Understanding of the Signup Process on FirstContact.lgbt
By: Rimmi Kaur
Position: Frontend/Backend/Full-Stack Developer
Date: 18/02/2025
Email: rimmikaurwriter@gmail.com

Introduction

Although FirstContact.lgbt uses **Wix with Velo (JavaScript)**, I am analysing the signup process from a **MERN stack perspective**, providing deeper insights into how this workflow would operate if implemented using React/Next.js Node.js, Express, and MongoDB.

Step-by-Step Breakdown of the Signup Process

Frontend Logic (User Interface and Client-Side Experience):

1. **Accessing the Signup Page:**
    - Upon clicking the signup link, I was redirected to `signupfree`. The system verified my login status using cookies and an API call. If logged in, I bypassed the form. If not, a signup modal opened without reloading the page.
2. **Signup Form Interface:**
    - The modal included input fields (e.g., text, dates, select inputs, and checkboxes) with required and optional fields.
    - Real-time input validation was implemented using event listeners (`onChange()`, `onBlur()`) to validate formats such as email and password strength.
3. **Form Submission Process:**
    - On clicking submit, the form data was collected into a JSON object and sent to the server using `axios` via a `POST` request.
    - A loading spinner appeared during submission, indicating asynchronous data transfer.
4. **Email Verification Prompt:**
    - The API responded with a message instructing me to verify my email. The screen updated without page reload using state management (e.g., `useState()` in React).
5. **Post-Verification Message:**
    - After email verification, a success message displayed stating that my profile would be reviewed, and I would receive a confirmation email.

Backend Logic (Server-Side and Database Interactions):

1. **Handling Form Submission:**
    - The server received the `POST` request and validated the payload for completeness and proper formatting.
    - Valid records were inserted into a MongoDB collection.
2. **Screening and Security Checks:**
    - An automated screening process compared responses to predefined criteria.
    - Duplicate email detection was performed using `findOne()` queries, returning an error if a match existed.
3. **Email Verification:**
    - A unique tokenized verification link was sent via `nodemailer`.
    - When clicked, a `PUT` request updated the user's status to "Verified" in MongoDB.
4. **Rejection and Data Handling:**
    - Failed screenings triggered a rejection email.
    - User records flagged for deletion were scheduled for removal using background jobs.
5. **Security Measures:**
    - Passwords were hashed with `bcrypt` before storage.
    - All endpoints were secured via HTTPS, and rate limiting was applied to prevent abuse.

UI Observations & Recommendations:

- **Footer Issue:** The footer scrolls into blank space, which may be caused by excessive container height or improper position styles. CSS adjustments such as min-height or flexbox layout can fix this.
- **Footer Alignment:** The footer misalignment may be due to unbalanced padding or margin settings. Implementing a grid or flexbox footer container could address this.

Conclusion:

From a MERN stack perspective, this signup process aligns with core full-stack principles such as client-side validations, API handling, secure data management, and efficient routing. Despite being built on Wix Velo, the issues noted in the UI, such as footer misalignment and excess scrolling, highlight areas for improvement that could be resolved through better front-end layout practices.