Frontend Engineer Screen Questions

The structure of these interview questions reflects some of the real challenges we face in our application today. While writing full code isn't required—we're not testing for syntax knowledge—explaining your approach clearly and including code snippets where helpful is encouraged. If you're unfamiliar with Vue, feel free to reference related concepts in React. That said, we're ultimately looking for developers who are excited about working with Vue.

Question #1

Consider this code snippet that uses Vue 3 and Pinia for state management:

```
import { useUserStore } from '@/stores/user'

export default {
  setup() {
    const userStore = useUserStore()
    const { username, email, isAdmin } = userStore

    function updateUsername() {
      userStore.setUsername('newUsername')
      console.log(username) // What will this log?
    }

    return {
      username,
      email,
      isAdmin,
      updateUsername
    }
```

```
  }
}
```

- What issue exists in this code?
  - What will be logged to the console when `updateUsername()` is called, and why?
  - Provide at least two different ways to fix this issue while maintaining the destructuring pattern.
- Most importantly: how would you explain this concept to a junior developer who is new to Vue 3 and Pinia?

Question #2

We've built a financial application that displays stock information to users.

Let's focus on the ability to create a **dashboard**—similar to tools like Tableau—where users can place multiple charts and/or tables on a single page.

- Users can add one or more **widgets**, which act as containers for these charts or tables.
- Each widget can have one or more **views**, which represent specific states of the chart or table (e.g., a selected date range). These states are stored within the view object.

In summary:

An application contains dashboards → dashboards contain widgets → widgets contain multiple saved views.

We want to track changes made to a dashboard or widget, so we can visually indicate to users when something needs to be saved—and specifically *what* needs to be saved.

How would you implement this in terms of code structure? For example, would you pass a saving prop around to different components?

Please describe your strategy for managing application state in a large-scale Vue 3 application with multiple interconnected components and views.

Answer this question with either code snippets or with a functional MVP.

Question #3

In the application described above, imagine you want to allow users to see each other's changes in real time—similar to the "multiplayer" functionality in tools like Google Docs or Figma, where users can view edits as they happen.

How would you implement this feature? You may suggest any third-party libraries or state management tools as needed, but assume Supabase (PostgreSQL) is used as the backend database.