Bank App

**User Registration/Signup:**

**Fields Required:**

1. **Full Name:**
   * Description: Enter your full name as per official records.
   * Validation: Ensure the name includes both first and last names.
2. **Email:**
   * Description: Provide a valid email address for communication and account verification.
   * Validation: Check for a valid email format.
3. **Address:**
   * Description: Enter your current residential address.
4. **Password:**
   * Description: Create a secure password to protect your account.
   * Validation: Enforce strong password criteria (e.g., minimum length, combination of letters, numbers, and special characters).

**Registration Process:**

* Users fill in the registration form with the required details.
* Validation check in the frontend itself
* Server Side validation takes place for the duplicate email
* Upon successful validation, store user information securely in the database.
* On successfully sign-up user will receive unique generated account number to their email
* Each User will have 1000 default.
* Send a verification email for account activation.
* During the registration process user schema from databse contain the fields are as follows
* name: String,
* email: String,
* pan: String,
* address: String,
* phone: Number,
* password: String,

password is stored in the db after encryption which will be done using bcrypt library of express

* accountno: String,
* money: Number,
* verify: String,
* Transaction[array ]

from this fields verify is a string for storing the random number created using the Math.random(5) function and also sent to the email which will work as a OTP

On the other hand transaction is a array in which the details of sender receiver ,date ,time, are stored during the transaction

**Login Process**

* Users enter their email and password in the login form.
* Server-side authentication verifies the user's credentials.
* If valid, generate a unique JWT token
* During the rendering every component the localstorage token is validated first before showing the component contents
* Store the token securely on the client side (e.g., cookies or local storage).
* Implement token expiration and refresh mechanisms for enhanced security.
* Redirect the user to the dashboard upon successful login.where user can make a payment by adding the recievers email address and amount to be send

**Logout:**

* Once user logout the account JWT token will be removed from local storage/Session storage
* User can’t access the dashboard without the log in