2026 – Fog Computing for Smart Services (6 ECTS)

Guidelines and Student Assignments

For laboratory assignments, students will be assigned a seminar project to complete by the end of the spring semester. Students are anticipated to diligently and weekly progress on their projects, ensuring they meet the given milestones and deadlines. Both teachers on this course will give a **weekly** overview of the process and keep notes on the students' ongoing progress.

To qualify for presenting their seminar work, students must submit a written report (10-15 pages), which can be composed in either Slovenian or English language.

Each student should provide a Git account to be included in the course repository, which the teachers will monitor.

Seminar Topics

1. **CD/CI pipeline with the example of EDUxCERT**

   Students will be given access to an existing EDUxCERT codebase that requires structural reorganization, modularization, refactoring and full containerization. The objective is to transform the system into a reproducible, production-ready application. Work must follow established DevOps and software engineering principles, with clear architectural justification.

   All design decisions must be technically justified.

2. **Small Language Model for Study Programme Analysis (UNP)**
   Students will design and deploy a small language model (SLM)–based analytical service focused on the UNP (Učni načrti predmetov) university study programme. The system must ingest heterogeneous data sources (e.g., official programme descriptions, course syllabi, teaching/learning methods from literature, and a PDF of the university programmes) and enable structured analysis of the curriculum. Various use cases should be supported (comparison, quality evaluation, literature evaluation etc.) The objective is to evaluate programme coherence, redundancy, methodological soundness, and alignment with current academic literature and comparable programmes.

   Scope of analysis must include:
   - Comparison of the existing UNP programme with relevant academic literature
   - Benchmarking against comparable national/international study programmes
   - Detection of content overlap and course repetition
   - Identification of methodological gaps or inconsistencies
   - Structural analysis (learning outcomes, ECTS balance, progression logic)
   - Any additional academically relevant observations regarding programme design

   Results include:

- Implemented SLM-based analysis service
- Data ingestion and preprocessing pipeline (including PDF processing)
- Analytical report with evidence-based conclusions
- Technical documentation of model choice, limitations, and evaluation methodology

3. LLM/SLM Pipeline for Legal Domain Analysis (LAW)

Students will design and implement a language-model–based analytical service for the legal domain. The system must integrate case law databases and other structured legal sources to support professional legal research tasks. The focus is on building a reproducible pipeline for developing, evaluating, and validating a domain-adapted LLM or small LM.

Project requirements:

- Integration of case law databases and relevant legal sources
- Definition of realistic use cases (e.g., legal argument preparation, precedent search, judicial reasoning support)
- Identification of primary users (lawyers, judges, legal researchers)
- Design and implementation of an LLM/SLM pipeline (data ingestion, preprocessing, indexing, prompting or fine-tuning)
- Evaluation framework with clearly defined precision, accuracy, and reliability metrics
- Systematic testing methodology (benchmark queries, hallucination analysis, error classification)

Results include:

- Functional legal AI service prototype
- Documented model development pipeline
- Empirical evaluation report (precision, accuracy, limitations)
- Critical discussion of ethical, methodological, and reliability aspects in legal AI use

4. LLM Legal Advisory Decision Service

Students will design an LLM-based advisory system that generates structured compliance recommendations from statutory texts, regulations, and case law (e.g. freely accessible databases of Anglo-Saxon case law, including any ontologies and knowledge bases used in the domain). The system must transform legal context into actionable guidance while preserving traceability to primary legal sources.

Core requirements:

- Integration of statutory and case law databases
- Context-aware compliance recommendation generation
- Structured output format (risk level, applicable norms, required actions)
- Source citation and traceability mechanisms
- Evaluation of precision and legal relevance
- Analysis of hallucination risk and mitigation strategies

Results include a functional advisory prototype, evaluation report, and methodological discussion of reliability and legal liability implications.

5. **LLM Legal Decision Justification Service**

   Students will develop a legal reasoning support system that produces structured, explainable justification traces for legal decisions. The system must generate argumentation chains grounded in legal norms and precedents.

   Core requirements:
   - Retrieval-augmented reasoning over case law and statutes
   - Generation of structured justification
     (facts → norms → interpretation → conclusion)
   - Explicit citation of supporting legal sources
   - Explainability framework for reasoning transparency
   - Evaluation of logical consistency and doctrinal correctness

   Results include the implemented justification engine, benchmark evaluation (precision, reasoning coherence), and critical analysis of explainability in legal AI systems.

6. **Quality in Education Services: Accreditation Quality Criteria Mapping Service**
   Students will design and implement an LLM-based service that maps institutional programme data to national and international accreditation standards. The system must automatically analyze study programme documentation (curricula, syllabi, learning outcomes, workload distribution) and evaluate compliance against defined accreditation criteria.

   The objective is to support higher education institutions in identifying compliance gaps, inconsistencies, and areas requiring improvement through structured, traceable analysis.

   Core requirements:
   - Integration of accreditation and nostrification standards (formal criteria, legal requirements, quality assurance frameworks)
   - Ingestion and preprocessing of institutional documentation (including structured and PDF sources)
   - Automated mapping of programme elements to accreditation criteria
   - Gap detection and compliance scoring
   - Traceable output linking findings to specific standards
   - Evaluation of precision, recall, and reliability of mapping

Results include a functional mapping prototype, a compliance evaluation report for a selected programme, and a methodological discussion of limitations, risks, and legal implications of AI-assisted accreditation analysis.

7. Environmental/Weather Forcasting Context Aggregation Service: LLM-Based Environmental Context Aggregation and Advisory System

Students will design and implement a service that aggregates heterogeneous environmental data and uses an LLM (or locally adapted small LM) to generate operational environmental advisories. This should certainly distniguish between all the spectrum from global to local weather/environmental models. The system must integrate sensor data (IoT), meteorological sources, satellite data, and other local datasets into a unified contextual representation suitable for AI-driven reasoning. Having in mind that this services usually run on the Edge, a small LM would be preferable option.

Core requirements:

- Data ingestion pipeline (sensor streams, meteo data, satellite inputs, local datasets)
- Data normalization and contextual feature construction
- Optional local model training or domain-specific fine-tuning
- LLM-based interpretation producing structured environmental advisories
- Explainability of recommendations
- Quantitative evaluation (accuracy, precision, advisory relevance)

Result: functional prototype, architecture specification, evaluation report, and discussion of limitations and deployment considerations.

8. Environmental Context Aggregation Service: Environmental Risk Detection and Human–AI Alignment Analysis

Students will develop a system that detects environmental risks from aggregated context data and evaluates alignment between AI-generated interpretations and human expert assessments. The focus is on pattern detection, operational reliability, and systematic comparison of AI and human decision-making.

Core requirements:

- Contextual environmental risk detection (pattern recognition, anomaly identification)
- Risk severity classification framework
- Generation of AI-based environmental assessments
- Collection of corresponding human expert interpretations
- Agreement metrics and divergence analysis
- Reliability and bias assessment

Result: risk detection prototype, human–AI comparison module, quantitative evaluation results, and methodological analysis of trustworthiness and operational applicability.

9. System Context Monitoring Service: LLM-Assisted Orchestration for (Re)Deployment in Things-to-Cloud Environments

Students will design and implement an LLM-assisted decision-support system that identifies optimal deployment options for a service in a Things-to-Cloud environment. The system must evaluate candidate deployment strategies across edge and cloud infrastructure and recommend one or more optimal placements based on measurable operational criteria. The LLM is used to reason over system state, constraints, and trade-offs when selecting deployment configurations.

The optimization process must incorporate both technical and user-centric attributes, as well as economic feasibility in case of redeployment.

Required components:

- Formal model of the Things-to-Cloud environment (edge nodes, cloud nodes, network characteristics)
- Definition of QoS metrics (latency, throughput, availability, reliability, resource utilization)
- Definition of QoE attributes (user-perceived responsiveness, SLA compliance, service continuity)
- Redeployment cost model (migration overhead, downtime impact, energy consumption, financial cost)
- LLM-based reasoning component for evaluating and ranking deployment options
- Mechanism for comparing current vs. alternative placements to determine whether redeployment is justified
- Experimental validation under varying workload and network scenarios

Deliverables:

- Architecture and optimization model specification
- Implemented prototype integrating LLM-assisted decision logic
- Evaluation demonstrating how optimal deployment options are identified
- Quantitative analysis of trade-offs between QoS, QoE, and redeployment cost

10. Things-to-Cloud: Trust Management and Verification

Design and apply multiple evaluation methodologies to rigorously assess an existing trust management mechanism with respect to correctness, robustness, performance, and architectural suitability.

Students are not required to implement a new trust model. Instead, they must critically evaluate the provided system using complementary validation strategies.

Requirements:

- Conduct a structured review of relevant trust models and evaluation frameworks (e.g., centralized vs. decentralized, hierarchical vs. federated trust).
- Analyze the architecture and enforcement logic of the provided trust management mechanism.
- Propose and implement multiple evaluation methodologies, such as:
  - Functional testing (validation of trust-list enforcement rules)
  - Formal verification or model-based validation
  - Negative and adversarial testing (revoked, forged, or non-listed entities)
  - Performance benchmarking (lookup latency, scalability, reliability under load)
  - Security analysis (attack surface, revocation handling, consistency guarantees)
- Specifically verify:
  - Only entities present in the trust list are accepted.
  - Revoked or unlisted entities are consistently rejected.
  - Correct interaction with verifiable credentials and revocation mechanisms.
- Compare empirical findings with alternative trust architectures and discuss trade-offs.

Deliverables:

- Evaluation methodology proposal (multiple approaches justified)
- Experimental results with quantitative metrics
- Critical comparison to alternative trust models
- Structured assessment of strengths, weaknesses, and improvement opportunities

11. Intelligent DIDs: Digital Twin DID Interface with Multimedia + LLM/SLM Access (IDID)

Students will design and implement a generic **Intelligent DID (iDID)** framework that extends a standard Decentralized Identifier into a **Digital Twin–like interface**. The system must enable policy-controlled access to structured subject data, linked multimedia resources, and an integrated LLM/SLM reasoning service directly via the DID interface.

The objective is to transform the DID from a static identifier into an **operational, auditable interaction layer** that supports human and machine clients while preserving decentralization, interoperability, and verifiability principles.

Requirements:

- DID-based capability model exposing services via DID Document (media, twin state, reasoning, policy, audit)
- Policy-controlled access to multimedia and structured twin data
- LLM/SLM reasoning endpoint with retrieval over DID-linked resources
- Structured outputs with citations and traceability

- Minimal Digital Twin schema (identity, attributes, events, lifecycle)
- Containerized, reproducible prototype

Deliverables:

- Functional DID prototype with resolver and service endpoints
- Multimedia ingestion and indexing pipeline
- LLM/SLM-based reasoning service (Q/A + structured reports with evidence)
- DID-based access control and audit mechanism
- Evaluation report (interoperability, security, reasoning quality, limitations)
- Technical documentation of architecture and design decisions

12. Autopoiesis Architecture in Action: Simulation of Quantum Autopoietic Gates and Identity Dynamics (see paper)

Students will design and implement a simulation framework that operationalizes the **quantum autopoiesis model** as an executable architecture. The system must simulate identity persistence, coherence-driven state transitions ("gates"), and symbolic trajectory accumulation within a minimal autopoietic loop.

The objective is to translate the formal model of recursive identity and admissible quantum transitions into a **functional computational prototype** that demonstrates autopoiesis dynamics and symbolic structure growth under varying coherence and constraint conditions.

The project is grounded in the operational model where identity is preserved under coherence and updated through admissible quantum-like gates determined by interaction constraints.

Requirements

- Simulation of the Binary Decision Operator (preserve / update / decay)
- Finite symbolic alphabet (phase classes) and admissible gate set
- Identity trajectory generation under coherence threshold conditions
- Configurable epistemic (Ψ) and deontic (Φ) potentials controlling transitions
- Visualization of symbolic trajectories and gate enactments
- Containerized, reproducible simulation prototype

Deliverables:

- Functional autopoiesis simulation engine (identity + gates + coherence loop)
- Quantum-autopoietic gate library and trajectory generator
- Visualization of identity persistence and symbolic accumulation dynamics

- Experimental scenarios (varying coherence, gate sets, channels)
- Evaluation report linking simulation behavior to theoretical model
- Technical documentation of architecture and assumptions

The topics are distributed by the rule: First come, first served.

General policy: AI is allowed and recommended. Submitting AI-generated nonsense, confusing, or otherwise cursory and blurry reports will result in an automatic failure. Please read the Generative AI Use Policy below.

Grading Scheme

The course assessment is based on a continuous project developed through five milestones. Each milestone is graded on a scale from 5 to 10 points. All milestones represent minimum requirements; therefore, no failing grade is assumed as long as the milestone is submitted and meets basic functionality and integration criteria.

All milestones are required. Since minimum functionality and integration are expected, grades below 5 are not assigned. Missing milestones may result in course incompletion.

Project goals and scope are defined collaboratively between the instructor and each student or team during the early phase of the seminar to ensure alignment with the overall architecture.

Milestones and Evaluation Responsibility

1. **Milestone 1 – Topic and Context Definition (M1: March 6)**
   Definition of the selected service, domain context, and inputs/outputs aligned with the shared architecture.
   Graded by Prof. Dr. Stankovski
2. **Milestone 2 – Component Architecture and Interfaces (M2: March 24)**
   Service architecture, integration interfaces, and placement in the layered system.
   Graded by Assist. Prof. Dr. Kochovski .
3. **Milestone 3 – Functions Implemented (M3: April 17)**
   Operational implementation of a service with domain logic and realistic data.
   Graded by Prof. Dr. Stankovski.
4. **Milestone 4 – Prototype, Integration and Deployment (M4: May 26th)**
   Integration of the services/components presented in Milestone 2 and 3 into a fully operational platform.
   Graded by Assist. Prof. Dr. Kochovski.
5. **Milestone 5 – Final Demo and Report (M5: June 10th)**
   Final integrated demonstration, documentation, and presentation of the service within the overall architecture.
   Graded by Prof. Dr. Stankovski.

Final Course Grade

Final grade = (M1 + M2 + M3 + M4 + M5) / 5

where M1–M5 are milestone grades (5–10).

Rounding

Intermediate calculations may use decimals.

The final grade is rounded to the nearest integer using standard rounding:

7.49 → 7

7.50 → 8

Only the final rounded integer grade (5–10) is recorded.

Generative AI Use Policy and Declaration

The use of generative AI and AI-assisted technologies ("AI tools") is allowed and recommended in this seminar as part of the development, coding, analysis, and writing process. When used responsibly, such tools can support efficient work, exploration of ideas, code generation, literature synthesis, and improvement of clarity and language.

However, AI tools must never be used as a substitute for human understanding, critical thinking, or technical contribution. All submitted work must represent the student's own comprehension, design decisions, implementation effort, and analysis.

Students remain fully responsible and accountable for all submitted content. This includes responsibility for:

• Carefully reviewing and verifying the correctness, coherence, and relevance of any AI-generated material

• Ensuring all outputs, code, and text are technically meaningful and accurate

• Editing and adapting AI-assisted content so that the submission reflects the student's own reasoning and contribution

• Ensuring transparency regarding the use of AI tools

• Respecting data privacy, intellectual property, and tool usage conditions

Submitting unedited, superficial, incorrect, confusing, or otherwise low-quality AI-generated material will be considered inadequate work and will result in automatic failure.

AI tools must not be listed as authors or contributors.

Declaration of Generative AI Use

If generative AI tools were used in preparing the report, code, or other materials, students must include a short declaration in their submission. The declaration should be placed at the end of the report before references.

Example declaration

Declaration of Generative AI Use

During the preparation of this work the author used [NAME OF TOOL / SERVICE] to support [coding / drafting / analysis / language editing]. The author reviewed and edited all outputs and takes full responsibility for the final content.

If no generative AI tools were used, no declaration is required.

Vision and Method: Convergence of Contextual Decisions Through Reality-Checked Orchestration

The architecture assumes a technological landscape in which artificial intelligence services are already widely available across domains. The central objective is therefore not the development of new AI models, but the orchestration of existing AI capabilities to support high-level contextual decisions such as sustainability, quality, compliance, and operational reliability and to enable measurable convergence between human and AI decisions over time.

The system is grounded in the principle that decisions are meaningful only relative to context. A contextual state is set within an active business process, and may include domain conditions, actors, constraints, and goals. Semantic orchestration invokes relevant AI services to generate decisions for that particular context. Decisions can be numeric, textual, or functional in nature.

Human experts may independently produce decisions for the same situation. The architecture persistently stores both human and AI decisions together with their originating context, creating a longitudinal repository of contextual decision observations across domains.

The core hypothesis is that, as contextualization and orchestration improve, AI decisions should increasingly align with human expert (science) decisions in comparable contexts. Convergence is therefore defined as increasing agreement and stability of decisions across repeated contextual situations.

Reality-Check Experiments as the Core Method

Convergence is evaluated through continuous reality-check experiments. Each experiment consists of presenting the same contextual situation to both AI services and human experts and recording their decisions. Contextual inputs may include institutional data, regulatory texts, environmental conditions, or system status, depending on the domain.

The resulting paired decisions are stored together with context and subsequent outcomes when available. This enables repeated comparison of decisions under similar conditions and observation of how alignment evolves over time. Reality-check experiments therefore ground evaluation in observable situations rather than abstract benchmarks, ensuring that orchestration remains anchored in real-world contexts.

Measurement of Precision and Accuracy in Contextual Decisions

Decision evaluation is based on relational comparison rather than isolated correctness labels. Accuracy measures agreement between AI and human decisions within the same context. Precision measures consistency of AI decisions across similar contextual situations. High precision indicates stable interpretation of context, while high accuracy indicates alignment with human expertise.

By analyzing stored context–decision pairs over time, the architecture evaluates whether AI recommendations become both more accurate relative to human judgment and more precise across repeated contexts. Convergence is observed when both measures increase longitudinally.

Four Motivating Use Cases

The demonstration evaluates convergence across four domains that share the same contextual orchestration architecture but differ in domain knowledge.

In the orchestration domain, context describes system state, resource availability, and operational constraints. Decisions concern deployment actions, scaling strategies, or service selection aligned with reliability and sustainability objectives. Human operators and orchestration AI produce comparable operational decisions.

In the legal domain, context includes regulatory texts, institutional conditions, and compliance objectives. Decisions consist of advisory interpretations or compliance recommendations. Human legal experts and legal AI services evaluate the same contextual situations.

In the quality-assurance domain for education, context includes institutional indicators, accreditation criteria, and improvement goals. Decisions concern quality evaluation or recommended actions. Human quality experts and QA AI services assess identical contexts.

In the environmental or weather domain, context includes meteorological or environmental data and operational implications. Decisions involve interpretation of conditions or risk advisories. Human environmental interpretation and AI forecasting outputs are compared under the same contextual conditions.

Across all four domains, the architecture records contextual states and corresponding human and AI decisions, enabling longitudinal evaluation of convergence within and across domains.

Vision Outcome

The envisioned system is a continuously learning contextual decision environment in which AI and human decisions progressively converge under shared contextual understanding. Through reality-check experiments, precision and accuracy measurements, and cross-domain comparison, the architecture transforms distributed AI services into a coherent decision ecosystem grounded in context, traceability, and longitudinal alignment.