Facultad Regional Rosario
Universidad Tecnológica Nacional

Trabajo práctico integrador:

Grupo 14

SignLearn: Juego Interactivo para Aprender Lengua de Señas

Alumnos:

Nombre y Apellido	Legajo
Pecoraro, Lucio	50239
Cordoba, Lucas	45091
Genoud, Franco	48992
Battistoni, Maria Paz	49641

Fecha de presentación: 11-11-2025

1. Narrativa del Proyecto

El aprendizaje de la Lengua de Señas Argentina (LSA) representa un desafío significativo, especialmente para los niños. Las herramientas de aprendizaje tradicionales, como libros o videos, carecen de la interactividad y la retroalimentación inmediata que son cruciales para dominar los gestos físicos. Esta falta de práctica dinámica puede llevar a la frustración y a una curva de aprendizaje lenta.

Sign Learn nace como una solución a este problema, proponiendo un puente entre la tecnología de visión por computadora y la educación inclusiva. El proyecto busca transformar el aprendizaje de la LSA en una experiencia lúdica y atractiva, eliminando las barreras de la práctica estática.

2. Abstract

SignLearn es un juego educativo interactivo programado en Python, diseñado para enseñar a niños el abecedario de la lengua de señas. El sistema utiliza la cámara web del usuario para capturar los gestos de la mano en tiempo real. Mediante el uso de librerías como OpenCV para el procesamiento de video y MediaPipe para el reconocimiento de landmarks de la mano, el juego analiza la seña realizada por el usuario y la compara con la seña objetivo.

El proyecto implementa una arquitectura de 3 capas (Presentación, Negocio y Datos) y sigue un modelo de dominio centrado en el Usuario y sus Sesiones de Juego. El sistema incorpora elementos de gamificación, como puntuaciones, combos y límites de tiempo, para motivar al jugador. La lógica de negocio valida la precisión de las señas y gestiona la progresión del juego, mientras que la capa de datos (diseñada para SQLite con SQLAlchemy) permite el almacenamiento del progreso y el desempeño histórico.

3. Requerimientos Funcionales (Casos de Uso)

CU01: Registro de Usuario

Permite que un nuevo jugador se registre en el sistema ingresando su nombre, edad, usuario y contraseña.

CU02: Inicio de Sesión

Permite que el usuario autenticado acceda al juego y a su historial de progreso.

CU03: Iniciar Juego / Sesión

El usuario inicia una nueva sesión de práctica. Se muestran las señas y se activa la cámara para el reconocimiento.

CU04: Detección y Evaluación de Señales

El sistema utiliza OpenCV y el modelo de IA para comparar la seña del usuario con la seña objetivo, calculando precisión y tiempo.

CU05: Cálculo de Puntuación y Nivel

El sistema calcula los puntos obtenidos según precisión y tiempo, y determina si el jugador avanza de nivel.

CU06: Almacenamiento de Resultados

Al finalizar la sesión, se guarda el puntaje, precisión promedio, nivel y fecha en la base de datos.

CU07: Visualización de Progreso

El usuario puede consultar un resumen de sus sesiones pasadas con estadísticas y evaluación de desempeño.

4. Requerimientos No Funcionales

Rendimiento: El sistema debe procesar el video y realizar el reconocimiento de señas en tiempo real (mínimo 20 FPS) para no afectar la jugabilidad.

Usabilidad: La interfaz debe ser simple, intuitiva y visualmente atractiva para niños, con retroalimentación clara e inmediata (visual y auditiva).

Seguridad: (Ref: RN05) Las contraseñas de los usuarios deben almacenarse cifradas (hash) en la base de datos.

Portabilidad: El juego debe poder ejecutarse en diferentes sistemas operativos (Windows, macOS, Linux) que soporten Python y las librerías requeridas.

Precisión: El modelo de reconocimiento debe tener una tasa de acierto superior al 80% (Ref: RN01) para ser considerado válido.

5. Stack Tecnológico

- ☐ Frontend / UI
- ☐ OpenCV (cv2)
- ☐ Interfaz gráfica de escritorio basada en la captura de video en tiempo real.
- ☐ Visión por Computadora
- ☐ OpenCV
- ☐ Captura y procesamiento de imágenes en tiempo real.
- ☐ IA / Reconocimiento de Señales
- ☐ MediaPipe (Hands)
- ☐ Modelo que detecta landmarks de la mano y permite la clasificación de gestos.
- ☐ Backend / Lógica de Negocio
- ☐ Python 3.13
- ☐ Contiene toda la lógica del juego (puntuación, combos, tiempo) y la clasificación geométrica de señas.
- ☐ Base de Datos
- ☐ SQLite + SQLAlchemy

- ☐ Almacenamiento de usuarios y sesiones.
- ☐ Testing
- ☐ Pytest
- ☐ Pruebas unitarias de reglas de negocio y detección.

6. Reglas de Negocio

RN01 – Validación de señas

El sistema solo considera una seña “correcta” si la precisión de la detección (según el modelo IA/lógica geométrica) supera el umbral del 80%.

Implementación (Parcial): La lógica en tpi_letras.py (ej. TH_CLOSE, angle(), is_extended_y) define geométricamente la validez de una seña.

RN02 – Asignación de puntos

El puntaje de cada seña depende de la precisión y el tiempo (bonificación por velocidad).

Implementación: tpi_juego.py implementa speed_bonus(elapsed_letter) y el cálculo de score_gain basado en base, bonus y combo.

RN03 – Progresión de niveles

Para avanzar de nivel, el usuario debe completar al menos 10 señas con una precisión promedio mayor o igual al 85%.

Implementación (Parcial): El juego actual avanza por palabras (target_word) pero la lógica de "niveles" (dificultad) no está explícita.

RN04 – Retroalimentación

Al finalizar cada sesión, el sistema genera un reporte visual.

Implementación: tpi_juego.py implementa draw_end_menu(img, score) que muestra el puntaje final.

RN05 – Registro y seguridad

Todas las contraseñas deben almacenarse en formato cifrado.

Implementación: (Pendiente, asociado a la Capa de Datos).

7. Modelo de Dominio

Entidades Principales:

Usuario:

- Representa al jugador (niño o usuario registrado).
- Guarda información básica: nombre, edad, usuario, contraseña (encriptada).
- Se relaciona con las partidas o sesiones de juego que ha realizado.

SesionJuego

- Registra cada sesión o ronda de práctica del usuario.
- Contiene información sobre:
- Fecha y hora de la sesión.
- Nivel o dificultad.
- Tiempo promedio por seña.
- Precisión promedio.
- Puntaje total obtenido.
- Resultado (por ejemplo, “superado” o “a mejorar”).
- Cada sesión pertenece a un único usuario.

DesempeñoSeña

- Representa el análisis de una seña específica durante una sesión.
- Guarda la seña mostrada, la precisión obtenida, el tiempo de reacción y si fue reconocida correctamente.
- Está vinculada a una sesión de juego.

Diagrama de Clases (Modelo de Dominio)

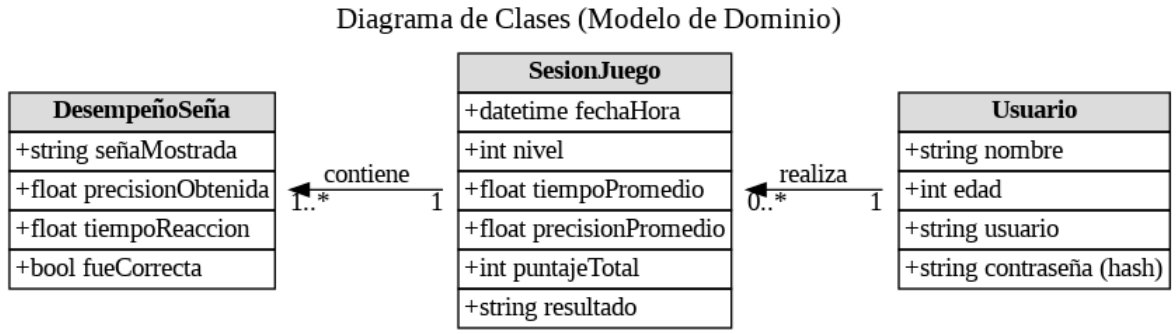
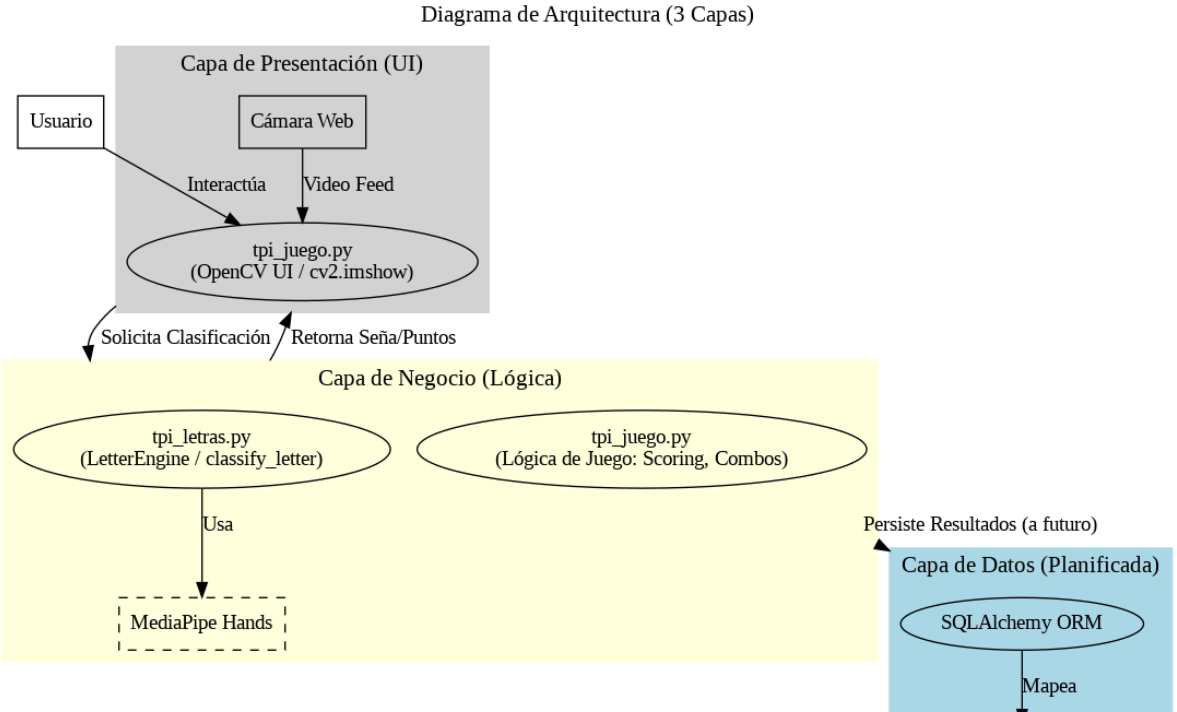


Diagrama de Arquitectura



10. Bibliografía y Documentación de Librerías

OpenCV: Open Source Computer Vision Library. Utilizada para la captura de video, procesamiento de imágenes y renderizado de la interfaz de usuario.

Documentación: https://docs.opencv.org/

MediaPipe: Framework de Google para la construcción de pipelines de percepción multimodal. Se utiliza específicamente la solución "Hands" para la detección de landmarks de manos.

Documentación: https://developers.google.com/mediapipe/solutions/vision/hand_landmarker

Python: Python es potente y rápido; se integra bien con otros lenguajes; funciona en cualquier lugar; es amigable y fácil de aprender; es de código abierto.

Su mayor fortaleza es El Índice de Paquetes de Python (PyPI - https://pypi.org/) ya que alberga miles de módulos de terceros para Python. Tanto la biblioteca estándar de Python como los módulos aportados por la comunidad ofrecen un sinfín de posibilidades.

Documentación: https://docs.python.org/3/

SQLite: SQLite es una biblioteca integrada que implementa un motor de base de datos SQL transaccional, autónomo, sin servidor y sin configuración. El código de SQLite es de dominio público, y por lo tanto, su uso es gratuito.

SQLite es un motor de base de datos SQL integrado. A diferencia de la mayoría de las bases de datos SQL, SQLite no requiere un proceso de servidor independiente. SQLite lee y escribe directamente en archivos de disco comunes. Una base de datos SQL completa, con múltiples tablas, índices, disparadores y vistas, se encuentra en un único archivo de disco.

El módulo sqlite3 de Python proporciona una interfaz estándar para interactuar con bases de datos SQLite, además se incluye en la biblioteca estándar de Python a partir de la versión 2.5, por lo que no es necesaria ninguna instalación adicional.

Documentación: https://sqlite.org/docs.html - https://docs.python.org/es/3/library/sqlite3.html

11. Link al Código Fuente

https://github.com/lcordoba844/frro-python-2025-14/tree/master/TPI_nuestro