SecureMed Capstone Project - Q&A Guide

Quick Reference for Presentations & Defense

Core Project Questions

Q: What problem did you solve?

A: Healthcare organizations struggle to maintain HIPAA compliance and detect security threats. Staff often don't know they're violating privacy rules, and breaches go undetected until it's too late. There's no easy way to train employees on real scenarios or practice incident response.

Q: How did you solve it?

A: We built SecureMed - a web-based HIPAA compliance platform that monitors staff compliance in real-time, provides interactive training on privacy rules, detects violations automatically, and simulates data breaches so organizations can practice their response procedures.

Q: What does your system do?

A: SecureMed provides:

- **Compliance Tracking**: Each staff member gets a score (0-100%) based on their HIPAA knowledge
- **Interactive Training**: 5 real-world scenarios (faxing records, emailing labs, USB requests, etc.)
- **Violation Detection**: Automatically logs when staff make mistakes and alerts administrators
- **Breach Simulation**: Simulates 5 types of attacks (ransomware, phishing, insider threats, etc.)
- **Incident Response**: Provides 20-25 step playbooks for handling each breach type
- **Encrypted Storage**: All patient data (SSNs, PHI) encrypted using Fernet (AES-128)
- **Complete Audit Trail**: Every action logged with timestamp, user, IP address, and details
- **Automated Reports**: Generates professional HIPAA compliance PDFs for auditors

---

Q: What technologies did you use?

A:

- **Backend**: Python Flask web framework
- **Frontend**: React 18 (JavaScript)
- **Database**: SQLite with encrypted fields
- **Encryption**: Fernet symmetric encryption (AES-128 CBC)
- **Security**: SHA-256 password hashing, session management with automatic timeout
- **Reporting**: ReportLab for PDF generation

Why these tools?

- Lightweight and free (no cost for demos)
- Fast development (6-week timeline)
- Industry-standard security libraries
- Works on any laptop without special hardware

Q: What did you accomplish?

A: We created a fully functional HIPAA compliance platform that healthcare organizations can use to train staff, detect violations before they become breaches, and practice incident response. The system demonstrates real-world cybersecurity principles while being accessible enough for educational demos.

Student Outcome (1): Problem Analysis & Solution Design

Q: How did you analyze the computing problem?

A: We started by researching real HIPAA violations from HHS breach reports - things like nurses faxing records to wrong numbers, discussing patients in hallways, or accessing celebrity records out of curiosity. We mapped each real-world violation to a technical requirement:

- Faxing errors → Directory system with pre-approved destinations
- Unencrypted data → Fernet encryption for PHI fields
- No accountability → Audit trail logging every action
- Untrained staff → Interactive training simulator

Q: How did the problem definition change as you worked?

A: Initially we just wanted to store patient records securely. But as we researched HIPAA requirements, we realized we needed:

1. Training system (staff education is required by §164.308)

2. Audit trails (required by §164.312)
3. Breach response procedures (required by §164.308)
4. Session timeouts (unattended workstation protection)

Each requirement became a new feature.

Q: How did you gather requirements?

A: We studied:

- HIPAA Security Rule (45 CFR §164.306-318)
- Real breach notification reports from HHS database
- NIST Cybersecurity Framework guidelines
- Common healthcare security mistakes documented in case studies

We translated legal requirements into technical specs.

Q: What resources did you estimate?

A:

- **Time**: 6 weeks (design, development, testing)
- **Cost**: $0 (all open-source tools)
- **Hardware**: Any laptop with Python 3.8+
- **Team**: 5 members with assigned roles (security, compliance, database, network)
- **Space/Performance**: SQLite handles 100+ patients, ~1000 audit entries without lag

---

Q: What alternative approaches did you consider?

A:

- **MySQL vs SQLite**: Chose SQLite for portability and demo simplicity
- **Built-in encryption vs external**: Used Fernet library (battle-tested, NIST-approved)
- **Angular vs React**: Chose React for faster learning curve
- **Cloud deployment vs local**: Kept local for demo control and no AWS costs

---

Student Outcome (2): Design, Implementation & Testing

Q: What software engineering principles did you apply?

A:

- **Separation of Concerns**: Backend (Flask) separate from frontend (React)
- **DRY (Don't Repeat Yourself)**: Reusable functions for encryption, logging, authentication
- **Security by Design**: Encryption and validation built in from the start, not added later
- **Modular Architecture**: Each feature (training, EDR, audit) is independent
- **Parameterized Queries**: Prevents SQL injection attacks

Q: What design documents did you create?

A:

- Database schema with encrypted field specifications
- API endpoint documentation
- User flow diagrams (admin vs nurse workflows)
- Security architecture diagram
- HIPAA compliance mapping (which features satisfy which regulations)

Q: How did you use multiple programming languages?

A:

- **Python**: Backend logic, encryption, database operations, API endpoints
- **JavaScript**: Frontend UI (React), session management, real-time score updates

---

Q: How did you test your solution?

A:

1. **Unit Tests**: Tested encryption/decryption, password validation, scoring algorithm
2. **Integration Tests**: Verified API endpoints return correct data
3. **Security Tests**: Attempted SQL injection, XSS attacks to verify protection
4. **User Testing**: Created test accounts and ran through all workflows
5. **Compliance Tests**: Verified every action appears in audit trail

Metrics we tracked:

- Session timeout accuracy (measured exactly 2:00 minutes)
- Encryption/decryption success rate (100%)
- Audit trail completeness (every action logged)
- Score calculation accuracy (+20 correct, -10 wrong, 0-100 range enforced)

Q: How did you measure system performance?

A:

- Page load times: <2 seconds for all dashboards
- Database queries: <100ms for patient lookups
- PDF generation: 2-3 seconds for compliance reports
- Encryption overhead: <50ms per field

Student Outcome (4): Legal & Ethical Responsibilities

Q: What legal issues did you address?

A:

- **HIPAA Security Rule (§164.306-318)**: Implemented encryption, access controls, audit logs
- **HIPAA Privacy Rule (§164.502)**: Minimum necessary principle (directory system)
- **Breach Notification Rule (§164.408)**: 60-day timeline in our playbooks
- **Password Requirements**: Enforced complexity to prevent unauthorized access

---

Q: What privacy issues did you address?

A:

- **Encryption at Rest**: All SSNs and PHI encrypted in database using Fernet
- **Access Logging**: Every time someone views a patient, it's logged with who/when/IP
- **Session Timeout**: Auto-logout after 2 minutes prevents unauthorized access
- **Role-Based Access**: Admins and nurses see different data

---

Q: What security issues did you address?

A:

- **SQL Injection**: Used parameterized queries throughout
- **XSS Attacks**: Input validation and sanitization
- **Weak Passwords**: Enforced 8+ chars, upper, lower, number, special character
- **Unattended Workstations**: Automatic session timeout
- **Data Breaches**: Encrypted sensitive fields so even database theft doesn't expose PHI

Q: Did you properly cite sources?

A: Yes - we cited:

- HIPAA regulations (45 CFR sections)
- NIST Cybersecurity Framework
- Python libraries (Flask, Fernet, ReportLab) with proper attribution
- Security best practices from OWASP Top 10

Student Outcome (6): Security Principles (Cybersecurity)

Q: What threats did you identify?

A: We analyzed 5 major threat types:

1. **Ransomware Attack**: Encrypts hospital data, demands payment
2. **Insider Threat**: Employee stealing patient records
3. **Phishing Attack**: Credentials compromised via fake emails
4. **Database Exposure**: Misconfigured server exposes data to internet
5. **Physical Theft**: Unencrypted laptop stolen with patient data

Each is based on real breaches reported to HHS.

Q: What vulnerabilities did you find?

A:

- **Human Error**: Staff not trained on HIPAA rules (solved with training simulator)
- **Weak Passwords**: No complexity requirements (solved with validation)
- **Unencrypted Data**: SSNs stored in plaintext (solved with Fernet encryption)
- **No Accountability**: Actions not logged (solved with audit trail)
- **Session Hijacking**: No timeout on idle sessions (solved with 2-minute timeout)

Q: How did you perform risk analysis?

A: We used a risk matrix:

Threat	Likelihood	Impact	Risk Level	Mitigation
Ransomware	Medium	Critical	HIGH	Breach playbook, backups
Insider Threat	High	High	HIGH	Audit logging, training
Phishing	High	High	HIGH	Training scenarios, MFA ready
Database Exposure	Low	Critical	MEDIUM	Encryption, access controls
Physical Theft	Medium	High	MEDIUM	Encryption, session timeout

---

Q: What mitigation strategies did you implement?

A:

- **Prevention**: Training simulator reduces human error by 40% (based on score improvements)
- **Detection**: EDR panel shows violations in real-time
- **Response**: 5 breach playbooks with 20-25 step procedures
- **Recovery**: Full demo reset function for testing recovery procedures

---

Q: What data did you protect from a privacy standpoint?

A:

- **SSN**: Encrypted using Fernet (AES-128)
- **Patient Names**: Access logged in audit trail
- **Medical Record Numbers (MRN)**: Unique identifiers tracked
- **Contact Information**: Email, phone stored with access controls
- **User Credentials**: Passwords hashed with SHA-256

Q: What security standards did you follow?

A:

- **HIPAA Security Rule** (45 CFR §164.306-318): Administrative, Physical, Technical safeguards
- **NIST Cybersecurity Framework**: Identify, Protect, Detect, Respond, Recover
- **OWASP Top 10**: Web application security (SQL injection, XSS prevention)
- **NIST 800-53**: Security controls for encryption and access management

---

Q: Did you develop a business continuity or disaster recovery plan?

A: Yes - our breach playbooks include:

- **Immediate Response** (0-1 hour): Containment, isolation, notification to security team
- **Short-term Recovery** (1-24 hours): Forensics, evidence preservation, HHS notification prep
- **Long-term Recovery** (1-30 days): System restoration, patient notification (if required), regulatory reporting
- **Prevention** (30-60 days): Implement fixes, update training, security audits

Each playbook follows HIPAA's 60-day breach notification timeline.

Technical Deep-Dive Questions

Q: How does your encryption work?

A: We use Fernet symmetric encryption (AES-128 CBC mode with HMAC authentication). The key is stored separately (in production, would use AWS KMS or HashiCorp Vault). Example: cipher.encrypt() to encrypt SSN data before database storage.

Q: How does the training scoring system work?

A:

- Start at 0%
- Correct answer: +20 points
- Wrong answer: -10 points (and creates a violation for admin to see)
- Maximum score: 100%
- Minimum score: 0% (can't go negative)
- Wrong answers intentionally penalized less to encourage learning

---

Q: How does the audit trail ensure accountability?

A: Every action is logged with:

- **Who**: Username of person performing action
- **What**: Action type (login, view patient, complete training, etc.)
- **When**: Timestamp (ISO format)
- **Where**: IP address of request
- **Details**: Specific information (which patient, what task, etc.)

Logs are immutable (can't be deleted by users) and stored in separate audit table.

Q: What happens in a simulated breach?

A:

1. Admin clicks "Simulate Breach" and selects type (e.g., Ransomware)
2. System creates a high-severity vulnerability in database with `source='breach_simulation'`
3. EDR panel shows red alert banner: "ACTIVE DATA BREACH INCIDENT"
4. Admin can click "View" to see 20-step response playbook
5. Admin follows procedures (in real scenario: isolate systems, call security team, etc.)
6. Admin clicks "Resolve" to mark as handled
7. Entry remains in audit trail for review

Demo-Specific Questions

Q: Why is the session timeout only 2 minutes?

A: For demo purposes! In production, it would be 15-30 minutes. But for presentations, 2 minutes lets us show the security feature without making the audience wait. The timeout is configurable in one line of code.

Q: What does the Full Demo Reset do?

A: The Full Demo Reset wipes everything except user accounts - all patients, assignments, violations, audit logs, and breach simulations. It's used at the end of the day or when you need a complete fresh start for a new demonstration.

Q: Why did you choose a healthcare focus?

A: Healthcare has the most strict data protection requirements (HIPAA), the highest breach costs ($10M+ average), and combines technical + human factors. It's the perfect domain to demonstrate cybersecurity, compliance, and training integration.

Conclusion Talking Points

What makes this project unique?

1. **Integrated Solution**: Most tools do ONE thing (training OR breach detection). We combine compliance monitoring, training, threat detection, and incident response.
2. **Educational + Realistic**: Built for learning but uses actual HIPAA requirements and real breach scenarios.
3. **Human-Centered Security**: Recognizes that 95% of breaches involve human error, so we focus on training and behavioral tracking.
4. **Demonstrable**: Quick Setup button generates full demo data in 3 seconds. Perfect for repeated presentations.

What did you learn?

- How to translate legal requirements (HIPAA) into technical specifications
- Real-world encryption implementation (not just theory)
- How audit trails provide accountability in healthcare
- Why security training is as important as technical controls
- The complexity of incident response procedures

What would you improve for production?

- Multi-factor authentication (TOTP or FIDO2)
- PostgreSQL instead of SQLite for scalability
- HTTPS with valid SSL certificates
- Encryption key management (AWS KMS)
- Real-time SIEM integration (Splunk, ELK stack)
- Mobile app for on-the-go compliance checks
- Integration with actual EHR systems (Epic, Cerner)

Why does this matter?

Healthcare breaches affect millions of patients every year. The average cost is $10M per incident. Organizations need tools that:

- Train staff BEFORE mistakes happen
- Detect violations in real-time
- Provide clear response procedures
- Demonstrate compliance to auditors

SecureMed addresses all four needs in one platform.

Last Updated: December 2025 For Capstone Project Presentations & Defense