Decomposition cheatsheet

Choosing a layer

Shared

Code that is not specific to your application, code that serves as a foundation.

Self-check question

Can this code be used in a pizza shop app or an online bank?

Example: a dropdown menu can appear in a pizza shop app, a social media post probably can't.

Entities

Code that represents a real-life concept that your app is working with.

Self-check question

When describing your app, does this word appear as a subject or an object? Do your users/clients understand that word?

Example: users can write posts. Clients want to be able to add videos to their posts.

Features

Interactions that provide real-life value to your app's users, the things people want to do with your entities.

Self-check question

When describing to a stranger what your app does, do you mention these actions?

Example: users can write and edit posts. Posts can be configured to auto-delete after 5 minutes.

Widgets

Code that combines the layers below to form meaningful blocks, interactive and complete with data.

Self-check question

When looking at your app's UI from a distance, does this stand out as a complete "block"?

Example: A list of posts with pagination and the header appear as standalone blocks.

Pages

Entire screens of your application, built mostly by combining the layers below. Similar to widgets, but on a larger scale.

Self-check question

Is this code ready to be plugged into the router and work for users as is?

Example: the home page of an online shop with login, fresh deals, categories, search, etc.

App

Infrastructural code that makes your app actually work.

Self-check question

Is this something your framework or technical stack needs for your app to function?

Example: an i18n provider and a router make the app work and display sensible text to the user.