**TECNOLOGÍAS DE LA INFORMACIÓN**

**PRODUCCIÓN TRABAJO AUTÓNOMO**

**DESCRIPCIÓN DE PROYECTO FINAL DE ASIGNATURA**

**PRIMERA ETAPA**

|  |  |  |
| --- | --- | --- |
| **Asignatura:** Programación orientada a objetos | |  |
| **Nivel:** 3 | **Docente:** Edgardo Panchana | |
| **Fecha de Entrega:** SEMANA 4  **Tiempo de la actividad:** 20 horas autónomas | | |

|  |
| --- |
| **Tema: Vista general de su proyecto de asignatura** |
| **Objetivos:**   * Describir conceptualmente la aplicación de consola a desarrollar * Realizar un diagrama de clases de la aplicación de consola a desarrollar * Configurar el entorno de desarrollo integrando todas las herramientas necesarias para diseñar aplicaciones de consola * Evaluar habilidades blandas tales como el trabajo en grupo y la comunicación efectiva. |
| **Resultado de aprendizaje de la unidad:** Aplica los componentes básicos de abstracción y encapsulamiento de la Programación Orientada a Objetos en una aplicación  **Instrucciones generales:**  **Conformación de grupos**  El trabajo es grupal (máximo 5 integrantes).  La integración será por afinidad entre los estudiantes.  **Etapas del trabajo autónomo**  El proyecto consta de tres etapas:  Etapa conceptual: definición y maquetación de la aplicación.  Etapa practica: desarrollo de la funcionalidad básica.  Etapa avanzada: adaptación del proyecto a uno o varios patrones de diseño.  **Sobre el documento:**  El informe debe presentarse en Word (las observaciones se harán en comentarios).  Las siguientes etapas se añadirán al mismo documento.  No existe una plantilla obligatoria: cada grupo dará formato propio a su entrega.  **Sobre Git:**  Descargar e instalar Git desde <https://git-scm.com>  Revisar comandos básicos en: <https://rogerdudler.github.io/git-guide/index.es.html> o en el material del aula virtual.  **Sobre el repositorio en la nube:**  El código fuente debe alojarse en un servicio Git (se recomienda GitHub).  El proyecto debe ser privado.  Agregar al docente con el usuario: edgardopanchana.  Todos los integrantes deben estar añadidos y aportar con código.  **Sobre la revisión:**  Esta entrega equivale a 1,5 puntos en el primer parcial, sobre 10 puntos, según rúbrica.  El plagio será sancionado con cero.  El documento debe subirse al aula virtual en la semana 4 (según fecha publicada en Moodle).  Entregas atrasadas: máximo una semana después, con penalización del 50%.  Aunque el trabajo es grupal, cada estudiante debe subir el archivo a Moodle para recibir nota.  Se evaluará la calidad de los commits: si un integrante no aporta código al proyecto, su nota será cero. |
| **Descripción de las actividades a realizar:**  El proyecto es un prototipo por lo que el tipo de aplicación será de consola.  El lenguaje de programación a utilizar es TypeScript  El editor de código a utilizar es Visual Studio Code  No debe solicitar que el usuario del sistema ingrese datos, toda las creación de las instancias de clases debe realizarse a través de un archivo separado de las clases.  **Integrantes:**  Escriba a continuación los nombres y apellidos de los integrantes del grupo:  **1.**Delly Gustavo Pico Arteaga  **2.**Edisson Santiago Quimiz Romero  **3.**Joselyn Milena Zambrano Mendoza  **4.**Leiver Wilfrido Carrasco Anchundia  **5.**Michael Andres Noboa Curillo  **Primera Etapa – Entregables**   1. **Presentación y temática de la aplicación**    * Explicar la idea elegida, los criterios de selección, las necesidades detectadas y el enfoque del producto o servicio.    * Extensión libre. 2. **Requisitos funcionales**    * Lista clara con al menos 10 requisitos funcionales, identificando los más relevantes. 3. **Diagrama de clases**    * Captura del diagrama creado con extensiones como PlantUML, UML Extension for VSC, Draw.io o Mermaid Markdown Syntax. 4. **Código fuente inicial**    * Desarrollo del modelo de datos (clases con propiedades y métodos, aún sin funcionalidad).    * Incluir la URL del repositorio Git: [DellyGustavoPicoArteaga/TurnosDeUnBanco: Sistemas de Trunos en un Banco](https://github.com/DellyGustavoPicoArteaga/TurnosDeUnBanco)   **Habilidades blandas a evaluar:**  **Trabajo en equipo:** se entiende como la capacidad de los estudiantes para organizarse, distribuir tareas y colaborar de manera equilibrada en el desarrollo del proyecto.  Cómo se refleja:   * Todos los integrantes participan activamente tanto en el documento como en el repositorio de Git. * Se observa un aporte equilibrado en los commits, evitando que el trabajo recaiga solo en uno o dos integrantes. * El grupo muestra coordinación en la integración de sus partes (documento coherente, sin repeticiones ni contradicciones). * La afinidad entre integrantes facilita la colaboración y el cumplimiento de responsabilidades.   **Comunicación efectiva:** se mide en la forma en que el grupo logra expresar sus ideas en el documento escrito y en el repositorio.  Cómo se refleja:   * El documento está bien estructurado, con redacción clara, precisa y sin ambigüedades. * Los requisitos funcionales están redactados de manera comprensible para cualquier lector. * Se utilizan herramientas de forma correcta (ejemplo: comentarios en Word, mensajes claros en commits de Git). * El resultado final del documento transmite de manera ordenada las ideas del grupo, evidenciando que hubo entendimiento y acuerdos entre los integrantes. |
| **Recursos tecnológicos:**  **Software**: TEAMS, Visual Studio Code, Git  **Hardware**: Computador  **Recursos bibliográficos:**  <https://code.visualstudio.com/docs/typescript/typescript-tutorial>  <https://www.typescriptlang.org/docs/handbook/2/classes.html> |
| **RÚBRICA DE CALIFICACIÓN**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **PARÁMETRO** | **TOTAL** | **PARCIAL** | **NINGUNO** | **CALIFICACIÓN** | | **Presentación** | Explica con claridad y profundidad la temática, justificando su elección  **1 punto** | La explicación es poco clara o incompleta.  **0.5 puntos** | No se entiende la temática escogida  **0 puntos** |  | | **Requisitos funcionales** | Define correctamente al menos 10 requisitos funcionales, claros y relevantes.  **2 puntos** | Define algunos requisitos incorrectos o incompletos.  **1 punto** | No define requisitos funcionales  **0 puntos** |  | | **Diagrama de clases** | Presenta un diagrama coherente con la temática y requisitos.  **2 puntos** | Presenta diagrama, pero no concuerda con requisitos  **1 punto** | No presenta diagrama de clases  **0 puntos** |  | | **Código fuente** | Clases desarrolladas conforme al diagrama, con propiedades, constructores y métodos  **2 puntos** | Clases incompletas o no concuerdan con el diagrama.  **1 punto** | No presenta código fuente  **0 puntos** |  | | **Trabajo en equipo (habilidad blanda)** | Evidencia colaboración de todos los integrantes (commits distribuidos y aporte en el documento).  **2 puntos** | Participación desigual, algunos integrantes aportan poco.  **1 punto** | Solo uno o pocos integrantes realizan todo el trabajo.  **0 puntos** |  | | **Comunicación efectiva (habilidad blanda)** | El documento está bien estructurado, con redacción clara y lenguaje formal.  **1 punto** | El documento tiene errores de redacción o carece de coherencia.  **0.5 puntos** | El documento es confuso o muy desordenado.  **0 puntos** |  | | **TOTAL** | | | |  | |

**1- Presentación y temática de la aplicación**

**Idea Principal:**

La idea elegida es desarrollar un Sistema de Turnos para un Banco, este sistema permitirá que los clientes tomen un turno de manera ordenada, ya sea desde una pantalla dentro del banco o por medio de una aplicación web, evitando aglomeraciones y mejorando la organización del servicio al público.

**Criterios de selección:**

Esta idea fue seleccionada porque responde a una problemática común en muchas entidades bancarias, las largas filas y el desorden en la atención al cliente y también se consideró que:

Es viable ya que puede desarrollarse con conocimientos básicos de programación.

Puede implementarse en bancos reales o en otras instituciones que manejen atención por turnos.

Representa una mejora en la experiencia del usuario, al reducir tiempos de espera y permitir un mejor control del flujo de personas.

Favorece la automatización de procesos, lo que contribuye a la modernización del servicio bancario.

**Necesidades detectadas:**

Los clientes suelen experimentar incomodidad y pérdida de tiempo al esperar su turno sin saber cuánto tardarán en ser atendidos. Además, el personal del banco enfrenta dificultades para organizar la atención de manera justa y eficiente, especialmente en horas pico.

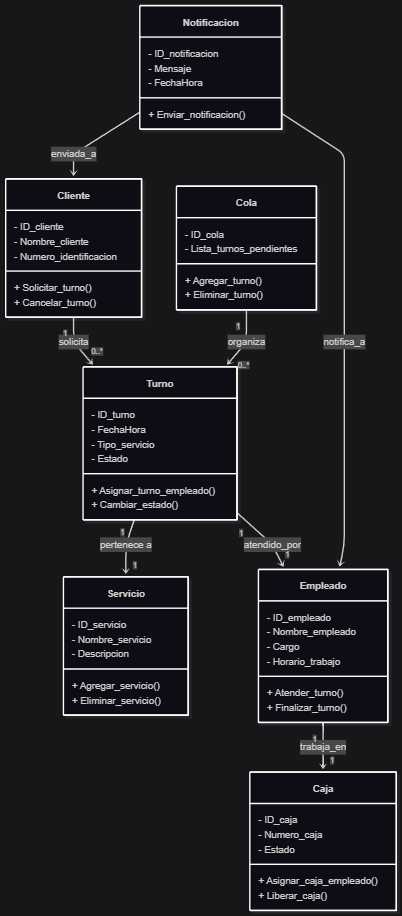
**Enfoque del producto o servicio:**

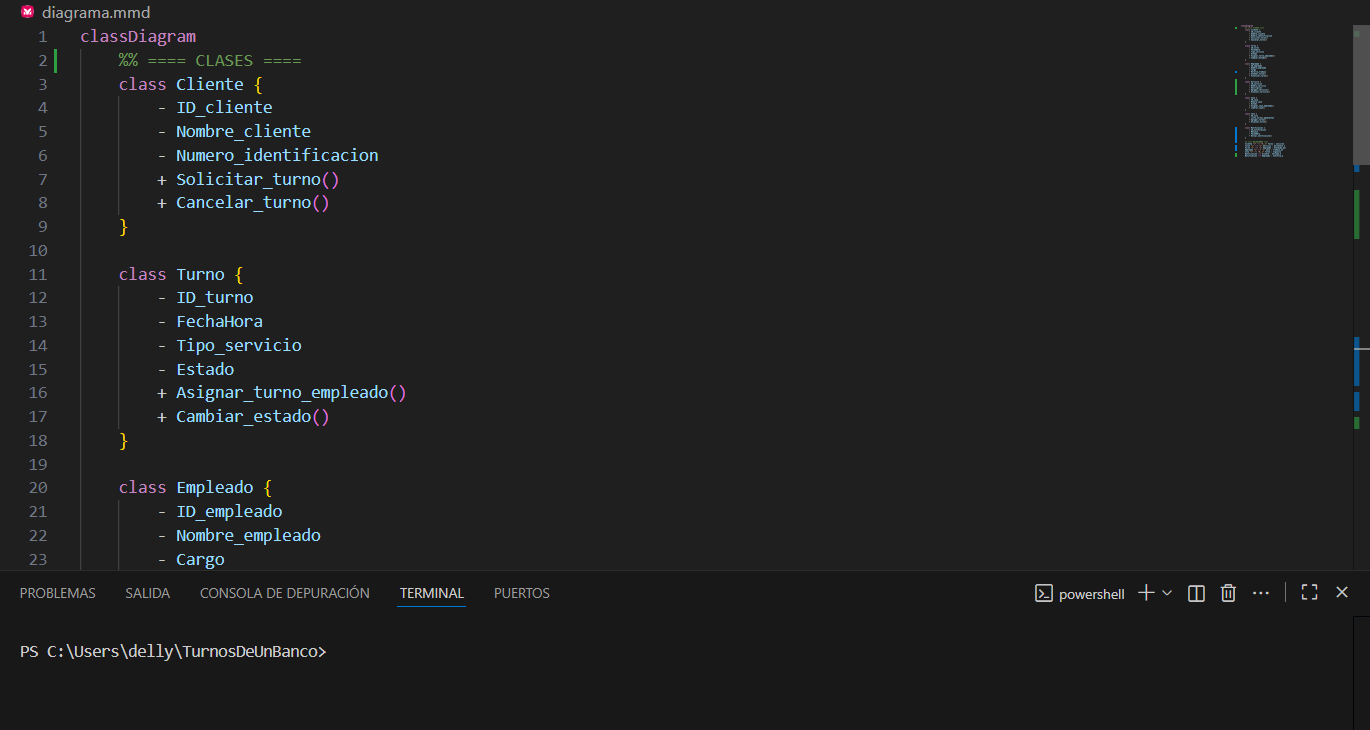
El enfoque del sistema es principalmente tecnológico y funcional, orientado a la optimización del servicio al cliente.

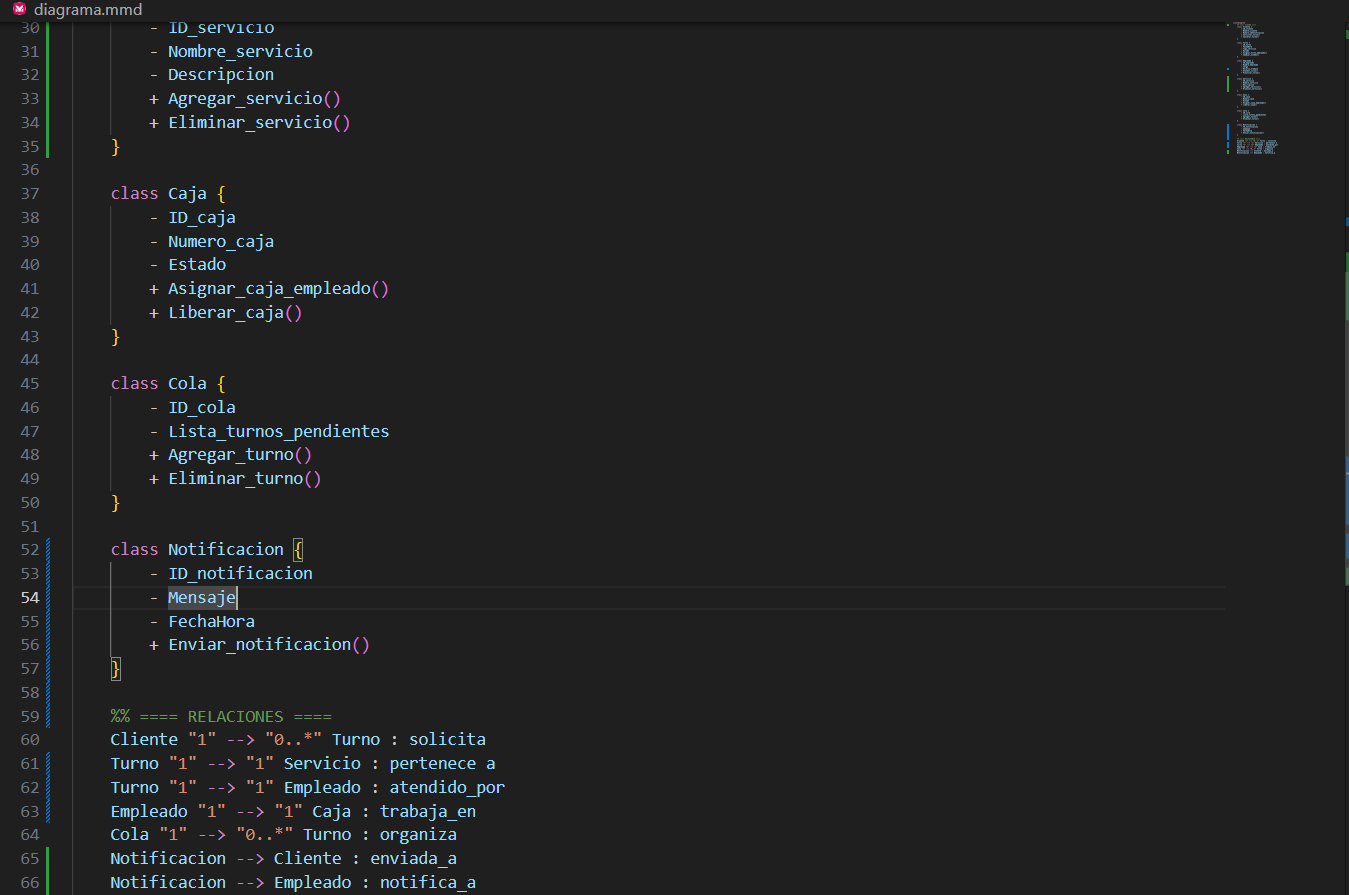
El sistema permitirá registrar la llegada de cada cliente, asignarle un número de turno y mostrar en una pantalla el orden de atención. Además, podrá incluir funciones como la selección del tipo de trámite, el seguimiento del turno en tiempo real y la notificación cuando el cliente deba acercarse al módulo correspondiente.

**2- Requisitos funcionales**

1. El sistema debe permitir registrar a los clientes que llegan al banco, ingresando sus datos básicos como nombre y tipo de trámite que desean realizar.
2. Cada cliente registrado debe recibir automáticamente un número de turno único que se genere de forma consecutiva.
3. El sistema debe mostrar en una pantalla los turnos en curso y los próximos por atender, junto con el número de módulo o ventanilla correspondiente.
4. El personal del banco debe poder llamar al siguiente cliente mediante el sistema, actualizando la pantalla principal y notificando el cambio de turno.
5. El sistema debe permitir asignar y controlar los turnos según el tipo de trámite o servicio que ofrece cada módulo.
6. El sistema debe guardar un registro de los turnos atendidos, incluyendo el tiempo de espera y el tiempo de atención de cada cliente.
7. El sistema debe permitir cancelar o reasignar un turno en caso de error o si el cliente no se presenta cuando es llamado.
8. Los empleados del banco deben iniciar sesión con un usuario y contraseña para acceder a las funciones de administración o atención.
9. El sistema debe contar con una interfaz gráfica fácil de usar, tanto para el cliente como para el personal del banco.
10. El sistema debe permitir generar reportes sobre el número de clientes atendidos, tiempos de espera promedio y módulos más utilizados.

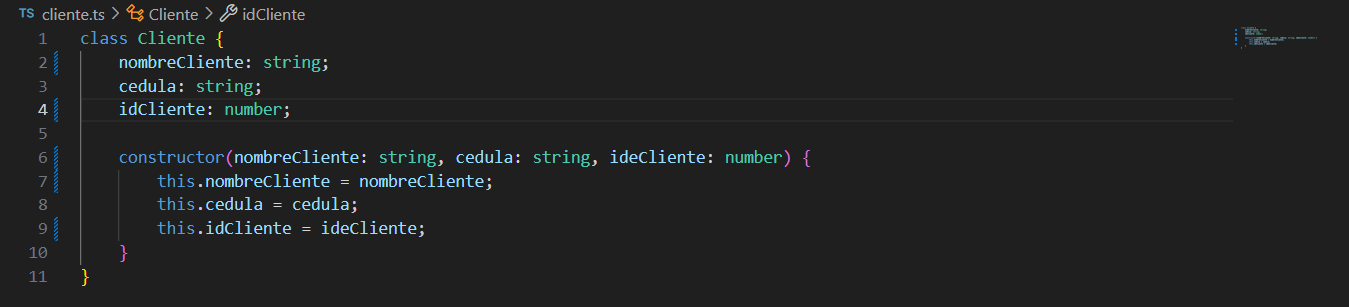
**3- Diagrama de clases**

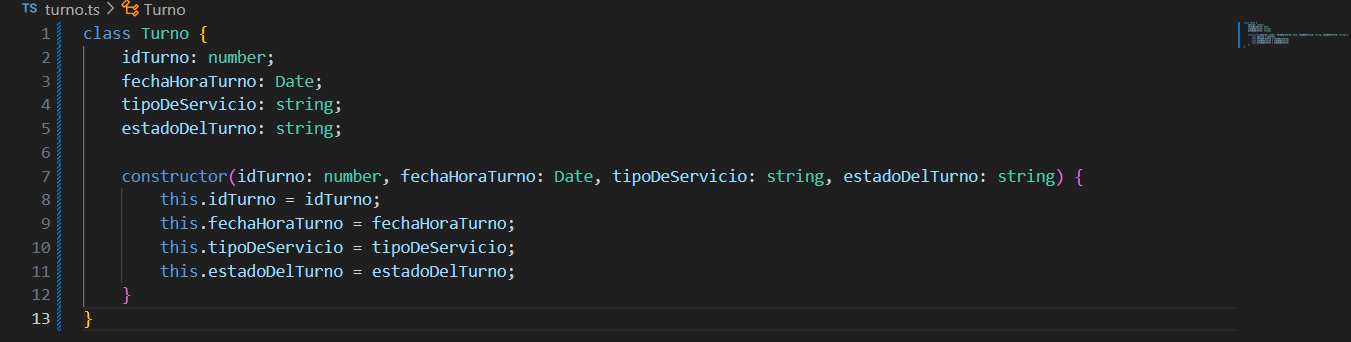


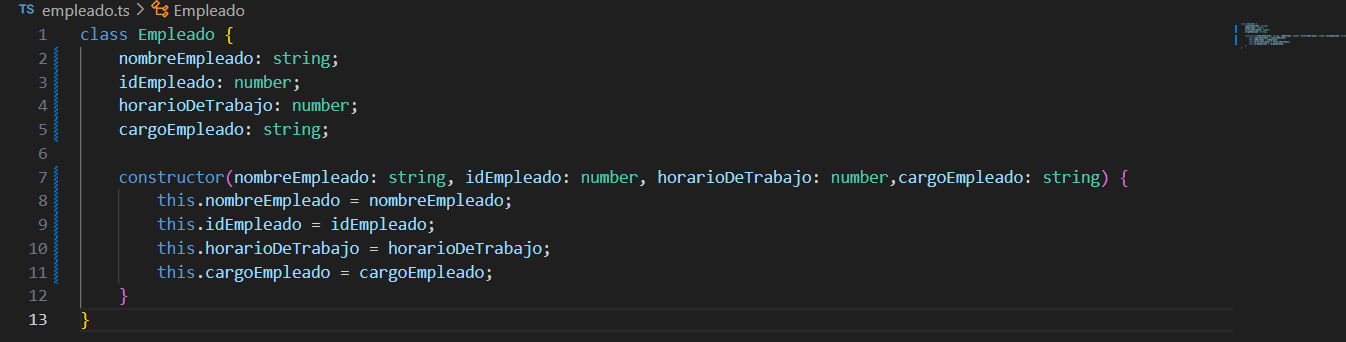
****

**Nota:** Para la creación del diagrama nos apoyamos de la IA utilizando las clases ya creadas por nosotros anteriormente.

**4- Código fuente inicial**

****

****

**  
Texto

El contenido generado por IA puede ser incorrecto.**

**Texto

El contenido generado por IA puede ser incorrecto.**

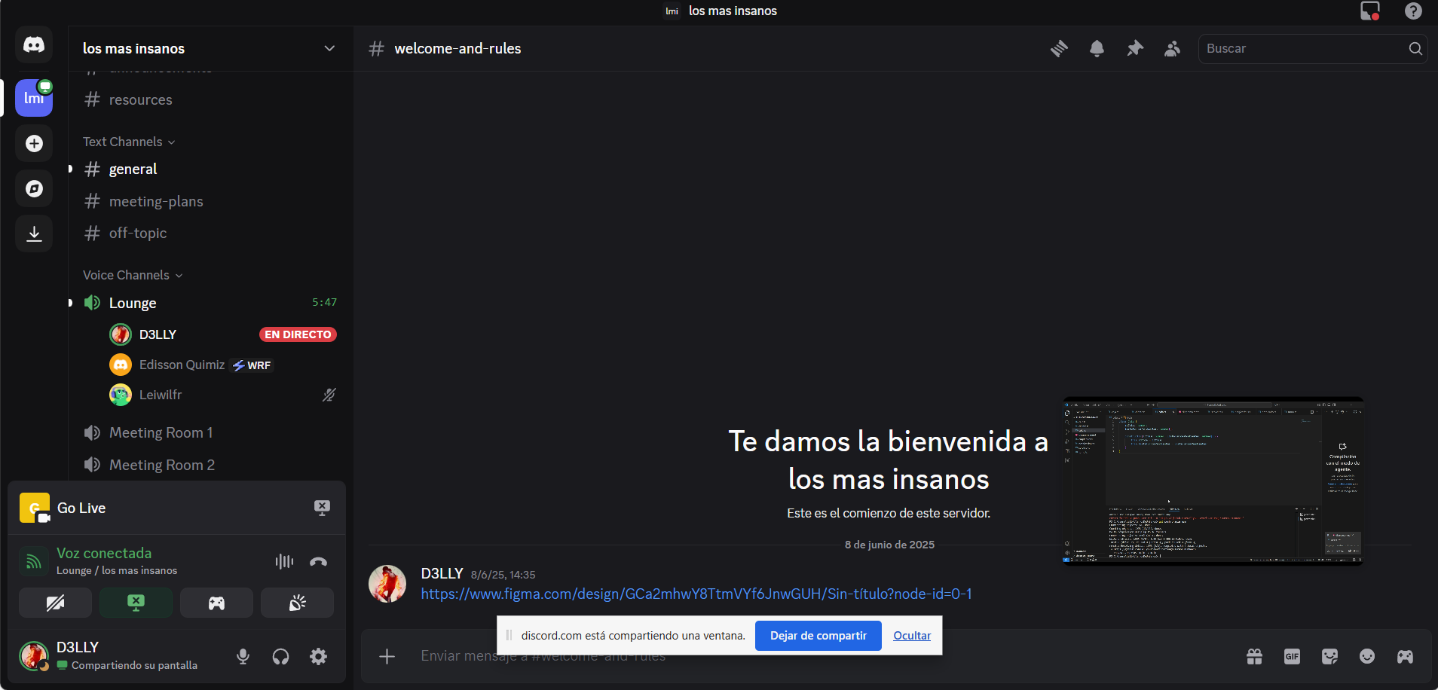
**Texto

El contenido generado por IA puede ser incorrecto.**

**Texto

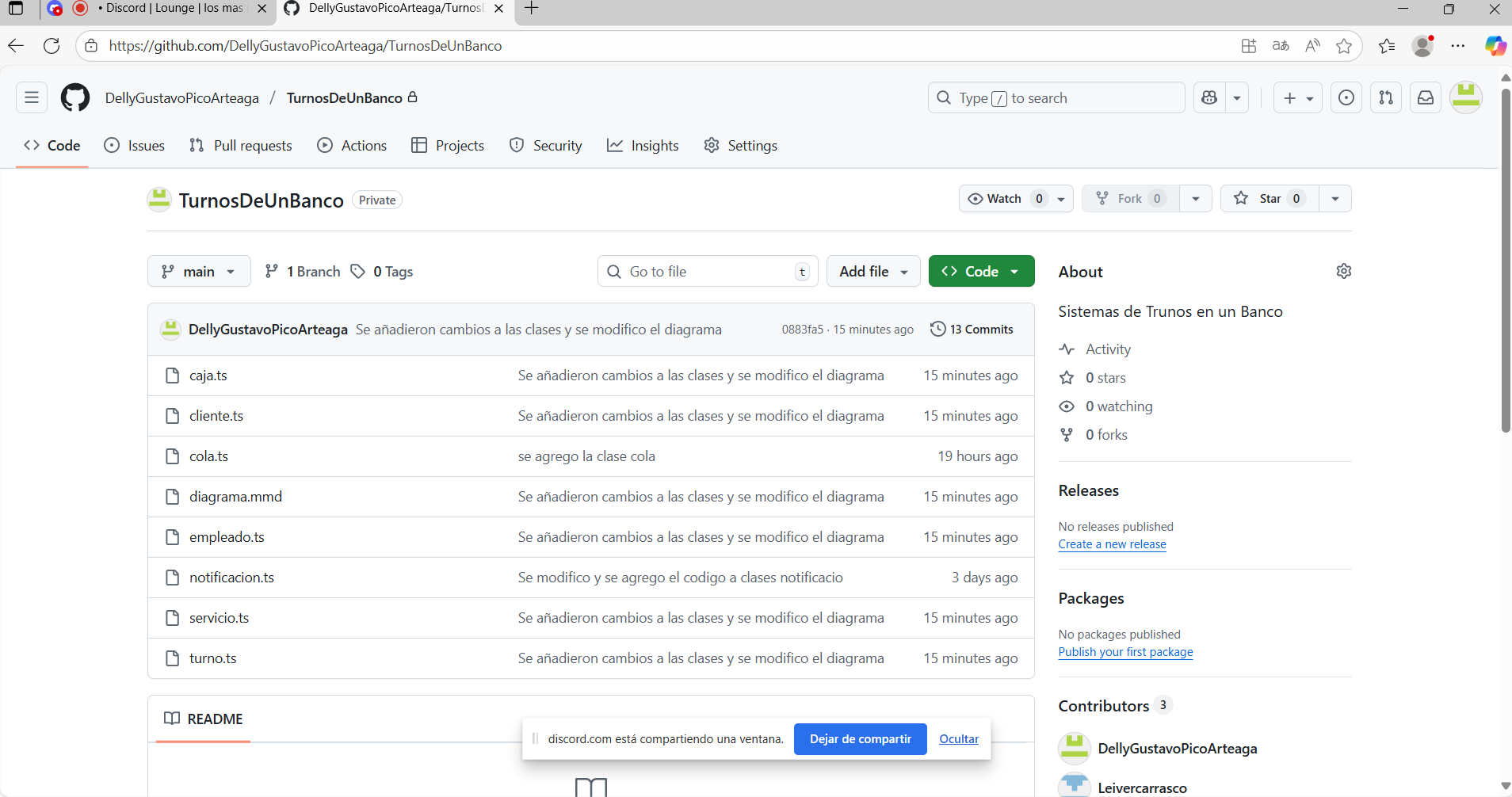
El contenido generado por IA puede ser incorrecto.**

**Anexos:**



Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.



**Link:** [**DellyGustavoPicoArteaga/TurnosDeUnBanco: Sistemas de Trunos en un Banco**](https://github.com/DellyGustavoPicoArteaga/TurnosDeUnBanco)