Tech Stack

Full Stack (Web)

- **MERN** (MongoDB, Express.js, React, Node.js): Ideal for building JavaScript-based applications, offering seamless data flow with JSON between client and server.
- **MEAN/MEVN** (MongoDB, Express.js, Angular/Vue, Node.js): Another JavaScript stack, suited for building dynamic web apps with Angular/Vue.
- **LAMP** (Linux, Apache, MySQL, PHP): A classic stack that supports web apps with PHP on the backend and MySQL for databases.

Frontend Stacks (Web)

- **React** (JavaScript): Paired with tools like Webpack, Babel, and CSS frameworks (Bootstrap, Tailwind CSS), it's versatile for building dynamic UIs.
- **Vue.js**: Known for simplicity and speed, often combined with Vue Router, Vuex, and a CSS framework like Vuetify.
- **Angular**: A complete framework with built-in state management, routing, and templating, making it ideal for large applications.

Backend Stacks (Generic)

Many modern backend stacks are designed to be headless, meaning they expose data through APIs, allowing any frontend to consume it independently:

- **Node.js + Express.js**: A popular JavaScript-based backend stack that can serve both websites and APIs. Combined with MongoDB, it enables highly scalable applications.
- **Ruby on Rails**: A Ruby-based backend known for rapid development. It can serve dynamic web pages or expose an API for other frontends.
- **Flask**: A lightweight Python framework, often used with SQLAlchemy for data handling and Flask-RESTful for building APIs.
- **Spring Boot**: A Java-based framework, popular for building secure, high-performance web apps and APIs.

Android Stacks

- **Native Android**: Built using **Java** or **Kotlin**. Android Studio is the official IDE, offering tools for UI design, testing, and debugging. Native development provides access to Android-specific features, like Google's Firebase services, sensors, and low-level device APIs.
- **Cross-Platform (Kotlin Multiplatform)**: With **Kotlin Multiplatform**, developers can share business logic across Android, iOS, and even other platforms, writing native UI for each. This allows sharing code without sacrificing the native feel.
- **Flutter**: A UI toolkit from Google, written in **Dart**, allowing Android and iOS apps to be developed from a single codebase. Flutter provides a rich set of customizable widgets and tools, making it suitable for high-performance applications.
- **React Native**: Built on **JavaScript** with React, allowing Android and iOS development from a single codebase. Although not as performant as native Kotlin or Java, it provides fast development cycles and a rich ecosystem of libraries.

iOS Stacks

- **Native iOS**: Developed using **Swift** or **Objective-C** in Xcode, Apple's official IDE. Native iOS apps allow access to all iOS-specific APIs, ensuring optimal performance and integration with Apple's services (such as iCloud, CoreML, and HealthKit).
- **Cross-Platform (SwiftUI with Kotlin Multiplatform)**: With **SwiftUI** for the UI and **Kotlin Multiplatform** for shared business logic, developers can target iOS alongside Android while preserving a native look and feel.
- **Flutter**: Flutter allows iOS development with Dart from a single codebase shared with Android. Its fast rendering engine offers close-to-native performance, especially for complex animations.
- **React Native**: Provides a shared codebase for iOS and Android using JavaScript. It supports custom native modules to access iOS-specific features when needed, giving more flexibility in mixed-platform environments.

Desktop Stacks

- **Electron**: Allows developers to build cross-platform desktop apps with **HTML, CSS, and JavaScript**. Popular applications like VS Code and Slack are built with Electron, making it a top choice for desktop apps with a web-like experience.
- **.NET (Windows Forms or WPF)**: For Windows applications, **C#** and .NET with Windows Presentation Foundation (WPF) are well-suited for high-performance, native applications with rich UIs.
- **Qt**: A cross-platform framework written in **C++** that can target Windows, macOS, and Linux. Qt is known for its performance and native look on each platform, commonly used for software needing high efficiency.
- **Flutter for Desktop**: Flutter also supports macOS, Windows, and Linux, offering a consistent UI across platforms. It's beneficial for apps that target desktop and mobile with a shared codebase.
- **JavaFX**: Built on Java, JavaFX is suitable for creating cross-platform desktop applications with a robust, native-like UI. It's especially popular for enterprise software where performance is critical across multiple operating systems.

Alternatives for Django

If you have previous experience in working with Django (i.e. Python based API frameworks), consider the following alternatives:

Flask

https://flask.palletsprojects.com/en/stable/
https://github.com/humiaozuzu/awesome-flask

If you used Django to just build an API, you can use Flask for the same purpose.
If you used Django to build and serve the web application, you can use Flask for the same purpose as well.

Tutorials:

- https://youtu.be/zsYIw6RXjfM?si=H8HssfMDn0S1m47Z
- https://www.youtube.com/playlist?list=PL-osiE80TeTs4UjLw5MM6OjgkjFeUxCYH

FastAPI

https://fastapi.tiangolo.com/
https://github.com/mjhea0/awesome-fastapi

If you used Django to just build an API, you can use FastAPI for the same purpose.

Tutorials:

- https://youtu.be/tLKKmouUams?si=kdEdrbjok0G5PE1_
- https://fastapi.tiangolo.com/tutorial/

MERN Stack

Written

- https://fullstackopen.com/en/
- https://react.dev/learn
- https://code.tutsplus.com/getting-started-with-mongodb-part-1--net-22879t
- http://expressjs.com/en/starter/installing.html
- https://github.com/max-mapper/art-of-node#the-art-of-node

Awesome MERN

Awesome (Collection of Resources): https://github.com/iamzeng/awesome-mern-stack
Please refer to this for information on libraries, components, etc.
Each of the individual Awesome on Mongo React etc. have a lot of resources

Video Tutorials

The following tutorials teach the MERN stack as a whole

- https://www.youtube.com/watch?v=98BzS5Oz5E4&list=PL4cUxeGkcC9iJ_KkrkBZWZRHVwnzLIoUE
- https://www.youtube.com/watch?v=CvCiNeLnZ00
- https://www.youtube.com/watch?v=mrHNSanmqQ4
- https://www.youtube.com/watch?v=xDCKcNBFsuI

This tutorial has Express, MongoDB and Node without React, so you can refer to this for both MERN and MEVN

- https://www.youtube.com/watch?v=-foo92lFIto&list=PL4cUxeGkcC9hAJ-ARcYq_z6lDZV7kT1xD

Interactive/Coding Based Tutorials

- https://nodeschool.io/

MEVN Stack

Generally follow the MERN tutorials, but replace React with Vue.js

Written

- https://vuejs.org/guide/introduction.html#getting-started
- https://signoz.io/blog/mevn-stack-tutorial/

Video Tutorials

Please note that the following one doesn't use MongoDB, but uses MySQL. You can choose to use MySQL but relational databases have some limitations that non-relational databases like MongoDB do not have. So technically it isn't fully MEVN, but you can learn Express, Vue and Node from this tutorial.

- https://www.youtube.com/watch?v=Fa4cRMaTDUI&list=PLWKjhJtqVAbnadueQ-C5keMQQiQau_i0D

Pure Vue tutorial

- https://www.youtube.com/watch?v=YrxBCBibVo0&list=PL4cUxeGkcC9hYYGbV60Vq3IXYNfDk8At1

TALL Stack or Laravel + Frontend Stack (Frontend library of your choice: React, Vue, HTML/CSS)

Written

- https://laravel.com/docs/11.x

Awesome

- https://github.com/chiraggude/awesome-laravel

Video Tutorials

- https://code.tutsplus.com/learn-laravel--cms-93248t

General Laravel + Frontend

- https://www.youtube.com/watch?v=iFOEU6YNBzw
- https://www.youtube.com/watch?v=qJq9ZMB2Was

TALL Stack

- https://www.youtube.com/watch?v=Ul3sfSDEt9U

Android

Java/Kotlin as Frontend (API Tutorials at the end)

- Frankly the best one - https://developer.android.com/courses/android-basics-compose/course
- https://github.com/JStumpp/awesome-android

Video Tutorials

- https://www.youtube.com/watch?v=BCSlZIUj18Y
- https://www.youtube.com/watch?v=kNghEbknLs8

React Native as Frontend (API Tutorials at the end)

- https://github.com/jondot/awesome-react-native

Written

- https://reactnative.dev/docs/environment-setup
- https://reactnative.dev/docs/tutorial

Video Tutorials

- https://www.youtube.com/watch?v=ur6I5m2nTvk&list=PL4cUxeGkcC9ixPU-QkScoRBVxtPPzVjrQ

Flutter as Frontend (API Tutorials at the end)

- https://github.com/Solido/awesome-flutter

Written

- https://docs.flutter.dev/get-started/codelab
- https://code.tutsplus.com/design-your-first-flutter-app--cms-37054t

Video Tutorials

- https://www.youtube.com/watch?v=1ukSR1GRtMU&list=PL4cUxeGkcC9jLYyp2Aoh6hcWuxFDX6PBJ
- https://www.youtube.com/watch?v=VPvVD8t02U8

API

RESTful API

Express + Node

Check some stuff from MERN stack portion too

- https://www.moesif.com/blog/technical/api-development/Rest-API-Tutorial-A-Complete-Beginners-Guide/
- https://blog.postman.com/how-to-create-a-rest-api-with-node-js-and-express/
- https://www.youtube.com/watch?v=b8ZUb_Okxro
- https://www.youtube.com/watch?v=nJMOTG-dfQ0

Laravel

Check some stuff from Laravel portion too

- https://www.toptal.com/laravel/restful-laravel-api-tutorial
- https://www.youtube.com/watch?v=YGqCZjdgJJk

Graphql API (GraphQL + Node)

- https://graphql.org/learn/
- https://www.youtube.com/watch?v=xMCnDesBggM&list=PL4cUxeGkcC9gUxtblNUahcsg0WLxmrK_y