# FORM-3 METHODOLOGY

**Team No:** 7

**Project Title:** DESIGNING A WEB BASED CHAT INTERFACE SERVER

**Proposed Method:**

**Set Up Your Environment:**
Install Node.js and npm for server-side development.
Install MongoDB for your database.
Set up a new React app using Create React App for the client-side.

**Server-side (Node.js, Express, MongoDB):**
Create a new Express application.
Set up routes for handling chat-related functionalities.
Create a MongoDB schema for storing messages.
Implement controllers to handle business logic.
Connect your Express app to MongoDB using Mongoose.

**Client-side (React):**
Design your chat interface components (e.g., chat window, message input).
Use state to manage messages and other relevant data.
Connect to the server-side using the fetch API or a library like Axios for HTTP requests.
Implement functionalities to send and receive messages.

**MVC Architecture:**
Organize your server-side code into separate folders for models, views, and controllers.
Models: Define MongoDB schemas and handle database interactions.
Views: Render the UI components using React.
Controllers: Implement the application logic, handling requests and responses.

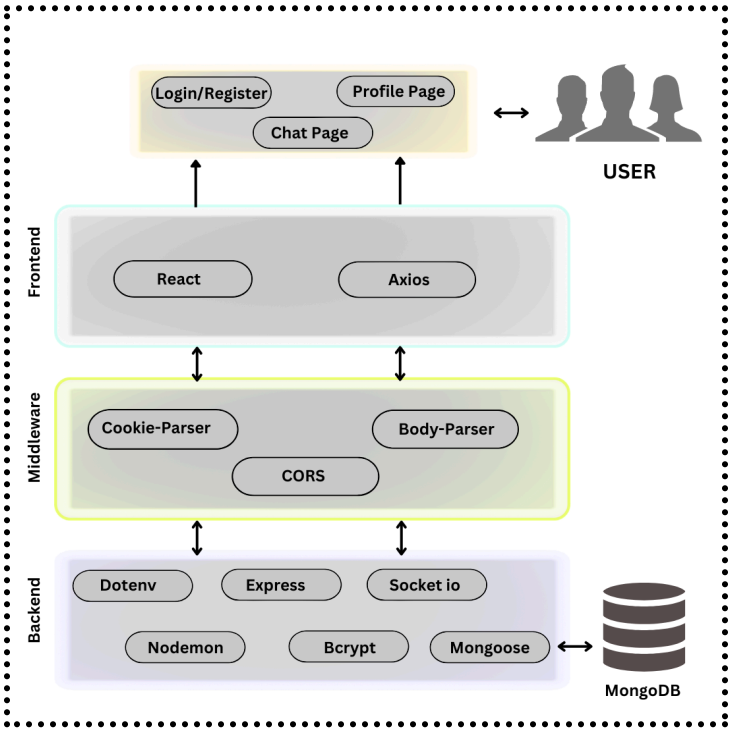
**Real-time Communication:**
Implement WebSocket communication for real-time updates.
Use a library like Socket.io for managing WebSocket connections.
Update the client and server logic to handle real-time messaging

**User Authentication:**
Implement user authentication to secure the chat application.
Use JSON Web Tokens (JWT) for authentication and authorization.
Protect your routes to ensure only authenticated users can access the chat.

**Testing:**
Write unit tests for server-side and client-side components.
Ensure that the application functions correctly under different scenarios.

**Deployment:**
Deploy your MongoDB database, Express server, and React app to a hosting service like Heroku or AWS.
Configure environment variables for sensitive information.

## Proposed Method illustration




**Signature of the Supervisor**