This Flutter code sets up a simple app with buttons to select both date and time. The selected date and time are displayed on the screen. Adjust the UI and logic according to your specific application requirements.

CODE:

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

void main() {
  runApp(MyApp());
}

class Task {
  String title;
  String description;
  DateTime dateTime;
  Task({required this.title, required this.description, required this.dateTime});
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'TODO List',
      home: MyHomePage(),
      debugShowCheckedModeBanner: false,
    );
  }
}

class MyHomePage extends StatefulWidget {
  @override
  _MyHomePageState createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  List<Task> tasks = [];
  List<Task> completedTasks = []; // New list to store completed tasks
  final TextEditingController titleController = TextEditingController();
  final TextEditingController descriptionController = TextEditingController();
  DateTime selectedDate = DateTime.now();
  TimeOfDay selectedTime = TimeOfDay.now();

  void _selectDate(BuildContext context) async {
    final DateTime? picked = await showDatePicker(
      context: context,
      initialDate: selectedDate,
      firstDate: DateTime.now(),
      lastDate: DateTime(2101),
```

```
    );
    if (picked != null && picked != selectedDate) {
      setState(() {
        selectedDate = picked;
      });
    }
  }

  void _selectTime(BuildContext context) async {
    final TimeOfDay? picked = await showTimePicker(
      context: context,
      initialTime: selectedTime,
    );
    if (picked != null && picked != selectedTime) {
      setState(() {
        selectedTime = picked;
      });
    }
  }

  void _addTask() {
    if (titleController.text.isNotEmpty) {
      Task newTask = Task(
        title: titleController.text,
        description: descriptionController.text,
        dateTime: DateTime(
          selectedDate.year,
          selectedDate.month,
          selectedDate.day,
          selectedTime.hour,
          selectedTime.minute,
        ),
      );
      setState(() {
        tasks.add(newTask);
      });
      _clearInputFields();
    }
  }

  void _deleteTask(int index) {
    setState(() {
      tasks.removeAt(index);
    });
  }

  void _editTask(int index) {
    Task task = tasks[index];
    titleController.text = task.title;
    descriptionController.text = task.description;
    selectedDate = task.dateTime;
```

```
  selectedTime = TimeOfDay.fromDateTime(task.dateTime);
  _showEditTaskDialog(context, index);
 }

 void _completeTask(int index) {
  setState(() {
   completedTasks.add(tasks[index]);
   tasks.removeAt(index);
  });
 }

 void _clearInputFields() {
  titleController.clear();
  descriptionController.clear();
 }

 void _showAddTaskDialog(BuildContext context) {
  showDialog(
   context: context,
   builder: (BuildContext context) {
    return AlertDialog(
     backgroundColor: Colors.yellowAccent,
     title: Text('Add Task', style: TextStyle(color: Colors.black)),
     content: Column(
      mainAxisSize: MainAxisSize.min,
      children: [
       TextField(
        style: TextStyle(color: Colors.grey),
        controller: titleController,
        decoration: InputDecoration(labelText: 'Title', labelStyle: TextStyle(color: Colors.purple)),
       ),
       TextField(
        style: TextStyle(color: Colors.grey),
        controller: descriptionController,
        decoration: InputDecoration(labelText: 'Description', labelStyle: TextStyle(color:
Colors.purple)),
       ),
       Row(
        children: [
         TextButton(
          onPressed: () => _selectDate(context),
          child: Text('Select Date', style: TextStyle(color: Colors.pinkAccent)),
         ),
         TextButton(
          onPressed: () => _selectTime(context),
          child: Text('Select Time', style: TextStyle(color: Colors.pinkAccent)),
         ),
        ],
       ),
      ],
     ),
```

```
        actions: [
          TextButton(
            onPressed: () {
              Navigator.of(context).pop();
            },
            child: Text('Cancel', style: TextStyle(color: Colors.red)),
          ),
          TextButton(
            onPressed: () {
              _addTask();
              Navigator.of(context).pop();
            },
            child: Text('Add', style: TextStyle(color: Colors.black)),
          ),
        ],
      );
    },
  );
}

 void _showEditTaskDialog(BuildContext context, int index) {
  showDialog(
    context: context,
    builder: (BuildContext context) {
      return AlertDialog(
        backgroundColor: Colors.blueGrey[900],
        title: Text('Edit Task', style: TextStyle(color: Colors.white)),
        content: Column(
          mainAxisSize: MainAxisSize.min,
          children: [
            TextField(
              style: TextStyle(color: Colors.grey),
              controller: titleController,
              decoration: InputDecoration(labelText: 'Title', labelStyle: TextStyle(color: Colors.white)),
            ),
            TextField(
              style: TextStyle(color: Colors.grey),
              controller: descriptionController,
              decoration: InputDecoration(labelText: 'Description', labelStyle: TextStyle(color:
Colors.white)),
            ),
            Row(
              children: [
                TextButton(
                  onPressed: () => _selectDate(context),
                  child: Text('Select Date', style: TextStyle(color: Colors.white)),
                ),
                TextButton(
                  onPressed: () => _selectTime(context),
                  child: Text('Select Time', style: TextStyle(color: Colors.white)),
                ),
```

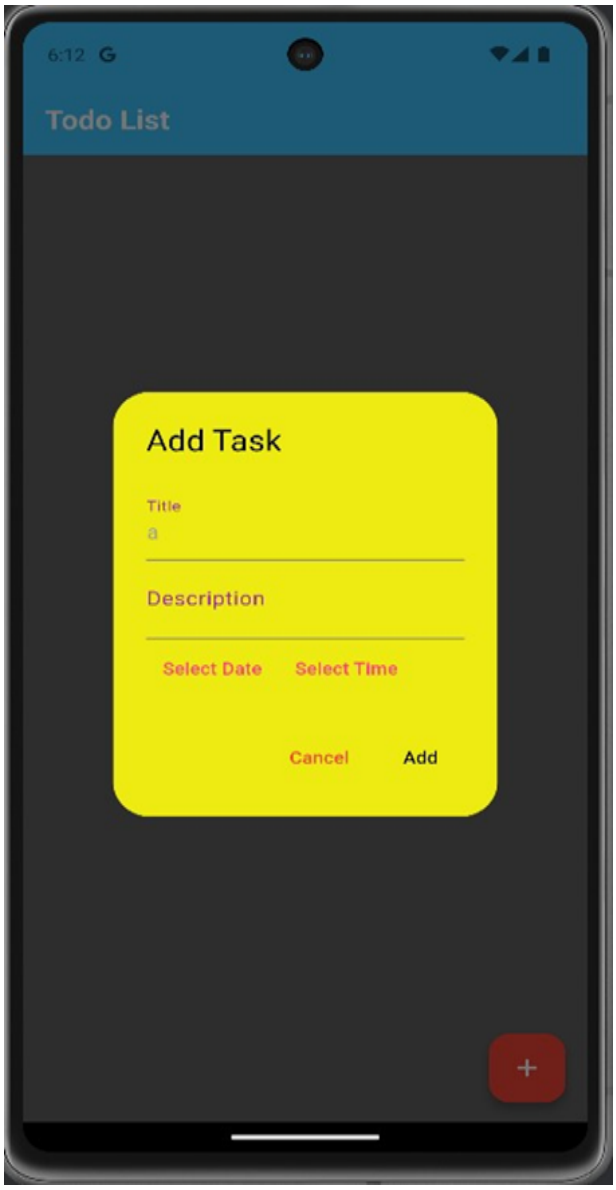
```
            ],
          ),
        ],
      ),
      actions: [
        TextButton(
          onPressed: () {
            Navigator.of(context).pop();
          },
          child: Text('Cancel', style: TextStyle(color: Colors.red)),
        ),
        TextButton(
          onPressed: () {
            Task editedTask = Task(
              title: titleController.text,
              description: descriptionController.text,
              dateTime: DateTime(
                selectedDate.year,
                selectedDate.month,
                selectedDate.day,
                selectedTime.hour,
                selectedTime.minute,
              ),
            );
            setState(() {
              tasks[index] = editedTask;
            });
            Navigator.of(context).pop();
          },
          child: Text('Save', style: TextStyle(color: Colors.white70)),
        ),
      ],
    );
  },
 );
}

@override
Widget build(BuildContext context) {
 return Scaffold(
  backgroundColor: Colors.grey[700],
  appBar: AppBar(
   title: Text(
    'Todo List',
    style: TextStyle(color: Colors.white, fontWeight: FontWeight.bold),
   ),
   backgroundColor: Colors.lightBlueAccent,
  ),
  body: SingleChildScrollView(
   child: Column(
    children: [
```

```
          // Widget to display tasks
          ListView.builder(
            shrinkWrap: true,
            itemCount: tasks.length,
            itemBuilder: (context, index) {
              return Container(
                decoration: BoxDecoration(
                  color: Colors.grey[900],
                  borderRadius: BorderRadius.circular(10),
                ),
                margin: const EdgeInsets.symmetric(vertical: 1.5, horizontal: 1.5),
                child: ListTile(
                  title: Text(
                    tasks[index].title,
                    style: TextStyle(color: Colors.white, fontWeight: FontWeight.bold),
                  ),
                  subtitle: Text(
                    tasks[index].description,
                    style: TextStyle(color: Colors.white70, fontStyle: FontStyle.italic),
                  ),
                  trailing: Row(
                    mainAxisSize: MainAxisSize.min,
                    children: [
                      Text(DateFormat('yyyy-MM-dd HH:mm').format(tasks[index].dateTime), style:
TextStyle(color: Colors.white)),
                      IconButton(
                        icon: Icon(Icons.delete),
                        onPressed: () => _deleteTask(index),
                        color: Colors.red,
                      ),
                      IconButton(
                        icon: Icon(Icons.edit),
                        onPressed: () => _editTask(index),
                        color: Colors.blue,
                      ),
                      IconButton(
                        icon: Icon(Icons.check),
                        onPressed: () => _completeTask(index),
                        color: Colors.green,
                      ),
                    ],
                  ),
                ),
              );
            },
          ),
          SizedBox(height: 20), // Adding space between the two lists
          // Widget to display completed tasks
          ListView.builder(
            shrinkWrap: true,
            itemCount: completedTasks.length,
```

```
            itemBuilder: (context, index) {
              return Container(
                decoration: BoxDecoration(
                  color: Colors.grey[900],
                  borderRadius: BorderRadius.circular(10),
                ),
                margin: const EdgeInsets.symmetric(vertical: 1.5, horizontal: 1.5),
                child: ListTile(
                  title: Text(
                    completedTasks[index].title,
                    style: TextStyle(color: Colors.white, fontWeight: FontWeight.bold),
                  ),
                  subtitle: Text(
                    completedTasks[index].description,
                    style: TextStyle(color: Colors.white70, fontStyle: FontStyle.italic),
                  ),
                  trailing: Text(DateFormat('yyyy-MM-dd
HH:mm').format(completedTasks[index].dateTime), style: TextStyle(color: Colors.white)),
                ),
              );
            },
          ),
        ],
      ),
    ),
    floatingActionButton: FloatingActionButton(
      foregroundColor: Colors.white,
      backgroundColor: Colors.red,
      onPressed: () => _showAddTaskDialog(context),
      tooltip: 'Add Task',
      child: Icon(Icons.add),
    ),
  );
 }
```


6:13
Todo List
a
1
2024-04-16 17:11
b
2
2024-04-23 19:11
c
3
2024-04-23 19:11


6:12
Todo List
Add Task
Title
a
Description
Select Date
Select Time
Cancel
Add