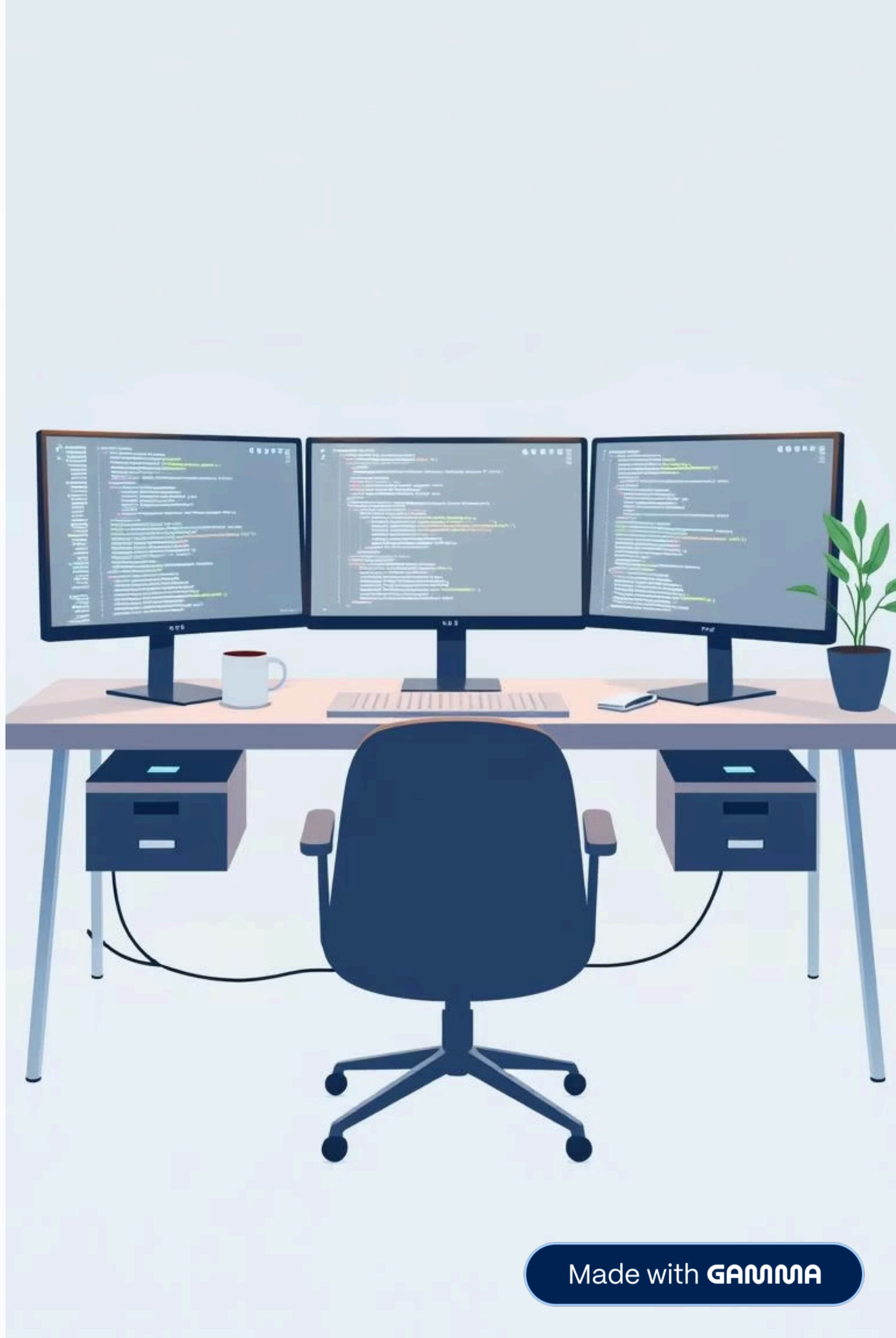
Flutter Bootcamp



Week 1 Schedule: Dart Fundamentals

Total Hours: 20 hours (5 hours daily)

Fundamentals

- **Introduction to Dart** - Its history and why Flutter uses it, running code (DartPad/VSCode)
- **Variables and Data Types** - var, final, const, int, double, String, bool, dynamic & Object, Null safety
- **Arithmetic and Logical Operations** - Arithmetic operations, comparison, logical operations
- **Control Flow** - if/else, switch/case, Loops (for, while, do-while, for-in, forEach)

Collections and Functions

- **Collections** - Lists, Sets, Maps, Spread operator (...), if/for in collections
- **Functions** - Definition and invocation, default and optional values, arrow functions, anonymous functions
- **Scope and Closures** - Variable scope, Closures (important in Flutter)

Object-Oriented Programming (OOP) in Dart

- **Classes and Objects** - Class, Object, Constructor, Properties & Methods
- **Core OOP Principles** - Inheritance, Polymorphism, Abstract Classes, Interfaces
- **Advanced Concepts** Getters & Setters, static, factory constructors, Enums

Week 2: Flutter Foundations & BMI Calculator

Kickstart your advanced Flutter journey by mastering core UI elements and user interaction. We'll solidify your understanding of widget composition and basic state management.

1

Advanced Widgets

Explore powerful widgets **ListView** for scrolling lists, and **GridView** for structured grids.

2

Layout Techniques

Dive deep into efficient layout with **Column**, **Row**, and **Container** for flexible UI arrangement.

3

Navigation & Routing

Learn seamless screen transitions and data passing using Flutter's navigation system.

4

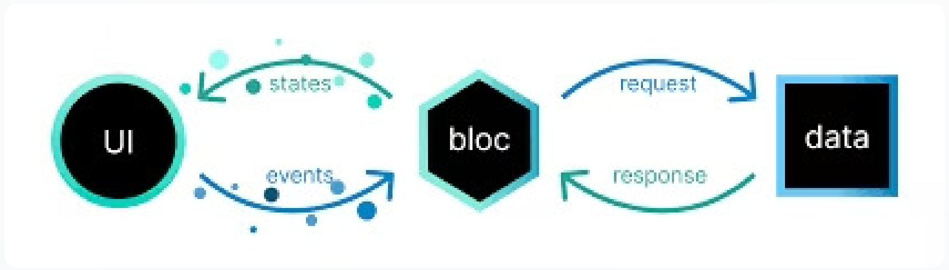
Forms

Build robust user input forms for a better user experience.

Hands-on Project: Develop a simple BMI Calculator app, focusing on **setState** for immediate UI updates.

Week 3: State Management with Bloc & Responsive Design

Elevate your app's architecture and adaptability by implementing advanced state management and ensuring your UI looks great on any screen.



Introduction to Bloc & Cubit

Understand the core concepts of Bloc and Cubit for scalable and predictable state management.

setState

Simple, local state management.

Cubit

Simpler Bloc for managing state changes with events.

Bloc

Advanced, event-driven state management for complex apps.

Responsive Design Principles

Learn to adapt your UI for various screen sizes using **MediaQuery** and **LayoutBuilder**, ensuring a consistent experience across devices.

Project Task: Refactor your BMI Calculator to use Bloc and make it responsive.

Week 4: Local Storage with Hive & To-Do App

Persist data locally in your applications using a lightweight database and enhance user interaction with dynamic UI elements.

Introducing Hive Database

Explore Hive, a fast and easy-to-use NoSQL database for local data storage in Flutter. Learn how to initialize, open boxes, and store various data types.

CRUD Operations

Master Create, Read, Update, and Delete operations for managing your local data effectively within the To-Do app.

UI/UX Enhancements

Implement interactive **AlertDialogs** for user confirmations, **Toasts** for transient messages, and state filtering for dynamic lists.

Hands-on Project: Build a fully functional To-Do List application, saving tasks locally with Hive and providing a smooth user experience.

Week 5: Firebase Integration & External APIs

Connect your Flutter apps to the cloud with Firebase and fetch data from external services to build dynamic, real-world applications.

Firebase Authentication

- Implement secure user login and registration.
- Manage user sessions and authentication states.

Firestore Database

- Perform CRUD operations (Add, Read, Update, Delete) on cloud data.
- Structure and query your NoSQL database in real-time.

Asynchronous Programming Essentials

- Understanding **Future** and how to work with asynchronous operations.
- Using **async** and **await** for cleaner asynchronous code.
- Handling errors and managing concurrency in network requests.

Consuming External APIs with Dio & Retrofit

Learn how to interact with third-party services to enrich your applications with real-time data and functionality.

Dio

- Robust HTTP client for making network requests.
- Handle interceptors, error handling, and more.

Retrofit

- Type-safe HTTP client for REST APIs.
- Simplify API declarations with annotations.

Hands-on Project: Begin developing a Doctor Appointment App, integrating Firebase for authentication and profile management, and using Dio/Retrofit for booking appointments from a mock API.

Ready to Build Amazing Flutter Apps?

This bootcamp has equipped you with the tools and knowledge to tackle complex Flutter projects. Keep exploring, keep building, and keep innovating!