

# Kapitel 7

## Overview Page

*Die Overview Page ist ein Floorplan, der eine Übersicht verschiedener Informationen für die Anwender\*innen aufbereitet darstellt. Die jeweilige Darstellung hängt dabei auch von der Rolle des jeweiligen Endanwenders oder der Endanwenderin ab. Diese sollen sich auf ihre wesentlichen Aufgaben fokussieren und rasch auf Ereignisse reagieren können.*

In diesem Kapitel widmen wir uns im Detail dem Floorplan Overview Page. Anhand eines praktischen Beispiels zeigen wir Ihnen die Umsetzung einer Anwendung auf Basis dieses Floorplans. Diese Anwendung implementieren wir einmal in der SAP Web IDE und einmal im SAP Business Application Studio, damit Sie den Umgang mit beiden Entwicklungsumgebungen erlernen können.

Zunächst sehen wir uns in Abschnitt 7.1 im Detail an, welche Möglichkeiten die SAP Fiori Elements Overview Page bietet und wie sie aufgebaut ist. In Abschnitt 7.2 behandeln wir anschließend die Annotationen, die notwendig sind, um diesen Floorplan umzusetzen. Die folgenden Abschnitte behandeln die Entwicklung einer Beispielanwendung auf Basis der Overview Page. Zunächst zeigen wir Ihnen in Abschnitt 7.3, wie Sie den zugrunde liegenden CDS View aufbauen können. Anschließend zeigen wir Ihnen die Implementierung der Benutzeroberfläche im SAP Business Application Studio (siehe Abschnitt 7.4) und in der SAP Web IDE (siehe Abschnitt 7.5).

### 7.1 Einführung

In den SAP Fiori Design Guidelines ist genau definiert, wie eine SAP Fiori Elements Overview Page auszusehen hat. Dieser Floorplan dient grundsätzlich als zentraler Einstiegspunkt. Von hier aus kann in andere Floorplans und Sichten abgesprungen werden. So könnten Sie beispielsweise auf der Overview Page einen Überblick über die Informationen zu einem Fachbereich aus verschiedenen Anwendungen geben. In weiterer Folge könnten Sie mithilfe eines List Report Floorplans oder einer Object Page detaillierte-

re Informationen darstellen. In Abbildung 7.1 sehen Sie, wie eine Overview Page nach den SAP Fiori Design Guidelines aussieht.

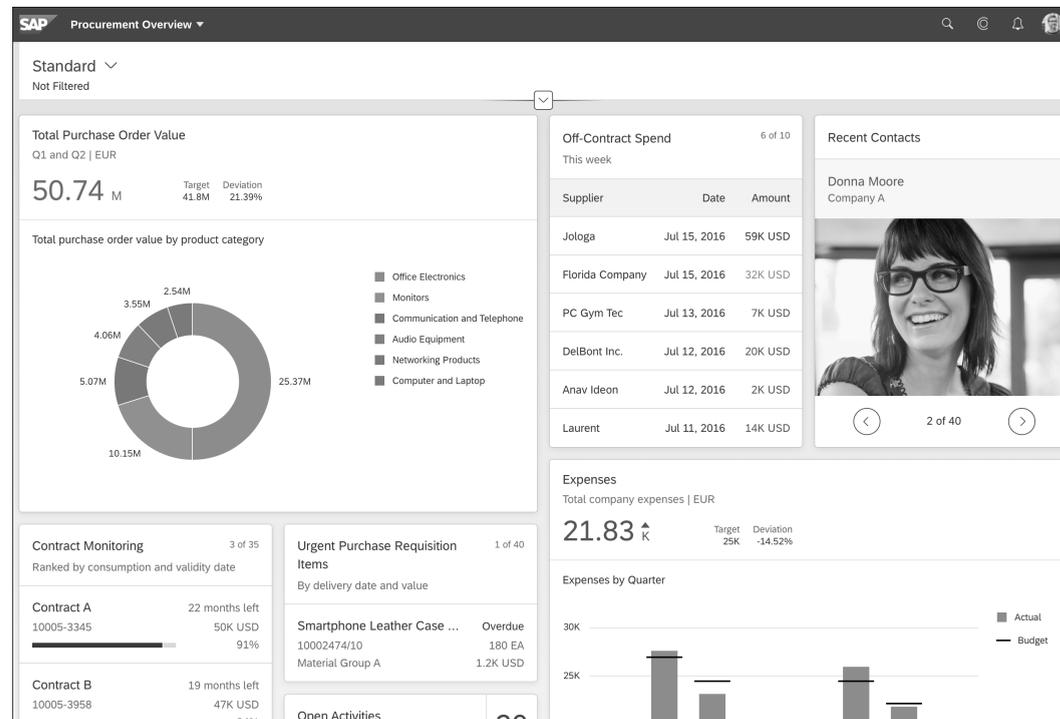


Abbildung 7.1 Overview Page in den SAP Fiori Design Guidelines (Quelle: SAP)

### 7.1.1 Aufbau einer Overview Page

#### Grundlegende Seitenelemente

In Abschnitt 6.1.1, »Overview Page«, haben wir den allgemeinen Aufbau einer Overview Page bereits grob vorgestellt. An dem Beispiel in Abbildung 7.1 erkennen Sie die folgenden grundlegenden Elemente:

#### ■ Dynamic Page Header

Im oberen Bereich dieser Seite finden Sie den *Dynamic Page Header*. Dieser Kopfbereich enthält eine Kopfzeile, die grundlegende Informationen wie einen Seitentitel beinhaltet. Des Weiteren findet sich hier ein Smart-Filter-Bereich, über den sich die dargestellten Informationen durch Filter und Suchen einschränken lassen. Dieser Bereich kann expandiert sowie komprimiert werden.

#### ■ Dynamic Page Content

Im unteren Bereich finden Sie den *Dynamic Page Content*. Diese Content Area beinhaltet die eigentlichen Informationen in verschiedenen Aufbereitungsvarianten. Wenn Sie eine Overview Page erstellen, obliegt es Ihnen,

welche Informationen Sie hier auf welche Art und Weise darstellen möchten. Grundsätzlich werden für die Darstellung der Daten Cards verwendet. Ob Sie auf diesen Cards eine Tabelle, ein Diagramm oder sonstige Inhalte einbetten, bleibt Ihnen überlassen.

Die im Dynamic Page Header eingebettete *Smart Filter Bar* verwendet den sogenannten *Live Update Mode*. Dieser Modus hat folgende Auswirkung: Setzt ein Anwender oder eine Anwenderin einen Filter, werden die Suchergebnisse im Inhaltsbereich sofort automatisch eingeschränkt. Ein Aktivieren des Filters, etwa durch einen **Go**-Button, ist dadurch nicht nötig.

Der Header-Bereich kann komprimiert und expandiert werden. Es besteht außerdem die Möglichkeit, diesen Bereich zu fixieren bzw. zu lösen. Durch die Fixierung bleibt der Header-Bereich sichtbar, wenn Sie nach unten scrollen.

Während die Vergabe und Anzeige eines Titels für die Overview Page im Header verpflichtend ist, ist das Einbinden weiterer Inhalte, wie der Smart Filter Bar, optional.

Grundsätzlich werden drei verschiedene Varianten des Dynamic Page Headers einer Overview Page unterschieden, auf die wir im Folgenden eingehen:

#### ■ Variante 1: Variantenmanagement und Smart Filter Bar

Die erste Variante ist die Standardvariante und wird vermutlich am häufigsten verwendet. In dieser Variante finden Sie die Smart Filter Bar im Header-Bereich. Anstelle des Titels wird das *Variantenmanagement* einblendet (erkennbar an dem Button **Standard**). Über diese Funktion können die Anwender\*innen einzelne Varianten erstellen und abrufen, die gewisse häufig verwendete Filterkonstellationen umfassen. Die Varianten müssen von den Benutzer\*innen jedoch selbst verwaltet werden.

Außerdem finden Sie in dieser Variante des Dynamic Page Headers noch einen Teilen-Button (🔗). Über diesen können Sie Optionen wie **Send Email** oder **Save as Tile** auswählen.



Abbildung 7.2 Dynamic Page Header – Variante 1 (Quelle: SAP)

Dynamic Page Header

Obligatorische und optionale Inhalte

Varianten

In Abbildung 7.2 sehen Sie diese Variante des Headers im ausgeklappten Zustand. Initial ist der Header-Bereich komprimiert, d. h., es wird lediglich die Titelzeile angezeigt.

■ **Variante 2: Text im Header-Titel und Smart Filter Bar**

Im Gegensatz zur ersten Variante wird in Variante 2 auf die Verwendung des Variantenmanagements verzichtet. Wie in Abbildung 7.3 zu sehen, wird zusätzlich zur Smart Filter Bar lediglich ein Titel angezeigt. Auch hier ist der Header-Bereich standardmäßig wieder komprimiert.



Abbildung 7.3 Dynamic Page Header – Variante 2 (Quelle: SAP)

■ **Variante 3: Text im Header-Titel**

Die dritte Variante ist sehr schlank gehalten. Außer dem Titel werden dabei keine weiteren Informationen oder Aktionen im Header-Bereich angezeigt (siehe Abbildung 7.4).



Abbildung 7.4 Dynamic Page Header – Variante 3 (Quelle: SAP)

**Dynamic Page Content**

Im Dynamic Page Content gestalten Sie den eigentlichen Inhalt der Overview Page. Hier können Sie Daten auf verschiedenste Arten aufbereiten. So können Sie Tabellen, Diagramme oder sonstige Designelemente verwenden, um eine Übersicht über ein bestimmtes Thema zu geben. In den meisten Fällen wird von der Overview Page in weiterer Folge auf andere Floorplans abgesprungen.



**Cards statt Kacheln verwenden**

Im Dynamic Page Content einer Overview Page sollten Sie auf die Verwendung von Kacheln verzichten. Stattdessen sind Cards vorgesehen, um Daten darzustellen. Innerhalb einer Card können beliebige Designelemente eingebettet werden.

Für den Content-Bereich der Overview Page existieren zwei Gestaltungsvarianten:

Varianten

- Fixed Card Layout
- Resizable Card Layout

Wie aus den Bezeichnungen der beiden Varianten bereits hervorgeht, unterscheiden sie sich hinsichtlich ihrer Anpassbarkeit.

In der Datei **manifest.json** Ihrer SAP-Fiori-Elements-Applikation haben Sie die Möglichkeit festzulegen, ob Sie das Fixed oder das Resizable Card Layout verwenden. Dafür konfigurieren Sie die Eigenschaft `containerLayout`, die die Werte `fixed` oder `resizable` annehmen kann. Zudem können Sie in der Datei **manifest.json** auch einsehen, welche Cards aktuell angezeigt werden. In Listing 7.1 sehen Sie einen Ausschnitt des relevanten Bereichs der Datei.

Wahl der Layoutvariante

7

```
"sap.ovp": {
  "globalFilterEntityType": "OVP_Filter",
  "containerLayout": "resizable",
  "enableLiveFilter": false,
  "considerAnalyticalParameters": false,
  "cards": {
    "cardDemo": {
      "template": "sap.ovp.cards.table",
      "settings": {
        "title": "{{cardDemo_title}}",
        "subTitle": "{{cardDemo_subTitle}}",
        "entitySet": "DemoSet",
        "sortBy": "Id",
        "sortOrder": "ascending",
        "addODataSelect": false
      }
    }
  }
}
```

Listing 7.1 Konfiguration der Cards in der Datei »manifest.json«

Damit die Cards und Informationen im Content-Bereich der Overview Page angezeigt werden, müssen sie dem zugrunde liegenden CDS View oder der eigens dazu vorgesehenen Annotationsdatei Annotationen hinzugefügt werden. Diese Annotationen stellen wir in Abschnitt 7.2 vor.

**Fixed Card Layout** Im Folgenden beschreiben wir die beiden Layoutvarianten genauer. Das Fixed Card Layout zeichnet sich dadurch aus, dass die Höhe und Breite jeder einzelnen Card vordefiniert werden muss. Diese Cards passen sich hinsichtlich ihrer Größe nicht an das Ausgabegerät an, und die Größe kann auch nicht von den Endanwender\*innen beeinflusst werden. Dadurch erhält man eine statische Ansicht. Zudem ist auch die maximale Anzahl an Spalten eingeschränkt. Es können maximal fünf Spalten angezeigt werden. In Abbildung 7.5 sehen Sie, wie eine solche Overview Page aussehen könnte. Die verschiedenen Arten von Cards, auf die hier mit Callout-Ziffern verwiesen wird, erläutern wir in Abschnitt 7.1.2, »Verwendung der Overview Page«. Das Aussehen wäre unabhängig vom verwendeten Gerät immer gleich. Der Implementierungsaufwand für dieses Layout ist geringer als der für das Resizable Card Layout. Außerdem ermöglicht diese Ansicht meist auch einen besseren und schnelleren Überblick, da die Platzierung der einzelnen Cards unabhängig vom verwendeten Gerät immer gleich ist.

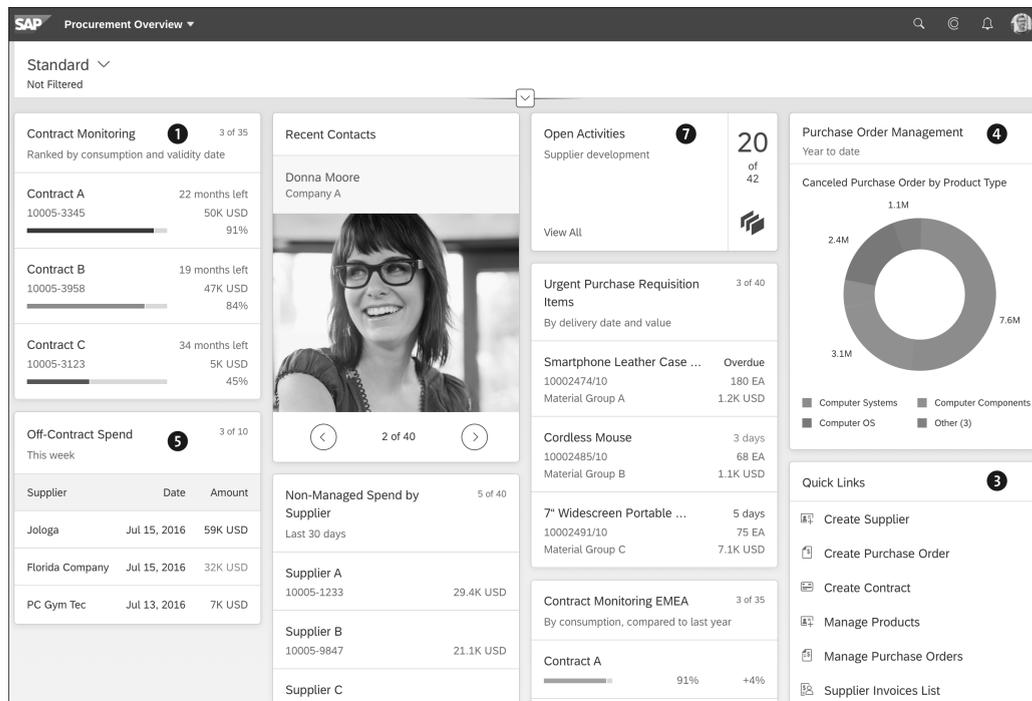


Abbildung 7.5 Fixed Card Layout (Quelle: SAP)

**Resizable Card Layout** Im Gegensatz dazu müssen im Resizable Card Layout die Dimensionen der einzelnen Cards nicht vorgegeben werden. Dieses Layout verfolgt einen responsiven Ansatz; die Cards passen ihre Größe der Bildschirmgröße des Geräts an. Zudem können die Dimensionen von den Endanwender\*innen

individuell angepasst werden – beispielsweise durch Zoomen. Zusätzlich können die einzelnen Cards per Drag & Drop beliebig verschoben werden.

Bei der Gestaltung der Overview Page sind Sie bei diesem Layout nicht an eine maximale Spaltenanzahl gebunden, sondern können diese selbst definieren. Durch die Anpassbarkeit und Flexibilität erreicht man zwar ein dynamisches User Interface, dies ist allerdings auch mit einem größeren Implementierungsaufwand verbunden. Wie eine flexible Overview Page aussehen könnte, sehen Sie in Abbildung 7.6.

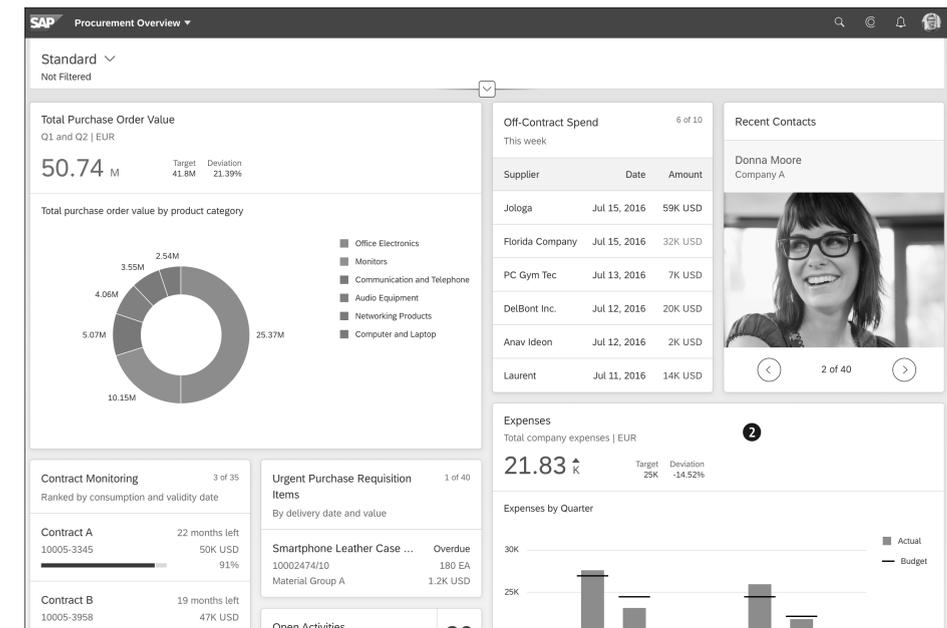


Abbildung 7.6 Dynamic Page Resizable Page Layout (Quelle: SAP)

## 7.1.2 Verwendung der Overview Page

Die Overview Page können Sie in folgenden Anwendungsfällen verwenden: **Anwendungsfälle**

- Sie möchten den Endanwender\*innen einen zentralen Einstiegspunkt zur Verfügung stellen, dessen Inhalte einen Bezug zu einer bestimmten Rolle haben.
- Die Anwender\*innen möchten für ihre Rolle relevante Informationen filtern und darauf reagieren können.
- Sie möchten Informationen darstellen, die aus zwei oder mehr verschiedenen Anwendungen stammen.
- Sie möchten Daten in verschiedenen Formaten aufbereiten, beispielsweise als Listen, Tabellen oder Diagramme.

Auf einer Overview Page sollten Sie mindestens drei Cards anbieten. Diese sollten unterschiedliche Designs aufweisen, um eine gewisse Bandbreite und unterschiedliche Perspektiven auf die Daten zeigen zu können.

#### Ausschlusskriterien für die Verwendung

In den folgenden Fällen sollten Sie die Overview Page dagegen nicht verwenden:

- Sie möchten eine Detailsicht auf die Daten anbieten. Hierfür ist die Overview Page nicht geeignet. Nutzen Sie stattdessen den Floorplan Object Page, um Details zu spezifischen Objekten anzuzeigen.
- Sie möchten den Anwender\*innen lediglich einen zentralen Einstiegspunkt bieten, über den sie andere Applikationen starten können. Für diesen Zweck sollten Sie das SAP Fiori Launchpad verwenden.
- Sie möchten Daten aus nur einer Anwendung anbieten bzw. weniger als drei Cards anzeigen. In diesem Fall sollten Sie stattdessen auf die Object Page setzen.

#### Arten von Cards

Wie Sie in Abbildung 7.5 und Abbildung 7.6 bereits gesehen haben, können Sie ganz verschiedene Arten von Cards verwenden:

##### 1 List Cards

Abhängig von der Konfiguration der `ListItem`-Annotation zeigen *List Cards* Listen von Einträgen an. Innerhalb einer List Card können Sie bis zu sechs Felder in einer Liste einfügen.

##### 2 Bar Chart List Cards

Die *Bar Chart List Card* wird verwendet, um eine Vielzahl von Elementen in einer vertikalen Liste anzuzeigen. Im Unterschied zur List Card wird diese Card in eine andere Komponente eingebettet, den *Comparison Micro Chart*.

##### 3 Link List Cards

In einer *Link List Card* können Sie eine Reihe von Links anzeigen. Zusätzlich können Sie den Links einen Titel, Untertitel, Bilder oder Icons hinzufügen.

##### 4 Analytical Cards

Eine *Analytical Card* bietet die Möglichkeit, analytische Daten darzustellen. Dazu können verschiedene Typen von Diagrammen herangezogen werden, z. B. Bar Charts, Column Charts, Donut Charts usw. Neben einem Diagramm beinhaltet eine Analytical Card immer auch einen Header-Bereich.

##### 5 Table Cards

Mit einer *Table Card* können Sie eine Liste von Einträgen im Tabellenformat darstellen. Auch hier ist wieder die Konfiguration der Annotation

`ListItem` ausschlaggebend. Mit diesem Card-Typ können bis zu drei Spalten angezeigt werden. Optional können Sie zusätzlich Smart Links definieren, die in der Table Card als sogenannte *Quick Access Links* angezeigt werden. Das sind Links, mit denen beispielsweise eine detaillierte Ansicht eines einzelnen Objekts geöffnet werden kann.

##### 6 Quick View Cards

Quick View Cards sind sogenannte *Single Object Cards*, die die grundlegenden Daten eines einzelnen Business-Objekts abbilden. Das könnten z. B. der Name, Adresdaten und die Telefonnummer eines Kontakts sein.

##### 7 Stack Cards

Mithilfe einer *Stack Card* können Sie mehrere Cards der gleichen Art zu einer Card aggregieren. Nach einem Klick auf diese Card können die Anwender\*innen die Applikation öffnen, die die Informationen aus dieser Card liefert, beispielsweise einen List Report. Außerdem können Sie den sogenannten *Object Stream* öffnen.

#### Object Stream

Der Object Stream erscheint als Overlay auf der Overview Page. Dieser dient dazu, diverse Cards mit hinterlegten Aktionen anzuzeigen. Innerhalb eines solchen Dialogs können Sie bis zu 20 Stacked Cards anzeigen. Zusätzlich haben Sie die Möglichkeit, zur übergeordneten Applikation dieser Card zu navigieren.

Alle Cards, die in diesem Dialog ersichtlich sind, beschaffen ihre Daten von der gleichen Applikation.

Es gilt zu beachten, dass die Stack Card und die Quick View Card nur unter Verwendung eines OData-Service der Version V2 verfügbar sind. Wenn Sie mit einem OData-Service der Version V4 arbeiten, stehen Ihnen diese beiden Arten von Cards nicht zur Verfügung.

Mit kundeneigenen Cards, den sogenannten *Custom Cards*, erhalten Sie die Möglichkeit, das Verhalten und das Erscheinungsbild einer Card selbst zu definieren. Dies ist besonders nützlich, wenn Sie Funktionen anbieten möchten, die von den Standard-Card-Typen nicht abgedeckt werden können.

Da die auf der Overview Page angeordneten Cards einen möglichst aussagekräftigen Überblick über den definierten Themenbereich geben sollen, ist es natürlich wichtig, dass die angezeigten Daten entsprechend aktuell gehalten werden. Das kann ganz ohne zusätzlichen Entwicklungsaufwand erreicht werden. In der Datei `manifest.json` der Overview Page finden Sie die



Custom Cards

Datenaktualisierung

Eigenschaft `refreshIntervalInMinutes`, mit der das Intervall definiert wird, in dem die Daten auf den Cards aktualisiert werden sollen. Es wird in Minuten angegeben. In Listing 7.2 sehen Sie den entsprechenden Ausschnitt der Datei, der zeigt, wie die Konfiguration dieses Parameters aussehen könnte.

```
"sap.ovp": {
  ...
  "refreshIntervalInMinutes": 5,
  ...
  "cards": { ... }
}
```

**Listing 7.2** Aktualisierungsintervall in der Datei »manifest.json« festlegen

In diesem Beispiel würden die Cards alle 5 Minuten aktualisiert. Wird diese Eigenschaft nicht definiert, wird standardmäßig ein Intervall von 1 Minute verwendet. Dieser Standardwert ist zugleich auch der Minimalwert.

#### Best Practices zur Seitengestaltung

Bevor Sie mit der Umsetzung einer Overview Page starten, sollten Sie sich Gedanken machen, was Sie mit der Seite bezwecken möchten und welches Aussehen Sie anstreben. Die folgenden Best Practices haben sich bewährt, um eine möglichst hohe Benutzerfreundlichkeit zu erzielen:

- **Legen Sie im Vorhinein schon fest, wie viele Cards Sie anzeigen werden**  
Stellen Sie sicher, dass nur Cards angezeigt werden, die den Benutzer\*innen einen Mehrwert bringen. So sollten die verfügbaren Cards beispielsweise immer auf die jeweilige Rolle oder Aufgabe der Anwender\*innen abgestimmt sein.
- **Heben Sie die wesentlichen Informationen hervor**  
Wählen Sie bedeutungstragende Farben, um wichtige Informationen hervorzuheben. Dies betrifft vor allem Texte und Diagramme, aber auch einzelne Informationen auf großen Cards.
- **Versuchen Sie, möglichst passende Arten von Cards zu mischen**  
Um Informationen anschaulicher darzustellen und einen hohen Wiedererkennungswert zu erreichen, sollten Sie auf eine möglichst passende Durchmischung der verschiedenen Card-Typen setzen.
- **Setzen Sie auf eine bewusste Reihenfolge**  
Anwender\*innen empfinden große Cards, die sich im oberen Bereich der Seite befinden, meist als wichtiger als andere Cards. Das sollten Sie bei der Reihenfolge Ihrer Cards beachten.

#### ■ Gruppieren Sie zusammengehörige Themen

Neben der Reihenfolge der einzelnen Cards ist es auch wichtig, darauf zu achten, dass zusammengehörige Informationen nah beieinander angezeigt werden. Auch dies trägt zu einer besseren Übersicht bei.

#### ■ Wählen Sie aussagekräftige Texte

Vor allem, wenn Sie Cards mit Aktionen verwenden, sollten Sie auf aktivierende Texte setzen.

#### ■ Achten Sie auf Durchgängigkeit

Endanwender\*innen sind es gewöhnt, dass z. B. bestimmte Status farblich gekennzeichnet werden. So sollten z. B. Erfolgsmeldungen grün dargestellt werden, Warnungen orange oder Fehler rot. Dabei ist es wichtig, dieselbe Kennzeichnung konsistent in der gesamten Applikation zu verfolgen.

## 7.2 Annotationen für Overview Pages

In diesem Abschnitt stellen wir Ihnen Annotationen vor, die für die Erstellung einer Overview Page essenziell sind. Diese Annotationen können sowohl direkt im CDS View als auch lokal in der SAPUI5-Anwendung hinzugefügt werden. In Bezug auf die Wiederverwendbarkeit der Annotationen spricht viel für das Hinzufügen direkt im CDS View.

#### ■ UI.SelectionVariant

– Beispiel: `UI.SelectionVariant [{qualifier: 'Default'}]`

Diese Annotation ermöglicht es, die Daten unter der Verwendung von Filtern vorzuselektieren. So können für spezifische Cards definierte Filter angewendet werden, um zum einen die Ladezeit zu verkürzen und zum anderen die Daten für die jeweilige Card zu aggregieren. Über eine bis zu 120 Zeichen lange Zeichenkette kann ein eindeutiger Bezeichner für die zu verwendende Variante angegeben werden.

#### ■ UI.HeaderInfo

– Beispiel: `UI.headerInfo: {typeName: 'Bestellung', typeNamePlural: 'Bestellungen', typeImageUrl: '/img/bestellung.png'}`

Die Annotation `UI.HeaderInfo` dient grundsätzlich dazu, die Eigenschaften eines Titels zu definieren. Im Fall der Overview Page wird mit dieser Annotation der Titelbereich einer Card definiert. Damit können Sie definieren, ob Sie einen Titel (für die Einzahl und für die Mehrzahl), eine Beschreibung und/oder optional ein Bild anzeigen möchten. Die Titel können hierbei bis zu 60 Zeichen lang sein.

- `UI.FieldGroup`
  - Beispiel: `UI.FieldGroup.label: 'Bezeichnung'`

Mithilfe dieser Annotation wird eine Gruppe von Datenfeldern definiert, wobei jedes Datenfeld ein Label (also einen Anzeigetext) und einen Wert umfasst. Auf diese Weise kann ein Formular aufgebaut werden. Dieser Anzeigetext weist eine maximale Länge von 60 Zeichen auf.
- `UI.Facets`
  - Beispiel: `UI.Facet: [{Label: 'Demo', targetElement: 'demoElement'}]`

Die Annotation `UI.Facets` wird verwendet, um einen Kontext zwischen einer Card und einer anderen Annotation herzustellen. So würde es beispielsweise nicht genügen, lediglich die Annotation `UI.FieldGroup` hinzuzufügen, sondern diese muss auch über eine `UI.Facets`-Annotation einer Card zugeordnet werden. Diese Zuordnung erfolgt über die Eigenschaft `Qualifier`.
- `UI.Chart`

Mit dieser Annotation kann definiert werden, welche Dimensionen und Maße für die Darstellung eines Diagramms verwendet werden sollen. Dazu wird mit `UI.Chart.DimensionAttributes.Dimension` die Dimension und mit `UI.Chart.MeasureAttributes.Measure` das Maß definiert. Ein detailliertes Beispiel dazu sehen Sie in Listing 7.8.

#### Card-spezifische Annotationen

Je nachdem, welche Art von Card Sie verwenden, unterscheiden sich die zusätzlich angebotenen Annotationen. Implementieren Sie eine List oder Table Card, benötigen Sie beispielsweise die Annotation `UI.LineItem`. Wählen Sie stattdessen eine Analytical Card, werden Sie die Verwendung einer `UI.Chart`-Annotation in Betracht ziehen müssen. Die `UI.LineItem`-Annotationen haben wir in Abschnitt 6.3, »Generische Annotationen«, vorgestellt.

### 7.3 CDS View für eine Overview Page

Wie so oft beginnt die Entwicklung einer SAP-Fiori-Elements-Applikation auch im Fall einer Overview Page mit einem CDS View. Dieser wird in den ABAP Development Tools (ADT) in Eclipse definiert (siehe auch Abschnitt 5.2.2, »Eclipse«).

#### CDS View mit Autorendaten

In Listing 7.3 definieren wir einen beispielhaften CDS View. Hier geben wir einerseits die Daten vor, die auf unserer Overview Page ausgegeben werden sollen, andererseits definieren wir auch schon über Annotationen, wie die-

se Daten angezeigt werden sollen. Über diese Annotationen werden gewisse Definitionen und Informationen an die Benutzeroberfläche durchgereicht. Das User Interface unserer SAP-Fiori-Elements-Applikation kann sich auf dieser Basis quasi »von selbst« aufbauen. Zunächst einmal soll unsere Overview Page Autorendaten anzeigen, die aus der Tabelle `ZAUTHOR` stammen. Wir geben an, welche Felder dieser Datenquelle angezeigt werden sollen, unter anderem sind Anrede (`title`), Vorname (`firstname`), Nachname (`lastname`), Geburtsdatum (`birthday`) und Mailadresse (`email`) sichtbar.

Die eingefügten Annotationen kennen Sie bereits. Die `UI.LineItem`-Annotationen sorgen für die Spaltendefinitionen, die dann von einer Tabelle ausgelesen werden, und die Annotation `UI.selectionField` sorgt für die Filterbedingungen, die in der Filter Bar stehen sollen. Weiter definieren wir die Assoziation, sprich die Bezeichnung, zwischen Autoren und ihren Büchern. Diese Annotationen sind externe Annotationen, da sie direkt im CDS View definiert sind und mit dem OData-Service Richtung Applikation durchgereicht werden (siehe Abschnitt 3.5, »OData- und CDS-Annotationen«).

Vergessen Sie nicht, den CDS View mit der Annotation `@OData.publish` direkt als OData-Service freizugeben, wie in Listing 7.3 gezeigt.

CDS View als OData-Service publizieren

```
@OData.publish: true
define view ZOV_P_AUTHORS as select from zauthor
  association [0..*] to ZOV_P_BOOKS as books
  on $projection.id = books.AuthorId {
    key id,
    title,
    @UI.lineItem: [{position: 10}]
    @UI.selectionField: [{position: 10}]
    firstname,
    @UI.lineItem: [{position: 20}]
    @UI.selectionField: [{position: 10}]
    lastname,
    birthday,
    CONCAT_WITH_SPACE(firstname, lastname, 1) as fullname,
    @UI.selectionField: [{position: 10}]
    email,
    books
  }
}
```

Listing 7.3 CDS View für das Lesen von Autorendaten

**Zweiter CDS View  
mit Buchdaten**

Auf unserer Overview Page wollen wir aber nicht nur die Autorendaten anzeigen, sondern auch Listen und Auswertungen zu deren Büchern darstellen. Für diesen Zweck definieren wir einen zweiten CDS View, den wir ebenfalls direkt als OData-Service freigeben und registrieren. Diesen zweiten CDS View sehen Sie in Listing 7.4.

```
define view ZOVP_BOOKS as select from zbook
  association [1] to ZAUTHOR_CV as author
  on author.Id = $projection.AuthorId{
    key id as Id,
    key author_id as AuthorId,
    @UI.lineItem: [{ hidden: false }]
    title as Title,
    @UI.lineItem: [{ hidden: false }]
    @Semantics.amount.currencyCode: 'Currency'
    price as Price,
    @Semantics.currencyCode: true
    currency as Currency,
    @UI.lineItem: [{ hidden: false }]
    isbn as Isbn,
    category as Category,
    @UI.lineItem: [{ hidden: false, label: 'Vorrat' }]
    stock as Stock,
    author as author
  }
```

**Listing 7.4** CDS View für das Lesen von Buchdaten

Dieser CDS View liest von der Datenquelle `zbook` und besitzt eine Assoziation zu dem vorhergehenden CDS View. Aus dieser Datenquelle lesen wir unter anderem die Attribute Anrede (`title`), Preis (`price`) und die dazugehörige Währung (`currency`), ISBN (`isbn`) und den Vorrat an Büchern (`stock`). Für all diese Attribute vergeben wir mit `as` einen Alias, also einen anderen Namen.

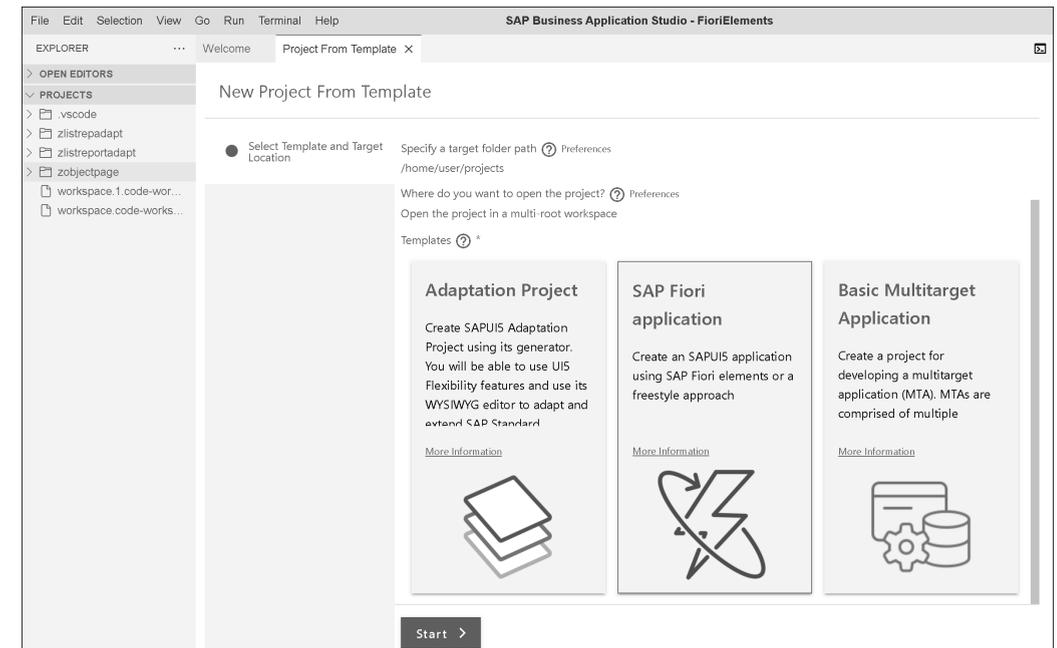
## 7.4 Entwicklung einer Overview Page im SAP Business Application Studio

Auf Basis der im vorangegangenen Abschnitt erstellten CDS Views können Sie die SAP-Fiori-Elements-Anwendung anlegen.

### 7.4.1 Projekt anlegen

Im SAP Business Application Studio müssen Sie dazu als Erstes ein neues Projekt anlegen:

1. Klicken Sie dazu auf der Startseite der Entwicklungsumgebung auf **Start from Template** oder in der oberen Menüleiste auf **File • Start from Template**. Nennen Sie das Projekt z. B. »zobjectpage«.
2. Als Nächstes können Sie den Projekttyp wählen, wie in Abbildung 7.7 gezeigt. Wählen Sie hier **SAP Fiori application** aus und bestätigen Sie mit **Start**.



**Abbildung 7.7** Ein neues Projekt im SAP Business Application Studio anlegen

3. Im nächsten Schritt werden Ihnen die verschiedenen Arten von SAP-Fiori-Elements-Applikationen bzw. deren Floorplans vorgeschlagen. Wählen Sie die **Overview Page** aus (siehe Abbildung 7.8).
4. SAP-Fiori-Elements-Applikationen benötigen einen OData-Service als Datenquelle. Deswegen müssen Sie im nächsten Schritt das Quellsystem auswählen, von dem Sie die Daten konsumieren möchten, und anschließend den OData-Service bestimmen (siehe Abbildung 7.9).

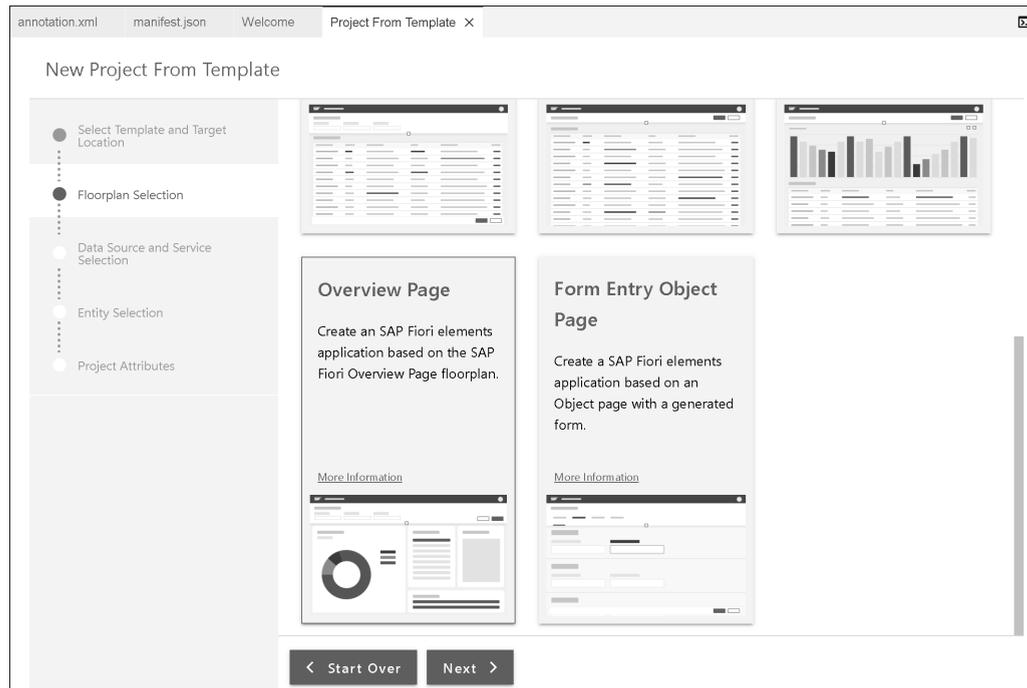


Abbildung 7.8 Floorplan auswählen

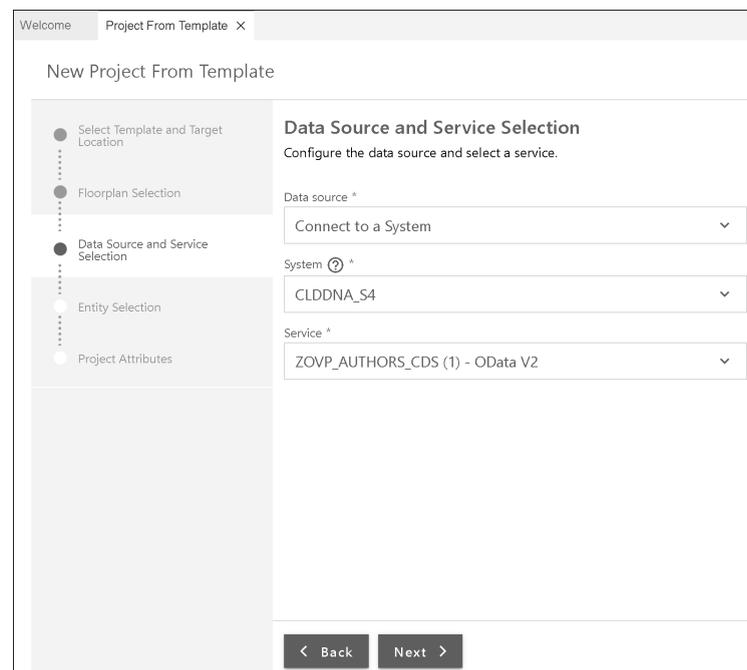


Abbildung 7.9 Datenquelle auswählen

5. Eine Eigenheit der Overview Page haben Sie schon in den vorangegangenen Abschnitten kennengelernt: Sie können Filterbedingungen auf einen bestimmten Entitätstyp anwenden. Diese Filter greifen über mehrere Cards hinweg, auf denen das Attribut, nach dem wir filtern, gefunden wird. Im nächsten Schritt müssen Sie dazu einen Entitätstyp auswählen (siehe Abbildung 7.10).

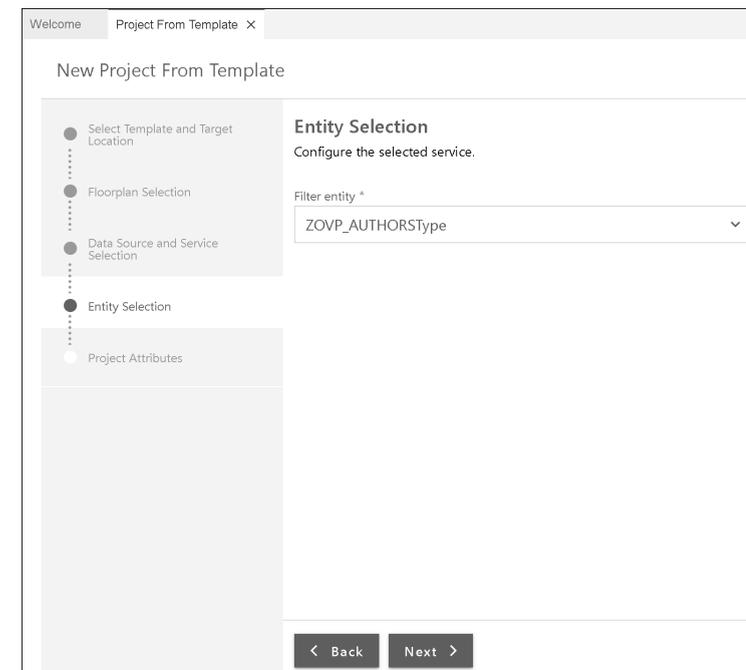


Abbildung 7.10 Entitätstyp auswählen

6. Im letzten Schritt des Erstellungsprozesses müssen Sie die Projektdetails, wie z. B. den Modulnamen, den Applikationstitel oder den Speicherort, angeben (siehe Abbildung 7.11).

#### 7.4.2 Table Card hinzufügen

Nachdem das Projekt definiert ist, können Sie mit der Gestaltung der Benutzeroberfläche beginnen. Als erste Card wollen wir eine Table Card auf unserer Overview Page anzeigen. Die externen Annotationen `@UI.lineItem`, die dazu notwendig sind, haben wir bereits im CDS View in Abschnitt 7.3, »CDS View für eine Overview Page«, definiert. Wir müssen nun nur noch die lokalen Definitionen in der Applikation selbst erledigen, damit die Card auch sichtbar wird.

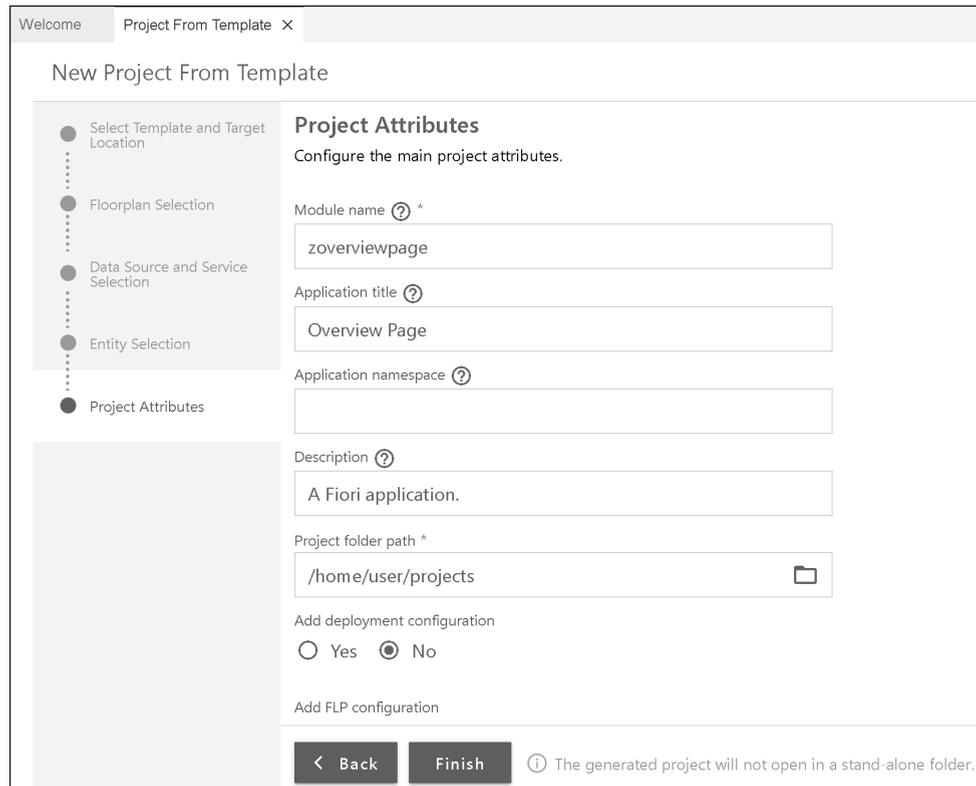


Abbildung 7.11 Projektdetails angeben

### Guided Development

Mit dem sogenannten *Guided Development* können Sie lokale Änderungen an der Applikation in einem geführten Dialog vornehmen. Da es sich bei solchen Änderungen an SAP-Fiori-Elements-Applikationen oft um spezifische Definitionen in den Metadaten handelt, die man nicht unbedingt kennt, erhält man an dieser Stelle eine passende Anleitung durch einen Assistenten, die man dann Schritt für Schritt durchgehen kann. In Kapitel 12, »Erweiterungen«, gehen wir noch näher auf Funktionsweise und Möglichkeiten des Guided Developments ein.

Rufen Sie den Assistenten für unser Beispiel wie folgt auf:

1. Öffnen Sie das Kontextmenü Ihres Projekts im Explorer, und wählen Sie den Eintrag **SAP Fiori Tools – Open Guided Development**. Alternativ können Sie über den Menüpfad **View • Find Command...** das Kommandofenster öffnen und nach **Fiori: Open Guided Development** suchen. Starten Sie dann diesen Befehl.
2. Sie gelangen zu der Ansicht in Abbildung 7.12. Aus dieser Liste wählen Sie die Anleitung **Add a table card to an overview page** aus.

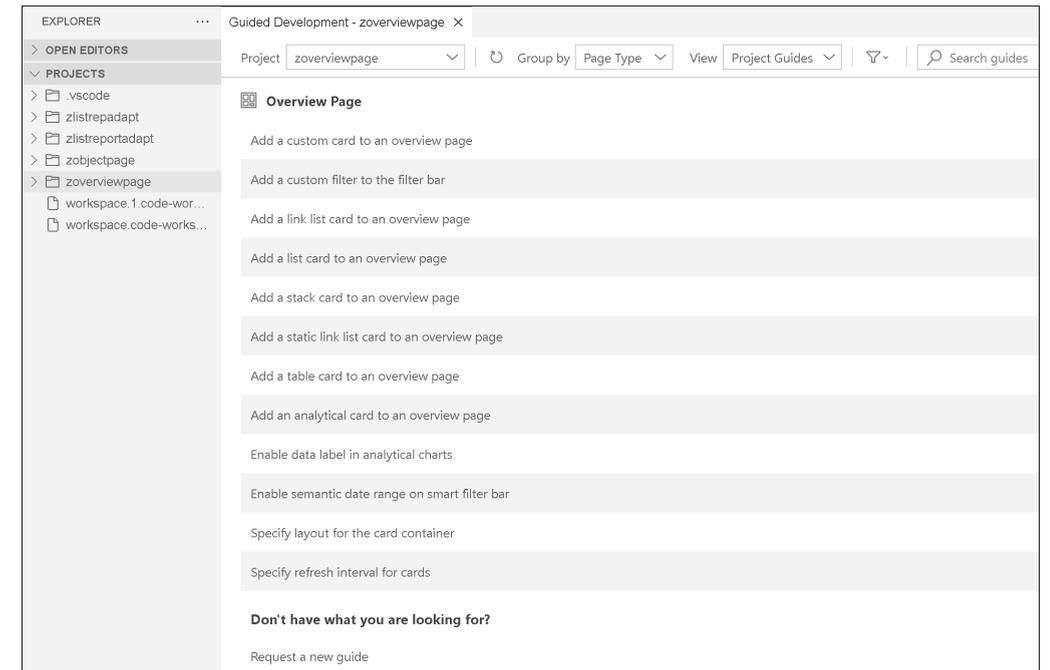


Abbildung 7.12 Guided Development

3. Sie gelangen zum ersten Schritt eines Assistenten (siehe Abbildung 7.13).

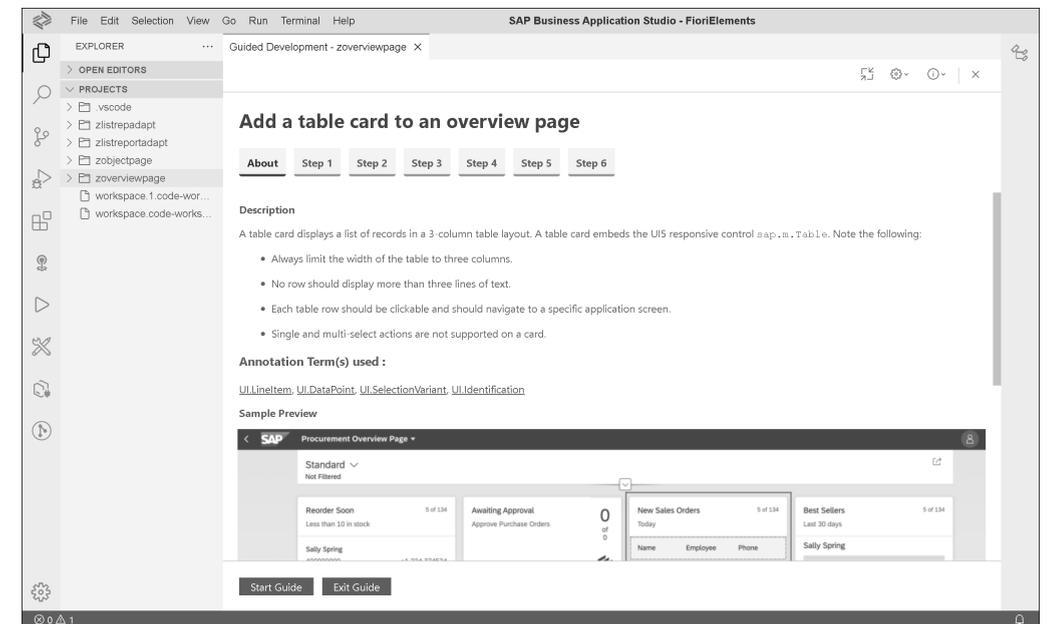


Abbildung 7.13 Table Card anlegen

Die Anleitungen gehen grundsätzlich davon aus, dass noch keine externen Annotationen gepflegt wurden und führt Sie daher sowohl durch die Anlage der nötigen Annotationen als auch durch die der Metadatendefinitionen. Da wir die nötigen Annotationen für die Table Card bereits in unserem CDS View definiert haben (siehe Abschnitt 7.3, »CDS View für eine Overview Page«), müssen wir lediglich die Schritte 5 und 6 erledigen. In diesen Schritten registrieren Sie die Cards der Overview Page in der Datei **manifest.json**.

4. Hier müssen Sie folgende Eingaben tätigen (siehe Abbildung 7.14):
  - **Model ❶**: Wählen Sie hier das OData-Model als Datenquelle aus. Das OData-Model `mainService` stammt von unserem in der Projekterstellung hinzugefügten OData-Service.
  - **Entity Set ❷**: Wählen Sie hier die Entitätsmenge aus dieser Datenquelle aus, die auf der Card angezeigt werden soll, in unserem Beispiel der in Listing 7.5 definierte CDS View `ZOVP_AUTHORS`.
  - **Card ID ❸**: Vergeben Sie hier eine eindeutige Identifikation für die Table Card.

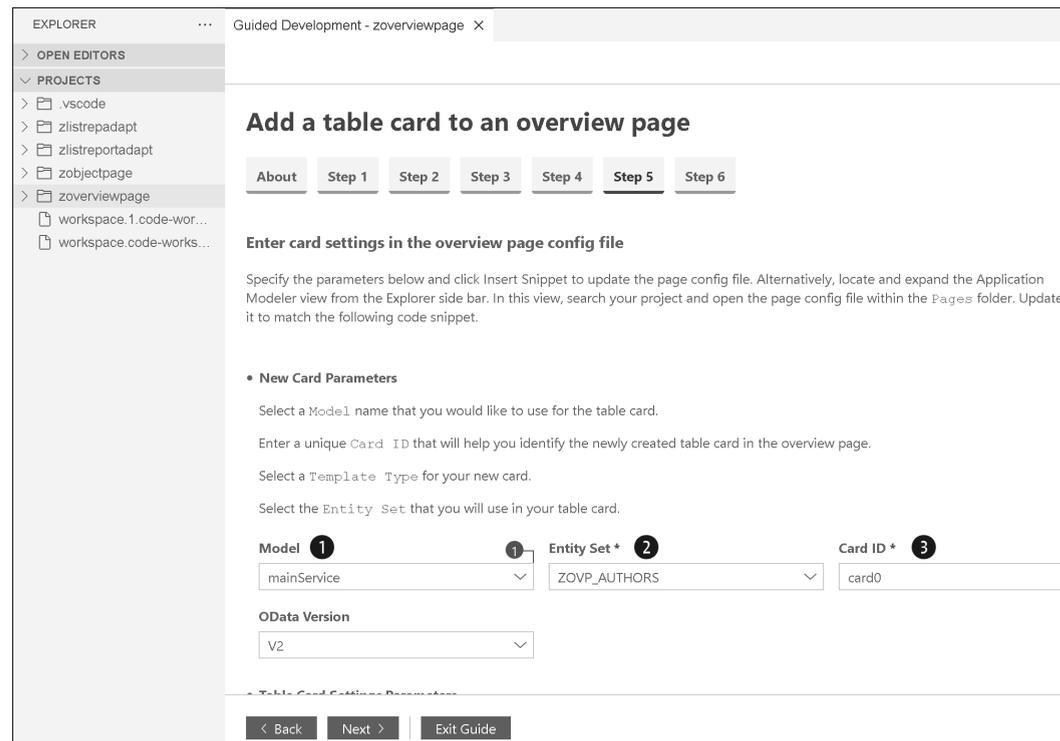


Abbildung 7.14 Guided Development zur Definition einer Table Card – Schritt 5 (Teil 1)

5. Scrollen Sie weiter nach unten. Hier sind folgende weitere Eingaben nötig (siehe Abbildung 7.15):
  - **Title ❹**: Wählen Sie Überschrift, die auf der Card angezeigt werden soll.
  - **Entity Type ❺**: Wählen Sie den Entitätstyp, in unserem Beispiel müssen wir den Entitätstyp `ZAUTHOR_CVType` auswählen, der für die Autoren angelegt worden ist.
6. Klicken Sie dann **Insert Snippet ❻**, woraufhin automatisch der Code zur Metadatendefinition generiert wird.
7. Klicken Sie anschließend auf **Next ❼**, um mit dem nächsten Schritt fortzufahren.

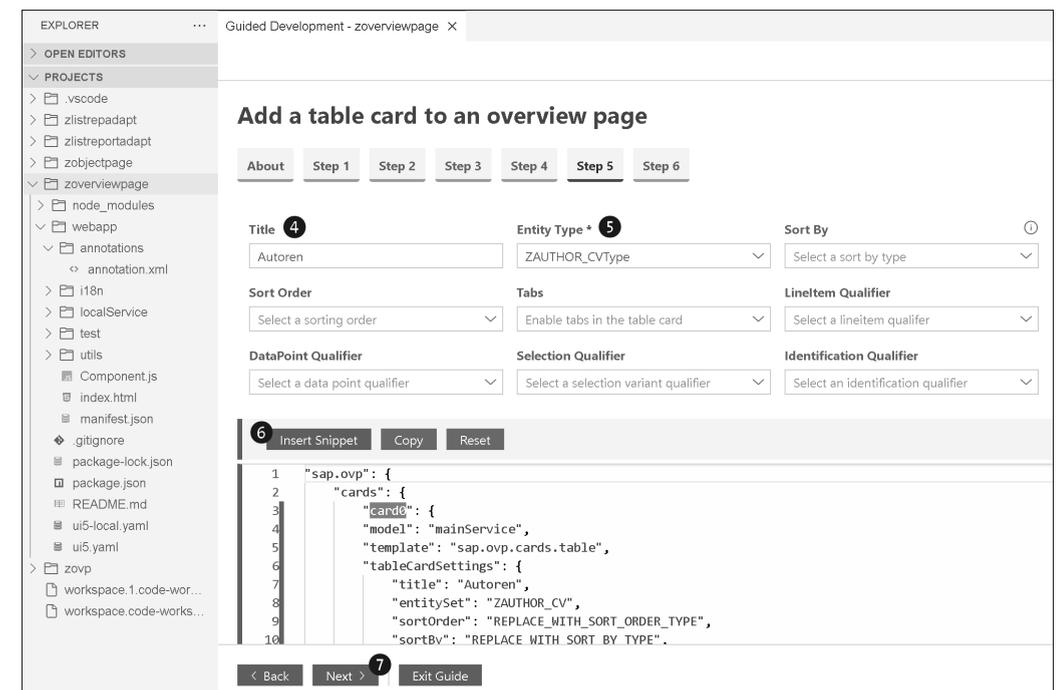


Abbildung 7.15 Guided Development zur Definition einer Table Card – Schritt 5 (Teil 2)

8. Im letzten Schritt dieser Anleitung müssen Sie folgende Eingaben ausfüllen (siehe Abbildung 7.16):
  - **Model ❶**: Wählen Sie hier das OData-Model als Datenquelle aus. Das OData-Model `mainService` stammt von unserem in der Projekterstellung hinzugefügten OData-Service.
  - **Card ID ❷**: Vergeben Sie hier eine eindeutige Identifikation für die Table Card.

- **Entity Set** ③: Wählen Sie hier die Entitätsmenge aus dieser Datenquelle aus, die auf der Card angezeigt werden soll, in unserem Beispiel der in Listing 7.5 definierte CDS View ZOVV\_AUTHORS.
9. Klicken Sie dann **Insert Snippet** ④, woraufhin automatisch der Code zur Metadatendefinition generiert wird.
  10. Klicken Sie nun auf **Exit Guide** ⑤, um diesen Dialog zu verlassen.

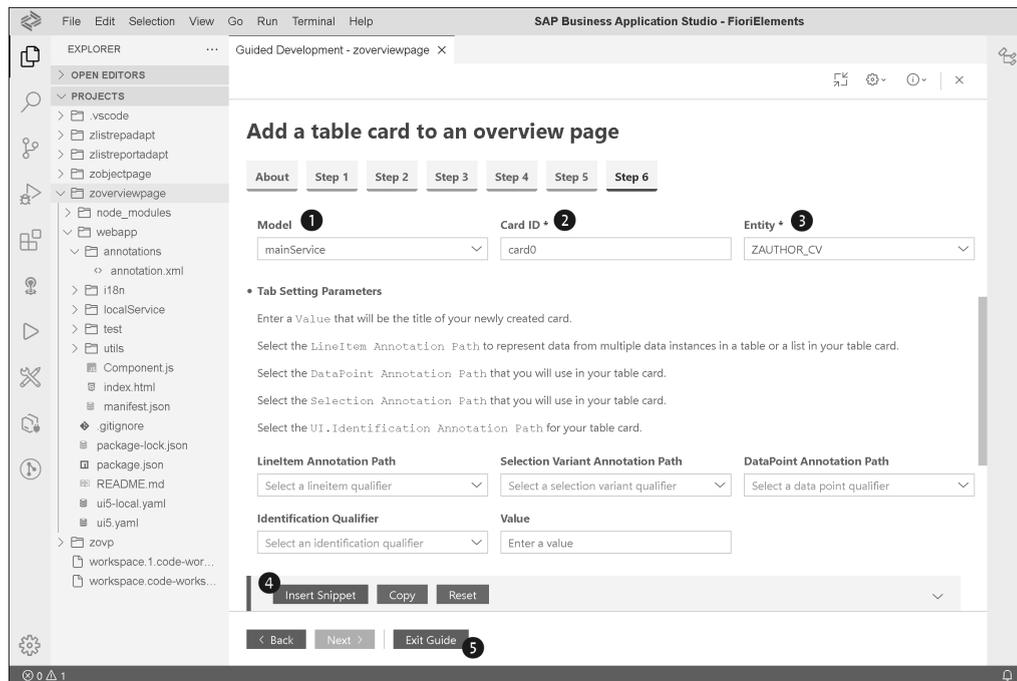


Abbildung 7.16 Guided Development zur Definition einer Table Card – Schritt 6

Als Ergebnis dieser Schritte wird die Definition der Table Card, die Sie in Listing 7.5 sehen, in die Datei **manifest.json** eingefügt. Sie müssen sie also nicht manuell vornehmen.

```
"sap.ovp": {
  "cards": {
    "card0": {
      "model": "mainService",
      "template": "sap.ovp.cards.table",
      "settings": {
        "title": "Autoren",
        "entitySet": "ZOVV_AUTHORS"
      }
    }
  }
}
```

```
}
}
}
```

Listing 7.5 Definition der Table Card in der Datei »manifest.json«

Damit haben wir bereits alles, was für die Anzeige einer Table Card auf unserer Overview Page notwendig ist. Über das Kontextmenü Ihres Projekts können Sie nun eine Vorschau aufrufen. Wählen Sie dazu den Eintrag **Preview Application**. Die Vorschau sehen Sie in Abbildung 7.17. Wir erhalten eine Overview Page mit einem Dynamic Page Header und einer ersten Card, die Autorennamen tabellarisch ausgibt.

Vorschau der Overview Page

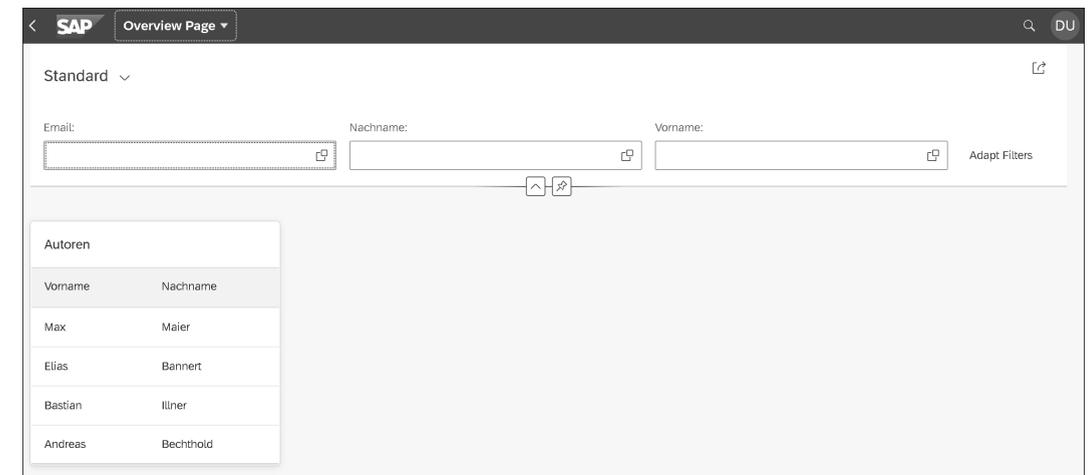


Abbildung 7.17 Vorschau auf die Overview Page und die erste Table Card

### 7.4.3 Analytical Card hinzufügen

Im nächsten Schritt möchten wir eine Auswertung aller lieferbaren Bücher dieser Autoren in einer weiteren Card hinzufügen. Dazu möchten wir ein Diagramm auf der Card anzeigen, das die Bestände der einzelnen Bücher zeigt, wozu wir eine Analytical Card anlegen. Da wir in dem CDS View bereits eine entsprechende Assoziation definiert haben, haben wir die Möglichkeit, direkt auf die Bestandsdaten der Bücher zuzugreifen und diese Entitätsmenge für die Auswertung zu verwenden. Oft haben Sie jedoch nicht die Möglichkeit, alle Daten aus einem einzigen OData-Service zu lesen. Deswegen zeigen wir Ihnen im Folgenden, wie Sie eine weitere Datenquelle, d. h. einen weiteren OData-Service, in Ihr Projekt einbinden können.

Card zur Anzeige der Buchbestände

### Weiteren OData-Service einbinden

Gehen Sie dazu wie folgt vor:

1. Wählen Sie im Menü **View • Find Command...**, und suchen Sie nach dem Befehl **Consume SAP Services** (siehe Abbildung 7.18).

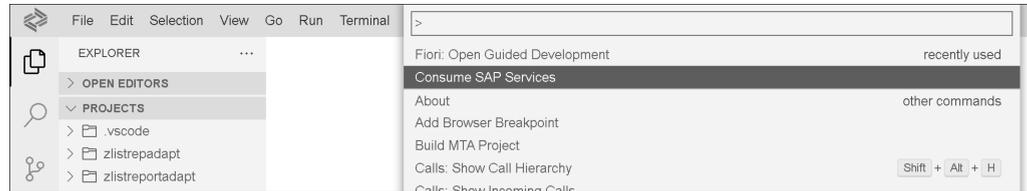


Abbildung 7.18 Neue Datenquelle einbinden

2. Sie werden gefragt, welchem Projekt der neue OData-Service hinzugefügt werden soll. Geben Sie den Projektennamen an, in unserem Beispiel **zoverviewpage** (siehe Abbildung 7.19).

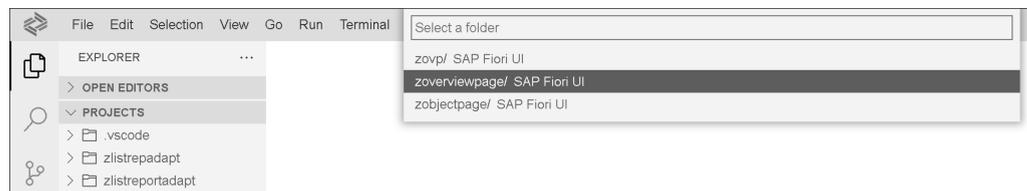


Abbildung 7.19 Applikation auswählen

3. Im nächsten Schritt müssen Sie das Quellsystem angeben, aus dem Ihr OData-Service stammt. Haben Sie an Ihrer SAP Business Technology Platform (SAP BTP) ein SAP-System angebunden, wählen Sie **My SAP systems**, um zur Auswahl der SAP-Systeme zu gelangen (siehe Abbildung 7.20).

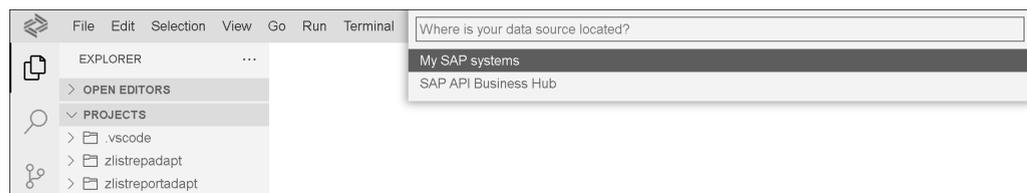


Abbildung 7.20 Quellsystem auswählen

4. Es werden Ihnen alle für Ihren Account der SAP BTP verfügbaren Destinationen angeboten (siehe Abbildung 7.21).
5. Im letzten Schritt wählen Sie den OData-Service aus (siehe Abbildung 7.22).

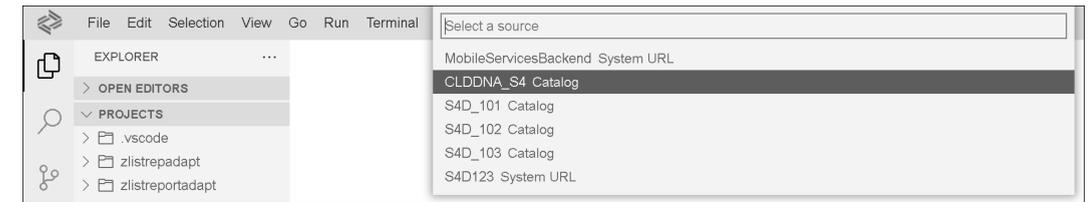


Abbildung 7.21 Destination des Quellsystems auswählen

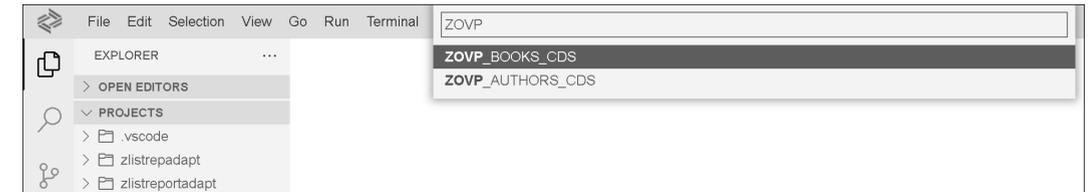


Abbildung 7.22 OData-Service auswählen

Im Architekturmuster Model View Controller, das allen SAPUI5- und damit auch allen SAP-Fiori-Elements-Applikationen zugrunde liegt, ist definiert, dass Daten, die in der View-Schicht dargestellt werden sollen, aus einem Model kommen müssen. Daher müssen wir noch ein OData-Model anlegen, über das wir mit unserem OData-Service kommunizieren können. Das OData-Model definieren Sie in der Datei **manifest.json** beim Attribut "models". Diese Definition sehen Sie in Listing 7.6.

**OData-Model anlegen**

```
"models": {
  "books": {
    "dataSource": "ZOVP_BOOKS_CDS",
    "preload": true,
    "settings": {
      "defaultBindingMode": "TwoWay",
      "defaultCountMode": "Inline",
      "refreshAfterChange": false
    }
  }
}
```

Listing 7.6 OData-Model-Definition

Die zweite Card, die wir anlegen, soll eine Analytical Card sein. Im Rahmen des Guided Developments können Sie dazu die Anleitung **Add an analytical card to an overview page** auswählen. Das Vorgehen ist sehr ähnlich wie bei der Anlage der Table Card. Da wir einige Eigenschaften der Analytical Card

**Analytical Card definieren**

über die Annotationen im CDS View pflegen, müssen Sie nicht alle Schritte des Assistenten durchgehen, sondern können direkt zu Schritt 5 navigieren. Machen Sie hier die folgenden Angaben für die Analytical Card (siehe Abbildung 7.23). Auf die Annotationen im CDS View kommen wir im Anschluss noch zurück:

- ❶ **Model:** Wählen Sie das soeben definierte OData-Model `books` aus.
- ❷ **Card ID:** Vergeben Sie eine eindeutige ID für die Card.
- ❸ **OData Version:** Wählen Sie die OData-Version aus, in unserem Beispiel `V2`.
- ❹ **Entity Set:** Wählen Sie die Entitätsmenge als Datenquelle aus.
- ❺ **Title:** Vergeben Sie eine Überschrift für die Card.

Mit einem Klick auf **Insert Snippet** ❻ wird der Code zur Definition der Card generiert. Mit einem Klick auf **Exit Guide** ❼ können Sie die Anleitung abschließen.

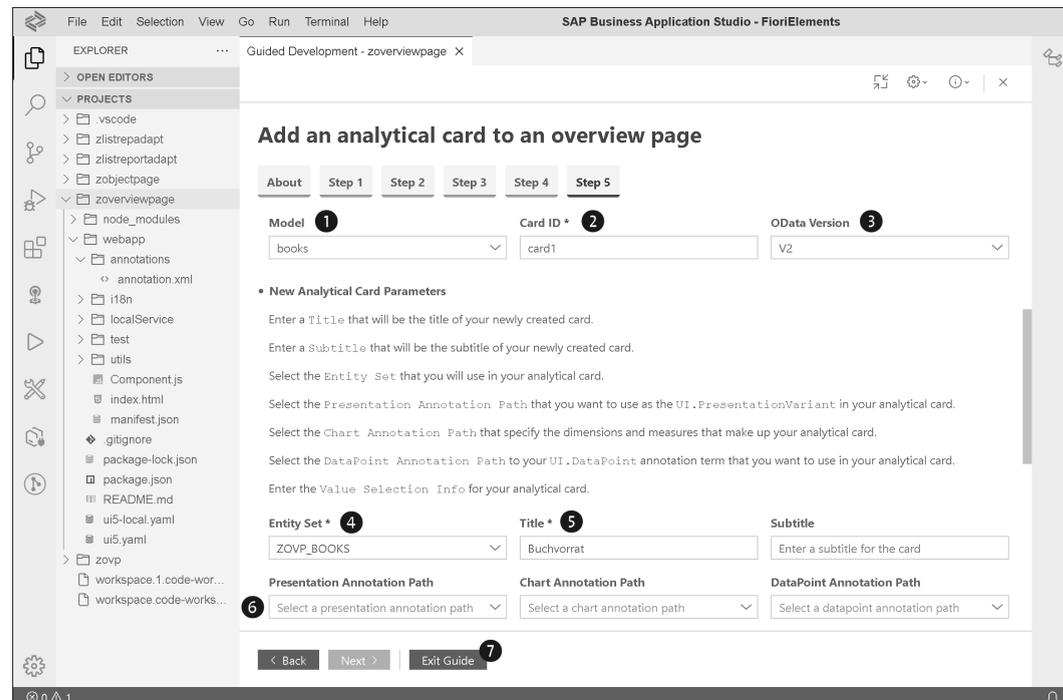


Abbildung 7.23 Guided Development zur Definition einer Analytical Card – Schritt 5

Im Ergebnis wird die Definition der Analytical Card in die Datei `manifest.json` eingefügt. Diese Definition sehen Sie in Listing 7.7.

```
"card1": {
  "model": "mainService",
  "template": "sap.ovp.cards.charts.analytical",
  "settings": {
    "title": "Buchvorrat",
    "entitySet": "ZOV_P_BOOKS",
    "chartAnnotationPath": "com.sap.vocabularies.UI.v1.Chart"
  }
}
```

Listing 7.7 Definition der Analytical Card in der Datei »manifest.json«

Während wir die Eigenschaften der Card über den Assistenten definiert haben, wollen wir das Aussehen des Diagramms der Buchbestände über UI-Annotationen in dem CDS View definieren (siehe Listing 7.8). Dazu nutzen wir die Annotation `@UI.chart`, über die wir den Diagrammtyp und die Werte der x- und y-Achse bestimmen. Mit dem Wert `#COLUMN` für das Attribut `chartType` geben wir an, dass es sich um ein Balkendiagramm handeln soll. Auf der x-Achse sollten die Buchtitel angezeigt werden (Attribut `dimensions`), auf der y-Achse die Anzahl der verfügbaren Exemplare (Attribut `measures`).

```
@UI.chart: [{
  title: 'Buchvorrat',
  chartType: #COLUMN,
  dimensions: ['Title'],
  measures: ['Stock'],
  dimensionAttributes: [{
    dimension: 'Title',
    role: #CATEGORY
  }],
  measureAttributes: [{
    measure: 'Stock',
    role: #AXIS_1
  }]
}]
```

Listing 7.8 Externe Annotation zur Definition des Diagramms für den Buchvorrat

Nun können wir uns das Ergebnis in der Vorschau ansehen. Starten Sie dazu die Applikation über den Kontextmenüeintrag **Preview Application**, oder laden Sie die bereits gestartete Applikation neu. Sie sehen, dass der Overview Page eine weitere Card hinzugefügt wurde (siehe Abbildung 7.24).

Definition eines Diagramms

Vorschau der Overview Page

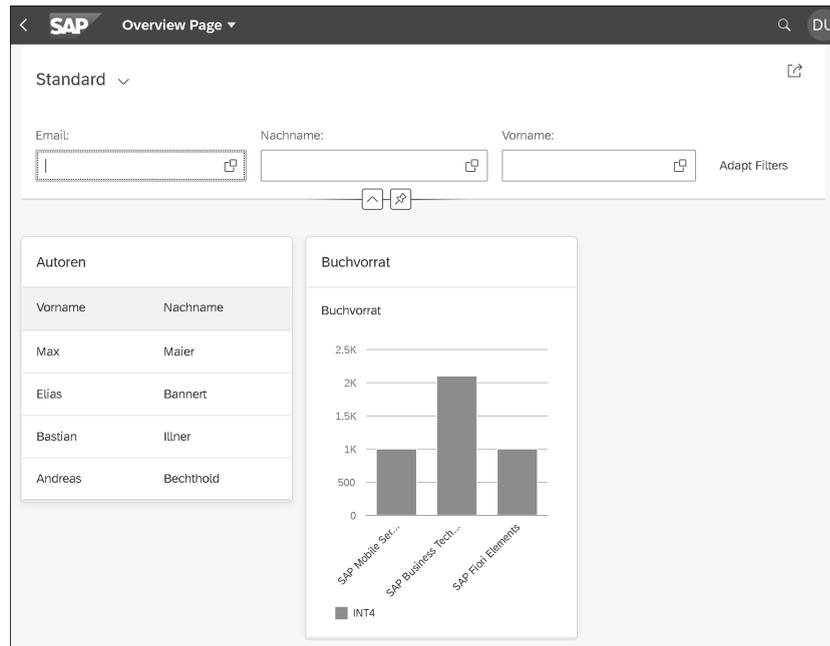


Abbildung 7.24 Vorschau der Overview Page mit der Analytical Card

#### 7.4.4 List Card hinzufügen

**List Card** Im nächsten Schritt fügen wir der Overview Page noch eine List Card hinzu, auf der wir die verfügbaren Bücher jeweils mit Preis und ISBN in einer Liste ausgeben möchten. Alle Vorbereitungen, die dazu nötig sind, d. h. die Anbindung der Datenquelle und die Definition der externen Annotationen, haben wir bereits erledigt. Sie brauchen also lediglich die Anleitung **Add a list card to an overview page** auszuwählen und können dann direkt mit Schritt 5 des Guided Developments fortfahren (siehe Abbildung 7.25). Wählen Sie hier das OData-Model **1** aus, das Sie für die Analytical Card angelegt hatten. Wählen Sie ebenfalls dieselbe Entitätsmenge **2** aus, und vergeben Sie eine eindeutige Identifikation **3** für die neue Card.

Im unteren Teil dieser Ansicht vergeben Sie eine Überschrift **4** für die Card, wählen den Entitätstyp **5** aus und klicken anschließend auf **Insert Snippet** **6**. Fahren Sie dann mit **Next** **7** fort (siehe Abbildung 7.26).

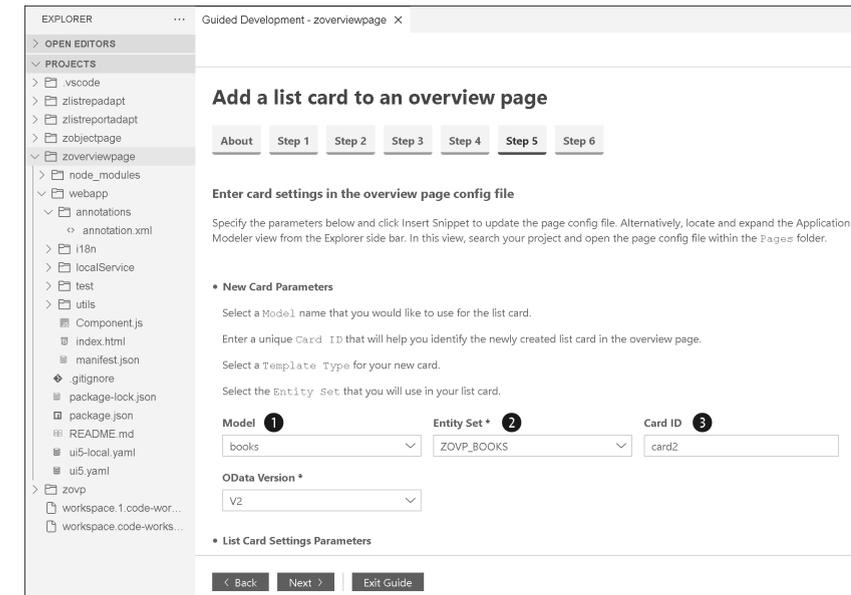


Abbildung 7.25 Guided Development zur Definition einer List Card – Schritt 5 (Teil 1)

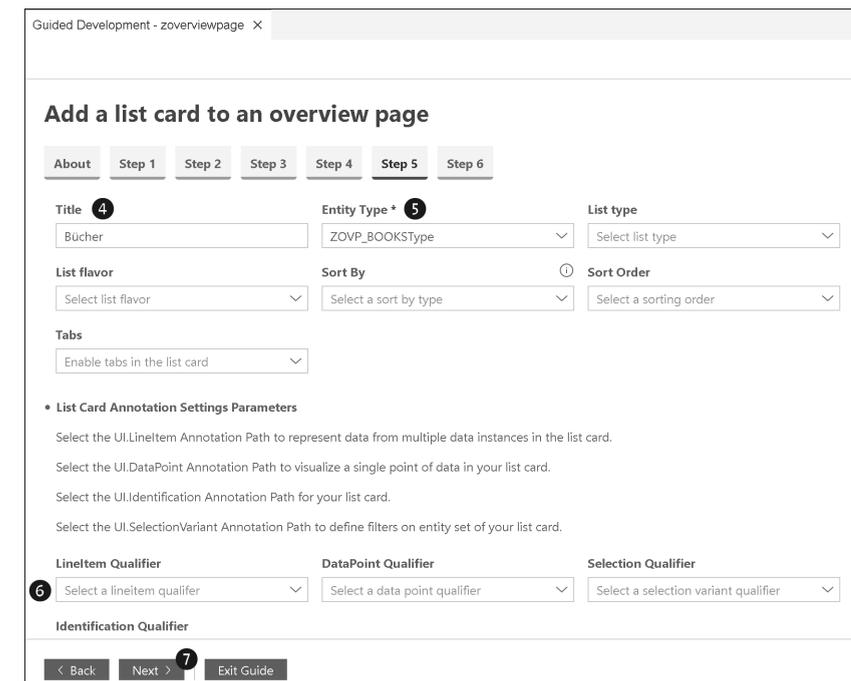


Abbildung 7.26 Guided Development zur Definition einer List Card – Schritt 5 (Teil 2)

Mit dem sechsten Schritt kann dieser geführte Dialog abgeschlossen werden. Wählen Sie das OData-Model **1** aus, geben Sie die ID der Card **2** und die Entitätsmenge **3** an, die Sie im vorherigen Schritt definiert hatten, und lassen Sie über den Button **Insert Snippet** **4** die Definition der Card generieren. Mit **Exit Guide** **5** können Sie den geführten Dialog verlassen (siehe Abbildung 7.27).

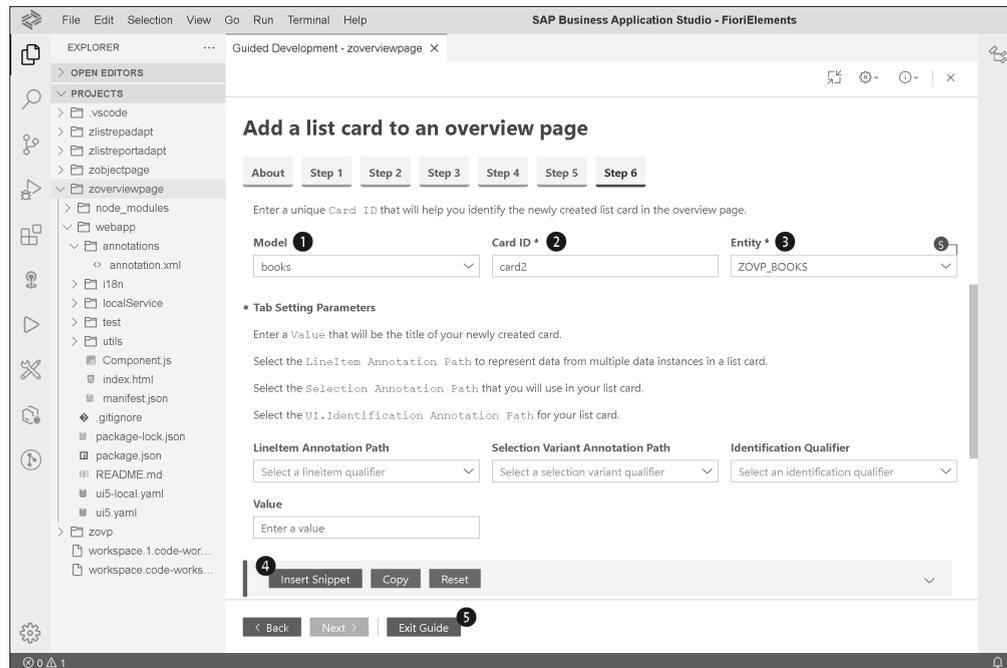


Abbildung 7.27 Guided Development zur Definition einer List Card – Schritt 6

Als Ergebnis wird die Definition aus Listing 7.9 in die Datei **manifest.json** eingefügt.

```
"card2": {
  "model": "mainService",
  "template": "sap.ovp.cards.list",
  "settings": {
    "title": "Bücher",
    "entitySet": "ZOVP_BOOKS"
  }
}
```

Listing 7.9 Definition der List Card in der Datei »manifest.json«

Das Ergebnis ist eine listenähnliche Darstellung der Bücher auf einer dritten Card (siehe Abbildung 7.28).

Vorschau der Overview Page

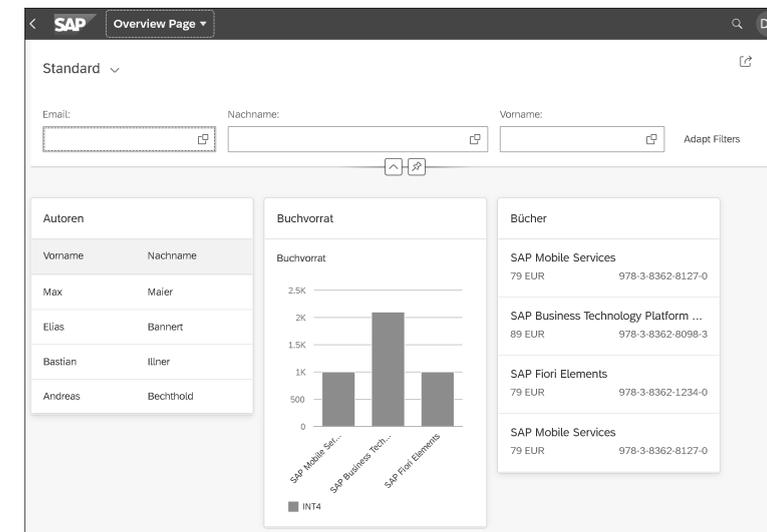


Abbildung 7.28 Vorschau der Overview Page mit der List Card

#### 7.4.5 Stack Card hinzufügen

Wir runden unsere Overview Page mit einer Stack Card ab. Diese soll eine Sammlung mehrerer Karten beinhalten und so eine Zusammenfassung bestimmter Entitäten darstellen. Verwenden Sie dazu die Anleitung **Add a stack card to an overview page**. In Schritt 5 des Assistenten legen Sie die Definition der Card an, indem Sie, wie schon gewohnt, unser OData-Model **1** und unsere Entitätsmenge **2** auswählen, eine ID **3** für die Card vergeben und eine Überschrift **4** definieren. Mit dem Klick auf **Insert Snippet** wird die Definition der Card generiert. Mit **Exit Guide** **5** verlassen Sie den Dialog.

Definition der Card

Im Ergebnis wird die Definition der Stack Card in die Datei **manifest.json** eingefügt (siehe Listing 7.10).

```
"card3": {
  "model": "books",
  "template": "sap.ovp.cards.stack",
  "settings": {
    "title": "Alle Bücher",
    "entitySet": "ZOVP_BOOKS"
  }
}
```

Listing 7.10 Stack-Card-Definition

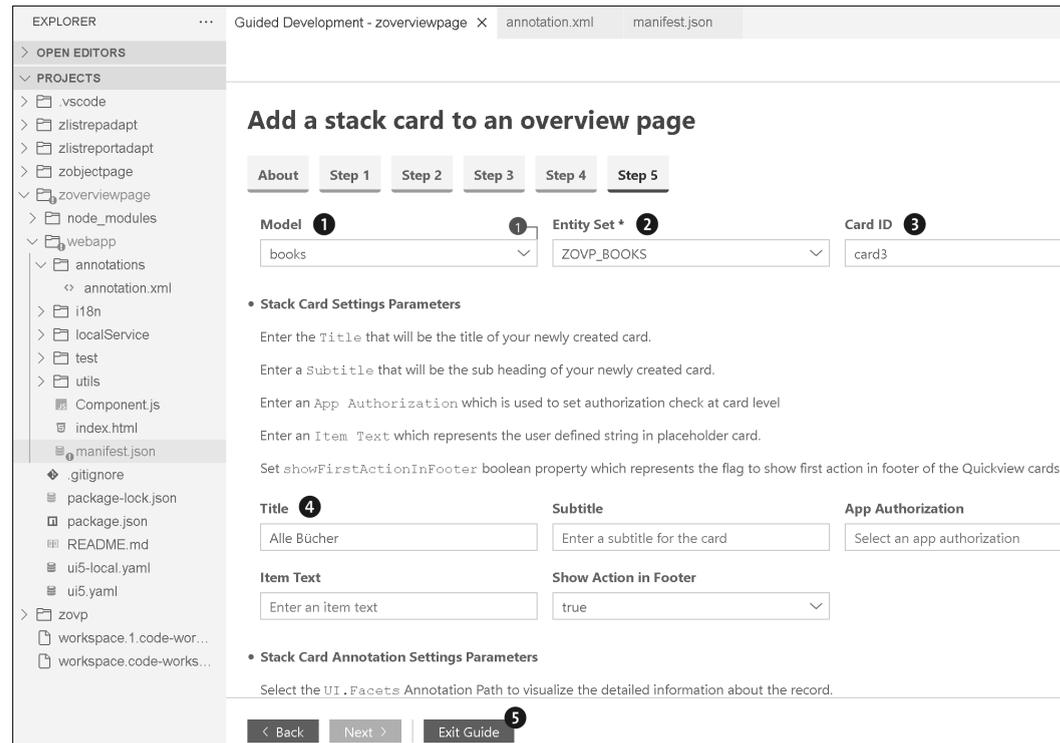


Abbildung 7.29 Guided Development zur Definition einer Stack Card – Schritt 5

### UI.facet-Annotation

In Abschnitt 7.2, »Annotationen für Overview Pages«, haben wir erklärt, dass die `UI.facet`-Annotationen dafür zuständig sind, formularähnliche Darstellungen von Informationen aufzubauen. Da wir mehrere Bücher im Detail in dieser Stack Card anzeigen möchten, fügen wir unserem CDS View `ZOVP_BOOKS` für die Buchdaten eine solche Annotation hinzu (siehe Listing 7.11).

```
@UI.facet: [{
  targetQualifier: 'BUCH',
  type: #FIELDGROUP_REFERENCE,
  isSummary: true
}]
```

Listing 7.11 UI.Facet-Annotation im CDS View

`UI.Facet` gehört zu einer Reihe von Annotationen, die mehrere weitere Annotationen beinhalten können. Zwischen diesen Annotationen muss eine Beziehung hergestellt werden. Dazu greift man oft auf sogenannte

`Qualifier-Attribute` zurück. Diese Qualifier können Sie sich wie IDs vorstellen, mit denen man auf die Annotation, die man definiert hat, zugreifen und darauf verweisen kann. Unter dem `targetQualifier` namens 'Buch' fassen wir eine Gruppe von Feldern zusammen, die auf der formularähnlichen Darstellung angezeigt werden sollen. Wir wollen in der formularähnlichen Darstellung unter anderem die Preise der Bücher anzeigen. Dazu verweisen wir mit der Annotation `@UI.fieldGroup` auf die Feldgruppe mit der ID 'BUCH':

```
@UI.fieldGroup: [{ position:10, qualifier: 'BUCH'}]
price as Price,
```

Wiederholen Sie diesen Schritt für alle Attribute, die dieser Feldgruppe zugeordnet werden sollen. Welche Darstellungen mit den Annotationen `UI.Facet` und `UI.FieldGroups` noch möglich sind, werden Sie in Kapitel 10, »Object Page«, sehen.

Zunächst einmal zurück zu unserer Stack Card: Wie diese auf der Overview Page ausgegeben wird, können Sie sich über die Vorschaufunktion ansehen (siehe Abbildung 7.30).

Vorschau der  
Stack Card

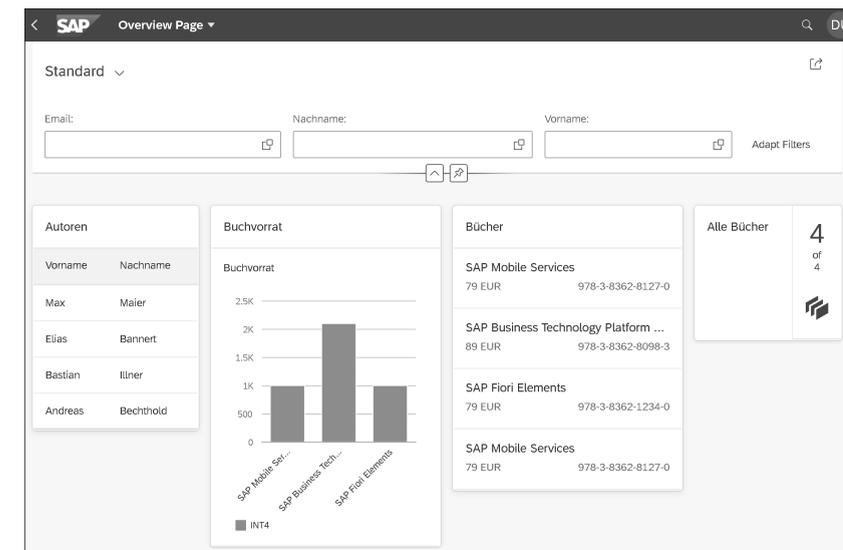


Abbildung 7.30 Vorschau der Overview Page mit der Stack Card

In diesem Abschnitt haben Sie die wichtigsten Annotationen für die Overview Page kennengelernt und auch einen ersten Einblick bekommen, wie man mit dem Guided Development im SAP Business Application Studio arbeitet. Einige der hier genutzten Annotationen werden Ihnen in den Folge-

kapiteln zu den weiteren Floorplans wieder begegnen. Auch werden wir weiterhin mit dem Guided Development arbeiten, zu dem Sie in Kapitel 12, »Erweiterungen«, noch ausführlichere Informationen finden. Dort werden wir nochmals auf die hier definierte Overview Page zurückkommen und Ihnen die Erweiterungsmöglichkeiten dieses Floorplans zeigen.

## 7.5 Entwicklung einer Overview Page in der SAP Web IDE

Nachdem Sie die Entwicklung einer Overview Page im SAP Business Application Studio gesehen haben, beschäftigen wir uns in diesem Abschnitt mit der Entwicklung einer Overview Page in der SAP Web IDE. Wir werden dazu die gleiche Anwendung mit den gleichen Cards nachbauen.

### 7.5.1 Projekt anlegen

Overview Page  
anlegen

Im ersten Schritt muss auch in der SAP Web IDE ein Projekt angelegt werden:

1. Wählen Sie dazu auf der Startseite die Option **New Project from Template**.
2. Anschließend wählen Sie den Floorplan **Overview Page** aus und klicken auf **Next** (siehe Abbildung 7.31).

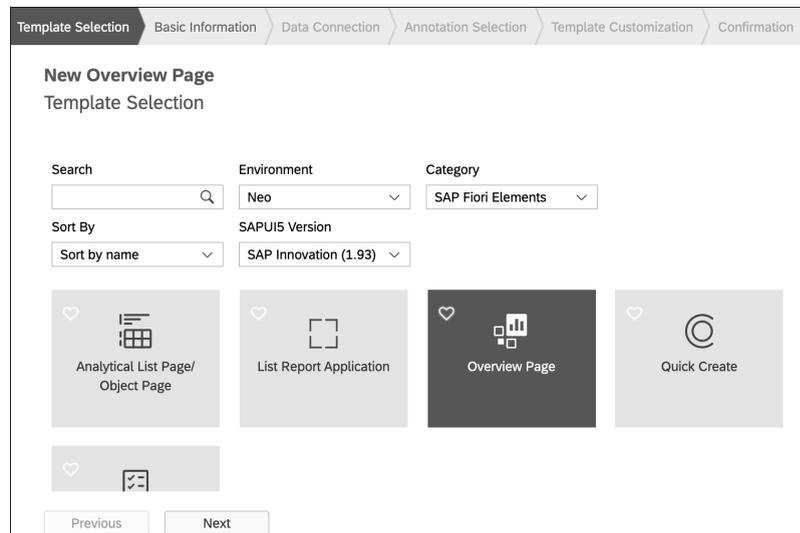


Abbildung 7.31 Floorplan in der SAP Web IDE auswählen

Diese Leseprobe haben Sie beim  
 [edv-buchversand.de](http://edv-buchversand.de) heruntergeladen.  
 Das Buch können Sie online in unserem  
 Shop bestellen.

[Hier zum Shop](#)