**Building a Real-Time Notification System with Socket.IO, Express, and MongoDB**

**Introduction:**

Using Node.js, Express, Socket.IO, and MongoDB, we created a real-time notification system. Subscribers could receive real-time notifications of changes to their favourite IPL teams. An additional challenge was to implement subscriptions at the field level of a document.

**Steps Taken:**

1. Admin Role: Only administrators can create IPL teams.
2. User Subscription: Users can subscribe to teams created by administrators.
3. Real-Time Updates: Logged-in users receive instant updates when a subscribed team is modified or deleted.

**Solution Approach:**

1. Express and MongoDB: Chose Express for backend and MongoDB for data storage due to their flexibility and integration ease.
2. Socket.IO: Used for real-time bidirectional communication between server and clients.
3. Field-Level Subscriptions: Enabled subscriptions at the field level, ensuring users receive updates only for specific fields of interest.

**Additional Points:**

1. Postman Collections Submitted: All APIs are documented in Postman collections for easy access and reference. All The Response is saved in Postman.
2. JWT Authentication: Implemented JSON Web Token (JWT) authentication to secure routes, allowing only authenticated users, especially administrators, to perform sensitive operations.
3. Error Handling and Validation: Implemented robust error handling and validation mechanisms to enhance application reliability and security.
4. Express Middleware’s: Utilized Express middleware’s for authentication verification, restricting access to certain routes to administrators only.
5. Scalability Considerations: Designed the system with scalability in mind, anticipating future expansion, increased user base, and additional features.

**Usage Manual:**

Development Requirements: Node.js and npm installed.

MongoDB installed and running.

API Usage:

1. **Socket.IO Connection:** Once all the Packages are installed Run this on your browser for Socket connection [IPL Notifications](http://localhost:3000/)
2. Create a Team
3. Register a User
4. Login with the Account
5. Subscribe to a Team
6. Receive Real-Time Updates

**Conclusion:**

Our approach, using Express, Socket.IO, and MongoDB, sets a sturdy foundation for a real-time notification system. It's adaptable, can handle growth, and delivers updates swiftly. By allowing users to subscribe to specific fields, we ensure they receive only relevant notifications.

**Future Considerations:**

Continuous testing and integration for ongoing reliability.

Implementation of additional features based on user feedback.

Regular updates to dependencies and security patches.

Only few Routes where considered need to add more functionality and error handlers.

**Sources:**

npm packages: Utilized for essential functionalities.

YouTube tutorials: Consulted for practical guidance and insights.

Bing AI: Used for documentation support and generation.

Stack Overflow: Tapped into community discussions for troubleshooting and additional insights.

ChatGPT: HTML content generation and assistance

Note: All responses displayed on the client have been simplified for ease of understanding, considering limited knowledge in HTML.