Entrenador en plataforma iOS caso de estudio “Eventos de Ciudad”

Estudiante:
Diego Andrés Mazo Muñoz

Informe final bajo la modalidad de práctica empresarial presentado para optar al título de:
Ingeniero de Sistemas

Asesor Interno:
Diana Margot López Herrera

Asesor Externo:
Jaime Laino

Departamento de Ingeniería de Sistemas
Facultad de Ingeniería
Universidad de Antioquia
Medellín, Colombia
2019

Resumen

Con el objetivo de aumentar el número de desarrolladores especializados para los proyectos que día a día llegan a Globant, se realiza un proceso de entrenamiento en el desarrollo de aplicaciones iOS de Apple para los practicantes, este entrenamiento es manual, ya que la guía se encuentra en un documento de texto donde los practicantes acceden a links externos para aprender los temas designados por el referente, por esto es que se propuso crear una aplicación iOS que sirva de "entrenador en la plataforma iOS" a partir de un caso de estudio llamado "Eventos de ciudad" con contenidos teóricos específicos, ejercicios explicativos y un componente evaluador.

Como resultado principal del proyecto se tiene el logro del objetivo general planteado en la propuesta inicial, ya que se construyó una aplicación iOS que sirve de entrenador para practicantes en el área de desarrollo de aplicaciones iOS con Swift, a partir de la implementación del caso de estudio mencionado anteriormente.

Introducción

Globant es una compañía líder en desarrollo de software fundada en 2003 en donde la ingeniería, el diseño y la innovación se encuentran para llegar a gran escala. Utiliza las últimas tecnologías en el campo digital y cognitivo para propulsar organizaciones en todo aspecto.

Globant tiene más de 7.000 profesionales trabajando para compañías como Google, LinkedIn, BBVA, EA, FOX y Coca Cola, entre otras. Cuenta con 35 centros de desarrollo en 25 ciudades distribuidas en 12 países.

En Colombia, Globant inició operaciones en Bogotá en el año 2009 y dos años después abrió sus oficinas en Medellín.

En el Site de Medellín, Globant cuenta con múltiples equipos y desarrolladores trabajando en proyectos para Disney, una compañía mundial de entretenimiento, para la cual se desarrollan diferentes aplicaciones.

Con el objetivo de aumentar el número de desarrolladores especializados para los proyectos que día a día llegan a la compañía el Site realiza un proceso de entrenamiento en el desarrollo de aplicaciones iOS de Apple para los practicantes que ingresan cada semestre a la compañía, dicho entrenamiento es guiado y coordinado por un referente en iOS, que es el encargado de asignar tareas y evaluarlas.

Actualmente, este entrenamiento es realizado de manera manual, ya que la guía se encuentra en un documento de texto donde los practicantes

acceden a links externos para aprender los temas designados por el referente, haciendo así el entrenamiento tedioso y vulnerable a perderse cuando el referente no esté presente.

Es por esto que se propuso crear una aplicación de ejemplo que sirva de "entrenador en la plataforma iOS" a partir del caso de estudio "Eventos de ciudad" con contenidos teóricos específicos, ejercicios explicativos y un componente evaluador a través de preguntas de selección múltiple, para que los practicantes de manera autodidacta puedan entrenar las habilidades requeridas para el desarrollo iOS y puedan realizar procesos de autoevaluación.

Objetivos

Objetivo General

Implementar un entrenador para practicantes en el área de desarrollo de aplicaciones iOS con Swift, a partir del caso de estudio "Eventos de ciudad" utilizando el IDE Xcode y el lenguaje de programación Swift.

Objetivos específicos

- Seleccionar las funcionalidades básicas que tiene el caso de estudio "Eventos de ciudad" y elegir las más representativas para ser implementadas en el entrenador.

- Diseñar una guía instruccional, de las funcionalidades básicas a implementar y del tratamiento que se dará a las demás funcionalidades en el contexto del caso de estudio.

- Seleccionar los conceptos de entrenamiento que se documentarán alrededor de las funcionalidades no implementadas y su pertinencia en la creación de una aplicación para dispositivos iOS.

- Diseñar las funcionalidades necesarias para el desarrollo de aplicaciones en dispositivos iOS, correspondiente al caso de estudio.

- Codificar la funcionalidad correspondiente del caso de estudio, colocando elementos conceptuales y explicativos en cada momento que sea pertinente.

- Construir un componente evaluador del conocimiento y de la habilidad de desarrollo que ha logrado el practicante, a partir de preguntas cerradas y de selección múltiple con única respuesta.

Marco Teórico

En Globant cada semestre ingresan nuevos practicantes al área de desarrollo móvil, específicamente para el desarrollo de aplicaciones iOS, y la empresa para asegurar la calidad técnica de dichos desarrolladores realiza un proceso de entrenamiento en dónde brinda las bases necesarias en el lenguaje de programación Swift y el correcto manejo del IDE Xcode.

El entrenamiento es guiado por un referente iOS que asigna tareas a los practicantes basado en una guía de entrenamiento creada y son evaluadas de manera manual. Con el fin de aportar una herramienta automatizada que ayude en el proceso de entrenamiento, se propone crear una aplicación iOS que sirva de entrenador y a la vez de evaluador, para que el proceso no sea manual, y que el practicante cuente con un mecanismo de autoevaluación que facilite el proceso de evaluación del referente e incentive el proceso de autoestudio en los practicantes.

El **autoaprendizaje**[1] es la forma de aprender por uno mismo. Se trata de un proceso de adquisición de conocimientos, habilidades, valores y actitudes, que la persona realiza por su cuenta ya sea mediante el estudio o la experiencia. Un sujeto enfocado al autoaprendizaje busca por sí mismo la información y lleva adelante las prácticas o experimentos de la misma forma. Para que las personas apuesten por llevar a cabo esa forma de aprender y para que consigan el éxito de la misma es importante resaltar que necesitan que tengan que llevarse a cabo tres elementos. Es decir, tienen que existir: responsabilidad, aprendizaje permanente y estudio independiente.

La **autoevaluación**[2] es un método que consiste en valorar uno mismo la propia capacidad que se dispone para tal o cual tarea o actividad, así como también la calidad del trabajo que se lleva a cabo, especialmente en el ámbito pedagógico.

Estos dos conceptos mencionados son habilidades de gran importancia para el desarrollo integral de un Ingeniero de Sistemas, ya que en el aprendizaje tecnológico es muy usual encontrarse con la necesidad de aprender por cuenta propia algún concepto o herramienta y si se brinda una herramienta a los practicantes que permita el uso y mejoramiento de estas habilidades se está facilitando el proceso de adaptación y de entrenamiento de los mismos en la empresa.

Para crear el entrenador se utilizarán las siguientes herramientas:

Xcode[3]

Xcode es un entorno de desarrollo integrado (IDE, en sus siglas en inglés) para el sistema operativo macOS que contiene un conjunto de herramientas

creadas por Apple destinadas al desarrollo de software para macOS, iOS, watchOS y tvOS.
Su primera versión tiene origen en el año 2003 y actualmente su versión número 10 se encuentra disponible de manera gratuita en el Mac App Store o mediante descarga directa desde la página para desarrolladores de Apple.

Xcode trabaja conjuntamente con Interface Builder, una herencia de NeXT, una herramienta gráfica para la creación de interfaces de usuario.

Xcode incluye la colección de compiladores del proyecto GNU (GCC), y puede compilar código C, C++, Swift, Objective-C, Objective-C++, Java y AppleScript mediante una amplia gama de modelos de programación, incluyendo, pero no limitado a Cocoa, Carbón y Java. Otras compañías han añadido soporte para GNU Pascal, Free Pascal, Ada y Perl.

Swift[4]

Es un lenguaje de programación multiparadigma creado por Apple enfocado en el desarrollo de aplicaciones para iOS y macOS. Fue presentado en WWDC 20147 y está diseñado para integrarse con los Frameworks Cocoa y Cocoa Touch, puede usar cualquier biblioteca programada en Objective-C y llamar a funciones de C. También es posible desarrollar código en Swift compatible con Objective-C bajo ciertas condiciones. Swift tiene la intención de ser un lenguaje seguro, de desarrollo rápido y conciso. Usa el compilador LLVM incluido en Xcode 6. En el año 2015 pasó a ser de código abierto.

Respecto a la curva de aprendizaje del lenguaje Apple buscando impulsar su aceptación a lo largo del globo y con esto poder incrementar la cantidad de desarrolladores que crean apps para el sistema operativo iOS y los demás sistemas, ha presentado varias instancias de aprendizaje, desde las más convencionales como ebooks publicados en su página web, blogs y presentaciones anuales en la WWDC con cursos de swift y sesiones prácticas con la posibilidad de hacer preguntas a los desarrolladores de la compañía, como así también otras más interesantes, como los cursos de Swift en determinadas universidades del mundo o una app para iPad llamada Swift Playgrounds, donde se puede aprender a programar "jugando" gracias al ambiente de desarrollo interactivo y táctil que presenta.

Aprender a programar en Swift permite que las aplicaciones que un desarrollador crea puedan correr en una amplia gama de productos, inicialmente en los dispositivos de Apple pero en un futuro cercano también en productos de otras compañías, ya que empresas como Facebook, Uber y hasta Google están estudiando las bondades de este lenguaje.

Swift está basado en lenguajes muy populares y reconocidos, como C y objective C, pero gracias a su desarrollo continuo, Apple planea llevarlo al siguiente nivel, en donde es posible crear código seguro usando menos líneas que su competencia pero a su vez dando la posibilidad armar

algoritmos de calidad, gracias a su integración con el compilador LLVM que permite hacer rendir al máximo el hardware en el que corre el programa. [5]

iOS[6]

iOS es un sistema operativo móvil de la multinacional Apple Inc. Originalmente desarrollado para el iPhone (iPhone OS), después se ha usado en dispositivos como el iPod touch y el iPad. No permite la instalación de iOS en hardware de terceros.

Actualmente es el segundo sistema operativo móvil más utilizado del mundo, detrás de Android, con una cuota de mercado de entre 10-15% al año 2017. La última versión del sistema operativo es iOS 12 aparecida en junio de 2018 sustituye a iOS 11 con el objetivo principal de mejorar la experiencia del usuario.

Los elementos de control consisten de deslizadores, interruptores y botones. La respuesta a las órdenes del usuario es inmediata y provee una interfaz fluida. La interacción con el sistema operativo incluye gestos como deslices, toques, pellizcos, los cuales tienen definiciones diferentes dependiendo del contexto de la interfaz. Se utilizan acelerómetros internos para hacer que algunas aplicaciones respondan a sacudir el dispositivo (por ejemplo, para el comando deshacer) o rotarlo en tres dimensiones (un resultado común es cambiar de modo vertical al apaisado u horizontal).

iOS y su arquitectura interna [7]

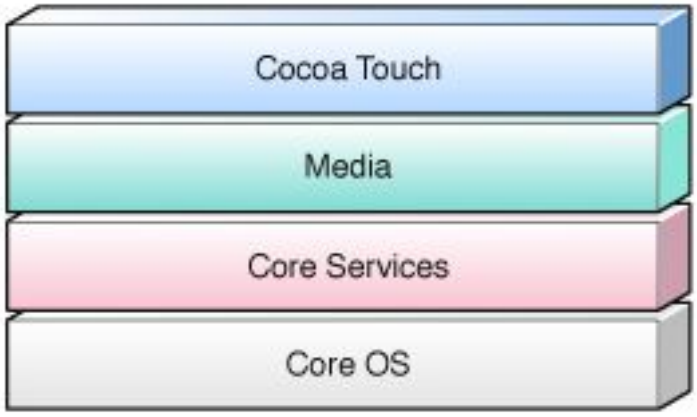

Figura 1. Arquitectura interna iOS [7].

El sistema operativo de Apple está distribuido en cuatro capas diferenciadas por su funcionalidad:

Cocoa Touch: Es la capa superior, la que los usuarios utilizan para interactuar con las aplicaciones, es decir, la capa visible. Es la zona donde nos

encontramos los componentes visuales, se trata de una capa de abstracción.

Media: Se trata de una capa basada en la mezcla de lenguaje C y Objective C que contiene las tecnologías que dan acceso a ficheros multimedia relacionados con audio, gráficos, vídeos, etc.

Core Services: Se trata de la capa de servicios principales disponibles en el dispositivo y que pueden ser utilizados por todas las aplicaciones, como pueden ser: base de datos SQLite, acceso a la red, soporte para XML.

Core OS: El núcleo del sistema. Recordar que el sistema operativo iOS está basado en el OS X de Apple, que fue desarrollado a partir de una base Unix. Elementos de seguridad, memoria, procesos o manejo de ficheros son los que podemos encontrar en esta capa.

Metodología

La metodología que se siguió para la realización del proyecto consistió en 3 etapas secuenciales que son mostradas a continuación. En la fase de implementación se utilizaron ciertas partes de Scrum para implementar el desarrollo.

Figura 2. Metodología [Fuente propia].

A continuación se explican en detalle cada una de las fases de la metodología.

- **Análisis:** En esta fase se definieron los requisitos funcionales y no funcionales que fueron implementados en la aplicación.

- **Implementación:**

  En esta fase se utilizó el marco de desarrollo ágil Scrum[8] que es el nombre con el que se denomina a los marcos de desarrollo ágiles caracterizados por:
  - Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto.

- Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos auto organizados, que en la calidad de los procesos empleados.
- Solapamiento de las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o en cascada.

Así que se definió un release con 8 sprints, cada uno con una duración de 15 días. A continuación se detallan los Sprints.

- **Sprint 1:**
  - Definición de las funcionalidades básicas del caso de estudio "Eventos de ciudad"

- **Sprint 2:**
  - Selección de las funcionalidades más representativas a ser implementadas.
  - Creación de la documentación necesaria acerca de los conceptos que no se van a implementar.

- **Sprint 3:**
  - Diseño de las funcionalidades que fueron elegidas para implementar.

- **Sprint 4:**
  - Codificación de las funcionalidades selecciones para implementar.

- **Sprint 5:**
  - Creación de un servicio web en donde se consulten los contenidos del tema y los ejemplos correspondientes.

- **Sprint 6:**
  - Creación de la aplicación "entrenador".

  - Creación de una tableView en donde se muestran los contenidos a estudiar por el practicante.
  - Creación de un DetailView donde se especifique el contenido de cada tema con sus respectivos detalles.

- **Sprint 7:**
  - Creación de una vista que se despliegue después de que el practicante aprenda un tema en donde se realice una evaluación con preguntas de única respuesta y selección múltiple.

- **Sprint 8:**
  - Creación de una vista de indicadores de avance en el desarrollo del caso de estudio "Eventos de ciudad".

- **Pruebas:** Durante el proceso de implementación se realizaron pruebas unitarias donde se probaron todas las funcionalidades implementadas.

Cronograma de Actividades

A continuación se muestra el cronograma de actividades que se planeó para llevar a cabo el desarrollo del proyecto.

Actividades	Mes 1		Mes 2		Mes 3		Mes 4	
Seleccionar las funcionalidades básicas que tiene el caso de estudio “Eventos de ciudad” y elegir las más representativas para ser implementadas en el entrenador.	Sprint 1							
Diseñar una guía instruccional, de las funcionalidades básicas a implementar y del tratamiento que se dará a las demás funcionalidades en el contexto del caso de estudio.		Sprint 2						
Codificar la funcionalidad correspondiente del caso de estudio, colocando elementos conceptuales y explicativos en cada momento que sea pertinente.			Sprint 3	Sprint 4				
Seleccionar los conceptos de entrenamiento que se documentarán alrededor de las funcionalidades no implementadas y su pertinencia en la creación de una aplicación para dispositivos iOS.					Sprint 5			
Diseñar las funcionalidades necesarias para el desarrollo de aplicaciones en						Sprint 6		

dispositivos iOS, correspondiente al caso de estudio.								
Codificar la funcionalidad correspondiente del caso de estudio, colocando elementos conceptuales y explicativos en cada momento que sea pertinente.							Sprint 7	
Construir un componente evaluador del conocimiento y de la habilidad de desarrollo que ha logrado el practicante, a partir de preguntas cerradas y de selección múltiple con única respuesta.								Sprint 8

Tabla 1. Cronograma de actividades [Fuente propia].

Resultados y análisis

El resultado principal del proyecto es el logro del objetivo general planteado en la propuesta inicial, ya que se construyó una aplicación iOS que sirve de entrenador para practicantes en el área de desarrollo de aplicaciones iOS con Swift, a partir de la implementación de un caso de estudio llamado "Eventos de ciudad".

En la propuesta se tenían como resultados esperados la creación de una guía instruccional con las funcionalidades básicas a implementar en el caso de estudio y la de una aplicación que permitiera mostrar los contenidos y desplegar un componente evaluador de lo aprendido.

Las funcionalidades básicas más representativas que fueron implementadas son:

- Uso de librerías mediante el manejador de dependencias "CocoaPods".
- Integración del iOS SDK de Google Maps en la aplicación.
- Visualización de mapas.
- Uso de los servicios de localización del celular mediante el gps.
- Implementación de TableViews para mostrar el contenido.
- Visualización de lugares de interés basados en la ubicación del usuario.

Todos estos resultados esperados se conjugan en la aplicación desarrollada que se describirá a continuación.

Aplicacion iOS desarrollada:

A continuación se mostrará de manera detallada el funcionamiento de la aplicación iOS creada en el proyecto.

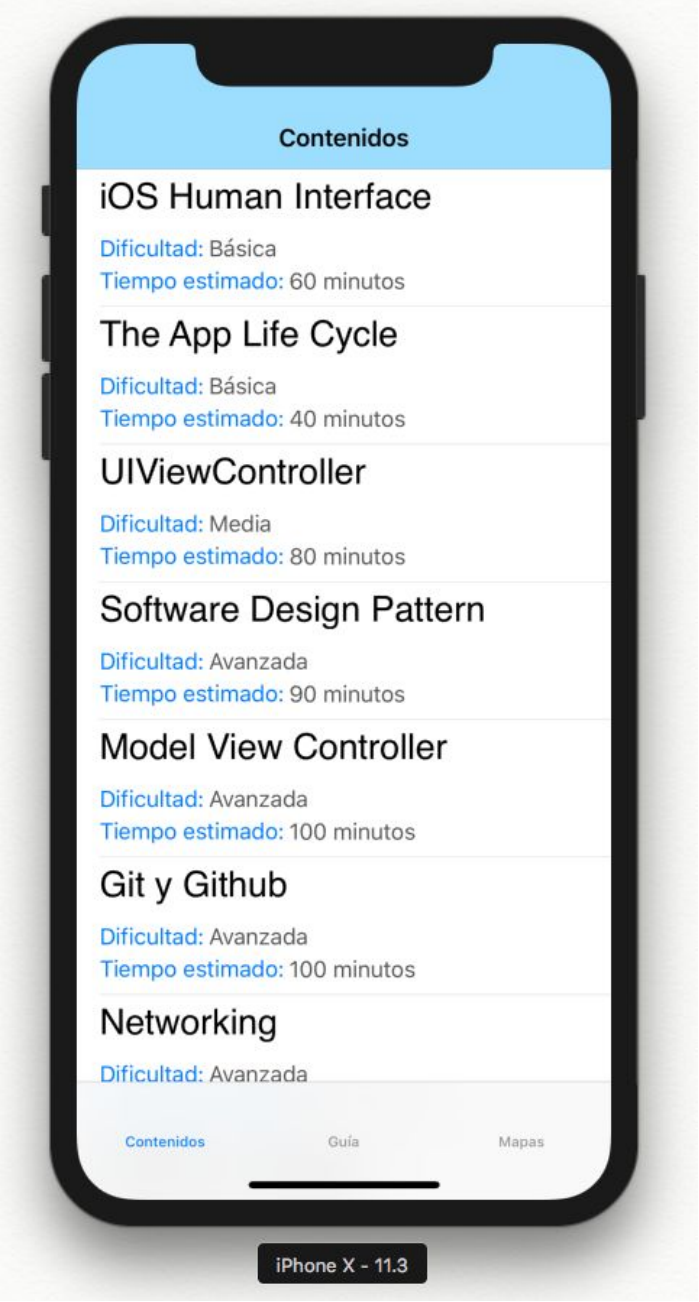

Figura 3. Aplicación iOS.

En la *figura 3* se observa la aplicación iOS después de haber sido ejecutada por primera vez. En la barra inferior (Tab Bar Controller) se encuentran los tres grandes módulos en los que está dividida la aplicación: Contenidos, Guía, Mapas. Dichos módulos se explicarán a continuación.

Módulo "Contenidos"

En este módulo de la aplicación se seleccionaron los conceptos teóricos de entrenamiento para ser mostrados a los practicantes, dichos conceptos fueron obtenidos de la guía creada por el referente iOS de Globant.

Los conceptos teóricos que fueron documentaron y se muestran de manera amigable en la aplicación son los siguientes:

- "iOS Human Interface"
- "The App Life Cycle"
- "UIViewController"
- "Software Design Pattern"
- "Model View Controller"
- "Git and Github"
- "Networking"
- "JSON y Parsing"
- "UITableView"
- "UICollectionView"
- "SOLID"
- "TDD"
- "Unit Testing"

En total son 13 conceptos teóricos documentados en la aplicación, estos son la base conceptual para convertirse en un desarrollador iOS Junior. Cabe aclarar que los títulos de los conceptos están en inglés ya que así son nombrados globalmente pero todas las explicaciones detalladas se encuentran en español.

A continuación se muestra la lista de contenidos mostrados en la aplicación.

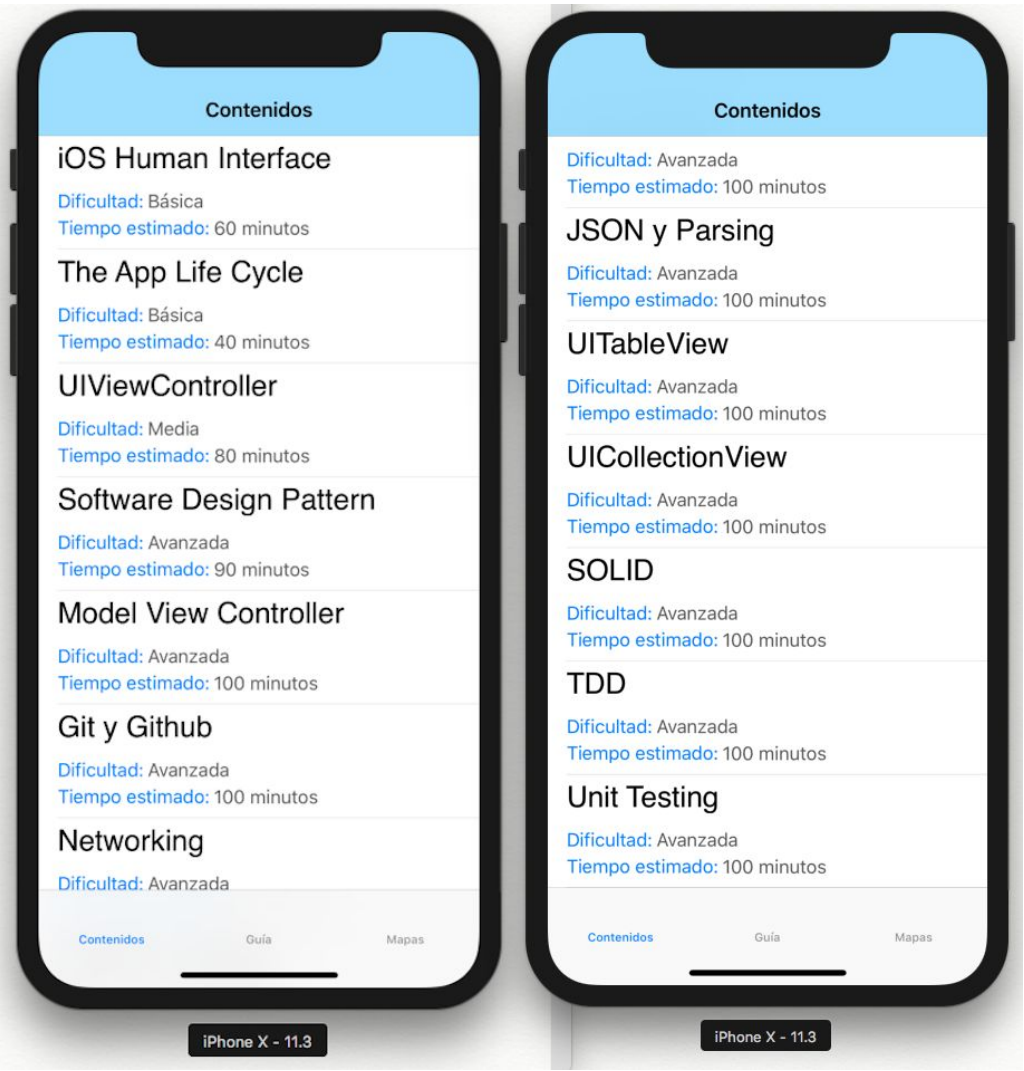

Figura 4. Lista de contenidos.

En la *figura 4* se observa la lista completa de los conceptos teóricos, cada concepto tiene una dificultad que puede ser básica, media o avanzada, y además se le muestra al usuario el tiempo estimado que le llevará leer y entrenar el concepto, facilitando así su planeación y los procesos de autoevaluación.

A continuación se muestra la vista en detalle que es desplegada después de seleccionar cualquier contenido.

Figura 5. Detalle del contenido, iOS Human Interface.

En la *figura 5* se muestra el detalle del contenido, allí se encuentra la explicación teórica del contenido, una imagen relacionada y diferentes páginas con ejercicios propuestos y enlaces de interés.

Figura 6. Detalle del contenido, Software Design Pattern.

En la *figura 6* se observa una de las funcionalidades creadas en la vista detalle: el contenido es mostrado imitando a un libro, así el usuario podrá navegar por el contenido como si estuviera pasando páginas en un libro, haciendo que la lectura sea más amena y atractiva.

Por último, en el módulo "Contenidos" se encuentra un componente evaluador del conocimiento y de la habilidad de desarrollo que ha logrado el practicante en el concepto que acaba de ser estudiado, a partir de preguntas cerradas y de selección múltiple con única respuesta. Dicho componente se encuentra al final de la lectura del contenido y será mostrado a continuación.

Figura 7. Componente evaluador, respuesta incorrecta.

En la figura anterior se muestra el componente evaluador que aparece después de cada concepto estudiado, allí se realizan una serie de preguntas y se evalúan mostrando mensajes al usuario.

El componente evaluador es mantenible y escalable, es decir es posible agregar más preguntas evaluadoras para cada concepto mediante la implementación de un servicio web utilizando JSONs, la aplicación consume este servicio y muestra el componente dinámicamente, permitiendo así modificar o agregar fácilmente nuevas preguntas.

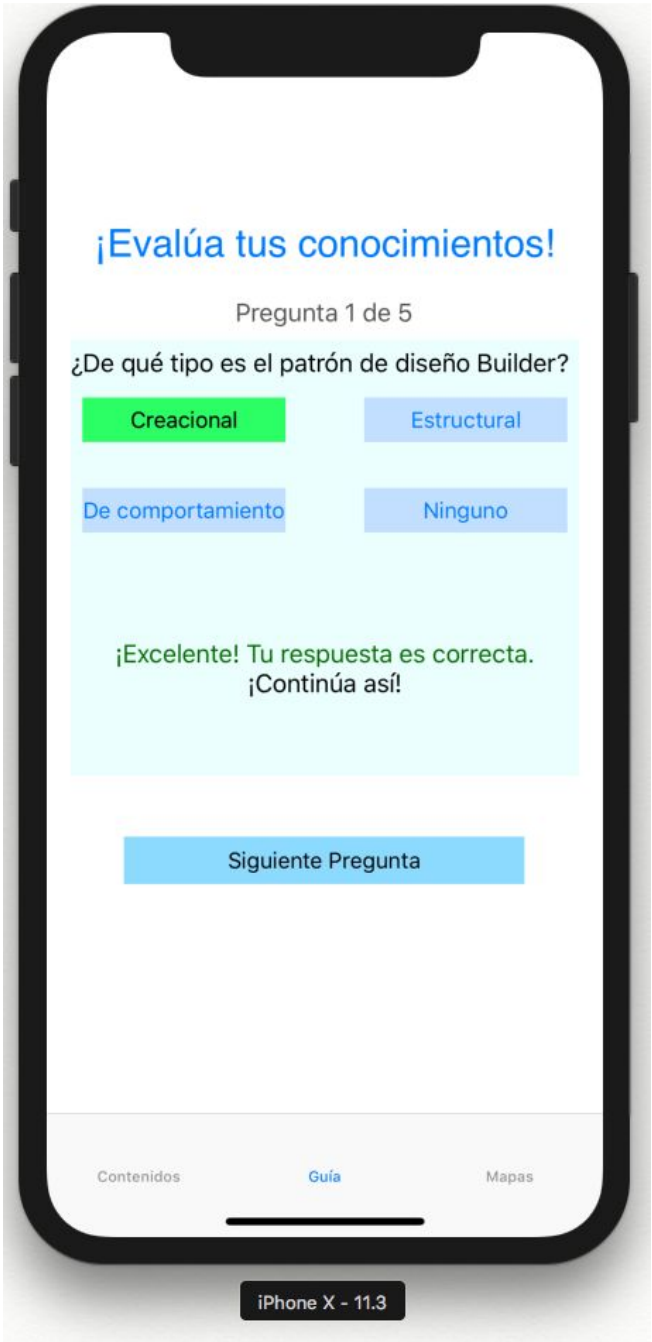

Figura 8. Componente evaluador, respuesta correcta.

En la *figura 8* se observa el comportamiento del componente evaluador cuando el usuario elige la respuesta correcta, allí se presenta la opción de continuar hacia la próxima pregunta.

En la actual implementación del componente evaluador no se obtienen puntajes para la evaluación, pero en futuras implementaciones se puede agregar dicha funcionalidad.

Módulo "Guía"

En este módulo de la aplicación se mostrará al usuario paso a paso cómo realizar la implementación del caso de estudio "Eventos de ciudad".

Se le mostrarán porciones de código para que las funcionalidades puedan ser codificadas por los practicantes mediante un tutorial detallado.

El caso de estudio "Eventos de ciudad" consiste en la implementación del iOS SDK de Google Maps para mostrar un mapa en la aplicación, en este se conocerá la ubicación en tiempo real del usuario mediante el gps del celular y se le mostrarán unos filtros para buscar eventos y lugares de interés cerca del usuario.

A continuación se mostrará el módulo en funcionamiento.

Figura 9. Módulo "Guía"

En la figura anterior se evidencia el inicio del tutorial en el módulo llamado "guía".

Figura 10. Tutorial para implementar el caso de estudio "Eventos de Ciudad"

En la *figura 10* se muestra cómo es mostrado el tutorial para implementar el caso de estudio en la aplicación.

El practicante conocerá que realizó el proceso correcto mediante la comparación de su implementación con la que existe en la aplicación en el módulo "Mapas", dichas implementaciones deberían ser similares. El tiempo estimado para la implementación del módulo es de una semana.

Módulo "Mapas"

En este módulo de la aplicación se muestra la implementación de las funcionalidades del caso de estudio "Eventos de Ciudad". Es el resultado tangible del tutorial enseñado en el módulo "Guía", es decir, el practicante debería desarrollar una funcionalidad similar a esta después de realizar correctamente el tutorial en el módulo "Guía".

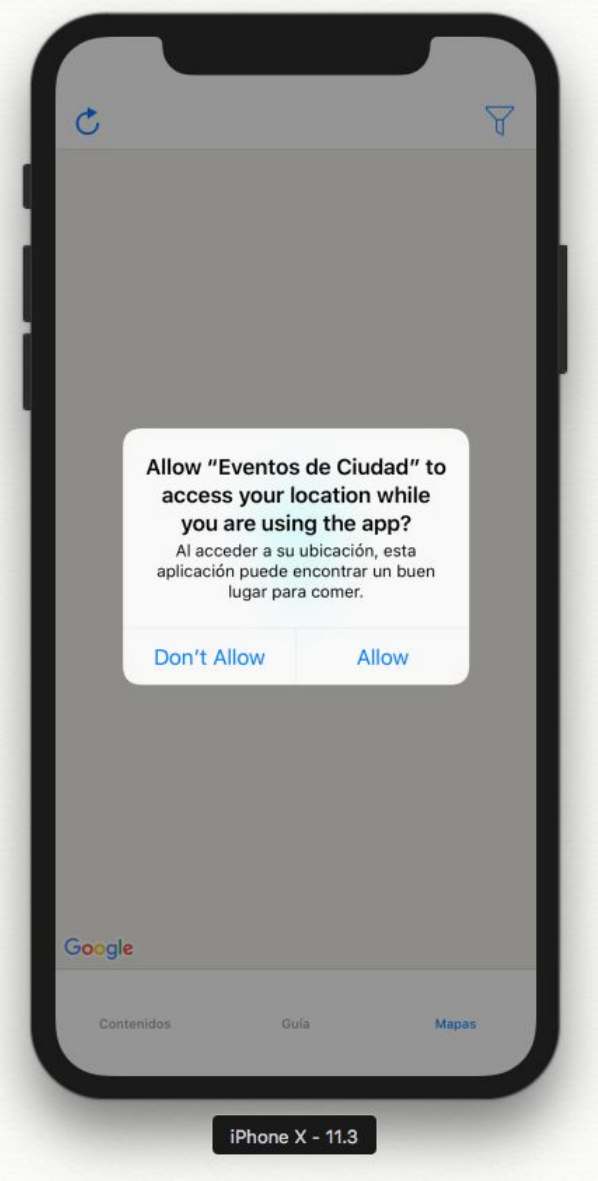

Figura 11. Petición de permisos, módulo "Mapas"

Cuando el usuario ingresa al módulo de "Mapas" lo primero que se encontrará es una ventana pidiéndole permisos para acceder al gps del celular mientras se usa la aplicación, el usuario deberá aceptar para que la

aplicación pueda conocer su ubicación y así mostrarle los sitios de interés cercanos.
Después, la aplicación mostrará la ubicación del usuario.

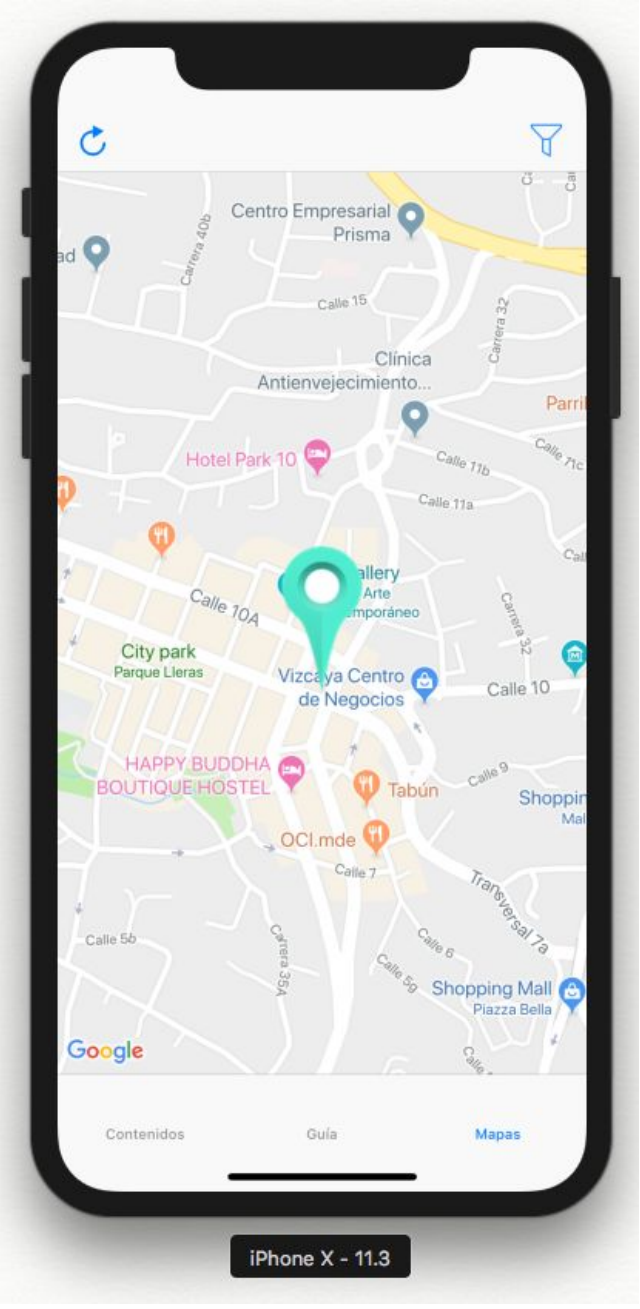

Figura 12. Ubicación del usuario.

En la *figura 12* se observa cómo la aplicación encuentra la ubicación exacta del usuario.

A continuación se mostrará cómo el usuario podrá acceder a unos filtros para buscar lugares de interés cercanos a su ubicación. Se accede

pulsando el botón con forma de embudo que se encuentra ubicado en la parte superior y a la derecha.

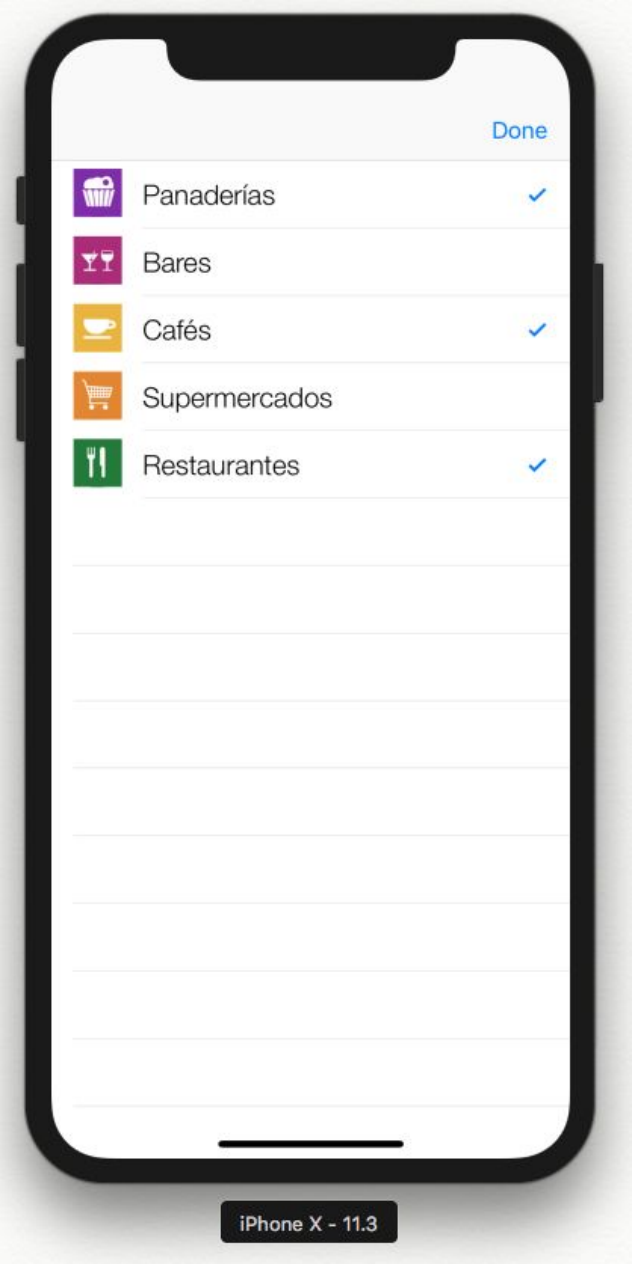

Figura 13. Filtros para buscar sitios de interés.

Como es mostrado en la *figura 13,* el usuario podrá buscar sitios de interés de 5 diferentes tipos: panaderías, bares, cafés, supermercados y restaurantes.

Después de seleccionar los filtros de interés, se mostrarán en el mapa los sitios cercanos.

Conclusiones

- El autoaprendizaje como proceso de adquisición de conocimientos es una habilidad muy valiosa y vital en la ingeniería de sistemas y debe ser entrenada al igual que se hace con las habilidades técnicas, en el día a día la mayoría de los conocimientos son adquiridos mediante este medio y desarrollarla hará que el aprovechamiento del tiempo sea más efectivo.

- Al igual que el autoaprendizaje, la autoevaluación es primordial en el proceso de adquisición de conocimientos, gracias a esta podemos mejorar de manera gradual y constante enfocandonos en lo que hacemos bien y cambiando lo que se hace mal.

- Depender de tareas manuales en los procesos de entrenamiento de practicantes hace que el entrenamiento sea lento y tedioso, con la aplicación desarrollada se mejora este proceso haciendo que el practicante organice mejor su tiempo y sea más eficaz, aprovechará su tiempo en aprender y no en procesos innecesarios que antes ocurrían.

- Swift es un lenguaje de programación amigable y cuenta con una curva de aprendizaje empinada, es decir, en poco tiempo se aprende mucho. En niveles avanzados se complica bastante pero con un buen proceso de entrenamiento como el ofrecido en la aplicación se hace más llevadero el proceso de adquisición de conocimiento.

- Implementar casos de estudio específicos para aprender conceptos generales resulta muy adecuado, ya que se estudia el concepto teórico e inmediatamente después se pone en práctica haciendo que se domine mucho más el conocimiento adquirido.

Referencias Bibliográficas

[1] Autoaprendizaje
Adell Segura, J. & Castañeda Quintero, L. (2010)
(PDF) Los Entornos Personales de Aprendizaje (PLEs): una nueva manera de entender el aprendizaje.
https://www.researchgate.net/publication/47721781_Los_Entornos_Personales_de_Aprendizaje_PLEs_una_nueva_manera_de_entender_el_aprendizaje

[2] Autoevaluación
CALATAYUD SALOM. A. (2002): "La cultura autoevaluativa, piedra filosofal de la calidad en educación".Revista: Educadores. Núm 204. Págs.357-375.

[3] Xcode
https://es.wikipedia.org/wiki/Xcode

[4] Swift
https://es.wikipedia.org/wiki/Swift_(lenguaje_de_programación)

[5] Swift
https://swiftlatino.com/

[6] iOS
https://es.wikipedia.org/wiki/IOS

[7] Arquitectura iOS
http://blog.ticsandroll.es/ios-y-su-arquitectura-interna-en-4-capas/

[8] Scrum
https://proyectosagiles.org/que-es-scrum/