

**Projekt-Documentation**

Luca Dal Corso



Table of contents

[Project Introduction 2](#_Toc192511570)

[Vision 3](#_Toc192511571)

[Product Backlog 3](#_Toc192511572)

[Architecture 3](#_Toc192511573)

[Tech Stack 3](#_Toc192511574)

[API-Struktur 3](#_Toc192511575)

[Roles and Permissions 4](#_Toc192511576)

[Error Handling & Validation 4](#_Toc192511577)

[Sprint Plan 4](#_Toc192511578)

[Sprint 1 4](#_Toc192511579)

[Sprint 2 5](#_Toc192511580)

[Sprint 3 6](#_Toc192511581)

[Sprint 4 7](#_Toc192511582)

[Sprint 5 8](#_Toc192511583)

[Test 9](#_Toc192511584)

[Test Cases for Login Endpoint: 9](#_Toc192511585)

[Test Cases for Lead Endpoint: 10](#_Toc192511586)

[Test Cases for Customer Endpoint: 11](#_Toc192511587)

[Decisions 12](#_Toc192511588)

[Project Summary 13](#_Toc192511589)

**Projectname:** Lead Management System  
**Autor:** Luca Dal Corso

**Startdate:** 02.03.2025

## Project Introduction

The Lead Management System (LMS) is designed to efficiently manage leads and convert them into customers. It supports CRUD operations for leads and customers, with JWT token authentication for secure access.

This project doesn’t have a traditional UI but uses Swagger UI to interact with the system’s API, allowing users to test and manage the system via API endpoints.

The LMS aims to streamline lead management processes with a secure, scalable, and easy-to-use system.

### Vision

The goal of the project is to develop a Lead Management System that allows leads to be created, updated, deleted, and converted into customers.  
The system should provide a simple user interface, JWT authentication, and CRUD functions.

## Product Backlog

| **Number** | **Must/Can** | **Category** | **Description** | **Priority** |
| --- | --- | --- | --- | --- |
| **1** | Must | Functional | Lead create | High |
| **2** | Must | Functional | Lead update | High |
| **3** | Must | Functional | Lead delete | High |
| **4** | Must | Functional | Lead view | High |
| **5** | Must | Functional | Lead to customer | High |
| **6** | Must | Functional | Customer update | High |
| **7** | Must | Functional | View customer | High |
| **8** | Must | Quality | JWT Token Auth | Low |
| **9** | Must | Presentation | UI Design + Presentation | High |

## Architecture

### Tech Stack

* **Swagger UI:** The API documentation and interaction are handled via Swagger, making it easy to test the endpoints and communicate with the AP**I.**
* **Backend** (C#): The business logic and API endpoints are implemented using C#.
* **JWT Authentication:** Secure access to the endpoints through JWT tokens**.**
* **Database** (MySQL): Data is stored in a MySQL database.

### API-Struktur

| **Methode** | **Endpunkt** | **Beschreibung** | **Benötigte Rolle** |
| --- | --- | --- | --- |
| POST | /api/v1/lead | Create a new lead | Editor, Admin |
| GET | /api/v1/lead | Retrieve all leads | Viewer, Editor, Admin |
| PUT | /api/v1/lead/{id} | Update a lead | Editor, Admin |
| DELETE | /api/v1/lead/{id} | Delete a lead | Admin |
| POST | /api/v1/customer/{id} | Convert lead to customer | Admin |
| GET | /api/v1/customer | Retrieve all customers | Viewer, Editor, Admin |
| POST | /api/v1/login | User login and get JWT token | - |
| DELETE | /api/v1/customer | Delete a customer | Admin |
| PUT | /api/v1/customer | Update a customer | Editor, Admin |

### Roles and Permissions

* **Viewer**: Can view leads and customers.
* **Editor**: Can view and update leads and customers.
* **Admin**: Full access, including creating, updating, deleting leads and customers, and managing the roles.

### Error Handling & Validation

* **401 Unauthorized**: The user is not authenticated or does not have the required permissions to access the endpoint.
* **404 Not Found**: The requested resource (lead or customer) does not exist.
* **400 Bad Request**: The request body is malformed or invalid.

## Sprint Plan

Sprint 1 **(02.03.2025 – 03.03.2025)**

Goal**:**

* Create Datbase
* CRUD-Function Create
* CRUD- Function View
* Check connection between backend → database

Tasks**:**

| **Task** | **Status** | **Reason** | **Problems** | **Solution** |
| --- | --- | --- | --- | --- |
| **Database Setup** | Done | Project foundation | None |  |
| **CRUD Create Function** | Done | No tests possible without create function | API connected with UI | Research + examples |
| **CRUD View Function** | Done | Validate & display data | None |  |

#### Retrospective Sprint 1

| **Question** | **Answer** |
| --- | --- |
| **What went well?** | Database connection works, CRUD Create tested and confirmed in database and CRUD View. |
| **Biggest problem?** | Connecting with UI took longer than expected to create and test CRUD functions. |
| **What can I improve?** | Plan more time for research. |
| **Next steps:** | Build update and delete functions. |

Sprint 2 **(04.03.2025 – 05.03.2025)**

Goal**:**

* CRUD **Update**
* CRUD **Delete**

Tasks**:**

| **Task** | **Status** | **Reason** | **Problems** | **Solution** |
| --- | --- | --- | --- | --- |
| **CRUD Update Function** | Done | Update leads | Update a lead with Id | Research and using of breakpoints |
| **CRUD Delete Function** | Done | Delete leads | None |  |

#### Retrospective Sprint 2

|  |  |
| --- | --- |
| **What went well?** | Creating the CRUD function and successfully updating data in an update request. |
| **Biggest problem?** | The request never accepted the ID. |
| **What can I improve?** | Working more with breakpoints. |
| **Next steps:** | Implement the Lead to Customer function and all Customer functions. |

Sprint 3 **(06.03.2025 – 07.03.2025)**

Goal**:**

* CRUD **Update Customer**
* CRUD **Delete Customer**
* CRUD **View Customer**
* LeadtoCustomer

Tasks**:**

| **Task** | **Status** | **Reason** | **Problems** | **Solution** |
| --- | --- | --- | --- | --- |
| **CRUD Update Customer Function** | Done | Update customers | None |  |
| **CRUD Delete Customer Function** | Done | Delete customers | None |  |
| **CRUD View Customer Function** | Done | View Customers | None |  |
| LeadtoCustomer | Done | To transfers lead to customers | How to transfer the leads to the customer database | With researching on the internet and using breakpoints to see what the code is doing |

#### Retrospective Sprint 3

|  |  |
| --- | --- |
| **What went well?** | Creating the other CRUD functions for Customers |
| **Biggest problem?** | Transferring the lead to the customer database |
| **What can I improve?** | More research on the internet |
| **Next steps:** | Add JWT Token and create presentation |

Sprint 4 **(08.03.2025)**

Goal**:**

* Add JWT Token
* Create Roles
* Presentation
* LeadtoCustomer

Tasks**:**

| **Task** | **Status** | **Reason** | **Problems** | **Solution** |
| --- | --- | --- | --- | --- |
| **CRUD JWT Authorization** | Done | That not every one can use customer function | None |  |
| **Presentation** | In work | Present my work i have done | None |  |

#### Retrospective Sprint 4

|  |  |
| --- | --- |
| **What went well?** | Creating the different Roles for using the customers function |
| **Biggest problem?** | Creating the JWT Token an dhow use it in Swagger UI |
| **What can I improve?** | Try more to research what the Problem is the Code to solve the problem |
| **Next steps:** | Finish presentation and documentation |

Sprint 5 **(10.03.2025)**

Goal**:**

* Finish Documentation
* Finish Presentation

Tasks**:**

| **Task** | **Status** | **Reason** | **Problems** | **Solution** |
| --- | --- | --- | --- | --- |
| **Documentation** | Done | That not every one can use customer function | None |  |
| **Presentation** | Done | Present my work I have done | None |  |

#### Retrospective Sprint 5

|  |  |
| --- | --- |
| **What went well?** | Designing the presentation |
| **Biggest problem?** | I have everything than for the documentation |
| **What can I improve?** | Structuring and planning for the Documentation |
| **Next steps:** | None |

## Test

### Test Cases for Login Endpoint:

you can test everything through postman**Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

KI-generierte Inhalte können fehlerhaft sein.**

**Here you have to fill in when you use Put or post**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case** | **Description** | **Input Data** | **Expected Result** | **Actual Result** | **Status** |
| **POST /api/v1/login** | Test if login works with valid credentials. | {"username": "admin", "password": "adminpass"} | 200 OK, {"status": "success", "message": "Login successful", "token": "jwt\_token\_here"} | 200 OK, {"status": "success", "message": "Login successful", "token": "jwt\_token\_here"} | Pass |
| **POST /api/v1/login** | Test if login works with invalid username. | {"username": "wrong\_user", "password": "adminpass"} | 401 Unauthorized, {"status": "error", "message": "Invalid username or password"} | 401 Unauthorized, {"status": "error", "message": "Invalid username or password"}, | Pass |
| **POST /api/v1/login** | Test if login works with invalid password. | {"username": "admin", "password": "wrong\_password"} | 401 Unauthorized, {"status": "error", "message": "Invalid username or password"} | 401 Unauthorized, {"status": "error", "message": "Invalid username or password"} | Pass |
| **POST /api/v1/login** | Test if login works without credentials. | {"username": "", "password": ""} | 400 Bad Request, {"status": "error", "message": "Username and password are required"} | 400 Bad Request, {"status": "error", "message": "Username and password are required"} | Pass |

### Test Cases for Lead Endpoint:

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

To show where you need to place the Token from the different logins

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | **Description** | **Input Data** | **Expected Result** | **Actual Result** | **Status** | **Used Role** |
| **POST /api/v1/lead** | Test if a new lead can be created. | {"id": 0,"name": "Max Meier",  "gender": "Male",  "address": "gasweg 3",  "leadSource": "website"  } | 201 Created, {"id": 0,"name": "Max Meier",  "gender":"Male",  "address": "gasweg 3","leadSource": "website"  } | 201 Created, {"id": 0,"name": "Max Meier",  "gender": "Male",  "address": "gasweg 3",  "leadSource": "website"  } | Pass | Admin |
| **GET /api/v1/lead** | Test if all leads can be retrieved. | No request data | 200 OK, [{"id": 0,"name": "Max Meier",  "gender": "Male",  "address": "gasweg 3",  "leadSource": "website"  }] | 200 OK, [{"id": 0,"name": "Max Meier",  "gender": "Male",  "address": "gasweg 3",  "leadSource": "website"  }] | Pass | Viewer |
| **PUT /api/v1/lead/1** | Test if a lead can be updated correctly. | {id} =1  {{"id": 0,  "name": "Max Hans",    "gender": "string",    "address": "string",    "leadSource": "string"  } | 200 OK, [{"id": 0,"name": "Max Hans",  "gender": "Male",  "address": "gasweg 3",  "leadSource": "website"  }] | 200 OK, [{"id": 0,"name": "Max Hans",  "gender": "Male",  "address": "gasweg 3",  "leadSource": "website"  }] | Pass | Editor |
| **DELETE /api/v1/lead/1** | Test if a lead can be deleted. | No request data | 204 No Content | 204 No Content | Pass | Admin |

### Test Cases for Customer Endpoint:

**Crate a Lead again**

{"id": 0,"name": "Jenny Meier",

"gender": "Female",

"address": "gasweg 3",

"leadSource": "website"

}

**Will have the id 2**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | **Description** | **Input Data** | **Expected Result** | **Actual Result** | **Status** | **Used Role** |
| **POST /api/v1/customer/2** | Test if a lead can be converted to a customer. |  | 201 Created, {"id": 0,"name": "Jenny Meier",  "gender": "Female",  "address": "gasweg 3",  "leadSource": "website"  } | 201 Created, {"id": 0,"name": "Jenny Meier",  "gender": "Female",  "address": "gasweg 3",  "leadSource": "website"  } | Pass | Admin |
| **PUT /api/v1/customer/2** | Test if a customer can be updated correctly. | {{"id": 0,  "name": "string",    "gender": "string",    "address": "Lochweg 4",    "leadSource": "string"  } | 200 OK, {"id": 0,"name": "Jenny Meier",  "gender": "Female",  "address": "Lochweg 4",  "leadSource": "website"  } | 200 OK, {"id": 0,"name": "Jenny Meier",  "gender": "Female",  "address": "Lochweg 4",  "leadSource": "website"  } | Pass | Editor |
| **DELETE /api/v1/customer/2** | Test if a customer can be deleted. | No request data | 204 No Content | 204 No Content | Pass | Admin |
| **GET /api/v1/Customer** | Test if all customers can be retrieved. | No request data | 200 OK, [{"id": 0,"name": "Jenny Meier",  "gender": "Female",  "address": "gasweg 3",  "leadSource": "website"  }] | 200 OK, [{"id": 0,"name": "Jenny Meier",  "gender": "Female",  "address": "gasweg 3",  "leadSource": "website"  }] | Pass | Viewer |

## Decisions

| **Date** | **Topic** | **Decision** | **Reason** | **Alternative** |
| --- | --- | --- | --- | --- |
| 02.03.2025 | Database | MySQL | Fast & flexible | PostgreSQL |
| 03.03.2025 | Authentication | JWT | Simple & secure | Session Tokens |
| 08.03.2025 | Roles | Viewer, Editor, Admin | Better use for JWT | - |

## Project Summary

The Lead Management System (LMS) allows users to perform CRUD operations (Create, Read, Update, Delete) on leads and customers. Access to these functions is secured using JWT tokens, which are assigned based on user roles (e.g., admin, editor, viewer).

While the system currently operates without a dedicated user interface (UI), the next step for future development would be to build a UI using JavaScript, allowing for a more interactive experience.

Overall, the system is fully functional and meets the requirements for managing leads and customers securely.