3. Contact Manager

✅ Planned Features

- [ ] 🌐 Multi-language support (Frontend using `react-i18next`)
- [ ] 🔔 Alarm/alerts for upcoming birthdays
- [ ] ✅ Account verification via OTP on registration (email-based)
- [ ] 🔐 Login → Account Verification → Dashboard flow
- [ ] 🌍 Display created/updated timestamps in user's local timezone
- [ ] 📂 Export contacts as `.csv` or `.json` & allow CSV upload/import
- [ ] 🗑️ Delete or favorite multiple contacts in bulk
- [ ] 📅 Set follow-up reminders on specific contacts

📦 NPM Packages to be Installed

Package	Purpose
`react-i18next`	Multi-language support on the frontend
`dayjs`	Lightweight date/time library for formatting & timezone handling
`moment`	Alternative to `dayjs`, mainly for handling complex date operations
`node-cron`	To schedule tasks like birthday alerts or reminder checks

Package	Purpose
`express-validator`	For validating inputs in backend APIs
`papaparse`	CSV parsing (e.g., for CSV import) on frontend
`json2csv`	Convert JSON to CSV (e.g., for exporting contacts)

1. User models

 `userSchema`

Field Name	Type	Required	Description
`name`	String	✅ Yes	Full name of the user
`email`	String	✅ Yes	Unique email in lowercase
`password`	String	✅ Yes	Hashed user password
`isVerified`	Boolean	❌ No (default: false)	Indicates if the user's email is verified
`otp`	String	❌ No	OTP sent to the user for account verification
`otpExpiresAt`	Date	❌ No	Expiry time for the OTP
`timezone`	String	❌ No	User's preferred timezone (e.g., `Asia/Kolkata`) for displaying created/updated dates properly
`createdAt`	Date	Auto	Automatically set on creation
`updatedAt`	Date	Auto	Automatically updated on modification

✅ User API Endpoints (Authentication + Account Management)

Method	Route	Auth Required	Description
`POST`	`/api/auth/register`	❌ No	Register a new user
`POST`	`/api/auth/login`	❌ No	Login and receive a JWT token
`POST`	`/api/auth/logout`	✅ Yes (client-side only)	Logout by clearing the token on the client side
`PUT`	`/api/users/password`	✅ Yes	Change the password of the logged-in user
`DELETE`	`/api/users/me`	✅ Yes	Delete the currently logged-in user's account

`GET`	`/api/users/forgot-password`	❌ No	Initiate password reset via email using NodeMailer + OTP/token validation

2. Profile (child) → User (parent) (1 to 1)

✅ Profile Model – Data Fields

Field Name	Type	Required	Description
`user`	ObjectId	✅ Yes	Reference to the associated `User` (_id)
`bio`	String	❌ No	Short user biography or about section
`avatarUrl`	String	❌ No	URL to the user's profile picture
`phone`	String	❌ No	User's phone number
`address`	String	❌ No	Physical or mailing address

🔁 Profile API List

Method	Route	Auth Required	Description
`GET`	`/api/profile/me`	✅ Yes	Get the profile of the currently logged-in user
`POST`	`/api/profile`	✅ Yes	Create a new profile (if it doesn't already exist)
`PUT`	`/api/users/me`	✅ Yes	Update user details like name and email
`PUT`	`/api/users/profile-pic`	✅ Yes	Upload or update user's profile picture

3. contact (child) → user (parent)

✅ Essential Contact API Endpoints

🧾

CRUD API Routes

Method	Route	Description
`POST`	`/api/contacts`	Create a new contact
`GET`	`/api/contacts`	Get all contacts for the logged-in user
`GET`	`/api/contacts/:id`	Get a single contact by ID
`PUT`	`/api/contacts/:id`	Update a contact by ID
`DELETE`	`/api/contacts/:id`	Delete a contact by ID

📂

Import & Export Routes

Method	Route	Description
POST	/api/contacts/import	Upload & import contacts from a CSV file
GET	/api/contacts/export	Export selected or all contacts to CSV

✅

Bulk Actions Routes

Method	Route	Description
POST	/api/contacts/bulk-delete	Delete multiple selected contacts
PUT	/api/contacts/bulk-update	Update multiple contacts (e.g., tags, favorite)
PUT	/api/contacts/bulk-favorite	Mark multiple contacts as favorite/unfavorite

🎯

Special Feature Routes

Method	Route	Description
GET	/api/contacts/birthday-today	Get all contacts with birthday today
GET	/api/contacts/reminders	Get all contacts with upcoming reminders
PUT	/api/contacts/:id/toggle-favorite	Toggle favorite status for a contact
PUT	/api/contacts/:id/toggle-alert	Toggle general alert status
PUT	/api/contacts/:id/toggle-birthday-alert	Toggle birthday alert status
PUT	/api/contacts/:id/toggle-selected	For UI selection handling (optional, client-only)

🔍

Search & Filter Routes

Method	Route	Description
GET	/api/contacts/search?query=xyz	Search contacts by name, email, or phone
GET	/api/contacts/filter?tag=family&favorite=true	Filter contacts by tags, favorites, etc.

Feature	Option	Description
Search/filter	`/api/contacts?search=rohit`	Filter by name, email, tags, etc.
Filter by tags	`/api/contacts?tag=family`	Return contacts with specific tag
Favorites only	`/api/contacts?favorite=true`	Show only favorite contacts
Pagination	`/api/contacts?page=1&limit=10`	Paginate large contact lists

✅ Contact Model – Data Fields (Tabular Format)

Field Name	Type	Required	Description
`Field Name`	Type	Required	Description
`user`	`ObjectId`	✅ Yes	Reference to User (_id)
`name`	`String`	✅ Yes	Full name
`email`	`String`	❌ No	Email address
`phone`	`String`	❌ No	Contact's phone number
`address`	`String`	❌ No	Contact's address
`birthday`	`Date`	❌ No	Date of birth
`birthdayAlert`	`Boolean`	❌ No	Enable birthday alert (default: false)
`reminderDate`	`Date`	❌ No	Follow-up or call-back reminder
`alertEnabled`	`Boolean`	❌ No	General alert toggle
`notes`	`String`	❌ No	Additional notes
`favorite`	`Boolean`	❌ No	Marks as favorite
`tags`	`[String]`	❌ No	Labels like work, family
`imported`	`Boolean`	❌ No	Flag for imported contacts
`createdByImport`	`Boolean`	❌ No	True if added via CSV
`exported`	`Boolean`	❌ No	True if exported
`isSelected`	`Boolean`	❌ No	Used for bulk actions (UI only)
`createdAt`	`Date`	Auto	Auto timestamp
`updatedAt`	`Date`	Auto	Auto timestamp

✅ Project Tech Stack Overview (Tabular Format)

Layer	Technology	Purpose / Responsibility
Frontend	**Next.js (React)**	User Interface, routing, API consumption, form handling

Backend	**Node.js**	Server runtime environment for executing JS
	Express.js	REST API routing, middleware, business logic
Database	**MongoDB**	NoSQL database to store users, profiles, and contacts
ORM	**Mongoose**	MongoDB object modeling, schema definitions
Auth	**JWT (jsonwebtoken)**	Stateless authentication using access tokens
	bcryptjs	Password hashing for secure storage
Middleware	Custom Express Middleware	Protect routes, extract user from JWT
Dev Tools	Nodemon, TypeScript (optional)	Dev server reload & static typing

✅ MERN Contact Manager App – Deployment Overview

Layer	Platform	Purpose / Description
Frontend	**Vercel**	Deploy your **Next.js** app (auto CI/CD with GitHub)
Backend	**Render**	Deploy your **Node.js + Express + MongoDB** API server
Database	**MongoDB Atlas**	Cloud-hosted MongoDB database (fully managed)
Auth	**JWT**	Stateless authentication across frontend & backend

🚀 Features & Enhancements (Categorized)

🔐 Advanced Authentication & Security

Feature	Description
Refresh Token System	Implement access + refresh token flow for better security and session persistence
Forgot/Reset Password	Email-based password reset using tokens (NodeMailer + OTP/token validation)
Google OAuth Login	Allow users to sign in with Google (Passport.js or Firebase Auth)

Rate Limiting	Prevent brute-force attacks using `express-rate-limit`
Account Verification	Send verification email upon registration
Device Detection & Session Management	Track user login sessions per device/browser

👤 Profile & User Experience

Feature	Description
Profile Picture Upload	Upload via **Multer** + **Cloudinary**
Dark/Light Theme Toggle	Store preference in DB or localStorage
Multi-language Support (i18n)	Add support for internationalization/localization
Timezone Handling	Show created/updated dates based on user's timezone

📇 Contact Features

Feature	Description
Contact Import/Export	Export contacts as `.csv` or `.json` and allow CSV upload/import
Bulk Actions	Delete or favorite multiple contacts at once
Activity Log	Track CRUD actions per contact with timestamps
Contact Reminders	Set reminders to follow-up or reach out to specific contacts
Birthday Tracking	Add `birthday` field and notify user when it's near