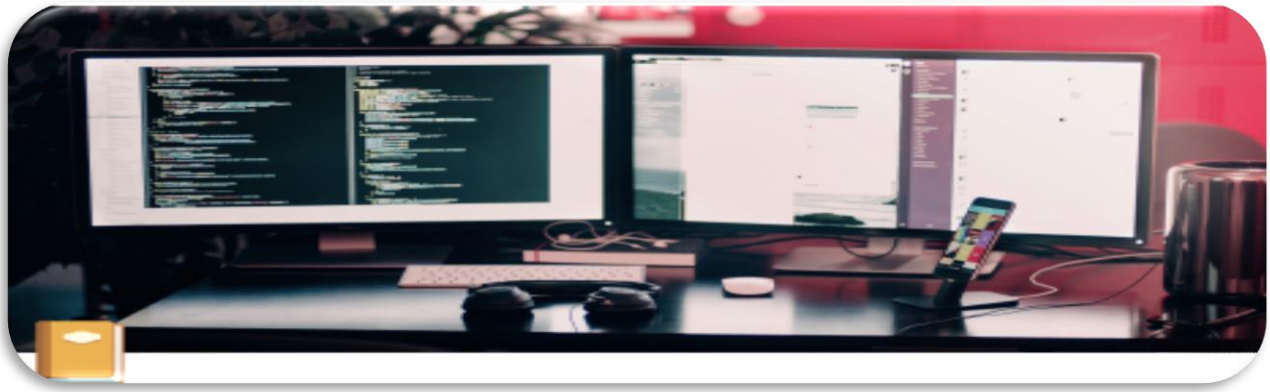


System Design Report for TextUp - Notes Taking Application

1. **System Architecture:**
   The TextUp application should follows a client-server architecture. It should consist of three main components: the client-side, the server-side, and the database.

- Client-Side: The client-side is responsible for rendering the user interface and handling user interactions. It is implemented using React.js, HTML, and CSS. The client-side communicates with the server-side through API endpoints.

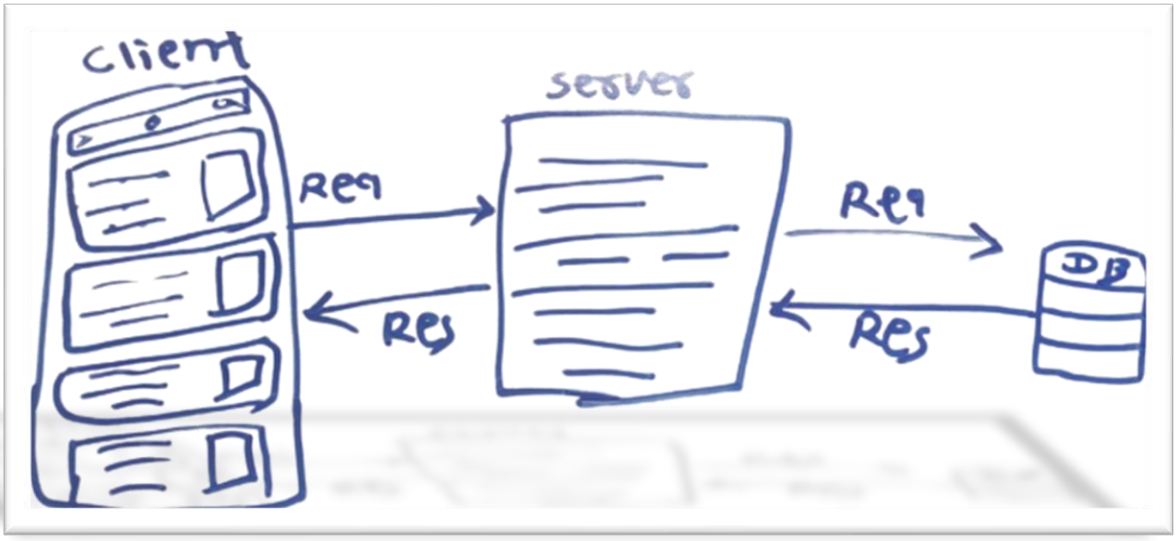
- Server-Side: The server-side handles the business logic and serves as the intermediary between the client-side and the database. It is implemented using Node.js. The server-side receives requests from the client-side, processes them, interacts with the database, and sends back the appropriate responses.

- Database: The database stores the application data, including user information, notes, and images. The application can use MongoDB as its database management system (DBMS) for storing user information and notes. MongoDB is a NoSQL database that provides flexibility and scalability for storing and retrieving data. To store images the application can use Cloudinary. it is a cloud-based image and video management platform that helps businesses of all sizes deliver fast, engaging visual experiences.

2. Data Flow:

The data flow in the TextUp application must involves the following steps:

- User Interactions: Users interact with the client-side user interface, such as creating, editing, and deleting notes, changing settings, and uploading images.
- Client-Server Communication: When a user performs an action, such as creating a note, the client-side sends an HTTP request to the server-side API endpoint associated with that action.
- Server-Side Processing: The server-side receives the request, extracts the necessary data, and performs the required operations. It interacts with the database to store or retrieve data as needed.
- Database Interaction: The server-side communicates with the MongoDB database to store notes, and user information. It uses MongoDB queries to read or modify data in the database. Images are directly get stored on Cloudinary from client-side, only image delete request is communicates through server-side.
- Response to Client: After processing the request and interacting with the database, the server-side sends an appropriate HTTP response back to the client-side.
- Client-Side Updates: Upon receiving the response, the client-side should update the user interface based on the server's response. For example, it may display a success message, refresh the note list, or show an error notification.



3. Modules and Components:

The key modules and components should be present in the TextUp application are as follows:

- Client-Side (React.js):
  - NoteList: Renders the list of notes and allows users to create, edit, or delete notes.
  - NoteEditor: Provides a text editor interface for creating or editing notes.

    - ImageUploader: Allows users to upload images and associate them with their notes.
    - Settings: Handles user preferences and settings such as changing account password.
    - Navigation: Facilitates navigation between different screens and components.
    - API Services: Handles HTTP requests to the server-side API endpoints.

- Server-Side (Node.js):
    - Express.js: Provides the framework for handling HTTP requests and defining API endpoints.
    - Authentication Middleware: Handles user authentication and authorization.
    - User Routes: Handles user-related API endpoints, such as registration, login, and account settings.
    - Note Routes: Handles note-related API endpoints, such as creating, updating, and deleting notes.
    - Image Routes: Handles image-related API endpoints, such as image deletion.
    - Email Service: Sends email notifications, such as email change requests.
    - Database Access Layer: Provides an interface to interact with the MongoDB database.

- Database (MongoDB):
    - Users Collection: Stores user information, including email, password, and preferences.
    - Notes Collection: Stores note data, including title, content, and associated user ID.

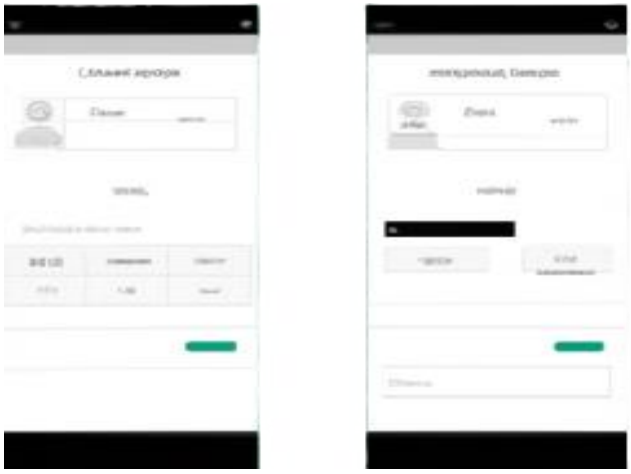
- Database (Cloudinary):
    - Image Collection: Stores user images.

4. User Interface:

The user interface components in the TextUp application must be designed to provide A seamless note-taking experience. Some key components include:

- NoteList: Displays a list of notes, allowing users to view, edit, or delete individual notes.
- NoteEditor: Provides a text editor interface for creating or editing notes, including options for formatting and organizing content.
- ImageUploader: Allows users to upload images and associate them with their notes, providing a visual representation for their notes.
- Settings: Enables users to customize their profile, including user profile or account access settings.
- Navigation: Provides navigation between different screens and components, allowing users to switch between the note list, note editor, and settings.

The user interface allows users to interact with their notes, create new notes, modify existing notes, upload images, and customize their profile settings.



5. Security:

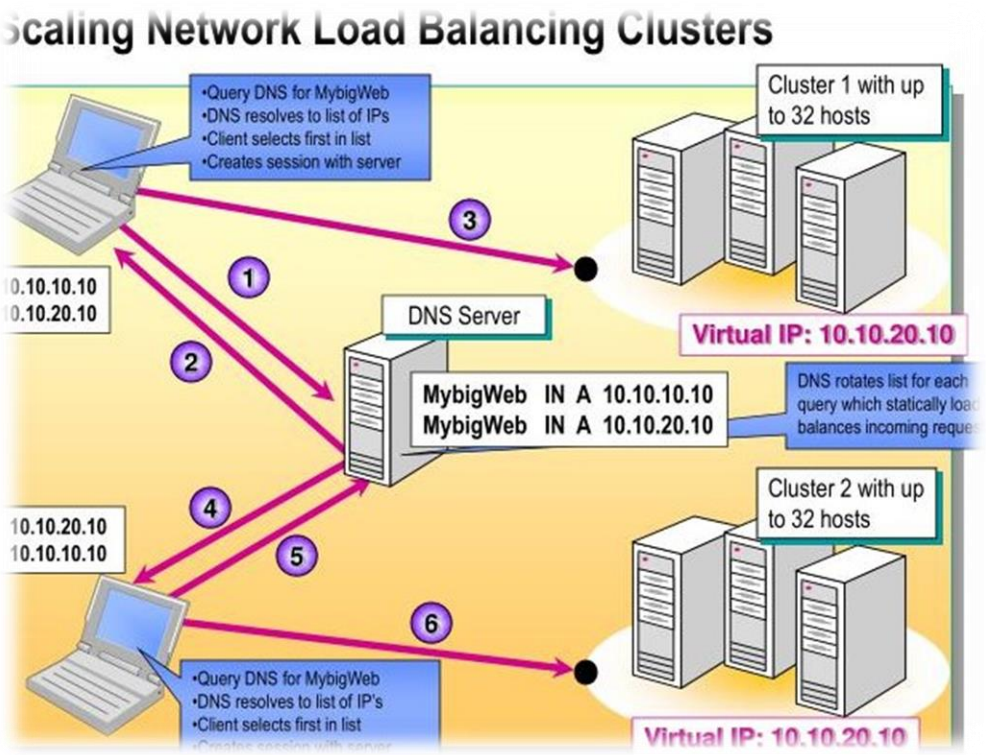
The TextUp application must focus on implementing several security measures to protect user data and ensure data privacy:

- User Authentication: The application requires users to register and login using their email and password. Passwords are securely hashed and stored in the database.
- Authorization: Certain API endpoints and actions require user authentication and authorization to ensure that only authorized users can access or modify their data.
- Password Hashing: User passwords are securely hashed using a strong hashing algorithm, such as bcrypt, to prevent unauthorized access even if the database is compromised.
- Input Validation: The application validates user inputs to prevent common vulnerabilities like SQL injection or cross-site scripting (XSS) attacks.
- Email Verification: The application uses email verification for important actions like changing email addresses to ensure that only the account owner can make such changes.
- HTTPS: The application should use HTTPS to encrypt the communication between the client and server, protecting user data during transmission.

6. Scalability and Performance:

To address scalability and performance considerations, the following strategies must be employed:

- Caching: Implement caching mechanisms to store frequently accessed data in memory, reducing the load on the database and improving response times.
- Load Balancing: Distribute incoming traffic across multiple servers to handle increased user load and improve application performance.
- Horizontal Scaling: Add more servers to the application infrastructure to distribute the load and handle increased user demand.
- Database Indexing: Properly index the database to optimize query performance, especially for frequently executed queries or queries involving large data sets.
- Asynchronous Processing: Utilize asynchronous programming techniques to handle concurrent requests and improve overall system throughput.
- Monitoring and Performance Testing: Regularly monitor application performance, identify bottlenecks, and perform load testing to ensure the system can handle increased user load.



7. Integration and Interfaces:

The TextUp application should be interacts with the following external services and interfaces:

- MongoDB Database: The application can use MongoDB as its text database management system to store and retrieve user data, notes.
- Email Service Provider: The application can integrate with an email service provider to send email notifications, such as email change requests or account verification emails.
- Cloudinary Database: The application can use Cloudinary as its image database management system to store and retrieve user note images data.

The application can interact with these services via API endpoints and utilizes the respective libraries or SDKs provided by these services to establish the connection and exchange data.

8. Deployment and Infrastructure:

To successfully deploy the TextUp application, the following infrastructure components are must be required:

- Web Server: A web server, such as Nginx or Apache, to serve the client-side React.js application and handle incoming HTTP requests.
- Node.js Runtime: A Node.js runtime environment to execute the server-side application code.
- MongoDB Database Server: A MongoDB database server to store and retrieve application data.
- Network Infrastructure: Proper networking setup, including firewalls and load balancers, to handle incoming traffic and ensure security.

The deployment process typically involves setting up the necessary infrastructure components, configuring the web server, deploying the client-side and server-side codebases, and establishing connections with the MongoDB database server. Continuous integration and deployment (CI/CD) pipelines can be implemented to automate the deployment process.

Name: Pranav Dharme
Role: SDE Intern
Company: Ace Online
Contact here