**Universidad de Santiago de Chile**

**Facultad de Ingeniería**

**Departamento de Ingeniería Informática**

**Paradigmas de Programación**

**Paradigmas de Programación**

**Proyecto semestral de laboratorio**

Versión 1.0 - actualizada al 22/03/2025.

# Descripción general

Durante el semestre se trabajará en distintos paradigmas de programación, los cuales serán puestos en práctica mediante distintos lenguajes de programación en tres entregas de laboratorio. El tema de laboratorio de este semestre se enfoca en la implementación de un juego de mesa basado en el intercambio y la compraventa de bienes raíces llamado CAPITALIA, el cual es similar a Monopoly, Gran Santiago, Mi Gran País o Mi Gran Capital, entre otros títulos del mismo tipo.



Los estudiantes desarrollarán soluciones a partir de los conceptos y principios de distintos paradigmas de programación. Estas soluciones permitirán a un grupo de jugadores(as) participar de una experiencia lúdica en que se simula el intercambio y compraventa de propiedades, además de situaciones que pueden afectar positiva o negativamente la fortuna de los jugadores. En su versión final la implementación del juego tendrá una interfaz de usuario a través de la cual se debe poder jugar a través de la línea de comandos.

Los aspectos de implementación del laboratorio quedan abiertos a la creatividad del estudiante, siempre cumpliendo con los requisitos obligatorios, los cuáles se presentan en las siguientes secciones y cada enunciado de laboratorio tiene un conjunto de requerimientos que debe ser abordado bajo el paradigma correspondiente.

# Enunciado[[1]](#footnote-0)

**CAPITALIA** es un juego de mesa de estrategia y negociación en el que los(as) jugadores(as) compiten por acumular riqueza comprando, vendiendo y administrando propiedades en un tablero. El objetivo principal es adquirir la mayor cantidad de propiedades posibles y llevar a la bancarrota a los demás jugadores mediante el cobro de rentas y el uso inteligente de los recursos financieros.

A diferencia de juegos como Monopoly y otros títulos similares, en CAPITALIA el sistema de reglas y variables económicas que impactarán el desarrollo del juego son dinámicos. En este juego se implementarán mecánicas avanzadas, como fluctuaciones en los mercados que pueden afectar impuestos, cambios en el valor de las propiedades y variaciones en las rentas y tributos, agregando un nivel adicional de estrategia e incertidumbre.

Los estudiantes deberán diseñar e implementar una versión funcional del juego, asegurando que los jugadores puedan comprar y vender propiedades, pagar y recibir rentas, gestionar sus recursos financieros y adaptarse a los cambios del mercado impuestos por el sistema.

### **Reglas generales del juego**

Las reglas generales de CAPITALIA se describen a continuación:

Cabe destacar que cada enunciado del laboratorio es una adaptación de las reglas de CAPITALIA, donde algunos aspectos pueden no ser tomados en cuenta.

### **1. Creación de una partida e inicio del juego**

* Al comienzo del juego se crea un tablero, se añaden todas las propiedades, las cartas de fortuna, de arca comunal, cantidad de dinero circulante, cantidad de capital inicial para jugadores, número de dados (entre 1 y 4), impuesto inicial, cantidad máxima de casas y de hoteles, entre otros elementos dinámicos del juego.
* Cada jugador elige una ficha y la coloca en la casilla de "Salida".
* El banco es administrado automáticamente por el sistema
* Se reparte una suma inicial de dinero (capital) a cada jugador

**2. Turnos y Movimiento**

* Los jugadores lanzan de 1 a 4 dados. El número de dados se define al comienzo del juego.
* Si un jugador saca dobles, triples o cuádruples (todos los dados con el mismo número), juega otro turno.
* Si saca dobles, triples o cuádruples tres veces seguidas, va directamente a la cárcel.
* Al caer en una casilla, el jugador debe seguir la acción correspondiente (comprar, pagar renta, extraer carta, etc.).

### **3. Compra de Propiedades**

* Si un jugador cae en una propiedad no comprada, puede adquirirla al precio indicado en la tarjeta.
* Al comprar una propiedad, el jugador recibe el título correspondiente.

### **4. Pago de Renta**

* Si otro jugador cae en una propiedad comprada, debe pagar la renta al dueño según lo indicado en la tarjeta.
* Si la propiedad tiene casas u hoteles, la renta aumenta.

### **4. Pago de Impuestos**

* Al recorrer el tablero completo y llegar a la salida, los(as) jugadores(as) deben pagar impuestos por el total de propiedades a su haber incluyendo las construcciones en cada una de ellas.

### **5. Construcción de Casas y Hoteles**

* Un jugador puede construir casas en sus propiedades durante su turno.
* Para efectos del laboratorio 1, construir una casa sólo se puede dar si i) la propiedad ya se ha comprado y ii) el precio de construir una casa es el mismo de la propiedad
* Al llegar al número máximo de casas (se define al momento de crear el tablero, para efectos del laboratorio 1 es global para todas las propiedades), se puede construir un hotel.

### **6. Cartas de Suerte y Comunidad**

* Al caer en una casilla de "Suerte" o "Comunidad", el jugador toma una carta y sigue las instrucciones (pagar, recibir dinero, moverse, etc.).

### **7. Ir a la Cárcel**

En el contexto de CAPITALIA, estar en la cárcel implica:

* Perder el turno de movimiento hasta salir de la cárcel.

Un jugador va a la cárcel si:

* Cae en la casilla "Ir a la cárcel".
* Saca dobles tres veces seguidas.
* Saca una carta que lo manda a la cárcel.

Para salir de la cárcel, el jugador puede:

* Pagar una multa.En el contexto de CAPITALIA se debe pagar una multa de $500.
* Usar una carta de “Salir de la Cárcel Gratis”.
* Sacar dobles en los siguientes tres turnos.

### **8. Hipotecas y Préstamos**

* Un jugador puede hipotecar propiedades para obtener dinero del banco.
* Para levantar una hipoteca, se paga el valor de la hipoteca más intereses.

### **9. Bancarrota y Eliminación**

* Un jugador entra en bancarrota si no puede pagar una deuda.
* Si debe dinero a otro jugador, le entrega sus propiedades.
* Si la deuda es con el banco, sus propiedades se subastan.
* El último jugador con dinero y propiedades gana la partida.

### **Funcionalidades a Implementar (generales del juego CAPITALIA, luego cada laboratorio tiene sus propia versión o implicancia de cada funcionalidad)**

1. **Tablero dinámico**
   * Se pueden incorporar o eliminar propiedades al tablero
2. **Gestión de jugadores**
   * Registro y control de jugadores con nombres y capital inicial.
   * Sistema de turnos secuenciales y reglas de movimiento (avanzar o retroceder) según dados o tarjetas de fortuna.
3. **Compra, venta e hipoteca de propiedades**
   * Permitir la adquisición de propiedades según el capital disponible.
   * Posibilidad de vender propiedades a otros jugadores o al banco.
   * Posibilidad de hipotecar propiedades al banco.
4. **Sistema de rentas y tributos**
   * Cobro automático de rentas cuando un jugador cae en una propiedad ajena.
   * Pago de impuestos y tributos según las reglas del juego.

1. **Fluctuación de impuestos y mercado**
   * Implementación de eventos aleatorios o periódicos que alteren los impuestos y tributos (en base a tarjetas de arca comunal y fortuna).
   * Ajuste del valor de propiedades en función de cambios en la economía del juego.
2. **Eventos económicos y políticos**
   * Introducción de cartas o eventos que modifiquen reglas temporalmente (ej. crisis económica, boom inmobiliario, exenciones fiscales).
3. **Banca y gestión de dinero**
   * Transacciones automáticas de compra, pago de rentas y tributos.
   * Posibilidad de solicitar préstamos o recibir subvenciones en situaciones específicas.
4. **Condiciones de victoria**
   * Definir criterios de victoria (ej. jugador con mayor riqueza tras X turnos, bancarrota de los demás jugadores, entre otros).

# Referencias

Hasbro. (s.f.). Connect Four [Instrucciones del juego].

https://hasbrostore.cl/product/monopoly/206-monopoly-clasico-630509528134.html?srsltid=AfmBOop-CrHJbA3DiBNawZ0nlijv3FhSDMevy-5ouYIsN-eavi-hW7jW

# 

# Integración con Google Gemini API

En cada enunciado de laboratorio, ya sea funcional, lógico u orientado a objetos, un requerimiento involucra realizar una integración con un servicio de inteligencia artificial generativa (GenAI). En este contexto se empleará Google Gemini debido a que permite uso gratuito sin ingresar tarjetas de crédito ni información personal. Puede considerar el uso de otros servicios similares previa consulta y autorización del profesor encargado del laboratorio.

Para lograr una integración Google Gemini primero debe ingresar a [prompt | Google AI Studio](https://aistudio.google.com/app/prompts/new_chat) usando su cuenta de correo Usach o su correo electrónico personal gmail. Para este laboratorio no es relevante con que tipo de correo gmail crea una cuenta en Google AI Studio.

Posteriormente se mostrará en pantalla el inicio de Google AI Studio (Figura 6). Esta es una plataforma con la cuál puede obtener la API KEY. Una API KEY es un clave propia generada por la plataforma para su cuenta y permite a distintos programas, incluyendo su laboratorio interactuar con la plataforma.

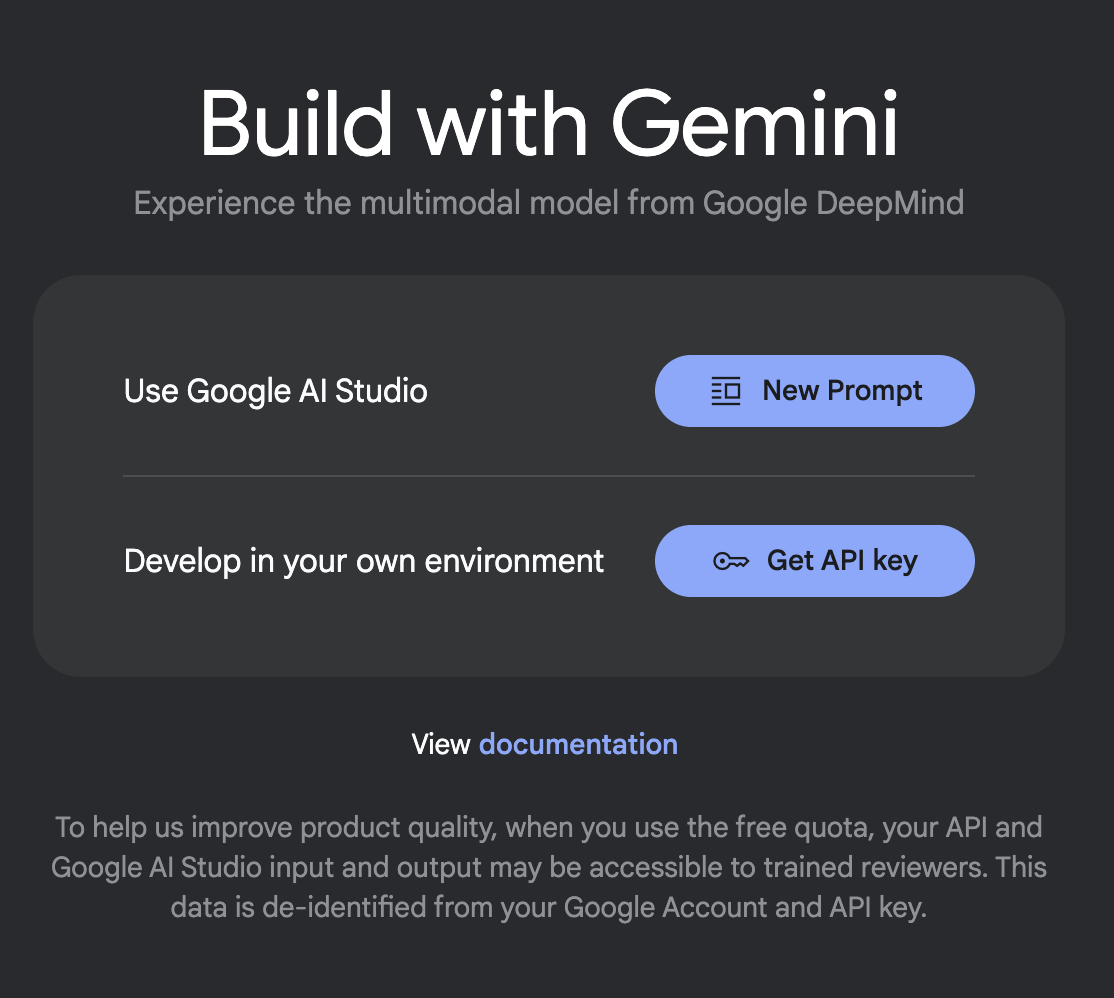
****

Figura 6. Pantalla inicio Google AI Studio

Debe hacer click en **"Get API Key"**

Al hacer click en **"Get API Key"**. Se cargará la pantalla "obtén una clave de API", es en esta página que usted debe hacer click en **"Crear clave de API"** (Figura 7). Esto generará una clave que usted puede utilizar para integrar cualquier programa con el servicio de Google Google Gemini.

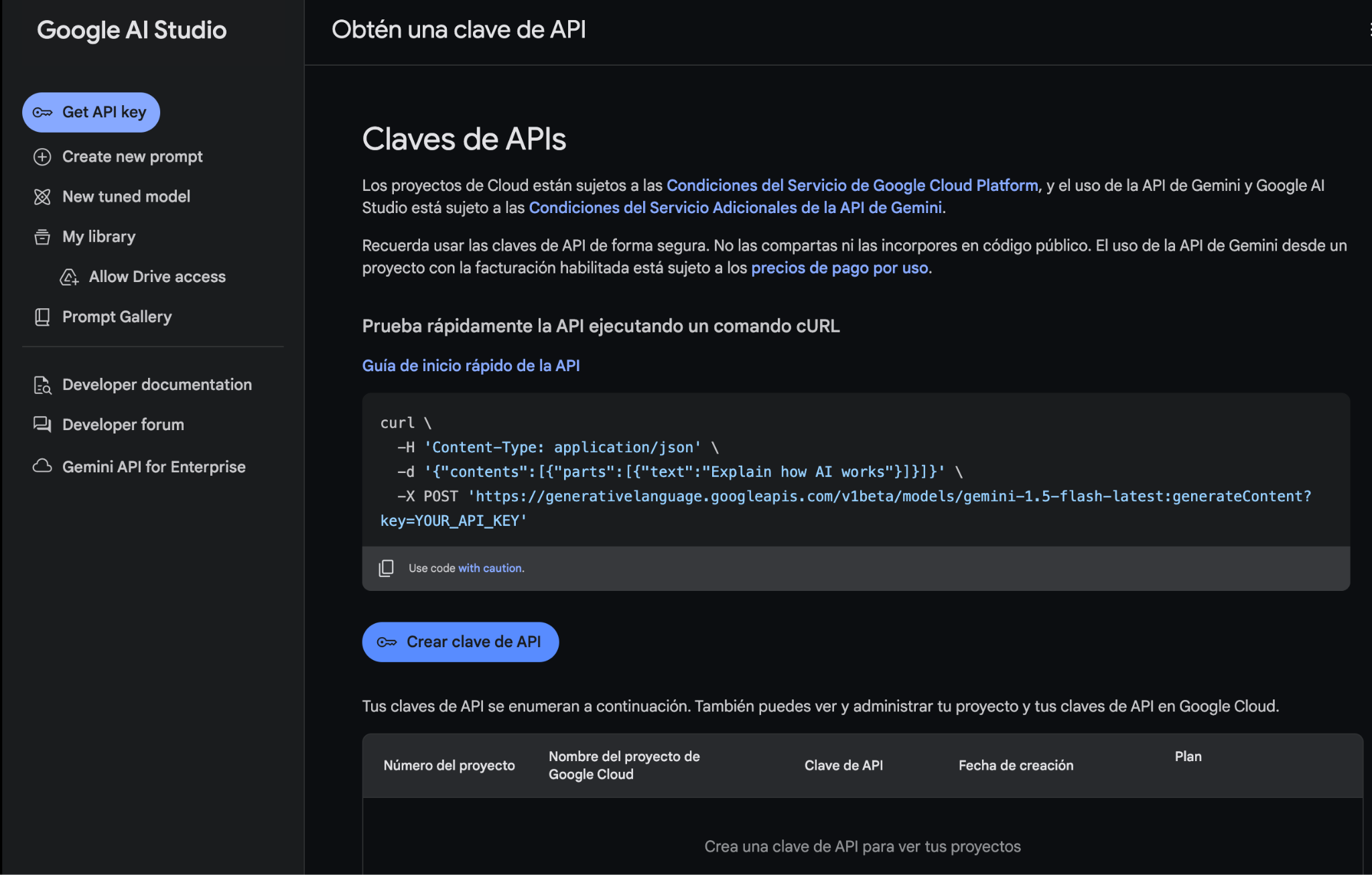
****

Figura 7. Generación clave API en Google AI Studio

Posteriormente aparecerá un aviso legal el cuál usted sólo debe presionar el botón **"Continuar"** (Figura 8). De igual forma se mostrará un aviso de safety setting reminder (Figura 9) el cuál debe presionar "Continuar".

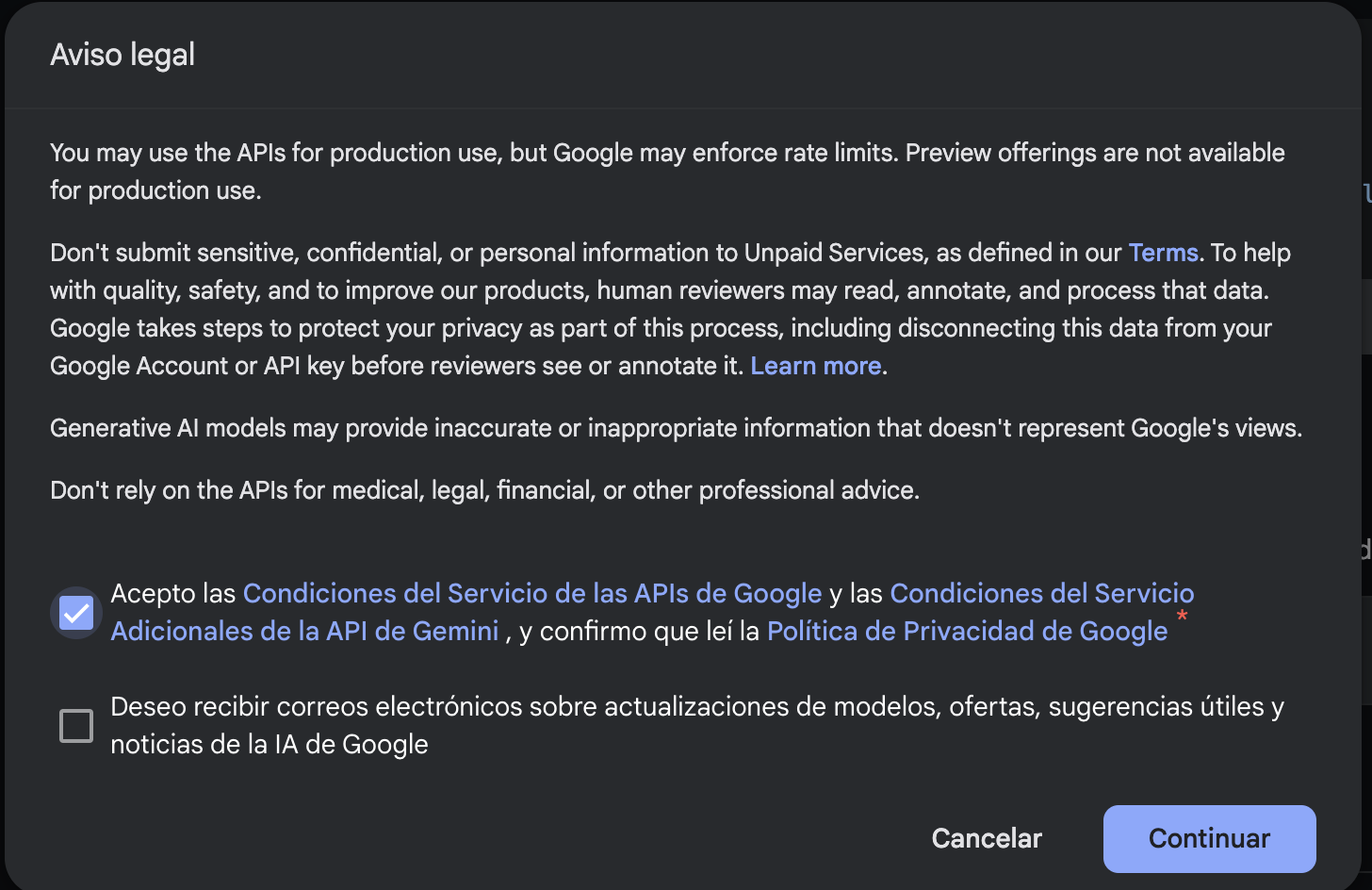
****

Figura 8. Aviso Legal Google AI Studio

## 

Figura 8. Aviso Legal Google AI Studio

Posteriormente aparecerá una ventana de creación de la API KEY. Es importante seleccionar la opción **"Crear clave de API en un proyecto nuevo"** (Figura 9 y Figura 10 donde el servidor comienza la creación de la API KEY).

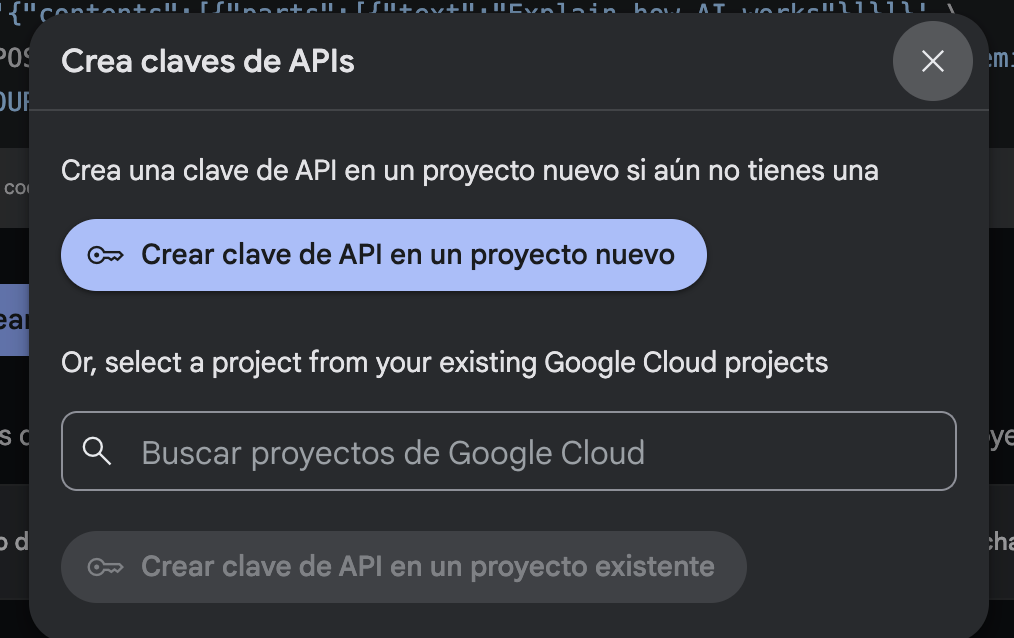


Figura 9. Crear clave de API en un proyecto nuevo

Posteriormente aparecerá una ventana de creación de la API KEY. Es importante seleccionar la opción **"Crear clave de API en un proyecto nuevo"** (Figura 9 y Figura 10 donde el servidor comienza la creación de la API KEY).

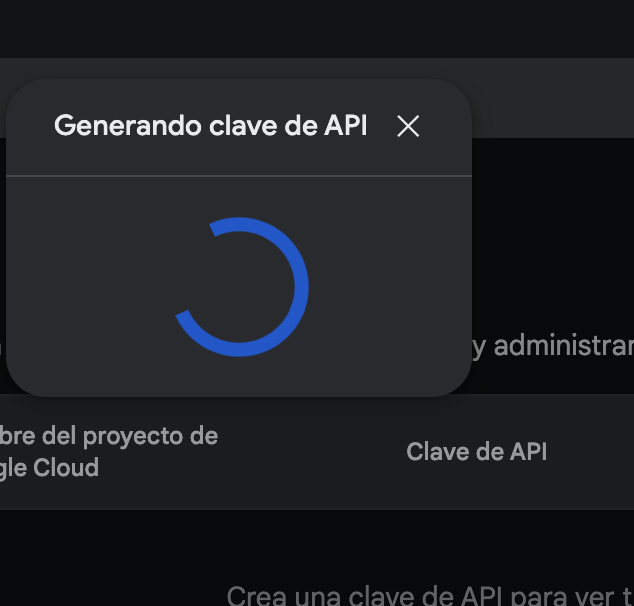


Figura 10. Estado de la generación de la API KEY

Finalmente la clave de la API (API KEY) se generará y mostrará en pantalla la clave generada. Es importante que usted copie y guarde dicha información. Esto se evidencia en la Figura 11 -por temas de privacidad la clave generada para realizar este tutorial fue marcada en rojo-. Es importante notar que la clave API KEY no puede ser compartida entre varios estudiantes, cada estudiante debe generar su propia API KEY.

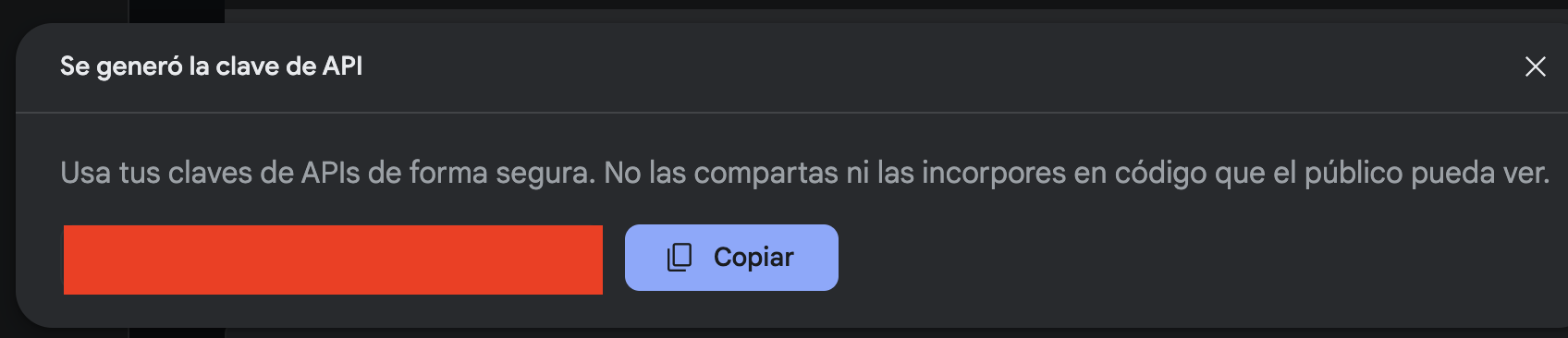


Figura 11. API KEY generada

La clave generada quedará disponible en su página inicial de la plataforma Google AI Studio (Figura 12).

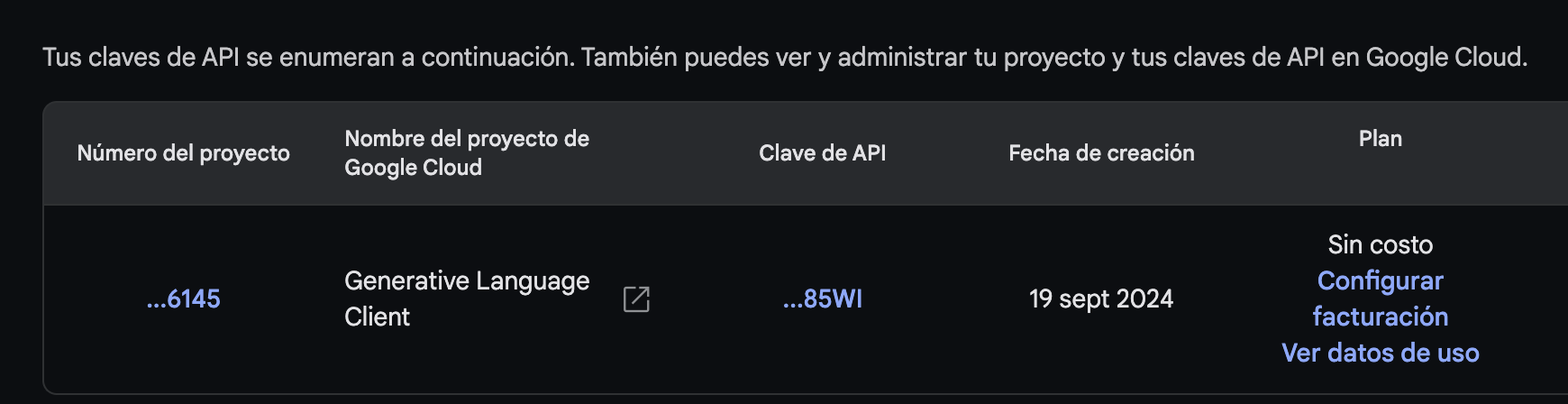


Figura 12. Resumen de las API KEY generadas.

**Nota: usted puede generar múltiples API KEY pero no es relevante para este laboratorio.**

Es responsabilidad del estudiante buscar cómo usar este servicio en cada laboratorio y cómo resolver los requerimientos solicitados en cada enunciado. En clases no se revisará ninguno de estos aspectos por lo que la investigación y consultar documentación, entre otros, es responsabilidad del estudiante.

# Comodines

**Comodines:** Durante el semestre cada estudiante dispondrá de **TRES comodines (uno para cada laboratorio)** que podrá utilizar **solo en el siguiente caso**:

* **Si su calificación** en los laboratorios 1, 2, 3 (Ya sea por requerimientos funcionales, no funcionales, informe) **es inferior a 4.0 y superior o igual a un 2 (es decir, que entrego un avance del proyecto)**.

**No entregar laboratorio ni entregar algún aspecto de él será calificado con un 1.0 y por tanto no tendrá derecho a usar comodín ni apelación.**

El comodín permitirá a el/la estudiante entregar lo necesario (requerimientos funcionales, no funcionales e/o informe) para alcanzar la nota mínima requerida (4.0) en la correspondiente entrega de laboratorio para la aprobación al final del semestre. Además, la nota máxima a la que puede optar por uso de comodín es un 4.0. **Es fundamental que lo/as estudiantes siempre entreguen un avance (dentro de los plazos establecidos para cada laboratorio) que permita alcanzar al menos una calificación igual o superior a 2.0 para poder optar al uso de comodines**. **Recordar que para aprobar el laboratorio de esta asignatura, todos los laboratorios deben tener una calificación igual o mayor a 4.0. En caso de que su calificación para un laboratorio sea menor a 2.0 debido a que en alguno de los entregables (ej.: código, versionamiento, informe, etc.) su calificación fue inferior a 2.0, quedará a criterio del profesor encargado de la revisión (en base al nivel de logro en otros entregables) si aplica o no el uso de comodín.**

**Se recomienda no emplear los comodines como una forma de aplazar la entrega de sus laboratorios, puesto que irá acumulando trabajo hacia el final de semestre. Por lo mismo, procure trabajar en el laboratorio dentro de los plazos señalados y entregarlo aún cuando no haya podido terminarlo.**

**Para el caso del uso de comodín por versionamiento solo dispondrá de 2 oportunidades. Si el estudiante tiene una calificación menor a 4.0 debido al no uso o uso inadecuado de git (frecuencia, constancia, periodo), el estudiante podrá enmendar esta calificación haciendo uso efectivo de git en los siguientes laboratorios o en los anteriores. Para estos casos, el estudiante debe enviar la solicitud de aplicación de comodín por versionamiento al término del semestre a través de la plataforma habilitada para tales efectos. De ser aceptado el comodín, la nota de cierre del correspondiente laboratorio será un 4.0.**

**Procure realizar la autoevaluación rigurosa** de cada laboratorio de manera que conozca a priori la calificación estimada que podría recibir. De esta manera se espera que tengan claridad sobre si debe o no usar comodín y en qué funcionalidades debería concentrarse para alcanzar la nota mínima. Las calificaciones de laboratorio tomarán tiempo (en el calendario Clase a Clase se presenta una fecha estimada donde debería recibir las calificaciones de cada laboratorio), por lo que se recomienda NO esperar a tener una calificación del laboratorio para recién determinar si requiere usar un comodín. Esto lo puede determinar a priori con una autoevaluación rigurosa y a conciencia. **Al respecto señalar que la evaluación de cada laboratorio está a cargo de un solo profesor, por lo que es probable que las calificaciones de un laboratorio no estén antes de la entrega del siguiente.**

Si su autoevaluación indica que su calificación hubiese sido mayor o igual a 4.0, se recomienda a lo/as estudiantes que empleen las instancias de recorrecciones/apelaciones de cada laboratorio antes de proceder a trabajar en el comodín. Si su autoevaluación fue a conciencia y rigurosa, pero aún así obtiene una calificación inferior a 4.0, podría tratarse de (1) un error/omisión en la revisión, (2) problemas en la implementación/ejecución que no fueron identificados por el/la estudiante, (3) problemas de versionamiento, interpretación, compilación debido a elementos que no fueron debidamente documentados y (4) aplicación incorrecta del paradigma, entre otras razones.

Finalmente, en caso de requerir el uso de comodín, podrá hacer entrega del correspondiente laboratorio corregido al final del semestre en las fechas y canal estipulados para estos efectos. La fecha específica de entrega de comodines se encuentran señalados en el Clase a Clase, mientras que los espacios de entrega para cada comodín estarán disponibles en CampusVirtual.

# Instrucciones generales para la entrega de TODOS los laboratorios

Versión 1.0

(Cambios menores pueden incorporarse en futuras versiones a fin de aclarar o corregir errores)

(Sus dudas las puede expresar en este mismo enunciado, incluso puede responder a preguntas de compañeros en caso de que conozca la respuesta)

**Entregables:** Archivo ZIP con el siguiente nombre de archivo: labN\_rut\_ApellidoPaterno.zip (ej: lab1\_12123456\_Perez.zip). **Notar que no incluye dígito verificador**. Dentro del archivo se debe incluir:

1. Informe en formato PDF.
2. Carpeta con código fuente.
3. Archivo leeme.txt para cualquier instrucción especial para ejecución u otro, según aplique.
4. Autoevaluación.txt donde debe incluir una lista completa de todos los requerimientos funcionales y no funcionales señalando una evaluación para cada uno según la siguiente escala:
   1. 0: No realizado.
   2. 0.25: Implementación con problemas mayores (funciona 25% de las veces o no funciona)
   3. 0.5: Implementación con funcionamiento irregular (funciona 50% de las veces)
   4. 0.75: Implementación con problemas menores (funciona 75% de las veces)
   5. 1: Implementación completa sin problemas (funciona 100% de las veces)
5. Archivo repositorio.txt que incluya la URL de su repositorio privado en GitHub[[2]](#footnote-1) el que deberá además compartirlo con los profesores y ayudantes al momento de la evaluación a la cuenta ***paradigmasdiinf*** de GitHub (no lo comparta antes ya que las invitaciones expiran después de un tiempo). **El nombre final del repositorio a compartir debe ser el mismo del entregable en Campus Virtual.**

El nombre de todos los archivos con código fuente, informes, autoevaluación, etc. debe seguir el siguiente formato: NombreArchivo\_rut\_Apellidos.extension (ej: labDobble\_12123456\_PerezPeña.rkt; funciones\_12123456\_PerezPeña.rkt)

**Se aplicarán descuentos en caso de no incluir estos datos e inclusive la nota mínima si el programa no puede pasar por los procedimientos automáticos de revisión por no respetar esta instrucción.**

**Informe del proyecto:** Extensión no debe superar las **7 planas** de contenido (se excluyen: portada, índice, referencias en formato APA, anexos), para más detalle de requerimientos del informe [ver descripción en nuestro espacio en Campus Virtual](https://uvirtual.usach.cl/moodle/mod/page/view.php?id=156475). El informe debe incluir introducción, descripción breve del problema, análisis del problema, diseño de la solución (esquematizada y explicada brevemente), consideraciones de implementación (ej: algoritmos, bibliotecas), **instrucciones con ejemplos claros de uso**, resultados obtenidos, evaluación completa y conclusiones.

**Repositorio en Git del Proyecto:** El laboratorio deberá ser desarrollado incluyendo el sistema de control de versiones Git y utilizando GitHub como servidor remoto. **El repositorio debe ser privado durante TODA la elaboración del proyecto**. Cualquier inconsistencia entre lo entregado en Campus Virtual y la versión final del repositorio será evaluado con nota mínima. **Para el proceso de revisión se considerará la versión subida a CampusVirtual**.

**IMPORTANTE: el nombre de cada repositorio en cada laboratorio debe contener la siguiente información:**

**LabX\_Rut\_NombreApellidos**

**Ejemplo, el nombre del repositorio 1: Lab1\_20123123K\_JuanArayaPerez**

**En caso de que su repositorio no tenga la información previamente descrita, no se podrá evaluar su repositorio, en consecuencia obtendrá la nota mínima.**

**Integridad:** Lo/as estudiantes deben desarrollar los laboratorios de forma individual. Si bien lo/as estudiantes pueden apoyarse y discutir aspectos del proyecto, el diseño e implementación del mismo, la solución construida se debe producir de manera individual. No se pueden compartir diseños o implementaciones, tampoco se pueden obtener de terceros. Cómo parte de la revisión existen procedimientos de revisión de copia/plagio tanto a nivel de código como en los informes. **Al detectarse plagio, el trabajo en cuestión no será revisado por no contar con evidencia concreta sobre el resultado de aprendizaje esperado para el correspondiente laboratorio. Por lo anterior, será calificado con un 1.0. El/la estudiante tampoco podrá hacer uso de comodín en este caso ya que es el equivalente a no haber entregado dicho laboratorio**. **Por último se iniciará investigación sumaria a través de la Facultad de Ingeniería con el objeto de determinar las sanciones correspondientes.**

**Sobre el uso de Inteligencia Artificial Generativa (GenAI)** **queda estrictamente prohibido el uso de GenAI para abordar aspectos de diseño e implementación de su laboratorio**. Por otro lado, para la confección del informe los posibles usos de GenAI se deben regir por lo señalado en el siguiente [instructivo](https://docs.google.com/document/d/1gwq42_wvBn4-FkYrI6eN-s37WjR_LWHmsclTeW-Xfcc/edit?tab=t.0).

**En caso de detectar uso de alguna herramienta GenAI en código se calificará con la nota mínima.**

**Evaluación:** El Informe (Inf), requerimientos funcionales (RF), requerimientos no funcionales (RNF), ejecución (Eje) se evalúan por separado. La nota final de cada laboratorio (NL) se calcula de la siguiente forma considerando sólo la calificación base a partir del cumplimiento de los RF y RNF obligatorios.

**En caso de que su calificación para un laboratorio (NL) sea menor a 2.0 debido a que en alguno de los entregables (ej.: código, versionamiento, informe, etc.) su calificación fue inferior a 2.0, quedará a criterio del profesor encargado de la revisión, en base al nivel de logro en los otros entregables, si aplica o no el uso de comodín.**

**Acceso a los laboratorios específicos**

[Laboratorio 1 (Funcional - Scheme)](https://docs.google.com/document/d/1c3Foi2RuNZimvkPhHDyuI8NUgG3TIjqGEgl-ZQZQXLY/edit?usp=sharing)

[Laboratorio 2 (Lógico - Prolog)](https://docs.google.com/document/d/1GptXF-yEW6P6hlJA1mGJ6TakuaUX8pBsUZfdsy4iKB0/edit?usp=sharing)

[Laboratorio 3 (Orientado a Objetos - Java)](https://docs.google.com/document/d/1XhIp4cC8Gs1HW_2bNLMi8Nr3LL-HmVzut2lw0Pp8jDo/edit?usp=sharing)

1. *Solo esta parte de la narrativa del enunciado fue elaborado con la asistencia de Chat GPT 3.5 para alguna de sus partes.* [↑](#footnote-ref-0)
2. Ver tutorial de Git en Campus Virtual donde se explica como activar el Student Pack para poder crear repositorios privados de forma gratuita. [↑](#footnote-ref-1)