

**Universidad**

**Tecnológica de Tijuana**

# **Software Requirements Specification (SRS)**

**Presentado por:**

Bautista Ordoñez Brian Ángel

Garcia Valenzuela Ernesto

Islas Reyes Luis Ivan

Orrantia González Germán Sisac

**Nombre del Proyecto:**

**Zero to Zero-Fatigue Zone**

Fecha: 13 de noviembre del  
2025

---

Tabla de versiones
--------------------

<p><b>Versión:</b> V.1</p> <p><b>Fecha:</b> 10 de Octubre de 2025</p> <p><b>Proyecto:</b> ZZZ</p> <p><b>Tipo:</b> Proyecto Escolar</p> <p><b>Estado:</b> Documento en primera version</p>
---

<p><b>Versión:</b> V.2</p> <p><b>Fecha:</b> 12 de Noviembre de 2025</p> <p><b>Proyecto:</b> ZZZ</p> <p><b>Tipo:</b> Proyecto Escolar</p> <p><b>Estado:</b> Documento Completo</p>
---

---

# TABLA DE CONTENIDOS

1. INTRODUCCIÓN .....	4
1.1 Propósito .....	4
1.2 Alcance del Sistema .....	4
1.3 Definiciones, Acrónimos y Abreviaturas .....	5
1.4 Referencias .....	6
1.5 Visión General del Documento .....	6
2. DESCRIPCIÓN GENERAL .....	7
2.1 Perspectiva del Producto .....	7
2.2 Funciones del Producto .....	7
2.3 Características de los Usuarios .....	9
2.4 Restricciones Generales .....	10
2.5 Suposiciones y Dependencias .....	11
3. REQUISITOS ESPECÍFICOS.....	12
3.1 Requisitos Funcionales Detallados.....	12
3.1.1 Autenticación y Gestión de Usuarios .....	12
3.1.2 Gestión de Dispositivos IoT.....	12
3.1.3 Captura y Almacenamiento de Datos de Sensores .....	12
3.1.4 Procesamiento de Métricas .....	13
3.1.5 Machine Learning y Predicción de Fatiga .....	13
3.1.6 Sistema de Alertas .....	13
3.1.7 Recomendaciones .....	13
3.1.8 Dashboards y Visualizaciones.....	13
3.2 Requisitos de Interfaz Externa .....	14
3.2.1 Interfaces de Usuario .....	14
3.2.2 Interfaces de Hardware.....	14
3.2.3 Interfaces de Software .....	14
3.4 Restricciones de Diseño.....	14
3.5 Atributos de Calidad del Sistema .....	14
3.5.1 Seguridad .....	14
3.5.2 Confiabilidad.....	15

3.5.3 Mantenibilidad .....	15
3.5.4 Usabilidad .....	15
3.5.5 Escalabilidad .....	15
4. APÉNDICES .....	16
4.1 Diagrama de Arquitectura .....	16
4.2 Modelos de Base de Datos (Resumen).....	16
4.3 Endpoints Principales de la API .....	17
4.4 Referencias Técnicas .....	17

## 1. INTRODUCCIÓN

### 1.1 Propósito

Este documento de Especificación de Requisitos de Software (SRS) describe las especificaciones funcionales y no funcionales del **Zero to Zero-Fatigue Zone (ZZZ)**. El propósito de este documento es:

- Definir claramente los requisitos del sistema para desarrolladores, testers y stakeholders
- Servir como base contractual para el desarrollo del proyecto
- Proporcionar una referencia para la validación y verificación del sistema
- Facilitar el mantenimiento y evolución futura del sistema

**Audiencia objetivo:** - Equipo de desarrollo (Backend y Frontend) - Supervisores del proyecto educativo

## 1.2 Alcance del Sistema

**Nombre del Sistema:** Zero to Zero-Fatigue Zone (ZZZ)

### **Descripción:**

El Sistema **ZZZ** es una aplicación web empresarial que permite monitorear en tiempo real el estado de fatiga de empleados mediante dispositivos wearables IoT (ESP32) equipados con sensores biométricos. El sistema utiliza Machine Learning para predecir niveles de fatiga, genera alertas automáticas y proporciona recomendaciones para optimizar rutinas laborales.

### **Objetivos principales:**

1. **Monitoreo en Tiempo Real:** Capturar y visualizar datos biométricos de empleados cada 5 segundos
2. **Predicción de Fatiga:** Utilizar algoritmos de ML (K-Means clustering) para calcular un índice de fatiga (0-100)
3. **Sistema de Alertas:** Notificar automáticamente cuando se detectan niveles críticos de fatiga
4. **Gestión Jerárquica:** Permitir administración de usuarios con roles (Admin → Supervisor → Empleado)
5. **Análisis y Reportes:** Generar dashboards interactivos y reportes de tendencias

6. **Optimización de Rutinas:** Proporcionar recomendaciones basadas en patrones detectados

**Beneficios esperados:**

- Reducción de accidentes laborales relacionados con fatiga
- Mejora en la productividad y bienestar de empleados
- Toma de decisiones basada en datos
- Optimización de turnos y rutinas de trabajo
- Cumplimiento de normativas de seguridad laboral

**Limitaciones del alcance:**

- El sistema NO realiza diagnósticos médicos
- El sistema NO reemplaza evaluaciones médicas profesionales
- La versión inicial soporta 1 dispositivo por empleado (no múltiples dispositivos)

### 1.3 Definiciones, Acrónimos y Abreviaturas

**Términos técnicos:**

Término	Definición
<b>Fatiga</b>	Estado de agotamiento físico o mental que reduce la capacidad de rendimiento
<b>Índice de Fatiga</b>	Valor numérico 0-100 que representa el nivel de fatiga calculado por ML
<b>Ventana de Tiempo</b>	Período de 30 segundos a 5 minutos usado para procesar métricas
<b>HRV</b>	Heart Rate Variability - Variabilidad del ritmo cardíaco

Término	Definición
<b>SpO2</b>	Saturación de oxígeno en sangre (porcentaje)
<b>Wearable</b>	Dispositivo electrónico que se lleva puesto en el cuerpo

#### Acrónimos:

Acrónimo	Significado
<b>SRS</b>	Software Requirements Specification
<b>ZZZ</b>	Zero To Zero-Fatigue Zone
<b>IoT</b>	Internet of Things
<b>ESP32</b>	Microcontrolador de Espressif Systems
<b>MQTT</b>	Message Queuing Telemetry Transport
<b>API</b>	Application Programming Interface
<b>REST</b>	Representational State Transfer
<b>JWT</b>	JSON Web Token
<b>ML</b>	Machine Learning
<b>HR</b>	Heart Rate (Ritmo Cardíaco)
<b>BPM</b>	Beats Per Minute (Latidos por Minuto)
<b>PPG</b>	Photoplethysmography (sensor óptico de ritmo cardíaco)
<b>ECG</b>	Electrocardiograma
<b>RMSSD</b>	Root Mean Square of Successive Differences
<b>SDNN</b>	Standard Deviation of NN intervals
<b>pNN50</b>	Percentage of successive RR intervals that differ by more than 50ms
<b>CRUD</b>	Create, Read, Update, Delete

Acrónimo	Significado
<b>QoS</b>	Quality of Service
<b>UI/UX</b>	User Interface / User Experience

#### Abreviaturas de roles:

- **Admin:** Administrador del sistema
- **Supervisor:** Usuario que gestiona empleados y dispositivos
- **Employee:** Empleado que porta el dispositivo wearable

## 1.4 Referencias

#### Estándares y frameworks:

- Django REST Framework Documentation: <https://www.django-rest-framework.org/>
- React Documentation: <https://react.dev/>
- MQTT Protocol: <https://mqtt.org/>
- PostgreSQL Documentation: <https://www.postgresql.org/docs/>

#### Librerías de ML:

- scikit-learn: <https://scikit-learn.org/>
- pandas: <https://pandas.pydata.org/>
- numpy: <https://numpy.org/>

## 1.5 Visión General del Documento

Este documento SRS está organizado de la siguiente manera:

- **Sección 1 (Introducción):** Proporciona una visión general del propósito, alcance y contexto del sistema
- **Sección 2 (Descripción General):** Describe la perspectiva del producto, funciones principales, usuarios y restricciones



- **Sección 3 (Requisitos Específicos):** Detalla todos los requisitos funcionales y no funcionales del sistema
- **Sección 4 (Apéndices):** Incluye información adicional, diagramas y referencias técnicas

## 2. DESCRIPCIÓN GENERAL

### 2.1 Perspectiva del Producto

El **ZZZ** es un sistema **independiente** desarrollado específicamente para este proyecto, aunque utiliza tecnologías y protocolos estándar de la industria.

#### Interfaces principales:

1. **ESP32 → MQTT Broker:** Dispositivos publican datos de sensores vía protocolo MQTT
2. **MQTT Broker → Django Backend:** Cliente MQTT suscrito recibe y procesa mensajes
3. **Frontend ↔ Backend:** Comunicación REST API con autenticación JWT
4. **Backend ↔ PostgreSQL:** Persistencia de datos y consultas
5. **Backend → ML Models:** Procesamiento de métricas y predicción de fatiga

### 2.2 Funciones del Producto

El sistema proporciona las siguientes funciones principales organizadas por módulo:

#### 2.2.1 Autenticación y Gestión de Usuarios

- **RF-AUTH-001:** Sistema de login/logout con autenticación JWT
- **RF-AUTH-002:** Gestión de roles jerárquicos (Admin, Supervisor, Employee)
- **RF-AUTH-003:** Cambio de contraseña
- **RF-AUTH-004:** Recuperación de perfil de usuario
- **RF-AUTH-005:** Registro de actividad (audit logs)

### *2.2.2 Gestión de Dispositivos IoT*

- **RF-DEV-001:** Registro de dispositivos ESP32
- **RF-DEV-002:** Asignación de dispositivos a empleados (relación 1:1)
- **RF-DEV-003:** Activación/desactivación de dispositivos
- **RF-DEV-004:** Monitoreo de estado de conexión
- **RF-DEV-005:** Visualización de última conexión

### *2.2.3 Captura y Almacenamiento de Datos de Sensores*

- **RF-SENS-001:** Recepción de datos vía MQTT cada 5 segundos
- **RF-SENS-002:** Almacenamiento de datos crudos (HR, SpO2, Acelerómetro)
- **RF-SENS-003:** Validación de formato y rangos de datos
- **RF-SENS-004:** Timestamp automático de cada lectura
- **RF-SENS-005:** Indexación para consultas rápidas

### *2.2.4 Procesamiento de Métricas*

- **RF-PROC-001:** Cálculo de métricas de ritmo cardíaco (avg, max, min, HRV)
- **RF-PROC-002:** Cálculo de métricas de oxigenación (avg, min, variance, desaturaciones)
- **RF-PROC-003:** Cálculo de métricas de movimiento (nivel de actividad, varianza, entropía)
- **RF-PROC-004:** Generación de features combinados (HR-Activity ratio, recovery time)
- **RF-PROC-005:** Ventanas de tiempo configurables (30s - 5min)

### *2.2.5 Machine Learning y Predicción de Fatiga*

- **RF-ML-001:** Predicción de índice de fatiga (0-100) mediante K-Means clustering
- **RF-ML-002:** Entrenamiento del modelo con datos históricos
- **RF-ML-003:** Actualización periódica del modelo
- **RF-ML-004:** Clasificación automática de niveles de fatiga
- **RF-ML-005:** Detección de patrones de fatiga

### *2.2.6 Sistema de Alertas*

- **RF-ALERT-001:** Generación automática de alertas según severidad (low, medium, high, critical)
- **RF-ALERT-002:** Tipos de alertas: fatigue\_detected, high\_heart\_rate, low\_oxygen, prolonged\_inactivity
- **RF-ALERT-003:** Notificaciones a supervisores
- **RF-ALERT-004:** Flujo de resolución de alertas
- **RF-ALERT-005:** Histórico de alertas
- **RF-ALERT-006:** Filtrado por estado (active, acknowledged, resolved)

### *2.2.7 Recomendaciones*

- **RF-REC-001:** Generación automática de recomendaciones
- **RF-REC-002:** Tipos: break, task\_redistribution, shift\_rotation
- **RF-REC-003:** Sistema de prioridades (1-5)
- **RF-REC-004:** Tracking de aplicación de recomendaciones
- **RF-REC-005:** Contexto detallado en formato JSON

### *2.2.8 Dashboards Interactivos*

**Dashboard de Administrador:** - **RF-DASH-ADM-001:** Estadísticas globales del sistema - **RF-DASH-ADM-002:** Gestión CRUD de supervisores - **RF-DASH-ADM-003:** Vista de todos los dispositivos activos - **RF-DASH-ADM-004:** Logs de actividad del sistema - **RF-DASH-ADM-005:** Métricas agregadas de todos los empleados

**Dashboard de Supervisor:** - **RF-DASH-SUP-001:** Métricas agregadas de empleados a cargo - **RF-DASH-SUP-002:** Gestión CRUD de empleados - **RF-DASH-SUP-003:** Gestión de dispositivos asignados - **RF-DASH-SUP-004:** Panel de alertas activas - **RF-DASH-SUP-005:** Sistema de recomendaciones - **RF-DASH-SUP-006:** Gráficas comparativas de fatiga - **RF-DASH-SUP-007:** Estadísticas de equipo

**Dashboard de Empleado:** - **RF-DASH-EMP-001:** Métricas personales en tiempo real - **RF-DASH-EMP-002:** Histórico de fatiga (gráficas) - **RF-DASH-EMP-003:** Mis alertas - **RF-DASH-EMP-004:** Estadísticas individuales - **RF-DASH-EMP-005:** Recomendaciones personales

### *2.2.9 Visualizaciones y Reportes*

- **RF-VIS-001:** Gráficas de series temporales (HR, SpO2, Fatiga)
- **RF-VIS-002:** Gráficas comparativas multi-empleado
- **RF-VIS-003:** Heatmaps de fatiga por hora/día
- **RF-VIS-004:** Distribución de alertas
- **RF-VIS-005:** Tendencias semanales/mensuales
- **RF-VIS-006:** Exportación de reportes en PDF
- **RF-VIS-007:** Exportación de datos en CSV/Excel

## *2.3 Características de los Usuarios*

El sistema tiene tres tipos de usuarios con diferentes niveles de acceso y responsabilidades:

### *2.3.1 Administrador*

**Perfil:** - Rol directivo o gerencial - Responsable de la supervisión general del sistema - Conocimientos técnicos básicos de sistemas web - No interactúa directamente con dispositivos

**Responsabilidades:** - Crear y gestionar cuentas de supervisores - Monitorear estadísticas globales del sistema - Revisar logs de actividad - Configurar parámetros del sistema - Acceso a reportes ejecutivos

**Nivel de acceso:** Total (excepto gestión directa de empleados)

**Frecuencia de uso:** Diaria a semanal

### *2.3.2 Supervisor*

**Perfil:** - Líder de equipo o jefe de área - Gestiona directamente a empleados - Usuario frecuente del sistema - Toma decisiones operativas basadas en datos

**Responsabilidades:** - Crear y gestionar cuentas de empleados bajo su supervisión - Asignar y gestionar dispositivos ESP32 - Monitorear alertas de fatiga de su equipo - Responder a alertas (acknowledge/resolve) - Revisar y aplicar recomendaciones - Analizar tendencias de su equipo - Generar reportes de equipo

**Nivel de acceso:** Medio (solo sus empleados y dispositivos)

**Frecuencia de uso:** Diaria

### *2.3.3 Empleado*

**Perfil:** - Usuario final del dispositivo wearable - Trabajador operativo - Mínima interacción con el sistema - Conocimientos técnicos básicos

**Responsabilidades:** - Portar el dispositivo ESP32 durante su turno - Revisar sus propias métricas y alertas - Ver recomendaciones personales - Consultar su histórico de fatiga

**Nivel de acceso:** Limitado (solo sus propios datos)

**Frecuencia de uso:** Ocasional (principalmente pasiva)

## *2.4 Restricciones Generales*

### *2.4.1 Restricciones Técnicas*

**Hardware:** - **REST-HARD-001:** Sistema diseñado para dispositivos ESP32 (no compatible con otros wearables) - **REST-HARD-002:** Relación 1:1 dispositivo-empleado (no se permiten múltiples dispositivos por empleado) - **REST-HARD-003:** Sensores requeridos: HR, SpO2, Acelerómetro 3-ejes

**Software:** - **REST-SOFT-001:** Backend requiere Python 3.11+ - **REST-SOFT-002:** Base de datos PostgreSQL 14+ obligatoria - **REST-SOFT-003:** Frontend

compatible con navegadores modernos (Chrome 90+, Firefox 88+, Edge 90+) -  
**REST-SOFT-004:** Node.js 18+ requerido para desarrollo frontend

**Protocolo:** - **REST-PROT-001:** Comunicación IoT exclusivamente vía MQTT -  
**REST-PROT-002:** API REST para comunicación frontend-backend - **REST-PROT-003:** Autenticación JWT obligatoria para todas las operaciones

#### *2.4.2 Restricciones de Despliegue*

- **REST-DEPLOY-001:** Sistema requiere broker MQTT (Mosquitto recomendado)
- **REST-DEPLOY-002:** Configuración mediante variables de entorno (.env)
- **REST-DEPLOY-003:** Soporte para Docker/Docker Compose

#### *2.4.3 Restricciones Regulatorias*

- **REST-REG-001:** No es un dispositivo médico certificado
- **REST-REG-002:** No reemplaza evaluaciones médicas profesionales
- **REST-REG-003:** Cumplimiento de GDPR/protección de datos personales (almacenamiento seguro)

#### *2.4.4 Restricciones de Seguridad*

- **REST-SEC-001:** Contraseñas hashadas con algoritmo seguro (PBKDF2)
- **REST-SEC-002:** Tokens JWT con expiración configurada
- **REST-SEC-003:** HTTPS obligatorio en producción
- **REST-SEC-004:** CORS configurado para dominios autorizados
- **REST-SEC-005:** Validación de entrada en todos los endpoints

### *2.5 Suposiciones y Dependencias*

#### *2.5.1 Suposiciones*

**SUPOS-001:** Se asume que los dispositivos ESP32 tienen acceso estable a WiFi

**SUPOS-002:** Se asume que los sensores están correctamente calibrados

**SUPOS-003:** Se asume que los empleados portarán el dispositivo durante su turno completo

**SUPOS-004:** Se asume que los supervisores revisarán alertas al menos cada 30

minutos

**SUPOS-005:** Se asume que el broker MQTT está disponible 24/7

**SUPOS-006:** Se asume conectividad a internet para acceso al sistema web

**SUPOS-007:** Se asume que los datos biométricos son representativos del estado de fatiga

### *2.5.2 Dependencias*

#### **Dependencias Externas:**

**DEP-EXT-001:** Broker MQTT (Mosquitto o compatible)

**DEP-EXT-002:** Servidor PostgreSQL

**DEP-EXT-003:** Conectividad de red WiFi para ESP32

**DEP-EXT-004:** Navegador web moderno para usuarios

#### **Dependencias de Software (Backend):**

Django==4.2.7

djangorestframework==3.14.0

djangorestframework-simplejwt==5.3.0

psycopg2-binary==2.9.9

paho-mqtt==1.6.1

python-decouple==3.8

django-cors-headers==4.3.0

drf-spectacular==0.26.5

scikit-learn==1.3.2

pandas==2.1.3

numpy==1.26.2

scipy==1.11.4

gunicorn==21.2.0

#### **Dependencias de Software (Frontend):**

```
{  
  "react": "^18.2.0",
```

```
"react-dom": "^18.2.0",  
"react-router-dom": "^6.20.0",  
"typescript": "^5.0.0",  
"axios": "^1.6.2",  
"chart.js": "^4.4.0",  
"react-chartjs-2": "^5.2.0",  
"daisyui": "^4.4.0",  
"tailwindcss": "^3.3.0"  
}
```

### **Dependencias de Servicios:**

**DEP-SERV-001:** Servidor web (Nginx recomendado para producción)

**DEP-SERV-002:** Servicio de tiempo sincronizado (NTP) para timestamps precisos

**DEP-SERV-003:** Sistema de logging centralizado (opcional pero recomendado)

## 3. REQUISITOS ESPECÍFICOS

### 3.1 Requisitos Funcionales Detallados

#### 3.1.1 Autenticación y Gestión de Usuarios

- El sistema debe permitir el inicio de sesión y cierre de sesión mediante JWT.
- El sistema debe permitir la gestión jerárquica de usuarios (Admin, Supervisor, Empleado).
- El sistema debe permitir el cambio de contraseña y recuperación de perfil.
- El sistema debe registrar todas las acciones importantes en un log de auditoría.

#### 3.1.2 Gestión de Dispositivos IoT

- El sistema debe permitir el registro y asignación de dispositivos ESP32 a empleados.
- El sistema debe permitir la activación/desactivación de dispositivos.



- El sistema debe mostrar el estado de conexión y la última conexión de cada dispositivo.

### 3.1.3 Captura y Almacenamiento de Datos de Sensores

- El sistema debe recibir datos de sensores (HR, SpO2, acelerómetro) vía MQTT cada 5 segundos.
- El sistema debe almacenar los datos crudos con timestamp y validación de formato.
- El sistema debe indexar los datos para consultas rápidas.

### 3.1.4 Procesamiento de Métricas

- El sistema debe calcular métricas de HR (promedio, máximo, mínimo, HRV).
- El sistema debe calcular métricas de SpO2 (promedio, mínimo, varianza, desaturaciones).
- El sistema debe calcular métricas de movimiento (nivel de actividad, varianza, entropía).
- El sistema debe generar features combinados y ventanas de tiempo configurables.

### 3.1.5 Machine Learning y Predicción de Fatiga

- El sistema debe predecir el índice de fatiga (0-100) usando K-Means clustering.
- El sistema debe entrenar y actualizar el modelo ML periódicamente.
- El sistema debe clasificar automáticamente los niveles de fatiga y detectar patrones.

### 3.1.6 Sistema de Alertas

- El sistema debe generar alertas automáticas según severidad y tipo.
- El sistema debe notificar a supervisores y permitir el flujo de resolución de alertas.
- El sistema debe mantener un histórico y permitir el filtrado de alertas.

### 3.1.7 Recomendaciones

- El sistema debe generar recomendaciones automáticas (break, redistribución de tareas, rotación de turnos).
- El sistema debe asignar prioridades y permitir el tracking de aplicación.

### 3.1.8 Dashboards y Visualizaciones

- El sistema debe mostrar dashboards personalizados por rol (Admin, Supervisor, Empleado).
- El sistema debe mostrar gráficas de series temporales, comparativas y heatmaps.
- El sistema debe permitir la exportación de reportes en PDF y datos en CSV/Excel.

## 3.2 Requisitos de Interfaz Externa

### 3.2.1 Interfaces de Usuario

- El sistema debe proveer una interfaz web responsiva y accesible.
- El sistema debe usar React + DaisyUI para una experiencia moderna y clara.
- El sistema debe mostrar notificaciones y alertas en tiempo real.
- El sistema debe permitir la navegación por dashboards, reportes y gestión de usuarios/dispositivos.

### 3.2.2 Interfaces de Hardware

- El sistema debe comunicarse con dispositivos ESP32 vía MQTT.
- El sistema debe recibir datos de sensores biométricos y movimiento.

### 3.2.3 Interfaces de Software

- El sistema debe exponer una API REST documentada con Swagger/OpenAPI.
- El sistema debe usar JWT para autenticación en todas las operaciones.
- El sistema debe permitir integración con sistemas de logging y monitoreo.

### 3.4 Restricciones de Diseño

- El sistema debe estar desarrollado usando Django REST Framework para el backend y React + TypeScript para el frontend.
- El backend debe seguir una arquitectura modular con apps separadas (users, devices, sensors, analytics, mqtt\_client).
- El frontend debe usar Context API para manejo de estado global y DaisyUI/TailwindCSS para la interfaz.
- La comunicación entre frontend y backend debe ser exclusivamente vía REST API con JWT.
- El sistema debe permitir configuración por variables de entorno (.env).
- El sistema debe ser compatible con PostgreSQL y Mosquitto MQTT.

### 3.5 Atributos de Calidad del Sistema

#### 3.5.1 Seguridad

- Todas las contraseñas deben almacenarse hasheadas.
- Todas las comunicaciones deben usar HTTPS en producción.
- El sistema debe validar y sanitizar todas las entradas de usuario.
- El sistema debe usar tokens JWT con expiración y refresh.
- El sistema debe registrar logs de actividad y accesos.

#### 3.5.2 Confiabilidad

- El sistema debe manejar errores de conexión y datos corruptos de sensores.
- El sistema debe tener tolerancia a fallos en el broker MQTT.
- El sistema debe permitir recuperación ante caídas del backend.

#### 3.5.3 Mantenibilidad

- El código debe estar documentado y seguir buenas prácticas de desarrollo.
- El sistema debe tener migraciones automáticas para la base de datos.
- El sistema debe permitir pruebas unitarias y de integración.

#### 3.5.4 Usabilidad

- La interfaz debe ser intuitiva y accesible para todos los roles.
- El sistema debe mostrar mensajes claros de error y confirmación.
- El sistema debe ser responsivo y funcionar en dispositivos móviles y escritorio.

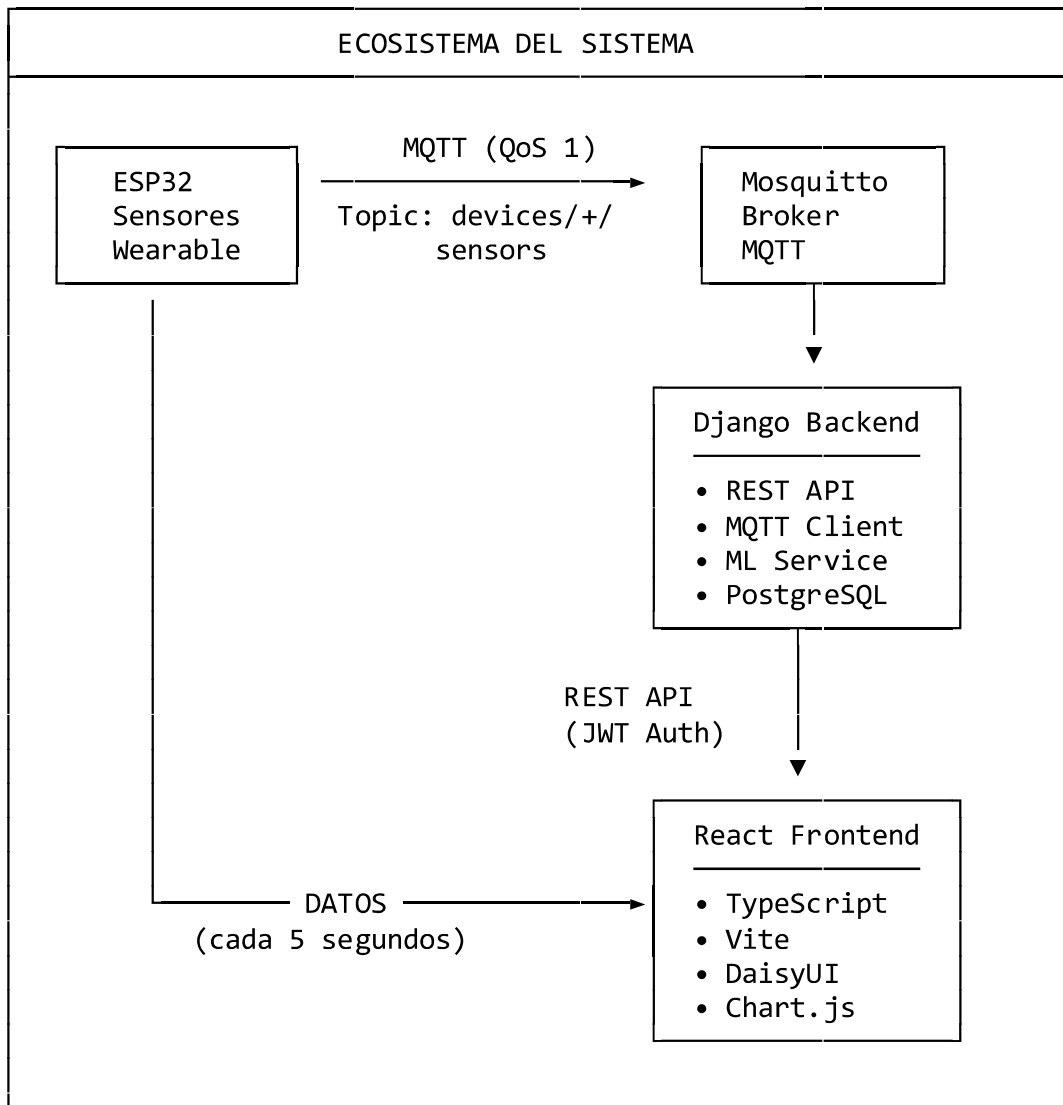
#### 3.5.5 Escalabilidad

- El sistema debe soportar la adición de más empleados y dispositivos sin pérdida de rendimiento.
- El sistema debe permitir la integración de nuevos módulos y servicios.

## 4. APÉNDICES

### 4.1 Diagrama de Arquitectura

#### Componentes del ecosistema:



### 4.2 Modelos de Base de Datos (Resumen)

- **CustomUser:** email, password, first\_name, last\_name, role, supervisor, admin, is\_active, timestamps
- **Device:** device\_identifier, employee, supervisor, is\_active, last\_connection, timestamps

- **SensorData:** device, timestamp, heart\_rate, spo2, accel\_x, accel\_y, accel\_z
- **ProcessedMetrics:** device, employee, window\_start, window\_end, hr\_avg, hr\_max, hr\_min, hrv\_rmssd, hrv\_sdn, hr\_trend, spo2\_avg, spo2\_min, spo2\_variance, desaturation\_count, activity\_level, etc.
- **FatigueAlert:** tipo, severidad, estado, timestamps, empleado, supervisor
- **RoutineRecommendation:** tipo, prioridad, contexto, timestamps, empleado, supervisor
- **ActivityLog:** usuario, acción, IP, user agent, detalles, timestamps

#### 4.3 Endpoints Principales de la API

- /api/auth/ - Autenticación y gestión de usuarios
- /api/devices/ - Gestión de dispositivos
- /api/sensor-data/ - Datos crudos de sensores
- /api/processed-metrics/ - Métricas procesadas
- /api/alerts/ - Alertas de fatiga
- /api/recommendations/ - Recomendaciones
- /api/dashboard/ - Dashboards por rol
- /api/visualizations/ - Gráficas y reportes
- /api/reports/ - Exportación de reportes

#### 4.4 Referencias Técnicas

- Documentación de Django REST Framework
- Documentación de React y DaisyUI
- Documentación de MQTT y Mosquitto
- Documentación de PostgreSQL
- Documentación de scikit-learn, pandas, numpy

## FIRMAS DE AUTORIZACIÓN DE PROYECTO V.1

Proyecto: Zero to Zero-Fatigue Zone

Por medio del presente documento, se hace constar que el equipo conformado por los estudiantes Bautista Ordoñez Brian Ángel, Orrantia González Germán Sisac, Islas Reyes Luis Ivan, García Valenzuela Ernesto, ha presentado su propuesta de proyecto para revisión. Los profesores abajo firmantes, tras haber revisado dicha propuesta, manifiestan su conformidad y autorizan su desarrollo conforme a los lineamientos académicos establecidos por la institución.



---

MLC. Resffa Elizabeth Fierro  
García



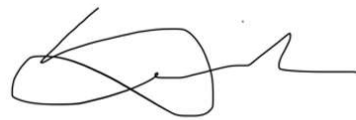
---

Dr. Ray Brunett Parra Galaviz



---

MC. Florencio Lopez Cruz



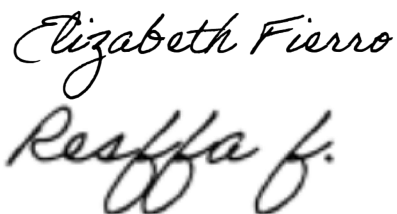
---

MTI. Daniel Torres Saldana

## FIRMAS DE AUTORIZACIÓN DE PROYECTO V.2

Proyecto: Zero to Zero-Fatigue Zone

Por medio del presente documento, se hace constar que el equipo conformado por los estudiantes Bautista Ordoñez Brian Ángel, Orrantia González Germán Sisac, Islas Reyes Luis Ivan, García Valenzuela Ernesto, ha presentado su propuesta de proyecto para revisión. Los profesores abajo firmantes, tras haber revisado dicha propuesta, manifiestan su conformidad y autorizan su desarrollo conforme a los lineamientos académicos establecidos por la institución.



---

MLC. Resffa Elizabeth Fierro  
García



---

Dr. Ray Brunett Parra Galaviz



---

MC. Florencio Lopez Cruz

FIRMAS DE INTEGRANTES



Se optó por implementar una maqueta con Internet de las Cosas (IoT) como alternativa al proyecto. Esta propuesta permite integrar comunicación en la plataforma web y demostrar aplicaciones prácticas de conectividad.



---

Brian Angel Bautista Ordoñez



---

German Sisac Orrantia Gonzalez



---

Luis Ivan Islas Reyes



---

Ernesto Garcia Valenzuela