Universidad de Tarapacá

UNIVERSIDAD DE TARAPACÁ

Universidad del Estado

Facultad de Ingeniería

Departamento de Ingeniería en Computación e Informática

Proyecto 1: Robot Garra con LEGO Spike Prime

Integrantes:

Guillermo Contreras

Ariel Colque

Daniel Flores

Jhilmar Solares

Fernanda Tobar

CC091: Proyecto 1

Profesor: Baris Klobertanz

Arica, Chile

26 de septiembre, 2025

Tabla 1

Historial de versiones del documento

Fecha	Versión	Descripción	Autores
22/09/2025	1.0	Concepción inicial del documento	Jhilmar Solares
29/09/2025	1.1	Elaboración del primer prototipo	Jhilmar Solares
01/10/2025	1.2	Ampliación y mejora de la tabla de contenidos	Jhilmar Solares
03/10/2025	1.3	Desarrollo de la sección de restricciones y gestión de riesgos	Jhilmar Solares
13/10/2025	1.4	Mejora de la sección de organización proporcional	Jhilmar Solares
15/10/2025	1.5	Extensión y revisión de la bibliografía	Jhilmar Solares
15/10/2025	1.6	Completación de la sección de gestión de riesgos	Jhilmar Solares
17/10/2025	1.7	Finalización de la tabla estimación de costos	Jhilmar Solares
17/10/2025	1.8	Corrección del formato APA y validación final del documento	Jhilmar Solares
17/10/2025	1.9	Redacción y finalización de la conclusión del informe, incorporando el cierre y proyección del proyecto.	Jhilmar Solares
12/11/2025	1.10	Incorporación de los cambios de roles que se hicieron en el equipo.	Ariel Colque
14/11/2025	1.11	Actualización en el cronograma de actividades para la fase 2.	Ariel Colque
29/12/2025	1.12	Actualización general de la carta gantt.	Ariel Colque
30/12/2025	1.13	Se agregó el inciso “Prueba de funcionamiento del sistema”	Guillermo Contreras
30/12/2025	1.14	Se agregó y actualizó el apartado 7 “Resultados”, incorporando el estado actual del proyecto y los problemas encontrados con sus respectivas soluciones.	Jhilmar Solares
30/12/2025	1.15	Se mejoró el apartado de “Análisis-diseño”	Jhilmar Solares
30/12/2025	1.16	Actualización de la conclusión fase 3.	Guillermo Contreras

Tabla de contenidos

Tabla de tablas

Panorama general

1.1 Introducción

En este proyecto se presenta un acercamiento a un escenario real: la minería moderna exige soluciones vanguardistas, sustentables y además seguras. Innovaciones como la teleoperación, la automatización y la robótica han reducido el riesgo para los trabajadores. Accidentes en Grasberg (Indonesia) y El Teniente (Chile) evidencian la urgencia de proteger la vida humana. Este proyecto aborda la seguridad mediante la implementación de la robótica.

Se dispondrá un kit LEGO Spike Prime. El robot será operado de forma remota por un dispositivo cliente con interfaz gráfica de usuario (GUI), que permitirá enviar órdenes al hub del robot, el cual actuará como un servidor. El enlace con el hub se realizará vía Bluetooth, y para el desarrollo se usará el lenguaje Python.

Nuestro objetivo será crear un brazo robótico con garra denominado: Spike, capaz de tomar, levantar y trasladar objetos de diferentes formas y pesos. Para la comunicación con el microcontrolador del hub utilizaremos la librería Pybricksdev, que permite conectarse por USB o Bluetooth, enviar programas Python, y controlar motores, sin depender de la app de LEGO. Elegimos Thonny como nuestro entorno de desarrollo y Tkinter para crear la interfaz gráfica.

La planificación se apoyará en una carta Gantt que mostrará actividades y plazos a lo largo del semestre, además de reuniones y bitácoras semanales para registrar los avances del equipo.

Esperamos que el proyecto contribuya a la seguridad minera y cumpla con estándares del sector. Además, nos permitirá adquirir conocimientos sobre construcción y programación de (Spike), y desarrollar habilidades de trabajo colaborativo, administración del tiempo y documentación mediante bitácoras y seguimiento constante.

1.2 Objetivo general

Diseñar, construir y programar un robot manipulador móvil, basado en LEGO SPIKE Prime y una arquitectura cliente–servidor con GUI, capaz de sujetar, elevar y trasladar objetos, para apoyar actividades formativas vinculadas a procesos de la minería moderna.

1.3 Objetivos específicos

- Experimentar con los componentes del kit LEGO Education SPIKE Prime para comprender el funcionamiento de sus motores, sensores y piezas estructurales.
- Diseñar la estructura del robot optimizando su equilibrio, movilidad y resistencia al peso de los objetos transportados.
- Construir una garra mecánica capaz de abrir, cerrar y mantener un agarre firme sobre objetos de distintas formas y tamaños.
- Probar y ajustar los movimientos del robot mediante iteraciones que mejoren la precisión del agarre y la eficiencia del desplazamiento.
- Evaluar el desempeño del robot en diferentes condiciones (peso del objeto, superficie de trabajo, velocidad de movimiento).
- Documentar el proceso de diseño, ensamblaje y programación para dejar registro técnico del desarrollo del proyecto.
- Comprobar el cumplimiento de los objetivos mediante pruebas que verifiquen la capacidad del robot para levantar y trasladar correctamente los objetos asignados.

1.4 Restricciones

Corresponden a los límites, condiciones o factores que restringen la forma en que será llevada a cabo la solución para este proyecto.

- Se debe programar exclusivamente en Python.
- Uso exclusivo del set de Lego spike prime y el set de expansión para la construcción del robot.
- Entrega del proyecto en la fecha establecida.
- El robot no se puede trasladar fuera del establecimiento.
- La documentación y seguimiento del proyecto debía realizarse exclusivamente a través de la plataforma Redmine.
- La garra debe ser capaz de agarrar, levantar y mover un objeto.

1.5 Entregables

- Bitácoras: Son informes semanales que describen el avance del equipo en el proyecto, abarcando actividades realizadas, dificultades encontradas, recomendaciones para mejorar y acciones tomadas. Preparadas por un individuo designado, ofrecen un panorama exhaustivo para apoyar decisiones estratégicas, asignan responsabilidades y resaltan asuntos a tratar en grupo.
- Carta Gantt: Representación visual de la programación del proyecto, mostrando en una línea de tiempo las tareas, su duración y secuencia, facilitando la gestión del tiempo y los recursos al visualizar la evolución de las actividades a lo largo del proyecto.
- Informe de Formulación: Este documento detalla nuestra organización y estrategia para alcanzar los objetivos de la asignatura. Abordaremos la asignación de roles, las metas del equipo y las medidas que implementaremos para lograr el propósito académico. Además, compartiremos nuestras primeras impresiones durante el proceso de desarrollo y presentaremos la documentación relevante recopilada a lo largo del semestre.
- Presentaciones: Se detallan los objetivos del proyecto, los retos superados y las soluciones aplicadas. También se resaltan los éxitos obtenidos, la distribución del equipo y se ofrece una visión general del robot.

Organización del Personal

En esta sección se define el personal asignado al proyecto, sus roles, sus responsabilidades y su dedicación semanal. También se establecen las competencias mínimas por puesto, la línea de reporte y los hitos bajo su cargo. Precisa sustituciones ante ausencias y los criterios de disponibilidad. Sirve como base para planificación, control de la carga y evaluación del desempeño.

2.2 Personal Designado Fase 1

Tabla 2

Personal designado fase 1

Rol	Descripción del Rol	Responsable
Jefe de proyecto	Coordina el equipo, organiza tareas y supervisa el cumplimiento de los objetivos del proyecto.	Guillermo Contreras
Ensamblador	Construye el robot utilizando el kit LEGO Spike Prime, asegurando su estabilidad y funcionamiento.	Ariel Colque
Programador	Desarrolla y ajusta el código del robot, configurando sensores y motores.	Daniel Flores
Documentador	Registra los avances del proyecto y redacta los informes técnicos.	Jhilmar Solares
Diseñadora	Diseña el material visual y la presentación final del proyecto.	Fernanda Tobar

2.3 Personal Designado Fase 2

Tabla 3

Personal designado fase 2

Rol	Descripción del Rol	Responsable
Jefe de proyecto	Coordina el equipo, organiza tareas y supervisa el cumplimiento de los objetivos del proyecto.	Jhilmar Solares
Ensamblador	Construye el robot utilizando el kit LEGO Spike Prime, asegurando su estabilidad y funcionamiento.	Daniel Flores
Programador	Desarrolla y ajusta el código del robot, configurando sensores y motores.	Guillermo Contreras y Fernanda Tobar
Documentador	Registra los avances del proyecto y redacta los informes técnicos.	Ariel Colque

Planificación del proyecto

En este apartado se explica cómo organizaremos el trabajo y los tiempos del proyecto. Muestra las tareas, quién las hará, en qué orden, cuánto durarán y con qué recursos. Incluye riesgos, presupuesto y puntos de control para revisar avances y corregir desvíos.

3.1 Cronograma de actividades Fase 1

Tabla 4

Cronograma de actividades fase 1

Actividad	Responsable (Rol)	Inicio	Término	Duración
Prototipo inicial del robot (boceto y pruebas rápidas)	Jefe de proyecto	22/09/2025	26/09/2025	1 semana
Construcción del robot: estructura y base.	Ensamblador	29/09/2025	03/10/2025	1 semana
Redacción del informe (resultados y conclusiones)	Documentador	22/09/2025	31/10/2025	6 semanas
Bitácoras y seguimiento semanal del equipo	Jefe de proyecto	22/09/2025	31/10/2025	6 semanas
Programación v1: motores y sensores en Pybricks	Programador	06/10/2025	31/10/2025	4 semanas
Investigación y pruebas de librerías (Pybricks/pybricksdev)	Programador	05/10/2025	15/10/2025	2.5 semanas
Pruebas funcionales con prototipo 1 (agarre–elevación–traslado)	Programador	22/09/2025	31/10/2025	3 semanas

3.2 Cronograma de actividades Fase 2

Tabla 5

Cronograma de actividades fase 2

Actividad	Responsable (Rol)	Inicio	Término	Duración
Bitácoras y seguimiento de la Fase 2	Documentador	03/11/2025	19/12/2025	7 semanas
Actualización del Informe 2 (nuevos resultados y conclusiones)	Documentador	03/11/2025	19/11/2025	5 semanas
Modificación del robot: mejoras de código para desplazamiento homogéneo	Programador	12/11/2025	14/11/2025	3 días
Pruebas de nuevas funciones del robot	Programador	14/11/2025	17/11/2025	3 días
Prueba cruzada del robot con otro equipo	Equipo Completo	24/11/2025	26/11/2025	3 días
Creación del PPT para la presentación final	Equipo Completo	21/11/2025	05/12//2025	2 semanas
Preparación de la presentación (roles y ensayo)	Equipo Completo	08/12//2025	19/12//2025	2 semanas

3.3 Carta Gantt

En este apartado se presenta la carta Gantt del proyecto: un cronograma por fases, actividades y entregables con fechas de inicio y fin, con dependencias y responsables. Define la ruta crítica, holguras e hitos de control que sirven como la línea base para el correcto seguimiento de los avances, con la reasignación de recursos y gestión de riesgos.

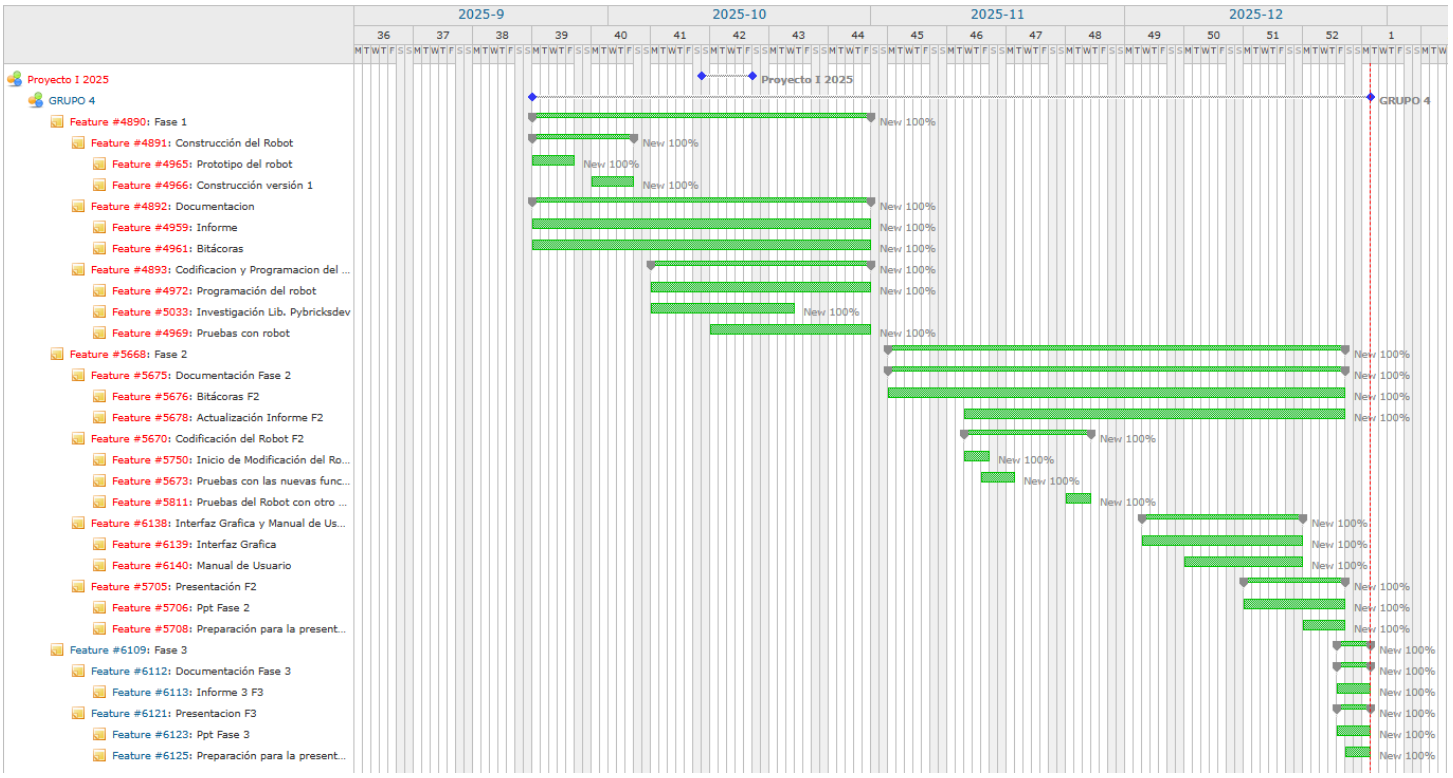

Figura 1. *Carta gantt del proyecto.*

Fase 1

Construcción del Robot 22/09 - 03/10

- Prototipo del robot (5 días)
- Construcción Versión 1 (5 días)

Documentación 22/09 - 31/10

- Informe (6 semanas)
- Bitácoras (6 semanas)

Codificación y Programación del Robot 06/10 - 31/10

- Programación del robot (4 semanas)
- Investigación de la Lib. Pybricksdev (4 semanas)
- Pruebas con robot (3 semanas)

Fase 2

Documentación F2 03/11 - 26/12

- Bitácoras F2 F2 (8 semanas)
- Actualización informe F2 (8 semanas)

Codificación del Robot F2 12/11 - 26/11

- Inicio de Modificación del Robot (3 días)
- Pruebas con las nuevas funciones del Robot (3 días)
- Pruebas con robot (3 días)
-

Interfaz Gráfica y Manual de Usuario 03/12 - 21/12

- Interfaz Grafica (3 semanas)
- Manual de Usuario (2 semanas)

Presentación F2 15/12 - 26/12

- Ppt Fase 2 (2 semanas)
- Preparación para la presentación (5 días)

Fase 3

Documentación F3 26/12 - 29/12

- Actualización informe F3 (4 días)

Presentación F3 26/12 - 29/12

- Ppt Fase 3 (4 días)
- Preparación para la presentación (3 días)

3.4 Gestión de Riesgos

A continuación se presenta una tabla que exhibe un desglose de los problemas que se han presentado a lo largo de la primera fase del proyecto. Esta tabla resume el impacto de cada desafío al clasificar el daño en cinco niveles distintos. Cada nivel está asociado con diferentes tipos de daño:

1. Daño catastrófico: Las medidas a tomar en el caso son de forma inmediata, puede provocar que el proyecto se detenga o retrase significativamente, teniendo que volver a empezar desde cero.

2. Daño crítico: Se deben tomar medidas necesarias para resolver el riesgo, debido a que puede provocar que el proyecto se retrase en varias etapas.

3. Daño circunstancial: El riesgo se debe resolver en el momento, debido a que puede retrasar el desarrollo de una etapa base del proyecto.

4. Daño irrelevante: El riesgo no es de mayor importancia, es un detalle imprevisto que no necesita mucha atención y se puede resolver en cualquier momento.

5. Daño recurrente: El riesgo no es significativo, pero es reiterativo, retrasa en las sesiones de trabajo, pero no en etapas.

Tabla 6

Gestión de riesgos del proyecto

Riesgo	Nivel de Impacto	Acción Remedial
Pieza rota	Daño circunstancial	Solicitar nuevamente la pieza faltante a los ayudantes, siguiendo el protocolo establecido
Falta de piezas	Daño circunstancial	Solicitar la pieza faltante a los ayudantes.
El robot sufre daños	Daño crítico	Asegurar el robot y realizar pruebas en entornos controlados.
Pérdida de piezas	Daño crítico	Buscar en el área donde se realizó la última manipulación de componentes.
Mala programación	Daño crítico	Revisar tutoriales e implementar soluciones en el código.
Ausencia del personal en el horario de trabajo	Daño crítico	Redistribución temporal de tareas entre el personal disponible para evitar retrasos
Dificultades con la conexión wifi	Daño Recurrente	Verificar la conectividad probando con un dispositivo alternativo

Estimación de Costos

En esta sección se presenta el cálculo estimado de los recursos necesarios para el desarrollo del proyecto, considerando tanto los materiales utilizados como las herramientas digitales y la participación del equipo de trabajo. Los costos se dividen en tres categorías principales:

- Hardware, que incluye los componentes físicos indispensables para la construcción y funcionamiento del robot.
- Software, donde se consideran las licencias o suscripciones empleadas para la programación y documentación.
- Costo por rol, que estima el valor del tiempo dedicado por cada integrante del equipo según sus funciones y horas de trabajo asignadas.
- El desarrollo del proyecto se llevó a cabo entre el 22 de septiembre de 2025 y el 30 de diciembre de 2025, período durante el cual se realizaron las etapas de diseño, ensamblaje, programación y documentación.

4.1 Hardware

Planificación:

- Kit LEGO Spike Prime: Componentes principales para construir la estructura, motores y sensores del robot. Es el núcleo físico del proyecto.

- Notebook Gaming: Herramienta esencial para programar el robot, realizar simulaciones y gestionar toda la documentación del proyecto.

- Mouse Ergonómico: Periférico para facilitar el trabajo prolongado en el diseño y la programación, mejorando la comodidad y eficiencia.

Tabla 7

Costo de hardware

Producto	Cantidad	Precio(CLP)
Set Lego Spike Prime	1	622.872
Notebook Gaming Victus 16-r1015la	1	1.299.990
Mouse Vertical Ergonómico Inalámbrico Negro	1	12.990
Huawei Intel Core i5	1	479.990
Lenovo N-9 15” AMD Ryzen 7 - 8GB RAM	1	649.990
Dell Inspiron 135301 15”	1	560.000
Acer Aspire 5 A515 15”	1	549.990
Total:	9	4.175.822

4.2 Software

Planificación:

- Microsoft 365 Personal: Suite de productividad utilizada para la documentación integral del proyecto (informes en Word, presentaciones en PowerPoint), la organización de datos en Excel y la comunicación y coordinación del equipo.

Tabla 8

Costo de software

Producto	Cantidad	Precio(CLP)
Microsoft 365 Personal (anual)	1	84.990
Total:	1	84.990

4.3 Costo por rol

Los costos por rol se estimaron a partir de las horas planificadas en el cronograma. Para cada actividad se asignó el rol responsable y se convirtieron las duraciones a horas. Las horas acumuladas por rol se multiplicaron por su tarifa horaria de referencia; la suma de estos valores entrega el costo total del proyecto.

Las tarifas horarias de referencia se obtuvieron a partir del portal de salarios de Indeed (Indeed, s. f.).

Tabla 9

Costo por rol

Rol	Horas	Precio por hora (CLP)
Jefe de proyecto	27	7.309
Ensamblador	19	3.711
Programador	27	7.860
Documentador	25	4.560
Analista tester	22	9.674
Total:	120	806.900

Análisis–Diseño

5.1. Especificación de Requerimientos

5.1.1. Requerimientos Funcionales

Identificación de Usuario y Cliente:

Usuario: Operador minero. Es la persona que, desde una estación de control, manipula la garra robótica para realizar tareas de recolección o manipulación de materiales en un entorno de simulación minera o en un prototipo a escala.

Cliente: Supervisión de la mina. Es el departamento o equipo de ingeniería que define las necesidades operativas y valida que el prototipo cumpla con los criterios de funcionalidad, seguridad y eficiencia establecidos para su aplicación.

Calidad de los Requerimientos Funcionales:

Completos: La lista abarca todos los aspectos críticos del sistema: entrada (comandos), proceso (movimiento, detección, regulación) y salida (agarre/liberación).

Claros: Cada requerimiento está redactado en un formato conciso que describe una capacidad observable y medible, sin ambigüedad.

Accionables y Concretos: Cada uno define una acción o comportamiento específico que los desarrolladores pueden implementar y probar.

Los requerimientos se alinean directamente con el objetivo de crear un efector final robótico autónomo y controlable para un entorno educativo/competitivo. Responden a necesidades como precisión, seguridad, integración con control externo (RF-04) y retroalimentación (RF-05), que son esenciales para un prototipo funcional y didáctico.

Lista de Requerimientos Funcionales (RF)

RF-01: Accionamiento Motorizado

La garra debe abrir y cerrar sus pinzas mediante un motor o servomotor controlable.

RF-02: Rango de Prensión

La garra debe poder agarrar objetos cuya dimensión se encuentre dentro del rango de apertura máximo y mínimo físicamente definido de sus pinzas.

RF-03: Liberación por Comando

La garra debe liberar el objeto sujeto al recibir un comando específico de apertura desde el usuario, en este caso utilizando el teclado del computador.

RF-04: Interfaz de Control Externa

El hub (motor) debe recibir comandos de operación (abrir/cerrar) desde un controlador externo mediante un protocolo de comunicación establecido.

RF-05: Confirmación de Agarre

La garra debe incorporar un mecanismo (sensor o límite mecánico) para detectar y confirmar si un objeto ha sido sujetado con éxito.

RF-06: Protección por Límites

El motor debe detenerse automáticamente al alcanzar los límites físicos de apertura o cierre completo, previniendo daños por bloqueo o sobretorque.

RF-07: Regulación de Fuerza

La interfaz debe permitir la regulación de la fuerza de agarre aplicada por las pinzas para manipular objetos de diferente fragilidad.

RF-08: Tiempo de Respuesta

La secuencia completa de agarre (desde abierto hasta sujetado firme) debe completarse en un tiempo definido.

5.1.2. Requerimientos No Funcionales

RNF-01: Seguridad física

Descripción: La garra no debe presentar bordes filosos ni aristas cortantes que puedan lesionar al usuario ni al entorno.

Métrica / Criterio: No hay superficies con radio < 0.5 mm ni aristas afiladas; cantos lijados o redondeados >= 0.5 mm.

Verificación: Inspección visual y táctil; medición con calibre y comprobación mediante checklist de seguridad.

RNF-02: Movimiento predecible / sin sorpresas

Descripción: No debe haber movimientos inesperados (aperturas/cierres involuntarios) bajo condiciones de operación normales.

Métrica / Criterio: Probabilidad de movimiento no solicitado < 0.1% durante 10.000 ciclos de prueba; no detectar comandos no intencionados.

Verificación: Prueba de estrés (10.000 ciclos abrir/cerrar con entradas válidas y sin entradas — observar comandos espurios), logs de control y revisión de la lógica de entradas (debounce, filtro).

RNF-03: Resistencia / vida útil

Descripción: El sistema debe soportar uso repetido sin fallas.

Métrica / Criterio: Soportar al menos 50.000 operaciones de apertura/cierre sin fallo funcional degradante (lubricación y mantenimiento según especificación).

Verificación: Prueba de durabilidad acelerada en bancada, inspección post-prueba (holguras, fatiga del material, fallos del motor) y registro de fallos.

RNF-04: Estabilidad del mecanismo

Descripción: El mecanismo no debe presentar aperturas o cierres involuntarios por vibración o golpes leves.

Métrica / Criterio: Resistencia a vibración: sin activación con vibraciones de hasta 2 g RMS en frecuencias 10–200 Hz durante 60 s; retención de posición con carga prevista ±10% sin deslizamiento.

Verificación: Ensayo de vibración en bancada; test de retención de posición con carga (peso).

RNF-05: Peso / Ligereza
Descripción: La garra debe ser ligera para no sobrecargar motores.
Métrica / Criterio: Masa total < 150 g (incluye estructura y fijaciones, excluye motor si motor no es parte de la garra).
Verificación: Pesaje en balanza con resolución 0.1 g. Documentar componentes que suman masa.

RNF-06: Materiales y durabilidad
Descripción: Materiales duraderos como PLA, PETG o aluminio.
Métrica / Criterio: Material documentado en especificación; propiedades mínimas: resistencia a tracción y al desgaste compatibles con la carga prevista.
Verificación: Lista de materiales (BOM) y certificaciones/hojas de datos; pruebas de desgaste (abrasión) y ensayo mecánico si es crítico.

RNF-07: Consumo energético bajo
Descripción: El sistema debe consumir un nivel de energía adecuado al microcontrolador/plataforma usada.
Métrica / Criterio: Consumo promedio en operación normal < X mA (definir X según tu plataforma; ejemplo: < 500 mA pico, < 200 mA promedio).
Verificación: Medición con multímetro/registrador de consumo durante uso típico y en máximo esfuerzo; comparar con límites definidos.

RNF-08: Ruido aceptable
Descripción: El ruido de funcionamiento del motor debe ser mínimo para evitar molestias.
Métrica / Criterio: Nivel de presión sonora (SPL) < 60 dB(A) a 1 m durante operación normal (valor orientativo — ajustar según entorno).
Verificación: Medición con sonómetro calibrado o app con referencia; comparar a 1 m de distancia en condiciones de carga.

RNF-09: Latencia / respuesta
Descripción: La garra debe tener respuesta rápida, sin retrasos evidentes para el usuario.

Métrica / Criterio: Latencia entre comando (UI/mando/ps) y comienzo de movimiento < 300 ms; tiempo hasta posición objetivo depende del movimiento (documentar).
Verificación: Medición con cronómetro/registro de eventos: generar comando y medir tiempo hasta que el motor inicia movimiento (sensor o cámara de alta velocidad) — repetir N=30 y reportar media/percentil 95.

RNF-10: Usabilidad de la Interfaz
Descripción:
La interfaz gráfica debe ser intuitiva y fácil de utilizar para cualquier tipo de usuario, incluso sin experiencia técnica previa.
Métrica / Criterio:
Tiempo promedio para identificar la función principal < 10 segundos para un usuario nuevo; número de errores de navegación < 2 durante una prueba básica de uso.
Verificación:
Prueba de usabilidad con usuarios sin experiencia; observación directa y registro de tiempos de interacción; checklist de facilidad de navegación.

5.2 Arquitectura

La arquitectura tiene como base la de Cliente-Servidor, este se basa entre la interacción entre un cliente(PC) y un servidor(Hub de lego), mediante un ciclo de petición-respuesta.

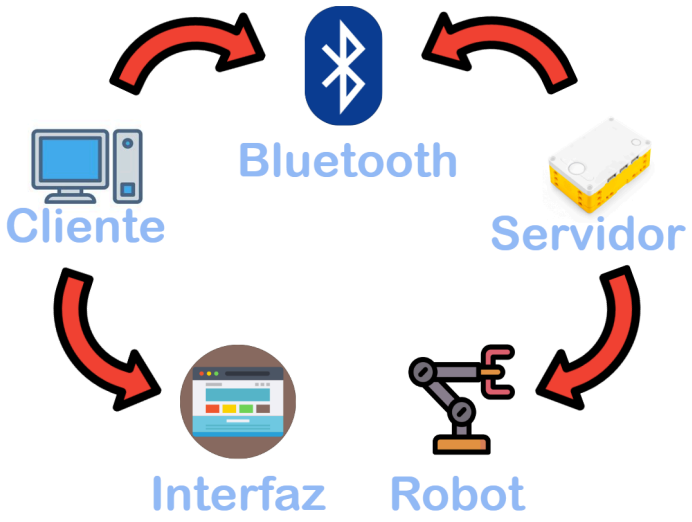

Figura 2. *Ciclo de petición-respuesta.*

1. Ambos dispositivos deben estar conectados a la misma red(cliente y servidor) para que se pueda realizar la comunicación.
2. Se encarga de la conexión remota entre cliente y robot , este se aloja en el Large Hub del spike army.
3. Por medio de una pc entramos a la interfaz, este se conectará al servidor del robot y se podrá controlarlo.
4. El robot garra(Spike) ejecuta acciones recibidas por el usuario y procede a realizarlas.
5. La interfaz gráfica de “Spike” es donde se encuentran las acciones que puede realizar el robot, está se encuentra programada usando la librería Tkinter.

5.3 Interfaz gráfica del sistema

La interfaz gráfica (GUI) se desarrolló en Python con Tkinter para controlar el movimiento del robot LEGO y la garra mediante Bluetooth, además de mostrar el estado de la conexión y un registro de acciones.

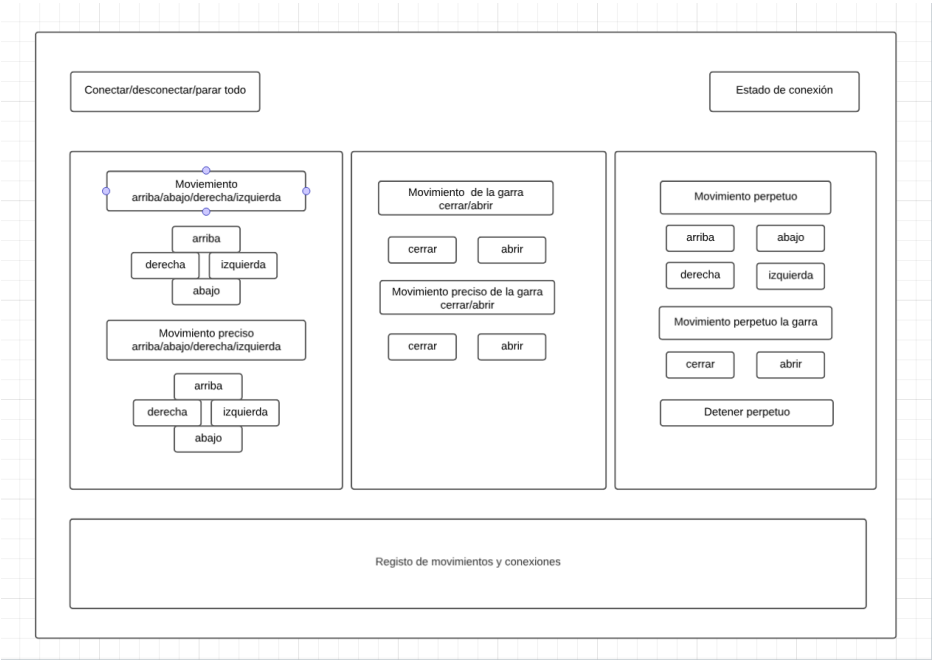

Figura 3. *Wireframe de la interfaz gráfica.*

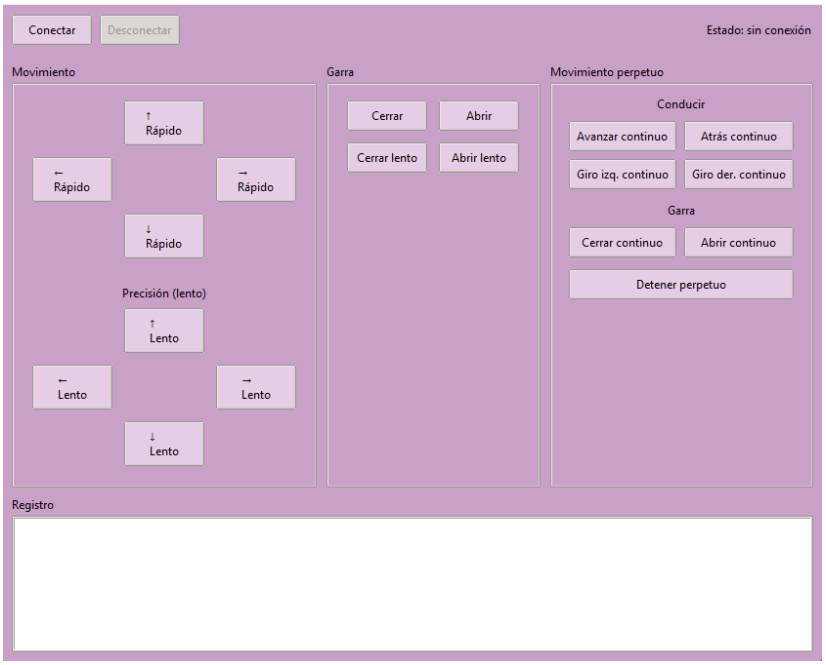

Figura 4. *Interfaz gráfica final para el control del robot LEGO.*

La **Figura 3** muestra el *wireframe* de baja fidelidad utilizado como boceto inicial, donde se planificó la disposición de los paneles de Movimiento, Garra, Movimiento perpetuo y Registro.

La **Figura 4** muestra la interfaz gráfica final implementada, con el mismo orden de paneles y un esquema de color lila para mejorar la visibilidad de los elementos

A continuación se describe brevemente la función de cada botón y zona de la GUI final.

Barra superior

- **Conectar:** inicia la búsqueda del hub LEGO y establece la conexión Bluetooth.
- **Desconectar:** corta la conexión con el hub LEGO.
- **Estado:** indica el estado actual del sistema, por ejemplo “sin conexión” o “conectado”.

Panel *Movimiento*

Permite desplazar el robot en dos modos: rápido y de precisión (lento).

- ↑ **Rápido:** el robot avanza a mayor velocidad.
- ↓ **Rápido:** el robot retrocede a mayor velocidad.
- ← **Rápido:** giro rápido hacia la izquierda.
- → **Rápido:** giro rápido hacia la derecha.
- ↑ **Lento (Precisión):** avance lento para aproximaciones finas.
- ↓ **Lento (Precisión):** retroceso lento.
- ← **Lento (Precisión):** giro lento a la izquierda.
- → **Lento (Precisión):** giro lento a la derecha.

Panel *Garra*

Controla el motor de la garra del robot.

- **Cerrar:** cierra la garra a velocidad normal.
- **Abrir:** abre la garra a velocidad normal.
- **Cerrar lento:** cierra la garra más lentamente, útil para objetos delicados.
- **Abrir lento:** abre la garra de forma lenta y controlada.

Panel *Movimiento perpetuo*

Agrupa los botones para mantener acciones continuas sin tener que mantenerlos presionados.

Conducir:

- **Avanzar continuó:** el robot avanza de forma continua.
- **Atrás continuó:** el robot retrocede de forma continua.
- **Giro izq. continuó:** giro continuo hacia la izquierda.
- **Giro der. continuó:** giro continuo hacia la derecha.

Garra:

- **Cerrar continuo:** mantiene la garra cerrando de forma continua.
- **Abrir continuo:** mantiene la garra abriendo de forma continua.
- **Detener perpetuo:** desactiva todos los movimientos continuos, tanto del robot como de la garra.

Área *Registro*

- **Registro:** área de texto donde se muestran los movimientos realizados, los cambios de conexión y posibles errores.
  Permite al usuario ver qué comandos se han enviado y comprobar el comportamiento del sistema.

Implementación

6.1 Fundamentos de los movimientos

En este proyecto se analizan dos movimientos fundamentales del sistema robótico: la fuerza necesaria para que la garra pueda levantar el objeto y el torque que debe entregar el motor para elevar dicho objeto considerando la geometría del brazo.

6.1.1. Fuerza necesaria para levantar el objeto con la garra

El primer movimiento corresponde a la acción de levantar un objeto mediante la garra. Para que el objeto pueda ser elevado, la garra debe ejercer al menos una fuerza igual al peso del objeto.

La fuerza peso se calcula mediante la expresión:

$$F = m \cdot g$$

donde:

m es la masa del objeto,

g es la aceleración de gravedad.

Considerando un objeto de masa **160 g**, equivalente a **0,16 kg**, se obtiene:

$$F = 0{,}16 \cdot 9{,}8 = 1{,}57\,N$$

Al incorporar un 30 % de margen de seguridad, la fuerza requerida queda:

$$F_{req} = 1{,}57 \cdot 1{,}3 = 2{,}04\,N$$

Por lo tanto, la garra debe ser capaz de ejercer al menos **2,04 N** para asegurar un agarre firme del objeto. Este margen permite compensar efectos como el deslizamiento, irregularidades en la superficie del objeto y pérdidas por fricción. Debido a esta exigencia de fuerza adicional, se optó por utilizar un **motor grande** en el mecanismo de apertura y cierre de la garra, garantizando un agarre seguro durante toda la operación.

6.1.2 Torque requerido del motor para elevar el objeto

El segundo movimiento analizado corresponde a la elevación del objeto mediante el motor que acciona el brazo. En este caso, el motor debe generar un torque suficiente para vencer el momento producido por el peso del objeto respecto al eje de giro.
El torque requerido se calcula con la expresión:

$$\tau = F \cdot r$$

donde:
F es la fuerza peso del objeto,
r es la distancia entre la garra y el eje del motor.

Considerando una distancia de **22 cm**, equivalente a **0,22 m**, y la fuerza que ejerce el peso del objeto:

$$\tau = 1{,}57 \cdot 0{,}22 = 0{,}35 \text{ Nm}$$

Al aplicar el 30 % de margen de seguridad, el torque requerido es:

$$\tau_{req} = 0{,}35 \cdot 1{,}3 = 0{,}46 \text{ Nm}$$

Esto indica que el motor encargado de levantar el brazo debe ser capaz de entregar al menos **0,46 Nm** de torque. Debido a esta exigencia, y considerando además el peso propio del brazo y su estructura, se decidió utilizar un motor grande para el levantamiento del objeto, asegurando un movimiento estable, controlado y sin sobrecargar el motor.

La incorporación de este margen de seguridad fue un criterio clave en la selección de los motores, permitiendo que el robot cumpla de manera confiable con la tarea de agarrar y levantar objetos dentro de las condiciones planteadas para el proyecto.

6.2 Descripción del sistema

El sistema implementado corresponde a un brazo robótico con garra, construido utilizando el kit LEGO Spike Prime, el cual es operado de forma remota mediante una interfaz gráfica de usuario (GUI).

Arquitectura general del sistema

El sistema se compone de dos partes principales:

- **Sistema físico (hardware):**
  - Hub LEGO Spike Prime
  - Motores encargados de:
    - Apertura y cierre de la garra
    - Elevación y descenso del brazo
    - Giro lateral de la garra
  - Estructura mecánica del brazo y la base

- **Sistema lógico (software):**
  - Programa en Python ejecutado en el hub del robot
  - Interfaz gráfica desarrollada con Tkinter
  - Comunicación entre la GUI y el robot mediante USB o Bluetooth, utilizando la librería Pybricksdev

Funcionamiento general

El operador interactúa con la interfaz gráfica desde un computador, enviando comandos que permiten controlar los distintos movimientos del robot. Estos comandos son transmitidos al hub, el cual ejecuta las instrucciones correspondientes en los motores.

El sistema permite:

- Tomar objetos mediante la garra
- Elevar y descender el objeto
- Girar la garra para posicionar o trasladar el material

La implementación del sistema asegura una operación intuitiva, modular y escalable, facilitando futuras mejoras tanto a nivel mecánico como de programación.

6.1.3 Cliente

El cliente corresponde a la aplicación ejecutada en el computador del operador, la cual proporciona una interfaz gráfica de usuario (GUI) que permite interactuar con el robot de manera intuitiva.

Desde la interfaz, el usuario puede:

- Enviar comandos de movimiento al robot
- Controlar la apertura y cierre de la garra
- Elevar, descender y girar el brazo robótico

La GUI fue desarrollada utilizando la librería Tkinter, lo que permitió crear botones y controles simples para la operación del sistema. El cliente se encarga de generar las órdenes de control y transmitirlas al servidor mediante una conexión USB o Bluetooth.

6.1.4 Servidor

El servidor corresponde al hub LEGO Spike Prime, el cual recibe las instrucciones enviadas por el cliente y ejecuta las acciones físicas del robot.

El hub ejecuta un programa desarrollado en Python, encargado de:

- Interpretar los comandos recibidos
- Controlar los motores según la acción solicitada
- Coordinar los movimientos del sistema de forma segura y ordenada

La comunicación entre el cliente y el servidor se realiza mediante la librería Pybricksdev, permitiendo una conexión directa sin depender de la aplicación oficial de LEGO. Esta arquitectura cliente–servidor facilita la separación entre la interfaz de control y la lógica de ejecución del robot, mejorando la organización y escalabilidad del sistema.

6.1.5 Interfaz gráfica

La interfaz gráfica de usuario (GUI) fue desarrollada con el objetivo de permitir una operación clara, simple y segura del robot por parte del operador. La GUI fue implementada utilizando la librería Tkinter en Python, lo que permitió crear una ventana de control con botones claramente identificados para cada acción del robot. Cada botón de la interfaz está asociado a una acción específica, la cual se envía al hub LEGO Spike Prime mediante la arquitectura cliente–servidor definida previamente. El diseño de la interfaz prioriza la claridad visual y la facilidad de uso, reduciendo la posibilidad de errores durante la operación.

Resultados

7.1 Estado actual del proyecto

Al término del desarrollo del proyecto, el sistema se encuentra implementado y operativo, cumpliendo con los objetivos funcionales planteados inicialmente. El robot manipulador basado en LEGO SPIKE Prime cuenta con su estructura final ensamblada, incluyendo la garra junto con los mecanismo de elevación y rotación necesarios para la manipulación de objetos.

Desde el punto de vista del software, se desarrolló una interfaz gráfica de usuario (GUI) que permite operar el robot de manera remota. Esta interfaz posibilita la conexión con el hub del robot, el control del movimiento, la operación de la garra y la ejecución de acciones continuas o precisas según lo requerido. La comunicación entre la interfaz y el robot se realiza correctamente, permitiendo el envío de comandos y la ejecución de las acciones definidas.

El sistema ha sido probado mediante una prueba de funcionamiento, en la cual se verificó el correcto desempeño del robot durante todo el ciclo operativo. Los resultados obtenidos demuestran que el proyecto se encuentra en un estado funcional estable, apto para su demostración y evaluación, validando la integración entre el hardware, el software de control y la interfaz gráfica.

7.2 Problemas encontrados y soluciones aplicadas

Durante el desarrollo del proyecto se presentaron diversos problemas, tanto a nivel de hardware como de software, los cuales fueron abordados y solucionados de manera progresiva:

- **Problemas de comunicación entre la GUI y el robot**:

  Inicialmente se presentaron dificultades para establecer una conexión estable entre la interfaz gráfica y el hub del robot. Este problema se soluciona revisando la configuración del entorno de desarrollo, ajustando el método de conexión y asegurando que el hub estuviera correctamente emparejado antes de ejecutar la aplicación.

- **Retrasos en la respuesta del robot**:

  En las primeras pruebas, algunas acciones presentaban latencia o respuestas imprecisas. Para solucionar esto, se optimizó la lógica de envío de comandos desde la GUI y se implementaron modos de control diferenciados (movimiento rápido y movimiento de precisión), mejorando la estabilidad del sistema.

- **Limitaciones mecánicas de la estructura**:

  Se identificaron restricciones en el movimiento de la garra, principalmente en la distancia entre las pinzas y la base del robot. Este aspecto fue considerado en el diseño funcional, adaptando las pruebas y los objetivos operativos a las capacidades reales del sistema mecánico, sin afectar el cumplimiento del proyecto.

- **Errores durante la ejecución de acciones continuas**:

  El modo de movimiento perpetuo generaba comportamientos no deseados en algunas pruebas iniciales. Esto se resolvió incorporando controles de detención desde la interfaz, permitiendo un control más preciso por parte del usuario.

La identificación y resolución de estos problemas permitió mejorar progresivamente el sistema, logrando un funcionamiento más robusto y confiable. Estos ajustes fueron fundamentales para alcanzar el estado final del proyecto y asegurar el correcto cumplimiento de los objetivos planteados.

Prueba de funcionamiento del sistema

En esta sección se describirán de forma completa todo lo que se realizó para llevar a cabo el funcionamiento del sistema. También mostraremos cuáles fueron los resultados obtenidos durante la prueba.

8.1 Descripción de la prueba de funcionamiento

La prueba de funcionamiento del sistema tuvo como objetivo verificar el desempeño del robot garra bajo condiciones de operación en tiempo real, analizando tanto el control mecánico como la comunicación cliente–servidor establecida mediante la interfaz gráfica desarrollada en Python con Tkinter.

8.2 Resultados observados para la prueba de funcionamiento

Durante las pruebas, el robot respondió de manera satisfactoria a todos los comandos enviados desde la interfaz gráfica. La conexión Bluetooth entre el cliente y el hub LEGO se mantuvo correctamente durante el procedimiento, permitiendo que el control remoto del robot continuará sin interrupciones.

El robot logró moverse de manera estable en los modos rápido y lento (Los únicos modos de movimiento hasta el momento.) cumpliendo con los movimientos y manteniendo un manejo efectivo durante los giros y desplazamientos. La garra presentó un funcionamiento correcto, siendo capaz de abrir y cerrar según los comandos ejecutados, así como de sujetar objetos de una manera firme sin soltarlos durante el levantamiento.

Conclusión

A lo largo del desarrollo del proyecto, nuestra experiencia con el robot *LEGO Education SPIKE Prime* ha sido sumamente didáctica. Aunque la etapa de construcción resultó ser la más relajada y sencilla, el verdadero desafío comenzó al momento de programar las acciones de la garra. Desde un inicio, nuestra idea principal era simple pero prometedora: sincronizar un mando de consola que nos permitiera ejecutar las funciones básicas del robot, como abrir y cerrar la garra o desplazarse sobre su propio eje.

Los problemas aparecieron cuando comenzamos a utilizar la plataforma oficial de LEGO Spike, la cual no soportaba funciones externas ni librerías ajenas a su sistema. Esto nos llevó a probar *LEGO Mindstorms*, pero su aplicación había sido descontinuada y no era compatible con nuestro hub. Luego intentamos trabajar con *Pybricks*, donde surgieron dificultades de conexión y líneas de código que no realizaban las acciones esperadas. Después de varios intentos, decidimos avanzar parte del código en *Visual Studio Code*, y fue allí donde logramos un progreso significativo: el programa se sincroniza correctamente con el hub, pudimos controlar la garra mediante el teclado, y se abrió la posibilidad de incorporar un mando más adelante.

En las etapas final del proyecto, nos dimos cuenta de nuestra efectividad como equipo y lo que nos permitió hacer, tuvimos dificultades como la mayoría, pero fuimos capaces de sobrellevar la adversidad, actualmente nuestro robot es capaz de moverse y sostener objetos de manera efectiva, en el transcurso también se agregó la interfaz para cualquier tipo de usuario sin mucho conocimiento en el tecnicismo, el teclado para quien prefiere más profesionalismo y un mando para aquellos que prefieren un uso de manejo simple pero igual de efectivo.

En general, todo el proceso del proyecto nos ha permitido aprender mucho sobre diseño, ensamblaje y programación. Nuestra idea que al inicio sonaba bastante ambiciosa, se logró con mucho éxito y dedicación, por limitaciones de tiempo no se llevó a cabo instalar una cámara capaz de hacer reconocer a la garra el ángulo en el que se encuentra. pero servirá como punto de partida cuando el equipo esté dispuesto a trabajar una vez más en el robot que nos unió en más de una ocasión.

Referencias

8.1 Fuentes de compra

Canva. (s. f.). *Canva Pro – Herramientas de diseño profesional (suscripción anual).* Recuperado de https://n9.cl/s6nf3i

Microsoft. (s. f.). *Microsoft 365 Personal (suscripción anual).* Recuperado de https://n9.cl/5od5j

Paris Chile. (s. f.). *Mouse vertical ergonómico inalámbrico (negro).* Recuperado de https://n9.cl/3ylfte

Paris Chile. (s. f.). *Notebook Gaming Victus 16-r1015la, Intel Core i7, 16 GB RAM, 1 TB SSD, RTX 4060, 16.1" FHD 144 Hz, Windows 11 Home.* Recuperado de https://n9.cl/l3vl7

Ubuy Chile. (s. f.). *LEGO Education SPIKE Prime Set.* Recuperado de https://n9.cl/tac8z

8.2 Fuentes de información

LEGO. (s. f.). *Acerca de LEGO Mindstorms.* Recuperado de https://n9.cl/udokh

YouTube. (2022, marzo 15). *LEGO Mindstorms – Introducción y funciones básicas* [Video]. YouTube. Recuperado de https://n9.cl/w42fs

YouTube. (2021, noviembre 10). *Programación del robot LEGO Mindstorms paso a paso* [Video]. YouTube. Recuperado de https://n9.cl/x6ptp

Indeed. (s. f.). *Salarios: búsqueda por cargo y ubicación*. Recuperado el 28 de noviembre de 2025, de https://cl.indeed.com/career/salaries?from=gnav-homepage

Visure Solutions. Requerimientos Funcionales https://n9.cl/0ys3e

Visure Solutions. Requerimientos No Funcionales https://n9.cl/rb9zy1

Gannouji, R. (s. f.). *Física 2 (mecánica clásica)*. Instituto de Física, Pontificia Universidad Católica de Valparaíso. Recuperado de https://surl.li/bafofc

Lucid Software. (s. f.). *Lucidchart* [Herramienta de diagramación en línea]. Recuperado de https://surl.lu/sifqjz