Systems Programming Final Project – Week 1 Proposal & Planning

Ye Khaung Soe – 68011975

Theint No No Aung - 68011567

Project Title

Lightweight System Resource Monitor

Project Idea

We are going to build a desktop application in **Rust,** using the **Iced GUI framework,** that displays real-time system resource usage. The app will include three functional screens:

- **Overview**: CPU, memory, disk usage, and system uptime
- **Processes**: Read-only process list with search/filter by name or PID
- **Settings**: Refresh interval and theme (light/dark), saved to a config file

The design emphasizes **stability and safety,** by avoiding destructive actions (e.g., killing processes). Instead, we implement safe state modification through configurable settings that persist across sessions.

Target OS

- **Windows 11** (primary target)
- Will try to implement cross-platform support, using **Docker**, if time allows

---

System Interaction

- Uses the sysinfo crate to retrieve CPU, memory, disk, and process data
- No elevated permissions required for read-only monitoring
- Configuration file stored in:

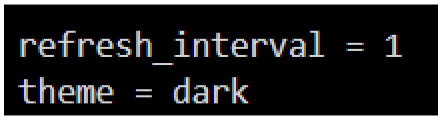
  %APPDATA%\LightMon\config.toml

System State Modification

- Users can adjust refresh interval and theme (light/dark)
- Settings are saved and reloaded on startup

Persistent Settings

- Config file format (.toml):



```
refresh_interval = 1
theme = dark
```

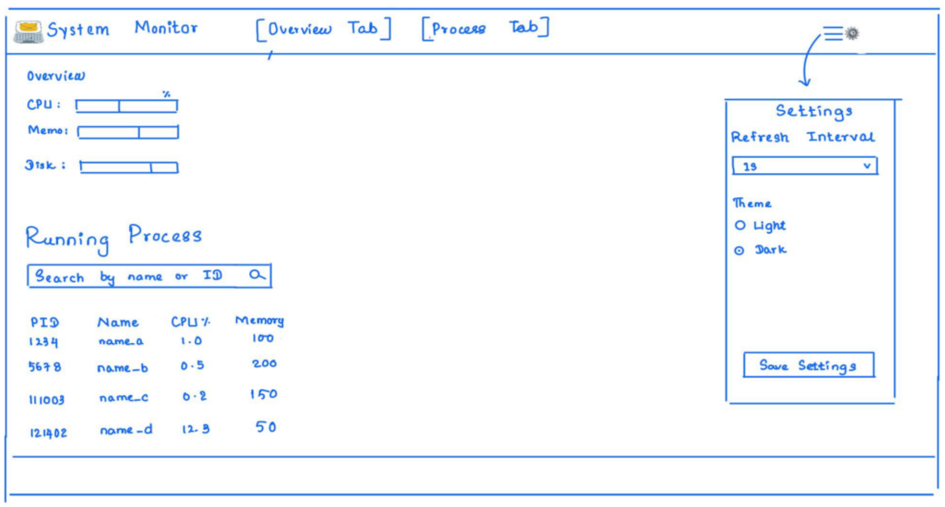
- Format will be documented in final deliverables

Risks & Mitigation

- **Rust/Iced learning curve** -> Begin with minimal working shell app
- **Async refresh logic** -> Use Iced subscriptions for safe background updates
- **Cross-platform complexity** -> Scope limited to Windows only for v1.0
- **Scope creep** -> Stick to monitoring; no process management in this version

---

UI Wireframe:



PID	Name	CPU %	Memory
1234	name_a	1.0	100
5678	name_b	0.5	200
111003	name_c	0.2	150
121402	name_d	12.3	50

Work Plan

- **Week 1 — Proposal & Planning**
  Write the project proposal, create a UI sketch, and finalize the work plan.
- **Week 2 — Architecture & Spike**
  Set up a minimal Iced shell app with mock data and produce the ARCHITECTURE.md file.
- **Week 3 — Data Layer**
  Connect to real system statistics using the sysinfo crate and add unit tests for data retrieval.
- **Week 4 — Core UI**
  Build the Overview and Processes screens, displaying live data updates.
- **Week 5 — Settings & Persistence**
  Add a Settings screen, implement configuration persistence (refresh interval and theme).
- **Week 6 — Testing & Hardening**
  Write integration tests, improve error handling, and add structured logging.
- **Week 7 — Beta & Documentation**
  Release a beta version, write the USER_GUIDE and TEST_PLAN, and include screenshots.
- **Week 8 — Release & Demo**
  Finalize the build, conduct the live demo, and write the postmortem report.