp\_creation\_tstmpl**;**

local\_last\_changed\_by **:** abp\_locinst\_lastchange\_user**;**

local\_last\_changed\_at **:** abp\_locinst\_lastchange\_tstmpl**;**

last\_changed\_at **:** abp\_lastchange\_tstmpl**;**

**}**

1. Create Data generator class **ZCL\_RAP100\_GEN\_DATA\_KS** with description as “**Generate demo data”** 🡪 Copy below code to fetch data and insert in our table 🡪 Execute this class

CLASS zcl\_rap100\_gen\_data\_KS DEFINITION

PUBLIC

FINAL

CREATE PUBLIC .

PUBLIC SECTION.

INTERFACES if\_oo\_adt\_classrun.

PROTECTED SECTION.

PRIVATE SECTION.

ENDCLASS.

CLASS zcl\_rap100\_gen\_data\_KS IMPLEMENTATION.

METHOD if\_oo\_adt\_classrun~main.

DATA:

group\_id TYPE string VALUE 'KS',

attachment TYPE /dmo/attachment,

file\_name TYPE /dmo/filename,

mime\_type TYPE /dmo/mime\_type.

\* clear data

DELETE FROM zrap100\_atrav\_KS.

"insert travel demo data

INSERT zrap100\_atrav\_KS FROM (

SELECT

FROM /dmo/travel AS travel

FIELDS

travel~travel\_id AS travel\_id,

travel~agency\_id AS agency\_id,

travel~customer\_id AS customer\_id,

travel~begin\_date AS begin\_date,

travel~end\_date AS end\_date,

travel~booking\_fee AS booking\_fee,

travel~total\_price AS total\_price,

travel~currency\_code AS currency\_code,

travel~description AS description,

CASE travel~status "[N(New) | P(Planned) | B(Booked) | X(Cancelled)]

WHEN 'N' THEN 'O'

WHEN 'P' THEN 'O'

WHEN 'B' THEN 'A'

ELSE 'X'

END AS overall\_status,

@attachment AS attachment,

@mime\_type AS mime\_type,

@file\_name AS file\_name,

travel~createdby AS created\_by,

travel~createdat AS created\_at,

travel~lastchangedby AS last\_changed\_by,

travel~lastchangedat AS last\_changed\_at,

travel~lastchangedat AS local\_last\_changed\_at

ORDER BY travel\_id UP TO 10 ROWS

).

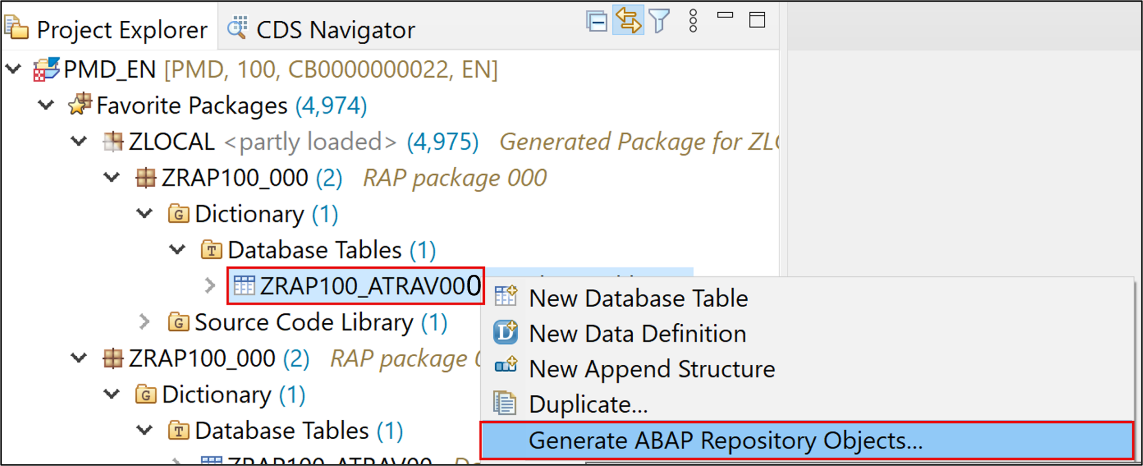
COMMIT WORK.

out->write( |[RAP100] Demo data generated for table ZRAP100\_ATRAV{ group\_id }. | ).

ENDMETHOD.

ENDCLASS.

1. Generate transaction UI services 🡪 Right click on Database table🡪Generate ABAP Repository Objects🡪Select ODATA UI Service🡪Click Next🡪 Select your package🡪Click Next

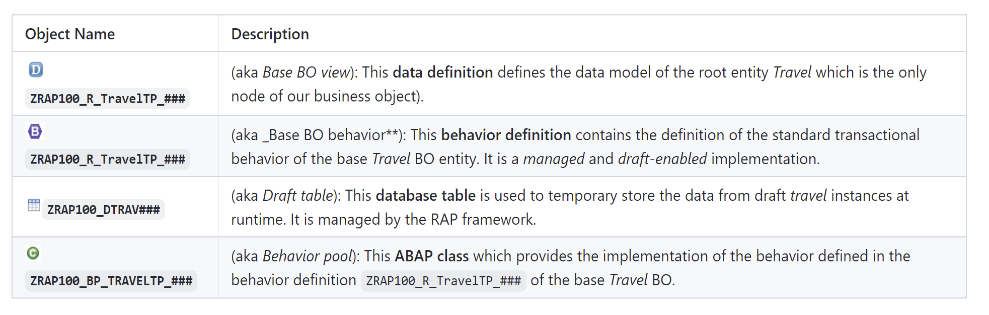


Put below details in each section🡪 check all artifacts before click next🡪 click next🡪all artifact will be generated🡪 Click on package and refresh it:-

| **RAP Layer** | **Artefacts** | **Artefact Names** |
| --- | --- | --- |
| **Business Object** |  |  |
|  | **Data Model** | CDS Entity Name: **ZRAP100\_R\_TRAVELTP\_KS** |
|  |  | CDS Entity Name Alias: **Travel** |
|  | **Behavior** | Implementation Behavior Class: **ZBP\_R\_RAP100\_ATRAV\_KS** |
|  |  | Draft Table Name: **ZRAP100\_DTRAV\_KS** |
| **Service Projection** | **Service Projection Entity** | CDS Entity Name: **ZRAP100\_C\_TRAVELTP\_KS** |
|  | **Service Projection Behavior** | Behavior Implementation Class: **ZRAP100\_BP\_C\_TRAVELTP\_KS** |
| **Business Service** |  |  |
|  | **Service Definition** | Service Definition Name: **ZRAP100\_UI\_TRAVEL\_###** |
|  | **Service Binding** | Service Binding Name: **ZRAP100\_UI\_TRAVEL\_O4\_###** |
|  |  | Binding Type: **OData V4 - UI** |

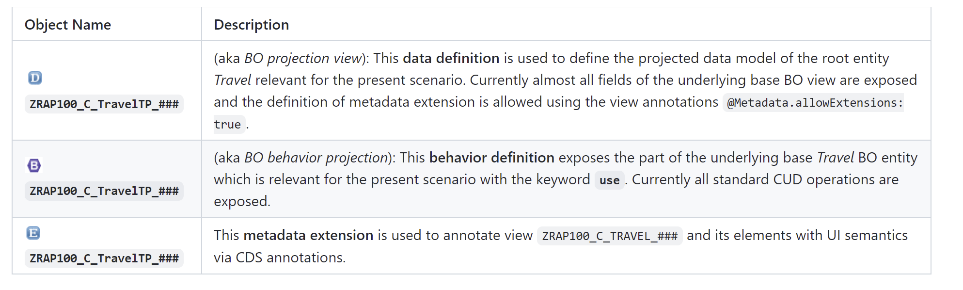
Below is short description of all object created:-

* **Base Business Object (BO)**

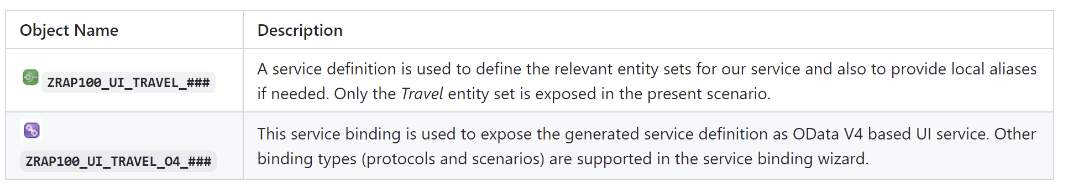


* **BO Projection ZRAP100\_C\_TRAVEL\_KS**

The BO projection represents the consumption specific view on the BO data model and behavior.



* **Business Service**



1. Now select service Binding ZRAP100\_UI\_TRAVEL\_O4\_KS 🡪 click on publish 🡪 Then select Travel entity from right side list and click on preview🡪 it will open preview in browser 🡪 Click on GO to preview data 🡪 Check result 🡪 this completes 1st part
2. Enhance the Business Object Data Model and Enable OData Streams
3. Open your data definition ZR\_RAP100\_ATRAV\_KSand Define the newassociations **\_Agency,  \_Customer,  \_OverallStatus, and  \_Currency 🡪 Replace entire code with below code**

@AccessControl.authorizationCheck: #CHECK

@Metadata.allowExtensions: true

@EndUserText.label: '###GENERATED Core Data Service Entity'

@ObjectModel.sapObjectNodeType.name: 'ZRAP100\_ATRAV\_KS'

**define** **root** **view** **entity** ZR\_RAP100\_ATRAV\_KS

**as** **select** **from** zrap100\_atrav\_ks **as** Travel

**association** **[**0**..**1**]** **to** /DMO/I\_Agency **as** \_Agency **on** **$projection.**AgencyId **=** \_Agency**.**AgencyID

**association** **[**0**..**1**]** **to** /DMO/I\_Customer **as** \_Customer **on** **$projection.**CustomerId **=** \_Customer**.**CustomerID

**association** **[**1**..**1**]** **to** /DMO/I\_Overall\_Status\_VH **as** \_OverallStatus **on** **$projection.**OverallStatus **=** \_OverallStatus**.**OverallStatus

**association** **[**0**..**1**]** **to** I\_Currency **as** \_Currency **on** **$projection.**CurrencyCode **=** \_Currency**.**Currency

**{**

**key** travel\_id **as** TravelId**,**

agency\_id **as** AgencyId**,**

customer\_id **as** CustomerId**,**

begin\_date **as** BeginDate**,**

end\_date **as** EndDate**,**

@Semantics.amount.currencyCode: 'CurrencyCode'

booking\_fee **as** BookingFee**,**

@Semantics.amount.currencyCode: 'CurrencyCode'

total\_price **as** TotalPrice**,**

@Consumption.valueHelpDefinition: [ {

entity.name: 'I\_CurrencyStdVH',

entity.element: 'Currency',

useForValidation: true

} ]

currency\_code **as** CurrencyCode**,**

description **as** Description**,**

overall\_status **as** OverallStatus**,**

attachment **as** Attachment**,**

mime\_type **as** MimeType**,**

file\_name **as** FileName**,**

@Semantics.user.createdBy: true

created\_by **as** CreatedBy**,**

@Semantics.systemDateTime.createdAt: true

created\_at **as** CreatedAt**,**

@Semantics.user.localInstanceLastChangedBy: true

local\_last\_changed\_by **as** LocalLastChangedBy**,**

@Semantics.systemDateTime.localInstanceLastChangedAt: true

local\_last\_changed\_at **as** LocalLastChangedAt**,**

@Semantics.systemDateTime.lastChangedAt: true

last\_changed\_at **as** LastChangedAt**,**

//public associations

\_Agency**,**

\_Customer**,**

\_OverallStatus**,**

\_Currency

**}**

1. In same interface view ZR\_RAP100\_ATRAV\_KS 🡪 We need to define semantics for attachment and mimetype so that images can stream through odata. Add below semantic for attachment and mimetype. 🡪 then save and activate

* For element MimeType:

@Semantics.mimeType: true

* For element Attachment:

@Semantics.largeObject: { mimeType: 'MimeType', //case-sensitive

fileName: 'FileName', //case-sensitive

acceptableMimeTypes: ['image/png', 'image/jpeg'],

contentDispositionPreference: #ATTACHMENT }

Short explanation: The attributes of the annotation @Semantics.largeObject

* mimeType: It indicates the name of the field containing the type of a MIME object. ⚠ The value is case sensitive.
* fileName: It indicates the name of the field containing the file name of a MIME object. ⚠ The value is case sensitive.
* acceptableMimeTypes: It provides the list of acceptable MIME types for the related stream property to restrict or verify the user entry accordingly. If any subtype is accepted, this can be indicated by \*.
* contentDispositionPreference: It indicates whether the content is expected to be displayed inline in the browser, i.e., as a Web page or as part of a Web page, or as an attachment, i.e., downloaded and saved locally.

1. Replace code in consumption view with below code ZRAP100\_C\_TRAVELTP\_KS to provide searchable functionality to some fields along with F4 help. Few fields are also added from associations.

@AccessControl.authorizationCheck: #CHECK

@Metadata.allowExtensions: true

@EndUserText.label: '##GENERATED Travel App (###)'

@Search.searchable: true

@ObjectModel.semanticKey: ['TravelID'] //case-sensitive

**define** **root** **view** **entity** ZRAP100\_C\_TRAVELTP\_KS

**provider** **contract** **transactional\_query**

**as** **projection** **on** ZR\_RAP100\_ATRAV\_KS

**{**

@Search.defaultSearchElement: true

@Search.fuzzinessThreshold: 0.90

**key** TravelId**,**

@Search.defaultSearchElement: true

@ObjectModel.text.element: ['AgencyName'] //case-sensitive

@Consumption.valueHelpDefinition: [{ entity : {name: '/DMO/I\_Agency\_StdVH', element: 'AgencyID' }, useForValidation: true }]

AgencyId**,**

\_Agency**.**Name **as** AgencyName**,**

@Search.defaultSearchElement: true

@ObjectModel.text.element: ['CustomerName'] //case-sensitive

@Consumption.valueHelpDefinition: [{ entity : {name: '/DMO/I\_Customer\_StdVH', element: 'CustomerID' }, useForValidation: true }]

CustomerId**,**

\_Customer**.**LastName **as** CustomerName**,**

BeginDate**,**

EndDate**,**

BookingFee**,**

TotalPrice**,**

@Consumption.valueHelpDefinition: [{ entity: {name: 'I\_CurrencyStdVH', element: 'Currency' }, useForValidation: true }]

CurrencyCode**,**

Description**,**

@ObjectModel.text.element: ['OverallStatusText'] //case-sensitive

@Consumption.valueHelpDefinition: [{ entity: {name: '/DMO/I\_Overall\_Status\_VH', element: 'OverallStatus' }, useForValidation: true }]

OverallStatus**,**

\_OverallStatus**.**\_Text**.**Text **as** OverallStatusText **:** **localized,**

Attachment**,**

MimeType**,**

FileName**,**

LocalLastChangedAt

**}**

1. Open your metadata extension **ZRAP100\_C\_TRAVELTP\_KS**and adjust the UI annotations to achieve the following changes on the Fiori elements based UI of the **Travel App**.

* Element **TravelID** - should also be a selection criteria in the filter bar and have high display importance on small windows.
* Element **AgencyID** - should also be a selection criteria in the filter bar and have high display importance on small windows.
* Element **CustomerID** - should also be a selection criteria in the filter bar and have high display importance on small windows.
* Element **BeginDate** - (no changes)
* Element **EndDate** - (no changes)
* Element **BookingFee** - should not be displayed in the list table.
* Element **TotalPrice** - should not be displayed in the list table.
* Element **CurrencyCode** - should not be explicitly displayed, neither in the list table nor on the object page. **Hint:** The currency code will be automatically displayed on the UI thanks to @consumption annotations specified for the element CurrencyCode in the BO projection view.
* Element **Description** - should not be displayed in the list table.
* Element **OverallStatus** - should have a high display importance on small windows and only its associated descriptive text should be displayed on the UI.
* Element Attachment - should only be displayed on the object page - not in the list table.
* Element MimeType - should be hidden.
* Element FileName - should be hidden.

Copy below code in metadata extension :-

@Metadata.layer: #CUSTOMER

@UI: {

headerInfo: {

typeName: 'Travel',

typeNamePlural: 'Travels',

imageUrl: 'Attachment', //case-sensitive

description: { type: #STANDARD, value: 'TravelID' } //case-sensitive

}

}

**annotate** **view** ZRAP100\_C\_TRAVELTP\_KS **with**

**{**

@UI.facet: [ {

id: 'idIdentification',

type: #IDENTIFICATION\_REFERENCE,

label: 'Travel',

position: 10

} ]

@UI: {

lineItem: [ { position: 10, importance: #HIGH } ],

identification: [ { position: 10 } ],

selectionField: [ { position: 10 } ]

}

TravelId**;**

@UI: {

lineItem: [ { position: 20, importance: #HIGH } ],

identification: [ { position: 20 } ],

selectionField: [ { position: 20 } ]

}

AgencyId**;**

@UI: {

lineItem: [ { position: 30, importance: #HIGH } ],

identification: [ { position: 30 } ],

selectionField: [ { position: 30 } ]

}

CustomerId**;**

@UI: {

lineItem: [ { position: 40, importance: #MEDIUM } ],

identification: [ { position: 40 } ]

}

BeginDate**;**

@UI: {

lineItem: [ { position: 50, importance: #MEDIUM } ],

identification: [ { position: 50 } ]

}

EndDate**;**

@UI.identification: [ { position: 60 } ]

BookingFee**;**

@UI.identification: [ { position: 70 } ]

TotalPrice**;**

@UI.identification: [ { position: 90 } ]

Description**;**

@UI: {

lineItem: [ { position: 100, importance: #HIGH } ],

identification: [ { position: 100 } ],

textArrangement: #TEXT\_ONLY

}

OverallStatus**;**

@UI.identification: [ { position: 110 } ]

Attachment**;**

@UI.hidden: true

MimeType**;**

@UI.hidden: true

FileName**;**

@UI.hidden: true

LocalLastChangedAt**;**

**}**

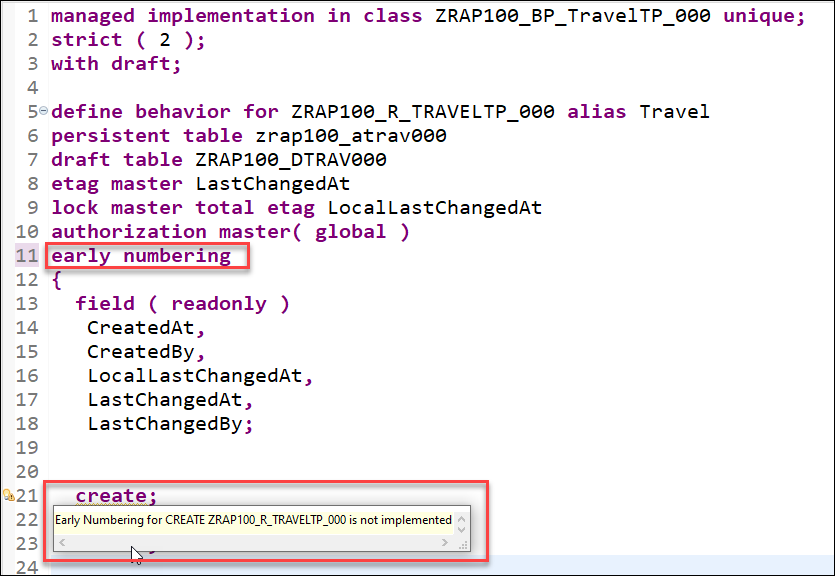
1. Enhance the Business Object Behavior With Unmanaged Internal Numbering

**Numbering:** Numbering is about setting values for primary key fields of entity instances during runtime. Different types of numbering are supported in RAP which can be divided into two main categories:

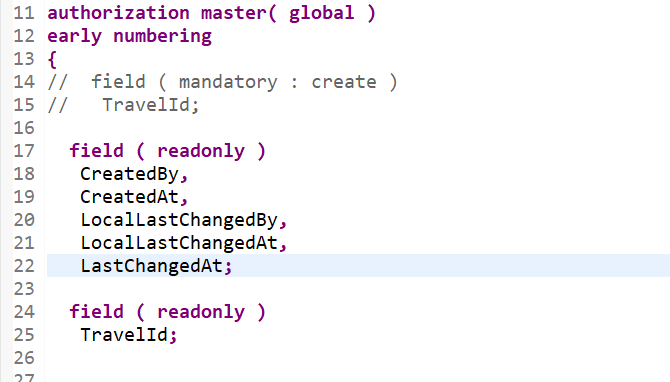
**Early numbering:** In an early numbering scenario, the primary key value is set instantly after the modify request for the CREATE is executed. The key values can be passed externally by the consumer or can be set internally by the framework or an implementation of the FOR NUMBERING method. The latter will be implemented in the present exercise.

**Late numbering:** In a late numbering scenario, the key values are always assigned internally without consumer interaction after the point of no return in the interaction phase has passed, and the SAVE sequence is triggered.

* 1. Goto Behavior implementation ZR\_RAP100\_ATRAV\_KS 🡪 add below line in code. The warning message **Early Numbering for CREATE ZR\_RAP100\_ATRAV\_KS** is not implemented is now displayed for the statement **create;**. You can hover the yellow underlined statement to display the message or have a look at the **Problems** view. CTRL + 1 and implement the suggestion.



* 1. Make Travel ID field as Readonly as it will be populated using early number.



* 1. Copy below code in earlynumbering\_create method

DATA:

entity TYPE STRUCTURE FOR CREATE zr\_rap100\_atrav\_ks,

travel\_id\_max TYPE /dmo/travel\_id,

" change to abap\_false if you get the ABAP Runtime error 'BEHAVIOR\_ILLEGAL\_STATEMENT'

use\_number\_range TYPE abap\_bool VALUE abap\_false.

"Ensure Travel ID is not set yet (idempotent)- must be checked when BO is draft-enabled

LOOP AT entities INTO entity WHERE TravelID IS NOT INITIAL.

APPEND CORRESPONDING #( entity ) TO mapped-travel.

ENDLOOP.

DATA(entities\_wo\_travelid) = entities.

"Remove the entries with an existing Travel ID

DELETE entities\_wo\_travelid WHERE TravelID IS NOT INITIAL.

IF use\_number\_range = abap\_true.

"Get numbers

TRY.

cl\_numberrange\_runtime=>number\_get(

EXPORTING

nr\_range\_nr = '01'

object = '/DMO/TRV\_M'

quantity = CONV #( lines( entities\_wo\_travelid ) )

IMPORTING

number = DATA(number\_range\_key)

returncode = DATA(number\_range\_return\_code)

returned\_quantity = DATA(number\_range\_returned\_quantity)

).

CATCH cx\_number\_ranges INTO DATA(lx\_number\_ranges).

LOOP AT entities\_wo\_travelid INTO entity.

APPEND VALUE #( %cid = entity-%cid

%key = entity-%key

%is\_draft = entity-%is\_draft

%msg = lx\_number\_ranges

) TO reported-travel.

APPEND VALUE #( %cid = entity-%cid

%key = entity-%key

%is\_draft = entity-%is\_draft

) TO failed-travel.

ENDLOOP.

EXIT.

ENDTRY.

"determine the first free travel ID from the number range

travel\_id\_max = number\_range\_key - number\_range\_returned\_quantity.

ELSE.

"determine the first free travel ID without number range

"Get max travel ID from active table

SELECT SINGLE FROM zrap100\_atrav\_ks FIELDS MAX( travel\_id ) AS travelID INTO @travel\_id\_max.

"Get max travel ID from draft table

SELECT SINGLE FROM zrap100\_dtrav\_ks FIELDS MAX( travelid ) INTO @DATA(max\_travelid\_draft).

IF max\_travelid\_draft > travel\_id\_max.

travel\_id\_max = max\_travelid\_draft.

ENDIF.

ENDIF.

"Set Travel ID for new instances w/o ID

LOOP AT entities\_wo\_travelid INTO entity.

travel\_id\_max += 1.

entity-TravelID = travel\_id\_max.

APPEND VALUE #( %cid = entity-%cid

%key = entity-%key

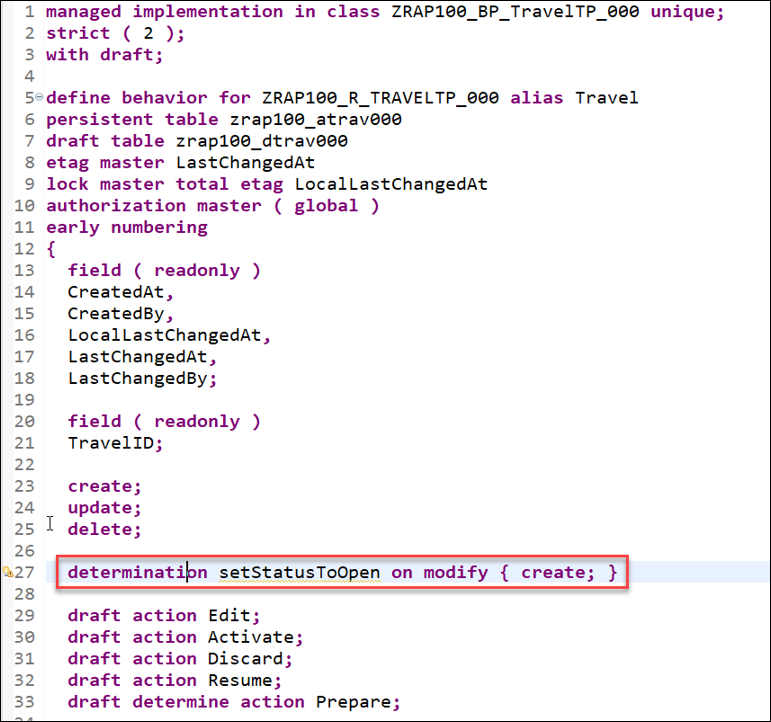
%is\_draft = entity-%is\_draft

) TO mapped-travel.

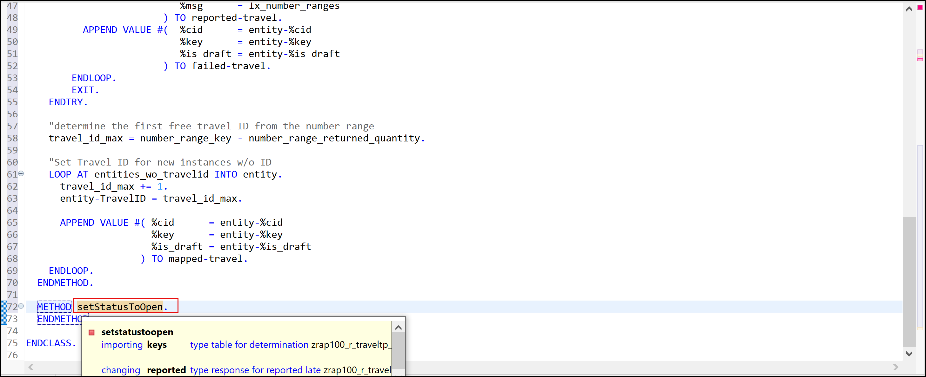
ENDLOOP.

* 1. Test enhanced travel app .

1. Enhance the Business Object Behavior with Determinations
2. Add below code in BO ZR\_RAP100\_ATRAV\_KS 🡪 CTRL+1 to add implementation in respective class. Activate both BO and class



1. Press F2 on method setStatusToOpen to check all importing/exporting parameter.



1. Copy Below code in method setStatusToOpen

The logic consists of the following steps:

* Read the travel instance(s) of the transferred keys **(keys)** using the EML statement **READ ENTITIES**
* The addition **IN LOCAL MODE** is used to exclude feature controls and authorization checks
* Removed all **Travel** instances where the overall status is already set
* Set the overall status to **open** **(O)** for the remaining entries using the EML statement **MODIFY ENTITIES**
* Set the changing parameter **reported**

CONSTANTS:

BEGIN OF travel\_status,

open TYPE c LENGTH 1 VALUE 'O', "Open

accepted TYPE c LENGTH 1 VALUE 'A', "Accepted

rejected TYPE c LENGTH 1 VALUE 'X', "Rejected

END OF travel\_status.

"Read travel instances of the transferred keys

READ ENTITIES OF ZR\_RAP100\_ATRAV\_KS IN LOCAL MODE

ENTITY Travel

FIELDS ( OverallStatus )

WITH CORRESPONDING #( keys )

RESULT DATA(travels)

FAILED DATA(read\_failed).

"If overall travel status is already set, do nothing, i.e. remove such instances

DELETE travels WHERE OverallStatus IS NOT INITIAL.

CHECK travels IS NOT INITIAL.

"else set overall travel status to open ('O')

MODIFY ENTITIES OF ZR\_RAP100\_ATRAV\_KS IN LOCAL MODE

ENTITY Travel

UPDATE SET FIELDS

WITH VALUE #( FOR travel IN travels ( %tky = travel-%tky

OverallStatus = travel\_status-open ) )

REPORTED DATA(update\_reported).

"Set the changing parameter

reported = CORRESPONDING #( DEEP update\_reported ).

1. Test the application🡪Create new travel id🡪you will see status will be set by default to Open.

A screenshot of a computer

AI-generated content may be incorrect.

1. Enhance the Business Object Behavior With Validations

In the present exercise, you’re going to define and implement two back-end validations, validateCustomer and validateDates, to respectively check if the customer ID that is entered by the consumer is valid and if the begin date is in the future and if the value of the end date is after the begin date. These validations are only performed in the back-end (not on the UI) and are triggered independently of the caller, i.e. Fiori UIs or EML APIs.

**Hint: Frontend validation & Backend validations** Validations are used to ensure the data consistency. As the name suggests, frontend validations are performed on the UI. They are used to improve the user experience by providing faster feedback and avoiding unnecessary roundtrip. In the RAP context, front-end validations are defined using CDS annotation or UI logic. On the other hand, backend validations are performed on the back-end. They are defined in the business object behavior definition and implemented in the respective behavior pools. Frontend validations can be easily bypassed - e.g. by using EML APIs in the RAP context. Therefore, backend validations are a MUST to ensure the data consistency.

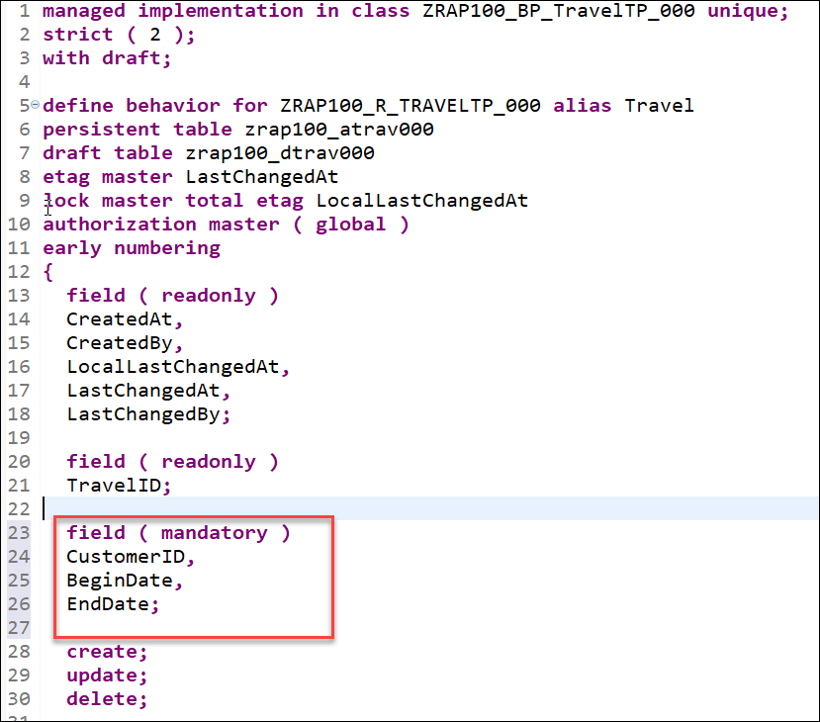
**About: Validations**

A validation is an optional part of the business object behavior that checks the consistency of business object instances based on trigger conditions.

A validation is implicitly invoked by the business object’s framework if the trigger condition of the validation is fulfilled. Trigger conditions can be MODIFY operations and modified fields. The trigger condition is evaluated at the trigger time, a predefined point during the BO runtime. An invoked validation can reject inconsistent instance data from being saved by passing the keys of failed instances to the corresponding table in the FAILED structure. Additionally, a validation can return messages to the consumer by passing them to the corresponding table in the REPORTED structure.

**Further reading**: [**Validations**](https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/171e26c36cca42699976887b4c8a83bf.html)

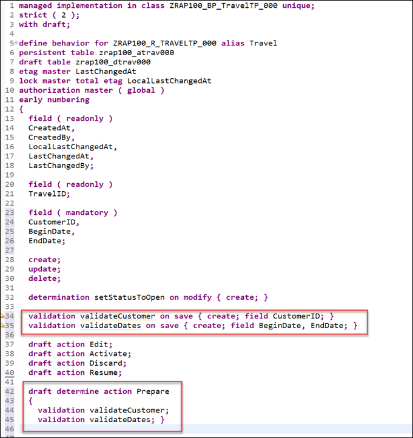
1. Define Validation validateCustomer and validateDates:- Open BO ZR\_RAP100\_ATRAV\_KS🡪Specify below field as Mandatory field as blank values will not be allowed.



1. After determination code, enter validation for customer and date 🡪 Validation always happens on Save for field and Action needs to be mentioned whether create; or create; update; 🡪 then field name.

To have draft instances being checked and determinations being executed before they become active, they must be specified for the **draft determine action prepare** in the behavior definition.

Replace the code line **draft determine action Prepare;** with the following code snippet as shown on the screenshot below🡪 Click CTRL+1 to implement method🡪F2 to check importing/Changing parameter.



1. Now implement code in validateCustomer.

READ ENTITIES OF zr\_rap100\_atrav\_ks IN LOCAL MODE

ENTITY Travel

FIELDS ( CustomerID )

WITH CORRESPONDING #( keys )

RESULT DATA(travels).

DATA: customers TYPE SORTED TABLE OF /dmo/customer WITH UNIQUE KEY customer\_id.

"optimization of DB select: extract distinct non-initial customer IDs

customers = CORRESPONDING #( travels DISCARDING DUPLICATES MAPPING customer\_id = CustomerId EXCEPT \* ).

DELETE customers WHERE customer\_id IS INITIAL.

IF customers IS NOT INITIAL.

SELECT FROM /dmo/customer FIELDS customer\_id

FOR ALL ENTRIES IN @customers

WHERE customer\_id = @customers-customer\_id

INTO TABLE @DATA(valid\_customers).

ENDIF.

"raise msg for non existing and initial customer id

LOOP AT travels INTO DATA(travel).

APPEND VALUE #( %tky = travel-%tky

%state\_area = 'VALIDATE\_CUSTOMER'

) TO reported-travel.

IF travel-CustomerID IS INITIAL.

APPEND VALUE #( %tky = travel-%tky ) TO failed-travel.

APPEND VALUE #( %tky = travel-%tky

%state\_area = 'VALIDATE\_CUSTOMER'

%msg = NEW /dmo/cm\_flight\_messages(

textid = /dmo/cm\_flight\_messages=>enter\_customer\_id

severity = if\_abap\_behv\_message=>severity-error )

%element-CustomerID = if\_abap\_behv=>mk-on

) TO reported-travel.

ELSEIF travel-CustomerID IS NOT INITIAL AND NOT line\_exists( valid\_customers[ customer\_id = travel-CustomerID ] ).

APPEND VALUE #( %tky = travel-%tky ) TO failed-travel.

APPEND VALUE #( %tky = travel-%tky

%state\_area = 'VALIDATE\_CUSTOMER'

%msg = NEW /dmo/cm\_flight\_messages(

customer\_id = travel-customerid

textid = /dmo/cm\_flight\_messages=>customer\_unkown

severity = if\_abap\_behv\_message=>severity-error )

%element-CustomerID = if\_abap\_behv=>mk-on

) TO reported-travel.

ENDIF.

ENDLOOP.

1. Implement code for validateDates:-

READ ENTITIES OF zr\_rap100\_atrav\_ks IN LOCAL MODE

ENTITY Travel

FIELDS ( BeginDate EndDate TravelId )

WITH CORRESPONDING #( keys )

RESULT DATA(travels).

LOOP AT travels INTO DATA(travel).

APPEND VALUE #( %tky = travel-%tky

%state\_area = 'VALIDATE\_DATES'

) TO reported-travel.

if travel-BeginDate IS INITIAL.

APPEND VALUE #( %tky = travel-%tky ) to failed-travel.

APPEND VALUE #( %tky = travel-%tky

%state\_area = 'VALIDATE\_DATES'

%msg = new /dmo/cm\_flight\_messages(

textid = /dmo/cm\_flight\_messages=>enter\_begin\_date

severity = if\_abap\_behv\_message=>severity-error )

%element-BeginDate = if\_abap\_behv=>mk-on )

to reported-travel.

ENDIF.

if travel-EndDate IS INITIAL.

APPEND VALUE #( %tky = travel-%tky ) to failed-travel.

APPEND VALUE #( %tky = travel-%tky

%state\_area = 'VALIDATE\_DATES'

%msg = new /dmo/cm\_flight\_messages(

textid = /dmo/cm\_flight\_messages=>enter\_end\_date

severity = if\_abap\_behv\_message=>severity-error )

%element-EndDate = if\_abap\_behv=>mk-on )

to reported-travel.

ENDIF.

if travel-BeginDate < cl\_abap\_context\_info=>get\_system\_date( ) AND travel-BeginDate IS NOT INITIAL.

APPEND VALUE #( %tky = travel-%tky ) to failed-travel.

APPEND VALUE #( %tky = travel-%tky

%state\_area = 'VALIDATE\_DATES'

%msg = new /dmo/cm\_flight\_messages(

textid = /dmo/cm\_flight\_messages=>begin\_date\_on\_or\_bef\_sysdate

begin\_date = travel-BeginDate

severity = if\_abap\_behv\_message=>severity-error )

%element-BeginDate = if\_abap\_behv=>mk-on )

to reported-travel.

ENDIF.

if travel-EndDate < travel-BeginDate and travel-BeginDate iS NOT INITIAL

and travel-EndDate IS NOT INITIAL.

APPEND VALUE #( %tky = travel-%tky ) to failed-travel.

APPEND VALUE #( %tky = travel-%tky

%state\_area = 'VALIDATE\_DATES'

%msg = new /dmo/cm\_flight\_messages(

textid = /dmo/cm\_flight\_messages=>begin\_date\_bef\_end\_date

begin\_date = travel-BeginDate

end\_date = travel-EndDate

severity = if\_abap\_behv\_message=>severity-error )

%element-BeginDate = if\_abap\_behv=>mk-on

%element-EndDate = if\_abap\_behv=>mk-on )

to reported-travel.

ENDIF.

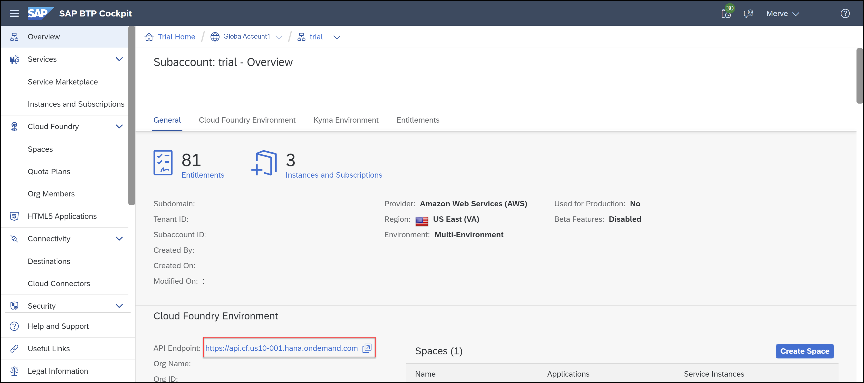
ENDLOOP.

1. Test the app for validation.
2. Create an SAP Fiori App and Deploy it to SAP BTP, ABAP Environment

You will learn

* How to assign role collections
* How to create dev spaces
* How to set up organization and space
* How to create list report object pages
* How to run SAP Fiori applications
* How to deploy applications
* How to check BSP library in Eclipse
* How to create IAM apps and business catalogs

1. Assign role collection to user 🡪 go to trail account🡪Sub-account🡪Users🡪Click on user🡪Select Role collections🡪Assign Role Collection🡪 Select **Business\_Application\_Studio\_Developer**.
2. Go to trial account🡪subaccount🡪Service and Marketplace🡪 SAP Application studio🡪Go to application🡪Create Dev Space🡪 When dev space starts running, select dev space.
3. Go to dev space🡪click on files & Folder🡪enter /home/user/projects/ 🡪Click Ok
4. Go back to subaccount🡪Copy Cloud Foundry Environment API Endpoint



1. Goto🡪 BAS🡪 **View > Command Palette🡪** **Search for CF: Login to Cloud Foundry and select it 🡪 give your universal email and password🡪you will see Cloud foundry endpoint and global account🡪Click Apply**

**Note:- Before you apply, just check if account is correct 🡪 if you don’t know username password, go with SSO login**

A screenshot of a computer

AI-generated content may be incorrect.

1. Click on view🡪Command palette🡪 **Fiori: Open Application Generator** and select it🡪Select List Report page🡪Click Next🡪Select “Connect to a system🡪Select Cloud Foundry Environment🡪Select Abap environment🡪 Select service ZRAP100\_UI\_TRAVEL\_O4\_KS 🡪Click Next🡪Select Main entry as Travel 🡪 Click Next🡪add below project attributes.

Note:- Application name should begin with Z and in lowercase

* Name: **ZRAP100\_ks**
* Title: **Travel App KS**
* Description: **A Fiori application.**
* Add deployment configuration: Yes
* Add FLP configuration: Yes
* Configure advanced options: No

1. Click Next🡪Configure Deployment as below

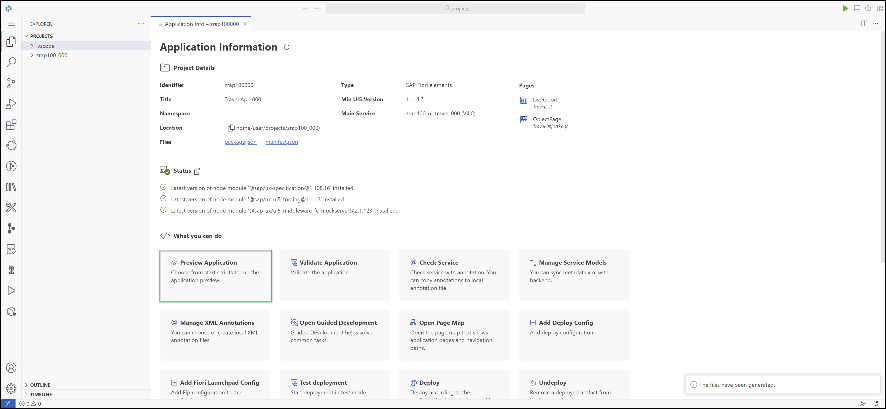
* Target: ABAP
* Destination: <your\_abap\_trial\_system>
* SAP UI5 ABAP Repository: ZRAP100\_KS
* Package: ZRAP100\_KS
* Transport Request: <your\_transport\_request>

1. Enter Fiori Launchpad Configuration 🡪 then click Finish 🡪Open Folder 🡪 you will see all files will be generated

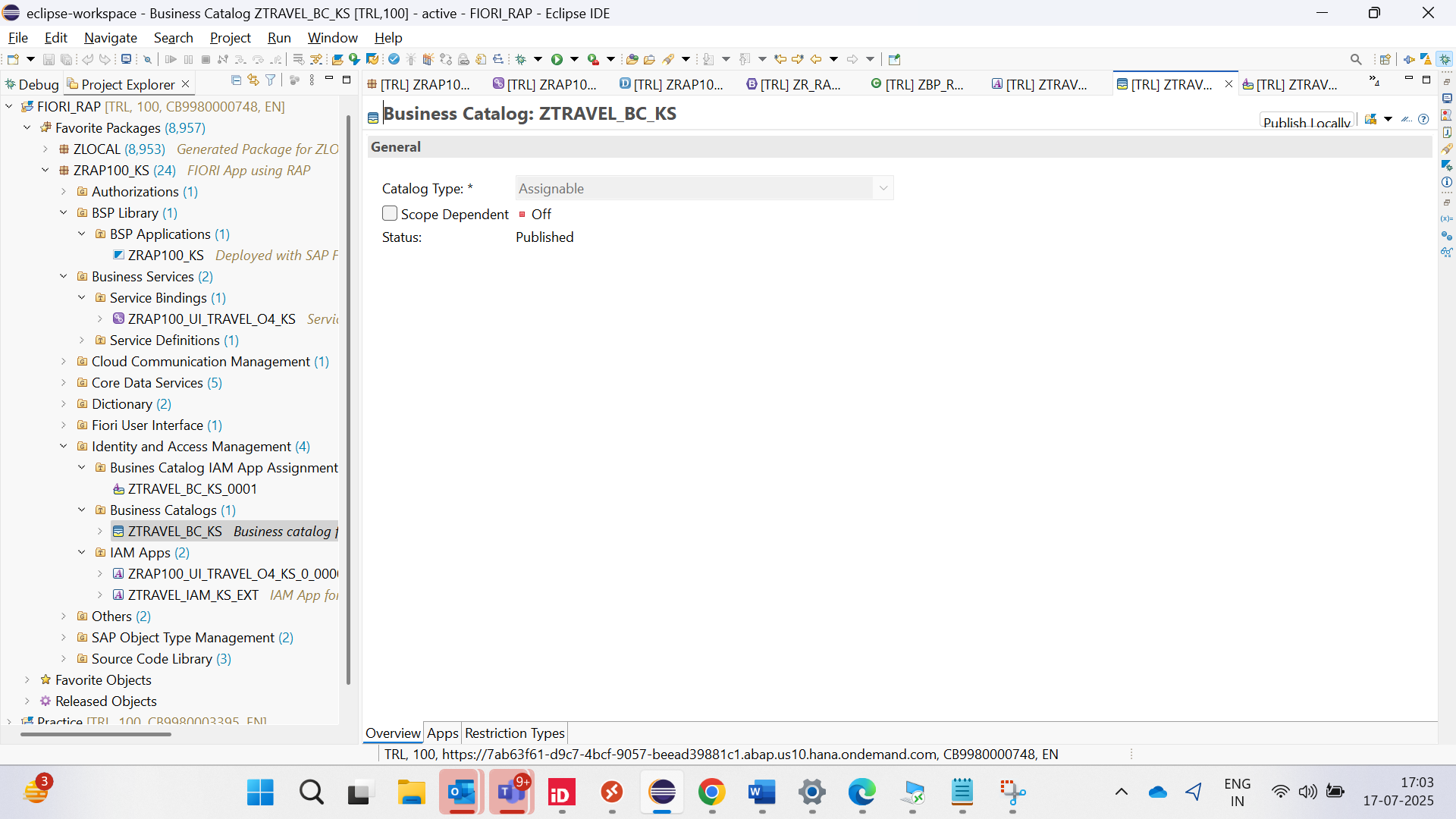
* Semantic Object: ZRAP100\_KS
* Action: display
* Title: Travel App KS

1. On application info screen🡪click on preview application🡪Select first entry🡪Click on GO🡪check the result

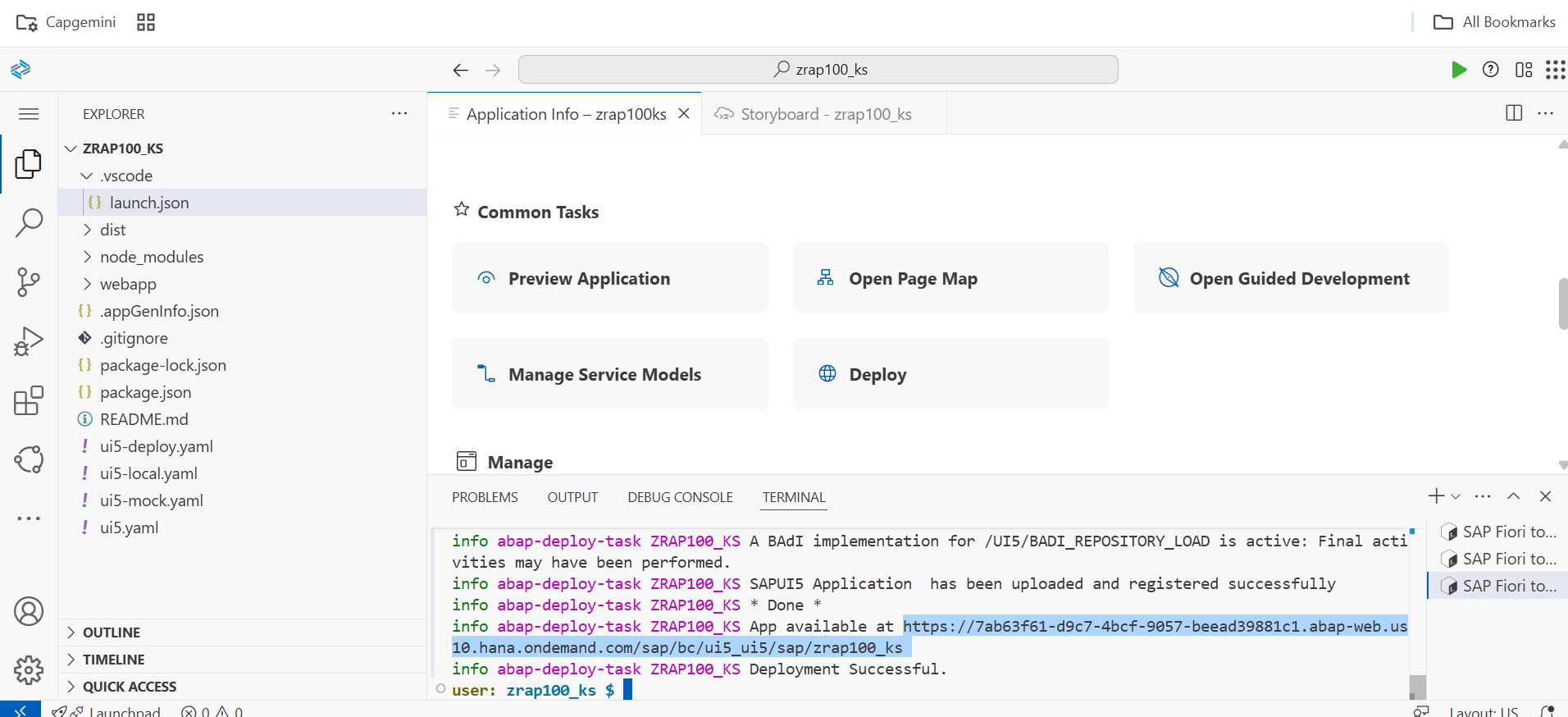
Note:- Remove Popup-blocker



1. On Application info🡪click “Deploy Application”🡪Press y🡪 When the deployment is successful, you will get this two information back as a result: **UIAD details** and **deployment successful 🡪 User zrap100\_ks**
2. Go to Eclipse🡪Refresh on package🡪You will see BSP application
3. Go to eclipse🡪Right click on package🡪select **New** > **Other Repository Object**🡪IAM APP🡪Click Next🡪Give name as ZTRAVEL\_IAM\_KS🡪 Description as “IAM App for travel app”🡪Application type as External🡪 Click next🡪Select TR and Finish.
4. Goto service tab🡪Insert🡪Service type as V4🡪Service name as “ZRAP100\_UI\_TRAVEL\_O4\_KS”🡪Click OK🡪Activate all object
5. Go to same IAM App🡪Click on create new Business catalog🡪Give name as **ZTRAVEL\_BC\_KS 🡪** Description as Business catalog for travel app🡪Select package🡪Click Finish🡪 Publish Locally.

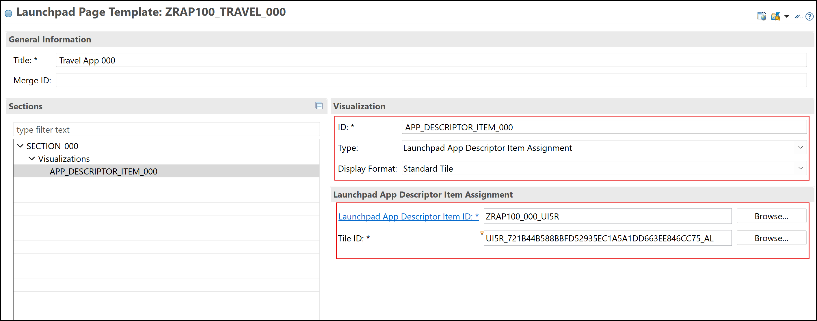


1. Go to BAS🡪Application Info🡪Deploy app🡪Press y🡪Copy url and paste in browser🡪Login and check.



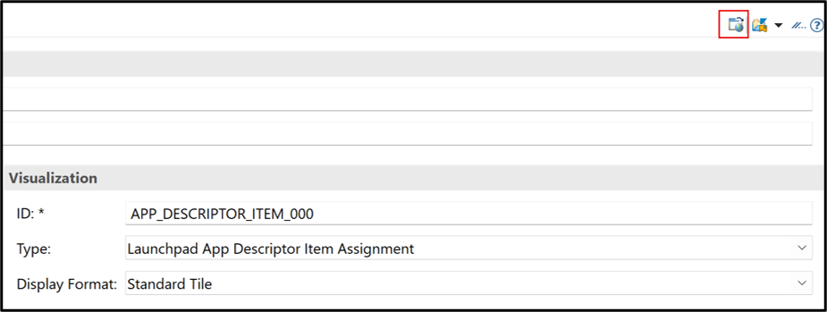
1. Create SAP Fiori Launchpad Space and Page Templates
   1. Create launchpad page template🡪 Open package ZRAP100\_KS in eclipse🡪 right-click **Fiori User Interface** and select **New** > **Launchpad Page Template**🡪 Name as “ZRAP100\_TRAVEL\_KS”🡪 Description as “Page for travel KS”🡪Title as Travel App KS🡪TR🡪Finish🡪Right click the list area in section and Select Add child>Section🡪Create new section with ID as “SECTION\_KS” 🡪 Title as SECTION KS🡪Right click and select **Add Child** > **Visualizations🡪 Right-click Visualizations and select Add Child > Visualization🡪** Create new visualization with ID APP\_DESCRIPTOR\_ITEM\_KS🡪add other details as below🡪 Then activate launchpad template

* Type: Launchpad App Descriptor Item Assignment
* Display Format: Standard Tile
* Launchpad App Descriptor Item ID:ZRAP100\_KS\_UI5R
* Title ID:

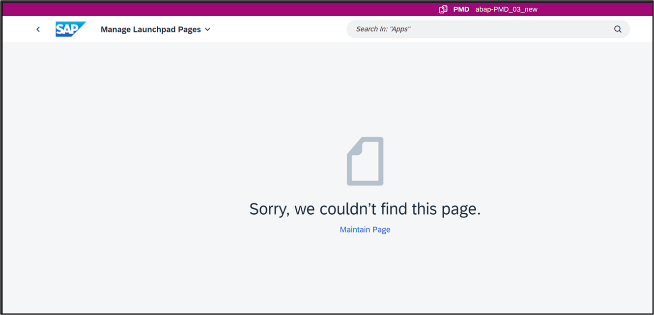


You can press CTRL + Space to load the existing data, to fill in your field entries.

b) Check Template availability in manage launchpad pages app



The launchpad page template will not be loaded. The page template needs to be scoped still in order to be able to use it.



c)