Take-home Assessment - Rubric

Category / Description	Needs improvement (below 40%)	Satisfactory (40-60%)	Good (60-75%)	Very Good (Top 25%)	Excellent (Top 10%)	Outstanding (Top 5%)
	What are we looking for?					
Correctness	● Passes all of our tests, and shows great attention to detail ● Goes above and beyond the requirements, such as handling validation of improper inputs, or completing bonus requirements and stretch goals (like deploying an application).					
Code Organization, Readability & Maintainability	● Code utilizes a consistent design pattern. The code is well organized and would be easy to extend. Design made with consideration to potential future features. ● Code is easy to read - proper indentation, vertical spacing, named constants (no magic numbers), big picture comments, and meaningful variable names. You can investigate using a linter to help make your code properly formatted. ● There are small amounts of repeated code, and is well factored into functions. Related items are grouped into classes (or something equivalent like structs or objects).					
Code Performance - Efficiency, Data Structures & Algorithms	● Code that is performant (has a good time and space complexity for the problem given). Avoids unnecessary nested for loops, and uses the appropriate data structures and algorithms. ● Efficient code that reduces unnecessary code. ● Implements additional features to improve code performance such as (but not limited to) caching, parallel API requests, reducing re-rendering on the front-end, memoization, and other code performance techniques.					
Best Practices & Proficiency in Language/ Framework	● Utilizes great programming practices such proper unit testing, reduced warnings, and defensive coding strategies (validation, asserts). ● Demonstrate a good grasp of the language and/or framework. Avoids major anti-patterns in the project (using let vs. const in Javascript, modifying the DOM in React, etc.) ● Utilizes the most advanced features in the framework (for example using functional components, Hooks in React).					
Completion Speed	● A fast completion time. This category is the least important category, so prioritize it the least. We will consider speed with respect to the complexity of your solution. You will receive a low mark in completion if you completed the application slowly and your solution is not complex.					