IMAP Integration for Interview & Google Meet Data Extraction

Intern Name: Avanindra
Mentor: Ashutosh Tripathi
Tech Stack: Next.js 14 (App Router), IMAP, Database (Mongo DB)

Objective

The goal of this project is to implement IMAP integration within a Next.js 14 application using the App Router architecture. The purpose of this integration is to extract structured information from emails related to interviews or Google Meet invitations and store this information in a database for frontend display.

Key Features

- Connect to an email inbox using IMAP protocol.
- Filter and parse relevant emails such as interview invites or Google Meet links.
- Extract structured data from email contents.
- Store extracted information in a database.
- Display the data on the frontend in a user-friendly format.

Extracted Fields

From each relevant email, the system extracts:

- **Interviewer**: Name and/or email of the person conducting the interview.
- **Interviewee**: Name and/or email of the candidate.
- **Meeting Time**: Scheduled date and time of the meeting.
- **Meeting Link**: Google Meet or other video conferencing link.

- **Participants**: List of attendees, if mentioned.
- **Other Details**: Calendar description, location, or duration (if available).

Architecture Overview

Backend

- **IMAP Integration**: The backend connects to a mail server using the IMAP protocol to access the inbox and fetch new emails.
- **Email Parsing**: Emails are parsed for patterns and structured data using a combination of regular expressions and MIME/HTML parsing.
- **Data Processing**: Relevant emails are filtered (e.g., subject line containing "Interview" or "Meet") and necessary information is extracted.
- **Database Storage**: Extracted data is inserted into a database for later use in the frontend.

Frontend

- **Data Fetching**: Uses Next.js 14 App Router (`app/api`) routes to fetch data from the backend.
- **UI Presentation**: A frontend component displays the parsed and structured information, possibly with filtering or sorting capabilities (e.g., by date or interviewer).

Database Schema (Example)

```
{
 interviewer: String,
 interviewee: String,
 time: String,
 link: String,
 participants: [String],
 subject: String,
}
```

Email Processing Workflow

1. **Connect to Mailbox** using IMAP protocol.
2. **Search for Relevant Emails** with subjects or contents related to interviews or Google Meet.
3. **Parse Emails** using content analysis or calendar event parsing.
4. **Extract Data** using custom logic or libraries.
5. **Store in Database** using an API route.
6. **Display on Frontend** in a list or card view.

Technologies Used

- **Next.js 14** with App Router
- **Node.js** (for backend email processing)
- **IMAP Library** (`node-imap`)
- **Email Parser** (`mailparser`)
- **Database** (MongoDB)

Testing

- Manually tested with sample emails for correct extraction.
- Verified database entries for accuracy.
- Checked frontend display rendering with dummy data.

Outcome

This project enables automatic extraction and visualization of interview-related emails, reducing manual effort in scheduling and coordination. It can be extended to support real-time updates or integrations with third-party calendar APIs.