WEEK 3 -PRACTICE

UI Architecture - Theme, Widgets, Screens

Learning objectives

- ✓ **Structure Flutter widgets** for extendibility and consistency
- ✓ Comply with **coding conventions**
- ✓ Create a library of **generic widgets** aligned with a **design system**
- ✓ Understand the dart concepts of **static methods**, **attributes** and **part of** syntax

How to submit?

- ✓ Make sure you follow the COMMIT standards (see below)
- ✓ Once finished, submit to MS Team:
  - GitHub URL
  - This document

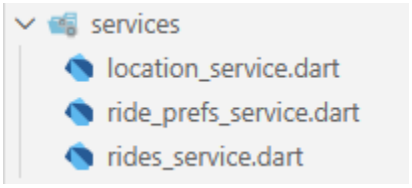
You are **not** allowed to **use AI** to submit this work

This practice focusses more **on clean code structure** rather than Flutter technical skills. We will evaluate how well **you follow the coding conventions**, how your name your class, variables etc.

BLA-001 – Rides Service

Complete the **service layer** and **test it**

The service layer – so far – is composed of 3 services:



All services are static - for now - and access to fake data (dummy data).

✏️ Complete the rides service to filter the rides starting from given departure location

```
List<Ride> filterByDeparture(Location departure)
```

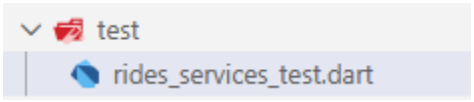
✏️ Complete the rides service to filter the rides for the given requested seat number

```
List<Ride> filterBySeatRequested(int seatRequested)
```

✏️ Complete the rides service to filter with several optional criteria *(flexible filter options)*

```
List<Ride> filterBy({Location? departure, int? seatRequested})
```

We use the /test folder to test our service and other layers :



✏️ Test your rides service filter methods using mock data: create a main() in the test folder :

```
RidesService.filterBy(
      departure: Location(name: "Dijon", country: Country.france),
      seatsRequested:2
);    // Shall return 1 ride
```

Example of test

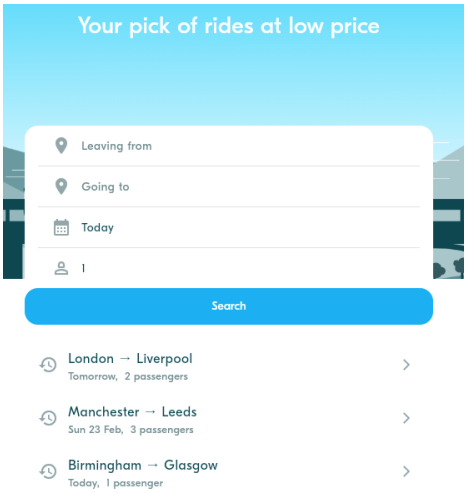
TODAY…

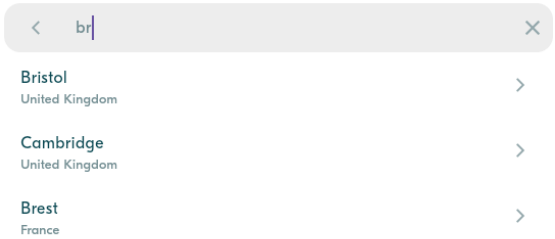
For this practice, you need to implement the **Rides Preferences** screen

- To input the locations, date and seats and press on Search
- Or to click on a past entered **Ride Preference**

You also need to implement the related input modals:

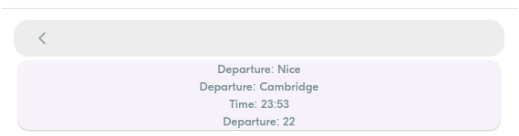
- Location Picker
- Seat picker
- Date chooser





–

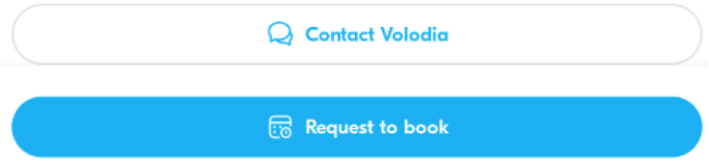
- Once selected, we should navigate to the **Rides screen**, displaying all rides matching preferences (fake view for now):



The Rides screen is just a fake screen for now

BLA-002 – Implement BlaButton

The BlaButton is used in many places in the application.



The BlaButton widget shall handle primary and secondary, without or with icons…

Q1 – First identify the **possible variations** of this button and complete the table

Widget	Screen / Screen Widget /App Widget	Parameters	Callbacks
BlaButton			

Q2 – Then implement the button and **test** all its variations using a **test screen**.

Q3 – Once validated, commit the code with the proper commit message.

```
BLA-003 - Implement BlaButton
```

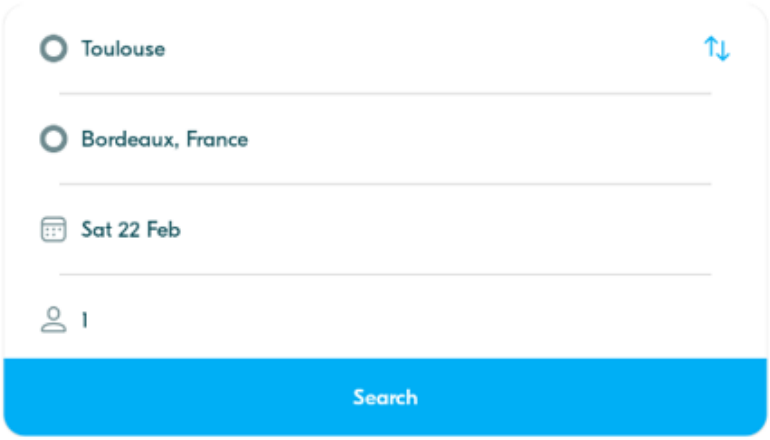
BLA-003 – Implement Form

The ride preferences from can be used either in the **Ride Preferences** screen (*as a screen component)* or in the **Ride screen** (*as a top modal dialog*)

⚠ Its s important to ensure this component **can be used on both screens**

⚠ You need to code the **component exactly as in the real app**.

What are the possible colors in the input fields ? The mandatory and optional icons? Actions?



Q1 – What are the widget parameters? What are the default values?

Q2 – What is the condition to validate the Search button? What type of data is popped if valid?

Q3 – What are the **form sub widgets** and **app widget** you plan to use for this form?

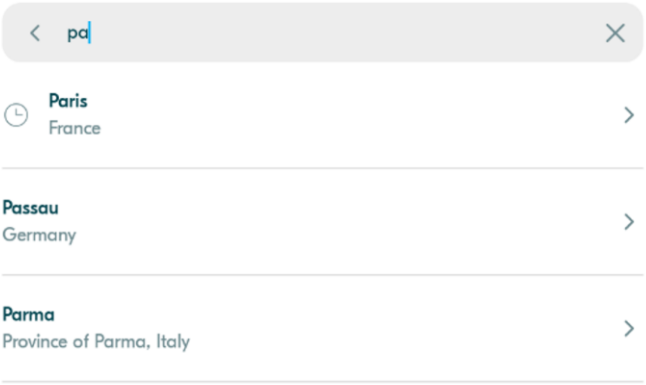
Widget	Screen / Screen Widget /App Widget	Parameters	Callbacks
BlaButton	App Widget		

Q5 – How to implement the **switch location action** () in a clean way?

Q6 – Implement the widget and **test** all use cases.

BLA-004 – Implement the Location Picker

The location picker will be used in many views (as a passenger as well as a driver).



Q1 – Analyze the real app picker behavior

How many actions can be done? When the list of locations is displayed? Etc.

Q2 – What are the **picker sub widgets** and **app widget** you plan to use for this form?

Widget	Screen / Screen Widget /App Widget	Parameters	Callbacks

Q3 – Implement the widget and **test** all use cases.

BLA-005 - BONUS - Add **a Tween animation** to show the Location Picker

The Location Picker shall be displayed with a bottom to top transition (slide).

Q1 – Learn how to use Tween and offset for animated transitions.

Q2 – Complete the code on animations_util.dart, to create a bottom to top transition.

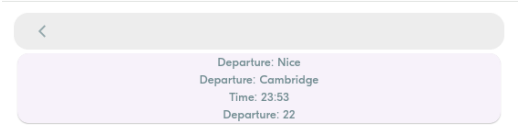
```
static Route<T> createBottomToTopRoute<T>(Widget screen) {
  const begin = Offset(0.0, 0.0);          //  TODO Change this line
  const end = Offset(0.0, 0.0);            //  TODO Change this line
  return _createAnimatedRoute(screen, begin, end);
}
```

Q3 – Use this route to **navigate** to the **Location Picker** form the **RidePreferenceForm**.

BLA-006 – BONUS - Implement a draft **Rides screen**

Once ride preferences have been selected, the Rides Screen shall be displayed with the matching rides.

For now, we just want to know if it works…



Not the final look, but just to test the preferences

Q1 – Widgets

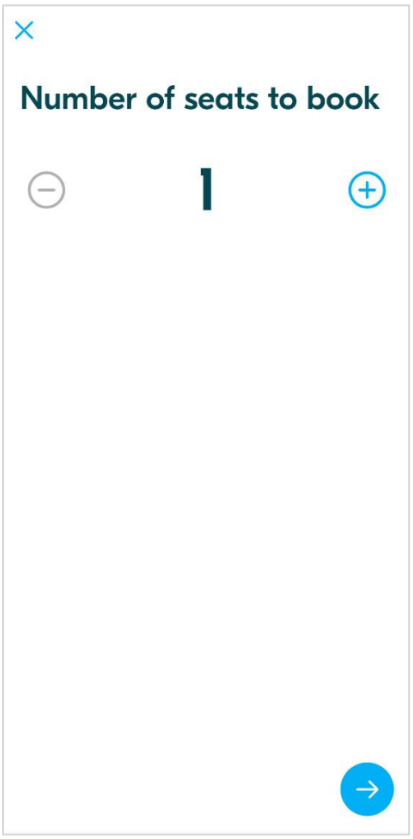
Analyze the Ride screen and complete the table with your widget strategy

Widget	Screen / Screen Widget /App Widget	Parameters	Callbacks

Q2 – Implement the widget and **test** all use cases.

BLA-007 – BONUS - Implement the Seat number spinner

Following the same methodology, you can as a bonus implement the seat number spinner.
It shall render and behave exactly as in the real app.



BLA-008 – BONUS - Implement the Date picker

Following the same methodology, you can as a bonus implement the date picker.
It should be rendered and behave exactly as in the real app.