Name : Vahora Mohammad Sauban

Enroll : 202101619010192 Flutter

Assignment -2

Q1

```
import 'package:flutter/material.dart';

// Task Class class
Task {   String
taskId;
  String title;
  String description;
  String dueDate;

  Task({     required this.taskId,
required this.title,     required
this.description,     required
this.dueDate,
  });
}

// Main Function void main() {
runApp(TaskManagementApp());
}

class TaskManagementApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(   title: 'Task Management',
    theme: ThemeData(primarySwatch: Colors.blue),
    home: TaskFormPage(),
```

```
    );
  }
}

// Page 1: TaskFormPage class TaskFormPage
extends StatefulWidget {
  @override
  _TaskFormPageState createState() => _TaskFormPageState();
}

class _TaskFormPageState extends State<TaskFormPage> {   final
_formKey = GlobalKey<FormState>();   final _taskIdController =
TextEditingController();   final _titleController =
TextEditingController();   final _descriptionController =
TextEditingController();   final _dueDateController =
TextEditingController();

  final List<Task> _taskList = [];

  // Add task function   void _addTask() {
if (_formKey.currentState!.validate()) {
      final newTask = Task(
      taskId: _taskIdController.text, title:
      _titleController.text, description:
      _descriptionController.text, dueDate:
      _dueDateController.text,
    );
    setState(() {
      _taskList.add(newTask);
    });
```

```
    // Clear input fields
    _taskIdController.clear();
    _titleController.clear();
    _descriptionController.clear();
    _dueDateController.clear();

    // Navigate to TaskListPage
Navigator.push(        context,
      MaterialPageRoute(            builder: (context) =>
TaskListPage(taskList: _taskList)),
    );
   }
 }

 @override
 Widget build(BuildContext context) {
return Scaffold(       appBar: AppBar(title:
Text('Add Task')),       body: Padding(
padding: const EdgeInsets.all(16.0),
              child: Form(
  key: _formKey,
        child: Column(     crossAxisAlignment:
      CrossAxisAlignment.start,     children: <Widget>[

          TextFormField(          controller: _taskIdController,
    decoration: InputDecoration(labelText: 'Task ID'),
    validator: (value) {
              if (value == null || value.isEmpty) {
                return 'Please enter Task ID';
              }
              return null;
```

```
            },
          ),
          TextFormField(              controller: _titleController,
decoration: InputDecoration(labelText: 'Task Title'),
validator: (value) {              if (value == null || value.isEmpty)
{                 return 'Please enter Task Title';
              }
              return null;
            },
          ),
          TextFormField(              controller: _descriptionController,
decoration: InputDecoration(labelText: 'Description'),
validator: (value) {              if (value == null || value.isEmpty) {
return 'Please enter Task Description';
              }
    return null;
            },
          ),
          TextFormField(
           controller: _dueDateController,         decoration:
    InputDecoration(labelText: 'Due Date (YYYY-MM-DD)'),         validator:
    (value) {          if (value == null || value.isEmpty) {
              return 'Please enter Due Date';
              }
              return null;
            },
          ),
          Padding(
            padding: const EdgeInsets.symmetric(vertical: 16.0),
child: ElevatedButton(              onPressed: _addTask,
child: Text('Add Task'),
```

```
            ),
          ),
        ],
      ),
    ),
  ),
 );
 }
}

// Page 2: TaskListPage class TaskListPage extends
StatelessWidget {
 final List<Task> taskList;

 TaskListPage({required this.taskList});

 @override
 Widget build(BuildContext context) {
 return Scaffold(    appBar: AppBar(title:
 Text('Task List')),    body: ListView.builder(
 itemCount: taskList.length,
     itemBuilder: (context, index) {
final task = taskList[index];
return ListTile(          title:
Text(task.title),          onTap: () {
          // Navigate to TaskManagePage to edit task details
Navigator.push(              context,
            MaterialPageRoute(              builder: (context) =>
TaskManagePage(task: task),
            ),
```

```
            );
          },
        );
      },
    ),
  );
 }
}

// Page 3: TaskManagePage (Edit Task) class
TaskManagePage extends StatefulWidget {
 final Task task;

 TaskManagePage({required this.task});
 @override
 _TaskManagePageState createState() => _TaskManagePageState();
}

class _TaskManagePageState extends State<TaskManagePage> {   late
TextEditingController _titleController;   late TextEditingController
_descriptionController;   late TextEditingController
_dueDateController;

 @override   void
initState() {
super.initState();
   _titleController = TextEditingController(text: widget.task.title);
   _descriptionController =
      TextEditingController(text: widget.task.description);
   _dueDateController = TextEditingController(text: widget.task.dueDate);
```

```
 }

 // Save the changes made to the task   void _saveTask() {
setState(() {      widget.task.title = _titleController.text;
widget.task.description = _descriptionController.text;
widget.task.dueDate = _dueDateController.text;
   });

   // Go back to TaskListPage with updated task
   Navigator.pop(context);
 }

 @override
 Widget build(BuildContext context) {
 return Scaffold(     appBar: AppBar(title:
 Text('Edit Task')),     body: Padding(
     padding: const EdgeInsets.all(16.0),        child: Column(
children: <Widget>[           TextField(            controller:
_titleController,            decoration:
InputDecoration(labelText: 'Task Title'),
         ),
         TextField(                    controller:  _descriptionController,
decoration: InputDecoration(labelText: 'Description'),
         ),
         TextField(            controller: _dueDateController,            decoration:
InputDecoration(labelText: 'Due Date (YYYY-MM-DD)'),
         ),
         Padding(
           padding: const EdgeInsets.symmetric(vertical: 16.0),
child: ElevatedButton(              onPressed: _saveTask,
child: Text('Save Changes'),
```

```
            ),
          ),
        ],
      ),
    ),
  );
 }
}
```

Q2

```
import 'package:flutter/material.dart';

// Product Class class Product {   String productId;   String productName;   double price;   int stockQuantity;

 Product({    required this.productId,    required this.productName,    required this.price,    required this.stockQuantity,
 });
}

// Main Function void main() {
runApp(ProductInventoryApp());
}
```

```
class ProductInventoryApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) { return
    MaterialApp(
      title: 'Product Inventory',
      theme: ThemeData(primarySwatch: Colors.blue), home:
      ProductFormPage(),
    );
  }
}

// Page 1: ProductFormPage class ProductFormPage
extends StatefulWidget {
  @override
  _ProductFormPageState createState() => _ProductFormPageState();
}

class _ProductFormPageState extends State<ProductFormPage> {   final
_formKey = GlobalKey<FormState>();   final _productIdController =
TextEditingController();   final _productNameController =
TextEditingController();   final _priceController = TextEditingController();
final _stockQuantityController = TextEditingController();

  final List<Product> _productList = [];

  // Add product function   void
_addProduct() {     if
(_formKey.currentState!.validate()) {
final newProduct = Product(        productId:
_productIdController.text,
```

```
      productName: _productNameController.text,   price:
    double.parse(_priceController.text),   stockQuantity:
    int.parse(_stockQuantityController.text),
    );
    setState(() {
     _productList.add(newProduct);
    });

    // Clear input fields
   _productIdController.clear();
    _productNameController.clear();
    _priceController.clear();
    _stockQuantityController.clear();

    // Navigate to ProductListPage
Navigator.push(       context,
     MaterialPageRoute(           builder: (context) =>
ProductListPage(productList: _productList)),
    );
   }
 }

 @override
 Widget build(BuildContext context) {    return
Scaffold(      appBar: AppBar(title: Text('Add
Product')),      body: Padding(        padding:
const EdgeInsets.all(16.0),        child: Form(
key: _formKey,     child: Column(
```

```
        crossAxisAlignment: CrossAxisAlignment.start,
    children: <Widget>[       TextFormField(
    controller: _productIdController,
          decoration: InputDecoration(labelText: 'Product ID'),
    validator: (value) {
            if (value == null || value.isEmpty) {
  return 'Please enter Product ID';
            }
            return null;
          },
        ),
        TextFormField(              controller: _productNameController,
decoration: InputDecoration(labelText: 'Product Name'),
validator: (value) {              if (value == null || value.isEmpty) {
return 'Please enter Product Name';
            }
            return null;
          },
        ),
        TextFormField(              controller: _priceController,
decoration: InputDecoration(labelText: 'Price'),              keyboardType:
TextInputType.numberWithOptions(decimal: true),              validator: (value) {
if (value == null ||                  value.isEmpty ||
double.tryParse(value) == null) {            return 'Please enter a valid Price';
            }
            return null;
          },
        ),
        TextFormField(         controller:
    _stockQuantityController,
```

```
            decoration: InputDecoration(labelText: 'Stock Quantity'),
keyboardType: TextInputType.number,
            validator: (value) {
              if (value == null ||                    value.isEmpty
||                  int.tryParse(value) == null) {
return 'Please enter a valid Stock Quantity';
              }
              return null;
            },
          ),
          Padding(
            padding: const EdgeInsets.symmetric(vertical: 16.0),
child: ElevatedButton(                onPressed: _addProduct,
child: Text('Add Product'),
            ),
          ),
        ],
      ),
    ),
  ),
 );
 }
}

// Page 2: ProductListPage class ProductListPage
extends StatelessWidget {
 final List<Product> productList;
 ProductListPage({required this.productList});

 @override
```

```
 Widget build(BuildContext context) {     return
Scaffold(      appBar: AppBar(title: Text('Product
List')),      body: ListView.builder(
itemCount: productList.length,       itemBuilder:
(context, index) {         final product =
productList[index];         return ListTile(
title: Text(product.productName),
subtitle: Text(
            'ID: ${product.productId}, Price: \$${product.price}, Quantity:
${product.stockQuantity}'),
          onTap: () {
           // Navigate to ProductDeletePage with selected product details
Navigator.push(              context,
            MaterialPageRoute(               builder:
(context) => ProductDeletePage(                  product:
product,                 productList: productList,
              ),
            ),
           );
          },
        );
      },
    ),
  );
 }
}

// Page 3: ProductDeletePage (Delete Product) class
ProductDeletePage extends StatelessWidget {   final
Product product;   final List<Product> productList;
```

```
 ProductDeletePage({required this.product, required this.productList});

 // Function to delete product and return to the previous page   void
_deleteProduct(BuildContext context) {     productList.remove(product);
   // Go back to ProductListPage with updated list
   Navigator.pop(context);
 }

 @override
 Widget build(BuildContext context) {     return
Scaffold(      appBar: AppBar(title: Text('Delete
Product')),      body: Padding(        padding: const
EdgeInsets.all(16.0),        child: Column(
children: <Widget>[
          Text('Are you sure you want to delete this product?'),
          SizedBox(height: 20),
          Text('Product ID: ${product.productId}'),
         Text('Product Name: ${product.productName}'),
         Text('Price: \$${product.price}'),
       Text('Stock Quantity: ${product.stockQuantity}'),
SizedBox(height: 40),          ElevatedButton(
onPressed: () => _deleteProduct(context),            child:
Text('Delete Product'),
            style: ElevatedButton.styleFrom(backgroundColor: Colors.red),
          ),
        ],
      ),
    ),
  );
 }
```

```
}
```

Q3

```
import 'package:flutter/material.dart';

//  Student  Class
class  Student  {
String studentId;
 String name;
 String gender;
 String grade;

 Student({  required
this.studentId,  required
this.name,  required
this.gender,  required
this.grade,
 });
}

// Main Function void main() {
runApp(StudentManagementApp());
}

class StudentManagementApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
return MaterialApp(  title: 'Student
```

```
Management',      theme: ThemeData(
primarySwatch: Colors.blue,
    ),
    home: StudentFormPage(),
   );
 }
}

// Page 1: StudentFormPage class StudentFormPage
extends StatefulWidget {
 @override
 _StudentFormPageState createState() => _StudentFormPageState();
}

class _StudentFormPageState extends State<StudentFormPage> {
 final _formKey = GlobalKey<FormState>(); final
 _studentIdController = TextEditingController(); final
 _nameController = TextEditingController(); String
 _gender = 'Male'; String _grade = 'A';

 final List<Student> _studentList = [];

 // Add student function   void
_addStudent() {    if
(_formKey.currentState!.validate()) {
final newStudent = Student(        studentId:
_studentIdController.text,        name:
_nameController.text,        gender: _gender,
grade: _grade,
    );
    setState(() {
```

```
      _studentList.add(newStudent);
    });

    // Clear input fields
    _studentIdController.clear();
    _nameController.clear();

    // Navigate to StudentListPage
Navigator.push(        context,
      MaterialPageRoute(       builder: (context) =>
 StudentListPage(studentList: _studentList)),
    );
   }
 }
 @override
 Widget build(BuildContext context) {
 return Scaffold(     appBar: AppBar(
 title: Text('Add Student'),
      centerTitle: true,
    ),
    body: Padding(        padding: const EdgeInsets.all(16.0),
child: Form(          key: _formKey,          child:
SingleChildScrollView(           child: Column(
crossAxisAlignment: CrossAxisAlignment.start,
children: <Widget>[              TextFormField(
controller: _studentIdController,                decoration:
InputDecoration(                 labelText: 'Student ID',
border: OutlineInputBorder(),                prefixIcon:
Icon(Icons.assignment),
              ),
```

```
              validator: (value) {                  if
(value == null || value.isEmpty) {
return 'Please enter Student ID';
                }
                return null;
              },
            ),
            SizedBox(height: 16),
            TextFormField(
              controller: _nameController,
              decoration: InputDecoration(
                labelText: 'Name',
                border: OutlineInputBorder(),
                prefixIcon: Icon(Icons.person),
              ),
              validator: (value) {                  if (value
== null || value.isEmpty) {                  return
'Please enter Student Name';
                }
                return null;
              },
            ),
            SizedBox(height: 16),
            DropdownButtonFormField<String>(
              value: _gender,
decoration: InputDecoration(
labelText: 'Gender',                  border:
OutlineInputBorder(),
              ),
              items: ['Male', 'Female', 'Other']
```

```
                .map((gender) => DropdownMenuItem<String>(
                    value: gender,
child: Text(gender),
                  ))
                .toList(),
             onChanged: (value) {
          setState(() {
          _gender = value!;
              });
            },
           ),
            SizedBox(height: 16),
            DropdownButtonFormField<String>(
              value: _grade,
              decoration: InputDecoration(
               labelText: 'Grade',
border: OutlineInputBorder(),
              ),
              items: ['A', 'B', 'C', 'D', 'E']
                .map((grade) => DropdownMenuItem<String>(
                    value: grade,
child: Text(grade),
                  ))
                .toList(),
              onChanged: (value) {
setState(() {                    _grade
= value!;
                });
              },
             ),
```

```
            SizedBox(height: 24),            ElevatedButton(
onPressed: _addStudent,                child: Text('Add
Student'),                style: ElevatedButton.styleFrom(
padding: EdgeInsets.symmetric(vertical: 16),
textStyle: TextStyle(fontSize: 18),
              ),
            ),
          ],
        ),
      ),
    ),
  );
 }
}

// Page 2: StudentListPage class StudentListPage
extends StatelessWidget {   final List<Student>
studentList;

 StudentListPage({required this.studentList});

 @override
 Widget build(BuildContext context) {
return Scaffold(      appBar: AppBar(
title: Text('Student List'),
centerTitle: true,
    ),
    body: Padding(        padding: const
EdgeInsets.all(16.0),        child: Column(          children:
<Widget>[          Expanded(            child:
```

```
ListView.builder(     itemCount: studentList.length,
itemBuilder: (context, index) {       final student =
studentList[index];      return Card(       margin:
EdgeInsets.symmetric(vertical: 8),         child: ListTile(
                title: Text(student.name),
                subtitle: Text(
                   'ID: ${student.studentId}, Grade: ${student.grade}'),
 trailing: Icon(Icons.arrow_forward),
                onTap: () {
                  Navigator.push(
context,
                   MaterialPageRoute(
builder: (context) =>
                       StudentFilterPage(studentList: studentList),
                   ),
                  );
                 },
               ),
             );
           },
         ),
        ),
        ElevatedButton(
onPressed: () {
Navigator.push(                context,
            MaterialPageRoute(builder: (context) => StudentFormPage()),
           );
         },
         child: Text('Add New Student'),
        ),
```

```
        ],
      ),
    ),
  );
 }
}

// Page 3: StudentFilterPage class StudentFilterPage
extends StatefulWidget {   final List<Student>
studentList;

 StudentFilterPage({required this.studentList});

 @override
 _StudentFilterPageState createState() => _StudentFilterPageState();
}

class _StudentFilterPageState extends State<StudentFilterPage> {
 String _selectedGrade = 'A';
 List<Student> _filteredStudents = [];

 // Filter students based on selected grade
void _filterStudents() {    setState(() {
    _filteredStudents = widget.studentList
        .where((student) => student.grade == _selectedGrade)
        .toList();
  });
 }

 @override
```

```
 void initState() {
  super.initState();
  _filteredStudents = widget.studentList;
 }

 @override
 Widget build(BuildContext context) {
return Scaffold(      appBar: AppBar(
title: Text('Filter Students by Grade'),
centerTitle: true,
    ),
    body: Padding(        padding: const
EdgeInsets.all(16.0),        child: Column(
children: <Widget>[
          DropdownButtonFormField<String>(
value: _selectedGrade,              decoration:
InputDecoration(              labelText: 'Select
Grade',              border:
OutlineInputBorder(),
           ),
           items: ['A', 'B', 'C', 'D', 'E']
              .map((grade) => DropdownMenuItem<String>(
                  value: grade,
child: Text(grade),
                 ))
              .toList(),
           onChanged: (value) {
    setState(() {
              _selectedGrade = value!;
            });
            _filterStudents();
```

```
            },
          ),
          SizedBox(height: 16),           Expanded(
child: ListView.builder(               itemCount:
_filteredStudents.length,               itemBuilder: (context,
index) {                final student =
_filteredStudents[index];                return Card(
margin: EdgeInsets.symmetric(vertical: 8),
                child: ListTile(
title: Text(student.name),
subtitle: Text(
                    'ID: ${student.studentId}, Grade: ${student.grade}'),
                ),
              );
            },
          ),
        ),
      ],
    ),
  ),
 );
 }
}
```

Q4

```
import 'package:flutter/material.dart';

// Expense Class
```

```
class Expense {   String
expenseId;   String title;
 double amount;
 String category;

 Expense({    required
this.expenseId,    required
this.title,    required
this.amount,    required
this.category,
 });
}

// Main Function void main() {
runApp(ExpenseTrackerApp());
}

class ExpenseTrackerApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
return MaterialApp(      title: 'Expense
Tracker',      theme: ThemeData(
primarySwatch: Colors.blue,
      textTheme: TextTheme(
        bodyText2: TextStyle(color: Colors.black, fontSize: 16),
      ),
    ),
    home: ExpenseFormPage(),
  );
 }
```

```
}

// Page 1: ExpenseFormPage class ExpenseFormPage
extends StatefulWidget {
  @override
  _ExpenseFormPageState createState() => _ExpenseFormPageState();
}

class _ExpenseFormPageState extends State<ExpenseFormPage> {   final
_formKey = GlobalKey<FormState>();   final _expenseIdController =
TextEditingController();   final _titleController = TextEditingController();
final _amountController = TextEditingController();
  String _category = 'Food';

  final List<Expense> _expenseList = [];

  // Add expense function   void _addExpense() {     if
(_formKey.currentState!.validate()) {       final
newExpense = Expense(         expenseId:
_expenseIdController.text,         title:
_titleController.text,         amount:
double.parse(_amountController.text),
       category: _category,
     );
     setState(() {
       _expenseList.add(newExpense);
     });
     // Clear input fields
     _expenseIdController.clear();
     _titleController.clear();
```

```
    _amountController.clear();

    // Navigate to ExpenseListPage
Navigator.push(        context,
      MaterialPageRoute(            builder: (context) =>
ExpenseListPage(expenseList: _expenseList)),
    );
   }
 }

 @override
 Widget build(BuildContext context) {
return Scaffold(      appBar: AppBar(
title: Text('Add Expense'),
centerTitle: true,
    ),
    body: Padding(        padding: const
EdgeInsets.all(16.0),        child: Form(
key: _formKey,
       child: SingleChildScrollView(
         child: Column(
           crossAxisAlignment: CrossAxisAlignment.start,
           children: <Widget>[
             TextFormField(
               controller: _expenseIdController,
               decoration: InputDecoration(   labelText:
               'Expense ID',
               border: OutlineInputBorder(),     prefixIcon:
           Icon(Icons.receipt),
               ),
```

```
            validator: (value) {                if
(value == null || value.isEmpty) {
return 'Please enter Expense ID';
              }
              return null;
            },
          ),
          SizedBox(height: 16),
TextFormField(              controller:
_titleController,              decoration:
InputDecoration(              labelText:
'Title',              border:
OutlineInputBorder(),
prefixIcon: Icon(Icons.title),
            ),
            validator: (value) {              if (value
== null || value.isEmpty) {              return
'Please enter Expense Title';
              }
              return null;
            },
          ),
          SizedBox(height: 16),
    TextFormField(          controller:
    _amountController,
          decoration: InputDecoration(
           labelText: 'Amount',
    border: OutlineInputBorder(),
    prefixIcon: Icon(Icons.attach_money),
          ),
```

```
            keyboardType: TextInputType.numberWithOptions(decimal: true),
validator: (value) {              if (value == null ||                  value.isEmpty ||
double.tryParse(value) == null) {                  return 'Please enter a valid
Amount';
              }
              return null;
             },
            ),
            SizedBox(height: 16),
            DropdownButtonFormField<String>(
             value: _category,
decoration: InputDecoration(
labelText: 'Category',              border:
OutlineInputBorder(),
            ),
            items: [
              'Food',
              'Transport',
              'Entertainment',
             'Utilities',
             'Others'
            ]
               .map((category) => DropdownMenuItem<String>(
          value: category,
                child: Text(category),
               ))
            .toList(),
          onChanged: (value) {    setState(()
         {
              _category = value!;
```

```
                });
              },
            ),
            SizedBox(height: 24),              ElevatedButton(
onPressed: _addExpense,                child: Text('Add
Expense'),                style: ElevatedButton.styleFrom(
padding: EdgeInsets.symmetric(vertical: 16),
textStyle: TextStyle(fontSize: 18),
              ),
            ),
          ],
        ),
      ),
    ),
  );
 }
}

// Page 2: ExpenseListPage class ExpenseListPage
extends StatelessWidget {
 final List<Expense> expenseList; ExpenseListPage({required this.expenseList});

 @override
 Widget build(BuildContext context) {   return
 Scaffold(
   appBar: AppBar(       title:
Text('Expense List'),
centerTitle: true,
   ),
```

```
    body: Padding(        padding: const
EdgeInsets.all(16.0),        child: Column(          children:
<Widget>[           Expanded(             child:
ListView.builder(               itemCount:
expenseList.length,               itemBuilder: (context,
index) {               final expense = expenseList[index];
return Card(                  margin:
EdgeInsets.symmetric(vertical: 8),                  child:
ListTile(                  title: Text(expense.title),
subtitle: Text(
                  'Amount: \$${expense.amount} - Category: ${expense.category}'),
trailing: Icon(Icons.arrow_forward),                  onTap: () {          Navigator.push(
context,
                  MaterialPageRoute(              builder:
        (context) => ExpenseManagePage(                expense:
        expense,
                    expenseList: expenseList,
                  ),
                ),
              );
            },
          ),
        );
      },
    ),
        ),
        ElevatedButton(
onPressed: () {
Navigator.push(                context,
            MaterialPageRoute(builder: (context) => ExpenseFormPage()),
```

```
            );
          },
          child: Text('Add New Expense'),
        ),
      ],
    ),
  ),
 );
 }
}

// Page 3: ExpenseManagePage
class ExpenseManagePage extends StatelessWidget {
 final Expense expense;
 final List<Expense> expenseList;
 ExpenseManagePage({required this.expense, required this.expenseList});

 // Delete expense function void
 _deleteExpense(BuildContext context) {
 expenseList.remove(expense);
  // Navigate back to ExpenseListPage with updated list
  Navigator.pop(context);
 }

 @override
 Widget build(BuildContext context) {
return Scaffold( appBar: AppBar(
title: Text('Expense Details'),
centerTitle: true,
  ),
```

```
    body: Padding(        padding: const
EdgeInsets.all(16.0),        child: Column(
children: <Widget>[
          Text('Expense ID: ${expense.expenseId}',
style: TextStyle(fontSize: 18)),
          SizedBox(height: 16),
          Text('Title: ${expense.title}', style: TextStyle(fontSize: 18)),
          SizedBox(height: 16),
          Text('Amount: \$${expense.amount}', style: TextStyle(fontSize: 18)),
         SizedBox(height: 16),
         Text('Category: ${expense.category}',     style:
         TextStyle(fontSize: 18)),
         SizedBox(height: 40),
         ElevatedButton(
          onPressed: () => _deleteExpense(context), child:
Text('Delete Expense'), style: ElevatedButton.styleFrom(
primary: Colors.red,   padding:
EdgeInsets.symmetric(vertical: 16),              textStyle:
TextStyle(fontSize: 18),
           ),
          ),
        ],
      ),
    ),
  );
 }
}
```

Q5

```
import 'package:flutter/material.dart';

//  Appointment  Class
class  Appointment  {
String appointmentId;
 String title;
 DateTime date;
 String clientName; String
 status;

 Appointment({ required
  this.appointmentId, required
  this.title, required this.date,
  required this.clientName,
  required this.status,
 });
}

// Main Function void main() {
runApp(AppointmentBookingApp());
}

class AppointmentBookingApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
return MaterialApp(  title:
'Appointment Booking',  theme:
ThemeData(  primarySwatch:
Colors.blue,
```

```
    ),
    home: AppointmentFormPage(),
   );
  }
}

// Page 1: AppointmentFormPage class
AppointmentFormPage extends StatefulWidget {
  @override
  _AppointmentFormPageState createState() => _AppointmentFormPageState();
}
class _AppointmentFormPageState extends State<AppointmentFormPage> {   final
_formKey = GlobalKey<FormState>();   final _appointmentIdController =
TextEditingController();   final _titleController = TextEditingController();   final
_clientNameController = TextEditingController();
  String _status = 'Pending';
  DateTime _appointmentDate = DateTime.now();

  final List<Appointment> _appointmentList = [];

  // Function to pick appointment date
  Future<void> _selectDate(BuildContext context) async {
final DateTime? pickedDate = await showDatePicker(
context: context,      initialDate: _appointmentDate,
firstDate: DateTime(2000),      lastDate: DateTime(2101),
   );
   if (pickedDate != null && pickedDate != _appointmentDate) {      setState(() {
      _appointmentDate = pickedDate;
    });
   }
```

```
  }

  // Add appointment function void
  _bookAppointment() {
    if (_formKey.currentState!.validate()) {
      final newAppointment = Appointment(
        appointmentId: _appointmentIdController.text,
        title: _titleController.text,
        date: _appointmentDate, clientName:
        _clientNameController.text, status: _status,
      );
      setState(() {
        _appointmentList.add(newAppointment);
      });

      // Clear input fields
      _appointmentIdController.clear();
      _titleController.clear();
      _clientNameController.clear();

      // Navigate to AppointmentListPage
Navigator.push(        context,
        MaterialPageRoute(
builder: (context) =>
              AppointmentListPage(appointments: _appointmentList)),
      );
    }
  }

  @override
```

```
 Widget build(BuildContext context) {     return
Scaffold(
    appBar: AppBar(   title: Text('Book
    Appointment'),   centerTitle: true,
    ),
    body: Padding(
     padding: const EdgeInsets.all(16.0),
     child: Form(   key: _formKey,   child:
     SingleChildScrollView(     child:
     Column(
          crossAxisAlignment: CrossAxisAlignment.start,
children: <Widget>[              TextFormField(
controller: _appointmentIdController,
decoration: InputDecoration(                  labelText:
'Appointment ID',                 border:
OutlineInputBorder(),                 prefixIcon:
Icon(Icons.assignment),
              ),
              validator: (value) {                 if (value ==
null || value.isEmpty) {                  return 'Please
enter Appointment ID';
               }
               return null;
              },
             ),
             SizedBox(height: 16),
TextFormField(                controller:
_titleController,               decoration:
InputDecoration(                  labelText:
'Title',
```

```
            border: OutlineInputBorder(),
          prefixIcon: Icon(Icons.title),
          ), validator: (value) {   if (value == null
          || value.isEmpty) {     return 'Please
          enter Appointment Title';
            }
            return null;
          },
          ),
          SizedBox(height: 16),
TextFormField(              controller:
_clientNameController,              decoration:
InputDecoration(              labelText: 'Client
Name',              border:
OutlineInputBorder(),              prefixIcon:
Icon(Icons.person),
            ),
            validator: (value) {              if (value
== null || value.isEmpty) {              return
'Please enter Client Name';
              }
              return null;
            },
          ),
          SizedBox(height: 16),
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: [
              Text(
                "Appointment Date: ${_appointmentDate.toLocal()}"
```

```
                .split(' ')[0],
            style: TextStyle(fontSize: 16),
          ),
           IconButton(     icon:
          Icon(Icons.calendar_today),
          onPressed: () => _selectDate(context),
           ),
          ],
         ),
         SizedBox(height: 16),
         DropdownButtonFormField<String>(
          value: _status,
decoration: InputDecoration(
labelText: 'Status',                 border:
OutlineInputBorder(),
          ),
          items: ['Pending', 'Confirmed', 'Completed']
.map((status) => DropdownMenuItem<String>(
                value: status,
child: Text(status),
                ))
            .toList(),
          onChanged: (value) {
setState(() {
_status = value!;
            });
          },
         ),
         SizedBox(height: 24),            ElevatedButton(
onPressed: _bookAppointment,     child: Text('Book
```

```
Appointment'),     style: ElevatedButton.styleFrom(
padding: EdgeInsets.symmetric(vertical: 16),
                textStyle: TextStyle(fontSize: 18),
              ),
            ),
          ],
        ),
      ),
    ),
  );
 }
}

// Page 2: AppointmentListPage class AppointmentListPage
extends StatelessWidget {   final List<Appointment>
appointments;

 AppointmentListPage({required this.appointments});

 @override
 Widget build(BuildContext context) {
return Scaffold(      appBar: AppBar(
title: Text('Scheduled Appointments'),
centerTitle: true,
    ),
    body: Padding(        padding: const
EdgeInsets.all(16.0),        child: Column(
children: <Widget>[
        Expanded(
```

```
          child: ListView.builder(
           itemCount:                  appointments.length,
           itemBuilder:   (context,   index)   {        final
           appointment = appointments[index];

            return Card(
              margin: EdgeInsets.symmetric(vertical: 8),
       child: ListTile(
              title: Text(appointment.title),
              subtitle: Text(
                'Client: ${appointment.clientName}, Date: ${appointment.date.toLocal()}'),
trailing: Text(appointment.status),                onTap: () {
                Navigator.push(
context,
                MaterialPageRoute(                builder:
(context) => AppointmentFilterPage(
appointments: appointments),
                  ),
                );
              },
            ),
          );
        },
      ),
    ),
        ElevatedButton(
onPressed: () {
Navigator.push(              context,
          MaterialPageRoute(
            builder: (context) => AppointmentFormPage()),
        );
```

```
          },
          child: Text('Book New Appointment'),
        ),
      ],
    ),
  ),
 );
 }
}

// Page 3: AppointmentFilterPage class
AppointmentFilterPage extends StatefulWidget {  final
List<Appointment> appointments;

 AppointmentFilterPage({required this.appointments});

 @override
 _AppointmentFilterPageState createState() => _AppointmentFilterPageState();
}

class _AppointmentFilterPageState extends State<AppointmentFilterPage> {
 String _selectedStatus = 'Pending';
 List<Appointment> _filteredAppointments = [];

 // Filter appointments based on selected status
void _filterAppointments() {   setState(() {
    _filteredAppointments = widget.appointments
      .where((appointment) => appointment.status == _selectedStatus)
      .toList();
  });
```

```
 }
 @override
 void initState() {   super.initState();
  _filteredAppointments = widget.appointments;
 }

 @override
 Widget build(BuildContext context) {     return
Scaffold(      appBar: AppBar(        title:
Text('Filter Appointments by Status'),
centerTitle: true,
    ),
    body: Padding(        padding: const
EdgeInsets.all(16.0),        child: Column(
children: <Widget>[
          DropdownButtonFormField<String>(
value: _selectedStatus,              decoration:
InputDecoration(              labelText: 'Select
Status',              border:
OutlineInputBorder(),
           ),
           items: ['Pending', 'Confirmed', 'Completed']
.map((status) => DropdownMenuItem<String>(
                  value: status,
                  child: Text(status),
                ))
              .toList(),
          onChanged: (value) {
           setState(() {
              _selectedStatus = value!;
```

```
            });
            _filterAppointments();
          },
        ),
        SizedBox(height: 16), Expanded( child:
ListView.builder( itemCount:
_filteredAppointments.length, itemBuilder: (context,
index) { final appointment =
_filteredAppointments[index]; return Card(
margin: EdgeInsets.symmetric(vertical: 8), child:
ListTile( title: Text(appointment.title),
subtitle: Text(
                'Client: ${appointment.clientName}, Date: ${appointment.date.toLocal()}'),
trailing: Text(appointment.status),
              ),
            );
          },
        ),
      ),
    ],
  ),
 ),
);
 }
}
```