Student Task Manager: Cloud-Native Development Plan & Timeline

Application Overview

The **Student Task Manager** is a cloud-native web application designed to help students manage assignments efficiently. Hosted on Google Cloud Platform (GCP), it allows task creation, deadline tracking, and reminders. Our goals are an intuitive user experience, scalability, and security, with delivery due by the end of Week 12 of the trimester.

Development Plan

1. **Requirements of the Application**
    - *Purpose*: Provide students with a simple task management system.
    - *Core Features*:
        - User authentication (login/signup with Firebase Authentication).
        - Task management (Create, Read, Update, Delete - CRUD).
        - Deadline reminders (in-app and email via Cloud Functions).
        - Responsive UI for mobile and desktop.
    - *Goals*: Easy navigation, fast performance, scalability for peak academic periods.
2. **Choosing the Cloud Platform**
    - *Platform*: Google Cloud Platform (GCP), via Deakin University resources.
    - *Justification*: Free tier credits, App Engine auto-scaling, built-in security (IAM, HTTPS).
    - *Services*: App Engine (hosting), Firestore (database), Cloud Functions (notifications).
3. **Architecting the Application**
    - *Data Architecture*: Firestore collections:
        - Users: (ID, email, name).
        - Tasks: (title, due date, status, owner ID).

- *Network Architecture*: HTTPS with GCP Load Balancer for secure traffic.
- *Application Architecture*:
  - Frontend: Vanilla JavaScript for a lightweight, dynamic UI.
  - Backend: Node.js/Express for API handling.
  - Microservices: Cloud Functions for reminders.

4. **Implementing the Application**
   - *Development Strategy*:
     - RESTful APIs in Node.js/Express.
     - Frontend built with vanilla JavaScript for UI and interactivity.
     - Git/GitHub for version control.
     - Environment variables for secure configuration.
   - *Setup Tasks*:
     - Configure Firestore database structure.
     - Implement Firebase Authentication.
     - Build API endpoints for task CRUD.
5. **Testing Strategy**
   - *Unit Testing*: Jest for backend logic (e.g., task validation).
   - *Integration Testing*: Verify frontend-backend-Firestore connectivity.
   - *System Testing*: Simulate multi-user load for scalability.
6. **Deployment & Maintenance**
   - *Deployment*: Deploy to App Engine using gcloud CLI.
   - *Monitoring*: Use Stackdriver for performance and error tracking.
   - *Maintenance*: Fix bugs and optimize post-deployment.
7. **Security & Compliance**
   - *Security*: HTTPS encryption, Firebase Authentication, input sanitization.
   - *Compliance*: Secure storage of user data per privacy standards.

Development Timeline (12 Weeks)

Week	Task
1-2	Define requirements, set up GCP environment
3-4	Backend implementation (API, Firestore)
5-6	Frontend development (JavaScript UI)
7-8	API integration, authentication, Cloud Functions
9-10	Testing (unit, integration, system)
11	Deployment on GCP, monitoring setup
12	Final optimizations, project delivery

Conclusion

The **Student Task Manager** is designed to be a **secure, scalable, and user-friendly** application that assists students in managing their academic workload efficiently. By leveraging **Google Cloud's robust infrastructure**, we ensure **reliability, cost-efficiency, and performance**. This structured **12-week plan** will guide the development process, ensuring the project meets its objectives on time.