AAD / Backend Development

Doing updates

Erik van der Arend / Jan Willem Boer

23/24 Q3

SAXION
UNIVERSITY OF
APPLIED SCIENCES

Assessment: 14-11, 16-11 (Enschede) 15-11, 16-11 (Deventer)

During the assessment:
- Demonstrate your API
- Demonstrate your advanced track

How to demonstrate? Examples:

- Deployment: push a change and show it arrives at production
- QM: push a change and show the process that follows
- Security: show how you prevented certain attacks, show that you use https
- External data: simulate that the service goes down and show your application still works

Do an integrated demo of all three modules.

Note: projects

- The projects have been published this week.
- Register for the project phase and choose a project or a team.
  Note: you can’t participate in a project if you didn’t register.
- Register before Friday the 20th of October (9AM). That is tomorrow.
- In week 9-10 you can contact the client already and start the project

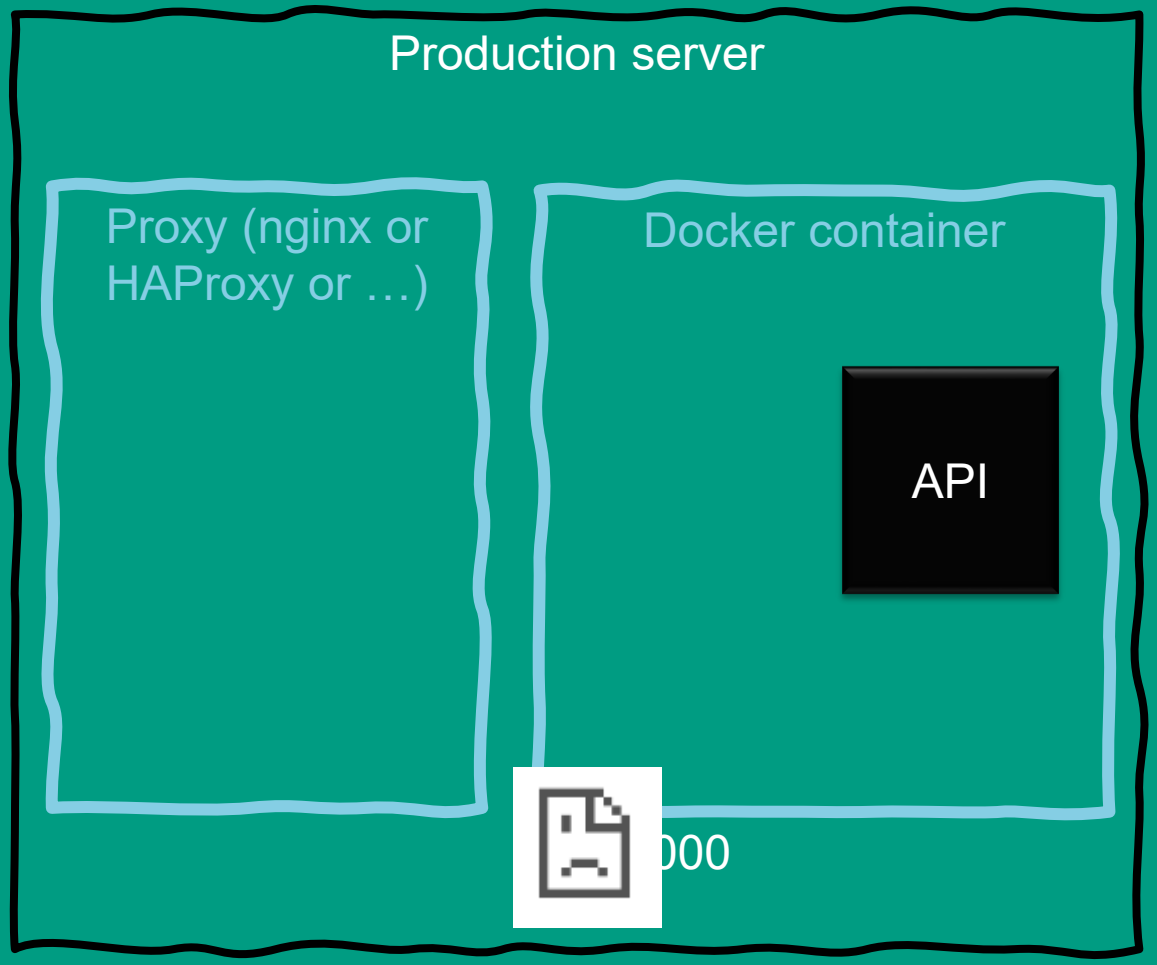
- In week 2.1: official start project
- In week 2.9-10: assessment for the team

Updating

- Program code / binaries
- Database

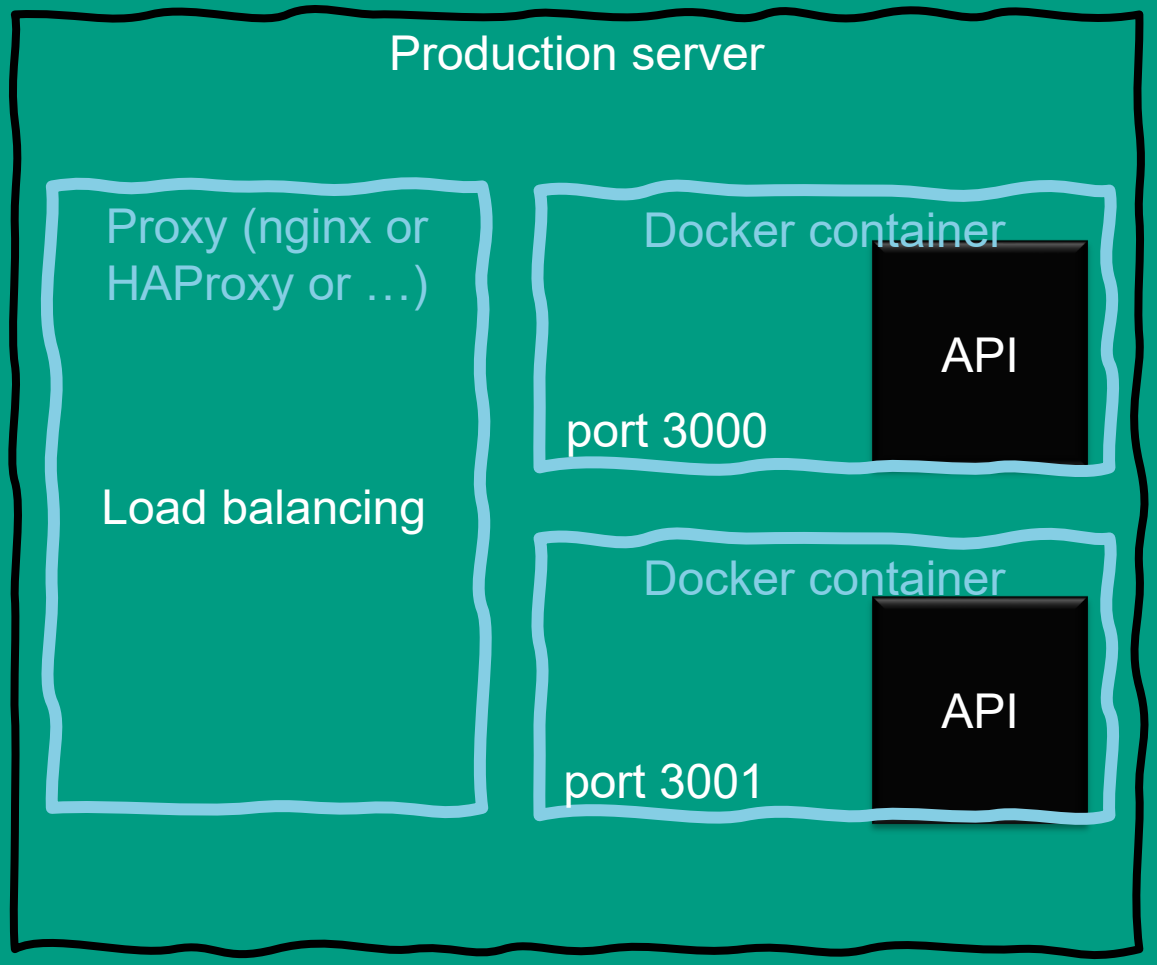
Updating program code / binaries

https://my-api.saxion.online/



Updating program code / binaries: keep uptime

https://my-api.saxion.online/



How to, step 1

Spin up two containers instead of one.

Could be on different hosts or the same host (but different ports)

```
sudo docker run -d -p 3000:3000 --name aad-demo-3000 janwillemboer/aad-demo
```

```
sudo docker run -d -p 3001:3000 --name aad-demo-3001 janwillemboer/aad-demo
```

How to, step 2

Tell nginx to do load balancing and restart nginx
(Note: you created this config if you followed the tutorial in week 4)

```
upstream aad-demo {
        server localhost:3000;
        server localhost:3001;
}
server {
        server_name aad-demo.saxion.online;
        location / {
                proxy_set_header X-Forwarded-For $proxy_add_x_
                proxy_set_header Host $host;

                proxy_pass http://aad-demo;

                proxy_buffering off;
                proxy_http_version 1.1;
                proxy_set_header Upgrade $http_upgrade;
```

How to, step 3

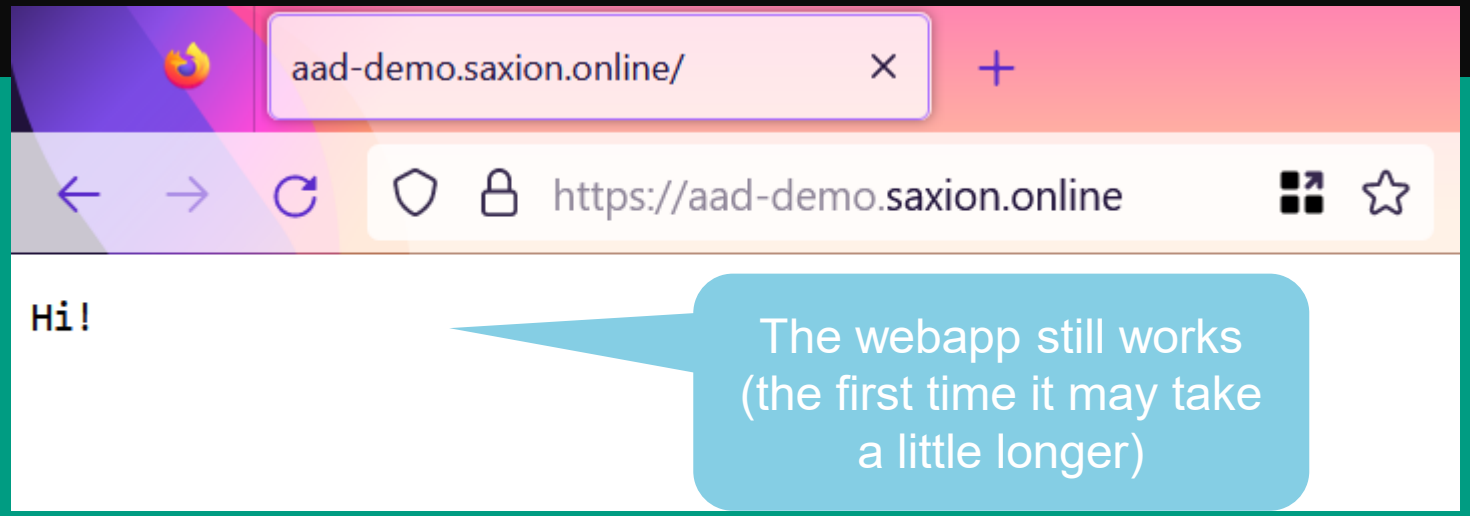
Get the container IDs

Test it: kill one of the instances and see that the website still works

```
ubuntu@ip-172-31-33-176:~$ sudo docker container ls
CONTAINER ID   IMAGE                     COMMAND                  CREATED         STATUS         PORTS
                    NAMES
acf3ad9a0849   janwillemboer/aad-demo    "docker-entrypoint.s…"   4 minutes ago   Up 7 seconds   0.0.0.0:3001->3000/tcp,
:::3001->3000/tcp   aad-demo-3001
da9effd56ff6   janwillemboer/aad-demo    "docker-entrypoint.s…"   6 minutes ago   Up 6 minutes   0.0.0.0:3000->3000/tcp,
:::3000->3000/tcp   aad-demo-3000
ubuntu@ip-172-31-33-176:~$ sudo docker container stop da9effd56ff6
da9effd56ff6
ubuntu@ip-172-31-33-176:~$
```

This is the one listening on port 3000

KILL IT



The webapp still works (the first time it may take a little longer)

SAXION UNIVERSITY OF APPLIED SCIENCES

But…

- Two different versions of the API coexist
  - Backwards compatibility
  - Database compatibility
- It is a complex process to do right.

Updating the database

Updating the database - STRUCTURE

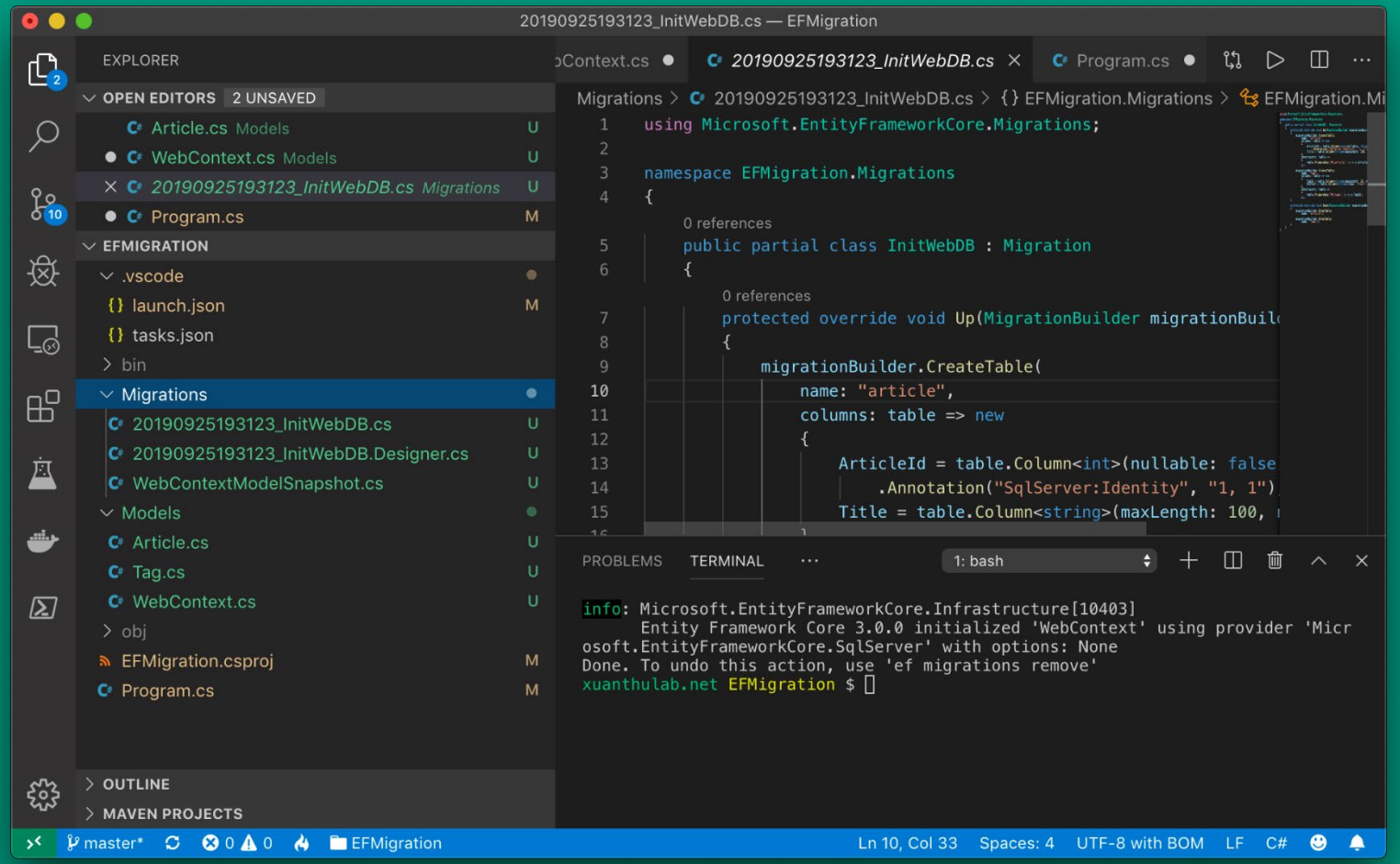
- Adding columns
- Deleting columns
- Renaming tables
- Adding tables
- …

Always done using scripts with database statements

```
9   ALTER TABLE Persons
10  ADD COLUMN City VARCHAR,
11  ADD COLUMN Phone_no VARCHAR;
12
```

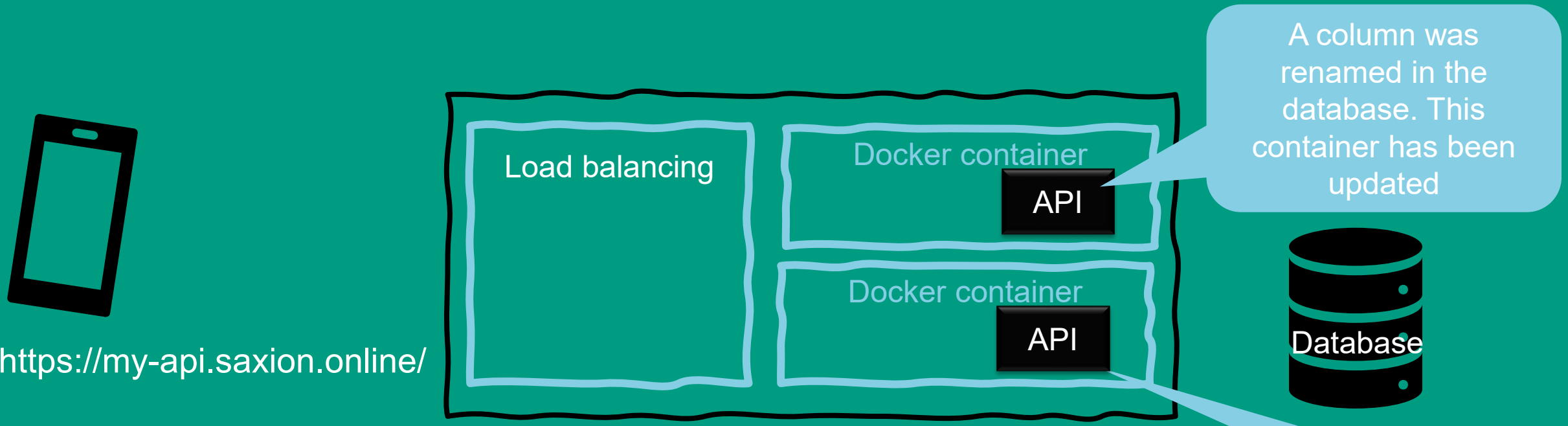
Updating the database - STRUCTURE

- Update scripts are created using a framework
- This is called **Schema Migrations**
  - Builtin into Dotnet core (EFCore)
  - Additional libraries for Hibernate
  - Knex for nodejs
  - …



SAXION UNIVERSITY OF APPLIED SCIENCES

Schema migrations may have to be done in phases



Solution: work in phases

1. Add migration that creates new column
2. Change the API, so it will use the new column, but if it is empty, then use the old column
3. Update the database and API
4. Add migration that moves data from the old column to the new column and drops the old column
5. Update the database

Schema migrations may have to be done in phases

2 Answers

Sorted by: Highest score (default)

36

You may edit created migration class (`Up` and `Down` methods) and include any SQL you want in correct place:

```
protected override void Up(MigrationBuilder migrationBuilder)
{
    migrationBuilder.DropColumn(
        name: "MyExtraColumn",
        table: "MySuperTable");

    migrationBuilder.Sql("DROP DATABASE [master]"); // <<< Anything you want :)

    migrationBuilder.DropColumn(
        name: "MyExtraColumn2",
        table: "MySuperTable");
}
```

It's your responsibility do not "break" the database schema (in EntityFramework point of view) and to provide reliable `Down` script to be able to migrate up/down/up/down multiple times.

How is this relevant

For the advanced track **availability** it is

For the rest: important to have this way of thinking about updates (later when you grow up)

- Always have a process that works.
- Small investment up front, nice returns later

Now what

This week

- Continue with Assignment 2