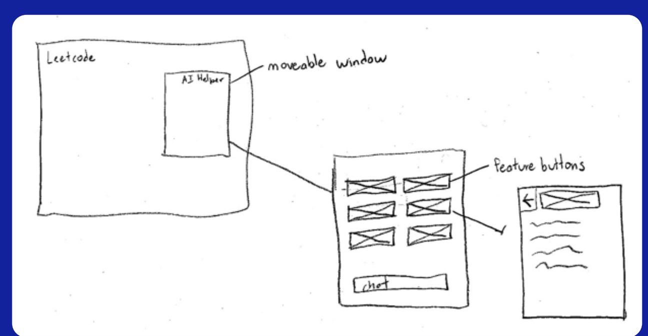
AI TUTOR : A BROWSER EXTENSION FOR SOLVING CODING PROBLEMS

Team 12:
Nolan Whittaker
Steve Duong
Tony Huang
Ryan Marwaha

THE INSPIRATION

Based on personal experiences, using AI for help with leetcode problems often inhibits learning because it isn't catered toward any learning process, it simply gives code when the user asks. In the long term, just using AI to generate leetcode solutions isn't an effective or honest way to learn, so this AI coding tutor's aim is to structure the use of AI in a way that assists the learning process. In the early stages, the solution was to use AI to generate visuals, give different levels of hints, recommend online resources, and give full/partial solutions to the problem. These ideas were refined throughout the 4 stages of this project: ideation, interviews and sketches, medium fidelity prototype and finally a high fidelity prototype.



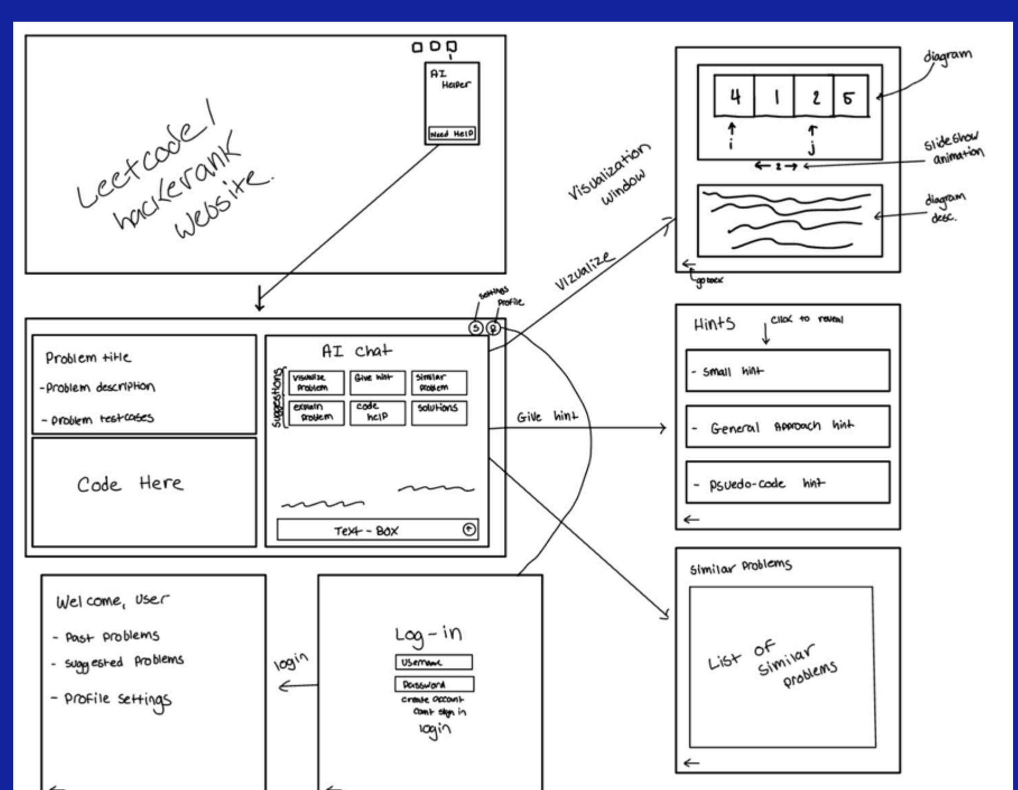
AN EARLY SKETCH

The original idea was to have some type of user interface on leetcode, where the user could press buttons for different features of the app (like hints/resources) and and an AI chat bar.

RESEARCH & USER REQUIREMENTS

Because the inspiration for this app was solely based on personal experiences with AI coding help, running 4 different interviews gave new insight into who the target users would be and filtered out the early ideas into features that should and shouldn't be added. A significant discovery was most of the interviewees (cs/engineering students) used ChatGPT for help, and were sometimes tempted to just get the full solution to the leetcode problem which reinforced the main reason for this app: to assist the learning process.

A SKETCH BASED ON INTERVIEWS



This sketch shows the app implemented where the screen is split between the app and leetcode. Six main functions were decided in this prototype: visuals, hints, similar problems, explanation of the problem, coding help, and solutions.

DESIGN ITERATIONS & MEDIUM FIDELITY PROTOYPE

Cognitive Walkthrough:
The cognitive walkthrough revealed switching back and forth between the application page and leetcode was harming the users workflow as it required them to switch tabs. So, the design was altered to become an pop-up extension to limit the nuisance. Additionally, certain features did not aid the user with their tasks, for example the extra problems, and partial solutions did not aid the user with their current problem and was space-consuming.

- A couple of challenges:
  - Understanding user's ideal learning methods and functionality needed for accessibility.
  - Creating a simple but useful design for the user.

HIGH FIDELITY PROTOTYPE & FUTURE IMPROVEMENTS

- How does this one solve the problem?

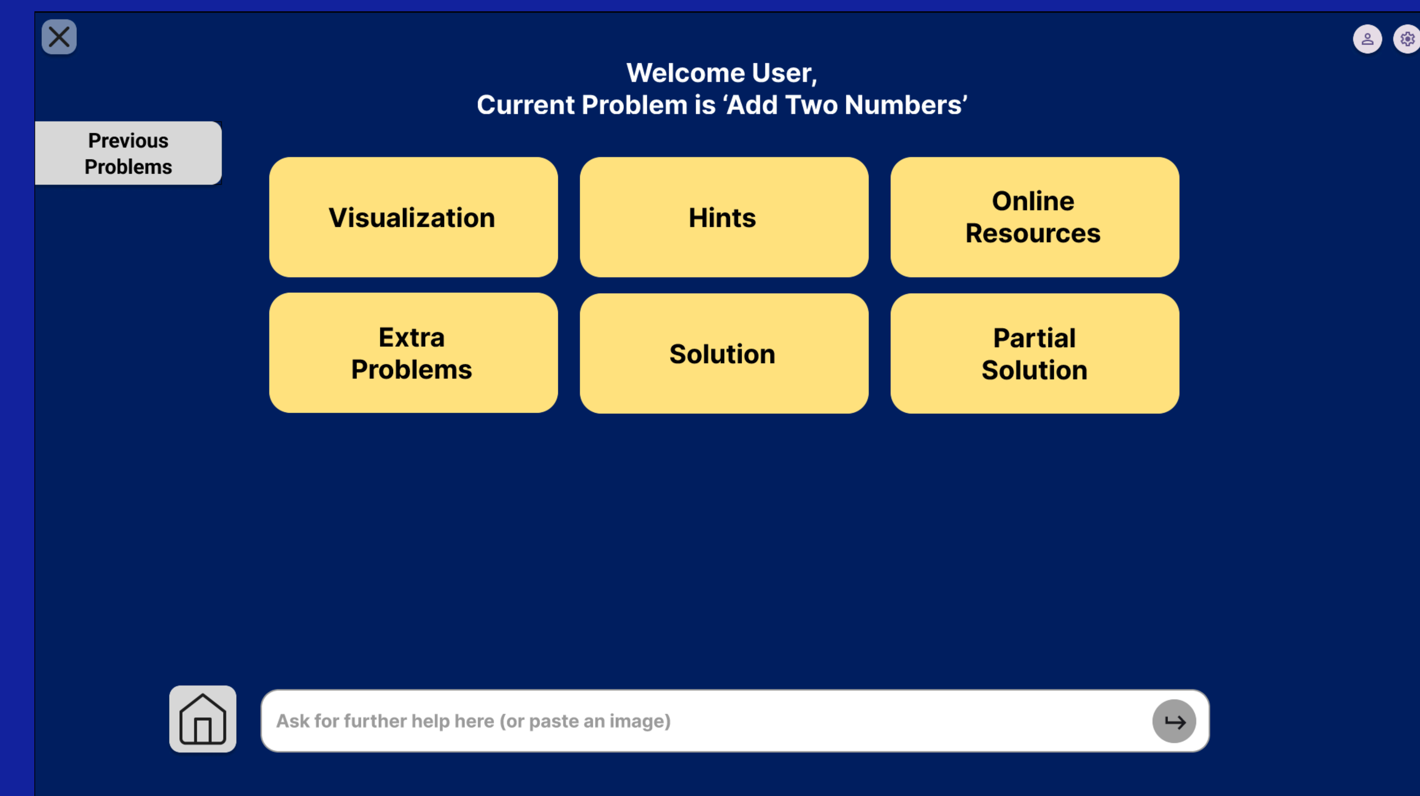
The high fidelity prototype solves issue of the user having to go back and forth by being a browser extension, which when active takes up half the screen. The two features, extra problems and partial solutions have been removed in the higher fidelity prototype. They weren't needed for its core function as an assistant and the partial solutions seemed unhelpful because of the hint function already available.

- Any impacts from this design?

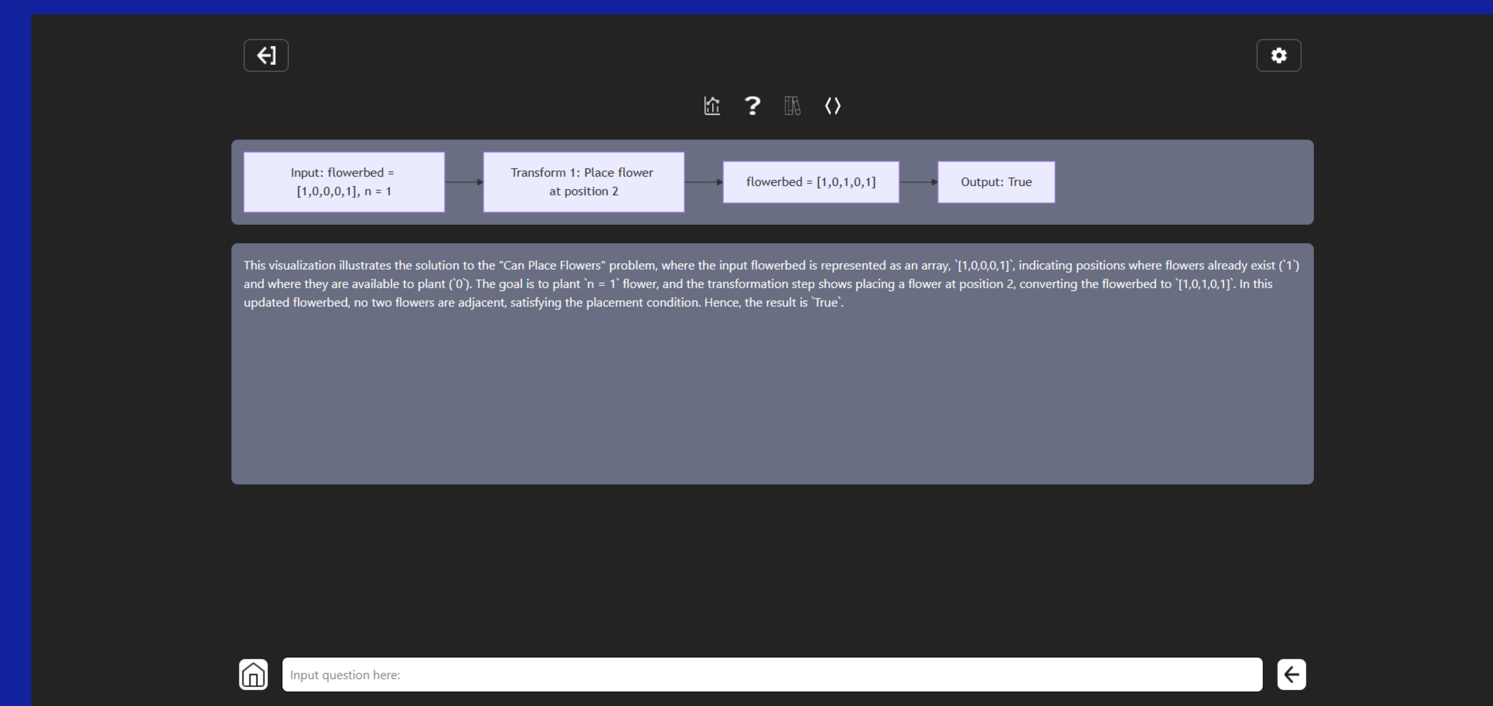
This design improves efficiency by reducing context switching and simplifying the interface. However, one limitation is that users still need to manually enter the link to the leetcode problem they need help with.

- Future improvements.

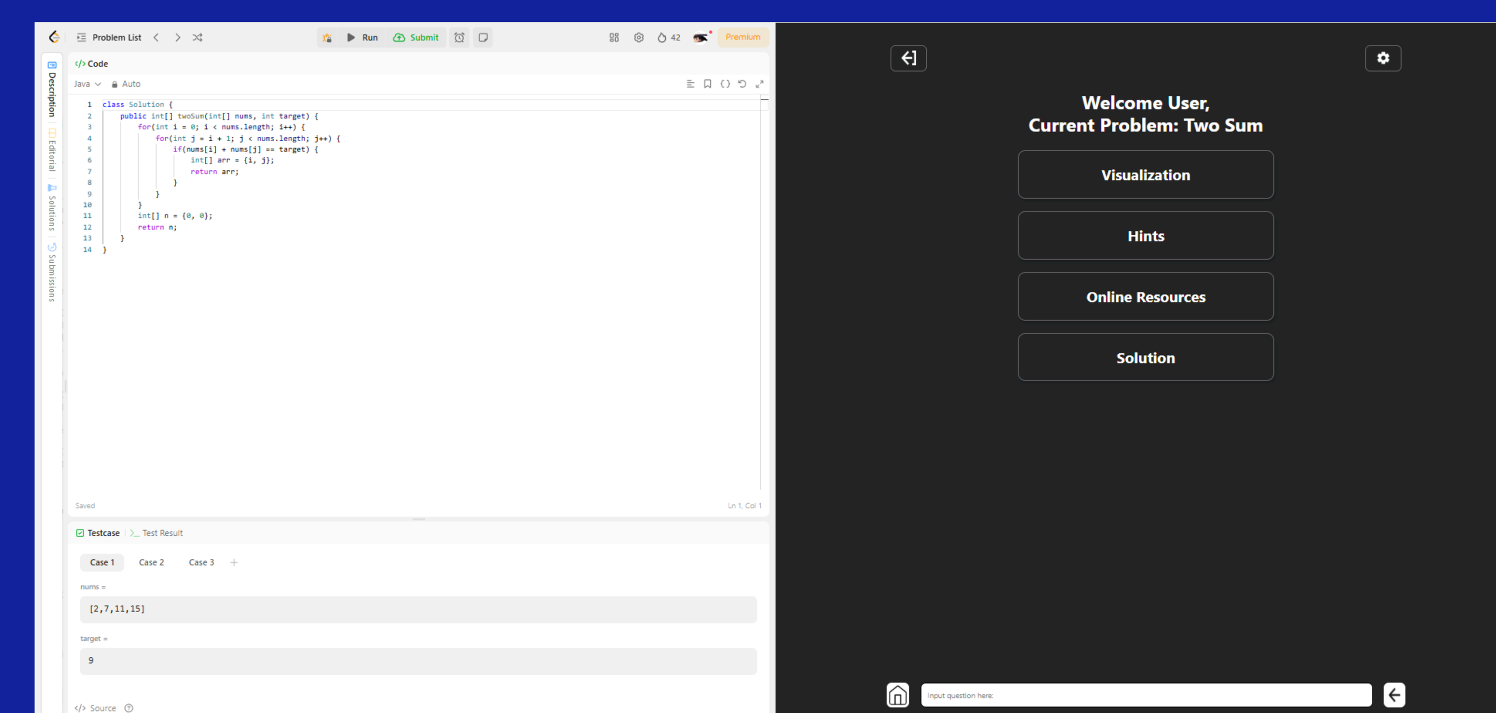
A future improvement would be removing the need for the user to enter the link to the leetcode problem they are getting help from by automatically getting it from the current tab when the extension is active. Similarly, fine-tuning AI responses for the users so visualizations and hints are relevant and useful.



Medium fidelity prototype takes the fullscreen



The visualization section gives the user an idea on the steps for the input and output of a problem



High fidelity prototype implemented as a browser extension. (leetcode left, AI tutor right)