ISPC
INSTITUTO SUPERIOR
POLITÉCNICO CÓRDOBA

TECNICATURA SUPERIOR EN

Telecomunicaciones

PROYECTO INTEGRADOR I

Capa de Procesamiento

Implementación

conceptual del Middleware

Dirección General de
EDUCACIÓN TÉCNICA Y
FORMACIÓN PROFESIONAL

Ministerio de
EDUCACIÓN

CÓRDOBA
Seguimos haciendo

ÍNDICE

Estructura del Repositorio

- **A** requisitos
- **B** investigación
- **C** prototipo
- **capa_de_analisis**:
    - `app.py`: Archivo principal de la aplicación, donde se inicializa Flask y se registran las rutas API.
    - `db_config.py`: Contiene la configuración de la base de datos MySQL, incluyendo las conexiones y parámetros de acceso.
    - `models.py`: Define los modelos de datos (tablas) para el sistema usando SQLAlchemy.
    - `routes`: Carpeta que contiene todos los Blueprints que manejan las diferentes rutas de la API.
    - `lectura.py`: Maneja las lecturas captadas por cada dispositivo.
    - `dispositivos.py`: Gestión de dispositivos registrados en el sistema.
    - `seguridad.py`: Implementa mecanismos de autenticación y autorización.
    - `cliente.py`: Gestiona la información de los usuarios registrados en el sistema.
    - `requirements.txt`: Lista de todas las dependencias del proyecto (incluyendo Flask y SQLAlchemy).

    - `Dockerfile`: Archivo que contiene las instrucciones para construir y desplegar la aplicación en un entorno Docker.
- **D** presentación
- **E** recursos

Stack Tecnológico

El stack tecnológico utilizado en este proyecto incluye:

- **Lenguaje de programación**: Python
- **Framework web**: Flask
- **Base de datos**: MySQL
- **ORM**: SQLAlchemy
- **Plataforma de despliegue**: Docker
- **Microcontrolador**: ESP32 (para la conexión IoT)
- **Sensor**: CNY 70 (para la captura de datos de glucosa)
- **Panel táctil**: Para la interacción del usuario con el dispositivo
- **Plataforma en la nube**: Conexión a un servidor brindado por el profesor.
    - **Comando**: `ssh opalo@gonaiot.com`
    - **Password**: `opalo`

Middleware

¿Qué es?

El desarrollo de dispositivos IoT está en auge, y una parte esencial en este ecosistema es el **middleware**, que actúa como un intermediario entre los dispositivos físicos y las aplicaciones de usuario o los sistemas backend. En este contexto, **Flask** se convirtió en una herramienta popular para implementar middleware debido a su simplicidad, flexibilidad y capacidad para manejar servicios web ligeros.

¿Cuál es su función?

En IoT, su función principal es facilitar la comunicación, gestión y control de los dispositivos conectados a la red. En lugar de que cada dispositivo tenga que comunicarse directamente con el servidor o la aplicación final, el middleware proporciona una capa que abstracta estas interacciones, manejando las complejidades de la comunicación, la seguridad, la integración de datos, y la escalabilidad.

¿Qué facilita el middleware?

- Gestión de dispositivos: Registrar, monitorear y actualizar dispositivos de manera remota.

- **Recopilación y almacenamiento de datos**: Capturar los datos enviados por sensores o actuadores, almacenarlos y prepararlos para su análisis.

- **Autenticación y seguridad**: Garantizar que solo dispositivos autorizados puedan enviar o recibir información.

- **Interoperabilidad**: Traducir diferentes protocolos y estándares para que dispositivos heterogéneos puedan interactuar entre sí.

¿Cómo lo implementaríamos?

Para la capa de procesamiento se debía integrar el middleware a nuestro proyecto de medidor de glucosa.

Los objetivos a lograr en tal capa eran los siguientes:

Unidad 6: Capa de Procesamiento en la Nube

- Fundamentos de procesamiento en la nube
- Implementación de servicios en la nube para almacenamiento y procesamiento de datos
- Middleware y adaptación de la capa de almacenamiento para procesamiento avanzado en la nube
- Integración del procesamiento en la nube en el sistema IoT

Funcionalidades clave que debe tener el middleware

- **Recepción de datos de los dispositivos Edge**: El middleware debe recibir las lecturas de glucemia y otros datos relevantes de los dispositivos en tiempo real.

- Transformación y adaptación de datos: Los datos que llegan desde los dispositivos necesitan ser adaptados antes de ser almacenados o enviados a la API.

- Convertir los datos de formato hexadecimal o binario a un formato legible (como JSON).
- Normalizar los valores de las mediciones de glucemia para asegurar que estén dentro de los rangos aceptados.
- Transformar los datos para almacenarlos en la Base de Datos SQL del proyecto.

- Validación y autenticación: Verificar que los datos provengan de dispositivos autorizados, especialmente si se maneja información sensible como la glucemia de pacientes.

- Gestión de eventos y alertas: Si la glucemia está fuera de los niveles normales, el middleware puede generar una alerta que luego se envíe al Contacto de Emergencia o al Historial Médico.

- Interacción con la API RESTful: Configurar el middleware para que sirva de interfaz entre los datos que recibe y las rutas de la API RESTful. Así, el middleware toma los datos de los dispositivos, los adapta y los envía a la API.

¿Qué logramos implementando un middleware?

En nuestro proyecto de sistema IoT para **monitoreo de glucosa**, el **middleware** cumple funciones clave para asegurar que los datos críticos de los sensores de glucosa lleguen a la **nube** para su **procesamiento y análisis**. Además, este middleware se encarga de **adaptar y normalizar** los datos antes de almacenarlos, verificando que provengan de dispositivos **autenticados y autorizados**, lo cual es fundamental dada la **sensibilidad** de la información médica manejada.

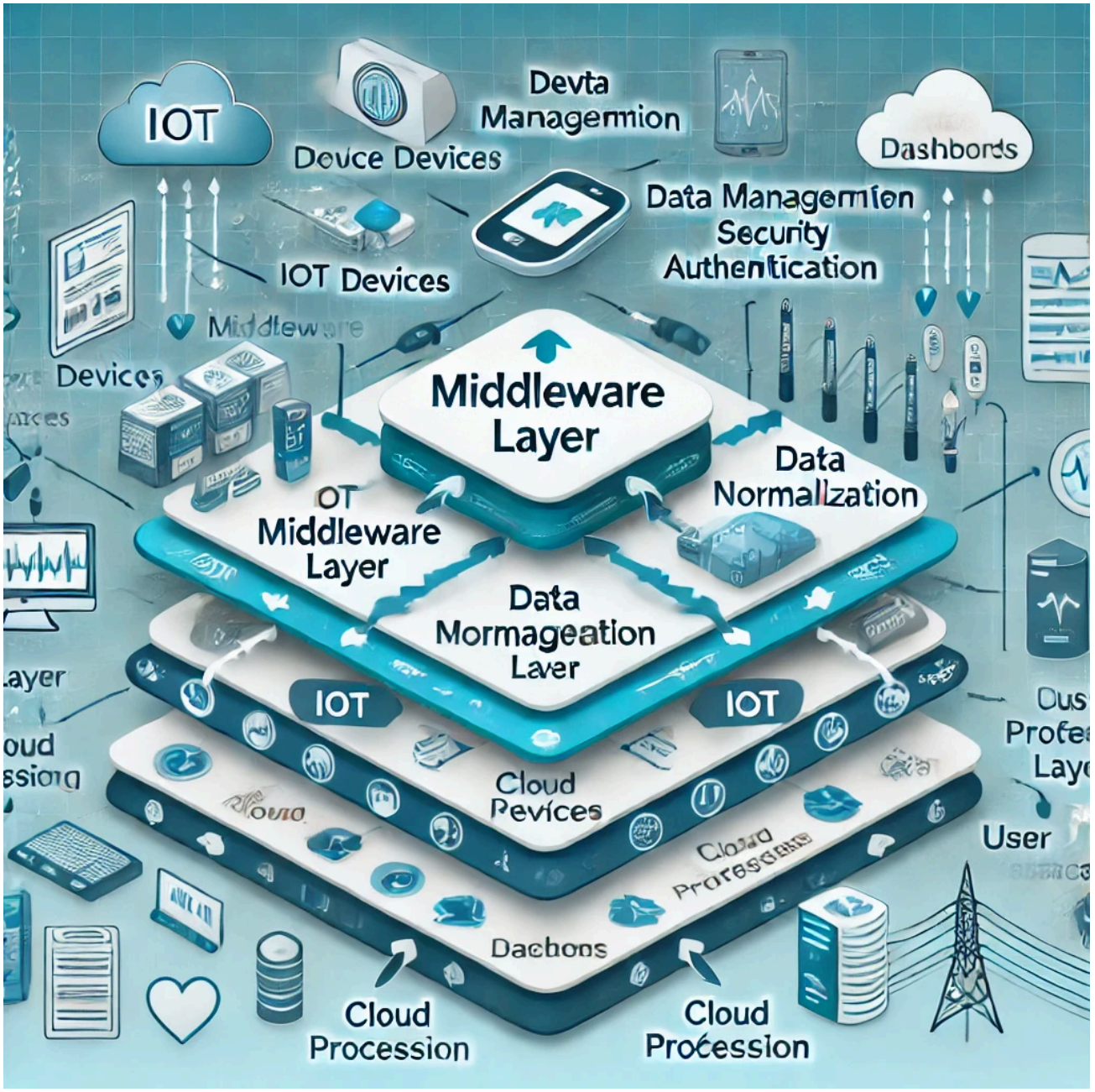
Al integrar middleware en la **capa de procesamiento** de nuestro sistema, logramos:

Gestión de Dispositivos y Seguridad: Facilitamos el registro y monitoreo de dispositivos en tiempo real.

Normalización y Almacenamiento de Datos: Adaptamos los datos en formatos como JSON para su almacenamiento adecuado y análisis posterior.

Interoperabilidad: Hacemos posible que distintos componentes del sistema, como sensores y la base de datos, se comuniquen fluidamente.

Con esta estructura, nuestro sistema de monitoreo de glucosa no solo captura datos precisos y en tiempo real, sino que también asegura una gestión eficaz de la información y su acceso controlado, fortaleciendo así la confiabilidad y escalabilidad del proyecto.


IOT
Devta
Managemion
Dashbords
Data Managemion
Security
Authentication
IOT Devices
Middleware Layer
Data Normallzation
OT Middleware Layer
Data Mormageation Laver
IOT
IOT
Cloud Pevices
User
Dachons
Cloud Procession
Cloud Procession

Funciones

- Conexión al servidor SSH "gonaiot": Esta conexión sería iniciada al arranque del middleware y reutilizada en cada ruta que necesite enviar datos al servidor.

- Función para registrar Cliente: Procesa los datos del cliente, valida que el DNI no exista ya en la base de datos e inserta los datos en la tabla `Cliente` de la base de datos.

- Función para registrar Dispositivo: Comprueba si el cliente ya tiene un dispositivo activo, registra el nuevo dispositivo y actualiza el estado del anterior (si lo hubiera) como retirado, luego, enlaza el dispositivo con el cliente mediante su DNI.

- Función para registrar Lecturas de Glucosa: Valida que el dispositivo esté activo y que el valor de glucosa sea válido, luego inserta los datos en la tabla `Lectura`, asociando la lectura con el dispositivo mediante su MAC.

Explicación de la Implementación

1. **Dispositivos IoT (Sensores de Glucosa)**: Los dispositivos de borde o edge (sensores de glucosa) recolectan datos en tiempo real sobre los niveles de glucosa del paciente. Estos datos son enviados hacia el middleware para su procesamiento inicial.
2. **Middleware**: Esta capa cumple múltiples funciones:

- **Gestión de Dispositivos**: Controla y monitorea los sensores conectados, verificando el estado de cada uno y gestionando sus registros en la red.
- **Normalización y Adaptación de Datos**: Convierte los datos recolectados en formatos estándar (como JSON) y los adapta para ser enviados al almacenamiento.
- **Autenticación y Seguridad**: Garantiza que solo los dispositivos autorizados envíen datos, aplicando capas de seguridad para proteger la información sensible del paciente.
- **Procesamiento de Alertas y Eventos**: Genera alertas cuando los niveles de glucosa están fuera de los rangos normales, notificando a los usuarios y a los proveedores de salud a través de la interfaz de usuario.

3. **Procesamiento en la Nube**: Los datos normalizados y autenticados por el middleware son enviados a la nube para su almacenamiento y análisis avanzado. Esto incluye el uso de bases de datos en la nube y análisis de datos históricos para detectar patrones.
4. **Interfaz de Usuario**: Los datos procesados son accesibles para el usuario final a través de una aplicación que permite a los pacientes y proveedores de salud visualizar los niveles de glucosa, recibir alertas y revisar tendencias históricas.

Este flujo garantiza una arquitectura robusta, en la cual el middleware actúa como un intermediario seguro, eficiente y escalable entre los dispositivos IoT y el almacenamiento y procesamiento en la nube, facilitando así una interacción ágil y confiable para el monitoreo de la salud.

Licencia

MIT License

Referencias Bibliográficas

- Manuales y cuadernillos digitales de Ciencias de la Computación

https://program.ar/material-didactico/

- Unidad 1. Fundamentos de Informática SOPORTE LÓGICO EN UN ORDENADOR PERSONAL: EL SOFTWARE

https://www3.gobiernodecanarias.org/medusa/ecoblog/mgoncal/