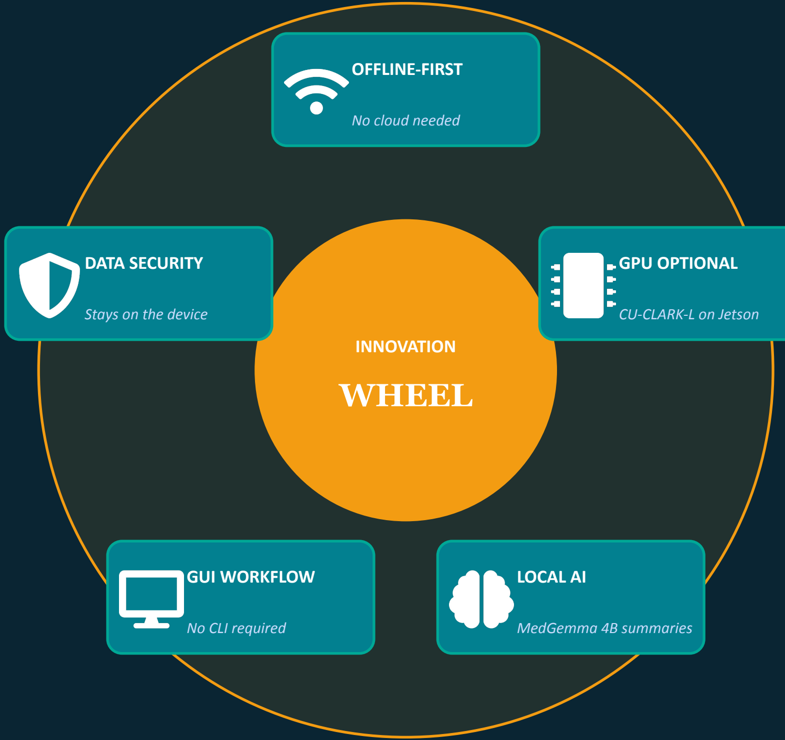
BILKENT CS491/2 • T2504 • FINAL IMPLEMENTATION

Pathogenius

Portable, offline-capable metagenomic pathogen analysis

THE ELEVATOR PITCH

Sequencing-based pathogen detection today depends on the cloud, HPC clusters, and bioinformaticians, none of which exist in the field. Pathogenius runs the entire pipeline on a consumer laptop, offline, with an on-device AI model that explains the findings.



INNOVATION

WHEEL

OFFLINE-FIRST

No cloud needed

DATA SECURITY

Stays on the device

GPU OPTIONAL

CU-CLARK-L on Jetson

GUI WORKFLOW

No CLI required

LOCAL AI

MedGemma 4B summaries

Team

Nazlı Apaydın • Ege Ateş • Yiğit Ali Doğan • Yunus Günay • Ata Uzay Kuzey

Supervisor: Can Alkan | Instructors: Mert Bıçakçı, İlker Burak Kurt | May 2026

Project Status Dashboard

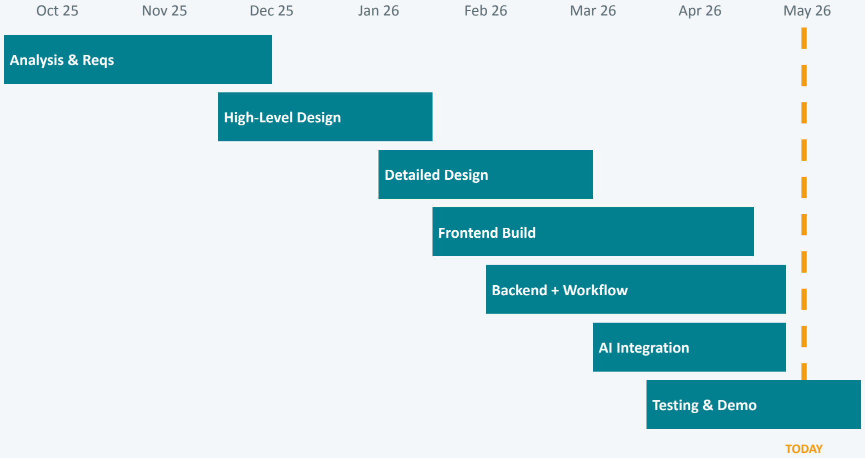
Final Implementation milestone, May 2026

✓ DELIVERED

SUBSYSTEM PROGRESS

Requirements & Design	100%
Frontend (Electron GUI)	100%
Backend & API	100%
Snakemake Workflow	100%
AI / Interpretation	100%
Testing & Integration	100%

PROJECT TIMELINE (GANTT)



74
Test cases executed

7
Main process services

3
Snakemake rules

4
Chart types delivered

HEALTH CARD

●	SCHEDULE	**On track**
●	SCOPE	**Stable**
●	QUALITY	**66 Tests Passed, 8 Failed**
●	TEAM	**5 / 5 active**
●	RISKS	**Resolved**
●	BLOCKERS	**None**

The Problem

Why pathogen analysis breaks down outside the central lab

Long-read sequencing can now identify almost any pathogen from a sample. But existing pipelines assume HPC clusters, cloud connectivity, and a bioinformatician, none of which exist in the field.

Cloud Dependence

Tools assume cloud. The field has none

HPC-Only Workflows

Pipelines built for servers, not laptops

CLI-Only Interfaces

Shell scripts lock non-experts out

Privacy Risks

Samples contain human DNA, uploads aren’t acceptable

THE EXISTING LANDSCAPE

RT-qPCR / ddPCR kits	*Target-driven; can't discover unknown pathogens; lab infrastructure required.*
Nanometa Live & similar dashboards	*Real-time, but still CLI-driven and tied to BLAST validation latency.*
Academic Nanopore pipelines	*Locally curated, but research-only; manual scripts, no GUI, no AI summary.*

How Pathogenius Solves It

From raw FASTQ on a laptop to a clinical-style summary, fully offline

Sample Input

Long-read FASTQ
(.fastq, .fastq.gz)

→

Preprocessing

QC, filtering,
read-only inputs

→

Classification

k-mer matching
CLARK-l or CU-CLARK-L

→

Aggregation

Abundance,
confidence scores

→

AI Summary

Plain-language
clinical summary

Offline Capable

Electron app. CLARK-l runs in the laptop. Optional Jetson Nano over SSH adds GPU acceleration.

Stays Local

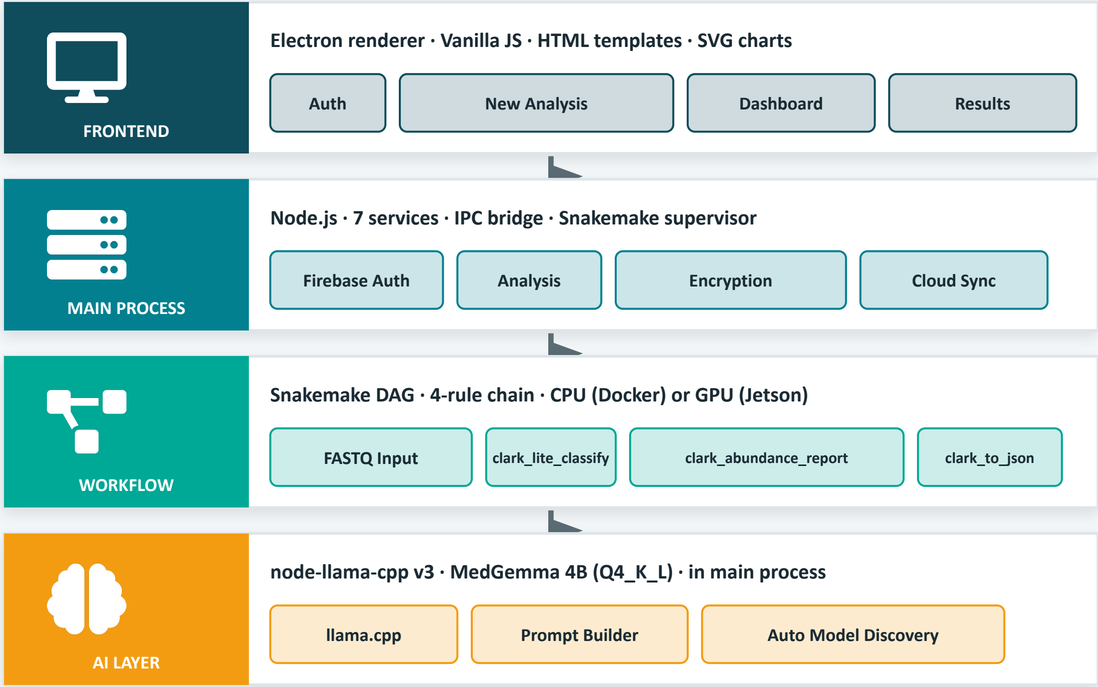
AES-256-GCM at rest. Never stored. FASTQ read-only. Cloud sync uploads ciphertext..

Speaks Plainly

MedGemma 4B (Q4_K_L) runs in the main process via node-llama-cpp. Turns pathogen JSON into a clinical summary.

System Architecture

Four cooperating layers, delivered and running on a single device



FRONTEND

Electron renderer · Vanilla JS · HTML templates · SVG charts

- Auth
- New Analysis
- Dashboard
- Results

MAIN PROCESS

Node.js · 7 services · IPC bridge · Snakemake supervisor

- Firebase Auth
- Analysis
- Encryption
- Cloud Sync

WORKFLOW

Snakemake DAG · 4-rule chain · CPU (Docker) or GPU (Jetson)

- FASTQ Input
- clark_lite_classify
- clark_abundance_report
- clark_to_json

AI LAYER

node-llama-cpp v3 · MedGemma 4B (Q4_K_L) · in main process

- llama.cpp
- Prompt Builder
- Auto Model Discovery

LOCAL STORAGE

Accessed by all four layers

FASTQ Inputs
10 MB — 10 GB · read-only

CLARK Reference DB
Built on first use · targets.txt

Encrypted Results
AES-256-GCM · per-analysis JSON

Analysis History
JSON index · plaintext metadata

OS Keychain
keytar · auth session (idToken, refreshToken)

Firebase (optional)
Firestore metadata · Storage ciphertext

Each layer is independently deployable, testable, and replaceable. The Snakemake workflow can be swapped without touching the GUI; the MedGemma model can be replaced by dropping a new GGUF file into frontend/models. The LLM service auto-discovers it at startup.

Technology Stack & Deployment

How the architecture maps onto a single consumer-grade machine

STACK BY LAYER

FRONTEND	Electron · Vanilla JavaScript · HTML templates · hand-rolled SVG charts
MAIN PROCESS	Node.js+ · IPC bridge · 7 services · Snakemake supervisor · keytar
WORKFLOW	Snakemake · CLARK-l (Docker) · CU-CLARK-L (Jetson) · Python
AI LAYER	node-llama-cpp v3 · MedGemma 4B · GGUF Q4_K_L
STORAGE	AES-256-GCM (Node crypto) · Local encrypted blobs · Firebase Storage + Firestore (optional)

HARDWARE / SOFTWARE MAPPING

MID-RANGE LAPTOP / JETSON NANO

CPU

Electron · Main process · Snakemake · CLARK-l (Docker) · MedGemma inference

RAM

CLARK in-memory index · GGUF model context · workflow buffers

GPU

CU-CLARK-L on Jetson Nano · SSH dispatch · batch=128 (avoids watchdog)

LOCAL DISK

FASTQ · CLARK reference DB · encrypted JSON · GGUF model · history index

No external compute servers required. The Electron app runs entirely on the user's machine. GPU mode dispatches CU-CLARK-L over SSH to a Jetson Nano (any reachable network; direct Ethernet, LAN, or VPN). Optional Firebase cloud sync stores result bodies as AES-256-GCM ciphertext; per-analysis metadata is stored in Firestore alongside.

Team & What We Delivered

Five contributors, focused ownership, end-to-end shipping

YAD

Yiğit Ali Doğan

Architecture Lead

System architecture, hardware/software mapping, CPU/GPU analysis workflows, batching

EA

Ege Ateş

Backend & Workflow

Snakemake workflow, CLARK reference database builder, CPU/GPU dispatch

NA

Nazlı Apaydın

Frontend & UX Lead

Requirements analysis, Electron renderer UI, theme system, cloud system, encryption schemes

AUK

Ata Uzay Kuzey

Reporting & AI Integration

Documentation, web reports, MedGemma 4B integration via node-llama-cpp v3

YG

Yunus Günay

Testing & Integration Lead

Testing, integration, Dockerized CLARK environment, chart visualizations, integration

WHAT WE SHIPPED

Mar — Apr ✓	**IPC bridge & renderer wiring**	*Renderer ↔ main process IPC bridge live; seven namespaced services routed through window.api; live Snakemake progress streaming.*
Mar — Apr ✓	**End-to-end pipeline running**	*Full FASTQ → CLARK-l (Docker) or CU-CLARK-L (Jetson Nano via SSH) → JSON results, with three Snakemake rules and AES-256-GCM at rest.*
Apr ✓	**AI summary integrated**	*MedGemma 4B (Q4_K_L) loaded by node-llama-cpp v3 in the main process; clinical summaries rendered above the chart suite on the Results page.*
May ✓	**Test execution & final demo**	*67 test cases executed across 12 categories (AM, FP, DM, NT, UA, RT, GPU, USE, REL, PER, SUP, SCA); demo build delivered.*

Pathogenius — genomic pathogen detection delivered, out of the data center and onto a laptop.