**Final Project – Contact Management**

The **Contact Management** is a simple web application that allows users to store and manage their contacts efficiently. The application is built with a React-based frontend and a Node.js & Express backend, using a MongoDB database to store contact information, enabling users to add, update and delete contacts.

**Features**

1. **Add New Contacts** – Users can enter a contact’s name, phone, email and address and add it to the list.
2. **Edit Existing Contacts** – Users can update details of a contact.
3. **Delete Contacts** – Users can remove a contact permanently.
4. **View Contacts** - A table displays all saved contacts. Users can click “View” to see the full details of a contact
5. **Validation for Duplicate Entries** – Prevents adding duplicate contacts based on phone and email fields.
6. **Confirmation Before Deletion** – A confirmation note displayed before deleting a contact.

**Project Format/Structure:**

The project is divided in two structures:

**Backend (MEN):**

The backend built with **Node.js and Express.js** as an API server that handles data operations. It communicates with a **MongoDB database** to store contacts.

* **RESTful API**– Provides API endpoints (/api/contacts) for CRUD operations.
* **Database Management** – Stores and retrieves contact data.
* **Request Handling** – Manages requests from the frontend.
* **Data Validation** – Ensures required fields are present.
* **Error Handling** – Returns appropriate error messages for failed operations.

The system uses **at least five API endpoints** to manage contacts:

1. POST /contacts Add a new contact
2. GET /contacts Retrieve all contacts
3. GET /contacts/:id Retrieve a specific contact
4. PATCH /contacts/:id Update contact details
5. DELETE /contacts/:id Delete a contact

**Frontend (React+Vite):**

**Key Components**

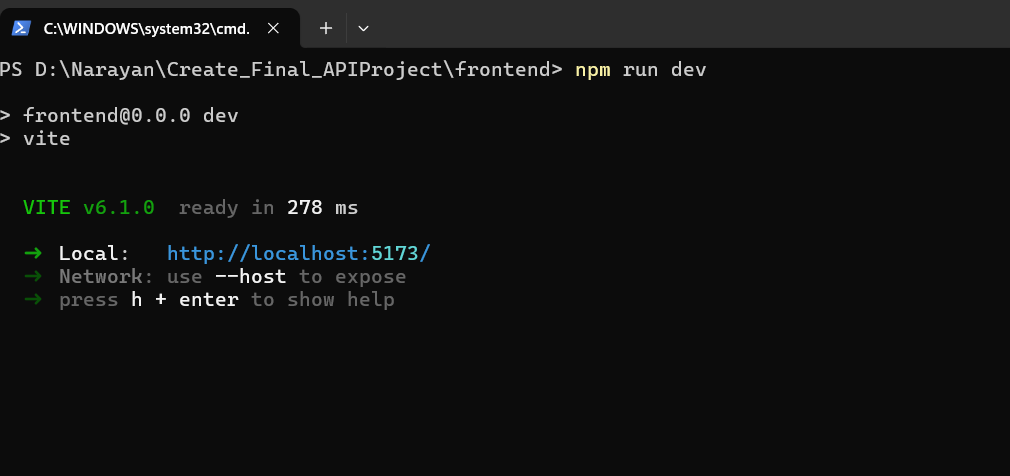
* **App.jsx** – The root component that renders ContactManager.
* **ContactManager.jsx** – Handles the main logic of managing contacts.
* **ContactList.jsx** – Displays the list of contacts in a table format.
* **AddContact.jsx** – Provides the form to add or update contacts.
* **DeleteContact.jsx** – Manages the deletion confirmation of a contact.

**Frontend Flow**

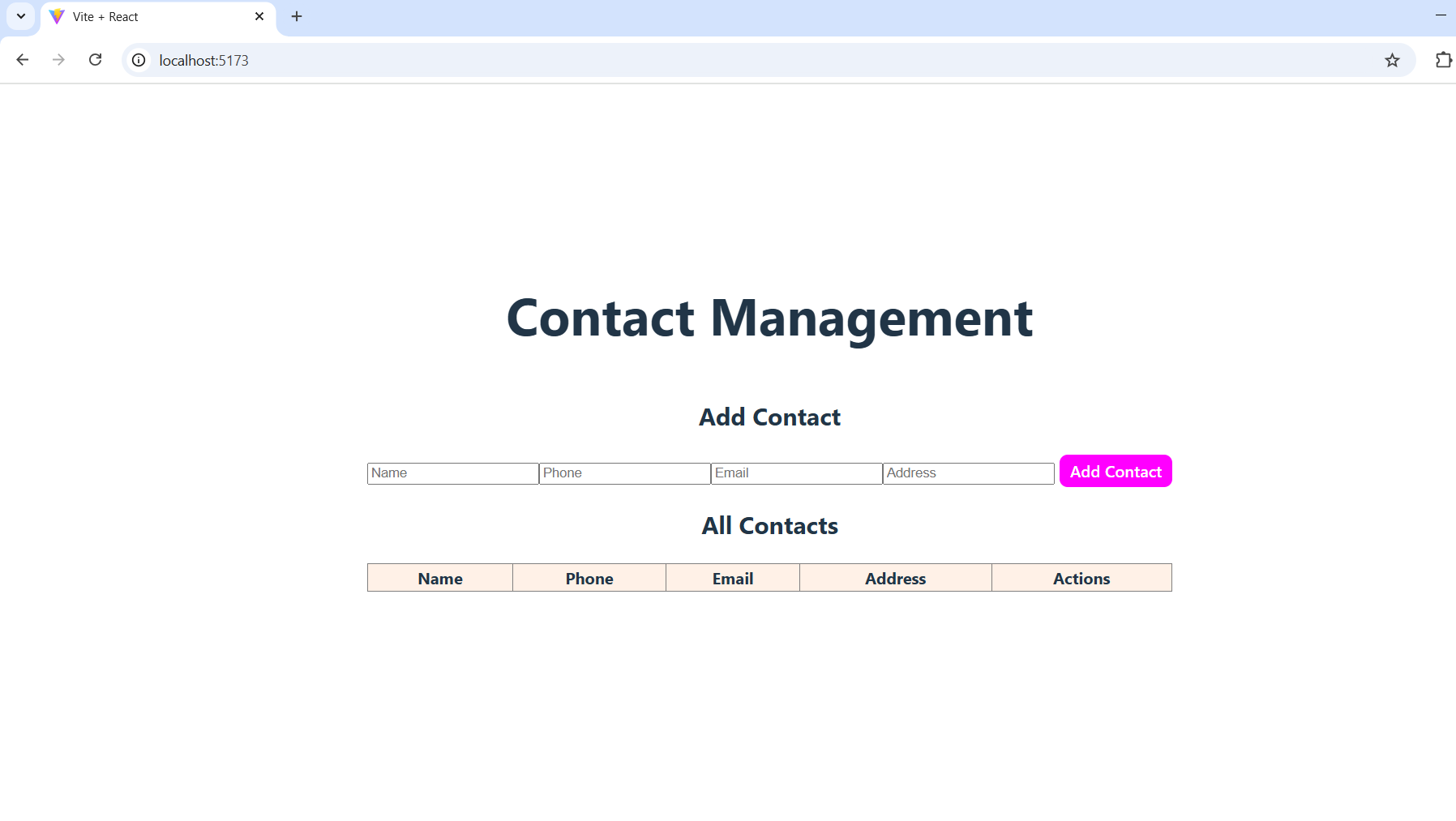
1. When the page loads, React’s useEffect fetches contacts from the backend.
2. Users can **add** a contact through the form, triggering an API request.
3. Users can **edit** a contact by clicking “Edit,” pre-filling the form.
4. Users can **delete** a contact, prompting a confirmation modal.
5. The app dynamically updates the UI without requiring a page refresh.

**Screenshots**:

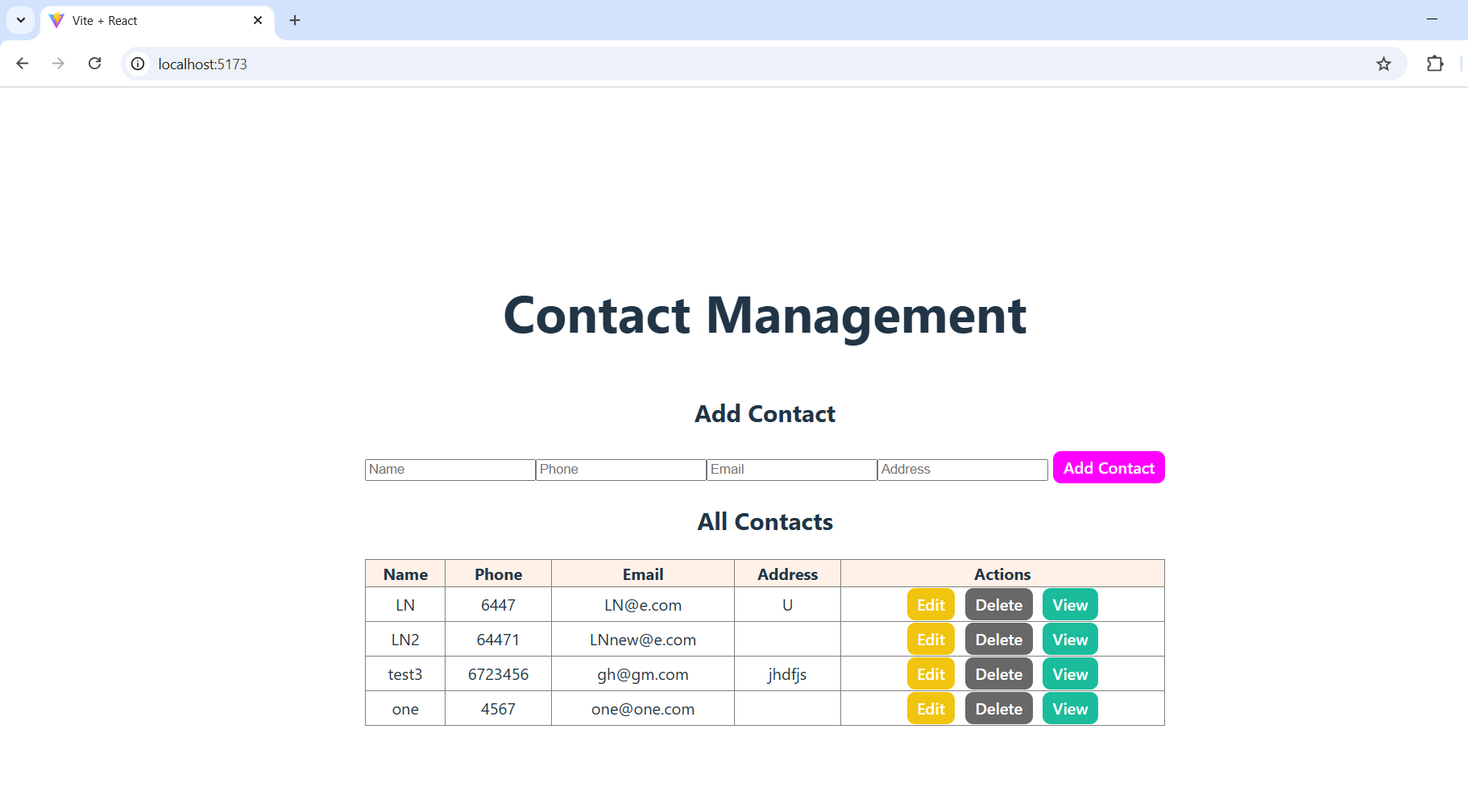
Started the front end and back-end server



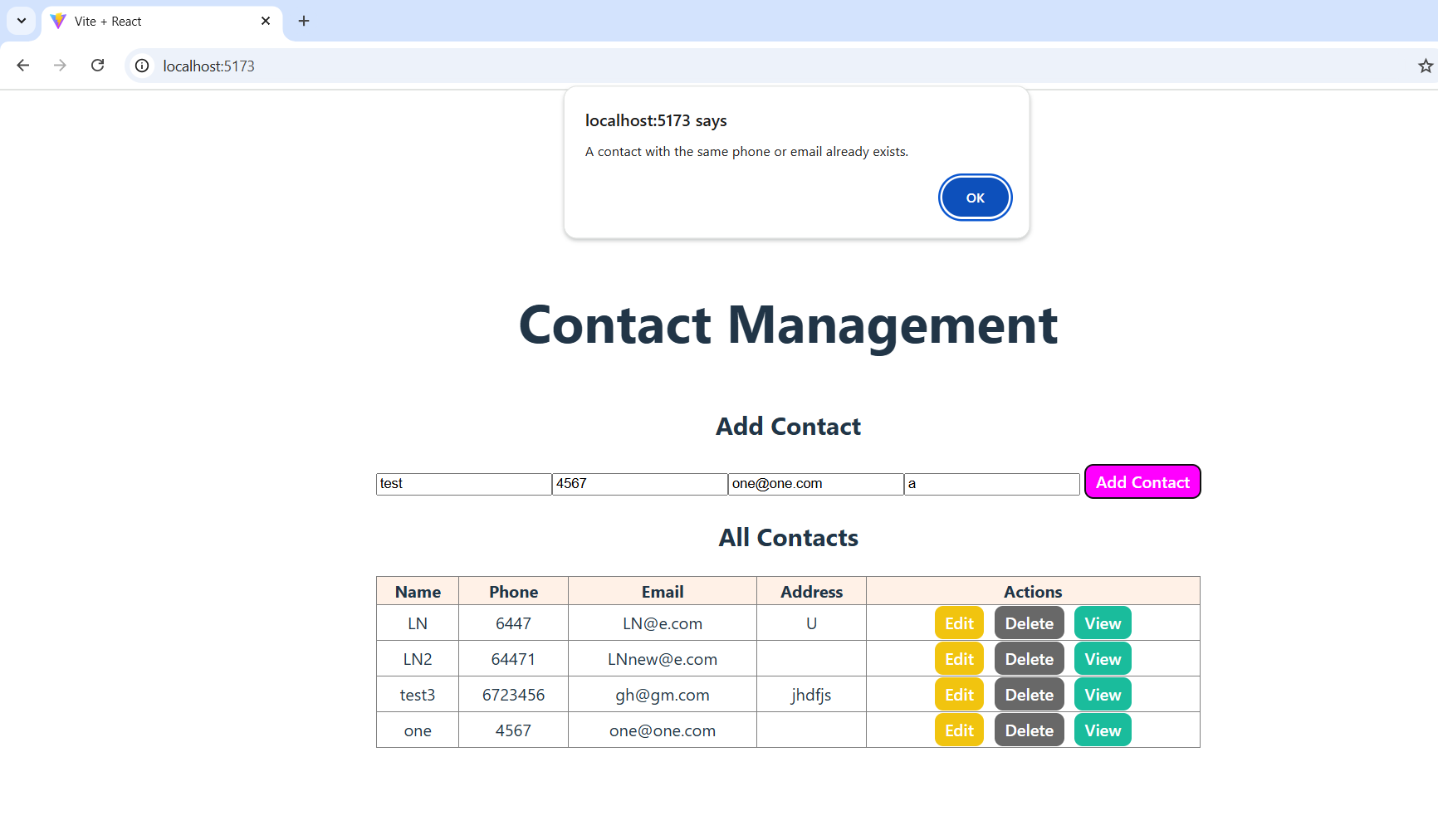
Application:

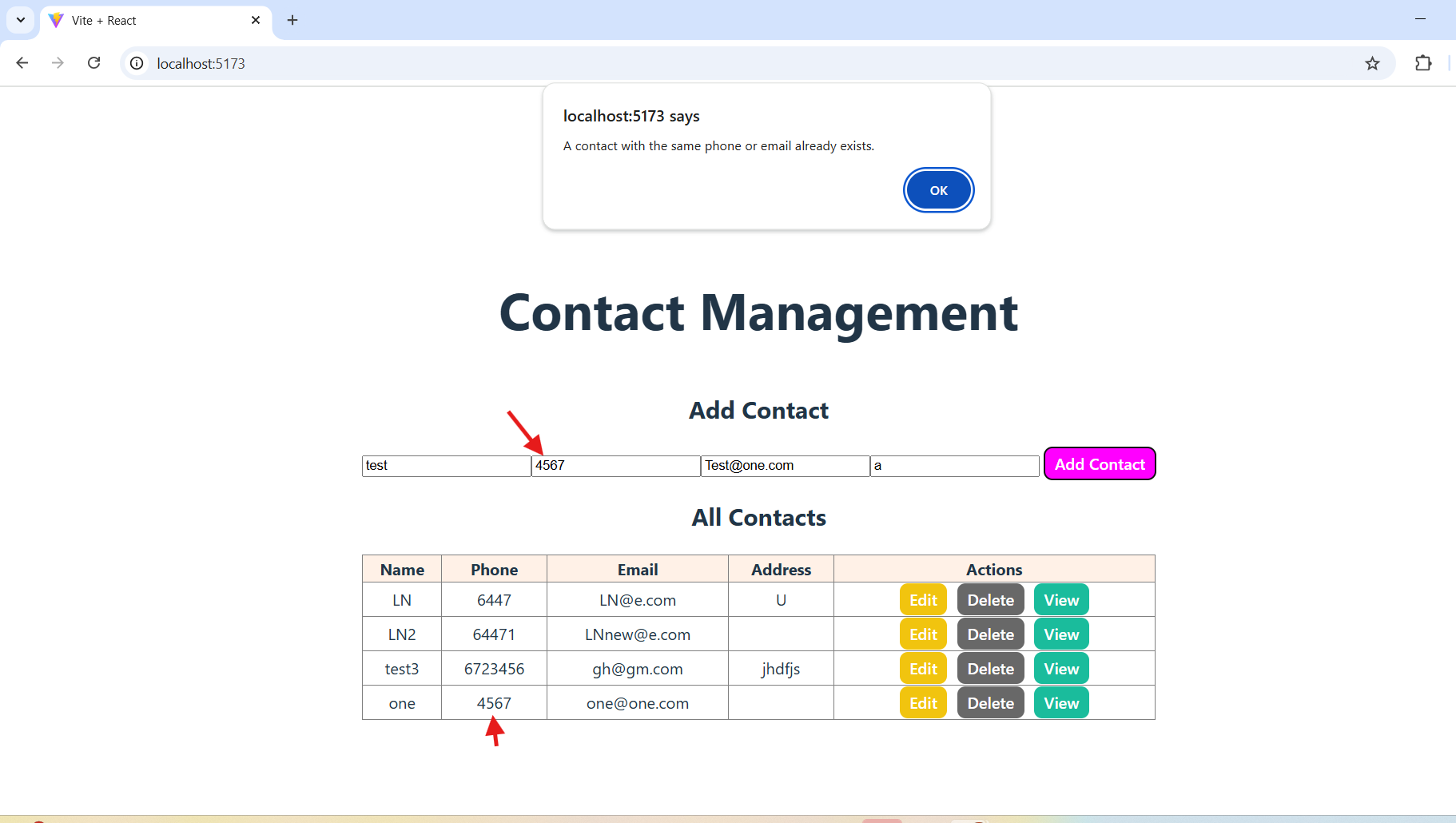


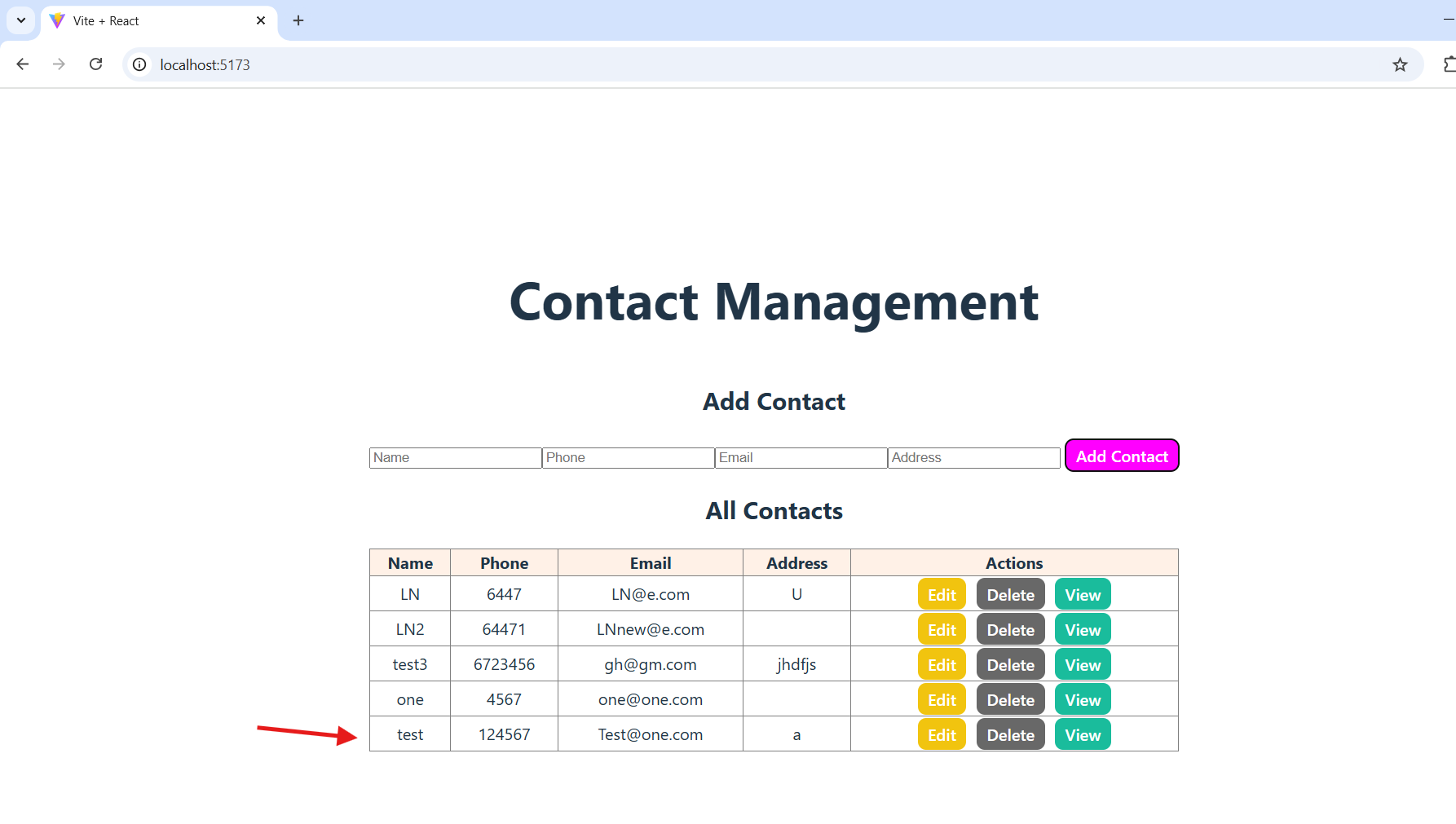
Adding (unique) multiple contacts



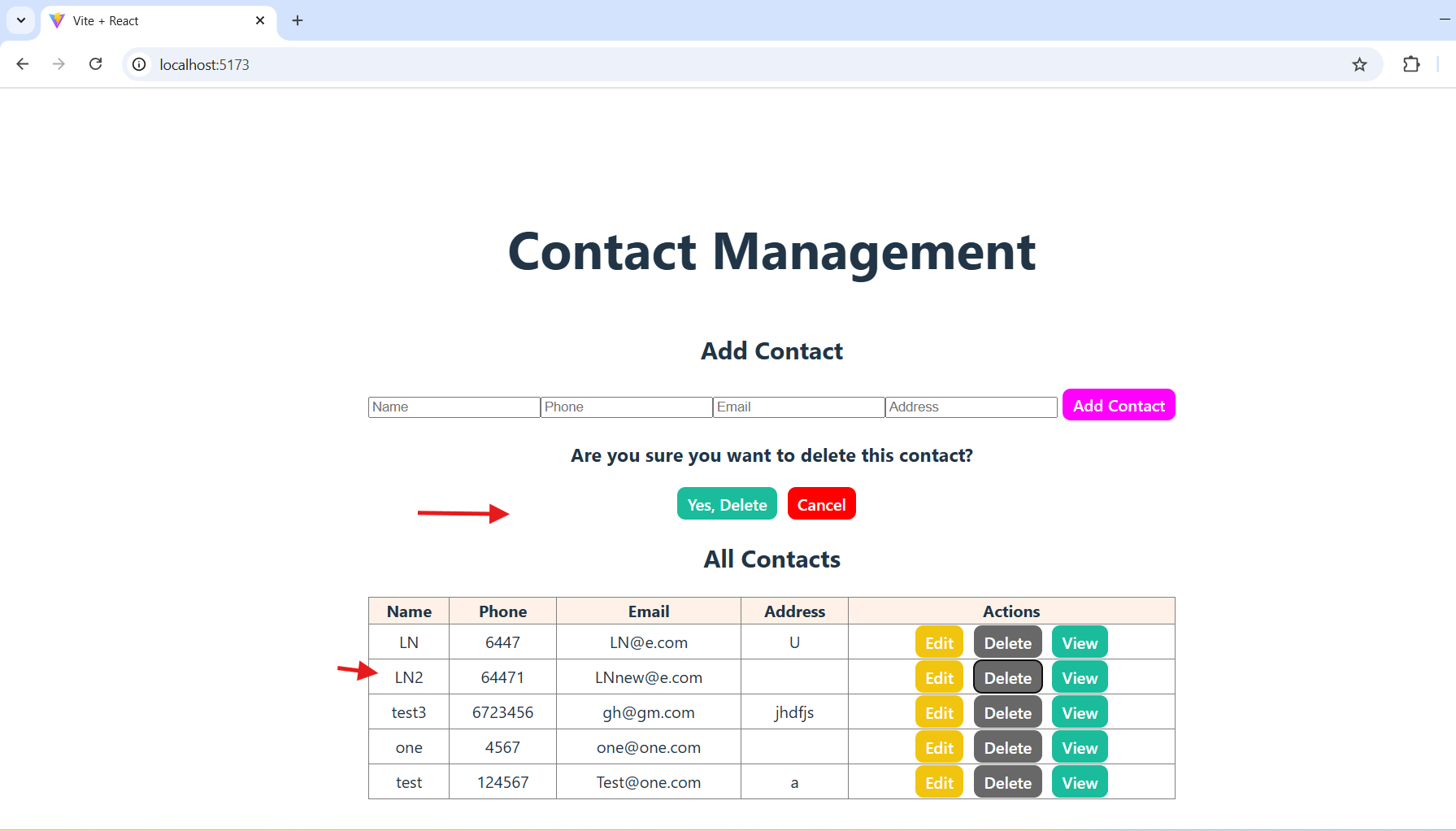
Validation when trying to add an existing error (both phone and Email should be unique)





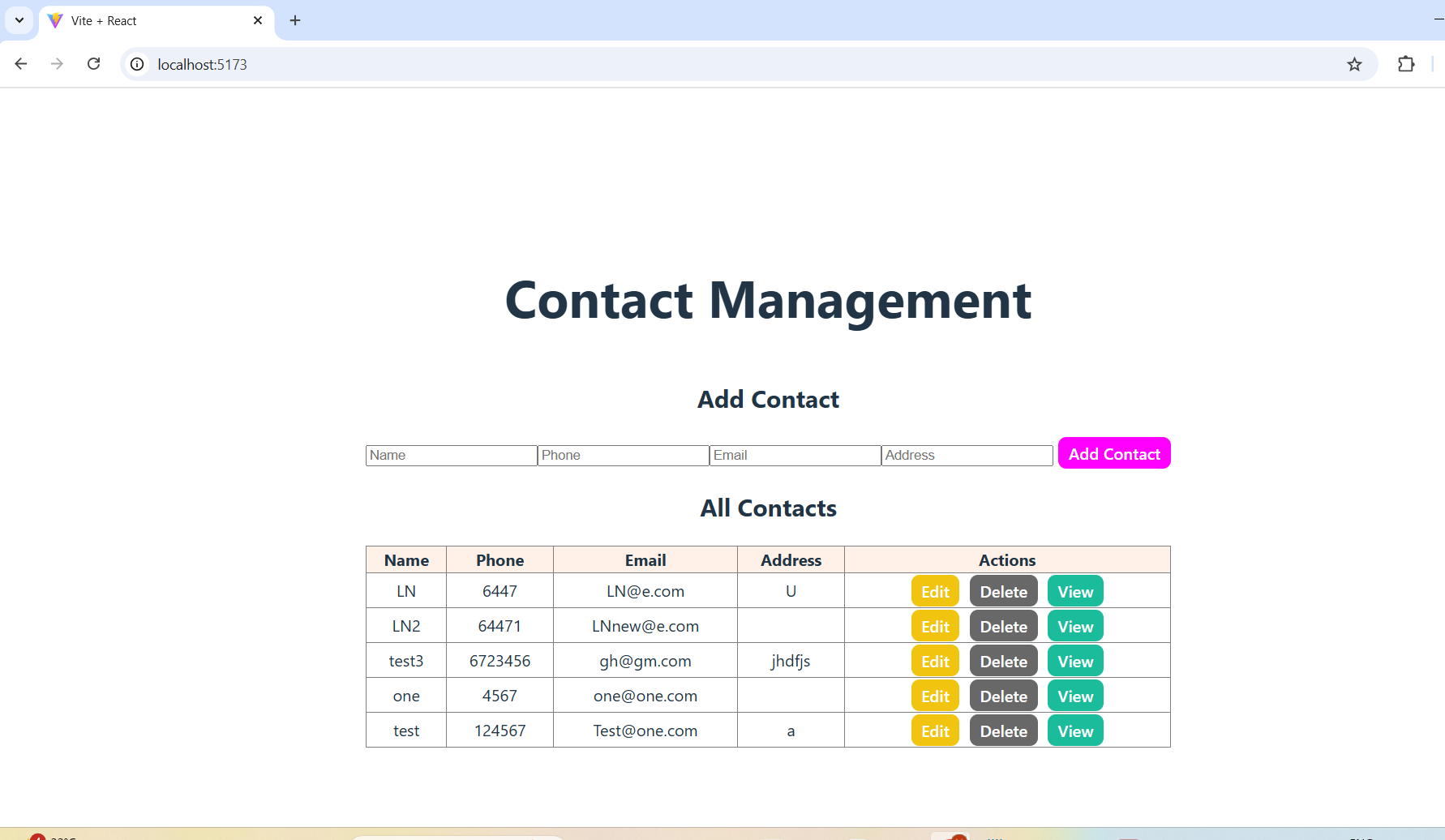


Delete Action

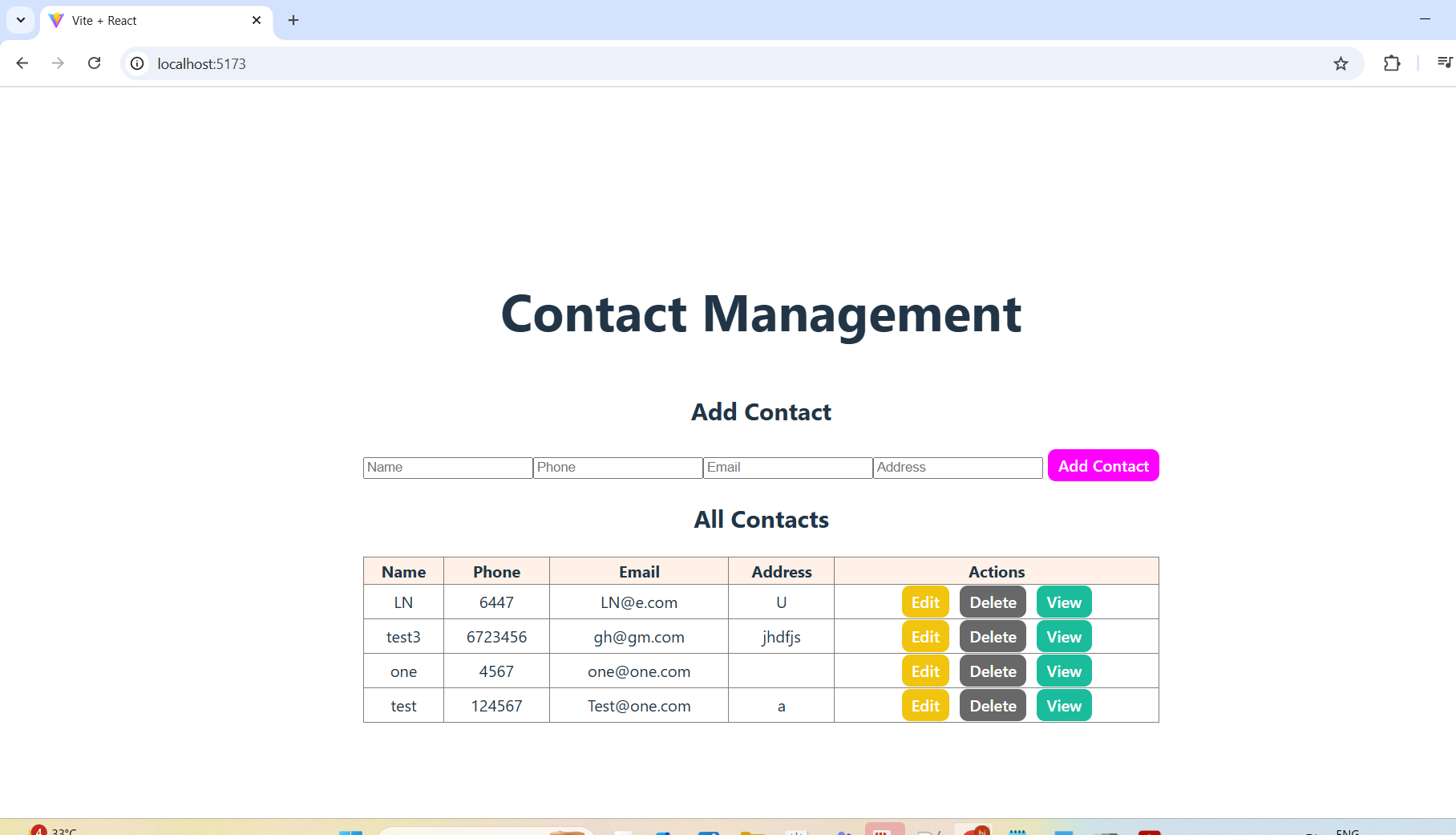


Cancel Action

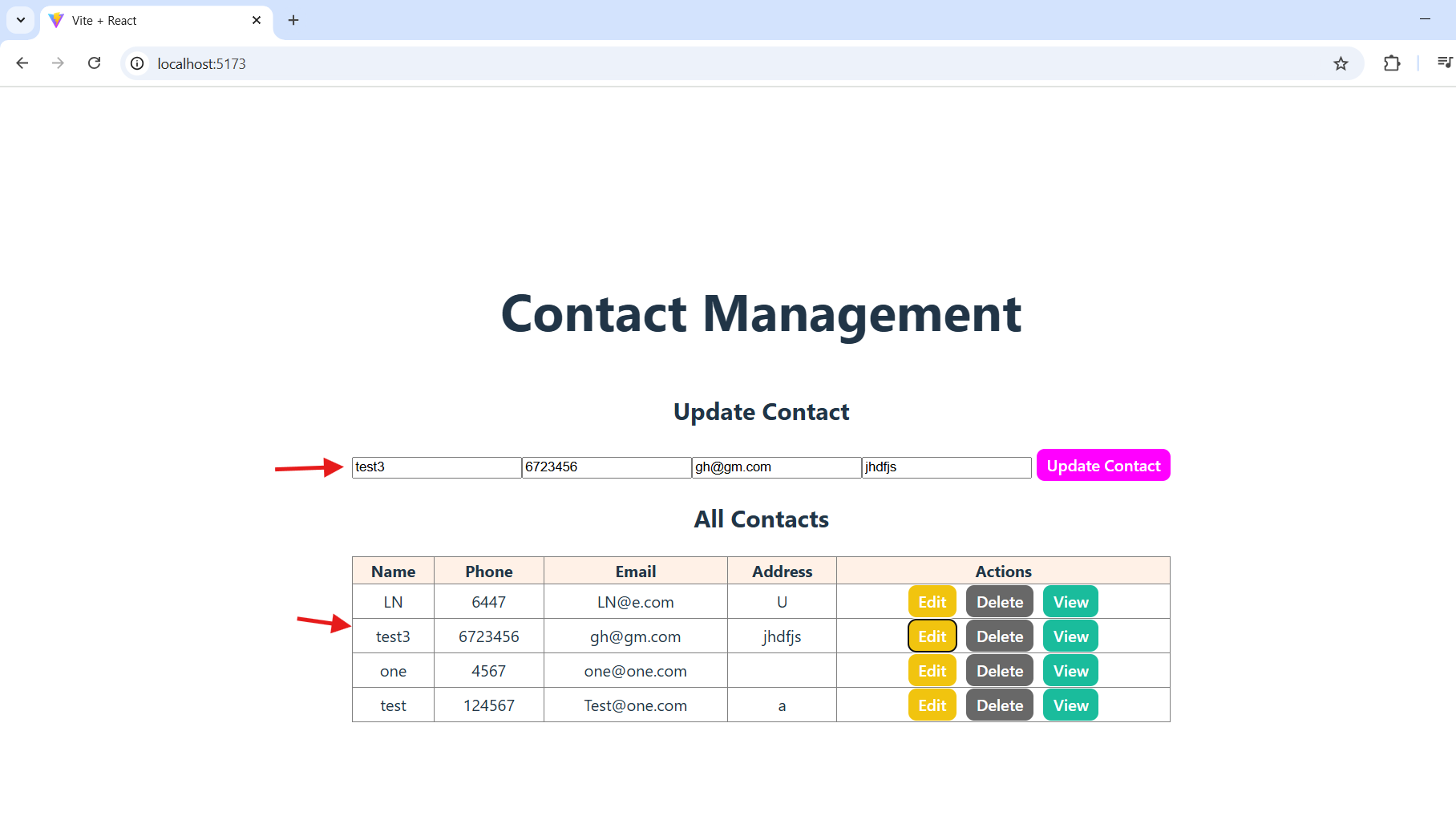
No contact deleted



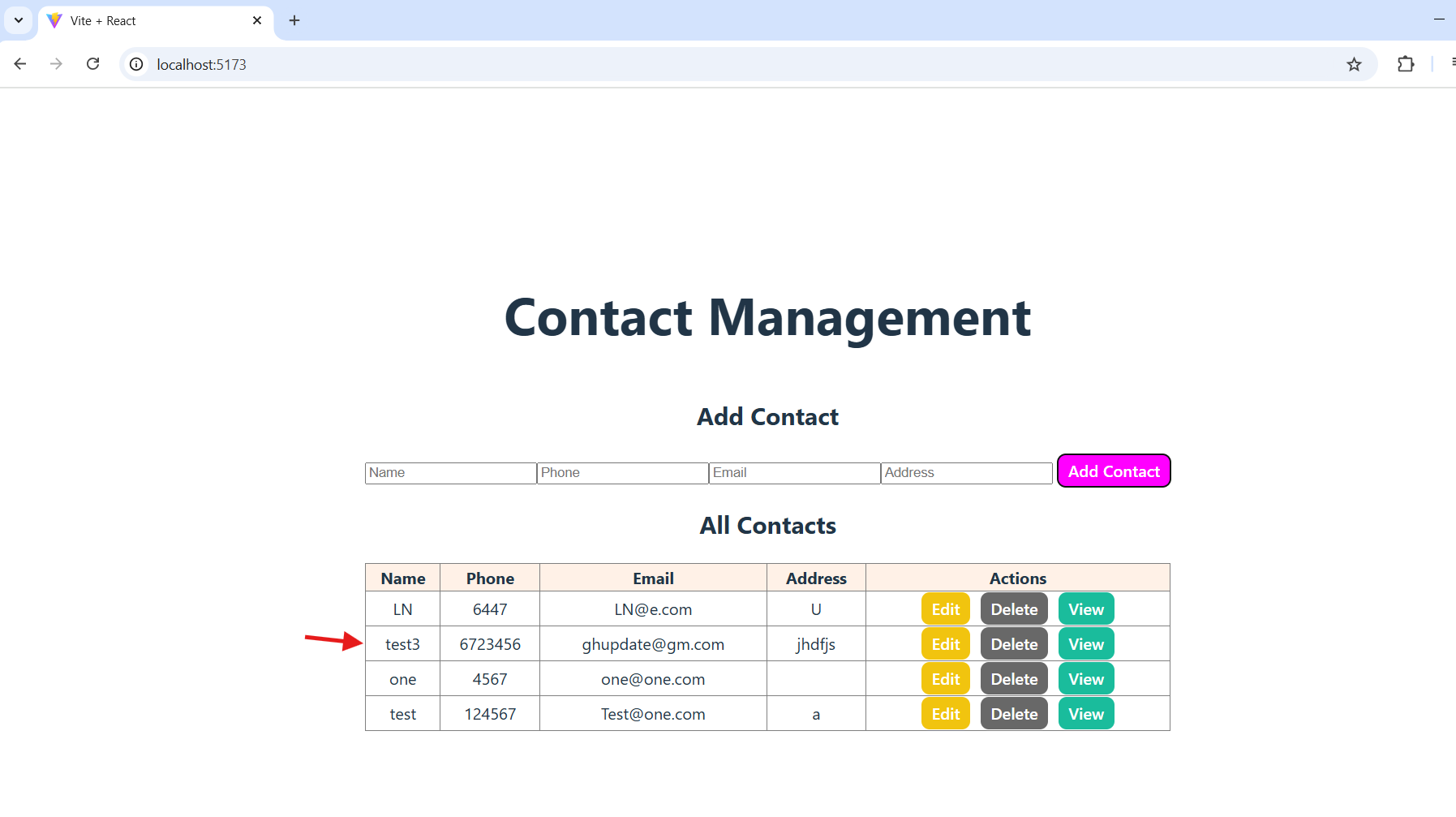
Yes, Delete option: - deleted the contact



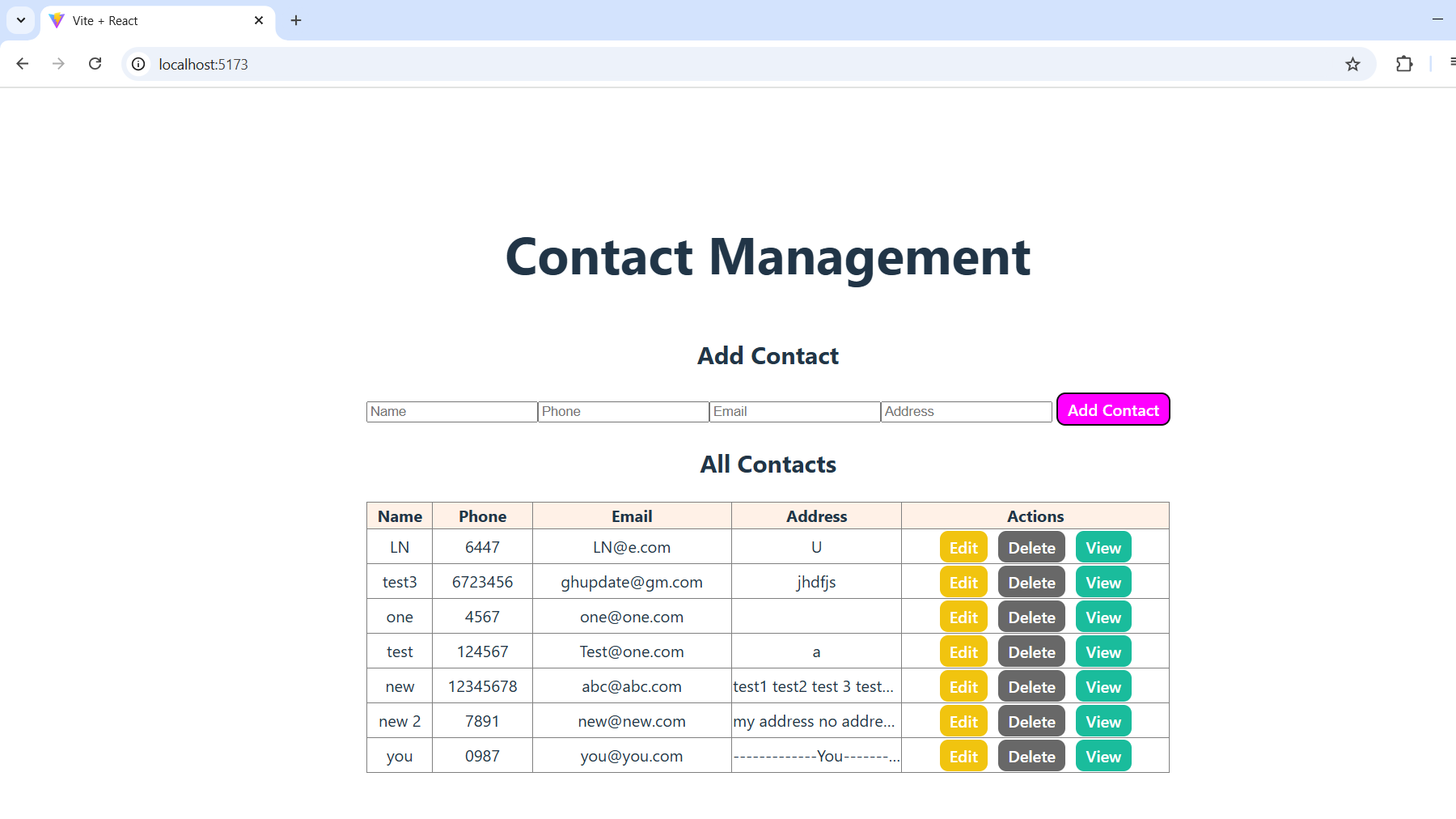
Update existing



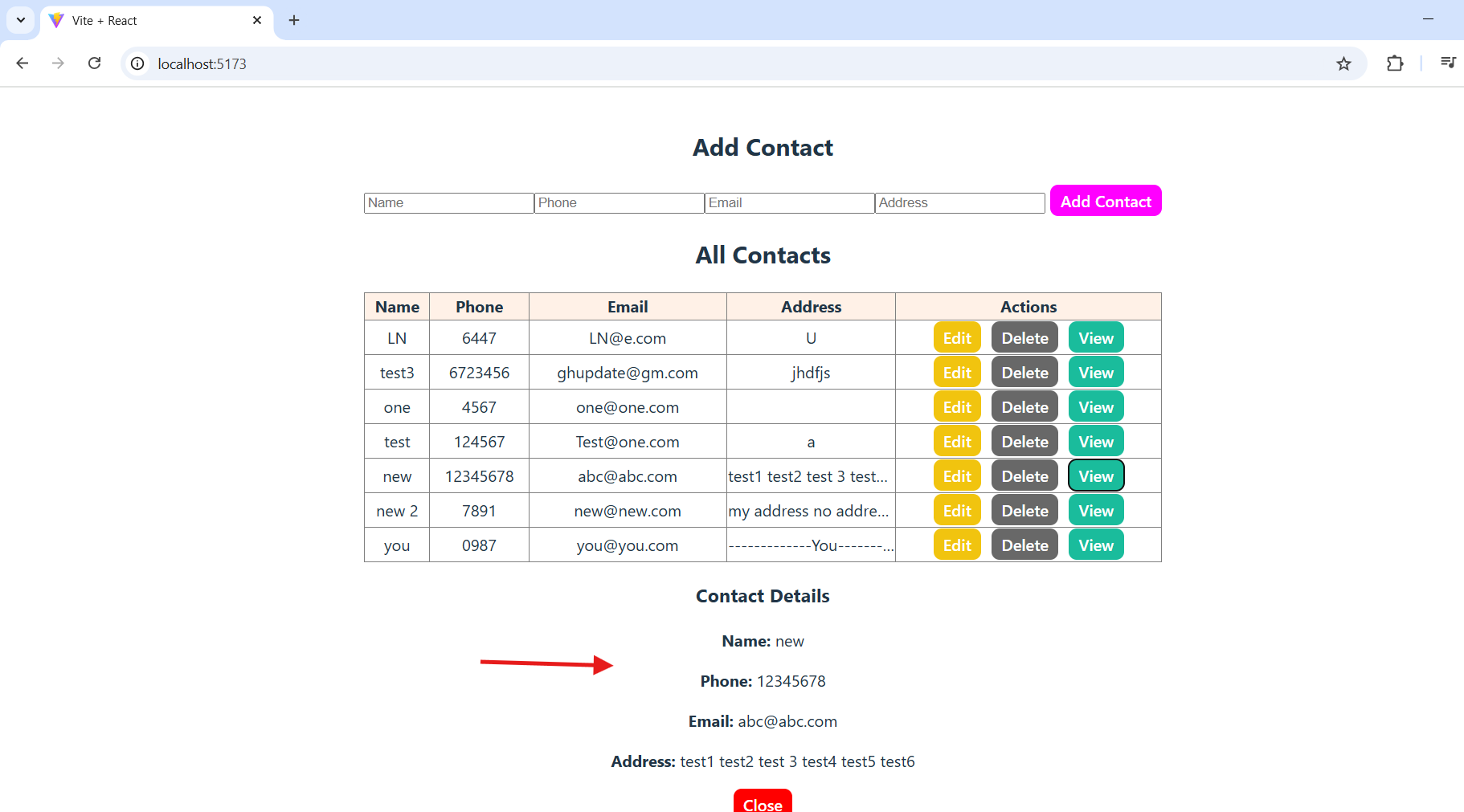
Updated email



Added new records



View Contact



Close button

