Desarrollo Basado en Plataformas

CS2B01

PROYECTO - 2019 II

Submitted By :
J. Bellido, PhD
jbellido@utec.edu.pe

Objetivo General

En el proyecto de este curso deberás de crear una aplicación del tema que más te guste, para lo cual todos en tu grupo deberán estar de acuerdo. Esta demás decir que el proyecto no puede ser un chat ni alguna variante de este. Busca un tema que les guste y empiecen a crearlo!

Este proyecto deberá tener dos versiones: 1 parte web y la otra mobile. La Figura 1. muestra el diagrama de architectura de esta aplicación. Lo importante aquí es que las funciones implementadas en el servidor funcionen para ambas partes: Web y Mobile.

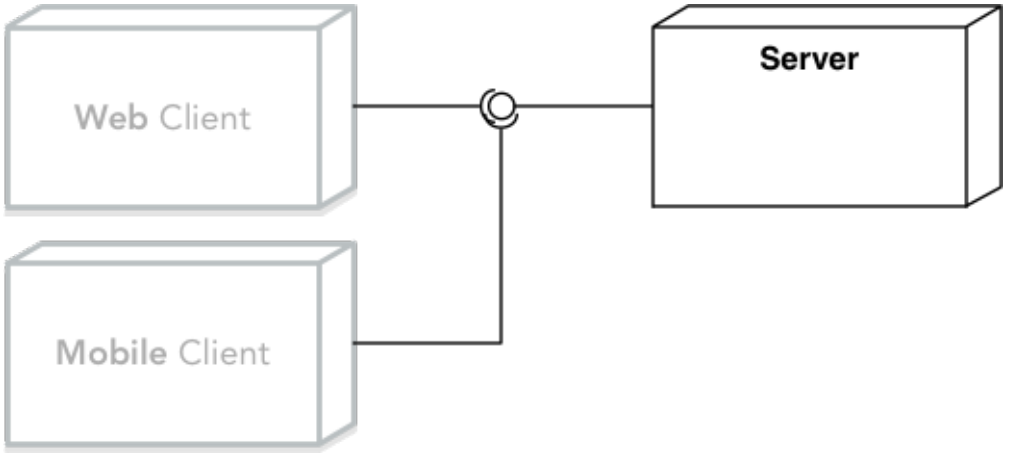

Figure 1: Diagrama de Arquitectura de Software del Proyecto

Parte I: Aplicación Web

La aplicación web deberá cumplir con las restricciones que las plataformas web ponen a los desarrolladores tales como:

Cliente-Servidor

Las primera restricción añadida es del estilo arquitectónico cliente-servidor. La separación de responsabilidades es el principio detrás de la restricción cliente-servidor. Al separar las responsabilidades de la interfaz de usuario de las de almacenamiento de datos, mejoramos la portabilidad de la interfaz de usuario en múltiples plataformas y mejoramos la escalabilidad al simplificar los componentes del servidor.

Stateless - Stateful

Esta restricción indica que la comunicación debe ser de naturaleza sin estado (stateless), de modo que cada solicitud del cliente a el servidor debe contener toda la información necesaria para comprender la solicitud y no puede aprovechar el contexto almacenado en el servidor. Por lo tanto, el estado de la sesión se mantiene completamente en el cliente.

Cache

Para mejorar la eficiencia de la red, agregamos restricciones de caché para formar el estilo cliente-caché-servidor-sin-estado . Las restricciones de caché requieren que los datos dentro de una respuesta a una solicitud se etiqueten implícita o explícitamente como cacheables o no. Si una respuesta es almacenable en caché, se le da a un caché de cliente el derecho de reutilizar los datos de respuesta para solicitudes equivalentes posteriores.

Uniform Interface

Al aplicar el principio de generalización de la ingeniería del software a la interfaz del componente, se simplifica la arquitectura general del sistema y se mejora la visibilidad de las interacciones. Las implementaciones están desacopladas de los servicios que proporcionan, lo que fomenta la capacidad de evolución independiente. La desventaja, sin embargo, es que una interfaz uniforme degrada la eficiencia, ya que la información se transfiere de forma estandarizada en lugar de una que sea específica para las necesidades de una aplicación.

Layered System

Con el fin de mejorar aún más el comportamiento para los requisitos de escala de Internet, agregamos restricciones del sistema en capas, el estilo de sistema en capas permite que una arquitectura se componga de capas jerárquicas al restringir el comportamiento de los componentes de modo que cada componente no pueda "ver" más allá de la capa inmediata con la que están interactuando. Al restringir el conocimiento del sistema a una sola capa, establecemos un límite en la complejidad general del sistema y promovemos la independencia del sustrato. Las capas se pueden usar para encapsular servicios heredados y para proteger servicios nuevos de clientes heredados, simplificando los componentes al mover la funcionalidad utilizada

con poca frecuencia a un intermediario compartido. Los intermediarios también se pueden usar para mejorar la escalabilidad del sistema al permitir el equilibrio de carga de los servicios en múltiples redes y procesadores.

Code-On-Demand

La adición final al conjunto de restricciones proviene del estilo de código según demanda. Esto permite extender la funcionalidad del cliente descargando y ejecutando código en forma de applets o scripts. Esto simplifica a los clientes al reducir el número de características requeridas para ser implementadas previamente. Permitir que las funciones se descarguen después de la implementación mejora la extensibilidad del sistema.

Parte II: Aplicación Móvil

La segunda parte de este proyecto deberás implementar una aplicación nativa móvil para la plataforma Android. Tu aplicación deberá implementar los siguientes elementos de Android:

Activity

Una Activity es un componente de la aplicación que contiene una pantalla con la que los usuarios pueden interactuar para realizar una acción, como marcar un número telefónico, tomar una foto, enviar un correo electrónico o ver un mapa. A cada actividad se le asigna una ventana en la que se puede dibujar su interfaz de usuario. La ventana generalmente abarca toda la pantalla, pero en ocasiones puede ser más pequeña que esta y quedar "flotando" encima de otras ventanas.

Intents

Un Intent es un objeto de mensajería que puede usar para solicitar una acción desde otro componente de la aplicación. Aunque los intents facilitan la comunicación entre los componentes de varias maneras, existen tres casos de uso fundamentales: iniciar una actividad, iniciar un servicio o enviar un mensaje masivo.

Layouts

Un diseño de Android define todo lo que el usuario puede ver y tocar. Un diseño se compone de objetos View (como botones y texto) y ViewGroup (como listas, tablas o más Vistas), todos combinados para crear una Jerarquía de Vista. Deberás crear tu diseño declarando los elementos de IU en XML.

Volley

Volley es una biblioteca HTTP que hace que las redes para las aplicaciones de Android sean más fáciles y, lo que es más importante, más rápidas de usar.

Volley ofrece los siguientes beneficios: Programación automática de solicitudes de red, Múltiples conexiones de red simultáneas, Almacenamiento en memoria caché transparente de memoria y disco con coherencia de caché HTTP estándar.

Indicaciones Generales

Entregables

- Recuerda que la tarea es grupal (máximo 3 personas y mínimo 2), y que el hecho de compartir código entre grupos no está permitido y es sancionado.
- La entrega de esta tarea es a través de los siguientes enlaces:
  - WebApp: https://classroom.github.com/g/bPa1DRgp
  - MobileApp: https://classroom.github.com/g/MVWkScN-
- Las fechas de entrega y de presentación de cada hito será en la segunda sesión de la semana 10 y 15, respectivamente.
- Revisa bien lo que entregas, después de la fecha de entrega no podrás hacer modificaciones.

Presentación

- Por cada hito tu equipo hará una presentación de 10 minutos de duración y 5 minutos más para responder preguntas.

- La presentación deberá tener la siguiente estructura:
  1. Presentación: Nombre del proyecto
  2. Integrantes: relación de integrantes del equipo y el porcentaje (%) de participación de cada uno de ellos en cada entrega.
  3. Introducción: Explicación clara del problema, necesidad y solución.
  4. Contenido:
     (a) Demostración **en vivo** de las principales funcionalidades de la aplicación.
     (b) Explicación del diseño de la solución y de las partes más importantes del código.
  5. Conclusiones y aprendizaje: Es importante enumerar las conclusiones más importantes y darte el tiempor de compartirlas.
  6. Preguntas