🧠 CodeSync.ai – Backend Functional Specification (Delta Version Control Edition)

🎯 Purpose

CodeSync.ai ka main goal hai ek **real-time collaborative IDE** create karna jahan developers ek hi workspace me coding, chatting, aur testing kar saken — bina traditional Git workflows ke.
Backend ka sabse strong module hai **Delta-Based Version Control Engine**, jo real-time edits ko efficiently track, merge aur restore karta hai — without manual commits or PRs.

⚙️ Core Backend Architecture

- **Framework:** Node.js + Express
- **Database:** MongoDB Atlas (cloud-hosted scalable DB)
- **Realtime Engine:** Socket.io
- **Sync Model:** CRDT (Conflict-free Replicated Data Type)
- **Versioning Engine:** Delta-based Version Control System
- **Testing Integration:** MCP with TestSprite for automated validation

---

🧮 Delta-Based Version Control System

🧩 Concept

Instead of storing full copies of code files, CodeSync.ai ka backend har change ko **delta (difference)** ke form me record karta hai.
Matlab, jab bhi user code edit karta hai, sirf **changed characters or lines** store hoti hain — pura file nahi.
Yeh approach storage efficient bhi hai aur fast rollback ke liye perfect hai.

⚙️ How It Works

1. **Change Detection:**
   Every keystroke ya line edit backend ke through Socket.io se capture hota hai.
   System identifies the exact delta (additions, deletions, or modifications).

Delta Storage:
Har delta MongoDB me JSON format me store hota hai —

```
{
  "session_id": "abc123",
  "timestamp": "2025-11-02T12:10:00Z",
  "delta": "+ line added / - line removed",
  "user_id": "u789"
}
```

2. Har delta uniquely identifiable hota hai aur CRDT layer ensure karti hai ki koi conflict na ho jab multiple users ek file edit kar rahe ho.

3. **State Reconstruction:**
   File ka latest version server "base file + all deltas" apply karke generate karta hai.
   Fast memory diff engine ensure karta hai ki reconstructing large files bhi instant ho.

4. **Version Restore:**
   Users kisi bhi timestamp pe revert kar sakte hain.
   System reverse deltas apply karke past stable version restore karta hai — bina Git commits ke.

5. **Auto Snapshotting:**
   Periodically, delta chain se ek consolidated snapshot create hota hai (for performance optimization).
   Isse rollback time aur memory usage dono efficient rehte hain.

🧰 Technical Implementation

- **Delta Engine:** Custom Node.js diff-patch module integrated with CRDT layer.

- **Transport Layer:** Socket.io ensures low-latency delta transmission among all collaborators.

- **Storage:** MongoDB GridFS for snapshot blobs, standard collections for delta logs.

- **Restore API:**
    - `GET /api/delta/:session_id` → fetches all deltas.
    - `POST /api/delta/restore` → reconstructs desired version.

🚀 Advantages

- Ultra-lightweight version control — only changes stored, not entire files.
- Real-time synchronization with guaranteed consistency.
- Fast rollback & replay system (no Git required).
- Handles 10+ parallel editors without conflict.
- Ideal for hackathons, AI coding bootcamps, and remote dev teams.

🔒 Security Layer

- JWT authentication for each delta stream.
- Encrypted sockets (WSS) for data privacy.
- User-level access control for restore & diff APIs.