CURRICULUM VITAE / 2026

Yassine Benomar



FULL-STACK SOFTWARE ENGINEER

yassine.benomar1@gmail.com · +33 7 58 17 41 88 · Nice, France

yassine-benomar.pages.dev · linkedin.com/in/yassine-benomar-11a8611b3

OPEN TO REMOTE ROLES · CET

01 PROFILE

Full-stack software engineer with 5+ years shipping production systems end to end: web frontends (React, Vue), backend services and internal tools, down to the embedded Linux devices they run on. One of four engineers on Hanover Displays' onboard stack: I built the web tool that cut provisioning errors to near zero, shipped passenger systems in daily service across French cities, and unified two product lines onto one codebase. Used to working remotely on hardware I can't touch.

5+	2	3	965
YEARS IN PRODUCTION	PRODUCT LINES UNIFIED	SITES IN DAILY SERVICE	TOEIC — ENGLISH C1

02 TECHNICAL SKILLS

FRONTEND	React.js · Vue.js · TypeScript · HTML/CSS · Qt Creator (C++)
BACKEND & INFRA	Spring Boot · Flask · gRPC · Docker · Jenkins · Maven · Git
LANGUAGES	C / C++ (gcc 12) · Python · Java (OSGi) · JavaScript (ES6+) / TypeScript · Bash
DATA & TESTING	MySQL · GraphQL · Pytest · Mockito · JUnit · Agile / Scrum
EMBEDDED & LINUX	Custom rootfs · Buildroot · TinyCore · init/boot sequence · systemd · busybox · cross-compilation (ARM/x86) · SSH tunnelling
PROTOCOLS	RS232 · RS485 · TCP/IP · serial device integration

03 EXPERIENCE

Embedded & Full-Stack Software Engineer

Hanover Displays · via S2I (Celad group)

Sept 2021 – Present
Saint-Laurent-du-Var, France

Hanover Displays builds on-board passenger information and fleet management systems for public transport operators. I work on the embedded product stack in a team of four engineers.

- **Unified two product lines onto one codebase** — the company's biggest technical challenge to date. Ported the modern C++/gcc 12 stack onto older onboard computers (512 MB RAM, 1 GB flash) previously considered too constrained.
  ↗ Adopted TinyCore Linux and rebuilt boot sequence, daemons, libraries and stripped binaries; one codebase now runs across the entire fleet.
- **Built an internal configuration-profile generator.** The web tool lets the team define a customer's needs — SIM/APN, display type, destination sign, emergency features — and generates all files ready to flash, replacing manual config editing.
  ↗ Provisioning errors on new projects dropped to near zero.
- **Built a real-time vehicle-tracking frontend (Vue.js)** with live map, service timeline, complaints and a replay of each vehicle's day. Also provide direct technical support to operators: remote diagnosis, bug fixing, on-site hardware replacement and wiring.
- **Deployed embedded passenger-information systems across France.** Built the full pipeline: RS232 serial wake-up, content rendering and live server sync. Rebuilt the driver dashboard (duty-takeover screen) as a lightweight embedded HTML/JS/CSS interface.
  ↗ In daily service across France: buses, train stations and airport shuttles.
- **Built a reverse-SSH remote diagnostics system.** On a server command, the onboard unit opens an outbound SSH tunnel back to the team, giving full shell access with no technician on site.
  ↗ Significantly cut travel and intervention costs across the operator network.
- **Integrated ticketing, passenger counting and peripheral hardware** for multiple operators. Connected onboard validators, two independent passenger-counting systems, destination signs, an emergency button and an audio-monitoring relay over serial protocols.
  ↗ Each integration opened a new billable configuration option.

03 EARLIER EXPERIENCE

Full Stack Developer Intern

Feb – Aug 2021
Sophia-Antipolis, France

Renault Software Labs

Built a test-automation HMI for Renault's connected-vehicle cloud services.

- React.js GUI to pilot and automate test scenarios, with real-time SVG visualisation of vehicle state from cloud signals.
- Python microservices for remote vehicle API requests; Docker/Bash scripts simulating a physical test bench.

R&D Intern — Embedded AI & Vision

Sept 2020 – Jan 2021
Mulhouse, France

STIMUL'ACTIV

Embedded facial recognition for the Mwoo companion robot, for people with communication disabilities.

- Benchmarked vision algorithms (LBPH, CNN) on Raspberry Pi under FPS and memory constraints, then integrated the selected solution into the robot.

C++ / Qt Developer Intern

Apr – Aug 2020
Colmar, France

UHA — Biopôle de Colmar

Energy-consumption data tool for 50 measuring instruments across a research campus.

- Qt/C++ desktop application: SD → CSV → MySQL pipeline with visualisation for energy analysts.

04 AREAS OF FOCUS

Embedded Linux
Constrained rootfs, boot sequence, systemd and daemons.

Onboard interfaces
Driver HMIs, passenger displays and operational states.

Full-stack tooling
React/Vue frontends, Python and Java services.

Hardware integration
Serial devices, ticketing, counting and peripherals.

05 SELECTED PROJECTS

Driver HMI
Embedded HTML/JS/CSS duty-takeover interface for driver authentication, journey state and operational messaging.
EMBEDDED WEB UI

Passenger information
Onboard route, next-stop and accessibility displays driven by live journey data; in service in French cities.
EMBEDDED DISPLAYS

Remote diagnostics
Reverse-SSH access to onboard units for remote shell diagnosis, avoiding site visits and shortening interventions.
PYTHON · SSH

Configuration management
Browser tool that turns equipment requirements into structured, ready-to-flash configuration profiles.
FULL-STACK TOOLING

Airport shuttle displays
Flight-connection boards for airport shuttle services: departures, status and terminal or zone information.
PASSENGER DISPLAYS

Hardware integration
Serial integration modules for third-party displays, ticketing validators and passenger-counting systems.
RS232 · RS485

06 EDUCATION

2021 **Engineering Degree — Computer Science & Networks**
ENSISA Mulhouse (Grande École)

2017 **Classe Préparatoire MPSI / MP**
Lycée Victor Hugo, Besançon

07 LANGUAGES

English TOEIC 965 / C1

French Native

08 INTERESTS

3D modelling & printing · AI, LLMs & model evaluation · Emerging technology & hardware.

09 ENGINEERING APPROACH

Start from the constraint
The device and network set the design space.

Make the state legible
Show the right complexity at the right moment.

Keep the system observable
Logs and failure states are part of the product.

Stay close to deployment
The last mile is where assumptions meet hardware.