Documentación CEstaciona-Prototipo de Parqueo Inteligente

Jimena Vargas Gómez

Los sistemas de de parqueo inteligente representan una solución tecnológica innovadora para optimizar el uso de espacios y mejorar la experiencia del usuario en centros comerciales. Este proyecto implementa un prototipo funcional de parqueo inteligente utilizando una Raspberry Pi Pico W como plataforma principal. El sistema desarrollado simula un parqueo con tres espacios disponibles, capaz de controlar el ingreso y salida de vehículos mediante una barrera automatizada, detectar la disponibilidad de espacios usando fotoresistencias, mostrar información en un display de 7 segmentos y administrar toda la operación a través de una aplicación remota desarrollada en Python.

Análisis del sistema

Sistema de control de espacios

El prototipo cuenta con tres espacios de estacionamiento, cada uno equipado con un LED indicador. Las fotoresistencias detectan la presencia de vehículos mediante cambios en la luminosidad, mientras que los LEDs señalan visualmente la disponibilidad del espacio. La lógica de detección se implementó estableciendo un umbral que diferencia entre un espacio ocupado y uno disponible.

Display de 7 segmentos

El display cumple una doble función: mostrar la cantidad de espacios disponibles durante la operación normal y desplegar el costo de estacionamiento cuando un vehículo está listo para salir.

Sistema de Ingreso y salida

El control de acceso se gestiona mediante dos botones físicos y un servomotor que actúa como barrera. El primer botón verifica la disponibilidad de espacios antes de permitir el ingreso, mientras que el segundo gestiona el proceso de salida calculando el costo y abriendo la barrera.

Construcción del prototipo

La maqueta física se construyó utilizando materiales reciclados, principalmente cartón, organizados en diferentes niveles para simular un parqueo real.

Recomendaciones

Calibración de Fotoresistencias: Las condiciones de iluminación ambiental afectan las lecturas significativamente. Es importante calibrar el umbral de detección en el ambiente donde se realizará la demostración. Estabilidad de WiFi: Implementar un mecanismo de reconexión automática y colocar el router cerca del área de demostración para evitar

pérdidas de conexión. Debouncing de Botones: Los botones físicos pueden generar múltiples señales por una sola presión. Implementar un filtro digital ayuda a evitar registros duplicados.

Conclusiones

El proyecto CEstaciona logró implementar exitosamente un sistema funcional de parqueo inteligente que integra componentes electrónicos con software de control remoto. La comunicación HTTP entre los microcontroladores y la aplicación de administración resultó eficiente y confiable. El sistema cumplió con todas las especificaciones del proyecto, demostrando que es posible crear soluciones tecnológicas útiles con recursos accesibles. Este proyecto fortaleció competencias técnicas en programación de sistemas empotrados, diseño de circuitos electrónicos y desarrollo de interfaces gráficas.

Bibliografía, recursos consultados

Documentación Oficial:

Raspberry Pi Pico W - https://www.raspberrypi.com/documentation/microcontrollers/ MicroPython - https://docs.micropython.org/en/latest/ Python Tkinter - https://docs.python.org/3/library/tkinter.html

Tutoriales:

Control de Servomotores con Raspberry Pi Pico Display 7 Segmentos con MicroPython HTTP Server en Raspberry Pi Pico W

Foros:

Raspberry Pi Forums - https://forums.raspberrypi.com/ Stack Overflow - https://stackoverflow.com/

Atributos de trabajo individual y en equipo

Estrategias para el Trabajo individual

1.1 Etapa de Planificación Como única integrante del equipo, desarrollé estrategias individuales enfocadas en la autogestión y organización eficiente:

Estrategias Aplicadas:

Creé un cronograma detallado dividiendo el proyecto en tareas alcanzables Utilicé la técnica Pomodoro (25 minutos de trabajo, 5 minutos de descanso) para mantener productividad

Enfoque Inclusivo hacia Mí Misma:

Reconocí mis fortalezas (conocimientos en programación Python) y debilidades (experiencia limitada con hardware) Asigné tiempo extra para aprender sobre componentes electrónicos Permití flexibilidad en el cronograma cuando enfrenté dificultades, ajustando fechas de manera realista

1.2 Etapa de Ejecución Estrategias de Trabajo:

Aprendizaje progresivo: Comencé con tareas simples (encender LEDs) antes de abordar las complejas Prototipado iterativo: Probé cada componente individualmente antes de integrar el sistema completo Consulta de recursos: Busqué activamente tutoriales, foros y documentación cuando enfrenté obstáculos

Distribución del Tiempo:

Semana última: Desarrollo hadward y software, Integración, pruebas y documentación (20

1.3 Etapa de Evaluación Autoevaluación Continua:

Reconocí mis logros para mantener la motivación alta durante los días más estresantes de mi vida

Planificación del Trabajo: Roles, Objetivos, Metas y Reglas

Roles asumidos

Como única desarrolladora, asumí múltiples roles rotándolos según la fase del proyecto: Diseñadora de Hardware:

Investigación de componentes electrónicos Diseño y ensamblaje de circuitos en protoboard Conexión de sensores, actuadores y display

Desarrolladora de Software Empotrado:

Programación en MicroPython para Raspberry Pi Pico W Control de componentes (servomotor, LEDs, 7 segmentos)

Desarrolladora de Aplicación de Escritorio:

Creación de interfaz gráfica con Tkinter Sistema de estadísticas e integración con API Integradora y Tester:

Pruebas de funcionalidad de cada componente Integración del sistema completo Construcción de la maqueta física

Objetivos y metas

Objetivo General: Desarrollar un sistema funcional de parqueo inteligente que cumpla con todas las especificaciones del proyecto. La meta de esta última semana era terminar con lo mejor posible de proyecto

Reglas de trabajo autoimpuestas

Gestión del Repositorio:

Crear branches para cada funcionalidad nueva antes de integrar a main Documentar el código inmediatamente después de escribirlo Realizar pruebas antes de cada commit importante

Disciplina Personal:

No posponer tareas difíciles; enfrentarlas cuando tengo más energía Pedir ayuda al profesor cuando esté bloqueada más de 4 horas en un problema Tomar descansos obligatorios para evitar burnout Mantener el espacio de trabajo organizado

Calidad del Trabajo:

Todo el código debe tener comentarios explicativos Probar cada funcionalidad inmediatamente después de implementarla No avanzar a la siguiente tarea si la actual no funciona correctamente Mantener la documentación actualizada semanalmente

Acciones que Promueven la Colaboración

Individual

Aunque trabajé sola, busqué activamente colaboración externa y apoyo:

Recursos de aprendizaje

Comunidades Online:

Participé en foros de Raspberry Pi Pico para resolver dudas específicas Consulté Stack Overflow para problemas de programación Revisé repositorios de GitHub con proyectos similares para inspirarme

Automotivación y Persistencia

Celebración de Logros:

Al completar cada meta, me permitía un descanso Mantuve una lista mental de "logros técnicos" para recordar mi progreso en momentos difíciles

Manejo de Frustración:

Cuando enfrentaba bugs complejos, tomaba un descanso de 30 minutos antes de reintentar Cambié de tarea cuando estaba bloqueada en una para mantener el avance general del proyecto

Ejecución de Estrategias para el Logro de Objetivos

Metodología de Trabajo

Implementé una metodología ágil personal Estructura Semanal:

Lunes: Planificación de tareas y compras para hac er el hardware Martes: Desarrollo del prototipo

Evaluación del desempeño individual

Autoevaluación de Logros

Objetivos Cumplidos:

Sistema funcional Aplicación de administración remota operativa Panel de estadísticas con integración de API Maqueta física construida y funcional Documentación técnica completa

Fortaleza y áreas de mejora

Organización: Mantuve un cronograma estructurado y lo seguí consistentemente Documentación: Código bien comentado y documentación técnica detallada Resolución de problemas: Capacidad para investigar y resolver obstáculos técnicos Persistencia: No me rendí ante dificultades complejas

Áreas de Mejora:

Gestión del tiempo: Tuve muy poco tiempo Pedir ayuda: Tardé demasiado en consultar al

conocidos de la carrera cuando estuve bloqueada Testing continuo: Concentré las pruebas al final; debí probar más frecuentemente durante el desarrollo

Reflexión personal

Trabajar sola en este proyecto fue desafiante pero extremadamente enriquecedor. Asumí completa responsabilidad del resultado, lo que me obligó a desarrollar autonomía y capacidad de autogestión. Enfrenté la soledad en momentos de frustración, pero también experimenté la satisfacción de resolver problemas complejos por mí misma. Este proyecto me enseñó la importancia de la planificación detallada, la documentación continua y la persistencia ante obstáculos. También valoré la importancia de las comunidades técnicas online, que fueron fundamentales cuando necesité orientación. Calificación Personal: 9/10

Evaluación de Estrategias de Equidad e Inclusión Personal

Equidad en la Distribución del Esfuerzo

Balance entre Áreas:

Hardware: 30 Software Empotrado: 35 Aplicación de Escritorio: 25 Documentación y Pruebas: 10

Esta distribución me permitió desarrollar competencias integrales en todas las áreas del proyecto, evitando especializarme excesivamente en una sola área.

Inclusión hacia Mí Misma

Respeto a Mis Ritmos de Aprendizaje:

No me presioné excesivamente cuando el hardware fue inicialmente difícil Reconocí que aprender toma tiempo y me di el espacio para equivocarme Celebré pequeños logros para mantener motivación

Cuidado Personal:

Tomé pausas cuando sentía frustración o cansancio Busqué apoyo cuando lo necesité sin sentirme inadecuada