AI Usage DISCLOSURE FORM

1. **Team Details**

Team Name: _____Coder4not4_____

Project / Product Name: _____ContextOS_____

Organization / Institution (if any): _____MSRIT_____

Submission Date: _____08-05-2026_____

2. **AI Usage Declaration**

Did your team use any Artificial Intelligence (AI) in developing this project? Yes / No

_____Yes_____

3. **Purpose of AI Usage (Brief Details)**

ChatGPT was used to brainstorm the initial product concept — exploring what a proactive, context-aware Android agent could look like and generating use-case scenarios (Busy Professional, Commuter, Traveler). Claude was used via Claude Code to refine the product architecture, develop the SituationModel framework, and produce the implementation roadmap for the 15-min background cycle.

Code generation or assistance ____OpenAI Codex and AntiGravity were the primary tool used to generate the core AI engine — including the OpenClawAgent, SituationModelBuilder, and the 6 skill implementations in Kotlin. Codex was also used in a second pass to compile and generate the final APK, assisting with Gradle build configuration, dependency management, and Jetpack Compose UI scaffolding.

UI / UX design ____Google Stitch was used to generate UI templates and design system components — including the notification card layouts, settings screen, and action log UI. Antigravity (Claude) was used to implement those templates into production-ready Jetpack Compose code, adapting the designs to Material Design 3 and the ContextOS design language.

Content creation ____No AI tools were used for content creation. All product copy, slide content, scenario narratives, and documentation were written entirely by the team.

Data analysis ____OpenAI Codex assisted with analyzing and processing contextual data collected from Android device APIs — including calendar events, GPS cluster patterns, battery telemetry, and usage logs. Codex helped design the data pipeline that feeds into the

SituationModelBuilder, defining how raw sensor data is structured, normalized, and scored for the OpenClaw AI layer.

Testing / debugging _____Cursor was used as the primary debugging environment — leveraging its AI-assisted code editor to identify runtime errors in the background service, resolve issues with WorkManager scheduling, fix cold-start failures, and validate the 15-min cycle budget (<15s total). Cursor's inline suggestions were particularly useful for debugging the Android Foreground Service lifecycle.

Other _____

4. **Feature Origin Classification (Please add/delete based on the number of the feature)**

1. Feature Name: Live SituationModel

Self-Generated/AI-Generated/Both: Both
Description: AI tools/platform used: OpenClaw AI, Gemini 2.0 Flash / Groq fallback, Android sensors, Calendar, GPS, battery, app usage. Prompt used: Analyze the user's current context every 15 minutes and identify what situation they are in, what action is useful, and why. Output summary: Creates a live context model such as meeting, commute, low battery, running late, or client-site mode. Modification: Connected sensor data with AI reasoning and added visible "Why?" explanations for every action.

2. Feature Name: Meeting Shield

Self-Generated/AI-Generated/Both: Both
Description: AI tools/platform used: Google Calendar API, Google Drive, Gmail, OpenClaw AI, Android DND controls. Prompt used: If a meeting is starting soon, prepare the phone by enabling DND, finding related files, opening meeting links, and explaining the reason. Output summary: Silences distractions, surfaces meeting documents, and prepares the user before the meeting starts. Modification: Added Samsung DeX angle, action log, and human-readable reasoning.

3. Feature Name: Traffic Rescue Drafts

Self-Generated/AI-Generated/Both: Both
Description: AI tools/platform used: Gemini / Groq, Calendar, location and ETA context, message drafting engine. Prompt used: If the user is running late, compare ETA with meeting time and draft a polite message to the meeting organizer. Output summary: Produces a ready-to-send "running late" message with one-tap send/edit/dismiss options. Modification: Final message is never auto-sent; the user stays in control.

4. Feature Name: Power Guardian

Self-Generated/AI-Generated/Both: Self-Generated
Description: AI tools/platform used: Android battery sensor, Calendar API, foreground service, rule-based fallback. Prompt used: Not AI-prompt based; the system checks if battery is low before a long meeting or important event. Output summary: Warns the user, reduces battery drain, and can prepare an emergency contact alert. Modification: Added as a proactive battery skill that triggers before the phone dies at the wrong time.

5. Feature Name: Memory Engine

Self-Generated/AI-Generated/Both: Both
Description: AI tools/platform used: OpenClaw AI, Room database, routine memory, preference memory, location memory. Prompt used: Learn from user routines, approvals, dismissals, locations, and repeated behavior to improve future recommendations. Output summary: ContextOS becomes smarter over time and adapts to the user instead of acting like a fixed automation app. Modification: Added local-first memory with clear-data control and Learn Mode as the default.

5. **Ethical & Compliance Confirmation**

AI usage complies with guidelines and policies. **Yes**

No proprietary or copyrighted data misused. **I Agree**

6. **Declaration & Sign-Off**

Name of Team Representative: Mukul Prasad

Role: Orchestration

Signature: MUKUL

Date:08-05-2026