- Cover page

**SUMMER TRAINING/INTERNSHIP**

**PROJECT REPORT**

(Term June-July 2025)

## MACHINE LEARNING MADE EASY: FROM BASICS TO AI APPLICATIONS SKILL DEVELOPMENT COURSE

## Project Title:

PRODUCTIVITY TRACKER FOR STUDENTS

Submitted by

**Hilario Unami Ngwenya**

Registration Number:

**12322069**

Course Code:

**PETV79**

Under the Guidance of:

**Mahipal Singh**

# School of Computer Science and Engineering

- Certificate

APRIL 2025

Lovely Professional University, Punjab

BONAFIDE CERTIFICATE

Certified that this project report "PRODUCTIVITY TRACKER FOR STUDENT" is the Bonafide work of "HILARIO UNAMI NGWENYA" who carried out the project work under my supervision.

SIGNATURE

<<Name of the Supervisor>>

HILARIO UNAMI GWENYA



<<Signature of the Head of the Department>>

**SIGNATURE**

<<Name>>

**HEAD OF THE DEPARTMENT**

<<Signature of the Supervisor>>

- Acknowledgement

I express sincere gratitude to my mentor, **Mr. Mahipal Singh**, for his guidance before and during my attempt at this project. I also thank the School of Computer Science and Engineering for providing the tools and structure that helped me conceptualize and develop this productivity-focused application.

- Table of Contents

|  |  |
| --- | --- |
| Chapter | Title |
| 1 | Introduction |
| 2 | Training Overview |
| 3 | Project Details |
| 4 | Implementation |
| 5 | Conclusion |

- Chapter-wise Report

**Chapter 1: Introduction**

**Company profile**

The project was conducted by Hilario, independently and individually.

**Overview of Training Domain**

The training domain of the StudySaver project centers on browser-based behavioral analysis for predicting student focus during online learning sessions, specifically within the context of an introductory AI/ML course. The system captures real-time activity metrics including tab switches, periods of inactivity, keystroke frequency, mouse movement totals, and scroll events. These indicators are compiled into a structured dataset, each session labeled with automated focus ratings such as Attentive, Semi-Attentive, or Distracted based on engagement patterns.

This dataset serves as the foundation for training supervised classification models using Python, with tools such as pandas, scikit-learn, and matplotlib. The selected algorithms—typically Decision Trees or Random Forests—learn to associate behavioral signals with cognitive focus levels. The approach demonstrates how non-optical browser telemetry can serve as a proxy for distraction tracking, enabling intelligent feedback systems for learners and providing an early proof of concept for practical machine learning applications in digital education environments.

StudySaver’s training domain bridges technical implementation and behavioral modeling. By treating engagement as a quantifiable, data-driven phenomenon, the system offers both pedagogical insight and an accessible entry point for independent research in applied machine learning.

**Objective of the Project**

StudySaver aims to deliver a modern browser-based dashboard that helps students monitor study behavior, visualize time-on-task, and improve focus through a clean and responsive interface. By tracking distractions—such as tab switching, idle periods, and typing rhythm—the system automatically assesses focus levels and generates labeled session data. Each study session is summarized for the user, offering behavioral snapshots that encourage personal habit-building.

Beyond personal feedback, all data is exportable for structured analysis and model training. This dual purpose—self-reflection and machine learning utility—positions StudySaver as a tool for both improving individual academic discipline and contributing to broader attention modeling efforts within educational technology.

**Chapter 2: Training Overview**

**Tools & technologies used**

The StudySaver system is developed using a combination of frontend and backend technologies. TypeScript and React form the foundation of the user interface, supporting dynamic component rendering and modular design. JavaScript’s Blob API enables seamless CSV export functionality, while localStorage is used for persisting session data across refreshes and closures. For machine learning analysis, Python and NumPy are utilized to process session metrics and train classification models. HTML and CSS contribute to layout and styling, and Visual Studio Code serves as the integrated development environment throughout development. All exported data is structured in CSV format to ensure compatibility with Python-based workflows.

**Areas covered during training**

The training phase covered a wide range of practical and theoretical domains. At its core, the system focuses on browser-based behavioral analytics, capturing nuanced indicators of distraction and evaluating session engagement in real time. Key interaction metrics such as keystrokes and mouse activity were tracked continuously, while scroll behavior and tab-switch timestamps added depth to the understanding of focus patterns.

Idle signal analysis allowed the platform to measure inactivity and identify disengaged moments within a session. These signals were transformed into labeled focus ratings using a custom logic engine, which grouped behavior into categories like Distracted, Semi-Attentive, and Attentive. The processed metrics were stored in CSV format and used to train machine learning models—including Random Forest and Decision Tree classifiers—using Python as the computational layer. Real-time UI updates were integrated to give users immediate feedback, while localStorage ensured their progress was saved across refreshes. The dashboard also introduced performance visualization techniques, reinforcing the link between study behavior and data-driven focus evaluation.

**Daily/weekly work summary**

* **Day 1** involved initial planning and component architecture, including Vite + TypeScript setup.
* **Day 2** focused on implementing timer logic, session structure, and tab tracking flow.
* **Day 3** was dedicated to UI refinement with the integration of ShadCN and Tailwind for aesthetic responsiveness.
* **Day 4** shifted toward final dashboard polish, user testing, and ongoing documentation.

**Chapter 3: Project Details**

**Title of the project**

Productivity Tracker for Students (**StudySaver**)

**Problem definition**

The core problem addressed by the StudySaver project is the absence of intelligent, non-intrusive systems capable of monitoring and improving student focus during online study sessions. Conventional productivity trackers often overlook real-time behavioral distractions such as tab switching, prolonged inactivity, and subtle drops in user engagement—particularly within browser-based learning environments. Additionally, most tools fail to provide immediate, meaningful feedback on cognitive focus, leaving students unaware of their study quality and attention patterns.

StudySaver responds to this gap by analyzing browser activity to surface actionable insights. It delivers real-time focus ratings, distraction breakdowns, and structured datasets compatible with machine learning workflows. By relying solely on in-browser signals—rather than external sensors or manual input—the system empowers learners to reflect on and improve their attention levels through a streamlined, autonomous interface.

**Scope and objectives**

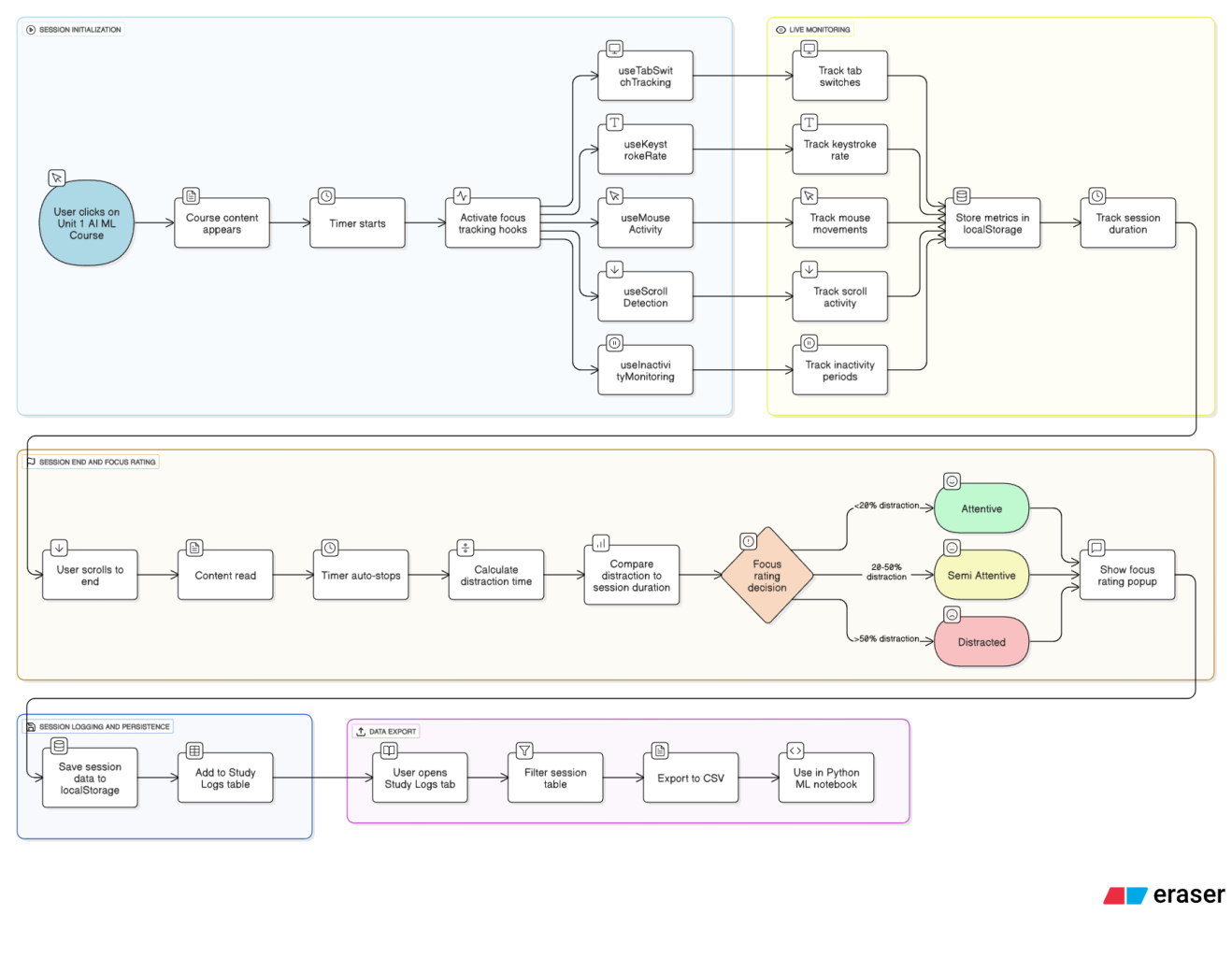
The scope of the StudySaver project includes the development of a responsive web dashboard specifically designed for students enrolled in online learning, particularly those engaging with AI/ML coursework. The system passively monitors browser behaviors including tab switches, inactivity intervals, keystroke rates, and scroll events to automatically assess focus levels during active study sessions.

Core features include real-time behavior tracking, post-session summaries, distraction visualizations, and structured CSV export options for downstream machine learning use. The central objective is to convert passive browser metrics into intelligent feedback mechanisms that support self-awareness, habit formation, and long-term improvement.

By uniting user interface development, behavioral analytics, and supervised machine learning design, StudySaver offers a complete productivity toolkit that operates entirely within the browser—making it accessible, scalable, and personalized for student use.

### ****Architecture Diagram****

### ****Data Flow****

**Chapter 4: Implementation**

**Tools used**

* TypeScript & React
* JavaScript's Blob API, localStorage.
* Python /w NumPy
* Custom React Hooks, HTML & CSS
* Visual Studio Code, CSV

**Methodology**

The methodology adopted for StudySaver follows a structured pipeline encompassing behavioral tracking, distraction evaluation, and machine learning integration. During each study session, the system continuously monitors browser-level activity using custom React hooks. These hooks track tab switches, idle periods, keystroke rate, mouse interactions, and scrolling behavior, generating a rich dataset of real-time engagement signals.

Once collected, the signals are processed into calculated features such as distraction ratios and productivity scores. Based on these values, the app automatically assigns focus ratings—Attentive, Semi-Attentive, or Distracted—at the end of each session. All session-level data is stored locally in the browser using localStorage and can be exported as structured CSV files.

These files form the basis for downstream analysis in Python, where machine learning models such as Decision Trees or Random Forests are trained to learn correlations between behavior and focus classification. This pipeline demonstrates how cognitive engagement can be quantified and modeled using non-optical, browser-based telemetry within a web environment.

**Modules**

The platform is built around several modular components that support both functionality and user experience.

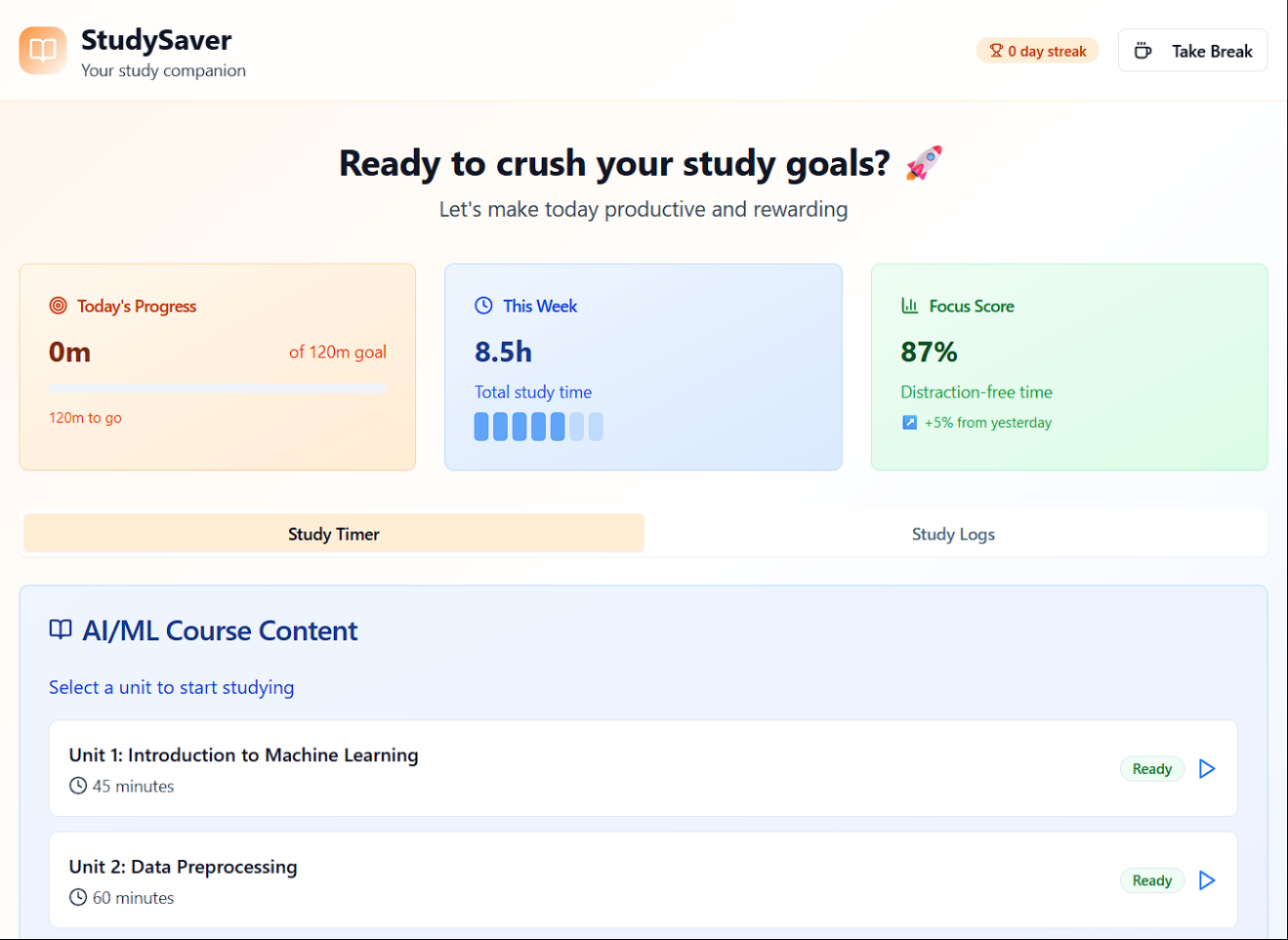
The User Interface Module renders study content, timers, focus ratings, and the central dashboard that displays session logs. The Focus Tracking Module is responsible for monitoring tab activity, keystrokes, scrolling intensity, mouse movements, and idle detection. The Session Management Module automates start-stop logic, enforces session boundaries, and manages timing events.

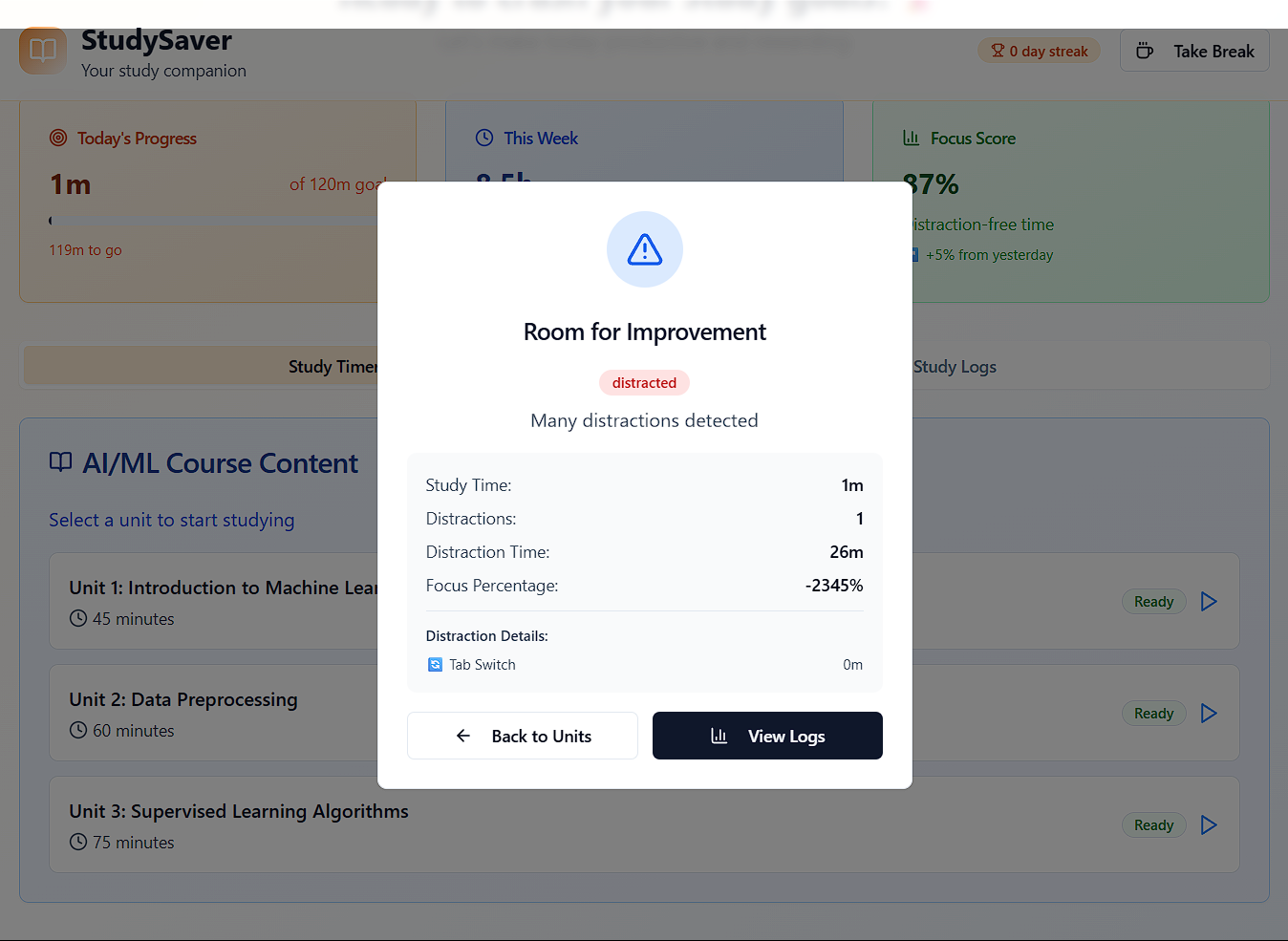
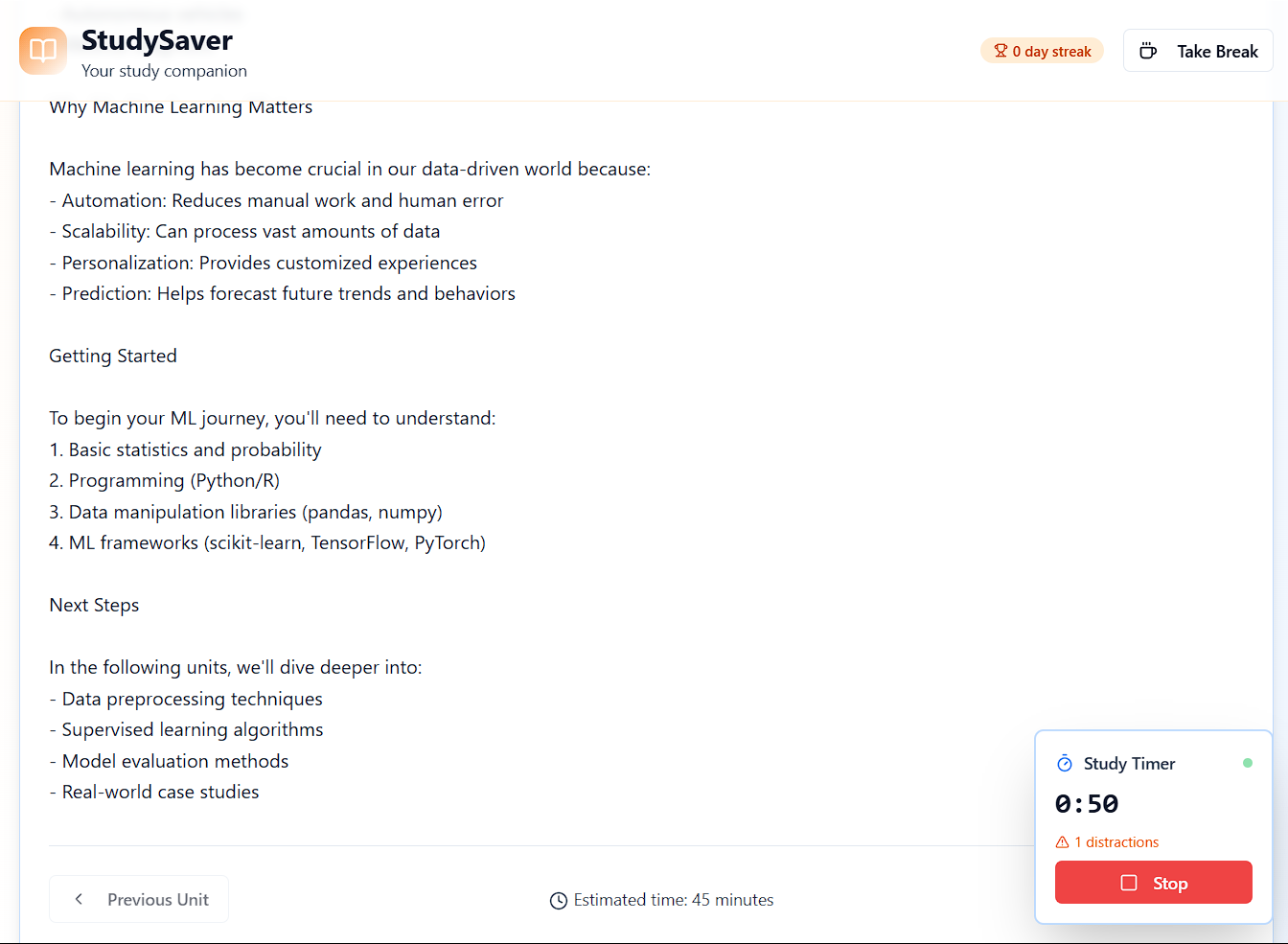
The Distraction Analysis Module processes raw metrics to calculate distraction ratios and assign the appropriate focus label. Data persistence is handled by the Data Storage Module, which saves metrics in localStorage and prepares them for export. The CSV Export Module generates download-ready datasets for external machine learning workflows.

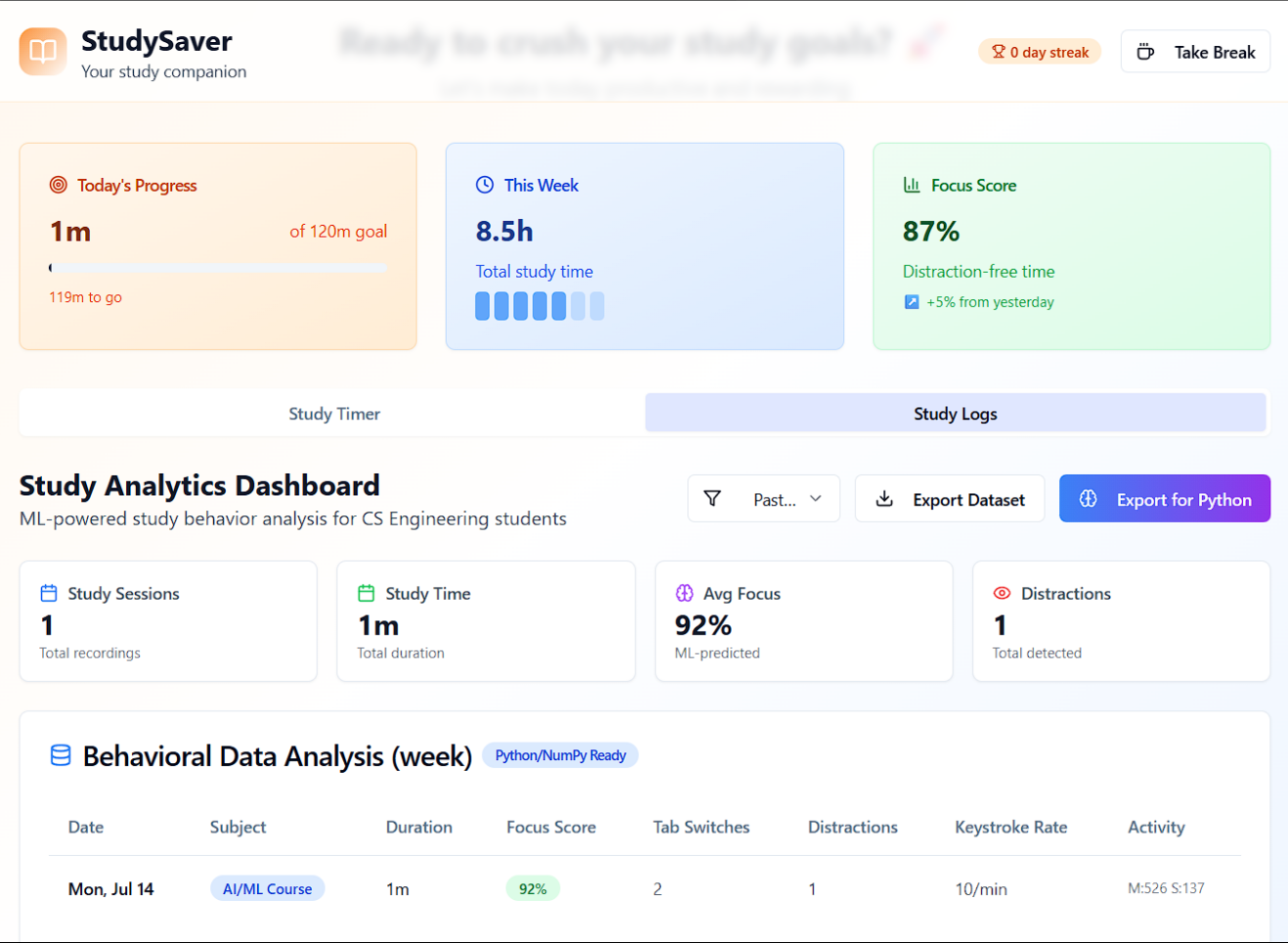
On the backend, the Machine Learning Module processes exported session data using Python and trains models to predict focus outcomes. The Course Content Module provides integrated study material (such as Unit 1 from the AI/ML curriculum) to ensure students engage with real content during tracked sessions. Finally, the Visualization Module displays charts and summaries inside the logs section, and the Onboarding Module introduces features and guides first-time users through a streamlined experience.

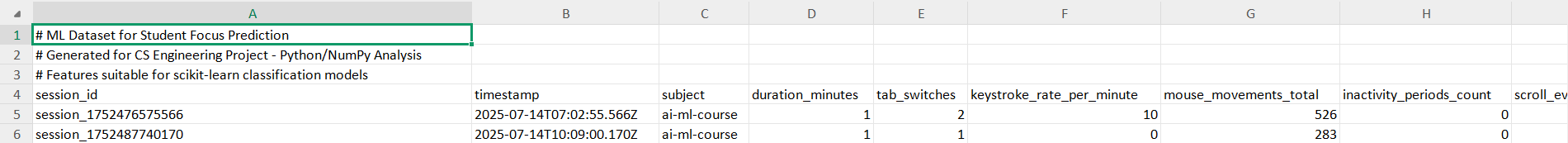
**Screenshots**

**Data Collection UI**

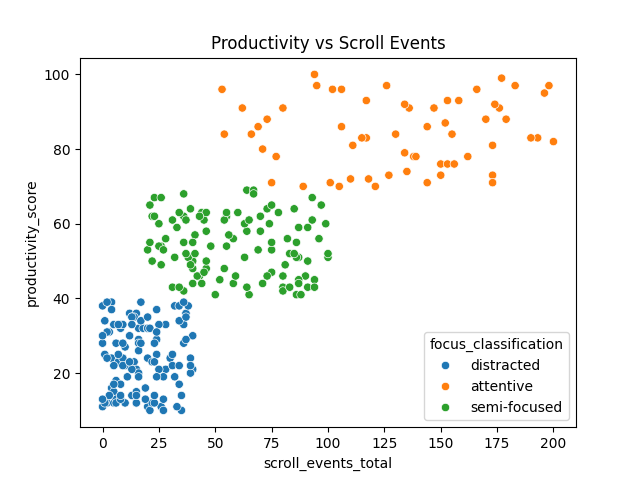
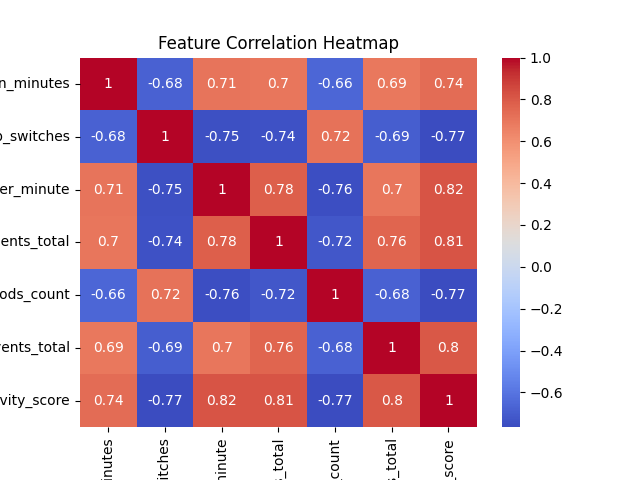
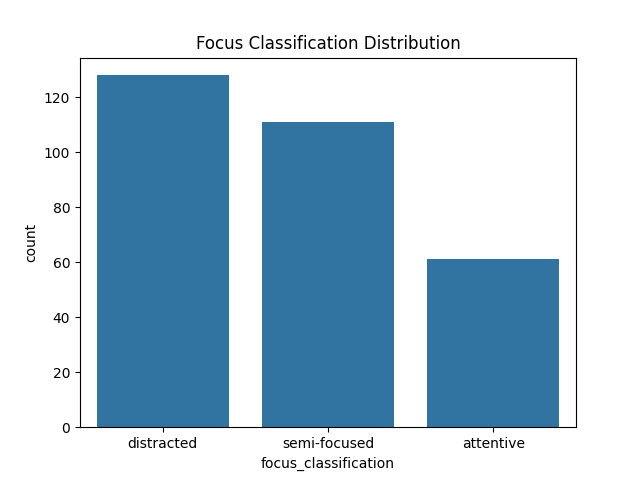
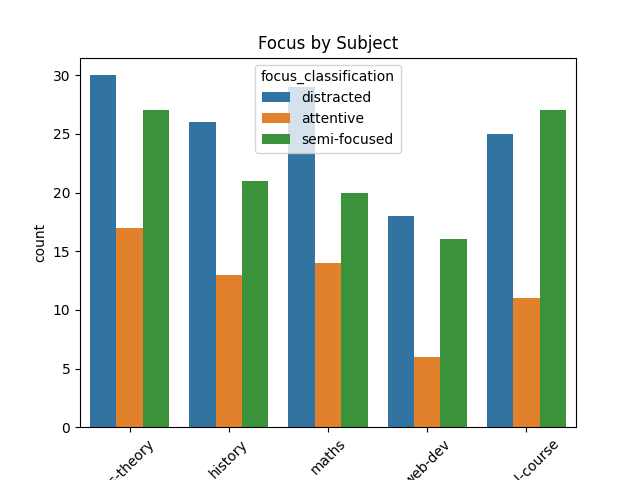
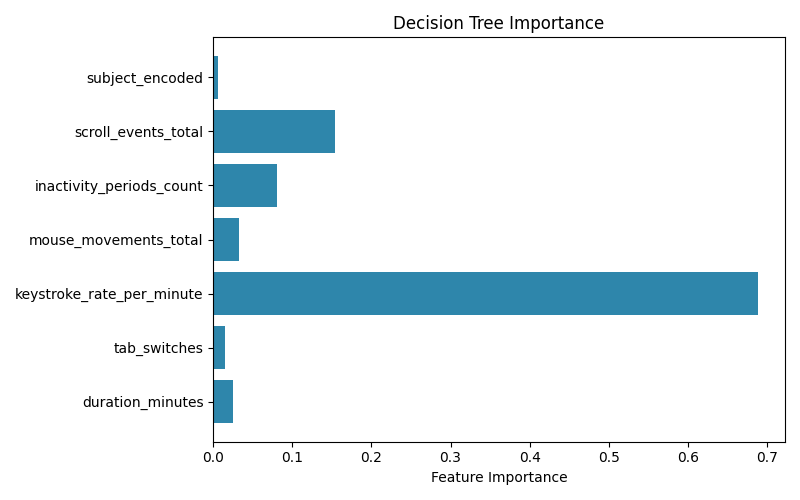
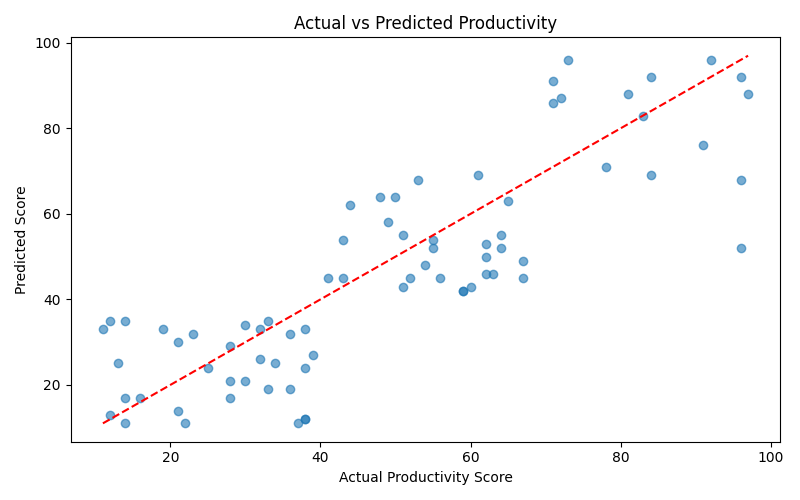
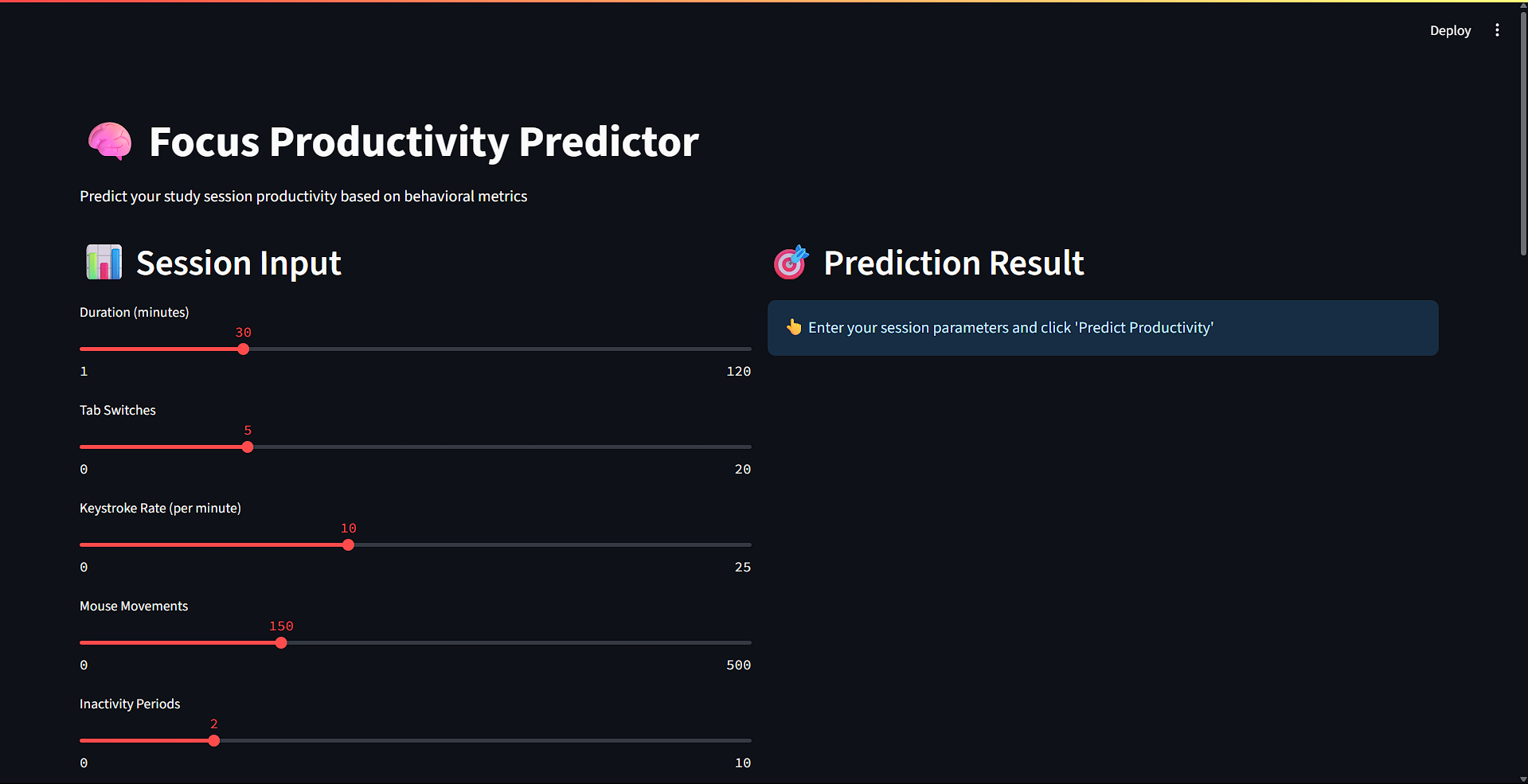


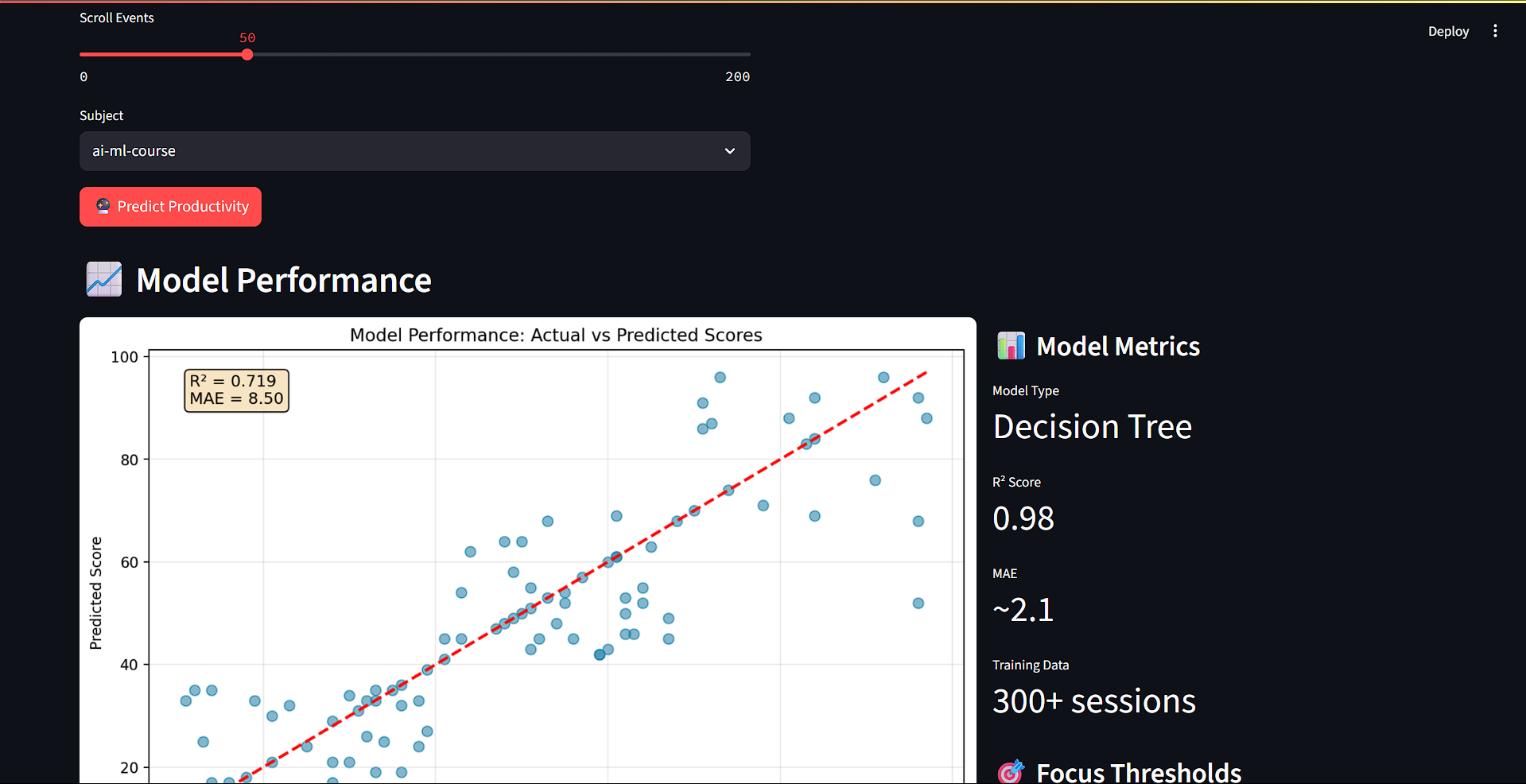


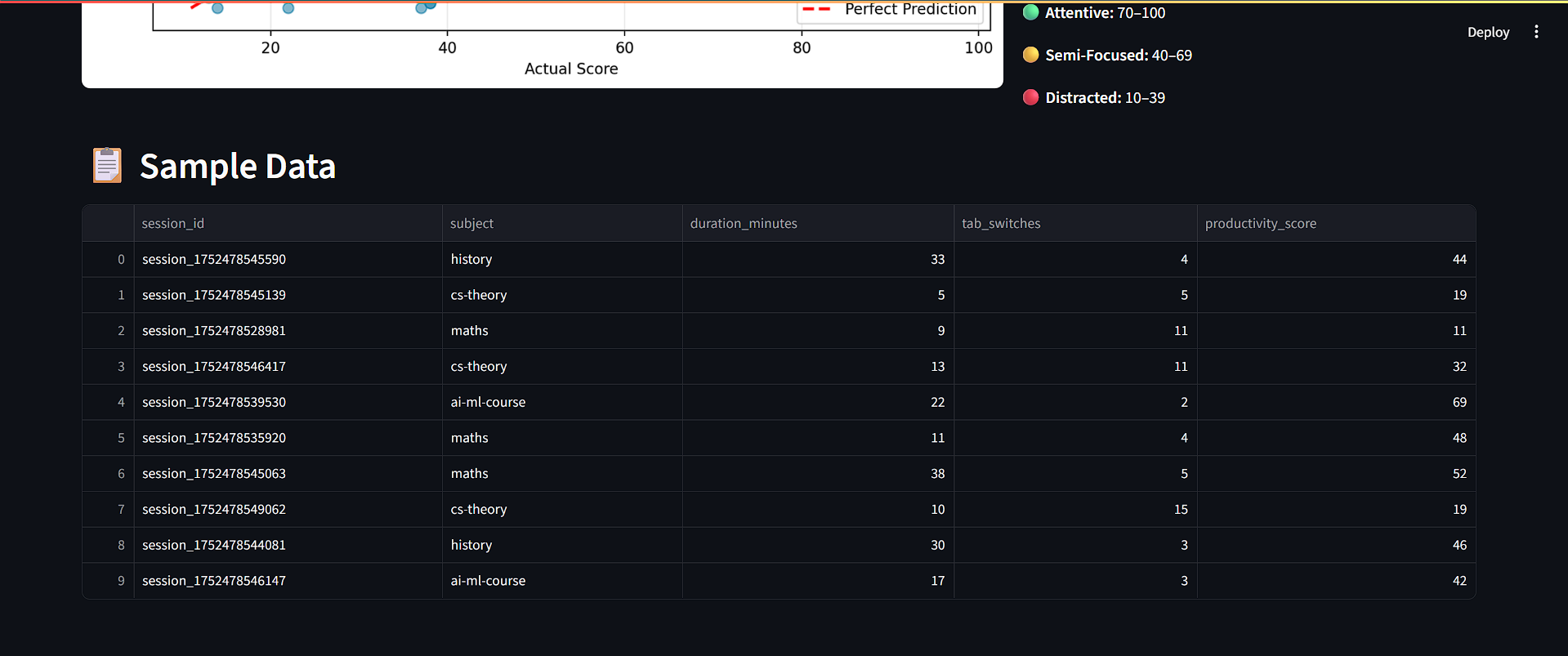




**Visualization**

**Dashboard Launch & UI**





**Code Snippets**

To illustrate StudySaver's internal mechanics, the following code samples highlight key implementation features within the frontend logic:

**Session Tracking Hook (Tab Switch Detection)** useSessionTracker.ts

typescript

useEffect(() => {

const sessionStart = Date.now()

const handleBlur = () => {

logDistraction('Tab switch', Date.now() - sessionStart)

}

window.addEventListener('blur', handleBlur)

return () => window.removeEventListener('blur', handleBlur)}, [])

This hook detects when a user switches away from the study tab by listening to the blur event. The distraction duration is calculated and logged, contributing to the distraction ratio used in focus classification.

**Saving Session to LocalStorage** storage.ts

typescript

export const saveSession = (data: Session) => {

const sessions = JSON.parse(localStorage.getItem('studyLogs') || '[]')

sessions.push(data)

localStorage.setItem('studyLogs', JSON.stringify(sessions))}

This function ensures each session is stored persistently in the browser, allowing users to maintain logs across refreshes. It forms the basis for session summaries and export functionality.

**CSV Export Functionality** exportCSV.ts

export const exportToCSV = (data: any[]) => {

const headers = ['Date', 'Duration']

const rows = data.map(session => [session.date, session.duration])

const csvContent = [headers, ...rows].map(e => e.join(',')).join('\n')

const blob = new Blob([csvContent], { type: 'text/csv' })

const url = URL.createObjectURL(blob)

const a = document.createElement('a')

a.href = url

a.download = 'study\_logs.csv'

a.click()}

This utility enables users to download their study logs in CSV format, preparing the dataset for external analysis and machine learning use in Python.

**Focus Prediction** focus\_model.ts

python

* Load the realistic dataset

df = pd.read\_csv("ml\_focus\_dataset\_realistic.csv")

* Train model

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

* Predict

y\_pred = model.predict(X\_test)

**Chapter 5: Results and Discussion**

**Output / Report**

The completed StudySaver system is publicly hosted and accessible via:

GitHub Repository – [unamihilario/focus-flow-dashboard-pal](https://github.com/unamihilario/focus-flow-dashboard-pal)

Deployment Link – [focus-flow-dashboard-pal](https://focus-flow-dashboard-pal.vercel.app/)

These resources include the full React-based frontend, session tracking hooks, and Python scripts for focus classification. The live deployment showcases how session data is collected and summarized, while the GitHub repository documents the technical structure of each module.

**Challenges faced**

The most significant challenge during development was designing a passive, non-intrusive system that could reliably detect distractions using only browser activity. Without access to sensors or manual feedback loops, StudySaver relied exclusively on event-driven metrics such as tab switching, idle intervals, and input frequency—requiring precise calibration to avoid false signals.

Integrating real-time tracking with a smooth user experience was another major hurdle. React hooks needed to operate continuously while maintaining performance, and localStorage had to handle asynchronous session data across refreshes and closures. Preserving timer continuity across tab transitions demanded custom logic to prevent resets and ensure accurate engagement duration.

Automatic focus rating based on behavior thresholds added further complexity, especially when accounting for sessions of variable length and differing interaction styles. It was also critical to shape the UI into a credible presentation platform that could function as a mentor-ready academic tool while reflecting the quality expected of a solo ML-focused engineering project.

Ultimately, designing CSV export functionality and formatting outputs for compatibility with Python workflows required attention to both technical detail and data hygiene—ensuring machine learning models could be trained without manual preprocessing.

**Learnings**

Through the development of StudySaver, deeper insight was gained into how everyday browser activity can be converted into meaningful datasets for machine learning. Tracking and modeling distraction behaviors highlighted the value of interaction signals such as tab switches, keystroke rates, and scrolling intensity when processed over time.

The project reinforced understanding of classification logic and how behavioral thresholds can be translated into real-time feedback. It also expanded frontend skills with TypeScript and React, while strengthening backend fluency in Python and NumPy-based preprocessing. Implementing localStorage persistence and CSV export logic proved essential for converting user behavior into reusable data.

Most importantly, StudySaver demonstrated how cognitive attention can be measured without hardware dependencies, connecting digital behavior with labeled training data in a way that supports academic modeling and personal self-improvement.

**Chapter 6: Conclusion**

**Summary**

The StudySaver project effectively demonstrates the potential of browser-based focus tracking as a tool for improving student productivity and attention. By combining behavioral signals such as tab switching, inactivity detection, keystroke analysis, and scroll behavior, the system evaluates session quality and assigns focus ratings without requiring manual input or external sensors.

Structured datasets exported via CSV allow for seamless integration with machine learning workflows, making StudySaver valuable for learners and researchers alike. The dashboard delivers intuitive feedback through clean design, automatic timers, distraction analytics, and persistent study logs—supporting both personal habit-building and academic understanding of attention patterns.

Ultimately, StudySaver bridges the gap between practical UI development and intelligent behavioral analytics. It stands as proof that solo, student-led applications can combine frontend sophistication with backend ML implementation to create meaningful educational tools. The system’s design, data pipeline, and predictive capabilities illustrate the power of combining user experience with machine learning insight, especially within the context of modern digital learning environments.