- Discord: https://discord.gg/SzZHZzBB
- MSTeams channel and folder: Fog Computing for Smart Services | Projects 2025 | Microsoft Teams
- Git: *will be made available soon*

ExtremeXP: https://extremexp.eu/

No	Student	Topic	Description
The ExtremeXP related tasks are supported by the following team from our laboratory: - Doc. dr. Petar Kochovski - on the overall requirements, implementation, MDP etc. - Day-to-day questions Mohammad Pezheski (MDP Swagger APIs and upgrades), Pouriya Miri (on conducting experiments), prof.dr. Kristina Veljković (on MDP) - Within the FOG, the activities are coordinated by Matic Conradi			
1	reserved	Contextualisation of experiments, mapping context to MDP	This task involves developing a comprehensive knowledge graph based on specific contextual criteria. In particular: (1) An approach for contextualization (2) An approach for mapping context to specific elements of the MDP process (see published articles) (3) Any upgrades of the existing MDP APIs that would make it possible to utilize complex knowledge graphs in the process (e.g. expressed in OWL2 or RDF). In addition, since this is a MSc work, contextualization of 100.000+ articles to be used for demonstration, testing, etc.
1	available	Figma design and GUI implementation of a Contextualizer and Options Explorer components	Interactive interfaces will be developed to enable efficient querying and management of a knowledge base of scientific experiments. As a minimum: (1) Context capturing (2) Exploration of 4 important variability - complex decision-making points: (a) selection of data, (b) selection of algorithm, c) parametrization, (d) visualization (3) The design should allow gradual building of an experiment, addition of new states, actions, activities etc. in the MDP and similar (4) MDPs states changes management by benefiting from / and improving the APIs
1	available	Five databases of ExtremeXP	The goal of this task is to prepare 5 separate knowledge graphs based on data from the 5 ExtremeXP experiments

1	reserved	Integration over the full-stack, dockerization and deployment	A CI/CD pipeline to automate the build, testing, and deployment processes to enable continuous integration of updates, ensuring that new features and refinements are deployed efficiently without downtime.

Swarmchestrate: https://www.swarmchestrate.eu/
ACES: https://www.aces-edge.eu/

	Student	Topic	Description
Engaged in these activities: - Andrea Roberta Costragiola - Univ. of Bologna (PhD student) - Doc. dr. Petar Kochovski - MSc thesis: Žan Pižmoht System for Ensuring Trust in the Pharmaceutical Supply Chain Using Blockchain			
1	reserved	System for Ensuring Trust in the Pharmaceutical Supply Chain Using Blockchain	**Define and present a trust management system architecture** Select blockchain framework (Hyperledger Fabric, Ethereum, or alternative). Set up blockchain nodes and consensus mechanism for trust-related transactions. Deploy smart contract execution environment for trust evaluation and credential issuance. 1: Ontology of Trust Design and formalize a trust ontology using OWL or RDF, defining key trust concepts, entities, relationships, and metrics. Develop a knowledge base to store trust-related data, including direct and indirect trust factors. Integrate multiple data sources (IoT, enterprise systems, third-party validators) to populate and update the knowledge base. Implement semantic reasoning capabilities to infer additional trust relationships from existing data. Ensure ontology alignment with industry standards (e.g., W3C, ISO, EBSI Verifiable Credentials). 2: Trust Algorithms & Reasoning Layer Develop trust algorithms that process the ontology and compute trust scores based on selected metrics. Implement a reasoning engine (e.g., Apache Jena, OWL Reasoner, RDF4J) to derive trust levels from the knowledge base. Design a modular framework that allows different trust models (reputation-based,

			context-aware, consensus-based) to be applied dynamically. Optimize algorithms for scalability and computational efficiency in handling large datasets. Integrate machine learning techniques to refine trust models based on historical performance and feedback. Ensure algorithmic transparency and explainability by providing logs and justifications for trust score calculations. 3: Smart Contract & Oracle for Trust Assessment Develop smart contracts to formalize trust evaluation logic on blockchain. Implement an oracle mechanism to query external data sources and feed verified trust data into the blockchain. Enable dynamic trust assessment calls where individual users, devices, or systems can invoke the smart contract to assess trust in real time. Ensure security and integrity of the trust assessment process through cryptographic signatures and verifiable credentials. Optimize smart contract gas efficiency for blockchain-based trust computations. Implement off-chain and on-chain interactions to balance performance and decentralization. Develop verification mechanisms that allow third parties to validate the issued trust credentials. Additional System Components & Deployment Develop APIs for trust queries and integration with ERP, supply chain, and identity management systems. Implement role-based access control (RBAC) for accessing trust assessment results. Provide visualization and analytics tools to track trust scores and trends. Conduct security audits and penetration testing to prevent manipulation or exploitation of trust mechanisms. Deploy monitoring and logging mechanisms for continuous trust system evaluation. Define update and maintenance procedures for the ontology, reasoning layer, and smart contracts. Perform real-world pilot deployment and refine trust models based on empirical data.
2	reserved	Trust management system for	The objective of this task is to design and implement a Trust Management System that estimates trust levels based on multiple types

		cloud-to-edge computing	of measurements, including direct trust (derived from firsthand interactions) and indirect trust (based on third-party recommendations or historical data). These trust measurements will be collected from various sources, requiring aggregation and processing to generate a reliable trust score. A key aspect of this system will be ensuring the correctness and integrity of incoming data before incorporating it into the trust calculations. Data validation mechanisms will be implemented to detect inconsistencies, filter out unreliable sources, and prevent manipulation or bias in trust estimation. Only verified and trustworthy data will be used for the aggregation process. Once the trust score is computed, the system will generate verifiable credentials to represent the trust assessment in a secure and tamper-proof manner. These credentials will adhere to recognized standards (e.g., W3C Verifiable Credentials) to ensure interoperability and cryptographic security. The system should support verification mechanisms that allow third parties to validate the authenticity of the issued credentials. (check also the description of the project above) Deploy using containerized microservices
1	reserved	Implementation of a Veramo-Based DID Issuing, Registry, and Verification Service	1. Set Up Infrastructure: Deploy Veramo framework, choose DID registry backend (blockchain, IPFS, Web2 storage), and configure API endpoints. 2. DID Issuance Module: Implement multi-method DID creation, key management, and secure storage. 3. DID Registry & Resolution: Develop a universal DID resolver with cross-network support, efficient lookup mechanisms, and metadata storage. 4. DID Verification Service: Implement Verifiable Credential (VC) life cycle management (issue, modify, revoke), cryptographic signature validation, and Zero-Knowledge Proof (ZKP) support. 5. Compliance: GDPR/eIDAS/EBSI compliance checks. 6. Integration & Testing: Deploy using containerized microservices (Docker/Kubernetes), perform API and blockchain interoperability testing, and optimize performance.

ALASTRIA: https://alastria.io/en/home/
EBSI: https://ec.europa.eu/digital-building-blocks/sites/display/EBSI/Home

No	Student	Topic	Description
This activity is coordinated by: - Doc. dr. Petar Kochovski - Luka Maček			
1	available	Test ALA network and their interfaces	Testing Plan for DID-Based Trust Management System Using Alastria Network 1. Infrastructure & Deployment Testing Verify correct deployment of Veramo and Alastria DID services. Ensure Alastria network nodes are properly connected and synchronized. Test API endpoints for DID issuance, resolution, and verification. Validate network latency and transaction processing times. 2. DID Issuance & Registration Testing Issue DIDs using different Alastria-supported methods. Validate correct cryptographic key generation and storage. Ensure DIDs are registered on-chain and retrievable. Test DID update and revocation mechanisms. 3. DID Resolution & Lookup Testing Query and resolve issued DIDs from the Alastria network. Check data integrity and metadata retrieval. Test DID interoperability with other networks. 4. Trust Management & Reasoning Engine Testing Verify integration of trust algorithms with the DID registry. Validate trust score computation based on direct and indirect trust. Test the system's response to data inconsistencies or manipulation attempts. 5. Smart Contract & Oracle Testing Deploy and test smart contracts handling trust assessments. Verify oracle integration for fetching off-chain trust data. Ensure transaction correctness and gas efficiency. 6. Verifiable Credentials (VC) Issuance & Validation Issue and verify W3C-compliant Verifiable Credentials. Test signature validation and credential revocation. Validate privacy-preserving proofs (ZKPs) for trust verification. 7. Security & Compliance Testing

			Perform penetration testing on API endpoints and smart contracts. Validate compliance with GDPR/eIDAS and Alastria trust model requirements. Ensure secure storage and transmission of trust-related data. 8. Performance & Load Testing Simulate high transaction loads and measure system performance. Test scalability of DID issuance and resolution. Optimize query and transaction efficiency under different network conditions. 9. Integration & Interoperability Testing Ensure compatibility with Alastria ID framework and external identity providers. Test interoperability with Hyperledger, EBSI, and other blockchain identity frameworks. Validate API integration with third-party applications. 10. User Acceptance & Pilot Testing Conduct real-world testing with sample entities (organizations, devices, AI models). Collect feedback on trust assessment accuracy and system usability. Finalize deployment strategy based on test results.
1	available	Implementation of an EBSI-Based DID Issuing, Registry, and Verification Service	The system of trust for EBSI environment. Same requirements as above.

BUILDCHAIN: https://buildchain-project.eu/

No	Student	Topic	Description
This activity is to be coordinated by: - M.Sc. thesis: Krivec, Jan: Uporaba decentraliziranih identitet pri decentraliziranem dnevniku stavb - M.Sc. thesis: Hoffman Jeglič, Lucas Anthony: Realizacija modela digitalnega dvojnika za samosuvereno identiteto - Rron Jahja - 3D modeling - One more student			
	reserved	Realizacija modela digitalnega dvojnika za samosuvereno identiteto Development	- AI or 3D browsing capability in a 3D BIM model of a building - Linked to DIDs - Linked to sensor data - Consider any IFC compatible BIM model Develop a comprehensive Digital Twin that integrates Building Information Modeling (BIM),

		of a Digital Twin for Identity & Role Management in BUILDCHAIN	digital identities for persons, objects, and services, and real-time sensor data. This system will leverage blockchain-based identity verification (ONCHAINID) and role-based access management (ERC-3643) to ensure secure and traceable interactions. The project draws inspiration from the BUILDCHAIN initiative, applying its methodologies to enhance interoperability, automation, and trust in digital asset management. Initially this project should start by cherry-picking what is interesting under BUILDCHAIN, and then, defining users, roles, data, and user journeys to be demonstrated. The architecture and design should allow for the management of user identities and roles efficiently using blockchain-based smart contracts. It may consist of: 1. Smart Contract Layer – Implements ERC-3643 & ONCHAINID. 2. GUI Layer – A web-based interface for managing identities and roles. 3. API Layer – Exposes REST and Web3-based endpoints for external service integration. 1. Smart Contract Architecture ● ONCHAINID Smart Contracts: ○ Manages decentralized identities (DID). ○ Stores verification information. ○ Links identities to roles and permissions. ● ERC-3643 (Permissioned Token Standard): ○ Ensures only verified identities can interact with tokenized assets. ○ Implements compliance rules for transactions based on roles. ● Role-Based Access Control (RBAC) Smart Contract: ○ Manages roles (Admin, Issuer, Validator, Holder, etc.). ○ Assigns permissions based on user roles. ○ Allows dynamic role modifications. 2. API Layer (can be additionally modified/updated during the semester)

			• User Identity Management
			  ○ POST /identities/register – Register a new identity.
			  ○ GET /identities/{id} – Get identity details.
			  ○ PUT /identities/{id} – Update identity information.
			  ○ DELETE /identities/{id} – Revoke an identity.
			• Role Management
			  ○ POST /roles/create – Create a new role.
			  ○ GET /roles/{role_id} – Get role details.
			  ○ PUT /roles/{role_id} – Update role permissions.
			  ○ DELETE /roles/{role_id} – Remove a role.
			• Role Verification
			  ○ POST /verify-role – Verify if an identity holds a specific role.
			  ○ GET /user-roles/{identity_id} – Fetch roles assigned to an identity.
			4. Technology Stack
			• Smart Contracts in Solidity (ERC-3643, ONCHAINID)
			Backend
			• Any Node.js based framework
			Infrastructure
			• Ethereum or Polygon test networks for smart contract execution.
			• Hardhat local network instance
			• Infura/Alchemy for blockchain connectivity.
			
			The Digital Twin acts as a virtual representation of buildings, infrastructure, and their operational environment, ensuring:
			• BIM-based modeling of building structures and components.
			• Integration of digital identities for stakeholders, materials, and services.
			• Real-time monitoring using sensor data for decision-making.
			• Blockchain-enabled verification of data provenance, transactions, and permissions.
			Key Components
			• Component Description: BIM Model Digital representation of buildings and assets
			• Digital Identities: ONCHAINID-based authentication for persons, objects, and services
			• IoT Sensors: Collect and log real-time

			environmental and structural data • Smart Contracts: Role-based access control and automation via ERC-3643 • Decentralized Knowledge Graph (DKG): Semantic mapping for interoperability and compliance Expected results: • Digital Twin Model (BIM, sensor data, identity layer) • Blockchain-Verified Identity & Access Control (ONCHAINID, ERC-3643) • Use Case Implementation & Smart Contracts • Ontology Mapping & Compliance Assurance • Testing Reports validating security, efficiency, and automation
1	available	Test ERC-3643 and ONCHAINID with BIM model data and digital identities for buildings	The assigned student will identify and collect all necessary components for the Digital Twin, including BIM model data, digital identities of buildings, persons, objects, and services, and sensor data, ensuring metadata standardization for Decentralized Knowledge Graphs (DKG); design a graphical user interface (GUI) for role-based identity management and lifecycle visualization; and define interoperability strategies between BIM models, IoT sensor data, smart contracts, and blockchain identity verification, ensuring seamless integration and compliance with BUILDCHAIN standards.

Summary of Use Cases for the topics above

Use Case ID	Use Case Name	Objective	Key Roles
UC-01	Structural Health Monitoring (SHM)	Secure logging & validation of sensor data	Owner, Inspector, Regulator
UC-02	Life-Cycle Analysis (LCA) & Carbon Footprint Tracking	Immutable tracking of material data & emissions	Owner, Auditor, Regulator
UC-03	Deep Renovation Workflows	Secure tracking of materials & approvals	Owner, Contractor, Regulator

UC-04	Smart Heritage Building Monitoring	Controlled access to cultural site maintenance	Conservator, Researcher, Regulator
UC-05	Post-Catastrophic Intervention System	Disaster response & recovery logging	Emergency Team, Regulator, Owner
UC-06	Construction Economics & Cost Estimation	Transparent financial reporting	Owner, Contractor, Auditor
UC-07	Management of Operational Energy Efficiency	AI-driven building energy optimization	Owner, Facility Manager, Regulator
UC-08	Flood Control & Precipitation Monitoring	IoT-driven early warning system	City Planner, Hydrologist, Regulator

EUDI — Physical Persons and Enterprise Wallets

Other wallets can be considered (e.g. ValidatedID)[1]

No	Student	Topic	Description
1	available	Test EUDI wallet	The goal is to identify, test, and develop an interoperable EUDI (European Digital Identity) wallet by evaluating open-source implementation(s), ensuring compliance with interoperability standards, and creating a mobile application capable of storing and managing digital identities. Tasks: ● Identify Open-Source EUDI Wallet Implementations: Research existing open-source EUDI wallet projects and evaluate their compatibility with interoperability frameworks and regulations. ● Test and Validate Open-Source Code: Deploy and test the selected wallet implementations, ensuring they support key functionalities such as digital identity storage, verification, and cryptographic security. ● Develop a Mobile Application: Build a prototype mobile app that integrates an EUDI wallet, allowing users to store, manage, and verify digital identities securely. ● Ensure Interoperability with BIM & Blockchain Identities: Test the wallet's

[1] The wallet must be first approved by the professor

			ability to interact with BIM-based digital identities, smart contracts, and Decentralized Knowledge Graphs (DKG). ● Security & Compliance Testing: Perform penetration tests, ensure GDPR compliance, and validate identity authentication mechanisms. Deliverables: A tested open-source EUDI wallet implementation with documented findings. A functional mobile application prototype capable of storing and verifying digital identities[2]. A technical report outlining interoperability, security, and compliance evaluations. A testing framework for future improvements and scalability.

No	Topic	Student	Description
1	Autopoietic Cognitive Edge-cloud Services	reserved	Theoretical study on AI and ML-enabled architecture to respond to the needs of cloud services at the edge Definition of the “AUTOPOIESIS” concept

[2] An operational mobile application will be provided upon request.