SOLO PROJECT

SOLO PROJECT ORIENTATION GUIDELINES, SUGGESTIONS, TIPS AND RESOURCES

> Version: 1.0.2

DESIGN PROCESS

1. Start by sketching out your application in a notebook or a whiteboard. This high level overview will help you define the different parts of the app and their responsibilities using a high level of abstraction. At this point, you shouldn't be thinking in code. Plain logic and common sense is all you need to set up a basic diagram/skeleton for your app. You can use Excalidraw for creating and sharing this diagram: https://excalidraw.com/. Other alternatives, include Figma, Sketch, InVision and others.

2. Once you've passed this stage, you are ready to think in terms of Modules/Components that make up the parts of your application and organize them in files and folders.

3. At an even lower level, you will go into these Modules/Components and start writing the code to implement their functionality. You might split these parts into even smaller, simpler and more manageable parts to keep things simple. Remember: Complexity is your enemy. Whatever reduces complexity, makes your life easier and reduces the possibility of introducing bugs, security issues and unintended side-effects.

PROJECT SETUP & DOCUMENTATION

1. Include all necessary setup instructions on both ends (backend/frontend) in short and easy-to-follow markdown files (preferably README.md). If you are using a bootstraping framework like Create React App, remove the default README.md content and replace it with your instructions. The files should include only the necessary information to have the application up and running on any local system (e.g. start MongoDB, install dependencies using npm install, run any other custom npm scripts, preferred ports, etc.). See an example README here. Your markdown can be even shorter and simpler.

2. Use npm scripts to populate the database with some sample data for someone to test. For example, if you are building an app that finds apartments, we want to be able to execute an npm script that will populate (seed) the database with a few entries so that we can quickly test the application withour having to manually enter data. You already saw a custom npm script that populates a database in the security exercise: `npm run db:seed`. You might even include a script to reset the database in case things get messy.

DATABASE

1. Create a diagram of your Database (entities, relationships, table or document structure) with one of these tools. It will help us review the database design and schemas and it will help you go from these ERD diagram straight into code much easier.

- https://app.dbdesigner.net/
- https://app.sqldbm.com/
- https://www.lucidchart.com/pages/

2. For relational databases, make sure your database enforces the basic normalization rules (1NF, 2NF, 3NF). You can study more about Database Normalization here: https://www.geeksforgeeks.org/introduction-of-database-normalization/

3. Designing your Database

    - Read: The 4 Phases of designing a Database
    - Watch 7min (for Relational DB's)
    - Watch 5min (for Relational DB's)
    - Extra: Watch 7min (for Relational DB's and ERD Diagrams)

STACK

1. Prefer libraries/frameworks/technologies we have covered in the lectures. Strengthen your skills on these technologies and let us help and support you in the most efficient manner. If you feel adventurous, don't stray too much from the course stack. You might pick a new CSS library or a third-party React component, but switching to a stack that is completely different from the course (e.g. Vue.JS + Next.JS + Firebase) is not recommended.

2. When picking a third-party module, library, plugin, npm package, make sure to check:

- Stars (prefer repos with at least a two-digit number of stars, but there are rare cases of small handy utilities with fewer stars)
- Issues (both open and closed, that might be related to the issues you are having with the package)
- Last update date
- DON'T pick deprecated or outdated/outlived packages
- Carefully read the documentation and study the examples
- Google for issues or problems that might arise when combining the package/library with another framework:
    - google: vue and redux (in case you want to use vue with redux)
    - google: expressjs and aws
    - google: react native and react-carousel-component
    - ...you get the idea If your implementation does not work, put the documentation examples side by side with your code and check to see if there's a mismatch. Start by copying and running the exact same code that you'll find in the documentation (almost all of the packages include a quickstart or basic example snippet), and once it's working, try refactoring (in a step-by-step-and-test process) the example code to fit your needs.

3. Leverage the power of third-party React/Angular Components if you want to easily implement certain features, e.g. charts, maps, tag-inputs, carousels, etc.

REACT

- If you are working with React Router:
    - Use the latest version: **v6**
    - Make sure you know about advanced routing options such as `(1) Nested Routes` and `(2) Protected Routes` that will help you (1) structure your routes in an organized and efficient way and also (2) divide your routes between public and private pages
    - This YouTube video provides a nice reference. **Scroll down to the comments** to find a `Table of Contents` (by the user `Rahul Rawat`) which you can use to quickly jump to the section of

interest. Bookmark this video for future reference also.
    - When searching on the Internet, add the `+v6` keyword to ensure that you are seeing results for the latest version and not v5 which include breaking changes

PRODUCTIVITY

1. Use the Pomodoro Technique to keep track of the time it takes to complete each task
2. Share insights and information about common technologies with your fellow students. For example, you might want to share information about geolocation or some maps API in case some of you want to integrate the same technology into your apps.

UI/UX

1. Spice up your UI with beautiful free stock photos from:

- Pexels
- Unsplash

Cool illustrations from: Undraw

And some awesome animations available from https://lottiefiles.com

2. Make use of CSS libraries like Bootstrap and Tailwind or CSS UI Kits like `Material Design` to quickly have a beautiful frontend setup, if you need to invest time on the actual functionality. You can customize the CSS as much as you can, once you have the basic MVP functionality on schedule. For example, see: https://mdbootstrap.com

PERFORMANCE

1. Use a Tool like ImageOptim (Mac) or Bamboo, or a free service like TinyPNG to decrease the file size of your images.