Healthcare Web Application Development

Capstone Project Guidelines

CSE 616: Lab on Web Engineering

Rokan Uddin Faruqui
Professor
Department of Computer Science and Engineering
University of Chittagong
Chittagong-4331, Bangladesh

September 11, 2025

1 Project Overview

This capstone project requires teams to develop a comprehensive healthcare web application following modern web engineering practices and agile software development methodologies. The project spans 4 months with 5 major deliverables scheduled at 2-week intervals.

1.1 Project Objectives

- Design and implement a full-stack healthcare web application
- Apply agile software engineering principles throughout development
- Demonstrate proficiency in modern web technologies
- Implement proper deployment and optimization strategies
- Document the entire development process

2 Technology Stack Requirements

2.1 Frontend Options

Choose ONE of the following frontend frameworks:

- **React.js** - Component-based library with hooks and state management
- **Angular** - Full-featured framework with TypeScript
- **Vue.js** - Progressive framework with reactive data binding
- **Next.js** - React-based framework with SSR/SSG capabilities

2.2 Backend Options

Choose ONE of the following backend technologies:

- **Java Spring Boot** - Enterprise-grade framework with dependency injection
- **Node.js** - JavaScript runtime with Express.js or similar frameworks

- **Python** - Flask/Django for rapid development and data processing
- **ASP.NET Core** - Microsoft's cross-platform web framework

2.3 Database Requirements

- Choose between SQL (PostgreSQL, MySQL) or NoSQL (MongoDB) databases
- Database must be managed through ORM (e.g., Hibernate, Sequelize, SQLAlchemy, Entity Framework)
- Justify your choice based on application requirements
- Implement proper data modeling and relationships

3 Healthcare Application Requirements

3.1 Core Features

Your healthcare application must include:

1. **User Authentication & Authorization**
   - Multi-role system (Patient, Doctor, Admin)
   - Secure login/logout functionality
   - Password reset and account management
2. **Patient Management**
   - Patient registration and profile management
   - Medical history tracking
   - Appointment scheduling system
3. **Doctor Dashboard**
   - Appointment management
   - Patient records access
   - Prescription management
4. **Administrative Features**
   - User management
   - System analytics and reporting
   - Hospital/clinic management
5. **Additional Features** (Choose 2-3)
   - Telemedicine video consultations
   - Medical imaging upload and viewing
   - Laboratory test results integration
   - Medication tracking and reminders
   - Health monitoring dashboard

4 Agile Framework Requirements

4.1 Project Management Tools

- **Trello**: Create boards for sprint planning, task tracking, and progress monitoring
- **Slack**: Set up team workspace for communication and integration with development tools
- **Git**: Version control with branching strategy (feature branches, pull requests)

4.2 Agile Practices

- Conduct weekly sprint planning meetings
- Maintain a product backlog with user stories
- Implement daily standups (documented in Slack)
- Perform sprint retrospectives after each deliverable
- Use story points for effort estimation

5 Deliverables Timeline

Week	Deliverable	Due Date	Key Components
2	D1: Requirements	September 17, 2025	Requirements document, tech stack selection, project setup
4	D2: Design	October 5, 2025	System architecture, UI/UX mock-ups, database schema
6	D3: Backend MVP	October 19, 2025	API endpoints, authentication, database integration
8	D4: Frontend Integration	November 2, 2025	Complete UI implementation, frontend-backend integration
10-12	D5: Deployment & Final	November 16 and 30, 2025	Production deployment, optimization, final documentation

Table 1: Deliverables Timeline

5.1 Deliverable 1: Requirements Document & Tech Stack (Week 2)

Components:

- Functional and non-functional requirements
- User stories with acceptance criteria
- Technology stack justification
- Team roles and responsibilities
- Project timeline and milestones
- Risk assessment and mitigation strategies

Deliverables:

- Requirements document (10-15 pages)
- Trello board setup with initial backlog
- Slack workspace configuration
- Git repository initialization

5.2 Deliverable 2: Design Document (Week 4)

Components:

- System architecture diagram
- Database design and ER diagrams
- API specification and endpoints
- UI/UX mockups
- Security architecture

Deliverables:

- Design document (15-20 pages)
- Interactive prototypes or mockups
- Updated project timeline

5.3 Deliverable 3: Backend MVP (Week 6)

Components:

- RESTful API implementation
- Authentication and authorization system
- Database models and migrations
- Unit and integration tests
- API documentation
- Basic error handling and logging

Deliverables:

- Functional backend with API endpoints
- Comprehensive API documentation
- Test suite with coverage report

5.4 Deliverable 4: Frontend Integration (Week 8)

Components:

- Complete frontend implementation
- Integration with backend APIs
- Responsive design implementation
- User authentication flow
- Frontend testing (unit and integration)
- Performance optimization

Deliverables:

- Fully functional web application
- Frontend test suite
- Performance analysis report

5.5 Deliverable 5: Deployment & Final Report (Weeks 10-12)

Components:

- Production deployment configuration
- Docker containerization
- Caching implementation (Redis/Memcached)
- Performance monitoring and analytics
- Security hardening
- Comprehensive documentation

Deliverables:

- Live deployed application
- Docker compose configuration
- Final project report (25-30 pages)
- Video demonstration (10-15 minutes)
- Project presentation slides

6 Technical Requirements

6.1 Security Requirements

- HTTPS encryption for all communications
- Proper input validation and sanitization
- JWT tokens for authentication
- Role-based access control (RBAC)
- Password hashing with salt
- Patient information must be encrypted at rest and in transit
- Protection against common vulnerabilities (atleast 2 known vulnerabilities)

6.2 Performance Requirements

- Implement caching strategies
- (Optional) Page load time under 3 seconds
- (Optional) API response time under 500ms
- (Optional) Database query optimization
- (Optional) Image optimization and lazy loading

6.3 Deployment Requirements

- Docker containerization
- Cloud deployment using Student Account/Free Cloude Services (AWS, Azure, GCP, or Heroku)
- Environment configuration management
- Database backup and recovery plan
- Monitoring and logging setup

7 Team Structure & Collaboration

7.1 Team Composition

Teams should consist of 3-4 members with defined roles:

- **Scrum Master**: Project coordination and agile process management
- **Frontend Developer(s)**: UI/UX implementation
- **Backend Developer(s)**: API and server-side logic
- **DevOps Engineer**: Deployment and infrastructure
- **QA Engineer**: Testing and quality assurance

7.2 Communication Guidelines

- Weekly team meetings (mandatory)
- Daily check-ins via Slack
- Code reviews for all pull requests
- Documentation updates with each commit
- Progress tracking in Trello boards

8 Evaluation Criteria

8.1 Grading Distribution

- Deliverable 1 (Requirements): 15%
- Deliverable 2 (Design): 15%
- Deliverable 3 (Backend MVP): 20%
- Deliverable 4 (Frontend Integration): 20%
- Deliverable 5 (Deployment & Final): 25%
- Team Collaboration & Process: 5%

8.2 Assessment Criteria

1. **Technical Implementation (40%)**
   - Code quality and architecture
   - Proper use of chosen technologies
   - Security implementation
   - Performance optimization
2. **Functionality (30%)**
   - Feature completeness
   - User experience quality
   - Error handling
   - Cross-browser compatibility
3. **Documentation (20%)**
   - Code documentation
   - Technical documentation
   - User documentation
   - API documentation
4. **Process & Collaboration (10%)**
   - Agile methodology adherence
   - Team collaboration effectiveness
   - Version control usage
   - Meeting participation

9 Submission Guidelines

9.1 Submission Format

- All code submitted via Git repository
- Documentation in PDF format
- Live application URL
- Video demonstration uploaded to course platform

Note: This project represents a significant portion of your course grade. Start early, collaborate effectively, and don't hesitate to seek help when needed. Good luck!