

SCADA Supervision System - Máquina Dosificadora IoT

Sistema de supervisión para máquina dosificadora basado en PyQt6 y ThingsBoard.

❖ Descripción

Aplicación de escritorio que muestra un dashboard de ThingsBoard para monitorear y supervisar una máquina dosificadora IoT. La interfaz está optimizada para eliminar elementos de navegación innecesarios y proporcionar una vista de pantalla completa del dashboard.

Como parte **complementaria** de este trabajo, se ha desarrollado una **aplicación ejecutable (.exe)** para la pantalla **HMI** con el objetivo de **tener un acceso más rápido a la plataforma y mejorar la interacción con el usuario**.

Para ello, se utilizó el lenguaje de programación **Python** y el framework **Qt** a través de la librería **PyQt6**. Esta solución integra un motor de navegación web **Chromium** mediante el módulo **QWebEngine**, lo que permite **incrustar contenido web** dentro de aplicaciones Qt para su visualización en una ventana dedicada en la HMI.

🚀 Características

- **Visualización de dashboard ThingsBoard** con ajustes visuales automáticos
- **Persistencia de cookies** para mantener la sesión
- **Pantalla completa optimizada** con eliminación de barras de navegación
- **Configuración mediante variables de entorno**
- **Logging estructurado** para debugging y monitoreo
- **Arquitectura modular** siguiendo principios SOLID y Clean Code
- **Cobertura de tests >85%** siguiendo principios TDD

Dr. Fa

📁 Contenido de la carpeta

- `main.py`: código fuente principal del launcher HMI.
- `requirements.txt`: dependencias del proyecto.
- `icono2-app.ico`: ícono de la aplicación (opcional).
- `.gitignore`: exclusiones para evitar subir artefactos generados (ej. `dist/`, `build/`).
- `README.md`: documentación de ejecución y generación del ejecutable.

Nota: no se recomienda versionar las carpetas `dist/` y `build/` porque se generan automáticamente al compilar con PyInstaller.

✓ Requisitos

- **Sistema Operativo:** Windows 10/11 (x64) o Linux

- **Python:** 3.11 o superior
 - **Dependencias principales:** PyQt6, PyQt6-WebEngine
-

Instalación y Ejecución

Entorno de Desarrollo

Si deseas ejecutar el código fuente directamente o realizar modificaciones:

1. Clonar el repositorio:

```
git clone https://github.com/Gran-Lider/Desarrollo-de-una-Metodologia-de-Integracion-IoT.git  
cd Desarrollo-de-una-Metodologia-de-Integracion-IoT/SCADA
```

2. Crear un entorno virtual (recomendado):

```
python -m venv venv  
# En Windows:  
.\\venv\\Scripts\\activate  
# En Linux/Mac:  
source venv/bin/activate
```

3. Instalar dependencias:

```
pip install -r requirements.txt
```

4. Ejecutar la aplicación:

```
python -m src.main  
# O usando el wrapper de compatibilidad:  
python main.py
```

Configuración

La aplicación puede configurarse mediante variables de entorno:

- **DASHBOARD_URL:** URL del dashboard de ThingsBoard
- **WINDOW_TITLE:** Título de la ventana
- **BROWSER_PROFILE:** Nombre del perfil del navegador (default: "CacheTesis")
- **INJECT_DELAY_MS:** Delay en milisegundos antes de inyectar ajustes visuales (default: 2000)

Ejemplo:

```
export DASHBOARD_URL="https://custom.thingsboard.url/dashboard"
export INJECT_DELAY_MS=3000
python -m src.main
```

🧪 Testing

Ejecutar tests:

```
pytest
```

Con cobertura:

```
pytest --cov=src --cov-report=term-missing
```

Ejecutar análisis de código:

```
# Formateo
black src/ tests/

# Linting
flake8 src/ tests/

# Type checking
mypy src/

# Análisis de seguridad
bandit -r src/
pip-audit
```

🏗 Estructura del Proyecto

```
agro5-scada/
├── src/
│   ├── __init__.py          # Package initialization
│   ├── main.py              # Entry point
│   ├── config.py            # Configuration management
│   └── viewer.py            # Dashboard viewer component
└── tests/
```

```
|   └── __init__.py
|   ├── test_main.py          # Main module tests
|   ├── test_config.py        # Configuration tests
|   └── test_viewer.py        # Viewer component tests
|   └── main.py               # Backward compatibility wrapper
|   (deprecated)
|   └── build.sh              # Build script for generating binaries
|   └── requirements.txt      # Project dependencies
|   └── README.md              # This file
|   └── .github/
|       └── instructions/
|           └── instructions.md # Copilot coding guidelines
|       └── workflows/
|           └── test.yml         # CI/CD pipeline
```

📦 Generación de Binarios Ejecutables

Script de Construcción Automatizado

El proyecto incluye un script bash que facilita la generación de binarios ejecutables para Windows y Linux.

Uso:

```
# Para Windows
./build.sh windows

# Para Linux
./build.sh linux

# Para ambas plataformas
./build.sh both

# Ver ayuda
./build.sh help
```

Dr. Fa

Características:

- ✓ Verifica dependencias automáticamente
- ✓ Limpia artefactos de compilaciones anteriores
- ✓ Soporta ícono para Windows (si existe `icono2-app.ico`)
- ✓ Genera ejecutables de un solo archivo (`--onefile`)
- ✓ Sin ventana de consola (`--noconsole`)
- ✓ Logs estructurados con colores

Salida:

Los binarios se generan en el directorio `dist/`:

- Windows: dist/Máquina Dosificadora IoT.exe
- Linux: dist/Máquina Dosificadora IoT

Generación Manual con PyInstaller

Si prefieres ejecutar PyInstaller manualmente, usa:

Para Windows:

```
pyinstaller --noconsole --onefile --clean --icon="icono2-app.ico" --  
name="Máquina Dosificadora IoT" main.py
```

Para Linux:

```
pyinstaller --noconsole --onefile --clean --name="Máquina Dosificadora  
IoT" main.py
```

Parámetros:

- **--noconsole**: Sin ventana de consola
- **--onefile**: Genera un solo ejecutable
- **--clean**: Limpia cache antes de compilar
- **--icon**: Ícono de la aplicación (solo Windows)
- **--name**: Nombre del ejecutable

Distribución Portable

El ejecutable generado es completamente portable y puede:

- ✓ Ejecutarse desde cualquier ubicación (USB, disco duro, red)
- ✓ No requiere instalación de Python en el equipo destino
- ✓ Incluye todas las dependencias necesarias
- ✓ Funciona en cualquier equipo con el mismo sistema operativo

Vista del ejecutable



Máquina Dosificadora IoT.exe