Projeto Segurança Informática em Redes e Sistemas

THE CORK, GROUP A57

96902 | Pedro Gomes | 13 de dezembro de 2022
100740 | João Salgueiro
105688 | Tiago Figueiredo

1. Infrastructure Report

I. BUSINESS CONTEXT

TheCork is a service that allows its users to taste the best food and wines in town by enabling easy restaurant reservations via a very user-friendly mobile application that integrates seamlessly with the restaurants' calendars.

TheCork offers a mobile app to its customers to display the list of restaurants available at the user's location. It also allows them to book a table for a specific number of people if the restaurant still has free tables. There is also a website that provides the same functionality.

TheCork also provides back-office for the restaurants which is used to manage the schedule and to approve or deny the customers reservations.

II. INFRASTRUCTURE OVERVIEW

The group decided to implement an infrastructure in OS Ubuntu v22.04. It consists of two VM's and one virtual network switch. VM CORK-DB has one network interface connected to the local network and works as a database server. CORK-WEB will work as a web application server being the service provider. CORK-WEB has two network interfaces, one configured for the local network, the other NATed to host computer network card. The switch emulates the network 192.168.0.0/24. CORK-WEB will also work as the firewall using Ubuntu built-in software firewall.

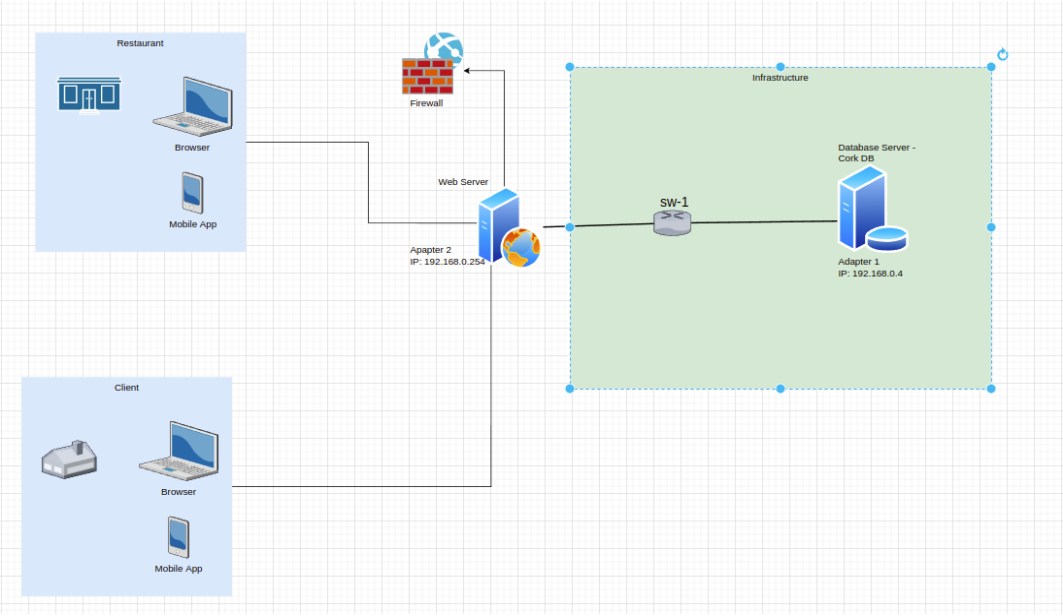

Figure 1 - Network diagram

For the database server, MySQL was installed, and an instance called cork was created, with it's own admin user "corkadmin". That instance will hold all the Tables for this project, as seen below.

To ease operation of the database server, the GUI MySQL Workbench was also installed.

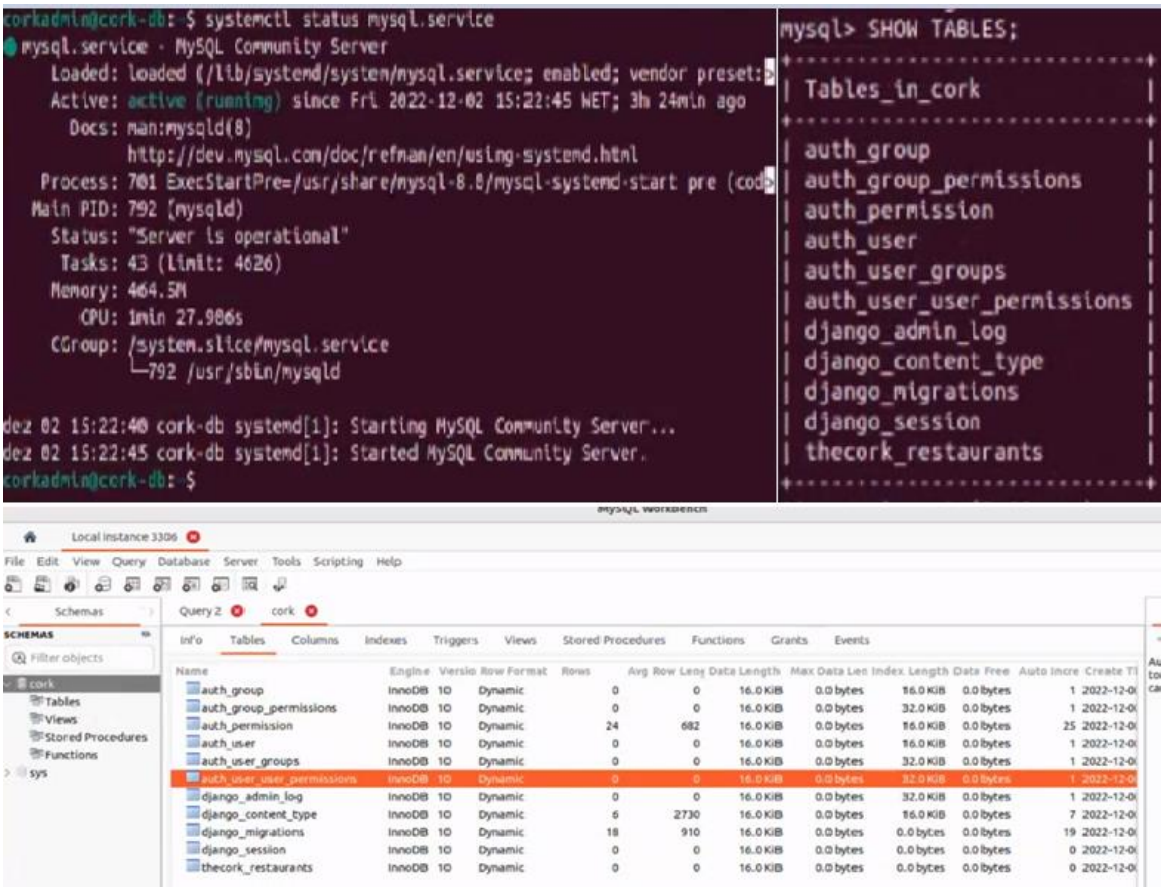

Figure 2 – Database details

To implement the Web App, the group decided to use a framework called Django. That framework uses Python as a programming language and is structured on a design pattern called MVT (Model-View-Template) while providing a lot of custom built in administration tools for user management along with the multitude of Python libraries which could prove necessary for this project.

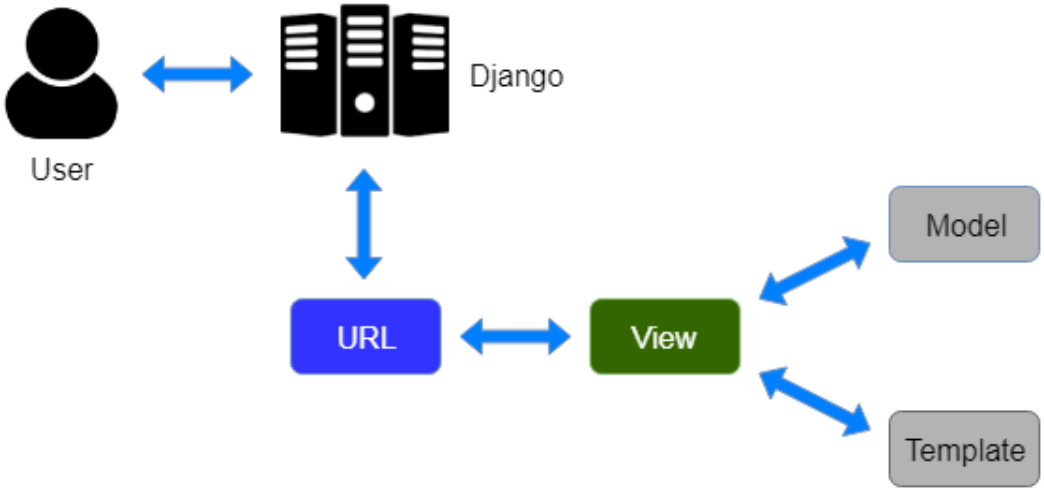

Figure 3 - MVT Pattern

The template part handles the user interface using html pages, the view executes all the logic and interacts with the model to carry data and the model handles the relational structure implemented in the database.

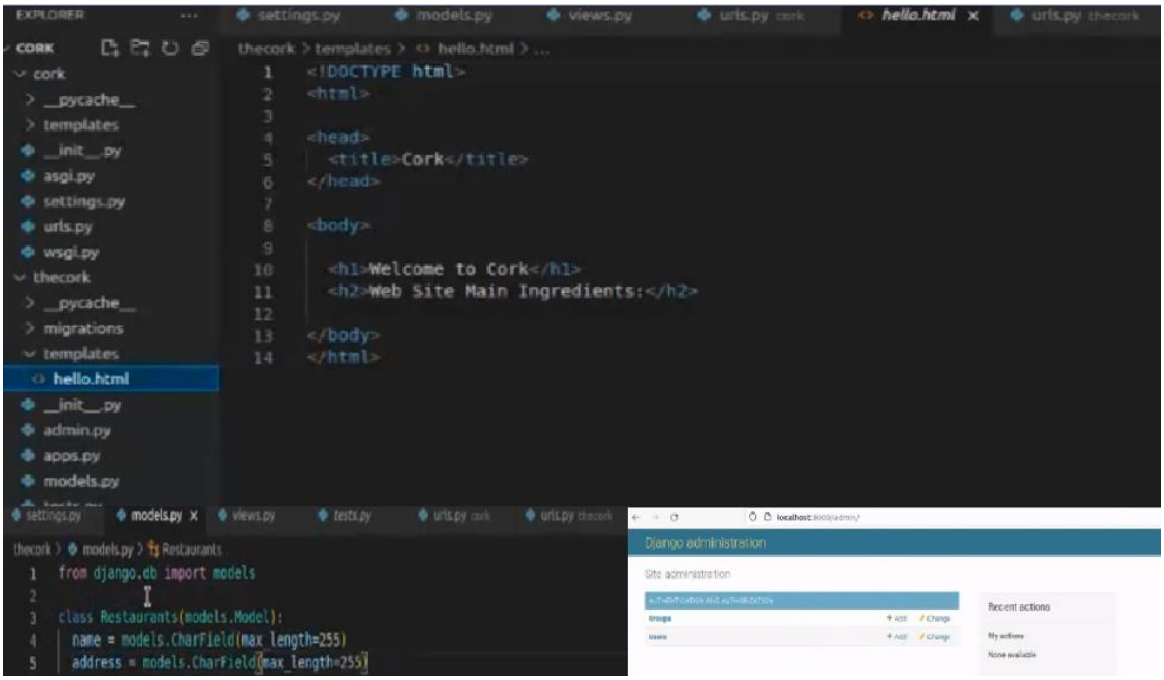

Figure 4 - Django Overview

2. Channels Report

I. SECURE COMMUNICATIONS

The group plans to configure two secure channels.

1.Between the users and the WebServer, located in a DMZ, through a firewall (initially assumed a software firewall in the WebServer but a new infra-structure will be projected to add a VM server working exclusively as firewall and gateway).

2.Between the WebServer and the DatabaseServer, through a Django connector that manages all database operations.

A login function is already implemented that ensures confidentiality and restricts the functions that each user can access. There are two user groups: Owner (for restaurant owners, allowing CRUD operations over an Owner's Restaurants) and Client (for restaurant clients, which allow CRUD operations over a Client's Reservations).

The aim is to implement TLS (transport layer security) on all the channels to ensure data confidentiality and integrity.

One goal is to ensure that all communication between the clients and the webserver (via browser) is made over HTTPS protocol, instead of HTTP.

Another goal is to ensure that all queries made from the WebServer to the DatabaseServer are safe. By default, Django uses query parametrization, preventing SQL Injections. The WebServer-DB connector can be implemented over SSL.

The implementation of TLS leads to the necessity of a Certificate Authority (CA) service on one of the servers (or a dedicated one), that issues certificates for establishing secure connections. The CA server will provide users with certificates based on their public key and the CA private key. One of the options is setting the Django server as a certificate authority, using python cryptography to encrypt data.

References

Security in Django - https://docs.djangoproject.com/en/4.1/topics/security/

Django Databases - https://docs.djangoproject.com/en/4.1/ref/databases/#mysql-notes