NATIONAL UNIVERSITY
of Computer & Emerging Sciences

| Project Proposal |

Secure E-Commerce Platform

Name: Talha Aamir Malik (22i1572), Muneeb Kashif (22i1659), Abdullah Najaf (22i1583), Abdul Sami (22i2358)

Section: CY - C

Table Of Contents

1. Title of the Project

Secure E-commerce Platform with End-to-End Payment Protection

2. Team Information

- Talha Aamir Malik - Project Lead & Fronted Developer
- Muneeb Kashif - Backend Developer & Database Security Engineer
- Abdullah Najaf - Security Engineer & Tester
- Abdul Sami - Documentation & Compliance Lead

3. Problem Statement

The rapid growth of Pakistan's e-commerce sector (Daraz, Foodpanda, fintech marketplaces) has introduced serious **security challenges**. Customers frequently face risks such as **payment fraud, phishing, weak authentication mechanisms, and data breaches**. Existing platforms often prioritize usability over **security-by-design**, leaving users vulnerable to identity theft and financial loss.

There is a clear need for an **e-commerce system designed around secure software development principles**, ensuring **confidentiality, integrity, and availability (CIA)** of financial transactions and customer data.

4. Objectives of the Project

- Develop an e-commerce platform that embeds **security at every stage of the SDLC**.
- Ensure **secure user authentication** (multi-factor or password-less login).
- Protect **payment transactions** using **end-to-end encryption** and **PCI DSS guidelines**.

- Apply **threat modeling** (STRIDE) and risk mitigation strategies.
- Provide **secure APIs** for payment gateways like Easypaisa, JazzCash, and PayFast.
- Deliver a **prototype with real-time fraud detection features** (e.g., location/device anomaly checks).

5. Proposed Solution

The system will be a **web-based e-commerce application** that integrates security principles throughout design and implementation.

Key Security Features:

- **Authentication & Authorization** → Multi-factor authentication (OTP/email), role-based access control for admins, sellers, buyers.
- **Secure Payments** → End-to-end encryption of transactions, integration with Pakistani gateways.
- **Data Protection** → AES-256 encryption for stored data, TLS 1.3 for communication.
- **Threat Countermeasures** → Defense against SQL injection, XSS, CSRF, brute force, and session hijacking.
- **Fraud Detection** → Risk-based checks (e.g., unusual transaction amount, new device, suspicious IP).
- **Audit & Logging** → Immutable logs for dispute handling and regulatory compliance.

6. Methodology

The project will follow the **Secure Software Development Lifecycle (S-SDLC)**:

1. **Requirements Phase** → Identify functional (shopping cart, payment) and security requirements (PCI DSS, MFA, encryption).

2. **Design Phase** → Threat modeling (STRIDE), secure architecture diagrams, database with least privilege.

3. **Implementation Phase** → Secure coding standards (OWASP ASVS, CERT Secure Coding Guidelines).

4. **Testing Phase** → Static & dynamic security testing (SAST/DAST), penetration testing on prototype.

5. **Deployment Phase** → Secure configuration (HTTPS-only, WAF, database hardening).

6. **Maintenance Phase** → Logging, monitoring, and patch management plan.

7. Tools and Technologies

- **Languages & Frameworks**: Python (Django/Flask) or Node.js (Express), React.js for frontend.

- **Databases**: PostgreSQL / MySQL with encryption.

- **Security Tools**: OWASP ZAP, SonarQube, Bandit (Python), JWT authentication.

- **Payment Gateways**: Easypaisa, JazzCash, PayFast (via sandbox APIs).
- **Encryption**: OpenSSL, bcrypt/Argon2 for password hashing.

8. Expected Deliverables

- Secure e-commerce prototype (web application).
- Documentation (architecture diagrams, threat model, secure SDLC steps).
- Security testing report (SAST, DAST, penetration testing results).
- Final presentation.

9. Timeline

Milestone	**Week**
Requirements & Threat Modeling	Week 1–2
System Design & Database Security	Week 3–4
Implementation (Frontend + Backend)	Week 5–7
Security Integration (MFA, encryption, payment security)	Week 8–9
Testing & Bug Fixing	Week 10–11
Final Report & Presentation	Week 12

10. References

- OWASP Application Security Verification Standard (ASVS) - https://owasp.org/www-project-application-security-verification-standard/

- PCI DSS Standards for Payment Security - https://www.pcisecuritystandards.org/standards/

- Microsoft STRIDE Threat Modeling Framework - https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling

- CERT Secure Coding Practices - https://wiki.sei.cmu.edu/confluence/display/seccode

- NIST Cybersecurity Framework - https://www.nist.gov/cyberframework