ITAHARI INTERNATIONAL COLLEGE · INNOVATION LAB

Creative Clash Main Event · February 24–27, 2026

Clash-A-Thon 2026

The Friction We Forget

Building Solutions for Problems We've Stopped Noticing

Kick-off	Submission Deadline	Final Presentation
Feb 24, 2026 | 09:30 AM	**Feb 26, 2026 | 4:00 PM**	**Feb 27, 2026 | On-site**

TEAM & PROJECT INFORMATION

Team Name	LoadSense
Project Title	LoadSense
Project Category	Educational

TEAM MEMBERS

#	Full Name	Role / Responsibility	GitHub / Contact
1	Amit Pokharel	Leader	9847226995
2	Aaryan Karki	Member	9805322257
3	Samir Bhandari	Member	9820533662
4	Anish Tamang	Member	9703077363

5	Isha Karki	Member	98472114444

SUBMISSION LINKS

GitHub Repository	LoadSense
Deployment URL	React , Nodejs
Tech Stack	MERN Stack, Flutter

Declaration

We declare that this project is our original work developed during Clash-A-Thon 2026. All external resources and libraries have been appropriately credited. We understand that plagiarism or misrepresentation will result in disqualification.

Table of Contents

Table of Figure

1 Technical Documentation

1.1 Section A: System Architecture

LoadSense is meant to be a cross-platform, academic workload management system that has been available either in a web application or a mobile application. The structure adheres to a three-layer model in the formed architecture, namely, Interface Layer, Logic Layer, and Storage Layer. Through this separation of concerns, scalability, maintainability and clarity of the system design is achieved.

Overall Architecture Flow

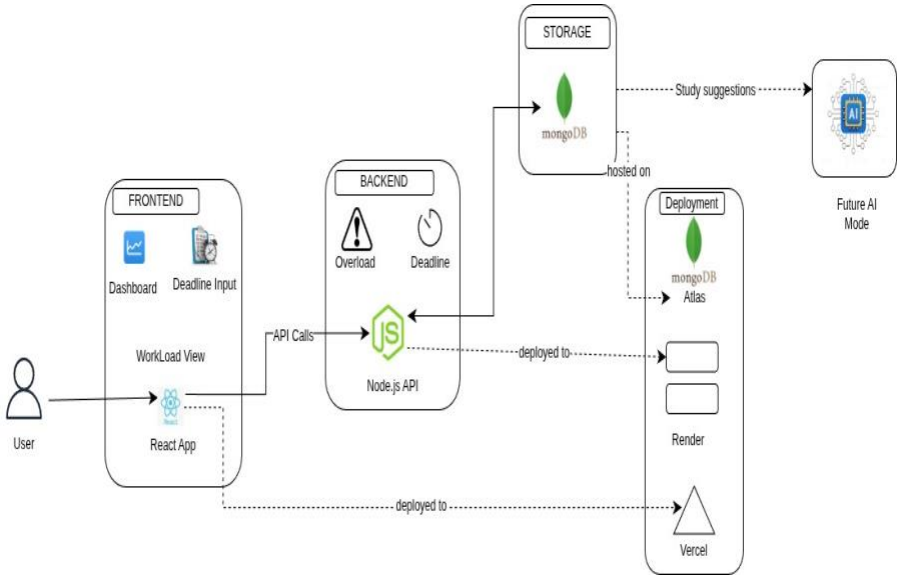

Figure 1: System Architecture

User

Interface Layer (Web & Mobile, RESTful API Communication,Logic Layer (Backend Processing), Storage Layer (Database), feedback sent back to Interface.

Interface Layer

Two platforms make up the interface layer:

Web Application: Built on React as MERN backends, Offers a reacting core user interface, Deployed via Vercel

Mobile Application: Developed using Flutter**,** Coded in Dart programming language , Android and iOS-compatible.

The interface layer has the following responsibility:

Registration and login of users, Entrys and management of the deadlines, Visualization of workload display, Notifications in case of overloading and Module management

Logic Layer

The logic layer is developed based on Node.js and Express.js. This is the base processing unit of the system.

The main tasks are:

JWT authentication and authorization,Checking of the input and processing of the request, Identification of deadlines, Calculation of the workload intensity, Overload alert generation,Formatted response processing of JSON.

Workload detection mechanism is rule-based, takes the deadlines into consideration in a 7-day rolling window. All types of academic evaluation will have a weight assigned to them to show its intensity:

Quiz - Low weight, Viva - Moderate weight, Assignment - High weight, Project - Very high weight

The work load score will be calculated by:

Workload Score = Σ **(Task weighted Estimated Hours)**

When the calculated score is more than some set limit, the system sends an overload alarm.This reasoning will make sure that the application can deliver meaningful insights as opposed to being just a deadline tracker.

Storage Layer: MongoDB Atlas is a cloud-hosted NoSQL database that is used in the implementation of the storage layer. The Object Data Modeling (ODM) tool is named as Mongoose and is used to design schemas and handle data interaction.

The storage layer ensures:

On-demand storage of user data, Guaranteed schedule and deadline control, Productive workload aggregation, Congruency of data between sessions and the ability to add features in the future.

1.2 Section B: Component Breakdown

LoadSense is divided into functional parts to ensure clarity and separations of functions.

1. Authentication Module

This module deals with the secure access to the system. It includes:

User registration, Login authentication,Hashing of passwords with bcypt, JWT token generation and Protected route middleware.

2. Deadline Management Module

In this module, we have data about academic tasks, and it has:

Creating new deadlines, Editing existing deadlines, Deleting deadlines, Re-calling impending appraisals and Connection of deadlines to courses.

3. Workload Analysis Engine

This component carries out non-trivial processing by:

Weighing of evaluation, identification of clustering in 5-7 day intervals, computing the scores of workload (weekly), triggering overload alerts, supporting Proactive academic.

4. Dashboard Module

The dashboard is applied to the web and mobile interfaces and it displays:

Visualization of the workload on a weekly basis, Upcoming deadline summary,Alert indicators and Workload ratio depiction.

Component Communication

The system has a pattern of request-response:

A backend receives an HTTP request sent by the interface,the request is processed by the backend controller, Business logic is executed, There are database queries done,The interface is given a structured JSON response, The interface is updated about the UI.

1.3 Section C: Data Design

LoadSense makes use of NoSQL data model based on documents. The design of the database schema is based on three main entities.

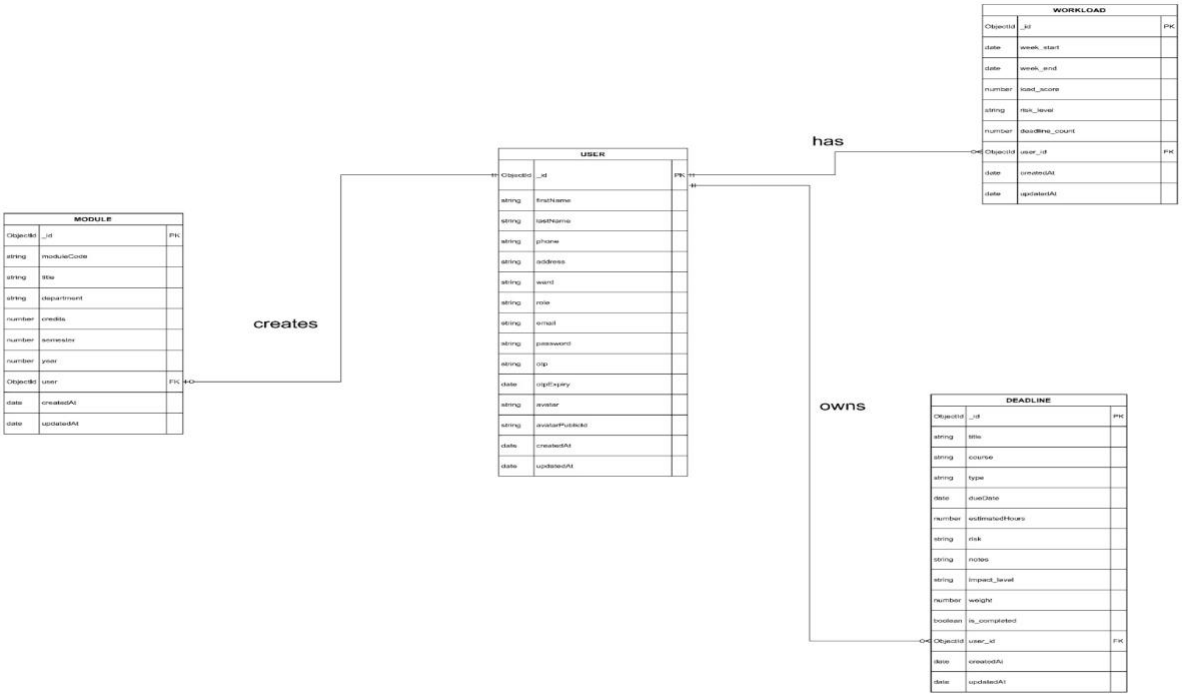

Figure 2: ERD

User Entity

Attributes:Unique identifier (id), First name,Last name,Email address,Hashed password,Account creation timestamp

The users are individual students who are using the system.

Module Entity

Attributes:Unique identifier (id),Course name,Course code, Reference to user (userId),Creation timestamp

The Module are also associated with their respective users.

Deadline Entity

Attributes: Unique identifier (id),Task title, Task type,Due date, Estimated completion hours, Optional notes, Reference to user (userId),Reference to module (moduleId),Creation timestamp

Relationships

Each User can enroll in multiple Modules per semester.

A User creates and manages multiple academic Deadlines.

A User may maintain multiple calculated Workload records.

One Workload record aggregates multiple Deadlines within timeframe.

Each Deadline links to specific Module via moduleCode.

1.4 Section D: API / Interface Specifications

The web and mobile applications make calls to RESTful APIs of the LoadSense.

Authentication Endpoints

POST /api/auth/register, POST /api/auth/login, GET /api/auth/me

Deadline Endpoints

POST /api/deadlines, GET /api/deadlines, PUT /api/deadlines/:id, DELETE /api/deadlines/:id

Response Structure

Effective answers should take the following format:

{

success: true,

data: { }

}

This structure of error responses is as follows:

{

success: false,

error description: Some description of the error.

}

1.5 Section E: Setup and Deployment.

Local Development Setup

Backend: Go into backend directory, Run npm install**,** Setting environment variables and Execute npm run dev

Web Frontend: Change to frontend directory, Run npm install**,** Execute npm run dev

Mobile Application (Flutter): Browse to mobile directory, Run flutter pub get**,** Execute flutter run.

Required Environment Variables

MONGOURI, PORT, JWTSECRET, JWTEXPIRESIN, EMAILUSER, EMAILPASS, CLOUDINARYCLOUDNAME, CLOUDINARYAPIKEY, CLOUDINARYAPISECRET, GEMINIAPIKEY, GOOGLEAPI_KEY

2 Business Documentation

2.1 Section F: Business Model Canvas

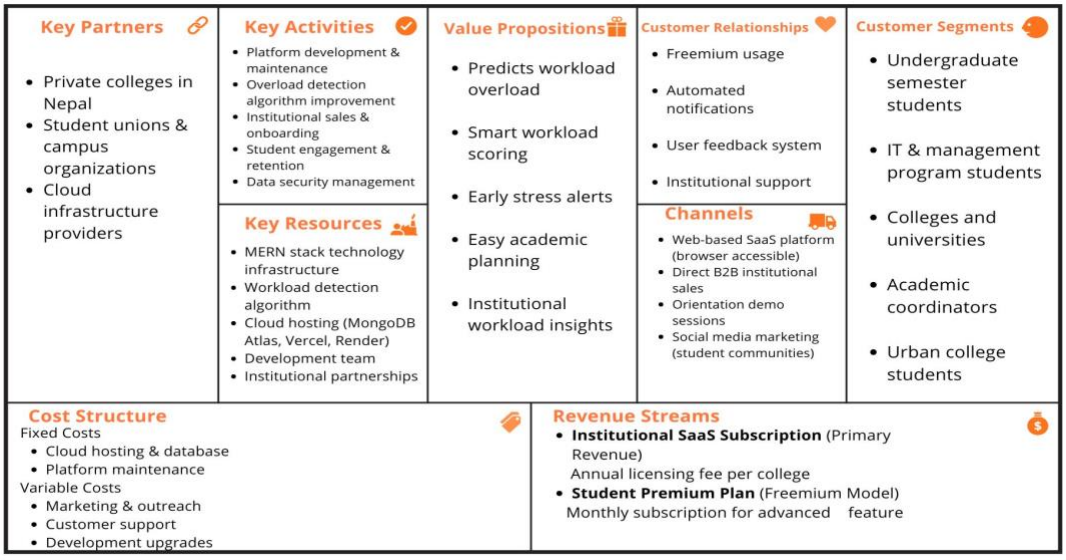

Figure 3: Business Model Canvas

2.2 Section G: Unique Selling Proposition

Core Differentiator: LoadSense is a workload intelligence system, not a deadline monitor. It detects academic overload before stress accumulates by analyzing clustering of high-stakes assessments and credit weight distribution.

What We Do Better: Weekly workload intensity scoring, Cross-course clustering detection, Automated overload alerts, Built specifically for Nepal's semester-based academic system, Institutional-level analytics dashboards

Why Customers Choose LoadSense: Solves a persistent academic stress problem digitally, Simple interface requiring no training, Actionable recommendations, not just raw data, Provides visibility for both students and administrators

2.3 Section H: Market Analysis

Target Market: LoadSense will focus on higher education and undergraduate students in universities and colleges that are semester-based in Nepal and in urban and semi-urban cities (such as Kathmandu, Pokhara, Biratnagar, Bharatpur, and Itahari) with over 70% institutional technology penetration post-COVID (KC, 2025). As research by (Rajesh

Karki, 2026) shows deadline clustering is an acute problem in this section with 46.7% indicating a high academic stress due to overlapping assessments.

Why Private Colleges: Higher EdTech adoption post-COVID, Competitive focus on student experience, Greater flexibility in adopting new digital tools,

Market Size Estimation: Nepal has more than 1,400 recognized colleges that are affiliates of Tribhuvan University, Pokhara University and others with a population of about 400,000+ undergraduate in-enrolment each year (University Grants Commission, 2024). The primary target of the company is private colleges with a number of about 600 and more likely to make an investment on the student facing technology.

Market Level	Estimate
TAM (Total Addressable Market)	Approx. 400,000 undergraduate students in Nepal across all institutions
SAM (Serviceable Addressable Market)	Approx. 180,000 students in private colleges with higher tech adoption
SOM (Serviceable Obtainable Market - Year 1)	Approx. 15,000 students across 30-50 pilot institutions

Nepal has over 1,400 affiliated colleges, with approximately 600+ private colleges, making institutional entry scalable.

Competitor Evaluation

Competitor	Strength	Weakness
Google Calendar	Widely used, reminders	No overload detection
LMS (Moodle, Classroom)	Course deadlines visible	No cross-course analysis
Notion/Trello	Customizable	Manual setup, no automation
Excel	Flexible	No alerts, not scalable

Market Gap: No dedicated academic workload detection tool exists in Nepal.

Market Opportunity & Timing: Increased EdTech adoption post-COVID, Growing, student mental health awareness,High smartphone & laptop penetration, Private colleges seeking differentiation, No direct competitor in this niche,Go-To-Market Strategy (First 100 Users)

Phase 1 - Campus Pilot (Month 1–2): Pilot in 1–2 private colleges, recruit 5 campus ambassadors, Conduct demo sessions, Offer first semester free for pilot institutions

Phase 2 - Organic Growth (Month 2–3): Social media campaigns,Peer-to-peer referrals,Orientation week promotions,Collect testimonials & case studies

Goal: Acquire first 100 active users and validate engagement.

2.4 Section I: Sustainability & Future Scope

LoadSense is designed to be financially self-sustaining through a B2B institutional licensing model. Rather than charging individual students, LoadSense monetizes through the colleges and universities that deploy it. This ensures a stable, recurring revenue base while keeping the platform free at the point of use for students. The core financial sustainability strategy rests on three pillars:

Institutional SaaS Subscriptions: Private colleges and universities pay an annual license fee based on student enrollment tiers, providing predictable recurring revenue.

Freemium-to-Premium Conversion: A free pilot period (one semester) allows institutions to experience the value of LoadSense before committing to a paid plan, reducing the barrier to adoption.

Low Operating Costs: LoadSense operates as a cloud-hosted web application with minimal infrastructure costs. The founding team handles development, marketing, and support in-house during the early stage, keeping the burn rate low.

Revenue Strategy Overview

LoadSense follows a B2B SaaS (Software as a Service) revenue model where the primary paying customers are educational institutions, not individual students.

The revenue model is structured into three tiers based on institutional size:

Plan	Students	Monthly Fee	Annual Fee	Features
Starter	Up to 500	NPR 8,000	NPR 80,000	Core workload scoring, overload alerts, basic dashboard
Growth	500 – 2,000	NPR 18,000	NPR 180,000	All Starter features + admin heatmaps, department analytics
Enterprise	2,000+	NPR 35,000	NPR 350,000	All Growth features + API access, LMS sync, priority support

Next Features to Build

- LMS Integration: Connect with learning management platforms to auto-import deadlines
- Faculty Analytics Dashboard: Enable instructors to view class-level workload impact of their scheduled evaluations
- Multi-department Collaboration: Allow academic regulators to view aggregated workload data across affiliated colleges

6-Month Vision: Pilot LoadSense in 3–5 private colleges, onboard 500+ student users, and refine workload analytics algorithms using real academic data. Secure primary institutional partnerships and validate the B2B licensing model, where colleges subscribe annually to offer LoadSense to their students.

1-Year Vision: Expand to 15–20 colleges, reach 5,000+ active users, and officially launch the premium subscription model for students.

Long-Term Sustainability Strategy: Initially, the B2B model ensures stable and predictable institutional revenue. As brand awareness and user trust grow, transitioning into B2C (student premium plans) will diversify revenue streams and reduce reliance on colleges. In future phases, LoadSense will also expand to corporate users facing workload and burnout challenges, further strengthening long-term financial sustainability and scalability.

3. References

KC, A., 2025. *Nepal EdTech Guide: Sustainable Digital Learning for Teachers.* [Online] Available at: https://gurkhatech.com/nepal-teachers-edtech-adoption-guide/ [Accessed 26 February 2026].

Rajesh Karki, A. T. M. K., 2026. Perceived Academic Stress and Its Correlates Among Secondary School Adolescents in Kathmandu, Nepal. *Public Health Chall.,* Volume 5, p. 1.

University Grants Commission, 2024. *Education management information system,* s.l.: University Grants Commission.