** Gopalganj Science and Technology University**

**Project Requirement Analysis on**

**“Hotel Management System”**

**Course Code: CSE 310**

**SUBMITTED GROUP BY**

              M. A. Forhad Aziz, ID: 20CSE007

Mohammad Nimour Hossain, ID: 20CSE010

  Jahid Hasan,   ID: 20CSE037

Session: 2020-21

Department of Computer Science and Engineering

**SUBMITTED TO**

Dr. Syful Islam,

Assistant Professor.

Department of Computer Science and Engineering

[1. INTRODUCTION 4](#_Toc198226204)

[1.1 Purpose of the Hostel Management System 4](#_Toc198226205)

[1.2 Scope of the System 4](#_Toc198226206)

[1.3 Goals of the Development Team 4](#_Toc198226207)

[1.4 Development Process Model 5](#_Toc198226208)

[1.5 Team Roles and Organization 5](#_Toc198226209)

[2. RESEARCH 6](#_Toc198226210)

[2.1 Literature Survey 6](#_Toc198226211)

[2.1.1 User Interface Design 6](#_Toc198226212)

[2.1.2 Data Management 6](#_Toc198226213)

[2.1.3 Authentication 6](#_Toc198226214)

[2.1.4 Performance 7](#_Toc198226215)

[2.2 Technology Stack 7](#_Toc198226216)

[3. DESCRIPTION 8](#_Toc198226217)

[3.1 System Modules 8](#_Toc198226218)

[3.1.1 Student Management 8](#_Toc198226219)

[3.1.2 Meal Management 8](#_Toc198226220)

[3.1.3 Request System 8](#_Toc198226221)

[3.1.4 Payment Module 8](#_Toc198226222)

[3.2 User Roles 9](#_Toc198226223)

[3.3 System Workflow 9](#_Toc198226224)

[3.4 Security 9](#_Toc198226225)

[4. REQUIREMENTS 10](#_Toc198226226)

[4.1 Functional Requirements 10](#_Toc198226227)

[4.2 Database Structure 10](#_Toc198226228)

[4.3 Performance 10](#_Toc198226229)

[5. SYSTEM MODELING 11](#_Toc198226230)

[5.1 Data Flow Diagrams 11](#_Toc198226231)

[5.2 Use Cases 11](#_Toc198226232)

[6. GANTT CHART 12](#_Toc198226233)

[7. Tech Stack 13](#_Toc198226234)

[8. Final Deliverables 13](#_Toc198226235)

[9. Conclusion 13](#_Toc198226236)

**Hostel Management System – Requirement Analysis Document**

# 1. INTRODUCTION

## 1.1 Purpose of the Hostel Management System

The Hostel Management System (HMS) is designed to automate and streamline hostel operations in educational institutions. Key objectives include:

Digitalizing room allocation (check-in/check-out, maintenance requests)

Managing meal services (menu planning, student reviews, dietary preferences)

Simplifying financial transactions (hostel fees, meal subscriptions)

Enhancing communication between students, staff, and administrators

## 1.2 Scope of the System

In-Scope

Student Portal: Room booking, meal reviews, payment gateway

Admin Dashboard: Room/meal management, approval workflows, reporting

Mobile Responsiveness: Accessible on all devices

Third-party Integrations: Stripe (payments), Firebase (auth), SendGrid (notifications)

**Out-of-Scope**

- Campus-wide ERP integration

- IoT-based room automation

- Offline functionality

## 1.3 Goals of the Development Team

- Deliver a scalable MERN stack applicationwithin 3 months

- Achieve 90% test coverage with Jest/React Testing Library

- Implement CI/CD pipeline using GitHub Actions

- Ensure GDPR compliancefor data protection

## 1.4 Development Process Model

Agile Scrum Framework with:

- 2-week sprints

- Daily standups

- Sprint reviews with stakeholders

- Tools: Jira (task tracking), Figma (UI prototyping)

## 1.5 Team Roles and Organization

Role Responsibilities Tools

Frontend Lead React components, Redux state Figma, Storybook

Backend Lead Node.js APIs, DB design Postman, MongoDB Atlas

QA Engineer Test automation Jest, Cypress

DevOps Deployment pipeline Docker, AWS

# 2. RESEARCH

## 2.1 Literature Survey

### 2.1.1 User Interface Design

-Material-UI for consistent components

Dark/Light modetoggle for accessibility

- Dashboard Analytics: Charts.js for occupancy/meal stats

### 2.1.2 Data Management

-MongoDB Schema:

`javascript

// Room Schema

{

roomNo: String,

type: { enum: ["Single", "Double", "Dorm"] },

status: { enum: ["Vacant", "Occupied", "Maintenance"] },

studentAllocations: [{ studentId: ObjectId, date: Date }]

}

### 2.1.3 Authentication

Firebase Auth with:

- Email/password

- Google OAuth

- Phone verification (optional)

### 2.1.4 Performance

Caching: Redis for frequent queries

-Lazy Loading: React code-splitting

CDN: Cloudflare for static assets

## 2.2 Technology Stack

| Layer | Technology | Justification |

|-------|------------|---------------|

| Frontend | React + Redux | Component reusability |

| Backend | Node.js + Express | Non-blocking I/O |

| Database | MongoDB | Flexible schema for hostel data |

| Auth | Firebase/JWT | Rapid implementation |

| Payments | Stripe | PCI-DSS compliance |

# 3. DESCRIPTION

## 3.1 System Modules

### 3.1.1 Student Management

- Features:

- Profile creation with ID proof upload

- Room preference selection

- Emergency contact registry

### 3.1.2 Meal Management

- \*\*Workflow\*\*:

1. Admin uploads weekly menu

2. Students rate meals (1-5 stars)

3. System auto-flags dishes with <2.5 avg rating

### 3.1.3 Request System

-Request Types:

```mermaid

graph TD

A[Student] -->|Submit| B(Room Change)

A -->|Submit| C(Maintenance)

B --> D[Admin Approval]

C --> E[Staff Assignment]

```

### 3.1.4 Payment Module

- Flow:

1. Student selects payment plan (Monthly/Quarterly)

2. Stripe checkout embedded

3. Receipt auto-generated in PDF

## 3.2 User Roles

Admin

- \*\*Permissions\*\*:

- `hostel:create` (Add rooms)

- `meal:delete` (Remove menu items)

- `finance:export` (Generate reports)

Student

- \*\*Permissions\*\*:

- `booking:create` (Max 1 active)

- `review:write` (3 reviews/day limit)

## 3.3 System Workflow

```plaintext

Student Login → Book Room → Pay Fee → Get Allocation

↓

Rate Meals → Notify Kitchen Staff

## 3.4 Security

Data Encryption: AES-256 for PII

- Rate Limiting: 5 requests/sec per IP

Audit Logs: MongoDB Change Streams

# 4. REQUIREMENTS

## 4.1 Functional Requirements

Student Must-Haves

- [ ] FR-01: Book room for academic year

- [ ] FR-02: View meal calendar with allergens

- [ ] FR-03: Pay fees in 3 installments

Admin Must-Haves

- [ ] FR-04: Bulk import students via CSV

- [ ] FR-05: Force-checkout during vacations

## 4.2 Database Structure

Relationships:

- One-to-Many: Student → Reviews

- Many-to-Many: Rooms → Students (through allocations)

## 4.3 Performance

Load Test: 1000 concurrent users @ <2s response

Stress Test: 10,000 room records searchable in <5s

# 5. SYSTEM MODELING

## 5.1 Data Flow Diagrams

**Level 0 DFD**

[Student] → (Booking Request) → [System] → [Database]

[Admin] → (Approval) → [System] → [Email Service]

**Level 1 DFD**

Auth

Booking

Payment

Notification

## 5.2 Use Cases

UC-01: Meal Review

1. Student logs in

2. Navigates to "Today's Meals"

3. Rates "Pasta" 4 stars

4. System updates average rating

**UC-02: Room Reallocation**

1. Admin views "Vacant Rooms"

2. Drags student "John" to Room 205

3. System emails John new allocation letter