MINISTRY OF EDUCATION

TECHNICAL UNIVERSITY

OF CLUJ-NAPOCA, ROMANIA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

DISTRIBUTED SYSTEMS

Assignment 1

Request-Reply Communication

Prof. Tudor Cioara
As. Liana Toderean
As. Gabriel Antonesi

S.l. Marcel Antal
As. Alexandru Rancea

Conf. Cristina Pop
As. Dan Mitrea

2025-2026

1. Requirements

Design and implement an **Energy Management System** that permits authenticated users to access, monitor, and manage smart energy metering devices. The system will be constructed as a set of loosely coupled, containerized microservices, each deployed independently, and orchestrated through a reverse proxy and API gateway.

The solution will contain the following components:

- **Frontend Application (Web Client):** A browser-based interface that manages user interactions and restricts access based on user roles
- **Microservices Layer:** A set of RESTful microservices, each responsible for a specific function: **User Management**, **Device Management**, and **Authentication**
- **Reverse Proxy and API Gateway:** Acting as the entry point to the system, this layer routes incoming requests, performs request validation, and applies security policies such as authentication and authorization

The platform supports two distinct categories of end-users:

- **Administrator:** Can execute CRUD operations on user accounts, device entities, and their associations
- **Client:** Can log in and see the devices assigned to them

The complete system will be deployed using **Docker-based containerization**.

At a minimum, a device must include the following attributes: an ID, a name, and a maximum consumption value. Similarly, a user must be defined by at least an ID, a username, and a password. Additional attributes are expected to be introduced to offer a better representation of users and devices.

1.1. Component description

1. Frontend

- Provides login and role-based pages.
- **Admin role:**
    - Perform CRUD (Create, Read, Update, Delete) operations on users.
    - Perform CRUD operations on devices.
    - Assign devices to users.
- **Client role:**
    - View all devices linked to their account.

2. API Gateway

- The **API Gateway** contains validation logic to check tokens and user permissions and route all requests to the correct microservice.
- Ensures that requests are only forwarded if the user is authenticated and authorized.

3. Authorization Service

- Handles user login and register.
- Stores and verifies credentials using the **Credential Database**.
- Generates and returns a token that the frontend uses in subsequent requests.
- Prevents unauthorized access to restricted resources.

4. User Management Microservice

- Provides CRUD operations for user accounts.
- Stores user data in the **User Database**.

5. Device Management Microservice

- Provides CRUD operations for devices.
- Stores data in the **Device Database**.
- Supports mapping devices to users.

1.2. Implementation technologies:

- We recommend the following technologies: REST for microservices (Java Spring REST or .NET Web API), JavaScript-based frameworks for client applications (Angular or ReactJS), for storing data we suggest using PostgreSQL or MySQL, and Traefik for reverse-proxy. The only **mandatory** technology is Docker for the deployment of the developed services.

2. Deliverables

- ➢ A solution description document containing:
  - a) UML Deployment diagram.
  - b) Readme file containing build and execution considerations.
- ➢ Source files. The source files and the database dump will be uploaded onto the personal *gitlab* account created at the *Lab resources* laboratory work, following the steps:
  - o Create a repository on *gitlab* with the exact name: *DS2025_Group_LastName_FirstName_Assignment_Number*
  - o Push the source code and the deployment diagram (push the code not an archive with the code or war files)
  - o Share the repository with the user *utcn_dsrl*

3. Evaluation

3.1. Assignment Related Basic Questions:

During project evaluation and grading you will be asked details about the following topics:

- URI and URL; Web Clients and Web Servers; HTTP protocol; HTTP methods; Reverse Proxy; Query strings; Hidden variables; Cookies; Session; JWT; Object-Relational Mapping (ORM); REST Services
- Additional concepts discussed during the previous laboratory sessions

3.2. Grading

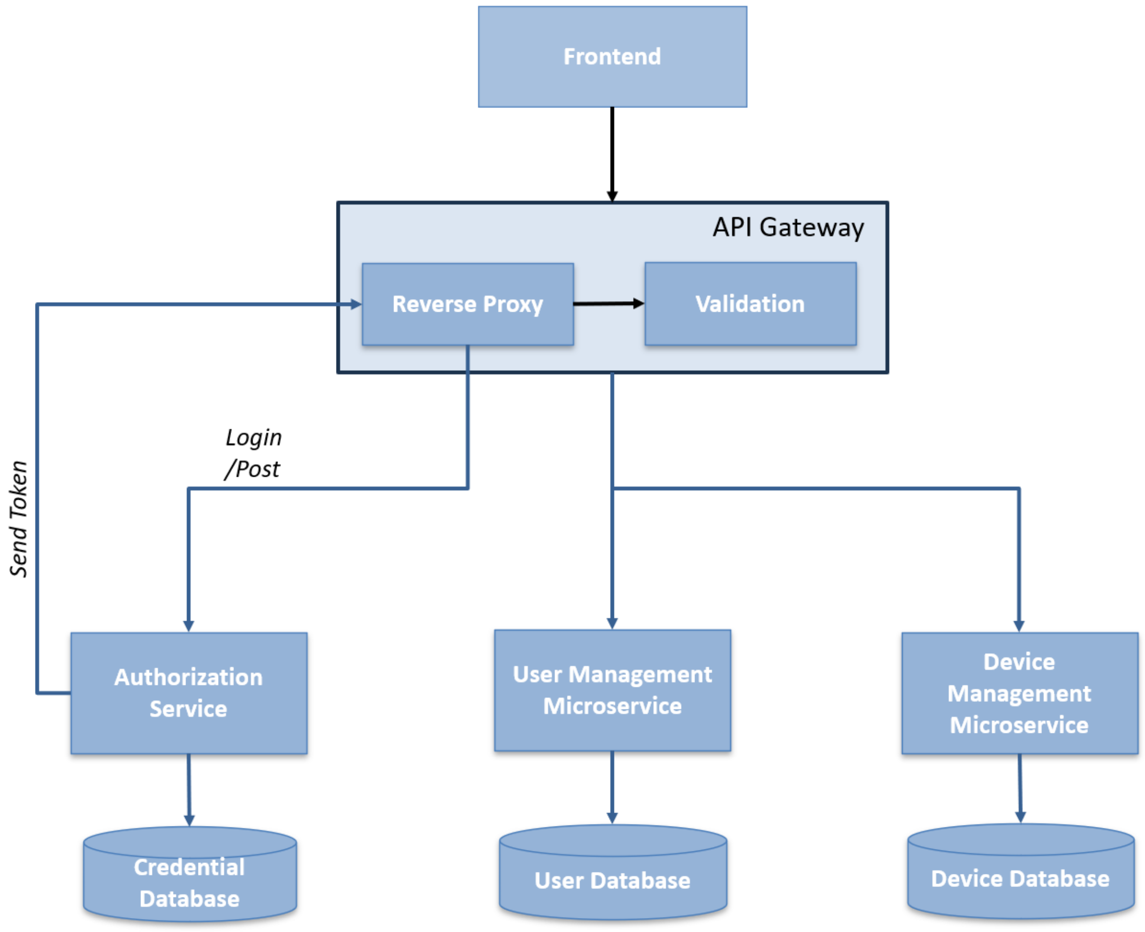
- The **assignment** will be graded as follows:

Points	Requirements
5 p	Minimum to pass: • User Microservice • Device Microservice • Basic Authentication • Frontend - CRUD on users and devices • Devices to users' associations • Readme file • Correct answers to 3.1 questions
2 p	Authentication Microservice
2 p	Frontend – Client & Admin role
1 p	Swagger documentation (path, HTTP method, short description, input params, response)

- The **project** will be graded as follows:

Points	Requirements
10 p	Minimum to pass: • Reverse Proxy • Docker deployment • Deployment diagram

4. Assignment conceptual diagram



5. Bibliography

1. Lab Book: M. Antal, C. Pop, D. Moldovan, T. Petrican, C. Stan, I. Salomie, T. Cioara, I. Anghel, Distributed Systems – Laboratory Guide, Editura UTPRESS Cluj-Napoca, 2018 ISBN 978-606-737-329-5, 2018, https://biblioteca.utcluj.ro/files/carti-online-cu-coperta/329-5.pdf

2. Lab Book: I. Salomie, T. Cioara, I. Anghel, T.Salomie, *Distributed Computing and Systems: A practical approach*, Albastra, Publish House, 2008, ISBN 978-973-650-234-7
3. https://swagger.io/docs/specification/v3_0/about/
4. https://docs.spring.io/spring-boot/index.html
5. https://react.dev/learn