GRUPO #8

NUÑEZ\_BERMEO\_BAQUERIZO\_LOOR\_GONZALEZ

TRABAJO AUTÓNOMO 5

desarrollo de UNA aplicación móvil USANDO componentes avanzados y REPOSITORIOS DE GITHUB

fkndsvd

PERÍODO 2020

ELABORADO POR: Adriana Collaguazo Jaramillo

MATERIA: PROGRAMACIÓN DE SISTEMAS TELEMÁTICOS

CARRERA DE INGENIERÍA EN MECATRÓNICA

FIEC-ESPOL



|  |  |
| --- | --- |
| **Trabajo Autónomo:** | **Capítulo 2** |
| **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 |

## INTRODUCCIÓN

Android Studio es un IDE, es decir, un entorno de desarrollo integrado oficial para el desarrollo apps para el SO Android basado en Intelli J 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]](#footnote-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]](#footnote-2)

## INSTRUCCIONES

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

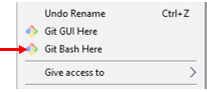
## ACTIVIDADES

**Paso 0: Creación de cuenta en Github.**

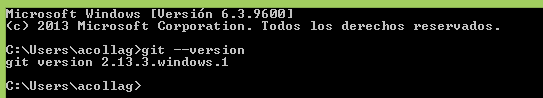
1. Crearse una cuenta en <https://github.com>**.**
2. Instalar Git Bash en el enlace <https://git-scm.com/downloads>. En caso de que no se disponga de GIT CLI (git bash), también puede utilizar CMD de Windows/Ubuntu.

**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”.



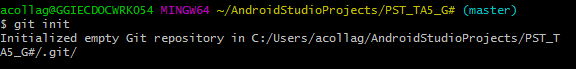
* *Para pobar que GIT ha sido instalado correctamente, utilice el comando “git --version”.*



GIT en línea de comandos de Windows  

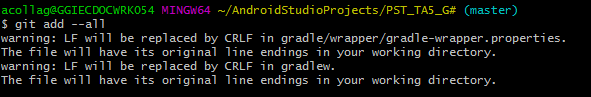

GIT BASH propia

1. Para crear un nuevo repositorio, utilice el siguiente comando “git init”.

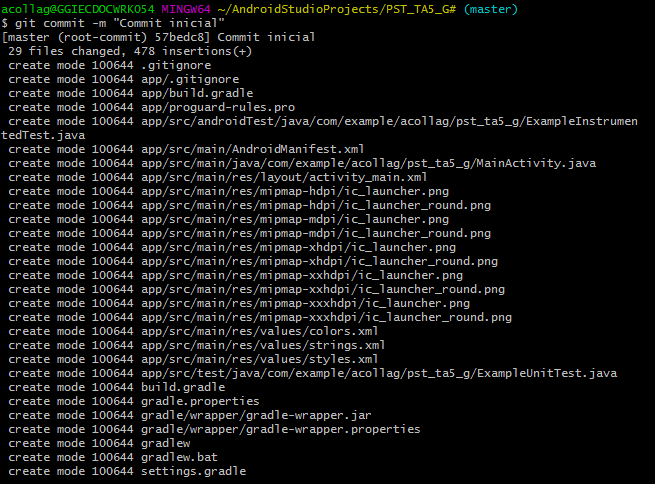


*Esto creará un archivo oculto [.git] para el manejo del repositorio y nos ubicará directamente en la rama “master”*

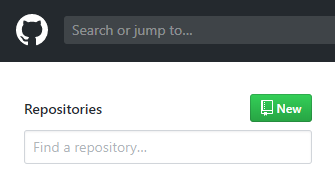
1. Agregamos todos los archivos del proyecto a nuestro **repositorio local** con el comando: “git add --all”.



1. 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.



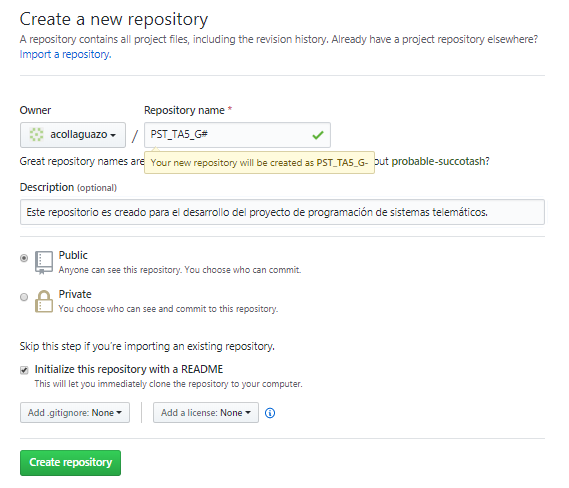
1. Creamos un repositorio en línea. Ahora usaremos Github (Requerira una cuenta gratuita). *Del lado superior izquierdo, encontrara el botón “NEW”.*



1. 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: OpenSouce, MIT, Apache, etc |

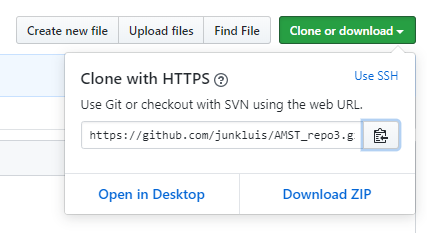
Debe asignarle el siguiente nombre a su repositorio “PST\_TA5\_G#”, reemplazando el “#” por su número de grupo, por ejemplo: “PST\_TA5\_G1”.



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



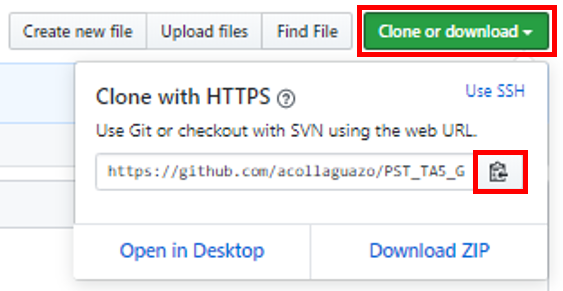
Vista de mi repositorio vacío.

1. 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]





1. 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)]

1. 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?

**Bitbucket:** Nos permite integrar otros servicios de Atlassian tales como Jira, HipChat, Bamboo, este destaca la posibilidad de que los usuarios puedan dejar análisis del código, seguimiento de tareas, permiso a los accesos a las diferente ramas de modificaciones y comparación de código..

**GitLab:** permite la configuración de los grupos de usuarios y las consultas y tareas con total libertad, además permite mover estas u otras tareas entre diferentes proyectos.

**SourceForge:** enfocado a la colaboración tradicional de blogs, foros, wiki, y listas de correo además de su función de repositorio y herramientas de rastreo.

Entre otras alternativas esta: GitKraken(atractivo y practica interaz, destacado por su velocidad de uso), LaunchPad (completamente gratuita con abundante herramientas tales como ratreo de bugs, análisis de cogidom hosting, proyectos de traducion entre otros) AWS CodeCommit (almacenamiento seguro, repositorios Git privados), GitKraken (enfocado a la velocidad, plataforma creada para ahorrar tiempo construyendo).

1. ¿Para qué sirve el archivo. gitignore y como se utiliza?

Es un archivo en el cual contiene la lista de los archivos o deterioros completos que son ignorados y los cuales no se encuentran en el repositorio de código. Se utiliza colocando en una lista especificando el nombre o directorio de los archivos que debe ignorar al realizar el commit o push.

1. ¿Qué limitaciones tiene GITHUB?

Las limitaciones de GITHUB son las siguientes: para poder acceder a todas las funciones de la plataforma, se debe pagar una mensualidad. Entre estas funciones están: un número ilimitado de usuarios colaboradores, más memoria de paquetes en el GITHUB, característica de dueño de código, entre otros. La otra limitación es el tamaño permitido de archivos en el repositorio.

1. ¿Qué es una rama?

Una rama es un entrono dentro del GITHUB, la cual al ser implementada en proyectos donde varias personas participan (como lo encargados del front-END o el back-END de la aplicación) pueden editar, cambiar y corregir partes del código sin que éstos cambios afecten a la estructura del código en general. Esto sirve para poder trabajar simultáneamente en un mismo proyecto sin esperar a que otra persona termine de editar su parte del código. Estas ramas se unen a la estructura principal del proyecto para ser testeadas y verificar su funcionalidad correctamente.

1. ¿Cuál es el link de su repositorio?

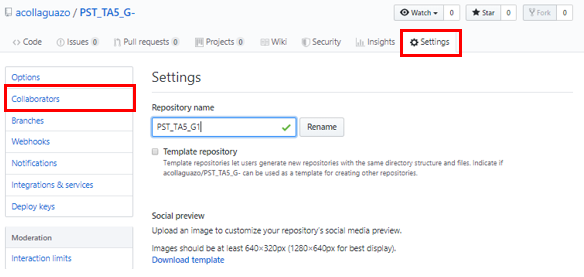
<https://github.com/bolisteward/PST_TA5_G8.git>

1. ¿Cómo utilizo un repositorio público (utilizando el comando git clone)?

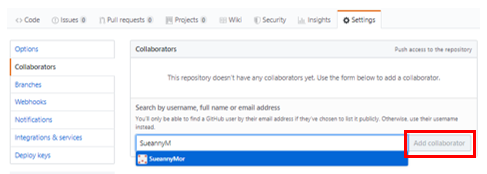
     Si se desea ser parte de un proyecto mediante la contribución de líneas de código en el mismo, se utiliza el comando git clone. Este comando, prácticamente copia toda la historia de archivos del proyecto y la descarga por defecto. Para poder hacer esto, se debe clonar con https el objetivo, luego se abre el terminal GitBash para cambiar el directorio de trabajo actual al que se copió la dirección https. Luego se escribe git clone seguido de dicha dirección obtenida.

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

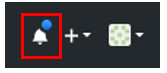
1. Para habilitar la modificación a otros miembros de mi grupo, debemos darle acceso. [Incluso si el proyecto es libre, solo pueden modificarlo quienes han sido invitados]. Para esto de clic en el tab “Settings/Configuración” > Collaborators / Colaboradores



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



1. 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”*



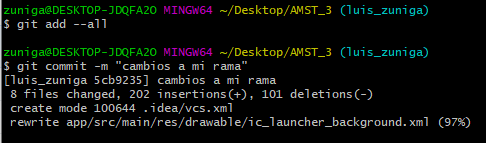
1. Ahora realizamos algunos cambios en nuestro proyecto local.

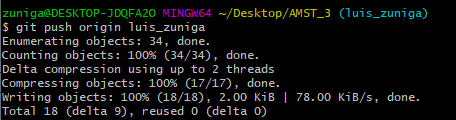


Sueanny\_Moreno\_Santos

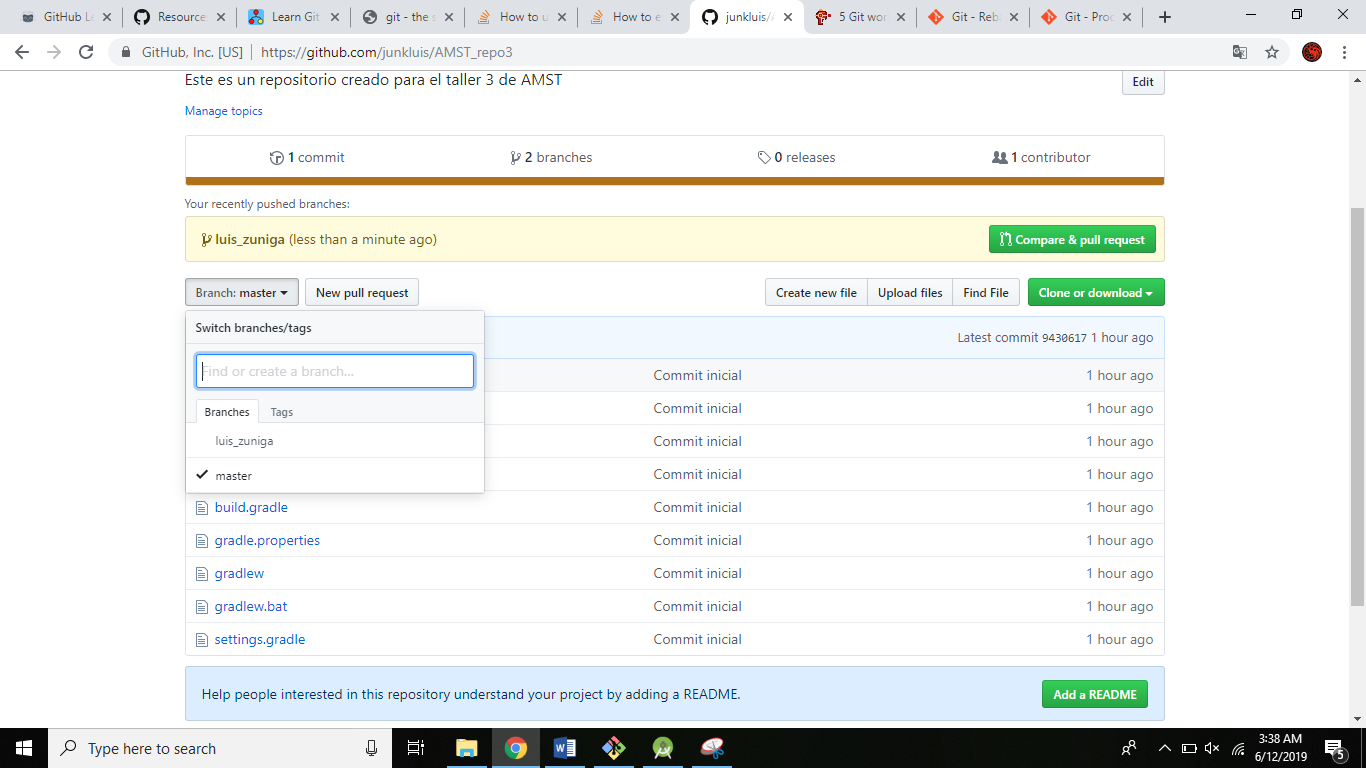
1. Estos cambios son únicamente reflejados dentro de nuestra rama. Ahora subimos los cambios, para ello realizamos los siguientes comandos.

|  |  |
| --- | --- |
| 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). |





1. 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?

Las ramas permiten acelerar el proceso de desarrollo de algún proyecto, como una aplicación por ejemplo, en donde a varias personas se les permite trabajar de manera paralela en el código general. Por lo general, estas ramas deben ser testeadas en el proyecto para verificar que el código implementado en ellas cumplan con los requerimientos de la aplicación.

1. ¿Qué tipos de conflictos puede ocurrir durante el manejo y creación de ramas?

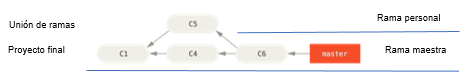
Puede ocurrir que no se haya creado una rama de prueba para el proyecto, y que se estén haciendo cambios drásticos al código empleado. El problema ocurre cuando se trabaja en la rama principal y por lo tanto se hace difícil lograr editar el código a la forma original, y en el peor de los casos empezar desde cero. También se puede cometer un error al momento de ponerle un nombre específico a la rama que se desea trabajar.

1. ¿Cuál es la diferencia entre **chekout** y **checkout -b?**

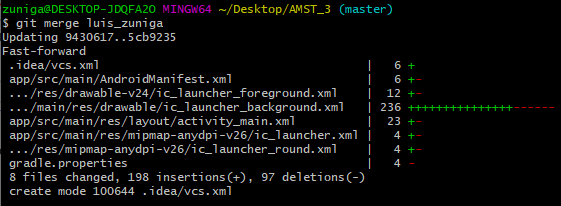
La diferencia entre estos dos comandos es que el comando checkout sirve para podernos cambiar a una rama ya existente dentro del GITHUB. Por otro lado, checkout -b es un comando que podemos utilizar para crear una rama y movernos a dicha rama en ese instante (si ya nos encontramos en otra rama, este comando nos permite seguir con el camino de ramas que se van creando).

**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.



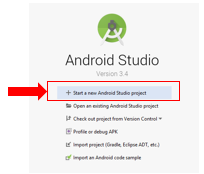
1. En caso de realizar algún cambio en el repositorio maestro.
   1. Git fetch origin master (obtiene todos los cambios realizados en master)
2. Nos cambiamos a la rama principal.
   1. Git checkout master
3. Traemos los cambios realizados en la rama única.
   1. Git merge [nombre rama]



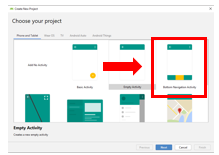
*Nota: Esto indica los archivos que han sido modificados.*

**Paso 5: Crear un nuevo proyecto en Android Studio**

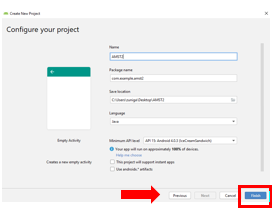
1. Al abrir Android Studio, podemos crear, abrir o importar proyectos. Seleccione “Start a new Android Studio project”.



1. 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.



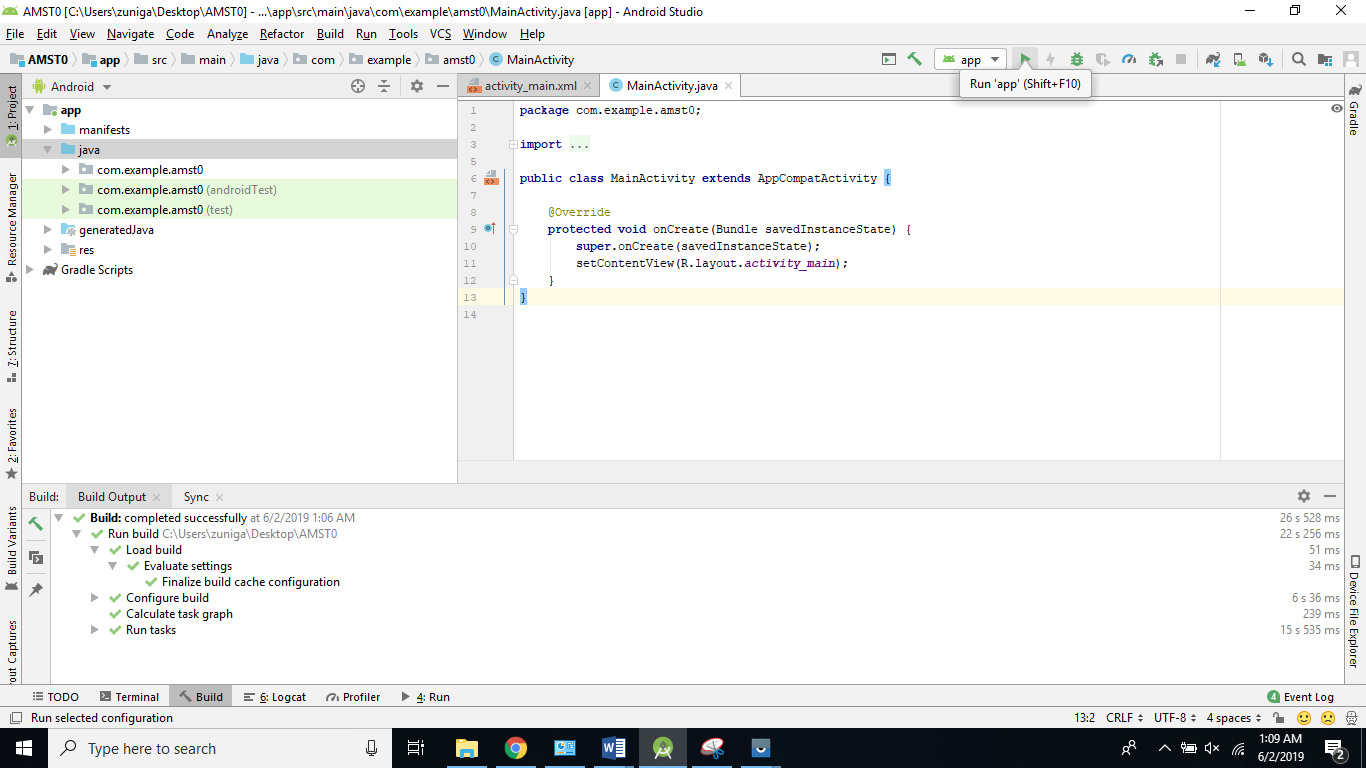
1. Configuración inicial del proyecto.
   1. [Name]: Colocaremos el nombre de nuestra app. (Recuerde que este nombre será reflejado en el PlayStore al momento de publicarlo). Para este taller, usaremos **AMST[numeroGrupo]**.
   2. [PackageName]: Paquete principal de código java, se obtiene automáticamente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



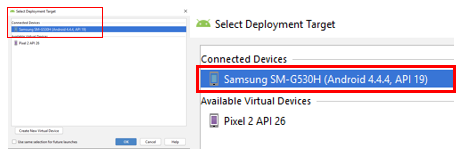
1. Como resultado se creará un proyecto vacío, solo presentado el mensaje “Hello World”

### **Ejecutar nuestra app en el teléfono**

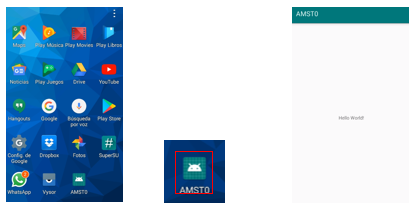
1. Del lado superior derecho de AndroidStudio aparecer la barra de ejecución (“RUN”).
2. Buscar el icono play para ejecutar nuestra app (o usar el atajo Shift+F10).



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



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



**TAREAS DE DESAFÍO:**

Para este desafío deberá desarrollar una aplicación con la temática de superhéroes. En la pantalla inicial se deberá mostrar en la pantalla completa una imagen representativa del universo cinematográfico que le guste más (Marvel o DC)



Ilustración 1 Pantalla Inicial

Seguido de eso deberá mostrar una pantalla de registro, para ello no es necesario que vincule su aplicativo a redes sociales, bastará con registrar un usuario y contraseña fijo (es decir, que puede definirlos/quemarlos en su código fuente).

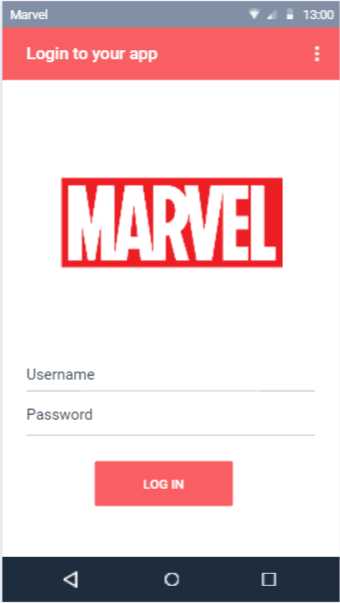


Ilustración 2 Pantalla de Registro

Posterior a ello una vez que valide su registro, será redirigido a una ventana en la que se muestren 6 películas o personajes del universo cinematográfico elegido. (Para este ejemplo se eligieron personajes, pero usted puede escoger películas si así lo desea.)

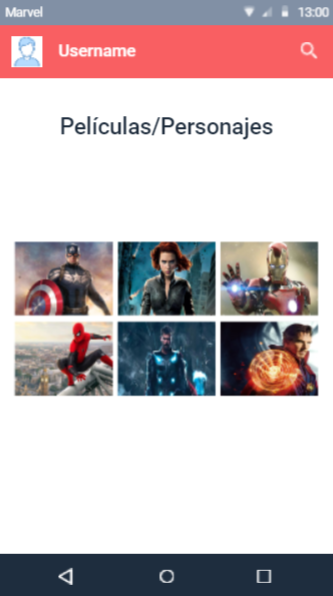


Ilustración 3 Ventana de Películas o Personajes

Finalmente, deberá asegurarse de que al pulsar la imagen de cada uno de esos personajes o películas deberá cambiar de pantalla, en la cual deberá mostrarse una descripción, como se muestra a continuación.

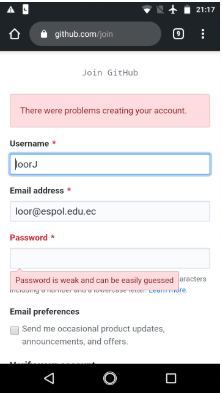


Ilustración 4 Descripción de personaje/película

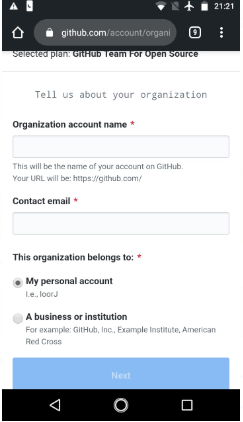
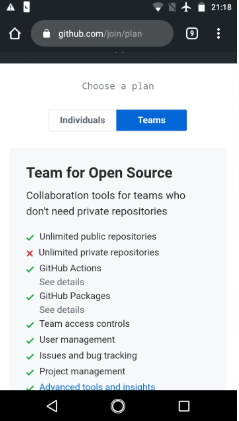
Nota: Es libre de cambiar el diseño de su aplicación mientras cumpla con los requerimientos.

**Desarrollo**

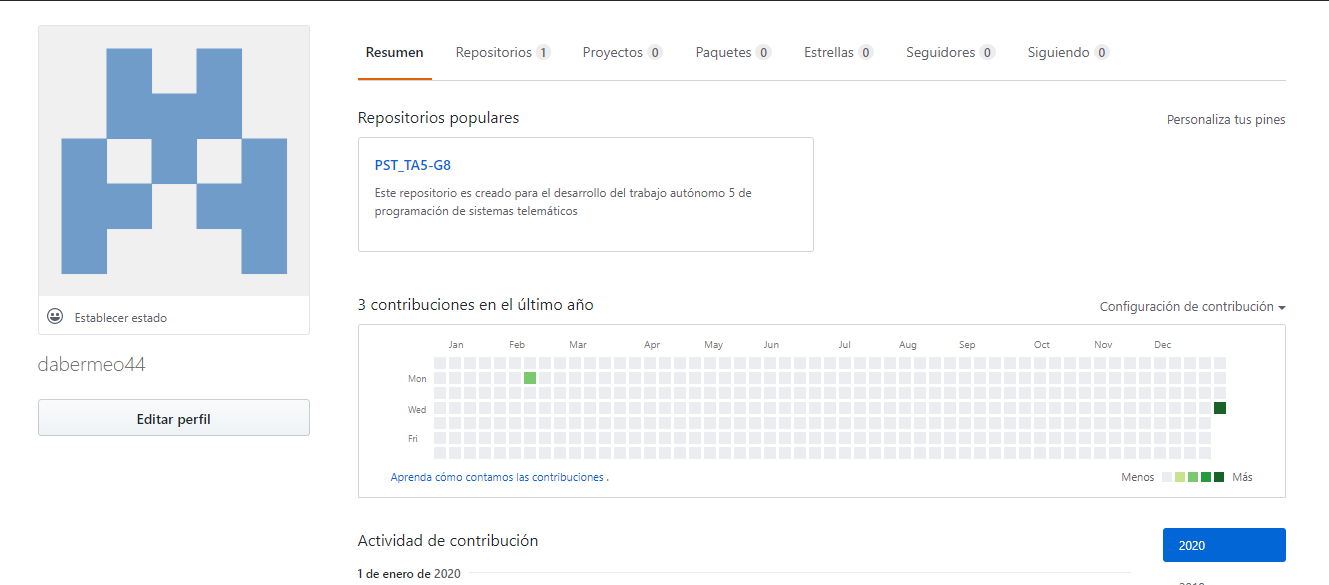
**Paso 0: Creación de una cuenta en GitHub.** Pasos que en la misma pagina de GitHub, va guiando para crear tu cuenta:



Se llenan todos los campos requeridos, se selecciona el plan que se requiera usar la cuenta, y una suscripción gratuita.

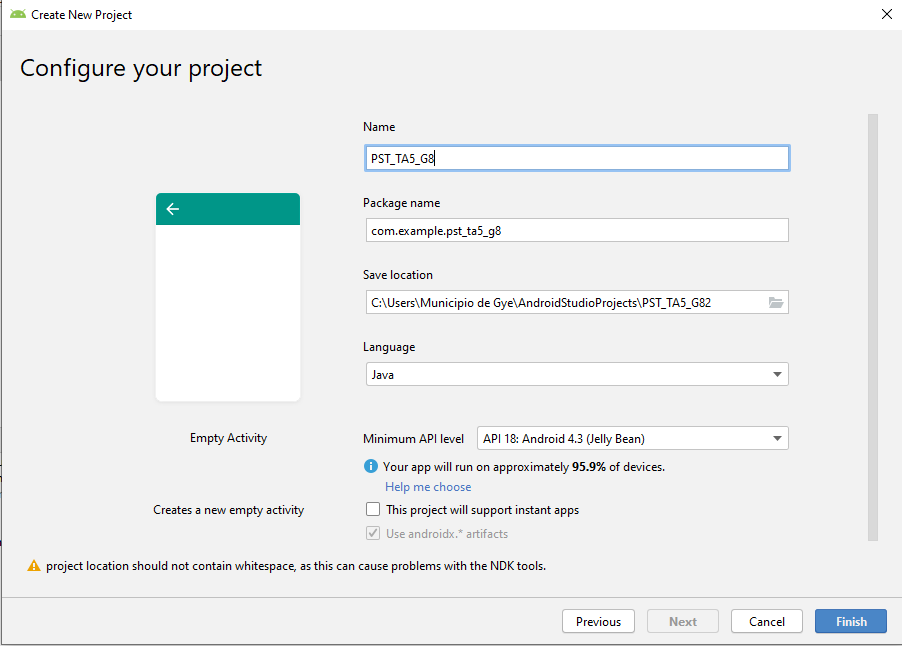


Y listo:

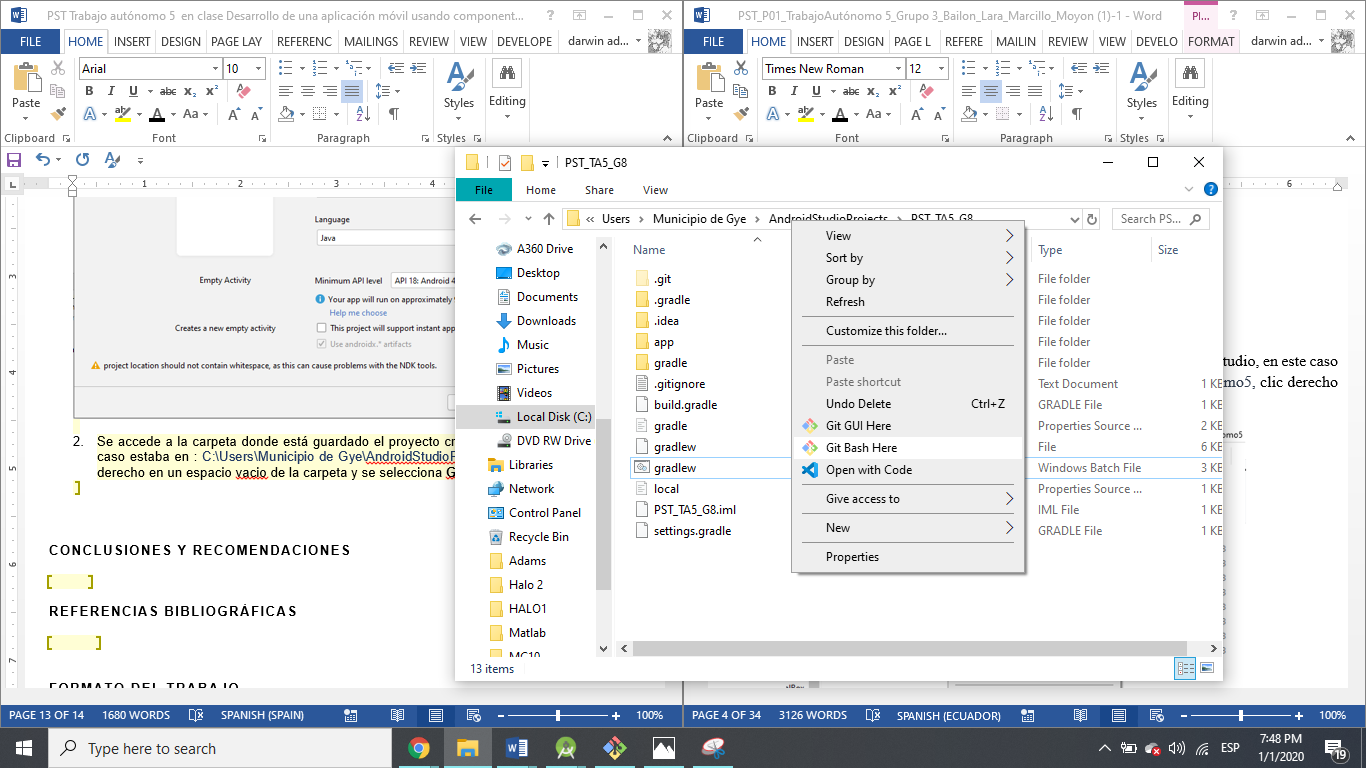


Paso 1: Crear un repositorio

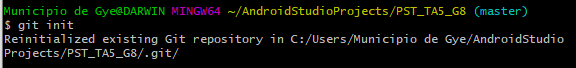
1. Se crea un proyecto de Android Studio, el cual será compartido en el repositorio de GitHub con todos los integrantes del grupo



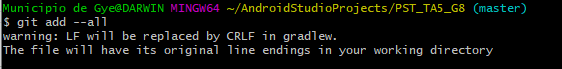
1. Se accede a la carpeta donde está guardado el proyecto creado en Android Studio, en este caso estaba en : C:\Users\Municipio de Gye\AndroidStudioProjects\PST\_TA5\_G8, se da clic derecho en un espacio vacio de la carpeta y se selecciona **Git Bash Here**



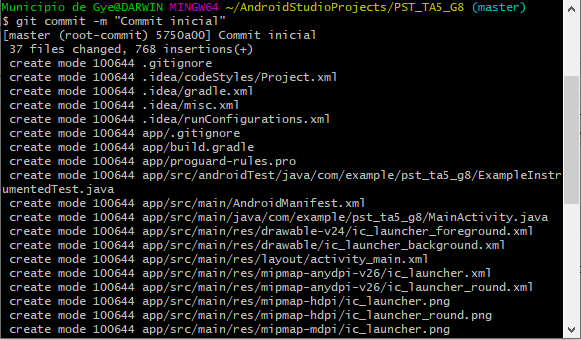
1. Creamos un repositorio vacío con el comando “git init” el cual permite conectar GitHub con el proyecto del TA5



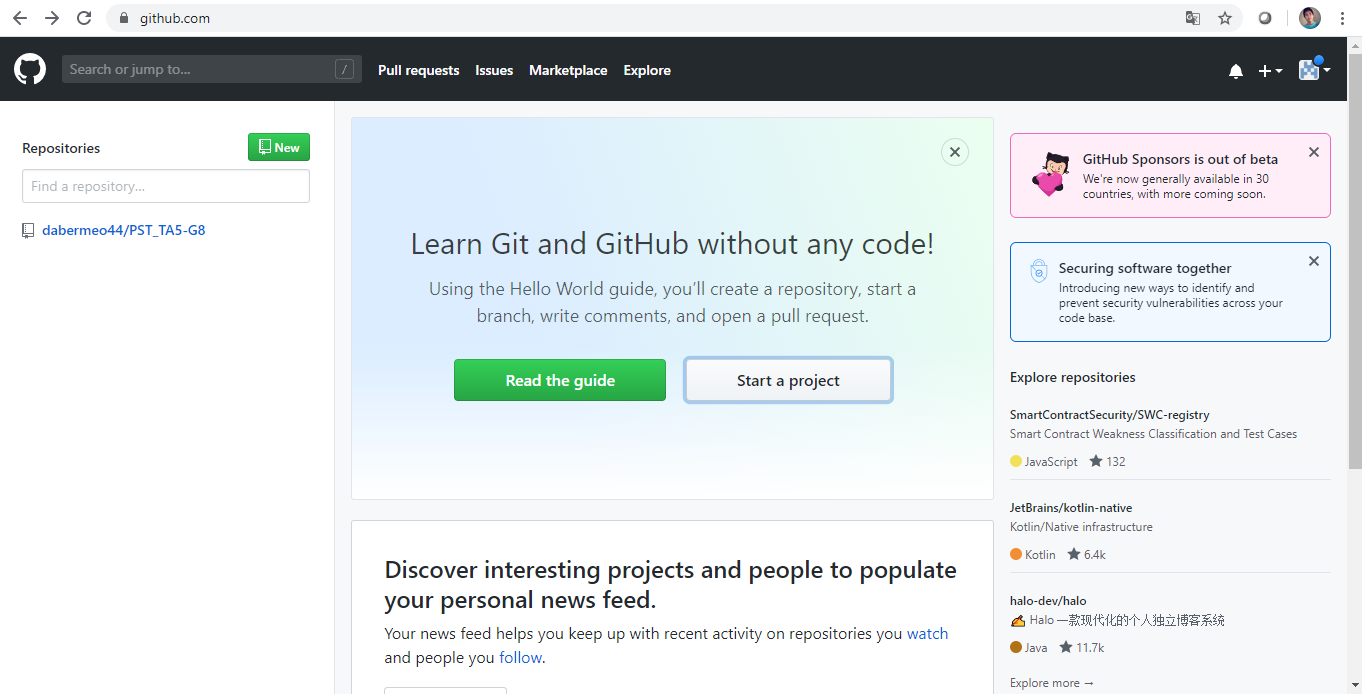
1. Agregamos todos los archivos del proyecto al repositorio local con el comando “git add –-all”



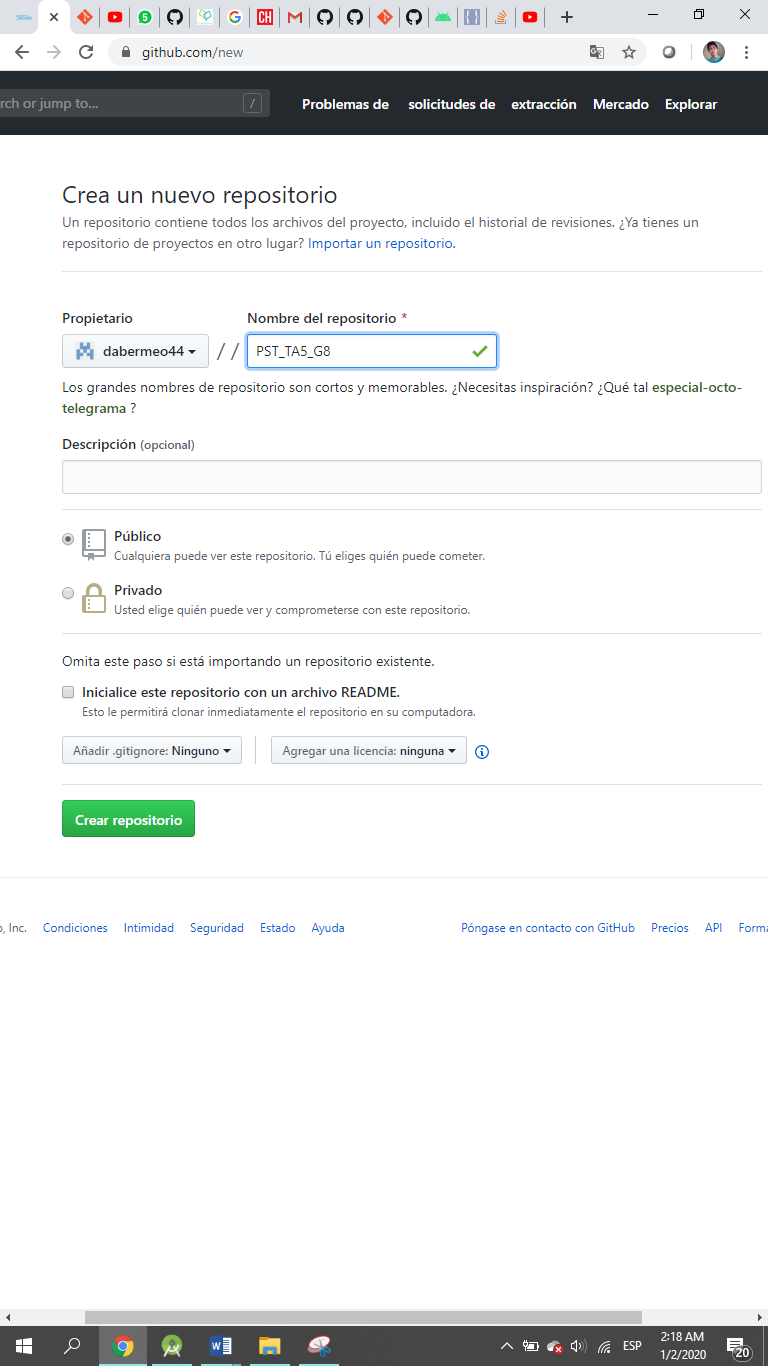
1. Se realiza un commit con el comando “git commit –m “mensaje del commit””, esto ayuda a guardar permanentemente los cambios realizado en el repositorio. Lo recomendable que los mensajes de los commit sea de acuerdo a los cambios que se ha realizado en el proyecto.



1. Creamos el repositorio en línea en la página principal de GitHub se hace click en el botón “New” que se encuentra en la parte superior izquierdo de la página.

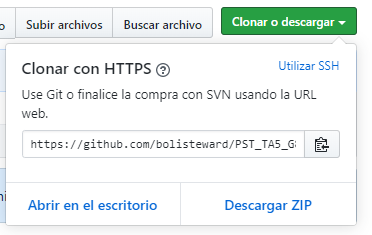


1. Se llenan los campos de acuerdo al requerimiento del presente trabajo autónomo y se da click en **Crear Repositorio**





1. En el repositorio creado, seleccionar “Clone or Download” y se copia el URL del repositorio



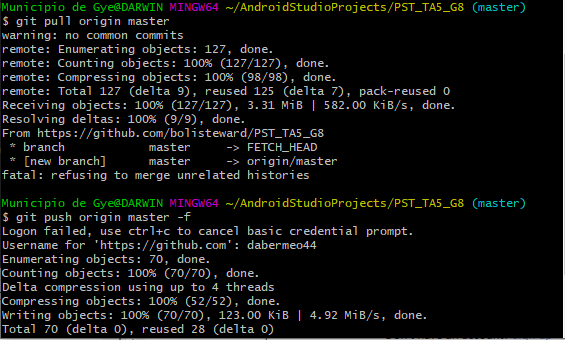
1. Obtener el repositorio de GitHub utilizando el comando “git remote add origin [URL del repositorio] ”, así se lo relaciona con el repositorio local.



1. Obtener la rama y publicar los cambios:

$git pull origin master

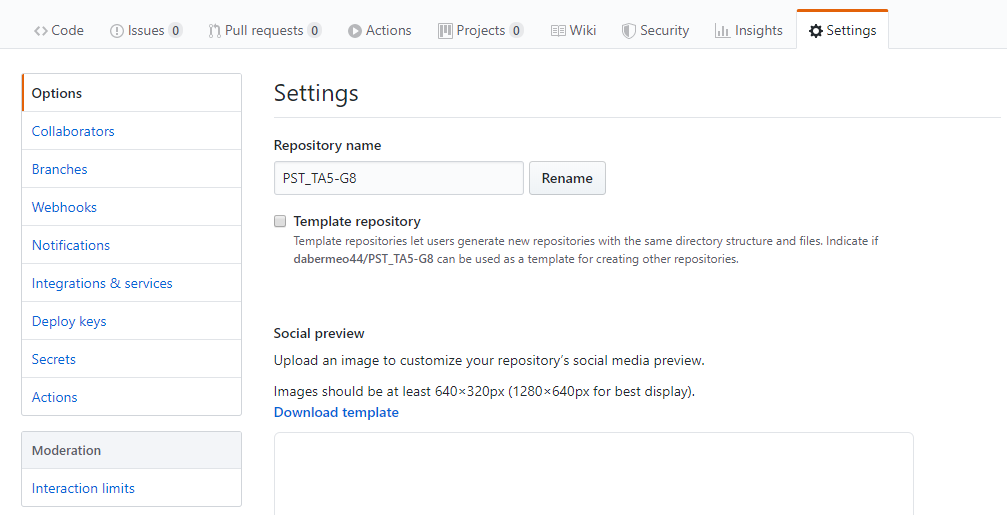
$git push origin master –f



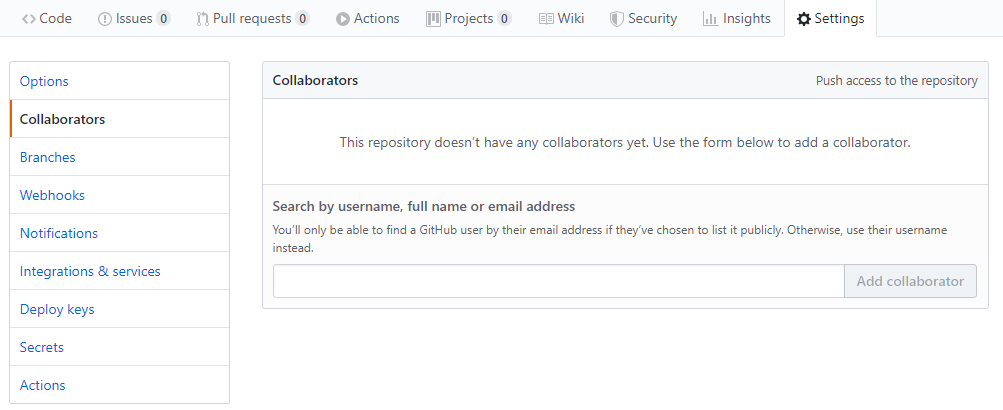
1. Para realizar cambios a la rama master, repetir los pasos 4, 5 y “git push origin master”.

**Paso 2: Añadir a los integrantes del grupo al repositorio**

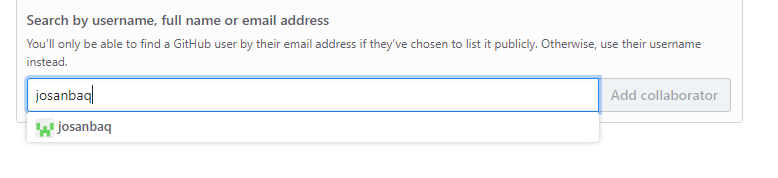
1. En el repositorio, hay que dirigirse a “Settings” y hacer click en la pestaña “Collaborators”



1. En la barra de búsqueda, escribir el nombre de usuario de tus compañeros.



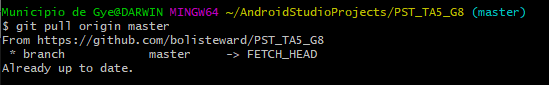
1. Finalmente, clickear el boton “Add collaborator”.



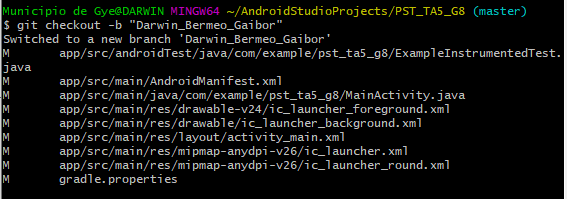
1. Repetir los pasos 2 y 3 hasta agregar a todos los compañeros y a la maestra titular.

**Paso 3: Crear una rama (branch)**

Se realizan los cambios hechos del proyecto en GitHuben la rama principal



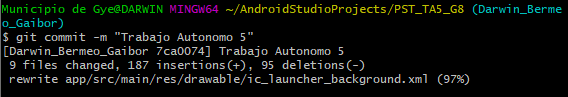
Se crean las ramas de un integrante del grupo:



Cuando se han realizado cambios en los archivos, se procede a listar todos los archivos a subir.

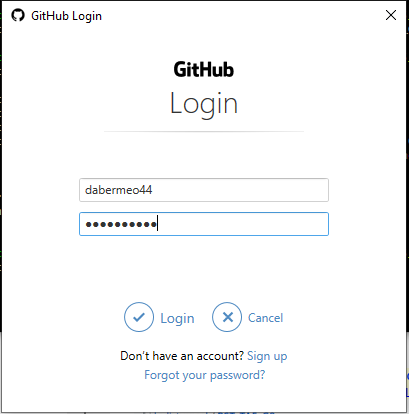


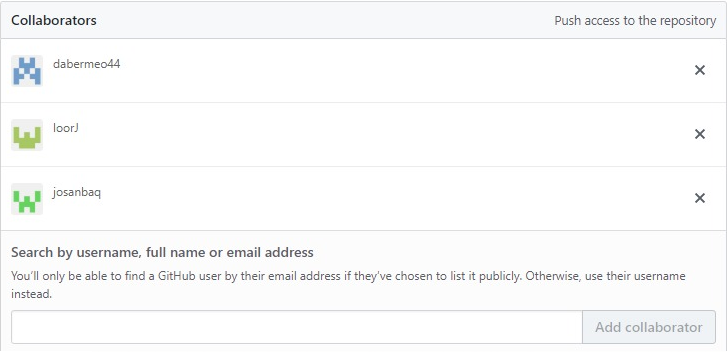
Se realiza un commit, esto es realizar un cambio en el repositorio local y se lo debe comentar



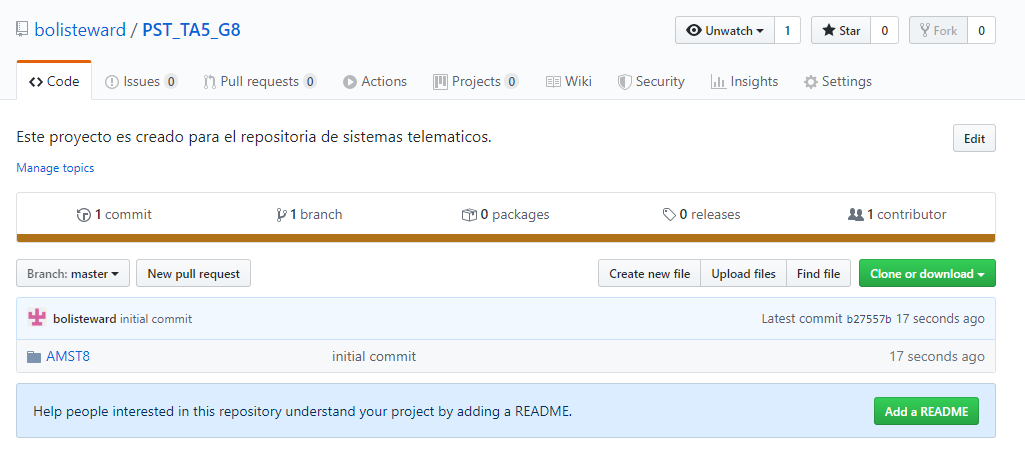
Ahora se suben los cambios en la rama personal



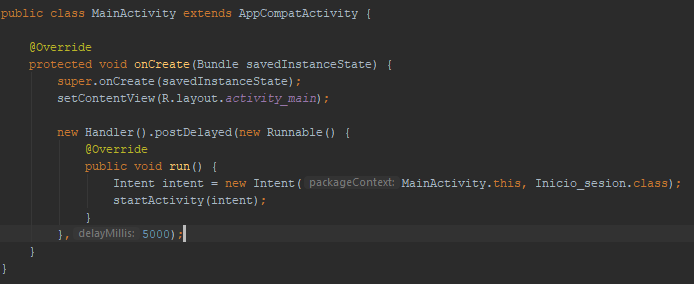
En el proceso de esto, preguntará por credenciales. Si se es colaborador, se podrá realizar con éxito.



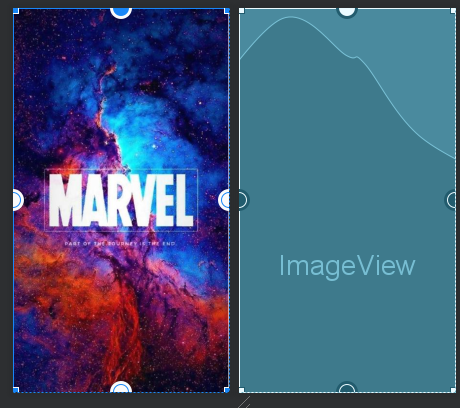
El repositorio con un nuevo proyecto en Android Studio en Git Hub cuyo enlace es el siguiente: <https://github.com/bolisteward/PST_TA5_G8.git>



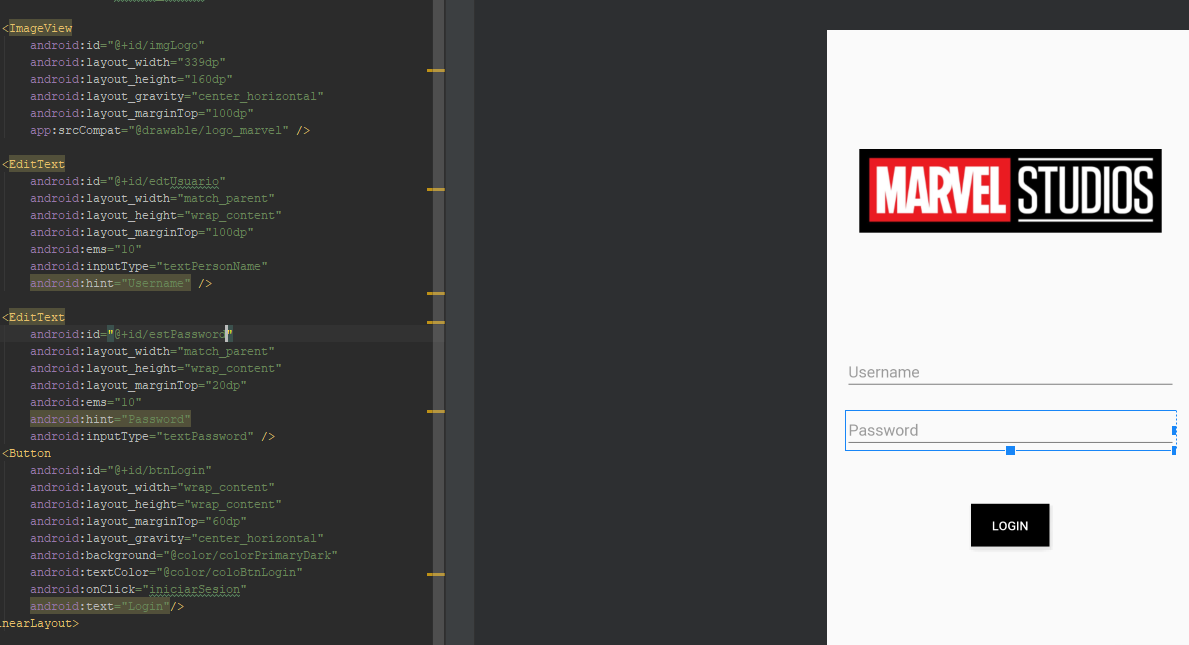
En el main activity se realiza un splash view mostrando el logo de Marvel Studios por 5000 ms y luego llama a la siguiente actividad Iniciar\_sesion mediante un Intent. El tiempo de retraso en el cambio de activity se realiza con el método postDelayed de clase Handler().



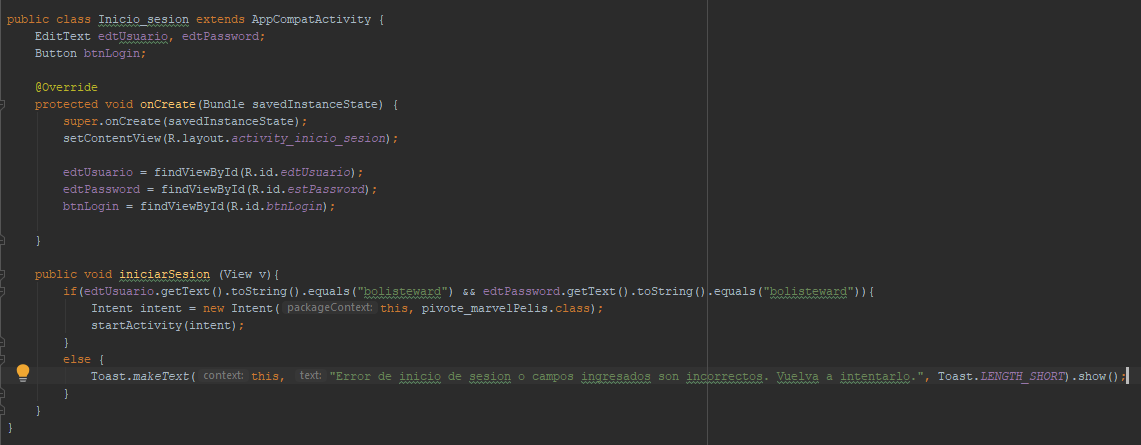
En la parte del archivo xml del main activity se coloco la respectiva imagen de portada de Marvel estudio.



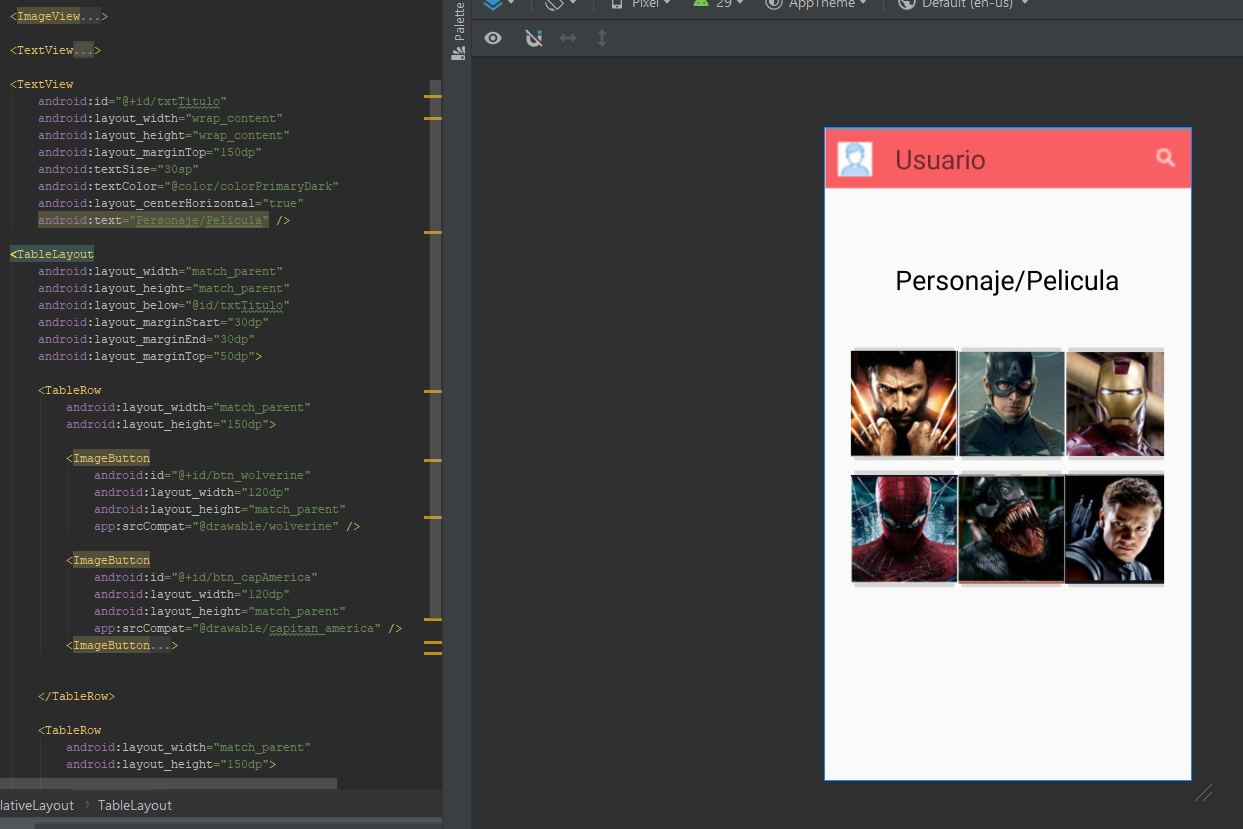
En el Activity Inici\_sesion, en su archivo xml se diseña con una image View con el logo, dos EditText para el usuario y contraseña que se ingrese, y el respectivo botón para validar y continuar con la siguiente actividad, el cual emplea el método iniciarSesion en el respectivo archivo java.



En su respectivo archive java se valida el usuario y contraseña verificando que ambos sean igual a “bolisteward”, el cual fue elegido predeterminadamente. En caso de colocar una respuesta errónea, mostrara un mensaje de error de inicio de sesión o campos llenados incorrectamente al pulsar el botón login, caso contrario se llama a la activity pivote\_marvelPelis.



En el pivote\_marvelPelis.xml se utilizo un tableLayout para colocar con mayor facilidad los imageButtom de forma ordenada y simétrica.



En su respectivo archivo java se coloca la función que realiza cada botón que es abrir una activity en la cual muestra información del personaje.

**Conclusiones y Recomendaciones**

conclusiones

GitHUB es una herramienta colaborativa que permite desarrollar trabajos grupales como proyectos a mayor velocidad y eficiencia. Esto es debido al uso e implementación de las ramas, las cuales son una característica de esta plataforma, que permiten desarrollar distintas partes del código general o del main branch al mismo tiempo y son testeadas previo a involucrarlas al proyecto que se encuentra desarrollándose.

EL USO DE ANDROID STUDIO PERMITE QUE SE UTILICEN HERRAMIENTAS DE DISEÑO PARA VISUALIZAR LAS IMÁGENES E INFORMACIÓN DE LAS PELÍCULAS DE MARVEL, LA MANERA EN LA QUE SE ABRE LA VENTANA POR PERSONAJE ES A TRAVÉS DE UN IMAGEBUTTON. LA MANERA EN LA QUE SE ORGANIZA ESTA PANTALLA ES A TRAVÉS DE UN TABLELAYOUT, ORGANIZANDO DE MEJOR MANERA LA PANTALLA.

EL USO DE RAMAS IMPLICA INCONGRUENCIAS ENTRE LOS TRABAJOS REALIZADOS POR LOS USUARIOS TRABAJANDO EN UN PROYECTO, POR LO QUE ES IMPORTANTE IDENTIFICAR QUE SUBIR Y QUE NO SUBIR EN EL PROYECTO, ESTO PUEDE SER APLICADO A TRAVÉS DE UN .GITIGNORE, DONDE NO SE TOMAN EN CUENTA CAMBIOS REALIZADOS EN EL PROYECTO Y NO SE SUBIRÁN A LA RAMA PRINCIPAL.

recomendaciones

se debe ser precavido y cuidadoso al trabajar con las ramas en github, ya que siempre se debe crear una rama de prueba previo a la unión de los códigos o de realizar commits. Esto es debido a que se puede estar trabajando en el main branch por error, editando así todo el código general de manera inoportuna.

se recomienda verificar que los permisos necesarios para implementar la aplicación correctamente estén habilitados en el android manifest, ya que de esta forma se puede trabajar con distintas funionalidades como el ethernet, google maps u otros factores.

se sugiere implementar la aplicación en un dispositivo móvil real en vez de uno virtual. Esto es debido a que de esta forma se puede conseguir un mejor retroalimentación por parte del usuario sobre cómo está estructurado el proyecto, su dinamismo, el dimensionamiento u otras características del mismo.

**Referencias Bibliográficas**

Gaby Briceño. (2017). GitHub vs. GitLab vs. Bitbucket: ¿Que repositorio elegir?. 1 de enero del 2020, de Club de Tecnología Sitio web: <http://www.clubdetecnologia.net/blog/2017/github-vs-gitlab-vs-bitbucket-que-repositorio-elegir/>

Ayuda de GitHub. (2019). Clonar un Repositorio. 1 de enero del 2020, de Ayuda de GitHub Sitio web: <https://help.github.com/es/github/creating-cloning-and-archiving-repositories/cloning-a-repository>

Git. (2019). Ramificaciones en Git - ¿Qué es una rama?. 1 de enero del 2020, de Software Freedom Conservancy Sitio web: <https://git-scm.com/book/es/v2/Ramificaciones-en-Git-%C2%BFQu%C3%A9-es-una-rama%3F>

Git. (21 de febrero del 2019). Que son las Ramas (Branches) en Git, como utilizarlas y otros detalles. 1 de enero del 2020, de NC Sitio web: <https://blog.nubecolectiva.com/que-son-las-ramas-branches-en-git-como-utilizarlas-y-otros-detalles/>

**Formato del trabajo**

El trabajo autónomo será desarrollado en el siguiente formato:

* Nombre del archivo: PST\_P01\_Trabajo Autónomo A\_Grupo B\_Apellido1\_Apellido2\_Apellido3
* *(\*) Siendo A el número del trabajo y B el número del grupo*
* Nombre de la materia
* Título del trabajo: Ejemplo: Trabajo Autónomo A - Tema
* Nombre de la profesora
* Número de grupo
* Nombres/Apellidos de los integrantes del grupo que hayan desarrollado el trabajo
* Fecha de inicio y fin del trabajo
* Resultados de las actividades planteadas: Explicación de las actividades ejecutadas, incluyendo las imágenes del proceso.
* Conclusiones y Recomendaciones: Respecto a lo aprendido durante el desarrollo del trabajo.
* Referencias bibliográficas: Colocar los documentos, enlaces web o libros consultados.

1. Andorid Studio <https://developer.android.com/studio/intro?hl=es> [↑](#footnote-ref-1)
2. GitHub https://github.com [↑](#footnote-ref-2)