Tutor JS Frontend

VIP Intelligent Tutoring Systems Spring 2021

The Team

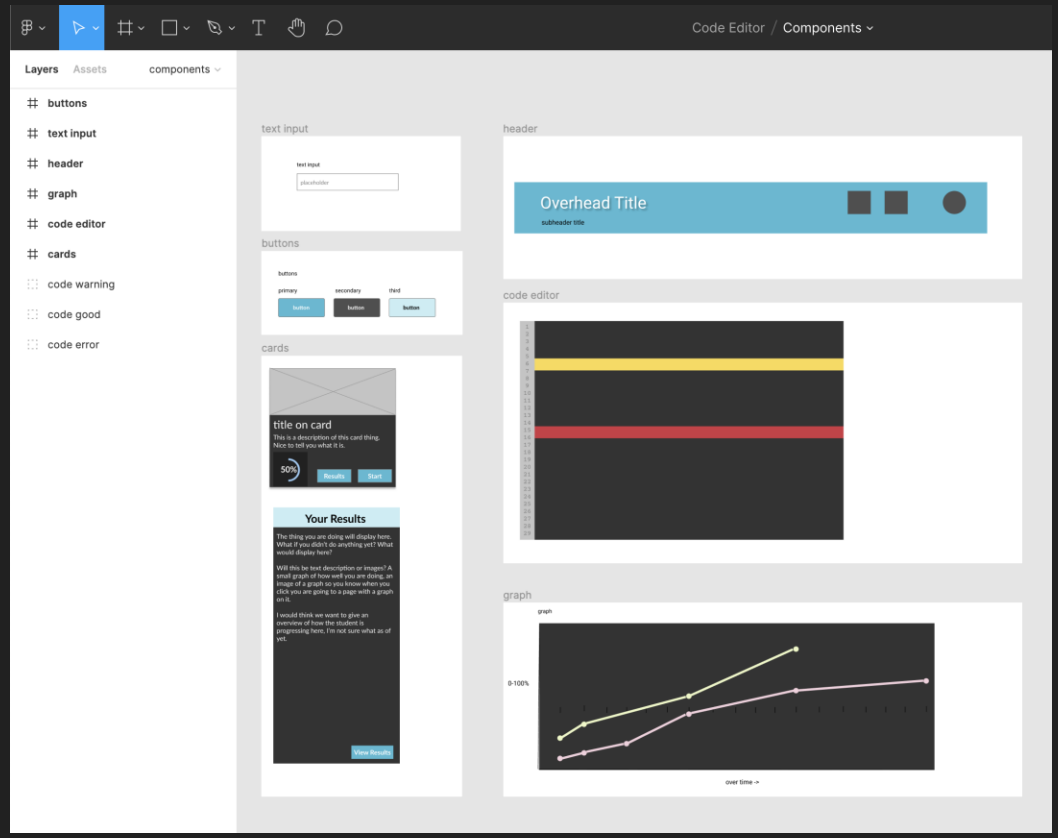
- Project Lead: Lisa Walkosz-Migliacio
- Front End team:
  - Roshni Dhanasekar, Khushi Magiawala, Vikas Barevadia

Motivation

- Create a user interface with challenges that allow students to learn JavaScript in a code editor
- Provide useful feedback on compile and run-time errors in natural and easy-to-understand language
- Save and present user progress to provide a meaningful learning experience

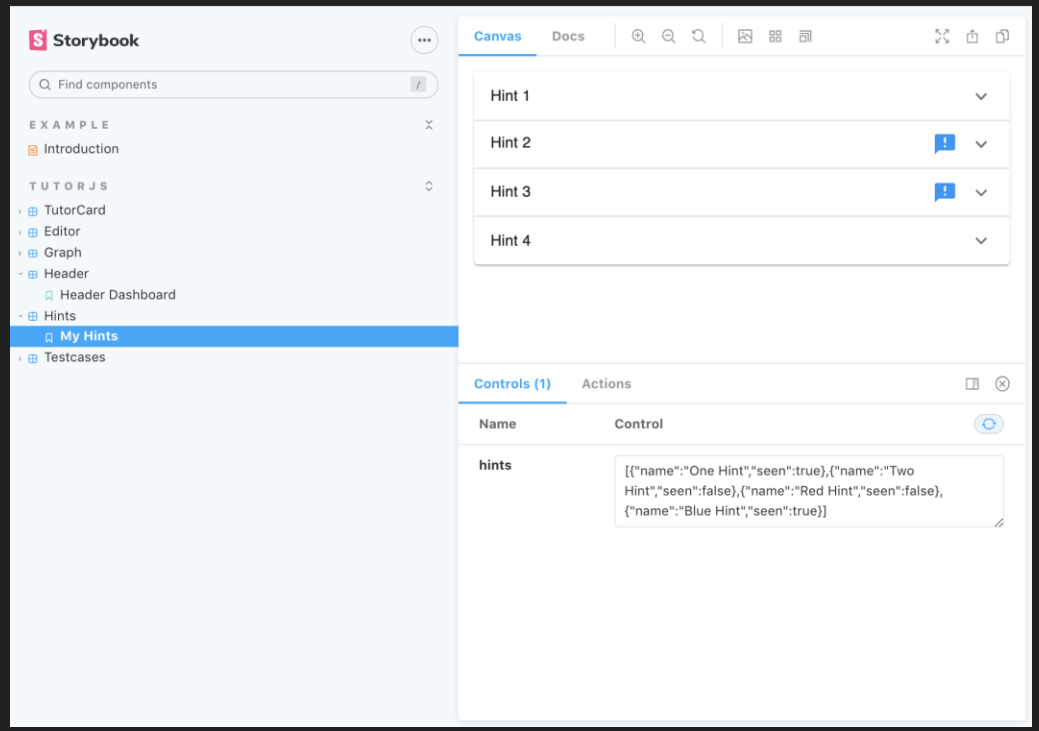
Project Outline

- Research and Needfinding
- **Figma -** To Make Wireframes
- **GitHub Kanban board** – To organize action items for the code editor development



Project Outline

- **React on Node.js** - for the Application
- **Material UI** - for Icons and standard Components
- **Storybook** - for the Components we created for the Application
- **Ace Editor** - for the Code Editor
- **D3** - for the Graphical Results
- **Axios** – to create a service to talk to the API



Research Study Outline

Mission: To measure the success of our application based on the questions we used to guide its functionality

Questions:

- How students best learn a coding language when much of the debugging is in exception errors?
- How students best learn a coding language when given multiple test cases for validation and practicing those failures again?
- How students best learn a mathematical formula from code and graphical visualization?

Things To Test:

1. Effectiveness of Hints
2. Human Readable Error Messages and Stack Trace with Helpful Links
3. Overall UI (Ease of Use) (ex: layout of testcases next to code editor)

Assessing Risk: Emotional Risk (losing confidence in coding ability)

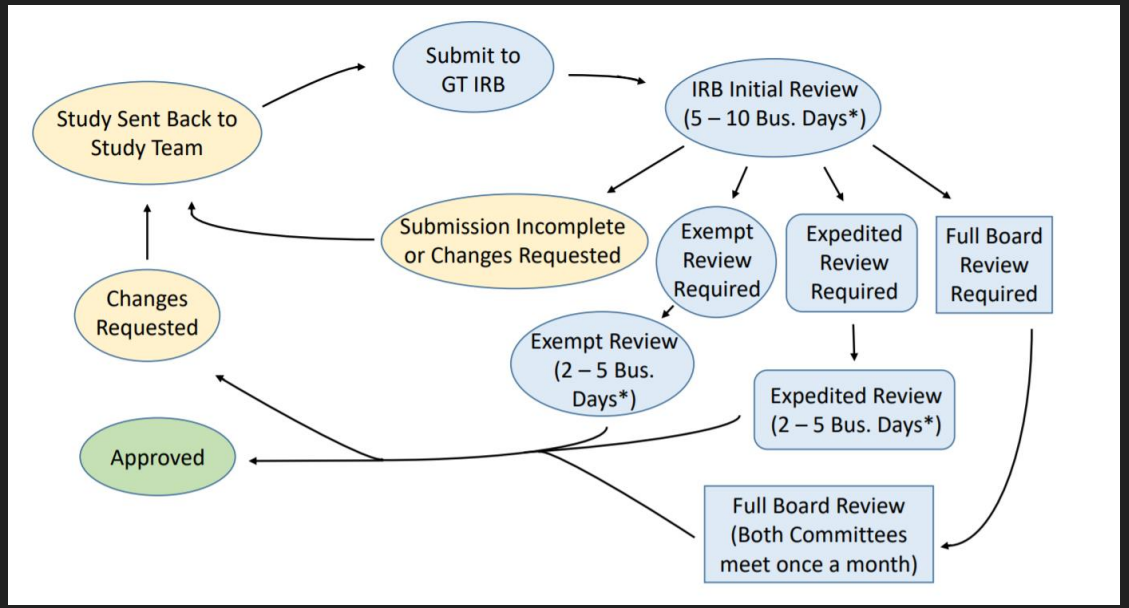
Potential Subject Population: Experienced vs Beginner Coders (establish a baseline), GT Students / ECE Students

CITI and the Institutional Review Board

CITI Relevant Concepts

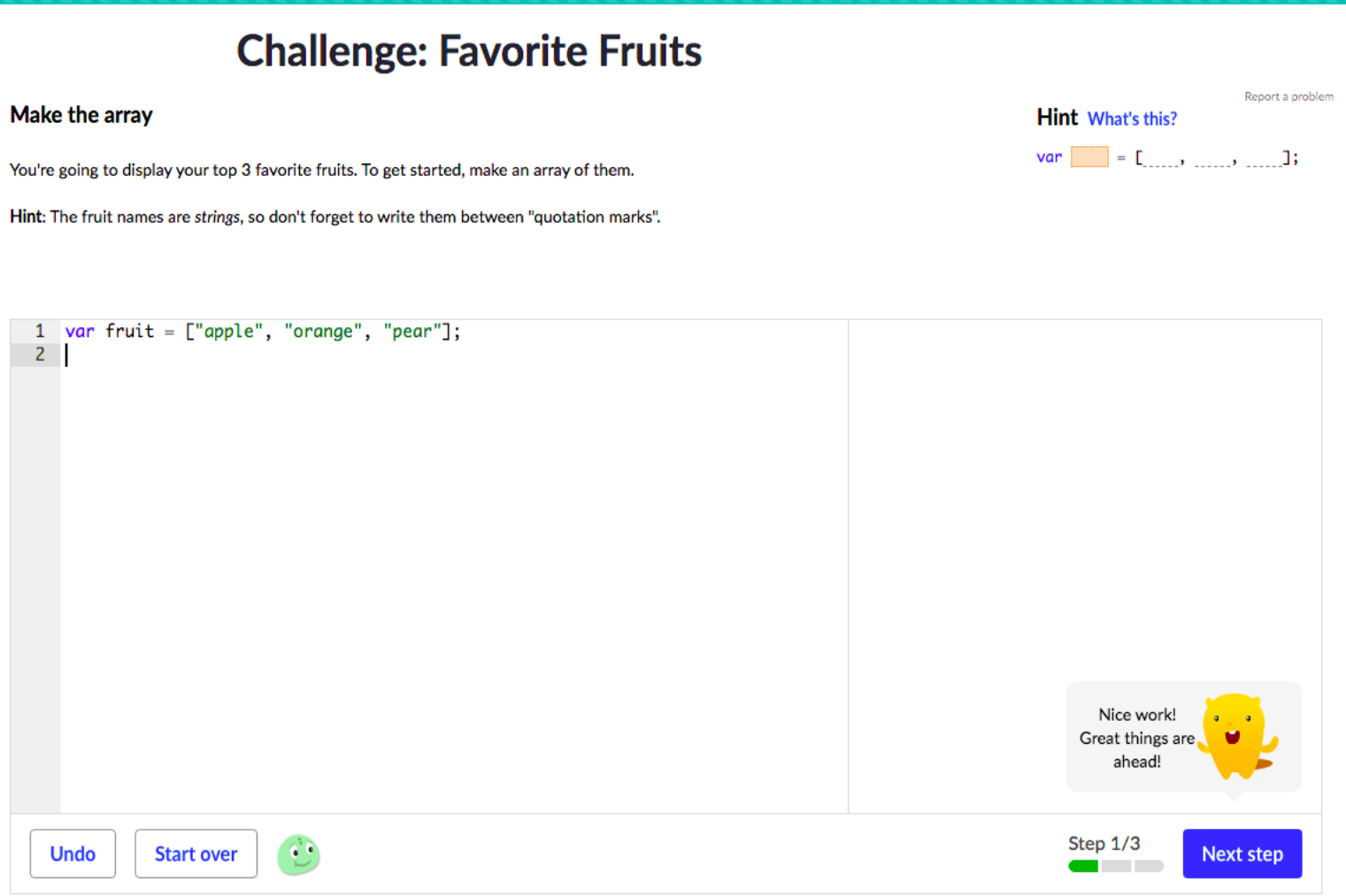
- **Confidentiality:** remove identifiers from study (name, age, major, email) and substitute with participant IDs
- **Informed Consent:** appropriate language level
- **Potential benefits:** helpful testcases > get questions correct > feel more confident in coding ability
- **Minimize emotional risk:** inform about difficulty level and emphasis that this is a test run of a possible solution, not a proven solution
- **Waivers of Documentation:** Study participation presents **minimal risk of harm** to the subject and the research involves **no procedures requiring consent outside the context of participation**

GT Central IRB Protocol



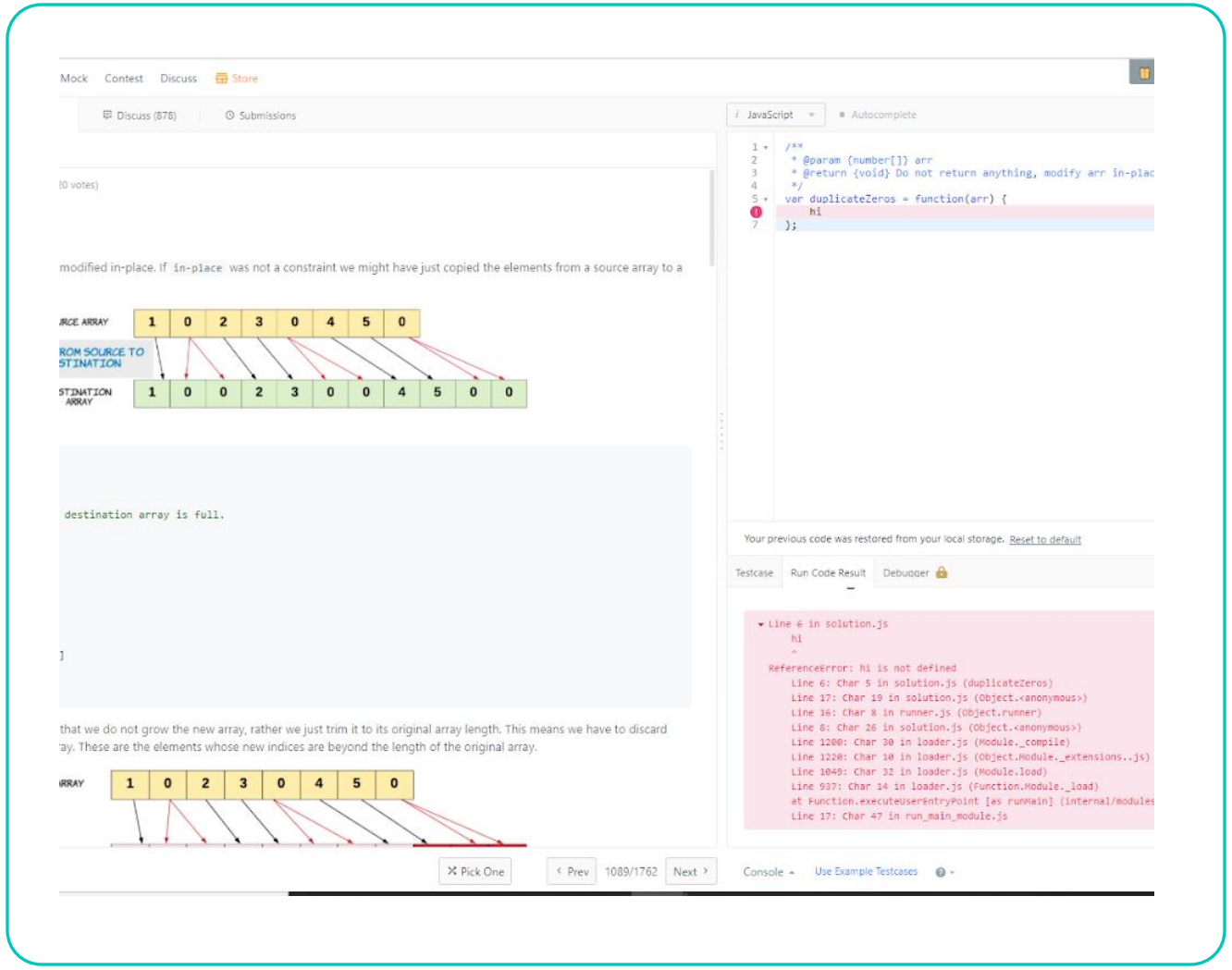
Frontend: Gather Insights

- Gathered insights from various CS problem solving platforms (ex. Khan Academy, Leetcode)
- Analyzed these platforms and formed a list of feature to include in our application



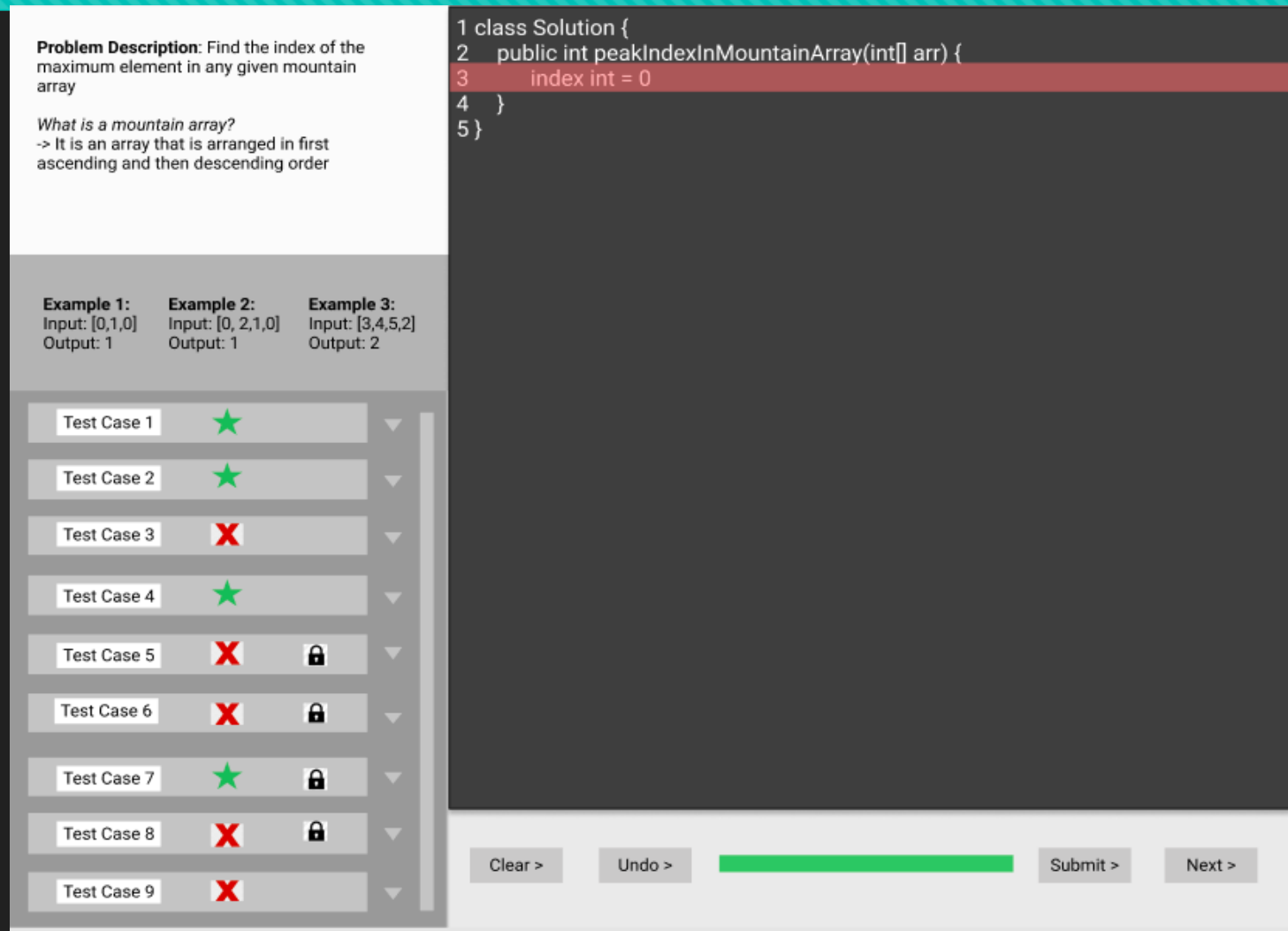
Frontend: Features List

- Main Editor Screen
- Place for Hints
- Description Box + Examples
- Progress Bar
- Submit, Undo, Clear, Next Buttons
- Layout: Horizontal, Side-by-side



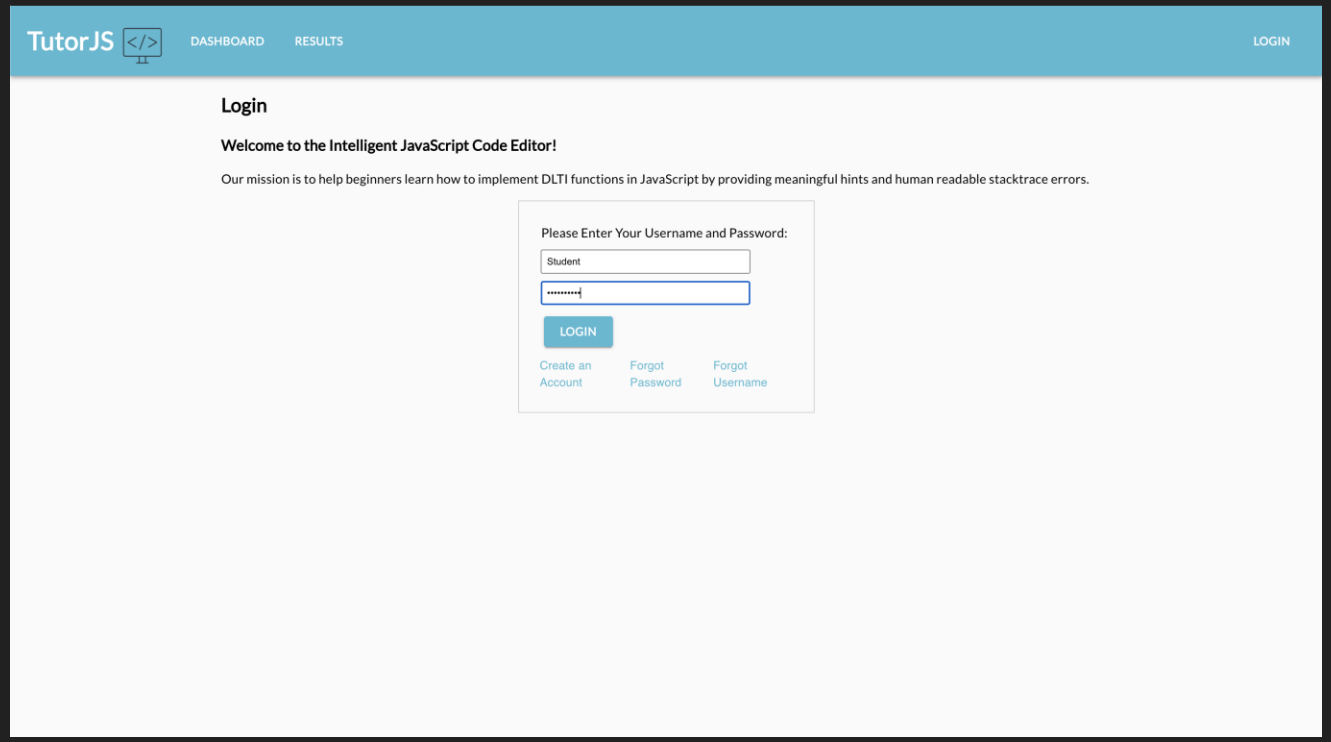
Frontend: Wireframe

- Designed a wireframe for our all pages of our code editor application using Figma
- Took an example problem from Leetcode to outline our problem statement, hints, and test cases



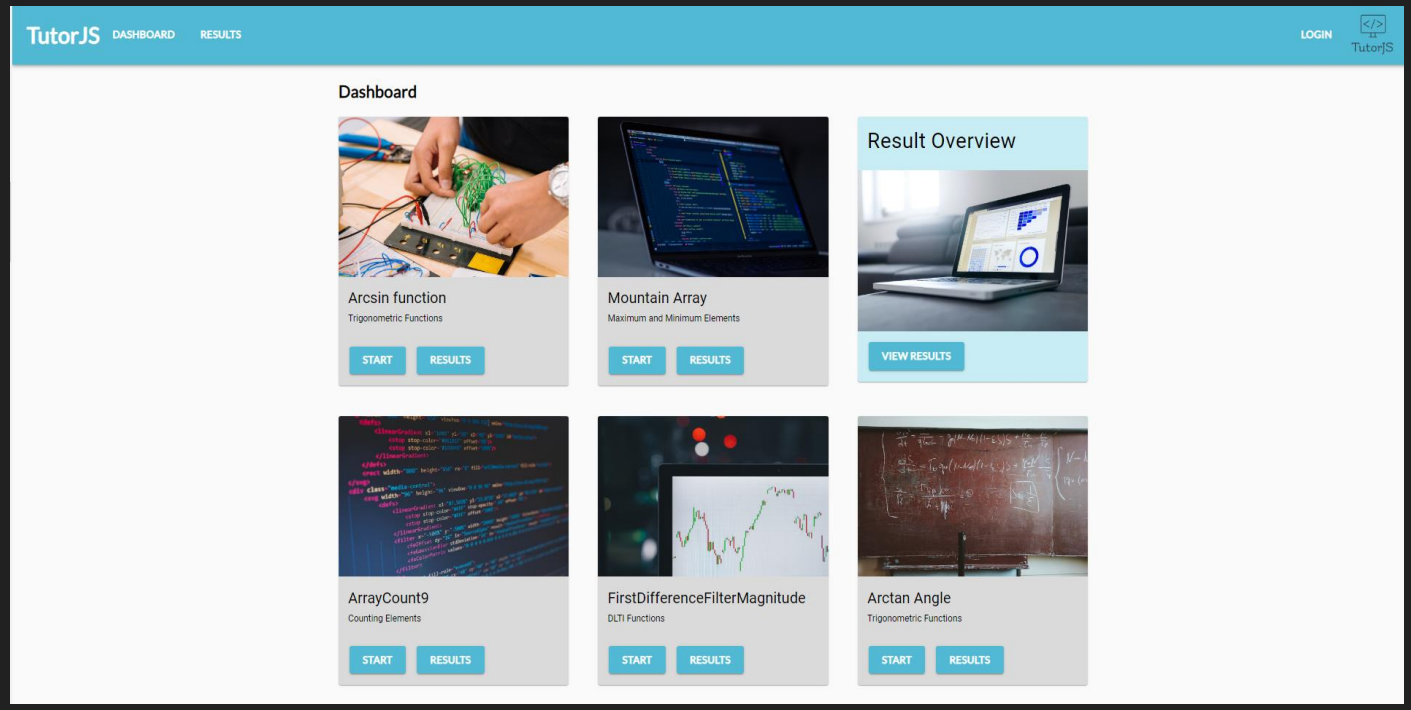
Frontend: Login Page

- Welcomes user to the application with mission statement, login fields and account options
- A username and password can log a user into a session until they logged out keeping their work associated with their userid.



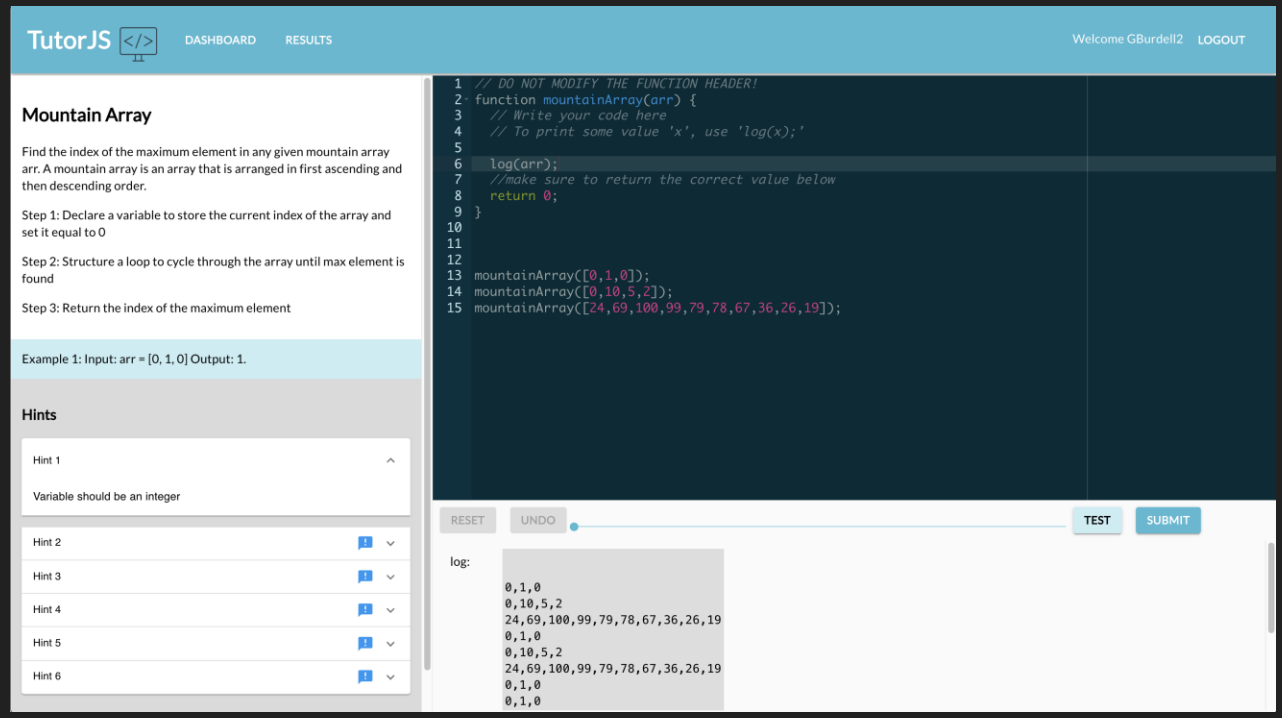
Frontend: Dashboard Page

- Found other example problems on LeetCode and incorporated it into our code editor format
- Developed a Dashboard page that links to each challenge and completion results



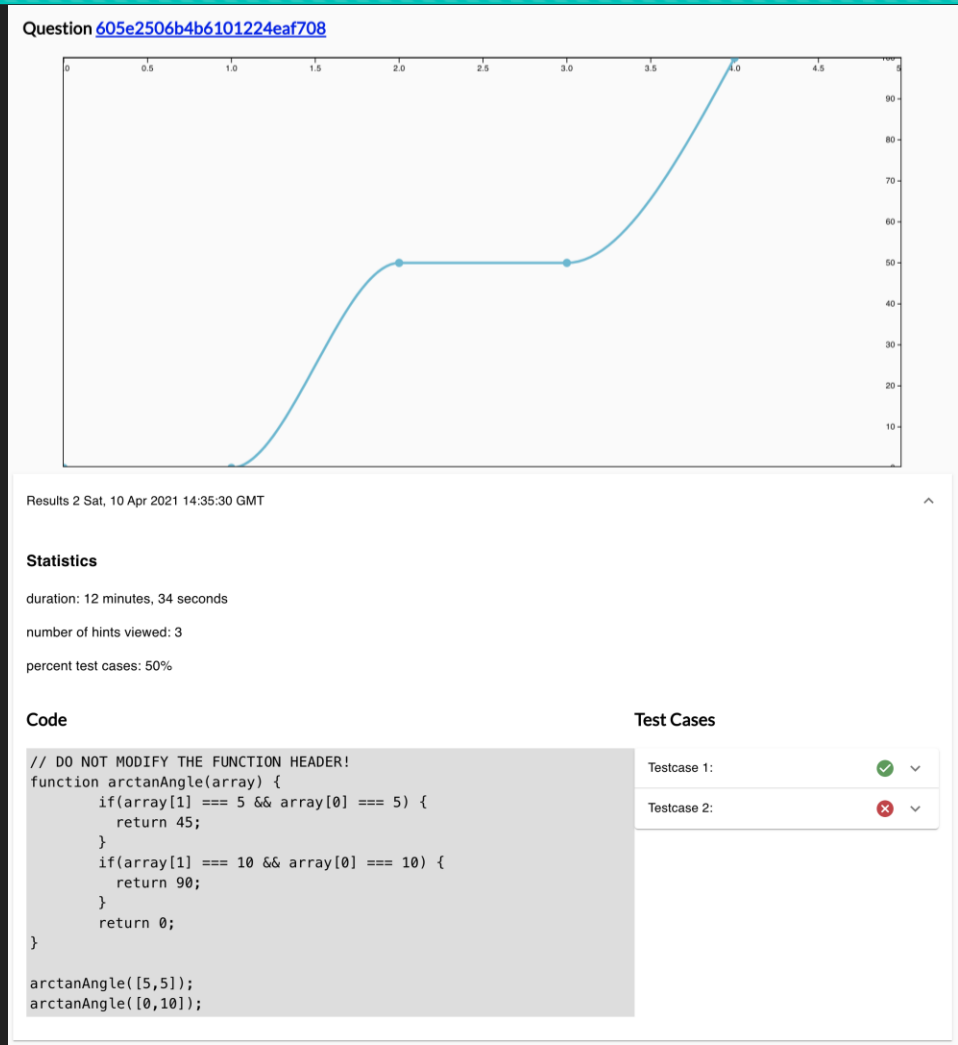
Frontend: Code Editor Page

- Created a functioning code editor that provides live feedback for syntax errors
- Learning hints, logging, and executed testcases results can be used when solving problems



Frontend: Results Page

- Graphical representation of submissions over time.
- Results report of the code, duration, hints, and test cases that passed or failed as well as error messages that were encountered.



Future Features Frontend

- Currently using js-interpreter, which only runs vanilla JS
  - Look into other Ace Editor compilers, such as BabelJS
- Display all errors on respective lines in code editor
- Clean up and format the test cases

DEMO