RestAPI with DB

Spring Boot Recipe

1. Spring Initializr: Web, H2, Data JPA (create online and import to Intellij)
2. Add an extra dependency: springdoc-openapi-ui
3. Create application layers (packages):
    - model
    - controller
    - service
    - repository
4. Add some classes/interfaces
    - model.Hobbit (a class)
    - controller.HobbitController (a class)
    - service.HobbitService (a class)
    - repository.HobbitRepository (an interface)
5. Add some superpowers to the classes/interfaces
6. Glue HobbitService and HobbitRepository (@Autowired)
7. HobbitService: implement List<Hobbit> getAll() in the service, using the repository method
8. Glue HobbitController and HobbitService (@Autowired)
9. HobbitController: implement @GetMapping("/hobbits") List<Hobbit> getAll() in the controller, using the service method
8. Fine-tune H2 DB
9. Run the app
    - use H2 / openapi console to add/get some hobbits
10. Add @PostMapping controller and use service and repository to save a hobbit

Spring Initializr

Web, H2, Spring Data(create online and import to Intellij)

spring initializr

Project

- (●) Maven Project
- ( ) Gradle Project

Language

- (●) Java
- ( ) Kotlin
- ( ) Groovy

Spring Boot

- ( ) 3.0.0 (SNAPSHOT)
- ( ) 3.0.0 (M2)
- ( ) 2.7.0 (SNAPSHOT)
- ( ) 2.7.0 (RC1)
- ( ) 2.6.8 (SNAPSHOT)
- (●) 2.6.7
- ( ) 2.5.14 (SNAPSHOT)
- ( ) 2.5.13

Project Metadata

Group	com.example
Artifact	demo
Name	demo
Description	Demo project for Spring Boot
Package name	com.example.demo
Packaging	(●) Jar ( ) War
Java	(●) 18 ( ) 17 ( ) 11 ( ) 8

Dependencies ADD DEPENDENCIES... ⌘ + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

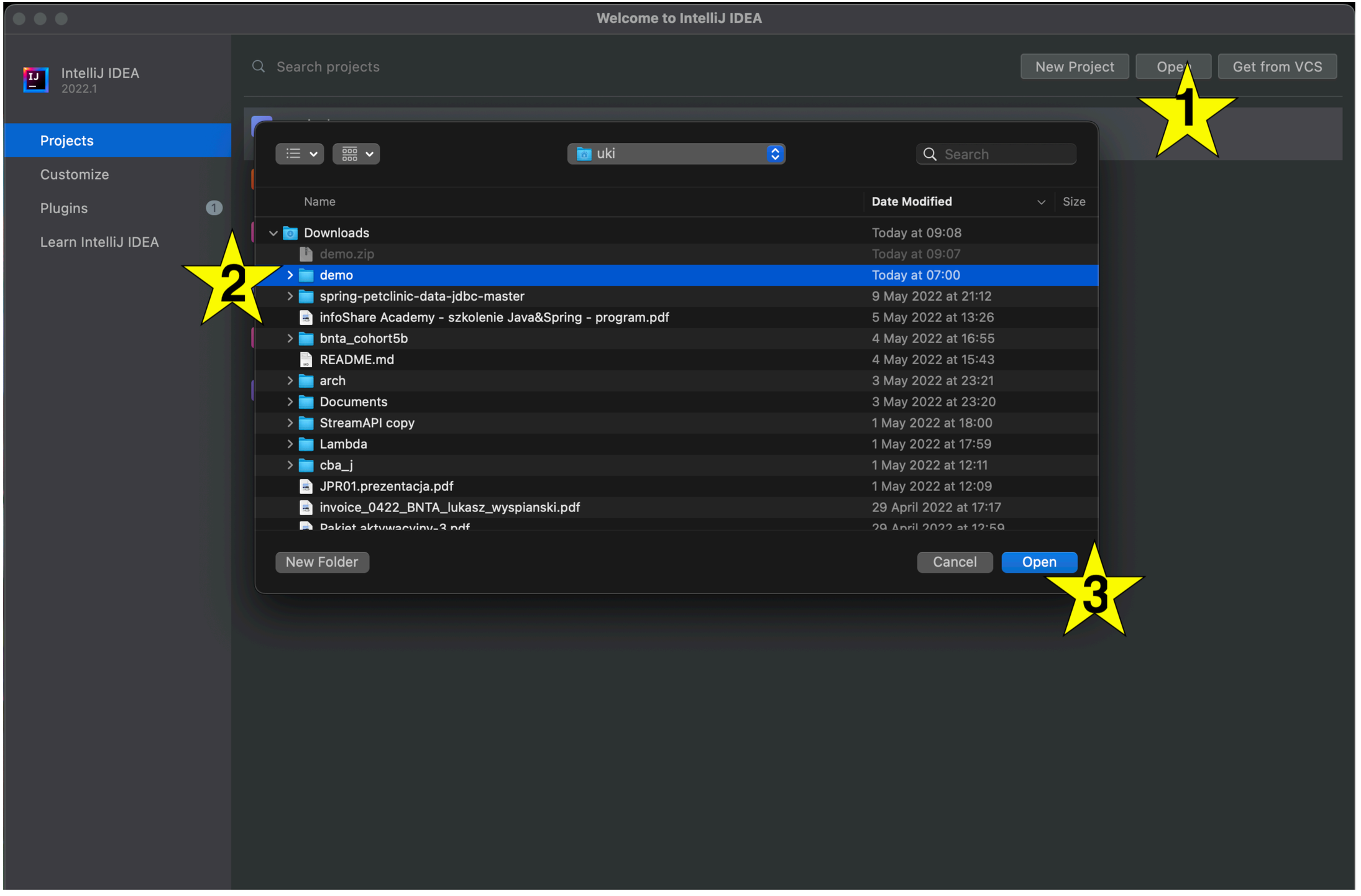
H2 Database SQL

Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.

Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

GENERATE ⌘ + ↵ EXPLORE CTRL + SPACE SHARE...


Welcome to IntelliJ IDEA
IntelliJ IDEA
2022.1
Projects
Customize
Plugins
1
Learn IntelliJ IDEA
Search projects
New Project
Open
Get from VCS
1
uki
Search
Name
Date Modified
Size
Downloads
Today at 09:08
demo.zip
Today at 09:07
demo
Today at 07:00
spring-petclinic-data-jdbc-master
9 May 2022 at 21:12
infoShare Academy - szkolenie Java&Spring - program.pdf
5 May 2022 at 13:26
bnta_cohort5b
4 May 2022 at 16:55
README.md
4 May 2022 at 15:43
arch
3 May 2022 at 23:21
Documents
3 May 2022 at 23:20
StreamAPI copy
1 May 2022 at 18:00
Lambda
1 May 2022 at 17:59
cba_j
1 May 2022 at 12:11
JPR01.prezentacja.pdf
1 May 2022 at 12:09
invoice_0422_BNTA_lukasz_wyspianski.pdf
29 April 2022 at 17:17
2
New Folder
Cancel
Open
3

Add an extra dependency

springdoc-openapi-ui

```xml
<dependencies>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<dependency>
  <groupId>com.h2database</groupId>
  <artifactId>h2</artifactId>
  <scope>runtime</scope>
</dependency>

  <dependency>
    <groupId>org.springdoc</groupId>
    <artifactId>springdoc-openapi-ui</artifactId>
    <version>1.6.8</version>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>

</dependencies>
```

Interact with HTTP clients.

Interact with relational databases,
use an ORM(Hibernate) framework under the hood.

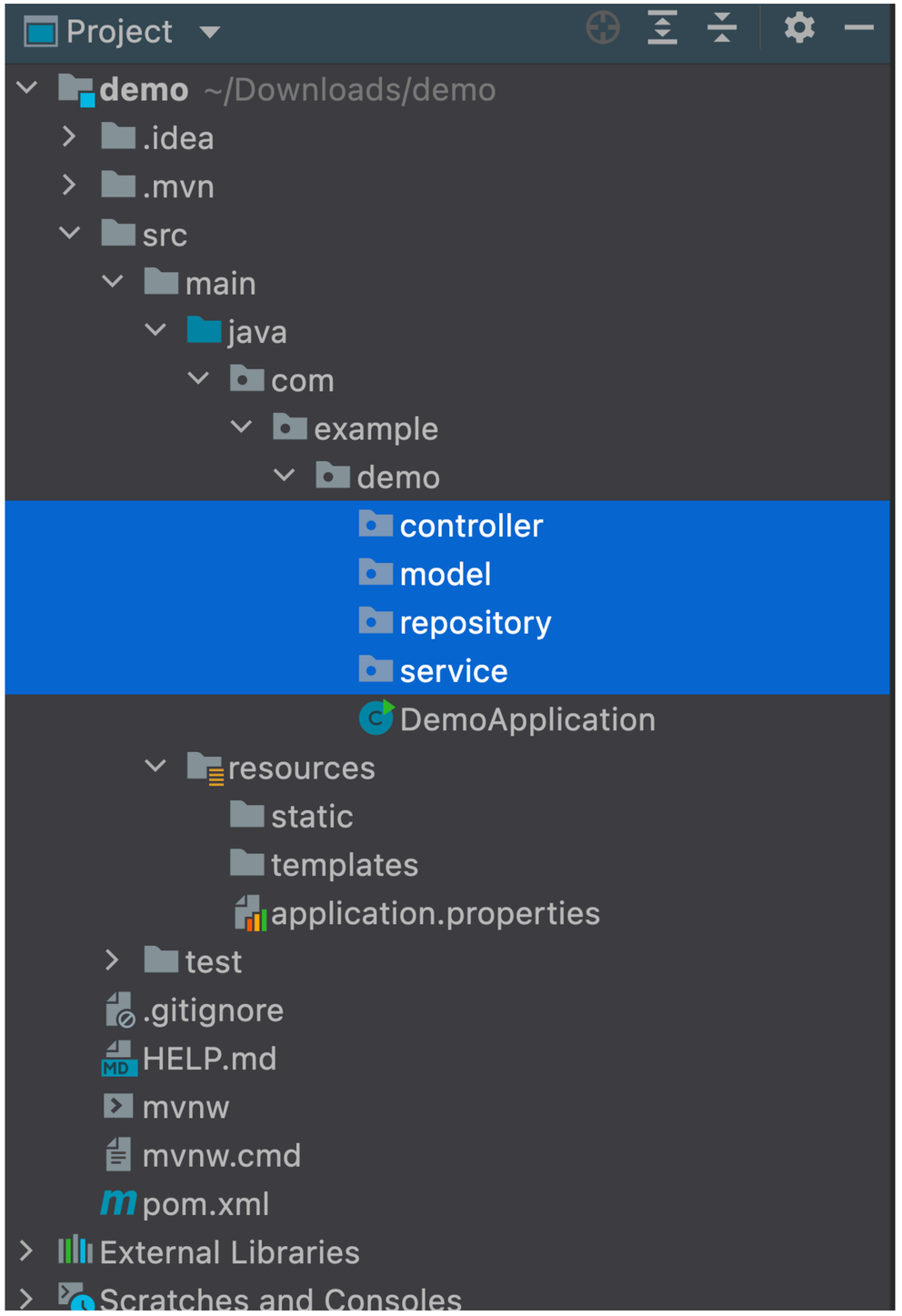
In-memory database.

UI to manually test the app's REST endpoints.

Junit & Spring libraries to write automatic tests.

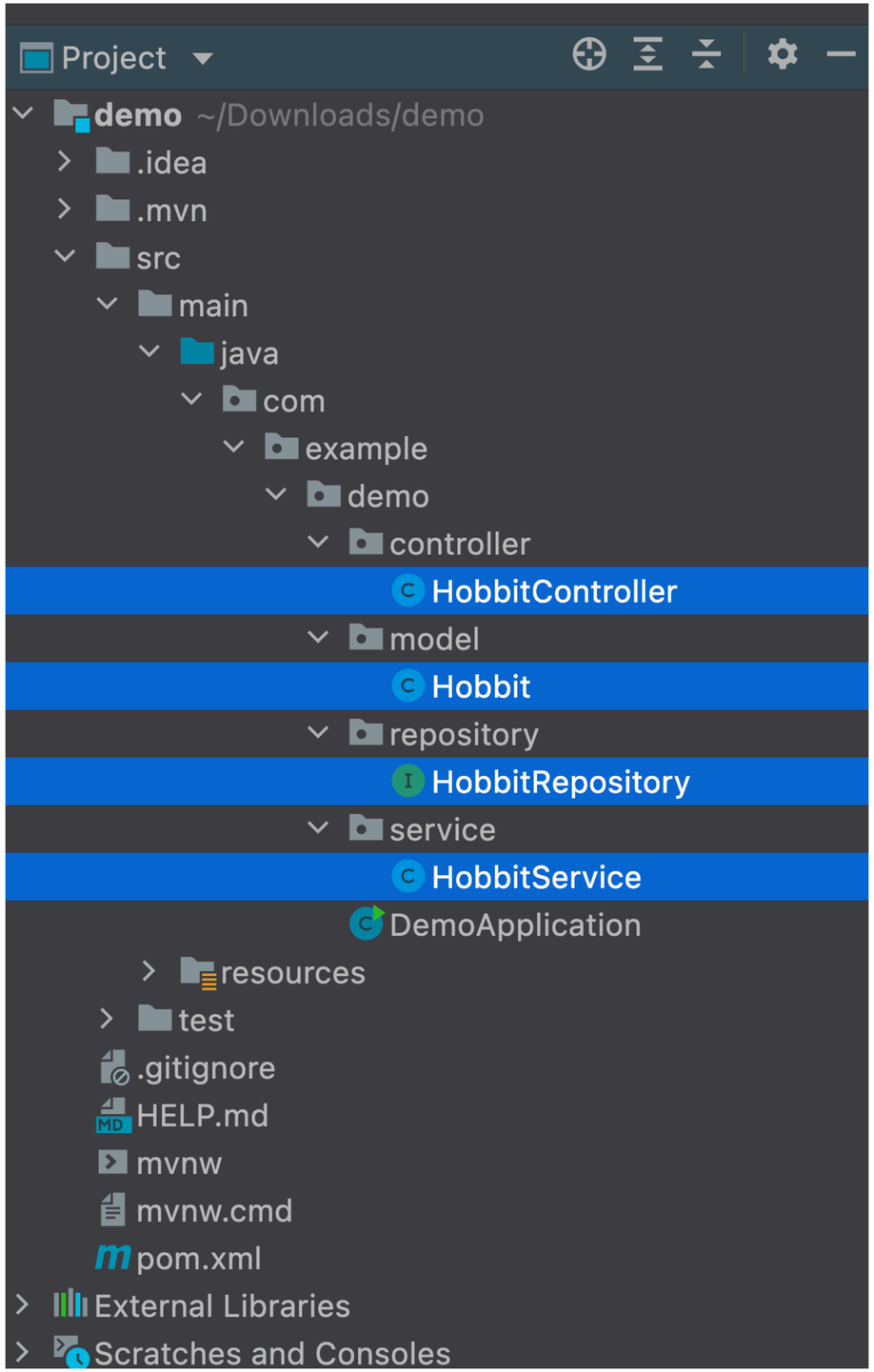
Create the application layers

Packages:
- model
- controller
- service
- repository


Project
demo ~/Downloads/demo
.idea
.mvn
src
main
java
com
example
demo
controller
model
repository
service
DemoApplication
resources
static
templates
application.properties
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles

Add some classes/interfaces (all empty)

- model.Hobbit (a class)
- controller.HobbitController (a class)
- service.HobbitService (a class)
- repository.HobbitRepository (an interface)


Project
demo ~/Downloads/demo
.idea
.mvn
src
main
java
com
example
demo
controller
HobbitController
model
Hobbit
repository
HobbitRepository
service
HobbitService
DemoApplication
resources
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles

Add some superpowers to the classes/interfaces

- Hobbit @Entity
- HobbitController @RestController
- HobbitService @Service
- HobbitRepository extends CrudRepository<T,ID>

Hobbit + (@Entity+@Id)

@Entity + @Id superpower:

-maps java class to a DB table
-maps object to a table row

```java
package com.example.demo.model;

import javax.persistence.Entity;
import javax.persistence.Id;


@Entity
public class Hobbit {
    @Id
    private Long id;

    private String name;
    private String lastName;

    public Hobbit() {
    }

    public Hobbit(Long id, String name, String lastName) {
        this.id = id;
        this.name = name;
        this.lastName = lastName;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
}
```

HobbitController + @RestController

@RestController + @GetMapping superpower:

1. This class will be managed by Spring (Spring Bean)
2. a java method in this class (with @GetMapping) can:
- be called by a browser and
- return some data in {JSON}

```java
package com.example.demo.controller;

import org.springframework.web.bind.annotation.RestController;

@RestController
public class HobbitController {
}
```

HobbitService +@Service

@Service superpower:

This class will be managed by Spring (Spring bean)

```java
package com.example.demo.service;

import org.springframework.stereotype.Service;

@Service
public class HobbitService {
}
```

HobbitRepository extends JpaRepository<Hobbit,Long>

extends JpaRepository<T,ID> superpowers:

- Easy access to a database
- ORM (Hibernate) under the hood

```java
package com.example.demo.repository;

import com.example.demo.model.Hobbit;
import org.springframework.data.jpa.repository.JpaRepository;

public interface HobbitRepository extends JpaRepository<Hobbit, Long> {
}
```

```java
@Entity
public class Hobbit {
    @Id
    private Long id;
```

HobbitService

@Autowired **1. Glue HobbitService and HobbitRepository**

2. Implement List<Hobbit> getAll() using HobbitRepository

```java
package com.example.demo.service;

import com.example.demo.model.Hobbit;
import com.example.demo.repository.HobbitRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class HobbitService {

    @Autowired
    private HobbitRepository hobbitRepository;

    public List<Hobbit> getAll(){
        return hobbitRepository.findAll();
    }
}
```

HobbitController

@Autowired **1. Glue HobbitController and HobbitService**

2. Implement List<Hobbit> getAll() using HobbitService

```java
package com.example.demo.controller;

import com.example.demo.model.Hobbit;
import com.example.demo.service.HobbitService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

import java.util.List;

@RestController
public class HobbitController {

    @Autowired
    private HobbitService hobbitService;

    @GetMapping("/hobbits")
    public List<Hobbit> getAll(){
        return hobbitService.getAll();
    }
}
```

Fine-tune H2

application.properties

```
spring.datasource.url=jdbc:h2:mem:testdb
spring.h2.console.enabled=true
```

Run the app

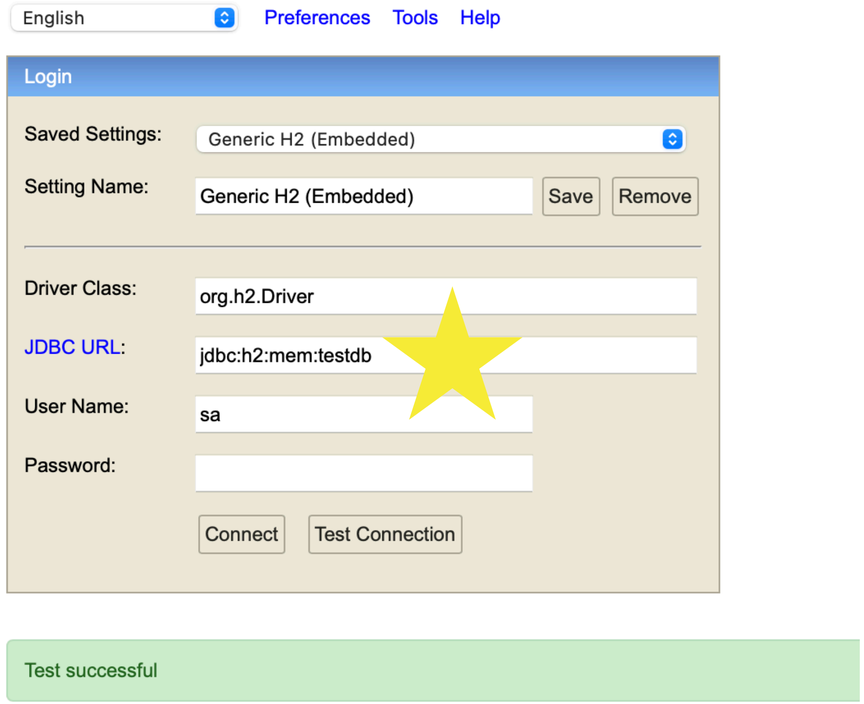
Test it with openapi-ui / H2

The H2 console

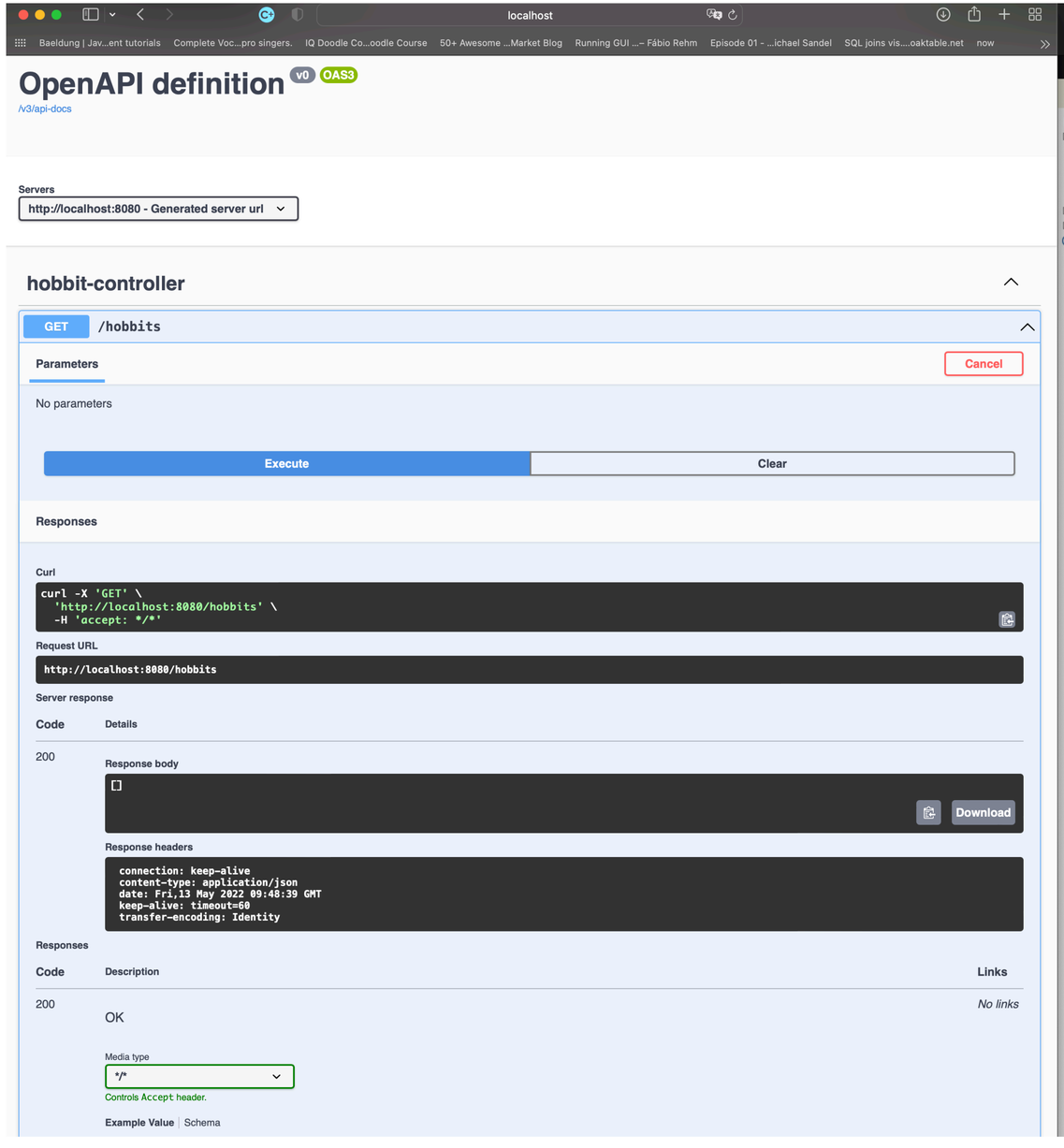
localhost:8080/h2-console

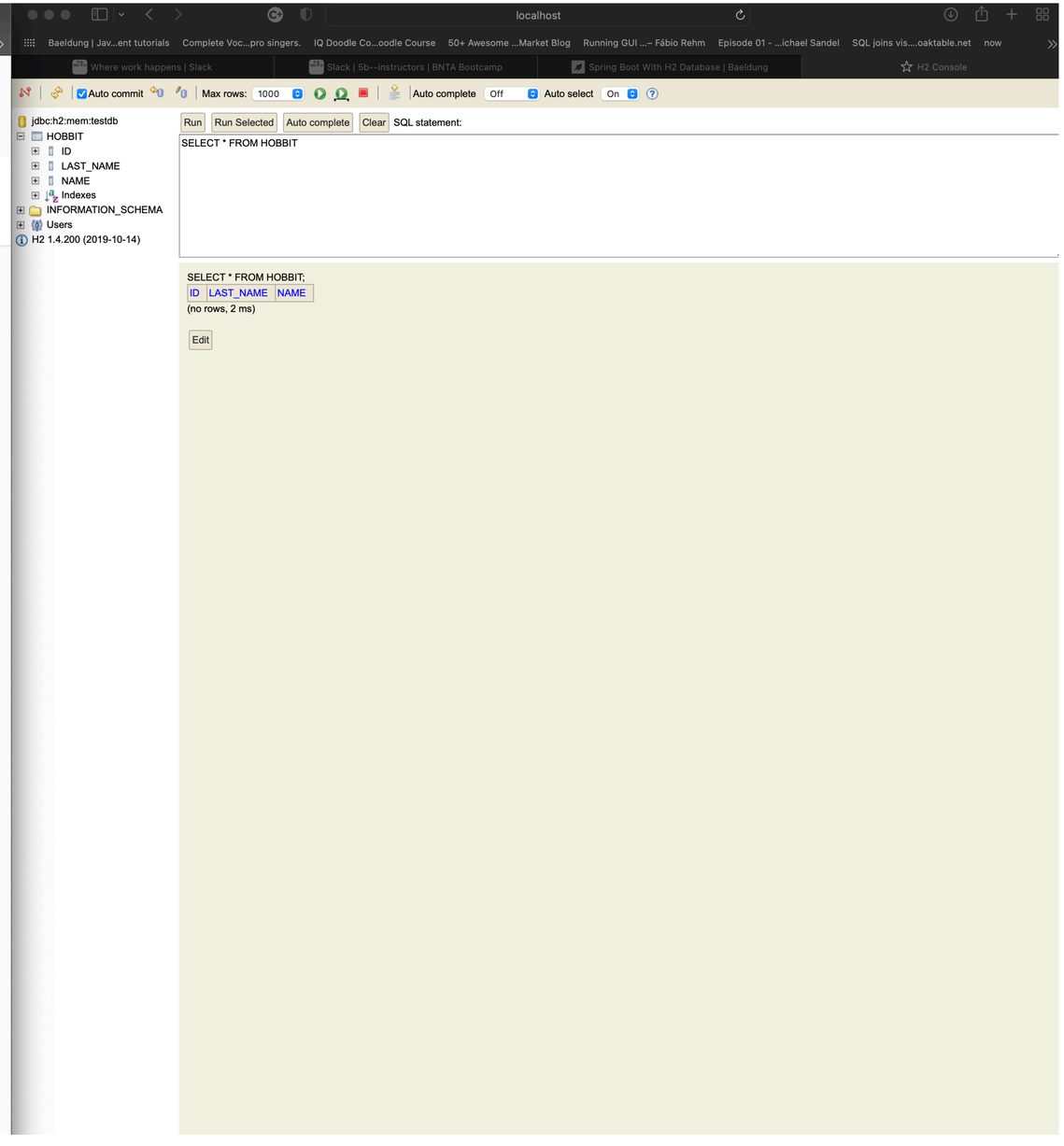
```
spring.datasource.url=jdbc:h2:mem:testdb
```

User name: sa
Password: <leave empty>

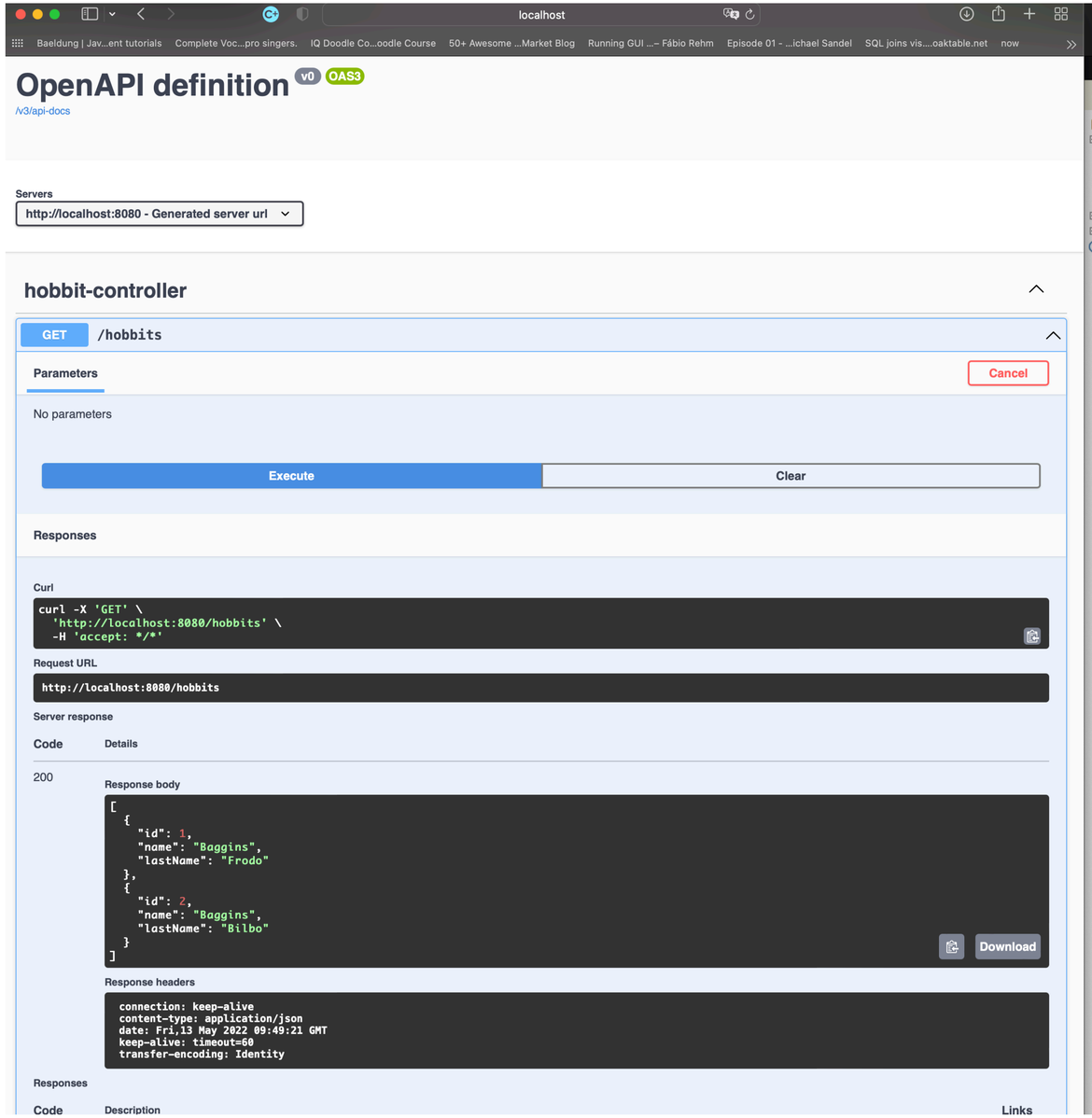


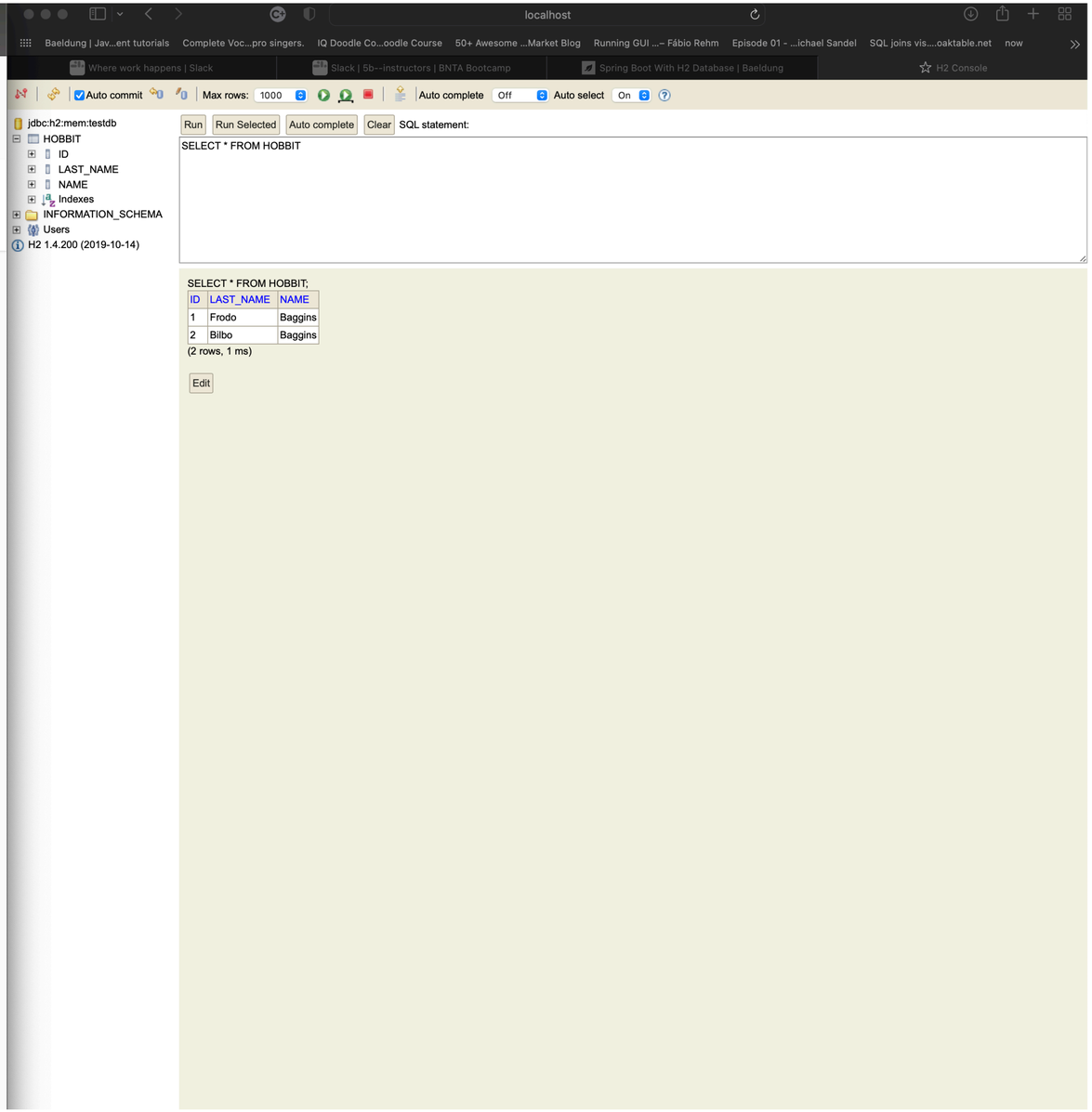
/hobbits with no hobbits





/hobbits with some hobbits





@PostMapping

Add controller, use service & repository to save a new hobbit


package com.example.demo.controller;
import com.example.demo.model.Hobbit;
import com.example.demo.service.HobbitService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;
import java.util.List;
@RestController
public class HobbitController {
@Autowired
private HobbitService hobbitService;
@GetMapping("/hobbits")
public List<Hobbit> getAll(){
return hobbitService.getAll();
}
@PostMapping("/hobbits")
public Hobbit post(@RequestBody Hobbit hobbit) {
return hobbitService.save(hobbit);
}
}
package com.example.demo.service;
import com.example.demo.model.Hobbit;
import com.example.demo.repository.HobbitRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.List;
@Service
public class HobbitService {
@Autowired
private HobbitRepository hobbitRepository;
public List<Hobbit> getAll(){
return hobbitRepository.findAll();
}
public Hobbit save(Hobbit hobbit) {
return hobbitRepository.save(hobbit);
}
}
package com.example.demo.repository;
import com.example.demo.model.Hobbit;
import org.springframework.data.jpa.repository.JpaRepository;
public interface HobbitRepository extends JpaRepository<Hobbit, Long> {
}

Is there a way to manually generate the ids?

POST /hobbits

We need to manually add an id or we'll see this in Spring's logs

Parameters

Cancel Reset

No parameters

Request body required

application/json

```
{
  "id": 150,
  "name": "string",
  "lastName": "string"
}
```

```
org.hibernate.id.IdentifierGenerationException Create breakpoint : ids for this class must be manually assigned before calling save(): com.example.demo.model.Hobbit
    at org.hibernate.id.Assigned.generate(Assigned.java:33) ~[hibernate-core-5.6.8.Final.jar:5.6.8.Final]
    at org.hibernate.event.internal.AbstractSaveEventListener.saveWithGeneratedId(AbstractSaveEventListener.java:115) ~[hibernate-core-5.6.8.Final.jar:5.6.8.Final]
    at org.hibernate.event.internal.DefaultPersistEventListener.entityIsTransient(DefaultPersistEventListener.java:185) ~[hibernate-core-5.6.8.Final.jar:5.6.8.Final]
    at org.hibernate.event.internal.DefaultPersistEventListener.onPersist(DefaultPersistEventListener.java:128) ~[hibernate-core-5.6.8.Final.jar:5.6.8.Final]
    at org.hibernate.event.internal.DefaultPersistEventListener.onPersist(DefaultPersistEventListener.java:55) ~[hibernate-core-5.6.8.Final.jar:5.6.8.Final]
    at org.hibernate.event.service.internal.EventListenerGroupImpl.fireEventOnEachListener(EventListenerGroupImpl.java:107) ~[hibernate-core-5.6.8.Final.jar:5.6.8.Final]
    at org.hibernate.internal.SessionImpl.firePersist(SessionImpl.java:756) ~[hibernate-core-5.6.8.Final.jar:5.6.8.Final]
    at org.hibernate.internal.SessionImpl.persist(SessionImpl.java:742) ~[hibernate-core-5.6.8.Final.jar:5.6.8.Final] <2 internal lines>
    at org.springframework.orm.jpa.ExtendedEntityManagerCreator$ExtendedEntityManagerInvocationHandler.invoke(ExtendedEntityManagerCreator.java:362) ~[spring-orm-5.3.19.jar:5.3.19]
    at jdk.proxy2/jdk.proxy2.$Proxy93.persist(Unknown Source) ~[na:na] <2 internal lines>
    at org.springframework.orm.jpa.SharedEntityManagerCreator$SharedEntityManagerInvocationHandler.invoke(SharedEntityManagerCreator.java:311) ~[spring-orm-5.3.19.jar:5.3.19]
    at jdk.proxy2/jdk.proxy2.$Proxy93.persist(Unknown Source) ~[na:na]
    at org.springframework.data.jpa.repository.support.SimpleJpaRepository.save(SimpleJpaRepository.java:647) ~[spring-data-jpa-2.6.4.jar:2.6.4] <2 internal lines>
```

Execute Clear

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:8080/hobbits' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 150,
  "name": "string",
  "lastName": "string"
}'
```

Request URL

```
http://localhost:8080/hobbits
```

Server response

Code	Details
200	Response body

```
{
  "id": 150,
  "name": "string",
  "lastName": "string"
}
```

Download

Response headers

```
connection: keep-alive
content-type: application/json
date: Sun,15 May 2022 19:16:18 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Responses

Code	Description	Links

```java
@Entity
public class Hobbit {
    @Id
// The database engine will automatically put the next available value for you.
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;
    private String lastName;

    // constructors / getters & setters
}
```

POST /hobbits

Parameters

Cancel Reset

No parameters

Request body required

application/json

```
{
  "id": null,
  "name": "string",
  "lastName": "string"
}
```

The next available id was put for us.

Execute Clear

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:8080/hobbits' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": null,
  "name": "string",
  "lastName": "string"
}'
```

Request URL

```
http://localhost:8080/hobbits
```

Server response

Code	Details
200	Response body

```
{
  "id": 1,
  "name": "string",
  "lastName": "string"
}
```

Download

Response headers

```
connection: keep-alive
content-type: application/json
date: Sun,15 May 2022 19:27:54 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Responses

Code	Description	Links