Proyecto App De Eventos

# Idea De Proyecto:

Crear una aplicación móvil multiplataforma (Android y iOS) a modo de plantilla que permita consultar, gestionar y manejar la información pertinente de un evento cualquiera permitiendo que la empresa Dynamik Collective Creative Group S.A.S pueda vender fácilmente versiones de la App para eventos diferentes a un precio cómodo para sus clientes.

# Justificación:

La empresa Dynamik Creative Collective Group S.A.S cuenta con clientes interesados en aplicaciones para eventos, desafortunadamente estos clientes optan por no asumir el proyecto debido a que el costo de desarrollo de una aplicación móvil excede los COP $6000000 que es un monto fuera de presupuesto para estas cotizaciones.

Dentro de la empresa, se propone no perder a los interesados del proyecto y formular una aplicación que no sea tan costosa para el cliente, sustentable para la empresa, fácilmente replicable y que al mismo tiempo no pierda ningún estándar de calidad como los que tiene una aplicación promedio.

# Alcance del Proyecto:

El proyecto se desarrollará en su totalidad; en un primer momento, se hace una descripción de su diseño y requisitos en este documento y se hará una implementación para Android y iOS de la especificación respectiva. Se busca en este primer lanzamiento mostrar el funcionamiento y propiedades que la aplicación ofrece contando con datos de un Evento ficticio para proceder al acercamiento con posibles clientes quienes comprarían la suscripción a la aplicación.

# Objetivo Principal del Proyecto:

Crear una oportunidad de negocio basada en la replicación de aplicaciones móviles para eventos que puedan ser vendidas por suscripción y a bajo costo aprovechando el interés de diversos clientes de la empresa Dynamik Creative Collective Group S.A.S.

# Objetivos Secundarios del Proyecto:

* Desarrollar una aplicación a modo de plantilla que permita visualizar, consultar y manejar los datos de un evento dado.
* Utilizar desarrollo móvil multiplataforma con Xamarin y una librería de componentes compartidos que funcione como modelo de datos genérico.
* Cargar la aplicación con datos de un evento ficticio para poder hacer una prueba piloto que permita venderla a los posibles clientes del producto.

# Requisitos:

## **6.1 Especificación en lenguaje natural:**

### **6.1.1 Requisitos Funcionales:**

* La aplicación debe permitir la consulta del lugar donde está ubicado el evento y los recintos de cada charla.
* La aplicación debe permitir la consulta de la información general del evento.
* La aplicación debe dar información de cada charla del evento así como de su expositor, el lugar, la fecha, hora y año donde se dará la charla.
* La aplicación debe permitir desplegar información importante sobre el expositor como su trabajo previo y su perfil.
* La aplicación debe permitir mostrar contenido fotográfico de: lugar, expositores, evento, recintos y trabajos del expositor.
* La aplicación debe dar acceso a preguntas frecuentes sobre el evento y sobre el uso de la interfaz de la App.
* La aplicación debe mostrar información de contacto y sobre los organizadores del evento.

### **6.1.2 Requisitos No Funcionales:**

### **6.1.2.1 Requisitos de Producto:**

#### 6.1.2.1.1 Requisitos de Usabilidad:

* La aplicación cuenta con interfaz gráfica que permita la interacción con el usuario tanto en Android como en iOS.
* La interfaz gráfica utilizará el lenguaje de diseño ‘Material design’ para Android y los estándares actuales de diseño para iOS.
* La interfaz gráfica debe ser sencilla de usar y llamativa en cuanto a color, iconografía, tipografía y diagramación de los elementos.

#### 6.1.2.1.2 Requisitos de Eficiencia:

#### 6.1.2.1.2.1 Requisitos de Rendimiento:

* La aplicación debe correr en dispositivos móviles capaces de utilizar android en su versión KitKat (4.4) y iOS 7.1.2 para dispositivos Apple.
* La aplicación no requiere de procesamiento intensivo de datos por lo que un teléfono de gama media/baja debería ser capaz de correrla adecuadamente.
* La aplicación debe tener tiempos de carga bajos para asegurar una usabilidad adecuada.
* La aplicación requiere de conexión a datos o red WiFi para asegurar la carga de los datos.

#### 6.1.2.1.2.2 Requisitos de Espacio:

* Se utilizará un CMS para el almacenamiento de los datos e imágenes, además, este CMS debe permitir la exportación del modelo de datos a un archivo JSON que será cargado desde el teléfono.
* Datos como imágenes, datos del evento e información relevante de la app no serán colocados dentro del paquete de la aplicación y estarán disponibles siempre en línea para evitar el consumo excesivo de memoria.
* La aplicación debe ocupar menos de 100 Mb y su archivo de instalación menos de 22 Mb para asegurar los recursos en la gran mayoría de dispositivos y asegurar recursos para sus posibles actualizaciones.

#### 6.1.2.1.3 Requisitos de Fiabilidad:

* Se harán pruebas de usabilidad para verificar que la interfaz gráfica propuesta funcione para el público objetivo.
* Se desarrollará la aplicación por prototipos y cada uno se creará de manera incremental buscando tener en cada fase la aprobación con el equipo de Dynamik Creative Collective Group S.A.S y de ser posible de los clientes posibles de la aplicación final.

#### 6.1.2.1.4 Requisitos de Portabilidad:

* La aplicación se desplegará en plataformas Android y iOS, no se prevé diseño e implementación para Windows Phone en este primer lanzamiento.

### **6.1.2.2 Requisitos Organizacionales:**

#### 6.1.2.2.1 Requisitos de entrega:

* Para inicios del mes de Septiembre de 2016 se debe entregar la documentación completa del proyecto, así como el diseño de sistema y su especificación.
* El prototipo piloto de proyecto está previsto para el dìa Viernes 30 de Septiembre, con avances en el transcurso de dicho mes.

#### 6.1.2.2.2 Requisitos de implementación:

* Se utilizará la plataforma .NET para desarrollar aplicaciones compatibles con Android y iOS.
* Se utilizará Xamarin para asegurar la compatibilidad de la aplicación con Android y iOS.
* Se utilizará y elaborará un diseño orientado a objetos para implementar una librería de componentes compartidos (PCL) escrita con C# capaz de incorporar el modelo de datos, así como la lectura de estos desde el servicio en línea y que pueda ser utilizada para las aplicaciones Android y iOS simultáneamente.
* Se utilizará un CMS para el almacenamiento de datos e imágenes dentro de la aplicación y se espera la exportación de un archivo formato JSON compatible con el modelo de datos descrito y propuesto para el desarrollo de la aplicación.

#### 6.1.2.2.3 Requisitos de estándares:

* Se utilizarán estándares convencionales de programación orientada objetos para la escritura de variables, propiedades, métodos, clases y atributos.
* El diseño del sistema se hará en inglés para asegurar una implementación ordenada de las clases, métodos requeridos y para poder utilizarlos para explicaciones ya sea a un cliente de habla inglesa o a algún miembro del equipo que no hable español.
* La resolución mínima de las imágenes para cargar a la galería de imágenes es de 1280 x 720 píxeles, en formato .png o .jpg
* La resolución de las fotografías de expositores ha de ser cuadrada es decir ‘nxn’ de un tamaño mínimo de 500 x 500 píxeles.

### **6.1.2.3 Requisitos Externos:**

#### 6.1.2.3.1 Requisitos de Interoperabilidad:

* La librería de componentes compartidos (PCL) será una y solo una para todas las aplicaciones replicadas del proyecto sean estas para Android y iOS.
* La aplicación debe permitir la lectura de información sobre un evento desde un servicio en línea.
* La aplicación y la PCL se desacoplan y funcionan como componentes separados asegurando facilidad para obtener distintas versiones de la aplicación y facilidad de mantenimiento.

#### 6.1.2.3.2 Requisitos Éticos:

* El contenido presentado en pantalla se hará en lenguaje común y corriente y el contenido desplegado no hiere la susceptibilidad de los usuarios objetivo.

#### 6.1.2.3.3 Requisitos Legales:

* Se utilizará Visual Studio Community 2015 con Xamarin, ambos tienen licencia libre siempre que la compañía no pase de tener 5 desarrolladores .NET o que utilicen tecnologías de Microsoft para creación de software.
* El diseño gráfico también es desarrollado por Dynamik Creative Collectiva S.A.S.
* Las imágenes y fotografías tienen copyright de la empresa que las suministra para crear cada una de las réplicas.

#### 6.1.2.3.3.1 Requisitos de seguridad:

* La seguridad de los datos sobre los eventos estará dada por la plataforma CMS que se adquiera, además, el llamado a un archivo JSON permite el encapsulamiento de la información gracias a datos de autenticación dentro de la plataforma.

#### 6.1.2.3.3.1 Requisitos de privacidad:

* No se requerirá login de los usuarios de la aplicación, ni información personal o de red para acceder a la App por lo que se asegura privacidad para los usuarios.

### **6.1.2.4 Otros Requisitos de Cálidad:**

#### 6.1.2.4.1 Requisitos de Internacionalización:

* La aplicación se desarrollará en lenguaje español; iniciando el propósito del proyecto (Generar la idea de negocio respectiva) desde un Dominio Local.
* La aplicación y sus réplicas se distribuirán por Google Play Store y App Store de Apple respectivamente, con excepción de la prueba piloto que sería llevada hasta el cliente de producto mediante una reunión.

#### 6.1.2.4.2 Requisitos de Configuración:

* Se tendrá en cuenta para el futuro extender la aplicación a una plataforma mixta con servicios WEB que permitan una experiencia personalizada y subida de la información y registro.

## **6.2 Especificación mediante casos de uso:**

A continuación se presenta el diagrama de casos de uso *[ver gráfico 1]* y la especificación en tablas de cada uno de ellos *[ver sección 6.2.2]*:

### **6.2.1 Diagrama de casos de uso:**

Casos_De_Uso (1)

*Gráfico 1: Diagrama de casos de uso, muestra las relaciones entre los casos de uso como pre-condiciones y post-condiciones, además, muestra a medida de caso de uso de relación, el enlace entre la aplicación y el servicio de datos.*

### **6.2.2 Tablas de casos de uso:**

|  |  |
| --- | --- |
| **Nombre:** | Show Menu |
| **Actor:** | Usuario |
| **Objetivo:** | Inicializar la app y mostrar el menú |
| **Disparador:** | Iniciar la aplicación |
| **Pre-Condición:** | N/A |
| **Post-Condición:** | El menú carga correctamente y el usuario puede interactuar con él. |
| **Flujo Básico:** | 1. El usuario inicia la app 2. La app hace su proceso de carga y muestra el menú |
| **Flujo Alternativo:** | N/A |
| **Excepciones:** | N/A |
| ***Calidades:*** | N/A |

|  |  |
| --- | --- |
| **Nombre:** | Data Instantiation |
| **Actor:** | Data Service |
| **Objetivo:** | Cargar datos desde un pedido a Data Service de la PCL |
| **Disparador:** | Un caso de uso en específico requiere de datos. La app requiere inicializar instancias de objetos con datos. |
| **Pre-Condición:** | Data Load, Map Info ***OR*** Event Info ***OR*** Presenters Info ***OR*** Calendar ***OR*** Image Gallery ***OR*** Frequent Questions |
| **Post-Condición:** | Los datos cargan correctamente en los objetos para cada flujo de uso. |
| **Flujo Básico:** | 1. La app hace una instancia de data service. 2. Data service carga los datos *[ver: Caso de uso DataLoad]*. 3. La app instancia sus objetos para la pantalla dada con la información requerida. |
| **Flujo Alternativo:** | 3a) Ocurre excepción 1:  La app instancia los objetos con datos por defecto y muestra un error al usuario, luego, reintenta hacer la lectura de datos, de no ser posible se cargan datos por defecto y la app procede a mostrarlos.  3b) Ocurre excepción 2:  La app intenta cargar de nuevo la instancia de Data Service, si esta es nula, ocurre excepción 1. |
| **Excepciones:** | 1. Data service envía una instancia nula. 2. La instancia del objeto dentro de la App es nula. |
| ***Calidades:*** | *El tiempo de carga de los datos no debe ser muy alto, lo mismo el consumo.* |

|  |  |
| --- | --- |
| **Nombre:** | Data Load |
| **Actor:** | CMS Service, Data Service |
| **Objetivo:** | Cargar los datos desde un servicio CMS |
| **Disparador:** | Se hace un llamado desde el repositorio de datos |
| **Pre-Condición:** | N/A |
| **Post-Condición:** | Los datos son correctamente cargados y la instancia de DataService los contiene. |
| **Flujo Básico:** | 1. Data Service hace un llamado al servicio CMS. 2. El servicio CMS abre el archivo JSON de datos. 3. Data Service carga y lee los datos del archivo JSON y se instancia correctamente. |
| **Flujo Alternativo:** | 1a) Ocurre excepción 1: DataService se instancia con datos por defecto y marca un error para ser desplegado en la app solicitando conexión a internet para hacer un reintento de carga de datos.  3a) Ocurre excepción 2:  Se hace un reintento de iniciar el caso de uso desde el paso 1.  3b) Flujo alternativo 3a Falló:  Se instancia DataService con datos por defecto y marca un error para ser desplegado en la app. |
| **Excepciones:** | 1. No hay conexión a datos o internet del dispositivo. 2. Los datos no se pueden leer correctamente. |
| ***Calidades:*** | N/A |

|  |  |
| --- | --- |
| **Nombre:** | Map Info |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar el mapa con las locaciones del evento |
| **Disparador:** | Hacer tap en ‘Lugar’ |
| **Pre-Condición:** | *Show Menu* |
| **Post-Condición:** | N/A |
| **Flujo Básico:** | 1. El usuario entra a la sección ‘mapa’. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos de locaciones del evento. 3. La app contacta el servicio google Maps y muestra las localizaciones en pantalla. 4. El usuario presiona en una ubicación de las posibles del mapa. 5. La app muestra información adicional sobre esa ubicación. |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app contacta el servicio de Google Maps, Muestra un error y una localización por defecto en la ciudad, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Show Menu*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan los datos por defecto para la/s localización/es del evento. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | *El usuario puede reportar si hay información que no existe o que no se presenta de manera adecuada en el sistema.* |

|  |  |
| --- | --- |
| **Nombre:** | Event Info |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar Información general del evento |
| **Disparador:** | Hacer tap en ‘información’ |
| **Pre-Condición:** | *Show Menu* |
| **Post-Condición:** | N/A |
| **Flujo Básico:** | 1. El usuario entra a la sección ‘información’. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos de información del evento. 3. La app muestra la información del evento. |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Show Menu*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | *El usuario puede reportar si hay información que no existe o que no se presenta de manera adecuada en el sistema.* |

|  |  |
| --- | --- |
| **Nombre:** | Presenters Info |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar menú de presentadores/expositores del evento |
| **Disparador:** | Hacer tap en ‘Expositores’ |
| **Pre-Condición:** | *Show Menu* |
| **Post-Condición:** | Presenter |
| **Flujo Básico:** | 1. El usuario entra a la sección ‘expositores’. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos del menú de presentadores. 3. La app muestra el menú de presentadores. 4. El usuario selecciona un presentador de interés dentro del menú. |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Show Menu*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | *El usuario puede reportar si hay información que no existe o que no se presenta de manera adecuada en el sistema.* |

|  |  |
| --- | --- |
| **Nombre:** | Presenter |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar información de un presentador/expositor en particular. |
| **Disparador:** | Escoger un presentador del menú de presentadores |
| **Pre-Condición:** | *Presenters Info* |
| **Post-Condición:** | N/A |
| **Flujo Básico:** | 1. El usuario ingresa a la información en particular de un expositor. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos del presentador en particular. 3. La app muestra la información particular del presentador. |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Presenters Info*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | *El usuario puede reportar si hay información que no existe o que no se presenta de manera adecuada en el sistema.* |

|  |  |
| --- | --- |
| **Nombre:** | Calendar |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar menú de calendario |
| **Disparador:** | Hacer tap en ‘Calendario’ |
| **Pre-Condición:** | *Show Menu* |
| **Post-Condición:** | Day Detail View |
| **Flujo Básico:** | 1. El usuario entra a la sección ‘Calendario’. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos del menú del calendario. 3. La app muestra el menú del calendario. 4. El usuario selecciona una fecha de interés dentro del menú. |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Show Menu*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | *El usuario puede reportar si hay información que no existe o que no se presenta de manera adecuada en el sistema.* |

|  |  |
| --- | --- |
| **Nombre:** | Day Detail View |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar información de un día del calendario en particular |
| **Disparador:** | Escoger un día en particular del calendario del menú |
| **Pre-Condición:** | *Calendar* |
| **Post-Condición:** | N/A |
| **Flujo Básico:** | 1. El usuario ingresa a la información en de un día del calendario. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos del día en particular. 3. La app muestra la información solicitada |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Calendar*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | *El usuario puede reportar si hay información que no existe o que no se presenta de manera adecuada en el sistema.* |

|  |  |
| --- | --- |
| **Nombre:** | Image Gallery |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar galería de imágenes |
| **Disparador:** | Hacer tap en ‘Fotos’ |
| **Pre-Condición:** | *Show Menu* |
| **Post-Condición:** | Image Zoom |
| **Flujo Básico:** | 1. El usuario entra a la sección ‘Fotos’. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos del menú de Fotos. 3. La app muestra la galería de fotos. 4. El usuario selecciona una foto de interés dentro del menú. |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Show Menu*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | *El usuario puede reportar si hay información que no existe o que no se presenta de manera adecuada en el sistema.* |

|  |  |
| --- | --- |
| **Nombre:** | Image Zoom |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar una imagen agrandada |
| **Disparador:** | Escoger una imagen de la galería |
| **Pre-Condición:** | *Image Gallery* |
| **Post-Condición:** | N/A |
| **Flujo Básico:** | 1. El usuario entra a una imagen en particular de la galería. 2. La imagen se muestra agrandada y en tamaño de pantalla completa dentro del teléfono. |
| **Flujo Alternativo:** | 2a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  2b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Image Gallery*  2c) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*.  2d) La imagen no carga:  La app intenta nuevamente cargar la imagen solicitada, de no ser posible se mostrará un mensaje de error al usuario y se devolverá al caso de uso *Image Gallery.* |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | *El usuario puede reportar si hay una imagen rota.* |

|  |  |
| --- | --- |
| **Nombre:** | Frequent Questions |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar las preguntas frecuentes. |
| **Disparador:** | Hacer tap en ‘Preguntas Frecuentes’ |
| **Pre-Condición:** | *Show Menu* |
| **Post-Condición:** | N/A |
| **Flujo Básico:** | 1. El usuario entra a la sección ‘Preguntas Frecuentes’. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos de preguntas frecuentes. 3. La app muestra las preguntas frecuentes. |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Show Menu*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | N/A |

|  |  |
| --- | --- |
| **Nombre:** | Contact Page |
| **Actor:** | Usuario |
| **Objetivo:** | Mostrar la Información de contacto del evento. |
| **Disparador:** | Hacer tap en ‘Contacto’ |
| **Pre-Condición:** | *Show Menu* |
| **Post-Condición:** | N/A |
| **Flujo Básico:** | 1. El usuario entra a la sección ‘Contacto’. 2. Se inicia caso de uso, DataInstantiation para solicitar los datos de contacto. 3. La app muestra la información de contacto para el evento. |
| **Flujo Alternativo:** | 3a) Ocurrió excepción 1:  La app muestra un mensaje de error, se devuelve a *Show Menu*.  3b) El usuario presiona el botón [<-] del dispositivo:  Se ejecuta el caso de uso *Show Menu*  2a) Ocurrió excepción 2:  La app muestra un mensaje de error al usuario y finaliza el caso de uso devolviendo a *Show Menu*. |
| **Excepciones:** | 1. Error fatal en recepción o lectura de datos: Se utilizan datos por defecto. 2. No hay conexión de datos o WiFi. |
| ***Calidades:*** | N/A |

# Diseño de sistemas:

## **7.1 Arquitectura del sistema:**

A continuación, se presenta el diagrama de paquetes con la arquitectura definida para el desarrollo de la aplicación *[ver: gráfico 2]*:

Arquitectura (1)

*Gráfico 2: Diagrama de despliegue, Muestra la arquitectura de manera explícita*

Como se puede observar en el gráfico anterior, la arquitectura definida para el sistema es por capas, habiendo dos capas principales que son la aplicación y el manejo de los datos que permiten desacoplar ambas partes para lograr la escalabilidad requerida por el jefe de proyecto.

A nivel de granularidad, la arquitectura también permite el desacople interno dentro de cada una de las capas, es decir, cada sección del producto está separada lo que facilita cuestiones de mantenimiento y replicación de la aplicación.

Las réplicas de la aplicación toman la implementación completa de la capa inferior del esquema, además esta capa inferior está diseñada de tal manera que se pueda asegurar la escalabilidad del producto mediante la añadidura de nuevos campos de datos para ser mostrados y presentados al usuario en la aplicación.

El desacople de la app y la inclusión permanente de conexión a datos para alimentar la capa de datos hace que el rendimiento sea reducido, además, este recurso queda supeditado a las condiciones de descarga de datos que tenga el usuario final y a nivel de desarrollo, implementar procesos de optimización para el manejo de datos desde servicios en línea para asegurar que el rendimiento de la aplicación sea óptimo.

Project_Packages

*Gráfico 3: Diagrama de paquetes, muestra dos sistemas donde uno representa la aplicación y el otro la librería de componentes compartidos para proveer el servicio de datos, así como las relaciones de importación de elementos de la PCL a la App.*

En el gráfico 3, se presentan los paquetes de cada capa principal de la aplicación y las dependencias internas de la capa de aplicación a la de datos; la PCL se define como la librería de componentes compartidos que traducido al esquema presentado en el gráfico 2 [ver: gráfico 2], representaría la capa *Data Layer,* Los paquetes *Model, DataService* y *Repository* permiten implementar las propiedades de esta capa descritas anteriormente.

Del paquete *App* se puede establecer la equivalencia con la capa superior del sistema que en este caso sería *Application Layer,* los paquetes presentados en este diagrama permiten hacer la implementación de la interfaz de usuario y la aplicación para sistema operativo Android [es decir: *fragments, activities, adapters, utilities, views*], en el caso de iOS la implementación de la capa se hace de manera diferente, pero, la arquitectura de sistema se conserva para la aplicación de prueba piloto y sus réplicas indiferentemente del sistema operativo de despliegue.

## **7.2 Diseño Orientado a Objetos:**

### **7.1.1 Modelo de datos y clases de la PCL:**

A continuación se presentan los diseños para el modelo de datos (*Model* dentro de gráfico 3 [ver: Gráfico 3]), los métodos de servicio y el repositorio de datos.



*Gráfico 4: Diagrama de clases, muestra el modelo de datos propuesto para el sistema contando con una clase principal llamada evento, y sus clases que lo componen con sus respectivas dependencias, composiciones y agregaciones (Se utiliza la palabra MainEvent en vez de Event debido a que Event es una palabra reservada en el lenguaje de programación).*

Como se puede observar en el gráfico 4 *[ver: Gráfico 4],* El modelo de datos incluye clases para todo tipo de dato que puede ser necesario en un evento cualquiera, además, de ser necesario, este modelo permite añadir clases adicionales para eventos o casos que lo requieran.

Actualizar el modelo de datos en una situación en la que aplicaciones y réplicas hayan sido previamente creadas, haría que se tuvieran que modificar todos los archivos JSON de dichas aplicaciones para que sean compatibles con el modelo nuevo para generar un repositorio de datos adecuado, en este caso, modelado como se presenta a continuación en el gráfico 5:



*Gráfico 5: Clase Repository, con un atributo y métodos para Query de datos sobre el atributo en situaciones particulares.*

La clase *Repository,* cuenta con solo un atributo llamado *event,* este atributo se llena con los datos leídos desde el servicio externo CMS siendo él, el repositorio; los métodos que componen esta clase permiten hacer queries a distintos datos en situaciones que pueden ser necesarias dentro de la aplicación, como llamar un dato en particular por ID, encontrar una lista dado un presentador en particular y encontrar una conferencia dado un lugar o un día.

Para asegurar el encapsulamiento de los datos en Repository, se modela la clase *DataService* que accede únicamente a los métodos de Repository, así:



*Gráfico 6: Clase DataService, con un llamado al repositorio y CallBacks a los métodos implementados en Repository* para su uso desde la capa de aplicación.