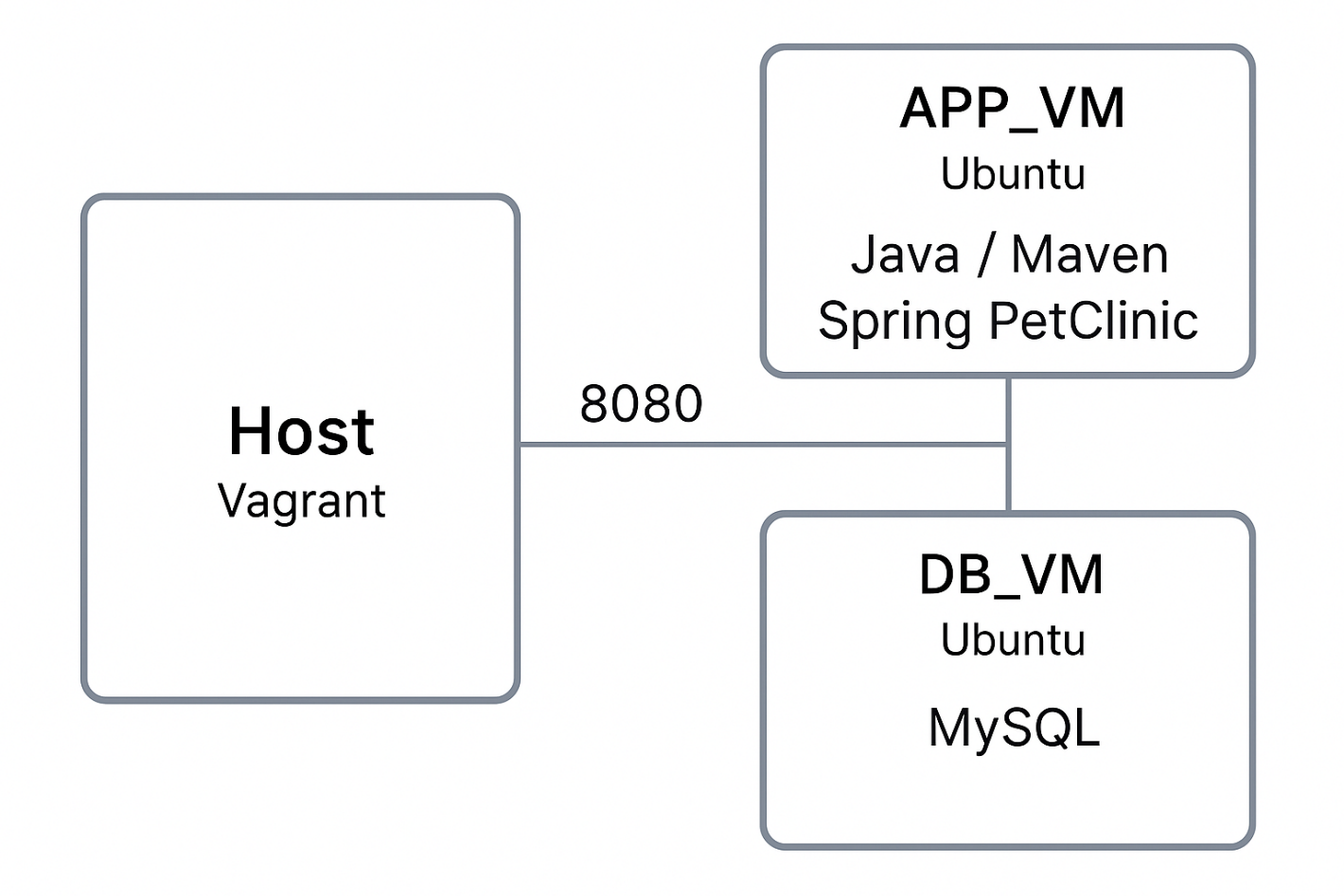
**Step Project 1**

**Topology:**



Step\_Project\_1/

├── Vagrantfile

├── provision/

│ ├── app\_provision.sh

│ └── db\_provision.sh

└── scripts/

Vagrantfile — Vagrant configuration file.

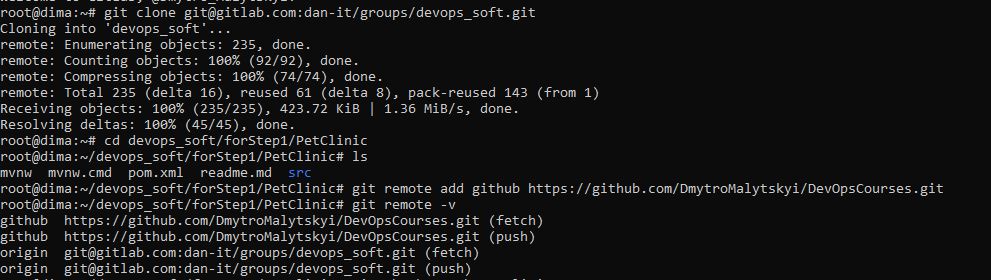
provision/ — directory for all configuration scripts:

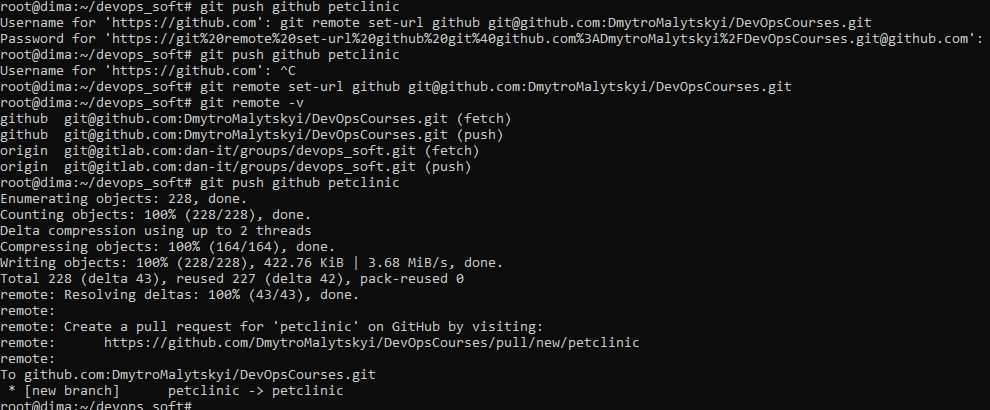
app\_provision.sh — Spring PetClinic configuration.

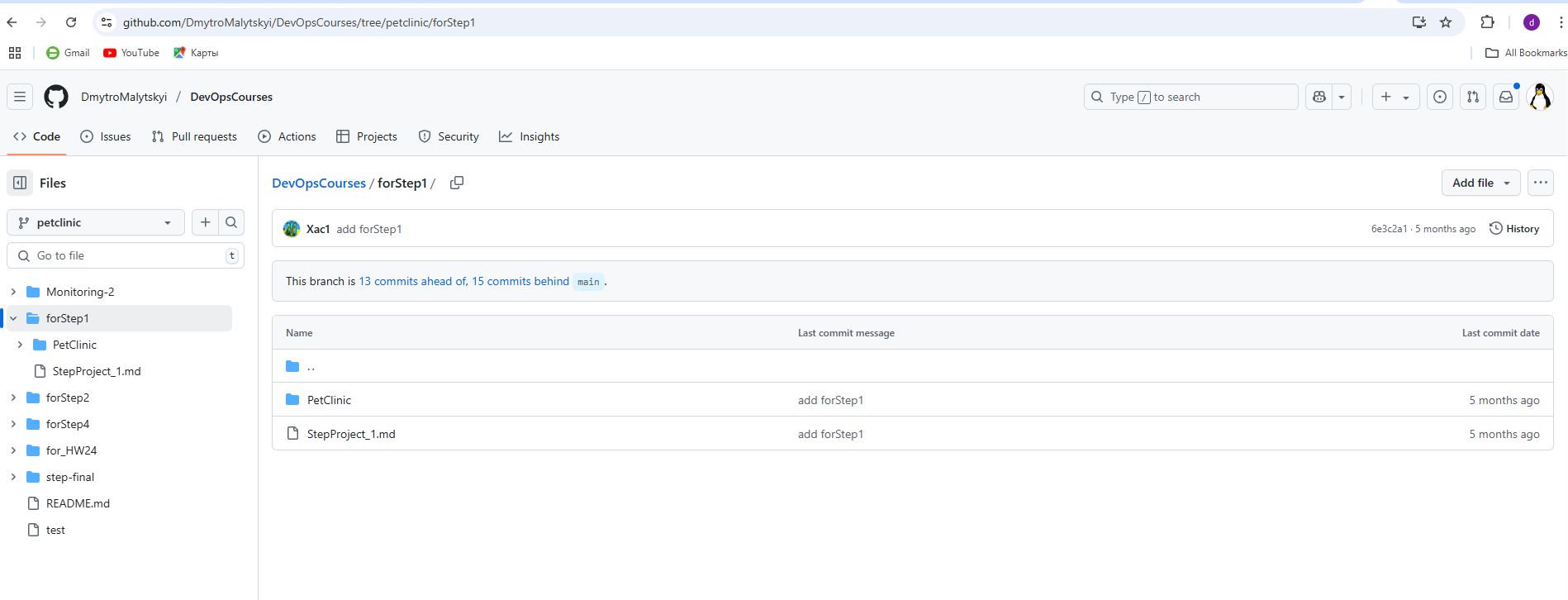
db\_provision.sh — MySQL configuration.

scripts/ — (optional) other additional scripts.

Download the repository from the gitlab account to the local machine and then upload this repository to your own github account.



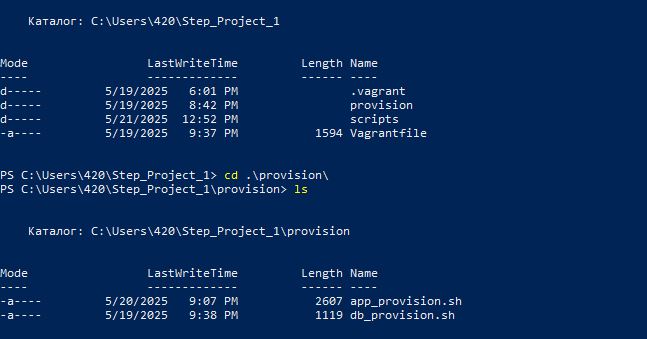




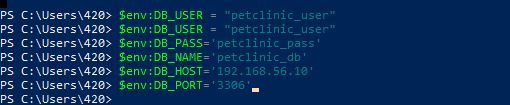
Initialize the Vagrant project



Check folders for project:



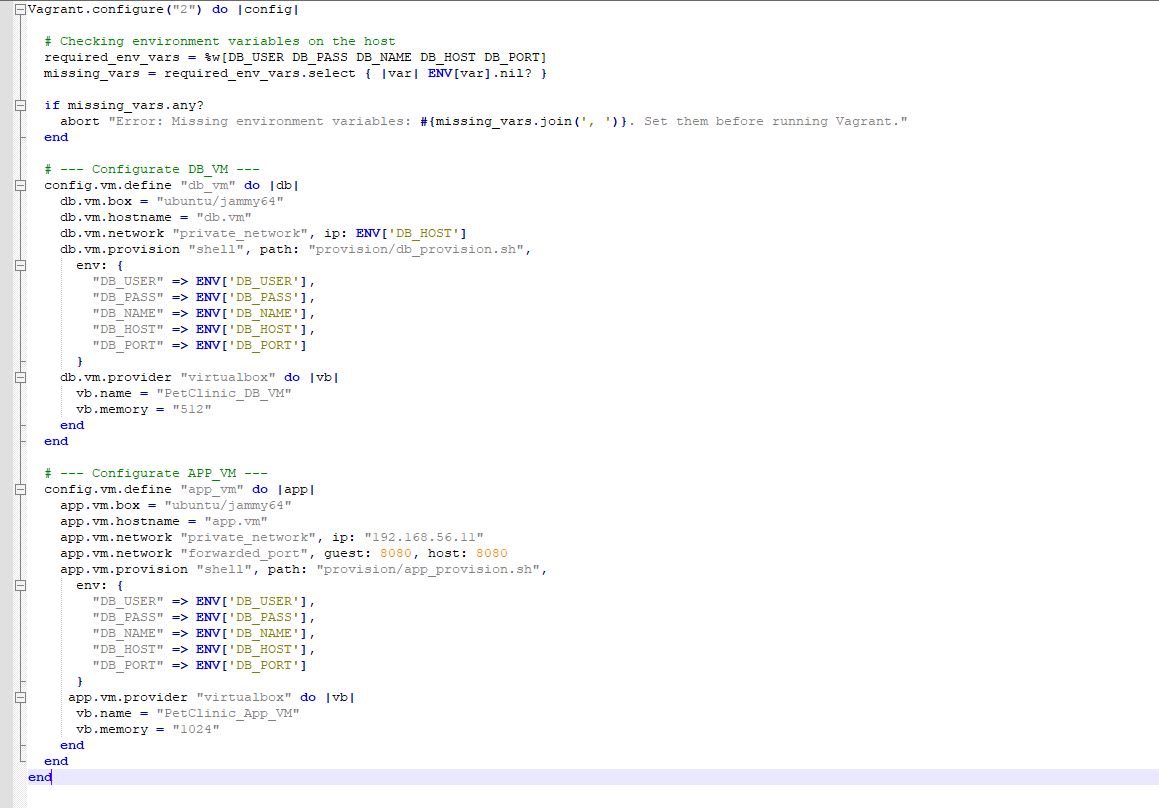
We create, configure environment variables and check what was created on the local host:





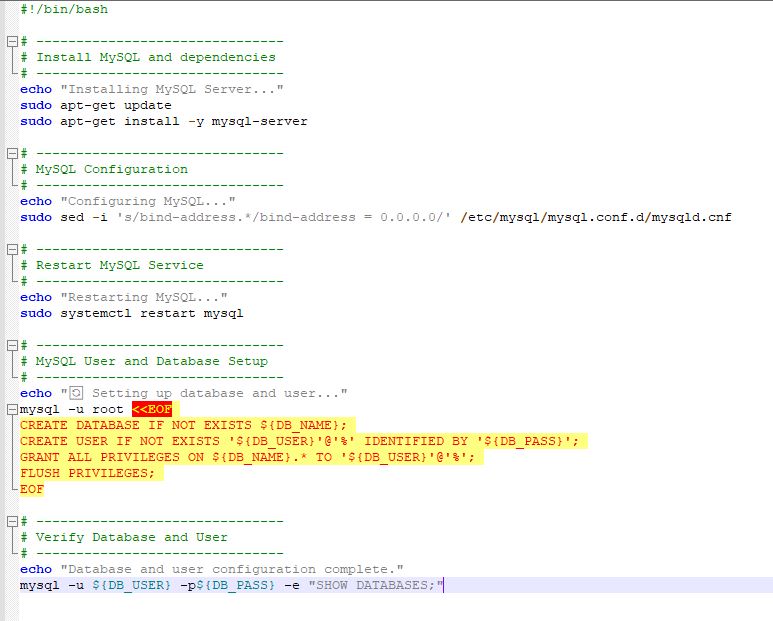
**Subtask I - Infrastructure (Vagrantfile)**

1. We create a Vagrantfile that describes two virtual machines:
   1. APP\_VM – for the application itself.
   2. DB\_VM – for the MySQL database.
2. The virtual machines are based on Ubuntu (ubuntu/jammy64). Each has its own IP address on a private network (does not require public internet).
3. Also, port 8080 is opened for APP\_VM so that you can view the application in a browser from the host machine.
4. ENV variables (such as DB\_USER, DB\_PASS, DB\_NAME) are passed from the host machine to the Vagrantfile to avoid hardcoding passwords in scripts.



**Subtask II - Database (provision/db\_provision.sh)**

1. This Bash script is responsible for installing and configuring MySQL on the DB\_VM virtual machine.
2. **Update packages**:
   * + Use apt-get update to update the list of available software.
3. **Install MySQL server:**
   * + Using apt-get install -y mysql-server.
4. **Restrict connections to private network only:**
   * + Change bind-address in MySQL configuration so that the server accepts connections only from IP address 192.168.56.10.
5. **Create a database and user:**
   * Using SQL commands, the database ${DB\_NAME} and the user ${DB\_USER} with the password ${DB\_PASS} are created.
   * This user is only allowed access from the application machine IP 192.168.56.11.



**Subtask III - Application (provision/app\_provision.sh)**

This script automates the deployment of the **Spring PetClinic** application on the **APP\_VM** virtual machine.

**1.Installing Dependencies:**

* + - Java (OpenJDK 8), Git, Unzip, dos2unix
    - Maven is not installed separately because the project uses the built-in mvnw (Maven Wrapper)

**2. Creating the petapp user (variable APP\_USER):**

* + - The application is run under a dedicated non-root user for security
    - Directories /home/petapp/app and /home/petapp/project are created

**3. Cloning the Spring PetClinic Repository:**

* + - Clones the petclinic branch from the DevOpsCourses GitHub repository
    - Project path: /home/petapp/project/source/forStep1/PetClinic

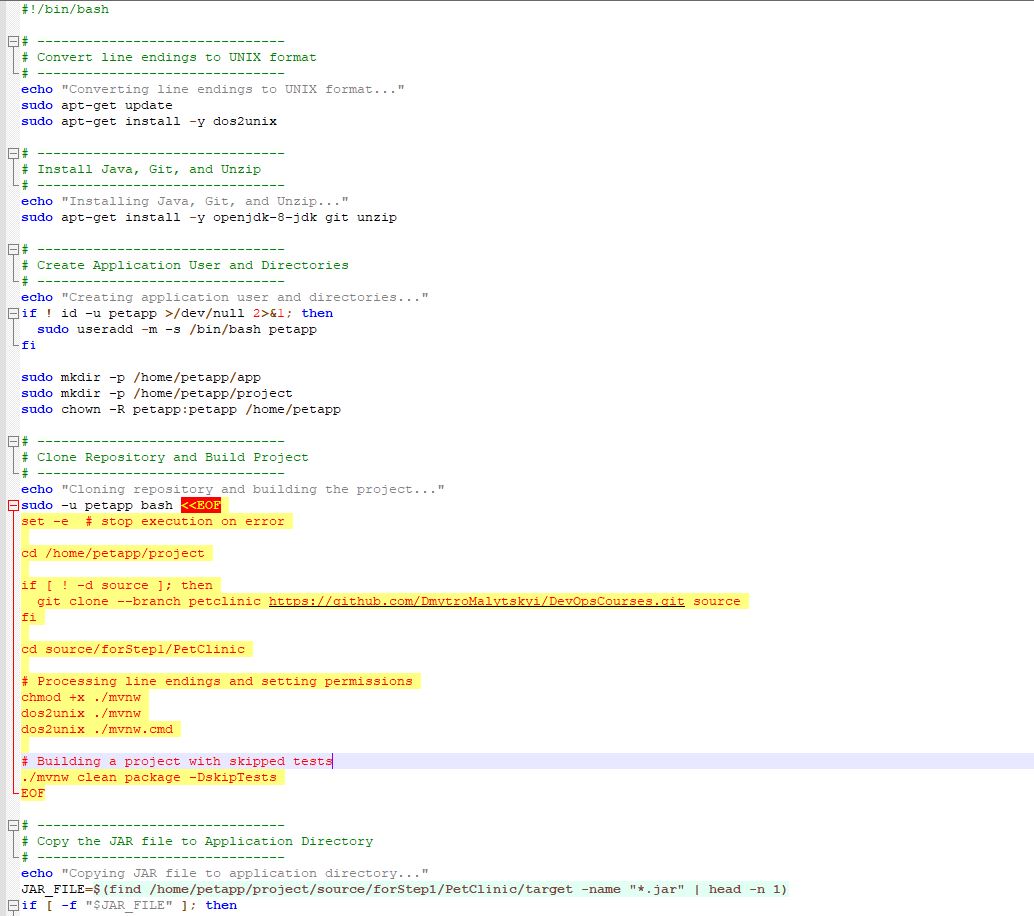
**4. Building the Project and Generating a JAR File**

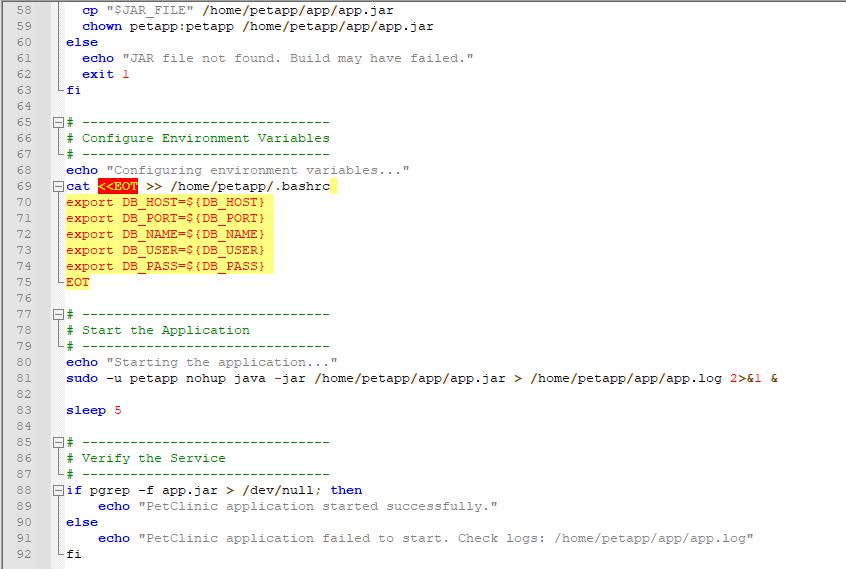
**5. Configuring Environment Variables**

**6. Running the Application in the Background**

**7. Verifying Application Startup**

* + - The script checks if the app.jar process is running and displays a status message accordingly

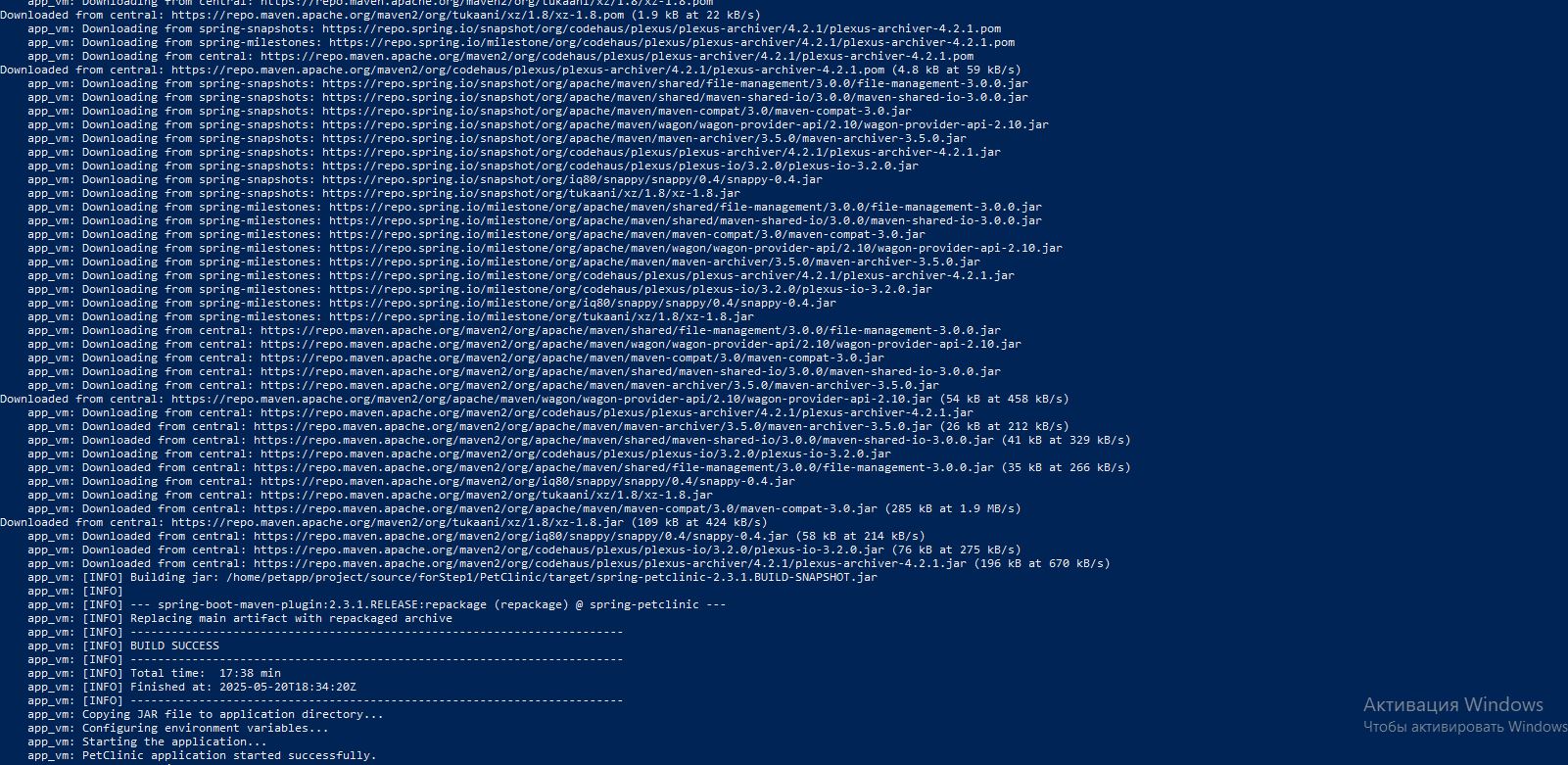




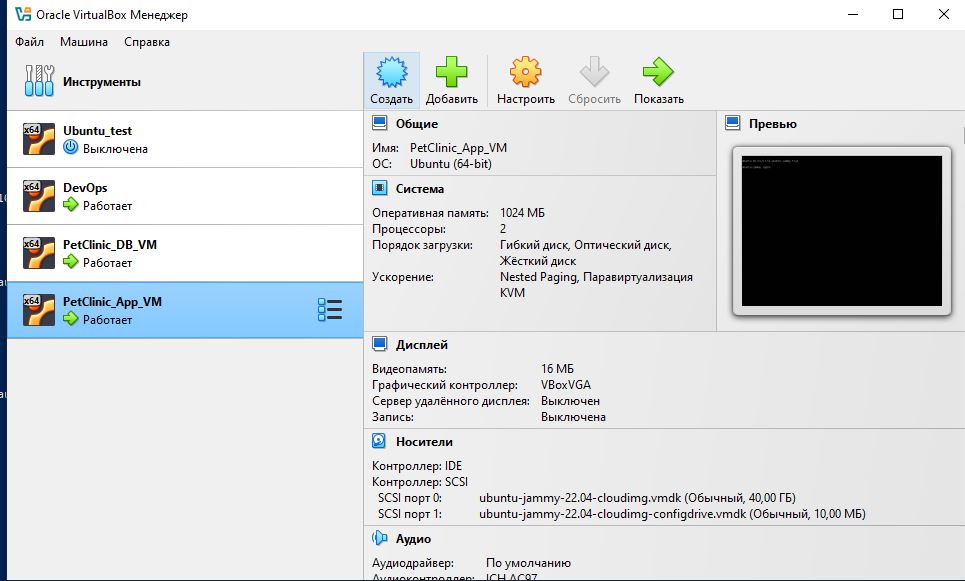
We start Vagrant

* + - vagrant up

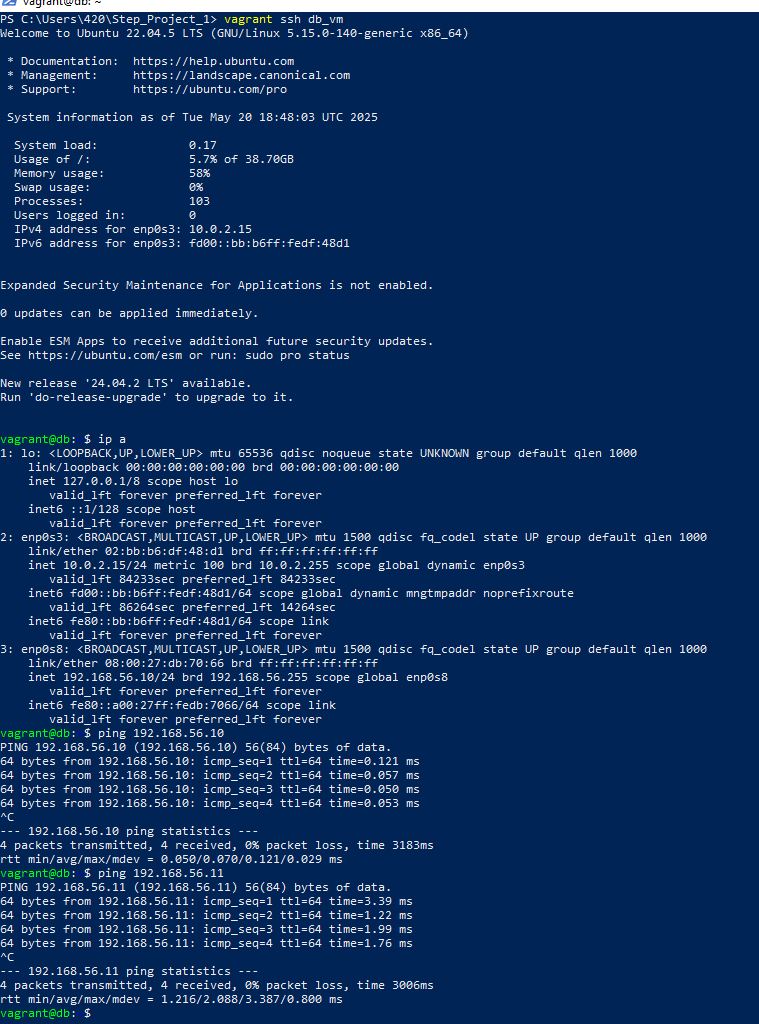
Let's check that the DB\_VM nf APP\_VM virtual machines were successfully created:

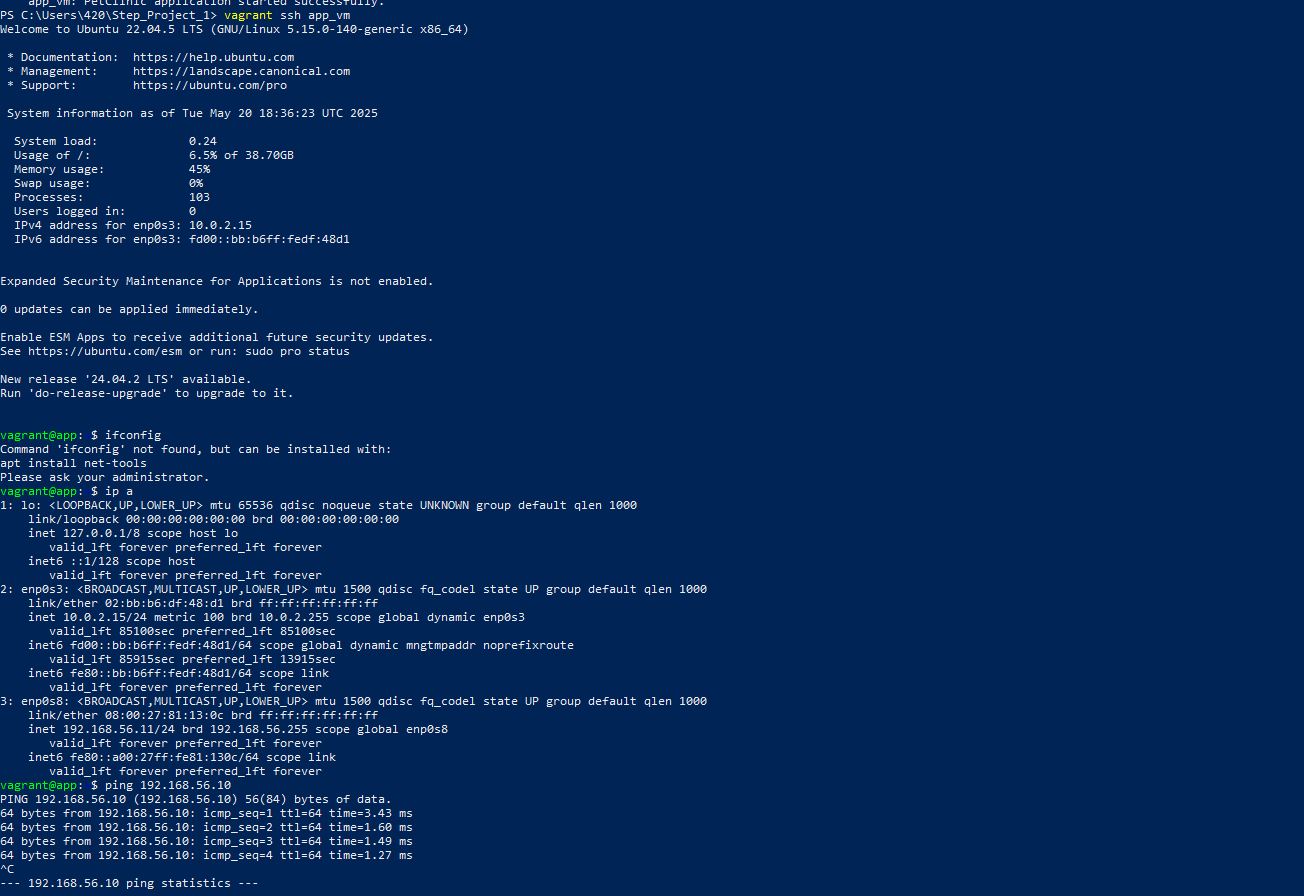


Let's check that the DB\_VM nf APP\_VM virtual machines were successfully created on virtual box::



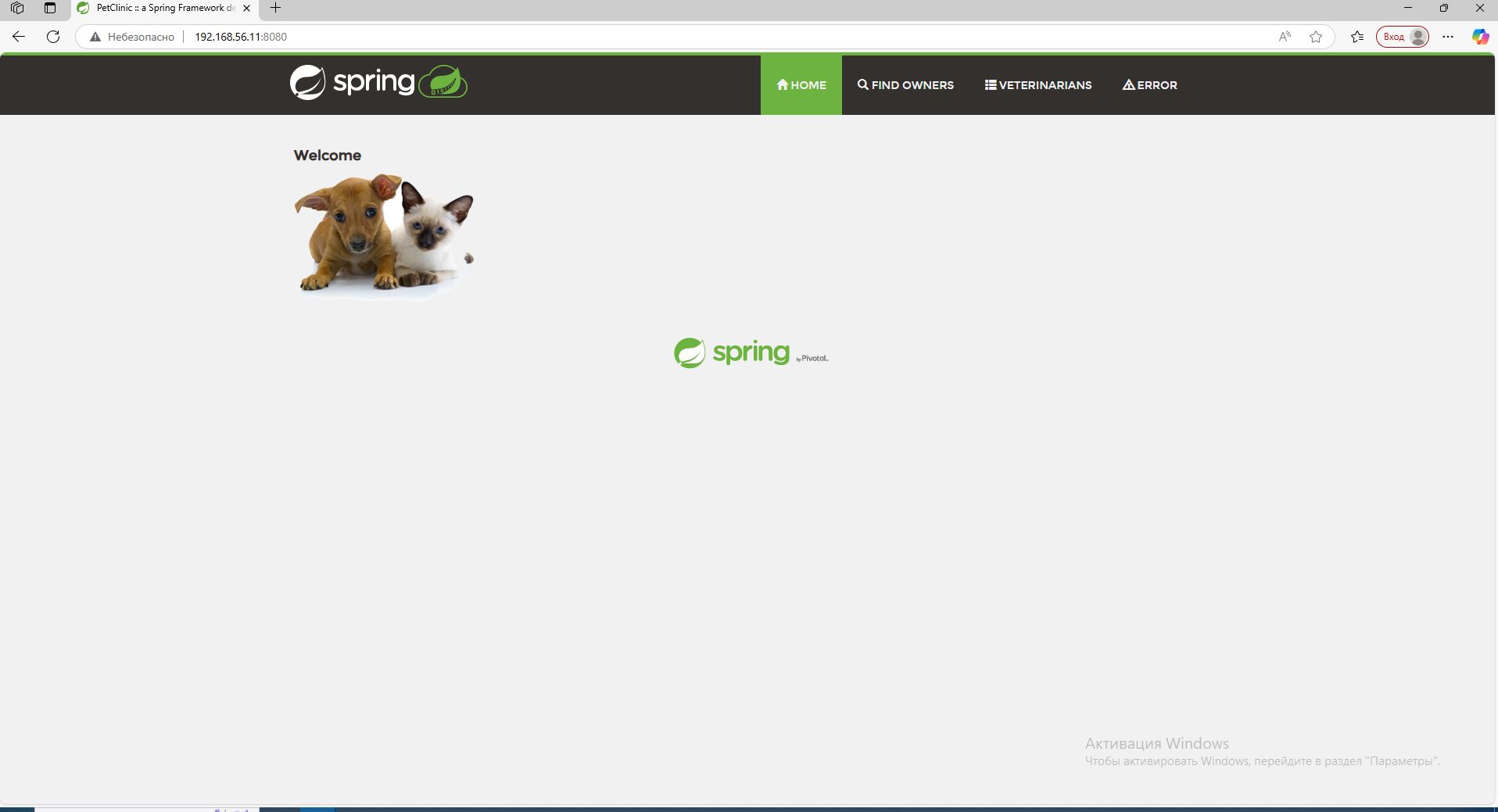
Go to PetClinic\_DB\_VM and check if there is a ping PetClinic\_APP\_VM and go to PetClinic\_APP\_VM and check if there is a ping PetClinic\_DB\_VM

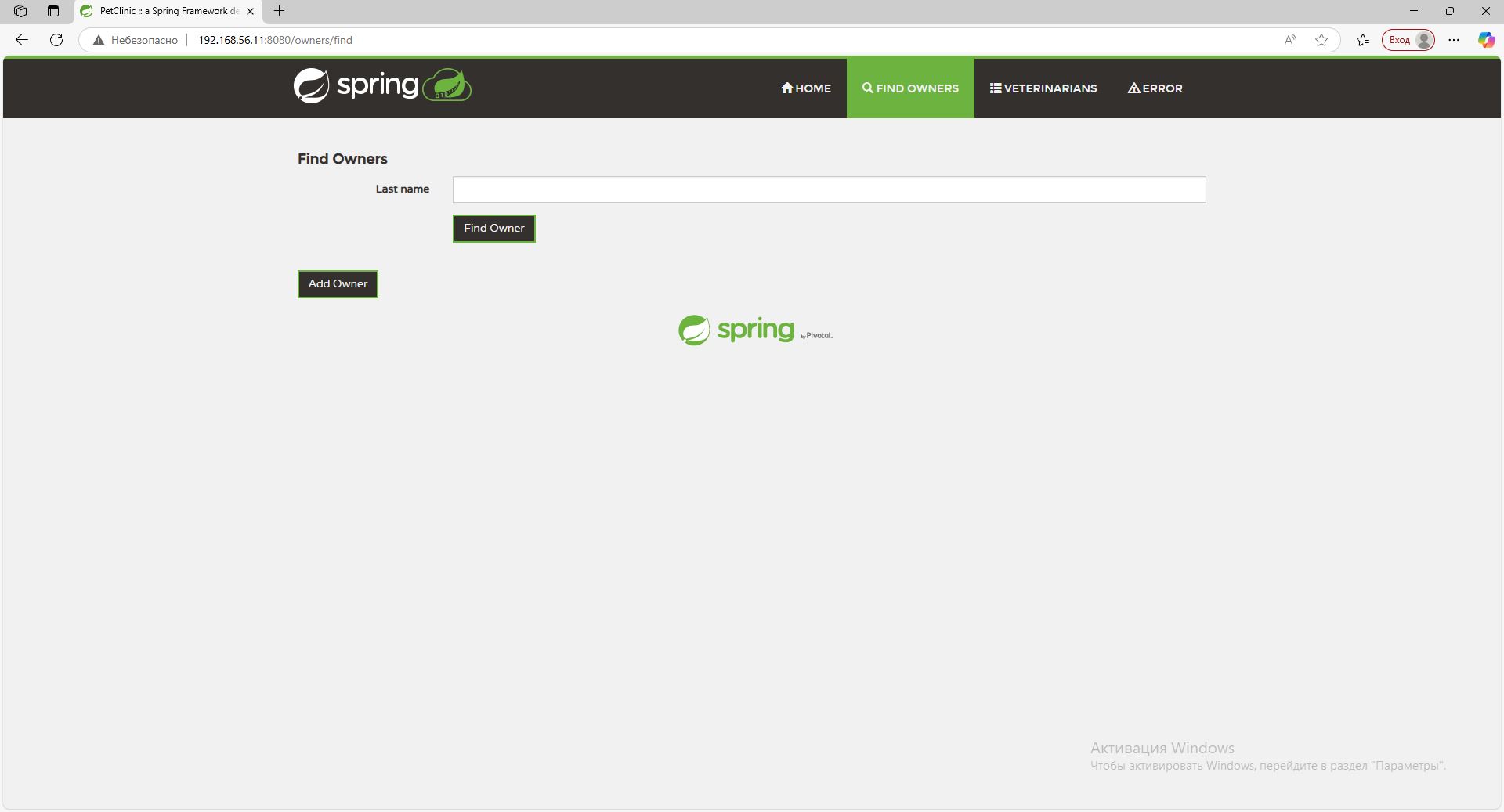


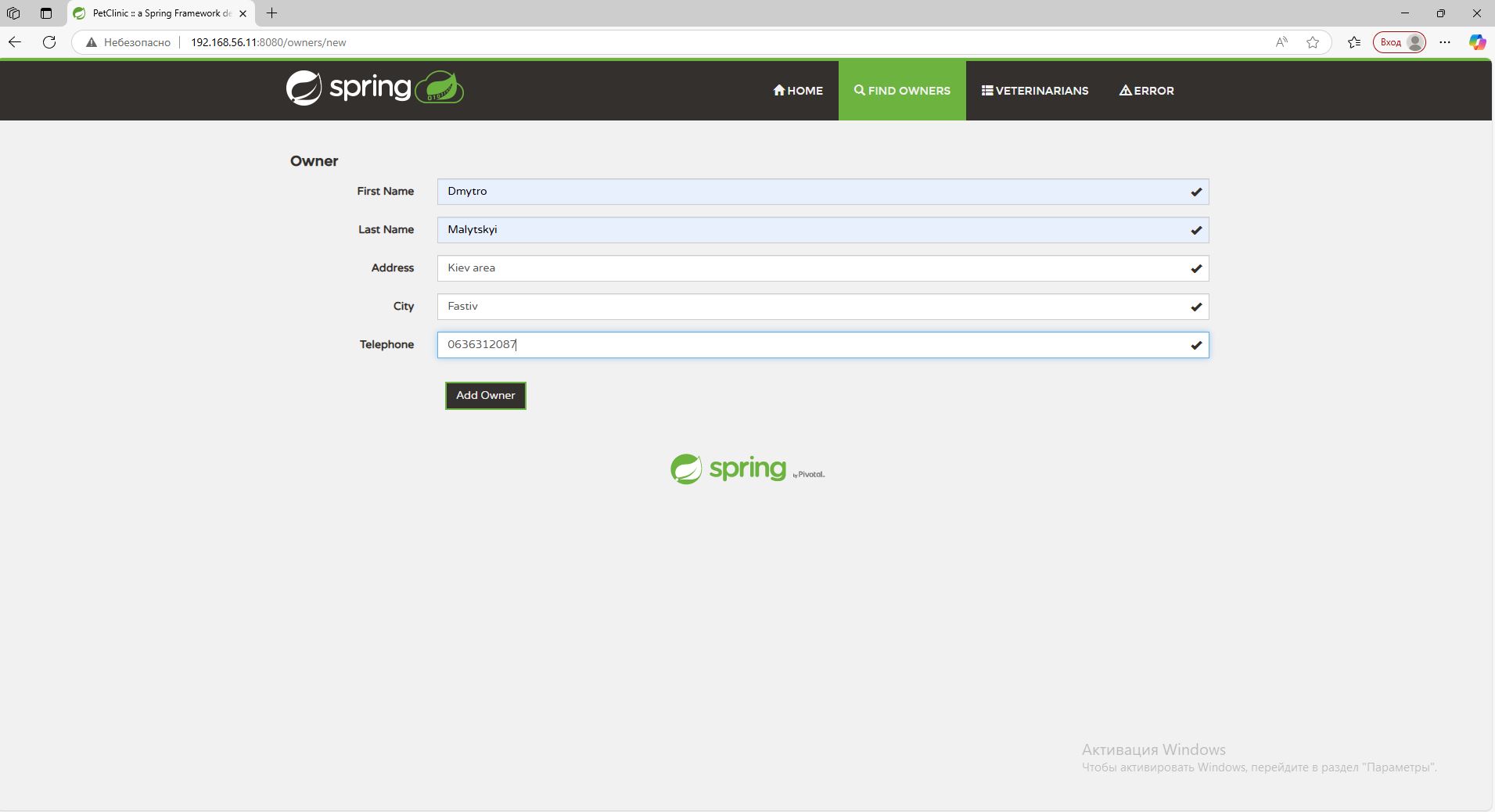


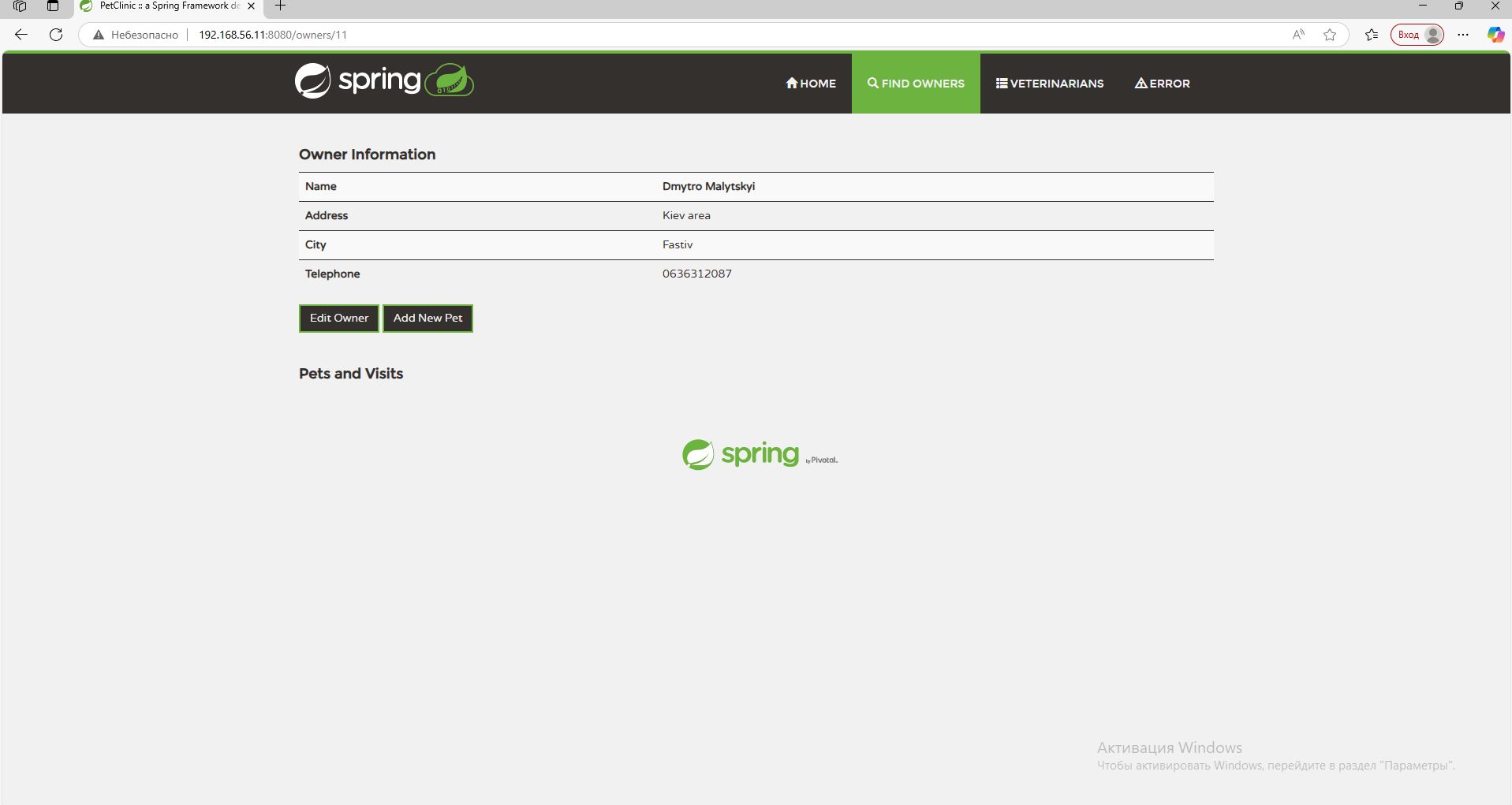
We check the operation of the application on port 8080 by adding our data.

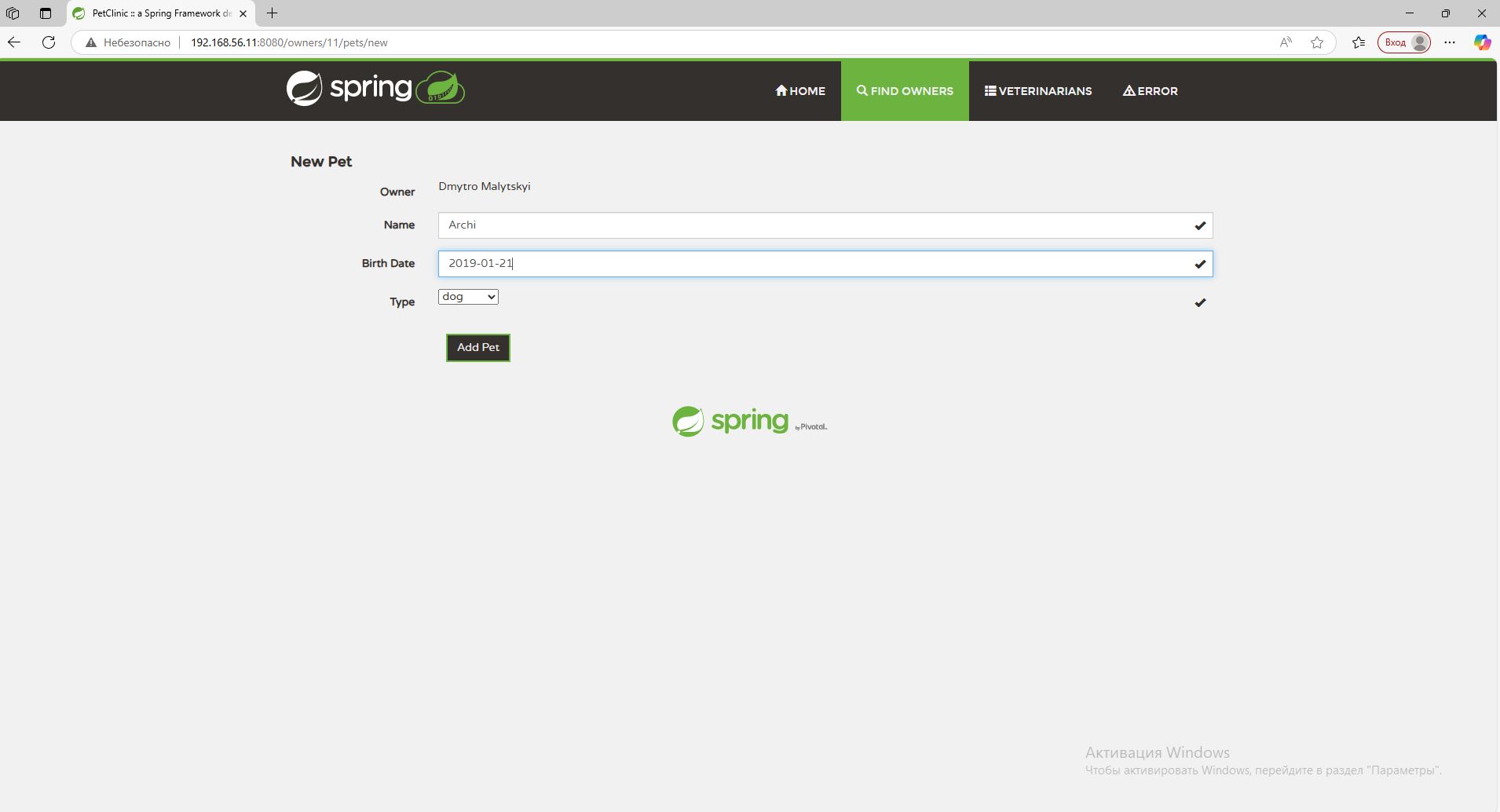
We also check the display of the added information.

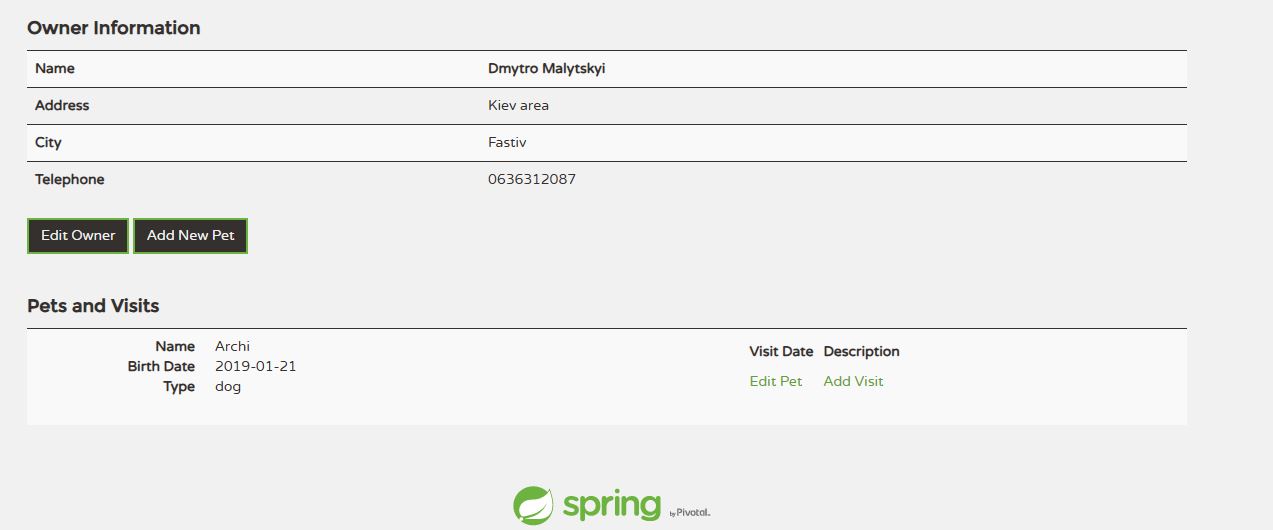












We connect to the database from PetClinic\_APP\_VM

