UNIVERSIDAD TECMILENIO

Guía para el profesor

Desarrollo de Aplicaciones Móviles

Clave: LSTI2316

Índice

Información general del curso

Modalidades

- Clave banner: LSTI2316
- Modalidad: Semestral

Competencia del curso

Desarrolla aplicaciones móviles innovadoras y funcionales para entornos organizacionales, utilizando metodologías ágiles, principios de experiencia de usuario y frameworks de desarrollo multiplataforma, mediante el diseño, implementación y optimización de soluciones tecnológicas en equipos multidisciplinarios y en un entorno real o simulado.

Metodología

Características del curso

- El curso se imparte con la técnica didáctica de **Aula Invertida**.
- Tiene una competencia y tres evidencias (una para cada módulo).
- Está conformado por tres módulos distribuidos en 15 temas que integran su contenido.
- Se desarrollan actividades dentro del aula (individuales o en equipo) y actividades previas que tiene que realizar el alumno para acudir preparado a clase (con excepción de la primera sesión).
- Se aplican exámenes rápidos y exámenes parciales, así como una evaluación final.

Estructura del curso

Tema	**Actividad**	**Actividad previa**
1	-	-
2	1	-
3	-	-
4	2	-
5	-	-
6	Avance del reto	-
7	-	-
8	3	-
9	-	-
10	4	-
11	-	-
12	Certificación	-
13	-	-
14	Entrega final del reto	-
15	-	-

Modelo didáctico

El modelo educativo de la Universidad Tecmilenio, cuya visión es "Formar personas positivas con propósito de vida y las competencias para alcanzarlo", está enfocado en el desarrollo de competencias que distingan a sus alumnos y los capaciten para actuar ante diversos contextos, previstos o impredecibles, dado que vivimos en constante cambio, empoderándolos para ser autoaprendices y para aprender a aprender. Todo esto para su florecimiento humano, tomando en cuenta los elementos del Ecosistema de Bienestar y Felicidad de la Universidad.

Nuestra meta más importante en el aula es lograr un aprendizaje centrado en el alumno, por lo cual, el modelo que seguimos para el diseño e impartición de cursos es también **constructivista**, al presentar un cambio en los roles.

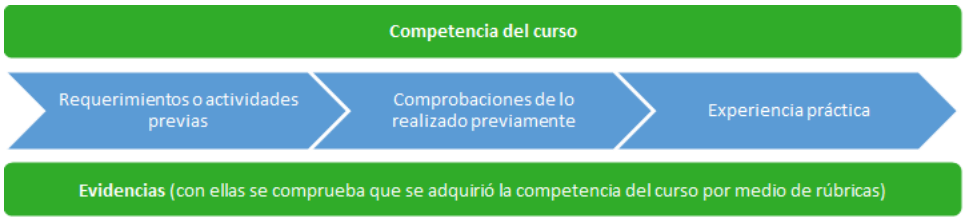
- **Los alumnos** obtienen las bases para hacer una interpretación de la realidad y construir su propio conocimiento, al aprender haciendo (no solamente viendo, escuchando y leyendo).
- **Los profesores**, al ser expertos en su disciplina y trabajar en la industria, aportan su experiencia laboral para guiar a los alumnos y construir ambientes de aprendizaje en contextos reales que los motiven a aprender, enriqueciendo así, su experiencia de aprendizaje.

Con esta visión constructivista, se ha incorporado la técnica didáctica de Aula Invertida para apoyar el aprendizaje activo. En seguida se explica la modalidad de este curso.

Modalidad: Aula Invertida con ciclo semanal

Los alumnos, comprometiéndose con su aprendizaje, realizan actividades previas o requerimientos **antes de la clase** para introducirlos a los conceptos que aplicarán en el aula. Para incentivar y evaluar lo realizado antes, los profesores deben desarrollar y aplicar comprobaciones de lo realizado. De esta manera, cuando los alumnos acudan al aula estarán más preparados para aclarar dudas, explorar, practicar, comprender la experiencia de sus profesores y ser guiados por ellos en la realización de actividades que buscan crear valiosas experiencias y oportunidades para el aprendizaje personal, al involucrar, estimular y retar a los alumnos en el descubrimiento de respuestas.

A continuación, se detallan las fases de esta modalidad.



¿Cómo impartir el curso?

El profesor debe revisar a fondo las actividades antes de que las realicen los alumnos y conocer todos los aspectos teóricos involucrados (capítulos de libros de texto o de apoyo y recursos), para brindar una respuesta o ayuda oportuna a los estudiantes dentro del modelo constructivista. Asimismo, debe indicar a los alumnos la información que requieren estudiar y buscar en internet para llevarla a las sesiones de clase, si se requiere.

A partir del tema 1, los alumnos se prepararán antes de la clase estudiando los temas incluyendo sus recursos, y a veces tendrán que realizar ejercicio como parte de la actividad previa o del apartado de requerimientos.

El profesor debe desarrollar y aplicar comprobaciones de lo que los alumnos debieron realizar antes y después de iniciar su clase, explicando la actividad y una visión de los conceptos más importantes en los que deben enfocar su atención. Considerando esta explicación, los alumnos inician su trabajo y el profesor monitorea su avance (no al frente del grupo, sino caminando entre mesas y a veces sentados con los alumnos para observar su trabajo), tratando de no interrumpir el aprendizaje, pero guiando la actividad para que los alumnos se enfoquen en lo que están haciendo.

Es muy importante que el profesor transmita a los alumnos sus experiencias relacionadas con los temas y aclare dudas.

Los exámenes parciales se desarrollarán por el profesor impartidor (considerando el contenido del curso), y pueden ser teóricos o prácticos.

Evaluación

Elemento	Evaluables	Puntos
1	Actividad 1	6
2	Actividad 2	6
3	Avance 1 del proyecto	25
4	Actividad 3	6
5	Actividad 4	6
6	Certificación	6
7	Entrega proyecto final	35
	Total	**100**

Bibliografía

Libro de texto

N/A

Libro de apoyo

Smyth, N. (2021). *Android Studio 4.2 Development Essentials - Kotlin Edition: Developing Android Applications Using Android Studio 4.2, Kotlin, and Android Jetpack*. Recuperado de https://research.ebsco.com/c/wzc6qz/search/details/yys6oyui5n?limiters=FT%3AY&q=android%20studio

Tips importantes

Material de capacitación en la plataforma tecnológica Canvas

- Tutorial digital para profesores:
- Tutorial digital para alumnos:

- **¿En dónde o a quién reporto un error detectado en el contenido del curso?**

Lo puedes reportar a la cuenta atencioncursos@servicios.tecmilenio.mx, pero te pedimos que también reportes sugerencias para el contenido y actividades del curso.

- **¿Quién me informa de la cantidad de sesiones y tiempo de cada sesión en las semanas?**

El coordinador docente te debe de proporcionar esta información.

- **¿En qué semanas se aplican los exámenes parciales y el examen final?**

Consulta con tu coordinador docente los calendarios de acuerdo con la modalidad de impartición.

- **¿Tengo que capturar las calificaciones en banner y en la plataforma educativa?**

Sí, es importante que captures calificaciones en la plataforma para que los alumnos estén informados de su avance y reciban retroalimentación de parte tuya de todo lo que realizan en el curso. En banner es el registro oficial de las calificaciones de los alumnos.

Temario

Tema 1	**Panorama general del desarrollo de aplicaciones móviles**
1.1	Definición de aplicaciones móviles y su relevancia en el entorno organizacional
1.2	Tipos de aplicaciones móviles: nativas, web y multiplataforma
1.3	Introducción a Android y su ecosistema
Tema 2	**Herramientas de desarrollo y configuración del entorno**
2.1	Instalación y configuración de Android Studio
2.2	Creación y ejecución de un primer proyecto en Android Studio
2.3	Familiarización con Kotlin y sus principales características
2.4	Primeras aplicaciones con "Hello World" en Kotlin
2.5	Tipos de datos, variables y estructuras de control
Tema 3	**Fundamentos de Kotlin**
3.1	Funciones en Kotlin: definición, parámetros y retorno
3.2	Manejo de excepciones y null en Kotlin

3.3	Clases, objetos e interfaces en Kotlin
3.4	Colecciones en Kotlin: listas, conjuntos y mapas
3.5	Funciones de orden superior y lambdas
Tema 4	**Fundamentos del Diseño Centrado en el Usuario (UX)**
4.1	Principios básicos del diseño UX: usabilidad, accesibilidad y emocionalidad
4.2	Técnicas de investigación de usuarios: encuestas, entrevistas y pruebas de usabilidad
4.3	Creación de personas y análisis de necesidades
4.4	Diseño iterativo y prototipos rápidos
Tema 5	**Herramientas y técnicas de diseño para aplicaciones móviles**
5.1	Introducción a las herramientas de diseño
5.2	Creación de wireframes y prototipos de baja fidelidad
5.3	Diseño de la estructura de navegación y flujos de usuario
5.4	Establecimiento de jerarquías visuales y accesibilidad en interfaces móviles
Tema 6	**Principios de Diseño de Interfaces de Usuario (UI)**
6.1	Elementos de la interfaz móvil: botones, formularios, menús
6.2	Uso de color, tipografía, iconos y espacio en blanco en interfaces
6.3	Diseño visual coherente y consistencia en la experiencia de usuario
6.4	Patrones de diseño y guías de estilo para Android
Tema 7	**Diseño avanzado de interfaces con Jetpack Compose**
7.1	Introducción a Jetpack Compose para la creación de UI en Android
7.2	Creación de composables, layouts y adaptación de la interfaz
7.3	Diseño de componentes interactivos y manejo de estados en Compose
7.4	Implementación de temas y personalización de la interfaz
Tema 8	**Manejo de eventos y entradas del usuario en Android**
8.1	Interacciones básicas: clicks, gestos, gestos multitouch
8.2	Formularios y validaciones de entrada
8.3	Implementación de notificaciones locales
8.4	Gestor de eventos y respuestas a interacciones
Tema 9	**Optimización de la interacción del usuario**
9.1	Creación de animaciones y transiciones fluidas en Android
9.2	Implementación de gestos y microinteracciones
9.3	Mejorando la experiencia táctil y la interacción en dispositivos móviles

9.4	Diseño para dispositivos con pantallas pequeñas y grandes
Tema 10	**Integración de bases de datos locales y persistencia de datos**
10.1	Uso de SQLite y Room para la persistencia de datos locales
10.2	Creación de bases de datos y almacenamiento de datos del usuario
10.3	Interacción con bases de datos a través de Kotlin
10.4	Diseño de pantallas que visualizan y gestionan datos de forma intuitiva
Tema 11	**Servicios en la nube y autenticación de usuarios**
11.1	Introducción a Firebase: autenticación y base de datos en tiempo real
11.2	Conexión y sincronización de datos con Firebase
11.3	Autenticación de usuarios con Firebase Authentication
11.4	Implementación de notificaciones push con Firebase Cloud Messaging
Tema 12	**Accesibilidad y diseño inclusivo en aplicaciones móviles**
12.1	Fundamentos de accesibilidad: compatibilidad con lectores de pantalla
12.2	Mejores prácticas para crear aplicaciones accesibles
12.3	Diseño para personas con discapacidades visuales, auditivas y motoras
12.4	Herramientas de prueba de accesibilidad en Android Studio
12.5	Optimización de interfaces para diferentes tamaños y resoluciones de pantalla
12.6	Pruebas y ajustes para garantizar la accesibilidad
12.7	Gestión de color y contraste para usuarios con deficiencias visuales
Tema 13	**Gestión del ciclo de vida de aplicaciones móviles**
13.1	Estrategias para la gestión de versiones y actualizaciones de aplicaciones
13.2	Uso de control de versiones con Git para diseñadores y desarrolladores
13.3	Técnicas de pruebas: unitarias y de integración en Android Studio
13.4	Manejo de excepciones y errores en aplicaciones móviles
Tema 14	**Despliegue y publicación de aplicaciones en Google Play**
14.1	Proceso de publicación en Google Play Store
14.2	Requisitos técnicos, políticas y optimización para la publicación
14.3	Creación de metadatos, capturas de pantalla y descripciones atractivas
14.4	Estrategias de mantenimiento post lanzamiento y actualizaciones

Notas de enseñanza

Tema 1 **Panorama general del desarrollo de aplicaciones móviles**

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Inicie la sesión contextualizando la importancia de las aplicaciones móviles en la transformación digital de las organizaciones.** Relacione cómo las aplicaciones han evolucionado de simples herramientas a plataformas estratégicas que impactan sectores como comercio, salud y educación, y destaque la necesidad de que los desarrolladores dominen tanto fundamentos técnicos como tendencias emergentes.
2. **Explique la definición de aplicaciones móviles y su relevancia organizacional.** Presente ejemplos de aplicaciones populares (Starbucks, Salesforce, aplicaciones bancarias) y cómo han transformado procesos internos y la relación con los clientes. Resalte características clave: portabilidad, usabilidad, conectividad, acceso a sensores y actualizaciones constantes.
3. **Analice los beneficios y retos de las aplicaciones móviles en el entorno empresarial.** Aborde cómo mejoran la productividad, la toma de decisiones en tiempo real, el *engagement* y la automatización de procesos. Utilice datos y casos de éxito para ilustrar el impacto real en organizaciones.
4. **Presente los tipos de aplicaciones móviles: nativas, web y multiplataforma.** Compare sus ventajas y desventajas en términos de rendimiento, costos, mantenimiento y experiencia de usuario. Utilice la tabla y el diagrama incluidos en el material para facilitar la comparación y analizar escenarios de elección tecnológica.
5. **Introduzca el ecosistema Android y su liderazgo en el mercado.** Destaque la cuota de mercado, las novedades de Android 14 (seguridad, privacidad, control de permisos) y la importancia de cumplir con los requisitos de Google Play para publicar aplicaciones. Explique el papel de herramientas como Android Studio, Firebase y ML Kit en el desarrollo moderno de aplicaciones.
6. **Demuestre la diferencia entre pruebas en emuladores y dispositivos físicos.** Explique cuándo conviene usar cada uno y sus limitaciones, reforzando la importancia de pruebas exhaustivas para asegurar la calidad y compatibilidad de la app1.
7. **Sugiera recursos de apoyo y actividades complementarias.** Recomiende videos, lecturas y podcasts para profundizar en tendencias, casos prácticos y experiencias reales del desarrollo móvil. Proponga preguntas de reflexión para fomentar el análisis crítico sobre la elección tecnológica y las tendencias emergentes (IA, IoT) en el desarrollo de aplicaciones.

Tema 2 Herramientas de desarrollo y configuración del entorno

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Introduzca la relevancia de dominar las herramientas de desarrollo móvil en el mercado actual.** Inicie la sesión destacando el crecimiento del mercado de aplicaciones móviles y la importancia de elegir y configurar correctamente el entorno de desarrollo, como Android Studio y Kotlin, para asegurar la calidad y competitividad de los proyectos.
2. **Explique y demuestre el proceso de instalación y configuración de Android Studio.** Guíe paso a paso la descarga, instalación y configuración inicial del IDE, enfatizando los requisitos técnicos recomendados (SSD, 16 GB de RAM) y mostrando cómo seleccionar el SDK adecuado y configurar un emulador desde el AVD Manager.
3. **Presente y explore las herramientas integradas de Android Studio.** Muestre el uso de Logcat para depuración, Android Profiler para monitoreo de rendimiento, y el emulador de dispositivos para pruebas. Explique cómo estas herramientas facilitan la identificación y solución de errores, y promueva el uso de la documentación oficial para resolver problemas comunes.
4. **Guíe la creación y ejecución del primer proyecto en Android Studio.** Demuestre la estructura básica de un proyecto Android (carpetas manifests, java, res, archivos build.gradle), la selección de plantillas (Empty Activity vs. Jetpack Compose), y la ejecución tanto en emulador como en dispositivo físico. Aborde errores frecuentes y sus soluciones, fomentando la autonomía en la resolución de problemas.
5. **Introduzca Kotlin y sus principales características.** Explique las ventajas de Kotlin frente a Java: sintaxis concisa, null safety, corrutinas para programación asíncrona e interoperabilidad con Java. Realice ejemplos prácticos de “Hello World” y muestre cómo personalizar la interfaz y agregar eventos básicos como onClickListener.
6. **Explique y practique el uso de variables, tipos de datos y estructuras de control en Kotlin.** Muestre la diferencia entre val y var, los tipos de datos básicos, y el uso de estructuras como when, for y while. Realice ejercicios prácticos que integren lógica condicional y repetitiva en el contexto de una aplicación móvil.
7. **Recomiende recursos de apoyo y actividades complementarias.** Sugiera videos, lecturas y podcasts para profundizar en la instalación, configuración y mejores prácticas con Android Studio y Kotlin. Proponga ejercicios de personalización de la aplicación “Hello World” y resolución de errores comunes para afianzar los conceptos aprendidos.

Tema 3 **Fundamentos de Kotlin**

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Introduzca la relevancia de Kotlin como lenguaje base para el desarrollo móvil moderno.** Explique cómo el 92% de las aplicaciones Android profesionales utilizan Kotlin por su sintaxis concisa, seguridad frente a errores comunes y compatibilidad con Java. Relacione cómo dominar sus fundamentos es clave para crear aplicaciones escalables y seguras en el entorno actual.
2. **Explique y demuestre la declaración y uso de funciones en Kotlin.** Muestre cómo definir funciones con parámetros, valores de retorno, parámetros por defecto, nombrados y variables (vararg). Realice ejemplos prácticos en clase, destacando la importancia de la reutilización y claridad del código. Enfatice la diferencia entre funciones de una sola expresión y funciones tradicionales.
3. **Guíe a los estudiantes en el manejo de excepciones y null safety.** Explique la estructura try/catch/finally y la creación de excepciones personalizadas. Demuestre cómo Kotlin previene errores comunes con null safety, usando tipos anulables, operadores seguros (?.), y estrategias como lateinit. Proponga ejercicios donde los estudiantes deban identificar y corregir posibles NullPointerException en fragmentos de código.
4. **Presente la estructura y uso de clases, objetos e interfaces.** Explique la sintaxis de clases, constructores primarios y secundarios, clases de datos, selladas y objetos singleton. Realice ejemplos aplicados a modelado de entidades en aplicaciones móviles (por ejemplo, Usuario, Producto). Destaque el uso de interfaces y la importancia del polimorfismo para desacoplar componentes y crear código flexible.
5. **Demuestre la gestión y manipulación de colecciones en Kotlin.** Muestre cómo crear y operar con listas, conjuntos y mapas, así como el uso de funciones avanzadas como map, filter y groupBy. Realice ejercicios prácticos de procesamiento de datos, por ejemplo, filtrando listas de usuarios o agrupando elementos por categoría.
6. **Explique y practique funciones de orden superior y lambdas.** Ilustre cómo estas herramientas permiten escribir código más modular y expresivo, facilitando operaciones como el procesamiento de datos y la gestión de eventos. Realice ejemplos aplicados a Android, como personalización de comportamientos en RecyclerView o manejo de callbacks en flujos de red.
7. **Sugiera recursos de apoyo y actividades complementarias.** Recomiende videos, lecturas y podcasts para profundizar en los fundamentos de Kotlin, su aplicación práctica en Android y tendencias actuales. Proponga ejercicios de integración, como diseñar jerarquías de clases, aplicar null safety en consultas de API y procesar colecciones con lambdas, para consolidar el aprendizaje.

Tema 4 **Fundamentos del Diseño Centrado en el Usuario (UX)**

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Inicie la sesión explicando la importancia del diseño centrado en el usuario en aplicaciones móviles.** Destaque cómo la competencia y las expectativas de los usuarios exigen experiencias memorables y funcionales, y presente ejemplos de aplicaciones exitosas que aplican estos principios. Relacione el impacto directo en la retención y satisfacción del usuario.
2. **Explique los principios básicos del diseño UX: usabilidad, accesibilidad y emocionalidad.** Desglose los componentes clave de la usabilidad según Nielsen (aprendizaje, eficiencia, memorabilidad, manejo de errores, satisfacción) y relacione con ejemplos prácticos en Android/iOS. Profundice en los principios de accesibilidad de la WCAG 3.0 (perceptible, operable, comprensible, robusto) y la importancia de la emocionalidad en la conexión usuario-aplicación.
3. **Demuestre cómo aplicar los principios de usabilidad y accesibilidad en interfaces móviles.** Realice ejemplos prácticos en clase, como diseñar botones accesibles en XML para Android, ajustar contrastes y tamaños de texto, y habilitar etiquetas contentDescription. Invite a los estudiantes a utilizar herramientas como Android Accessibility Scanner para validar sus diseños.
4. **Presente y compare técnicas de investigación de usuarios: encuestas, entrevistas y pruebas de usabilidad.** Explique cuándo y cómo aplicar cada método, mostrando ejemplos de preguntas efectivas y tareas para pruebas. Recomiende plataformas como Google Forms, Typeform y Maze, y enfatice la importancia de observar y analizar el comportamiento real de los usuarios antes de tomar decisiones de diseño.
5. **Guíe la creación de personas y el análisis de necesidades.** Explique cómo recolectar datos (entrevistas, encuestas, analytics), identificar patrones y construir arquetipos de usuario. Realice un ejercicio práctico de creación de una persona y priorización de necesidades funcionales, sociales y emocionales, utilizando herramientas como mapas de empatía y matrices de impacto.
6. **Explique el proceso de diseño iterativo y prototipos rápidos.** Describa el ciclo ideación → prototipo low-fi → pruebas con usuarios → análisis, y muestre cómo este enfoque reduce riesgos y acelera la mejora del producto. Proponga la regla 5-5-5 (máximo 5 pantallas, 5 días, 5 usuarios para pruebas) y el uso de herramientas como Figma y ProtoPie para prototipado rápido.
7. **Sugiera recursos de apoyo y actividades complementarias.** Recomiende videos, lecturas y podcasts para profundizar en principios de UX, creación de personas y diseño iterativo. Proponga preguntas de reflexión para fomentar el análisis crítico, como la importancia de involucrar a usuarios desde el inicio y los riesgos de omitir el prototipado rápido.

Tema 5 Herramientas y técnicas de diseño para aplicaciones móviles

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Comience la sesión contextualizando la importancia del diseño en la retención y éxito de aplicaciones móviles.** Explique que, según datos recientes, el 68% de las aplicaciones son desinstaladas en la primera semana por problemas de usabilidad, navegación confusa o interfaces inaccesibles. Relacione cómo el dominio de herramientas y técnicas de diseño es hoy una necesidad estratégica y no solo estética.
2. **Presente y compare las principales herramientas de diseño colaborativo utilizadas en la industria.** Demuestre las ventajas de Figma (colaboración en tiempo real, plugins), Adobe XD (integración con el ecosistema Adobe), Sketch (diseño vectorial en Mac), InVision y Proto.io (prototipado interactivo). Incluya una breve demostración práctica de Figma y destaque el papel de las herramientas potenciadas por IA como Uizard y Adobe Sensei para acelerar la iteración de prototipos.
3. **Explique y guíe la creación de wireframes y prototipos de baja fidelidad.** Describa la metodología en cuatro pasos: investigación y definición de requerimientos, bocetado manual o digital, prototipado interactivo y validación con usuarios. Realice un ejercicio práctico de sketching rápido y muestre ejemplos de wireframes exitosos, enfatizando la importancia de identificar problemas de usabilidad antes de invertir en desarrollo.
4. **Describa el diseño de la estructura de navegación y los flujos de usuario.** Analice patrones de navegación móvil como tab bars, menús hamburguesa, navegación por gestos y wizard. Explique cómo definir tareas críticas, segmentar usuarios y crear wireflows que incluyan estados de error y alternativas. Recomiende probar los flujos con usuarios reales y utilizar métricas como tasa de éxito y System Usability Scale para validar la experiencia.
5. **Demuestre la aplicación de principios de jerarquía visual en interfaces móviles.** Ilustre cómo el contraste, color, tamaño, tipografía, espaciado y profundidad guían la atención del usuario y facilitan la toma de decisiones. Realice un análisis de casos prácticos y proponga ejercicios de rediseño aplicando los principios Gestalt y Material Design para mejorar la claridad y el enfoque de las pantallas.
6. **Enfatice la importancia de la accesibilidad en el diseño móvil.** Explique los principios clave: contraste suficiente, tamaño mínimo de elementos, etiquetas descriptivas, navegación simplificada y pruebas con usuarios con discapacidad. Muestre cómo implementar contentDescription en Android y cómo validar el cumplimiento de estándares WCAG y UNE-EN 301 549:2019. Proponga la integración de opciones de personalización visual y el uso de lectores de pantalla en pruebas.
7. **Sugiera recursos de apoyo y actividades complementarias.** Recomiende videos, lecturas y podcasts sobre jerarquía visual, accesibilidad y diseño inclusivo. Proponga ejercicios de validación de prototipos con usuarios reales y el uso de herramientas automatizadas para pruebas de accesibilidad. Plantee preguntas de reflexión como ¿qué impacto tiene la ubicación del botón principal en la tasa de conversión?, o ¿cómo adaptarías tu diseño para usuarios con daltonismo?

Tema 6 **Principios de Diseño de Interfaces de Usuario (UI)**

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Inicie la sesión contextualizando la relevancia del diseño UI en el éxito de las aplicaciones móviles.** Explique que, en 2025, las aplicaciones se desinstalan la primera semana por problemas de usabilidad y navegación confusa. Destaque cómo la correcta implementación de principios UI es una necesidad estratégica para la retención y la inclusión, y no solo una cuestión estética.
2. **Explique y demuestre la correcta selección y diseño de los elementos fundamentales de la interfaz móvil: botones, formularios y menús.** Analice las mejores prácticas para cada componente: tamaño mínimo de botones (48x48dp), estados visuales, etiquetas claras en formularios, validaciones accesibles y menús ubicados en lugares esperados. Realice ejercicios prácticos de diseño y revisión de estos elementos en Android Studio y Figma.
3. **Analice el uso efectivo de color, tipografía, iconos y espacio en blanco en interfaces móviles.** Explique la psicología del color, la importancia de la legibilidad tipográfica (mínimo 17sp para texto, 22sp para títulos), la selección y consistencia de iconos (Material Icons, Fluent UI) y el uso del espacio negativo para mejorar la jerarquía visual y reducir la carga cognitiva. Proponga ejercicios de rediseño aplicando la regla 60-30-10 y el grid de 8dp.
4. **Guíe a los estudiantes en la creación de interfaces visualmente coherentes y consistentes.** Explique la diferencia entre consistencia visual, funcional y contextual. Utilice ejemplos reales (Spotify, Google Maps) para mostrar cómo la coherencia en colores, patrones de navegación y comportamientos reduce la carga cognitiva y aumenta la confianza del usuario. Sugiera el uso de sistemas de diseño y design tokens para mantener la uniformidad entre pantallas y equipos de trabajo.
5. **Presente los patrones de diseño y guías de estilo recomendados para Android, especialmente Material Design 3.** Explique la relevancia de patrones como MVVM, Singleton y Observer para la mantenibilidad y escalabilidad del código. Demuestre la aplicación de componentes clave de Material Design 3 (FAB adaptable, Navigation Bar, Cards) y la importancia de cumplir con estándares de accesibilidad, contraste y touch targets ampliados.
6. **Realice una actividad práctica de revisión y mejora de una interfaz móvil basada en los principios aprendidos.** Proponga que los estudiantes identifiquen errores comunes (inconsistencias, bajo contraste, iconos confusos) y apliquen soluciones utilizando guías de estilo y herramientas de validación como Color Contrast Checker y Android Accessibility Scanner.
7. **Sugiera recursos de apoyo y actividades complementarias.** Recomiende videos, lecturas y podcasts sobre herramientas colaborativas, patrones de arquitectura, tendencias UI/UX y accesibilidad. Proponga preguntas de reflexión, por ejemplo, ¿cómo adaptarías tu diseño para usuarios con daltonismo?, ¿qué impacto tiene la ubicación del botón principal en la tasa de conversión?, o ¿cómo priorizar entre simplicidad visual y funcionalidad compleja en un prototipo?

Tema 7 Diseño avanzado de interfaces con Jetpack Compose

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Inicie la sesión contextualizando la relevancia de Jetpack Compose en el desarrollo Android actual.** Explique que las aplicaciones en Play Store priorizan experiencias fluidas y personalizadas, y que Compose se ha convertido en el estándar gracias a su enfoque declarativo, reducción de tiempos de desarrollo y facilidad para crear interfaces adaptativas y accesibles.
2. **Explique la transición del enfoque XML al paradigma declarativo de Jetpack Compose.** Destaque cómo Compose permite describir la UI directamente en Kotlin, eliminando la duplicidad de archivos y facilitando la personalización y mantenimiento. Presente ejemplos de funciones @Composable y analice las ventajas frente a XML: menos código, mayor legibilidad y reactividad automática ante cambios de estado.
3. **Demuestre la creación de composables, layouts y la adaptación de la interfaz a diferentes dispositivos.** Guíe la construcción de componentes reutilizables y layouts con Column, Row y LazyColumn, mostrando cómo anidar composables para estructurar la interfaz. Explique el uso de WindowSizeClass para lograr diseños responsivos en móviles, tablets y pantallas plegables, y realice ejercicios prácticos de adaptación de UI.
4. **Guíe a los estudiantes en el diseño de componentes interactivos y el manejo de estados en Compose.** Explique conceptos clave como mutableStateOf, remember y state hoisting. Realice ejemplos con TextField, Switch y LazyColumn, mostrando cómo la UI se actualiza automáticamente ante cambios de estado. Introduzca arquitecturas avanzadas como MVI para la gestión de estados complejos y fomente buenas prácticas para evitar errores comunes.
5. **Explique la implementación de temas y la personalización avanzada de la interfaz.** Muestre cómo utilizar MaterialTheme para definir esquemas de color, tipografía y formas, y cómo crear temas dinámicos y modo oscuro. Demuestre el uso de herramientas como Material Theme Builder y Compose Preview para validar la coherencia visual y la accesibilidad, asegurando el cumplimiento de WCAG 3.0 y la identidad visual de la marca.
6. **Proponga actividades prácticas de revisión y optimización de la UI con Compose.** Invite a los estudiantes a analizar y mejorar la estructura de composables, la gestión de estados y la aplicación de temas en proyectos reales o ejemplos propuestos. Sugiera el uso de @Preview para validar la UI en diferentes dispositivos y fomente la integración de tokens personalizados para sistemas de diseño unificados.
7. **Sugiera recursos de apoyo y actividades complementarias.** Recomiende videos, lecturas y podcasts sobre Jetpack Compose, gestión de estados y tendencias UI/UX en Android. Plantee preguntas de reflexión como ¿qué ventajas ofrece Compose frente a XML en proyectos colaborativos?, ¿cómo asegurarías la accesibilidad en un tema oscuro?, o ¿cómo adaptarías una interfaz para pantallas plegables sin sacrificar coherencia visual?

Tema 8 Manejo de eventos y entradas del usuario en Android

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Inicie la sesión contextualizando la importancia del manejo eficiente de eventos y entradas en la experiencia de usuario móvil.** Destaque que, según Google (2025), las aplicaciones con validaciones robustas, gestos intuitivos y notificaciones contextuales logran una mayor retención de usuarios y mejor percepción de calidad. Relacione cómo la evolución de dispositivos (pantallas plegables, wearables) exige respuestas inmediatas y adaptativas en las interacciones.
2. **Explique y demuestre la implementación de interacciones básicas: clicks, gestos y multitouch.** Muestre cómo manejar eventos de clic tanto en XML como en Jetpack Compose, subrayando la diferencia en integración y mantenimiento. Realice ejemplos prácticos de gestos (deslizar, arrastrar, zoom) con Modifier.pointerInput y GestureDetector en Compose. Destaque la importancia de la retroalimentación háptica y la adaptación a dispositivos modernos.
3. **Guíe la construcción de formularios y la validación de entradas de usuario.** Explique la relevancia de formularios bien diseñados para la eficiencia, seguridad y satisfacción del usuario. Demuestre la validación en tiempo real con Jetpack Compose y el uso de expresiones regulares. Resalte la importancia de mensajes de error claros, validación accesible y autocompletado seguro, incluyendo tendencias como biometría y validación con IA (ML Kit).
4. **Explique la implementación de notificaciones locales y su impacto en el compromiso del usuario.** Presente los conceptos de canales de notificación, prioridades y categorías. Demuestre cómo crear y personalizar notificaciones en Android Studio, programar notificaciones exactas y cumplir con las políticas de privacidad de Android 12+. Analice tendencias como notificaciones adaptativas y la integración de IA para personalización.
5. **Presente arquitecturas y patrones modernos para la gestión de eventos y respuestas a interacciones.** Explique el uso de ViewModel, StateFlow y patrones como MVVM y MVI para centralizar la lógica de eventos y evitar fugas de memoria. Realice ejemplos prácticos de manejo de eventos en Compose, integración de corrutinas y pruebas unitarias para validar interacciones complejas.
6. **Proponga actividades prácticas de revisión y optimización del manejo de eventos.** Invite a los estudiantes a implementar formularios con validación en tiempo real, crear gestos personalizados y programar notificaciones locales. Sugiera el uso de herramientas como Android Accessibility Scanner para validar la accesibilidad de las interacciones y el cumplimiento de objetivos táctiles de ≥48dp en botones.
7. **Sugiera recursos de apoyo y actividades complementarias.** Recomiende videos, lecturas y podcasts sobre validación de formularios, seguridad en notificaciones y gestión avanzada de eventos en Android. Plantee preguntas de reflexión, por ejemplo, ¿cómo diseñar formularios accesibles y no invasivos?, ¿qué estrategias usar si una notificación crítica falla por restricciones de privacidad?, o ¿cómo migrar a MVI manteniendo estados predecibles en flujos complejos?

Tema 9 **Optimización de la interacción del usuario**

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Comience la sesión contextualizando la importancia de la optimización de la interacción en la retención de usuarios.** Explique que, en un mercado saturado de aplicaciones, la fluidez, adaptabilidad y precisión de las interacciones son factores críticos para evitar desinstalaciones tempranas y destacar frente a la competencia. Presente estadísticas recientes sobre la tasa de abandono y la preferencia de los usuarios por aplicaciones con animaciones y gestos bien implementados.
2. **Explique y demuestre la creación de animaciones y transiciones fluidas en Android.** Destaque que las animaciones no solo embellecen, sino que guían la atención, refuerzan acciones y mejoran la percepción de rendimiento. Muestre ejemplos prácticos con Jetpack Compose (animate*AsState, transiciones de layouts) y discuta los principios de duración, suavizado y consistencia según Material Design 3.0. Realice ejercicios de animaciones de retroalimentación y transiciones entre pantallas.
3. **Presente la implementación de gestos y microinteracciones como lenguaje de interacción moderno.** Explique la diferencia entre GestureDetector y los modificadores de gestos en Compose. Realice demostraciones de gestos estándar (deslizar, toque largo, pellizcar) y microinteracciones visuales/hápticas (ripple effect, vibración). Introduzca el uso de MediaPipe para reconocimiento avanzado de gestos y discuta buenas prácticas para evitar fatiga y errores de interpretación.
4. **Guíe a los estudiantes en la mejora de la experiencia táctil y la interacción en dispositivos móviles.** Explique la importancia de los tamaños mínimos de elementos interactivos (≥48dp), el espaciado adecuado y la retroalimentación inmediata. Realice ejercicios para implementar zonas seguras, debounce en eventos y prevención de toques accidentales. Analice cómo calibrar la sensibilidad táctil y adaptar la experiencia para dispositivos con protectores de pantalla o necesidades especiales.
5. **Explique el diseño adaptativo para dispositivos con pantallas pequeñas y grandes.** Analice los principios mobile-first, la priorización de contenido esencial y la jerarquía visual clara en móviles. Demuestre el uso de multi-pane layouts, Navigation Rail y contenido adaptable en tablets y plegables. Realice ejercicios prácticos con ConstraintLayout y WindowSizeClass para validar la adaptabilidad de la interfaz y la optimización de recursos gráficos (VectorDrawables, WebP, Lazy Loading).
6. **Proponga actividades prácticas de revisión y optimización de la interacción.** Invite a los estudiantes a validar la duración y suavizado de animaciones, implementar gestos estándar con retroalimentación, verificar el tamaño y espaciado de elementos táctiles y probar diseños responsivos en múltiples dispositivos. Sugiera el uso de herramientas como Android Accessibility Scanner y Color Contrast Checker para asegurar accesibilidad y calidad.
7. **Sugiera recursos de apoyo y fomente la reflexión crítica.** Recomiende videos, lecturas y podcasts sobre animaciones avanzadas, gestos intuitivos y diseño responsivo. Plantee preguntas de reflexión, por ejemplo, ¿cómo equilibrarías animaciones complejas en dispositivos de gama baja?, ¿qué criterios

seguirías para diseñar un gesto intuitivo para diferentes culturas?, y ¿qué elementos priorizarías al adaptar una interfaz móvil a una tablet para mantener la coherencia de marca?

Tema 10 Integración de bases de datos locales y persistencia de datos

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Inicie la sesión destacando la importancia de la persistencia de datos locales en aplicaciones móviles modernas.** Explique que los usuarios esperan experiencias fluidas incluso sin conexión a internet, y que la capacidad de almacenar y recuperar información localmente es clave en aplicaciones de productividad, salud y redes sociales. Relacione ejemplos de aplicaciones populares que dependen de la persistencia local para mantener la funcionalidad y la satisfacción del usuario.
2. **Explique y compare las soluciones de almacenamiento local: SQLite y Room.** Analice las ventajas de Room como capa de abstracción sobre SQLite, resaltando la reducción de errores, el soporte para corrutinas, la integración con Jetpack Compose y la facilidad para mantener el código. Utilice la tabla comparativa y ejemplos incluidos para mostrar cómo Room simplifica la gestión de entidades, DAOs y bases de datos.
3. **Demuestre la creación de bases de datos y almacenamiento de datos del usuario con Room.** Guíe paso a paso la definición de entidades (@Entity), DAOs (@Dao) y la clase de base de datos (@Database). Realice ejemplos prácticos de operaciones CRUD, migraciones de esquema y relaciones entre tablas (uno a muchos, muchos a muchos), enfatizando el uso de anotaciones y buenas prácticas como el patrón Singleton y la inyección de dependencias.
4. **Presente la interacción con bases de datos usando Kotlin y Jetpack.** Explique cómo utilizar funciones suspendidas y Flow en los DAOs para operaciones asíncronas y reactivas, evitando bloqueos en la UI. Realice ejemplos de transacciones atómicas, búsquedas avanzadas y paginación, mostrando la integración con ViewModel y la actualización automática de la interfaz con collectAsState o LiveData.
5. **Guíe el diseño de pantallas que visualizan y gestionan datos de forma intuitiva.** Analice principios clave como jerarquía visual, agrupación lógica y feedback contextual. Demuestre la implementación de listas dinámicas con LazyColumn, gráficos interactivos y pantallas adaptativas usando WindowSizeClass. Proponga ejercicios para aplicar Material Design 3 y asegurar la accesibilidad en la visualización de datos complejos.
6. **Enfatice tendencias y mejores prácticas en persistencia de datos.** Explique la importancia del cifrado con SQLCipher, el uso de migraciones seguras, la integración multiplataforma con Kotlin Multiplatform y la implementación de pruebas automáticas para garantizar la integridad y seguridad de los datos. Analice casos reales y discuta cómo elegir entre Room, SQLite y DataStore según el contexto de la aplicación.
7. **Sugiera recursos de apoyo y actividades complementarias.** Recomiende videos, lecturas y podcasts sobre implementación de Room, diferencias con SQLite, mejores prácticas y tendencias en persistencia

de datos. Plantee preguntas de reflexión, por ejemplo, ¿cómo priorizarías la seguridad y el rendimiento en una aplicación de salud?, o ¿qué retos enfrentarías al migrar una base de datos en producción?

Tema 11 Servicios en la nube y autenticación de usuarios

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Comience la sesión contextualizando la importancia de los servicios en la nube y la autenticación segura en el desarrollo móvil actual.** Explique cómo la demanda de aplicaciones colaborativas, la necesidad de sincronización en tiempo real y el cumplimiento de normativas de privacidad (como GDPR) hacen indispensable dominar herramientas como Firebase. Destaque que, según Google (2025), el uso de Firebase puede reducir el tiempo de lanzamiento al mercado en un 40% y mejorar la seguridad y escalabilidad de las apps.
2. **Explique los conceptos fundamentales de Firebase y su ecosistema.** Presente a Firebase como una plataforma integral de "backend como servicio" que ofrece autenticación, bases de datos en tiempo real y notificaciones push. Compare brevemente Firebase Realtime Database y Firestore, resaltando cuándo conviene usar cada uno según la necesidad de sincronización instantánea o consultas avanzadas.
3. **Demuestre la integración de Firebase Authentication en Android.** Explique los distintos métodos de autenticación soportados (correo/contraseña, Google, Facebook, teléfono, autenticación anónima y multifactor). Realice una demostración práctica de configuración e implementación de autenticación con email/contraseña y Google Sign-In. Enfatice la importancia de una gestión segura de credenciales y la personalización de la experiencia de inicio de sesión.
4. **Guíe la configuración y uso de Firebase Realtime Database para sincronización de datos en tiempo real.** Muestre cómo conectar la aplicación con Firebase, crear referencias a la base de datos y realizar operaciones CRUD (lectura y escritura) en tiempo real. Explique el modelo de publicación-suscripción y cómo la persistencia offline mejora la experiencia del usuario en condiciones de conectividad variable. Realice ejemplos prácticos de sincronización bidireccional y manejo de conflictos con transacciones atómicas y reglas de seguridad.
5. **Explique y practique la implementación de reglas de seguridad y mejores prácticas en Firebase.** Enseñe cómo definir reglas de acceso granular para proteger datos sensibles y restringir accesos a usuarios autenticados. Realice ejemplos de reglas de seguridad en la consola de Firebase y discuta la importancia de la validación de datos y la configuración de autenticación multifactor (MFA) para fortalecer la seguridad.
6. **Demuestre la implementación de notificaciones push con Firebase Cloud Messaging (FCM).** Explique el flujo de envío y recepción de notificaciones, la creación de canales de notificación y la personalización de mensajes. Realice un ejemplo práctico de integración en Android, destacando la protección del token FCM, la validación de mensajes y la importancia de la privacidad en la gestión de notificaciones, especialmente en aplicaciones sensibles como salud o finanzas.

7. **Sugiera recursos de apoyo y fomente la reflexión crítica.** Recomiende videos, lecturas y podcasts sobre integración avanzada de Firebase, mejores prácticas de seguridad y tendencias en servicios en la nube. Plantee preguntas de reflexión, por ejemplo: ¿cómo priorizarías los métodos de autenticación para diferentes públicos?, ¿qué estrategias usarías para resolver conflictos de sincronización en tiempo real?, y ¿qué consideraciones éticas aplicarías al diseñar notificaciones push en aplicaciones de salud mental?

Tema 12 Accesibilidad y diseño inclusivo en aplicaciones móviles

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Inicie la sesión contextualizando la accesibilidad como un imperativo ético, legal y de mercado.** Explique que una de cada siete personas vive con alguna discapacidad y que, según la OMS y Google (2025), las aplicaciones que ignoran la accesibilidad pueden perder hasta el 40% de su audiencia potencial en tres meses. Destaque la relevancia de normativas como WCAG 2.2, Material Design 3 y la Ley General de Inclusión Digital, y cómo la accesibilidad impacta la competitividad y reputación de las aplicaciones.
2. **Explique los fundamentos de accesibilidad y la compatibilidad con lectores de pantalla.** Demuestre cómo TalkBack (Android) y VoiceOver (iOS) permiten a millones de usuarios con discapacidad visual interactuar con aplicaciones móviles. Ilustre el uso de etiquetado semántico (contentDescription en XML, semantics en Compose), el manejo lógico del foco y la navegación por grupos lógicos. Realice ejemplos prácticos de implementación y validación con herramientas como Accessibility Scanner y Lint de Android Studio.
3. **Presente las mejores prácticas para crear aplicaciones accesibles.** Aborde principios clave como el contraste mínimo de 4.5:1, objetivos táctiles de al menos 48x48 dp, navegación predecible y soporte para múltiples modos de entrada. Realice ejercicios prácticos de ajuste de contraste, etiquetado y validación automatizada. Destaque la importancia de construir accesibilidad desde el diseño y no como un añadido posterior.
4. **Analice el diseño inclusivo para personas con discapacidades visuales, auditivas y motoras.** Explique técnicas como alternativas visuales y hápticas para alertas, subtítulos para contenido multimedia, soporte para Switch Access y navegación por voz. Realice ejemplos de adaptación de la interfaz para distintos perfiles de usuario y promueva la empatía técnica como base de la inclusión real.
5. **Demuestre el uso de herramientas de prueba de accesibilidad en Android Studio y dispositivos reales.** Enseñe a utilizar Lint, Accessibility Scanner, el panel de accesibilidad en el Layout Editor y pruebas manuales con TalkBack. Explique cómo integrar pruebas automatizadas con Espresso y la importancia de validar con usuarios reales para identificar barreras no detectadas por herramientas automáticas.
6. **Guíe la optimización de interfaces para diferentes tamaños y resoluciones de pantalla.** Explique el uso de WindowSizeClass en Jetpack Compose, ConstraintLayout, recursos alternativos y simuladores de

densidad para asegurar la adaptabilidad de la app en móviles, tablets y dispositivos plegables. Realice ejercicios prácticos de diseño responsivo y adaptación visual.

7. **Sugiera recursos de apoyo y fomente la reflexión crítica.** Recomiende videos, lecturas y podcasts sobre accesibilidad, diseño inclusivo y tendencias legales y técnicas. Plantee preguntas como ¿qué cambios priorizarías al rediseñar una aplicación bancaria para usuarios con discapacidad visual?, ¿cómo equilibrarías la estética visual con los requisitos de contraste WCAG?, ¿qué criterios usarías ante un conflicto entre innovación visual y normas de accesibilidad?

Tema 13 Gestión del ciclo de vida de aplicaciones móviles

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Comience la sesión contextualizando la gestión del ciclo de vida como pilar de calidad y competitividad en el desarrollo móvil.** Explique que una estrategia clara de versiones, actualizaciones y mantenimiento es fundamental para la retención de usuarios, la seguridad y la visibilidad en tiendas de aplicaciones. Destaque cómo la gestión integral del ciclo de vida abarca desde la planificación y el desarrollo hasta el soporte y la eventual retirada de la aplicación, promoviendo eficiencia y reducción de riesgos comerciales.
2. **Explique las estrategias de gestión de versiones y actualizaciones en Android.** Presente el versionado semántico (SemVer: MAJOR.MINOR.PATCH) y la importancia de actualizar correctamente los campos versionName y versionCode en cada nueva versión. Analice las diferencias entre actualizaciones incrementales y forzadas, y la relevancia de la automatización y la comunicación clara con los usuarios para mejorar la percepción y el posicionamiento de la aplicación en Google Play.
3. **Demuestre el uso de Git como sistema de control de versiones para equipos multidisciplinarios.** Explique cómo Git permite la colaboración eficiente entre diseñadores y desarrolladores, el manejo de ramas (feature, release, main), el uso de .gitignore para mantener repositorios limpios y la integración de herramientas como Git LFS para archivos pesados. Realice ejercicios prácticos de commits semánticos y resolución de conflictos, y enfatice la importancia de la trazabilidad y la seguridad en el flujo de trabajo.
4. **Guíe la implementación de pruebas unitarias y de integración en Android Studio.** Explique la diferencia entre pruebas unitarias (JVM local, JUnit, Mockito) y pruebas de integración (dispositivos/emuladores, Espresso, UI Automator). Demuestre cómo estructurar y automatizar pruebas para asegurar la calidad continua, reducir errores en producción y validar la interacción entre componentes. Proponga ejercicios de cobertura de casos críticos y edge cases.
5. **Explique técnicas y mejores prácticas para el manejo de excepciones y errores en aplicaciones móviles.** Analice el uso de estructuras try-catch, excepciones personalizadas, manejo global de errores y herramientas de monitoreo como Firebase Crashlytics. Realice ejemplos prácticos de validación de

entradas, manejo de errores en corrutinas y estrategias para evitar cierres inesperados, enfatizando la importancia de la resiliencia y la experiencia del usuario.

6. **Proponga actividades prácticas de integración de ciclo de vida y colaboración.** Invite a los estudiantes a aplicar versionado semántico en un proyecto real, gestionar ramas y assets con Git, implementar pruebas automatizadas y diseñar un flujo de manejo de errores robusto. Sugiera el uso de herramientas de automatización y monitoreo para simular un entorno de desarrollo profesional.
7. **Sugiera recursos de apoyo y fomente la reflexión crítica.** Recomiende videos, lecturas y podcasts sobre gestión de versiones, control de código, pruebas automatizadas y manejo de errores. Plantee preguntas, por ejemplo, ¿cómo minimizarías el impacto en usuarios durante una actualización crítica?, ¿qué prácticas de Git son esenciales para equipos mixtos?, ¿cómo asegurarías la calidad continua en un ciclo ágil?

Tema 14 Despliegue y publicación de aplicaciones en Google Play

Notas para la enseñanza del tema

Al profesor impartidor, se le recomienda lo siguiente:

1. **Comience la sesión contextualizando el despliegue y la publicación como la culminación técnica y estratégica del desarrollo de aplicaciones.** Explique que publicar en Google Play es más que un trámite: es un proceso que impacta la visibilidad, adopción y éxito comercial de la aplicación. Destaque que, según Google (2025), el 65% de las descargas provienen de búsquedas orgánicas, lo que subraya la importancia de la optimización de metadatos y el cumplimiento normativo.
2. **Explique el proceso técnico y administrativo para publicar una aplicación en Google Play Store.** Guíe paso a paso: creación de cuenta de desarrollador (pago único de $25 USD), generación y firma del Android App Bundle (AAB), configuración de la ficha de Play Store (título, descripción, capturas, icono), cumplimiento de requisitos técnicos (minSdkVersion, icono 512x512 px), y carga del paquete en la consola. Resalte la importancia de pruebas cerradas para cuentas nuevas (mínimo 20 testers por 14 días).
3. **Demuestre el cumplimiento de políticas y requisitos técnicos para evitar rechazos.** Analice las políticas de privacidad, uso de SDKs actualizados, clasificación de contenido y declaración de permisos sensibles. Realice una revisión de errores comunes (AAB sin firmar, iconos incorrectos, falta de política de privacidad) y cómo solucionarlos antes de enviar la aplicación a revisión de Google, que puede tardar de 2 a 7 días.
4. **Guíe la optimización de la ficha de Play Store y los elementos visuales para maximizar descargas (ASO).** Explique la importancia de títulos claros (≤30 caracteres), descripciones cortas con keywords (≤80 caracteres), descripciones largas estructuradas y capturas de pantalla en HD mostrando el valor diferencial de la aplicación. Destaque la localización de metadatos en al menos cinco idiomas y el uso de herramientas como Device Art Generator y Google Play Translation Service.

5. **Presente estrategias de mantenimiento post-lanzamiento y gestión de actualizaciones.** Explique que el lanzamiento es solo el inicio: la relevancia y seguridad dependen de actualizaciones frecuentes, monitoreo proactivo (Firebase Crashlytics, Google Analytics), gestión de incidencias (Jira, GitLab) y recopilación de feedback para priorizar mejoras. Analice tipos de mantenimiento (correctivo, evolutivo, adaptativo) y la importancia del despliegue continuo (CI/CD).
6. **Proponga actividades prácticas de simulación de publicación y mantenimiento.** Invite a los estudiantes a generar y firmar un AAB, completar una ficha de Play Store con metadatos optimizados, realizar pruebas cerradas y simular el proceso de revisión y lanzamiento. Sugiera ejercicios de monitoreo post-lanzamiento y respuesta a incidencias reales o simuladas.
7. **Sugiera recursos de apoyo y fomente la reflexión crítica.** Recomiende videos, lecturas y podcasts sobre publicación, optimización ASO y mantenimiento post-lanzamiento. Plantee algunas preguntas: ¿cómo priorizarías keywords en una categoría saturada?, ¿qué estrategias usarías para reducir la tasa de desinstalación en la primera semana?, ¿cómo ajustarías los metadatos tras un rechazo por “contenido engañoso” sin perder la esencia comercial?

Evidencia

Avance 1

Mencionar a los alumnos que pueden trabajar el proyecto en equipos de dos o tres personas.

Ayuda a los estudiantes a desarrollar habilidades en la creación de mensajes y campañas que promuevan estilos de vida sostenibles de manera efectiva. Es importante que se fomente la investigación de público objetivo, la identificación de barreras psicológicas y culturales para la adopción de comportamientos sostenibles y la creación de estrategias de comunicación persuasivas que puedan motivar y comprometer a la comunidad.

Enseñar técnicas para medir la efectividad de la campaña en términos de conciencia pública, cambio de comportamiento y reducción de impacto ambiental. Incluye el uso de indicadores cuantitativos y cualitativos para analizar el éxito de la campaña y ajustar estrategias según sea necesario.

Ayudar a una creación efectiva de logotipo y un brief completo que incluya todo lo requerido en las instrucciones del avance.

Entrega final

En esta entrega final es importante recordar al alumno que esta es una continuación del avance de proyecto.

Apoyar al alumno en la realización de la serie de diseños que se requieren en esta última entrega. Revisar que se cumpla lo aprendido a lo largo de los temas del curso.

Es importante que el alumno realice la presentación del desarrollo de su proyecto e incentivarlo a utilizar medios interactivos para brindar un trabajo dinámico e innovador.