



Universidad  
Continental

Taller de Proyectos 2

# **MODELO BASADO EN INTELIGENCIA ARTIFICIAL PARA DETECTAR SOMNOLENCIA EN CONDUCTORES**

Realizado por Tirza Buendia

# ÍNDICE

**01**

Introducción

**02**

Visión del proyecto

**03**

Especificación de Requisitos  
del Software (SRS)

**04**

Backlog del Producto

**05**

Sprint Backlog

# INTRODUCCIÓN


En el Perú, el transporte público es una de las principales formas de movilidad, y sus conductores suelen enfrentarse a **jornadas laborales largas, intermitentes y con poco descanso**. Esta realidad expone a muchos de ellos a altos niveles de fatiga y somnolencia, especialmente en el **transporte interprovincial**, donde los **trayectos** suelen ser **prolongados y durante la noche**.

**La somnolencia al volante es un problema frecuente y peligroso** en este sector, ya que **afecta** directamente la atención, los reflejos y la **capacidad de reacción del conductor**. Esta situación es preocupante, pues incrementa considerablemente el riesgo de accidentes de tránsito, algunos con consecuencias fatales.

Frente a ello, como medida de control, se plantea el **desarrollo de un modelo de monitoreo basado en inteligencia artificial**, capaz de detectar en tiempo real signos de fatiga mediante el análisis de gestos faciales. Esta herramienta busca alertar al conductor ante posibles estados de somnolencia, contribuyendo así a una **conducción más segura y a la reducción de accidentes viales**.



# VISIÓN DEL PROYECTO

- Objetivos
  - Público Objetivo
  - Funcionalidades Principales
  - Requisitos Técnicos
  - Riesgos y Limitaciones
  - Alcance
- 

# OBJETIVOS

Mitigar los riesgos asociados a la conducción bajo condiciones altas de somnolencia.





**01**

Identificar patrones faciales indicativos de somnolencia a partir de técnicas de visión por computadora.



**02**

Implementar un sistema de alerta basado en la detección de somnolencia para mejorar la seguridad vial.

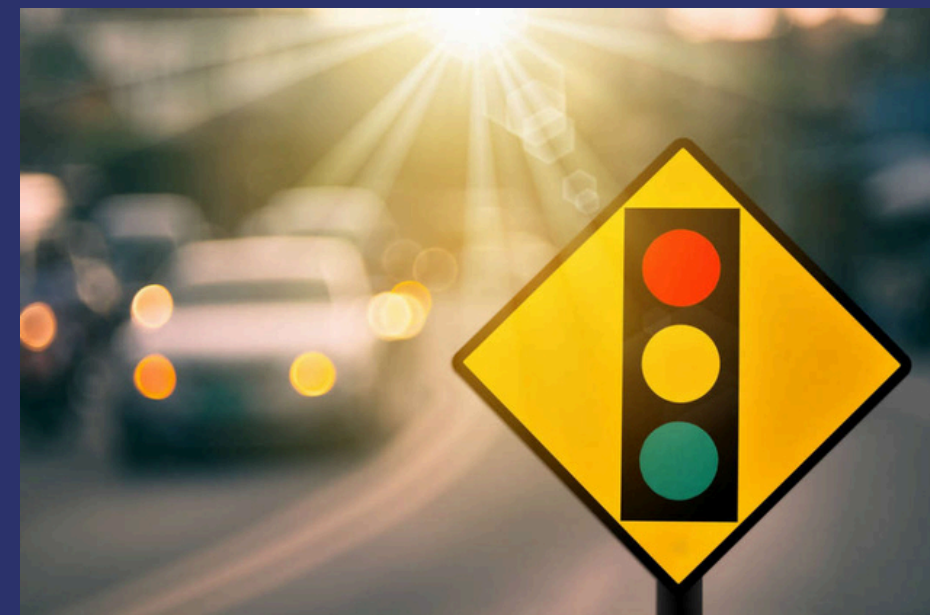
# PÚBLICO OBJETIVO

## Usuarios principales

- Conductores de transporte público.
- Conductores de transporte inter urbano.
- Conductores particulares.

## Beneficios esperados

Mayor seguridad vial



# FUNCIONALIDADES PRINCIPALES

## Esenciales

- Detección en tiempo real de gestos faciales indicativos de somnolencia.
- Generación de alertas y/o auditivas cuando se detecte somnolencia moderada o grave.
- Registro de eventos de somnolencia para análisis posterior.

## Futuras

- Predicción del estado del conductor basada en historial de somnolencia.
- Integración con hardware externo (cámaras infrarrojas de alta resolución).
- Integración con hardware como Raspberry Pi .



# REQUISITOS TÉCNICOS



- Python (para el desarrollo del modelo de IA y visión por computadora).
- OpenCV y TensorFlow/Keras (para el procesamiento de imágenes y entrenamiento del modelo).
- Windows 10 y versiones superiores.

# RIESGOS Y LIMITACIONES

- Posible resistencia por parte de los conductores a la adopción del sistema.
- El modelo no presenta respuestas precisas.
- Costos imprevistos (datasets de paga, necesidad de hardware más caro, etc.).

- Falta de acceso a tecnología avanzada para pruebas en entornos reales.
- Necesidad de mayor capacitación en desarrollo de modelos de IA.
- Dependencia de la calidad y cantidad de datos disponibles para el entrenamiento del modelo.



# ALCANCE DEL PROYECTO

- Herramientas de inteligencia artificial de aprendizaje

- Implementación de criterios de detección de somnolencia moderada.

- Generación de alertas de advertencia en tiempo real.

- Análisis de somnolencia en personas con trastornos del sueño.

- Uso de sensores fisiológicos como frecuencia cardíaca o monitoreo bioeléctrico.

- Implementación en hardware embebido como Raspberry Pi o dispositivos de a bordo en vehículos reales.



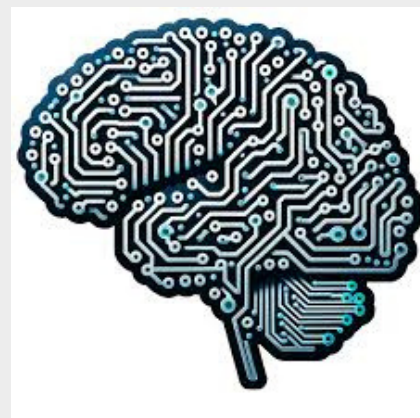
# ESPECIFICACIÓN DE REQUISITOS DEL SOFTWARE

- Definiciones, acrónimos, abreviaturas
- Descripción general del producto
- Requisitos Específicos

# DEFINICIONES

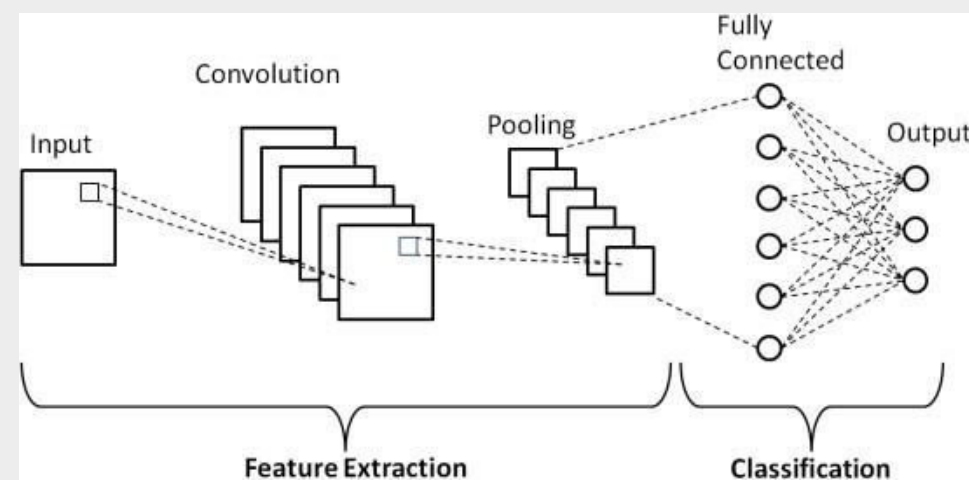
- **IA**

Inteligencia Artificial



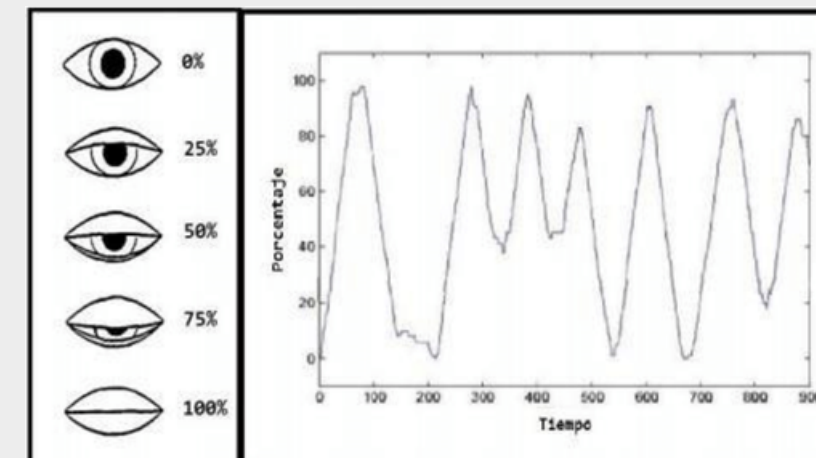
- **CNN**

Redes neuronales convolucionales (Convolutional Neural Network)



- **PERCLOS**

Porcentaje de cierre ocular (Percentage of Eye Closure)



- **OpenCV**

Biblioteca para procesamiento de imágenes



# DESCRIPCIÓN DEL PRODUCTO

- Frontend: interfaz en Python (Tkinter).
- Backend: Integración de IA (TensorFlow o PyTorch) para la detección de somnolencia.
- Hardware: Cámara para la captura de imágenes en tiempo real y GPU potente (procesamiento eficiente del modelo de IA).



## Restricciones

- Compatible con cámaras estándar.
- Requiere Python 3.9+ y sistemas operativos Windows/Linux.



## Suposiciones y dependencias

- El sistema operará en entornos con iluminación adecuada para el correcto funcionamiento de la cámara convencional.
- Los conductores no usarán accesorios que obstruyan el rostro (ej: máscaras, gafas oscuras).
- Dependencias a bibliotecas de IA para el procesamiento de imágenes y detección facial.
- Hardware con GPU para el entrenamiento del modelo.



# REQUISITOS

REQUERIMIENTOS FUNCIONALES	
RF1	El sistema detectará rostros en tiempo real con una cámara.
RF2	Calculará el porcentaje de cierre ocular (PERCLOS) y la frecuencia de bostezos, al igual que el ángulo de inclinación de la cabeza.
RF3	Emitirá una alerta sonora y/o visual si: PERCLOS >20% o Frecuencia de cabeceo >3 veces/minuto.
RF4	Guardará registros de eventos (detectados como somnolencia) en un archivo CSV para análisis posterior.

REQUERIMIENTOS NO FUNCIONALES	
RNF1	Latencia máxima de 1 segundo en la detección en tiempo real.
RNF2	Precisión mínima del 85% en condiciones de iluminación óptima.
RNF3	Compatibilidad con cámaras estándar de 720p o superior

# REQUISITOS

## REQUISITOS DE UI

La interfaz debe ser intuitiva y minimalista.

Mostrará video en vivo de la cámara.

Métricas en tiempo real (PERCLOS, bostezos, cabeceos).

Alertas visuales (cambio de color en la pantalla) y sonoras (tono continuo hasta respuesta del conductor).

## REQUISITOS DE HW Y SW

Cámara web convencional (resolución mínima: 720p).

GPU para entrenamiento y inferencia

Python 3.9+ como lenguaje principal.

TensorFlow o PyTorch para el modelo de IA.

OpenCV para procesamiento de imágenes





# BACKLOG DEL PRODUCTO

- Épicas - Historias de usuario

# ÉPICAS

## Detección Facial en tiempo Real

- **HU 1.1:** Como usuario, quiero que el sistema detecte mi rostro en tiempo real, para realizar el monitoreo de somnolencia.
- **HU 1.2:** Como usuario, quiero que el sistema mida con precisión mi porcentaje de cierre ocular (PERCLOS), para detectar fatiga visual.
- **HU 1.3:** Como usuario, quiero que el sistema identifique mis bostezos, para evaluar mi nivel de somnolencia.
- **HU 1.4:** Como usuario, quiero que el sistema detecte si mi cabeza se inclina por somnolencia, para alertarme.

# ÉPICAS

## Sistema de Alertas

- **HU 2.1:** Como usuario, quiero recibir alertas claras cuando el sistema detecte signos de somnolencia, para tomar medidas preventivas.

## Registro de Eventos

- **HU 3.1:** Como usuario, quiero un archivo CSV con eventos de somnolencia, para analizar patrones.

Prioridad	Historia de Usuario	Estado	Estimación
Alta	Detectar rostros con cámara	Pendiente	8
Alta	Monitoreo de Cierre Ocular	Pendiente	5
Media	Detección de Bostezos	Pendiente	5
Media	Medición de Inclinação de Cabeza	Pendiente	4
Alta	Alertas Temprana de somnolencia	Pendiente	4
Baja	Exportar Datos a CSV	Pendiente	3



# SPRINT BACKLOG

Sprints - Tareas - Esfuerzo estimado



• **SPRINT 1**

HU 1.1.: Detectar rostros con cámara

HU 1.2.: Monitoreo de Cierre Ocular

ID	Tarea	Responsable	Estimación (Horas)	Estado
1.1.1	Configurar cámara y flujo de video en tiempo real	Backend Dev	18	Pendiente
1.1.2	Implementar detección facial con OpenCV/MediaPipe	Computer Vision Engineer	24	Pendiente
1.1.3	Optimizar latencia (<1 segundo por frame)	Backend Dev	18	Pendiente
1.1.4	Pruebas con diferentes ángulos y condiciones de luz	QA Engineer	10	Pendiente
1.1.5	Documentar configuración y requisitos de hw	Technical Writer	6	Pendiente
1.2.1	Implementar cálculo de PERCLOS con landmarks oculares	Computer Vision Engineer	22	Pendiente
1.2.2	Filtrar parpadeos rápidos (<0.5 segundos)	Data Scientist	14	Pendiente

**Estimación Total de Esfuerzos**

Función	Responsable	Horas Totales
Computer Vision Development	Computer Vision Engineer	46
ML/Data Science	Data Scientist/ML Engineer	14
Backend Development	Backend Developer	36
Frontend Development	Frontend Developer	0
UX/UI Design	UX/UI Designer	0
Quality Assurance (QA)	QA Engineer	10
Technical Writing	Technical Writer	6
Total General		112

• **SPRINT 2**

HU 1.2.: Monitoreo de Cierre Ocular

HU 1.3.: Detección de Bostezos

HU 1.4.: Medición de Inclinación de Cabeza

ID	Tarea	Responsable	Estimación (Horas)	Estado
1.2.3	Calibrar sensibilidad para usuarios con gafas/lentes	QA Engineer	12	Pendiente
1.2.4	Documentar algoritmo y métricas de validación	Technical Writer	6	Pendiente
1.3.1	Implementar Mouth Aspect Ratio (MAR) para bostezos	Computer Vision Engineer	18	Pendiente
1.3.2	Entrenar modelo con dataset de bostezos reales	Data Scientist	22	Pendiente
1.3.3	Filtrar falsos positivos (habla, risa)	Machine Learning Engineer	18	Pendiente
1.3.4	Pruebas de precisión	QA Engineer	12	Pendiente
1.4.1	Implementar detección de ángulo de cabeza (pitch/yaw)	Computer Vision Engineer	21	Pendiente

**Estimación Total de Esfuerzos**

Función	Responsable	Horas Totales
Computer Vision Development	Computer Vision Engineer	39
ML/Data Science	Data Scientist/ML Engineer	40
Backend Development	Backend Developer	0
Frontend Development	Frontend Developer	0
UX/UI Design	UX/UI Designer	0
Quality Assurance (QA)	QA Engineer	24
Technical Writing	Technical Writer	6
Total General		109

• **SPRINT 3**

- HU 1.4.: Medición de Inclinación de Cabeza
- HU 2.1.: Alerta Temprana de Somnolencia
- HU 3.1.: Exportar datos a CVS

ID	Tarea	Responsable	Estimación (Horas)	Estado
1.4.2	Definir umbrales para cabeceos por fatiga (>15°)	Data Scientist	14	Pendiente
1.4.3	Filtrar movimientos voluntarios (ej: mirar espejos)	Machine Learning Engineer	14	Pendiente
1.4.4	Pruebas de función	QA Engineer	12	Pendiente
2.1.1	Diseñar interfaz de alerta visual	UX/UI Designer	14	Pendiente
2.1.2	Implementar alarma sonora (85 dB, no intrusiva)	Backend Dev	10	Pendiente
2.1.3	Configurar persistencia de alerta hasta confirmación	Frontend Dev	16	Pendiente
2.1.4	Pruebas de usabilidad	QA Engineer	18	Pendiente
3.1.1	Definir estructura del CSV (timestamp, PERCLOS, etc.)	Data Scientist	12	Pendiente

**Estimación Total de Esfuerzos**

Función	Responsable	Horas Totales
Computer Vision Development	Computer Vision Engineer	0
ML/Data Science	Data Scientist/ML Engineer	40
Backend Development	Backend Developer	10
Frontend Development	Frontend Developer	16
UX/UI Design	UX/UI Designer	14
Quality Assurance (QA)	QA Engineer	30
Technical Writing	Technical Writer	0
Total General		110

• **SPRINT 4**

HU 3.1.: Exportar datos a CVS

ID	Tarea	Responsable	Estimación (Horas)	Estado
3.1.2	Implementar generación automática de CSV	Backend Dev	22	Pendiente
3.1.3	Validar compatibilidad con Excel/Pandas	QA Engineer	10	Pendiente
3.1.4	Documentar formato y ejemplos de CSV	Technical Writer	8	Pendiente

**Estimación Total de Esfuerzos**

Función	Responsable	Horas Totales
Computer Vision Development	Computer Vision Engineer	0
ML/Data Science	Data Scientist/ML Engineer	0
Backend Development	Backend Developer	22
Frontend Development	Frontend Developer	0
UX/UI Design	UX/UI Designer	0
Quality Assurance (QA)	QA Engineer	10
Technical Writing	Technical Writer	8
Total General		40

**TOTAL: 371 hrs (2 meses y medio) aprox.**





# MUCHAS GRACIAS

[www.unsitiogenial.es](http://www.unsitiogenial.es)

