FICHA DE ACTIVIDAD PRÁCTICA								
CARRERA	COMPUTACIÓN E INFORMÁTICA MENCIÓN PROGRAMACIÓN							
NOMBRE DOCENTE	SEBASTIÁN SOLAR CUEVAS							
NOMBRE MÓDULO	DESARROLLO DE APLICACIONES MÓVILES							
APRENDIZAJE(S) ESPERADO(S)	1. Identificar las redes inalámbricas, los dispositivos móviles y sus diferentes plataformas y lenguajes de programación, empleados en dispositivos móviles. 2. Analizar el entorno de desarrollo Flutter y el lenguaje Dart. 3. Diseñar interfaces de usuario usando widget y layouts de Flutter. 4. Programar la lógica de negocio y el manejo de estado de una aplicación móvil. 5. Integrar servicios web y persistencia de datos en Flutter para el desarrollo de aplicaciones funcionales. 6. Publicar una aplicación móvil Flutter en PlayStore.							
N° ACTIVIDAD			TIEMPO EJECUCIÓN			FECHA DESARROLLO	01-12-2025	
LUGAR EJECUCIÓN	X	SALA		TALLER		LABORATORIO		TERRENO

OBJETIVO DE LA ACTIVIDAD
Desarrollar una aplicación móvil Flutter para la gestión integral de una flota vehicular que permita controlar vehículos, programar mantenimientos, gestionar recursos y sincronizar información con Firebase, usando persistencia local y siguiendo buenas prácticas de desarrollo.

DESCRIPCIÓN DE LA ACTIVIDAD

En la empresa “Zenith Transporte”, la gestión de su flota vehicular enfrenta varios desafíos críticos. Actualmente, el control de vehículos se realiza mediante registros manuales dispersos en hojas de cálculo o anotaciones físicas, lo que ha generado problemas como falta de actualización en tiempo real, pérdida de información y retrasos en la programación de mantenimientos fundamentales. Esto provoca que algunos vehículos sufran paradas imprevistas por fallas mecánicas evitables, perjudicando la operación diaria, aumentando costos por reparaciones de emergencia y generando retrasos en los servicios.

Además, la asignación y seguimiento de recursos como conductores y consumibles (combustible, repuestos) tampoco está centralizada, dificultando el control y la optimización del uso de cada vehículo. La empresa carece de alertas automáticas que avisen anticipadamente sobre mantenimientos próximos o el vencimiento de certificados y seguros asociados a los vehículos.

Frente a esta situación, Zenith Transporte busca una solución tecnológica móvil que permita:

- Registrar y mantener actualizados los datos detallados de cada vehículo y su estado operacional.
- Programar mantenimientos preventivos y registrar los correctivos realizados, asegurando la continuidad operativa.
- Gestionar recursos asociados a la flota, como conductores y consumibles, optimizando su asignación.
- Centralizar la información en tiempo real con capacidad de operar offline y sincronizar cuando haya conexión.
- Personalizar la experiencia del usuario mediante preferencias almacenadas localmente.
- Garantizar seguridad mediante autenticación de usuarios y roles definidos.

El estudiante deberá desarrollar una aplicación móvil utilizando Flutter que permita administrar la flota vehicular de Zenith Transporte cumpliendo con los siguientes requerimientos funcionales y técnicos:

1. Gestión de vehículos
   - Crear, editar, visualizar y eliminar vehículos con datos completos: matrícula, modelo, año, kilometraje, estado actual, y otros campos relevantes.
   - Visualizar listado de vehículos con información resumida y detalle individual de cada uno.

2. Mantenimientos
   - Registrar nuevos mantenimientos preventivos y correctivos indicando tipo, fecha programada, fecha de realización, observaciones y técnico responsable.
   - Mostrar alertas automáticas dentro de la app para mantenimientos próximos a vencer o retrasados.
   - Mantener historial completo de mantenimientos por vehículo con accesos fáciles desde el detalle del vehículo.

3. Gestión de recursos
   - Registrar y asignar conductores a vehículos, mantener datos de contacto y disponibilidad.
   - Registrar consumos de recursos como combustible o materiales asociados a cada vehículo de forma detallada.

4. Autenticación y seguridad
   - Implementar inicio de sesión seguro a través de Firebase Authentication con gestión de roles para diferenciar administradores y técnicos.

5. Persistencia y sincronización
   - Almacenar datos en Firebase Firestore para sincronización en tiempo real entre dispositivos.
   - Implementar almacenamiento local usando SharedPreferences para guardar preferencias de usuario como tema visual, alertas y filtros.
   - Diseñar función de modo offline para consulta y modificación de datos, sincronizándose automáticamente con Firebase al volverse a conectar.

6. Interfaz y navegación
   - Construir una interfaz clara, ordenada y funcional, utilizando widgets propios de Flutter (TextField, ElevatedButton, ListView, etc.).
   - Implementar navegación dinámica entre pantallas, pasando datos relevantes como argumentos.
   - Incorporar formularios con validación para ingreso correcto de datos.

7. Modelo y manejo de datos
   - Definir clases modelo en Dart que representen los datos de vehículos, mantenimientos y recursos.
   - Implementar parseo de datos JSON obtenidos de Firebase a objetos modelo y adaptar estos al evolucionar la estructura de datos.

8. Generación y pruebas
   - Realizar pruebas funcionales documentando los resultados.
   - Generar un APK funcional listo para su instalación y uso.

9. Documentación y reflexión
   - Documentar todo el proceso de desarrollo, incluyendo configuración del entorno, análisis comparativo de entornos, explicación del uso de widgets, gestión de estado, conexión a Firebase y persistencia local.
   - Describir el uso de inteligencia artificial para la generación y mejora de código, justificando su aplicación y reflexionando sobre su impacto en el proyecto.
   - Mantener orden y limpieza en recursos, código y entorno de desarrollo.

Entregables:

1. Reporte de especificación de requisitos de software
   - Documento formal que describa el problema, los objetivos, los requerimientos funcionales y no funcionales, usuarios y restricciones técnicas de la aplicación.
2. Cuadro comparativo de entornos de desarrollo móviles
   - Tabla que compare al menos tres entornos/marcos de desarrollo (incluyendo Flutter), detallando ventajas, desventajas y justificación de la elección para el proyecto.
3. Evidencia de configuración del entorno de desarrollo
   - Capturas, logs o documentación breve que muestre la correcta instalación y configuración de Flutter, Firebase CLI, emuladores o dispositivos usados.
4. Informe de experimentación con widgets básicos
   - Código fuente y breve explicación sobre diferencias entre widgets Stateless y Stateful.
   - Aplicación mínima funcional que implemente widgets básicos como Text, ElevatedButton y muestre su funcionamiento.
5. Análisis comparativo y justificación del uso de IA
   - Texto que contraste fragmentos de código base generados con IA frente a código propio.
   - Justificación documentada y reflexionada sobre el uso de IA para mejorar el desarrollo, con ejemplos específicos.
6. Código fuente de la aplicación completa

- Todo el código organizado y comentado que implementa las funcionalidades requeridas:
- Gestión de vehículos, mantenimientos y recursos.
- Autenticación con Firebase.
- Sincronización con Firestore y almacenamiento local SharedPreferences.
- Navegación entre pantallas con paso de argumentos.
- Formularios con validación.
- Gestión de estado (Provider, etc.).

7. APK funcional
   - Archivo APK generado listo para instalarse en dispositivos Android.
8. Documentación técnica del desarrollo
   - Explicaciones técnicas sobre la estructura general de la app, modelos de datos, manejo de estado, parseo JSON, validación, uso de Firebase y persistencia local.
   - Descripción clara del flujo de navegación y la interfaz de usuario.
9. Reporte de pruebas funcionales
   - Constancia con resultados de pruebas realizadas, incluyendo casos de uso validados, errores encontrados y soluciones aplicadas.
10. Documento de reflexión sobre el uso de IA y orden en el proyecto
    - Reflexión personal sobre cómo la integración de IA influyó en la eficiencia y calidad del código.
    - Descripción de las medidas tomadas para mantener un entorno de trabajo ordenado, tanto en recursos como en código.

Cada entregable deberá cumplir con un nivel de detalle y formalidad adecuados, reflejando conocimientos técnicos, claridad conceptual y buenas prácticas de desarrollo móvil según las habilidades y competencias esperadas

RECURSOS DIDÁCTICOS

- Flutter - Getting Started: Guía oficial para instalar Flutter en Windows, macOS o Linux.
- Flutter - Widgets Catalog: Catálogo oficial de widgets organizados por tipo y funcionalidad.
- Flutter - Stateless vs Stateful Widgets: Explicación clara de las diferencias entre widgets estáticos y dinámicos.
- Dart - Language Tour: Recorrido completo por la sintaxis y características del lenguaje Dart.
- Flutter Codelabs – First App: Tutorial interactivo paso a paso para crear tu primera aplicación en Flutter.
- GeeksForGeeks – Flutter vs React Native: Comparación entre dos plataformas de desarrollo multiplataforma.
- UOC – Alternativas en programación móvil: Artículo que presenta otras opciones para desarrollo móvil.
- GitHub Copilot – Getting Started: Guía para activar y utilizar la asistencia de código basada en IA.
- ChatGPT Prompts for Developers: Ejemplos de prompts útiles para generar código, aprender y depurar.

ESTRATEGIA DE EVALUACIÓN (BITÁCORA)

La actividad práctica se evaluará en forma individual, mediante formato bitácora correspondiente a la tercera de la asignatura.