Smart Campus Navigation and Facility Booking System (SCNFBS)

Systems Analysis and Design - SDLC Implementation

Course: COSC333 Systems Analysis and Design

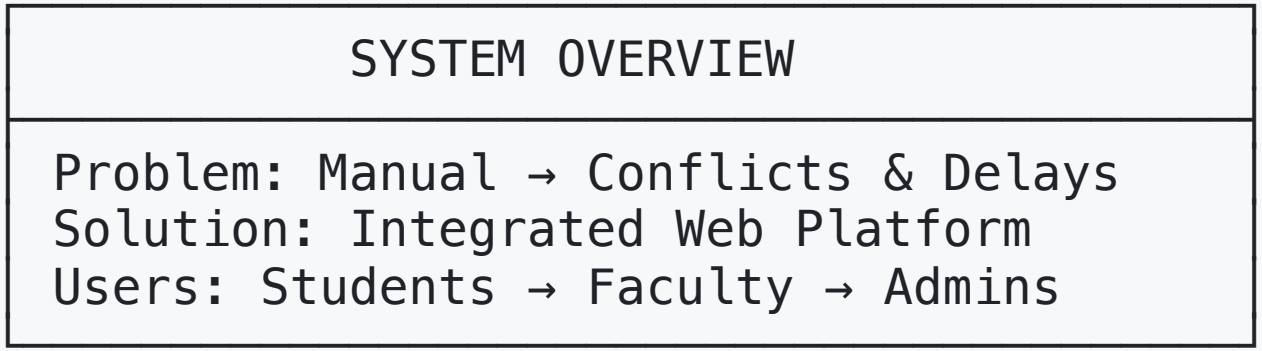
Professor: Dr. Kamal Taha

Chapters 1-5 SDLC Application

Slide 1: Project Overview

SDLC Phase 1: System Identification and Planning

- **Problem**: Manual booking causes 30% conflicts, students waste 15+ min navigating
- **Solution**: Integrated platform for navigation and facility booking with real-time conflict prevention
- **Scope**: Node.js/Express serving 1000+ users with navigation, booking, user management



Slide 2: Requirements Analysis and Engineering

SDLC Phase 2: Requirements Gathering and Analysis

- **Functional**: Authentication, mapping, booking, admin controls (FR001-020)
- **Non-Functional**: <2s response, 100+ users, HTTPS/JWT security, WCAG compliance
- **Validation**: Traceability matrix, stakeholder review, prototype testing

```
Requirements Hierarchy:
├─ Functional Requirements (FR001–020)
│  ├─ User Management
│  ├─ Navigation System
│  └─ Booking System
└─ Non-Functional Requirements
   ├─ Performance (<2s)
   ├─ Security (HTTPS/JWT)
   └─ Usability (WCAG 2.1)
```

Slide 3: Stakeholder Analysis and Use Case Modeling

SDLC Phase 2 Continued: Stakeholder Identification

- **Stakeholders**: Students (navigation/booking) → Faculty (advanced booking) → Admins (oversight)
- **Use Cases**: Navigate Campus, Book Facility, Manage Users with actor hierarchy
- **Validation**: Stakeholder interviews, scenario walkthroughs, requirements traceability

Slide 4: System Design and UML Modeling

SDLC Phase 3: System Design and Architecture

- **UML Models**: Use Case (4 actors), Class Diagram (User/Building/Room/Booking), Sequence flows
- **Patterns**: Repository, MVC, Factory with association/inheritance relationships
- **Architecture**: Layered (Presentation → Business → Data) with high cohesion, loose coupling

Slide 5: Database Design and Data Architecture

SDLC Phase 3 Continued: Data Design and Modeling

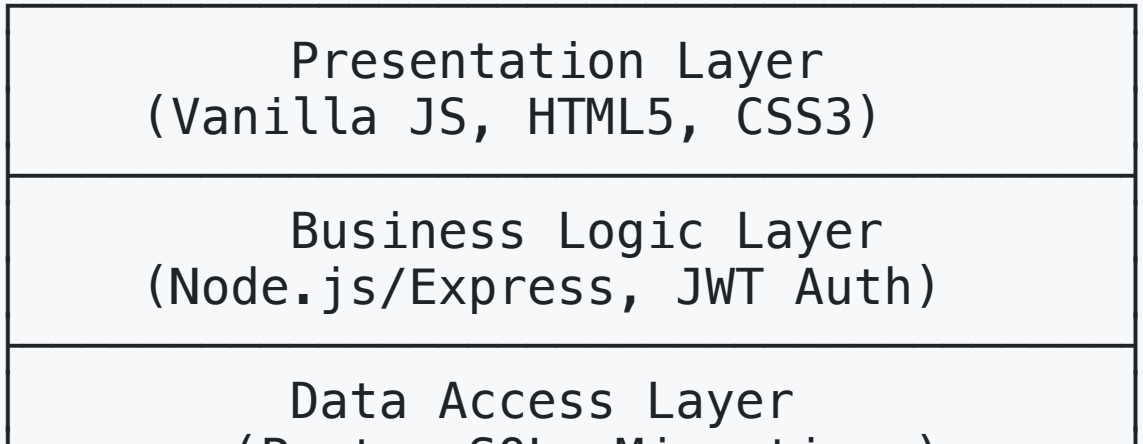
- **Data Model**: 8 entities in 3NF with 1:N/M:N relationships, business rule constraints
- **Physical Design**: PostgreSQL with 15+ indexes, JSONB config, UUIDs, ACID compliance
- **Implementation**: Automated migrations, role-based access, point-in-time recovery

Slide 6: System Implementation and Construction

SDLC Phase 4: Implementation and Development

- **Tech Stack**: Node.js/Express backend, PostgreSQL DB, Vanilla JS frontend, JWT security
- **Strategy**: Modular architecture, feature-driven development, Git workflow
- **Core Modules**: Authentication (login/sessions), Navigation (mapping), Booking (real-time)

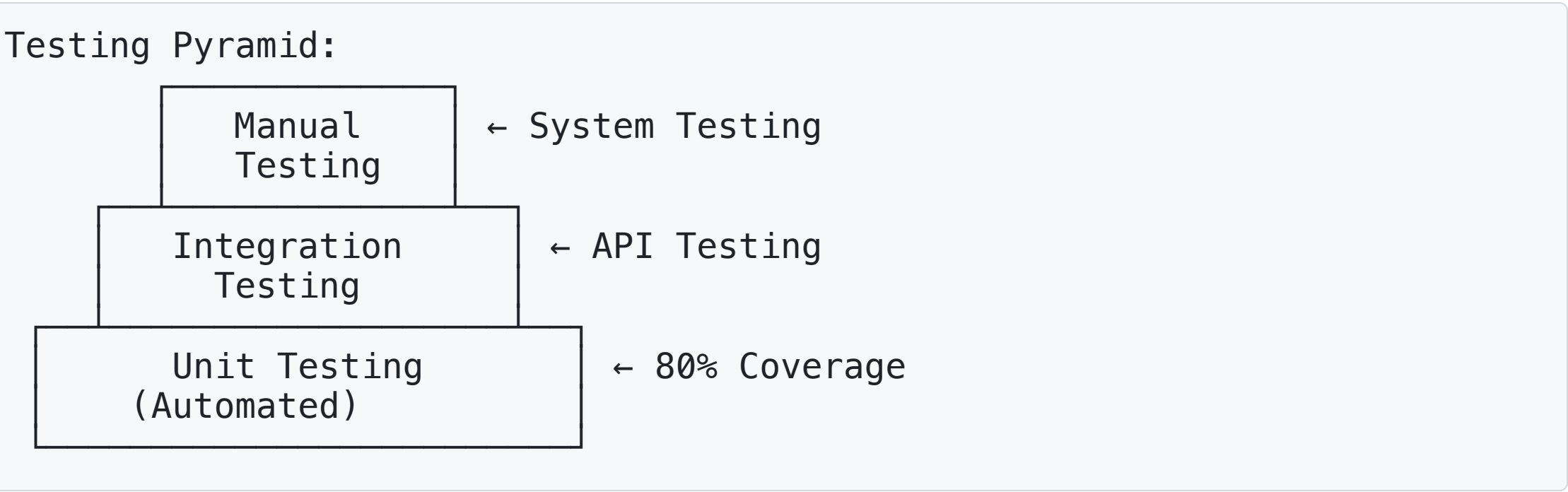
```
System Architecture:
```



Slide 7: Testing and Quality Assurance Strategy

SDLC Phase 5: Testing and Validation

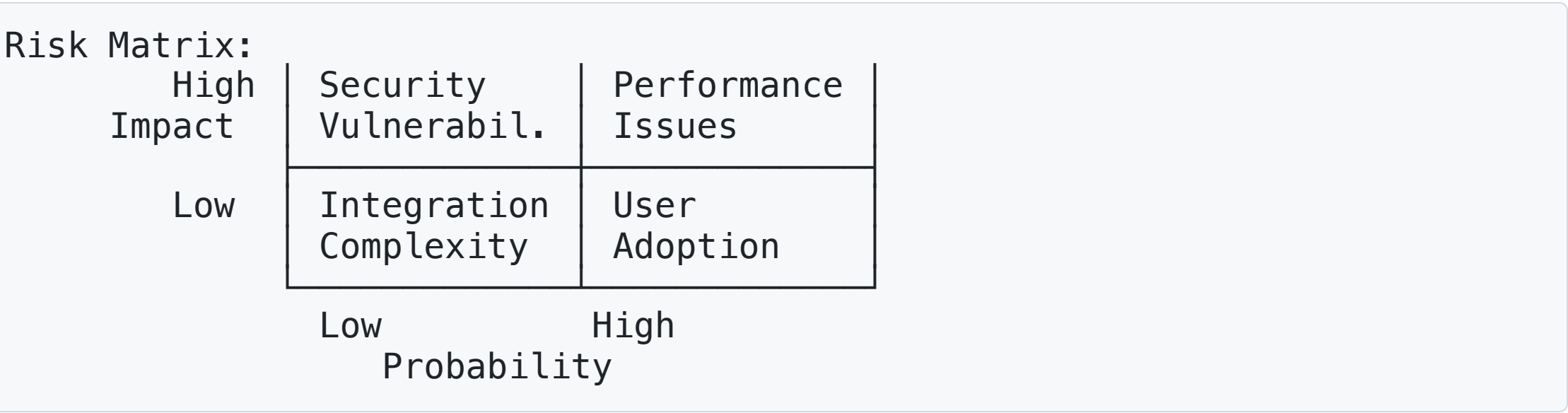
- **Testing Levels**: Unit (80% coverage) → Integration (API) → System (E2E workflows)
- **QA Focus**: Performance (100+ users), Security (penetration), Usability (accessibility)
- **Management**: Requirements → Test Cases → Defects traceability, CI/CD automation



Slide 8: Feasibility Analysis and Risk Management

SDLC Phase 1 Continued: Feasibility Assessment

- **Technical**: Proven Node.js/PostgreSQL stack, team JS expertise, standard hosting
- **Economic**: 6-month timeline, 40+ hrs/week savings justify costs, minimal operations
- **Risk Mitigation**: Performance testing, iterative development, phased deployment



Slide 9: Project Conclusions and Future Evolution

SDLC Implementation Success

- **SDLC Applied**: Requirements engineering, UML modeling, normalized DB design
- **Achievements**: Layered architecture, full-stack security focus, multi-level testing
- **Future Roadmap**: Phase 2 (Mobile/notifications) → Phase 3 (IoT/AI) → AR navigation

```
Evolution Timeline:
```

Phase 1	Phase 2	Phase 3	Long-term
Web App Core Features	Mobile Real-time Notify	IoT AI Rec Multi-Campus	AR Nav Predict Analytics
6mo	12mo	18mo	24mo+

This project successfully demonstrates the application of Systems Analysis and Design principles through a complete SDLC implementation, from initial system identification through testing and deployment, providing a foundation for continued system evolution and enhancement.