Welcome to Seafaring-IT

Navigate your digital transformation.

Who We Are

Seafaring-IT is a premium IT solutions company helping businesses navigate digital transformation with confidence. We combine modern web and cloud technologies, AI innovation, and DevOps excellence to build scalable, secure, and future-ready solutions. From strategy to deployment, we chart a clear course from complexity to success.

Our Vision & Mission

Our Vision

To become a trusted global technology partner, empowering businesses to navigate digital change with innovation, reliability, and long-term impact.

Our Mission

Our mission is to deliver cutting-edge IT solutions that drive growth and efficiency. We guide our clients through complex digital challenges by combining modern development, cloud infrastructure, AI innovation, and strategic consulting—turning ideas into scalable, real-world success.

What Sets Us Apart

Cutting-Edge Innovation

We continuously invest in R&D, ensuring our solutions are always at the forefront of technological advancements, providing a distinct competitive advantage.

Strategic Partnerships

Our strong collaborations with industry leaders and academic institutions drive mutual growth and foster a rich ecosystem of shared knowledge and resources.

Exceptional Talent

Our team comprises highly skilled and dedicated professionals, passionate about solving complex challenges and delivering outstanding results.

Introducing Our Projects for Final Year Students

Innovative, Real-World Challenges Await

Project 1: Intelligent Retail Supervision & Real-Time Analytics Platform

Goals:

- Build a real-time monitoring platform for retail stores combining sales tracking, cashier oversight, and AI-powered analytics.
- Detect anomalies, prevent fraud, and predict inventory shortages automatically.
- Create mobile and web dashboards for store owners to monitor performance instantly.

Student Roles:

- Full-stack development (React, Node.js), AI/ML engineers, data specialists, DevOps engineers.

Expected Outcomes:

A complete platform with real-time sales monitoring, fraud detection, inventory forecasting, and intelligent alerting system deployed with Docker.

Technologies:

JavaScript/TypeScript, React, Node.js, Python, PostgreSQL, Redis, Docker, Machine Learning, API development.

Project 2: AI-Powered Intelligent E-Learning Platform

Goals:

- Develop an intelligent e-learning platform that adapts to each student's learning pace and level.
- Use AI to understand educational content and create personalized learning paths.
- Help students overcome knowledge gaps with intelligent tutoring and recommendations.

Student Roles:

- Full-stack developers (React, Node.js), AI/ML engineers, data specialists, security engineers.

Expected Outcomes:

A complete e-learning platform with personalized learning paths, AI-powered tutoring, student performance tracking, and secure data management.

Technologies:

Python, Node.js, React/Next.js, NLP, LLM models, PostgreSQL, MongoDB, API development, Docker, recommendation systems, data security.

Project 3: Centralized Multi-Store & Multi-Warehouse Management Platform

Goals:

- Build a centralized platform to manage multiple stores and warehouses from a single system.
- Optimize product availability and prevent unnecessary stock shortages.
- Streamline order preparation and delivery across all locations.

Student Roles:

- Full-stack developers (React, Node.js), systems engineers, logistics specialists, DevOps engineers.

Expected Outcomes:

A complete platform with real-time inventory management, automated order orchestration, warehouse notifications, order tracking, and comprehensive dashboards for all locations.

Technologies:

JavaScript/TypeScript, React/Next.js, React Native, Node.js, PostgreSQL, Redis, WebSockets, API development, Docker, real-time notifications, role-based access control.

Project 4: Rabbehni – Global Gamified Customer Loyalty Platform

Goals:

- Build an intelligent, gamified loyalty platform that transforms one-time customers into loyal repeat visitors.
- Create engaging experiences through gamification, habit-building, and cultural adaptation.
- Provide businesses with actionable data to understand and influence customer behavior.

Student Roles:

- Full-stack developers (React, Node.js), product engineers, data analysts, AI/ML engineers, cloud architects.

Expected Outcomes:

A complete SaaS loyalty platform with QR code check-ins, gamification mechanics, real-time analytics, multi-language support, and white-label capabilities for global deployment.

Key Features:

- QR code check-in system with instant rewards
- Gamification engine (spin wheels, scratch cards, treasure chests, challenges)
- Smart Habits system to encourage repeat visits
- Real-time business analytics dashboard (retention, engagement, frequency)
- Multi-language (Arabic-first), multi-currency, regional personalization
- Automatic inactive customer recovery mechanisms
- White-label and franchise-ready architecture

Technologies:

JavaScript/TypeScript, React/Next.js, React Native, Node.js, Python, PostgreSQL, Redis, QR systems, Docker, cloud computing, machine learning, data analytics, gamification systems.

Project 5: Hospital System Observability & Patient Journey Monitoring

Goals:

- Build complete observability for hospital systems to monitor patient journeys across all critical applications
- Detect, correlate, and resolve incidents quickly to minimize impact on patient care
- Track performance metrics across admission, triage, appointments, and laboratory systems

Student Roles:

- DevOps engineers, SRE specialists, infrastructure engineers, monitoring specialists

Expected Outcomes:

A complete observability platform with distributed tracing, centralized logging, real-time dashboards, automated alerting, and operational runbooks for hospital incident management.

Key Features:

- Distributed tracing for patient journey tracking across services
- Centralized log aggregation and correlation
- Real-time dashboards (wait times, latency, errors, system saturation)
- SLO/SLA definition with intelligent alerting and escalation
- Operational runbooks for common incidents (outages, overload, unavailability)
- Performance monitoring for critical hospital workflows

Technologies:

Linux, Docker, Kubernetes, OpenTelemetry, Prometheus, Grafana, Loki/ELK, alerting systems, SRE practices

Project 6: Microservices Architecture for Community Patient Records System

Goals:

- Design a lightweight, modular patient records system for community health centers and NGOs
- Build a scalable microservices architecture that enables care continuity without hospital system complexity
- Ensure interoperability, security, and resilience across all services

Student Roles:

- Backend engineers (Node.js, Spring Boot), DevOps engineers, API specialists, security engineers

Expected Outcomes:

A complete microservices platform with API Gateway, distributed services, event-driven architecture, Kubernetes deployment, and comprehensive security and audit documentation.

Key Features:

- Modular microservices (patient management, consultations, prescriptions, reference data)
- API Gateway with authentication, rate limiting, routing, and versioning
- Event-driven architecture for business notifications and audit trails
- Resilience patterns (idempotence, retries, circuit breaker)
- Kubernetes deployment with auto-scaling and health monitoring
- Role-based access control and audit logging

Technologies:

Node.js, Spring Boot, REST APIs, PostgreSQL, Kafka/RabbitMQ, Docker, Kubernetes, API Gateway, event streaming, security best practices

Project 7: Secure CI/CD Pipeline for Social Assistance Platform

Goals:

- Build a secure, automated CI/CD pipeline for a social assistance platform handling sensitive data
- Ensure rapid, reliable deployments while maintaining quality, security, and compliance
- Implement comprehensive testing, security scanning, and deployment automation

Student Roles:

- DevOps engineers, CI/CD specialists, security engineers, infrastructure engineers

Expected Outcomes:

A complete CI/CD infrastructure with automated testing, security scanning, containerization, staging/production deployment, rollback capabilities, and comprehensive documentation.

Key Features:

- Containerized application (frontend, backend, database) with standardized environments
- Automated CI pipeline (tests, linting, builds, artifact management)
- Security integration (SAST, dependency scanning, image scanning)
- Automated CD pipeline (staging and production deployments)
- Automated rollback mechanisms for failed deployments
- Secrets and configuration management following best practices
- Release documentation, backup/restore procedures, and disaster recovery planning

Technologies:

GitHub Actions/GitLab CI, Docker, Docker Compose, Kubernetes, SonarQube, Trivy, SAST tools, automated testing frameworks, VPS/K8s deployment

Project 8: Mini Lakehouse for Water & Energy Data Analytics

Goals:

- Build a governed data pipeline for water and energy consumption data
- Implement multi-layer architecture (bronze/silver/gold) for data quality and analytics
- Create reliable KPIs and dashboards for resource management and decision-making

Student Roles:

- Data engineers, analytics engineers, data quality specialists, DevOps engineers

Expected Outcomes:

A complete data lakehouse with multi-source ingestion, quality controls, analytical models, SQL exposure, and comprehensive documentation for operational use.

Key Features:

- Multi-source data ingestion (CSV, IoT sensors, APIs) into bronze/silver/gold layers
- Data quality controls (outlier detection, missing values, duplicates, consistency checks)
- Analytical KPI models (energy per m^2, water per day, losses, peak analysis)
- SQL-ready datasets optimized for BI tools and dashboards
- Data lineage tracking and governance documentation
- Reproducible demo and operational guide

Technologies:

Apache Spark, Delta Lake/Iceberg, MinIO/S3, Apache Airflow, SQL, Great Expectations, data quality frameworks, BI tools integration

Project 9: Anomaly Detection for Water & Energy Consumption

Goals:

- Build an intelligent anomaly detection system to identify water leaks and energy overconsumption
- Provide explainable insights into the causes of anomalies
- Prioritize interventions based on impact and cost savings

Student Roles:

- Machine learning engineers, data scientists, backend engineers, MLOps specialists

Expected Outcomes:

A complete ML-powered anomaly detection system with explainability, API exposure, model tracking, and estimated impact analysis for resource optimization.

Key Features:

- Dataset construction (open data and synthetic data) with feature engineering
- Multiple anomaly detection models (statistical and machine learning) with comparison
- Explainability layer (SHAP) with error analysis
- Alert API with anomaly scoring and prioritization
- Model tracking and versioning (MLflow)
- Concept drift monitoring and model performance tracking
- Business impact report (estimated savings and environmental impact)

Technologies:

Python, machine learning libraries (scikit-learn, XGBoost, Isolation Forest), SHAP, MLflow, FastAPI, PostgreSQL, Elasticsearch, data visualization tools

Project 10: RAG Chatbot for Health Centers & NGOs

Goals:

- Build a reliable conversational AI assistant for health center staff and social workers
- Provide accurate, sourced answers about procedures, rights, and referrals
- Ensure traceability, access control, and audit logging for sensitive information

Student Roles:

- Backend engineers (Python), AI/ML engineers, full-stack developers, security engineers

Expected Outcomes:

A complete RAG-powered chatbot system with vector indexing, role-based access control, audit logging, quality evaluation, and a simple web interface for staff and administrators.

Key Features:

- Document ingestion and preparation (chunking, cleaning, preprocessing)
- Vector indexing with semantic search (embeddings)
- Sourced responses with citations and document references
- Role-based access control (RBAC) with audit logging
- Quality evaluation framework (test questions, hallucination detection)
- Simple web interface for end users
- Admin dashboard for index management and document updates
- Conversation history and audit trails

Technologies:

Python, RAG frameworks, embeddings (OpenAI, Hugging Face), Qdrant/FAISS, FastAPI, RBAC, audit logging, web frontend (React/Vue), vector databases

Project 11: Social & Health Monitoring - Sentiment Analysis & Weak Signal Detection

Goals:

- Build an automated social listening system to monitor health and social sector conversations
- Detect sentiment trends, emerging issues, and weak signals before they become crises
- Provide actionable insights for proactive decision-making

Student Roles:

- Data engineers, NLP specialists, data scientists, BI developers, ethics specialists

Expected Outcomes:

A complete social monitoring platform with data collection, sentiment analysis, trend detection, dashboards, and comprehensive ethical evaluation and documentation.

Key Features:

- Data collection from public APIs and sources with normalization
- Data anonymization and privacy protection
- Sentiment analysis and topic modeling
- Peak detection and weak signal identification
- Real-time monitoring dashboards with alerts
- Trend analysis and reporting
- Data quality evaluation and bias assessment
- Improvement roadmap and recommendations
- Ethical guidelines and limitations documentation

Technologies:

Python, NLP libraries (NLTK, spaCy, transformers), sentiment analysis models, topic modeling (LDA, BERTopic), data collection APIs, data anonymization tools, BI tools (Tableau, Power BI), visualization libraries

Project 12: MLOps for Energy Demand Forecasting

Goals:

- Build a production-ready machine learning system for energy demand forecasting
- Optimize energy production, storage, and purchasing decisions
- Implement continuous monitoring, retraining, and model improvement

Student Roles:

- ML engineers, MLOps specialists, data engineers, backend engineers

Expected Outcomes:

A complete MLOps pipeline with model training, deployment, monitoring, and automated retraining for reliable energy demand predictions.

Key Features:

- Dataset preparation (consumption, production, weather, and feature engineering)
- Multiple time series forecasting models with comparison
- MLflow integration for model versioning and experiment tracking
- API deployment with batch prediction jobs
- Performance monitoring (latency, errors, concept drift)
- Automated retraining pipeline
- Operational guide and maintenance procedures
- Cost and emissions impact analysis

Technologies:

Python, time series libraries (statsmodels, Prophet, LSTM), scikit-learn, MLflow, FastAPI, Docker, Docker Compose, CI/CD (GitHub Actions/GitLab CI), PostgreSQL, monitoring tools

Project 13: MRV Platform for Green Projects - Measurement, Reporting & Verification

Goals:

- Build a complete MRV (Measurement, Reporting, Verification) system for green projects
- Quantify, track, and audit carbon impact with full traceability
- Enable evidence-based reporting for climate action projects

Student Roles:

- Backend engineers, data engineers, BI developers, sustainability specialists, workflow engineers

Expected Outcomes:

A complete MRV platform with baseline modeling, evidence management, CO2 impact calculation, validation workflows, and auditable reporting for green projects.

Key Features:

- Baseline vs. actual modeling (energy, water, CO2e) with documented assumptions
- Evidence management system (documents, photos, sensor measurements)
- CO2e impact calculation with uncertainty quantification
- Multi-stage validation workflow (field data → expert review → audit)
- Real-time MRV dashboards (impact, costs, project progress)
- Auditable exports and compliance reporting
- Methodology documentation and best practices guide
- Data quality controls and traceability logs

Technologies:

Python, PostgreSQL, data validation frameworks, workflow engines, BI tools (Tableau, Power BI), document management, geospatial tools, API development, audit logging

Project 14: Offline Field Data Collection PWA for Social & Green Projects

Goals:

- Build a robust offline-first mobile application for field teams working without connectivity
- Ensure reliable data collection, synchronization, and complete audit trails
- Implement quality controls and multi-level validation for field interventions

Student Roles:

- Full-stack developers (PWA/React), mobile engineers, backend engineers, security specialists, QA engineers

Expected Outcomes:

A complete offline-capable PWA with robust synchronization, quality controls, multi-level validation, evidence management, and comprehensive security and audit documentation.

Key Features:

- Offline-first forms with attachments (photos, documents, signatures)
- Robust synchronization with conflict resolution and recovery
- Data quality controls (validation rules, consistency checks, duplicate detection)
- Multi-level validation workflow (field agent → supervisor → admin)
- Evidence file management with audit trails
- Complete audit logging of all actions and changes
- Data encryption and security best practices
- Offline-to-online transition handling
- Progress tracking and sync status visibility

Technologies:

React, PWA (Service Workers), IndexedDB/SQLite, offline-first sync libraries, Node.js backend, PostgreSQL, encryption libraries, audit logging, mobile-responsive design, file management APIs

Project 15: Real-Time Streaming Detection for Water & Energy Network Leaks & Faults

Goals:

- Build a real-time streaming pipeline for detecting leaks and anomalies in water and energy networks
- Enable near-instantaneous detection to minimize losses and operational risks
- Provide actionable insights for network supervision and maintenance

Student Roles:

- Data engineers, streaming specialists, ML engineers, backend engineers, DevOps engineers

Expected Outcomes:

A complete real-time streaming platform with sensor data ingestion, anomaly detection, contextual correlation, supervision dashboards, and operational procedures for network management.

Key Features:

- Real-time data ingestion from sensors (simulated and real)
- Streaming anomaly detection with scoring and prioritization
- Contextual correlation (zones, historical patterns, weather)
- Real-time supervision API and dashboards
- Alert management and escalation
- Performance monitoring (latency, false positive rates)
- Operational procedures and runbooks
- Integration with existing SCADA/monitoring systems

Technologies:

Apache Kafka, Apache Spark/Flink, anomaly detection algorithms, stream processing, PostgreSQL/TimescaleDB, Redis, FastAPI, BI tools (Grafana, Kibana), Docker, Kubernetes

How to Get Involved

01

Review Project Briefs

Explore all **15 exciting projects** to find your perfect fit.

02

Submit Your Application

Email us at **hello@seafaring-it.com** with the subject line "PFE: [Project Title]" and include your **CV**.

03

Optional: Motivation Letter

Help us understand your **passion and goals** for the project in a brief letter.

04

Interview & Selection

We'll review applications and **contact shortlisted candidates** for an interview with our project leads.

05

Begin Your Journey

Successful applicants will embark on an exciting journey of **innovation and learning** with their final year project.

Don't see a project that fits? We have more unique ideas! Send us an email with your interests and we'll find the perfect match for you.

Contact Us: hello@seafaring-it.com

Thank You & Next Steps

Explore, Innovate, Succeed with Seafaring-it

Visit our website at **https://www.seafaring-it.com/**

For further inquiries, contact us at **hello@seafaring-it.com**