PM4 FocusBot Application
Michael Pang, Matthew Topping, Elijah Tynes, Ajay Seethana

Process Deliverable III

1. Prototype Overview

Our prototype focuses on key features, now enhanced with event-based architecture to improve responsiveness and scalability:

- Break Reminders: Notifications triggered in real-time based on productivity data, reducing fatigue by encouraging timely breaks.
- Hydration Tracking: Alerts for staying hydrated throughout the workday, monitored and delivered asynchronously for efficiency.
- Movement Suggestions: Real-time prompts to stretch or walk, leveraging event-driven insights for optimal timing.

2. Feature Mockups

The prototype now includes mockups reflecting event-based and fault-tolerant designs:

- Smart Watch Integration: Real-time notification screens that adapt to productivity drops or extended work periods.
- User Dashboard: A simplified, responsive layout tracking hydration levels, activity, and break statuses.
- Break Activity Suggestions: Fault-tolerant notifications providing actionable suggestions for breaks, even if other components fail.

3. User Flow Demonstration

The user flow incorporates event-driven processing to ensure responsiveness:

- Detection of Inactivity: Events captured by the productivity tracker initiate a notification.
- Real-Time Break Management: Follow-up notifications are issued if no action is taken.
- Hydration Tracking: Regular event-based prompts for water consumption with a status update for completion.

4. Requirements Prioritization

- Health Reminders: Minimizing burnout with accurate, actionable notifications.
- Fault Tolerance and Responsiveness: Ensuring reliability and real-time responses to user needs.
- Usability: Maintaining a user-friendly and intuitive interface.

5. Feedback Collection Plan

After presenting the prototype, we will gather feedback on:

- Effectiveness and timing of notifications, considering event-based triggers.
- Usability of the fault-tolerant notification system and user interface design.
- Suggestions for improved integration of productivity data for retrospectives.

This feedback will guide iterative improvements, leveraging Agile development principles for refinement.

Black Box Testing

	Test ID	Description	Expected Results	Actual Results
1	Productivity Tracking Validation Testing	Preconditions: Users have tasks to complete Steps: 1. User maintains various levels of productivity 2. Observers monitor the productivity rating of the user based on the data being obtained by the application Input: Eye-tracking data, Key-stroke data, productivity on JIRA tickets	Productivity dips when inactive and increases when active; when the productivity drops below a threshold, the user will be notified	
2	Access tasks posted on JIRA through the application Validation Testing	Preconditions: Tasks assigned to the test user on a test JIRA account that allows users to move the tickets amongst the different columns and update their progress Steps: 1. User moves task from backlog to In Progress 2. User moves task from In Progress to Completed	Users access the test tickets through a JIRA embedding that allows them to change progress on different tasks	

		Input: JIRA task data		
3	Ability to customize reminder notifications Integration Testing	Preconditions: User annoyed with popups Steps: 1. User use the system to modify notification location, size, etc. 2. Saving the edits will allow the notification system to be updated for future instances of the notification system Input: User notifications	Users are able to change the notification sounds/messages, updating them for future instances	
4	Break Timers Integrated Testing	Preconditions: User is starting a break Steps: 1. Break starts and a timer should display the time remaining on a break 2. Productivity tracker shuts off; observer checks to make sure it has turned off Input: Start/end time, break duration	Starting the break starts a timer; Break ends when the timer ends, starting the Work Timer and tracking the productivity of the user	
5	Work Timer Integrated Testing	Preconditions: User is starting to work Steps: 1. Break ends 2. User starts working again until another break is granted 3. Check to make sure productivity tracker is back to tracking data Input: Start/end time, distance	A notification should infrequently display the distance from the next break	

		from break		
6	Application starts up when user logs in and shuts down when user's workday ends Unit Testing	Preconditions: Developer starts or shuts down their computer Steps: 1. Program launches with startup, beginning work cycle. 2. Program closes with shutdown, logging data. 3. Check to ensure proper logging of events Input: Startup/shutdown signal sent from computer to program	Logging into laptop leads to the application starting up; Application shuts down at end of the workday	
7	Do not Disturb/Meeting mode Validation Testing	Preconditions: User wants privacy while working Steps: 1. User enables do not disturb mode. 2. Make sure icon notification works. 3. User can disable do not disturb mode. Input: User clicks Do Not Disturb button	Users are unable to be notified for breaks during this period	
8	Eye Tracking Validation Testing	Preconditions: Software is running, user is in front of computer, Camera is on and working Steps: 1. User ensures program is running and camera is on 2. User eyes follows a predetermined pattern 3. Eye tracking output is compared to expected results	Software can accurately track and record user's eye movements and locations	

		Input: User's eye movements		
9	Administrator Accounts Validation Testing	Preconditions: List of usernames, and which ones need administrator access, test User has admin privileges Steps: 1. User (with sufficient privileges) inputs usernames one by one, submitting them 2. User verifies that desired usernames are in admin list Input: Usernames of admins	List of administrators can be changed at will by user with sufficient privileges	
10	Admins can change start/end times Validation Testing	Preconditions: Using a test administrator account Steps: 1. Login to FocusBot on a test account with administrator privileges 2. Navigate to admin portal 3. Edit start/end times 4. Click 'submit' and observe the results Input: New desired start/end times	Start/end times can be altered by administrators	