TASCHED — Full System Specification and Architecture

0) Product Name and Purpose

Product Name: TaSched (Task Scheduler + Countdown Orchestrator)
Tagline (optional): "Schedule your day. Run it hands-free."
Primary Purpose: A desktop-first task scheduling application that allows users to plan and run **single or batched tasks** across up to **24 hours**, with **auto-run sequencing**, **warning popups**, **time-up alerts**, **sound profiles**, **minimize/restore**, and optional **screen ticker (scrolling message)** while the timer is running.
Core Promise: Users schedule tasks once, then TaSched executes them automatically with reminders—without blocking the user from doing other work.

1) Scope and Target Platforms

1.1 Phase Scope (TaSched v1.0)

Primary platform: Windows Desktop (Python + Tkinter)
App type: GUI desktop app, offline-first
Storage: Local JSON + optional local SQLite (recommended)

1.2 Future Scope (Roadmap)

- Cross-platform Desktop: macOS + Linux (Phase 2)
- Mobile companion: Android + iOS (Phase 3)
- Cloud sync + multi-device schedule sharing (Phase 4)

2) User Roles and Access Control

2.1 Roles

- **Standard User:** Create and run schedules, configure alerts, manage templates.
- **Admin (optional for enterprise):** Lock settings, enforce UI policies, manage shared templates.

2.2 Authentication (v1.0)

- Not required (single-user desktop tool).
- Enterprise edition can add optional PIN lock for settings.

3) Functional Requirements

3.1 Task Scheduling

3.1.1 Task Creation

User can create a task with:

- Task name/title
- Duration (minutes/seconds)
- Mode:
  - **Relative/Sequential:** starts after previous task ends
  - **Absolute/Clock-based:** starts at a specific time today
- Repeat:
  - None
  - Daily
  - Custom (Mon–Sun)
- Alerts:
  - Warning points (e.g., 10 min, 5 min, 1 min)
  - Time-up behavior (fullscreen time-up panel or popup)
- Sound profile:
  - Alert sound (time-up)
  - Warning sound (pre-alert)

    - Optional “background music” during breaks/tasks
- Display options:
    - Show countdown window or mini timer
    - Enable ticker message (scrolling text), choose direction and position

3.1.2 Batch Scheduling

User can create a **Schedule** comprising multiple tasks:

- Tasks can be reordered
- Tasks can be duplicated
- Batch can be saved as a template
- Batch execution is automatic:
    - Task 1 → Task 2 → Task 3 … until completion

3.1.3 24-Hour Constraint

Schedules must not exceed 24 hours within a day for v1.0:

- Sequential tasks: sum(duration) must be ≤ 24 hours
- Absolute tasks: must fall within today’s date, with validation for conflicts

3.2 Run Engine (Execution)

3.2.1 Schedule Execution Modes

- **Run Now:** executes immediately from first task
- **Run From Selected Task:** start at any task index
- **Auto-Start at Clock Time:** schedule begins at a chosen time

3.2.2 Auto-Run Behavior

When a task completes:

- Log completion
- Fire time-up alert
- Start next task automatically after:
    - immediate (default), or
    - configurable “gap” (e.g., 10 seconds)

3.2.3 Pause/Resume/Skip

While running:

- Pause current task
- Resume
- Skip to next task
- End schedule early

3.3 Warnings and Alerts

3.3.1 Warning Popups

As countdown approaches warning points:

- Popup shows task name + remaining time + next task (optional)
- Popup behavior:
    - auto-dismiss after X seconds
    - optional sound
    - optional “Snooze 1 min”

3.3.2 Time-Up Alert

At 00:00:

- Primary time-up window:
    - fullscreen overlay (optional)
    - WAEC-themed UI (optional theme pack)
    - background image (Flora / WAEC_Background)

    - sound/music playback
- User dismiss action:
    - Enter / ESC
- Auto-close after configured duration (e.g., 15 seconds)

3.4 UI Behavior Requirements

3.4.1 Setup Window Auto-Hide

When user starts a schedule:

- Setup window should **withdraw (hide)** automatically
- Running timer window becomes active

3.4.2 Minimize / Restore

While running:

- User can minimize timer window to continue other work
- App keeps running in background
- Restore via:
    - taskbar click
    - optional hotkey (future)
    - optional tray icon (Phase 1.1 or Phase 2)

3.4.3 Horizontal Scrolling Message (Ticker)

When timer is running:

- User can display a scrolling message across screen:
    - direction: left→right or right→left
    - position: top or bottom
    - speed: slow/medium/fast
    - text source:
        - "Current Task: X"
        - "Next Task: Y"
        - Custom message

3.5 Templates and Presets

3.5.1 Templates

Users can save a batch of tasks as a template:

- Meeting Day template
- Study Sprint template
- WAEC session template
- Pomodoro template

3.5.2 Presets (Quick Add)

One-click presets:

- Pomodoro 25/5 cycles (configurable)
- Deep Work 50/10
- Lunch break 60
- Tea break 15

3.6 Logging and History

3.6.1 Run Log

Log entries:

- schedule started/ended
- task started/ended
- pause/resume events

- skipped tasks
- missed absolute start times (if user was late)

3.6.2 Export

Export history as:

- CSV (v1.0)
- JSON (v1.0)
- PDF summary (future)

3.7 Settings

- Theme selection (WAEC theme vs Corporate theme vs Indigenous theme if you want)
- Default warning points
- Default sound profile
- Default time-up display mode (fullscreen/popup)
- Default auto-start next task setting
- Default ticker settings

4) Non-Functional Requirements

4.1 Reliability

- Timer must remain accurate under minimize/background
- Must survive window hide/show
- Must recover gracefully from sound device failures

4.2 Performance

- Low CPU usage (ticker must be efficient)
- No UI freezing; timer updates via after() loop

4.3 Usability

- Clear, big timer display
- Fast schedule building
- Keyboard shortcuts (Phase 1.1)
- Accessible color contrast

4.4 Security

- Local-only by default
- No network operations required
- Optional settings PIN for enterprise edition

4.5 Maintainability

- Modular architecture (core engine separate from UI)
- Clear data models and storage layer

5) System Architecture

5.1 High-Level Architecture (Component View)

A) Presentation Layer (UI)

- **Setup UI** (Schedule Builder)
- **Run UI** (Timer window)
- **Time-Up UI** (Alert window)
- **Warning UI** (Popup notifications)
- **Ticker UI** (Scrolling overlay)

B) Application Layer (Controllers)

- ScheduleController
- RunController
- AlertController

- SettingsController

C) Domain Layer (Core)

- Task model
- Schedule model
- Scheduler engine (state machine)
- Warning engine
- Time services

D) Infrastructure Layer

- Storage (JSON/SQLite)
- Audio service (pygame mixer)
- Logging service
- Resource manager (assets)

5.2 Core Modules (Recommended File Structure)

```
tasched/
  app.py                    # entry point
  ui/
    setup_window.py
    run_window.py
    timeup_window.py
    warning_popup.py
    ticker_overlay.py
  core/
    models.py               # Task, Schedule, Settings
    scheduler_engine.py     # state machine + run loop
    warning_engine.py       # threshold checks + events
    time_service.py         # clock/format helpers
  services/
    audio_service.py        # pygame mixer wrapper
    storage_service.py      # json/sqlite persistence
    log_service.py          # run logs
    resource_service.py     # asset discovery / pyinstaller support
  assets/
    WAEC_Background.png
    WAEC_Logo.webp
    WAEC_Tone.mp3
    WAEC_Icon.ico
  data/
    schedules.json
    templates.json
    settings.json
    logs.txt
```

You can keep it single-file for now, but the architecture above is the “adult version” that won’t collapse later.

6) Scheduler Engine Design

6.1 State Machine

Schedule States

- IDLE

- READY
- RUNNING
- PAUSED
- COMPLETED
- CANCELLED

Task States

- PENDING
- ACTIVE
- PAUSED
- COMPLETED
- SKIPPED

6.2 Execution Loop

- Uses tk.after(1000, tick) style timing (UI safe)
- Each tick:
  - decrement remaining seconds
  - update UI
  - evaluate warning thresholds
  - dispatch warning events
  - at 0 → dispatch time-up event → task completion → advance to next

7) Data Model Specification

7.1 Task Object (JSON Example)

```
{
 "id": "task_001",
 "title": "Report Writing",
 "duration_seconds": 3600,
 "mode": "sequential",
 "absolute_start_time": null,
 "repeat": "none",
 "warning_points_seconds": [600, 300, 60],
 "sound_profile": {
  "warning_sound": "WAEC_Tone.mp3",
  "timeup_sound": "WAEC_Tone.mp3"
 },
 "display": {
  "fullscreen_timeup": true,
  "ticker_enabled": true,
  "ticker_text": "Current Task: Report Writing",
  "ticker_position": "bottom",
  "ticker_direction": "left",
  "ticker_speed": "medium"
 }
}
```

7.2 Schedule Object

```
{
 "id": "schedule_2026_01_05",
 "name": "Workday Sprint",
 "date": "2026-01-05",
 "tasks": ["task_001", "task_002", "task_003"],
```

```
  "auto_start": true,
  "auto_advance": true,
  "created_at": "2026-01-05T09:00:00"
}
```

7.3 Settings Object

- default warning points
- default theme
- default sounds
- default ticker settings

8) Workflows

Workflow A — Create and Run Batch Tasks

1. User opens TaSched setup window
2. Adds multiple tasks to schedule
3. Enables “Auto-run next task”
4. Clicks “Start Schedule”
5. Setup window hides automatically
6. Run window appears and starts task 1
7. Warnings show at thresholds
8. Time-up triggers, auto-advances to task 2
9. After last task, schedule completes and summary is shown

Workflow B — Absolute Scheduled Task

1. User creates task with clock time (e.g., 14:30)
2. Adds it to schedule
3. TaSched waits until time, then starts automatically
4. If user is late (app closed/off), logs “missed start” and prompts on next open

Workflow C — Minimize/Restore While Running

1. User minimizes timer window
2. Engine continues running in background
3. Warning popup appears even when minimized
4. User restores timer when needed

9) Packaging and Distribution

9.1 Windows EXE (PyInstaller)

- Embed icon
- Bundle assets
- Include version metadata

9.2 MSI Installer (Next Phase)

- Use WiX Toolset or Advanced Installer
- Add Start Menu shortcut
- Optional auto-start on login

10) Future Enhancements (Your “blow my mind” list, but architecture-ready)

- Pomodoro cycles generator (25/5, 50/10, custom)
- Day planner timeline view (drag/drop schedule)
- Tray icon controls (Pause/Resume/Next)
- Hotkeys (global)
- Analytics dashboard (weekly focus time)
- Multi-profile (personal / work)

- Calendar import (ICS)
- Notifications integrated with Windows toast