Slide 1: Introduction

Purpose of HMS v1.0

Streamlining hostel operations

Modules: Incident, Room Allocation, Feedback, Staff Management, Mess, Fee Submission, Authentication

Slide 2: Document Conventions

Overview of industry-standard conventions

Naming, formatting, and document structure

Version control using semantic versioning

Slide 3: Project Scope

Inclusions: Modules overview

Exclusions: No financial system integration, limited analytics

Slide 4: References

Relevant documents (IEEE Std 830-1998, University Policies)

External references (DBMS, Django, Web Browsers)

Slide 5: Product Perspective

Standalone software for hostel management

Components: UI, Database, Functional Modules

Subsystem Interconnections and External Interfaces

Slide 6: User Classes and Characteristics

Administrator, Staff, Residents, System Guests, Support/IT Personnel

Characteristics and functions of each user class

Slide 7: Operating Environment

Tailored for the university environment

Server requirements: Windows, Linux

Client devices: Modern web browsers

Slide 8: Design and Implementation Constraints

Regulatory Compliance

Hardware Limitations

Database Technology

Integration with Existing Systems

Design Conventions and Programming Standards

Slide 9: Assumptions and Dependencies

Assumptions about third-party components, regulatory compliance, hardware stability, user availability, and development resources

Dependencies on external system integration, web browser compatibility, university policies, and external software updates

Slide 10: System Features

Add Incident

Authentication, Recording, Tracking

Slide 11: System Features

Allocate Room

Availability Check, Allocation, Transfer

Slide 12: System Features

Add Feedback, Manage Staff, Maintain Mess

Feedback Submission, Staff Management, Mess Operations

Slide 13: System Features

Submit Fee, Register, Login/Logout

Fee Submission, User Registration, Secure Authentication

Slide 14: Logical Data Model

Overview: Representation of data entities for effective hostel management

Entities: Users (Warden, Staff, Students), Modules, Common Entities (Room, Maintenance, Visitor, Event, Notification, Security Logs)

Attributes and Relationships: Details of attributes and relationships for user entities, rooms, incidents, and feedback

Slide 15: Data Dictionary

Overview: Comprehensive reference for data elements

Key Data Elements: User Table, Room Table, Incident Table, Feedback Table, Maintenance Table, Visitor Table

Relationships: User-to-Incident, Room-to-Maintenance, User-to-Feedback

Slide 16: Reports

Overview: Reporting system for valuable insights

Key Reports: Occupancy, Incident Summary, Fee Submission, Feedback Analysis, Maintenance History

Slide 17: Data Acquisition, Integrity, Retention, and Disposal

Data Acquisition: User Registration, Incident Recording, Room Allocation

Data Integrity: Referential Integrity, Validation Rules

Data Retention: Archiving Historical Data, Retention Policies

Data Disposal: Secure Deletion, Compliance with Regulations

Slide 18: External Interface Requirements

User Interfaces

Overview: Seamless and intuitive interfaces for administrators, staff, residents, system guests, support/IT personnel

User Classes and Interfaces: Specific features for each user class

Slide 19:

Software Interfaces

Database Management System (DBMS): Interaction for data storage

Web Browsers: Compatibility with major browsers

Third-Party Integrations: Integration with financial systems

Slide 20:

Hardware Interfaces

Server Hardware: Robust server for data management

Client Devices: Access through various devices

Slide 21: External Interface Requirements

Communications Interfaces

Internet Communication: Communication over the internet

Email Notifications: Important updates via email

API Integration: Potential integration with university systems

Security Protocols: Secure communication protocols

Slide 22: Quality Attributes

Availability: Aim for 99.9% availability

Correctness: Consistent and correct results

Flexibility: Customizable settings

Portability: Compatibility with major browsers and operating systems

Reliability: Minimize system failures and errors

Reusability: Modular and well-documented code

Robustness: Effective error handling

Testability: Modular structure for efficient testing