Aurora: Focus App

Oghenetejiri Etaghene, Adrian Hautea, Mohammad Khan, Sravya Kotamraju, Vrishti Misra, Nihal Paul, Ece Yobas

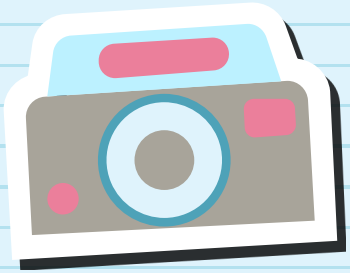
01

Objective

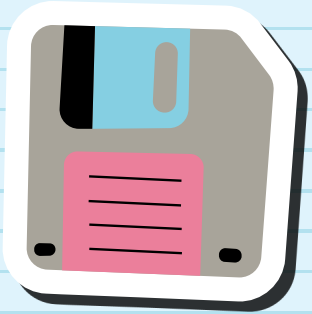
Objective of the Project Description

Project Objective

- **Objective:**
  Design and implement a productivity application that helps users maintain focus, manage tasks efficiently, and track progress through a *Pomodoro*-based system with streak tracking, statistics, and leaderboard features.
- **Core Goals:**
  - Provide a reliable focus timer with automatic transitions.
  - Allow users to create, organize, and link tasks to focus sessions.
  - Track session history, focus minutes, completed tasks, and productivity streaks.
  - Motivate users through progress analytics and optional leaderboard rankings.
  - Deliver a simple, efficient, and intuitive interface that supports consistent study and work habits.



02



Cost Estimation

Function Point Method

Cost & Effort Estimation

Function Category	Description	Count	Complexity	Weight	Calculation	Total
User Inputs (EI)	Updates Internal Data	16	Simple	3	16x3	48
User Outputs (EO)	System Outputs	12	Simple	4	12x4	48
User Queries (EQ)	Read-Only Requests	12	Complex	6	12x6	72
Data Files (ILF)	Internal Logical Files	5	Moderate	10	5x10	50
External Interfaces (EIF)	External Data Sources	5	Complex	10	5x10	50
GFP	-	-	-	-	Sum	268

Processing Complexity (PC) & Adjustment (PCA)

Factor	Rating	Reason
PC1: Reliable Backup	5 (High)	Local + Cloud Sync Reliability
PC14: User-friendliness	5 (High)	Necessary for Convenience

Total Processing Complexity (PC):

PC = (12 x 3) + (2 x 5) = **46**

Final Function Points (FP):

FP = 268 x 1.11 = **298 FP**

Processing Complexity Adjustment (PCA):

PCA: 0.65 + 0.01 (46) = 0.65 + 0.46 = **1.11**

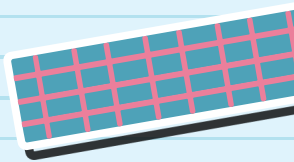
Effort and Duration

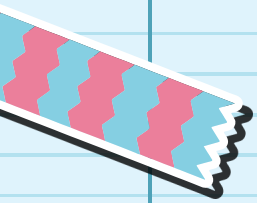
Total/Final Project Cost

- Productivity Rate: 30 FP per person-week
- Effort (pw): E = 298/30 = 9.93 person-weeks
- Effort (hrs): 9.93 pw x 40 hrs/pw = 397 hours
- Duration (6 person team) = 4 months
  *(adjusted for student team working max 5hrs/week)
- Personnel & Labor: $7200 (6 ppl @ 15/hr for 5 hrs/week)
- Hardware (Cloud Server): $1150 (Amazon S3)
- Software/Services: $124 (Google Play + App Store)
- Total Project Cost: $8474
- Recommended Project Cost: 1.5 x $8474 = $12,711

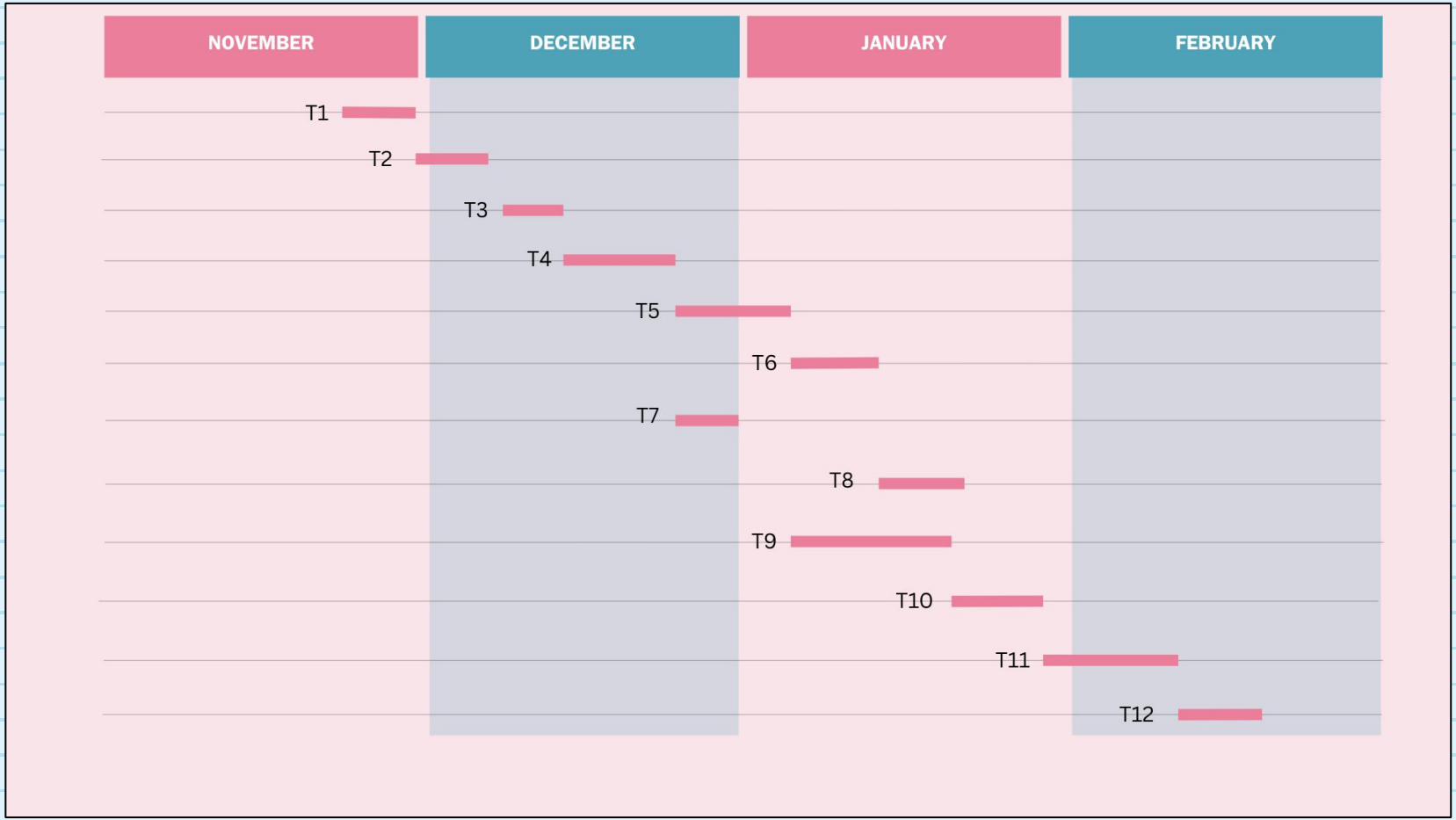
03

Project Timeline





Task	Effort (person-days)	Duration (days)	Dependencies	Start date
T1: define scope and requirements	12	4		11/24/25
T2: create system architecture and UML diagrams	15	5	T1: define project scope	11/28/25
T3: set up development environment	6	3	T2: system architecture	12/5/25
T4: implement timer functionality	32	8	T3: development environment	12/10/25
T5: develop task management system	32	8	T4: timer	12/22/25
T6: build session history tracking	20	5	T5: task management	1/5/26
T7: create streak tracker	20	5	T4: timer	12/22/25
T8: implement leaderboard	20	5	T6: session history	1/12/26
T9: design UI	36	9	T5: task management	1/5/26
T10: integrate frontend and backend	42	7	T4: timer, T9: UI	1/16/26
T11: testing and bug fixes	48	12	T10: integration	1/27/26
T12: documentation	36	6	T11: testing	2/12/26


NOVEMBER
DECEMBER
JANUARY
FEBRUARY
T1
T2
T3
T4
T5
T6
T7
T8
T9
T10
T11
T12

04

Functional & Non-Functional Requirements

Functional Requirements

Session Control

Start, pause, resume, stop sessions

Progress Tracking

Log completed sessions, track streaks, track total minutes

Session Persistence

Save and restore and timer state when closed

Notifications

Send alerts at start and end of focus/break sessions

Task Management

Create, edit, categorize, prioritize, and delete tasks

Leaderboard

Rank users by focus time and streaks

Non-Functional Requirements

Product Requirements

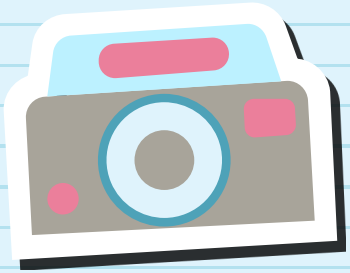
- Dashboard loads within 3 secs
- Less than 200 MB stored
- Less than 350 MB cached
- Local data saved
- HTTPS encrypted
- Intuitive interface
- Works on all devices

Organizational Requirements

- Team uses agile with weekly updates
- GitHub commits with great documentation
- Diagrams created using draw.io

External Requirements

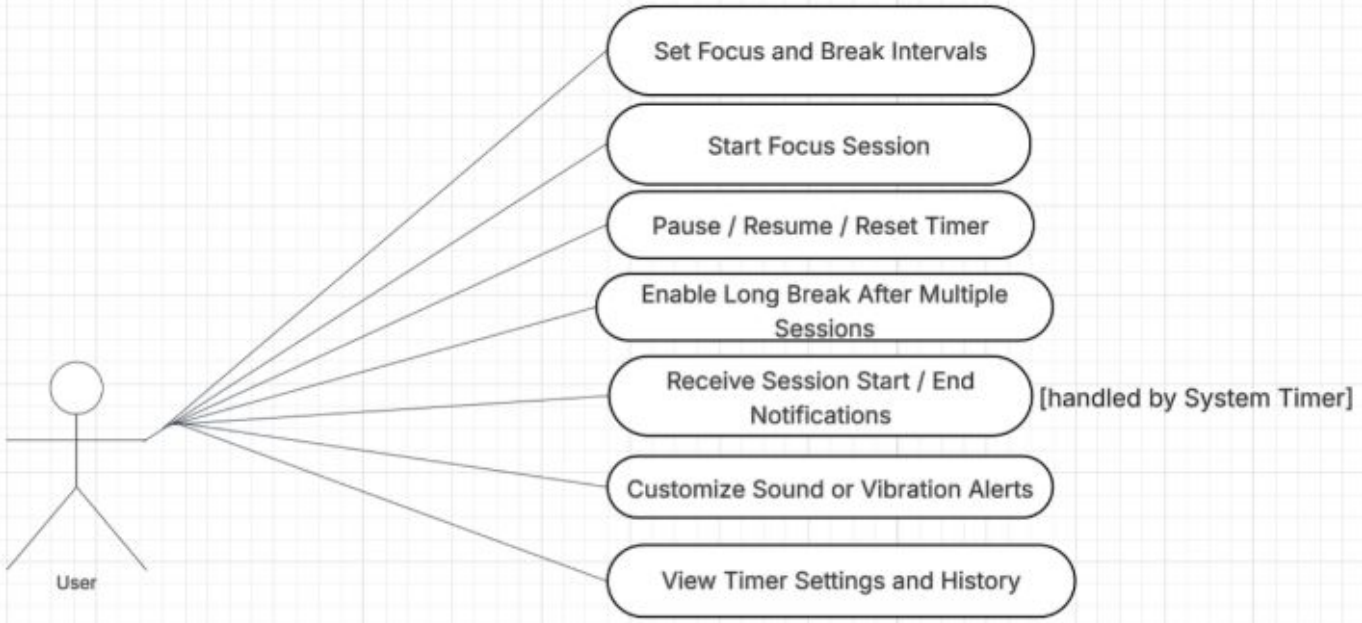
- Meets data handling rules
- Meets privacy rules
- Store approved payment systems for purchases
- No unnecessary data collection
- Restrict underage users
- Follow platform publishing policies



05

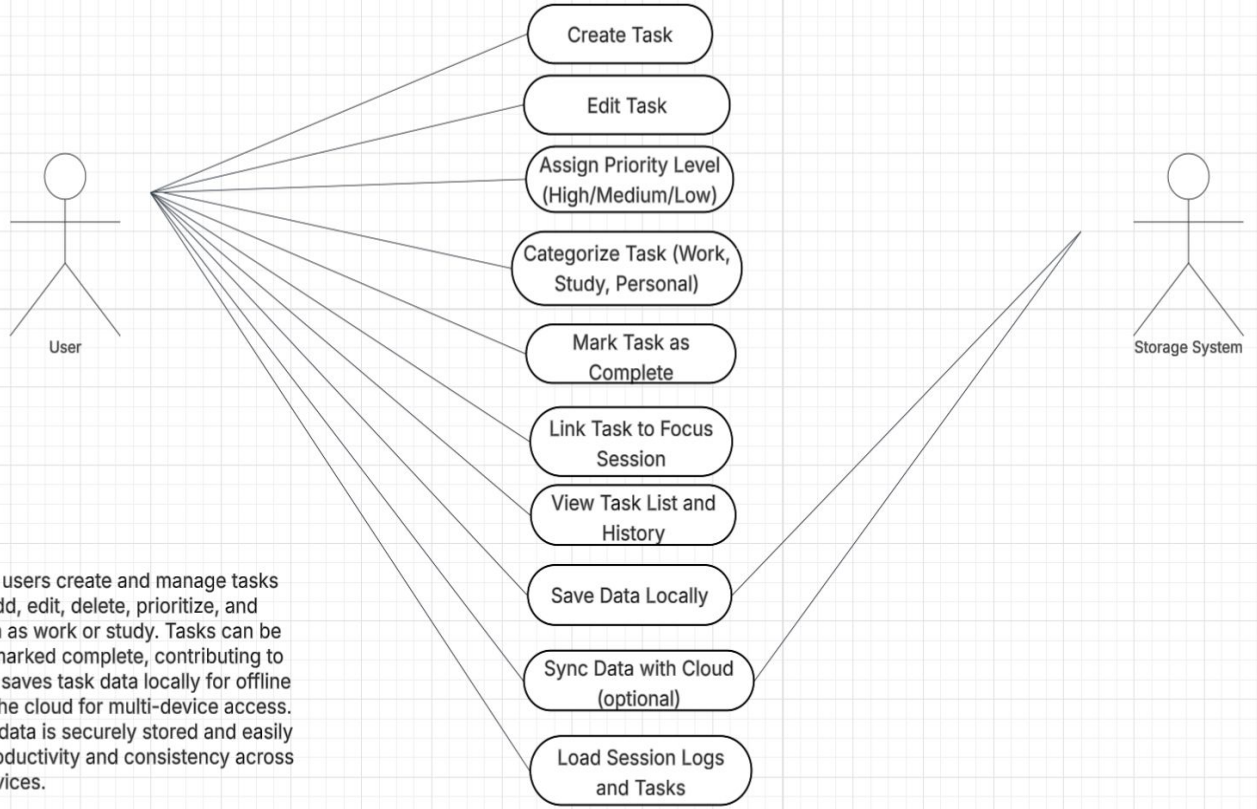
Use-Case Diagrams

Figure 1: Focus Timer and Notifications



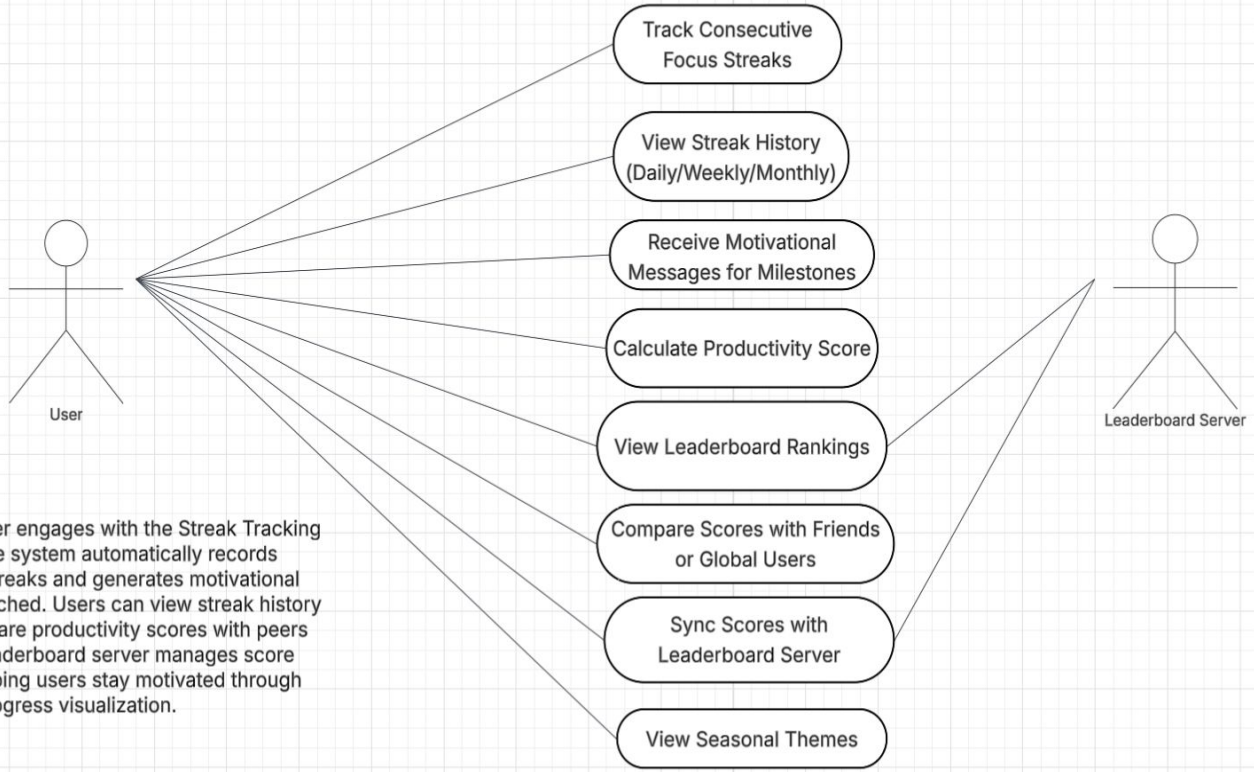
This diagram shows how the user interacts with the Focus Timer module to manage work sessions. The user can set customized focus and break durations, start or pause sessions, and enable long breaks after several completed cycles. The system timer automatically transitions between focus and break phases and sends notifications through sound or vibration. This module ensures that time management and reminders operate automatically, minimizing user distraction.

Figure 2: Task Management and Data Handling



This diagram represents how users create and manage tasks within Aurora. Users can add, edit, delete, prioritize, and categorize tasks by type such as work or study. Tasks can be linked to focus sessions and marked complete, contributing to progress tracking. The system saves task data locally for offline use and can synchronize it to the cloud for multi-device access. This module ensures that user data is securely stored and easily retrievable, supporting both productivity and consistency across devices.

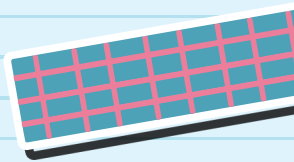
Figure 3: Streak Tracking and Leaderboard

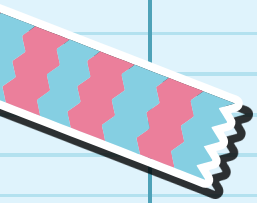


This diagram illustrates how the user engages with the Streak Tracking and Leaderboard modules. The system automatically records consecutive focus sessions as streaks and generates motivational messages when milestones are reached. Users can view streak history by day, week, or month and compare productivity scores with peers through the leaderboard. The leaderboard server manages score synchronization and ranking, helping users stay motivated through competition and progress visualization.

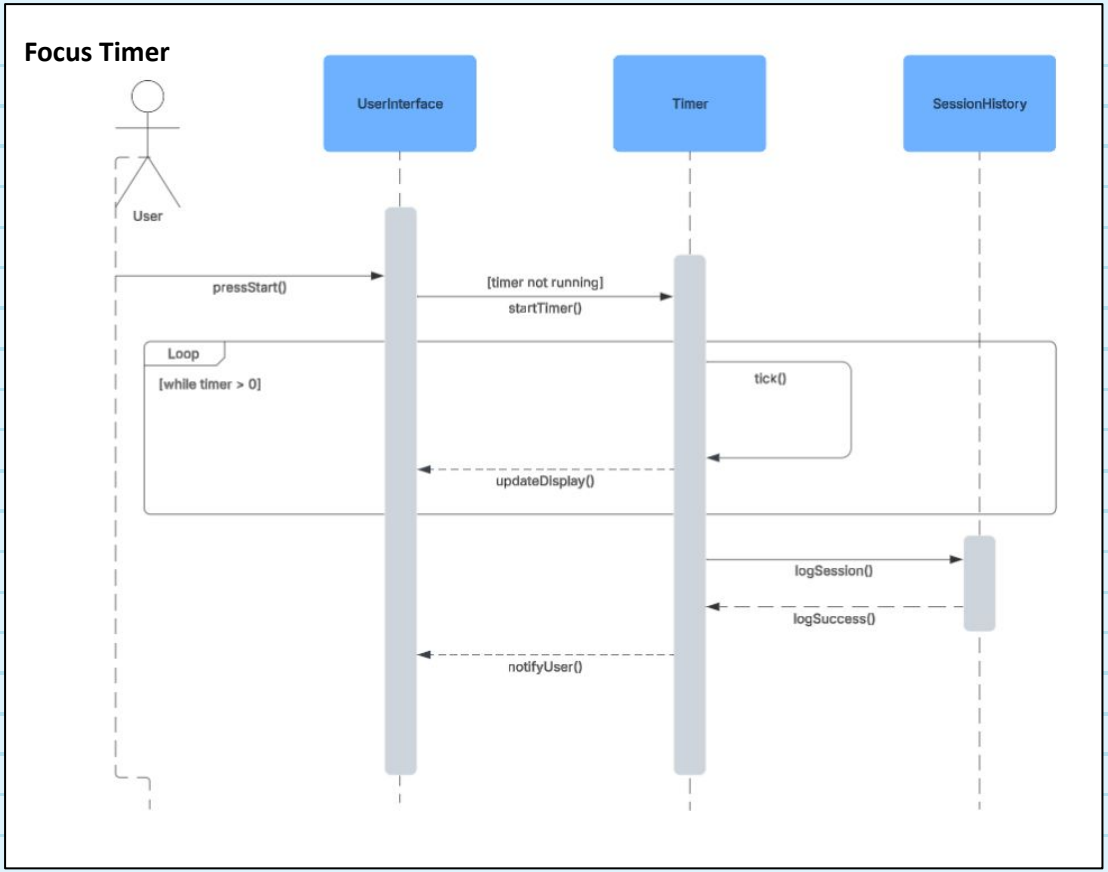
06

Sequence Diagrams

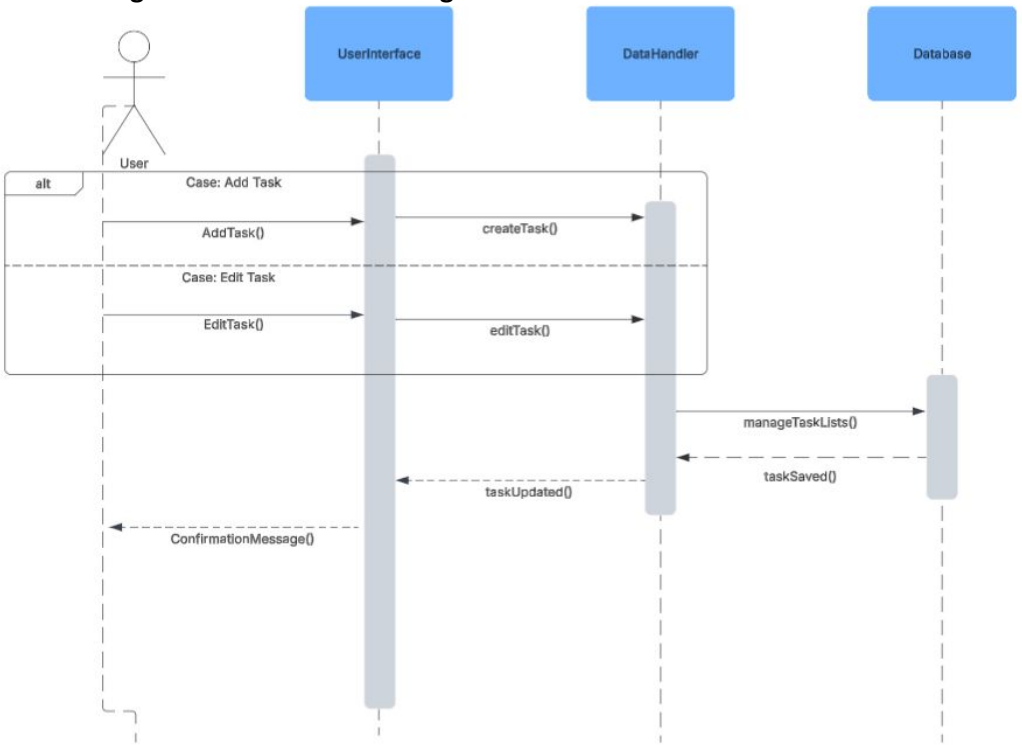




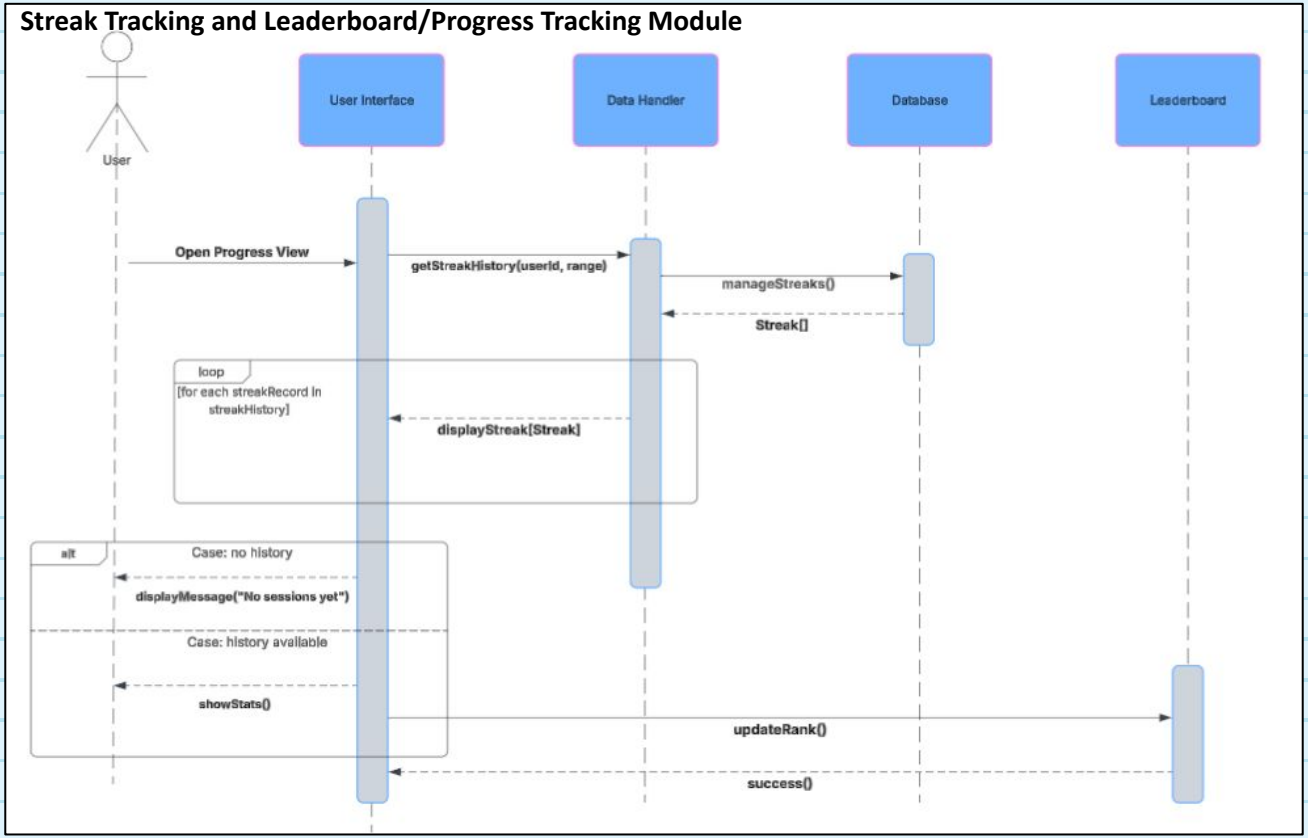
Focus Timer


UserInterface
Timer
SessionHistory
User
pressStart()
[timer not running]
startTimer()
Loop
[while timer > 0]
tick()
updateDisplay()
logSession()
logSuccess()
notifyUser()

Task Management and Data Handling

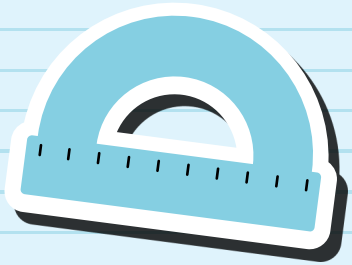


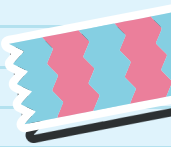
Streak Tracking and Leaderboard/Progress Tracking Module

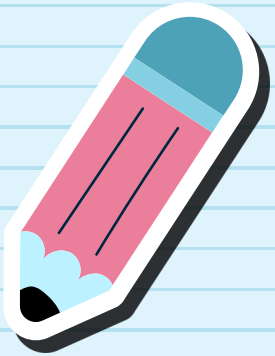


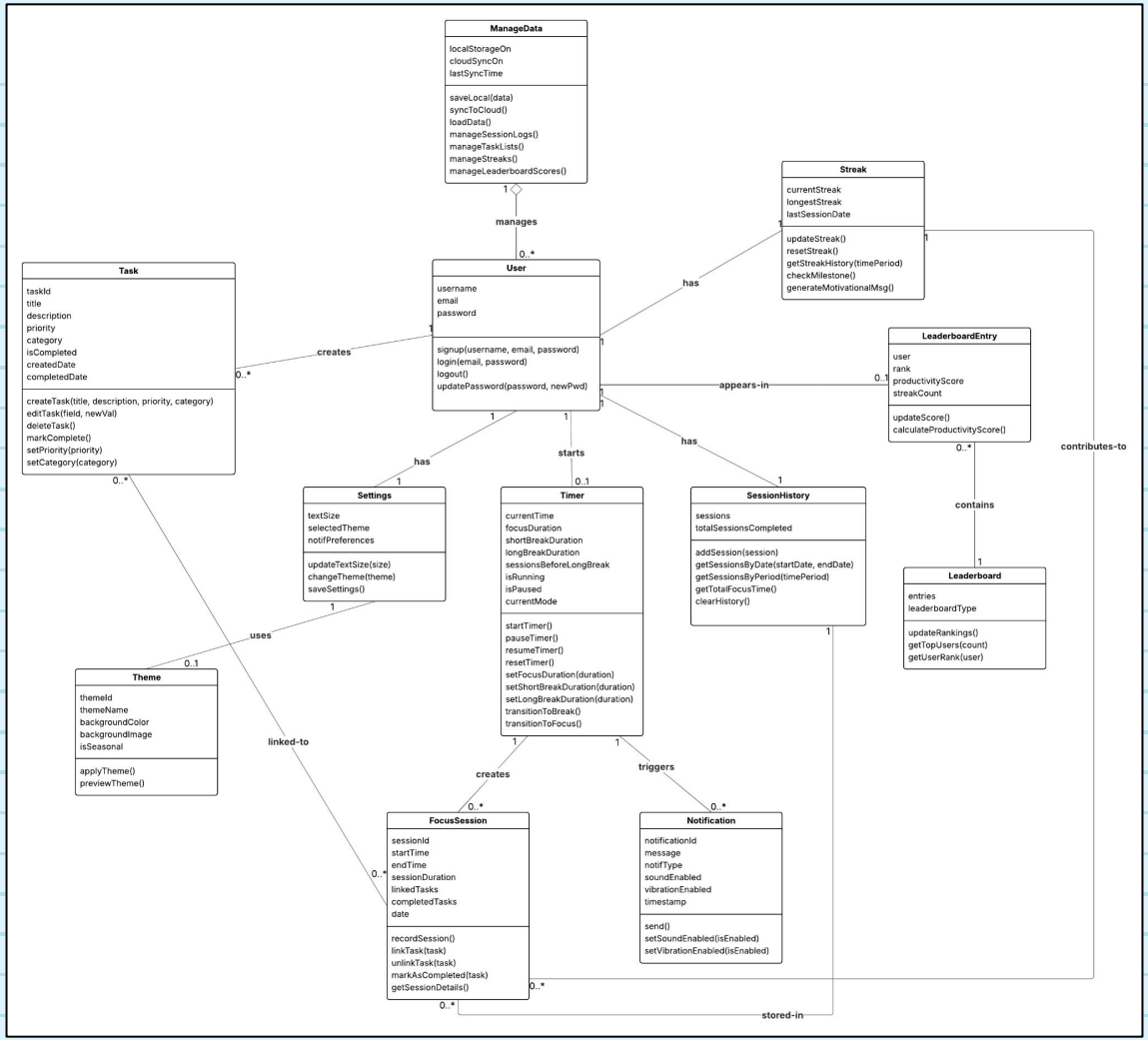
07

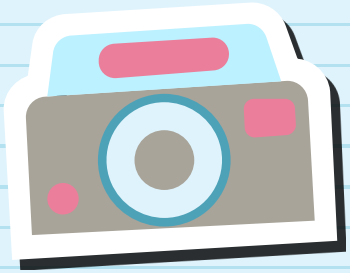
Class Diagram



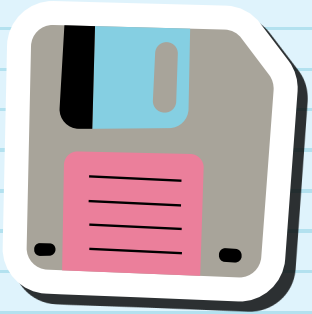





ManageData
localStorageOn
cloudSyncOn
lastSyncTime
saveLocal(data)
syncToCloud()
loadData()
manageSessionLogs()
manageTaskLists()
manageStreaks()
manageLeaderboardScores()
manages
Streak
currentStreak
longestStreak
lastSessionDate
updateStreak()
resetStreak()
getStreakHistory(timePeriod)
checkMilestone()
generateMotivationalMsg()
has
Task
taskId
title
description
priority
category
isCompleted
createdDate
completedDate
createTask(title, description, priority, category)
editTask(field, newVal)
deleteTask()
markComplete()
setPriority(priority)
setCategory(category)
creates
User
username
email
password
signup(username, email, password)
login(email, password)
logout()
updatePassword(password, newPwd)
appears-in
LeaderboardEntry
user
rank
productivityScore
streakCount
updateScore()
calculateProductivityScore()
contributes-to
contains
Leaderboard
entries
leaderboardType
updateRankings()
getTopUsers(count)
getUserRank(user)
has
starts
has
Settings
textSize
selectedTheme
notifPreferences
updateTextSize(size)
changeTheme(theme)
saveSettings()
Timer
currentTime
focusDuration
shortBreakDuration
longBreakDuration
sessionsBeforeLongBreak
isRunning
isPaused
currentMode
startTimer()
pauseTimer()
resumeTimer()
resetTimer()
setFocusDuration(duration)
setShortBreakDuration(duration)
setLongBreakDuration(duration)
transitionToBreak()
transitionToFocus()
SessionHistory
sessions
totalSessionsCompleted
addSession(session)
getSessionsByDate(startDate, endDate)
getSessionsByPeriod(timePeriod)
getTotalFocusTime()
clearHistory()
uses
Theme
themeId
themeName
backgroundColor
backgroundImage
isSeasonal
applyTheme()
previewTheme()
linked-to
creates
triggers
FocusSession
sessionId
startTime
endTime
sessionDuration
linkedTasks
completedTasks
date
recordSession()
linkTask(task)
unlinkTask(task)
markAsCompleted(task)
getSessionDetails()
Notification
notificationId
message
notifType
soundEnabled
vibrationEnabled
timestamp
send()
setSoundEnabled(isEnabled)
setVibrationEnabled(isEnabled)
stored-in



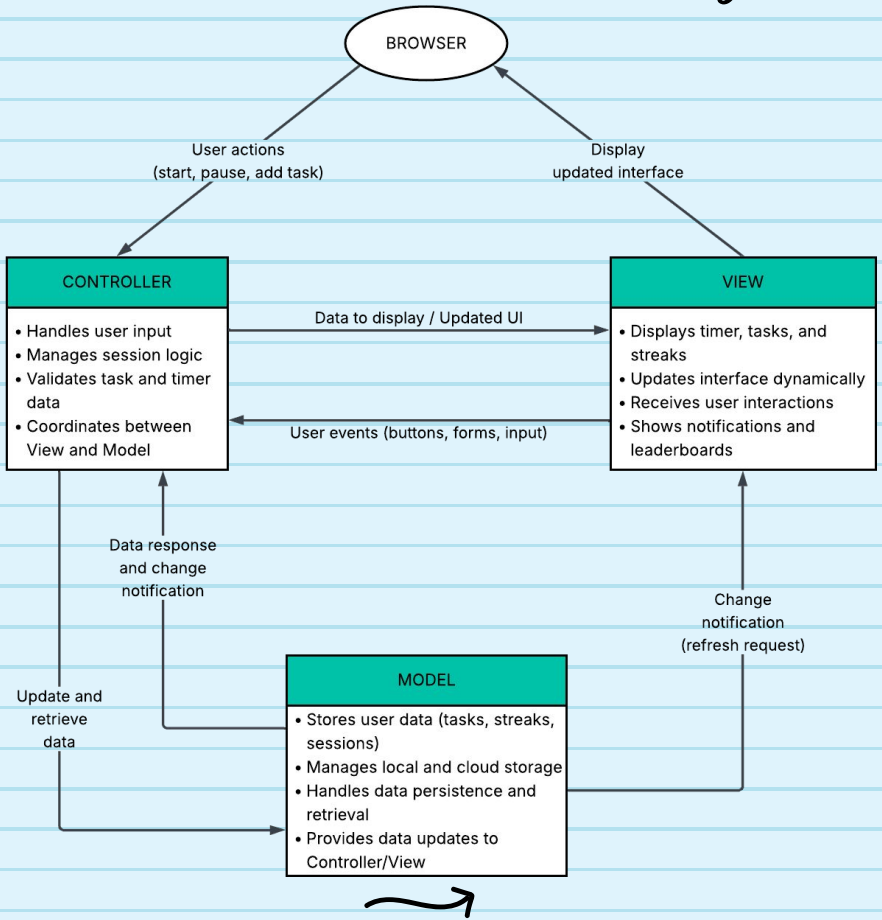
08



Architectural Design

Model-View-Controller (MVC) Pattern

Architectural Design



Final Implementation

https://aurora-focus-app.vercel.app/

Framework: Next.js 16
Language: TypeScript
Styling: Tailwind CSS
UI Components: Radix UI + shadcn/ui
Icons: Lucide React
Charts: Recharts
State Management: React Hooks