NebulaRAT

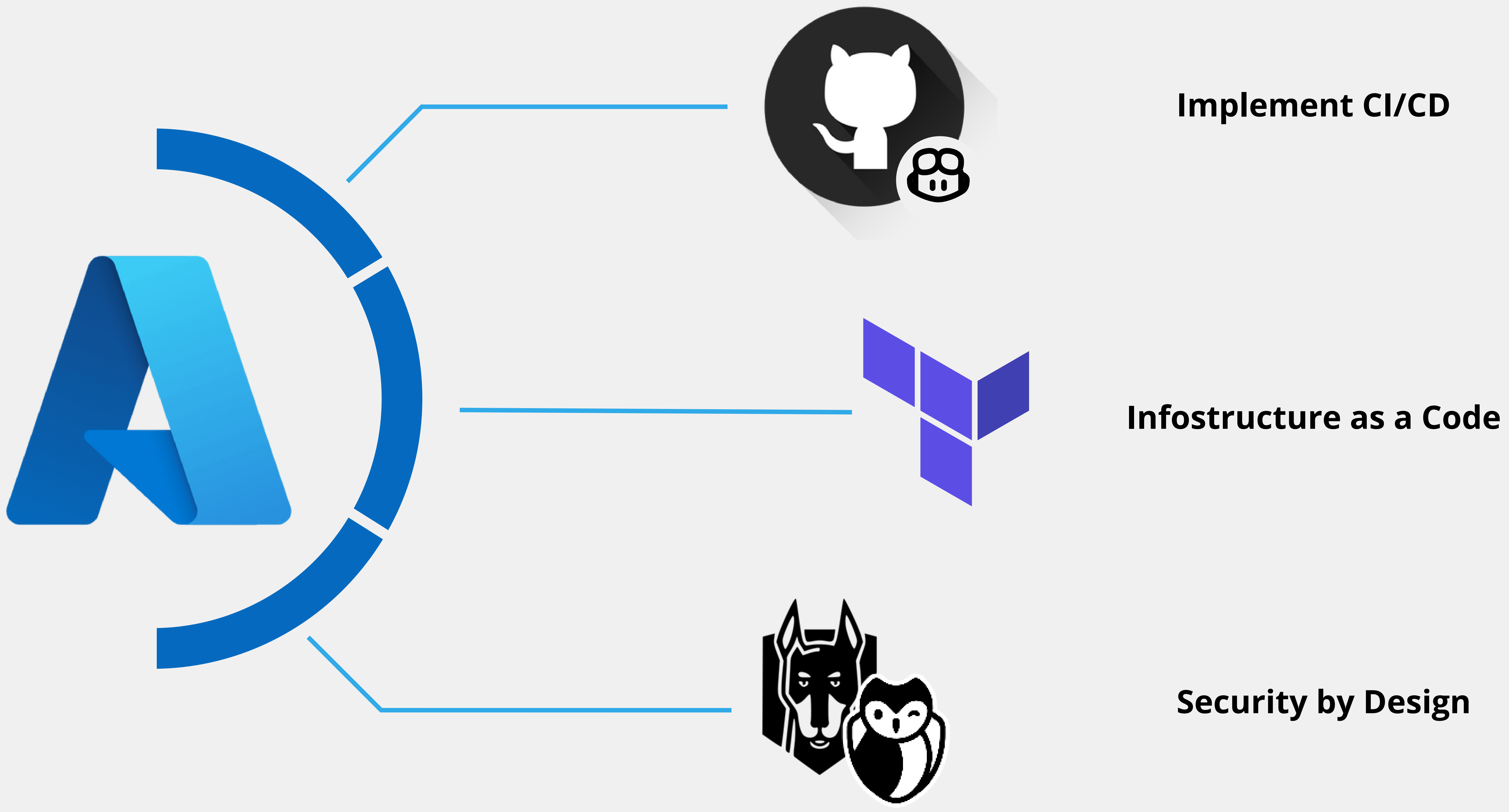
Remote Access Tool

Hosts:

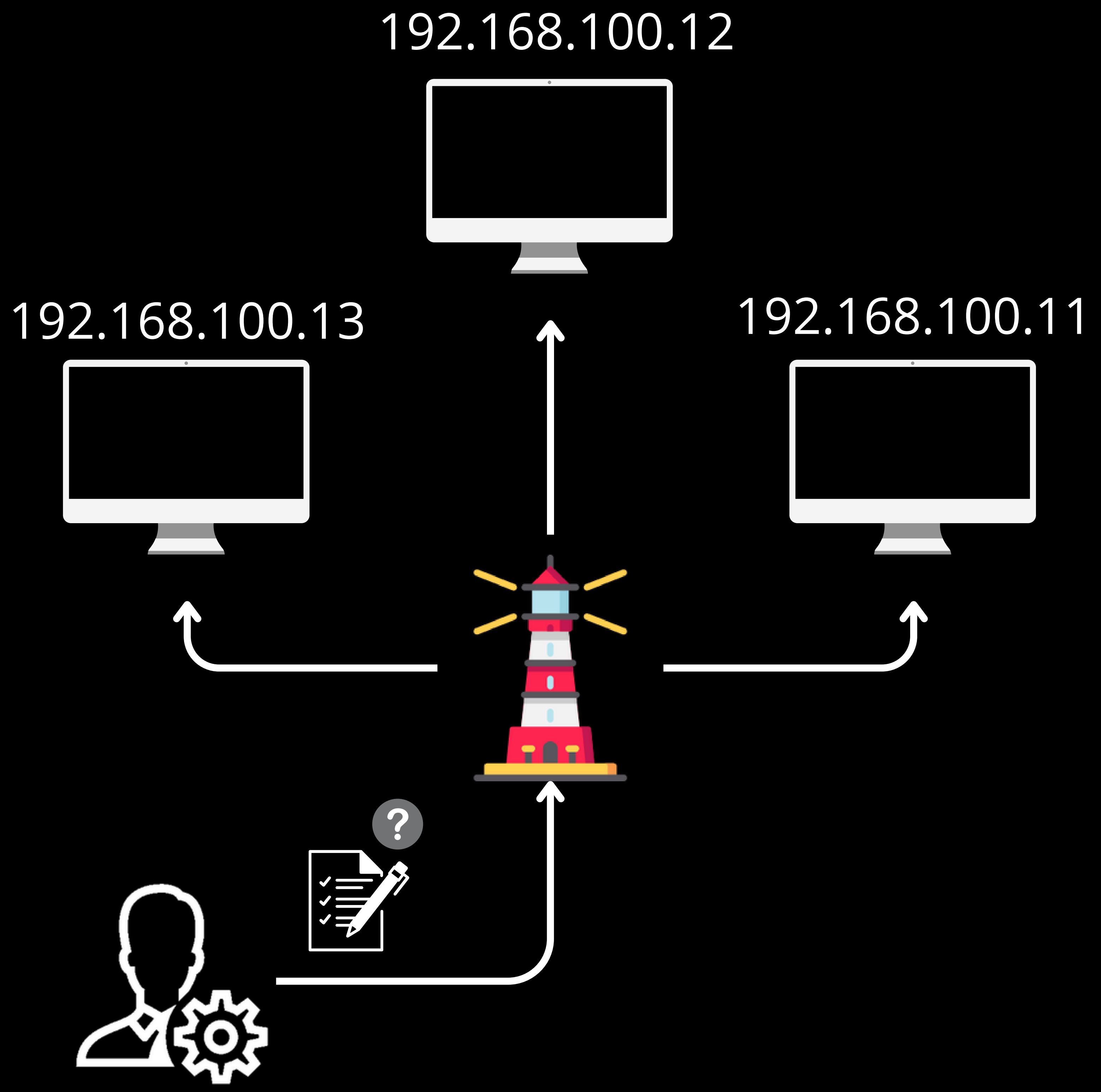
Luca Lombardi,
Stefano Paparella,
Pasquale Basta

Date:
07/19/2024

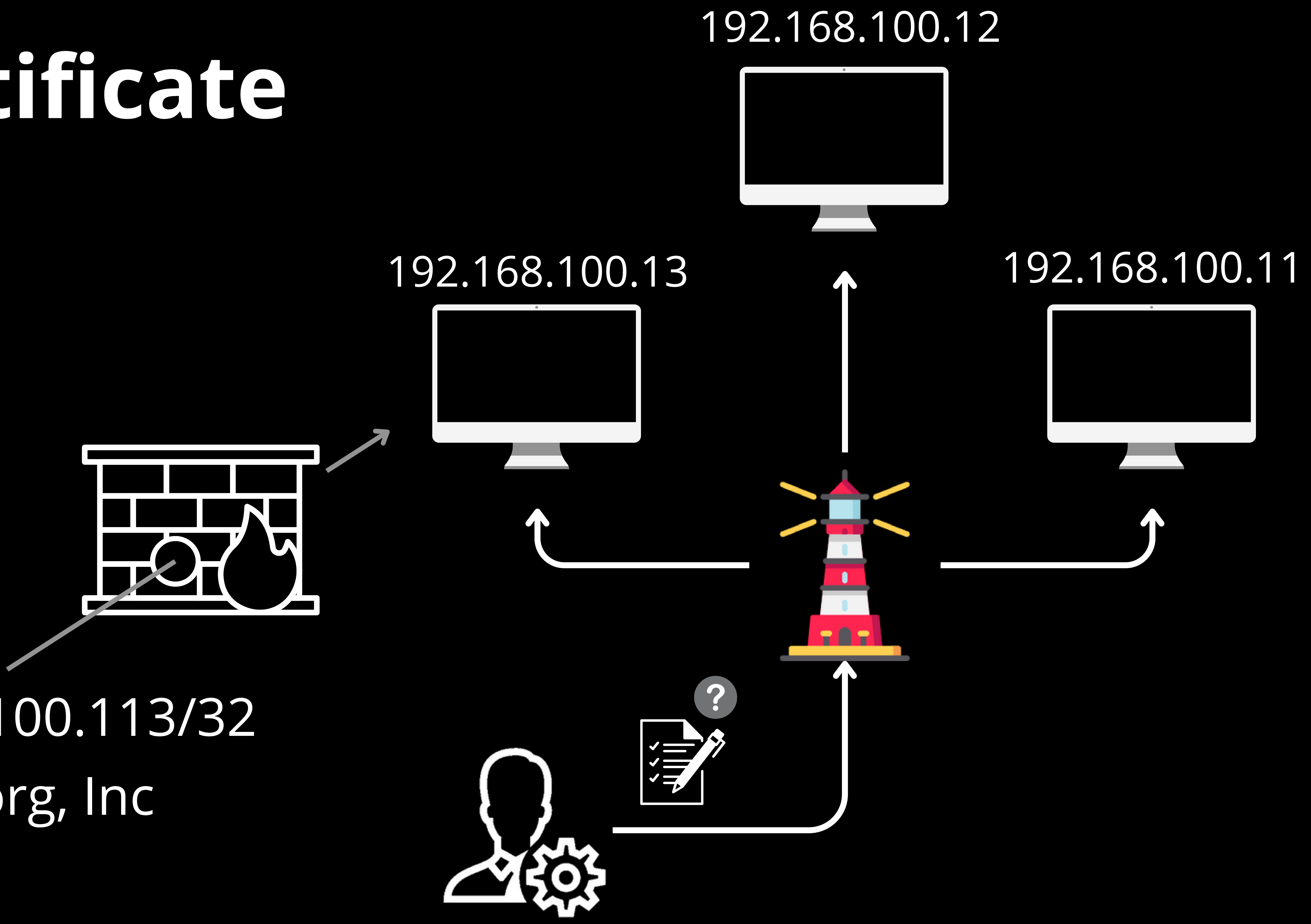
Technology Used

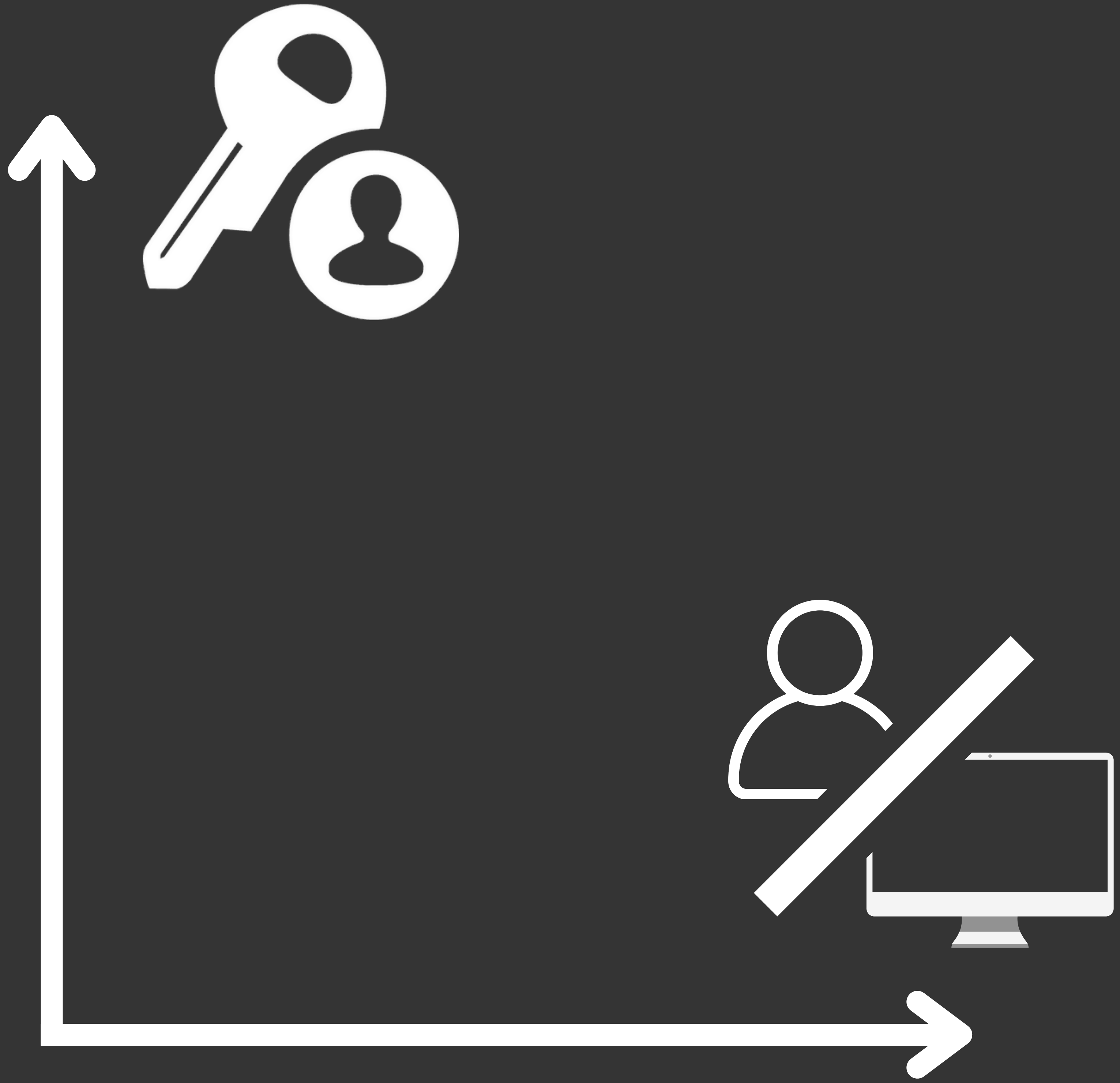


Short-lived certificate generation



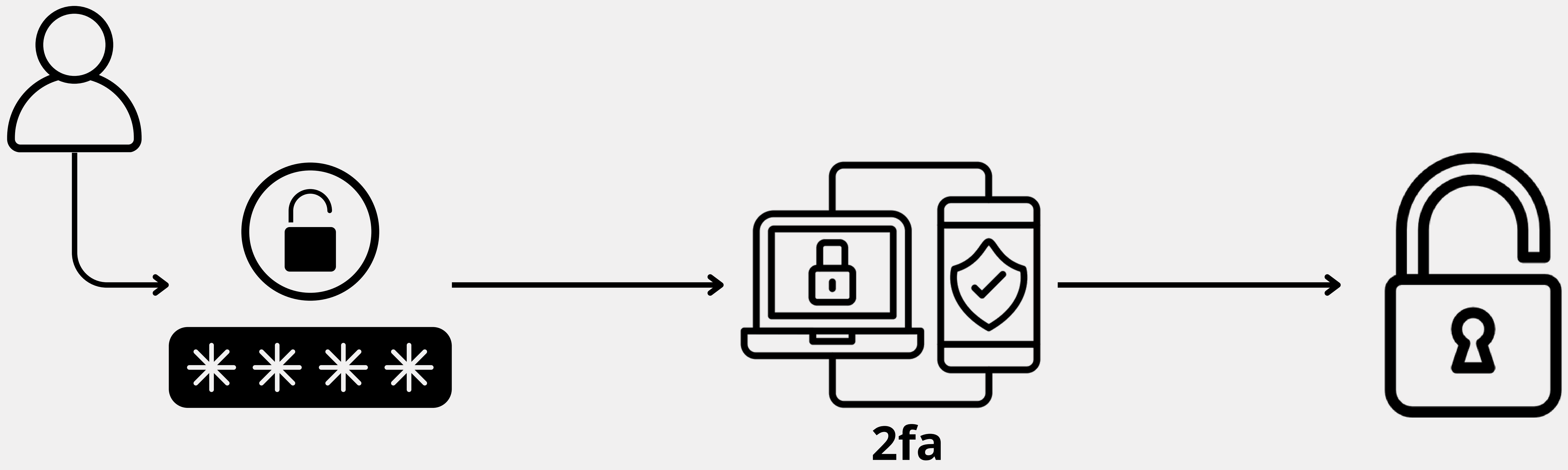
Short-lived certificate generation



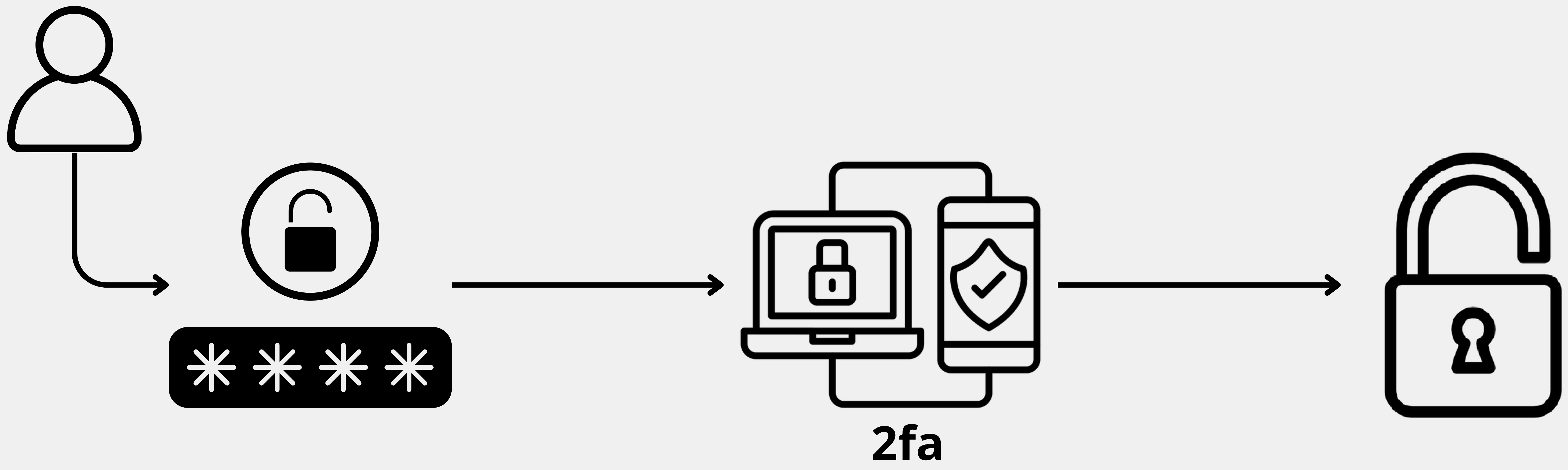


Roles and permission

Authentication

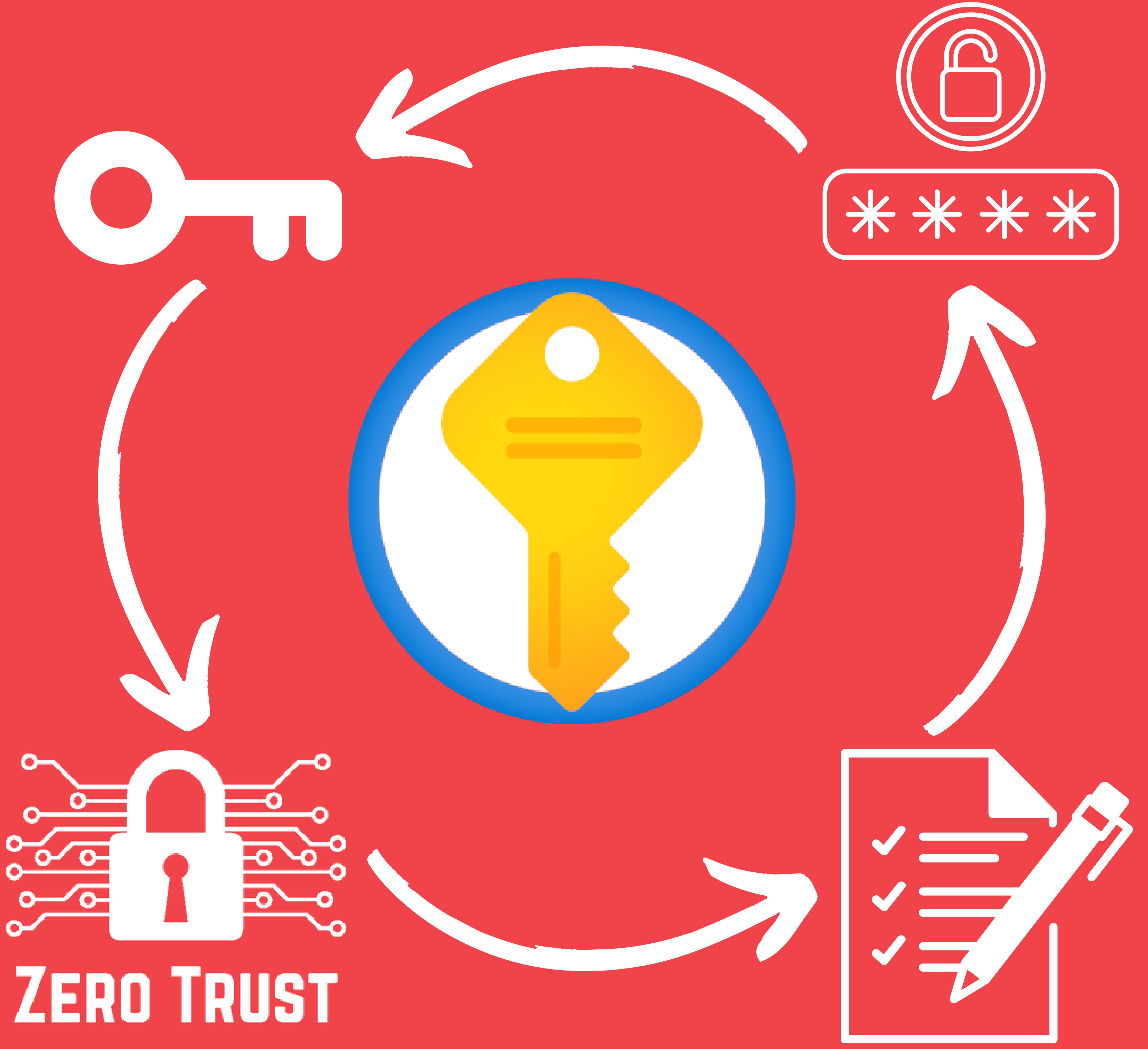


Authentication



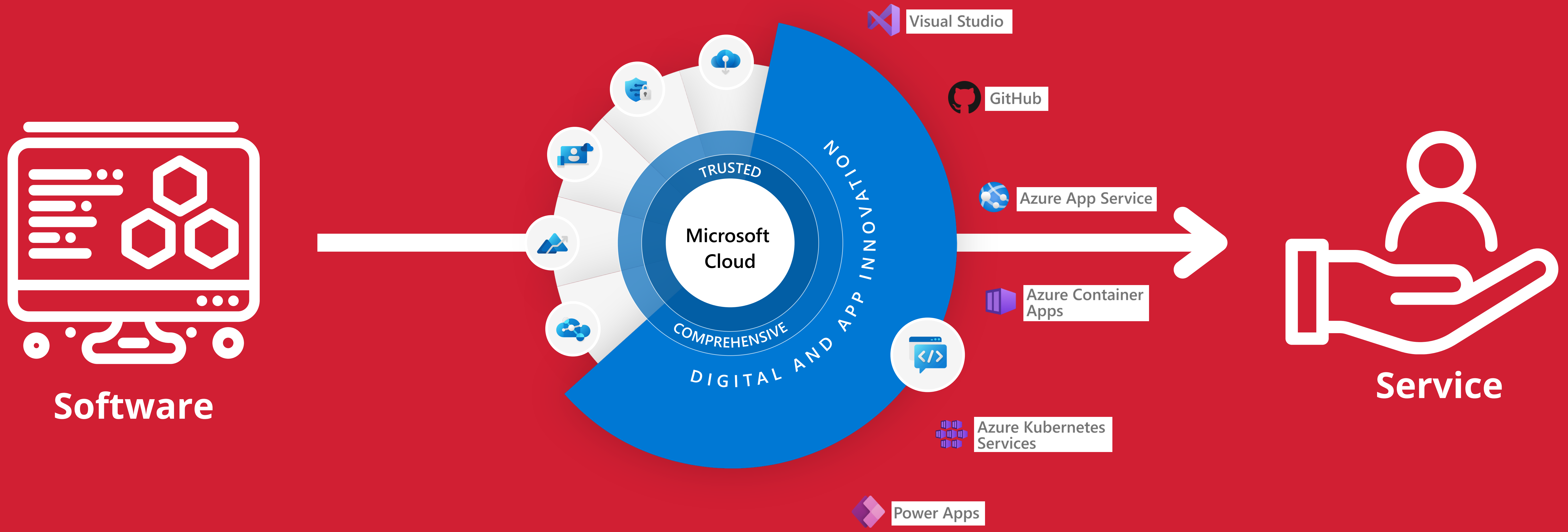
What about Zero Trust?

Managing secrets



Azure Cloud Service

Was it worth it?



Database, important features and security issues

Which DB?

PostgreSQL

MongoDB

Azure DB for SQL

Which DB?

PostgreSQL

MongoDB

Azure DB for SQL

DB IMPLEMENTATION HISTORY

First version

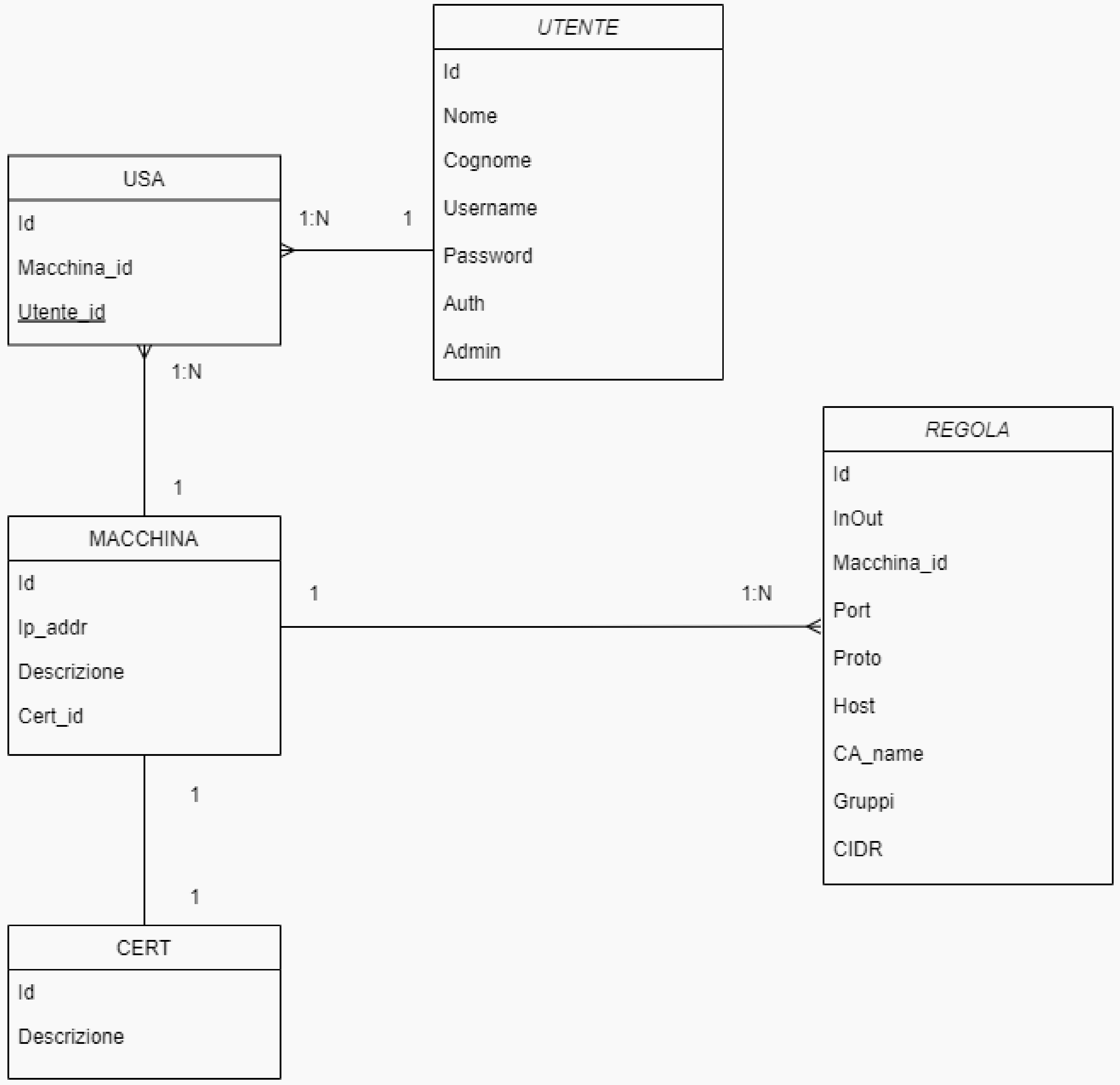
Three tables, only for testing

Second version

Introduced CERT, CONF and REGOLA tables, but not optimised.

Final version

Optimised DB, addedd AUTH and ADMIN fields.



DB Connection Framework

Psycopg2

SQLAlchemy

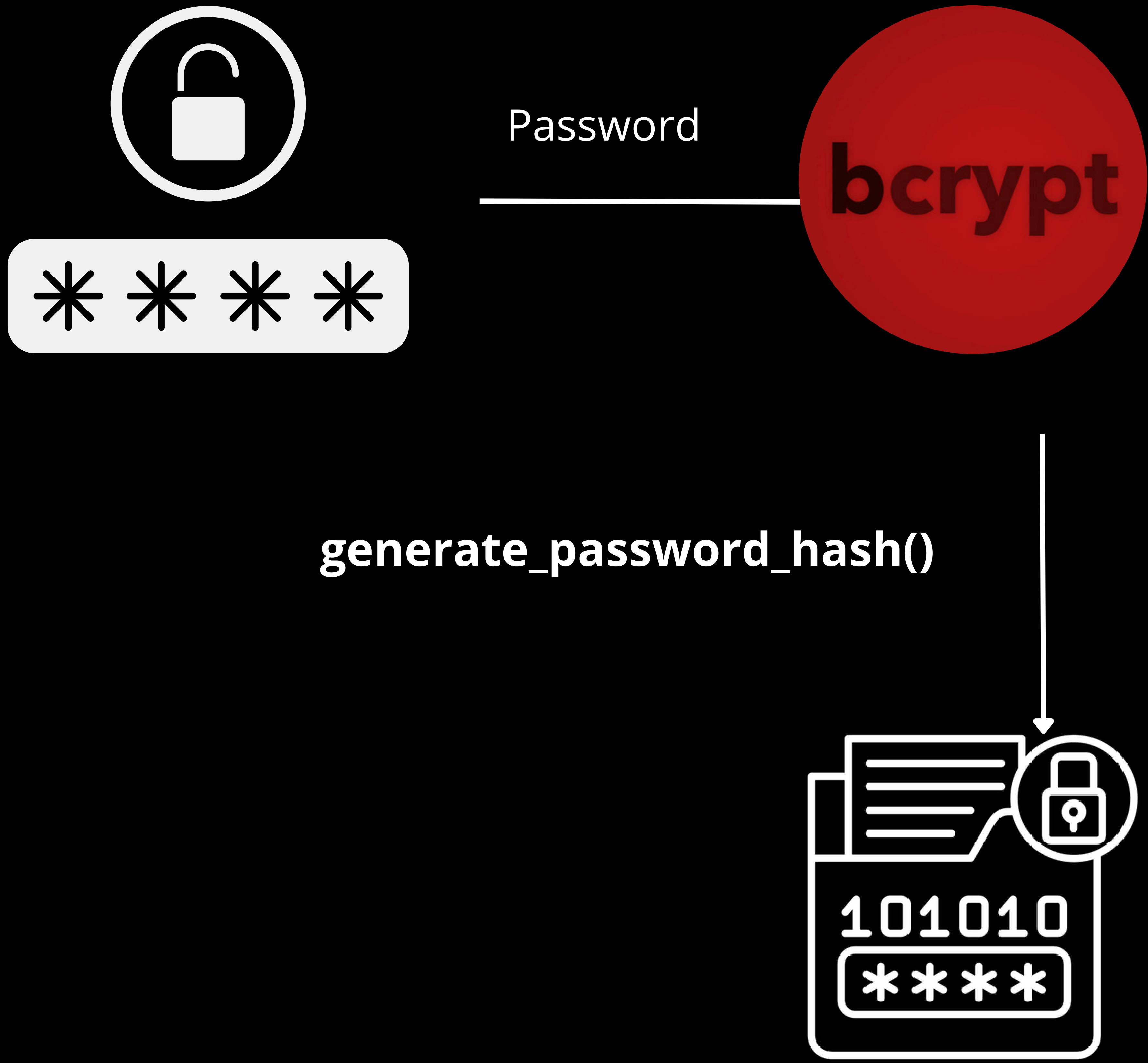
DB Connection Framework

Psycopg2

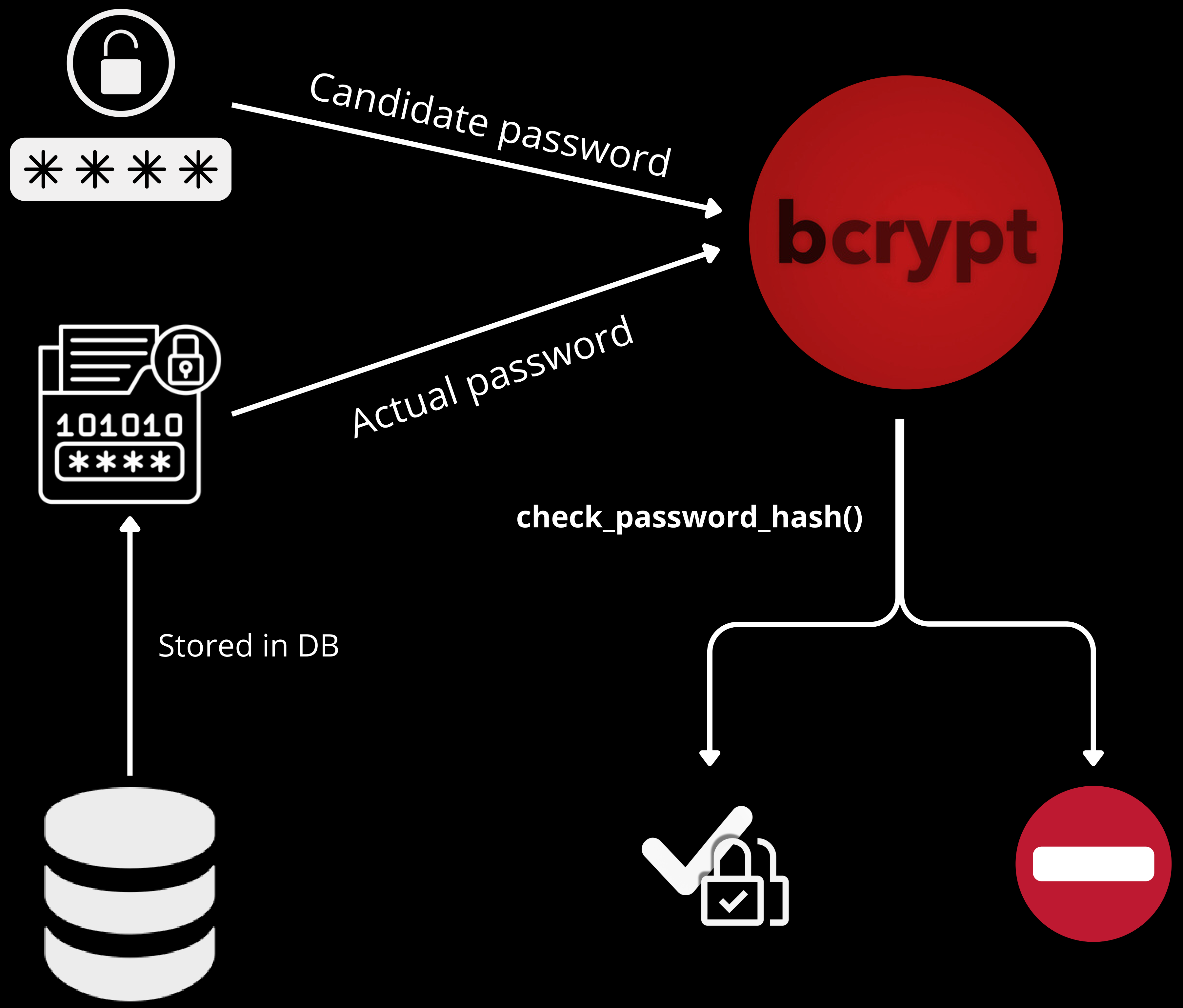
SQLAlchemy

Password management

- Based on Blowfish
- Slower but Safer
- Easy to use



Password storage



Password check

Downloading Short-Lived Certificate & Keys

First idea: use of Tkinter

- user chooses **where to store files**
- *apt-get install* required for it $\longrightarrow$ **NOT POSSIBLE**

$$\Big\Downarrow$$

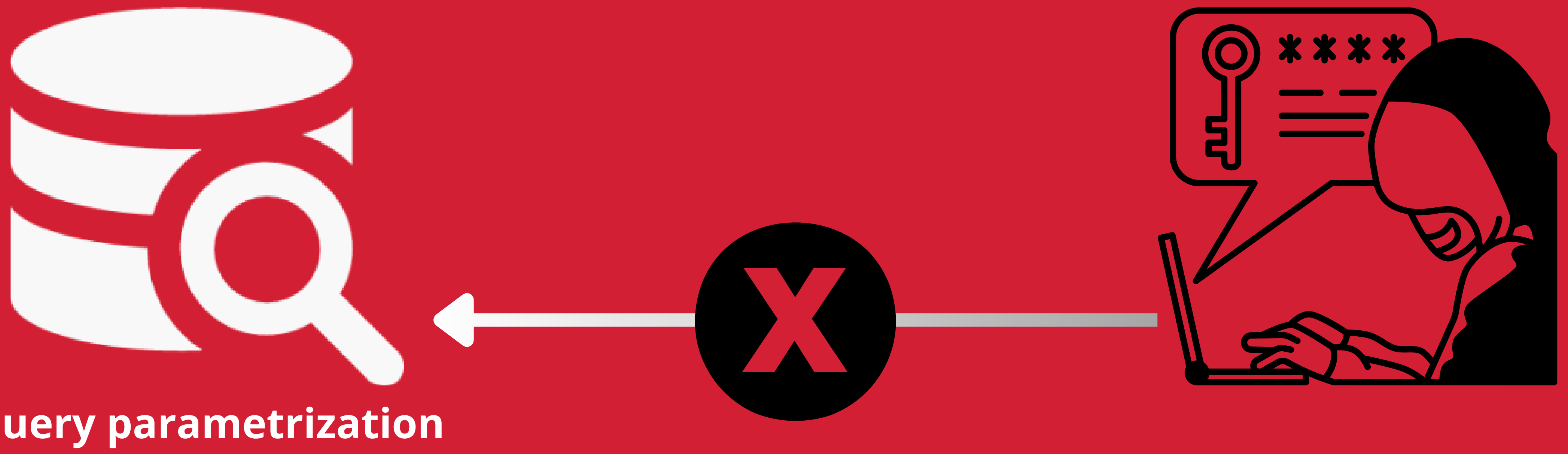
SOLUTION: Flask *send_file()* method

- downloads directly into **Downloads folder**
- **only one file** at time $\longrightarrow$ **MAKE A ZIP FILE!**

NebulaRAT vulnerabilities

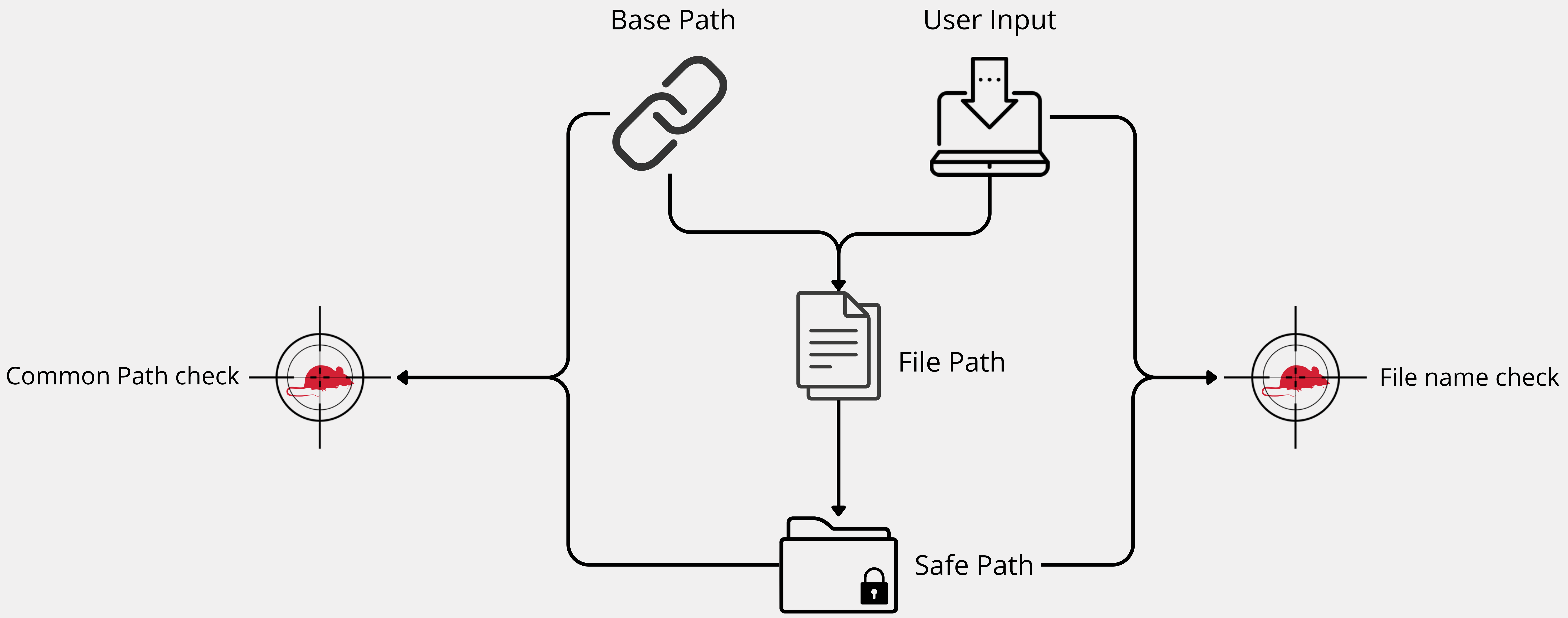
1. Old versions of Python libraries
2. Raw SQL code → **SQL INJECTION**
3. Use of paths → **PATH TRAVERSAL**

How did we solve SQL Injection?



Escapes the input parameters, so that they are **not treated** as SQL statements.

Path Traversal mitigation



Web Interface

FRONT END

HTML

CSS

BACK END

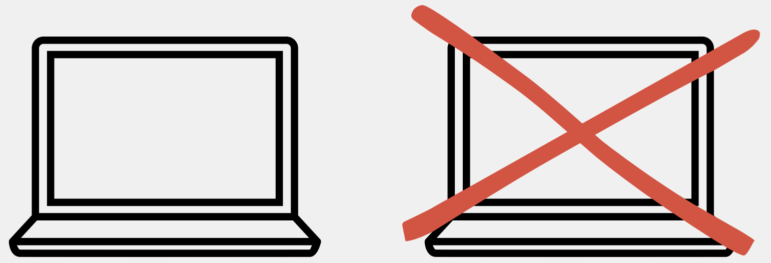
Flask

Why Flask?

- Simplicity and Lightness
- Flexibility
- Azure and Flask
- Steeper learning curve in Django

App Flask

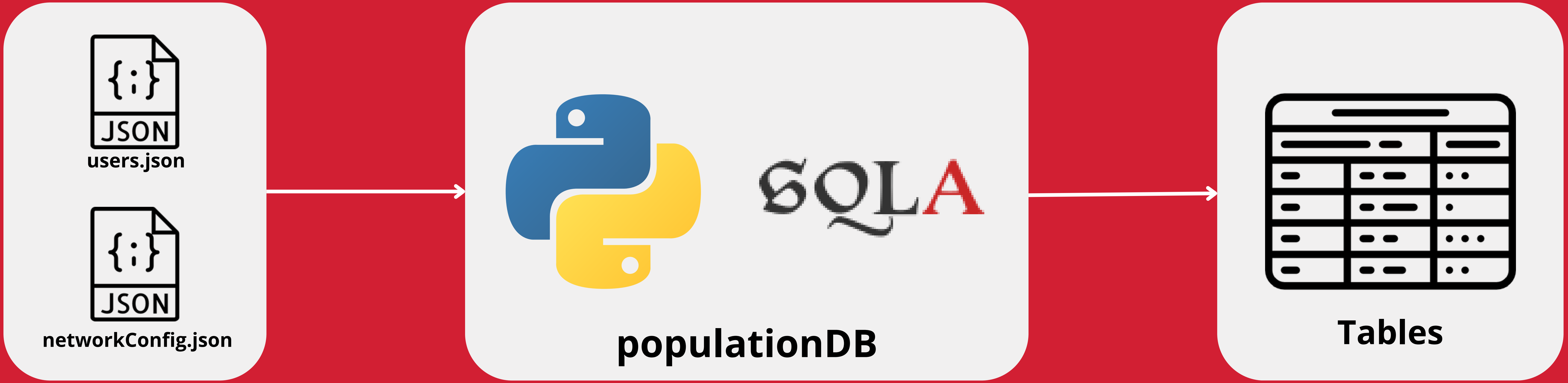
Definition of the route



2 Routes for machine assignment/revocation

2 Routes for the dashboard

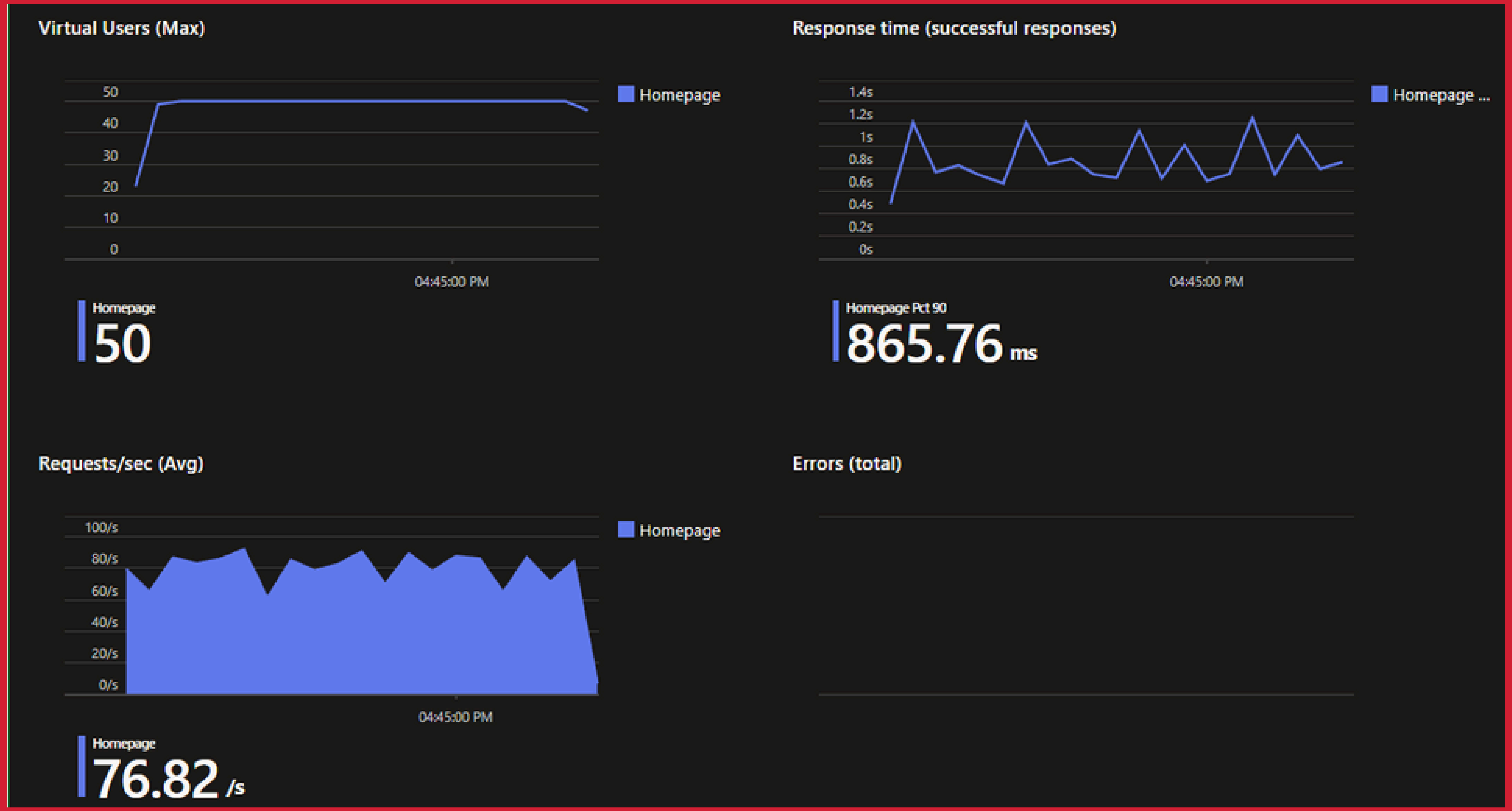
PopulateDB



JSON
users.json
JSON
networkConfig.json
SQLA
populationDB
Tables

Load test and cost analisys

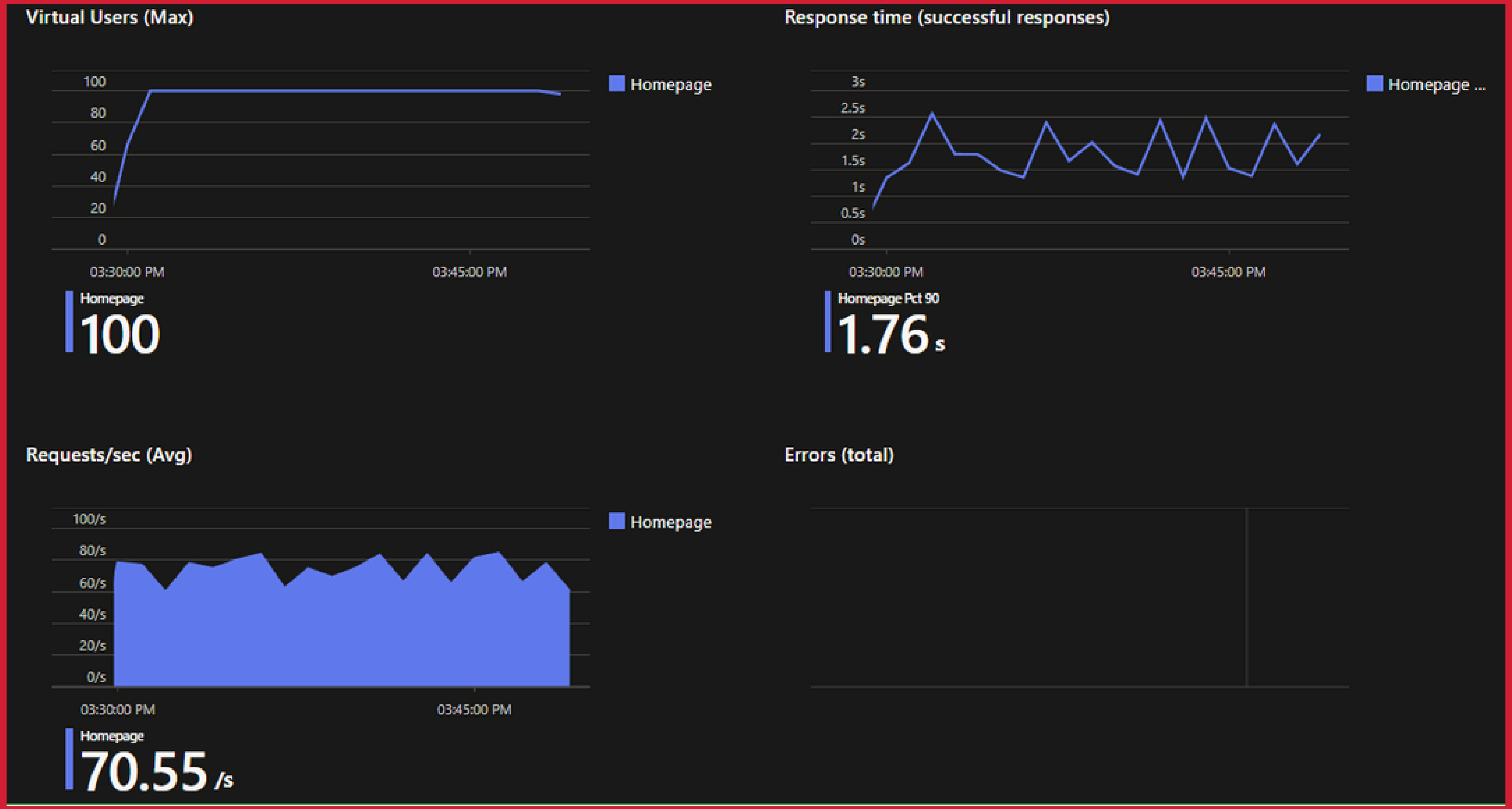
First Basic Test



Homepage 50

Homepage Pct 90 865.76 ms

Homepage 76.82 /s

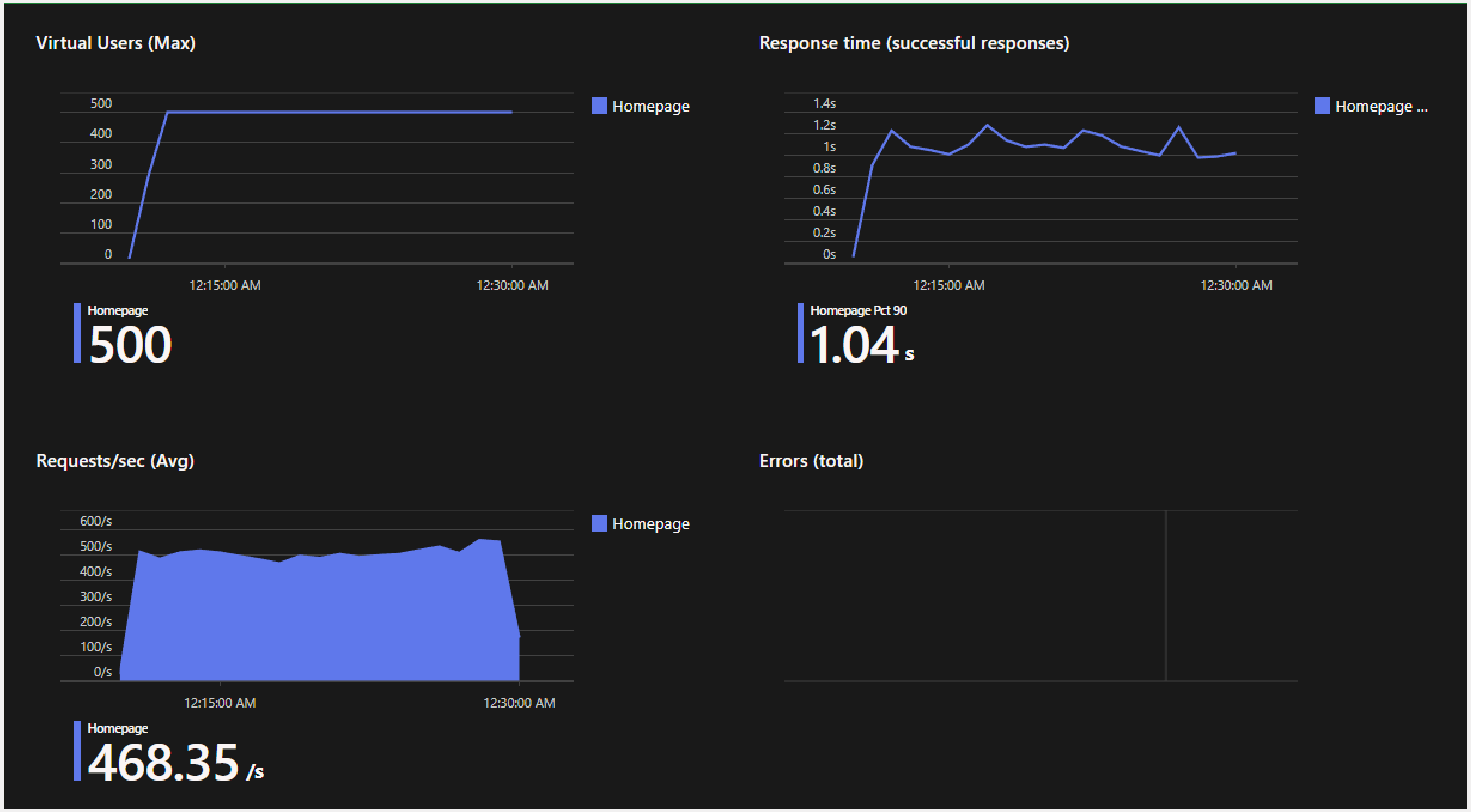


Homepage 100

Homepage Pct 90 1.76 s

Homepage 70.55 /s

Test with premium services



Thanks for your attention!

TRY DEMO