System Architecture for ElevanceNet v1.0.0

1. Overview

ElevanceNet leverages a modern client-server architecture to provide a scalable, reliable, and distraction-free social networking platform for professionals. The architecture integrates frontend, backend, database, and caching layers with seamless CI/CD pipelines and efficient deployment strategies.

---

2. Architecture Overview

2.1 Architecture Type

- Client-Server Architecture

2.2 Components

- Frontend: React.js (Deployed on Netlify)
- Backend: Node.js with Express.js (Deployed on Vercel)
- Database: MongoDB (Hosted on MongoDB Atlas)
- Cached Server: Redis (Hosted on Upstash)
- CI/CD: GitHub Actions
- Containerization: Docker (Images stored on Docker Hub)

---

3. Component Details

3.1 Frontend

- Technology: React.js
- Features:
    - Responsive design for desktop and mobile devices.
    - User interface optimized for professional usage.
    - Minimal distractions with a clean and professional layout.
- Deployment: Hosted on Netlify.

3.2 Backend

- Technology: Node.js with Express.js
- Features:
    - RESTful API for communication with the frontend.
    - Authentication using JWT and email verification.
    - Handling of business logic for feed curation, post management, and reporting.
- Deployment: Hosted on Vercel for high availability and scalability.

3.3 Database

- Technology: MongoDB
- Hosting: MongoDB Atlas
- Features:
    - Document-based storage for flexible and scalable data modeling.
    - Entities: User, Post, Comment, Like, Report, Feedback, Education, Work, Location, Social Links, User Preferences.

3.4 Cache Layer

- Technology: Redis
- Hosting: Upstash
- Features:
    - Caching frequently accessed data to improve performance.
    - Reducing latency for feed generation and user searches.

---

4. Deployment Workflow

4.1 Version Control

- Version Control System: Git
- Repository Hosting: GitHub

4.2 Continuous Integration and Delivery

- Tool: GitHub Actions
- Pipeline:
    1. Code pushed to GitHub.

2. Automated testing and linting.
3. Docker images built and pushed to Docker Hub.
4. Deployment triggered for frontend and backend.

4.3 Containerization

- Tool: Docker
- Repository: Docker Hub
- Usage:
    - Backend and frontend services packaged into Docker containers.
    - Ensures consistent environments across development, staging, and production.

4.4 Production Deployment

- Frontend: Deployed on Netlify for responsive and fast delivery.
- Backend: Deployed on Vercel for serverless scalability.

---

5. Communication Interfaces

- Frontend to Backend:
    - RESTful API endpoints for user authentication, feed retrieval, post management, and reporting.
- Backend to Database:
    - Secure database queries using MongoDB drivers.
- Backend to Cache:
    - Redis integration for faster data retrieval.

---

6. Scalability and Future Enhancements

6.1 Scalability

- Horizontal Scaling:
    - Frontend: Netlify’s CDN ensures fast delivery worldwide.
    - Backend: Vercel’s serverless architecture handles high traffic efficiently.
    - Database: MongoDB Atlas’s shard-based scaling supports increased data loads.

- Caching: Upstash Redis reduces backend load by caching frequently accessed data.

6.2 Planned Enhancements

- Chat System: Real-time messaging using WebSockets.
- Follow System: Build user networks with follow/unfollow functionality.
- Online Status: Display user availability with real-time presence updates.

---

7. Security and Compliance

- Authentication: JWT-based secure token authentication with email verification.
- Data Protection: Encrypted storage for sensitive data like passwords.
- Uptime: Achieving 99.9% availability via redundant infrastructure.
- Compliance:
    - GDPR and other relevant privacy regulations.

---

8. Diagrams

8.1 Entity-Relationship Diagram (ERD)

- Entities: User, Post, Comment, Like, Report, Feedback, Education, Work, Location, Social Links, User Preferences.
- Relationships:
    - A user can create multiple posts, comments, likes, and reports.

8.2 User Flow Diagram

- Authentication: Signup, Login, Email Verification.
- Main Page Navigation: Profile, Feed, Posts, Settings.
- Post Interaction: Like, Comment, Report.

---

9. Conclusion

The architecture of ElevanceNet ensures a robust, scalable, and distraction-free professional networking platform. By leveraging modern technologies and adhering to best practices, ElevanceNet is well-positioned to provide a seamless user experience while maintaining high performance and reliability.