



PERÍODO 2021



TRABAJO AUTÓNOMO 5

**DESARROLLO DE UNA APLICACIÓN MÓVIL
USANDO COMPONENTES AVANZADOS Y
REPOSITORIOS DE GITHUB**

GRUPO#
APELLIDOS1_APELLIDOS2_APELLIDOS3_APELLIDOS4

ELABORADO POR: ADRIANA COLLAGUAZO JARAMILLO
MATERIA: PROGRAMACIÓN DE SISTEMAS TELEMÁTICOS
CARRERA DE INGENIERÍA EN MECATRÓNICA
FIEC-ESPOL

Trabajo Autónomo:	Unidad 3 Control remoto de sistemas telemáticos
Objetivo de Aprendizaje:	Desarrollar interfaces de usuario en entornos de desarrollo web y móvil para el control remoto de los sistemas telemáticos.
Recursos:	Android Studio, GIT (software). Github (online).
Duración:	8 horas

INSTRUCCIONES

- El formato del trabajo tiene habilitado recuadros para que llenen las respuestas.
- Los trabajos se reciben hasta la fecha planificada en el SIDWeb.
- Coloque el nombre del archivo así "PST_TAA_GrupoB_Apellido1_ApellidoN", siendo A el número del trabajo, B el número del grupo, N el último apellido del integrante del grupo.
- Una vez que haya desarrollado el trabajo, cada integrante del grupo contestará la encuesta de evaluación de los trabajos autónomos ingresando al enlace. → <https://forms.gle/d2p61cJrjd2ZFgsu8>

INTRODUCCIÓN

Android Studio es un IDE, es decir, un entorno de desarrollo integrado oficial para el desarrollo apps para el SO Android basado en IntelliJ IDEA. Entre sus funciones destacan la compilación flexible basada en Gradle; emulación rápida y cargado de funciones; proporciona un entorno unificado de desarrollo para cualquier versión de Android; es compatible con C++, NDK, Google Cloud Platform y permite la generación de APK.¹

Este IDE puede además integrarse con GitHub, el cual es una plataforma de desarrollo colaborativo de software para alojar proyectos utilizando el sistema de control de versiones Git. Esta plataforma es usada por desarrolladores a nivel global para facilitar el trabajo en equipo a través de sus opciones de clonación, merge, etc. Y al mismo tiempo compartir de manera libre su trabajo contribuyendo a la mejora continua de software.²

Desarrolle un aplicativo móvil usando componentes avanzados como menú, y cargue el código fuente en un repositorio de GitHub.

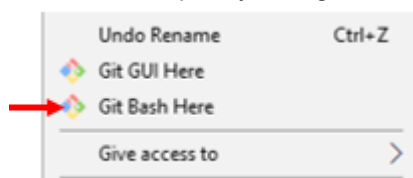
ACTIVIDADES

Paso 0: Creación de cuenta en Github.

1. Crearse una cuenta en <https://github.com>.
2. Instalar Git Bash en el enlace <https://git-scm.com/downloads>. En caso de que no se disponga de GIT CLI (git bash), también puede utilizar CMD de Windows/Ubuntu.

Paso 1: Crear un repositorio.

1. Creamos un proyecto de Android Studio, el cual vamos a alojar en nuestro repositorio de github.
2. Dentro de la carpeta del proyecto, abra la línea de comandos de GIT (GIT CLI). Podemos encontrarlo, dando clic derecho dentro de la carpeta y escogemos la opción "GIT BASH HERE".



¹ Android Studio <https://developer.android.com/studio/intro?hl=es>

² GitHub <https://github.com>

- Para probar que GIT ha sido instalado correctamente, utilice el comando "git --version".

```
Microsoft Windows [Versión 6.3.9600]
(c) 2013 Microsoft Corporation. Todos los derechos reservados.

C:\Users\acollag>git --version
git version 2.13.3.windows.1
```

GIT en línea de comandos de Windows

```
acollag@GGIECDOCWRK054 MINGW64 ~/AndroidStudioProjects/PST_TA5_G# (master)
$ git --version
git version 2.13.3.windows.1
```

GIT BASH propia

- Para crear un nuevo repositorio, utilice el siguiente comando "git init".

```
acollag@GGIECDOCWRK054 MINGW64 ~/AndroidStudioProjects/PST_TA5_G# (master)
$ git init
Initialized empty Git repository in C:/Users/acollag/AndroidStudioProjects/PST_TA5_G#/.git/
```

Esto creará un archivo oculto [.git] para el manejo del repositorio y nos ubicará directamente en la rama "master"

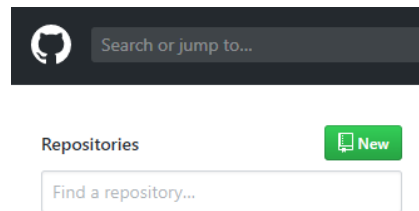
- Agregamos todos los archivos del proyecto a nuestro **repositorio local** con el comando: "git add --all".

```
acollag@GGIECDOCWRK054 MINGW64 ~/AndroidStudioProjects/PST_TA5_G# (master)
$ git add --all
warning: LF will be replaced by CRLF in gradle/wrapper/gradle-wrapper.properties.
The file will have its original line endings in your working directory.
warning: LF will be replaced by CRLF in gradlew.
The file will have its original line endings in your working directory.
```

- Ahora realizamos un **commit**, esto realizará nuestros cambios permanentes en el repositorio local. Pero debemos asignarle un mensaje [-m "mensaje"] para indicar los cambios que hemos realizado.

```
acollag@GGIECDOCWRK054 MINGW64 ~/AndroidStudioProjects/PST_TA5_G# (master)
$ git commit -m "Commit inicial"
[master (root-commit) 57bedc8] Commit inicial
29 files changed, 478 insertions(+)
create mode 100644 .gitignore
create mode 100644 app/.gitignore
create mode 100644 app/build.gradle
create mode 100644 app/proguard-rules.pro
create mode 100644 app/src/androidTest/java/com/example/acollag/pst_ta5_g/ExampleInstrumentedTest.java
create mode 100644 app/src/main/AndroidManifest.xml
create mode 100644 app/src/main/java/com/example/acollag/pst_ta5_g/MainActivity.java
create mode 100644 app/src/main/res/layout/activity_main.xml
create mode 100644 app/src/main/res/mipmap-hdpi/ic_launcher.png
create mode 100644 app/src/main/res/mipmap-hdpi/ic_launcher_round.png
create mode 100644 app/src/main/res/mipmap-mdpi/ic_launcher.png
create mode 100644 app/src/main/res/mipmap-mdpi/ic_launcher_round.png
create mode 100644 app/src/main/res/mipmap-xhdpi/ic_launcher.png
create mode 100644 app/src/main/res/mipmap-xhdpi/ic_launcher_round.png
create mode 100644 app/src/main/res/mipmap-xxhdpi/ic_launcher.png
create mode 100644 app/src/main/res/mipmap-xxhdpi/ic_launcher_round.png
create mode 100644 app/src/main/res/values/colors.xml
create mode 100644 app/src/main/res/values/strings.xml
create mode 100644 app/src/main/res/values/styles.xml
create mode 100644 app/src/test/java/com/example/acollag/pst_ta5_g/ExampleUnitTest.java
create mode 100644 build.gradle
create mode 100644 gradle.properties
create mode 100644 gradle/wrapper/gradle-wrapper.jar
create mode 100644 gradle/wrapper/gradle-wrapper.properties
create mode 100644 gradlew
create mode 100644 gradlew.bat
create mode 100644 settings.gradle
```

6. Creamos un repositorio en línea. Ahora usaremos Github (Requerira una cuenta gratuita). *Del lado superior izquierdo, encontrara el botón “NEW”.*



7. La información requerida para crear un repositorio, se muestra a continuación:

Nombre del repositorio	El nombre de nuestro repositorio como será publicado en línea
Descripción (Opcional)	Descripción sobre lo que realiza nuestro proyecto
Tipo	Público o privado (para saber si es visible en línea)
Archivo Readme	Archivo inicial del repositorio. Agregamos indicaciones a otros programadores
Agregar. gitignore	Archivo para seleccionar los archivos que no queremos subir a nuestro repositorio
Licencia	Tipo de licencia: OpenSouse, MIT, Apache, etc

Debe asignarle el siguiente nombre a su repositorio “PST_TA5_G#”, reemplazando el “#” por su número de grupo, por ejemplo: “PST_TA5_G1”.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner: acollaguazo / Repository name *: PST_TA5_G# ✓

Great repository names are Your new repository will be created as PST_TA5_G- but probable-succotash?

Description (optional): Este repositorio es creado para el desarrollo del proyecto de programación de sistemas telemáticos.

☒ Public: Anyone can see this repository. You choose who can commit.

☐ Private: You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☒ Initialize this repository with a README: This will let you immediately clone the repository to your computer.

Add .gitignore: None | Add a license: None ⓘ


Create repository

8. Una vez ingresados todos los campos, damos clic en “Create repository”.



Vista de mi repositorio vacío.

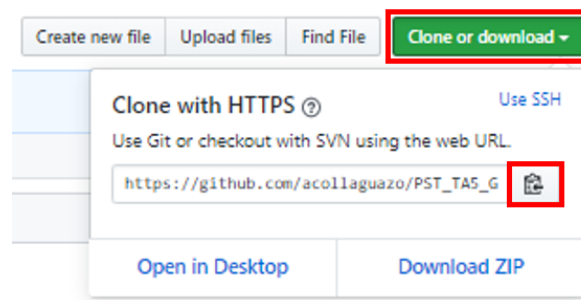
9. Damos clic en el botón verde “clone or download” donde estará visible el **URL** para su manejo y

presionamos en para  copiarlo.

Para obtener el repositorio en línea, obtenemos la rama de externa con el comando:

\$ git remote add origin [web URL del repositorio]

```
acollag@GGIECDOCWRK054 MINGW64 ~/AndroidStudioProjects/PST_TA5_G# (master)
$ git remote add origin https://github.com/acollaguazo/PST_TA5_G-.git
```



10. Ahora tendremos que obtener la rama y publicar los cambios:

\$ git pull origin master [obtiene la rama externa de Github (master)]

\$ git push origin master -f [publica el proyecto local (-f para forzar los cambios)]

11. Repetir los pasos 4, 5 para actualizar los cambios realizados y el comando git push origin master.

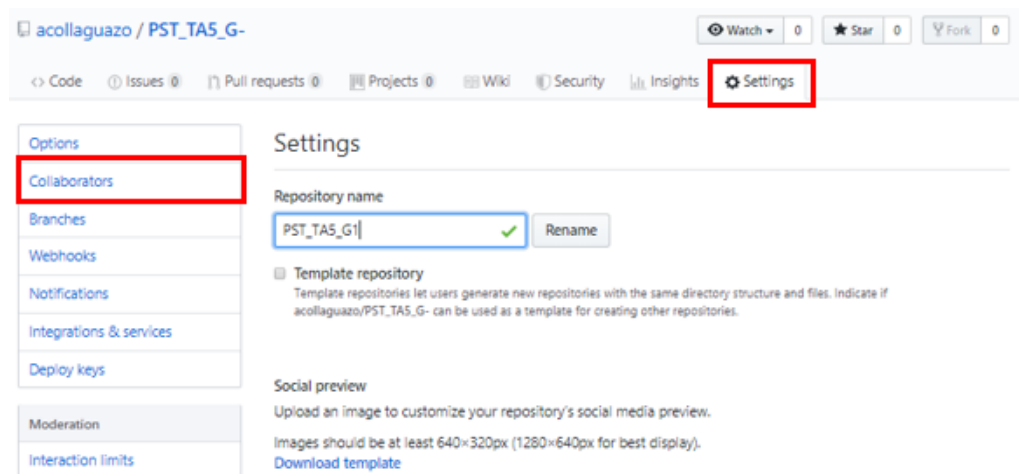
PREGUNTAS DE INVESTIGACIÓN:

1. ¿Qué otro tipo de servicios en línea (como GITHUB) existen?
2. ¿Para qué sirve el archivo .gitignore y como se utiliza?
3. ¿Qué limitaciones tiene GITHUB?

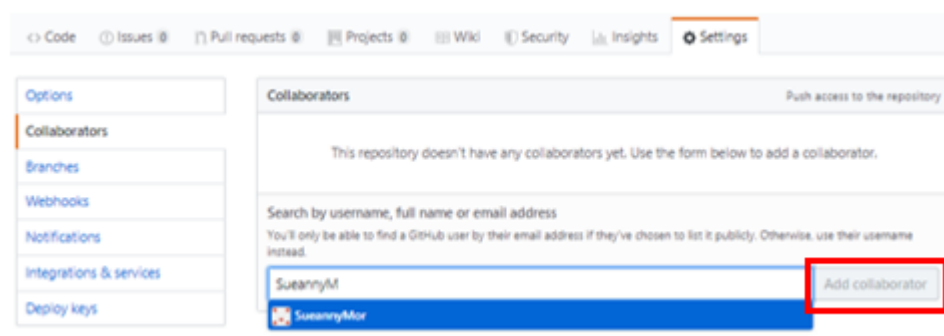
4. ¿Qué es una rama?
5. ¿Con cuál nombre, comúnmente, se le denomina a la rama principal?
6. ¿Cuál es el link de su repositorio?
7. ¿Cómo utilizo un repositorio público (utilizando el comando git clone)?

Paso 2: Invitar a otros miembros del grupo a mi proyecto (incluya a todos los miembros del grupo)

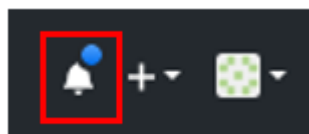
1. Para habilitar la modificación a otros miembros de mi grupo, debemos darle acceso. [Incluso si el proyecto es libre, solo pueden modificarlo quienes han sido invitados]. Para esto de clic en el tab "Settings/Configuración" > Collaborators / Colaboradores



2. Buscamos y agregamos a otros usuarios (usuario de github).



3. Una vez agregados, es necesario aceptar las invitaciones para poder realizar PUSH (cambios al repositorio). Puede revisar las invitaciones en la campana a lado del usuario.



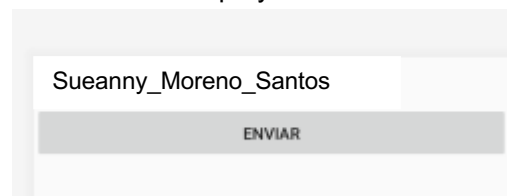
Paso 3: Crear una rama [branch] (Trabajo individual)

1. Tenemos el proyecto principal en master, cualquier otro cambio puede ser realizado sin dañar el proyecto principal. Utilizamos el comando:
git checkout -b "nombre_rama"

Para este taller, crearemos una rama de la siguiente forma: "nombre_apellido1_apellido2"

```
acollag@GGIECD0CWRK054 MINGW64 ~/AndroidStudioProjects/PST_TA5_G# (SueannyMor)
$ git checkout -b "Sueanny_Moreno_Santos"
Switched to a new branch 'Sueanny_Moreno_Santos'
```

2. Ahora realizamos algunos cambios en nuestro proyecto local.



3. Estos cambios son únicamente reflejados dentro de nuestra rama. Ahora subimos los cambios, para ello realizamos los siguientes comandos.

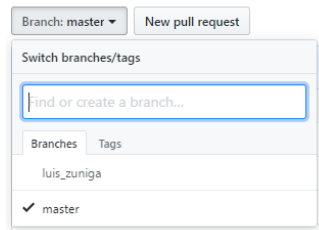
Git add --all	Agrega todos los cambios a nuestra rama.
Git commit -m "cambios a mi rama"	Agrega un commit en mi rama, indicando los cambios que realice.
Git push origin [nombre rama]	Subimos los cambios al repositorio (pero solo dentro de la página).

```
zuniga@DESKTOP-JDQFA20 MINGW64 ~/Desktop/AMST_3 (luis_zuniga)
$ git add --all

zuniga@DESKTOP-JDQFA20 MINGW64 ~/Desktop/AMST_3 (luis_zuniga)
$ git commit -m "cambios a mi rama"
[luis_zuniga 5cb9235] cambios a mi rama
8 files changed, 202 insertions(+), 101 deletions(-)
create mode 100644 .idea/vcs.xml
rewrite app/src/main/res/drawable/ic_launcher_background.xml (97%)
```

```
zuniga@DESKTOP-JDQFA20 MINGW64 ~/Desktop/AMST_3 (luis_zuniga)
$ git push origin luis_zuniga
Enumerating objects: 34, done.
Counting objects: 100% (34/34), done.
Delta compression using up to 2 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (18/18), 2.00 KiB | 78.00 KiB/s, done.
Total 18 (delta 9), reused 0 (delta 0)
```

4. Podemos revisar todas las ramas dentro de Github (así mismo podemos cambiar entre ramas para revisar diferentes versiones de código).

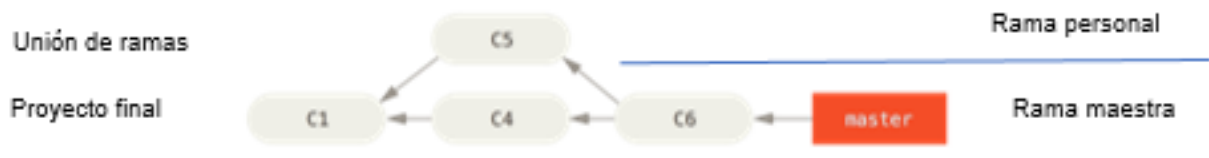


PREGUNTAS DE INVESTIGACIÓN

1. ¿Qué utilidades tiene el manejo de ramas?
2. ¿Qué tipos de conflictos puede ocurrir durante el manejo y creación de ramas?
3. ¿Cuál es la diferencia entre **chekout** y **checkout -b**?
4. ¿Qué es la herramienta **lint**?
5. Escriba al menos **5 buenas prácticas de estándares de programación** para creación de aplicaciones móviles en Android Studio/Java.
6. Investigue acerca de los tipos de aplicaciones móviles, su principal característica, las tecnologías con las que se programan y ejemplos de aplicaciones de cada tipo.

Paso 4: Unir ramas al proyecto principal [branch].

Las ramas funcionan como proyectos paralelos del proyecto principal, pero para avanzar con el proyecto es necesario unir las ramas una vez han sido probadas.



1. En caso de realizar algún cambio en el repositorio maestro.
 - a. Git fetch origin master (obtiene todos los cambios realizados en master)
2. Nos cambiamos a la rama principal.
 - a. Git checkout master
3. Traemos los cambios realizados en la rama única.
 - a. Git merge [nombre rama]


```

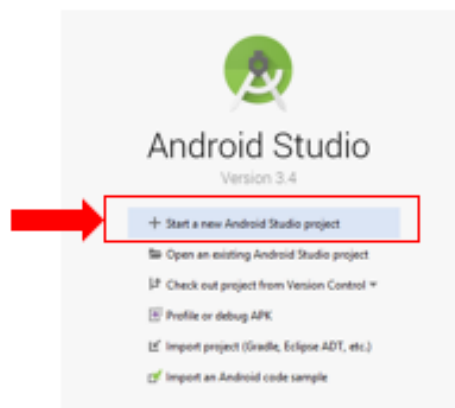
zuniga@DESKTOP-JDQFA20 MINGW64 ~/Desktop/AMST_3 (master)
$ git merge luis_zuniga
Updating 9430617..5cb9235
Fast-forward
 .idea/vcs.xml | 6 +
 app/src/main/AndroidManifest.xml | 6 +
 .../res/drawable-v24/ic_launcher_foreground.xml | 12 +-
 .../main/res/drawable/ic_launcher_background.xml | 236 ++++++-----
 app/src/main/res/layout/activity_main.xml | 23 +-
 app/src/main/res/mipmap-anydpi-v26/ic_launcher.xml | 4 +-
 .../res/mipmap-anydpi-v26/ic_launcher_round.xml | 4 +-
 gradle.properties | 4 -
 8 files changed, 198 insertions(+), 97 deletions(-)
 create mode 100644 .idea/vcs.xml

```

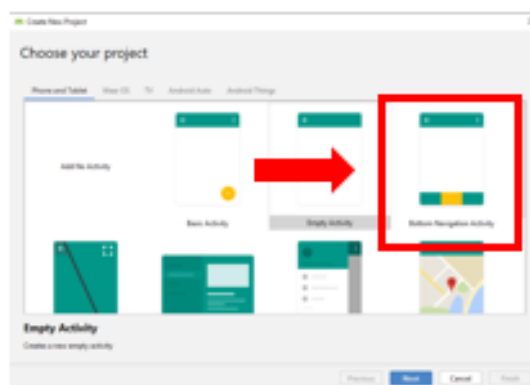
Nota: Esto indica los archivos que han sido modificados.

Paso 5: Crear un nuevo proyecto en Android Studio

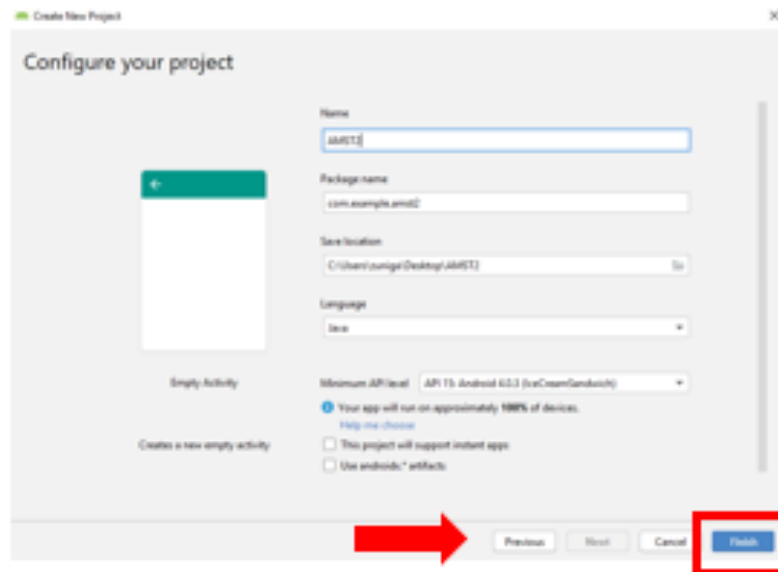
1. Al abrir Android Studio, podemos crear, abrir o importar proyectos. Seleccione “Start a new Android Studio project”.



2. Seleccionar el tipo de proyecto: Para esta práctica escogeremos la pestaña **Phone and Tablet > Empty Activity**. Otro tipo de actividades viene por defecto con componentes no necesarios para este taller.



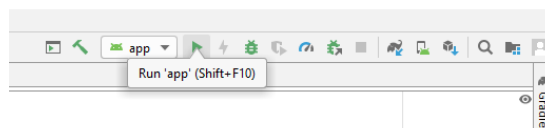
3. Configuración inicial del proyecto.
 - a. [Name]: Colocaremos el nombre de nuestra app. (Recuerde que este nombre será reflejado en el PlayStore al momento de publicarlo). Para este taller, usaremos **AMST[numeroGrupo]**.
 - b. [PackageName]: Paquete principal de código java, se obtiene automáticamente del nombre
 - c. [Save Location]: Dirección donde se ubica el proyecto en nuestra PC
 - d. [Language]: java
4. Seleccionamos **FINISH**



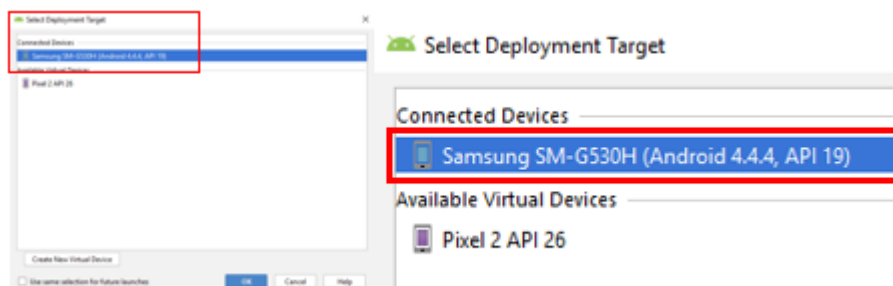
5. Como resultado se creará un proyecto vacío, solo presentado el mensaje “Hello World”

Ejecutar nuestra app en el teléfono

1. Del lado superior derecho de AndroidStudio aparecer la barra de ejecución (“RUN”).
2. Buscar el icono play para ejecutar nuestra app (o usar el atajo Shift+F10).



3. Ahora aparecerá la ventana “Select Deployment target” (Seleccionar dispositivo a ejecutar). Ahora seleccionamos en “**Connected devices**” > [modelo del teléfono conectado]:
Samsung SM G530




4. Esto creara una aplicación local en nuestro celular, y podemos probarla en vivo.



TAREAS DE DESAFÍO:

Una empresa dedicada a la venta de libros quiere mostrar su catálogo mediante una aplicación móvil (el modo de venta no se ha definido aún). Ha realizado un prototipo de baja resolución de lo que espera que tenga la aplicación.

1	2
	
<p>Ilustración 1 Splash art con el logo de la empresa</p>	<p>Ilustración 2 Pantalla de Inicio de sesión</p>
<p>Una pantalla inicial donde solo se presente el logo de la empresa.</p>	<p>Seguido, se presenta una pantalla para realizar inicio de sesión.</p>

3	4
	
<p>Ilustración 3 Ventana principal o home</p>	<p>Ilustración 4 Descripción del libro</p>
<p>Una vez que se ingresó se procede a mostrar la página principal que cuenta con una lista de todos los libros con los que se cuenta y un cuadro de búsqueda por nombre de libro.</p>	<p>Cuando se selecciona un ítem de la lista se abrirá un cuadro de diálogo personalizado indicando la sinopsis del libro en cuestión.</p>

5	6
 <p><i>Ilustración 5 Perfil de usuario</i></p>	 <p><i>Ilustración 6 Sección categorías de libros</i></p>
<p>Se cuenta con una sección perfil donde se detallan los datos del usuario.</p>	<p>En la sección de categorías se muestran las categorías de libros que existen en la empresa. Una vez que se seleccione una categoría, esta debe mostrar una lista de libros de solo esa categoría como la página de home.</p>

A usted y su equipo le han pedido que realice un prototipo de alta resolución para Android donde se implemente la idea del prototipo de baja.

Observaciones:

- La interfaz de usuario debe ser creativa e interactiva por cada equipo. No es necesario que sea una réplica del prototipo de baja resolución.
- La experiencia de usuario debe ser similar a la presentada: debe existir una forma de buscar libros, de categorizarlos, de presentar la sinopsis en un cuadro de dialogo personalizado y de hacer un login (inicio de sesion)
- No es necesario usar una base de datos interna, salvo que usted lo desee.
- No es necesario que ponga las mismas categorías de libros ni todas las existentes en el mercado. Con 3 o 4 categorías bastan. Así mismo con 3 o 4 libros por categoría es suficiente.

Entregables:

- El presente documento en Word con las actividades desarrolladas: preguntas de investigación, desarrollo de la aplicación donde se escribe las clases, activitys y archivos xml con una explicación corta acerca de lo que hace la activity (máximo de 3 líneas), captura de pantalla de las estadísticas de los commits realizados al proyecto (se encuentra en Insights>Contributors).
- Enlace del repositorio público de github, con el archivo ejecutable (.apk), y un video corto de ejecución de la aplicación.
- Los archivos (documento y apk) deben ser subidas por separado a Sidweb en la misma tarea no como un archivo comprimido.

RETO:

El equipo que presente el mejor prototipo de alta resolución puede recibir un punto extra. Se calificará en base a estos criterios:

- Interfaz de usuario
- Experiencia de usuario
- Código (buenas prácticas de programación)
- Elemento adicional que el equipo implemente con su prototipo de alta resolución.

DESARROLLO

LINK VÍDEO

ESTADÍSTICAS

REPOSITORIO GITHUB