**MANUAL CONFIGURACION APLICACIÓN**

**SMARTCITIES - PARKINGS**

Cliente: Telefónica

Proyecto: Smartcities - Parkings

Versión: 1.0

Autor: José Colina

Contenido

[1. Objeto 3](#_Toc134129759)

[2. Arquitectura / Tecnología 3](#_Toc134129760)

[3. Requerimientos e instalaciones necesarias. 4](#_Toc134129761)

[3.1 Requerimientos previos 4](#_Toc134129762)

[3.2 Instalación backend 5](#_Toc134129763)

[3.3 Instalación del frontend 7](#_Toc134129764)

**Control de Versiones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Responsable** | **Fecha** | **Descripción** |
| 1.0 | José Colina | 01/04/2023 | Primera versión de manual de configuración e instalación de las aplicaciones de smartcities. |
| 1.1 | José Colina | 04/05/2023 | Ajustes en configuraciones de frontend y backend. |
|  |  |  |  |
|  |  |  |  |

# Objeto

El objeto del presente documento es detallar la instalación y configuración del servidor para despliegue de las aplicaciones backend y frontend de SmartCities - Parking

# Arquitectura / Tecnología

La arquitectura está compuesta por las siguientes tecnologías:

* Net Core
* SQL Server
* Vuejs

La interfaz de frontend hace uso de VUEjs conectada por medio de API REST al backend con tecnología .NET CORE.

Tecnología Aplicación Web.

La aplicación está desarrollada usando .Net Core Framework versión 3.1 para el backend y documentación OPEN API y para el FrontEnd VUEjs 2

# Requerimientos e instalaciones necesarias.

Tanto los artefactos de backend y frontend necesita de un servidor Linux Centos versión superior a 8.

# Requerimientos previos

* Instalación de git, manejador de repositorios de fuentes para realizar el clone de los repositorios frontend y backend.
  + sudo dnf install git -y
* Instalación de servidor apache con módulos para proxy reverso
  + sudo yum -y install httpd mod\_ssl
  + sudo firewall-cmd --permanent --add-service=https http
  + sudo firewall-cmd --reload
  + sudo systemctl start httpd
  + sudo systemctl status httpd
  + Creamos el virtualhost en la carpeta /etc/httpd/conf.d/
    - vi /etc/httpd/conf.d/smartcities.conf

<VirtualHost \*:80>

ProxyPreserveHost On

ProxyPass /smartcities http://127.0.0.1:5000/api/1

ProxyPassReverse /smartcities http://127.0.0.1:5000/api/1

ErrorLog /etc/httpd/logs/smartcities-error.log

CustomLog /etc/httpd/logs/smartcities-access.log common

</VirtualHost>

* Crear un directorio para descargar los fuentes de las aplicaciones
  + mkdir smartcities
  + Ingresamos a la carpeta
  + cd smartcities
* Clonamos los repositorios
  + git clone <https://calimatechnologiesSL@dev.azure.com/calimatechnologiesSL/Smart%20Cities/_git/SmartCitiesParkingBackend>
  + git clone <https://calimatechnologiesSL@dev.azure.com/calimatechnologiesSL/Smart%20Cities/_git/SmartCitiesParkingFrontend>

Usuario: usuario\_del\_repositorio

Password: password

# Instalación backend

Instalaciones necesarias para el backend

* Instalación del runtime de .NET CORE 3.1
  + [dnf](https://www.server-world.info/en/command/html/dnf.html) -y install dotnet-runtime-3.1 dotnet-sdk-3.1 dotnet-targeting-pack-3.1 dotnet-templates-3.1 dotnet-hostfxr-3.1
* Instalación de SQL SERVER EXPRESS
  + Agregamos los repositorios de microsoft
    - sudo curl https://packages.microsoft.com/config/rhel/8/mssql-server-2019.repo -o /etc/yum.repos.d/mssql-server-2019.repo
    - sudo curl [https://packages.microsoft.com/config/rhel/8/prod.repo -o /etc/yum.repos.d/msprod.repo](https://packages.microsoft.com/config/rhel/8/prod.repo%20-o%20/etc/yum.repos.d/msprod.repo)
  + Instalamos el SQL SERVER EXPRESS
    - sudo dnf -y install mssql-server
    - sudo yum -y install mssql-tools unixODBC-devel
    - Se acepta la licencia escribiendo YES
    - Verificamos la instalación
      * rpm -qi mssql-server
      * rpm -qi mssql-tools
    - Inicializamos el SQL SERVER
      * sudo /opt/mssql/bin/mssql-conf setup
      * Seleccionamos la opción 3(EXPRESS).
      * Aceptamos la licencia
      * Seteamos el password de administrador del SQLSERVER sa(Colocar aquí el password que luego se utilizará para realizar la conexión a la base de datos).
      * Verificamos que el status del servicio de SQL SERVER
        + systemctl status mssql-server.service
      * Agregamos al PATH la configuración
        + echo 'export PATH=$PATH:/opt/mssql/bin:/opt/mssql-tools/bin' | sudo tee /etc/profile.d/mssql.sh
        + source /etc/profile.d/mssql.sh
      * Si el firewall se encuentra activo, realizamos la configuración para recibir conexiones en el puerto default del SQL SERVER
        + sudo systemctl start firewalld
        + sudo firewall-cmd –reload
      * Realizamos una conexión de test
        + sqlcmd -S localhost -U SA
      * Realizamos del restore con el siguiente comando (Archivo de backup debe colocarse en /var/opt/mssql/data)
        + sqlcmd -S localhost -U SA -Q "RESTORE DATABASE [smartcities] FROM DISK = N'/var/opt/mssql/data/sqldb-smartcities-dev.bak' WITH REPLACE"
* Configuraciones para el backend
  + Generamos el artefacto del backend para ejecutarlo
    - Nos ubicamos en la carpeta del repositorio del backend (SmartCitiesParkingBackend)
    - Editamos el archivo
      * sudo vi SCParking.API/appsettings.json
        + Definimos la cadena de conexión

"connectionStrings": {

"defaultConnection": "Server=localhost,1433;Initial Catalog=smartcities;User ID=UsrSmartCity;Password=Usr$m4rt;"

},

* Si la base de datos se encuentra en el mismo servidor la configuración actual bastaría para funcionar o sino habría que cambiar con los valores correctos.
  + - Ejecutamos el siguiente comando
      * sudo dotnet publish --configuration Release -o /var/www/smartcities/back
    - Configuramos el servicio de Kestrel para la ejecución del backend
      * sudo vi /etc/systemd/system/kestrel-smartcities.service
      * El contenido del archivo sería

[Unit]

Description=Example .NET Web API App running on CentOS 8

[Service]

WorkingDirectory=/var/www/smartcities/back

ExecStart=dotnet /var/www/smartcities/back/SCParking.API.dll

Restart=always

# Restart service after 10 seconds if the dotnet service crashes:

RestartSec=10

KillSignal=SIGINT

SyslogIdentifier=dotnet-example

User=[usuario con el que se hace la instalación]

Environment=ASPNETCORE\_ENVIRONMENT=Production

[Install]

WantedBy=multi-user.target

* + - sudo systemctl enable kestrel-smartcities.service
    - sudo systemctl start kestrel-smartcities.service
    - sudo systemctl status kestrel-smartcities.service

# Instalación del frontend

* Instalación del nodejs y npm
  + Instalamos nodejs
    - sudo dnf install nodejs
  + Nos ubicamos en la carpeta en la carpeta clonada del repositorio de frontend
  + cd SmartCitiesParkingFrontend
  + Ejecutamos los comandos
    - npm install
    - Editamos el archivo .env
      * sudo vi .env
      * Cambiamos el valor de VUE\_APP\_BASE\_URL a la ip del servidor http://<ipdelservidor>/smartcities.
    - sudo dnf install @nodejs
    - curl --silent --location https://dl.yarnpkg.com/rpm/yarn.repo | sudo tee /etc/yum.repos.d/yarn.reposudo rpm --import https://dl.yarnpkg.com/rpm/pubkey.gpg
    - sudo rpm --import <https://dl.yarnpkg.com/rpm/pubkey.gpg>
    - sudo dnf install yarn
* Preparamos la aplicación frontend para producción
  + npm run build
  + Esto creará una carpeta dist la cual la copiamos en /var/www/smartcities
  + Habilitar conexión al apache
    - sudo /usr/sbin/setsebool httpd\_can\_network\_connect true
  + Setear el content root del apache a /var/www/smartcities/dist
  + Cambiamos el documentRoot del archivo
    - sudo vi /etc/httpd/conf/httpd.conf
    - Se cambia el documentroot de /var/www/html por /var/www/smartcities/dist