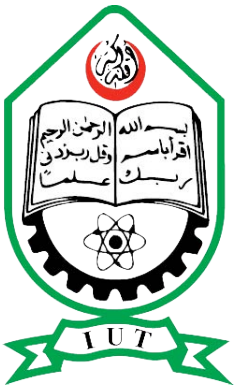
Islamic University of Technology



Web programming lab

Cse 4540

Project Proposal Report

Project Title: *IdleInsights*

Team Members:

Nayeemul Hasan Prince - 220041125

Ferdous Reza Habib - 220041138

Alfi Shahrin - 220041153

Supervisors:

Farzana Tabassum

Lecturer, Computer Science and Engineering, IUT

Njayou Youssouf

Lecturer, Computer Science and Engineering, IUT

Contents

1 Project Overview

The *IdleInsights* is a highly specialized web application built entirely on the MERN (MongoDB, Express, React, Node.js) stack. It acts as a detailed self-diagnostic tool by moving beyond traditional time-tracking. The system captures both a user's Intended Tasks (with a subjective Difficulty Score) and their subsequent Confessional Log (detailing the distraction activity). This comparative data feeds an asynchronous algorithm that calculates the user's Procrastination Velocity Index (PVI) and identifies their specific behavioral Root Cause Pattern, which is then used to provide interactive, data-driven countermeasure.

2 Motivation

The core motivation for this specific application stems from the gap between traditional productivity tools and the psychological reality of procrastination. Most "To-Do" apps focus on what was *not* done, leading to a cycle of guilt without providing insight into behavioral triggers.

IdleInsights is built to turn "wasted time" into actionable data. By forcing users to categorize their distractions and link them to the difficulty of avoided tasks, we create a feedback loop that identifies *why* a user is procrastinating. For example, a user may discover they only clean their house (Activation Delay) when faced with high-difficulty coding tasks. By quantifying these patterns through a **Procrastination Velocity Index (PVI)**, the app shifts from being a passive observer to an active analyst, providing specific "Modes" to break individual cycles of avoidance. This project fulfills the **"Uselessly Useful"** theme by over-engineering a highly clinical, data-heavy solution to an everyday human habit.

3 Key Features

The application's functionality is divided into four robust, interconnected systems:

3.1 Core Infrastructure and Data Capture

1. **Secure User Authentication**

Use-Case Description: Users can register and log in via email/password secured by JWT. All API access requires a valid token via Express middleware.

Technical Requirement Highlight: Token-based Authentication

2. **Intent Manager (CRUD)**

   Use-Case Description: Users create documents detailing their work intentions, including the task title and their personal Difficulty Score (1-5). This score is a mandatory input for the PVI algorithm.

   Technical Requirement Highlight: Multiple Schemas (IntendedTask) & Basic CRUD

3. **Confessional Log (CRUD)**

   Use-Case Description: The primary input where users log a distraction session, linking to the Avoided Task. Captures Duration, Actual Activity (e.g., 'Watching Videos'), and Activity Detail (e.g., 'YouTube/Cooking').

   Technical Requirement Highlight: Multiple Schemas (LogEntry) & Basic CRUD

3.2 Advanced Backend Logic & Analysis Engine

1. **PVI Algorithm** ***Use-Case Description:*** Upon saving a new LogEntry, a Mongoose post-save hook (middleware) triggers an asynchronous function that calculates the log's impact. The overall PVI (Procrastination Velocity Index) is updated based on total duration and difficulty of avoided tasks.

   Technical Requirement Highlight: Asynchronous Programming & Middleware Usage

2. **Root Cause Pattern ID**

   Use-Case Description: The asynchronous analysis function analyzes the user's logs for recurring patterns (e.g., 'Social Media' before 'Difficult' tasks). It then assigns a behavioral label (e.g., Overwhelm Avoidance) to the user's profile.

   Technical Requirement Highlight: Middleware Usage

3. **In-Dashboard Benchmarking Chart**

Use-Case Description: An endpoint that returns aggregated, anonymized user data, enabling the frontend to display a direct bar chart comparison of the user's PVI versus the overall system average PVI. The system generates a downloadable text/image summary of the user's stats compared to the system average.

Technical Requirement Highlight: Middleware & Frontend Data Support

3.3 The Analyst Dashboard & Intervention System

1. **PVI Velocity Gauge**

   Use-Case Description: The central dashboard visualization that dynamically changes color (Green/Yellow/Red) and accompanying text based on the user's current PVI score.

   Technical Requirement Highlight: Conditional Rendering & Custom Design

2. **Activity/Time Charts**

   Use-Case Description: Simple visualization components displaying the user's most time-wasting platform and their most distracted hours of the day.

   Technical Requirement Highlight: Frontend Data Visualization

3. **Countermeasure Generator (Modes)**

   Use-Case Description: The core interactive solution. The system uses Conditional Rendering to recommend a specific "Mode" based on the identified Root Cause Pattern (e.g., suggests "Task Deconstructor Mode" for Overwhelm Avoidance).

   Technical Requirement Highlight: Conditional Rendering

4. **Interactive Modes**

   Use-Case Description: Dedicated views accessed via React Router where users can use tailored micro-tools: e.g., a simple Pomodoro timer (using React Hooks) in the "Frequent Break Mode," or a sub-task lister in the "Task Deconstructor Mode".

   Technical Requirement Highlight: Proper Routing & Effective Use of Hooks

4 Tools and Technologies

- **Frontend:** React.js (Hooks, Components), React Router, Custom CSS (or modern framework like Tailwind/Bootstrap used exclusively for styling, not templates), Axios for API interaction.
- **Backend:** Node.js (Runtime), Express.js (Framework).
- **Database:** MongoDB Atlas (Cloud Database), Mongoose ODM (Object Data Modeling).
- **Security:** JSON Web Tokens (JWT), Bcrypt.js (Password Hashing).
- **Other Tools:** Git/GitHub (Version Control).

5 Proposed Timeline

A Gantt chart outlines the timeline for the development of our project, with key milestones:

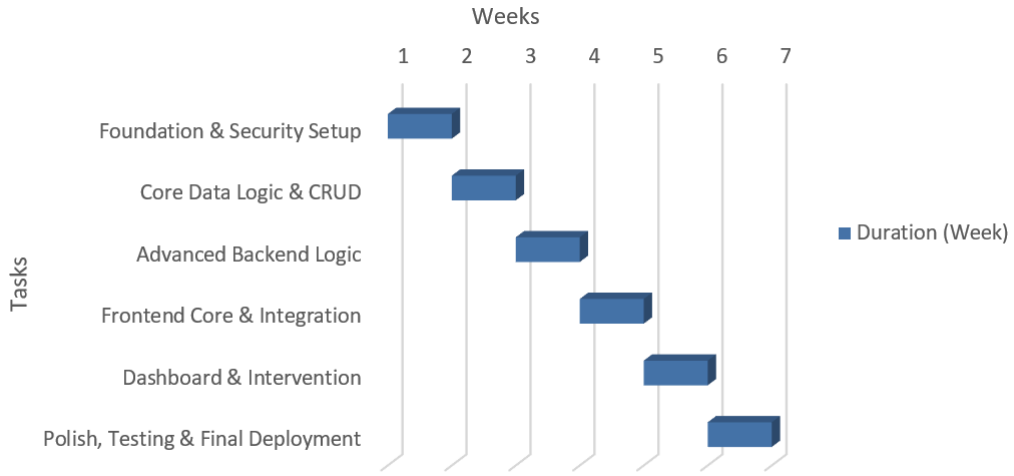


Figure-02: Proposed Timeline for Our Project *IdleInsights*

5.1 Key Milestones and Deliverables

Week 1 (Foundation & Security Setup): Define all Mongoose Schemas (User, IntendedTask, LogEntry). Set up Express server and database connection. Implement

secure JWT authentication (Register/Login routes) and JWT Middleware for route protection.

Week 2 (Core Data Logic & CRUD): Complete all CRUD operations for IntendedTask and LogEntry. Begin data aggregation methods required for the PVI.

Week 3 (Advanced Backend Logic): Implement the PVI Algorithm (Asynchronous function). Integrate Mongoose Middleware hooks to trigger analysis on log entry creation. Finalize Root Cause Pattern ID logic.

Week 4 (Frontend Core & Integration): Set up React component structure and React Router. Build the login/registration views, Intent Manager, and Confessional Log forms. Establish secure API connections across the application.

Week 5 (Dashboard & Intervention): Build the Analysis Dashboard. Implement Conditional Rendering for the PVI Gauge and Mode suggestions. Create the Activity/-Time Charts and the In-Dashboard Comparison Chart.

Week 6 (Polish, Testing & Final Deployment): Develop the Interactive Modes components (e.g., timers, listers). Perform comprehensive system-wide testing (unit and integration tests). Finalize custom CSS/styling, documentation, and prepare the project for deployment and presentation.