Personal Productivity Dashboard

Use Case Diagram

Jayson Dixon

CSMS Degree Program

Full Sail University

11/26/2025

Contents

Use Cases

Use Case 1: Create a New Habit

Actor: User **Goal:** Add a new habit to track (e.g., “Meditate for 10 minutes”)

Flow:

1. User selects “Add Habit” from the dashboard.
2. System prompts for habit name, frequency, and goal type.
3. User enters details and confirms.
4. System stores the new habit and integrates it into the tracking system.
5. Dashboard updates to include the new habit in visualizations and feedback loops.

Postconditions:

- Habit is saved and ready for daily logging.
- Visual and sound feedback systems are linked to the new habit.

Use Case 2: Review Habit History

Actor: User Goal: View historical data and trends for a specific habit

Flow:

1. User selects a habit from the dashboard.
2. System displays a timeline graph showing completion streaks, missed days, and overall consistency.
3. User can filter by week, month, or custom date range.
4. System overlays motivational milestones (e.g., “Longest streak: 14 days”).

Postconditions:

- User gains insight into their performance.
- Data is used to inform future suggestions and feedback.

Use Case 3: Receive Motivational Challenge

Actor: System (initiated by user interaction or inactivity) Goal: Encourage user engagement through a personalized challenge

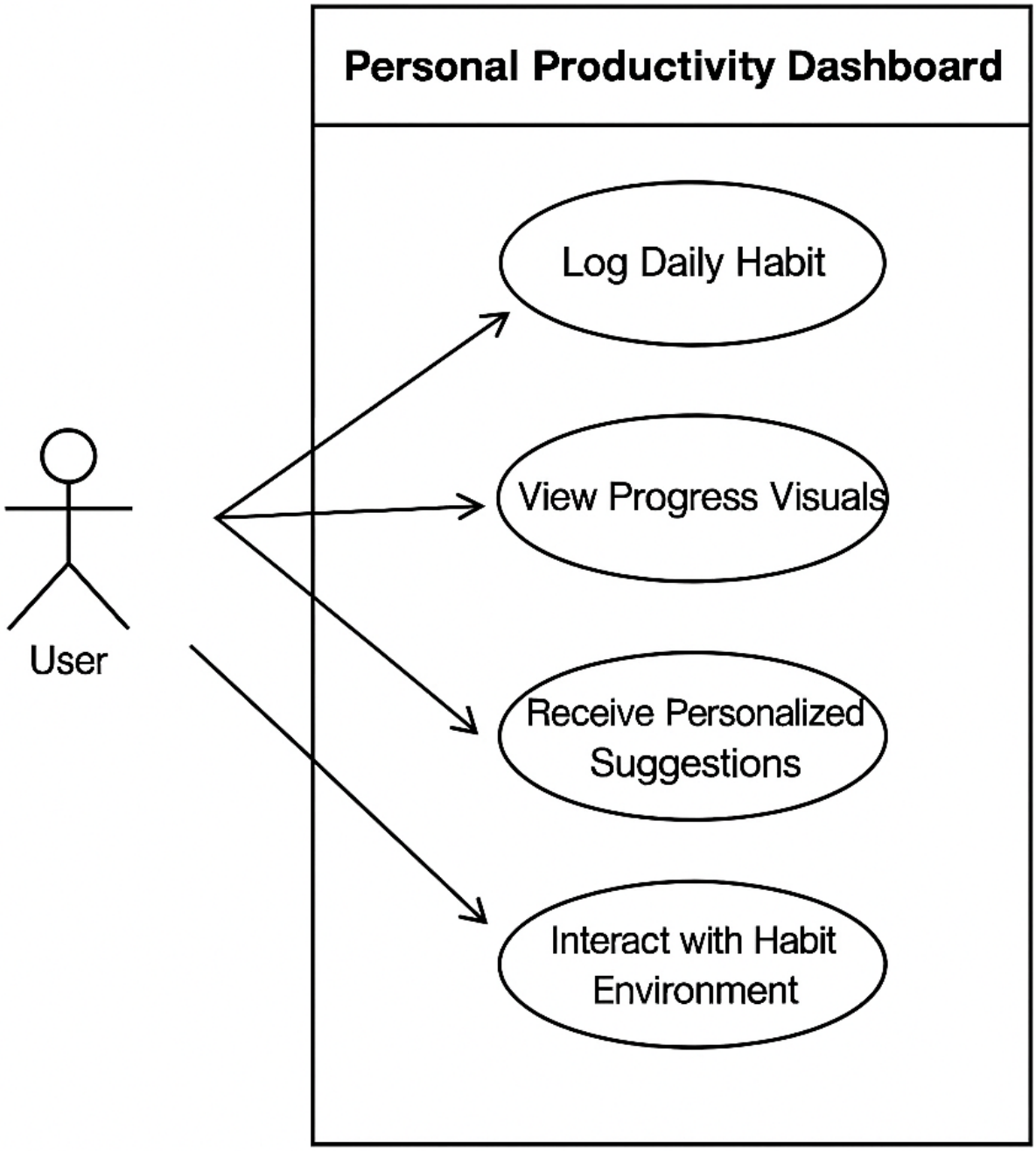
Flow:

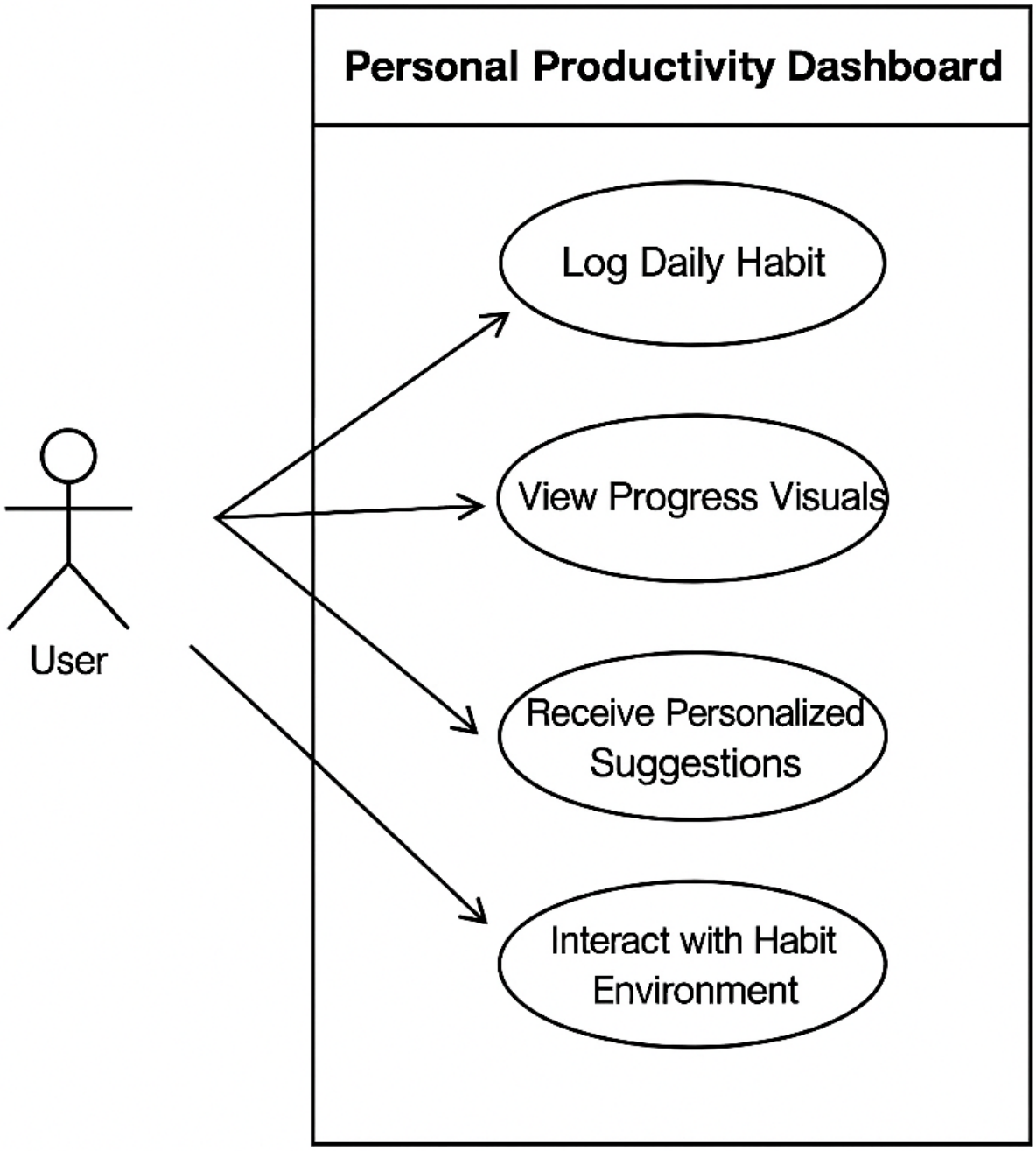
1. System detects a drop in habit consistency or user inactivity.
2. It generates a motivational challenge (e.g., “Complete 3 habits daily for the next 5 days to unlock a new visual upgrade”).
3. User accepts or declines the challenge.
4. If accepted, system tracks progress and provides enhanced feedback.
5. Upon completion, system rewards user with a visual upgrade (e.g., new tree in garden, badge, sound effect).

Postconditions:

- User re-engages with the dashboard.
- System reinforces positive behavior through gamification.

Use Case Diagram


Personal Productivity Dashboard
Log Daily Habit
View Progress Visuals
Receive Personalized Suggestions
Interact with Habit Environment
User


Personal Productivity Dashboard
Log Daily Habit
View Progress Visuals
Receive Personalized Suggestions
Interact with Habit Environment
User

Evaluation

Evaluation of Use Cases

Easier Use Cases

1. **Create New Habit**
    - *Challenge Level:* Low
    - *Why:* Involves basic UI input, local data storage, and list management. Well-supported by SFML and standard C++ libraries.
2. **Review Habit History**
    - *Challenge Level:* Moderate
    - *Why:* Requires data parsing and visualization. SFML doesn't have built-in charting, but libraries like HMViz offer SFML-based support for bar, line, and pie charts.
3. **Receive Motivational Challenge**
    - *Challenge Level:* Moderate
    - *Why:* Involves basic logic and UI feedback. Can be implemented with timers, streak counters, and simple conditionals.

Most Challenging Use Case: AI-Driven Habit Suggestions

Why It's Challenging

- **Algorithm Complexity:** Requires trend detection, pattern recognition, and personalized feedback.
- **Data Modeling:** Must analyze user behavior over time and generate meaningful insights.
- **Unknown Territory:** SFML doesn't natively support machine learning or recommendation systems.
- **Time Constraints:** Building a robust model from scratch could exceed the 15–20 hour major feature window.

🛍 Research & Feasibility Review

Recommendation Engine Techniques

Collaborative Filtering & Content-Based Filtering are standard approaches. These can be simplified for local use without large datasets.

- **Cosine Similarity & Embeddings** (used in OpenAI-powered engines) are powerful but may be overkill for a small desktop app.

🔧 Simplified Solution Strategy

- **Local Profile Learning:** Use basic rule-based logic (e.g., “If habit X is skipped 3 times on weekends, suggest shorter goals”).
- **Matrix-Based Analysis:** Track habits in a 2D array (days × habits), then apply simple scoring.
- **Offline Recommendations:** No need for real-time AI; batch analysis after each week is sufficient.

Testing Plan

- Prototype a basic scoring system using mock data.
- Validate that recommendations are logical and relevant.
- Ensure integration with UI and feedback systems is smooth.

Conclusion & Action Plan

- **Feasible with Simplification:** Full AI modeling is too complex, but simplified rule-based logic is achievable.
- **Time Allocation:** Week 3 will be dedicated to developing and testing this feature.
- **Fallback Strategy:** If recommendation logic proves too time-consuming, pivot to a static motivational system (e.g., milestone-based encouragement).