Contact

[Your phone]
ruyangearnold@gmail.com

[Your city, country]

arnold-rho.vercel.app

GitHub

SKILLS

Frontend

React

Next.js

TypeScript

Astro

Tailwind CSS

Backend

Node.js

Python

Java

PostgreSQL

MongoDB

Mobile

React Native

Expo

Cross-platform UI

Mobile APIs

AI / ML

TensorFlow

CNN

Machine Learning

Python

Data Preprocessing

DevOps

Docker

AWS

GitHub Actions

MQTT / IoT

Security

Cybersecurity

Penetration Testing

Network Security

OWASP

LANGUAGES

English (Fluent)

[Add language]

CERTIFICATIONS

Cybersecurity Fundamentals

Self-Directed Study · 2024

Full-Stack & AI Project Portfolio

Project-Based Learning · 2025

RUYANGE ARNOLD

FULL-STACK DEVELOPER

PROFILE

Full-stack developer with 3+ years of hands-on experience building Java, Python, AI/ML, and web applications. I design and ship end-to-end systems — from desktop GUIs and REST APIs to CNN models and IoT pipelines.

Portfolio of RUYANGE Arnold — focused on practical software engineering through open-source work on GitHub, project-based learning, and production-ready user experiences.

WORK EXPERIENCE

University / Self-Directed Learning 2023 - Present

Computer Science Student

Studying computer science fundamentals while building personal projects in Java, Python, AI/ML, and web development. Focus on practical software engineering through open-source work on GitHub.

Java

Python

Next.js

TensorFlow

SQL

Personal Projects & Open Source 2023 - Present

Independent Developer

Designed and shipped projects including an e-commerce platform (arnshop), handwriting recognition CNN, MQTT IoT weather system, web scraping pipelines, and this portfolio site.

Java

Python

Flask

Docker

Machine Learning

EDUCATION

Computer Science 2023 - Present

University / Self-Directed Learning · Remote

Studying computer science fundamentals while building personal projects in Java, Python, AI/ML, and web development. Focus on practical software engineering through open-source work.

FEATURED PROJECTS

Selected work across full-stack, backend, AI/ML, DevOps, and mobile-focused development. Each project includes the problem, implementation, outcome, and repository link.

FULLSTACK PROJECTS

Ingoboka Platform — Digital Microinsurance PWA

Problem: Microinsurance products need a mobile-first, accessible frontend for citizens to browse coverage and complete flows without heavy native apps.

Built: Next.js 14 App Router PWA with TypeScript, responsive layouts, and API-driven insurance workflows designed for real-world deployment.

Outcome: Production-grade PWA frontend ready to integrate with insurance backend services.

Tech: Next.js, TypeScript, PWA, React, Tailwind CSS

GitHub: https://github.com/Arn-The-Wolf/ingoboka-platform

Key results

Next.js 14 PWA

Citizen-facing insurance UI

Production-ready frontend

Challenges solved

PWA offline behavior

Complex form flows

Mobile-first UX

BestBuy Electronics — Full-Stack E-Commerce

Problem: Electronics retail needed a modern storefront with a scalable backend for products, orders, and customer data.

Built: Split repos: responsive React storefront with product catalog & checkout; Next.js API backend with Supabase for auth, inventory, and orders.

Outcome: End-to-end online store with separated frontend/backend deployment.

Tech: TypeScript, React, Next.js, Supabase, E-Commerce

GitHub: https://github.com/Arn-The-Wolf/bestbuyelectronics_frontend

Key results

React e-commerce UI

Next.js + Supabase API

Product & order management

Challenges solved

Supabase schema design

Cart & checkout flow

Responsive storefront

AviaServe — Airline Reservation System

Problem: Airline booking flows need reliable search, seat selection, and reservation management across routes.

Built: Full-stack TypeScript application with route search, booking logic, and reservation management.

Outcome: Complete reservation workflow for multi-route airline scenarios.

Tech: TypeScript, Node.js, React, REST API, PostgreSQL

GitHub: https://github.com/Arn-The-Wolf/aviaserve

Key results

Multi-route reservations

TypeScript full stack

Booking management UI

Challenges solved

Route & schedule modeling

Booking concurrency

Search performance

ArnShop Full Stack — Java + React E-Commerce

Problem: Needed a complete e-commerce system with a modern web frontend and robust Java backend for electronics retail.

Built: React client connected to Java REST services with MySQL persistence, inventory controls, and admin order management.

Outcome: Production-style e-commerce split across modern frontend and Java API layer.

Tech: Java, React, MySQL, REST API, E-Commerce

GitHub: https://github.com/Arn-The-Wolf/arnshop_fullstack

Key results

React + Java full stack

Product catalog & cart

Order management

Challenges solved

Frontend/backend integration

Inventory sync

Checkout reliability

Impact metrics

Features: 0 → Full cart + checkout (100%)

GUI Screens: N/A → Multi-panel UI (Complete)

Data Layer: None → MySQL integrated (Production-ready)

ArnLibrary — Library Management System

Problem: Small libraries needed a simple system to track books, students, and borrowing without complex enterprise software.

Built: Java Swing GUI with MySQL persistence for books, students, due dates, and admin operations.

Outcome: End-to-end library workflow with CRUD operations.

Tech: Java, Swing, MySQL, GUI

GitHub: https://github.com/Arn-The-Wolf/arnlibrary

Key results

Full library workflow

Interactive admin panel

Challenges solved

CRUD operations

GUI design

Data persistence

ARNOLD.DEV Portfolio — Personal Developer Portfolio

Problem: Needed a professional online presence to showcase projects, skills, and case studies with fast performance and a distinctive modern UX.

Built: Full-stack portfolio with Next.js 14, TypeScript, Tailwind CSS, Three.js visuals, JSON-backed CMS APIs, and Vercel deployment.

Outcome: Live portfolio at arnold-rho.vercel.app with project showcase, skills arsenal, blog, case studies, and contact flow.

Tech: Next.js, TypeScript, Tailwind CSS, React, Three.js, Vercel

GitHub: https://github.com/Arn-The-Wolf

Key results

Live production site

CMS-driven content

Performance-optimized UX

Challenges solved

3D hero performance

Content management

Responsive design

BACKEND PROJECTS

WASAC & REG Unified Billing — Enterprise Java Billing Backend

Problem: Water and electricity postpaid billing required automated unified invoicing across WASAC and REG systems.

Built: Spring Boot services for billing logic, customer accounts, invoice generation, and unified postpaid workflows.

Outcome: Unified billing automation replacing manual cross-utility reconciliation.

Tech: Java, Spring Boot, REST API, MySQL, Enterprise

GitHub: https://github.com/Arn-The-Wolf/wasac-reg-unified-billing-system

Key results

Unified WASAC/REG billing

Spring Boot backend

Automated postpaid flows

Challenges solved

Multi-utility billing rules

Invoice accuracy

Enterprise data modeling

MongoDB Transaction Manager — ACID Transactions Demo

Problem: Multi-document updates in MongoDB require transactions to stay consistent — needed a clear reference implementation.

Built: Node.js service with MongoDB session-based transactions, rollback handling, and example transfer workflows.

Outcome: Reusable pattern for transactional MongoDB operations in production APIs.

Tech: JavaScript, Node.js, MongoDB, ACID, Backend

GitHub: https://github.com/Arn-The-Wolf/transactional-management

Key results

MongoDB transactions

Commit/abort flows

Transfer demo

Challenges solved

Session lifecycle

Rollback edge cases

Replica set setup

Web Scraping Pipeline — Python Data Extraction

Problem: Manual data collection from multi-page websites was slow and error-prone for analysis workflows.

Built: Python pipeline: paginated crawling, HTML parsing, record normalization, and JSON/CSV export.

Outcome: Automated extraction pipeline producing clean structured output.

Tech: Python, BeautifulSoup, Web Scraping, Data Extraction

GitHub: https://github.com/Arn-The-Wolf/webscraping

Key results

Automated data pipeline

Clean structured export

Defensive parsing

Challenges solved

Multi-page navigation

Data structuring

Rate limiting

BFS & DFS Algorithms — Graph Traversal Implementation

Problem: Understanding graph traversal strategies for pathfinding and connected-component problems.

Built: Java BFS/DFS with adjacency structures, traversal logic, and performance comparison.

Outcome: Correct, tested traversal implementations demonstrating algorithm trade-offs.

Tech: Java, Algorithms, Data Structures

GitHub: https://github.com/Arn-The-Wolf/Breadth-First-Searching-Deep-First-Searching-

Key results

Working BFS/DFS implementations

Challenges solved

Algorithm optimization

Graph representation

AI / ML PROJECTS

Clause Lens — Insurance Policy Clause Extractor

Problem: Insurance policies are dense PDFs — extracting coverage facts, exclusions, and limits manually is slow, inconsistent, and error-prone.

Built: Full pipeline: PDF ingest, clause chunking & classification, concurrent Claude JSON extraction, embeddings, pgvector cosine search, labeled eval benchmark, and a built-in web UI.

Outcome: End-to-end policy analyzer with retrieval accuracy targets above 80% and field-level extraction scoring on a labeled benchmark set.

Tech: TypeScript, Node.js, Claude API, PostgreSQL, pgvector, Python, Express, Docker, LLM, RAG

GitHub: https://github.com/Arn-The-Wolf/clause-lens

Key results

PDF → chunk → extract → embed pipeline

Semantic search with clause-type filters

Labeled evaluation report

Challenges solved

PDF clause segmentation

Structured LLM extraction at scale

Vector retrieval accuracy

ArcFace Face Recognition — 5-Point Alignment + ONNX Inference

Problem: Face recognition systems often depend on GPU infrastructure — needed an accurate CPU-first pipeline for enrollment and verification.

Built: Python pipeline: face detection, landmark alignment, ONNX ArcFace inference, embedding comparison, and enrollment/verification workflow.

Outcome: Portable face recognition stack suitable for edge and server deployment.

Tech: Python, ONNX, ArcFace, OpenCV, Computer Vision, AI

GitHub: https://github.com/Arn-The-Wolf/face-recognition-5pt-arcface-onnx

Key results

ArcFace ONNX pipeline

5-point alignment

Enrollment & verification flow

Challenges solved

CPU inference performance

Alignment accuracy

Embedding threshold tuning

ANPR — Plate Detection & OCR — Automatic Number Plate Recognition

Problem: Manual plate reading does not scale for parking, security, or traffic monitoring use cases.

Built: Python CV pipeline combining detection models, alignment, OCR, and structured plate text output (ANPR-Q11).

Outcome: Working ANPR pipeline from raw images to readable plate strings.

Tech: Python, OpenCV, OCR, Computer Vision, YOLO, AI

GitHub: https://github.com/Arn-The-Wolf/ANPR-Detection-Alignment-OCR

Key results

Detection + alignment + OCR

Structured plate output

End-to-end ANPR flow

Challenges solved

Plate detection accuracy

OCR on skewed plates

Real-world lighting

Face Tracker Live Dashboard — AI + IoT Recognition System

Problem: Live environments need face recognition tied to hardware actuators with a real-time monitoring dashboard.

Built: Python recognition stack, MQTT/device integration, and a dashboard for live tracking events.

Outcome: Integrated CV + IoT demo suitable for access control and monitoring prototypes.

Tech: Python, OpenCV, MQTT, ESP32, Computer Vision, IoT

GitHub: https://github.com/Arn-The-Wolf/face-tracker-live-dashboard

Key results

Live face tracking

ESP32 integration

Monitoring dashboard

Challenges solved

Real-time inference

Hardware integration

Dashboard latency

Handwriting Recognition AI — CNN Machine Learning Model

Problem: Classifying handwritten digits manually does not scale — needed a trained model for automated recognition.

Built: Python CNN: dataset preprocessing, model training, evaluation metrics, and inference on new samples.

Outcome: Working classifier with extensible training pipeline.

Tech: Python, TensorFlow, CNN, Machine Learning, AI

GitHub: https://github.com/Arn-The-Wolf/Handwriting-Recognition-Ai

Key results

Working CNN model

Digit classification

Extensible training pipeline

Challenges solved

Model training optimization

Dataset preprocessing

Overfitting control

Impact metrics

Model Type: None → CNN classifier (Deployed)

Dataset Handling: Raw → Preprocessed (Optimized)

Accuracy: Baseline → High confidence (Strong)

Print Text Scanner — OCR Desktop + Web App

Problem: Extracting text from scanned documents and images required a unified tool for desktop and web users.

Built: Python OCR engine with GUI and web frontends, image preprocessing, and export to text formats.

Outcome: Dual-interface OCR tool for document digitization workflows.

Tech: Python, OCR, Tesseract, Flask, Computer Vision

GitHub: https://github.com/Arn-The-Wolf/Print_Text_Scanner

Key results

Desktop + web OCR

Text export

Document scanning flow

Challenges solved

Image preprocessing

OCR accuracy

Cross-platform UI

DEVOPS PROJECTS

MQTT Weather App — IoT Weather Monitoring

Problem: Weather sensor data was scattered across devices with no unified real-time view or reliable ingestion path.

Built: Flask API subscribing to MQTT topics, processing payloads, and exposing a live weather dashboard. Dockerized.

Outcome: Live MQTT pub/sub pipeline with working environmental dashboard.

Tech: Python, Flask, MQTT, IoT, Docker

GitHub: https://github.com/Arn-The-Wolf/mqtt-weather-app

Key results

Live weather dashboard

MQTT pub/sub pipeline

Docker deployment

Challenges solved

MQTT broker integration

Real-time processing

Sensor calibration

Impact metrics

Data Pipeline: Manual → MQTT live stream (Real-time)

Backend: None → Flask + MQTT (Complete)

Deployment: Local → Dockerized (Portable)

MOBILE — SKILLS & FOCUS

Mobile development is an active focus area. Current stack centers on React Native and Expo for cross-platform apps, with API-first backends from my full-stack work.

Technologies

React Native

Expo

Cross-platform UI

Mobile APIs

No dedicated mobile app repositories in the portfolio yet — upcoming work will extend existing backends (Flask, Node.js) with React Native clients.