Excel Analytices

Platform Overview:

This platform provides a comprehensive solution for users to upload, analyze, and visualize data, with a strong emphasis on security and user experience. It caters to individual users for their data analysis needs and includes an administrative interface for platform management.

User-Facing Modules:

Landing Page:

The landing page serves as the initial gateway to the platform. It's designed to be visually appealing and informative, guiding potential users through what the platform offers.

- **Navbar**: This persistent navigation bar ensures users can easily access key sections like Features, Pricing, and Contact information. It also provides quick access for existing users to Login or new users to Signup.
- **Hero Section**: This prominent area uses a compelling headline, a concise description, and a clear call-to-action (CTA) button (e.g., "Get Started") to immediately grab user attention and encourage engagement.
- **Features Section**: This part highlights the platform's core capabilities. This likely includes:
    - Data Import: Support for various file formats like Excel and CSV.
    - Analytics: Tools for processing and extracting insights from data.
    - Visualization: Options for creating various types of charts and graphs.
    - Security: Emphasizing data protection and user authentication.

- **Pricing Section**: Clearly outlines the different subscription plans (e.g., "Free," "Professional") available, detailing the features included in each plan and their respective costs.
- **Contact Section**: A dedicated form for users to directly communicate with the platform administrators. Users can submit their name, email, and message.
    - Backend Integration: Upon submission, this form's data is sent to a backend API. The backend then either stores the message in a database for later review or immediately forwards it to the admin's email, ensuring prompt follow-up.

Signup Page:

The signup page is where new users register their accounts securely.

- **User Input**: Users provide their Name, Email, and a Password.
- **Email Verification (OTP)**: A critical security step. After form submission, the backend sends a One-Time Password (OTP) to the user's provided email address. The user must then enter this OTP on the platform to verify their email. This ensures that the email address belongs to the user and adds a layer of security against fraudulent sign-ups.
- **Backend Actions**:
    - User details are validated to ensure data integrity and proper formatting.
    - Passwords are hashed using a strong algorithm using **bcrypt** before being stored in the database. This is a crucial security measure, preventing passwords from being stored in plain text and protecting them even if the database is compromised.
    - The email is marked as verified only after successful OTP entry using **SMTP** (Simple Mail Transfer Protocol).

    - Email uniqueness is enforced to prevent multiple accounts from being created with the same email address.

Login Page:

The login page allows registered users to securely access their accounts.

- **Credential Input**: Users enter their email and password.
- **OTP Verification (Two-Factor Authentication)**: Similar to signup, after submitting credentials, the backend sends an OTP to the user's registered email. This acts as a second factor of authentication, significantly enhancing account security. The user must enter this OTP to complete the login process.
- **Session Management**:
    - Upon successful OTP verification, the backend generates a JSON Web Token (JWT).
    - The frontend stores this JWT token and the user's email in local storage. This token acts as a digital key, authenticating the user for the duration of their session.
    - The JWT token is then included in the headers of all subsequent API requests to access protected resources, ensuring that only authenticated users can interact with the platform's features.
- **Security**:
    - OTPs have a short expiration period, reducing the window for potential misuse.
    - All sensitive operations on the platform are gated, requiring a valid JWT token for authorization.

Dashboard (After Login):

The dashboard is the central hub for authenticated users, providing access to the platform's core functionalities.

- **Import Section**:
  - Users can upload Excel/CSV files containing their data.
  - These files are sent to the backend and stored securely in MongoDB using GridFS. GridFS is a specification for storing and retrieving files that exceed the BSON-document size limit, making it ideal for handling large data files.
- **Analysis Section**:
  - Users select an uploaded file for analysis.
  - The backend then processes the file, extracts column headers, and computes summary statistics (e.g., mean, median, mode, standard deviation) to provide initial insights into the data.
- **Visualization Section**:
  - Users can choose from a variety of chart types (e.g., bar, line, pie, doughnut, radar, polar area).
  - They select the columns for the X and Y axes based on their analysis needs.
  - The backend generates the chosen chart based on the selected data and returns it to the frontend for display.
- **Reports Section**:
  - Users can generate and download reports based on their data analysis. These reports can be exported in common formats like PDF or CSV.

Administrative Module:

Admin Dashboard:

The Admin Dashboard is a restricted area accessible only to users with administrative privileges, providing tools for platform management.

- **User Management**: Admins can view, edit, or remove user accounts. They can also manage admin privileges, granting or revoking administrative access to other users.
- **File Overview**: Provides a comprehensive view of all uploaded files across the platform, allowing admins to monitor data usage and potentially identify issues.
- **Report Management**: Admins have access to and can manage all generated reports, which could involve reviewing content or deleting outdated reports.

<u>Core System Integrations & Security:</u>

Contact Integration:

The contact form on the landing page is seamlessly integrated with the backend. User submissions are sent to a backend API, which then stores the message and/or forwards it to the admin's email, ensuring that inquiries are promptly addressed.

Authentication & Security:

Security is a cornerstone of this platform, with multiple layers implemented throughout the user journey.

- **Signup Security:**
  - User details undergo validation before storage.
  - Email verification via OTP is mandatory, ensuring legitimate email addresses are linked to accounts.
- **Login Security**:

- Two-step authentication (2FA) is enforced, requiring both a password and an OTP for login. This significantly reduces the risk of unauthorized access even if a user's password is compromised.
- JWT tokens are used for secure session management, providing a stateless and efficient way to verify user authenticity for subsequent requests.
- Tokens and user emails are stored in local storage, providing persistent login sessions within the browser.

- **Password Security**:
  - Passwords are hashed using bcrypt (a strong, adaptive hashing algorithm) before being stored in the database. This one-way hashing makes it virtually impossible to reverse-engineer passwords from the stored hash.
- **Session Security**:
  - All protected routes (features requiring user authentication) are designed to only be accessible with a valid JWT token, preventing unauthorized access to sensitive data and functionalities.

File Handling & Analysis:

The platform's ability to handle and process data files is a core feature.

- **Upload**: Users can easily upload Excel/CSV files. These files are then securely stored in MongoDB using GridFS, which is optimized for handling large binary files.
- **Analysis**: The backend intelligently processes the uploaded files, extracting columns and computing essential summary statistics. This provides users with immediate insights into their data.
- **Visualization**: A wide array of chart types (bar, line, pie, doughnut, radar, polar area) are supported, allowing users to visually represent their data in diverse and meaningful ways.

- **Export**: The results of the analysis and generated reports can be conveniently exported as PDF or CSV files, facilitating sharing and further use of the insights.

End-to-End User Journey:

This section outlines the typical workflow for different types of users on the platform:

1. **New User Registration**:
   - A new user navigates to the landing page and clicks "Sign Up."
   - They fill in their name, email, and password.
   - An OTP is sent to their email, which they enter to verify their identity.
   - Upon successful verification, their account is created and activated.
2. **User Login**:
   - A registered user enters their email and password.
   - An OTP is sent to their email, which they enter to complete the login.
   - A JWT token and their email are stored in local storage, establishing their session.
3. **File Upload & Analysis**:
   - The user uploads an Excel/CSV file, which is securely transferred to the backend and stored in GridFS.
   - They then select the uploaded file for analysis.
   - The backend processes the file, extracting columns and computing statistics.

    - The user selects a desired chart type and specifies the X and Y axes, prompting the backend to generate the visual representation.

4. **Report Generation**:
    - After analysis, the user can generate a detailed report.
    - This report can then be downloaded in either PDF or CSV format.

5. **Admin Management**:
    - An administrator logs in using their specialized admin credentials.
    - They gain access to the Admin Dashboard, allowing them to view and manage users, uploaded files, and generated reports.

6. **Contact Form**:
    - Any user (registered or not) can submit a message through the contact form on the landing page.
    - The message is securely transmitted to the backend and then forwarded to the administrator for prompt action.

Technical Stack:

The platform is built using a modern and robust technology stack, ensuring high performance, scalability, and maintainability.

Frontend:

- **React**: A popular JavaScript library for building user interfaces, known for its component-based architecture and efficiency.
- **Vite**: A fast build tool that provides a rapid development experience for web projects.
- **Tailwind CSS**: A utility-first CSS framework for rapidly building custom designs without leaving your HTML.

- **Chart.js**: A flexible JavaScript charting library for creating various types of data visualizations.
- **Axios**: A promise-based HTTP client for making API requests from the frontend to the backend.
- **React Router**: A standard library for routing in React, enabling declarative navigation within the application.
- **Framer Motion**: A production-ready motion library for React, making it easy to create animations and gestures.

Backend:

- **Node.js**: A JavaScript runtime built on Chrome's V8 JavaScript engine, enabling server-side execution of JavaScript.
- **Express.js**: A fast, unopinionated, minimalist web framework for Node.js, used for building APIs.
- **MongoDB (with GridFS)**: A NoSQL document database used for storing all platform data, including user information, file metadata, and analysis results. GridFS is specifically used for efficient storage of large files (Excel/CSV).
- **Mongoose**: An object data modeling (ODM) library for MongoDB and Node.js, simplifying interactions with the database.
- **JWT (JSON Web Tokens)**: Used for secure, stateless authentication and authorization.
- **Multer**: A middleware for Node.js used for handling multipart/form-data, primarily for file uploads.
- **Nodemailer**: A module for Node.js applications to allow easy email sending, used for sending OTPs and contact form messages.
- **Zod**: A TypeScript-first schema declaration and validation library, ensuring data integrity and type safety.

- **Passport (Google OAuth)**: An authentication middleware for Node.js, specifically used here for integrating Google OAuth for alternative login methods (though "Google OAuth" is only listed in the tech stack, its implementation details are not explicitly described in the flow, so it's a potential future or optional feature).

Security:

- **JWT authentication**: Standard for securing API endpoints.
- **OTP verification**: Adds a crucial second layer of authentication.
- **Bcrypt password hashing**: Protects user passwords from being compromised.
- **CORS (Cross-Origin Resource Sharing)**: Properly configured to prevent unauthorized cross-origin requests.
- **Input validation**: Ensures that all data received from users is clean and conforms to expected formats, preventing common vulnerabilities like injection attacks.

API Endpoints (Summary):

This provides a quick reference for the various API endpoints available on the platform, categorizing them by purpose and indicating their HTTP methods.

- **/api/auth/send-otp (POST):** Initiates the sending of an OTP for user registration.
- **/api/auth/verify-otp (POST):** Verifies the OTP entered during registration.
- **/api/auth/register (POST):** Handles the actual user registration process after OTP verification.
- **/api/authLogin/send-login-otp (POST):** Sends an OTP to the user's email for login verification.

- **/api/authLogin/verify-login-otp (POST):** Verifies the OTP entered during the login process.
- **/api/v1/files (POST):** Used for uploading new data files.
- **/api/v1/files/getfiles (GET):** Retrieves a list of files uploaded by the current user.
- **/api/v1/files/:fileId (DELETE):** Deletes a specific file based on its ID.
- **/api/v1/analysis (GET):** Retrieves a summary of the analysis for a selected file.
- **/api/v1/analysis/generate (GET):** Generates a specific analysis or chart based on user selections.
- **/api/contact (POST):** Handles submissions from the contact form.

Data Models (Backend):

These are the blueprints for the data stored in the MongoDB database, defining the structure and types of information for each entity.

- **User_Registration**:
  - name: User's full name.
  - email: User's email address (unique).
  - password: Hashed password.
  - isAdmin: Boolean flag indicating administrator privileges.
  - googleId: (Optional) ID if registered via Google OAuth.
  - isEmailVerified: Boolean flag indicating if email has been verified via OTP.
  - signupComplete: Boolean flag indicating if the full signup process (including OTP verification) is complete.

- **Admin**:
  - name: Admin's name.
  - email: Admin's email address (unique).
  - password: Hashed password.
- **File**:
  - gridFsId: Reference to the file stored in GridFS.
  - originalName: Original name of the uploaded file.
  - size: Size of the file.
  - userEmail: Email of the user who uploaded the file.
  - status: Current processing status of the file.
  - columns: Array of column headers extracted from the file.
  - timestamps: Metadata for creation and update times.
- **Analysis**:
  - userEmail: Email of the user who performed the analysis.
  - fileId: Reference to the file being analyzed.
  - fileName: Name of the file being analyzed.
  - chartType: Type of chart selected for visualization.
  - xAxis: Column selected for the X-axis.
  - yAxis: Column selected for the Y-axis.
  - reportGridFsId: (Optional) Reference to the generated report in GridFS.
  - createdAt: Timestamp of creation.
  - updatedAt: Timestamp of last update.

- **Otp**:
    - email: Email address for which the OTP was generated.
    - otp: The One-Time Password itself.
    - type: (e.g., 'registration', 'login') indicating the purpose of the OTP.
    - expiresAt: Timestamp when the OTP becomes invalid.

<u>Security & Best Practices:</u>

The platform adheres to several security best practices to protect user data and ensure system integrity.

- **Data Validation and Sanitization**: All incoming sensitive data is rigorously validated and sanitized to prevent common vulnerabilities like SQL injection, cross-site scripting (XSS), and other data manipulation attacks.
- **Password Hashing**: Passwords are never stored in plain text; instead, they are hashed using bcrypt before being saved to the database. This makes it impossible to retrieve original passwords even if the database is breached.
- **JWT Tokens for Protected Routes**: JSON Web Tokens (JWTs) are used for every protected route, ensuring that only authenticated and authorized users can access specific functionalities and data.
- **OTP for Registration and Login**: The mandatory use of OTPs for both registration and login significantly enhances the security of user accounts, providing a robust second factor of authentication.
- **CORS and Secure File Upload Handling**: CORS (Cross-Origin Resource Sharing) is properly configured to prevent unauthorized requests from different domains. Additionally, secure file upload

handling mechanisms are in place to prevent malicious file uploads and ensure file integrity.

This detailed breakdown provides a comprehensive understanding of the platform's architecture, features, and underlying security measures.