3. Section

RESTful Service 기능 확장

- Validation
- Internationalization
- XML format으로 반환하기
- Filtering
- Version 관리

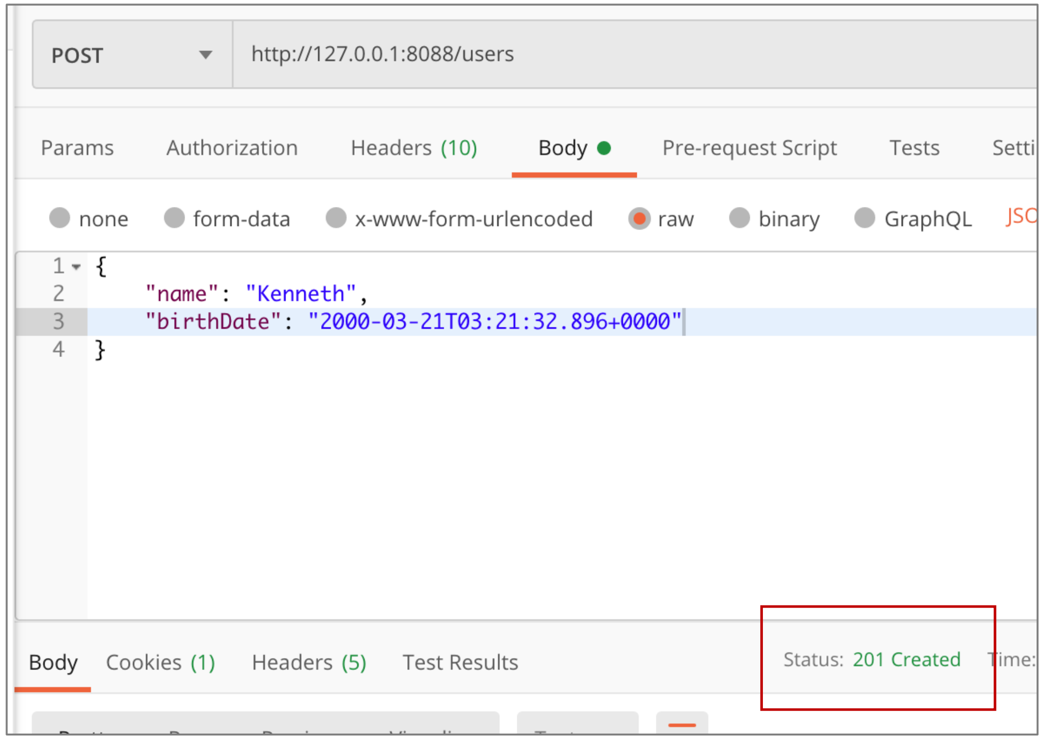
step14- Implementing Validations for RESTful Services

```
@PostMapping("/users")
public ResponseEntity<User> createUser(@Valid @RequestBody User
user) {
   User savedUser = service.save(user);
```

```
public class User {
   private Integer id;

   @Size(min=2)
   private String name;

   @Past
   private Date birthDate;
}
```



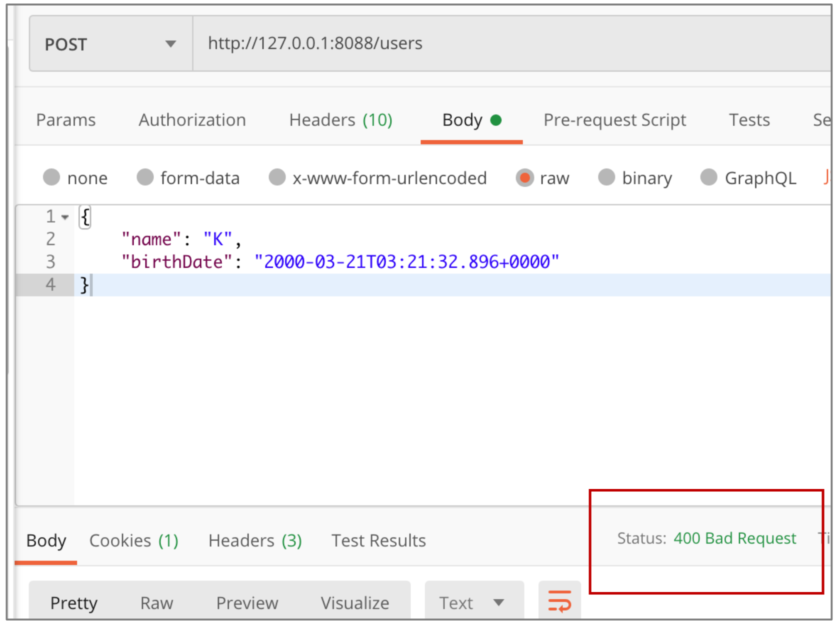
POST http://127.0.0.1:8088/users

Params Authorization Headers (10) Body ● Pre-request Script Tests Setti

none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSO

```
{
    "name": "Kenneth",
    "birthDate": "2000-03-21T03:21:32.896+0000"
}
```

Body Cookies (1) Headers (5) Test Results Status: 201 Created Time:



POST http://127.0.0.1:8088/users

Params Authorization Headers (10) Body ● Pre-request Script Tests Se

none form-data x-www-form-urlencoded [✓ raw] binary GraphQL J

```
{
    "name": "K",
    "birthDate": "2000-03-21T03:21:32.896+0000"
}
```

Body Cookies (1) Headers (3) Test Results Status: 400 Bad Request

Pretty Raw Preview Visualize Text

- *execute project*

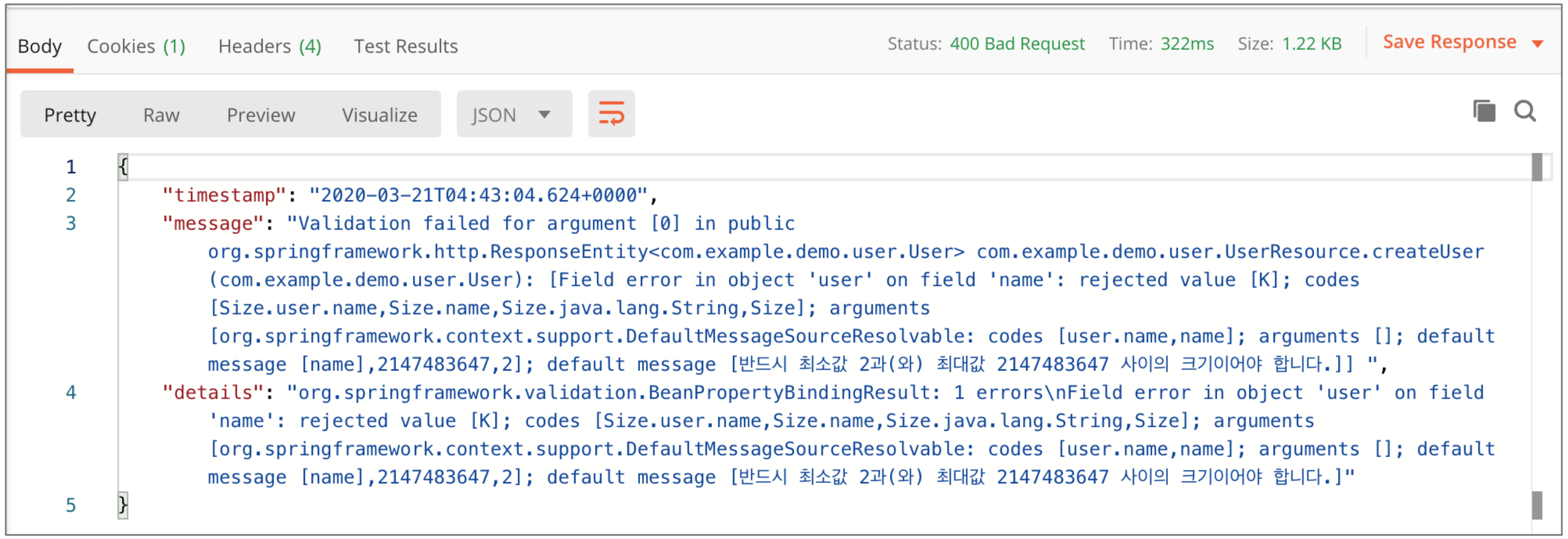
step14- Implementing Validations for RESTful Services

- ResponseEntityExceptionHandler → handleMethodArgumentNotValid

```java
@Override
protected ResponseEntity<Object> handleMethodArgumentNotValid(MethodArgumentNotValidException ex,
                                                              HttpHeaders headers, HttpStatus status, WebRequest request) {
    ExceptionResponse exceptionResponse =
            new ExceptionResponse(new Date(), ex.getMessage(), ex.getBindingResult().toString());

    return new ResponseEntity(exceptionResponse, HttpStatus.BAD_REQUEST);
}
```

CustomizedResponseEntityExceptionHander.java



Status: 400 Bad Request   Time: 322ms   Size: 1.22 KB

```
{
    "timestamp": "2020-03-21T04:43:04.624+0000",
    "message": "Validation failed for argument [0] in public
        org.springframework.http.ResponseEntity<com.example.demo.user.User> com.example.demo.user.UserResource.createUser
        (com.example.demo.user.User): [Field error in object 'user' on field 'name': rejected value [K]; codes
        [Size.user.name,Size.name,Size.java.lang.String,Size]; arguments
        [org.springframework.context.support.DefaultMessageSourceResolvable: codes [user.name,name]; arguments []; default
        message [name],2147483647,2]; default message [반드시 최소값 2과(와) 최대값 2147483647 사이의 크기이어야 합니다.]] ",
    "details": "org.springframework.validation.BeanPropertyBindingResult: 1 errors\nField error in object 'user' on field
        'name': rejected value [K]; codes [Size.user.name,Size.name,Size.java.lang.String,Size]; arguments
        [org.springframework.context.support.DefaultMessageSourceResolvable: codes [user.name,name]; arguments []; default
        message [name],2147483647,2]; default message [반드시 최소값 2과(와) 최대값 2147483647 사이의 크기이어야 합니다.]"
}
```

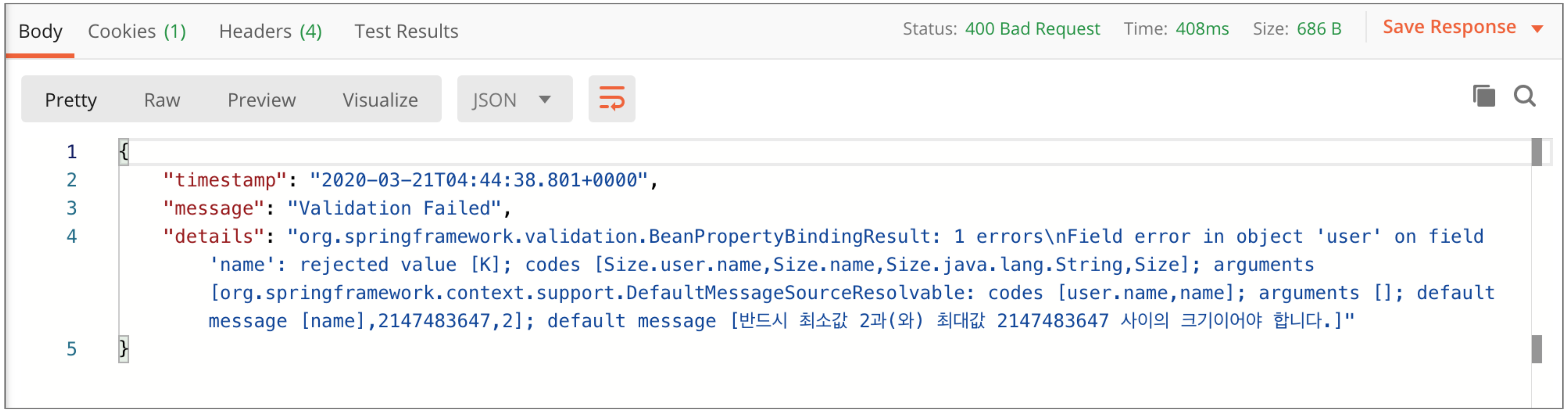
- *execute project*

step14- Implementing Validations for RESTful Services

```java
@Override
protected ResponseEntity<Object> handleMethodArgumentNotValid(MethodArgumentNotValidException ex,
                                                              HttpHeaders headers, HttpStatus status, WebRequest request) {
    ExceptionResponse exceptionResponse =
            new ExceptionResponse(new Date(), "Validation Failed", ex.getBindingResult().toString());

    return new ResponseEntity(exceptionResponse, HttpStatus.BAD_REQUEST);
}
```

CustomizedResponseEntityExceptionHander.java



Status: 400 Bad Request   Time: 408ms   Size: 686 B

```
{
    "timestamp": "2020-03-21T04:44:38.801+0000",
    "message": "Validation Failed",
    "details": "org.springframework.validation.BeanPropertyBindingResult: 1 errors\nField error in object 'user' on field
        'name': rejected value [K]; codes [Size.user.name,Size.name,Size.java.lang.String,Size]; arguments
        [org.springframework.context.support.DefaultMessageSourceResolvable: codes [user.name,name]; arguments []; default
        message [name],2147483647,2]; default message [반드시 최소값 2과(와) 최대값 2147483647 사이의 크기이어야 합니다.]"
}
```

- *execute project*

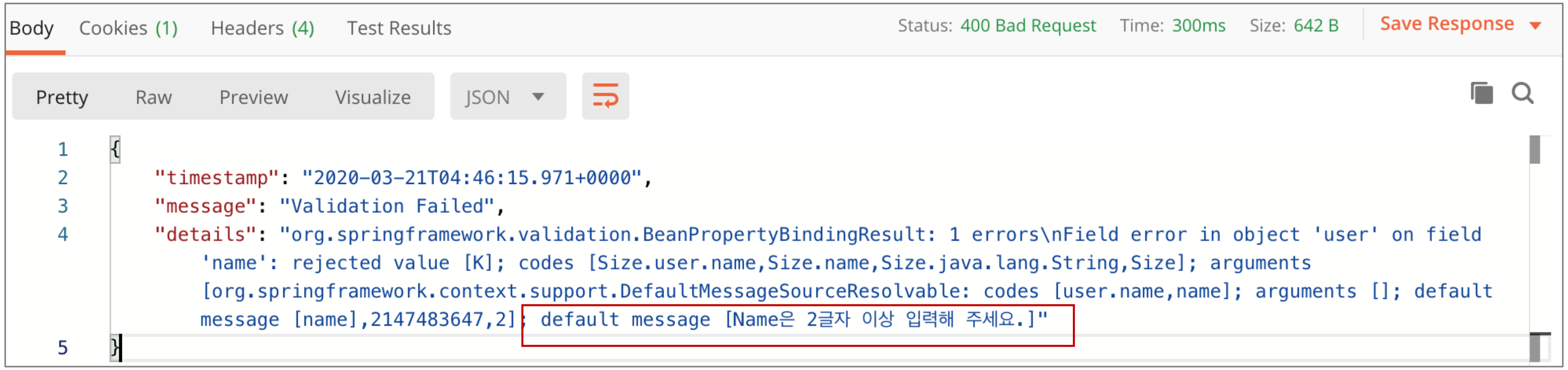
step14- Implementing Validations for RESTful Services

```java
public class User {
   private Integer id;

   @Size(min=2, message="Name은 2글자 이상 입력해 주세요.")
   private String name;

   @Past
   private Date birthDate;
}
```

- Validation API
  - jakarta.validation-api
  - hibernate-validator



Status: 400 Bad Request Time: 300ms Size: 642 B

```json
{
    "timestamp": "2020-03-21T04:46:15.971+0000",
    "message": "Validation Failed",
    "details": "org.springframework.validation.BeanPropertyBindingResult: 1 errors\nField error in object 'user' on field
        'name': rejected value [K]; codes [Size.user.name,Size.name,Size.java.lang.String,Size]; arguments
        [org.springframework.context.support.DefaultMessageSourceResolvable: codes [user.name,name]; arguments []; default
        message [name],2147483647,2]; default message [Name은 2글자 이상 입력해 주세요.]"
}
```

- *execute project*

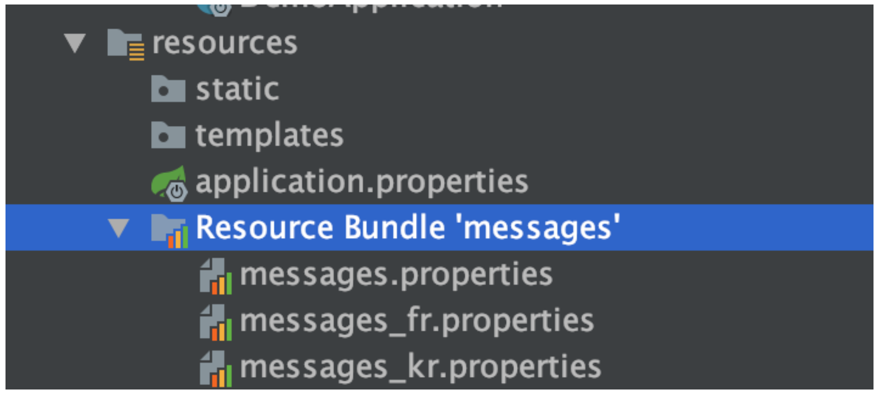
step15- Internationalization for RESTful Services

```
@GetMapping(path = "/hello-world-internationalized")
public String helloWorldInternationalized() {
   return "Good Morning";
}
```

HelloWorldController.class

- Internationalization
- @Configuration 등록
  - LocaleResolver
  - Default Locale
    - Locale.US or Locale.KOREA
  - ResourceBundleMessageSource
- Usage
  - generate message budle files
  - @Autowired MessageSource
  - @RequestHeader(value = “Accept-Language”, required = false) Local local
  - messageSource.getMessage(“greeting.message”, null, local)

step15- Internationalization for RESTful Services



```
resources
    static
    templates
    application.properties
    Resource Bundle 'messages'
        messages.properties
        messages_fr.properties
        messages_kr.properties
```

```
greeting.message=Hello
greeting.message=Bonjour
greeting.message=안녕하세요
greeting.message=こんにちは
```

```java
@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

    @Bean
    public LocaleResolver localResolver() {
        SessionLocaleResolver localeResolver = new SessionLocaleResolver();
        localeResolver.setDefaultLocale(Locale.US);
        return localeResolver;
    }
}
```

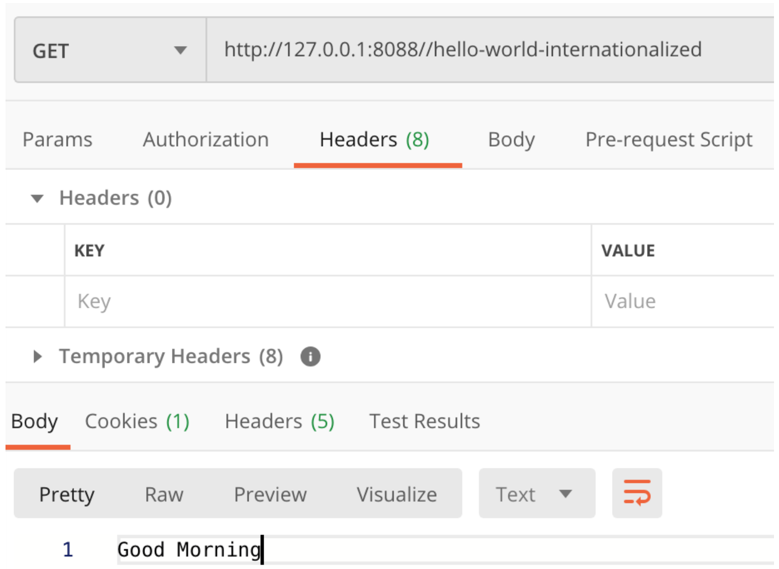
- application.yml

```yaml
spring:
  messages:
    basename: messages
```

step15- Internationalization for RESTful Services

```
@GetMapping(path = "/hello-world-internationalized")
public String helloWorldInternationalized(
        @RequestHeader(name="Accept-Language", required = false ) Locale locale) {
    return messageSource.getMessage(”greeting.message", null, locale);
}
```

```
@GetMapping(path = "/hello-world-internationalized")
public String helloWorldInternationalized() {
    return messageSource.getMessage("good.morning.message", null, LocaleContextHolder.getLocale());
}
```


GET http://127.0.0.1:8088//hello-world-internationalized

Params | Authorization | Headers (8) | Body | Pre-request Script

Headers (0)

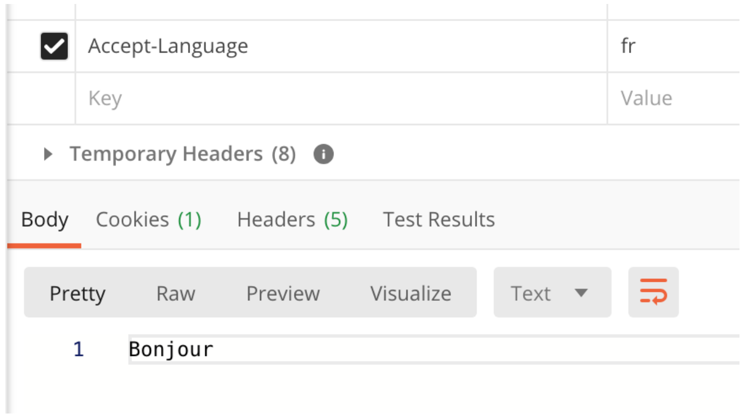
KEY	VALUE
Key	Value

Temporary Headers (8)

Body | Cookies (1) | Headers (5) | Test Results

Pretty | Raw | Preview | Visualize | Text

```
Good Morning
```



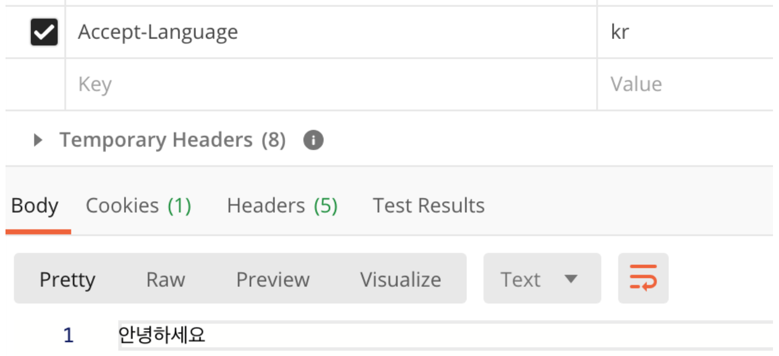
✓ Accept-Language	fr
Key	Value

Temporary Headers (8)

Body | Cookies (1) | Headers (5) | Test Results

Pretty | Raw | Preview | Visualize | Text

```
Bonjour
```



✓ Accept-Language	kr
Key	Value

Temporary Headers (8)

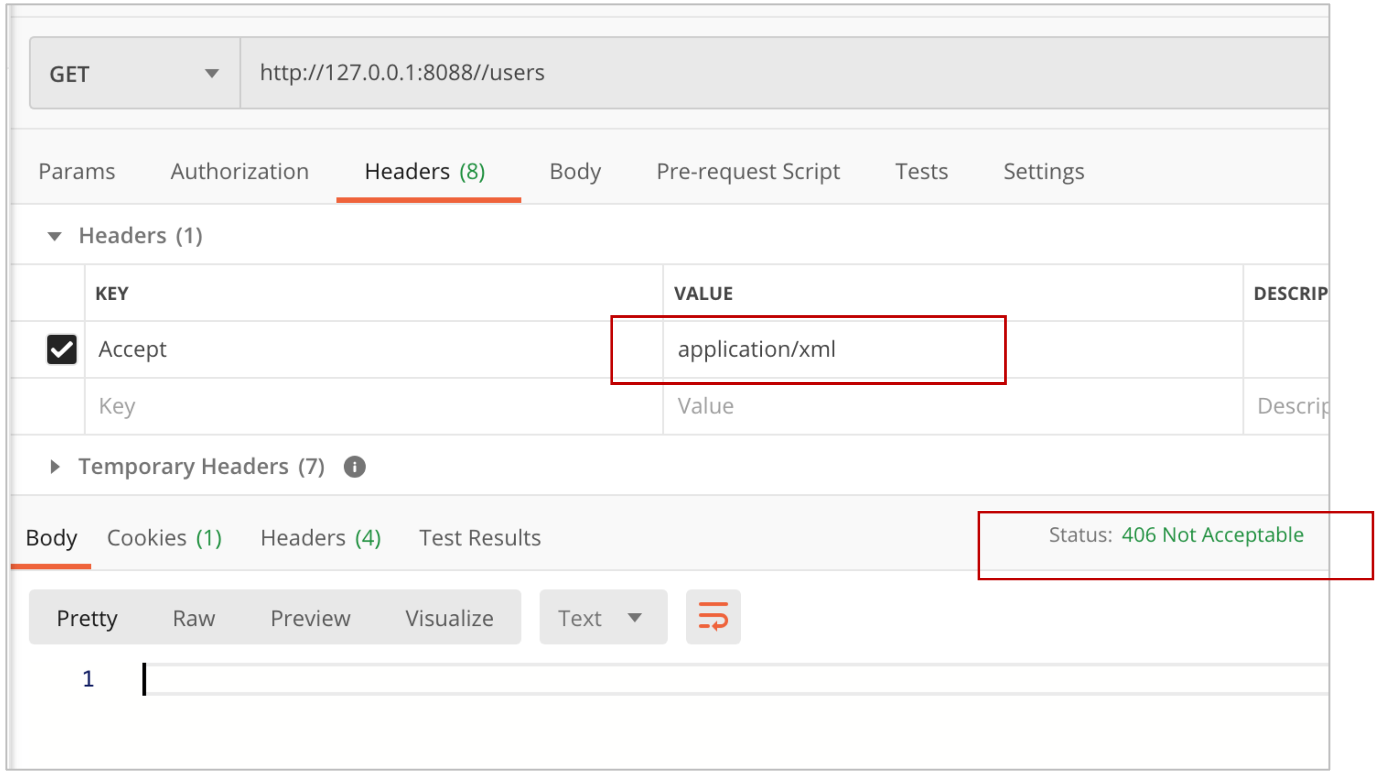
Body | Cookies (1) | Headers (5) | Test Results

Pretty | Raw | Preview | Visualize | Text

```
안녕하세요
```

- *execute project*

step16- Content Negotiation - Implementing Support for XML



GET http://127.0.0.1:8088//users

Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings

Headers (1)

	KEY	VALUE	DESCRIP
✓	Accept	application/xml	
	Key	Value	Descrip

Temporary Headers (7)

Body | Cookies (1) | Headers (4) | Test Results

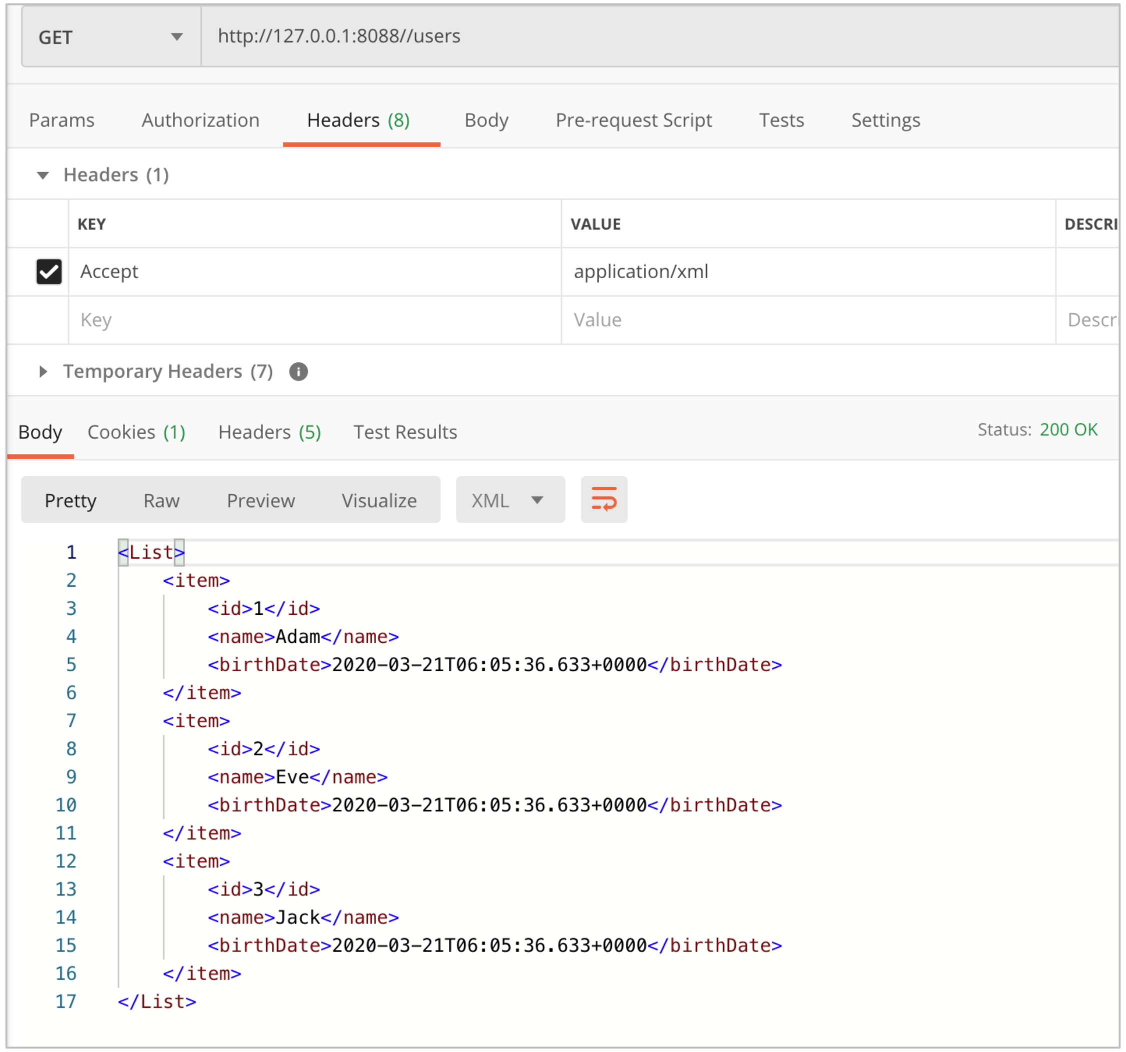
Status: 406 Not Acceptable

Pretty | Raw | Preview | Visualize | Text

step16- Content Negotiation - Implementing Support for XML

- pom.xml
  - add dependency

```
<dependency>
  <groupId>com.fasterxml.jackson.dataformat</groupId>
  <artifactId>jackson-dataformat-xml</artifactId>
  <version>2.10.2</version>
</dependency>
```



GET http://127.0.0.1:8088//users

Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings

Headers (1)

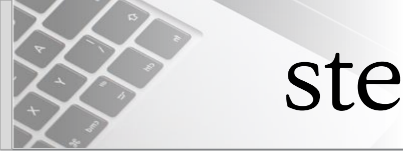
KEY	VALUE	DESCRI
✓ Accept	application/xml	
Key	Value	Descr

Temporary Headers (7)

Body | Cookies (1) | Headers (5) | Test Results — Status: 200 OK

Pretty | Raw | Preview | Visualize | XML

```
<List>
    <item>
        <id>1</id>
        <name>Adam</name>
        <birthDate>2020-03-21T06:05:36.633+0000</birthDate>
    </item>
    <item>
        <id>2</id>
        <name>Eve</name>
        <birthDate>2020-03-21T06:05:36.633+0000</birthDate>
    </item>
    <item>
        <id>3</id>
        <name>Jack</name>
        <birthDate>2020-03-21T06:05:36.633+0000</birthDate>
    </item>
</List>
```



step17- Implementing Static Filtering for RESTful Service

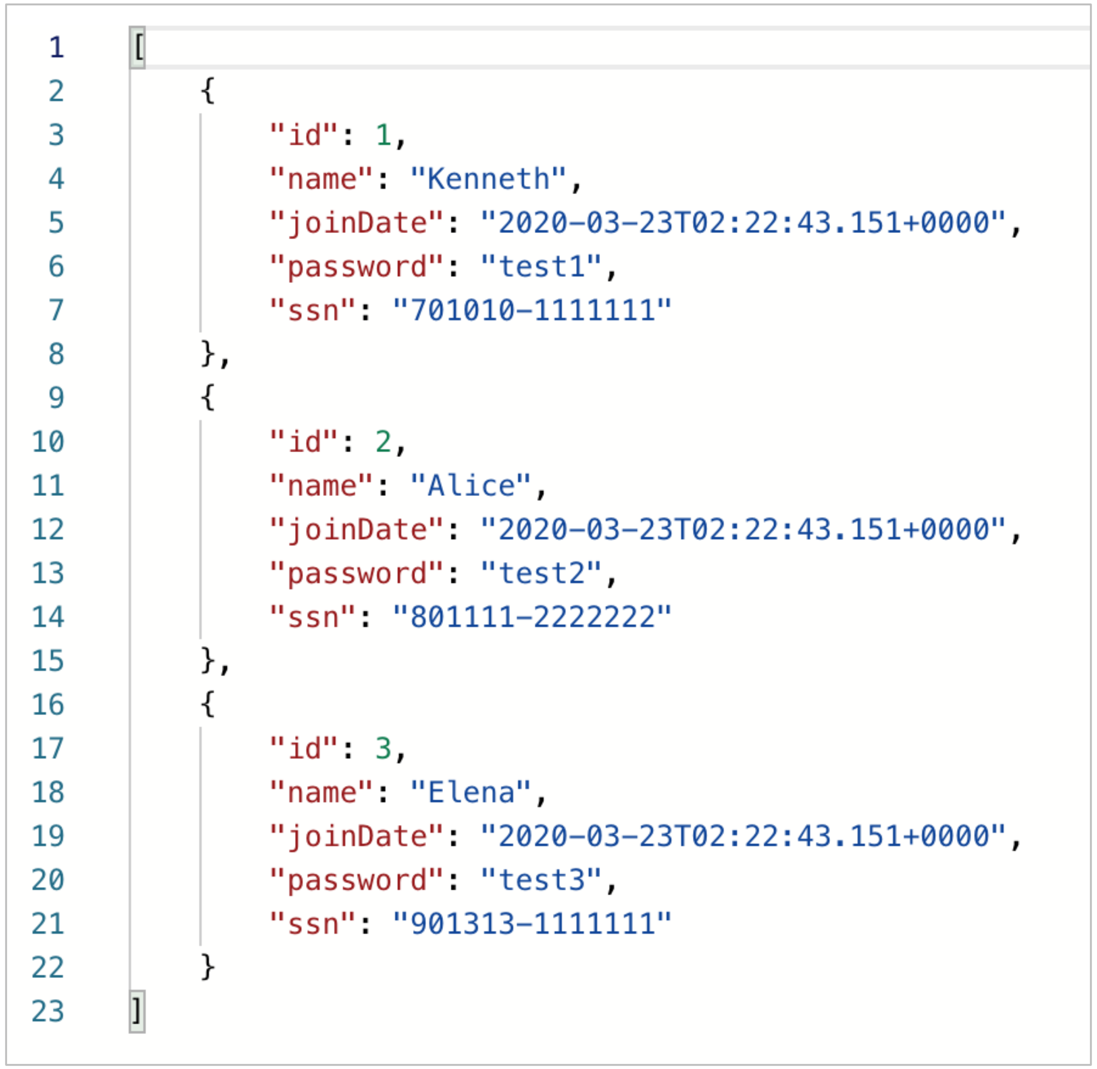
```
@Data
@AllArgsConstructor
public class User {
    ...

    private String password;
    private String ssn;
}
```

```
@Component
public class UserDaoService {
    private static List<User> users = new ArrayList<>();

    private static int usersCount = 3;

    static {
        users.add(new User(1, "Kenneth", new Date(), "test1", "701010-1111111"));
        users.add(new User(2, "Alice", new Date(), "test2", "801111-2222222"));
        users.add(new User(3, "Elena", new Date(), "test3", "901313-1111111"));
    }
```



```
[
    {
        "id": 1,
        "name": "Kenneth",
        "joinDate": "2020-03-23T02:22:43.151+0000",
        "password": "test1",
        "ssn": "701010-1111111"
    },
    {
        "id": 2,
        "name": "Alice",
        "joinDate": "2020-03-23T02:22:43.151+0000",
        "password": "test2",
        "ssn": "801111-2222222"
    },
    {
        "id": 3,
        "name": "Elena",
        "joinDate": "2020-03-23T02:22:43.151+0000",
        "password": "test3",
        "ssn": "901313-1111111"
    }
]
```

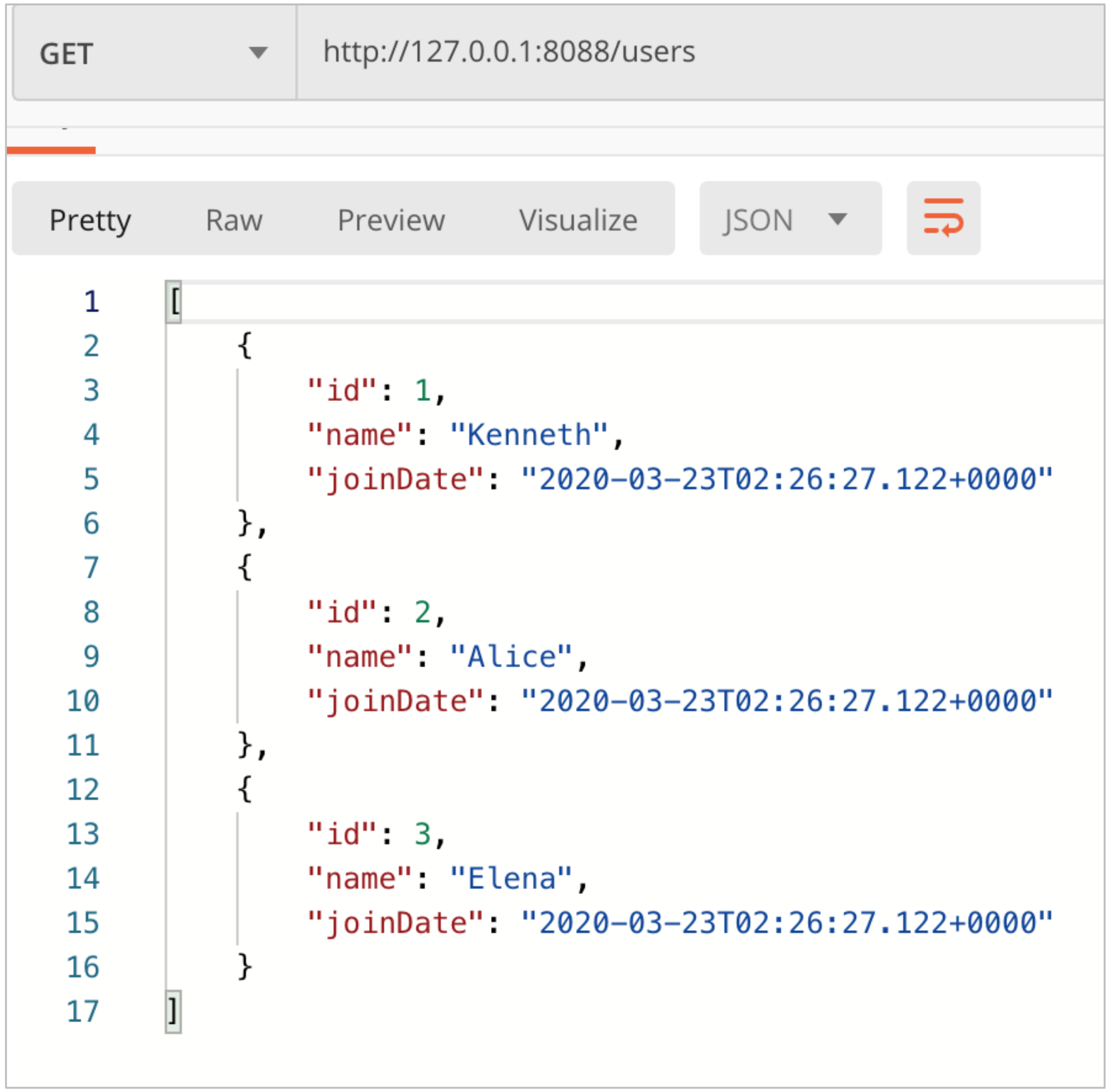
- *execute project*

step17- Implementing Static Filtering for RESTful Service

- @JsonIgnore

```
@Data
@AllArgsConstructor
public class User {
   ...

   @JsonIgnore
   private String password;
   @JsonIgnore
   private String ssn;
}
```



http://127.0.0.1:8088/users

```
[
    {
        "id": 1,
        "name": "Kenneth",
        "joinDate": "2020-03-23T02:26:27.122+0000"
    },
    {
        "id": 2,
        "name": "Alice",
        "joinDate": "2020-03-23T02:26:27.122+0000"
    },
    {
        "id": 3,
        "name": "Elena",
        "joinDate": "2020-03-23T02:26:27.122+0000"
    }
]
```

step17- Implementing Static Filtering for RESTful Service

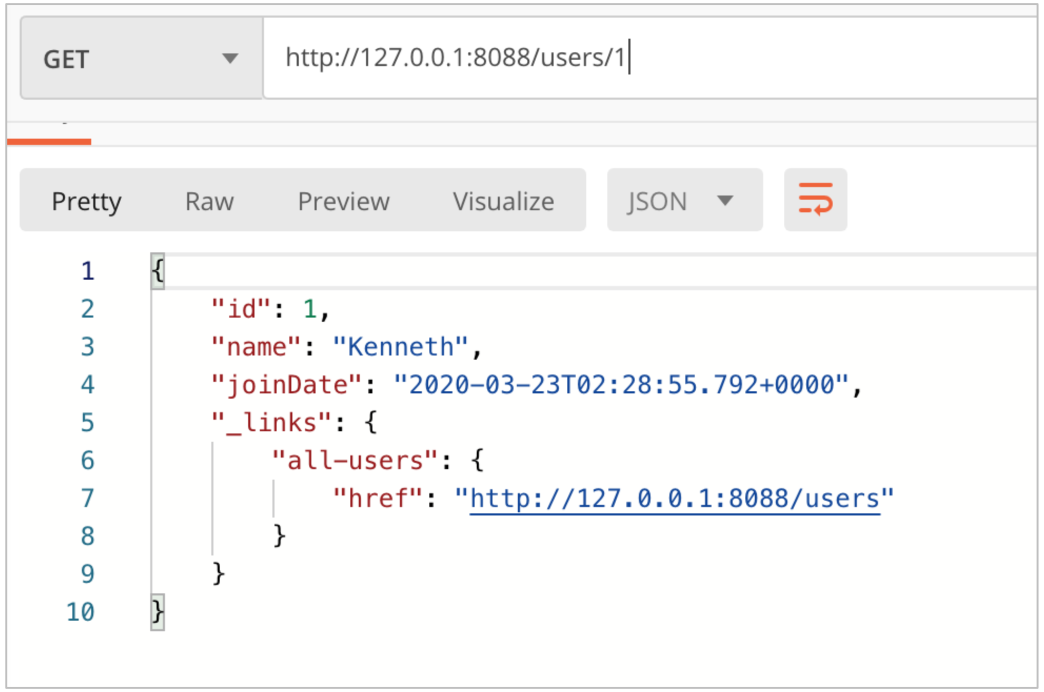
- @JsonIgnoreProperties

```
@Data
@AllArgsConstructor
@JsonIgnoreProperties(value={"password", "ssn"})
public class User {
    private Integer id;

    @Size(min=2, message="Name은 2글자 이상 입력해 주세요.")
    private String name;

    @Past
    private Date joinDate;

//    @JsonIgnore
    private String password;
//    @JsonIgnore
    private String ssn;
}
```



http://127.0.0.1:8088/users/1

```
{
    "id": 1,
    "name": "Kenneth",
    "joinDate": "2020-03-23T02:28:55.792+0000",
    "_links": {
        "all-users": {
            "href": "http://127.0.0.1:8088/users"
        }
    }
}
```

step18- Implementing Dynamic Filtering for RESTful Service

- Retrieve user info for admin

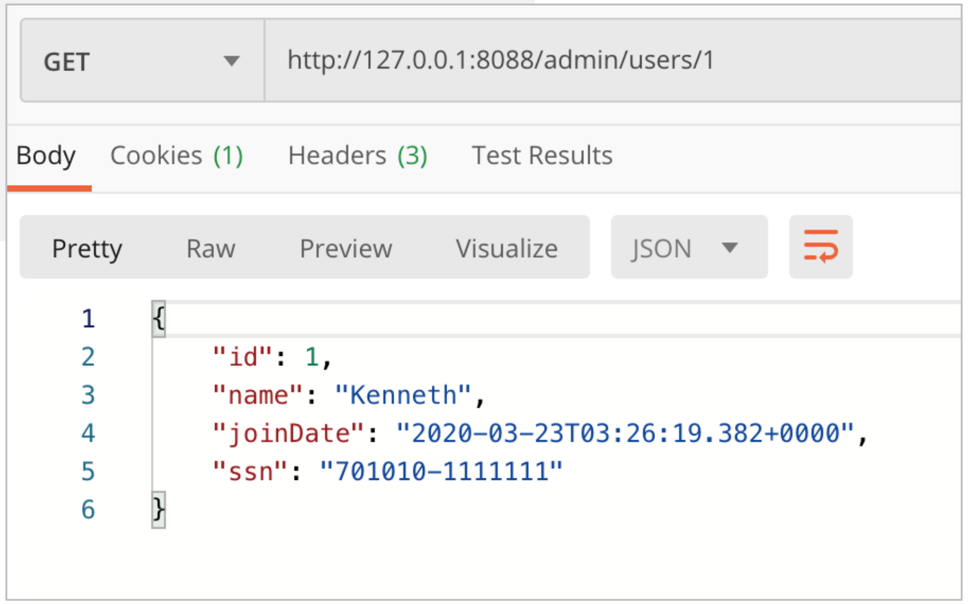
```
@GetMapping("/admin/users/{id}")
public MappingJacksonValue retrieveUser4Admin(@PathVariable int id) {
    ...
    SimpleBeanPropertyFilter filter = SimpleBeanPropertyFilter.filterOutAllExcept("id", "name", "joinDate", "ssn");

    FilterProvider filters = new SimpleFilterProvider().addFilter("UserInfo", filter);

    MappingJacksonValue mapping = new MappingJacksonValue(user);
    mapping.setFilters(filters);

    return mapping;
}
```

```
//@JsonIgnoreProperties(value={"password", "ssn"})
@JsonFilter("UserInfo")
public class User {
        ...
//    @JsonIgnore
    private String password;
//    @JsonIgnore
    private String ssn;
}
```



http://127.0.0.1:8088/admin/users/1

```
{
    "id": 1,
    "name": "Kenneth",
    "joinDate": "2020-03-23T03:26:19.382+0000",
    "ssn": "701010-1111111"
}
```

step18- Implementing Dynamic Filtering for RESTful Service

- Retrieve users list for admin

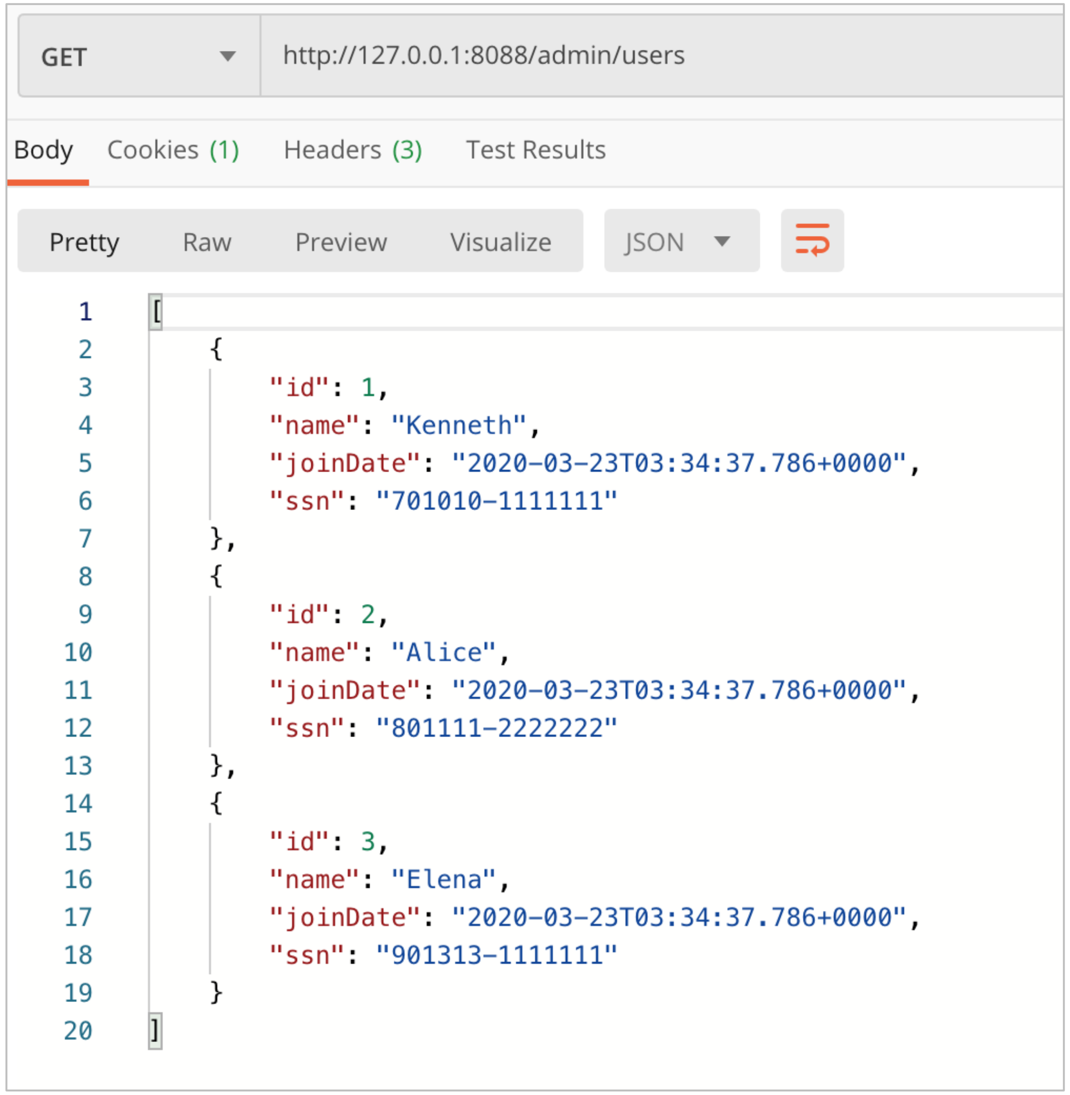
```
@GetMapping("/admin/users")
public MappingJacksonValue retrieveUsers4Admin(@PathVariable int
id) {
    List<User> users = service.findAll();

    SimpleBeanPropertyFilter filter =
SimpleBeanPropertyFilter.filterOutAllExcept("id", "name", "joinDate",
"ssn");

    FilterProvider filters = new
SimpleFilterProvider().addFilter("UserInfo", filter);

    MappingJacksonValue mapping = new MappingJacksonValue(users);
    mapping.setFilters(filters);

    return mapping;
}
```



http://127.0.0.1:8088/admin/users

```
[
    {
        "id": 1,
        "name": "Kenneth",
        "joinDate": "2020-03-23T03:34:37.786+0000",
        "ssn": "701010-1111111"
    },
    {
        "id": 2,
        "name": "Alice",
        "joinDate": "2020-03-23T03:34:37.786+0000",
        "ssn": "801111-2222222"
    },
    {
        "id": 3,
        "name": "Elena",
        "joinDate": "2020-03-23T03:34:37.786+0000",
        "ssn": "901313-1111111"
    }
]
```

step19- Versioning - Basic Approach with URIs

- /admin/users/{id} → /v1/admin/users/{id}

```
@GetMapping("/admin/users/{id}")
public MappingJacksonValue retrieveUser4Admin(@PathVariable int id) {
   ...
```

URI 변경

```
@GetMapping("/v1/admin/users/{id}")
public MappingJacksonValue retrieveUser4AdminV1(@PathVariable int id) {
   ...
```

- Add a new user bean

```
@Data
@NoArgsConstructor
@AllArgsConstructor
@JsonFilter("UserInfoV2")
public class UserV2 extends User {

    private String grade;

}
```

step19- Versioning - Basic Approach with URIs

- URI versioning

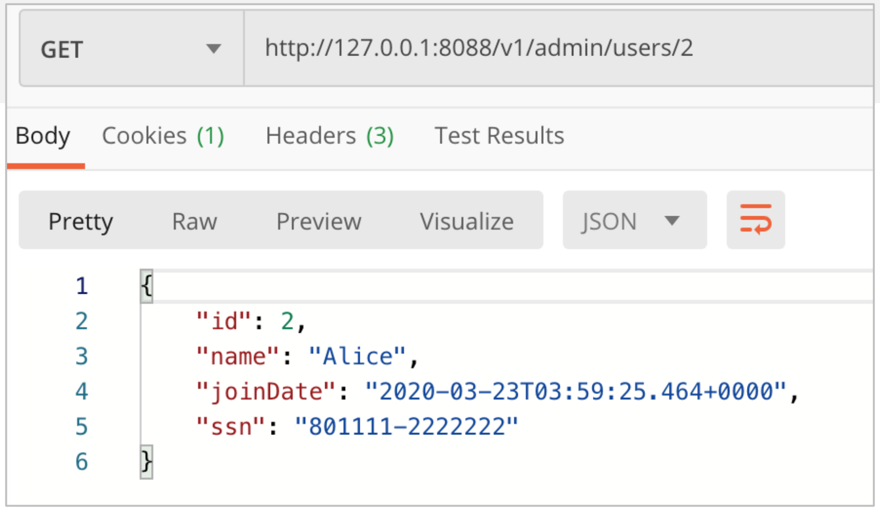
```
@GetMapping("/v2/admin/users/{id}")
public MappingJacksonValue retrieveUser4AdminV2(@PathVariable int id) {
    ...
    UserV2 userV2 = new UserV2();
    BeanUtils.copyProperties(user, userV2);
    userV2.setGrade("VIP");

    SimpleBeanPropertyFilter filter = SimpleBeanPropertyFilter.filterOutAllExcept("id", "name", "joinDate", "grade");

    FilterProvider filters = new SimpleFilterProvider().addFilter("UserInfoV2", filter);

    MappingJacksonValue mapping = new MappingJacksonValue(userV2);
    mapping.setFilters(filters);

    return mapping;
}
```

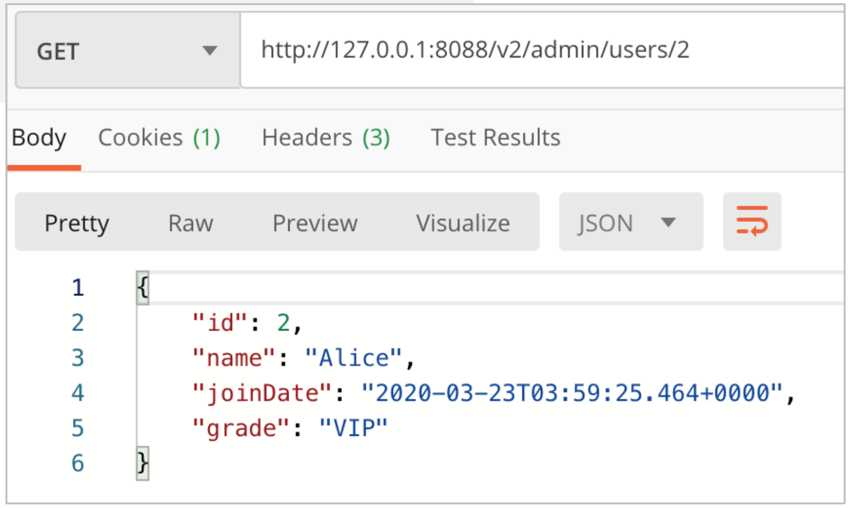


GET http://127.0.0.1:8088/v1/admin/users/2

Body | Cookies (1) | Headers (3) | Test Results

Pretty | Raw | Preview | Visualize | JSON

```
{
    "id": 2,
    "name": "Alice",
    "joinDate": "2020-03-23T03:59:25.464+0000",
    "ssn": "801111-2222222"
}
```



GET http://127.0.0.1:8088/v2/admin/users/2

Body | Cookies (1) | Headers (3) | Test Results

Pretty | Raw | Preview | Visualize | JSON

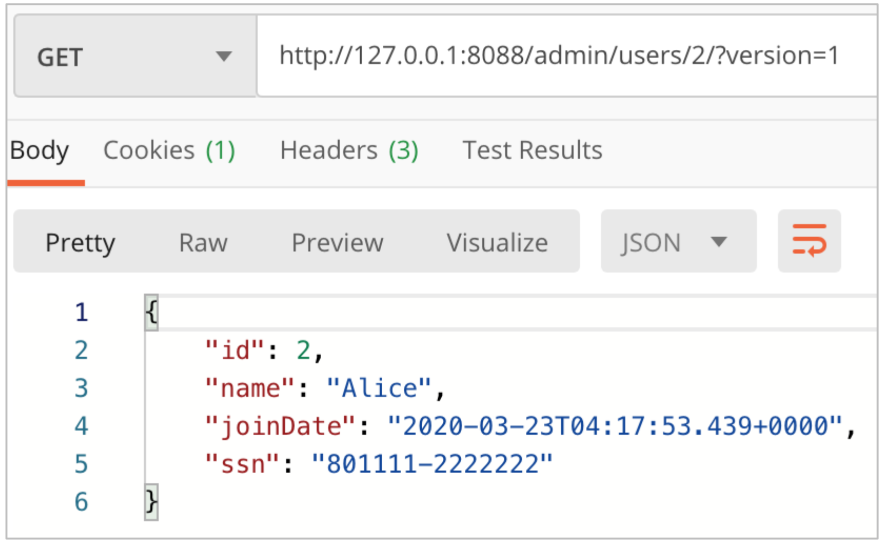
```
{
    "id": 2,
    "name": "Alice",
    "joinDate": "2020-03-23T03:59:25.464+0000",
    "grade": "VIP"
}
```

- *execute project*

step20- Versioning - Header and Content Negotiation Approach

- request parameter versioning

```
@GetMapping(value = "/admin/users/{id}/", params = "version=1")
public MappingJacksonValue retrieveUser4AdminV1(@PathVariable int id) {
    User user = service.findOne(id);
```

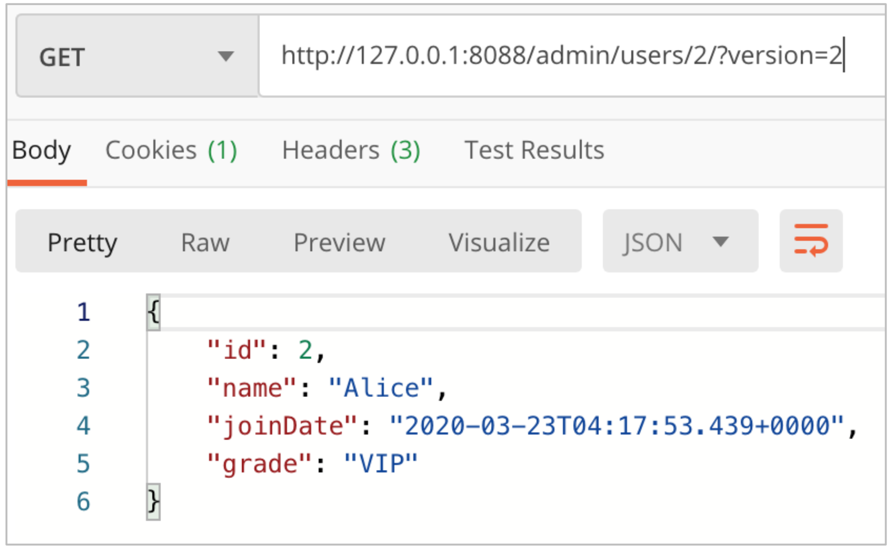

GET http://127.0.0.1:8088/admin/users/2/?version=1

Body Cookies (1) Headers (3) Test Results

Pretty Raw Preview Visualize JSON

```
{
    "id": 2,
    "name": "Alice",
    "joinDate": "2020-03-23T04:17:53.439+0000",
    "ssn": "801111-2222222"
}
```

```
@GetMapping(value = "/admin/users/{id}/", params = "version=2")
public MappingJacksonValue retrieveUser4AdminV2(@PathVariable int id) {
    User user = service.findOne(id);
```


GET http://127.0.0.1:8088/admin/users/2/?version=2

Body Cookies (1) Headers (3) Test Results

Pretty Raw Preview Visualize JSON

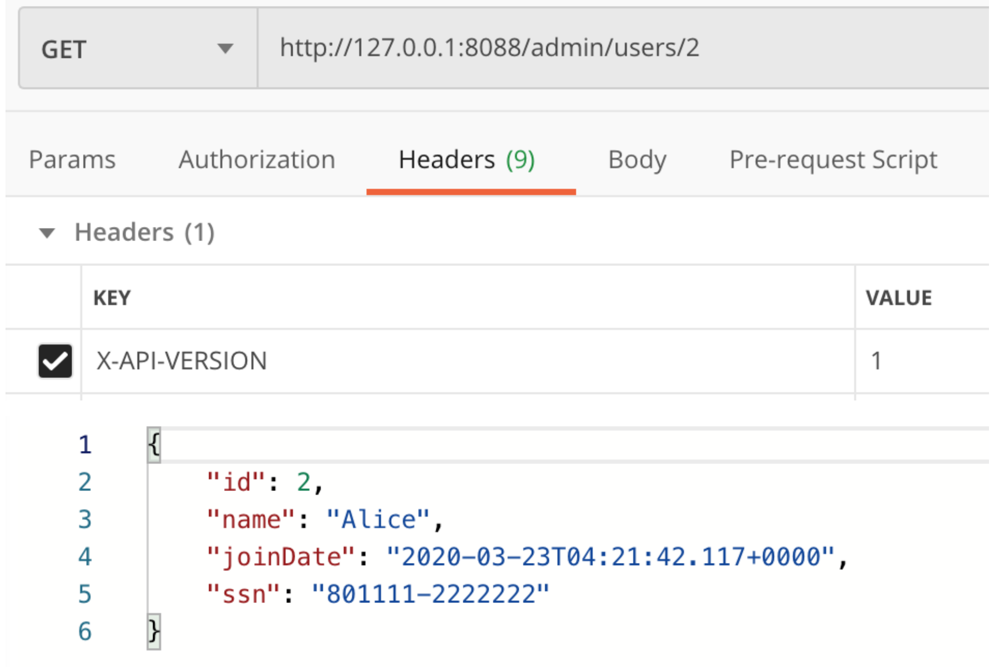
```
{
    "id": 2,
    "name": "Alice",
    "joinDate": "2020-03-23T04:17:53.439+0000",
    "grade": "VIP"
}
```

- *execute project*

step20- Versioning - Header and Content Negotiation Approach

- header versioning

```
@GetMapping(value = "/admin/users/{id}", headers = "X-API-VERSION=1")
public MappingJacksonValue retrieveUser4AdminV1(@PathVariable int id) {
   User user = service.findOne(id);
```



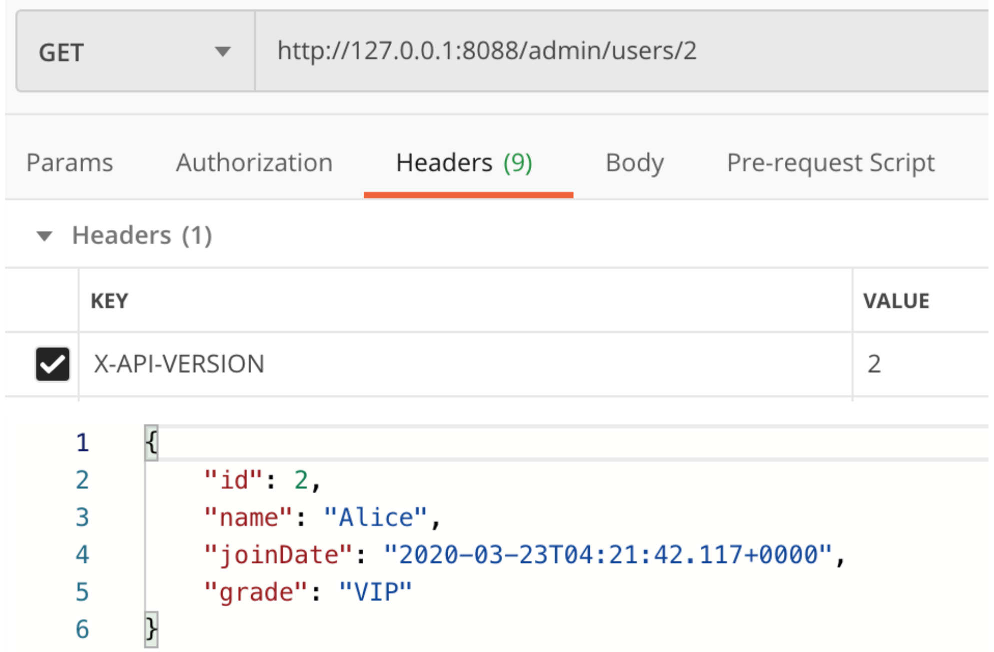
GET http://127.0.0.1:8088/admin/users/2

Params | Authorization | Headers (9) | Body | Pre-request Script

Headers (1)

	KEY	VALUE
✓	X-API-VERSION	1

```
{
    "id": 2,
    "name": "Alice",
    "joinDate": "2020-03-23T04:21:42.117+0000",
    "ssn": "801111-2222222"
}
```



GET http://127.0.0.1:8088/admin/users/2

Params | Authorization | Headers (9) | Body | Pre-request Script

Headers (1)

	KEY	VALUE
✓	X-API-VERSION	2

```
{
    "id": 2,
    "name": "Alice",
    "joinDate": "2020-03-23T04:21:42.117+0000",
    "grade": "VIP"
}
```

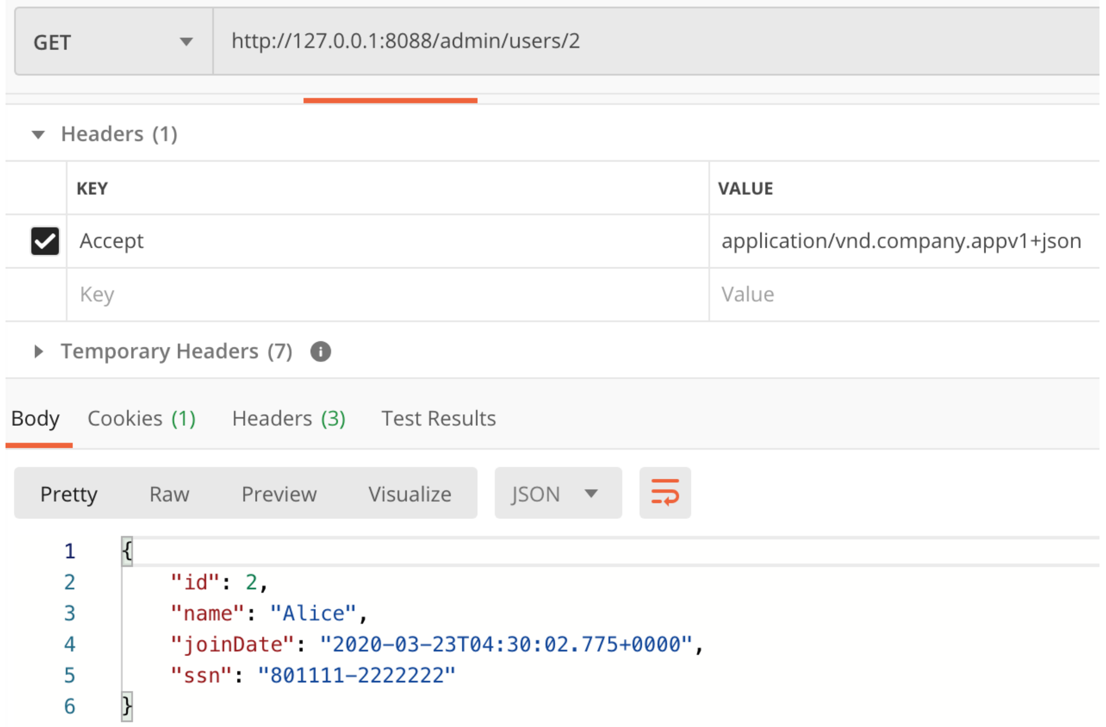
```
@GetMapping(value = "/admin/users/{id}", headers = "X-API-VERSION=2")
public MappingJacksonValue retrieveUser4AdminV2(@PathVariable int id) {
   User user = service.findOne(id);
```

- *execute project*

step20- Versioning - Header and Content Negotiation Approach

- mime-type or accept header versioning

```
@GetMapping(value = "/admin/users/{id}", produces = "application/vnd.company.appv1+json")
public MappingJacksonValue retrieveUser4AdminV1(@PathVariable int id) {
   User user = service.findOne(id);
```



GET http://127.0.0.1:8088/admin/users/2

Headers (1)

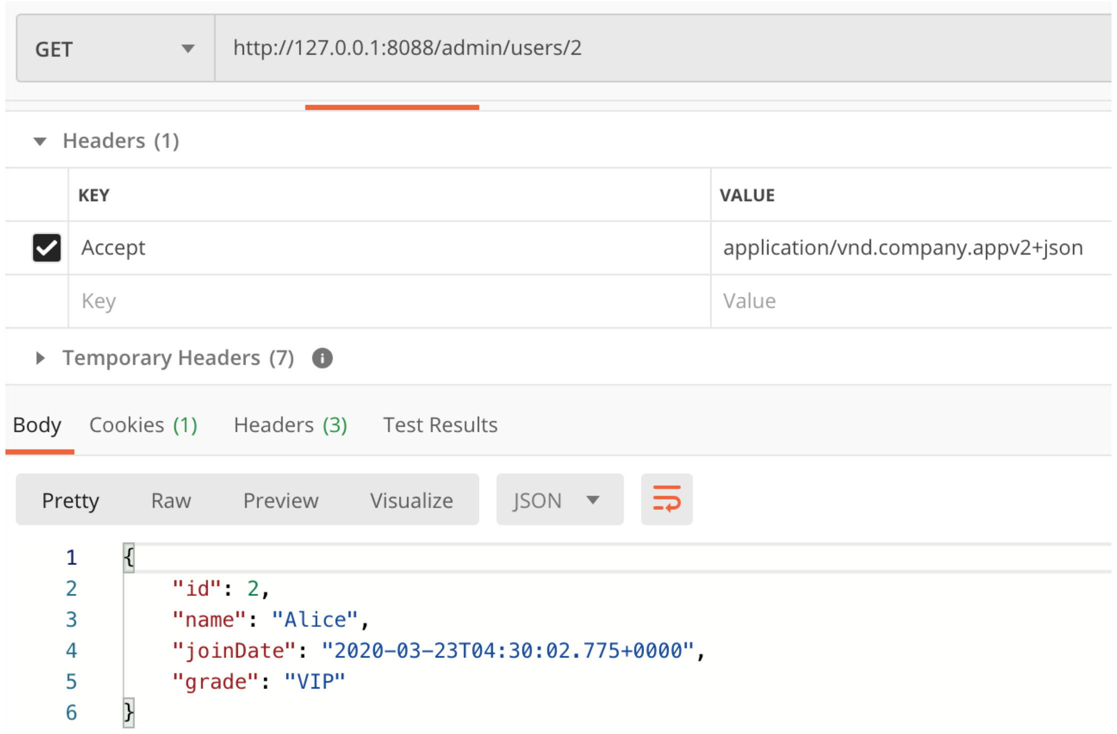
KEY	VALUE
✓ Accept	application/vnd.company.appv1+json
Key	Value

Temporary Headers (7)

Body · Cookies (1) · Headers (3) · Test Results

Pretty · Raw · Preview · Visualize · JSON

```
{
    "id": 2,
    "name": "Alice",
    "joinDate": "2020-03-23T04:30:02.775+0000",
    "ssn": "801111-2222222"
}
```



GET http://127.0.0.1:8088/admin/users/2

Headers (1)

KEY	VALUE
✓ Accept	application/vnd.company.appv2+json
Key	Value

Temporary Headers (7)

Body · Cookies (1) · Headers (3) · Test Results

Pretty · Raw · Preview · Visualize · JSON

```
{
    "id": 2,
    "name": "Alice",
    "joinDate": "2020-03-23T04:30:02.775+0000",
    "grade": "VIP"
}
```

```
@GetMapping(value = "/admin/users/{id}", produces = "application/vnd.company.appv2+json")
public MappingJacksonValue retrieveUser4AdminV2(@PathVariable int id) {
   User user = service.findOne(id);
```

- *execute project*

step20- Versioning - Header and Content Negotiation Approach

- Versioning

1) Media type versioning (a.k.a “content negotiation” or “accept header”)
   - GitHub

2) (Custom) headers versioning
   - Microsoft

→ 일반 브라우저에서 실행 안됨

3) URI Versioning
   - Twitter

4) Request Parameter versioning
   - Amazon

→ 일반 브라우저에서 실행 가능

- Factors
  - URI Pollution
  - Misuse of HTTP Headers
  - Caching
  - Can we execute the request on the browser?
  - API Documentation
- No Perfect solution