Senior Frontend Technical Test: Multi-Step User Onboarding Flow

🧩 Scenario

You're building a **multi-step onboarding flow** for a SaaS dashboard using **Vue 3**. The process includes collecting user and business information, verifying the user's identity, and mocking submission of the data to a server.

👣 Steps in the Flow

1. **Personal Details**
    - Name
    - Email
    - Phone number
    - Profile image upload (JPG/PNG, max 2MB)
2. **Business Details**
    - Business name
    - Business Logo
    - Industry (populated by mock API)
    - Company size
    - Document upload (PDF only, e.g., license, max 5MB)
3. **Verification & Summary**
    - Simulate sending and entering an **email verification code**
    - Visualise a **summary** of all collected data and its submission to the mock API
    - Display a success message

✅ Requirements

1. Componentization & Prop Contracts

- Each step is a **standalone component**.
- Use **props** and **emits** to pass data, trigger actions and manage state updates.
- The parent handles:
    - Current step
    - Aggregated data
    - Navigation

2. State Management & API Integration

- Use **Pinia** (or Vuex) to:
    - Track global onboarding state (including files)
    - Manage step navigation and error/loading states

API-related tasks:

- **Fetch industry options** from a mock endpoint
- **Simulate verification code** sending and validation
- **Mock submission of all onboarding data**, including references to uploaded files

  ✅ You may mock APIs using any method you prefer (e.g., timeouts, mock service workers - MSW, packages). Ensure you provide a way to simulate both **successful** and **failing** API responses.

3. Form Handling & UX

- Each step must include input validation, preferably using zod (e.g., required fields, email format, file type/size).
- Allow users to **navigate back/forth** without losing data.
- On the final step:
    - Show a summary of all entered data
    - Include file names/previews
- Show a **loading indicator** during final submission
- Display a **success message** upon completion

📎 File Upload Details

Step	Field	Accepted Types	Max Size
Personal Details	Profile Image	JPG, PNG	2MB
Business Details	Business Document	PDF	5MB

🎁 Bonus Features

1. **Autosave to localStorage**
    - Save form state (including file references) to localStorage.
    - Restore form on refresh if data exists.
    - Add support for clearing and restarting the process.
2. **Unit Tests**
    - Write unit tests for at least **one full step component**, including validation logic.

📁 Deliverables

- A **Vue 3 project** using **Vite** or **Nuxt 3**
- Structured codebase (`components/`, `store/`, `api/`, etc.)
- **Readme.md** with:
    - Setup and run instructions (`yarn install && yarn run dev`)
    - A brief overview of your process and choices while developing the flow.
    - Instructions for how API mocking was implemented
    - (Bonus) How to run the unit tests

🧪 Evaluation Criteria

Area	Expectation
Component Design	Modular, testable, reusable components with clear prop contracts
State Management	Proper use of Pinia or Vuex, well-organised global state
Form UX & Validation	Robust validation, seamless navigation, and file handling
API Integration	Correct handling of async operations, success/error flows
Code Quality	Clean, maintainable, readable codebase
Bonus (if implemented)	Smart autosave + well-structured test cases

---

⏳ Estimated Time

- **Core features**: 5 – 7 hours
- **With bonus (autosave + testing)**: 7 – 9 hours