SOA Credit Project

PARTICIPANTS

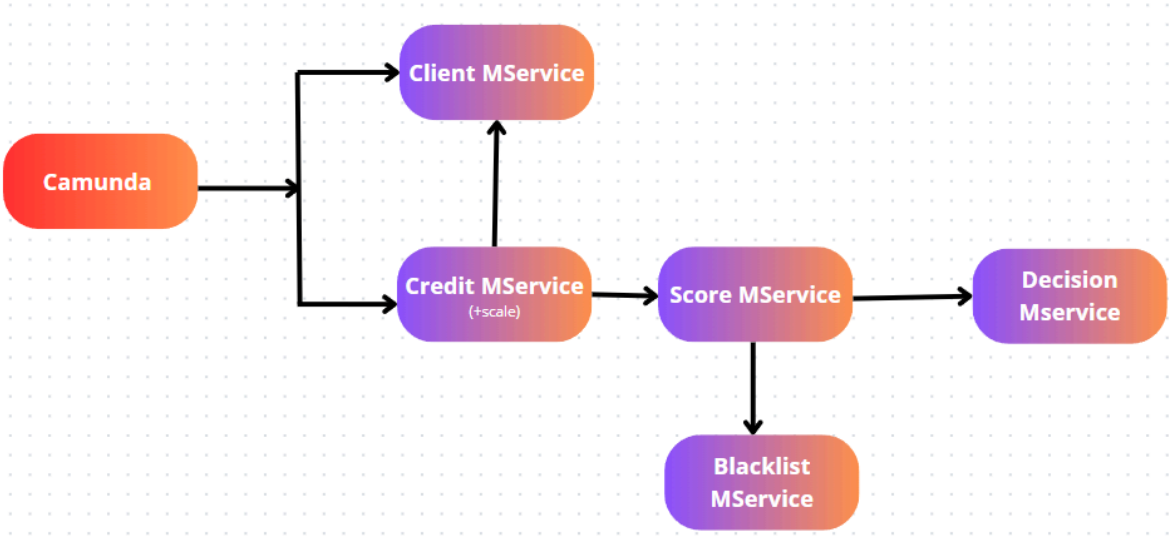
Mohamed Rayen Khlifi & El Kahla Nour & Ines Boukhris & Malak Zangar

3IDL01

System Architecture

High-Level Architecture Overview:

1. Client Service: Manages client data (create and retrieve client records).
2. Credit Service: Handles credit requests, and rate matching based on the client's request and selects the appropriate rate.
3. Scoring Service: Calculates the client's score and evaluates it.
4. Blacklist Service: Check if the client is blacklisted ensuring that clients in the blacklist are automatically flagged
5. Decision Service: Makes the final decision based on the calculated score.
6. Camunda BPMN Workflow Engine: Manages the credit granting process workflow, orchestrating communication between microservices.



Database Design

Every Microservice has a database (managed with PHPMyAdmin), consisting of multiple tables to store data from various services.

- For the credit Microservice we have 2 tables in the credit database :

credit table: id (pk), cin, scaleId, amount, interest, durationInMonths

Scale table: id (pk), interestRate,minimumDurationInMonths, maximumDurationInMonths, minimumAmount, maximumAmount

- For the client Microservice we have 1 table in the client database :

client table: id (pk) name, lastname, cin, salary, birthdate, contract

- For the blacklist Microservice we have 1 table in the bct database :

BlackListEntity table: id (pk), clientCin

- For the score Microservice we have 1 table in the score database :

Score table: id (pk), creditId, score

- For the decision Microservice we have 1 table in the decision database :

Score table: id (pk), credit_Id, created_at, Status_decision

Communication Between Microservices

In this project, all communication between the microservices is implemented using REST APIs, ensuring a lightweight, platform-independent, and scalable solution for data exchange.

- **For the client Microservice :**

  1. Create a New Client

Endpoint: POST /clients
Description:

- This API is used to create a new client.
- It checks if a client with the provided CIN already exists.
- If the client exists, it returns a conflict response with an appropriate message.

- If not, it creates a new client and returns the created client's details along with a success message.

2. Get Client by CIN

Endpoint: `GET /clients/{cin}`
Description:

- This API retrieves the details of a client using their CIN.
- It returns the `ClientDto` containing the client's details if found.

- **For the blacklist Microservice :**

1. Create a New Blacklist Entry

Endpoint: `POST /bct`
Description:

- This API creates a new entry in the blacklist.
- Accepts a `BlacklistDto` in the request body.
- Returns the created `BlacklistDto` along with a success message.

2. Check if a Client is Blacklisted

Endpoint: `GET /bct/blacklist/{clientCIN}`
Description:

- This API checks if a client, identified by their CIN, is in the blacklist.
- Returns a boolean value:
  - `true` if the client is blacklisted.
  - `false` if the client is not blacklisted.

- **For the score Microservice :**

1. Create a Score and Make a Decision

Endpoint: `POST /scores`
Description:

- **Input**: Accepts a ScoreRequestDto containing the CIN, creditId, and monthlyPayment.
- **Process**:
    1. Retrieves the client details using the CIN via the ClientService.
    2. Checks if the client is blacklisted using the BCTService.
        - If blacklisted, the score is set to 0.
        - If not blacklisted, computes a score using client details (salary, contract) and the requested monthlyPayment.
    3. Creates a ScoreEntity with the creditId and the computed score, then saves it via the ScoreService.
    4. Evaluates the score to determine the decision and sends the decision details (creditId and score evaluation) to the DecisionService.
- **Output**: Returns a ScoreDto containing the score and evaluation along with a success message.

- **For the decision Microservice :**

1. Create a Decision

Endpoint: POST /decisions
Description:

- **Input**: Accepts a DecisionRequestDto containing:
    1. creditId: The ID of the credit associated with the decision.
    2. evaluation: The evaluation result used to make the decision (e.g., "Red" or "Green").
- **Process**:
    1. Creates a DecisionEntity using the creditId and evaluation.
    2. Saves the decision to the database using the DecisionService.
- **Output**: Returns a DecisionDto containing the details of the saved decision along with a success message.

- **For the credit Microservice :**

1. Create a Scale

Endpoint: POST /scales
Description:

- **Input**: Accepts a ScaleDto containing scale details (e.g., interest rate, amount range, duration range).
- **Process**:
  - Adds the scale to the database using the ScaleService.
- **Output**: Returns the created ScaleDto along with a success message.

1. Create a Credit

Endpoint: POST /credits
Description:

- **Input**: Accepts a CreditRequestDto containing:
  1. CIN: Client identification number.
  2. amount: Requested credit amount.
  3. durationInMonths: Requested credit duration in months.
- **Process**:
  1. Uses the ScaleService to find the most suitable ScaleEntity based on the requested amount and durationInMonths.
     - If no suitable scale is found, returns an error message.
  2. Creates a CreditEntity using the CIN, scale ID, amount, calculated interest, and duration.
  3. Saves the credit entity via the CreditService.
  4. Initiates a score calculation via the ScoreService using the newly created credit details.
  5. Appends the score details (ScoreDto) to the created credit entity.
- **Output**: Returns the created CreditDto with associated score details along with a success message.

2. Retrieve Credits by Client ID

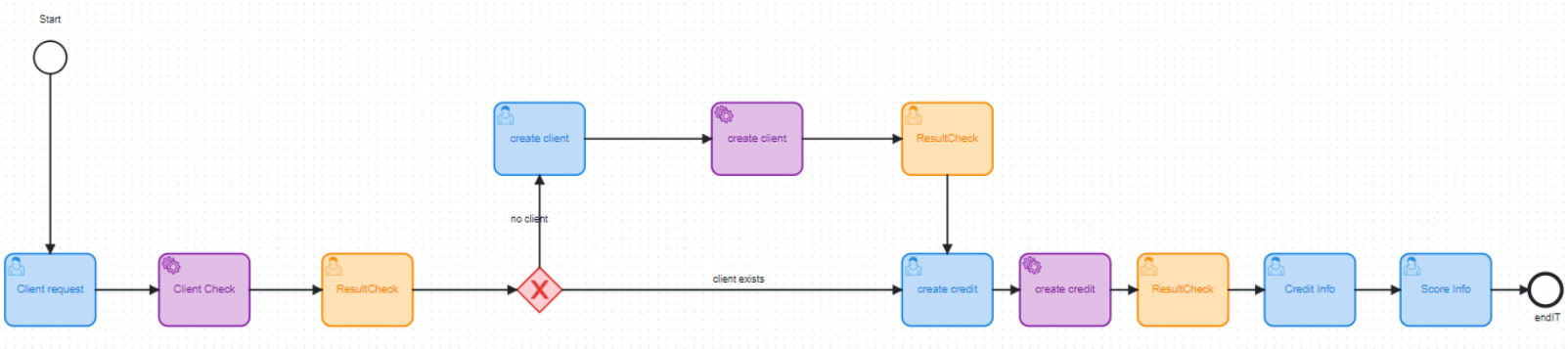
Endpoint: GET /credits/{clientId}
Description:

- **Input**: A clientId as a path variable.
- **Process**:
  - Fetches all credits associated with the given clientId using the CreditService.

- **Output:** Returns a list of `CreditDto` objects representing the client's credits.

Camunda BPM Workflows Design

Credit Workflow :



Proposed Workflow :