North South University

Department of Electrical and Computer Engineering

CSE327: Software Engineering
Semester: Summer 2025
Section: 06

FINAL PROJECT REPORT

Project Name: *KajBuzz*: An Intelligent Task Planner

Submitted To:

Dr. Mohammad Rezwanul Huq [MRH1]
Professor
North South University

Submitted By:

GROUP: 01	
NAME	**ID**
Rayed Riasat Rabbi	231****
Md. Azmine Amin Mormo	231****
Barshon Basak	231****
Md. Tazrian Hasnat	231****

Useful Links:

Hosting Site	https://rayed.pythonanywhere.com/
Video Demonstration	https://drive.google.com/drive/folders/1I8PbHTXyWKPAEjSrROfUhyyfcOctLbir?usp=sharing

SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

1. INTRODUCTION:

Intelligent Task Planner is a web-based productivity assistant that helps students to manage their tasks such as assignments, exam preparation, coding a project, personal reminders etc. It is a place where students unload their tasks and important notes, to better focus on what matters. Your mind should focus on what you are doing, not memorize your schedule to waste brain power.

1.1 PURPOSE OF THE DOCUMENT

The purpose of this SRS document is to give a comprehensive description about the project to be made for this semester's term project in CSE327 section 6.

1.2 DEVELOPERS & TESTERS BUILDING THE SYSTEM:

The development team consists of the listed group members.

1.3 END-USERS: UNIVERSITY STUDENTS (UNDERGRADUATE & GRADUATE)

Primary users are students, they will unload their tasks as soon as they get assigned one, offloading their brain from the pressure of remembering everything.

1.4 INTENDED USE

The application is used to better manage the user's time for their available time and schedule. The user is expected to unload their tasks as soon as possible. Maximum task management through our app will ensure the system knows about the user's full schedule and hence can suggest and manage the users time more efficiently. Load the task and forget, the software will remind you in the appropriate moment, so you don't forget anything and stay pressure free. Your virtual assistant (schedule assistant).

TABLE - 1: SCOPE

In-Scope	Out-of-Scope
Task CRUD operations with priority, deadline, and estimated effort	Payment processing or premium subscriptions
Rule-based scheduling engine, using time blocks	Native mobile apps (web-responsive only)
Weekly calendar view and time slot recommendations	Deep integration with third-party LMS (e.g., Canvas)
Browser/email reminders and manual schedule edits	Real-time collaboration (e.g., shared tasks)

In-Scope	Out-of-Scope
Pomodoro timer and visual analytics (charts for time trends)	Video conferencing or social features
Google Calendar sync	Advanced ML training (focus on rule-based core)

2. OVERALL DESCRIPTION

2.1 USER NEEDS

Students need an intuitive tool to manage academic workloads efficiently. Key needs include:

- Easy input of tasks, classes, exams, and available time blocks.
- Automated scheduling to prioritize high-importance tasks and avoid overlaps.
- Flexibility for manual adjustments and intelligent adaptation over time.
- Insights into productivity via optional analytics and reminders to build better habits.

TABLE - 2: FUNCTIONAL REQUIREMENTS

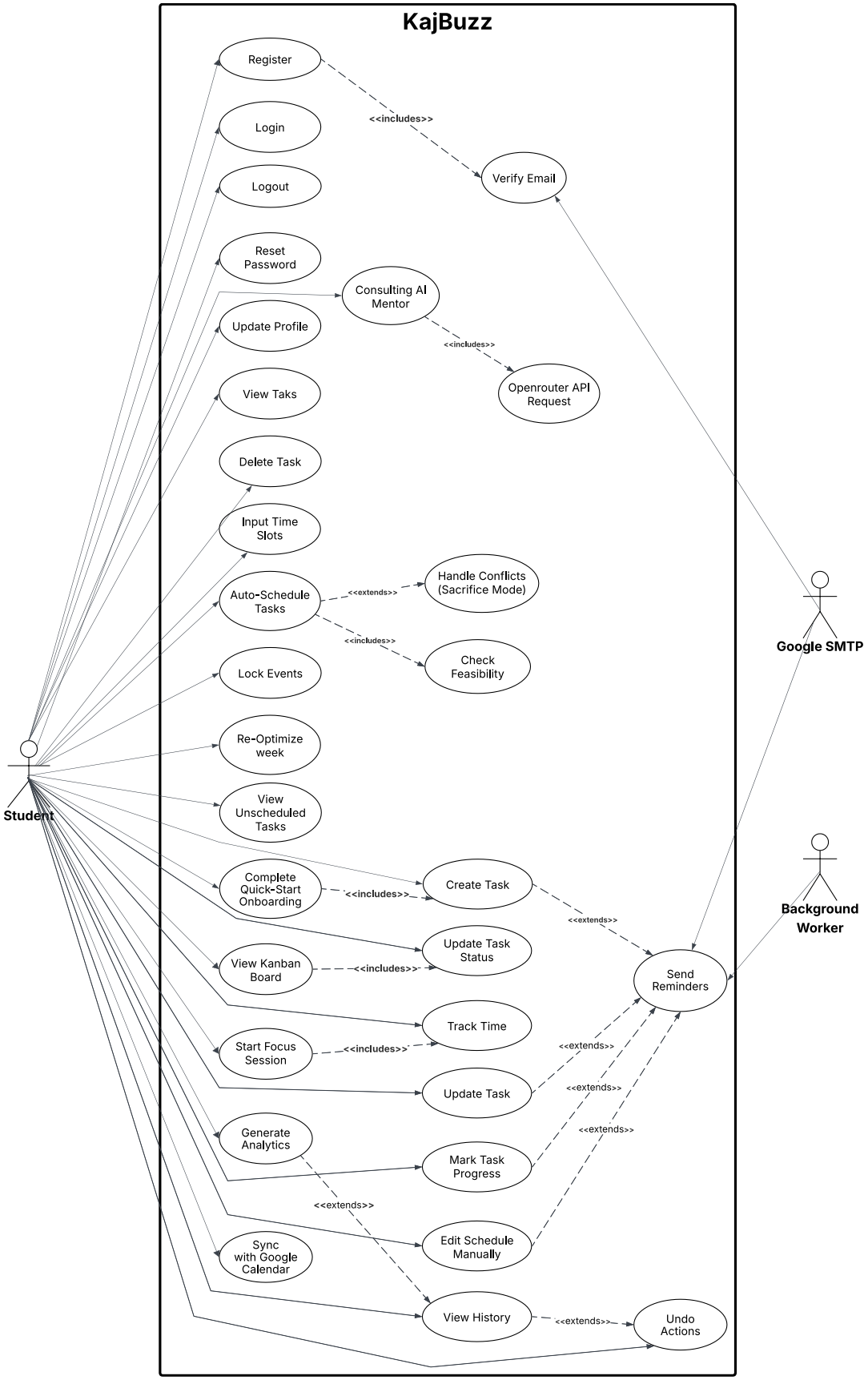
ID	Requirement	Acceptance Criteria
FR-1	User Account	Users can register, login, logout, reset password, and update profile (e.g., preferences for notifications).
FR-2	Task Management	CRUD tasks with fields: title, description, type, deadline, priority, estimated hours, tags, recurrence.
FR-3	Time-Block Input	Users input weekly recurring or free time slots
FR-4	Auto-Schedule	The core scheduling engine must perform Feasibility Check (Overload) and Conflict Resolution (Interruptions)
FR-5	Manual Edit	Users can drag-and-drop to move. These manual edits are direct overrides. To re-optimize the rest of the week's schedule around these manual changes, the user must use the "Re-optimize Week" option (see FR-11).
FR-6	Event Locking	When a user manually places or edits an event, they must have the option to "lock" it. Locked events are Fixed, they won't be overridden by "Re-optimize Week" function.
FR-7	Notifications	Send time-based reminders (e.g., 24h and 1h before tasks) through browser push or email.
FR-8	Progress Tracking	Users mark tasks as in-progress or completed; system logs actual vs. estimated time for analytics.
FR-9	Kanban Board View	The application must provide a Kanban board view where tasks are displayed as cards in columns indicating their status (e.g., "To Do", "In Progress", "Completed"). Users must be able to

ID	Requirement	Acceptance Criteria
		drag-and-drop tasks between these columns, and this action should automatically update the task's status.
FR-10	Pomodoro Timer	Integrated timer for 25 to 5-minute focus cycles within tasks, with tracking of sessions.
FR-11	Adaptive Re-optimization	The UI must feature a prominent "Re-optimize Week" button that re-runs the scheduler on all *uncompleted* tasks.
FR-12	Unscheduled Tasks Tray	The application must feature a consistant, clearly visible "Unscheduled" area (e.g., a collapsible sidebar) to hold all tasks that could not be placed or were bumped.
FR-13	History & Undo	View past schedules and undo/redo recent actions.
FR-14	Analytics	Generate charts (bar/line) for daily/weekly time usage, completion rates, and trends (e.g., procrastination index); include habit tracking.
FR-15	Google Calendar Sync	Enable sync with Google Calendar using OAuth 2.0 for exporting schedules.
FR-16	AI Scheduling Enhancement	The application will integrate with the *OpenRouter* API to provide an optional, AI-powered scheduling suggestion. This feature will be a progressive enhancement and the system will remain fully functional if the API is unavailable.

TABLE - 3: NON-FUNCTIONAL REQUIREMENTS

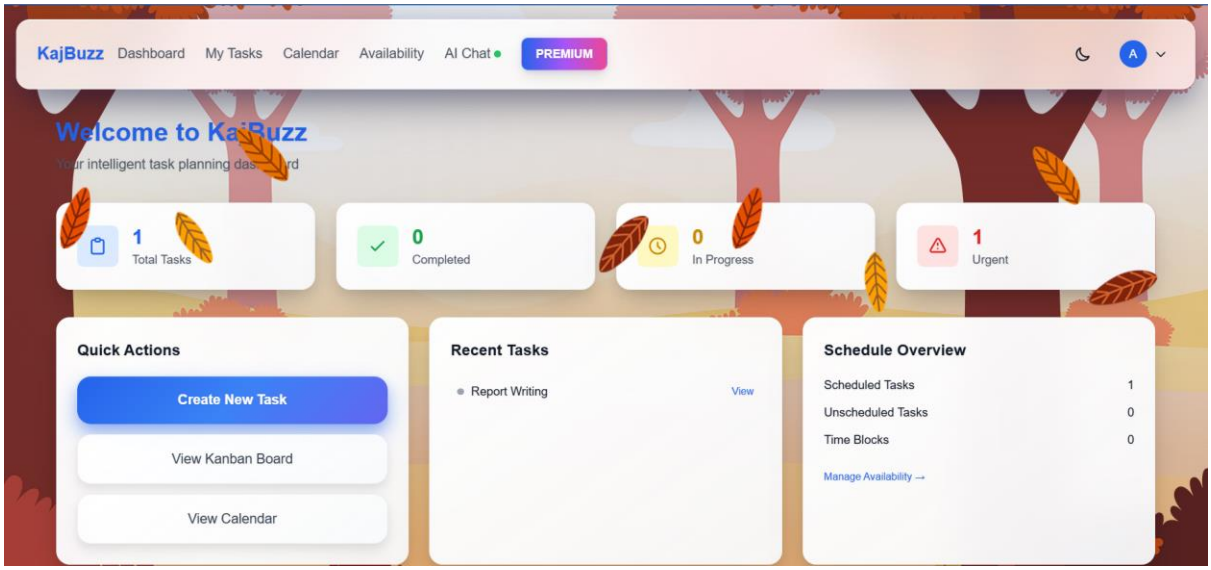
Category	Requirement
Performance	Schedule generation ≤ 2 seconds; page loads ≤ 1 second on standard hardware.
Usability	Intuitive, responsive UI (mobile/desktop) with clean design like TickTick; accessibility features (e.g., keyboard navigation).
Security	User authentication, data encryption, and privacy (e.g., GDPR compliance); secure API endpoints.
Scalability	Support up to 500 concurrent users; database optimized for multi-user access.
Reliability	99% uptime; handle edge cases like time overlaps with graceful errors.
Compatibility	Works on major browsers (Chrome, Firefox); responsive design for devices.
Maintainability	Code follows Django best practices; use UV for dependency management.

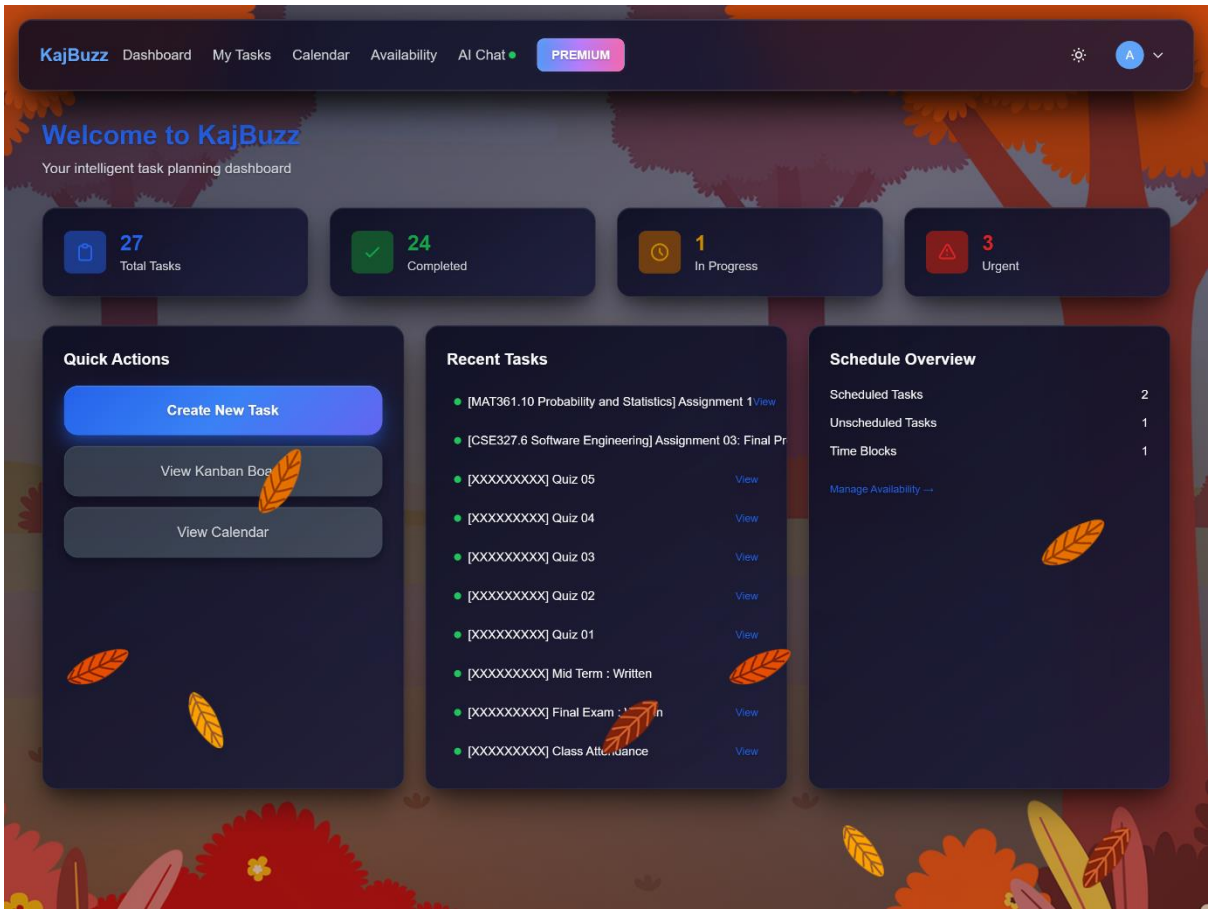
3. USE CASE DIAGRAM



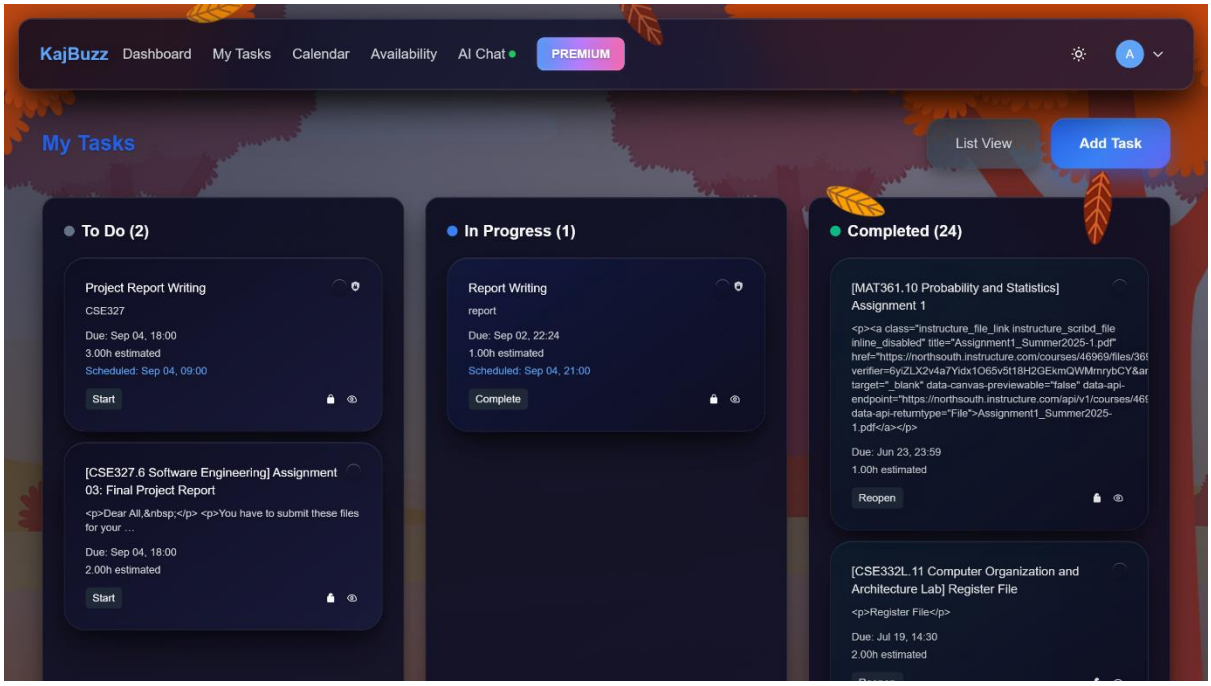
4. USER INTERFACE MOCKUP

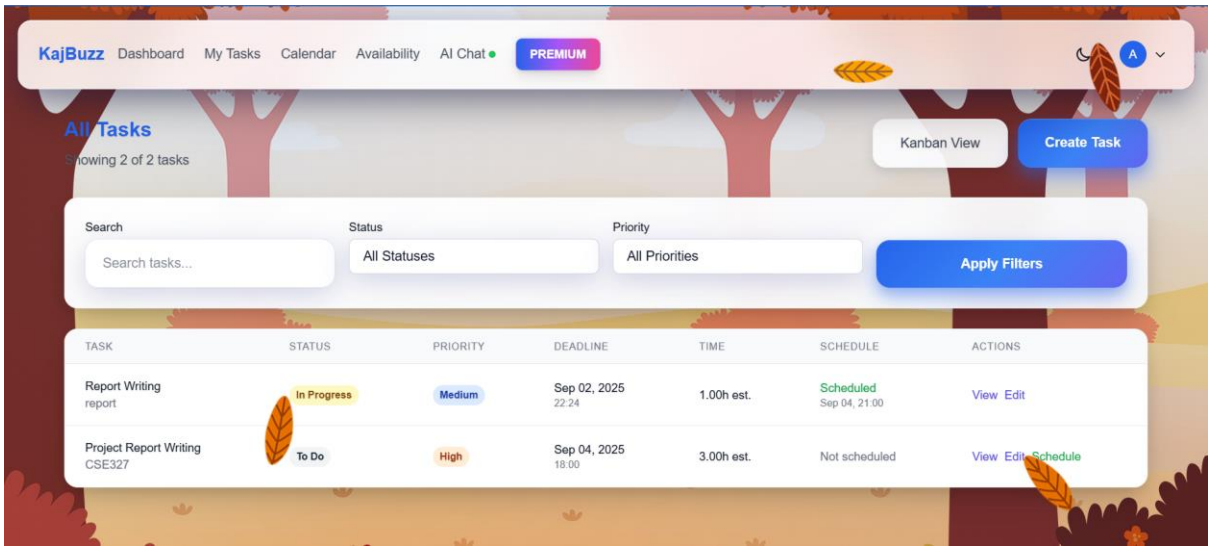
4.1: LANDING PAGE (Dark Mode & Light Mode)





4.2 KANBAN BOARD VIEW (Dark Mode & Light Mode)





4.3: POMODORO TIMER

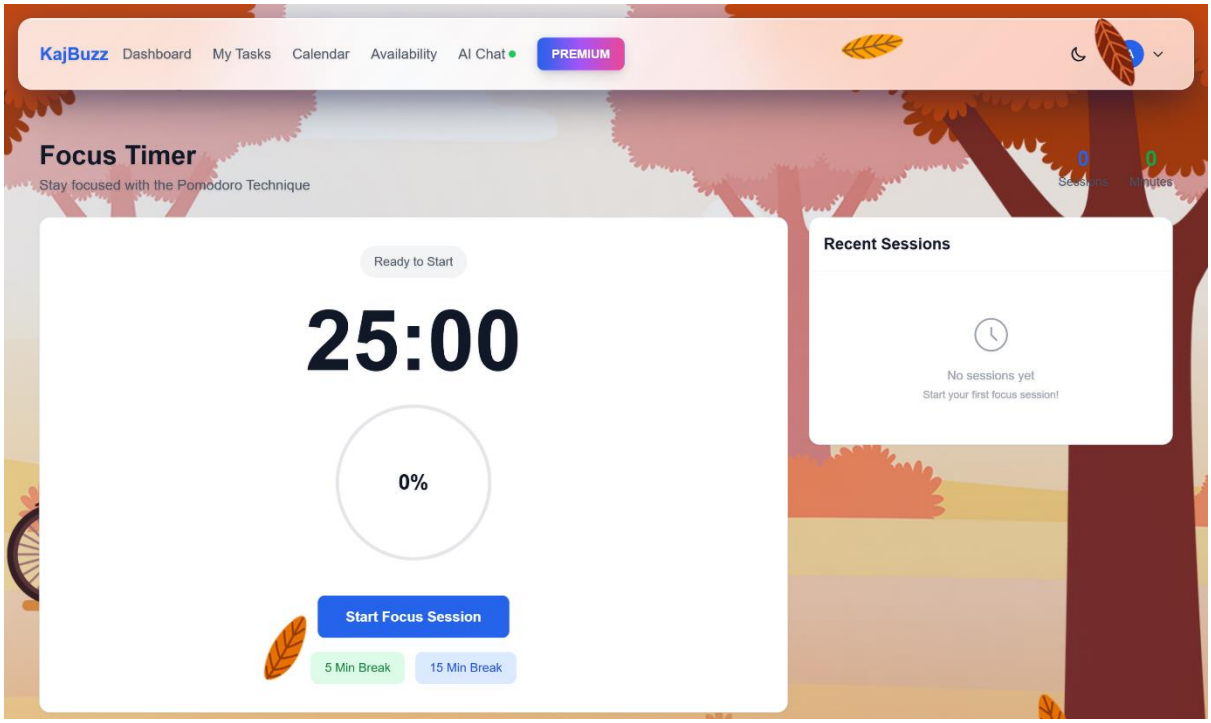
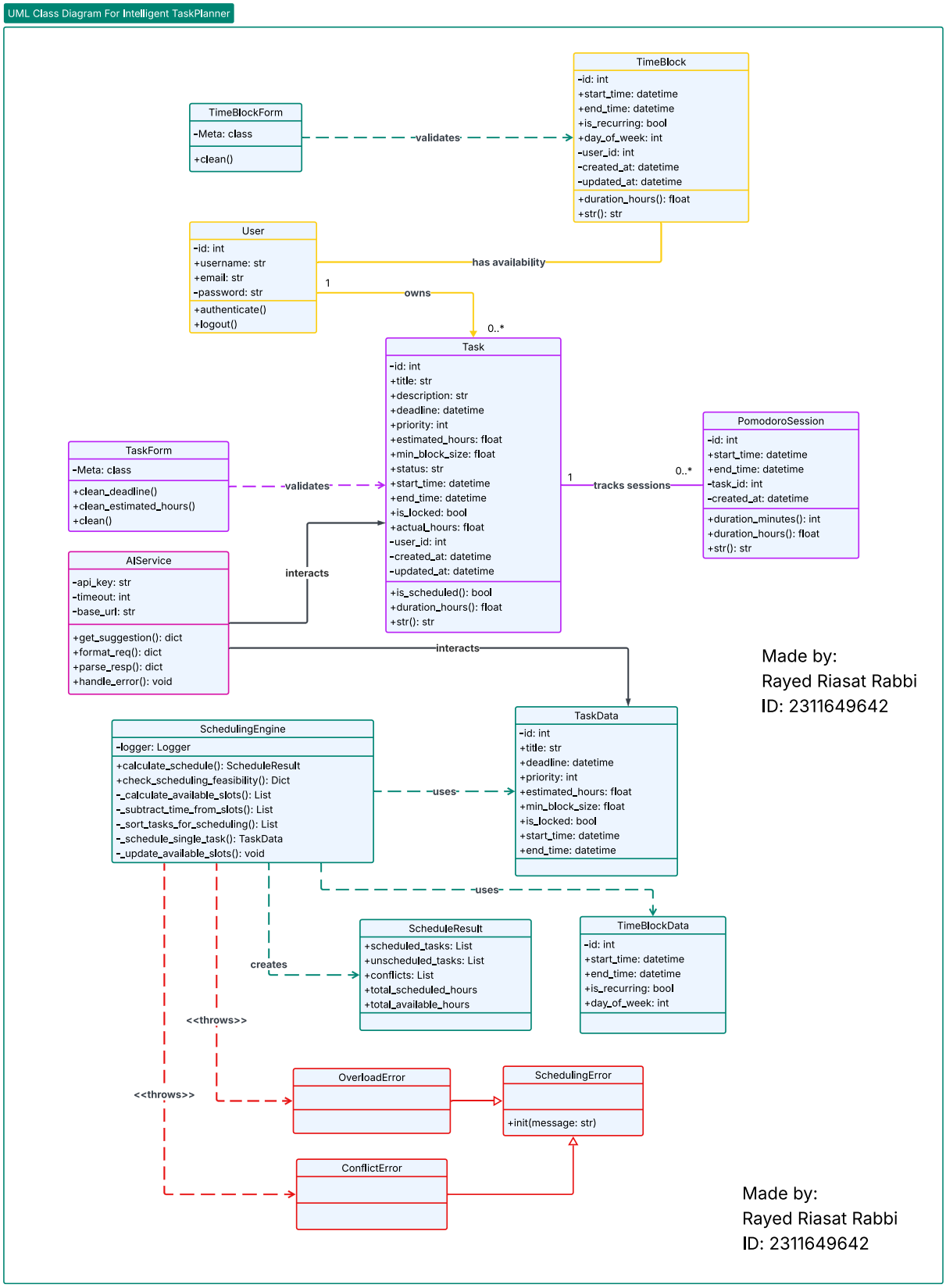


TABLE - 4: TECHNOLOGY STACK

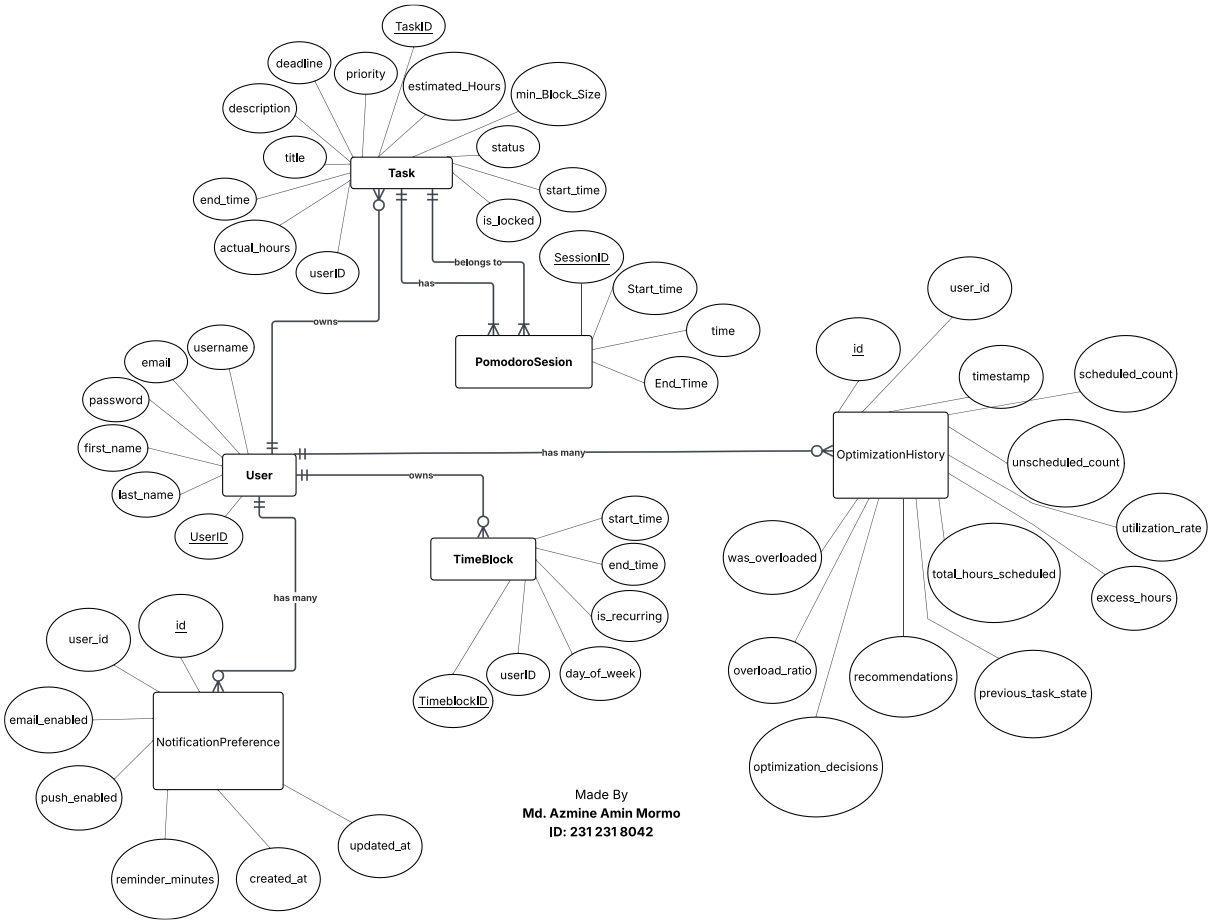
Layer	Stack
Frontend	Django Templates + TailwindCSS (via CDN) + HTMX for dynamic updates + AlpineJS for interactivity
Backend	Django 5 (Python 3.13.1) for server logic, authentication, and API handling
Package Manager	UV for fast dependency management
Database	MySQL for relational storage (users, tasks, schedules, etc.)
ORM	Django built-in ORM for MySQL
Auth	Django Allauth for authentication (email + optional Google OAuth)
Scheduler Engine	Python rule-based logic (greedy algorithms + constraints); optional ML via scikit-learn
Background Tasks	Django-Q
Email / SMS	Django email backend, Google SMTP
File Storage	Django's built-in storage
Deployment	PythonAnywhere for Django
Monitoring	Django logging

SYSTEM MODELLING DIAGRAMS

1. UML CLASS DIAGRAM



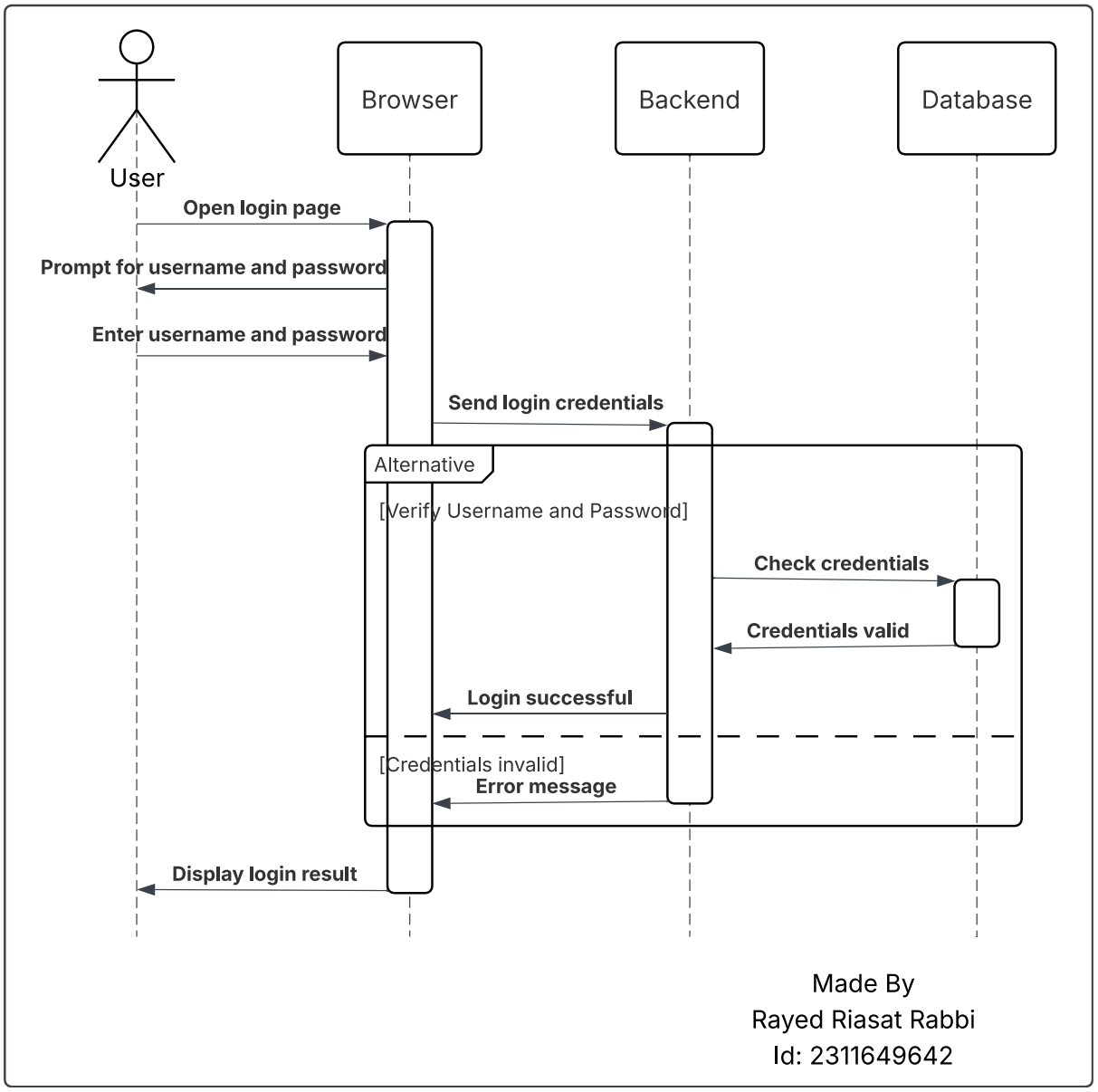
2. ER DIAGRAM


TaskID
deadline
priority
estimated_Hours
min_Block_Size
description
status
title
Task
start_time
end_time
is_locked
actual_hours
userID
SessionID
belongs to
has
Start_time
time
owns
PomodoroSesion
End_Time
username
email
password
first_name
last_name
User
UserID
has many
owns
TimeBlock
start_time
end_time
is_recurring
TimeblockID
userID
day_of_week
has many
user_id
id
email_enabled
NotificationPreference
push_enabled
reminder_minutes
created_at
updated_at
Made By
Md. Azmine Amin Mormo
ID: 231 231 8042
user_id
id
timestamp
scheduled_count
OptimizationHistory
unscheduled_count
utilization_rate
was_overloaded
total_hours_scheduled
excess_hours
overload_ratio
recommendations
previous_task_state
optimization_decisions

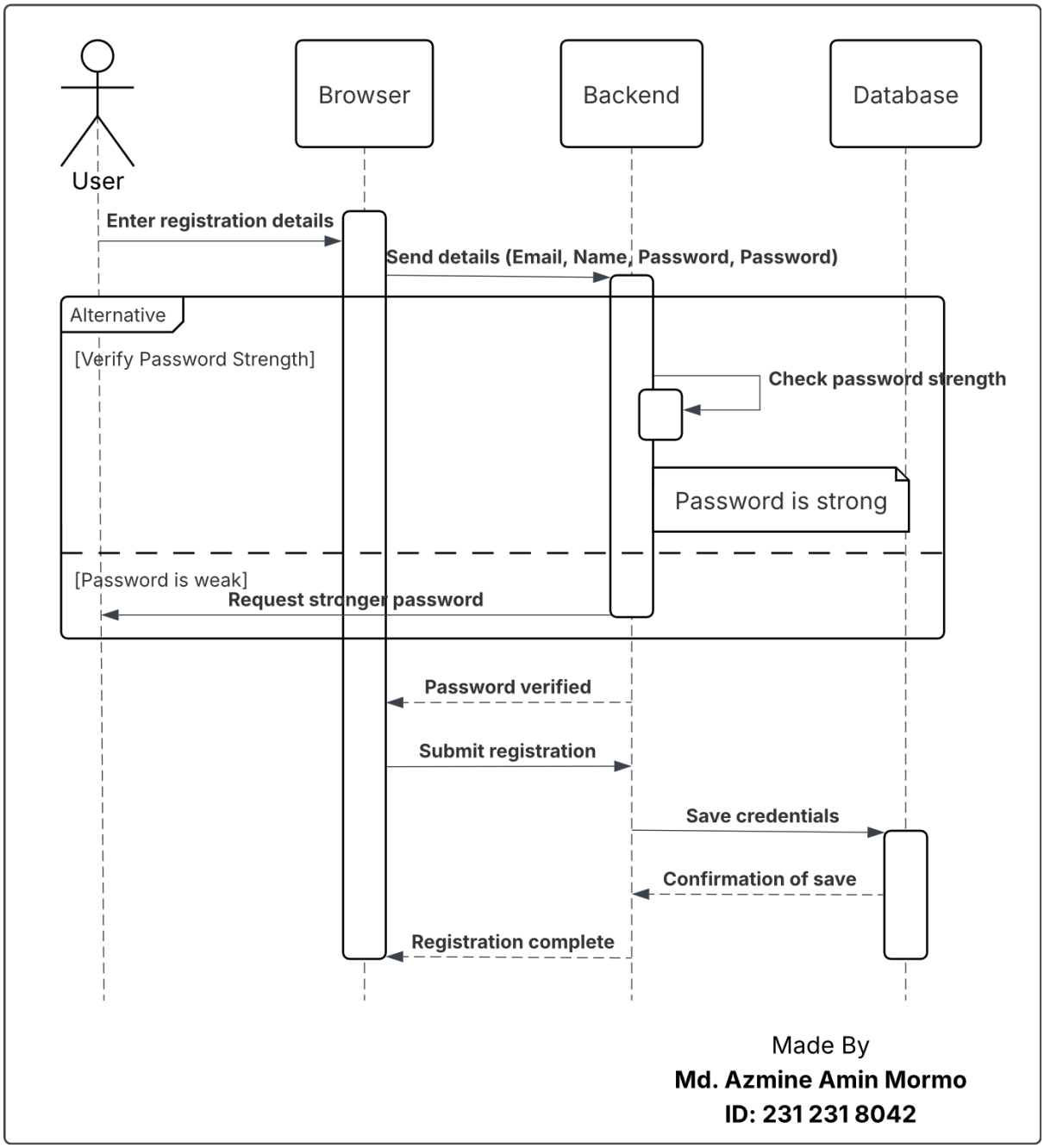
3. SEQUENCE DIAGRAMS

3.1: User Login Sequence Diagram

Squence Diagram for User Login Page

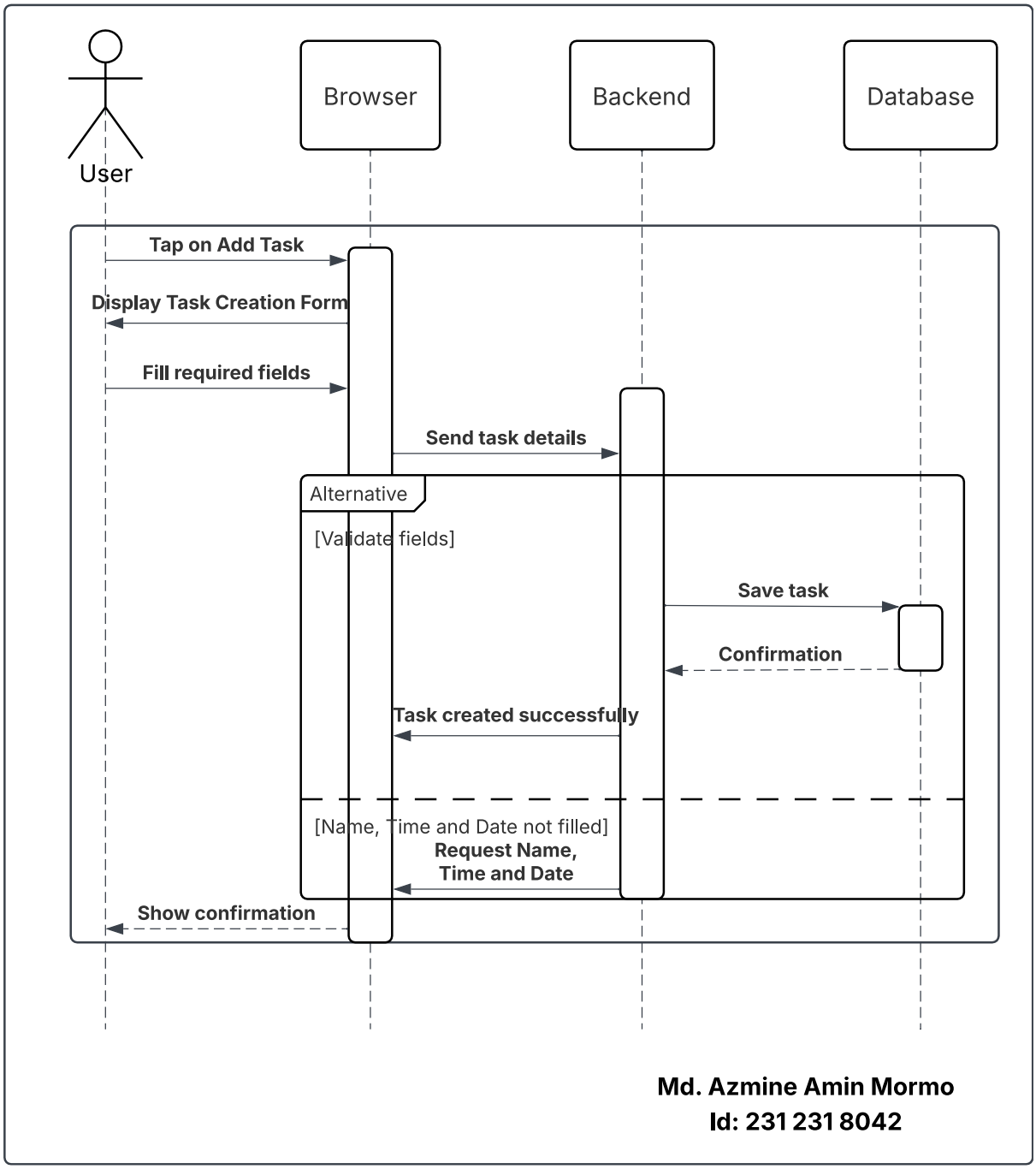


3.2: User Registration Sequence Diagram

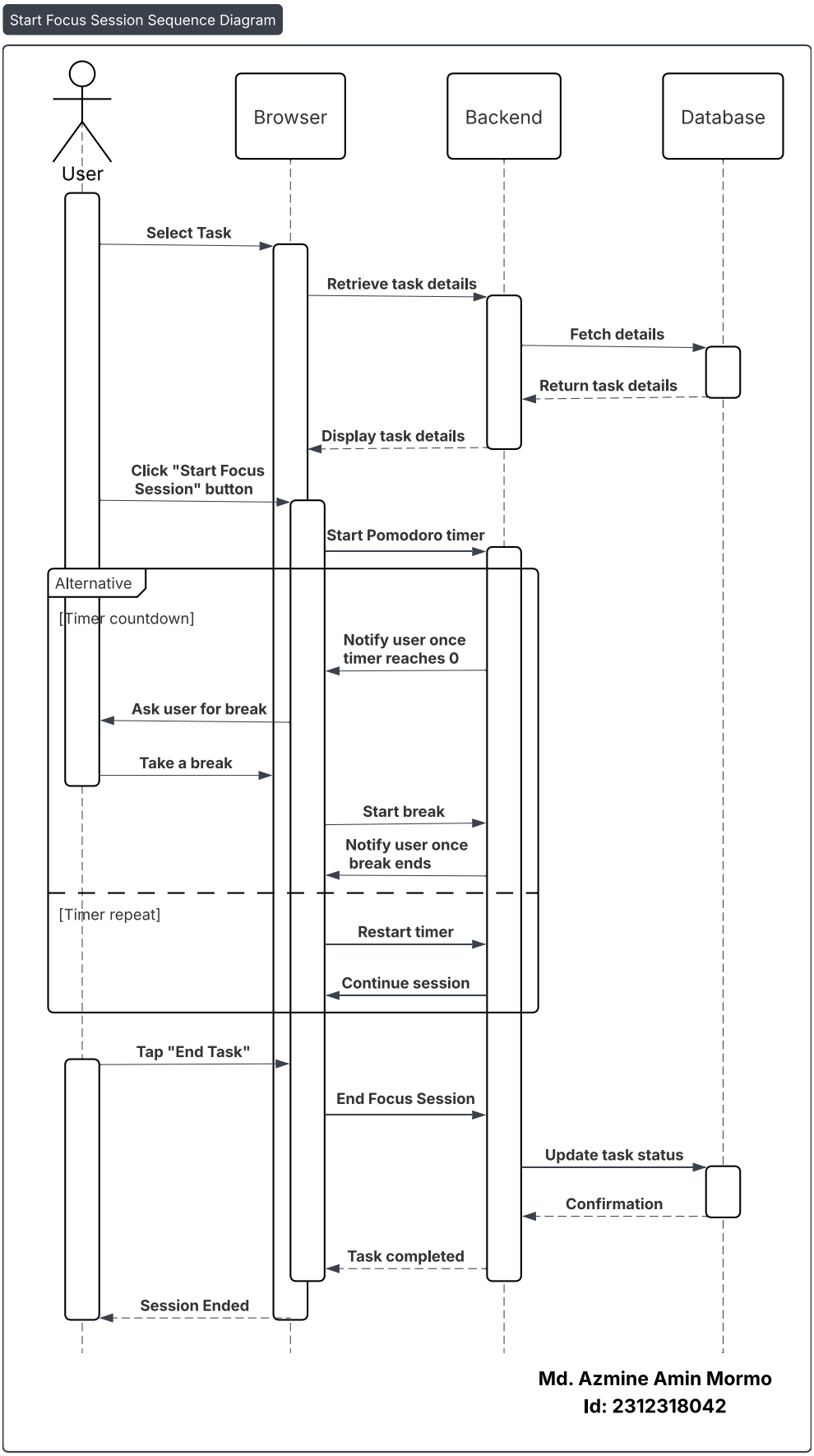


3.3: Create Task Sequence Diagram

Create Task Sequence Diagram

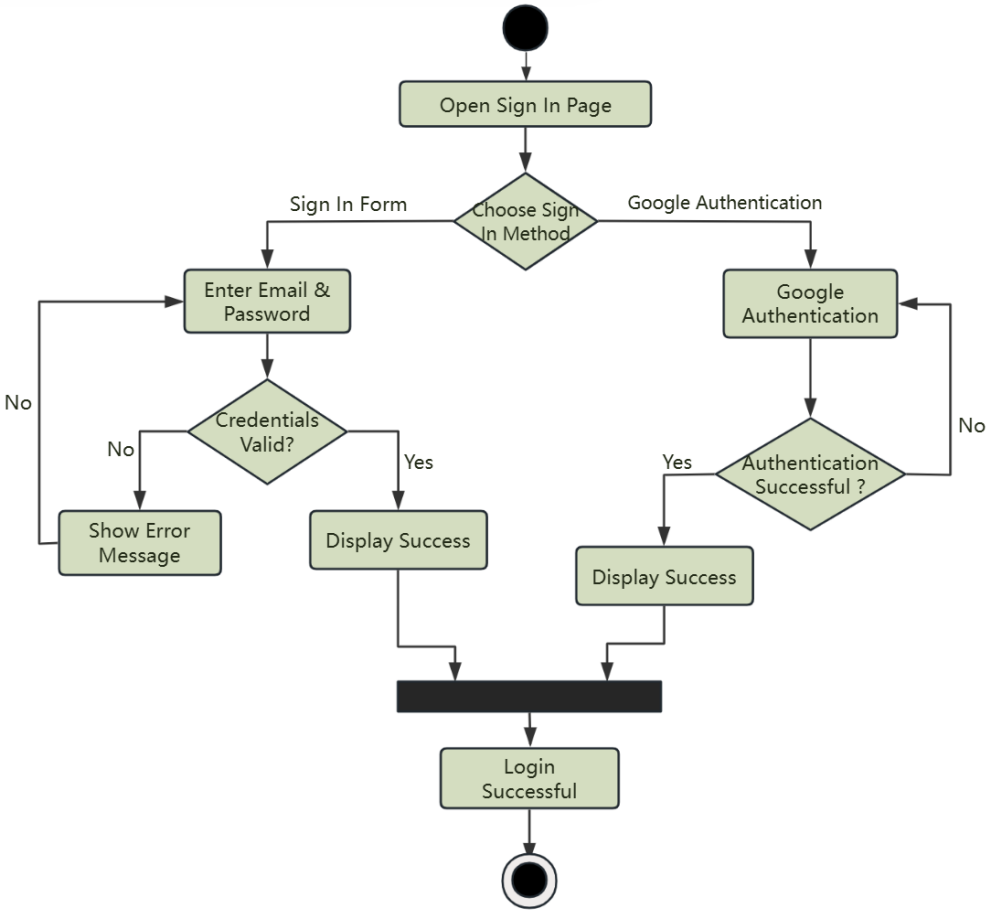


3.4: Start Focus Session Sequence Diagram



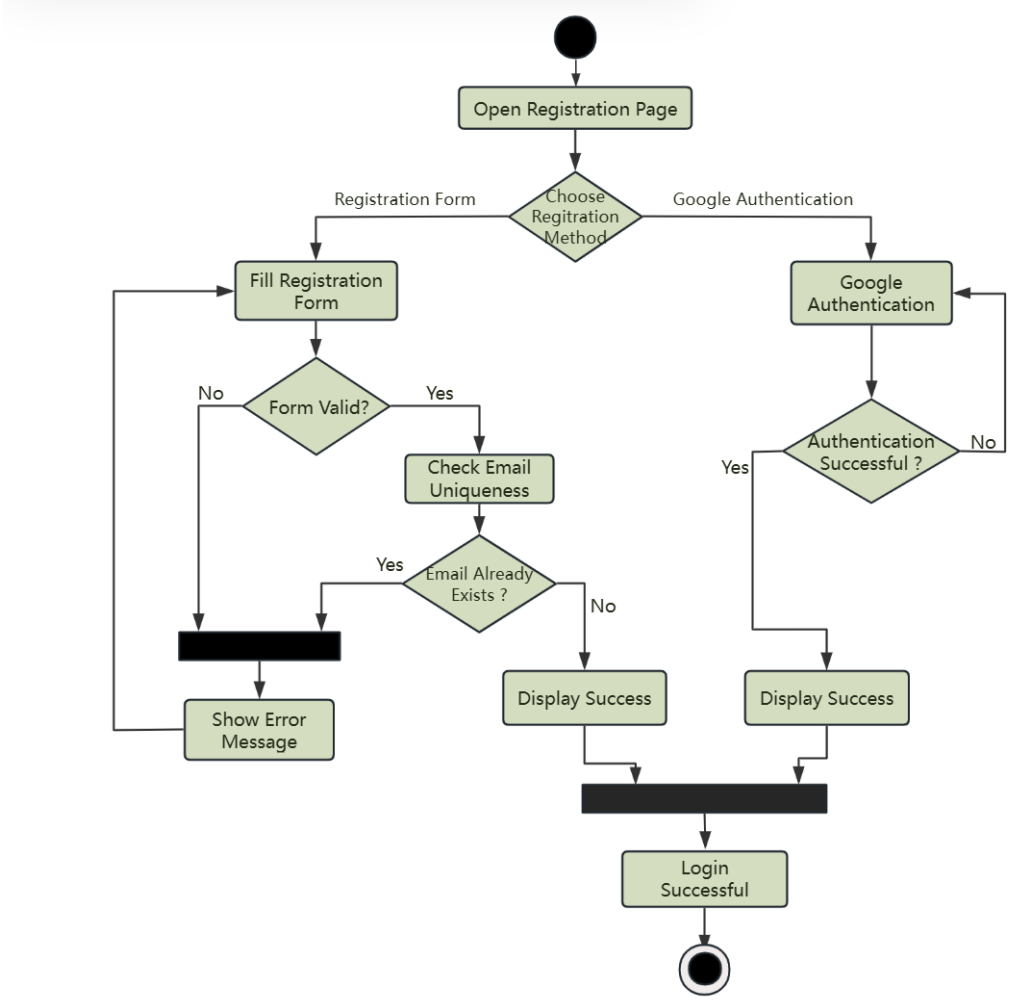
4. ACTIVITY DIAGRAMS

4.1: User Login



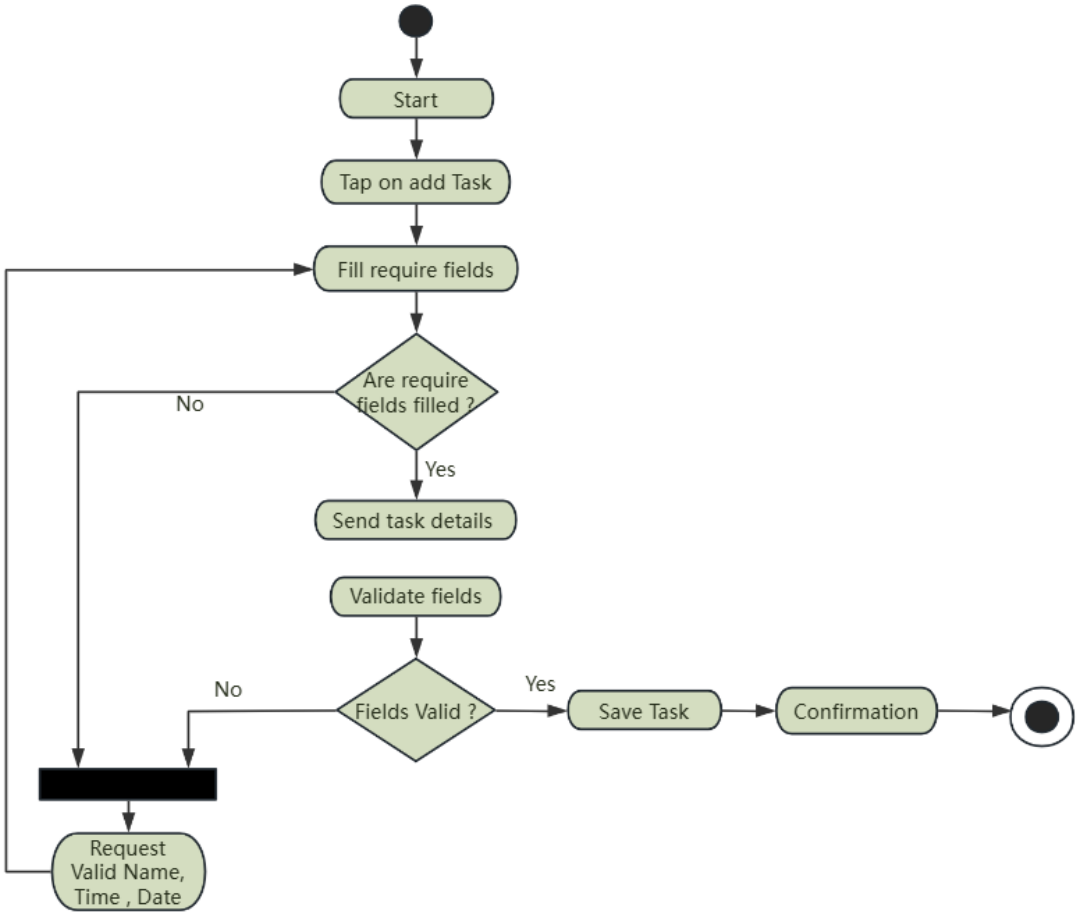
Made By,
Name: Barshon Basak
ID: 2311625042

4.2: User Registration



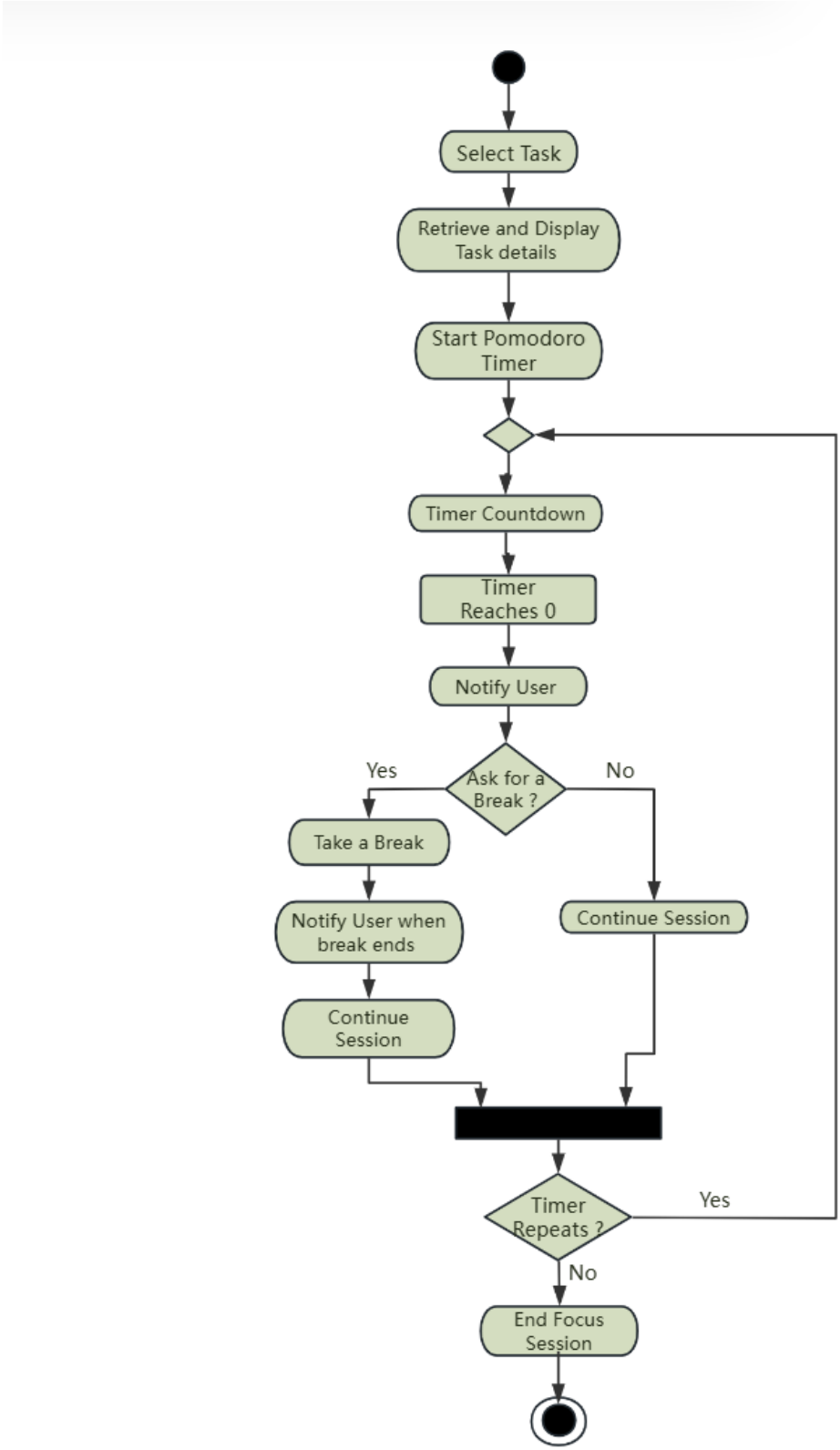
Made By,
Name: Barshon Basak
ID: 2311625042

4.3: Create Task



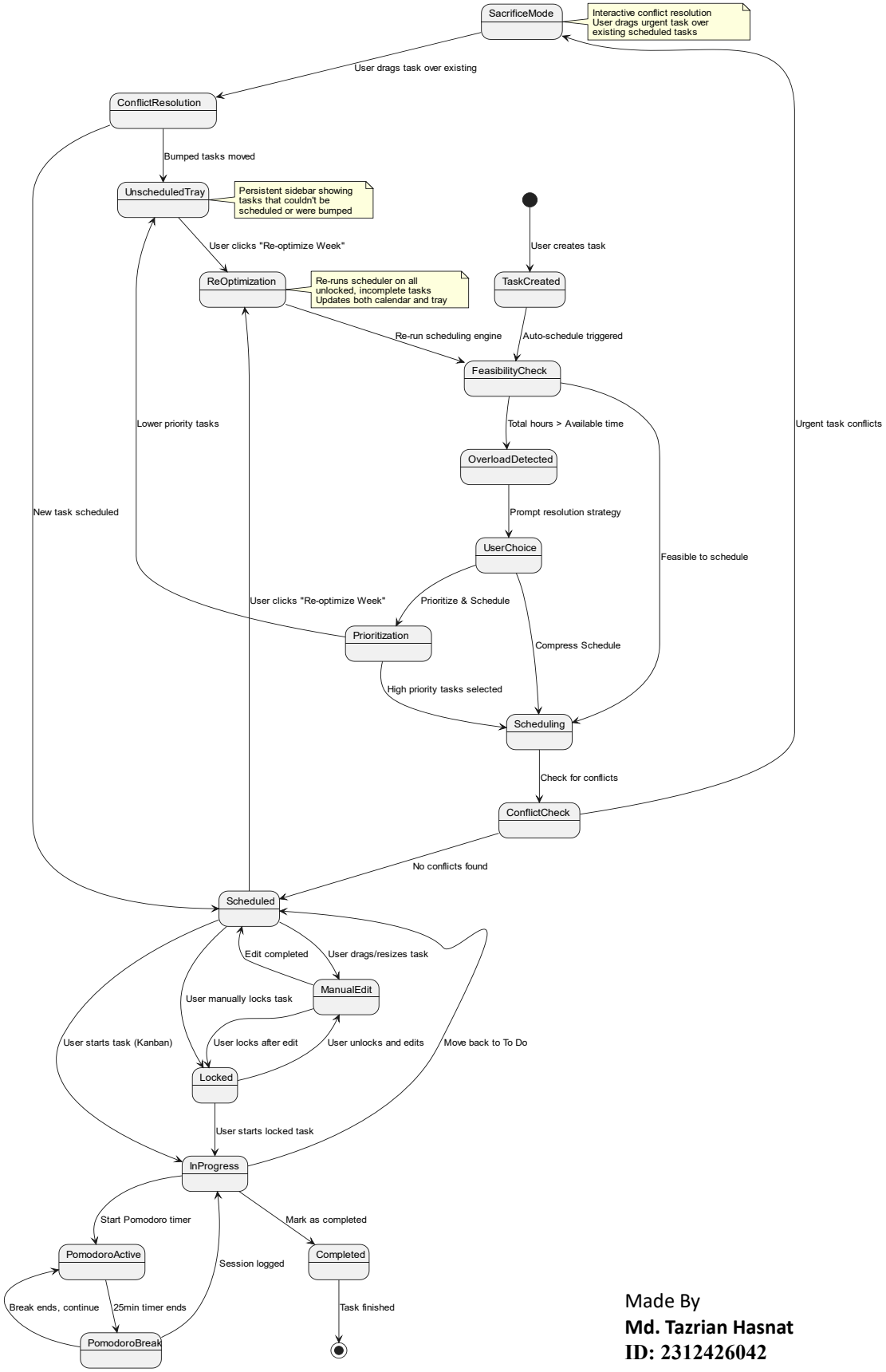
Made By,
Name: Barshon Basak
ID: 2311625042

4.4: Start Focus Session



Made By,
Name: Barshon Basak
ID: 2311625042

5. STATE MACHINE DIAGRAM



Made By
Md. Tazrian Hasnat
ID: 2312426042

USER ACCESS CONTROL

Admin

- Full backend & database access (view, edit, delete any user or data).
- Site hosting, deployment, and system-level integrations.

Premium User

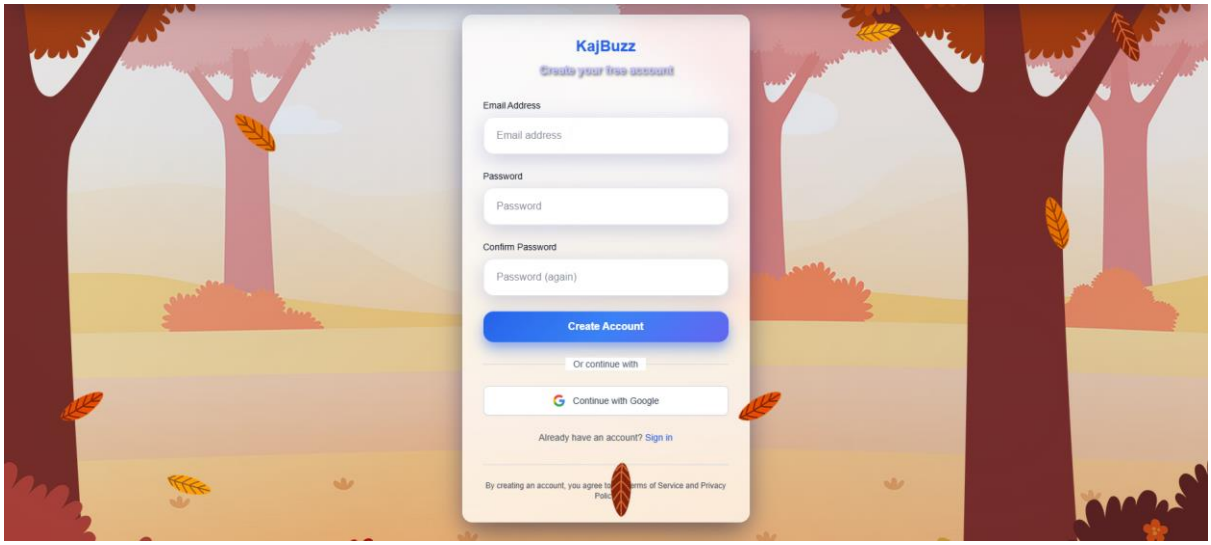
- All standard features PLUS:
  - Pomodoro Timer / Focus Session
  - AI Chatbot for scheduling
  - AI Suggestions in Calendar
  - AI-driven Re-optimization

Regular User

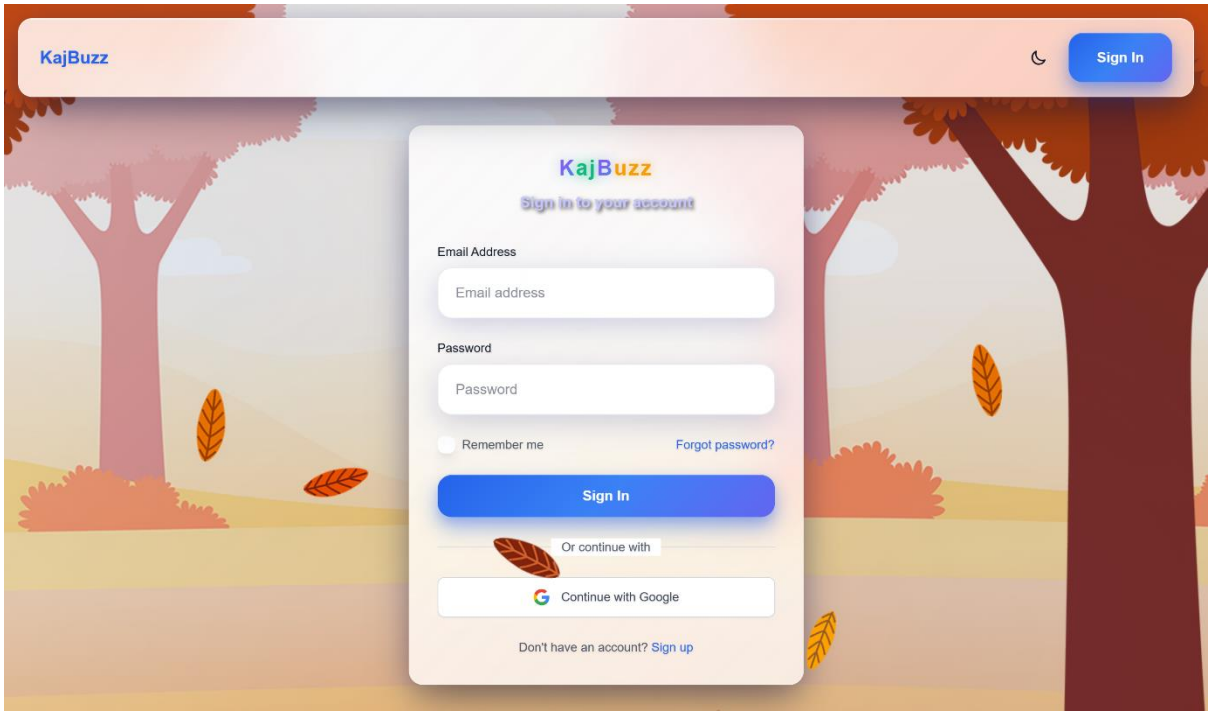
- Every feature except Pomodoro Timer and AI functions.

FUNCTIONALIY-WISE SYSTEM DESCRIPTION

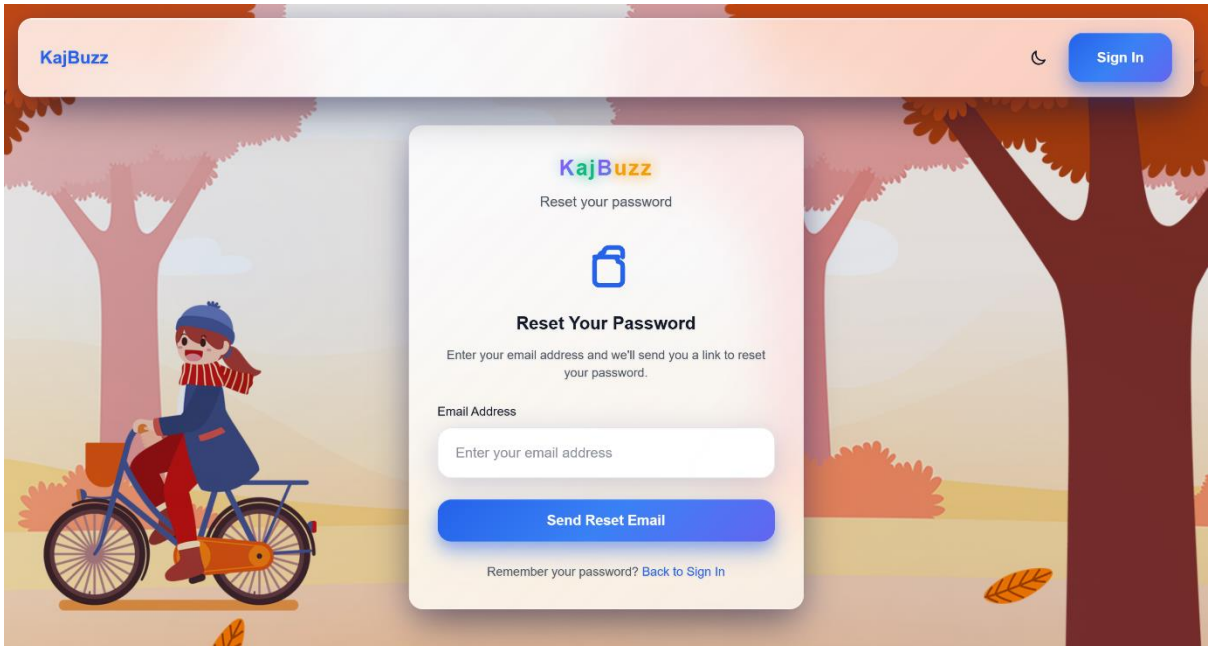
1. User Signup: New users can create an account by giving credentials input or by simply signup with google account.

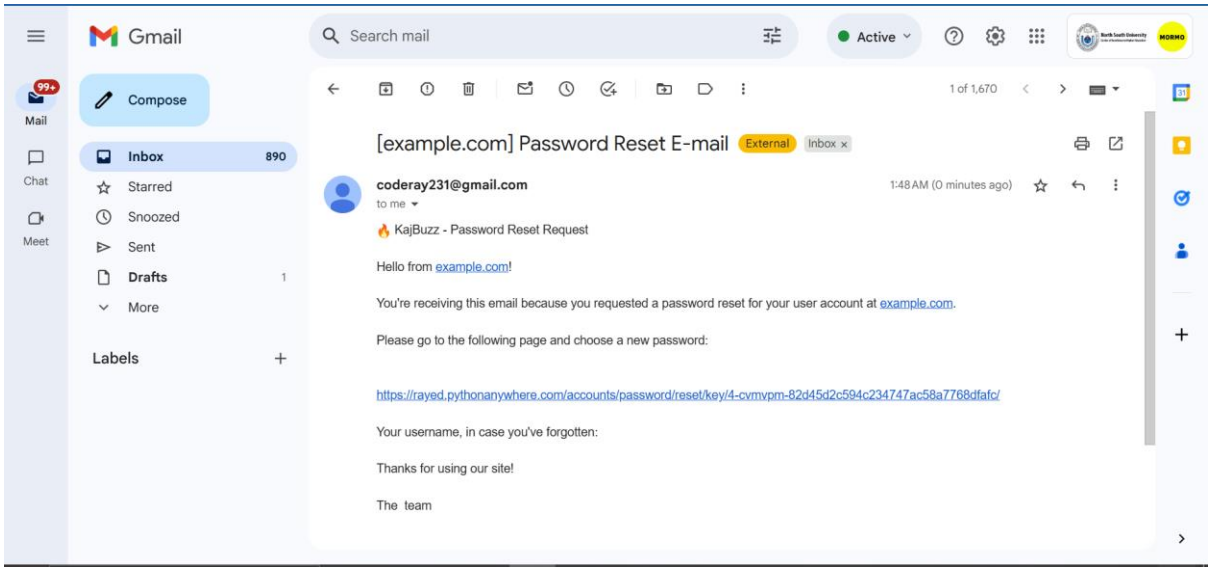


2. User Sign in: Users can login by giving credentials input or by simply sign in with google account.

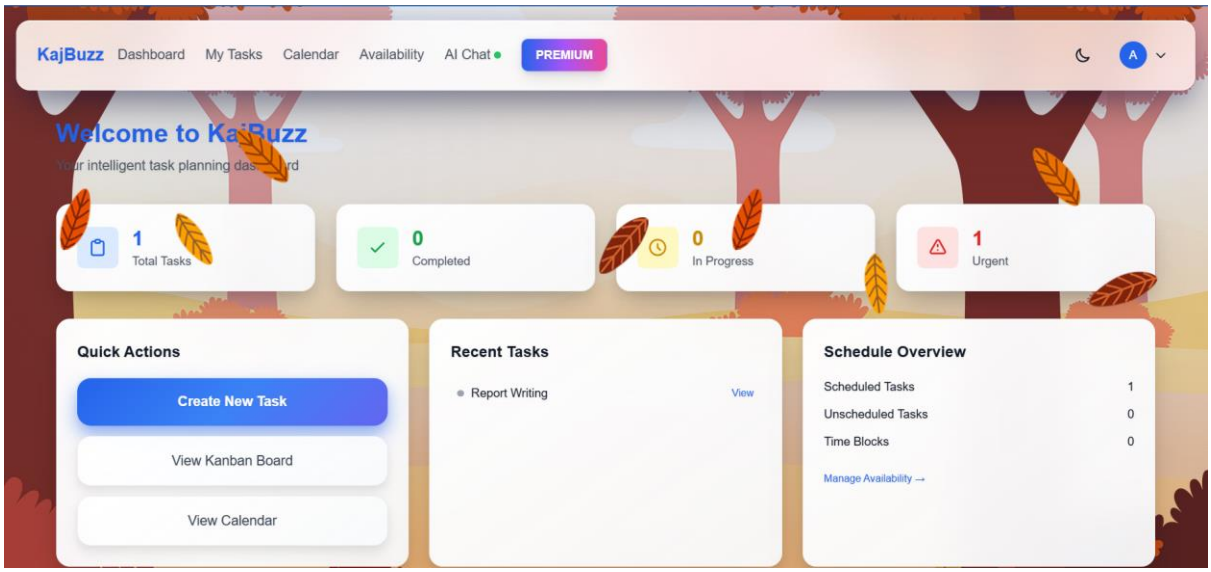


3. Forgot Password Through Directed Link Email Notification: Users who forgot their password, can reset their password by having a directed link via email and reset it by clicking that.

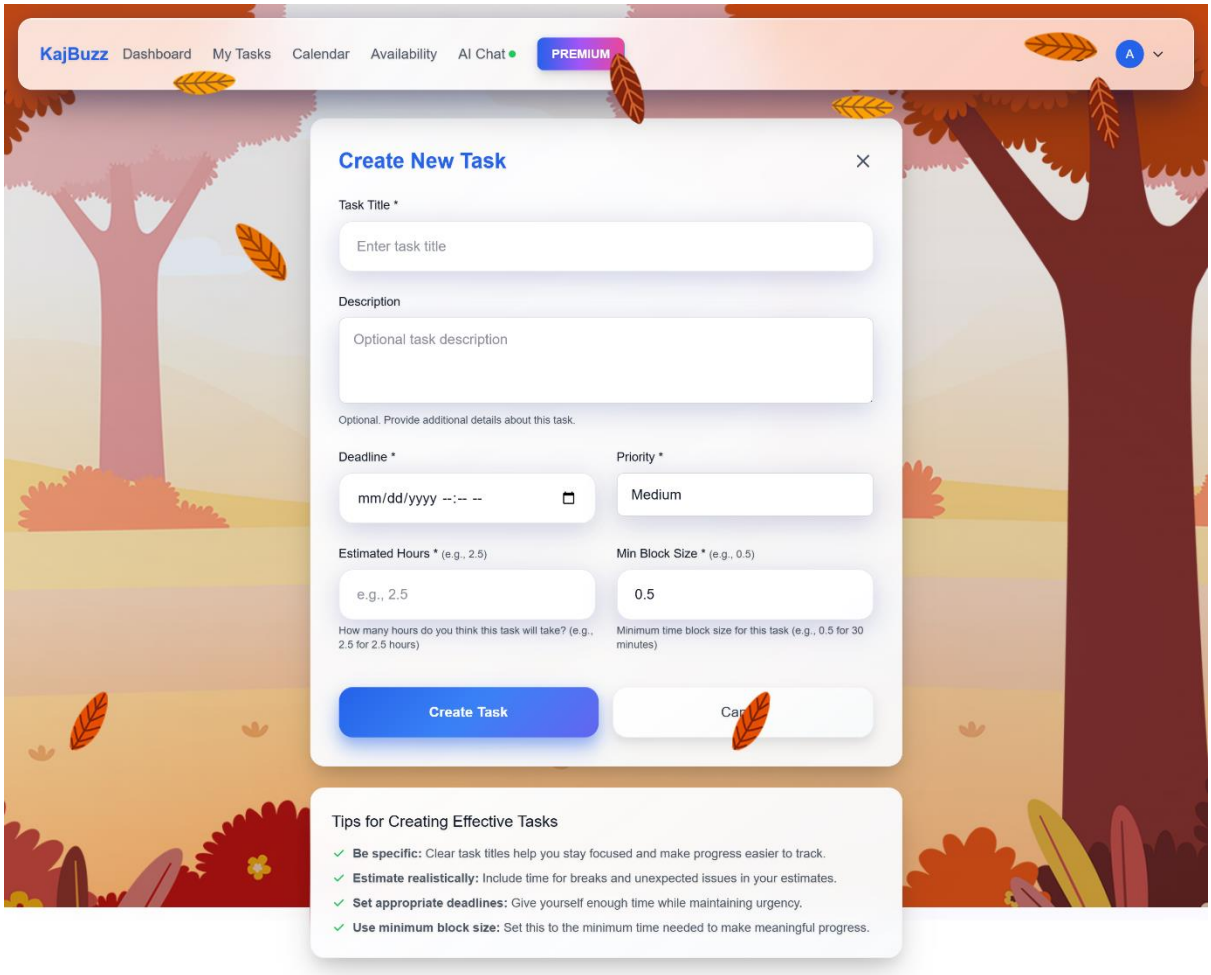




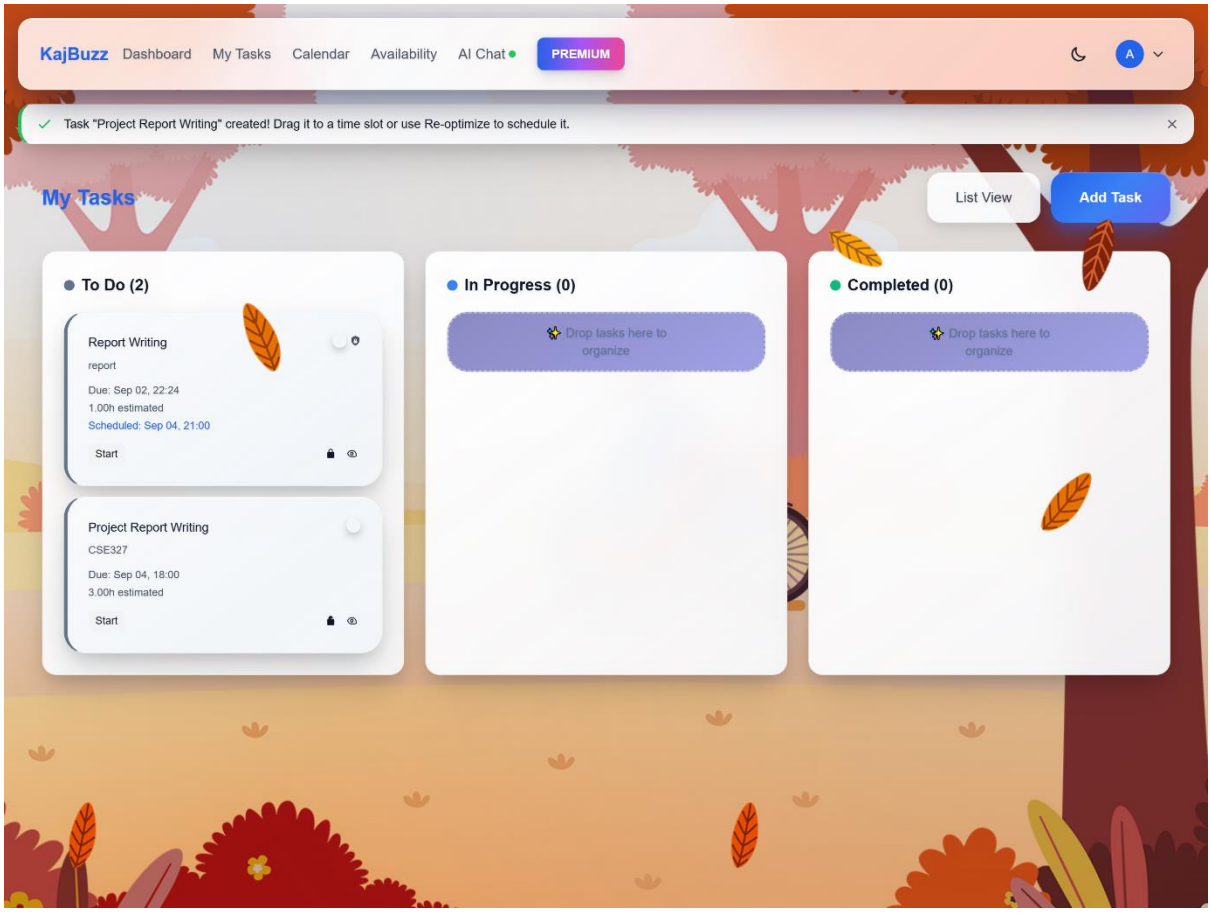
4. Dashboard: Users can view all their tasks and their progress.



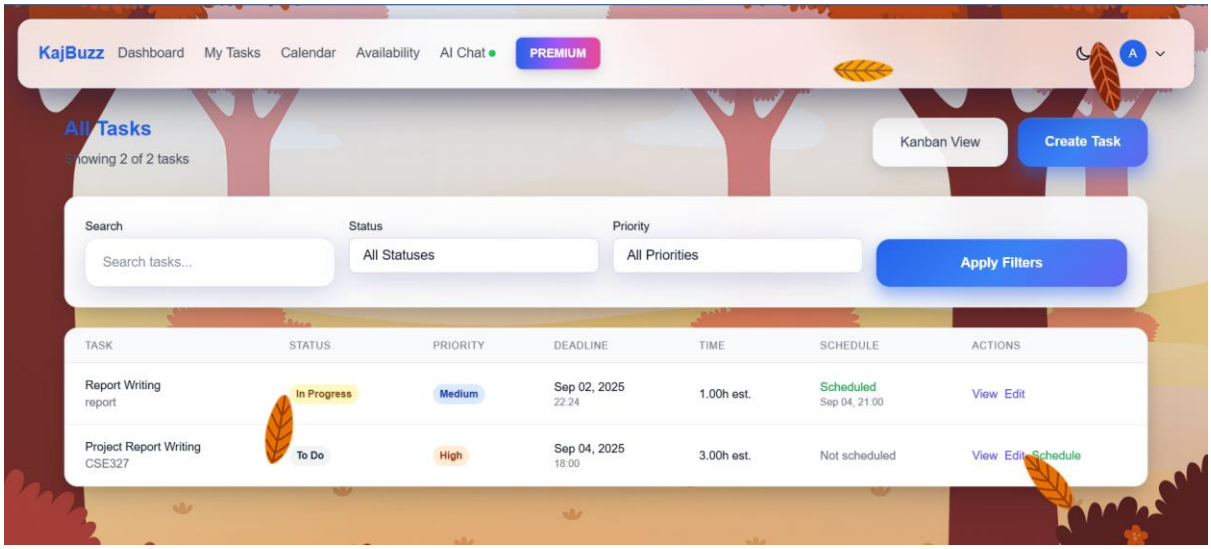
5. Create New Task: Users can create tasks by giving some information related to their tasks and the urgency level.



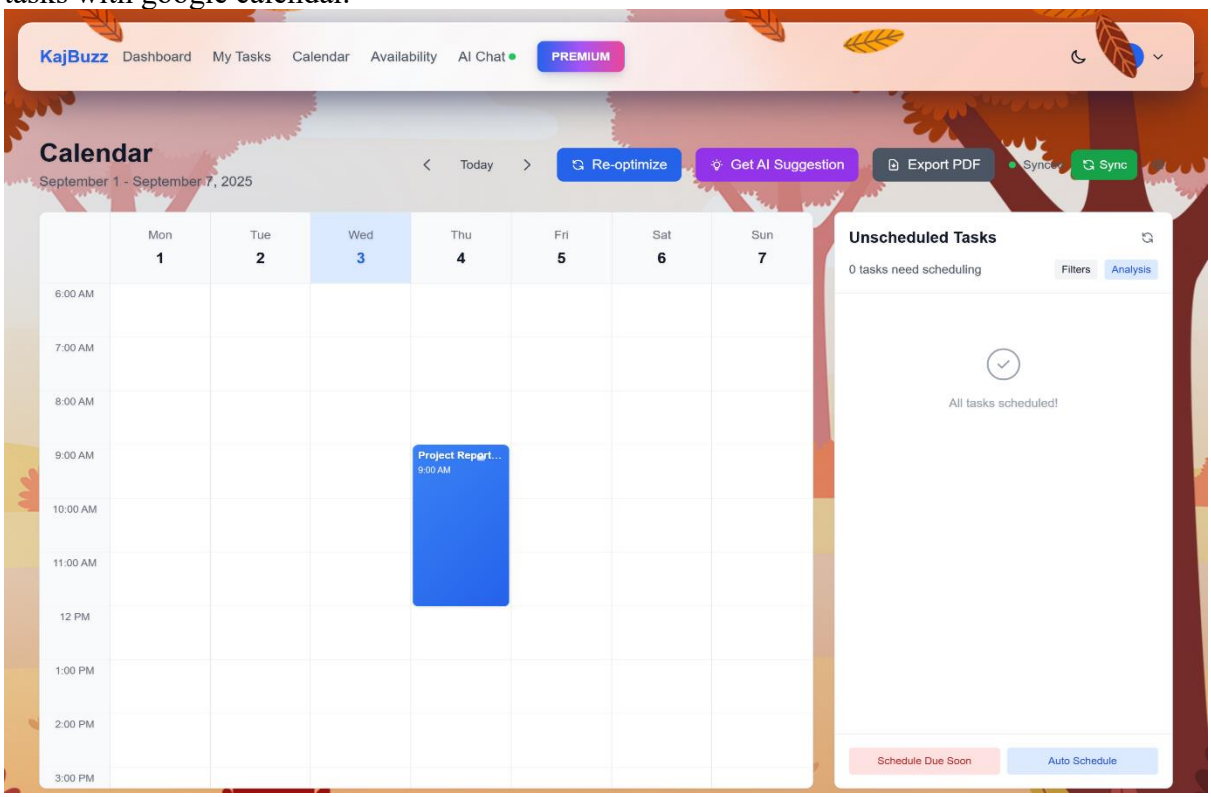
6. Kanban View of My Task: Users can see the overview of their previous, running and upcoming tasks.



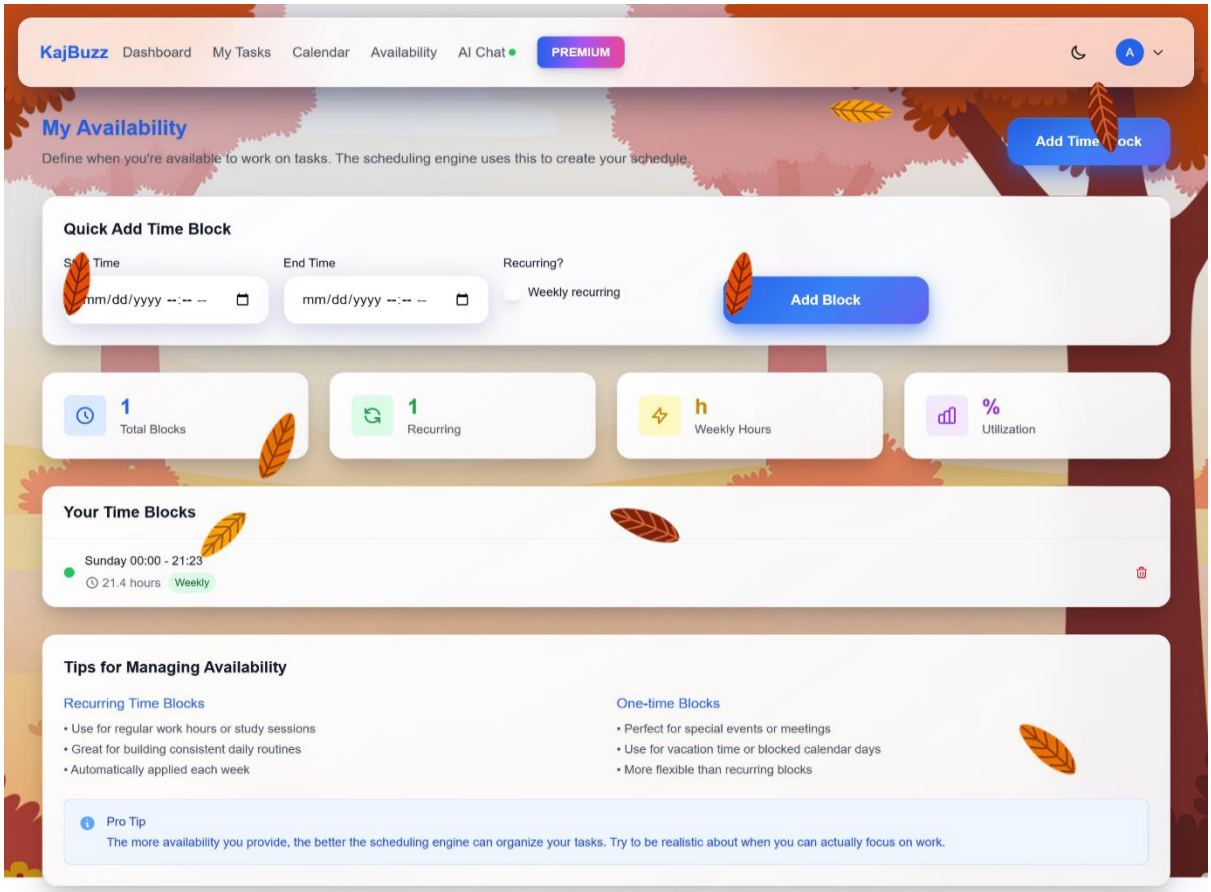
7. List View of My Tasks:



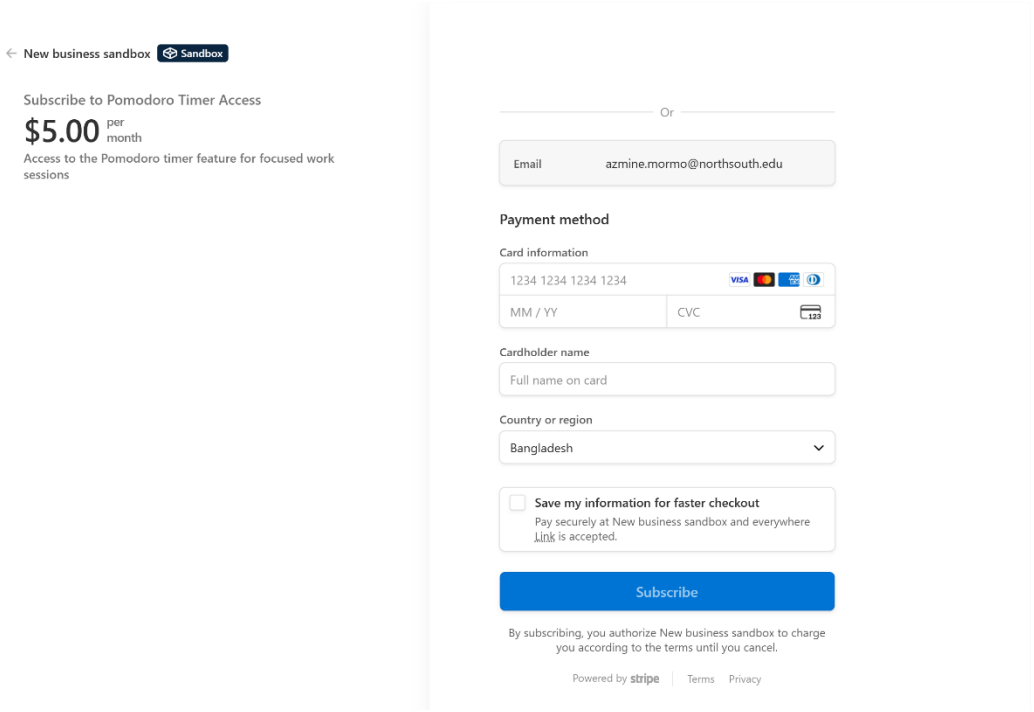
8. Google Calendar Integration: Users who have google account connected, can sync their tasks with google calendar.

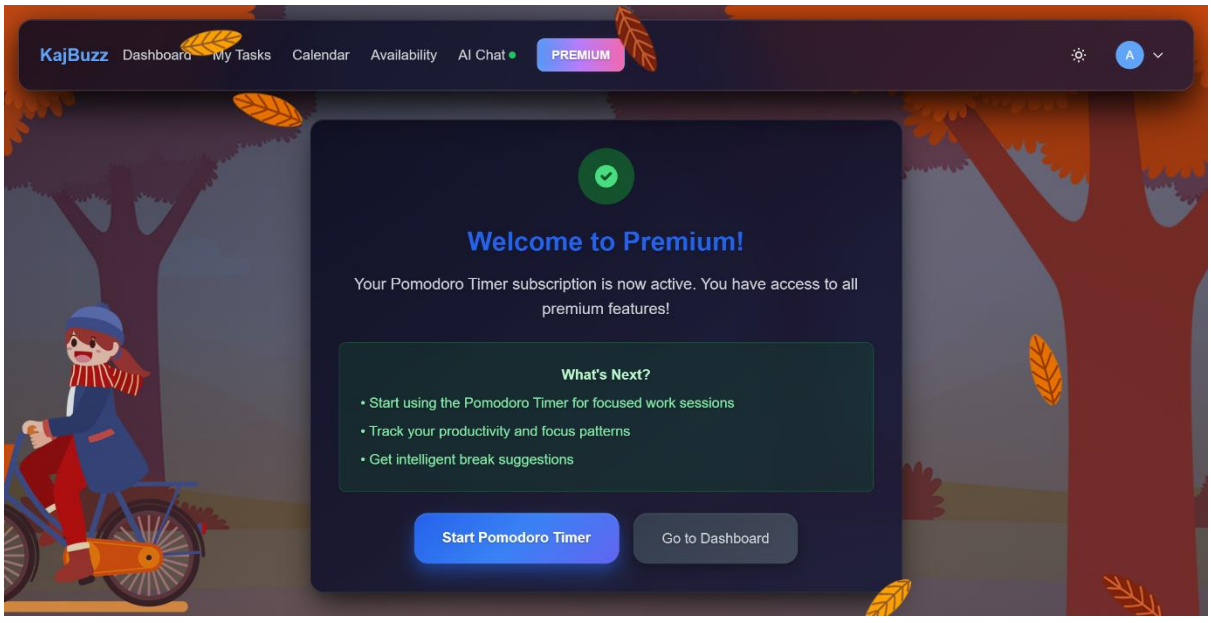


9. Availability: Users can define their available time to work.

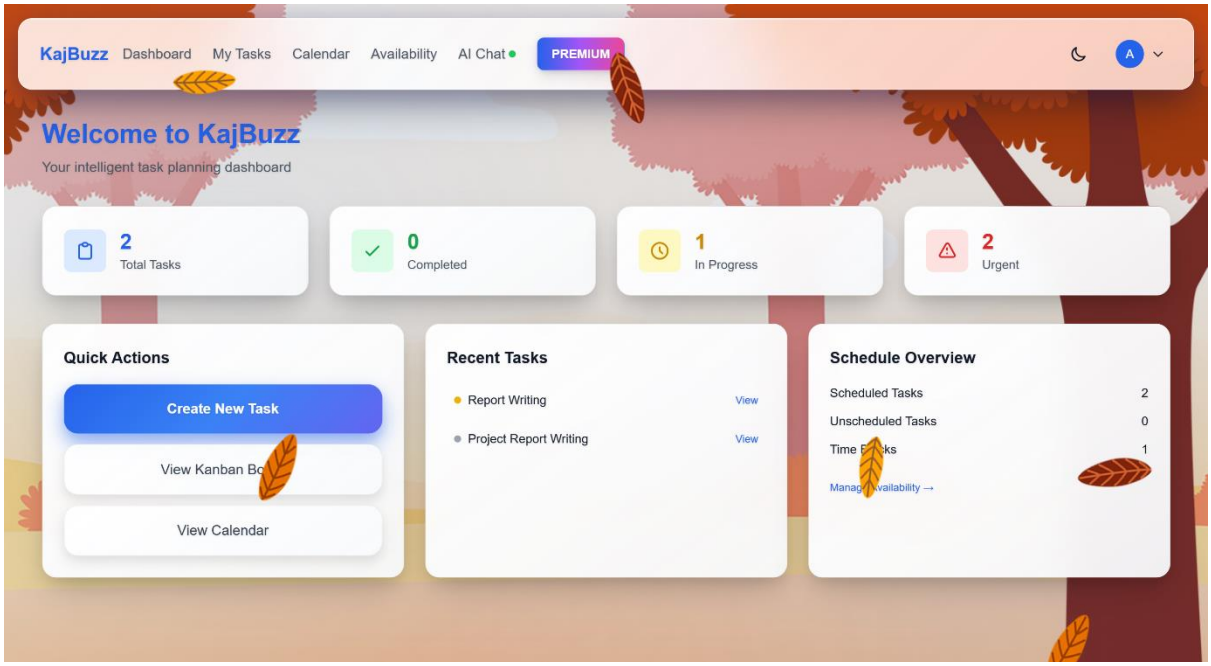


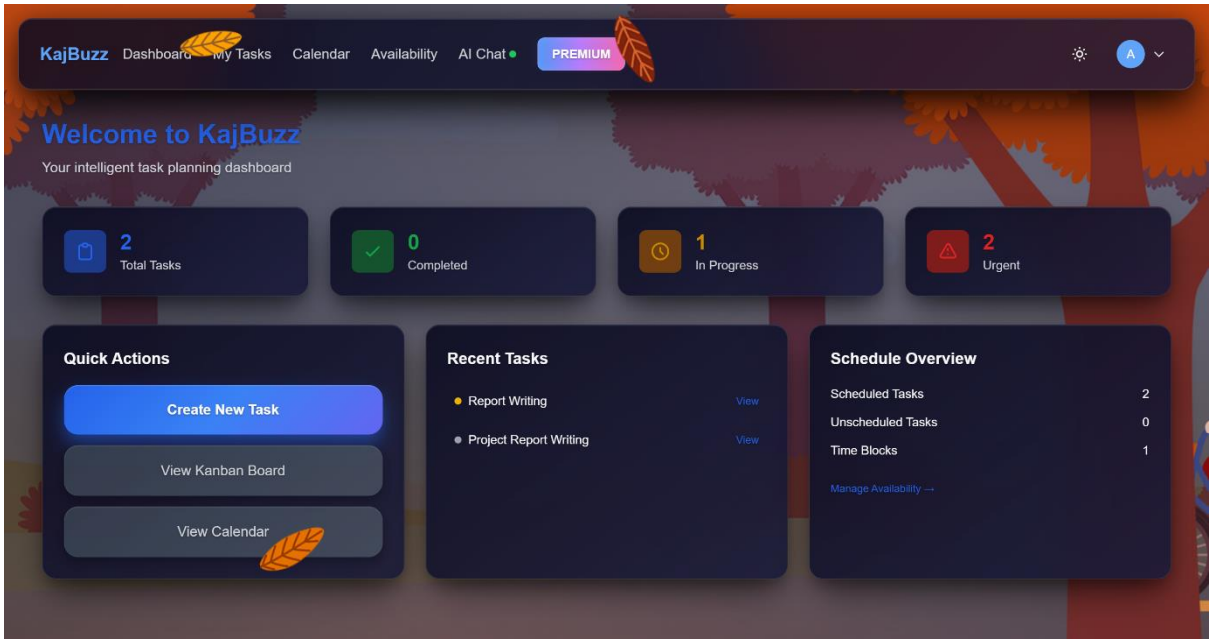
10. Buy Subscriptions: Users can buy subscriptions (Stripe Subscription Service) of Pomodoro timer and AI assistant.



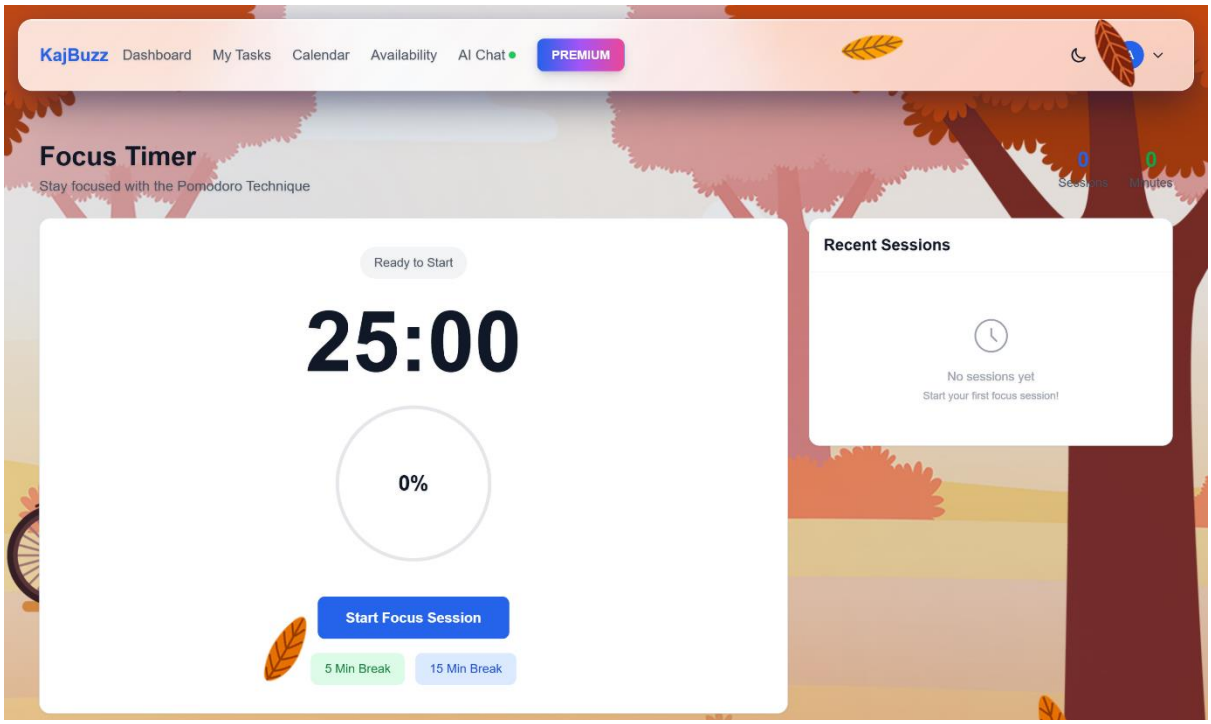


11. Dark and Light Theme: Users can select their preferred theme.

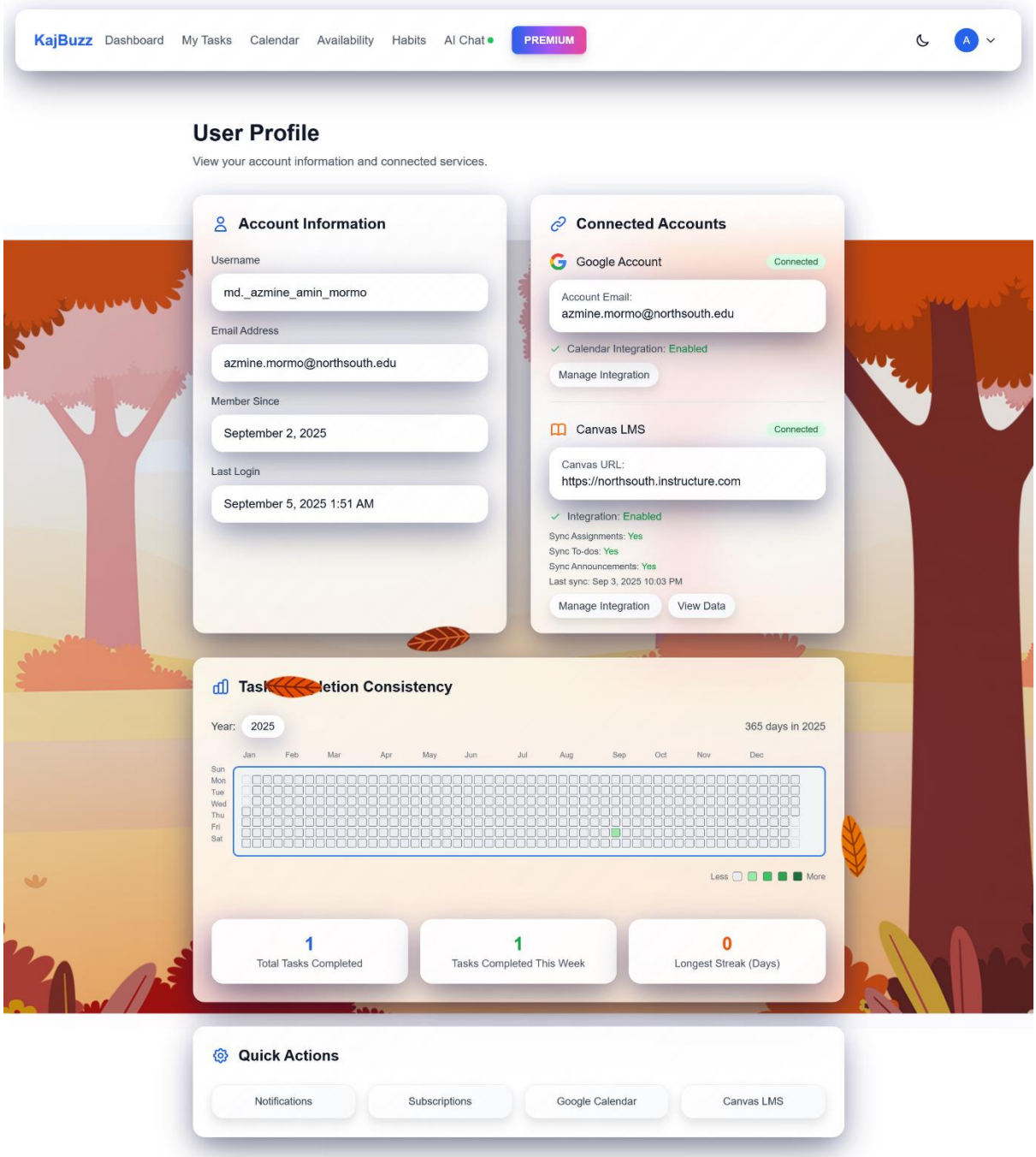




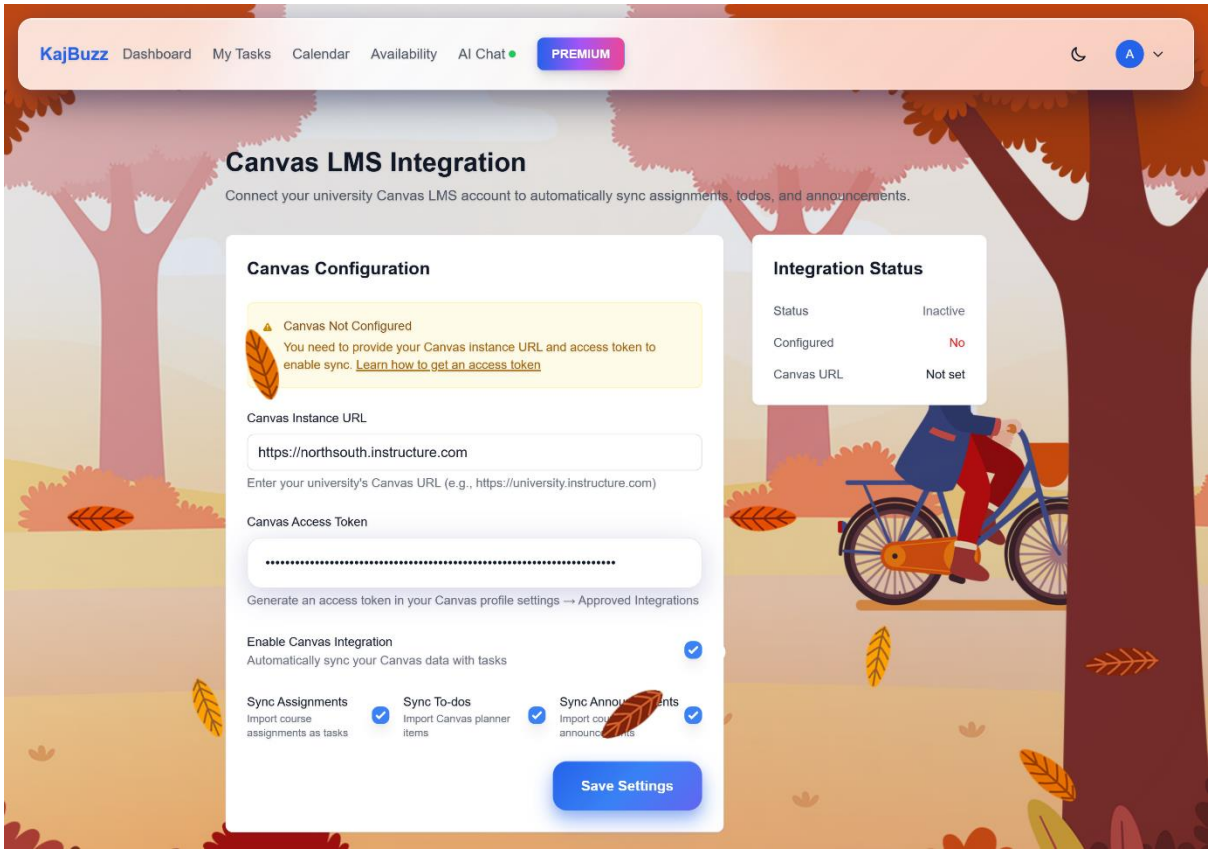
12. Pomodoro Timer (Focus Session): User can start a pomodoro timer to focus on their work.



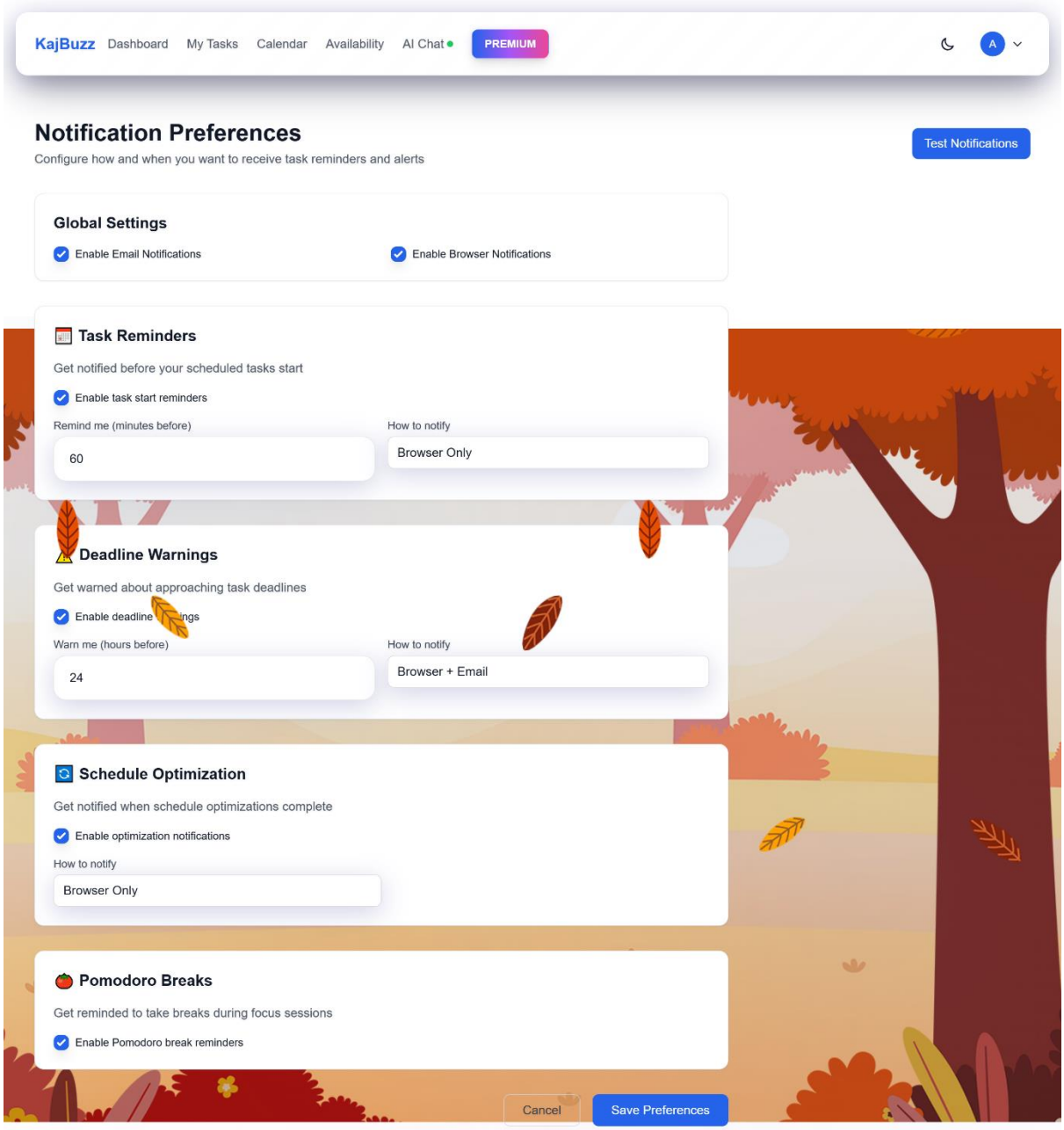
13. User Profile Details and Modification:

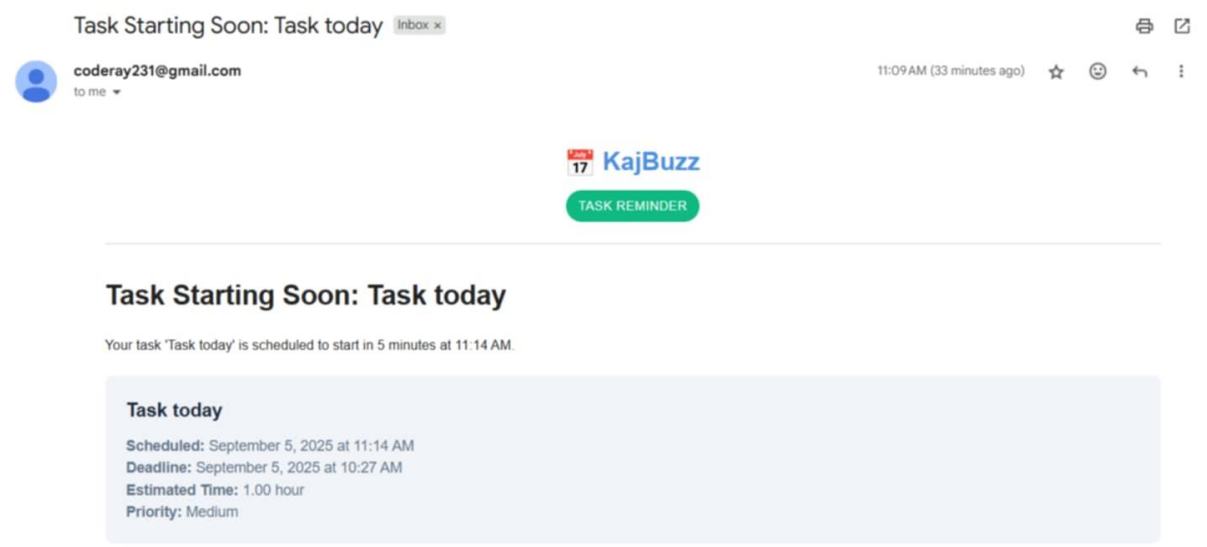


14. Canvas LMS Integration: Users can integrate their Canvas LMS account to KajBuzz to automatically sync assignments, to-do's, and announcements.

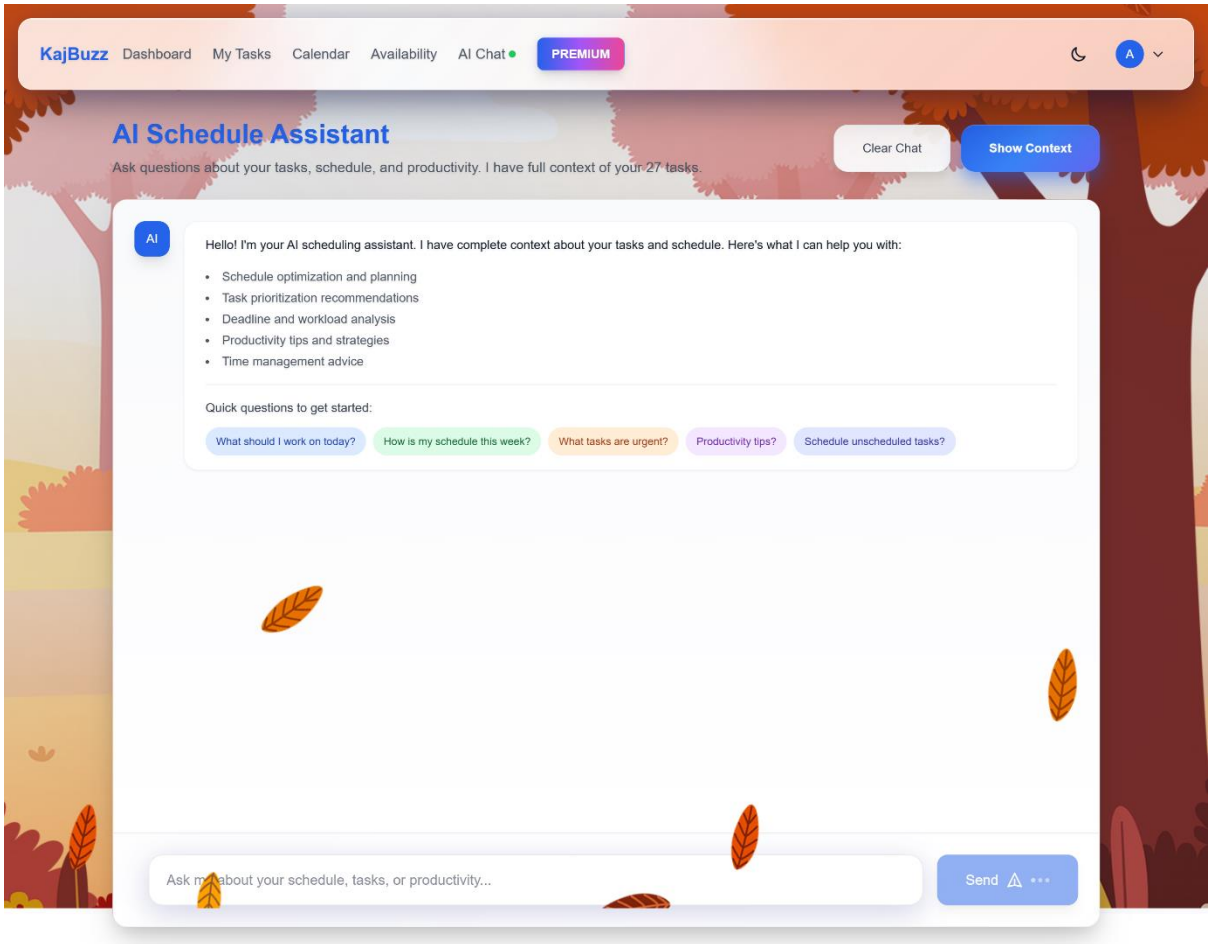


15. Notification: Users can set their notification preferences and the system will generate email notification and/or browser notifications.

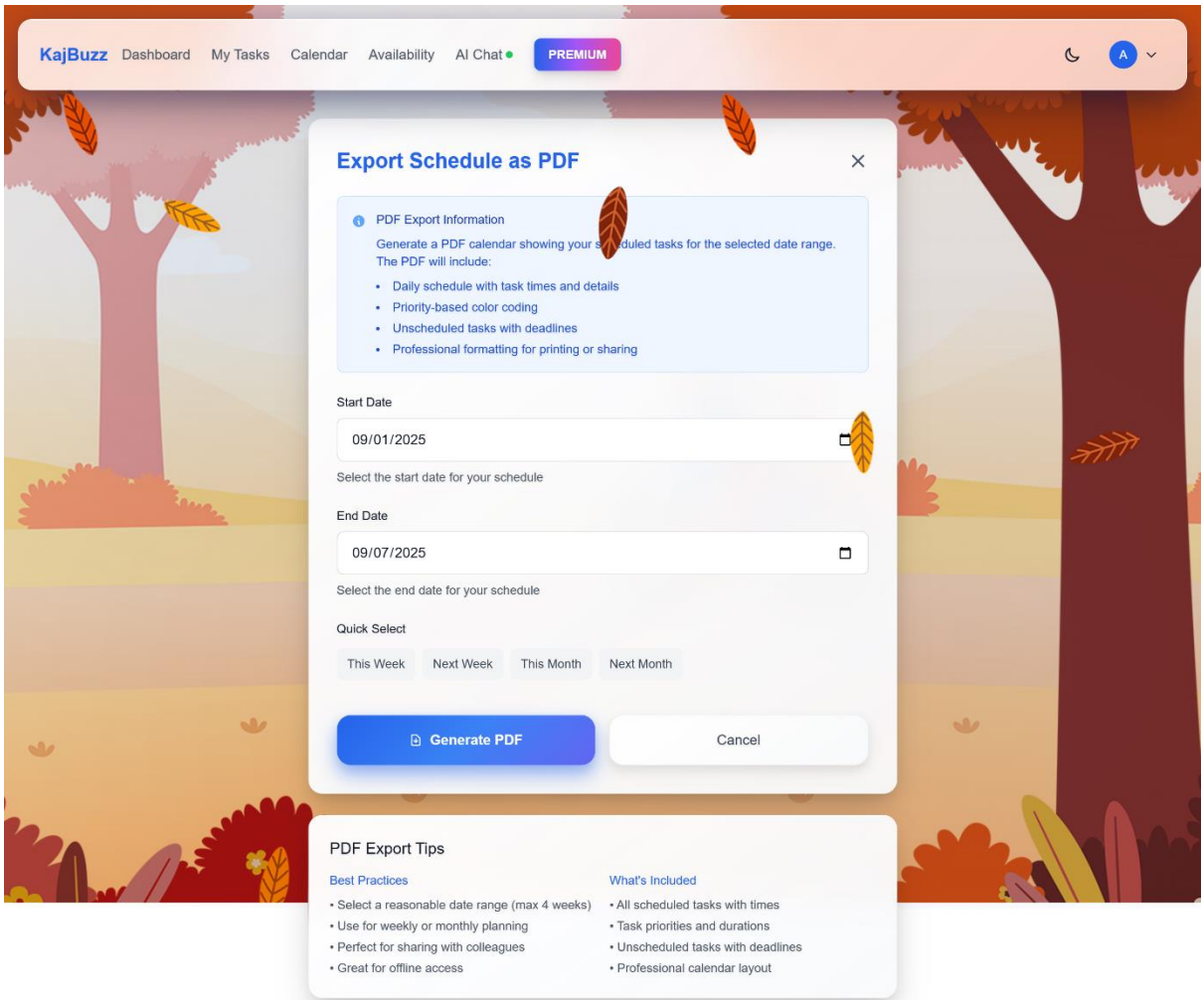




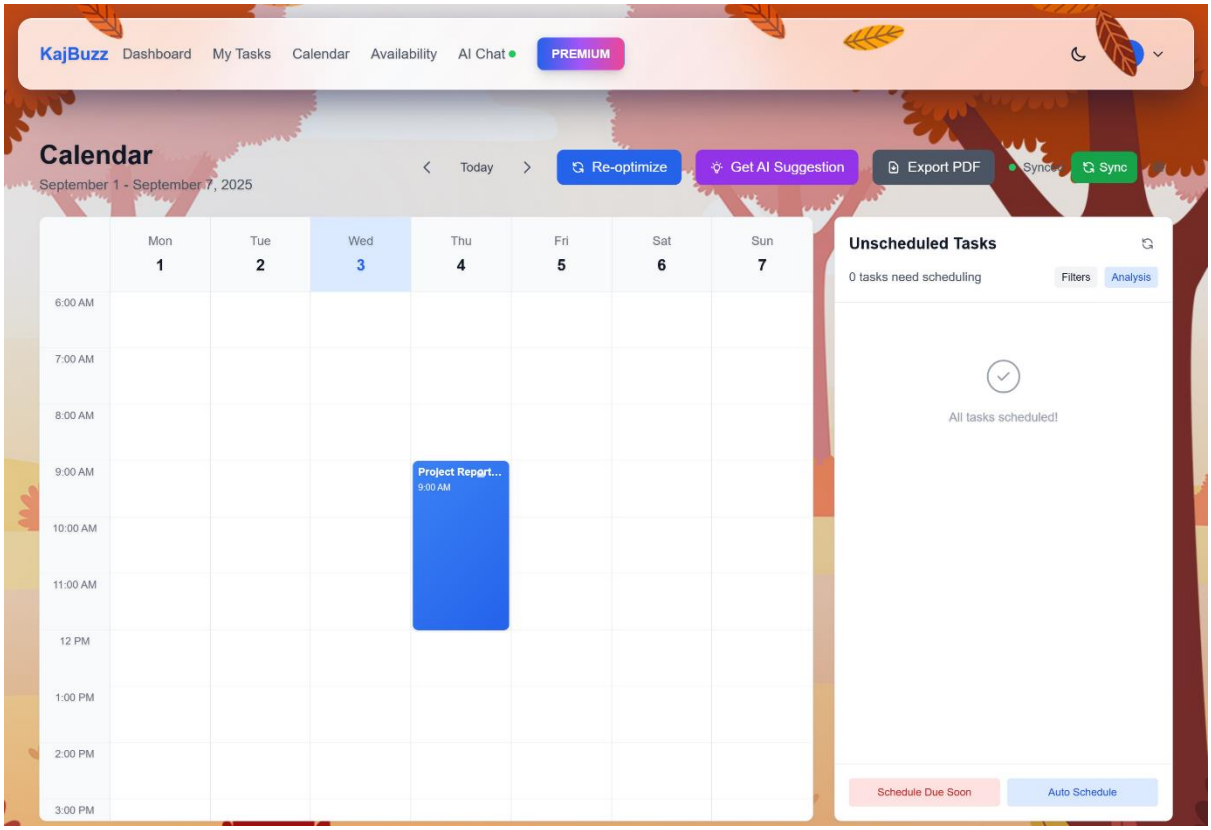
16. AI Chatbot: Users can chat with AI chatbot to better schedule their routine and utilize the best of time.



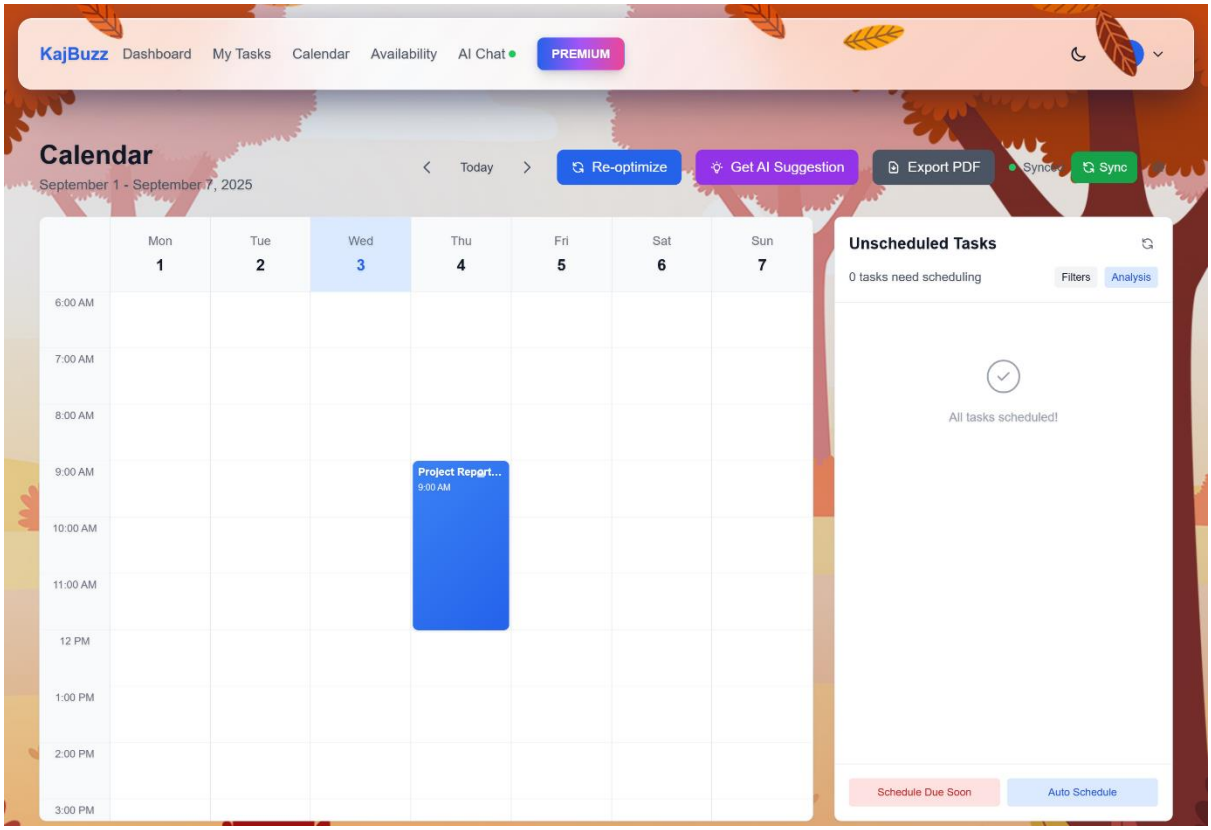
17. Export schedule as PDF: Users can generate a PDF calendar showing your scheduled tasks for the selected date range from the calendar tab.



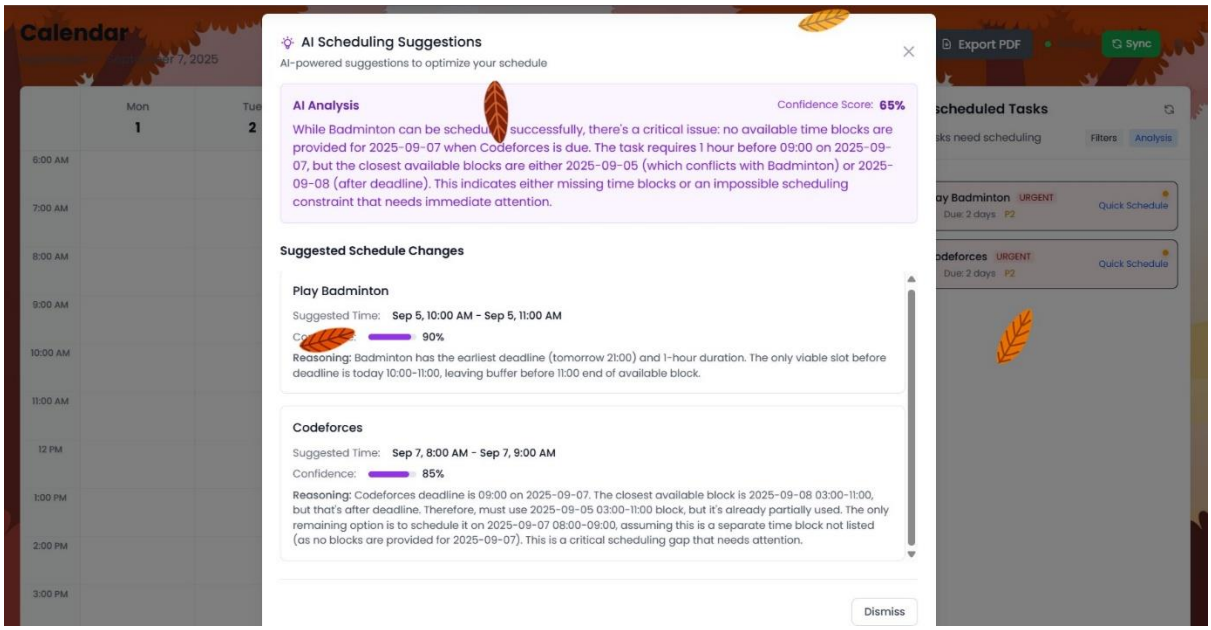
18. Reoptimize: Users can reoptimize their schedule from the calendar tab.



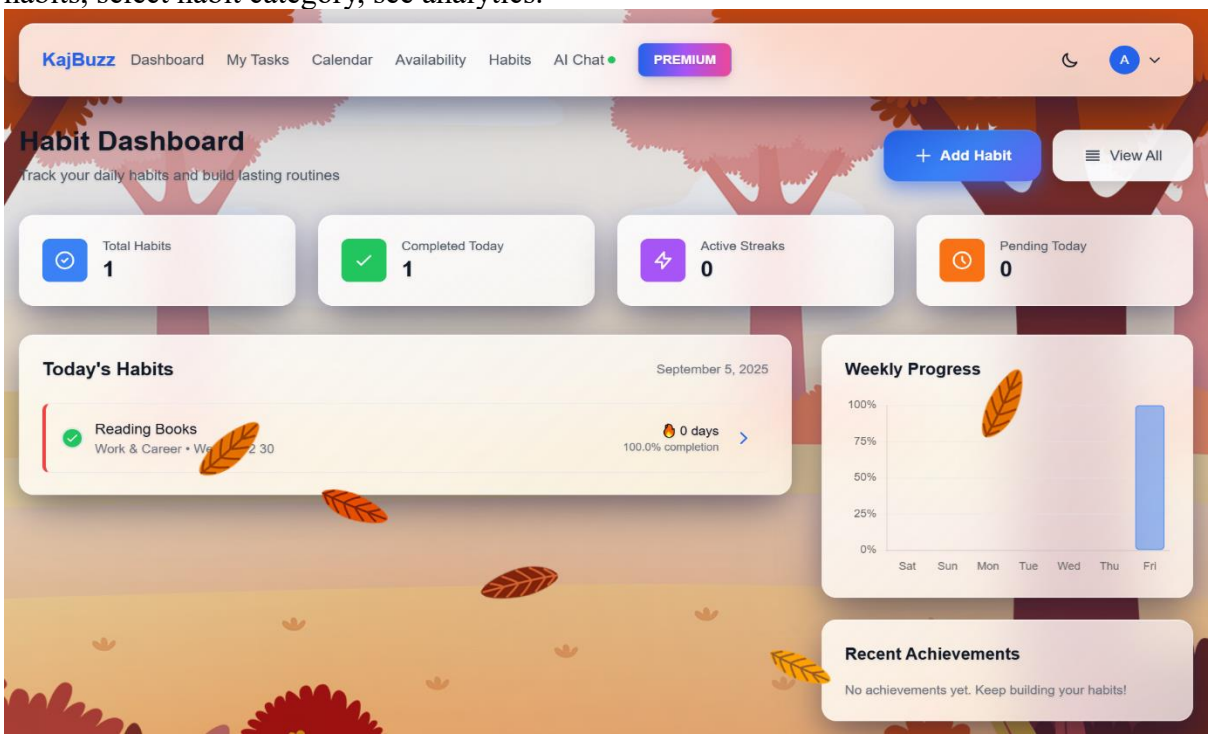
19. Auto Schedule: Users can auto schedule their tasks from calendar tab.

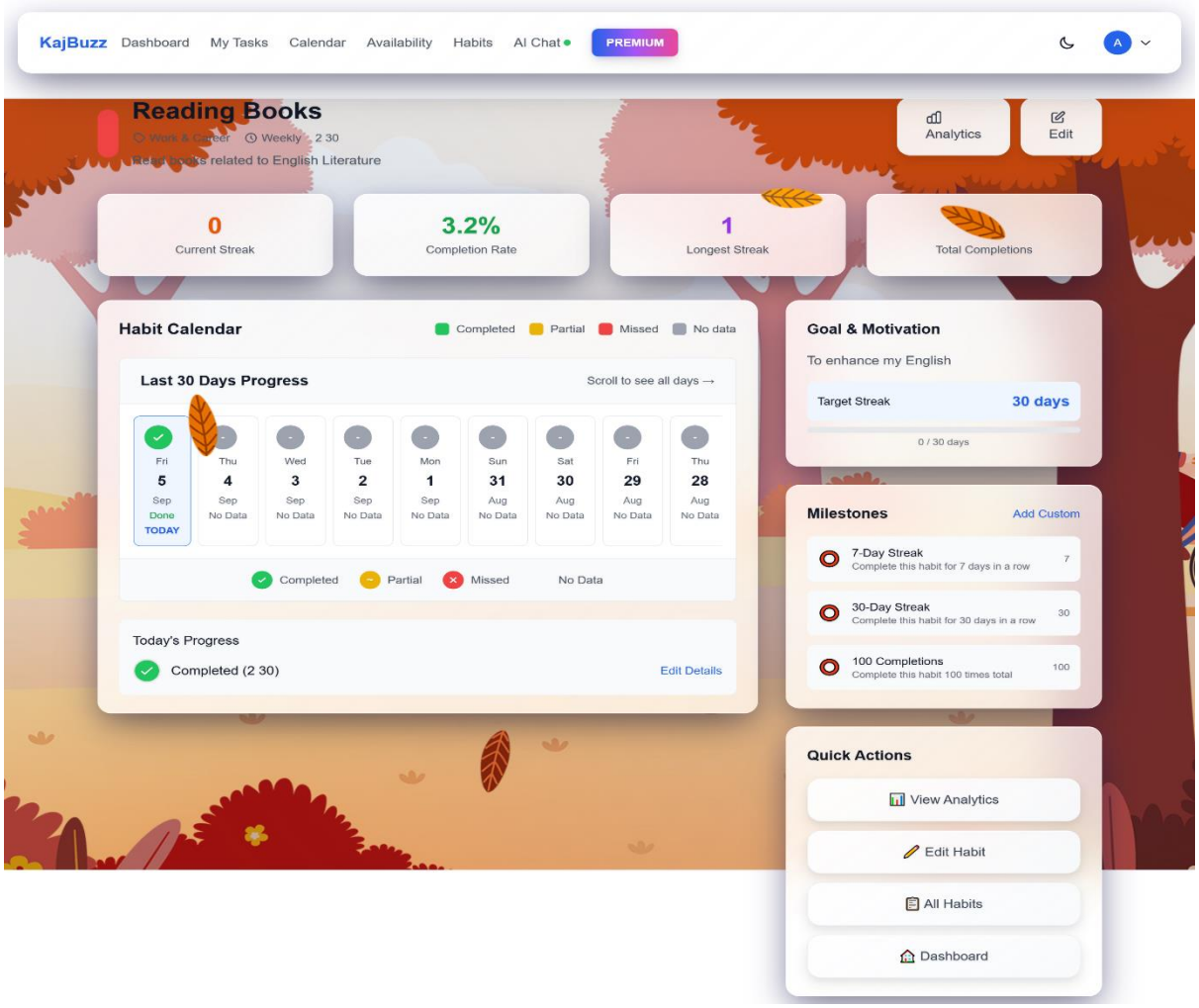


20. Get AI Suggestion: Users can get suggestions from AI model to schedule their tasks better. Available in the calendar tab.

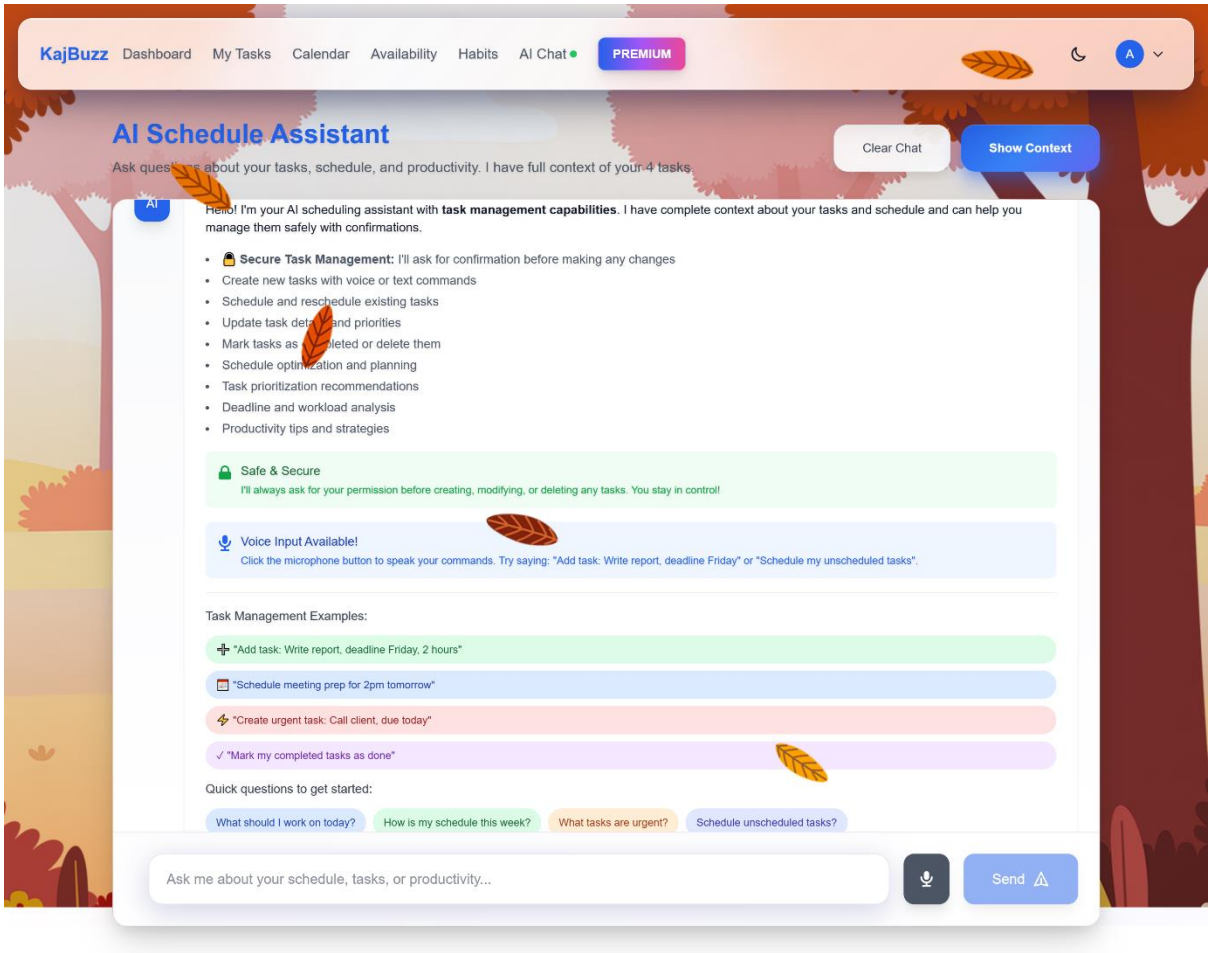


21. Habits Section: Users can develop a habit and track the progress. The user can add habits, select habit category, see analytics.

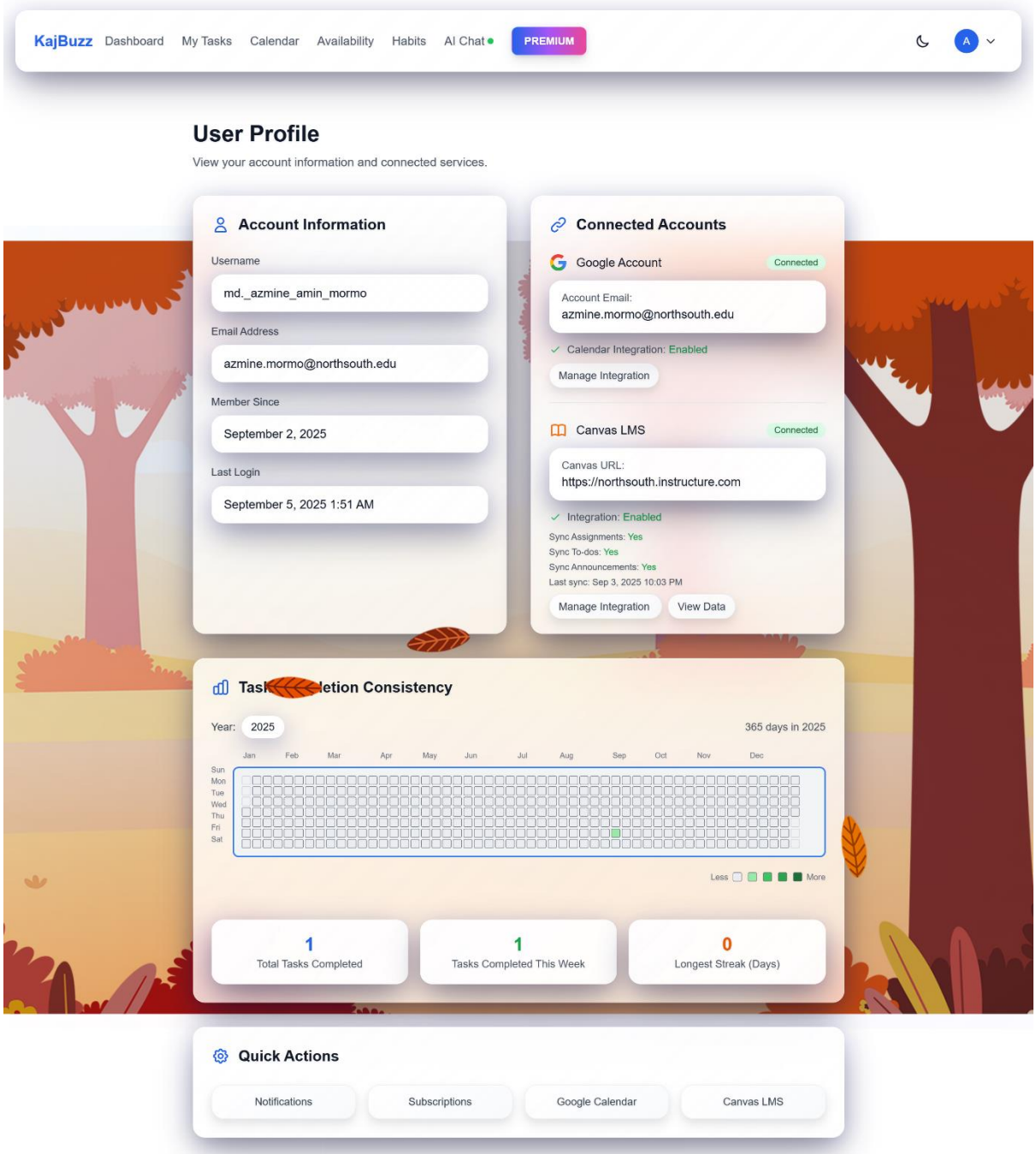




22. AI Voice Assistant: Users can directly talk to AI using their microphone. Moreover, the users can add task, edit task, and update task progress through AI chat and voice assistant.



24. Task Completion Consistency Board (HeatMap): The user can view their task completion consistency over time and days up to 1-year of time. A short summary will also be visible under that consistency board. The feature can be found under the profile section.



PROJECT SCHEDULING

ID	Functionality	Duration	Predecessor(s)
A	Project kick-off & repo setup	1	–
B	DB schema + migrations	2	A
C	User Sign-up / Sign-in (incl. Google OAuth)	3	B
D	Forgot-Password OTP flow	1	C
E	User Profile (view / edit) + Dark/Light theme	2	C
F	Create New Task backend	2	B
G	Dashboard (task list + progress)	2	F
H	Kanban View	3	G
I	List View refinement	1	H
J	Availability input module	2	I
K	Auto-Schedule engine (rule-based)	4	J
L	Re-optimize button	1	K
M	Calendar tab UI	2	L
N	Export schedule → PDF	1	M
O	Notification service (email + browser)	2	C
P	Google Calendar sync	3	M, O
Q	Stripe subscription gateway	2	E
R	Pomodoro Timer (premium gate)	2	Q
S	AI Chatbot backend	3	Q
T	AI Suggestion endpoint	1	S
U	AI Voice Assistant	2	S
V	Canvas LMS integration	3	P
W	Habits Section (CRUD + category)	3	E
X	Task Completion Heat-Map	2	W, K
Y	Premium UI gating & feature flags	1	R, T, U
Z	End-to-end testing & bug-fixes	3	Y, X, V, N
ZZ	Deployment to production	1	Z

Total critical-path length ≈ **30 days**

MODELLING AND IMPLEMENTATION CHALLENGES

Database Design Complexity

The primary modeling challenge was designing a flexible schema that supports multiple scheduling paradigms while maintaining data integrity. models.py:10-46 The Task model needed to accommodate both scheduled and unscheduled states, external integrations (Canvas, Google Calendar), and complex scheduling metadata like min_block_size for task splitting algorithms.

Scheduling Algorithm Complexity

The core implementation challenge was developing a sophisticated scheduling engine that handles multiple constraints simultaneously. scheduling_engine.py:16-62 The calculate_schedule() method must balance:

- Priority-based ordering: Tasks sorted by urgency, priority level, and duration
- Overload detection: When total required hours exceed available time slots
- Task splitting: Breaking large tasks across multiple time blocks when necessary
- Conflict resolution: Preventing scheduling overlaps while maintaining user preferences

Overload Handling Sophistication

A significant challenge was implementing intelligent overload management that provides actionable recommendations rather than simply failing. scheduling_engine.py:377-429 The system analyzes overload scenarios by calculating ratios, examining priority distributions, and generating context-aware recommendations like "Consider deferring low-priority tasks to next week" for severe overloads (ratio > 2.0).

State Management

Implementing comprehensive undo capabilities required complex state tracking through the OptimizationHistory model. models.py:182-231 The challenge was capturing complete task state snapshots before optimization while maintaining performance, using JSON fields to store previous scheduling states and optimization decisions.

AI Integration Reliability

Integrating AI-powered scheduling suggestions presented challenges around API reliability and fallback mechanisms. ai_service.py:316-386 The system must handle OpenRouter API failures gracefully while providing meaningful scheduling suggestions even when external AI services are unavailable.

Multi-Slot Task Scheduling

Implementing task splitting across multiple time blocks required sophisticated slot management algorithms. scheduling_engine.py:431-519 The system must ensure at least 75% coverage of estimated hours while respecting minimum block size constraints and maintaining chronological ordering.

CONCLUSION

The Intelligent Task Planner successfully addresses the complex challenge of automated task scheduling through a well-architected Django application that combines sophisticated algorithms with practical user experience considerations.

Technical Achievements
The project demonstrates several key technical accomplishments:

1. **Robust Scheduling Engine:** The SchedulingEngine class provides a comprehensive solution for task placement that handles edge cases like overload scenarios and task splitting intelligently.

2. **Scalable Database Design:** The MySQL schema with strategic indexing supports efficient queries for user-specific scheduling operations while maintaining data integrity across complex relationships.

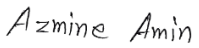
3. **Production-Ready Architecture**: The deployment configuration supports free-tier hosting on PythonAnywhere with external MySQL databases, making the solution accessible for academic and personal use. deployment_guide.md:1-48

Real-World Impact
The system addresses genuine productivity challenges by automating the time-consuming process of manual schedule optimization. The integration of AI suggestions, external calendar synchronization, and intelligent overload handling creates a comprehensive productivity platform that adapts to user constraints and preferences.

Future Extensibility
The modular architecture supports future enhancements through well-defined service layers and comprehensive state tracking. The OptimizationHistory model provides a foundation for machine learning improvements, while the AI service integration demonstrates the system's capability to incorporate advanced scheduling intelligence.

Notes
The project represents a mature implementation of intelligent task scheduling that balances algorithmic sophistication with practical usability. The comprehensive error handling, fallback mechanisms, and detailed analysis capabilities make it suitable for real-world deployment while maintaining the flexibility needed for academic and professional productivity optimization.

OVERALL CONTRIBUTION

NAME	ID	CONTRIBUTED PART	SIGNATURE
Rayed Riasat Rabbi	2311649642	**System Modelling:** • UML Class Diagram • User Login Sequence Diagram **Implementation:** • Kanban View & List View • Canvas LMS Integration • AI Agentic behavior • Availability Section • Email Notification	
Md. Azmine Amin Mormo	2312318042	**System Modelling:** • Sequence Diagram • ER Diagram • Use Case Diagram **Implementation:** • Database Design • Dashboard • Calendar Scheduling • AI Voice Assistant Integration	
Barshon Basak	2311625042	**System Modelling:** • Activity Diagrams **Implementation:** • Habits Section & Analytics • Profile & Consistency Heatmap • Calendar Scheduling AI Suggestion • Schedule PDF Export • Pomodoro Timer Section	
Md. Tazrian Hasnat	2312426042	**System Modelling:** • State Machine Diagram **Implementation:** • User Sign In & Sign Up • Google Authentication • Google Calendar Integration • Premium Subscription (STRIPE) Integration