This course unit [CM-W-WAS] describes the content and the organization of the lecture "Web Applications and Service-oriented Architectures" (WASA) and the practical course Microservice2Go (M2Go) provided by the research group Cooperation & Management (C&M, Prof. Abeck). Since the lecture WASA is closely linked with the practical course M2Go, the lecture and the practical course can only be taken in combination (WASA-M2Go). Since the number of WASA-M2Go places is limited, interested students must apply for a place.

WASA1_M2Go1 (Bachelor): Current concepts of software development and architectures (including Microservices, REST, gRPC, Domain-Driven Design, DevOps, CI/CD, Build Pipelines, Container-virtualized Infrastructures) as well as related technologies and tools (including HTTP, Go, Swagger, Postman, JavaScript/TypeScript, Angular, GitLab-CI, Docker, Kubernetes, Prometheus) are introduced. These concepts and technologies are applied in a systematic engineering approach, called Unified Microservice Engineering (UME), to develop and deploy microservice-based web applications. A practical course, called Microservice2Go (M2Go), is offered in combination with the WASA1 lecture in which the UME approach is practically applied with the example of a car rental application.

WASA2_M2Go2 (Master): A compact summary of the concepts covered by WASA1 is provided. In WASA2, Identity and Access Management (IAM) as an advanced topic is presented. IAM is a highly relevant part of the digitization strategy of each organization. In the lecture, leading IAM solutions and products are introduced to illustrate how the IAM challenges are solved in IT practice. A practical course, called Microservice2Go (M2Go), is offered in combination with the WASA2 lecture in which an existing car rental application is extended by IAM functionality dealing with different topics, such as OIDC/OAuth-based authentication and authorization, decentralized identities, and authorization policies.

The WASA kickoff lecture will take place

=== on Wednesday, 25th October 2023 at 9:45 am
=== in the Building 50.34 (Informatikgebäude am Fasanengarten), Room SR301

Each student who wants to take part in the WASA kickoff lecture must

=== send an email (in German) to cm.research@lists.kit.edu

to apply for one of the restricted WASA_M2Go places. Please do only use your depseudonymized KIT student email address (see https://my.scc.kit.edu/shib/pseudonymisierung.php for further information).

The lecture material is made available in English. During the lecture, the content is presented and discussed in German. The oral examination is conducted exclusively in German. All students write their practical/seminar thesis in English. Thesis templates are made available in LaTeX. Overleaf is used for the writing of the practical and seminar thesis.

C&M Cooperation & Management
KIT Karlsruhe Institute of Technology
M2Go Microservice2Go
WASA Web Applications and Service-oriented Architectures
The research work, carried out by C&M, can be divided into two main areas:

(Microservice Engineering) For the business domain Connected Car, applications based on the concept of domain modeling and microservice architectures are developed. Relevant concepts applied in the microservice engineering approach include Behavior-Driven Development (BDD) and Domain-Driven Design (DDD). In addition to development (Dev), the operational aspects (Ops) are intensively taken into account. DevOps concerns the continuous integration (CI) and continuous deployment (CD) of microservice-based software systems into a container-virtualized (Docker/Kubernetes-based) cloud infrastructure.

(Identity and Access Management) Identity and Access Management (IAM) is a highly relevant crosscutting concern appearing in every web application. Advanced IAM topics are (i) decentralized identities which make the user the owner and controller of their identity data and (ii) fine-grained authorization which takes access decisions based on a broad spectrum of different attributes.

BDD Behavior-Driven Development  
CI/CD Continuous Integration / Continuous Deployment  
DDD Domain-Driven Design  
IAM Identity and Access Management  
DevOps Development and Operations
A microservice architecture is located on the application plane as it is introduced in ++Network, System, and Application Plane++. While the software architecture is described by the logical layers specified by a specific DDD pattern LAYERED ARCHITECTURE, the system architecture introduces several subsystems (domain microservices, application microservices, API gateway). Two types of application programming interfaces (domain microservice API, application microservice API) are separating the logical layers on the software architecture side and the microservices on the system architecture side.

(Presentation Layer) This layer renders the UI elements in the browser. Technologies that support the implementation are Angular and Bootstrap. The presentation includes a logic which controls the interaction with the application microservice API. An optional API gateway is often used to provide cross-cutting concerns, such as load balancing or security aspects.

(Application Logic Layer, Application Microservice) This layer realizes the orchestration of domain microservices in order to provide the application logic to fulfill the requirements made to the application. A technology that supports the implementation of this functionality is Spring.

(Domain Logic Layer, Domain Microservice) This layer implement the domain microservices which mainly are CRUD operations on the domain objects.

(1) In contrast to a traditional three-layer application architecture, the business logic layer in a microservice architecture is split into two layers, the domain logic layer and the application logic layer. The reason for that is to promote the reuse of business logic functionality by distinguishing between application-agnostic (= domain logic) and application-specific (= application logic) functionality.

<table>
<thead>
<tr>
<th>API</th>
<th>Application Programming Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUD</td>
<td>Create, Read, Update, Delete</td>
</tr>
</tbody>
</table>
The Unified Microservice Engineering (UME) approach unifies two former approaches developed by C&M, called CMEng and MuleEng. UME consists of one process, the structured development process consisting of the well-known phases analysis, design, implementation and test, and deployment and operations. In the UME approach, the domain-driven aspects are not necessarily part of the engineering process (i.e., they are optional).

(Application Requirements Analysis) The requirements are expressed by use cases with a specific structure. Optional analysis artifacts are vision and goals or the application sketch.

(Application Term List) Application-specific terms are not introduced as a ubiquitous language. Ubiquitous languages are only provided by domains.

(External System Requirements Analysis) The consideration and integration of the external systems (esp. enterprise applications, business services, or databases) into the microservice application which is to be developed is a main characteristic of MuleEng. This is a highly relevant aspect in practice which was not sufficiently taken into account in CMEng.

(Domain-specific Analysis) In UME, the consideration of domain-specific aspects based on the concept of Domain-Driven Design (DDD) is intentionally kept optional. This analysis leads to Domain APIs which become part of the software architecture.

(DDD Artifacts) These include the ubiquitous language(s) and the domain model(s) of the domains relevant for the application to be developed.

(Software Architecture Design) The software architecture consists of (i) Process APIs which are derived from the use cases, (ii) System APIs which integrate the external systems, (iii) optional Domain APIs which provide the domain-specific logic, and (iv) Experience APIs which support the requirements of the different user interfaces.

(User Interface Design) User Interface Implementation and Test) (Experience API) As soon as the experience API is specified, the User Interface (UI) can be designed and implemented and tested in parallel to the design and implementation of the different API types.

(API Design and API Specification) The API is systematically specified based on an API diagram. The API diagram of a Process API is derived from the use case descriptions and the software architecture diagram.

(API Implementation and Test) In UME, the implementation and test usually follows the CMEng approach in which the microservices are coded by using a Java- or Go-based framework. Alternatively, the MuleEng approach can be applied, in which the implementation is based on so-called Mule flows (low code / no code).

Deployment and Operations) The template-based deployment approach used in both the CMEng and the MuleEng approach is applied in the UME approach.

UI: User Interface
UME: Unified Microservice Engineering
The software architecture of the DLAKaApp and BestRentalApp contains no domain layer, as considering domain-specific aspects is optional in the Unified Microservice Engineering C&M (UME) approach, along which the DrivingLicensePoC is developed. The consideration of domain-specific aspects would add more complexity to the DrivingLicensePoC and would not contribute to the demonstration of decentralized identities, which is the goal of the DrivingLicensePoC.

(Presentation Layer) Contains the user interfaces (UI) of the DLAKaApp and the BestRentalApp.

(Application Logic Layer) Both application microservices AM-DrivingLicenseManagement and AM-RentalManagement offer a gRPC API to the UI and require an interface to a verifiable credential system, which assists the application microservices with issuing or verifying VC-DLs.

(CallbackAPI) Via this API the verifiable credential system can notify the microservice about updates or errors in the issuance or verification process.

(Infrastructure Layer) The modeling of the infrastructure layer is based on the concrete realization of a verifiable data system by Microsoft, called Entra Verified ID.

Remark: The names of the components provided by Microsoft did not align with the upper camel case naming convention of C&M. However, the names were not adapted to be more comprehensible and consistent.

(Verifiable Credential Request Service) Part of Microsoft Entra Verified Id. It offers a REST API that can be used to initiate an issuance and verification process.

(Request Service REST API) API via which AM-DrivingLicenseManagement initiates the issuance process and via which "VC Wallet" requests a verifiable credential.

(VC Wallet) Part of the Authenticator App and is responsible for requesting and storing verifiable credentials and providing verifiable presentations.

(DLAKaWellKnown, BestRentalWellKnown) Web servers which provide the public keys the "VC Wallet" needs to verify the signature of DLAKa and BestRental.
The acronym WASA stands for "Web Applications and Service-oriented Architectures". The following courses are offered: (i) lecture courses WASA1 and WASA2 (ii) practical courses M2Go1 and M2Go2 associated to the lecture courses (iii) proseminal course and seminar (iv) key qualification course (germ. Schlüsselqualifikation SQ).

Remarks: In the Wirtschaftsinformatik study programme the name of the module is "Microservice-basierte Web-Anwendungen".

(1) The lecture courses WASA1 and WASA2 each comprise 2 semester hours. A student who attends one of the lectures acquires 4 credit points (German: Leistungspunkt).

(2) The practical course M2Go is closely linked with the lecture course. The practical course counts 5 credit points meaning a workload of 150 hours.

(3) A proseminal and seminar count 3 credit points meaning a workload of 90 hours. A student who wants to carry out a (pro)seminar in parallel to WASA_M2Go will have a workload of 360 hours (4+5+3 = 12 credits) in the semester.

(4) The examiners are Prof. Abeck and one of the C&M's PhDResearchers. Since the examination is in the last week of the lecture term, the students should have a good personal resource management in order to have enough time for the preparation of the examination.

**SQ** Schlüsselqualifikation (Key Qualification)
Remark: The lecture plan can only be accessed by students who will be accepted as WASA_M2Go participants.
The WASA lecture content and the M2Go practical course content are closely coupled.

M2Go Participants can make contributions to the presentation of the M2Go content in the WASA lecture.

Each M2Go Participant documents the found solutions to the exercises in an individual thesis document.

A participation in the practical course M2Go requires about 12 hours per week.

The workload is accruing continuously and immediately. Therefore, students interested in participating in M2Go, must make sure that they have enough time resources before they decide to apply for a WASA_M2Go place.
(1) (2) Besides the traditional (one-way) lecture part, each lecture event additionally consists of an interactive part which is shaped in a more dynamic way.

(2.1) (2.2) For each lecture event, the agenda is made available in [CM-G-WAS]. The content of the markdown file is in German since this is the language spoken in the WASA lecture.

(1) One of the goals pursued by chatbot systems is to support the transfer of knowledge. Based on the found literature, scenarios of the usage of such systems in the area of education should be investigated in this seminar thesis. Proposals for the support of C&M's teaching and research activities C&M's should be made.

(2) In addition to authorizing requests from human users, requests between microservices must also be authorized. This is especially necessary when aiming for a zero-trust architecture. In this seminar, the existing literature on service-to-service authorization will be reviewed and the concepts applied to the CarRentalApp.

(3) End-to-end testing ensures that the acceptance criteria of an application are met. Various approaches and tools found in the literature which address the testing of distributed systems should be examined and discussed. The result of the seminar should be a discussion/proposal on how to advance end-to-end testing in C&M's Unified Microservice Engineering (UME) approach and using the CarRentalApp covered by the lecture WASA and the practical course Microservice2Go.
(1) It is absolutely important that a student who participates in the WASA lecture and practical/seminar course has the necessary resources to cope with the workload (lecture: 120 hours, practical course: 150 hours).

(2.1) Check on the page of the KIT Steinbuch Computing Centre if your email is already depseudonymized.
(2.1.2) The motivation and the experiences should be summarized in at least one or two paragraphs.

(3) This email will be sent by the project team leads.
The current WASA_M2Go material is stored on the C&M Data Storage in the following folder:
\sccfs.sce.kit.edu\OE\TM\VR\Mitglieder\2-1.WASA_Aktuell
This is the most important folder which is available via \sccfs.scc.kit.edu\OE\TM\VR\Mitglieder\2-1.WASA_Aktuell
This function can be found in the Studierendeportal by clicking on "Meine Benutzerdaten" > "De-/Pseudonymisierung" and accepting "Ich stimme der Sichtbarkeit meiner namensbezogenen Daten zu". The name-related E-Mail-Adresse <prenom><surname>@student.kit.edu" exists additionally to the "uxxx@student.kit.edu" email address.