Islamic University of Technology
Department of Computer Science and Engineering

# Project RPS:
# Retake Prevention System

## Web Programming Project Proposal

*Submitted by:*

[Group - 11]

Sadman Shaharier Mahim - 220041133

Didhiti Nahid - 220041117

Ridita Alam - 220041110

*Submitted to:*

Farzana Tabassum, Lecturer, CSE

Islamic University of Technology

Course Code : CSE 4540
Course Title : Web Programming Lab
Section : Lab 1
Date of Submission : Monday, 15 December, 2025

# Table of Contents:

# Project Overview

RPS or Retake Prevention System is a satirical yet functional academic productivity web application. It's function is to act as an academic management platform designed to aggressively combat procrastination through psychological pressure and gamification.

This is not just any other "to-do" list app. This passively waits for user input and actively calculates a "Pain Score" based on the academic load of the user and their procrastination habits. It analyzes courses, assignments, deadlines, and the users behaviors and estimates burnout risk when the load starts to become overwhelming. As the situation worsens, the UI of the application changes from Calm mode to other modes like Grind, Panic or Doom mode and imposes humorous penalties on the user to force them into a productive mode.

This project has been made with the theme "Life, But make it weird" by making something uselessly useful through making everyday activity like task tracking into an over engineered and dramatic experience.

# Motivation Behind the Project

University students always have a lot on their plate. They frequently struggle with managing their assignments, deadlines and the biggest problem, their procrastination. The added stress of finals week really makes it difficult to focus because it apparently seems impossible to address it all.

Most productivity tools passively track tasks but fail to adapt their behavior based on increasing academic pressure or repeated procrastination patterns.

The solution: A system that visualises this stress as a quantifiable metric (the “Pain score” ) and then prompts the user to react to it using a myriad of different methods. It reprimands the student for procrastinating by using “Digital shame” and again rewards the student for completing tasks in time. This also allows the user to reflect on their actions more and see the consequence of their slacking-off and improve their productivity habits while keeping the system entertaining and non-intrusive.

From a technical perspective, the project provides an opportunity to implement a complete MERN-based system with authentication, middleware, asynchronous processing, file handling, and multiple interacting schemas in a cohesive and meaningful way. It also allows to explore how predictive logic and reactive system behavior can make task management more engaging and self-regulating.

# Key Features

Didhiti

### 1. User Authentication and Profile Management:

- Secure registration and log in using JWT-based authentication.
- ~~Optional~~ OAuth login (e.g., Google) may be supported.
- Personalized dashboard showing courses, tasks, and workload predictions for every user.

**Use Case Description:**

A student registers using their email or Google account. Upon successful authentication, the system retrieves their unique user ID and loads a personalized dashboard. This dashboard immediately visualizes their current "Pain Score," creating a secure and custom environment for tracking their specific academic load.

### 2. Course and Academic Data Management:

- Adding, updating, and deleting courses.
- Assignments and exams creation with details such as: → Difficult
  - Deadline
  - Weight or importance
- All academic data stored in a structured database for analysis.
- Organize materials in the system to keep their instructive PDFs in a unified place.
- Task sorting based on both the deadline and the priority/difficulty of the task.

**Use Case Description:**

The user creates a new assignment entry (e.g., "Lab Report") and uploads the relevant instruction PDF. The system stores this data in the MongoDB backend and automatically re-sorts the user's "To-Do" list. Instead of a simple chronological list, the system prioritizes tasks that have both high difficulty and an approaching deadline, ensuring the most critical work appears at the top.

## 3. The "Pain Score" Algorithm & Dashboard

- **Dynamic Stress Calculation:** The system calculates a real-time "Pain Score" (0–100) using a weighted algorithm based on assignment difficulty, credit hours, and time remaining.

$$\text{Pain Score} = (\text{Task Count} \times \text{Avg Difficulty}) + \sum_{i=1}^{n} \left( \frac{\text{Weight}_i \times \text{Difficulty}_i}{\text{Days Remaining}_i} \right) + (\text{Procrastination Score})$$

This score model effectively separates the impact of academic load, deadline urgency, and behavioral factors into independent components.

- **Visual Mode Switching:** The UI reacts to the Pain Score. As the score increases, the dashboard switches modes in real time:
  - Relaxed Mode (0–30): Calm blue/green UI while data loads. ⟷ Red
  - Grind Mode (30–60): Focused interface as rising workload is detected.
  - Panic Mode (60–80): High-contrast visuals and urgent reminders when deadlines cluster.
  - Doom Mode (80+): Red theme, intense alerts, and a highlighted “High Workload Pressure” warning.
- The Weekly Survival Forecast updates.

**Use Case Description**

The user logs in three days before midterms. The system recalculates all upcoming tasks using the weighted Pain Score formula.  The Weekly Survival Forecast updates with predicted high-stress days based on upcoming deadlines. The user immediately sees they are entering a heavy academic week and can adjust priorities accordingly.

## 4. Procrastination Management & Penalties

The system identifies procrastination patterns, such as:

- Tasks left untouched for long periods
- Frequent pauses during work sessions
- Deadlines being missed

Based on detected behavior, the system applies responses such as:

- Stronger reminders.
- **Digital "Shame" Penalties:** If a deadline is missed, the user must perform a "redemption task" (e.g., typing a formal apology letter to their future self in a modal window) to regain full access to their dashboard.
- **Dramatic Notifications:** The system sends unnecessarily dramatic alerts if tasks are ignored. Example: “Your task weeps in your absence…” or “The database is disappointed in you.”
- **Variable Physics Timer:** A focus timer that physically speeds up depending on the user's productivity history. If the user has been slacking, a “1 hour” timer might actually take 50 minutes to tick down, punishing them with the sense that time is moving a lot faster.

**Use Case Description:**

A user ignores a deadline for a major project. The system detects this failure and locks the main navigation bar. To unlock it, the user is forced to open a modal and type a 50-word “Apology Letter” to themselves. Additionally, when they finally start working, the “Variable Physics Timer” runs 10% faster than real-time as a penalty for their previous inaction, forcing them to work quicker to log the same amount of hours.

## 5. Satirical Achievements:

- The system awards badges for questionable academic behavior. Examples:
  “Master of Procrastination” (for waiting until the last hour)
  or “Sleepless Warrior” (for submitting between 3 AM and 5 AM).

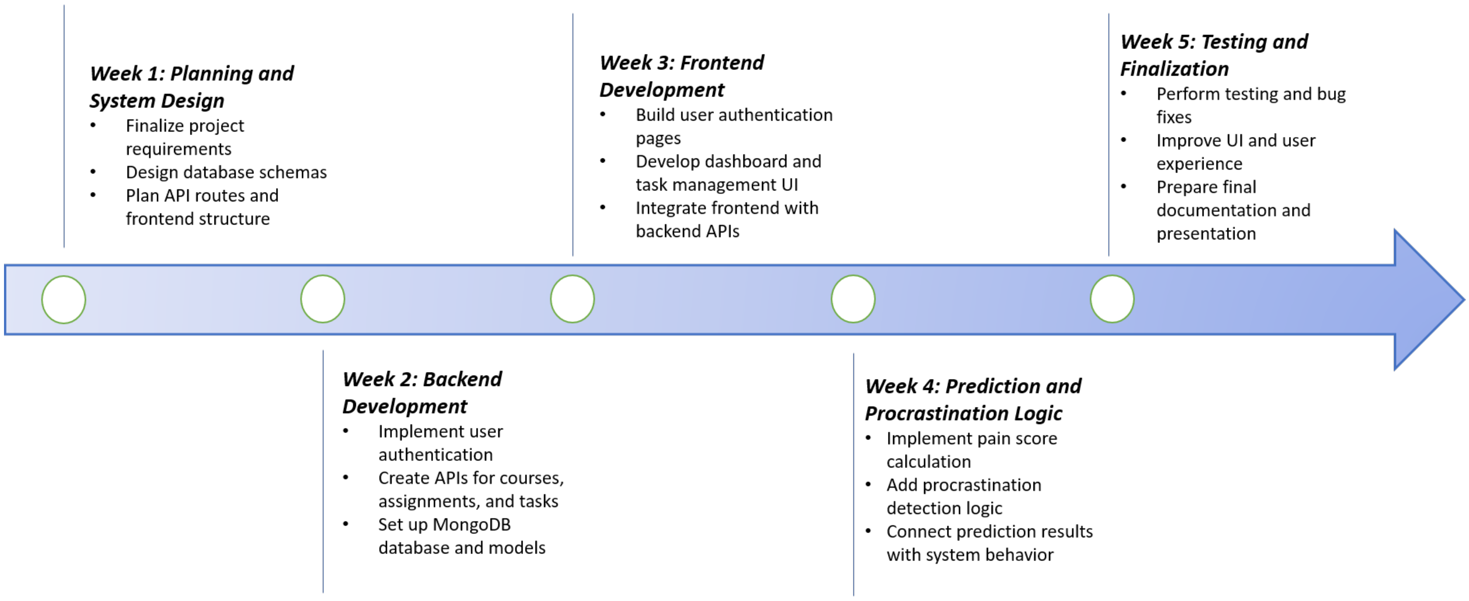
**Use Case Description:**

A user marks an assignment as “Complete” at 4:15 AM. The system checks the timestamp, recognizes the unhealthy work hour, and unlocks the “Sleepless Warrior” badge on the user's profile, providing a humorous acknowledgement of their “all-nighter.”

## Tools and technologies

| Category | Tool / Technology | Purpose |
|---|---|---|
| Frontend | React.js | User interface development |
| | HTML5 & CSS3 | Structure and styling |
| | JavaScript (ES6+) | Client-side logic |
| Backend | Node.js | Server-side runtime environment |
| | Express.js | Backend framework and REST API development |
| Database | MongoDB | NoSQL database for storing users, courses, tasks, and predictions |
| Authentication & Security | JWT (JSON Web Tokens) | Secure user authentication |
| | OAuth (optional) | Third-party login support |
| Development Tools | Git & GitHub | Version control and collaboration |
| | Postman | API testing |
| | VS Code | Code editor |

## Proposed Timeline



***Week 1: Planning and System Design***

- Finalize project requirements
- Design database schemas
- Plan API routes and frontend structure

***Week 2: Backend Development***

- Implement user authentication
- Create APIs for courses, assignments, and tasks
- Set up MongoDB database and models

***Week 3: Frontend Development***

- Build user authentication pages
- Develop dashboard and task management UI
- Integrate frontend with backend APIs

***Week 4: Prediction and Procrastination Logic***

- Implement pain score calculation
- Add procrastination detection logic
- Connect prediction results with system behavior

***Week 5: Testing and Finalization***

- Perform testing and bug fixes
- Improve UI and user experience
- Prepare final documentation and presentation