Security Implementation Guide for Developers

This guide provides essential security considerations and implementation instructions for developers working on the CoAce web app project. The goal is to ensure every part of the application is built with secure coding standards from the start.

1. Authentication & Authorization

- Enforce strong password policies (min 8 characters, mix of letters, numbers, symbols).
- Use secure authentication flows (e.g., OAuth 2.0 if third-party login is enabled).
- Implement role-based access control (RBAC) to separate privileges for Admins and Students.
- Store passwords securely using hashing algorithms like bcrypt.

2. Input Validation & Error Handling

- Sanitize all user inputs on both frontend and backend.
- Validate inputs using regex or libraries (e.g., Joi for Node.js).
- Prevent common vulnerabilities like XSS, SQL Injection, and CSRF.
- Ensure meaningful but non-technical error messages are shown to users.

3. Secure Session Management

- Use HTTP-only, Secure cookies for session tokens.
- Implement session timeout and auto-logout for inactivity.
- Regenerate session tokens after login.

4. HTTPS & Data Protection

- Use HTTPS across the platform to encrypt data-in-transit.
- Avoid storing sensitive data in local storage or cookies.
- Encrypt sensitive user data in the database (e.g., phone number, DOB if required).

5. API & Backend Security

- Use authentication middleware to protect sensitive API endpoints.
- Implement rate limiting and input validation for API calls.

- Do not expose internal server errors or stack traces to users.

6. Deployment & Monitoring

- Ensure environment variables are used for credentials and keys.
- Remove test accounts and dummy data before deployment.
- Set up logging and basic intrusion detection to monitor suspicious behavior.