

# Flutter Coding Exam (20 Practical Tasks)

Beginner → Advanced • Hands-on tasks with acceptance criteria

## A) Fundamentals & UI (1–5)

### 1) Hello Material

- Build a MaterialApp with theme and a home Scaffold.
- Acceptance: AppBar title 'Flutter Exam'; uses primary color and a custom font from assets.

### 2) Profile Card

- Centered profile card with avatar, name, role, phone.
- Acceptance: Rounded corners, elevation, responsive padding; tapping phone opens tel: with url\_launcher.

### 3) Product Grid

- 2-column GridView of 8 products from a local list.
- Acceptance: Each tile has asset image, name, price; tapping opens details with a Hero image.

### 4) Todo List (Local State)

- Add todos via TextField + Add; toggle complete.
- Acceptance: setState only; in-session persistence; shows counts total/completed.

### 5) Responsive Layout

- Dashboard adapts to device size.
- Acceptance: Mobile single-column; Tablet/Web 3-col grid using LayoutBuilder/MediaQuery.

## B) Networking & Persistence (6–10)

### 6) Fetch Posts (HTTP)

- GET <https://jsonplaceholder.typicode.com/posts>
- Acceptance: List titles; pull-to-refresh; loading & error states.

### 7) Post Details + Comments

- Navigate to details and fetch related comments.
- Acceptance: Two parallel requests (post + comments) handled safely.

### 8) Offline Cache (Hive)

- Cache posts to Hive for offline access.
- Acceptance: If offline, show last cache; visible 'Offline' banner.

## 9) Auth Mock + Guarded Route

- Fake login saved in SharedPreferences; protect /dashboard.
- Acceptance: Logout clears session; guarded navigation works.

## 10) Image Upload (Multipart)

- Pick image from gallery and upload to mock endpoint.
- Acceptance: Show progress %, retry on error; store uploaded URL locally.

## C) State Management (11–12)

### 11) Provider / Riverpod Store

- Move Todo state to Provider or Riverpod.
- Acceptance: Separate state class; unit tests for add/toggle/remove.

### 12) Bloc or Cubit Counter

- Implement CounterCubit with + / – / reset.
- Acceptance: BlocBuilder updates UI; states verified by tests.

## D) Navigation & Deep Links (13–14)

### 13) Named Routes & Arguments

- Routes: /, /post/:id, /settings.
- Acceptance: Use onGenerateRoute or go\_router; invalid IDs handled gracefully.

### 14) Deep Link

- Handle app link myexam://post/42 opening detail page.
- Acceptance: Works on Android & Web (path /post/42).

## E) Testing (15)

### 15) Tests

- Write 2 unit tests, 1 widget test, 1 integration test.
- Acceptance: Use flutter\_test & integration\_test; CI-ready.

## F) Performance & Accessibility (16)

### 16) Large List Perf

- Render 5k items efficiently with ListView.builder.

- Acceptance: Smooth scrolling; item reuse; helpful Semantics labels.

## **G) Firebase (17)**

### **17) Firestore Notes**

- CRUD notes collection with live updates.
- Acceptance: Add/edit/delete; snapshot stream; rules deny unauthenticated writes.

## **H) Advanced (18–19)**

### **18) Isolate for Heavy JSON**

- Parse 10k-item JSON without blocking UI.
- Acceptance: Uses compute or Isolate; shows spinner while parsing.

### **19) Platform Channel**

- Method channel `device_info` returns a native string.
- Acceptance: Android manufacturer+model; iOS systemName+version; exposed via `DeviceInfoService.getSummary()`.

## **I) Packaging & Delivery (20)**

### **20) Flavors + Icons + Splash**

- Create dev/prod flavors with different app names and base URLs.
- Acceptance: `flutter_launcher_icons` & `flutter_native_splash` set up; flavor configs applied.

# Starters

```
main.dart (minimal) ----- import 'package:flutter/material.dart'; void main() =>
runApp(const ExamApp()); class ExamApp extends StatelessWidget { const ExamApp({super.key});
@override Widget build(BuildContext context) { return MaterialApp( title: 'Flutter Exam', theme:
ThemeData(useMaterial3: true, colorSchemeSeed: Colors.indigo), debugShowCheckedModeBanner: false,
home: const HomePage(), ); } } class HomePage extends StatelessWidget { const
HomePage({super.key}); @override Widget build(BuildContext context) { return Scaffold( appBar:
AppBar(title: const Text('Flutter Exam')), body: Center( child: ElevatedButton( onPressed: () =>
Navigator.push( context, MaterialPageRoute(builder: (_) => const TodoPage()), ), child: const
Text('Start Tasks'), ), ), ); } } class TodoPage extends StatefulWidget { const
TodoPage({super.key}); @override State createState() => _TodoPageState(); } class _TodoPageState
extends State { final _c = TextEditingController(); final _items = []; @override void dispose() {
_c.dispose(); super.dispose(); } @override Widget build(BuildContext context) { return Scaffold(
appBar: AppBar(title: const Text('Todos')), body: Padding( padding: const EdgeInsets.all(16),
child: Column( children: [ Row(children: [ Expanded(child: TextField(controller: _c, decoration:
const InputDecoration(hintText: 'Add todo'))), const SizedBox(width: 8), FilledButton(onPressed:
() { if (_c.text.isNotEmpty) setState(() { _items.add(_c.text); _c.clear(); }); }, child: const
Text('Add')), ], const SizedBox(height: 12), Expanded( child: ListView.builder( itemCount:
_items.length, itemBuilder: (_, i) => ListTile(title: Text(_items[i])), ), ), ), ), ); } }
```

```
counter_test.dart (unit test) ----- import
'package:flutter_test/flutter_test.dart'; class Counter { int value = 0; void inc() => value++;
void dec() => value--; } void main() { test('counter increments and decrements', () { final c =
Counter(); c.inc(); c.inc(); expect(c.value, 2); c.dec(); expect(c.value, 1); }); }
```