Security Requirements Generated (GEN1+GEN2)

1. Implement role-based access control to ensure students can only book exam sessions for their own courses. – Web Application Security

2. Ensure that course coordinators can only modify schedules for courses they are assigned to. – Secure Design

3. Restrict technical staff from viewing or modifying user data without proper authorization. – Data Protection

4. Use secure cryptographic algorithms to store passwords and sensitive data in the database. – Authentication

5. Ensure all API endpoints use HTTPS to protect data in transit. – Secure Coding

6. Implement proper session management to prevent session hijacking. – Web Application Security

7. Sanitize all user inputs to prevent SQL injection in the class scheduling feature. – Input Validation

8. Validate and escape all user-generated content in messages between students and professors. – Client-Side Security

9. Use parameterized queries when accessing the database to prevent injection attacks. – Secure Coding

10. Conduct regular security audits to identify and fix insecure design flaws. – Security Testing

11. Implement rate limiting to prevent brute force attacks on login pages. – Web Application Security

12. Ensure all third-party libraries are up-to-date and free from known vulnerabilities. – Software Maintenance Security

13. Configure security headers to prevent clickjacking and other client-side attacks. – Client-Side Security

14. Disable unnecessary HTTP methods to reduce the attack surface. – Web Application Security

15. Implement proper CORS policies to restrict cross-origin requests. – Secure Design

16. Encrypt all sensitive data stored in the database, including user messages and schedules. – Data Protection

17. Ensure that error messages do not reveal sensitive information about the system. – Privacy Security

18. Use secure cookies with HttpOnly and Secure flags for session management. – Authentication

19. Implement multi-factor authentication for professors and course coordinators. – Authentication

20. Enforce strong password policies for all user accounts. – Web Application Security

21. Monitor and log all authentication attempts for suspicious activity. – Security Testing

22. Ensure that all user data is anonymized in logs and error reports. – Privacy Security

23. Implement data minimization practices to collect only necessary user information. – Secure Design

24. Provide users with the ability to delete their account and associated data. – Data Protection

25. Regularly update and patch the system to address known vulnerabilities. – Software Maintenance Security

26. Conduct penetration testing to identify and remediate security weaknesses. – Security Testing

27. Ensure that all security configurations are documented and reviewed periodically. – Secure Design

28. Monitor system logs for unusual activity that may indicate a security breach. – Security Testing

29. Implement automated alerts for suspicious login attempts. – Web Application Security

30. Ensure that backup and recovery procedures are in place to restore data in case of an incident. – Data Protection

31. All user inputs, including form fields and API parameters, must be validated server-side to prevent injection attacks. – Input Validation

32. Input fields must enforce strict data type and format validation (e.g., dates, emails, numeric IDs) to prevent malformed data processing. – Input Validation

33. User-supplied data used in database queries must be parameterized or sanitized to prevent SQL injection. – Input Validation

Total: GEN1=30, GEN2=3, Total=33

Counting per category (total = gen1 + gen2):

- Authentication: total=3 (gen1=3, gen2=0)

- Client-Side Security: total=2 (gen1=2, gen2=0)

- Data Protection: total=4 (gen1=4, gen2=0)

- Input Validation: total=4 (gen1=1, gen2=3)

- Privacy Security: total=2 (gen1=2, gen2=0)

- Secure Coding: total=2 (gen1=2, gen2=0)

- Secure Design: total=4 (gen1=4, gen2=0)

- Security Testing: total=4 (gen1=4, gen2=0)

- Software Maintenance Security: total=2 (gen1=2, gen2=0)

- Web Application Security: total=6 (gen1=6, gen2=0)