**What is a Gantt Chart?**

A Gantt chart is a visual project management tool that represents a project schedule over time. It provides a timeline view of tasks, their durations, dependencies, and progress. Gantt charts are particularly useful for organizing and tracking the work of teams, especially when multiple tasks need to be completed in parallel or sequentially.

Key components of a Gantt chart:

1. Tasks : The individual activities or milestones required to complete the project.
2. Timeline : A horizontal axis showing the duration of the project (e.g., days, weeks, months).
3. Bars : Horizontal bars representing the start and end dates of each task.
4. Dependencies : Lines connecting tasks to show which tasks must be completed before others can begin.
5. Milestones : Important checkpoints or deliverables marked on the timeline.

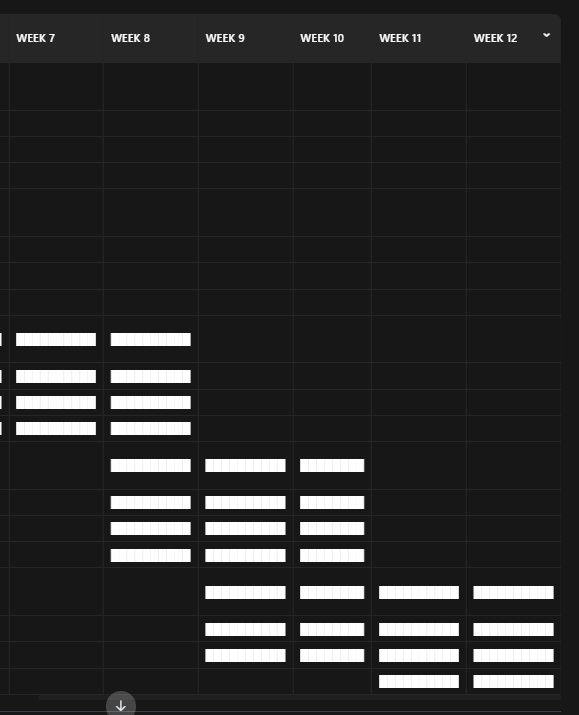
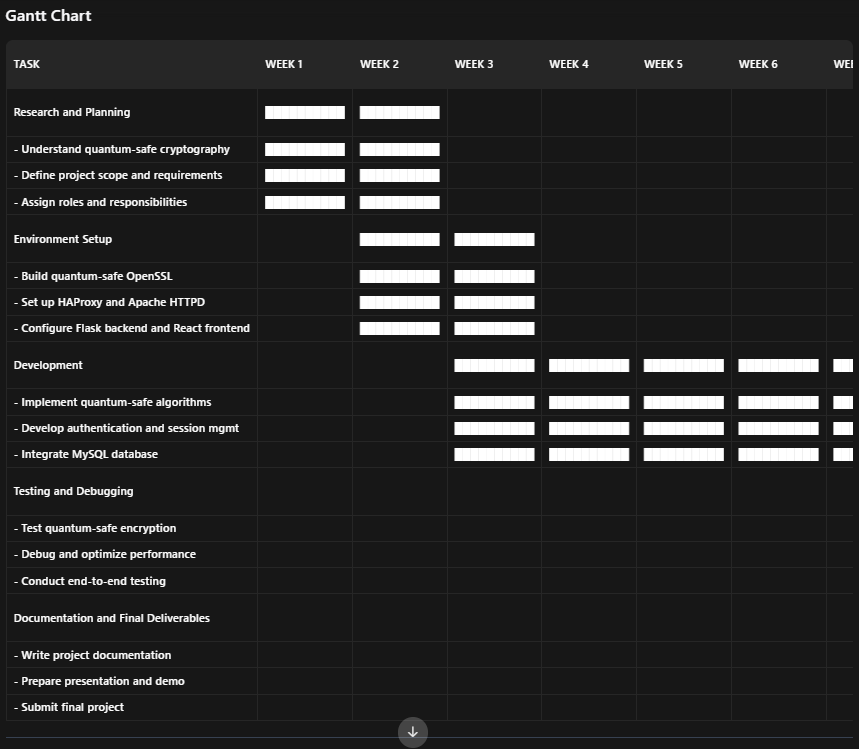
**Gantt Chart for Quantum-Safe Messaging Project**

Below is a Gantt chart for your quantum-safe messaging project. The project spans 3 months (12 weeks) , starting from today. The team consists of 4 students, so tasks are divided into manageable chunks with clear responsibilities.

**Project Phases**

The project is divided into the following phases:

1. Research and Planning (Weeks 1–2)
2. Environment Setup and Development (Weeks 3–8)
3. Testing and Debugging (Weeks 9–10)
4. Documentation and Final Deliverables (Weeks 11–12)



**Explanation of the Gantt Chart**

**1. Research and Planning (Weeks 1–2)**

* Understand quantum-safe cryptography : Study quantum-safe algorithms like Kyber and Dilithium, and their integration with OpenSSL.
* Define project scope and requirements : Clearly outline the features of the messaging app (e.g., login, messaging, quantum-safe encryption).
* Assign roles and responsibilities : **Divide tasks among the 4 team members (e.g., one for frontend, one for backend, one for testing, and one for documentation).**

**2. Environment Setup (Weeks 3–4)**

* Build quantum-safe OpenSSL : Follow the tutorial to build OpenSSL with quantum-safe algorithms using **liboqs** and **oqs-provider**.
* Set up HAProxy and Apache HTTPD : Configure HAProxy for SSL termination and Apache HTTPD as the web server.
* Configure Flask backend and React frontend : Set up the Flask backend for APIs and React for the user interface.

**3. Development (Weeks 5–10)**

* Implement quantum-safe algorithms : Use Kyber for key exchange and Dilithium for signatures in session management and database encryption.
* Develop authentication and session management : Implement session-based authentication with quantum-safe encryption.
* Integrate MySQL database : Store encrypted session data and messages in MySQL.

**4. Testing and Debugging (Weeks 11–12)**

* Test quantum-safe encryption : Verify that all communication and data storage use quantum-safe algorithms.
* Debug and optimize performance : Address any performance bottlenecks caused by quantum-safe algorithms.
* Conduct end-to-end testing : Ensure the entire system works seamlessly from client to server.

**5. Documentation and Final Deliverables (Weeks 11–12)**

* Write project documentation : Document the architecture, setup instructions, and usage of the application.
* Prepare presentation and demo : Create slides and a live demo showcasing the quantum-safe messaging app.
* Submit final project : Deliver the completed project by the deadline.

**Key Notes**

* Dependencies : Tasks like "Set up HAProxy" depend on "Build quantum-safe OpenSSL." Ensure these dependencies are respected.
* Team Collaboration : Regular meetings should be scheduled to track progress and address issues.
* Buffer Time : Include some buffer time in each phase to account for unexpected delays.

This Gantt chart provides a clear roadmap for completing your quantum-safe messaging project within the 3-month timeframe. Let me know if you'd like further adjustments!