

**UNIVERSIDAD INTERNACIONAL DEL ECUADOR**

**CARRERA DE INGENIERÍA EN SOFTWARE**

**Materia:**

Lógica de Programación

**Proyecto integrador:**

El impacto de las nuevas tecnologías en la sociedad: visualización del futuro

**AUTORA:**

Dayanna Margarita Castro Jaime

**ECUADOR**

**AÑO 2025**

**Introducción**

El proyecto “Impacto Tecnológico” tiene como objetivo el analizar y poner en práctica los distintos conocimientos adquiridos en la materia de Lógica de Programación, por medio de la organización en diferentes etapas. La idea es aplicar lo aprendido mediante la elaboración de diagramas, el desarrollo de un programa sencillo y la gestión de un repositorio en GitHub que muestre todo el proceso de manera ordenada.

Para lograrlo, se ha diseñado una estructura que incluye un cronograma de actividades semanales, la creación de diagramas que representen la lógica del sistema, la codificación del programa y la documentación de los resultados junto con una reflexión crítica del proceso, fortaleciendo así las habilidades tanto tecnológicas como de organización e investigación, fomentando la planificación el manejo de herramientas digitales y la capacidad de trabajar de manera autónoma con los recursos disponibles.

En lo personal, el realizar este proyecto me permitió organizar mejor mis ideas y además darme cuenta de que cada paso, por pequeño que parezca, suma en el resultado final. En definitiva, este proyecto representa una oportunidad para unir teoría y práctica, fortaleciendo competencias que serán útiles tanto en el ámbito académico como en el profesional, además de dejarme una experiencia valiosa de aprendizaje que va más allá de la materia misma.

**Cronograma**

El siguiente cronograma tiene como objetivo mostrar las actividades planeadas para el desarrollo del proyecto integrador "El impacto de las nuevas tecnologías en la sociedad: visualización del futuro".

En cada semana se detallan las tareas a realizar y las entregas correspondientes, organizando el trabajo de manera eficiente y asegurando la integración de todas las unidades del curso.

|  |  |  |  |
| --- | --- | --- | --- |
| Semana | Unidad / Temas | Actividad | Entrega |
| 1 | Unidad 1 (Tema 1 y 2) | Investigación de tipos de diagramas y selección del software | Documento de análisis o nota de investigación (DOCS/investigacion\_unidad1.docx) |
| 2 | Unidad 1 (Tema 1 y 2) | Diseño de funcionalidades y diagramas de casos de uso | Diagramas de casos de uso (DIAGRAMS/casos\_de\_uso.png) |
| 3 | Unidad 2 (Tema 3 y 4) | Configuración de GitHub y preparación del entorno | Captura del repo y configuración (DOCS/config\_github.docx) |
| 4 | Unidad 2 (Tema 3 y 4) | Avance inicial de codificación y diagramas de flujo | Diagrama de flujo (DIAGRAMS/flujo.png) |
| 5 | Unidad 3 (Tema 5 y 6) | Implementación de funcionalidades y estructuras lógicas | Código fuente parcial (SRC/main.py) |
| 6 | Unidad 3 (Tema 5 y 6) | Comentarios, pruebas de funciones y ajustes | Código comentado y probado (SRC/main.py) |
| 7 | Unidad 4 (Tema 7 y 8) | Integración de todas las unidades, revisión y documentación | Documentación final (DOCS/documentacion\_final.docx) |
| 8 | Unidad 4 (Tema 7 y 8) | Preparación de la presentación final y entrega | Presentación y video demo (MEDIA/presentacion.pdf, MEDIA/demo.mp4) |

Este cronograma nos sirve para tener todo bien organizado y saber qué toca hacer cada semana y qué tenemos que entregar. Al dividir las tareas por unidad y tema, es más fácil avanzar paso a paso sin perdernos, y así al final podemos juntar todo el proyecto completo y cumplir con lo que nos propusimos desde el inicio.

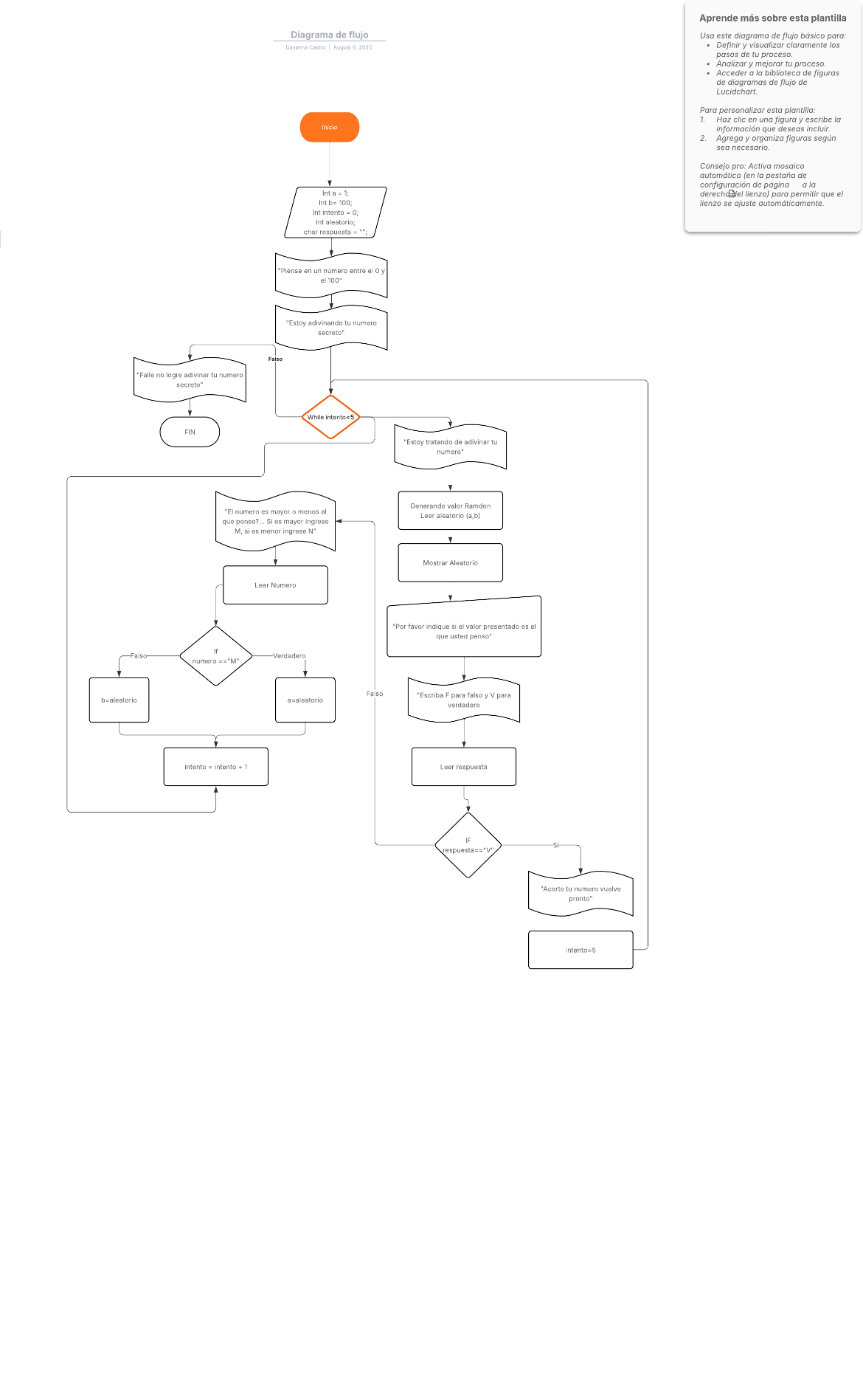
**Diagramas**

1. **Diagrama de Flujo**

Los diagramas de flujo fueron herramientas clave para visualizar la lógica y el flujo de datos del sistema, permitiendo planificar las funcionalidades de forma clara y ordenada. Como menciona (Pressman, 2014) el diseño visual mediante diagramas ayuda a estructurar el software antes de la codificación.

Este diagrama describe la secuencia lógica de nuestro programa Adivinador de Números. Se inicia con la definición de variables principales como los límites del rango, los intentos y la respuesta del usuario. Luego, el sistema genera un número aleatorio y, mediante decisiones, se compara la respuesta del jugador: si el número es mayor, menor o el correcto.

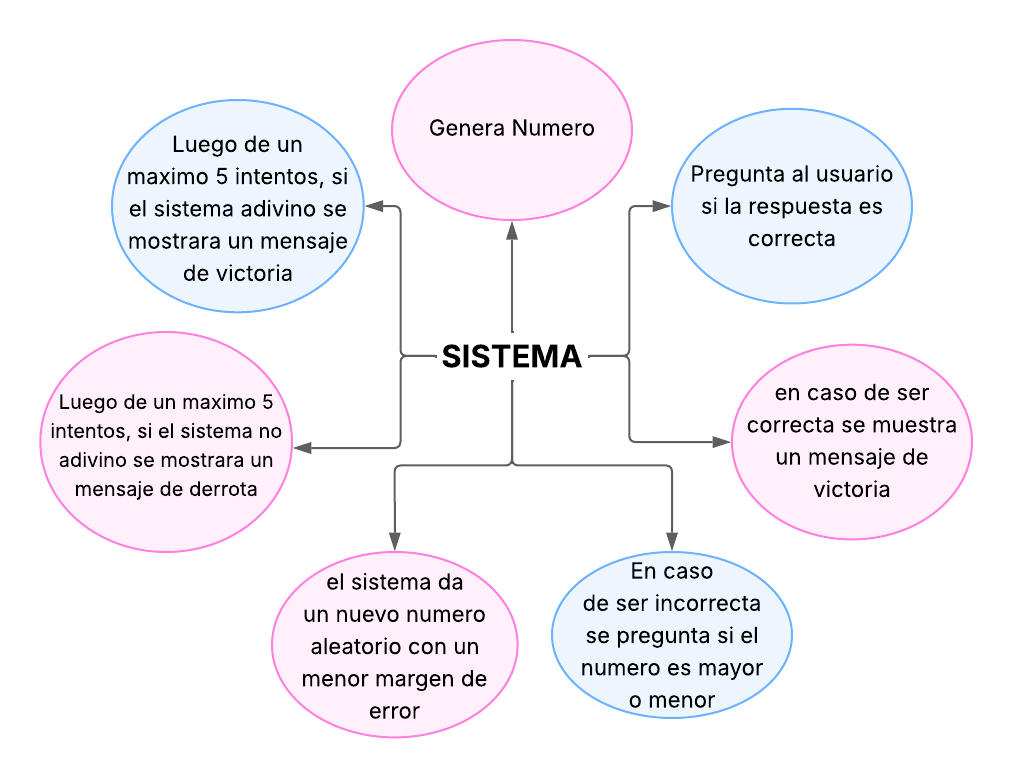
De esta manera, el diagrama permite visualizar de forma clara cómo avanza el proceso desde el inicio hasta que se acierta el número.



1. **Diagrama de Casos de Uso**

Es uno de los más utilizados por su claridad y sencillez, ya que muestra las  
funcionalidades esperadas del sistema y la interacción de los usuarios con cada una de ellas. “Los  
diagramas de casos de uso permiten identificar de forma temprana los requisitos funcionales de  
un sistema de manera sencilla y centrada en el usuario, representando mediante “actores” y  
“casos de uso” las interacciones principales que se esperan del software” (Fowler, 2003).

Aquí se representan las interacciones entre los participantes del juego. Por un lado, está el usuario, que responde si el número propuesto es mayor, menor o correcto; y por el otro, la computadora, que genera el número aleatorio y ajusta el proceso según las respuestas recibidas. Este diagrama ilustra de manera sencilla cómo se da la comunicación entre ambos actores a lo largo del juego.

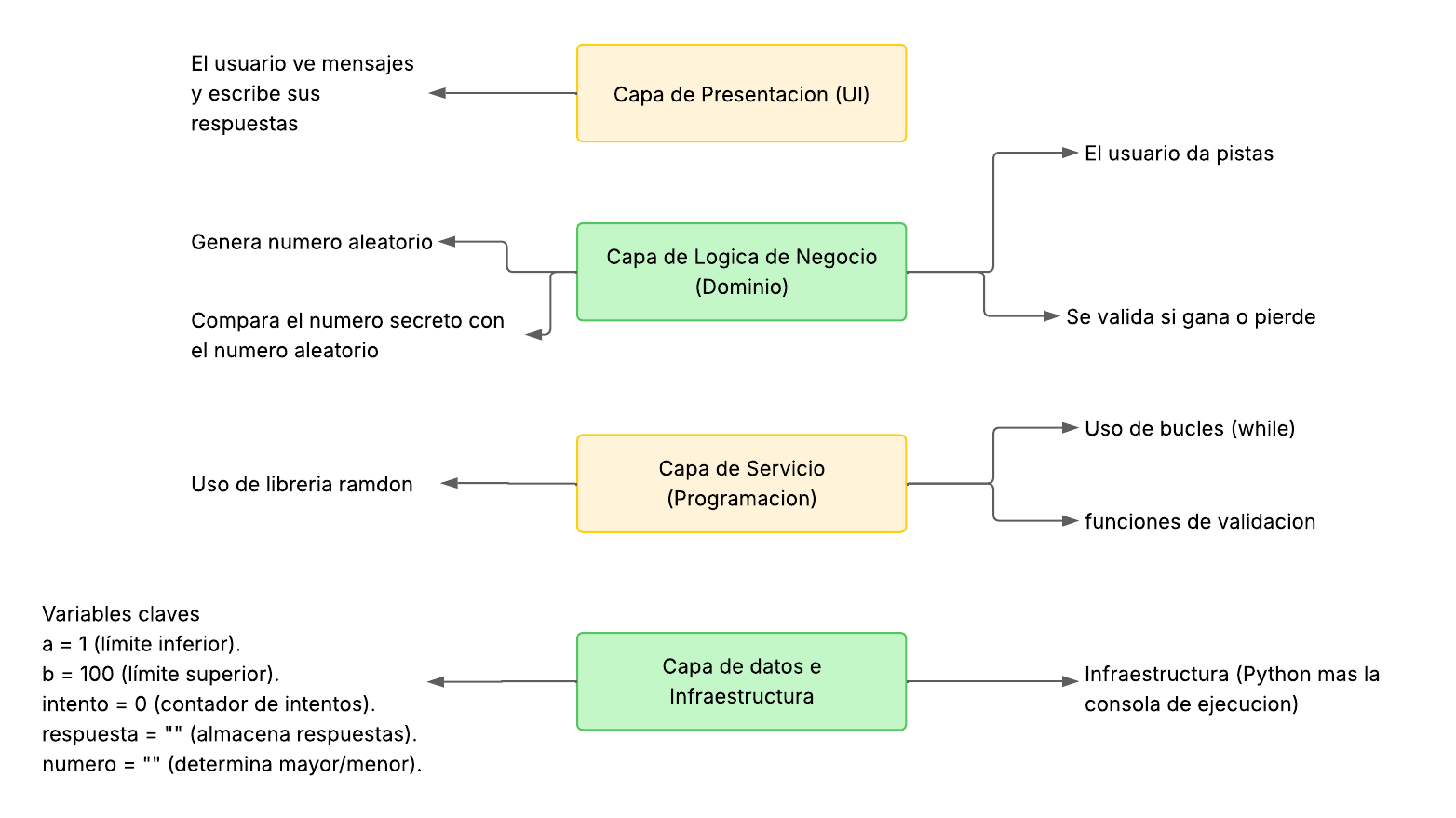


1. **Diagrama de Arquitectura en Capas**

Este diagrama organiza el funcionamiento del programa en diferentes niveles:

* **Presentación (UI):** corresponde al intercambio de mensajes entre el usuario y la computadora, esta capa se encuentra conformada por la interfaz de la consola, realiza distintas muestras de mensajes dependiendo de las respuestas del usuario y recibe las respuestas del mismo, por ejemplo, si el número es mayor o menor.
* **Lógica de Negocio/Dominio:** se encarga de manejar las reglas del juego, como validar si la respuesta es correcta o la reducción del rango, además de que es el encargado de generar el numero aleatorio y llevar el control del ciclo hasta finalizar el juego.
* Datos e Infraestructura: incluye las variables que guardan información clave del sistema, como el límite inferior y superior, el número generado y la cantidad de intentos.

Gracias a esta organización en capas, se separan claramente las responsabilidades del programa, lo que hace más sencillo comprender y modificar el sistema si es necesario.



**Conclusiones Finales del Proyecto**

Al finalizar este proyecto, logre comprender mucho mejor cómo las nuevas tecnologías afectan nuestra vida y la sociedad. Me permitió ver que los algoritmos y la automatización no son conceptos abstractos, sino herramientas que realmente influyen en decisiones y procesos diarios. Incluso un programa sencillo, como la computadora que adivina números, puede mostrar de forma clara cómo funciona la lógica detrás de la tecnología y generar reflexión sobre su uso.

Sobre la introducción y planificación del proyecto, aprendí que organizar cada paso antes de empezar es fundamental. Dividir el trabajo por unidades, temas y semanas hizo que cada actividad tuviera un objetivo claro y que todo el proyecto avanzara de manera ordenada. Esto me enseñó que planificar no es solo cumplir con un cronograma, sino una guía práctica para facilitar todo el desarrollo.

En relación con el programa, aunque sea básico, cumplió su función de mostrar cómo los algoritmos toman decisiones según datos y condiciones. Fue una forma práctica de aplicar la teoría y de comprender cómo un software, aunque pequeño, puede ser útil para aprender y reflexionar.

Respecto a la codificación, aprendí a cuidar la lógica, las estructuras y los comentarios, pensando siempre en cómo alguien más podría entender el código. Esto me ayudó a desarrollar precisión y claridad en la programación.

En cuanto a GitHub, subir el proyecto y organizar carpetas y archivos me enseñó a documentar un proyecto de manera profesional. Tener todo centralizado facilita la revisión y el seguimiento, y me mostró lo importante que es mantener un control de versiones y una buena organización para cualquier proyecto.

En general, este proyecto integró todos los conocimientos de las unidades: desde analizar el fenómeno tecnológico hasta codificar, hacer diagramas y documentar correctamente. La experiencia fue muy valiosa, aunque al inicio fue un poco abrumadora, porque ahora veo claramente cómo cada parte se conecta y tiene sentido dentro de un proyecto completo.

En conclusión, este proyecto no solo me enseñó programación y tecnología, sino también planificación, organización y reflexión sobre el impacto social de la tecnología. Me siento más segura para desarrollar proyectos futuros y más consciente de cómo la tecnología puede influir en la vida de las personas.

**Referencias**

Pressman, R. S. (2014). Software Engineering: A Practitioner’s Approach (8th ed.). McGraw-Hill Education.

<https://books.google.com.ec/books?id=Y7fyAgAAQBAJ>

Fowler, M. (2003). UML Distilled: A Brief Guide to the Standard Object Modeling  
Language (3rd ed.). Addison-Wesley.

[https://books.google.com/books?id=bZzYRQAACAAJ](%20https:/books.google.com/books?id=bZzYRQAACAAJ)