Deployment Plan

Version	Date	Author(s)	Changes
0.1	22/04/21	David La Gordt Dillie, Florin Deleanu	First draft
0.2	10/05/21	David La Gordt Dillie, Florin Deleanu	Added steps to run local instance
0.3	17/6/21	David La Gordt Dillie	Added instructions for docker images

Contents

1. Overview

1.1 Purpose

The purpose of the deployment plan are as follows:

- Lay out the steps for running the application.
- Cover the capabilities of the application.
- Keep a record of the dependencies used.
- Describe the deployment steps

1.2 Assumptions

The website should have a place for users to:

- Login with their outlook account
- Provide a list of upcoming appointments
- Provide a search feature to find a specific appointment
- Have sorting capabilities to order appointments by date or alphabetically
- Provide an indication if the appointment has started and whether it started on-time

1.3 Dependencies

The back end of the application which takes images and transforms them into usable string variables relies on a license plate recognition system as an external dependency which is added as a module into our project.
We are also using a JavaCV dependency which allows us to input a video and split it into images to be further used by the license plate recognition module. Alternatively we get direct video stream from a camera and periodically take screenshots to input into the plate recognition system

1.4 Constraints

These factors can limit the ability to deploy the application:

- Change of client's needs for the project
- License plate recognition or API failure
- Email sending failure
- Database mismatch/ wrong database credentials

2. Deployment

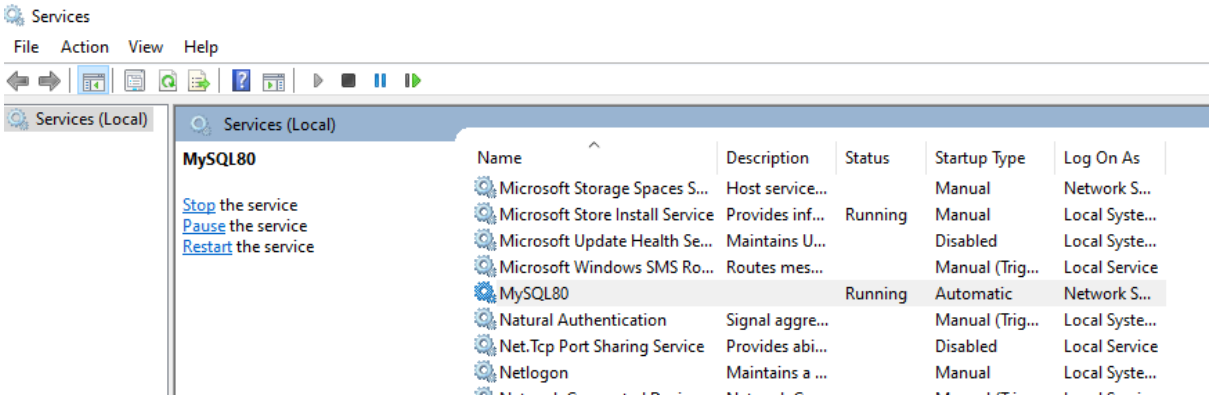
2.1 Schedule

Once all tests pass and the needs of the client have been met, the application will be ready for deployment. The deployment process will happen every 3 weeks/ every iteration.

3. Local

3.1 Prerequisites

- Docker installed
- MySQL80 Service is stopped (Windows search -> Type “Services” -> Find MySQL80) and stop the service (This is to prevent the mysql service on your pc colliding with the docker mysql service).



This is what MySQL80 service looks like while it is running.

3.2 Front-end

1. Open a powershell.
2. Run the command `docker pull davekhaki/certiorem-front` to pull the image.
3. Run the command `docker run -it --rm -p 3000:3000 davekhaki/certiorem-front` to start the frontend.

3.3 Back-end

1. Open a new powershell.
2. Run the command `docker pull mysql` to pull the latest mysql image.
3. Run the command `docker run -d -p 3306:3306 --name=docker-certiorem --env=”MYSQL_ROOT_PASSWORD=Password1234”`

`--env=”MYSQL_DATABASE=interfacesean” mysql` to start the mysql docker container.

4. Open a new powershell.
5. Run the command `docker pull davekhaki/certiorem` to pull the latest image for the backend.
6. Run the command `docker pull --link docker-certiorem -p 8081:8081 davekhaki/certiorem` to start the backend.
7. After all three containers are running, visiting localhost:3000 will show the website, and due to the fact that the database is created from scratch and therefore empty, no records will be visible when the app first loads.