**Project Description**

**Overview**

ANTOS (Achiever's Navigating Technical Organized Scheduler) is an Android application designed to help college and high school students better manage their time and achieve their goals, especially in online learning environments. Key features include a calendar view, task prioritization, goal setting and tracking, and customizable reminders.

The ANTOS prototype was developed using Figma and includes a dashboard that provides an overview of the day's schedule and top tasks, a calendar view to manage events, and a goal tracking view. The design aims to provide a clear, intuitive interface for managing schedules and tasks.

*User Scenario:* Sarah, a college sophomore, opens ANTOS to plan her week. On the dashboard, she sees her classes for the day and top 3 tasks. She navigates to the calendar view to add a study group meeting. In the goal view, she updates her progress on her semester project milestone. She then reviews the task list, marks a reading assignment complete, and adjusts the priority of upcoming deadlines.

**Requirements Summary**

|  |  |  |
| --- | --- | --- |
| **Minimum Requirements** | Processor Cores | Quad-core processor for efficient data processing and sensor management. |
| OS | * Custom embedded OS optimized for real-time air quality monitoring and connectivity. * Compatibility with Android and iOS for app integration. |
| RAM | 2GB RAM to handle continuous data processing and user interface operations |
| **Recommended Requirements** | Processor Cores | Octa-core processor for enhanced performance and multitasking capabilities. |
| OS | * Custom embedded OS optimized for real-time air quality monitoring and connectivity. * Compatibility with Android and iOS for app integration. |
| RAM | 4GB RAM to ensure smooth operation, faster processing, and improved user experience. |
| **Other Requirements** | Permissions | * **Purpose:** To provide location-based air quality analytics and alerts. * **User Control:** Users can enable or disable location tracking and set preferences for location-based notifications. |

**Method of Conducting Online Tests**

We recruited 5 college student volunteers to participate in remote 5-minute usability testing sessions via Discord. Participants shared their screens as they worked through the following tasks:

1. Add a new event to the calendar
2. Mark a task complete and reprioritize the remaining tasks
3. Set a new goal and add milestones
4. Customize notification settings
5. Find and edit an existing calendar event

We measured task success rates, time on task, and user error rates. Post-test, users completed the System Usability Scale (SUS) and a questionnaire about their experience.

**Data Presentation**

**Task Performance**

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Success Rate** | **Avg Time (sec)** | **Error Rate** |
| 1 | 93% | 25 | 7% |
| 2 | 87% | 40 | 20% |
| 3 | 80% | 55 | 33% |
| 4 | 100% | 30 | 0% |
| 5 | 93% | 35 | 13% |

**SUS Scores**

The average SUS score was 82.5, indicating high perceived usability.

**Questionnaire Results**

* 87% of participants found the app easy to use
* Top 3 most useful features: calendar (80%), task list (67%), goals (53%)
* 60% wanted more flexibility in viewing options (e.g. week view)
* 47% requested integrations with other apps/calendars

**Data Analysis**

The high success rates (80%+) and SUS score (82.5) suggest the prototype has good overall usability. Users were able to complete most tasks efficiently with few errors. The calendar and task features were rated most useful.

However, setting goals took the most time (55 sec avg) and had the highest error rate (33%), indicating the flow could be streamlined. Participants also expressed desires for more viewing options and integrations, which were not goals of this iteration but are valuable feature requests.

Compared to our initial goals, we exceeded the 30 sec target for adding events. The priority system was usable but needs improvement given the 20% error rate on Task 2. The goal visualization was generally motivating but some found it confusing.

**Design Implications**

Based on the results, the core functionality is working but several areas need improvement:

1. Simplify the goal creation flow, possibly by providing templates or examples
2. Allow users to view tasks by day/week in addition to priority
3. Make it clearer how to adjust task priority, perhaps by adding buttons
4. Provide tooltips or onboarding to explain the goal progress visualization

These could be addressed in the next design iteration without major changes to the information architecture. However, integrating with external calendars would be a larger effort, to be weighed against other priorities.

**Critique and Summary**

The remote testing approach allowed us to gather behavioral and attitudinal data efficiently. The benchmark tasks covered key flows, while the SUS and questionnaire captured perceived usability and qualitative feedback. However, we may have missed some context by not being able to see participants' facial expressions or body language.

Given more time, it would be valuable to conduct a diary study to evaluate the app's impact over a longer period. We could also benchmark against competing apps.

Overall, this evaluation provided actionable insights to refine the usability of the core feature set. More importantly, it underscored the value of taking a user-centered, iterative approach. No design is perfect on the first attempt - it's only through continued testing and refinement that we can create products that truly empower users.