Note Taking Application – Bug Report and Refactor Summary

1. Overview

This document describes the debugging and refactoring of a broken React note-taking application with a single Home route. The goal was to ensure that a user can type a note in a text field, click a button, and see all notes displayed as an unordered list on the same page. Initially, the project was not runnable as a React app and only showed the default Vite starter page once a proper project was created.

2. Environment and Final Behavior

Environment

- Tech stack: React + Vite (created with `npm create vite@latest note-app -- --template react`).
- Platform: Windows, Node.js and npm installed.

Expected Final Behavior

- The Home route shows a heading, a text input, and an "Add Note" button.
- When the user enters text and clicks the button (or presses Enter), the note is added to an unordered list displayed below the input field.
- All notes remain visible as the user keeps adding new notes.

---

3. Bugs and Fixes

Bug 1 – Project not runnable as a React app

Symptom

- Running `npm install`, `npm run dev`, or `npm start` in the initial folder (`C:\innomatics`) failed with errors like:
  - "Could not read package.json: ENOENT: no such file or directory, open 'C:\innomatics\package.json'".

There was a JSON file with only

json

```
{ "name": "innomatics", "lockfileVersion": 3, "requires": true,
"packages": {} }
```

but the app could not run.

Root Cause

- The folder did not contain a valid React project.
- There was no proper `package.json` with `dependencies` and `scripts` such as `"dev"` or `"start"`, so npm had nothing to install or execute.

Fix

- Created a new React project using Vite in `C:\innomatics\note-app`:

Bash

```
npm create vite@latest note-app -- --template react
```

- Changed into the new project folder and installed dependencies:

Bash

```
cd C:\innomatics\note-app npm install
```

Why This Works

- Vite's React template generates a complete `package.json` containing React, ReactDOM, Vite, and the necessary scripts to build and run the app.
- With this structure in place, `npm install` can install dependencies, and `npm run dev` can successfully start the development server.

Bug 2 – Application only showed the default "Vite + React" starter page

Symptom

- After running `npm run dev` and opening `http://localhost:5173/`, the browser showed the default "Vite + React" page with logos and a counter.
- The required note-taking UI (text field, button, notes list) was not visible.

Root Cause

- The generated `src/App.jsx` still contained the default Vite starter component, which displays the logos and counter.
- The Home route was rendering this default component instead of a custom note-taking component.

Fix

- Opened `C:\innomatics\note-app\src\App.jsx`.
- Deleted all starter template code and replaced it with the custom note-taking component:

jsx

```
import React, { useState } from "react"; function App() { const [noteText, setNoteText] = useState(""); const [notes, setNotes] = useState([]); const handleInputChange = (event) => { setNoteText(event.target.value); }; const handleAddNote = () => { const trimmed = noteText.trim(); if (!trimmed) return; setNotes((prevNotes) => [...prevNotes, trimmed]); setNoteText(""); }; const handleKeyDown = (event) => { if (event.key === "Enter") { handleAddNote(); } }; return ( <div style={{ maxWidth: "500px", margin: "2rem auto", fontFamily: "sans-serif", padding: "1rem", border: "1px solid #ddd", borderRadius: "8px", }} > <h1>Note Taking App</h1> <input type="text" placeholder="Type your note here..." value={noteText} onChange={handleInputChange} onKeyDown={handleKeyDown} style={{ width: "100%", padding: "0.5rem", boxSizing: "border-box", }} /> <button type="button" onClick={handleAddNote} style={{ marginTop: "0.5rem", padding: "0.5rem 1rem", cursor: "pointer", }} > Add Note </button> <ul style={{ marginTop: "1rem" }}> {notes.map((note, index) => ( <li key={index}>{note}</li> ))} </ul> </div> ); } export default App;
```

Why This Works

- The Home route now renders a component that implements the required UI and behavior: text input, button, and notes list.
- The default Vite template is completely removed, so the app shows the note-taking interface instead of the starter logos and counter.

Bug 3 – Notes not stored in React state (conceptual issue)

Symptom

- In a broken design, notes might be stored in a normal array or not stored at all, meaning the UI would not update when the user adds a note.
- Even if `push` is called on an array, React will not re-render unless state is updated properly.

Root Cause

- React requires state (e.g., via `useState`) to trigger re-renders when data changes.
- Directly mutating a plain array or mutating a state array without creating a new reference does not cause React to update the DOM.

Fix

- Introduced state to hold the notes list:

jsx

```
const [notes, setNotes] = useState([]);
```

- Updated the `handleAddNote` function to add notes immutably:

jsx

```
const handleAddNote = () => { const trimmed = noteText.trim(); if
(!trimmed) return; setNotes((prevNotes) => [...prevNotes, trimmed]);
setNoteText(""); };
```

Why This Works

- `useState` keeps the notes in React-managed state.
- Using `setNotes((prevNotes) => [...prevNotes, trimmed])` creates a new array reference on each update, which signals React to re-render the component and show the new list of notes.

Bug 4 – Input not controlled and not cleared properly

Symptom

- The input field could be out of sync with the internal data, or it might not clear after adding a note.
- This leads to a confusing user experience.

Root Cause

- The input was not configured as a controlled component, or the state was not reset after adding a note.
- Without binding `value` and `onChange` to state, React cannot reliably track what is in the field.

Fix

- Managed the input as controlled state:

jsx

```
const [noteText, setNoteText] = useState("");
```

- Bound the input to state and reset it after adding a note:

jsx

```
<input type="text" placeholder="Type your note here..."
value={noteText} onChange={(e) => setNoteText(e.target.value)}
onKeyDown={handleKeyDown} /> // In handleAddNote: setNoteText("");
```

Why This Works

- The input now uses `noteText` as its single source of truth.
- Every keystroke updates the state via `onChange`, and after adding a note, setting `noteText` to an empty string clears the field, providing a clear and consistent UX.

---

4. Final Result

After applying these fixes:

- The project is a valid React + Vite application that starts with `npm run dev`.
- The Home route shows a note-taking interface with a text field, an “Add Note” button, and an unordered list of notes.
- Users can add multiple notes, and each note appears immediately in the list below the input on the same page.
- The debugging process has been documented with symptoms, root causes, and code-level fixes for each issue