

Name: Andi Dwi Saputro

Date: April 9<sup>th</sup>, 2020

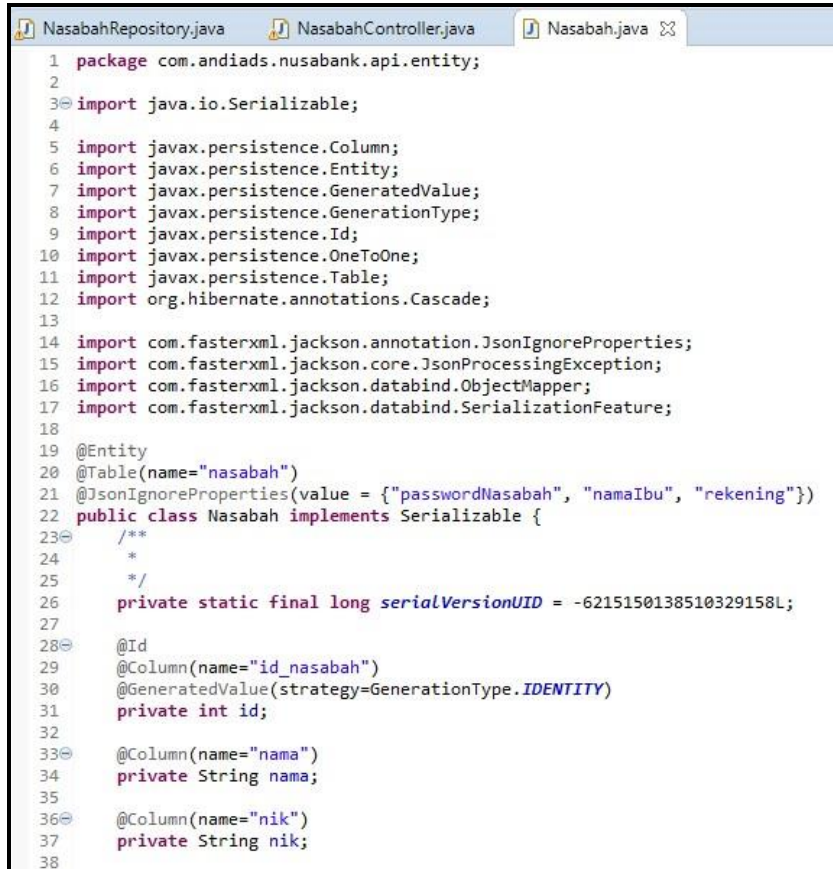
### Main Cases: 1. Create RESTful API

#### 2. Paginated Results

#### 3. Search by Resources

### 1. Create RESTful API

#### a. POJO Nasabah class



```
1 package com.andiads.nusabank.api.entity;
2
3 import java.io.Serializable;
4
5 import javax.persistence.Column;
6 import javax.persistence.Entity;
7 import javax.persistence.GeneratedValue;
8 import javax.persistence.GenerationType;
9 import javax.persistence.Id;
10 import javax.persistence.OneToOne;
11 import javax.persistence.Table;
12 import org.hibernate.annotations.Cascade;
13
14 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
15 import com.fasterxml.jackson.core.JsonProcessingException;
16 import com.fasterxml.jackson.databind.ObjectMapper;
17 import com.fasterxml.jackson.databind.SerializationFeature;
18
19 @Entity
20 @Table(name="nasabah")
21 @JsonIgnoreProperties(value = {"passwordNasabah", "namaIbu", "rekening"})
22 public class Nasabah implements Serializable {
23     /**
24      *
25      */
26     private static final long serialVersionUID = -6215150138510329158L;
27
28     @Id
29     @Column(name="id_nasabah")
30     @GeneratedValue(strategy=GenerationType.IDENTITY)
31     private int id;
32
33     @Column(name="nama")
34     private String nama;
35
36     @Column(name="nik")
37     private String nik;
38 }
```

POJO Class nasabah menggunakan annotation `JsonIgnoreProperties` untuk meng-ignore atau tidak menyisipkan beberapa fields diantaranya password, namaIbu, dan rekening nasabah pada result JSON. For the rest, terdapat field2 lainnya dan getters setters.

```

227 @Override
228 public String toString() {
229     ObjectMapper mapper = new ObjectMapper();
230
231     String jsonString = "";
232     try {
233         mapper.enable(SerializationFeature.INDENT_OUTPUT);
234         jsonString = mapper.writeValueAsString(this);
235     } catch (JsonProcessingException e) {
236         e.printStackTrace();
237     }
238
239     return jsonString;
240 }
241
242
243 }

```

Snippets diatas dimaksudkan untuk menampilkan result JSON dengan pretty print.

## b. Application.properties

```

application.properties
1 ## Spring DATASOURCE CONFIG
2 spring.datasource.url=jdbc:mysql://localhost:3306/db_nusa?useUnicode=true&useJDBCCompliant
3 spring.datasource.username=andi
4 spring.datasource.password=wFU75Pc87a9n?mS
5
6 ## Hibernate PROPERTIES CONFIG
7 spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLInnoDBDialect
8
9 ## Hibernate ddl-auto (create, create-drop, validate, update)
10 spring.jpa.hibernate.ddl-auto=update
11
12 ## Pretty-print JSON responses
13 spring.jackson.serialization.indent-output=true
14

```

Application.properties file untuk mensetup koneksi database dan hibernate serta JSON pretty print.

## c. Nasabah Repository

```

NasabahRepository.java
1 package com.andiads.nusabank.api.Repository;
2
3 import java.util.List;
4
5 import org.springframework.data.domain.Page;
6 import org.springframework.data.domain.Pageable;
7 import org.springframework.data.jpa.repository.JpaRepository;
8 import org.springframework.data.jpa.repository.Query;
9 import org.springframework.data.repository.PagingAndSortingRepository;
10 import org.springframework.data.repository.query.Param;
11 import org.springframework.data.rest.core.annotation.RestResource;
12 import org.springframework.stereotype.Repository;
13
14 import com.andiads.nusabank.api.entity.Nasabah;
15
16 @Repository
17 public interface NasabahRepository extends PagingAndSortingRepository<Nasabah, Integer>,
18     JpaRepository<Nasabah, Integer>{
19
20     @RestResource(path = "namaContains")
21     public Page<Nasabah> findByNamaContaining(@Param("nama") String nama, Pageable p);
22
23     @RestResource(path = "nikContains")
24     public Page<Nasabah> findByNikContaining(@Param("nik") String nik, Pageable p);
25 }
26

```

Pada nasabah repository digunakan untuk melakukan proses parsing data menjadi restful. Disini terdapat custom resource untuk fetching data berdasarkan nama dan nik. Karena interface ini mengimplement methods dari JpaRepository dan PagingAndSortRepository jadi tidak perlu membuat method seperti findAll, findById, dsb karena sudah di generate oleh spring.

#### d. Nasabah Controller

```
NasabahRepository.java  NasabahController.java  ⌵
1 package com.andiads.nusabank.api.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.data.domain.Page;
5 import org.springframework.data.domain.Pageable;
6 import org.springframework.data.rest.core.annotation.RepositoryRestResource;
7 import org.springframework.data.rest.core.annotation.RestResource;
8 import org.springframework.http.ResponseEntity;
9 import org.springframework.ui.Model;
10 import org.springframework.web.bind.annotation.*;
11 import javax.validation.Valid;
12 import java.util.List;
13 import java.util.stream.Collectors;
14
15 import com.andiads.nusabank.api.Repository.NasabahRepository;
16 import com.andiads.nusabank.api.entity.Nasabah;
17 import com.andiads.nusabank.api.exception.*;
18
19 @RestController
20 @RequestMapping("/nusabank/api")
21 public class NasabahController {
```

Pada gambar diatas merupakan import pada controller nasabah. Terdapat requestmapping ke path /nusabank/api untuk menampilkan fetch semua data.

```
23 private static final String LinkFormat =
24     "<a href='http://localhost:8080/nusabank/api/nasabah/page/?page=%s&size=%s'>%s</a><br/>";
25
26 @Autowired
27 private NasabahRepository nasabahRepository;
28
29 public NasabahController(NasabahRepository nr) {
30     this.nasabahRepository=nr;
31 }
```

Pada gambar diatas, dilakukan instansiate nasabah repository object dan String LinkFormat untuk hasil fetch pagination.

```
33 // get all of nasabah lists
34 @GetMapping("/nasabah")
35 public List<Nasabah> getAllNasabah(){
36     return this.nasabahRepository.findAll();
37 }
```

Pada gambar diatas merupakan method dan mapping untuk menampilkan semua data nasabah tanpa pagination.

```

39 // get all paged nasabah lists
40 @RequestMapping("/nasabah/page")
41 public String showPagedNasabahList(Pageable pageable) {
42     Page<Nasabah> pgNasabah = this.nasabahRepository.findAll(pageable);
43     List<Nasabah> listNasabah = pgNasabah.getContent();
44     String response =
45         listNasabah.stream()
46             .map(Nasabah::toString)
47             .collect(Collectors.joining("<br/>"));
48     response += "<br/><br/>";
49     if (pgNasabah.hasPrevious()) {
50         response += String.format(LinkFormat,
51             pgNasabah.getNumber()-1,
52             pgNasabah.getSize(),
53             "Previous Page");
54     }
55     if (pgNasabah.hasNext()) {
56         response += String.format(LinkFormat,
57             pgNasabah.getNumber()+1,
58             pgNasabah.getSize(),
59             "Next Page");
60     }
61     return response;
62 }

```

Pada gambar diatas, digunakan untuk menampilkan list data nasabah dengan pagination dengan memanfaatkan PagingAndSortRepository. Result nya juga akan menampilkan tombol next dan prev halaman.

```

64 // get nasabah by ID
65 @GetMapping("/nasabah/{id}")
66 public Nasabah getNasabahById(@PathVariable(value = "id") int id) {
67     return this.nasabahRepository.findById(id)
68         .orElseThrow(() -> new NasabahNotFoundException("Nasabah", "id", id));
69 }
70
71 // search nasabah by nama
72 @GetMapping("/nasabah/search/name/{namaContains}")
73 public Page<Nasabah> getNasabahByNamaContaining(@PathVariable(value = "namaContains") String nama, Pageable pageable) {
74     return this.nasabahRepository.findByNamaContaining(nama, pageable);
75 }
76
77 // search nasabah by nik
78 @GetMapping("/nasabah/search/nik/{nikContains}")
79 public Page<Nasabah> getNasabahByNikContaining(@PathVariable(value = "nikContains") String nik, Pageable pageable) {
80     return this.nasabahRepository.findByNikContaining(nik, pageable);
81 }

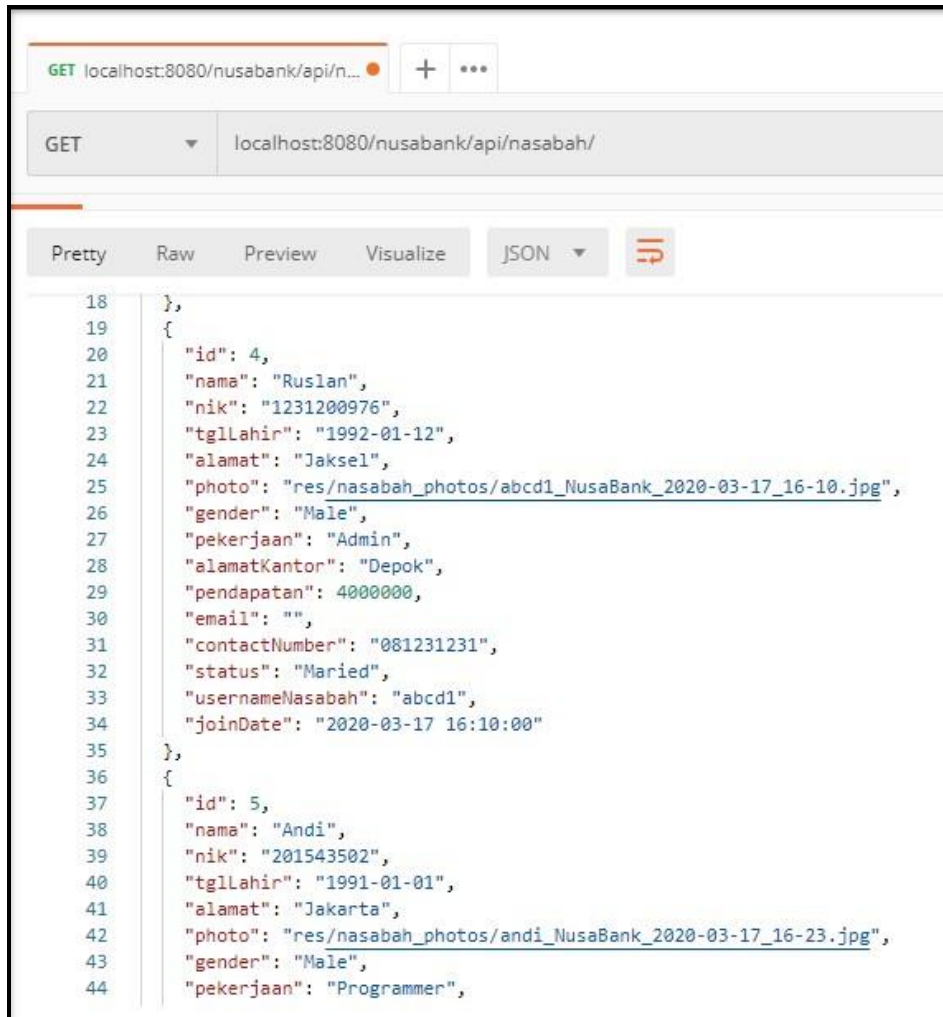
```

Ketiga method diatas digunakan untuk mengambil data berdasarkan ID, Nama, dan NIK nasabah. Dengan method HTTP GET.



e. Result (Fetch All and findById)

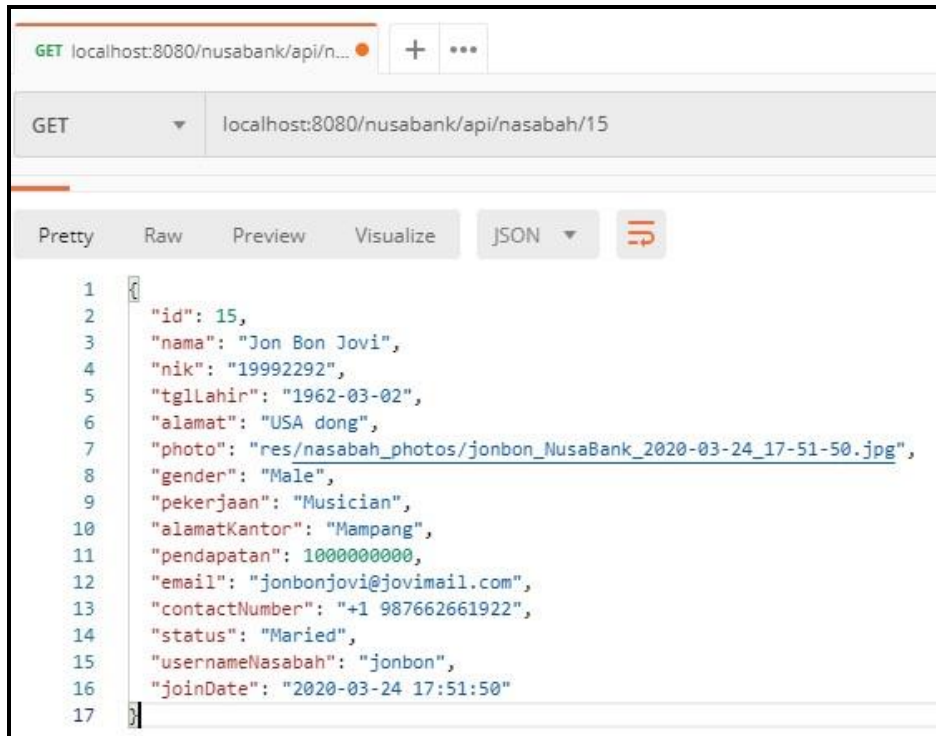
Berikut ada result dari fetchall normal dan findByID



The screenshot shows a REST client interface with a GET request to `localhost:8080/nusabank/api/nasabah/`. The response is displayed in JSON format, showing a list of two users. The first user has an ID of 4, name Ruslan, and is an Admin. The second user has an ID of 5, name Andi, and is a Programmer.

```
18 },
19 {
20   "id": 4,
21   "nama": "Ruslan",
22   "nik": "1231200976",
23   "tgllahir": "1992-01-12",
24   "alamat": "Jaksel",
25   "photo": "res/nasabah_photos/abcd1_NusaBank_2020-03-17_16-10.jpg",
26   "gender": "Male",
27   "pekerjaan": "Admin",
28   "alamatKantor": "Depok",
29   "pendapatan": 4000000,
30   "email": "",
31   "contactNumber": "081231231",
32   "status": "Maried",
33   "usernameNasabah": "abcd1",
34   "joinDate": "2020-03-17 16:10:00"
35 },
36 {
37   "id": 5,
38   "nama": "Andi",
39   "nik": "201543502",
40   "tgllahir": "1991-01-01",
41   "alamat": "Jakarta",
42   "photo": "res/nasabah_photos/andi_NusaBank_2020-03-17_16-23.jpg",
43   "gender": "Male",
44   "pekerjaan": "Programmer",
```

(Fetch All)

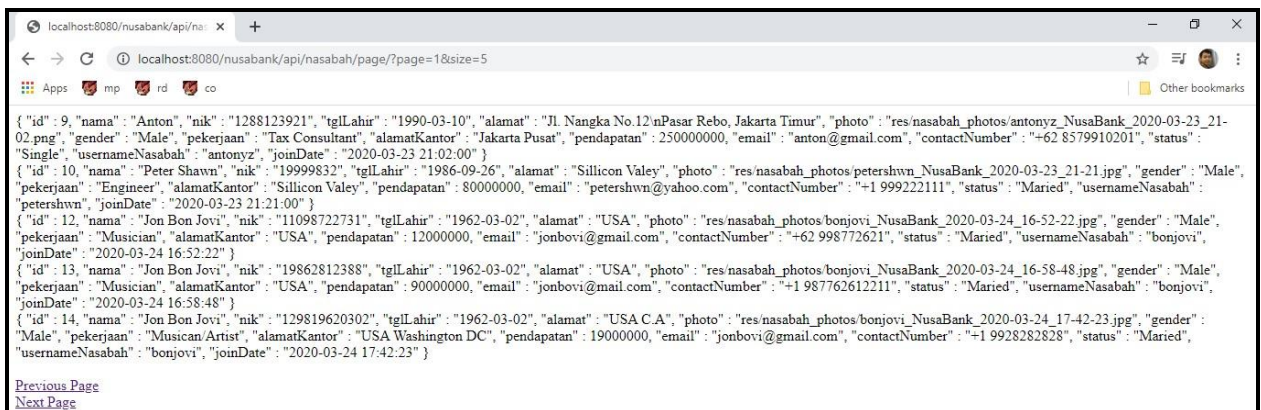


(Fetch ByID 15)

## 2. Paginated Results



(Page 0, Size 5)



(Page 1, Size 5)

```
localhost:8080/nusabank/api/na: x +
localhost:8080/nusabank/api/nasabah/page/?page=2&size=5
Apps mp rd co Other bookmarks

{"id": 15, "nama": "Jon Bon Jovi", "nik": "19992292", "tglLahir": "1962-03-02", "alamat": "USA dong", "photo": "res/nasabah_photos/jonbon_NusaBank_2020-03-24_17-51-50.jpg", "gender": "Male", "pekerjaan": "Musician", "alamatKantor": "Mampang", "pendapatan": 100000000, "email": "jonbonjovi@jovimail.com", "contactNumber": "+1 987662661922", "status": "Maried", "usernameNasabah": "jonbon", "joinDate": "2020-03-24 17:51:50"}
{"id": 16, "nama": "Abon Jovi", "nik": "19292839212", "tglLahir": "1962-03-02", "alamat": "New York, USA", "photo": "res/nasabah_photos/bonjovi11_NusaBank_2020-03-24_17-54-44.jpg", "gender": "Male", "pekerjaan": "Musician", "alamatKantor": "Pejaten Barat", "pendapatan": 90000000, "email": "jonbovi@mail.com", "contactNumber": "+1 299838277122", "status": "Maried", "usernameNasabah": "bonjovi11", "joinDate": "2020-03-24 17:54:44"}
{"id": 17, "nama": "SpringBoot", "nik": "03040304", "tglLahir": null, "alamat": "Osaka, Japan", "photo": null, "gender": null, "pekerjaan": null, "alamatKantor": null, "pendapatan": 0, "email": null, "contactNumber": null, "status": null, "usernameNasabah": null, "joinDate": null}
{"id": 25, "nama": "Monacar", "nik": "190804212", "tglLahir": "1976-04-06", "alamat": "Monacar Street", "photo": "", "gender": "Female", "pekerjaan": "Driver", "alamatKantor": null, "pendapatan": 90000000, "email": "monica@monacar.com", "contactNumber": "", "status": null, "usernameNasabah": "", "joinDate": null}

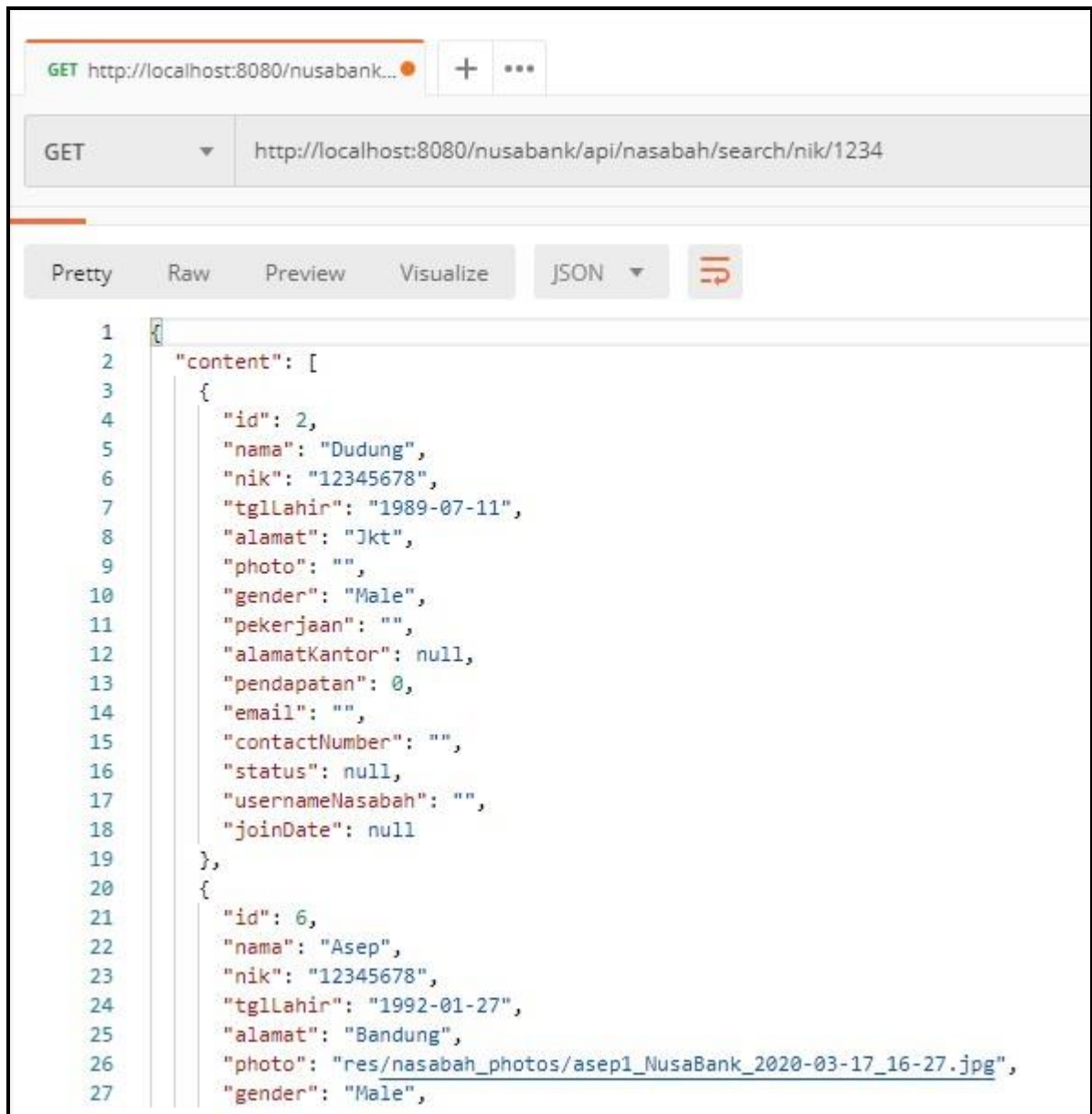
Previous Page
```

(Page 2, Size 5)

### 3. Search By Resources (Name, NIK)

```
GET http://localhost:8080/nusabank... + ...
GET http://localhost:8080/nusabank/api/nasabah/search/name/bon
Pretty Raw Preview Visualize JSON
1 [{"content": [
2   {
3     "id": 12,
4     "nama": "Jon Bon Jovi",
5     "nik": "11098722731",
6     "tglLahir": "1962-03-02",
7     "alamat": "USA",
8     "photo": "res/nasabah_photos/bonjovi_NusaBank_2020-03-24_16-52-22.jpg",
9     "gender": "Male",
10    "pekerjaan": "Musician",
11    "alamatKantor": "USA",
12    "pendapatan": 12000000,
13    "email": "jonbovi@gmail.com",
14    "contactNumber": "+62 998772621",
15    "status": "Maried",
16    "usernameNasabah": "bonjovi",
17    "joinDate": "2020-03-24 16:52:22"
18  },
19  {
20    "id": 13,
21    "nama": "Jon Bon Jovi",
22    "nik": "19862812388",
23    "tglLahir": "1962-03-02",
24    "alamat": "USA",
25    "photo": "res/nasabah_photos/bonjovi_NusaBank_2020-03-24_16-58-48.jpg",
26    "gender": "Male",
27  }
```

(Search By Name)



(Search By NIK)