Requirements and Design Document

for

NITC Hostel Mess Management System

Version 1.0

Prepared by Team 19:

Nishant Tanaji Bhandigare	**M250923CS**	**nishant_m250923cs@nitc.ac.in**
Sikta Roy	**M251045CS**	**sikta_m251045cs@nitc.ac.in**
Vengidesan K	**M251103CS**	**vengidesan_m251103cs@nitc.ac.in**

Project Manager: **Maneesha**

Project Client: **NIT Calicut**

Course: **CS6103E - Software Systems lab**

Date: **28.09.2025**

Glossary

NITC	National Institute of Technology Calicut
MERN	MongoDB, Express, React, Node.js (technology stack)
API	Application Programming Interface
JWT	JSON Web Token; standard for user authentication
SPA	Single Page Application
UI	User Interface
MongoDB	NoSQL Document Database
REST	Representational State Transfer; architectural style for networked applications

Table of contents

Glossary 2
Table of contents 3
MERN Application Design Document: NITC Hostel Mess Management System 4
1. Introduction 4
2. Functional Requirements 5
3. UI Design 6
4. System Architecture 9
4.1. High-Level Diagram 9
4.2. Technology Stack 10
4.3. Database Schema (MongoDB Collections) 10

MERN Application Design Document: NITC Hostel Mess Management System

1. Introduction

Goal: To digitize and streamline NITC hostel mess operations by providing an online portal for menu posting, feedback, attendance, inventory, and automated billing.

- **Project Vision:** Empower students and mess staff with a transparent, efficient digital system for managing hostel mess activities.
- **Problem Statement:** Manual processes for mess management are inefficient, error-prone, and lack transparency. There is a need for a centralized system to manage attendance, feedback, menu, inventory, and billing.
- **Target Audience:** Students residing in hostels, mess managers, and hostel administrators.

2. Functional Requirements

Key Features & Scope:

- **User Authentication and Management:**
  - F1: The system shall provide secure user registration and login functionality with role-based access control for students, mess managers, and administrators.
  - F2: The system shall maintain user profiles with personal information, hostel details, and role-specific permissions.

- **Menu Management:**
  - F3: The system shall allow mess managers to create, edit, and publish weekly meal menus with detailed descriptions and nutritional information.
  - F4: The system shall enable students to view current and upcoming weekly menus in an organized, user-friendly format.

- **Feedback and Rating System:**
  - F5: The system shall allow students to submit meal ratings on a 1-5 scale for each meal they consume.
  - F6: The system shall enable students to provide written feedback and suggestions for meal quality improvement.
  - F7: The system shall provide mess managers with feedback analytics including average ratings.

- **Attendance Management:**
  - F8: The system shall allow students to register non-attendance for specific dates during vacations or leave periods.
  - F9: The system shall maintain accurate attendance records for all registered students with date-wise tracking.

- **Billing and Payment Management:**
  - F10: The system shall automatically generate monthly mess bills based on student attendance records and applicable rates.
  - F11: The system shall allow administrators to view, modify, and finalize monthly billing for all students.
  - F12: The system shall track payment status and provide payment history for each student account.

- **Administrative Functions:**
  - F13: The system shall provide administrators with comprehensive reporting capabilities including attendance summaries, payment reports, and system usage analytics.
  - F14: The system shall enable inventory management functionality for mess managers to track food stock levels and consumption patterns.

3. UI Design

The NITC Hostel Mess Management System will provide three distinct user interfaces tailored to the specific roles and requirements of each user group. All interfaces will be responsive web-based applications built using React.js with Material-UI components to ensure consistency and modern design standards.

GUI of some of the pages made using Figma is given below:

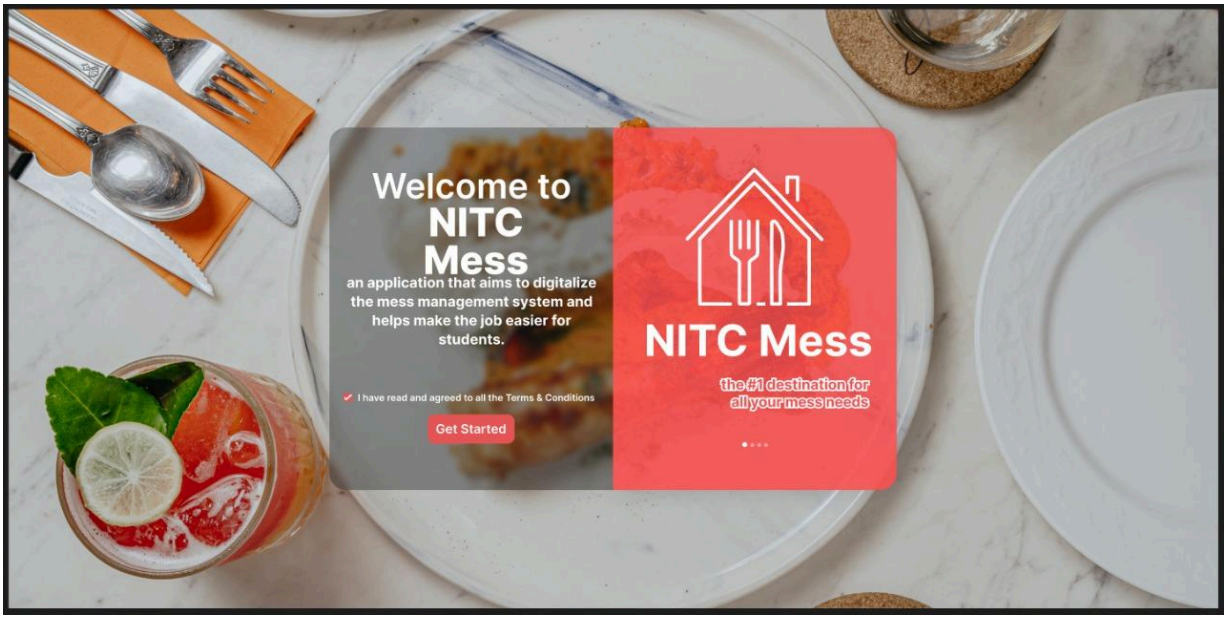


Figure 1: The starting page

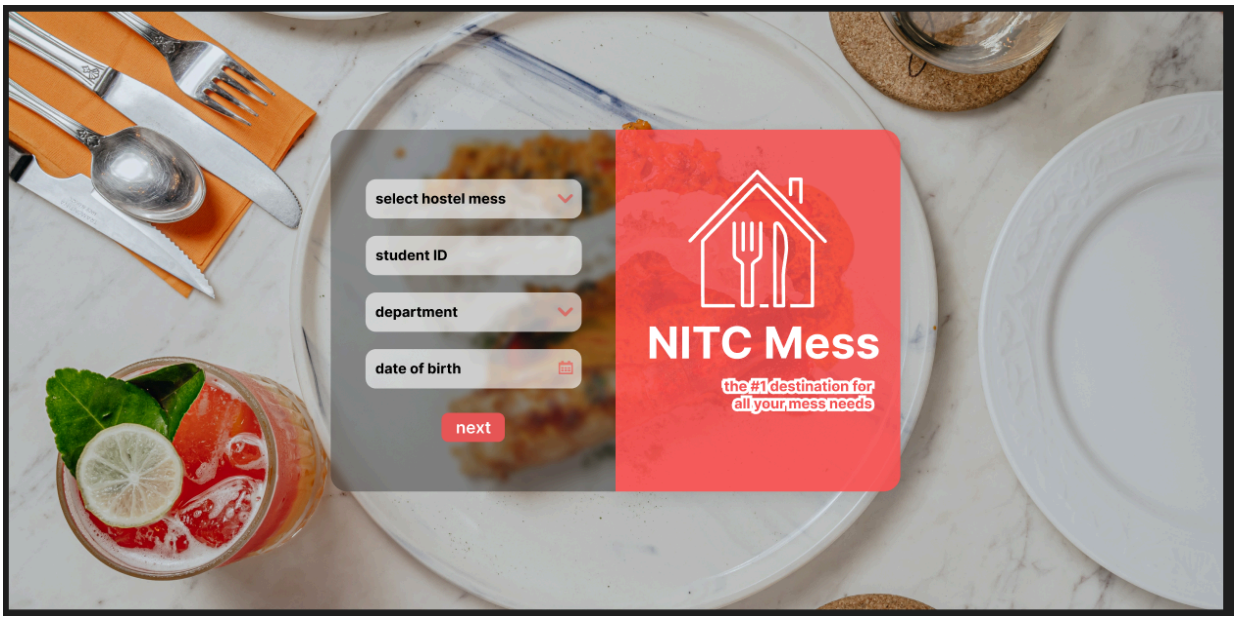


Figure 2: The register page

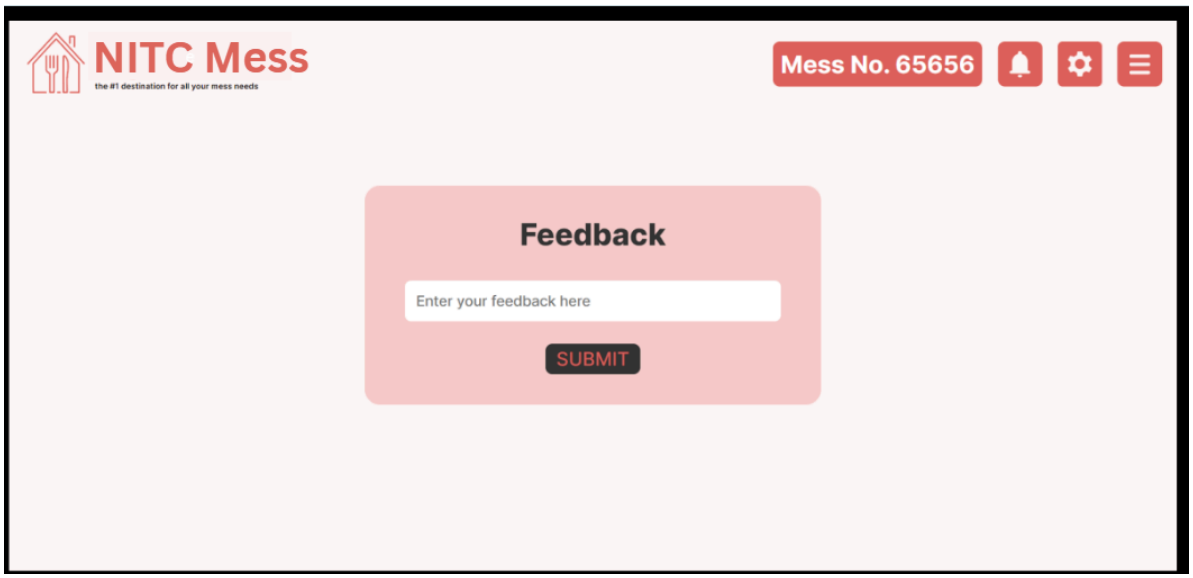


Figure 3: The Feedback page

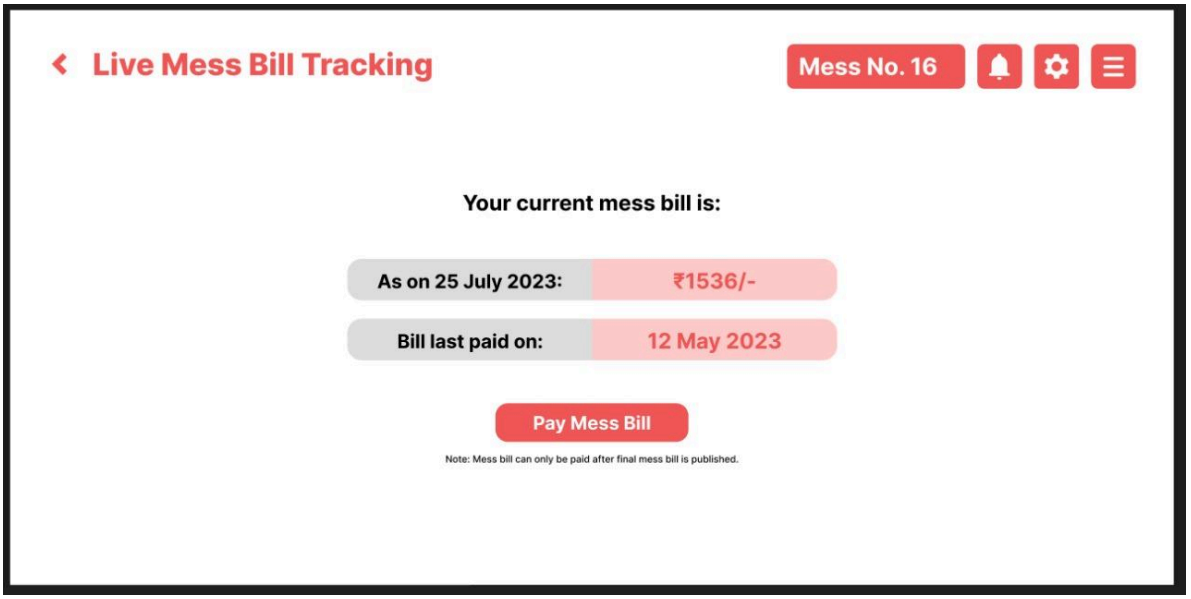
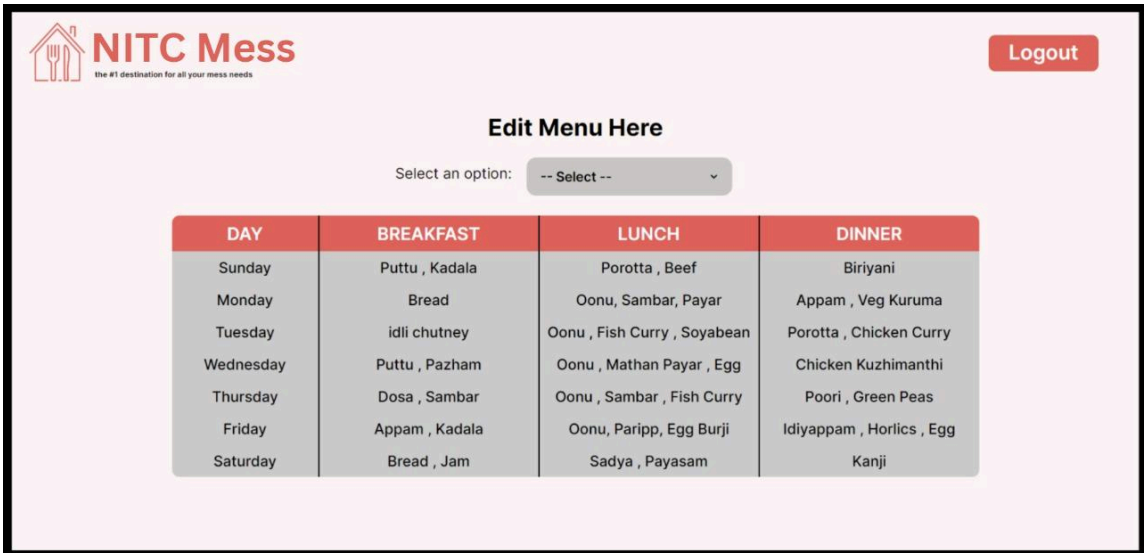


Figure 4 : Bill Tracking Page



DAY	BREAKFAST	LUNCH	DINNER
Sunday	Puttu , Kadala	Porotta , Beef	Biriyani
Monday	Bread	Oonu, Sambar, Payar	Appam , Veg Kuruma
Tuesday	idli chutney	Oonu , Fish Curry , Soyabean	Porotta , Chicken Curry
Wednesday	Puttu , Pazham	Oonu , Mathan Payar , Egg	Chicken Kuzhimanthi
Thursday	Dosa , Sambar	Oonu , Sambar , Fish Curry	Poori , Green Peas
Friday	Appam , Kadala	Oonu, Paripp, Egg Burji	Idiyappam , Horlics , Egg
Saturday	Bread , Jam	Sadya , Payasam	Kanji

Figure 5 : Menu Editing Page

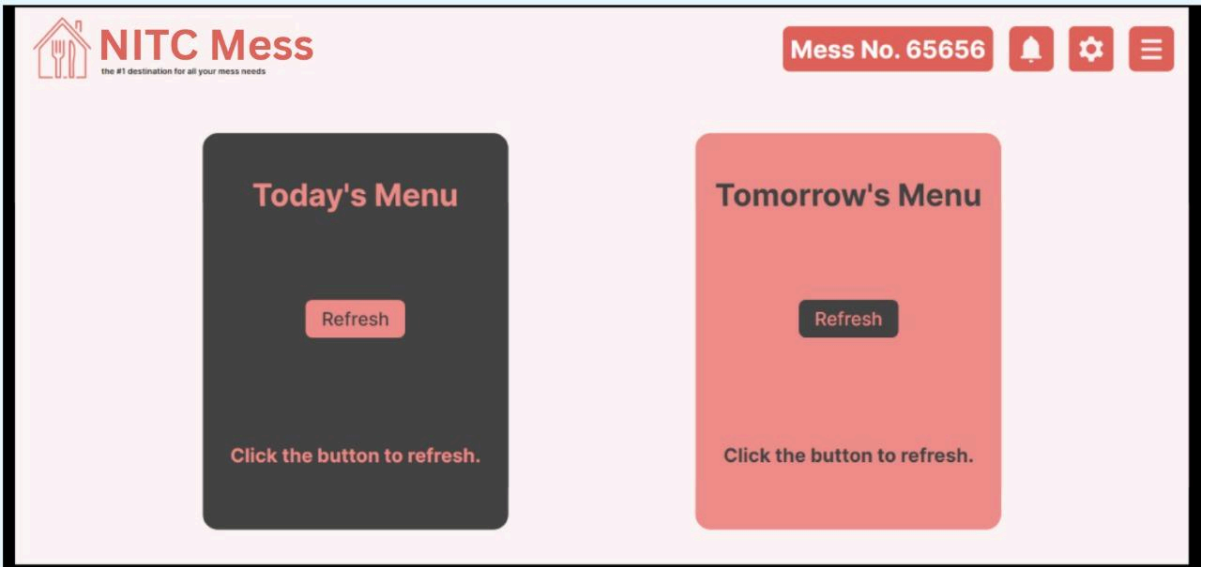


Figure 6 : Viewing Menu Page

Student Interface:

The student dashboard will feature a clean, intuitive layout with four primary sections: Menu View (displaying weekly meals in card format with images and nutritional information), Feedback Center (rating system with 1-5 stars and text comment fields), Attendance Manager (calendar interface for marking vacation/absence dates), and Bill Summary (displaying current month charges and payment history). Navigation will be through a bottom tab bar on mobile devices and side navigation on desktop, optimized for quick access to frequently used features.

Mess Manager Interface:

The mess manager portal will include a comprehensive dashboard with Menu Editor (drag-and-drop interface for weekly meal planning with ingredient lists), Feedback Analytics (charts and graphs showing meal ratings and common feedback themes using Chart.js), Inventory Management (table-based CRUD interface for stock tracking with low-stock alerts), and Attendance Reports (filtered views of student attendance patterns). The interface will support bulk operations and provide export functionality for reports.

Administrator Interface:

The admin panel will offer system-wide oversight capabilities including User Management (role-based user administration with search and filter options), Billing System (automated bill generation interface with batch processing capabilities), Payment Tracking (comprehensive payment status dashboard with overdue alerts), and System Analytics (comprehensive reporting dashboard showing usage statistics, popular meals, and financial summaries). The interface will feature advanced filtering, sorting, and export capabilities.

4. System Architecture

4.1. High-Level Diagram

A visual representation of the application's architecture.

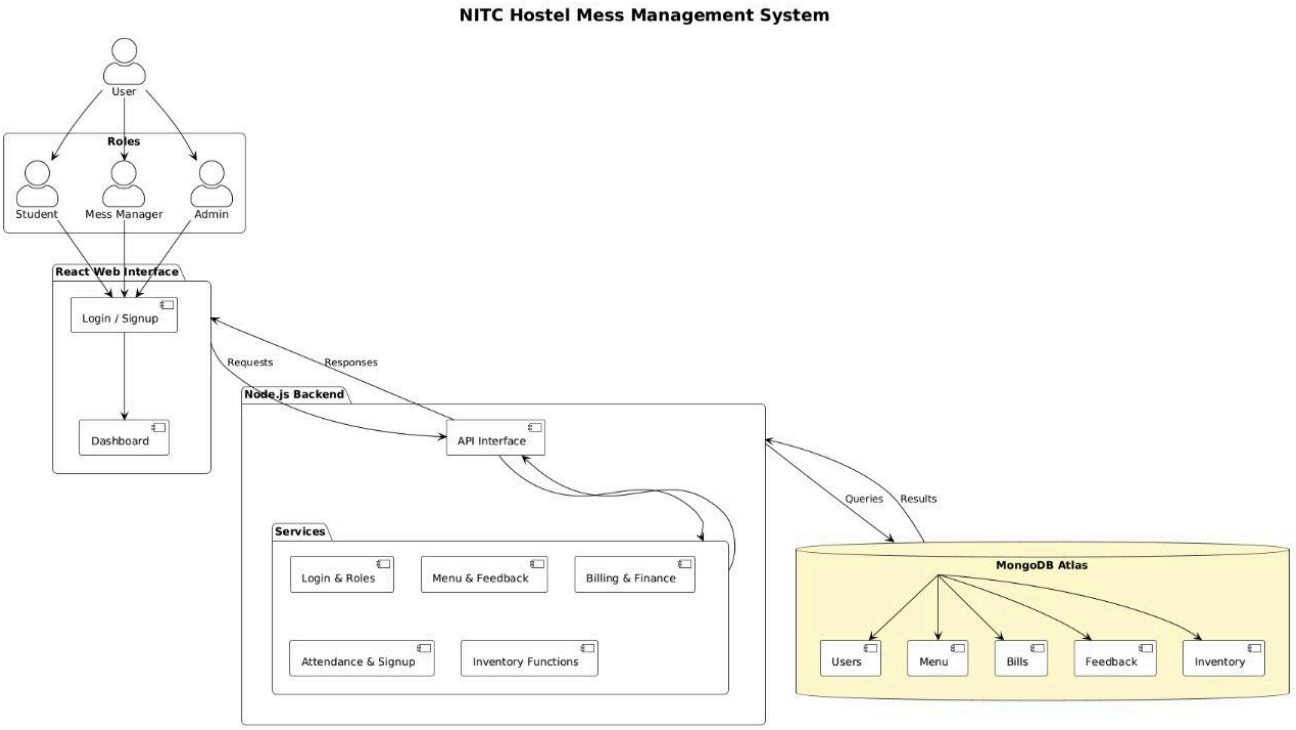


Figure 7: High level diagram of the application's architecture

- **Client (React):** A web application running in the user's browser with three distinct interfaces - Student Dashboard (menu viewing, feedback submission, attendance marking, bill tracking), Mess Manager Dashboard (menu posting, feedback analytics, inventory management), and Admin Dashboard (user management, bill generation, payment tracking). The frontend communicates with the backend via REST API calls.

- **Server (Node.js/Express):** A RESTful API server that handles all business logic including user authentication with JWT tokens, role-based access control, menu management operations, feedback processing, attendance tracking, automated billing calculations, and data validation. The server manages communication between the frontend and database while enforcing user permissions.

- **Database (MongoDB):** A NoSQL database hosted on MongoDB Atlas that stores all application data including user profiles, weekly menus, student feedback, attendance records, and billing information. The database is optimized with proper indexing for efficient queries and supports the scalable document structure needed for the mess management system.

4.2. Technology Stack

- **Frontend:** React 18.x, React Router 6.x, Axios, Tailwind CSS
- **Backend:** Node.js 20.x, Express 4.x, Mongoose 8.x
- **Database:** MongoDB 7.x
- **Authentication:** JSON Web Tokens (JWT)
- **Deployment:** Vercel (Frontend), Heroku (Backend), MongoDB Atlas (Database)

4.3. Database Schema (MongoDB Collections)

4.3.1 Users Collection:

- Fields: user_id, name, email, password (hashed), role (Student/Manager/Admin), roll_number, created_at
- Purpose: Stores authentication and profile data for all system users including students, mess managers, and administrators.

4.3.2 Menus Collection:

- Fields: menu_id, week_start_date, meals (array of day, breakfast, lunch, dinner), created_by, created_at
- Purpose: Stores weekly meal schedules posted by mess managers for student viewing.

4.3.3 Feedback Collection:

- Fields: feedback_id, student_id, meal_date, meal_type, rating (1-5), comment, created_at
- Purpose: Stores meal ratings and comments submitted by students for quality improvement.

4.3.4 Attendance Collection:

- Fields: attendance_id, student_id, date, is_present (boolean), created_at
- Purpose: Tracks daily attendance records for students to calculate monthly billing accurately.

4.3.5 Bills Collection:

- Fields: bill_id, student_id, month, year, total_days, present_days, amount, is_paid (boolean), created_at
- Purpose: Stores monthly billing information generated automatically based on student attendance records.