



Aprendiendo Programación



Implementación de tres juegos utilizando **Streamlit** para la interfaz de usuario y **Python puro** para la lógica del backend. Además, se integran bases de datos **PostgreSQL** y **MongoDB** para el almacenamiento de datos y culminando con un despliegue real en la nube.

Este taller tiene un enfoque escalonado y práctico, permitiendo cubrir desde conceptos básicos hasta temas más avanzados de manera una manera integrada.

Objetivos de Aprendizaje

- Comprender los principios básicos de programación en Python.
- Aplicar conocimientos en bases de datos relacionales (PostgreSQL) y NoSQL (MongoDB).
- Desarrollar aplicaciones interactivas con Streamlit.
- Integrar lógica de backend con una interfaz de usuario funcional.
- Implementar y desplegar aplicaciones en la nube con herramientas gratuitas.

Beneficios para el Estudiante

Al completar este taller, el estudiante será capaz de:

- Desarrollar juegos interactivos usando Python.
- Conectar aplicaciones con bases de datos PostgreSQL y MongoDB.
- Crear y desplegar aplicaciones en la nube sin costo.
- Mejorar habilidades en programación orientada a la resolución de problemas.

Requisitos Previos

Para aprovechar mejor el taller, se recomienda:

- Conocimientos básicos de programación en cualquier lenguaje.
- Familiaridad con lógica de programación y estructuras de control.
- Conocimientos básicos de bases de datos (opcional pero recomendado).



🔧 Requisitos Técnicos

- PC con sistema operativo Windows, macOS o Linux.
- Conexión a Internet.
- Editor de código recomendado: VS Code o PyCharm.
- Python 3.12 ó superior ó Anaconda Navigator con Python 3.12 superior.
- Cuenta en GitHub para despliegue en Streamlit Community Cloud.
- Cuenta en MongoDB Atlas y Render (para bases de datos).

🔧 Herramientas Necesarias e instalaciones Requeridas

1. **Python** y opcional para manejo de entornos Python usar **Anaconda Navigator**
 - [Descargar Python 3.12 ó superior](#)
 - [Descargar Anaconda Navigator con Python 3.12 ó superior](#)
2. **Pip (Gestor de paquetes de Python)** (viene con Python ó Anaconda Navigator)
3. **Editor de Código (VS Code ó PyCharm)**
 - [Descargar VS Code Última Version](#)
 - [Descargar PyCharm Community Edition Última Version](#)
4. **PostgreSQL** (para almacenamiento estructurado)
 - [Descargar PostgreSQL Version 17](#)
5. **MongoDB y MongoDB Atlas** (para almacenamiento NoSQL)
 - [Descargar MongoDB Community Server Version 8.0.6 \(current\)](#)
 - [Crear cuenta en MongoDB Atlas](#)
6. **Render (para desplegar PostgreSQL)**
 - [Crear cuenta en Render](#)
7. **Streamlit** (para la interfaz) y **Cuenta en Streamlit Community Cloud**
 - Se brindará el comando correspondiente para instalar Streamlit.
 - [Crear cuenta en Streamlit](#)

8. Git y Cuenta en GitHub

- [Descargar Git Version 2.49.0](#)
- [Crear cuenta en GitHub](#)

9. DBeaver (herramienta para gestión de bases de datos)

- [Descargar DBeaver Community 25.0.1](#)

Tecnologías utilizadas:

- **Backend:** Python  puro sin frameworks, utilizando módulos de la biblioteca estándar para manejar la lógica y las peticiones.



- **Frontend:** Streamlit  , una biblioteca ligera para la creación de aplicaciones web interactivas en Python.



• Bases de datos:

- **PostgreSQL:** usando bibliotecas como [psycopg2](#), para almacenar datos estructurados.



PostgreSQL

- **MongoDB:** Para el almacenamiento de datos en formato NoSQL, pymongo almacena datos semiestructurados.



MongoDB®

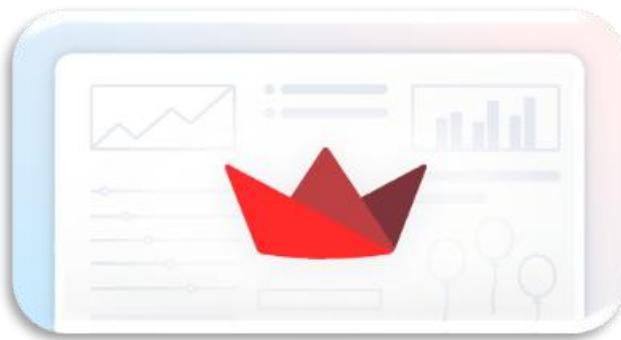
- **Despliegue en la nube (gratuito):**

- **Backend:** Se desplegará en un servicio gratuito Replit, donde se pueden correr scripts en Python sin frameworks.



- **Frontend:**

- a) **Streamlit Community Cloud**, para alojar y desplegar la aplicación de forma gratuita, directamente desde un repositorio de GitHub.



- **Bases de datos:**

- a) **MongoDB Atlas**, ofrece un tier gratuito para MongoDB suficiente para el taller.



- b) **Postgres**, la base de datos se desplegará en **Render**.





Nivel Básico

"Adivina el Número": Se introduce a los estudiantes a la lógica básica de programación y a la conexión con una base de datos relacional (**Postgres**) mediante **Python** puro. La interfaz en **Streamlit** les permite ver de forma interactiva cómo se gestionan las peticiones y se muestran respuestas dinámicas.

Objetivo del Juego: El usuario debe adivinar un número aleatorio generado por el backend.

Base de Datos: PostgreSQL

Backend: Python Puro

Frontend: Streamlit

Juego: Adivina el Número

- El usuario intenta adivinar un número aleatorio entre 1 y 100.
- Se proporciona retroalimentación en cada intento (mayor/menor).
- Se registra la cantidad de intentos hasta acertar.
- Se registra la puntuación y se guarda en Postgres usando psycopg2.

Aprendizaje

- Conexión entre la UI de Streamlit y funciones backend en Python.
- Operaciones CRUD básicas en una base de datos relacional.

Nivel Medio

"Ahorcado": Se profundiza en la gestión de estados dinámicos y el uso de bases de datos NoSQL (MongoDB) para almacenar el progreso del juego. Con Streamlit se crea una experiencia interactiva donde se visualiza el progreso del usuario en tiempo real, fortaleciendo la comunicación entre la lógica del juego y la interfaz.



Objetivo del Juego: El usuario debe adivinar una palabra oculta, letra por letra visualizando el progreso del juego.

Base de Datos: MongoDB

Backend: Python Puro

Frontend: Streamlit

Juego: Ahorcado

- Se selecciona una palabra secreta y el usuario intenta adivinarla letra por letra.
- Se muestra el estado actual de la palabra con los espacios ocultos.
- Se indican las respuestas correctas e incorrectas.

Aprendizaje

- Manejo de datos dinámicos en una base de datos NoSQL.
- Integración entre la lógica de juego y la visualización interactiva.

Nivel Avanzado Simplificado

"Trivia Express": Se integra la lógica de selección y validación de preguntas de opción múltiple, combinando el uso de Postgres (para datos estructurados como perfiles y puntuaciones) y MongoDB (para logs y detalles de cada partida). Además, se enfatiza el despliegue en la nube con herramientas gratuitas, mostrando cómo llevar un proyecto desde el desarrollo hasta la producción en un tiempo limitado (aproximadamente 2 horas).

Objetivo del Juego: El usuario debe adivinar una palabra oculta, ingresando letras y viendo el progreso del juego.

Base de Datos: PostgreSQL y MongoDB

Backend: Python Puro

Frontend: Streamlit



Juego: Trivia de Programación

- Se presentan preguntas de opción múltiple sobre temas de programación.
- El usuario selecciona una respuesta y recibe retroalimentación sobre su elección.

Aprendizaje

- Diseño de una aplicación interactiva y la integración de dos tipos de bases de datos.
- Desarrollo rápido y deployment en la nube, con énfasis en la comunicación entre frontend y backend en Python.

