CRM API Documentation

Overview

This API provides access to the CRM (Customer Relationship Management) system, allowing users to perform operations such as customer management, billing, product sales, and order management. The API supports essential functions like user login, customer search, adding new customers, and updating or deleting existing customers.

Getting Started

Authentication

Authorization: All API requests require Bearer Token authentication. The token is obtained after login and must be included in every request.

```
Authorization: Bearer <token>
```

Base URL:

http://localhost:8090/api

Endpoints

1. Login

Method: `POST`

Endpoint: `/auth/login`

Description: Used for logging in a user and returns a token.

Name	Type	In	Description
email	string	body	**Required** - Specifies the user's email address.

password	string	body	**Required** - Specifies the user's password.
Authorization	string	header	**Required** - Specifies the bearer token of the API client.

Request Body (JSON):

```
{
  "email": "string",
  "password": "string"
}
```

Response Format:

Success (`200 OK`):

```
{
  "token": "string",
  "success" : true
}
```

Example Error (`401 Unauthorized`):

```
{
  "error": "Invalid username or password"
}
```

Code Examples:

JavaScript (Fetch API):

```
const url = 'http://localhost:8090/api/auth/login';
const data = {
    username: "jane.doe@example.com",
    password: "password123"
};

fetch(url, {
```

```
    method: 'POST',
    headers: {
        'Content-Type': 'application/json'
    },
    body: JSON.stringify(data)
})
.then(response => response.json())
.then(result => console.log(result))
.catch(error => console.error('Error:', error));
```

C# (.NET HttpClient):

```
using System;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;

class Program
{
    static async Task Main(string[] args)
    {
        var client = new HttpClient();
        var url = "http://localhost:8090/api/auth/login";
        var data = new
        {
            username = "jane.doe@example.com",
            password = "password123"
        };

        var content = new StringContent(
            Newtonsoft.Json.JsonConvert.SerializeObject(data)
            Encoding.UTF8,
            "application/json"
        );

        var response = await client.PostAsync(url, content);
        var responseString = await response.Content.ReadAsStr
```

```
        Console.WriteLine(responseString);
    }
}
```

Java (HttpURLConnection):

```
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;

public class Main {
    public static void main(String[] args) {
        try {
            String url = "http://localhost:8090/api/auth/logi
            URL obj = new URL(url);
            HttpURLConnection con = (HttpURLConnection) obj.o
            con.setRequestMethod("POST");
            con.setRequestProperty("Content-Type", "applicati

            String jsonInputString = "{\"username\": \"jane.d

            con.setDoOutput(true);
            try (OutputStream os = con.getOutputStream()) {
                byte[] input = jsonInputString.getBytes(Stand
                os.write(input, 0, input.length);
            }

            int responseCode = con.getResponseCode();
            System.out.println("Response Code : " + responseC

            // Process the response...
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

PHP (cURL):

```
<?php
$url = "http://localhost:8090/api/auth/login";
$data = array(
    "username" => "jane.doe@example.com",
    "password" => "password123"
);

$options = array(
    'http' => array(
        'header'  => "Content-Type: application/json\r\n",
        'method'  => 'POST',
        'content' => json_encode($data),
    ),
);

$context  = stream_context_create($options);
$response = file_get_contents($url, false, $context);
if ($response === FALSE) {
    // Handle error
}

var_dump($response);
?>
```

2. Password Reset

Method: `POST`

Endpoint: `/password-reset`

Description: Used to reset a user's password.

Name	Type	In	Description
email	string	body	**Required** - Specifies the user's email address to receive the reset link.
Authorization	string	header	**Required** - Specifies the bearer token of the API client.

Request Body (JSON):

```
{
  "email": "jane.doe@example.com"
}
```

Response Format:

Success (`200 OK`):

```
{
  "message": "Password reset link sent"
}
```

Error (`404 Not Found`):

```
{
  "error": "Email not found"
}
```

Code Examples:

JavaScript (Fetch API):

```
const url = 'https://api.crm-example.com/v1/password-reset';
const data = {
    email: "jane.doe@example.com"
};

fetch(url, {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json'
    },
    body: JSON.stringify(data)
})
.then(response => response.json())
.then(result => console.log(result))
.catch(error => console.error('Error:', error));
```

C# (.NET HttpClient):

```
using System;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;

class Program
{
    static async Task Main(string[] args)
    {
        var client = new HttpClient();
        var url = "https://api.crm-example.com/v1/password-re
        var data = new { email = "jane.doe@example.com" };

        var content = new StringContent(
            Newtonsoft.Json.JsonConvert.SerializeObject(data)
            Encoding.UTF8,
            "application/json"
        );

        var response = await client.PostAsync(url, content);
        var responseString = await response.Content.ReadAsStr

        Console.WriteLine(responseString);
    }
}
```

Java (HttpURLConnection):

```
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;

public class Main {
    public static void main(String[] args) {
        try {
```

```
            String url = "https://api.crm-example.com/v1/pass
            URL obj = new URL(url);
            HttpURLConnection con = (HttpURLConnection) obj.o
            con.setRequestMethod("POST");
            con.setRequestProperty("Content-Type", "applicati

            String jsonInputString = "{\"email\": \"jane.doe@

            con.setDoOutput(true);
            try (OutputStream os = con.getOutputStream()) {
                byte[] input = jsonInputString.getBytes(Stand
                os.write(input, 0, input.length);
            }

            int responseCode = con.getResponseCode();
            System.out.println("Response Code : " + responseC

            // Process the response...
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

PHP (cURL):

```
<?php
$url = "https://api.crm-example.com/v1/password-reset";
$data = array(
    "email" => "jane.doe@example.com"
);

$options = array(
    'http' => array(
        'header'  => "Content-Type: application/json\r\n",
        'method'  => 'POST',
        'content' => json_encode($data),
    ),
```

```
);

$context  = stream_context_create($options);
$response = file_get_contents($url, false, $context);
if ($response === FALSE) {
    // Handle error
}

var_dump($response);
?>
```

3. Customer Transactions

3.1. Individual Customer Transactions

Controller: `Individual Customer Controller`

Bringing Individual Customers

Method: `GET`

Endpoint: `/search`

Description: Retrieves a list of all individual customers with optional filters for searching by customer details.

Name	Type	In	Description
firstName	string	query	Optional - Filter by customer's first name.
lastName	string	query	Optional - Filter by customer's last name.
nationalityId	string	query	Optional - Filter by customer's nationality ID.
accountNumber	string	query	Optional - Filter by customer's account number.
mobilePhone	string	query	Optional - Filter by customer's mobile phone number.
Authorization	string	header	**Required** - Specifies the bearer token of the API client.

Example Request URL:

```
GET /search?firstName=Nuray&lastName=Altuğ
```

Example Request:

The request can include optional query parameters, but no request body is required.

Example Response:

`200 OK`


```
{
  "totalElements": 0,
  "totalPages": 0,
  "first": true,
  "last": true,
  "size": 0,
  "content": [
    {
      "id": "string",
      "firstName": "string",
      "middleName": "string",
      "lastName": "string",
      "role": "string",
      "nationalityId": "string",
      "accountNumber": "string",
      "mobilePhone": "string"
    }
  ],
  "number": 0,
  "sort": [
    {
      "direction": "string",
      "nullHandling": "string",
      "ascending": true,
      "property": "string",
      "ignoreCase": true
    }
```

```
  ],
  "numberOfElements": 0,
  "pageable": {
    "offset": 0,
    "sort": [
      {
        "direction": "string",
        "nullHandling": "string",
        "ascending": true,
        "property": "string",
        "ignoreCase": true
      }
    ],
    "paged": true,
    "pageNumber": 0,
    "pageSize": 0,
    "unpaged": true
  },
  "empty": true
}
```


Possible Error Examples:

`400 Bad Request`


```
{
  "error": "Invalid filter parameters"
}
```


`404 Not Found`


```
{
  "message": "No customers found"
}
```


3.2. Create New Individual Customer

Method: `POST`

Endpoint: `/customers/individual`

Description: Creates a new individual customer in the system.

Name	Type	In	Description
firstName	string	body	**Required** - Customer's first name.
middleName	string	body	Optional - Customer's middle name.
lastName	string	body	**Required** - Customer's last name.
birthDate	date	body	**Required** - Customer's birth date (YYYY-MM-DD).
gender	string	body	**Required** - Customer's gender (M/F).
fatherName	string	body	Optional - Customer's father's name.
motherName	string	body	Optional - Customer's mother's name.
turkishNationality	boolean	body	**Required** - Indicates if the customer is of Turkish nationality.
nationalityId	string	body	Conditional - Required if `turkishNationality` is true. Customer's nationality ID.
Authorization	string	header	**Required** - Specifies the bearer token of the API client.

Request Body (JSON):

The request body should contain all necessary details about the customer, such as name, national ID, birthdate, address, and contact information.

Example Request:

```
{
  "firstName": "string",
  "middleName": "string",
  "lastName": "string",
  "birthDate": "2024-10-25T10:55:30.203Z",
  "gender": "char"
  "fatherName": "string"
```

```
  "motherName": "string"
  "nationality" : true
  "nationalityId": "stringstrin",
}
```

Response:

`200 OK`:

```
{
    "id": 0
    "firstName": "string"
    "middleName": "string"
    "lastName": "string"
}
```

Example Response:

`201 Created`:

```
{
  "message": "Customer created successfully",
  "customerId": 1002,
  "status": "ACTIVE",
  "createdAt": "2024-10-24T08:30:00Z"
}
```

`400 Bad Request` (Example for missing or invalid fields):

```
{
  "error": "Invalid input data",
  "validationErrors": {
    "nationalId": "National ID must be 11 digits",
    "email": "Invalid email format"
  }
}
```

`500 Internal Server Error`:

```
{
  "error": "Internal Server Error",
  "message": "An unexpected error occurred while processing y
}
```

Code Examples:

JavaScript (Fetch API):

```
const url = 'http://localhost:8090/api/customers/individual';

const data = {
    firstName: "Jane",
    lastName: "Doe",
    nationalId: "12345678901",
    birthDate: "1990-05-01",
    address: {
        street: "Main St",
        city: "Istanbul",
        zipCode: "34000"
    },
    contact: {
        email: "jane.doe@example.com",
        phone: "+905555555555"
    }
};

fetch(url, {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer your_token_here'
    },
    body: JSON.stringify(data)
})
.then(response => response.json())
```

```
.then(data => console.log('Success:', data))
.catch((error) => console.error('Error:', error));
```

Java (HttpURLConnection)

```
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;

public class CreateCustomer {
    public static void main(String[] args) {
        try {
            URL url = new URL("http://localhost:8090/api/cust
            HttpURLConnection conn = (HttpURLConnection) url.
            conn.setRequestMethod("POST");
            conn.setRequestProperty("Content-Type", "applicat
            conn.setRequestProperty("Authorization", "Bearer
            conn.setDoOutput(true);

            String jsonInputString = "{\"firstName\":\"Jane\"

            try (OutputStream os = conn.getOutputStream()) {
                byte[] input = jsonInputString.getBytes(Stand
                os.write(input, 0, input.length);
            }

            int responseCode = conn.getResponseCode();
            System.out.println("Response Code: " + responseCo

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

PHP (cURL)

```php
<?php

$curl = curl_init();

$data = [
    "firstName" => "Jane",
    "lastName" => "Doe",
    "nationalId" => "12345678901",
    "birthDate" => "1990-05-01",
    "address" => [
        "street" => "Main St",
        "city" => "Istanbul",
        "zipCode" => "34000"
    ],
    "contact" => [
        "email" => "jane.doe@example.com",
        "phone" => "+905555555555"
    ]
];

curl_setopt_array($curl, [
    CURLOPT_URL => "http://localhost:8090/api/customers/indiv
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_CUSTOMREQUEST => "POST",
    CURLOPT_HTTPHEADER => [
        "Content-Type: application/json",
        "Authorization: Bearer your_token_here"
    ],
    CURLOPT_POSTFIELDS => json_encode($data)
]);

$response = curl_exec($curl);

if (curl_errno($curl)) {
    echo 'Error:' . curl_error($curl);
} else {
    echo $response;
}
```

```
curl_close($curl);

?>
```

C# (.NET HttpClient)

```
using System;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;

class Program
{
    private static readonly HttpClient client = new HttpClien

    static async Task Main(string[] args)
    {
        var url = "http://localhost:8090/api/customers/indivi
        var requestBody = new
        {
            firstName = "Jane",
            lastName = "Doe",
            nationalId = "12345678901",
            birthDate = "1990-05-01",
            address = new
            {
                street = "Main St",
                city = "Istanbul",
                zipCode = "34000"
            },
            contact = new
            {
                email = "jane.doe@example.com",
                phone = "+905555555555"
            }
        };
```

```
        var json = Newtonsoft.Json.JsonConvert.SerializeObjec
        var content = new StringContent(json, Encoding.UTF8,

        client.DefaultRequestHeaders.Add("Authorization", "Be

        var response = await client.PostAsync(url, content);
        string responseString = await response.Content.ReadAs

        Console.WriteLine(responseString);
    }
}
```

4. Update Customer Address

Method: `PUT`

Endpoint: `/customers/addresses/{id}`

Description: Updates the address of an existing customer. The `{id}` in the endpoint is the unique identifier of the address being updated.

Name	Type	In	Description
id	integer	path	**Required** - Unique ID of the address to be updated.
customerId	integer	body	**Required** - Customer ID associated with the address.
neighbourhoodId	integer	body	**Required** - ID of the neighbourhood.
addressName	string	body	**Required** - Name/label of the address.
street	string	body	**Required** - Street name.
houseNumber	integer	body	**Required** - House or flat number.
zipCode	string	body	Optional - Zip code of the address.
Authorization	string	header	**Required** - Specifies the bearer token of the API client.

Path Parameter:

- `id` (integer): The unique ID of the address to be updated.

Example Request URL:

```
PUT /customers/addresses/123
```

Example Request:

Request Body (JSON):

```
{
     "customerId": 0,
     "neighbourhoodId": 0,
     "addressName": "string"
     "street": "string"
     "houseNumber": 0
}
```

Response Format:

Success (`200 OK`):

```
{
    "id": 0,
    "customerId": 0,
    "neighbourhoodId": 0,
    "addressName": "string"
    "street": "string"
    "houseNumber": 0
}
```

Error (`404 Not Found`):

```
{
  "error": "Address not found",
  "addressId": 123
}
```

400 Bad Request (Example for invalid address format)

```
{
  "error": "Invalid address format",
  "validationErrors": {
    "zipCode": "Zip Code must be a valid postal code"
  }
}
```

Code Examples:

JavaScript (Fetch API):

```
const addressId = 123; // Replace with the actual address ID
const url = `http://localhost:8090/api/customers/addresses/${

const data = {
    street: "Updated Tech St",
    city: "Ankara",
    zipCode: "06000"
};

fetch(url, {
    method: 'PUT',
    headers: {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer your_token_here'
    },
    body: JSON.stringify(data)
})
.then(response => response.json())
.then(data => console.log('Success:', data))
.catch((error) => console.error('Error:', error));
```

C# (.NET HttpClient):

```
using System;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;
```

```
class Program
{
    private static readonly HttpClient client = new HttpClien

    static async Task Main(string[] args)
    {
        int addressId = 123; // Replace with the actual addre
        var url = $"http://localhost:8090/api/customers/addre
        var requestBody = new
        {
            street = "Updated Tech St",
            city = "Ankara",
            zipCode = "06000"
        };

        var json = Newtonsoft.Json.JsonConvert.SerializeObjec
        var content = new StringContent(json, Encoding.UTF8,

        client.DefaultRequestHeaders.Add("Authorization", "Be

        var response = await client.PutAsync(url, content);
        string responseString = await response.Content.ReadAs

        Console.WriteLine(responseString);
    }
}
```

Java (HttpURLConnection):

```
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;

public class UpdateCustomerAddress {
    public static void main(String[] args) {
        try {
```

```
            int addressId = 123; // Replace with the actual a
            URL url = new URL("http://localhost:8090/api/cust
            HttpURLConnection conn = (HttpURLConnection) url.
            conn.setRequestMethod("PUT");
            conn.setRequestProperty("Content-Type", "applicat
            conn.setRequestProperty("Authorization", "Bearer
            conn.setDoOutput(true);

            String jsonInputString = "{\"street\":\"Updated T

            try (OutputStream os = conn.getOutputStream()) {
                byte[] input = jsonInputString.getBytes(Stand
                os.write(input, 0, input.length);
            }

            int responseCode = conn.getResponseCode();
            System.out.println("Response Code: " + responseCo

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

PHP (cURL):

```
<?php

$addressId = 123; // Replace with the actual address ID
$curl = curl_init();

$data = [
    "street" => "Updated Tech St",
    "city" => "Ankara",
    "zipCode" => "06000"
];

curl_setopt_array($curl, [
```

```
    CURLOPT_URL => "http://localhost:8090/api/customers/addre
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_CUSTOMREQUEST => "PUT",
    CURLOPT_HTTPHEADER => [
        "Content-Type: application/json",
        "Authorization: Bearer your_token_here"
    ],
    CURLOPT_POSTFIELDS => json_encode($data)
]);

$response = curl_exec($curl);

if (curl_errno($curl)) {
    echo 'Error:' . curl_error($curl);
} else {
    echo $response;
}

curl_close($curl);

?>
```

5. Product Operations

5.1. Retrieving Products

Method: `GET`

Endpoint: `/catalogs/products`

Description: Retrieves a list of all available products with optional filtering and pagination.

Example Request URL:

```
GET /catalogs/products?category=Electronics
```

Example Response:

Success (`200 OK`):

```
{
    "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "name": "string",
    "categoryId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "price": 0
}
```


`404 Not Found`


```
{
  "message": "No products found"
}
```


`401 Unauthorized`:

Example Response:


```
{
  "error": "Unauthorized",
  "message": "You must provide a valid authentication toke
n."
}
```


`403 Forbidden`:


```
{
  "error": "Forbidden",
  "message": "You do not have permission to view this resourc
}
```


JavaScript:

```
fetch('http://localhost:8090/api/catalogs/products?page=1&siz
    .then(response => {
        if (!response.ok) {
            throw new Error('Network response was not ok ' +
        }
        return response.json();
    })
```

```
    .then(data => {
        console.log('Response:', data);
    })
    .catch(error => {
        console.error('Error:', error);
    });
```

C#:

```
using System;
using System.Net.Http;
using System.Threading.Tasks;

class Program
{
    static async Task Main(string[] args)
    {
        using (HttpClient client = new HttpClient())
        {
            client.BaseAddress = new Uri("http://localhost:80
            HttpResponseMessage response = await client.GetAs

            if (response.IsSuccessStatusCode)
            {
                string responseData = await response.Content.
                Console.WriteLine("Response: " + responseData
            }
            else
            {
                Console.WriteLine("Error: " + response.Status
            }
        }
    }
}
```

Java:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

public class RetrieveProducts {
    public static void main(String[] args) {
        try {
            String url = "http://localhost:8090/api/catalogs/
            HttpURLConnection conn = (HttpURLConnection) new 
            conn.setRequestMethod("GET");

            if (conn.getResponseCode() == HttpURLConnection.H
                BufferedReader in = new BufferedReader(new In
                String inputLine;
                StringBuilder response = new StringBuilder();

                while ((inputLine = in.readLine()) != null) {
                    response.append(inputLine);
                }
                in.close();
                System.out.println("Response: " + response.to
            } else {
                System.out.println("Error: " + conn.getRespon
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

PHP:

```
<?php
$url = 'http://localhost:8090/api/catalogs/products?page=1&si

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $url);
```

```
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);

$response = curl_exec($ch);
$httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);

if ($httpCode == 200) {
    echo "Response: " . $response;
} else {
    echo "Error: " . $httpCode;
}

curl_close($ch);
?>
```

HTTP Codes:

HTTP Code	Description
200 OK	The request was successful.
201 Created	A new resource was successfully created.
400 Bad Request	Invalid parameters or missing data.
401 Unauthorized	Authentication failed.
403 Forbidden	Access is denied.
404 Not Found	Resource not found.
409 Conflict	Conflicting resource (e.g., duplicate ID).
500 Internal Server Error	Server encountered an error.

Rate Limits and Usage

- **Rate Limiting**: API request limit is 1000 requests per hour. Exceeding this limit will return a `429 Too Many Requests` error.
- **Request Size**: Maximum request size is limited to 10 MB.
- **Timeout**: Maximum processing time for a request is 30 seconds.

Changelog

Version	Date	Change
1.0	2024-10-01	Initial version of the API.

Frequently Asked Questions (FAQ)

1. What do I need to do before using the API?

- Before using the API, you need to log in and obtain an authorization token. This token must be included in the `Authorization` header for all requests.

2. What happens if I hit the rate limit?

- If you exceed the rate limit, you will receive a `429 Too Many Requests` error. Wait for a while before retrying your request.

3. How can I stay updated with API changes?

- Check the changelog section for any updates, or contact the support team for more information.

Glossary

- **Bearer Token**: The token used for API access authentication.
- **Endpoint**: The URL paths used for performing API operations.
- **Rate Limit**: The maximum number of requests allowed in a specific time frame.
- **HTTP Method**: The method used to send or retrieve data (e.g., GET, POST, PUT, DELETE).