Flyway, açık kaynaklı database migration'ları yapmamızı sağlayan bir tool'dur.
Sadeliği ve konfigürasyondan bağımsız konvensiyonel bir mekanizmayı ön plana çıkarmayı amaçlar.

Flyway temelde 6 adet komut - *migrate, info, validate, repair, clean* - üzerine kurulmuş bir mekanizmaya sahiptir ve bu komutlar birçok araç ile entegre şekilde çalışabildiğinden farklı platformlarda hem uygulanabilirlik hem de tümleşiklik anlamında geliştiriciler tarafından tercih edilmektedir.
Değişiklik kümelerini dosyalar şeklinde yönetir.
Ham SQL formatlarında değişiklik kümelerini çalıştırabildiği gibi, Java tabanlı değişiklik kümelerini de desteklemektedir.

https://flywaydb.org/documentation/

flywaydb.org/documentation/concepts/migrations#naming-1

- Advanced bulk data changes (Recalculations, advanced format changes, ...)

Naming §

In order to be picked up by Flyway, Java-based Migrations must implement the `JavaMigration` interface. Most users however should inherit from the convenience class `BaseJavaMigration` instead as it encourages Flyway's default naming convention, enabling Flyway to automatically extract the version and the description from the class name. To be able to do so, the class name must comply with the following naming pattern:

Versioned Migrations

```
Prefix  Separator
   ↓    ↓
   V 2  __  Add_new_table
     ↑      ↑
  Version  Description
```

Undo Migrations

```
Prefix  Separator
   ↓    ↓
   U 2  __  Add_new_table
     ↑      ↑
  Version  Description
```

Repeatable Migrations

```
Prefix Separator
   ↓   ↓
   R   __  Add_new_table
           ↑
       Description
```

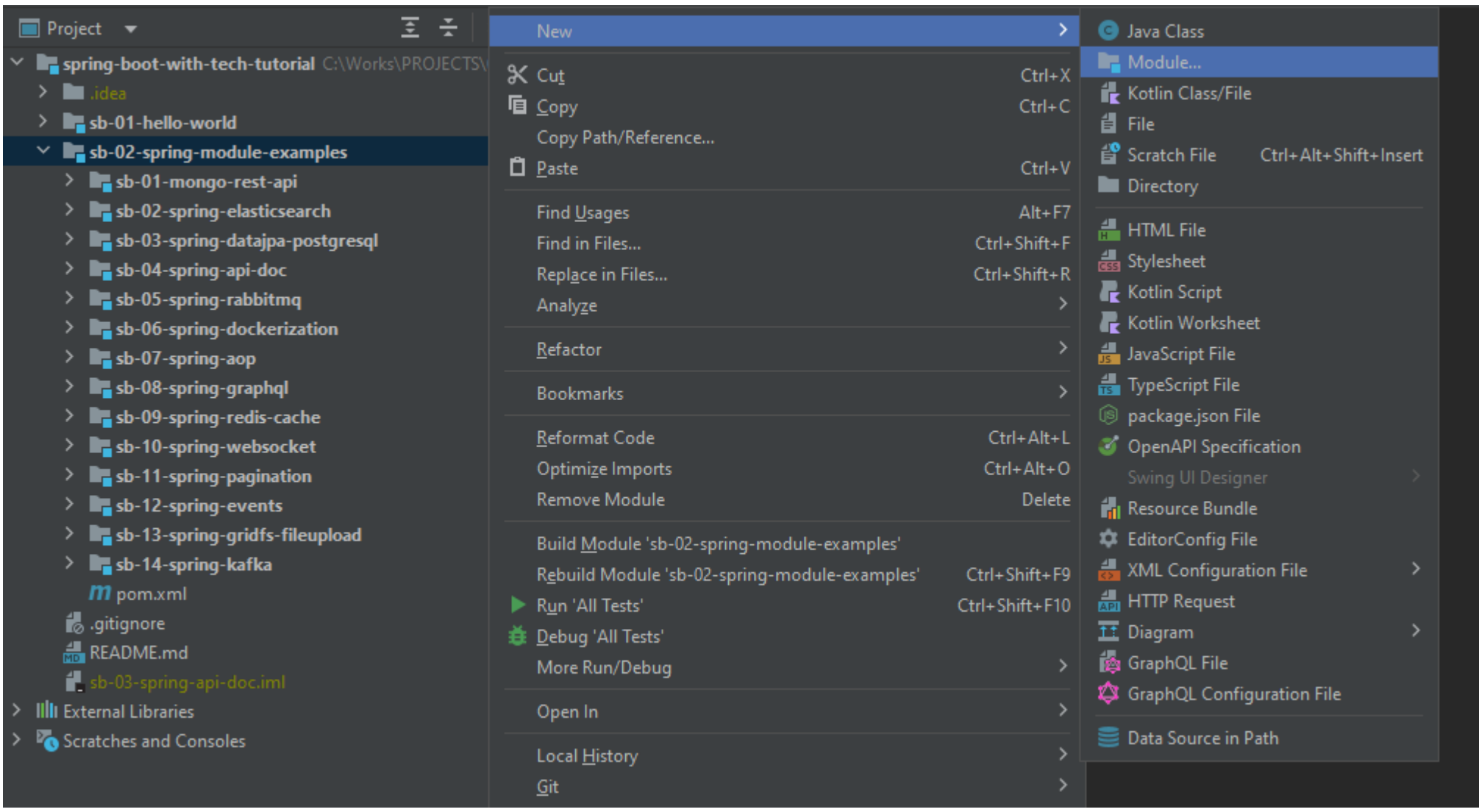
The class name consists of the following parts:

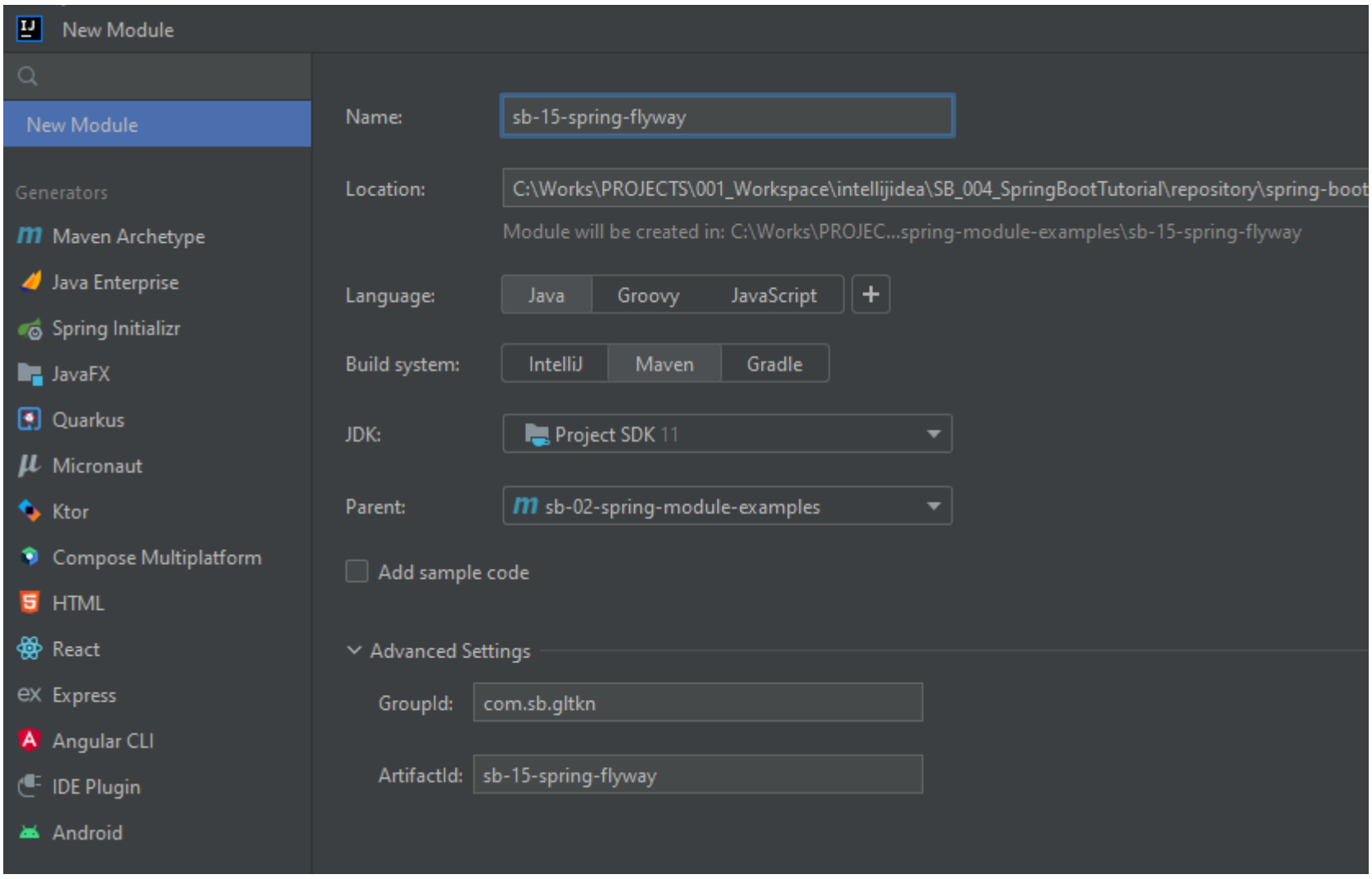
- **Prefix**: `V` for versioned migrations, `U` for undo migrations, `R` for repeatable migrations
- **Version**: Underscores (automatically replaced by dots at runtime) separate as many parts as you like (Not for repeatable migrations)
- **Separator**: `__` (two underscores)
- **Description**: Underscores (automatically replaced by spaces at runtime) separate the words

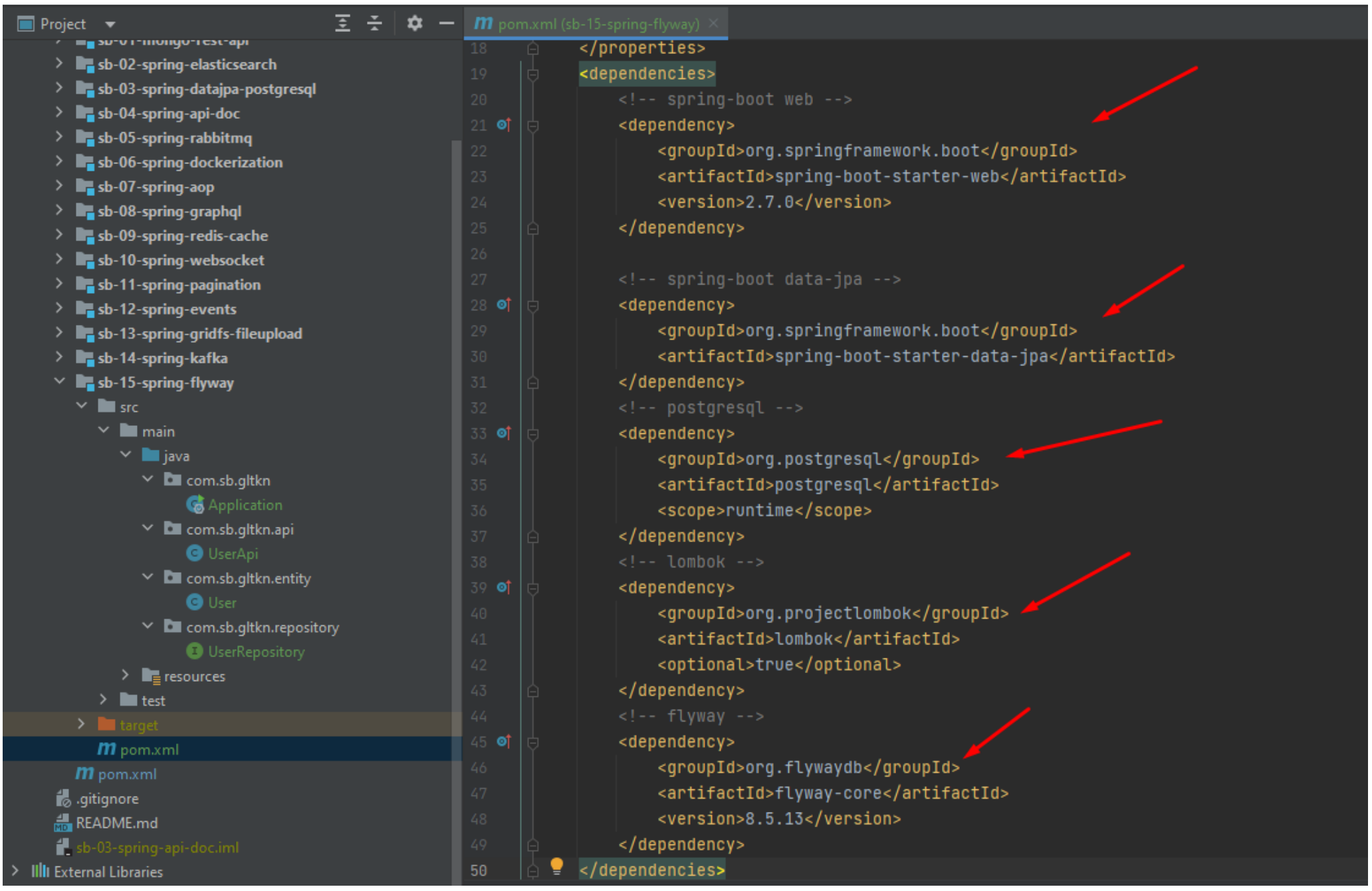
If you need more control over the class name, you can override the default convention by implementing the `JavaMigration` interface directly.

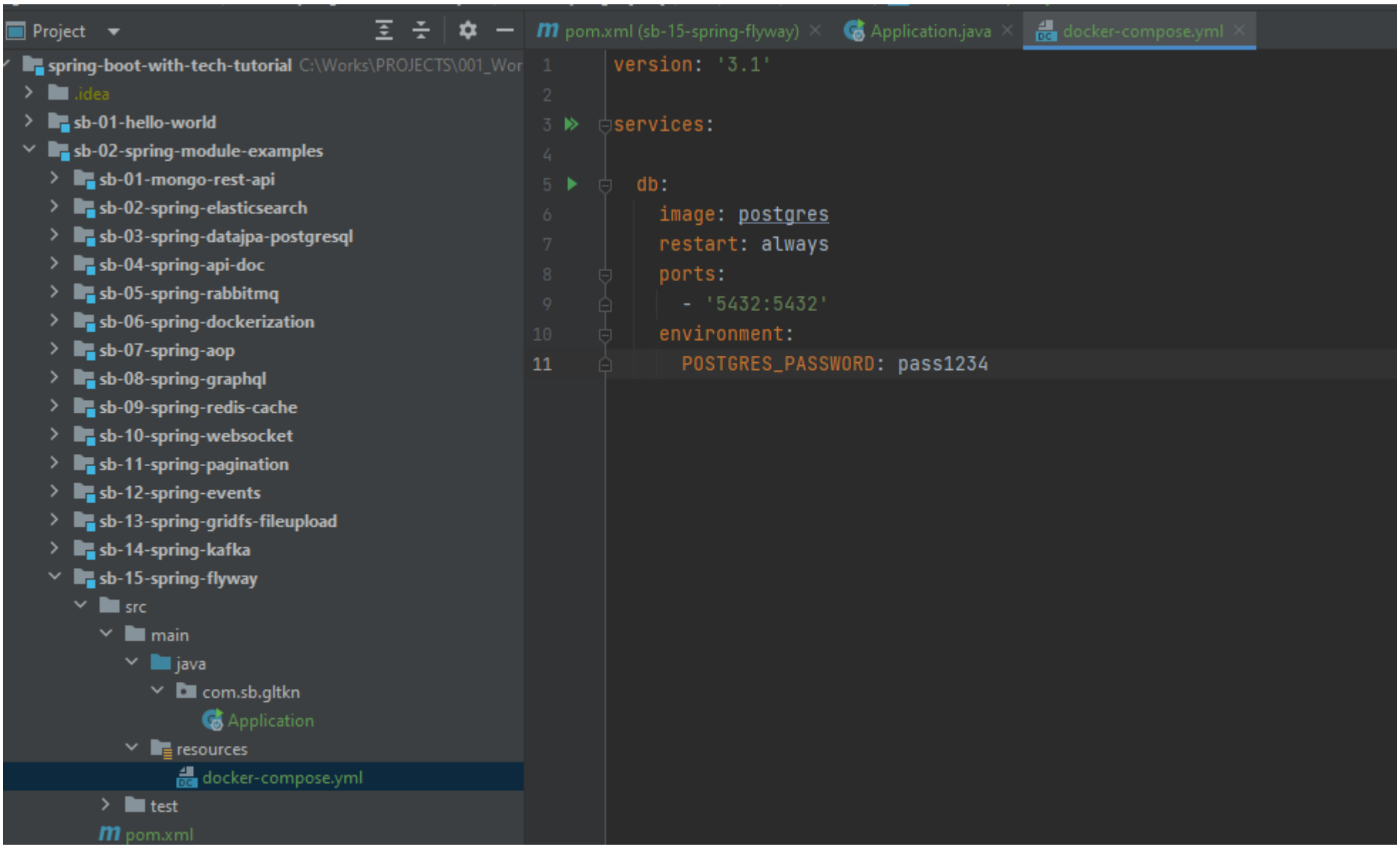
This will allow you to name your class as you wish. Version, description and migration category are provided by implementing the respective methods.

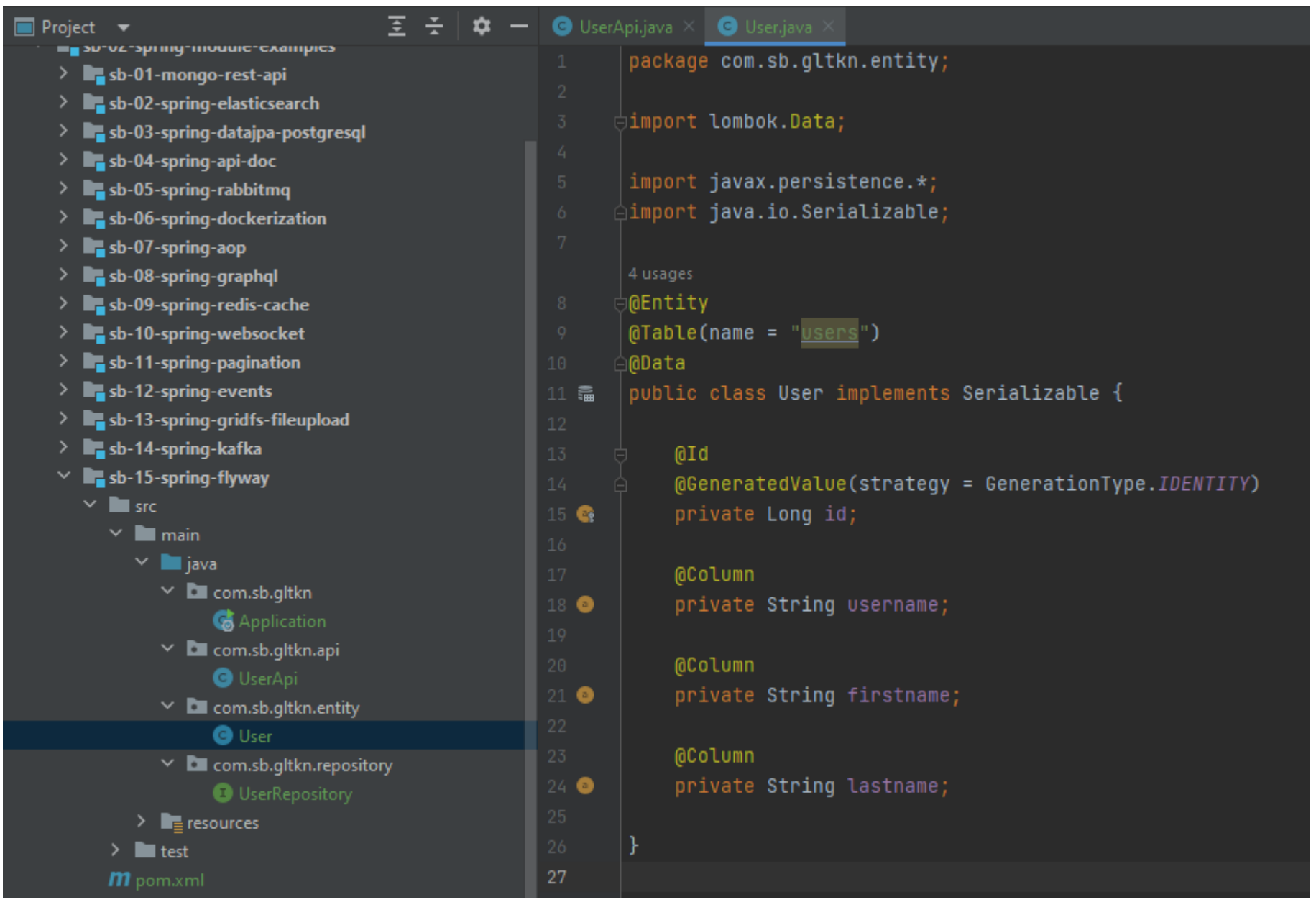
Veritabanına uygulanmamış bir değişiklik varsa bu script veritabanına flyway ile migrate ediliyor.



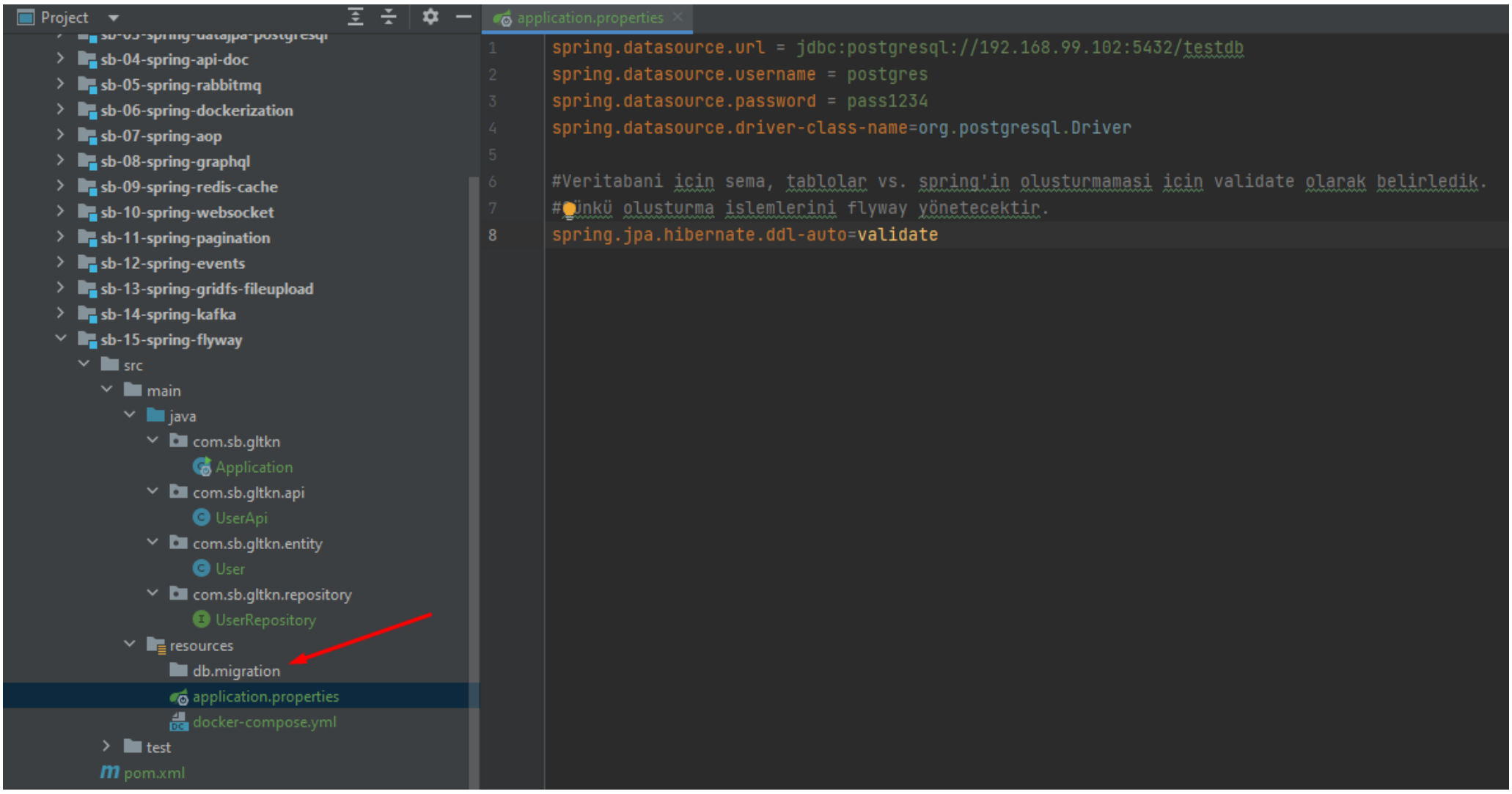

New Module
New Module
Generators
Maven Archetype
Java Enterprise
Spring Initializr
JavaFX
Quarkus
Micronaut
Ktor
Compose Multiplatform
HTML
React
Express
Angular CLI
IDE Plugin
Android
Name: sb-15-spring-flyway
Location: C:\Works\PROJECTS\001_Workspace\intellijidea\SB_004_SpringBootTutorial\repository\spring-boot
Module will be created in: C:\Works\PROJEC...spring-module-examples\sb-15-spring-flyway
Language: Java Groovy JavaScript
Build system: IntelliJ Maven Gradle
JDK: Project SDK 11
Parent: sb-02-spring-module-examples
Add sample code
Advanced Settings
GroupId: com.sb.gltkn
ArtifactId: sb-15-spring-flyway

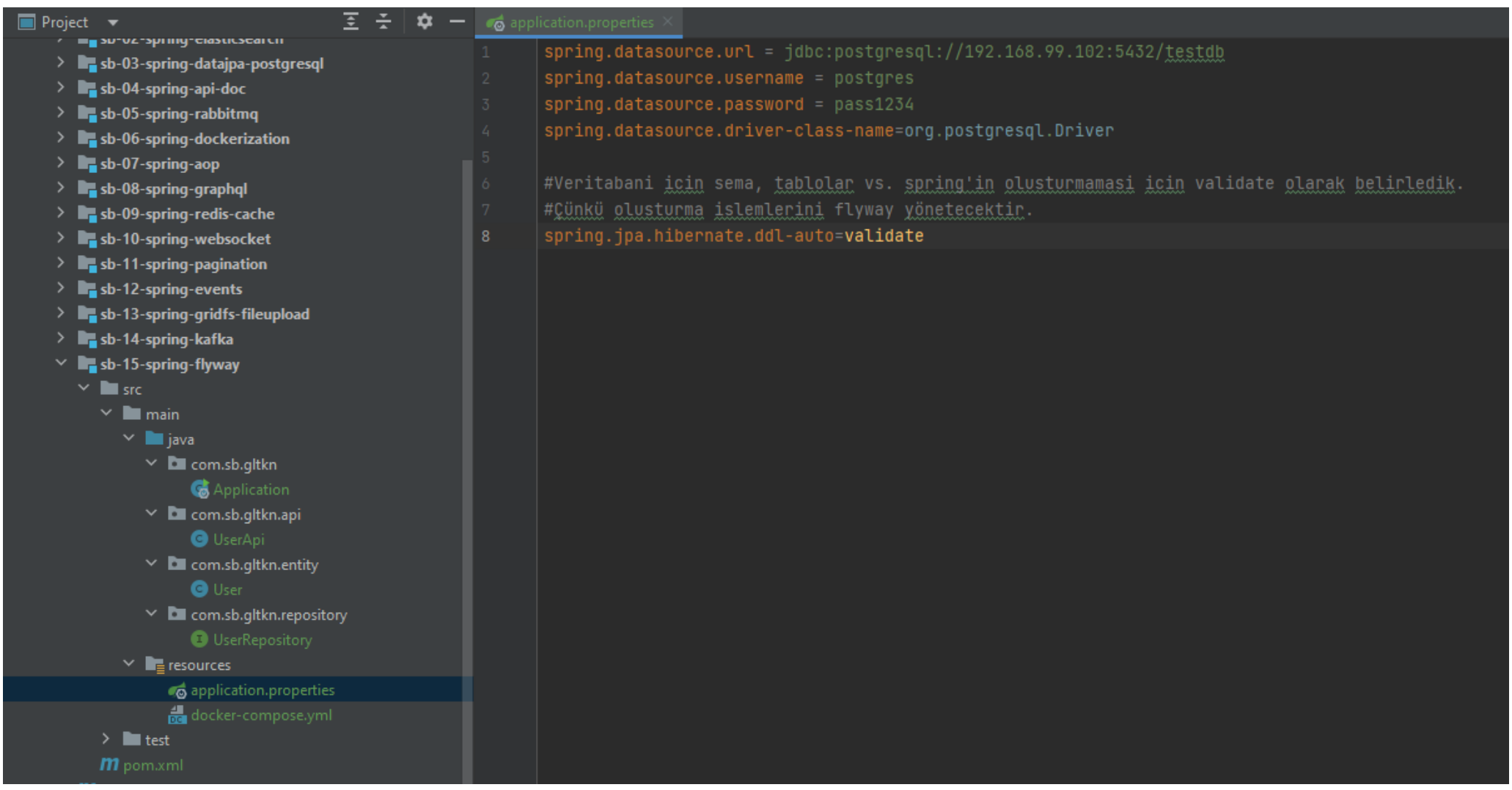

Project
sb-02-spring-elasticsearch
sb-03-spring-datajpa-postgresql
sb-04-spring-api-doc
sb-05-spring-rabbitmq
sb-06-spring-dockerization
sb-07-spring-aop
sb-08-spring-graphql
sb-09-spring-redis-cache
sb-10-spring-websocket
sb-11-spring-pagination
sb-12-spring-events
sb-13-spring-gridfs-fileupload
sb-14-spring-kafka
sb-15-spring-flyway
src
main
java
com.sb.gltkn
Application
com.sb.gltkn.api
UserApi
com.sb.gltkn.entity
User
com.sb.gltkn.repository
UserRepository
resources
test
target
pom.xml
pom.xml
.gitignore
README.md
sb-03-spring-api-doc.iml
External Libraries
pom.xml (sb-15-spring-flyway)
</properties>
<dependencies>
<!-- spring-boot web -->
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
<version>2.7.0</version>
</dependency>
<!-- spring-boot data-jpa -->
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<!-- postgresql -->
<dependency>
<groupId>org.postgresql</groupId>
<artifactId>postgresql</artifactId>
<scope>runtime</scope>
</dependency>
<!-- lombok -->
<dependency>
<groupId>org.projectlombok</groupId>
<artifactId>lombok</artifactId>
<optional>true</optional>
</dependency>
<!-- flyway -->
<dependency>
<groupId>org.flywaydb</groupId>
<artifactId>flyway-core</artifactId>
<version>8.5.13</version>
</dependency>
</dependencies>

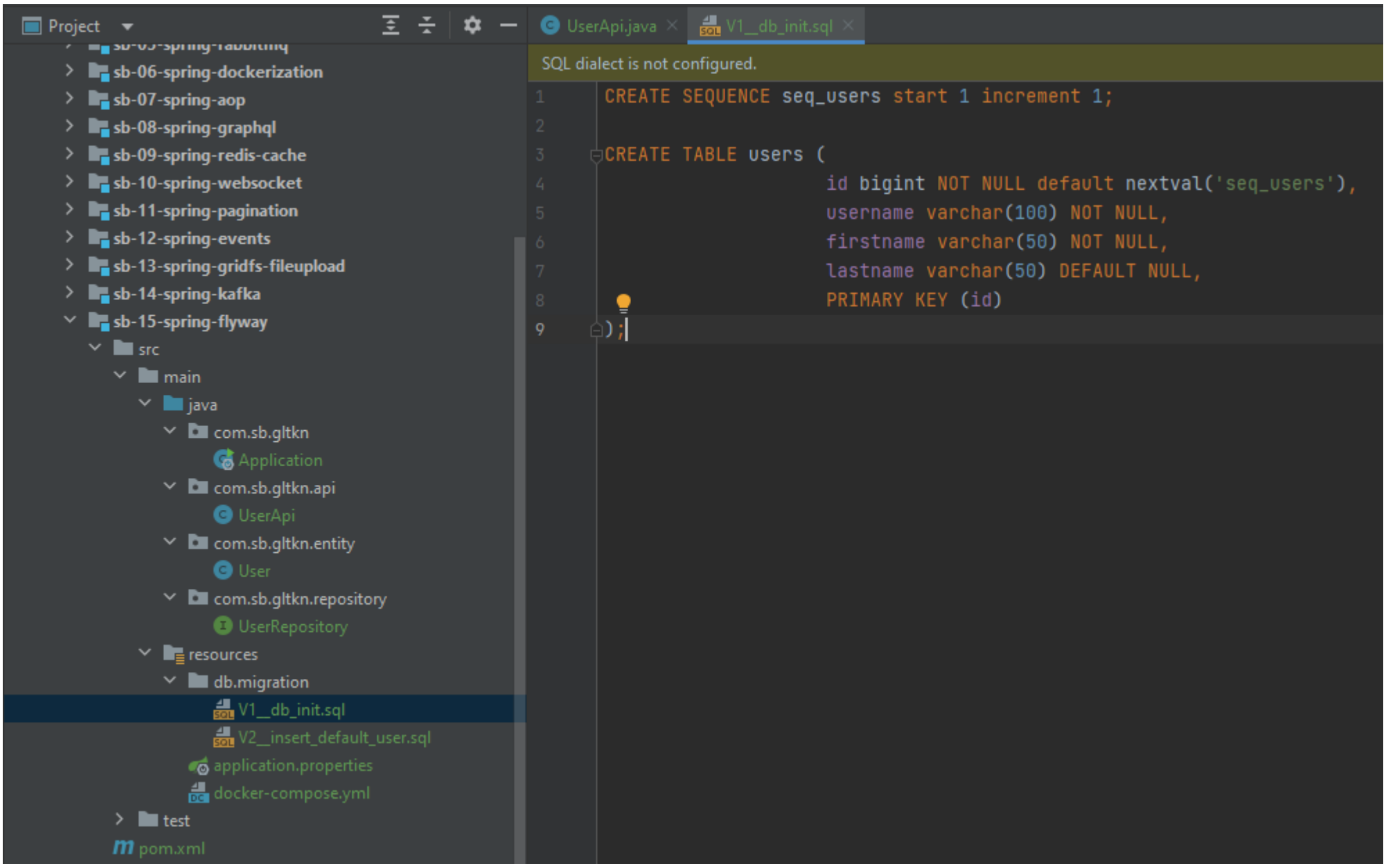

Project
spring-boot-with-tech-tutorial C:\Works\PROJECTS\001_Wor
.idea
sb-01-hello-world
sb-02-spring-module-examples
sb-01-mongo-rest-api
sb-02-spring-elasticsearch
sb-03-spring-datajpa-postgresql
sb-04-spring-api-doc
sb-05-spring-rabbitmq
sb-06-spring-dockerization
sb-07-spring-aop
sb-08-spring-graphql
sb-09-spring-redis-cache
sb-10-spring-websocket
sb-11-spring-pagination
sb-12-spring-events
sb-13-spring-gridfs-fileupload
sb-14-spring-kafka
sb-15-spring-flyway
src
main
java
com.sb.gltkn
Application
resources
docker-compose.yml
test
pom.xml
pom.xml (sb-15-spring-flyway)
Application.java
docker-compose.yml
version: '3.1'
services:
db:
image: postgres
restart: always
ports:
- '5432:5432'
environment:
POSTGRES_PASSWORD: pass1234

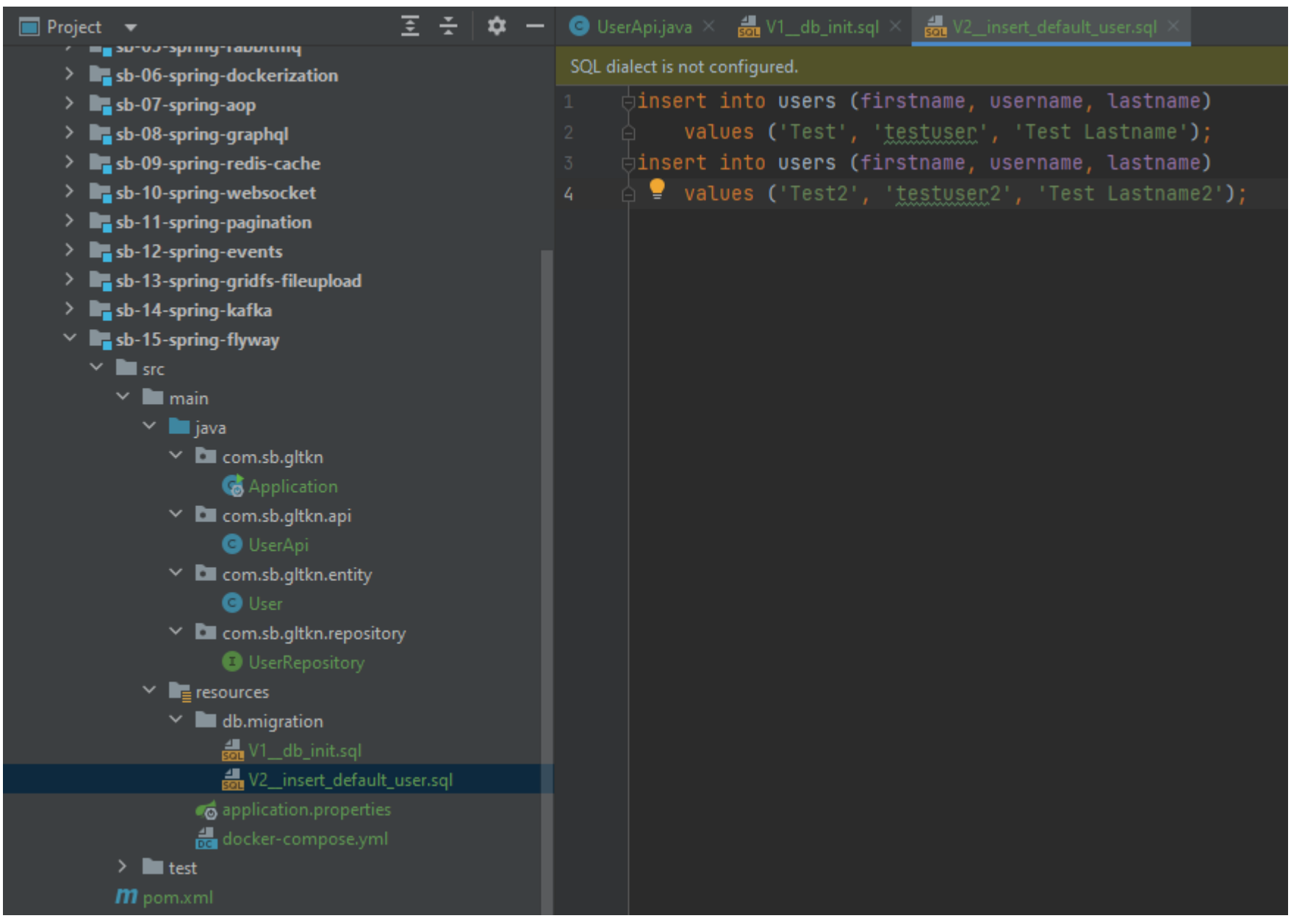

Project
sb-01-mongo-rest-api
sb-02-spring-elasticsearch
sb-03-spring-datajpa-postgresql
sb-04-spring-api-doc
sb-05-spring-rabbitmq
sb-06-spring-dockerization
sb-07-spring-aop
sb-08-spring-graphql
sb-09-spring-redis-cache
sb-10-spring-websocket
sb-11-spring-pagination
sb-12-spring-events
sb-13-spring-gridfs-fileupload
sb-14-spring-kafka
sb-15-spring-flyway
src
main
java
com.sb.gltkn
Application
com.sb.gltkn.api
UserApi
com.sb.gltkn.entity
User
com.sb.gltkn.repository
UserRepository
resources
test
pom.xml
UserApi.java
User.java
package com.sb.gltkn.entity;
import lombok.Data;
import javax.persistence.*;
import java.io.Serializable;
4 usages
@Entity
@Table(name = "users")
@Data
public class User implements Serializable {
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
@Column
private String username;
@Column
private String firstname;
@Column
private String lastname;
}

Flyway default olarak resources altında **db/migration** dizinin olmasını beklemektedir.






Project
sb-06-spring-dockerization
sb-07-spring-aop
sb-08-spring-graphql
sb-09-spring-redis-cache
sb-10-spring-websocket
sb-11-spring-pagination
sb-12-spring-events
sb-13-spring-gridfs-fileupload
sb-14-spring-kafka
sb-15-spring-flyway
src
main
java
com.sb.gltkn
Application
com.sb.gltkn.api
UserApi
com.sb.gltkn.entity
User
com.sb.gltkn.repository
UserRepository
resources
db.migration
V1__db_init.sql
V2__insert_default_user.sql
application.properties
docker-compose.yml
test
pom.xml
UserApi.java
V1__db_init.sql
SQL dialect is not configured.
CREATE SEQUENCE seq_users start 1 increment 1;
CREATE TABLE users (
id bigint NOT NULL default nextval('seq_users'),
username varchar(100) NOT NULL,
firstname varchar(50) NOT NULL,
lastname varchar(50) DEFAULT NULL,
PRIMARY KEY (id)
);


Project
sb-06-spring-dockerization
sb-07-spring-aop
sb-08-spring-graphql
sb-09-spring-redis-cache
sb-10-spring-websocket
sb-11-spring-pagination
sb-12-spring-events
sb-13-spring-gridfs-fileupload
sb-14-spring-kafka
sb-15-spring-flyway
src
main
java
com.sb.gltkn
Application
com.sb.gltkn.api
UserApi
com.sb.gltkn.entity
User
com.sb.gltkn.repository
UserRepository
resources
db.migration
V1__db_init.sql
V2__insert_default_user.sql
application.properties
docker-compose.yml
test
pom.xml
UserApi.java
V1__db_init.sql
V2__insert_default_user.sql
SQL dialect is not configured.
insert into users (firstname, username, lastname)
values ('Test', 'testuser', 'Test Lastname');
insert into users (firstname, username, lastname)
values ('Test2', 'testuser2', 'Test Lastname2');

Şimdi docker-compose'u çalıştıralım.

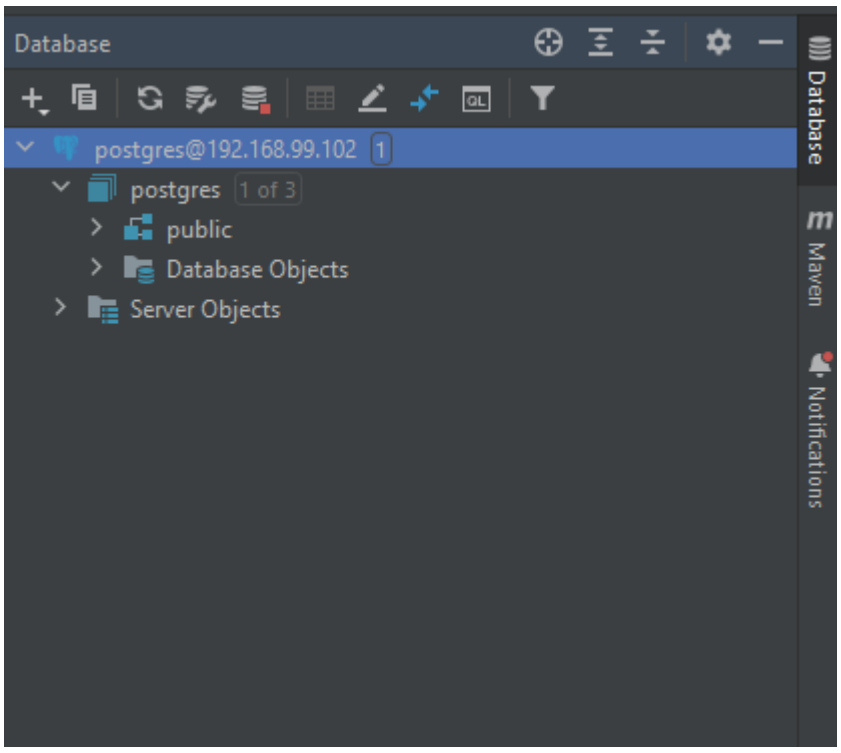
```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker-machine start default
Starting "default"...
(default) Check network to re-create if needed...
(default) Windows might ask for the permission to configure a dhcp server. Sometimes, such confirmation window is minimized in the taskbar.
(default) Waiting for an IP...
Machine "default" was started.
Waiting for SSH to be available...
Detecting the provisioner...
Started machines may have new IP addresses. You may need to re-run the `docker-machine env` command.
```

```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker-machine ls
NAME      ACTIVE   DRIVER       STATE     URL                         SWARM   DOCKER      ERRORS

EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker-machine env default
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.99.102:2376"
export DOCKER_CERT_PATH="C:\Users\EmreGltkn\.docker\machine\machines\default"
export DOCKER_MACHINE_NAME="default"
export COMPOSE_CONVERT_WINDOWS_PATHS="true"
# Run this command to configure your shell:
# eval $("C:\Users\EmreGltkn\bin\docker-machine.exe" env default)
```

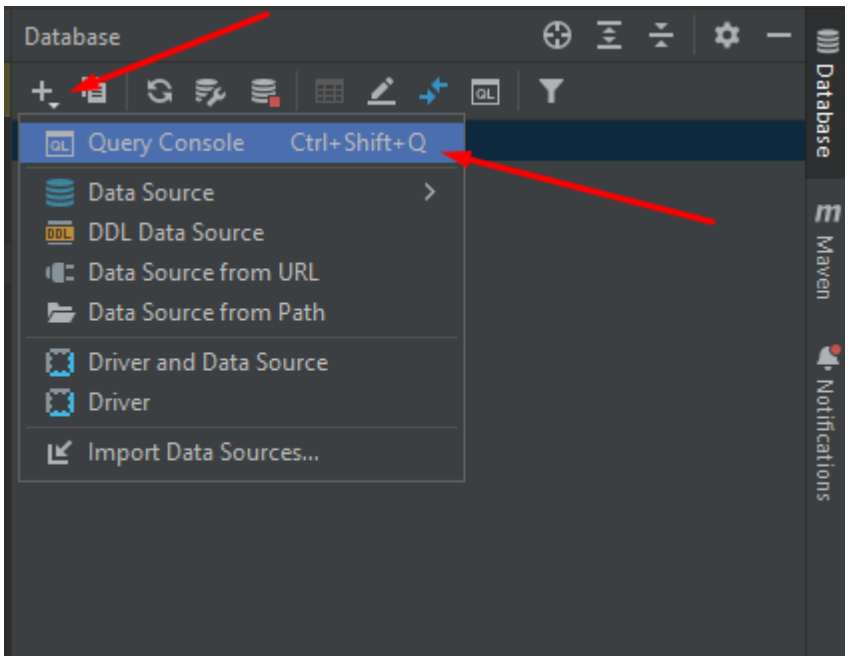
```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ eval "$(docker-machine env default)"

EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker-compose -f /c/Works/PROJECTS/001_Workspace/intellijidea/SB_004_SpringBootTutorial/repository/spring-boot-with-tech-tutorial/sb-02-spring-module-examples/sb-15-spring-flyway/src/ma
in/resources/docker-compose.yml up -d
Found orphan containers (gltkn-kafka) for this project. If you removed or renamed this service in your compose file, you can run this command with the --remove-orphans flag to clean it up.
Creating resources_db_1 ...
Creating resources_db_1 ... done
```

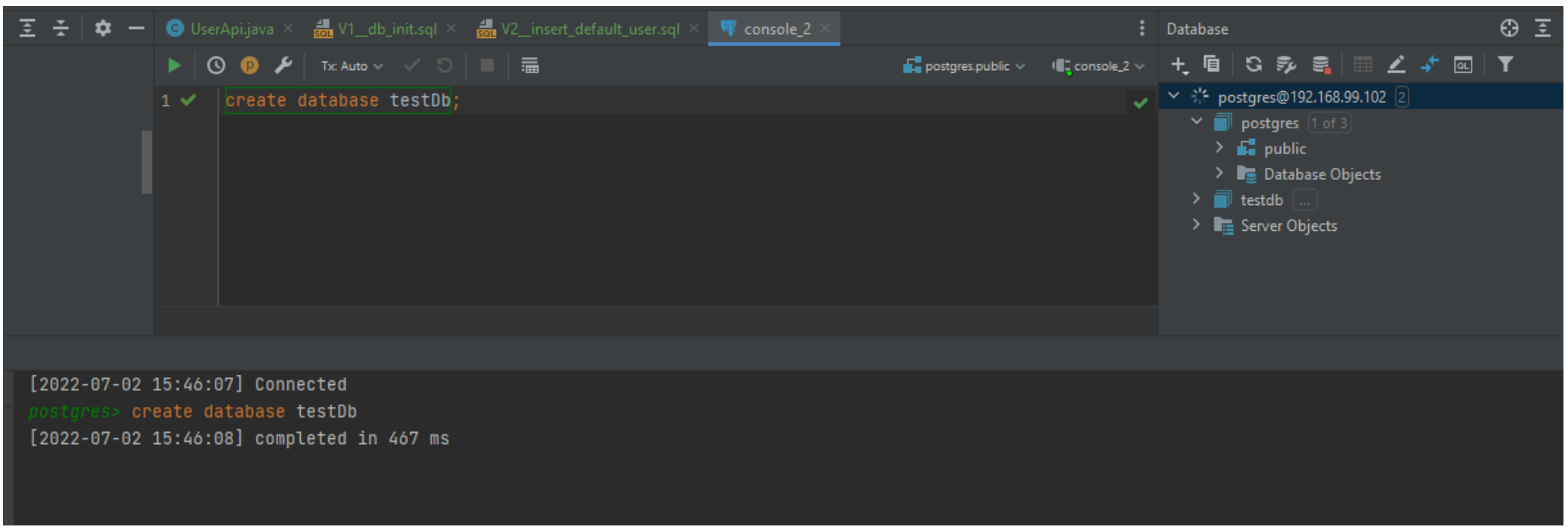
docker-compose -f /c/Works/PROJECTS/001_Workspace/intellijidea/SB_004_SpringBootTutorial/repository/spring-boot-with-tech-tutorial/sb-02-spring-module-examples/sb-15-spring-flyway/src/main/resources/docker-compose.yml up -d

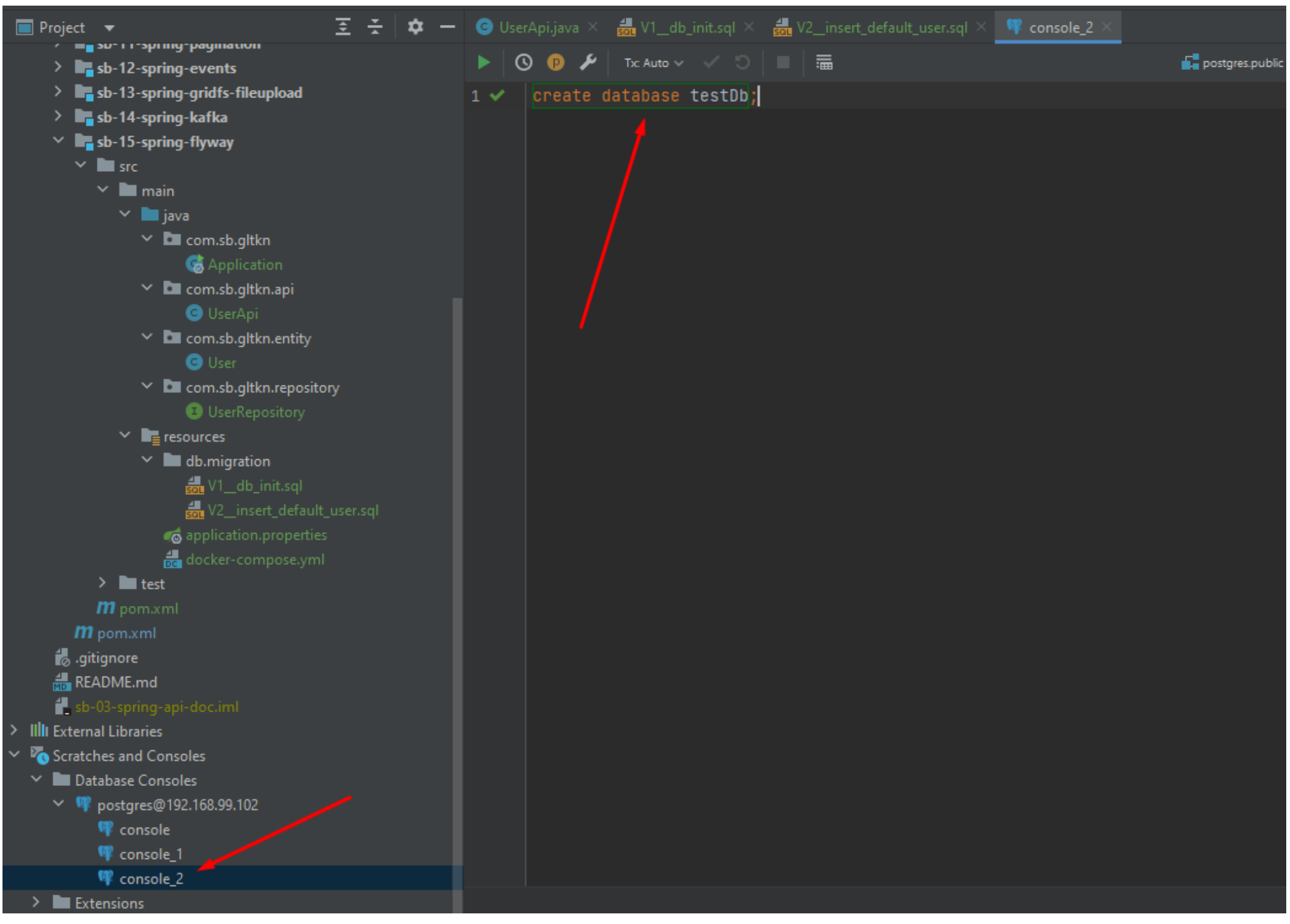
```
EmreGltkn@DESKTOP-DN9PH1A MINGW64 ~
$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED              STATUS              PORTS                    NAMES
91917e52bc12   postgres   "docker-entrypoint.s…"   About a minute ago   Up About a minute   0.0.0.0:5432->5432/tcp   resources_db_1
```



Şimdi database'imizi oluşturalım.



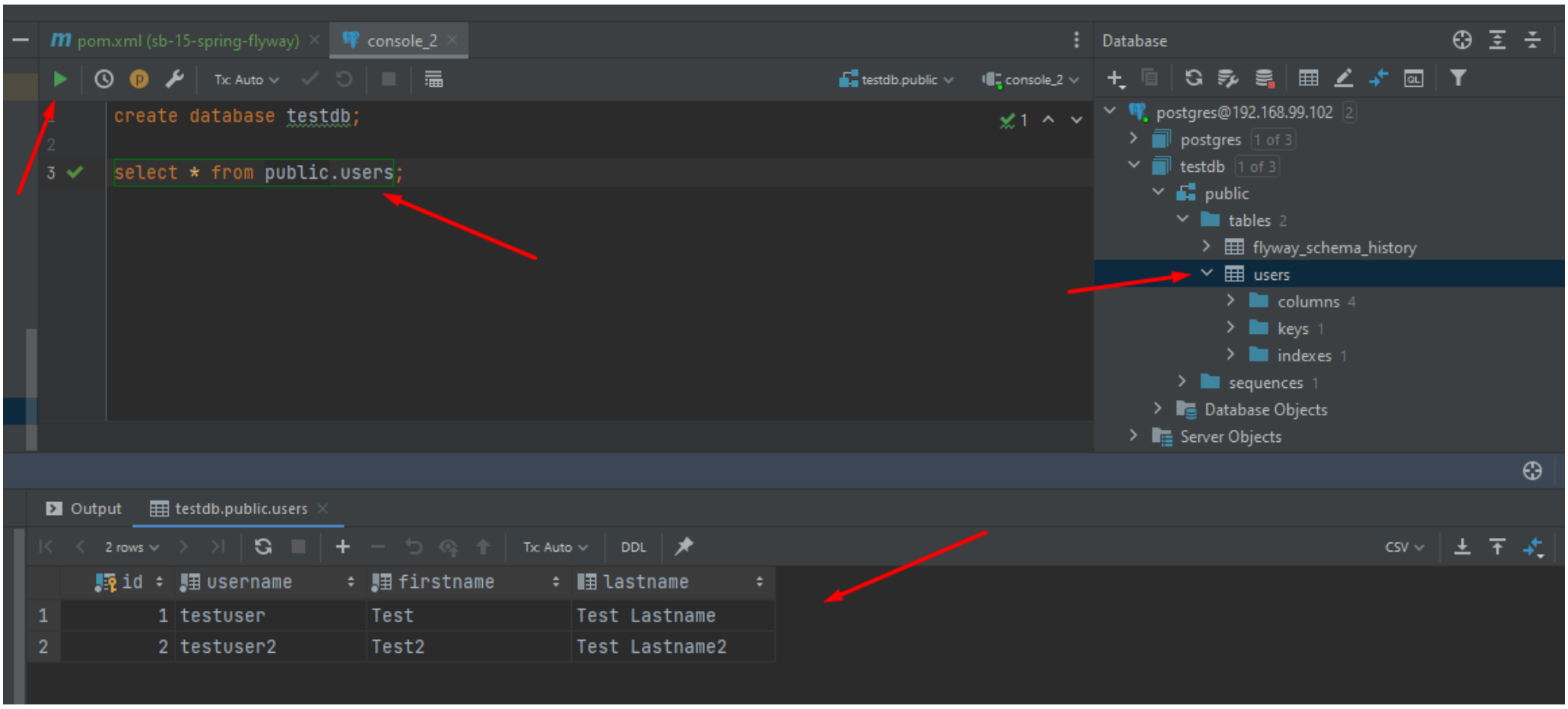




Şimdi spring boot uygulamamızı başlatalım.

Console | Actuator

```
main] com.sb.gltkn.Application               : Starting Application using Java 11.0.15 on DESKTOP-DN9PH1A with PID 17508 (C:\Works\PROJECTS\001_Workspace\
main] com.sb.gltkn.Application               : No active profile set, falling back to 1 default profile: "default"
main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 45 ms. Found 1 JPA repository interfaces.
main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.63]
main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1592 ms
main] o.f.c.internal.license.VersionPrinter    : Flyway Community Edition 8.5.13 by Redgate
main] o.f.c.internal.license.VersionPrinter    : See what's new here: https://flywaydb.org/documentation/learnmore/releaseNotes#8.5.13
main] o.f.c.internal.license.VersionPrinter    :
main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
main] o.f.c.i.database.base.BaseDatabaseType   : Database: jdbc:postgresql://192.168.99.102:5432/testdb (PostgreSQL 14.3)
main] o.f.core.internal.command.DbValidate     : Successfully validated 2 migrations (execution time 00:00.027s)
main] o.f.core.internal.command.DbMigrate      : Current version of schema "public": << Empty Schema >>
main] o.f.core.internal.command.DbMigrate      : Migrating schema "public" to version "1 - db init"
main] o.f.core.internal.command.DbMigrate      : Migrating schema "public" to version "2 - insert default user"
main] o.f.core.internal.command.DbMigrate      : Successfully applied 2 migrations to schema "public", now at version v2 (execution time 00:00.618s)
main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.6.9.Final
main] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
main] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.PostgreSQL10Dialect
main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaP
main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view ren
main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
main] com.sb.gltkn.Application               : Started Application in 5.122 seconds (JVM running for 6.551)
```

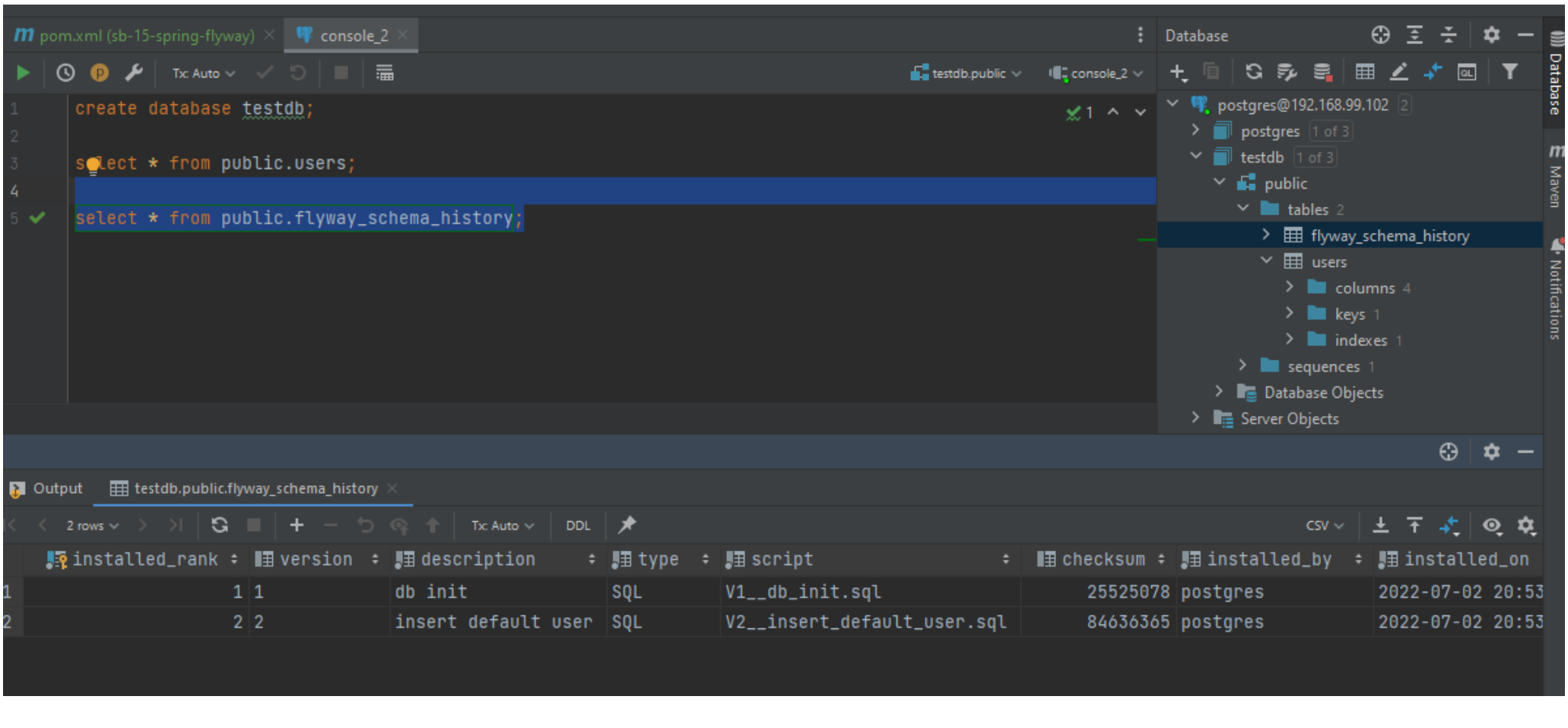


	id	username	firstname	lastname
1	1	testuser	Test	Test Lastname
2	2	testuser2	Test2	Test Lastname2

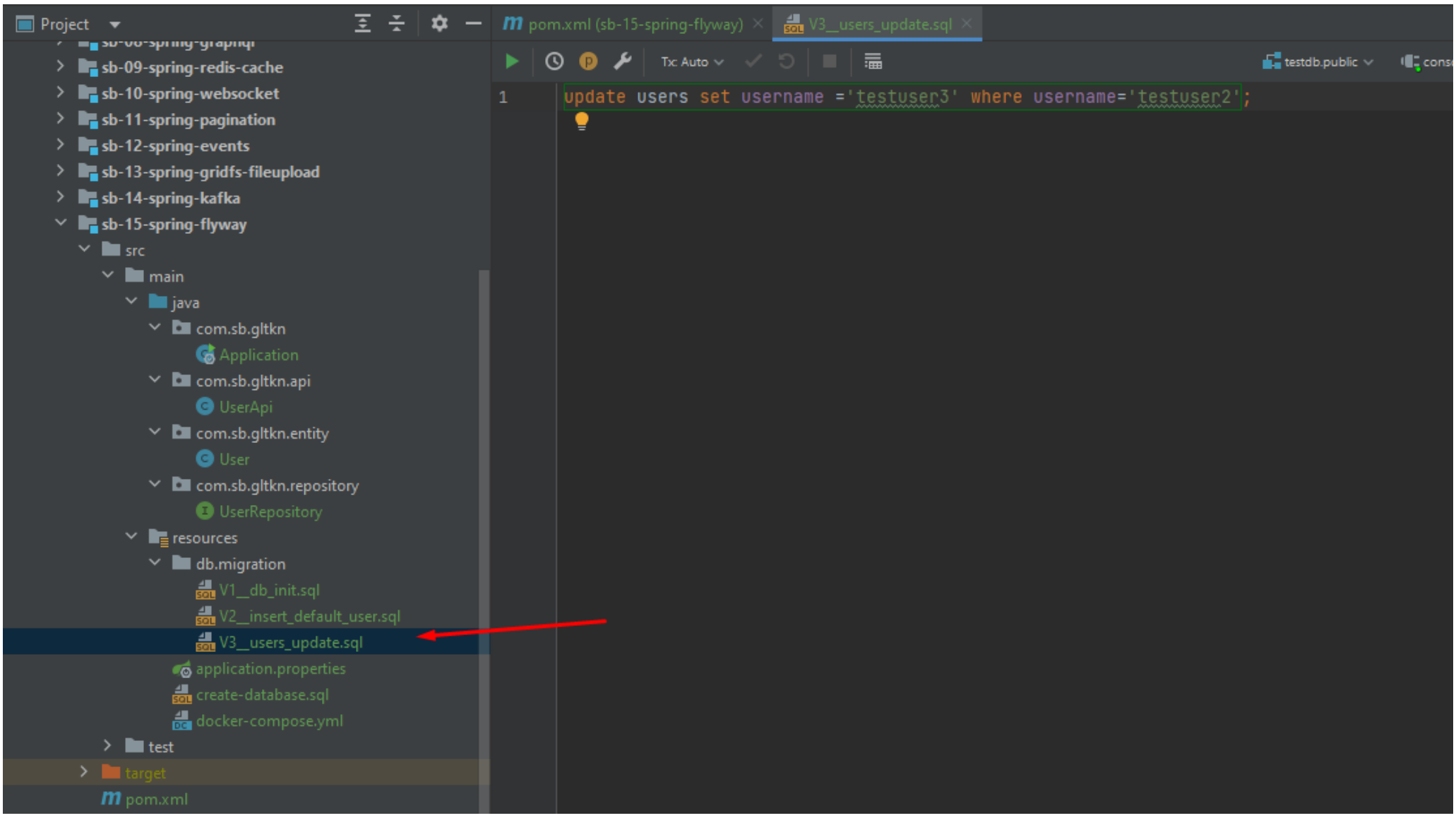
localhost:8080/users

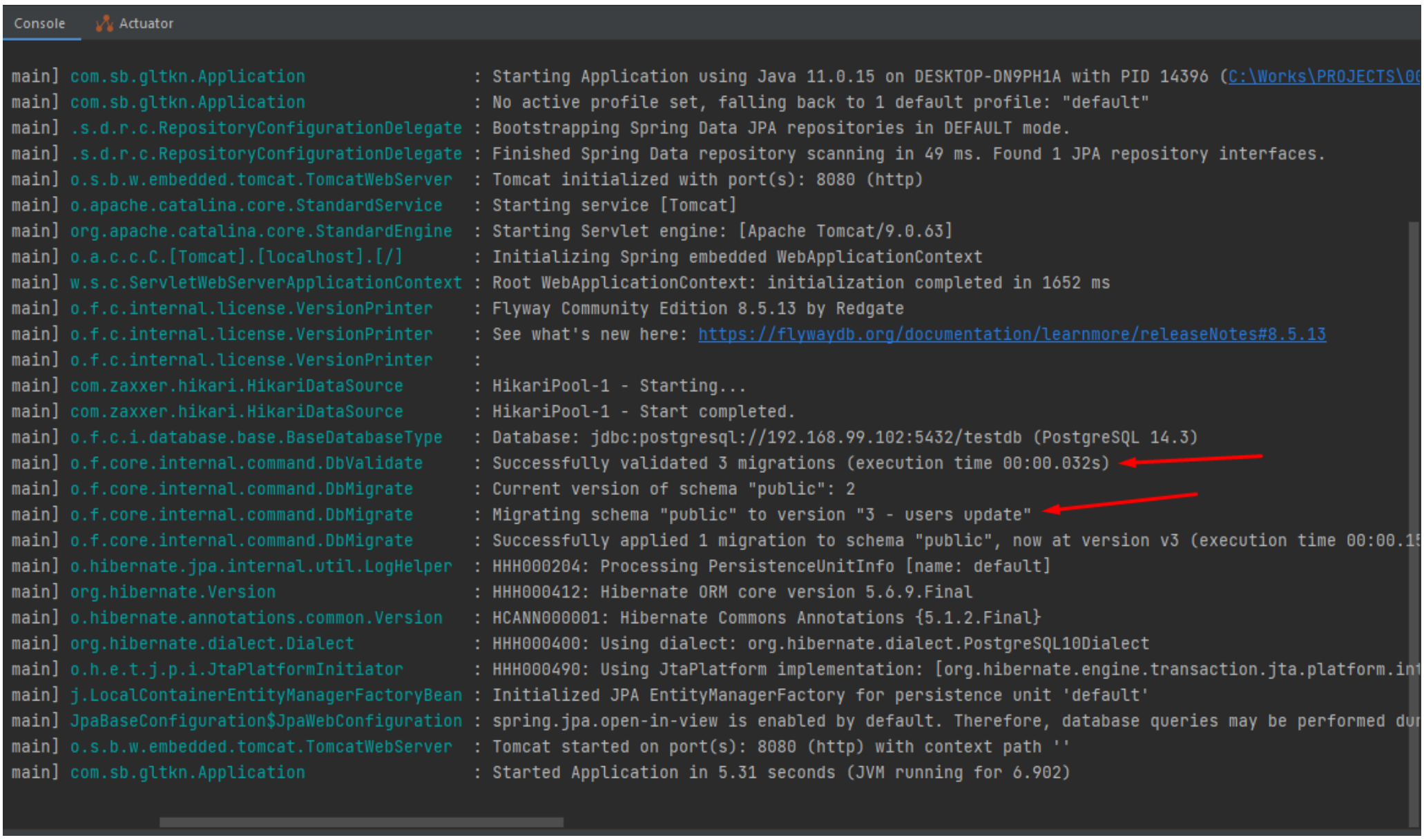

```
// 20220702205851
// http://localhost:8080/users

[
  {
    "id": 1,
    "username": "testuser",
    "firstname": "Test",
    "lastname": "Test Lastname"
  },
  {
    "id": 2,
    "username": "testuser2",
    "firstname": "Test2",
    "lastname": "Test Lastname2"
  }
]
```


Flyway_schema_history tablosu içerisinde maigration bilgileri tutulmaktadır.



Şimdi bir update yapalım. Flyway ile migration gerçekleşecektir.





pom.xml (sb-15-spring-flyway) | V3__users_update.sql | console_2

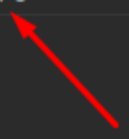
Tx: Auto | testdb.public

```
1  create database testdb;
2
3  select * from public.users;
4
5  select * from public.flyway_schema_history;
```

Output | testdb.public.users

2 rows | Tx: Auto | DDL

	id	username	firstname	lastname
1	1	testuser	Test	Test Lastname
2	2	testuser3	Test2	Test Lastname2



pom.xml (sb-15-spring-flyway) | V3__users_update.sql | console_2

Tx: Auto | testdb.public | console_2

```
1  create database testdb;
2
3  select * from public.users;
4
5  select * from public.flyway_schema_history;
```

Database

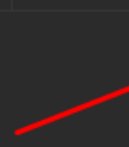
postgres@192.168.99.102 2

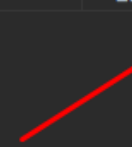
- flyway_schema_history
- users
  - columns 4
  - keys 1
  - indexes 1
- sequences 1
- Database Objects
- Server Objects

Output | testdb.public.flyway_schema_history

3 rows | Tx: Auto | DDL | CSV

	installed_rank	version	description	type	script	checksum	installed_by	installed_on
1	1	1	db init	SQL	V1__db_init.sql	25525078	postgres	2022-07-02 20:5
2	2	2	insert default user	SQL	V2__insert_default_user.sql	84636365	postgres	2022-07-02 20:5
3	3	3	users update	SQL	V3__users_update.sql	-450686414	postgres	2022-07-02 21:0





localhost:8080/users


```
// 20220702210906
// http://localhost:8080/users

[
  {
    "id": 1,
    "username": "testuser",
    "firstname": "Test",
    "lastname": "Test Lastname"
  },
  {
    "id": 2,
    "username": "testuser3",
    "firstname": "Test2",
    "lastname": "Test Lastname2"
  }
]
```