ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

DEPARTMENT OF MANAGEMENT SCIENCE AND TECHNOLOGY

Course: Programming II

ChatAUEB

Career Orientation System for the Departments of the Athens University of Economics and Business

---

Technical Report – Version 2.0

chataueb.fly.dev

DEVELOPMENT TEAM — V1.0

Arnold Mulai	8220092
Emmanouil Papagiannis	8220116
Stavros Vlachos	8220019
Nikolaos Sinani	8220225
Spilios Dimakopoulos	8220035
Ath.-Pan. Sakkatos	8220132
Antzela Ntasi	8220231
Evgenia Vekiou	8220015
Pavlos Stamatis	8220141

UPGRADE & DEPLOYMENT — V2.0

Spilios Dimakopoulos – 8220035

UI/UX · PostgreSQL/Neon · Docker · noVNC
Fly.io · CI/CD · AI · Security

Athens, March 2026

Abstract

This technical report describes ChatAUEB, a career orientation system designed for prospective students of the Athens University of Economics and Business (AUEB). Leveraging a large language model (LLM) through the free Groq API, the system analyzes a structured 18-question questionnaire and produces a personalized department recommendation with complete justification.

The application was originally developed (v1.0) by a team of nine students as part of the Programming II course. The present version (v2.0) was fully upgraded in terms of architecture, interface, security, infrastructure, and cloud deployment by Spilios Dimakopoulos (Student ID: 8220035), and is publicly available at chataueb.fly.dev.

Application type	Desktop GUI (Java 21 Swing) with browser access via noVNC
Database	Neon PostgreSQL (cloud, serverless, SSL/TLS required)
AI model	llama-3.3-70b-versatile via Groq API (free)
Hosting	Fly.io · Amsterdam · 512 MB · €0/month
Live URL	chataueb.fly.dev

Keywords: Java Swing, Artificial Intelligence, LLM, Groq API, PostgreSQL, Docker, noVNC, Fly.io, Career Orientation

Contents

1 Introduction

1.1 Motivation and Problem Statement

Choosing a field of study is one of the most significant decisions in a young person's life, with direct implications for their professional prospects. Despite its importance, the process is often characterized by insufficient information and a lack of personalized guidance. At AUEB, this challenge is particularly pronounced: the university offers eight departments with substantial thematic overlap and subtle distinctions that are not always obvious to a high school student.

ChatAUEB addresses this gap by providing a digital advisor that: (a) collects data through a structured questionnaire, (b) leverages an LLM for personalized analysis, and (c) provides a justified department recommendation. The goal is not to replace a human advisor, but to offer an objective and always-available first point of reference.

1.2 The Departments of AUEB

Table 1: Departments of the Athens University of Economics and Business

#	Department	Abbr.	Main Subjects
01	International, European & Economic Studies	IEES	International relations, European integration, multilingualism
02	Economics	ECON	Macroeconomics, econometrics, public economics
03	Management Science & Technology	MSIT	Management, digital innovation, technology
04	Business Administration	BA	Strategy, human resources, logistics
05	Accounting & Finance	AF	Accounting, financial analysis, investments
06	Marketing & Communication	MC	Marketing strategy, digital marketing
07	Informatics	INF	Algorithms, artificial intelligence, cybersecurity
08	Statistics	STAT	Statistics, operations research, data analysis

1.3 Report Structure

The report is organized as follows: §2 Functional Analysis, §3 Technical Architecture, §4 Database, §5 Artificial Intelligence, §6 Security, §7 Deployment, §8 Installation, §9 Development Team, §10 Conclusions.

2 Functional Analysis

2.1 Use Case Scenario

The user visits chataueb.fly.dev from any browser, with no software installation required. The typical usage flow is presented in Figure 1.

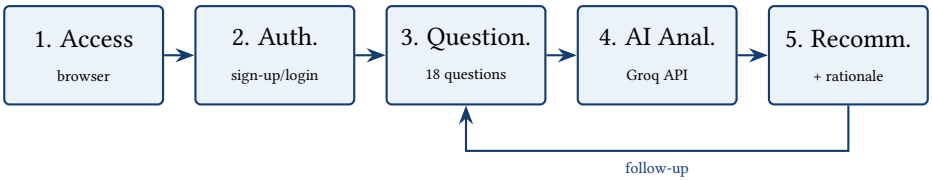

Figure 1: System usage flow

Table 2: Typical usage scenario

Step		Description
1	Access	Open URL – application loads immediately, no installation required
2	Authentication	Sign up / Log in / Guest (no data persistence)
3	Questionnaire	18 multiple-choice questions – at least 10 required
4	AI Analysis	Request sent to the Groq API – loading indicator ≈5–15 sec.
5	Result	Justified recommendation: primary + alternative + outlook
6	Follow-up	Free-form dialogue via the AUEB Advisor for clarifications

2.2 The Questionnaire

The 18 questions cover five thematic axes: academic interests, career goals, skills, values, and future direction. Scoring is not performed through hard-coded rules but by the LLM, which can assess complex interactions between answers. The requirement of at least 10 answers ensures sufficient data for reliable analysis.

2.3 AUEB Advisor

Beyond the questionnaire, the user may submit free-form queries through the AUEB Advisor. The model's system prompt: prevents off-topic responses, enforces the use of a professional tone, ensures consistency with AUEB's department information, and guards against prompt injection.

3 Technical Architecture

3.1 Layered Architecture

The system follows a 5-layer architecture, implementing the principles of loose coupling and high cohesion. Figure 2 illustrates the structure and external dependencies.

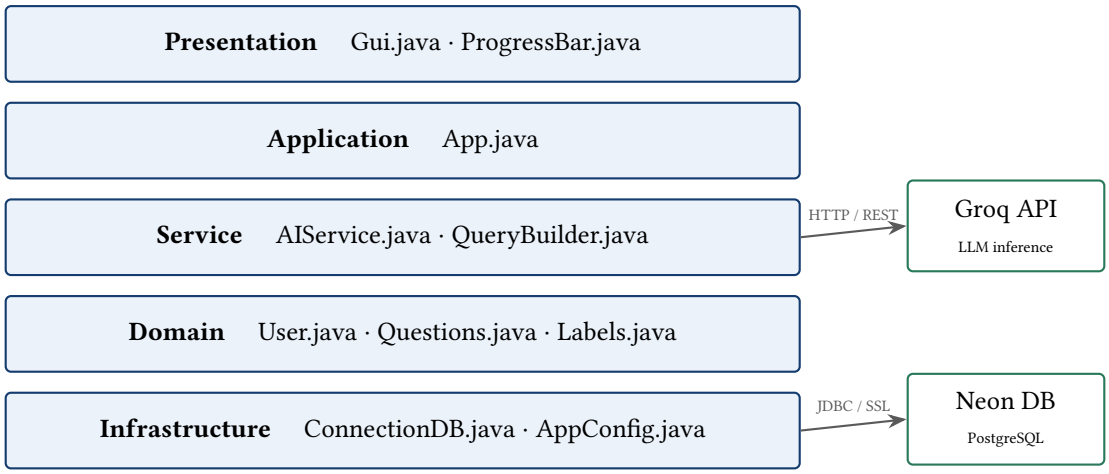

Figure 2: Layered architecture and external dependencies

Table 3: Responsibility per layer

Layer	Classes	Responsibility
Presentation	Gui.java, ProgressBar.java	Swing frames, event handling
Application	App.java	5-step bootstrap, coordination
Service	AIService.java, QueryBuilder.java	Groq API client, prompt assembly
Domain	User.java, Questions.java, Labels.java	Business logic, authentication
Infrastructure	ConnectionDB.java, AppConfig.java	JDBC, .env, schema bootstrap

3.2 Data Flows

Table 4: Main data flows

Flow	Sequence
Recommendation	Gui → QueryBuilder → AIService → Groq API → openResponseFrame()
Sign-up	Gui → User.signUp() → SHA-256 → ConnectionDB → Neon
Login	Gui → User.logIn() → hash comparison → main frame
Startup	AppConfig → Questions → initSchema() → download() → invokeLater()

3.3 File Structure

Table 5: Main project files

File	Purpose
App.java	Entry point – bootstrap across 5 stages
config/AppConfig.java	Singleton .env loader with fail-fast validation
db/ConnectionDB.java	JDBC DAL: initSchema, download, uploadCred, updateAns
model/User.java	User entity, SHA-256 hashing, static UserList
questionnaire/Questions.java	[18][7] array: questions and choices
service/AIService.java	Groq HTTP client, JSON parser, awaitResponse()
service/QueryBuilder.java	Prompt construction from questions/answers
ui/Gui.java	All Swing frames and dialogs
Dockerfile	Multi-stage: JDK builder + jlink JRE + Debian runtime
start.sh	Xvfb → fluxbox → Java → x11vnc → noVNC
fly.toml	Fly.io IaC: region, port, VM size

4 Database

4.1 Technology Selection

The original version used AUEB's institutional SQL Server, creating a dependency on university infrastructure. For v2.0, Neon PostgreSQL was selected – a serverless cloud database with a free tier – due to its zero cost, accessibility from anywhere, mandatory SSL/TLS, and simple configuration via a single JDBC URL. The migration was completed without any code modification.

4.2 Database Schema

Table 6: Database schema

Table	Column	Type	Description
users	username	VARCHAR(100)	PRIMARY KEY – unique identifier
	pass	VARCHAR(64)	SHA-256 hex – never plain text
answers	username	VARCHAR(100)	FK → users ON DELETE CASCADE
	question_idx	INT	Question index: 0–17
	answer_idx	INT	Choice index: 1–6, composite PK

The schema is automatically created at startup via `CREATE TABLE IF NOT EXISTS`. The `ON DELETE CASCADE` clause ensures referential integrity.

5 Artificial Intelligence Integration

5.1 Model Selection

The original version used the paid OpenAI API. For v2.0, the Groq API was selected – a free inference service using LPU hardware that delivers exceptionally high throughput. The API is OpenAI-compatible, allowing migration with minimal code changes. The model used is llama-3.3-70b-versatile with max_tokens=1500 and temperature=0.5.

5.2 Prompt Design and AI Flow

Figure 3 illustrates the horizontal flow of prompt construction and processing.

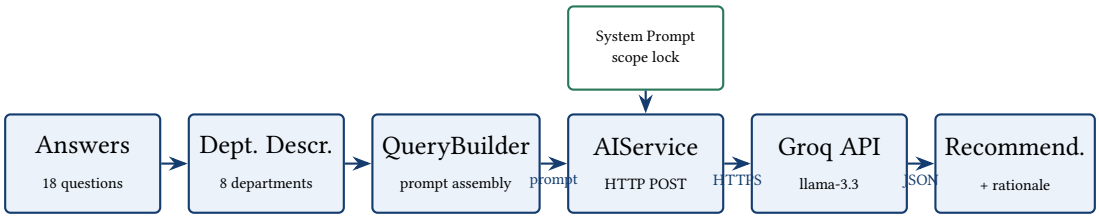

Figure 3: Prompt construction flow and Groq API call

AIService runs on a background thread, preventing the EDT from being blocked. Communication between threads occurs via synchronized wait/notifyAll. The JSON parser is implemented manually without external libraries.

6 Security Measures

The system implements a multi-layered security approach (Table 7).

Table 7: Security measures by area

Area	Mechanism	Implementation
Authentication	Password Hashing	SHA-256 hex digest (64 chars.) – password is never stored in plain text
Secrets	.env Mgmt.	API keys exclusively via .env. Excluded from Git and Docker image.
Database	SQL Injection	PreparedStatement on every query. Zero string concatenation in SQL.
Transport	SSL/TLS	sslmode=require in the JDBC URL. Module jdk.crypto.ec.
Resources	Resource Mgmt.	try-with-resources for every Connection and Statement.
AI Safety	Scope Lock	System prompt with explicit rejection of off-topic queries and prompt injection.
	Input Validation	Empty credentials, duplicate usernames, minimum 10 answers.

7 Containerization and Cloud Deployment

7.1 Infrastructure Architecture

Figure 4 presents the overall infrastructure architecture: the user, the Docker container on Fly.io, and the external services.

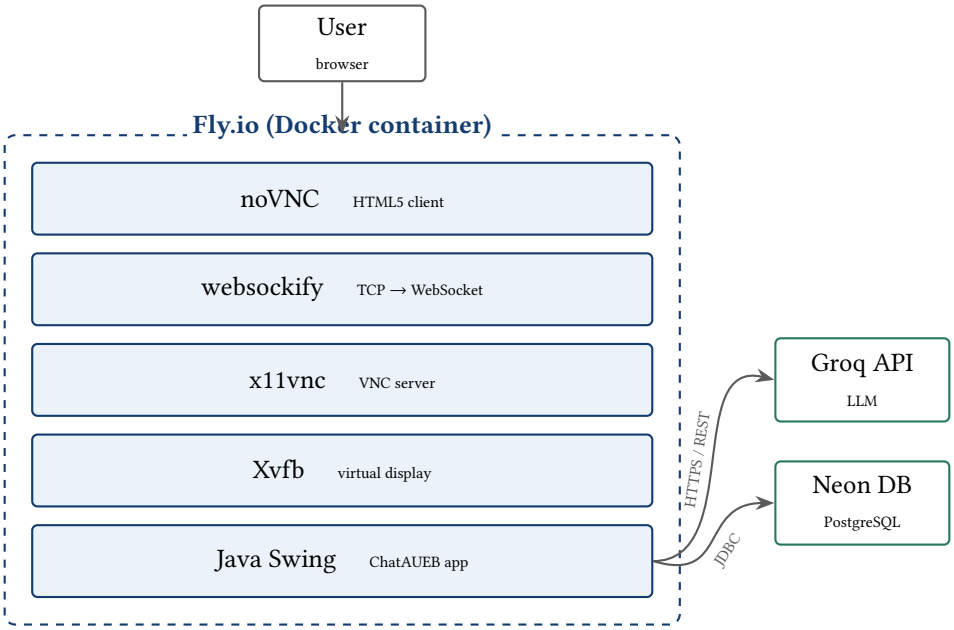

Figure 4: Infrastructure architecture: Docker container on Fly.io

7.2 Docker Multi-Stage Build

The Dockerfile uses a multi-stage build (Table 8). jlink creates a custom JRE containing only the 11 required modules, reducing the image size from ≈650 MB to ≈95 MB.

Table 8: Docker build stages

Stage	Base Image	Actions
builder	eclipse-temurin:21-jdk-jammy	Maven build → fat JAR → jlink custom JRE
runtime	debian:bookworm-slim	Copy JRE + JAR + start.sh + noVNC. apt: xvfb x11vnc fluxbox

7.3 Fly.io Cloud Deployment

Table 9: Fly.io deployment parameters

Parameter	Value
Region	ams (Amsterdam, Netherlands)
VM Size	shared-cpu-1x, 512 MB RAM
Port	6080 → HTTPS (automatic TLS termination)
Live URL	https://chataueb.fly.dev
Cost	€0/month – within free tier
Cold Start	None – VMs remain alive
Auto-deploy	GitHub Actions: git push → flyctl deploy automatically

8 Installation Guide

8.1 Local Execution

Only Docker Desktop is required. No Java or Maven installation is needed.

1. `git clone https://github.com/USERNAME/chataueb.git`
2. `cp .env.example .env` – fill in DATABASE_URL and GROQ_API_KEY
3. `docker compose up –build`
4. Open http://localhost:6080

8.2 Environment Variables

Table 10: Environment variables

Variable	Req.	Default	Description
DATABASE_URL	Yes	—	JDBC URL from Neon
GROQ_API_KEY	Yes	—	API key from console.groq.com
GROQ_MODEL	No	llama-3.3-70b-versatile	Groq model identifier
GROQ_API_URL	No	Groq endpoint	AI API base URL

9 Development Team

9.1 Version 1.0 – Initial Development

Table 11: Development team v1.0

Student ID	Name	Contribution
8220092	Arnold Mulai	Repository management, database
8220116	Emmanouil Papagiannis	Code coordination, graphical interface
8220019	Stavros Vlachos	Questionnaire design
8220225	Nikolaos Sinani	Domain model, quality control
8220035	Spilios Dimakopoulos	Functionality, AI interface
8220132	Ath.-Pan. Sakkatos	Infrastructure classes, architecture
8220231	Antzela Ntasi	AUEB department information
8220015	Evgenia Vekiou	AUEB department information
8220141	Pavlos Stamatis	Development and integration

9.2 Version 2.0 – Upgrade and Deployment

Table 12: v2.0 upgrades – Spilios Dimakopoulos (ID: 8220035)

Area	Upgrades
UI/UX	Dark theme, card-based layout, chip-style choices, consistent headings
Database	SQL Server → Neon PostgreSQL. SSL required. Prepared statements.
Docker	Multi-stage build. jlink custom JRE (≈95 MB). debian:bookworm-slim.
Browser Access	noVNC HTML5 stack. Auto-connect. Fullscreen 1600 × 900.
Cloud / CI-CD	Fly.io fly.toml IaC. GitHub Actions auto-deploy. Zero cold start.
Artificial Intelligence	OpenAI → Groq. System prompt scope lock. Structured output.
Security	SHA-256, .env secrets, PreparedStatement, AI scope restriction.
Code Quality	== → .equals(). try-with-resources. SLF4J logging.

10 Conclusions and Future Directions

10.1 Goal Achievement

ChatAUEB v2.0 achieves its original goals: it provides a reliable, personalized department recommendation, operates as a production application at chataueb.fly.dev, and incurs no monthly operating cost. The exclusive use of free services (Fly.io, Neon, Groq) demonstrates the viability of production applications with zero operational cost.

10.2 Known Limitations

- **Single user:** Java Swing does not support concurrent sessions
- **Mobile UX:** Degraded experience on mobile devices due to VNC scaling
- **Password Reset:** Email-based recovery is not implemented
- **AI Non-determinism:** temperature=0.5 produces non-deterministic responses

10.3 Proposed Extensions

1. **[High]** Migration to a Web Application (Spring Boot + React) for native multi-user support
2. **[High]** Email authentication and password recovery
3. **[Medium]** Retrieval-Augmented Generation using official AUEB texts
4. **[Medium]** Analytics dashboard for usage statistics
5. **[Low]** Multilingual support for international students

References

[1] Oracle Corporation. (2024). *Java SE 21 Documentation*. https://docs.oracle.com/en/java/javase/21/

[2] Groq Inc. (2025). *Groq API Documentation*. https://console.groq.com/docs

[3] Neon Inc. (2025). *Neon PostgreSQL Documentation*. https://neon.tech/docs

[4] Fly.io. (2025). *Fly.io Documentation*. https://fly.io/docs

[5] Meta AI. (2024). *Llama 3 Technical Report*. https://ai.meta.com/research/publications/llama-3/

[6] Docker Inc. (2025). *Docker Documentation*. https://docs.docker.com

[7] noVNC Project. (2024). *noVNC: VNC client using HTML5*. https://novnc.com

[8] Athens University of Economics and Business. (2025). *AUEB Departments*. https://www.aueb.gr/el/content/tmimata