

Autenticación

Grupo 1
ID Opera: 18

Manuel Gómez Suárez
Federico Alvaro Plasencia García de Diego
Alejandro Polvillo Hall
Raúl Romero Palomo
Tomás Ruán Rollán
Francisco Javier Santos Velázquez

Repositorio: <https://github.com/Proyecto-EGC-G1/Autenticacion-EGC-G1>
Sistema desplegado: <https://g1login.egc.duckdns.org/login>
Ópera: <http://opera.eii.us.es/egc/public/trabajo/ver/id/91>

Índice

Índice	2
Resumen	3
Introducción y contexto	3
Descripción del sistema	4
Cambios desarrollados	5
Planificación del proyecto	6
Milestone 1	6
Milestone 2	6
Milestone 3	6
Milestone 4	6
Reparto de tareas	7
Entorno de desarrollo	7
Instalación del entorno de desarrollo	8
Gestión del cambio, incidencia y depuración	8
Evidencias	10
Gestión del código fuente	12
Gestión de ramas	12
Lecciones aprendidas	13
Roles en la gestión del código	13
Lecciones aprendidas	14
Gestión del código en las ramas	14
Lecciones aprendidas	15
Gestión de la construcción e integración continua	15
Gestión de la liberaciones, despliegues y entregas	17
Mapas de herramientas	18
Ejercicio de propuesta de cambio	19
Conclusiones y trabajo futuro	26
Conclusiones	26
Mejoras	27

Resumen

El proyecto ha consistido en realizar todas las funciones relacionadas con la autenticación del sistema, de manera que un usuario (votante o admin) pase una serie de pasos de seguridad en el inicio de sesión antes de poder acceder a la aplicación

Entre las funcionalidades implementadas se encuentran:

- Login, con el correspondiente formulario.
- Autenticación y seguridad del proceso de login y logout.
- Asignación de los roles de los usuarios (votante o admin).

Introducción y contexto

El proyecto ha consistido en realizar todas las funciones relacionadas con la autenticación del sistema, de manera que un usuario (tanto un votante como un administrador) al logearse en el sistema tiene que pasar un proceso de seguridad, para asegurar que es quién dice ser (para que no se produzca ningún tipo de fraude, al ser nuestro proyecto un sistema de votaciones) y preparar una sesión acorde con su correspondiente rol.

Hemos implementado las funcionalidades del proceso que sigue un usuario (tanto un votante como un administrador del sistema) desde el momento del login para entrar en el sistema hasta el momento de la redirección al listado de votaciones.

Entre las funcionalidades implementadas se encuentran:

- Login, con el correspondiente formulario.
- Autenticación y seguridad del proceso de login (con la comprobación de contraseñas (incluida la contraseña de WordPress con la nuestra para permitir el logueo), la inclusión de captchas, comprobación de cookies, etc.) y logout (con el correspondiente borrado de cookies).
- Asignación de los roles de los usuarios (votante o admin), con la posible modificación de éstos.

Descripción del sistema

El sistema desarrollado es un sistema de autenticación por contraseña. Este sistema te permite acceder a una aplicación web previa comprobación de que eres un usuario que pertenece al sistema. Al ser un sistema integrado en una aplicación con una arquitectura de microservicios la función de nuestro sistema es doble: Permitir a los usuarios iniciar sesión en el sistema y dar a los otros sistemas que componen la aplicación una vía para que puedan comprobar si los usuarios que hacen peticiones a su sistema pertenecen o no a la aplicación. También nos encargamos de asignar los roles a los usuarios a través la misma API que exponemos para los otros sistemas.

Arquitectónicamente nuestro sistema está compuesto por dos elementos principales:

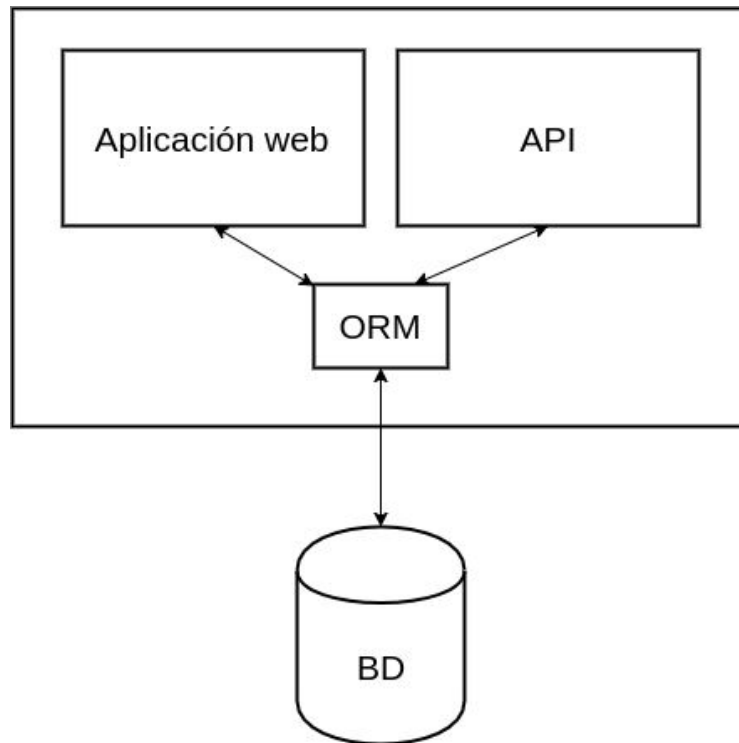
- Una aplicación web
- Una API

La primera se encarga de tener toda la funcionalidad que permita al usuario iniciar sesión. La segunda tiene como objetivo facilitar cierta funcionalidad a los demás subsistemas, siendo esta funcionalidad la siguiente:

1. Comprobación de usuarios
2. Asignación de roles
3. Cierre de sesión

Nosotros nos comunicamos con todos los subsistemas, no iniciamos nosotros esta comunicación, sino el resto de subsistemas, pues el primer paso que tienen que hacer todos los subsistemas al recibir una petición es comprobar si el usuario pertenece al sistema y eso lo tienen que hacer a través de nosotros.

Para la conexión con la base de datos usamos un ORM que nos facilita la gestión de los datos con ésta.



Cambios desarrollados

Nosotros no hemos partido de código heredado así que el proyecto ha sido desarrollado desde cero, la funcionalidad desarrollada es la siguiente:

- Inicio de sesión
- Comprobación de cookies (comprobación de usuarios)
- Asignación de roles
- Cierre de sesión

Planificación del proyecto

El proyecto se ha realizado en 4 milestones que se describen a continuación:

Milestone 1

En el primer milestone principalmente se ha preparado el ecosistema de desarrollo y se ha subido la primera plantilla del proyecto.

Milestone 2

En el segundo milestone se creó el mapeo de clases Python a las tablas de la base de datos y se desarrolló la funcionalidad de inicio de sesión tanto la lógica como la plantilla HTML.

Milestone 3

El milestone 3 fue el más productivo, una vez puesto en marcha se desarrolló toda la parte de la API, se mejoró la parte de inicio de sesión para que fuera más segura, se crearon tests de la aplicación, se configuró la integración continua y la generación de la imagen de Docker en la integración continua.

Milestone 4

Se solucionaron algunos fallos respecto a la base de datos y el login. A parte se añadió un captcha en el Login, aparte de la generación de este documento

Reparto de tareas

	Feature	Bug	Enhancement	Documentation	Investigation
Alejandro Polvillo Hall	5	1	4	5	0
Tomás Ruán Rollán	3	0	2	3	0
Fco. Javier Santos Velázquez	3	0	1	2	2
Manuel Gómez Suárez	5	0	0	2	1
Raúl Romero Palomo	3	0	0	3	0
Álvaro Plasencia García de Diego.	6	2	1	3	0

Entorno de desarrollo

Para desarrollar el sistema hemos usado los siguientes componentes:

Python 3.5 -> Lenguaje usado para el desarrollo

Flask 0.12.1 -> Framework web usado para el desarrollo de la aplicación web y la API

SQLAlchemy 1.1.15 -> ORM para la conexión con la base de datos

MariaDB 10.6 -> Base de datos

Instalación del entorno de desarrollo

1) Lo primero que necesitamos es desplegar la base de datos para poder levantar el proyecto exitosamente, para ello haremos uso de Docker para desplegar la base de datos MariaDB, los scripts de la base de datos se encuentran en este mismo directorio del repositorio clonado, en la carpeta 'database'. Abriendo una consola en el directorio 'database', tendrás que construir la imagen de Docker la primera vez, para ello se ejecutará:

```
docker-compose up -d --build
```

2) Lo segundo que necesitamos es instalar las dependencias del proyecto, como nosotros vamos a usar Python 3, haremos uso de su gestor de dependencias 'pip3'. Instalaremos todas las dependencias con el archivo 'requeriments.txt' provisto en este mismo directorio:

```
pip3 install -r requeriments.txt
```

3) Hay que configurar las siguientes variables de entorno:

```
MARIADB_HOST=127.0.0.1  
MARIADB_PORT=3306  
MARIADB_USER=root  
MARIADB_PASSWORD=root123
```

4) Ya se puede usar el sistema para desarrollar. Con el objetivo de ejecutar el sistema simplemente se podría ejecutar lo siguiente:

```
python3 controllers.py
```


Gestión del cambio, incidencia y depuración

Para las nuevas características a añadir al proyecto o bugs que se encuentren se creará una tarea, esta tarea estará compuesta por la siguiente información:

- Título: Título de la tarea.
- Descripción: Descripción de la tarea con detalles.
- Asignación: Asignación a persona, si no tiene asignación, se deja como está.
- Etiqueta: Dependiendo del tipo de tarea.
- Proyecto: El proyecto al que pertenezca.
- Milestone: Si pertenece a alguna milestone fijada, o ninguna si no pertenece a ninguna milestone.

Una vez se cree ésta tarea, se situará en un tablero Kanban, que indica el estado en el que se encuentra actualmente la tarea. Estos estados pueden ser 'Need Assignment', 'To Do', 'Need Revision' y 'Done'.

El flujo que sigue una tarea es el siguiente: si una tarea recientemente creada no tuviera asignación, al crearla se establecería en la columna 'Need assignment', así sabríamos las issues que están sin asignar. Una vez se asigne a una persona, ésta misma tendrá que cambiarla a la columna 'To Do'. Cuando esta tarea se complete pasará al estado 'Need revision' con el objetivo que otra persona del equipo la revise. Una vez otra persona del equipo la revise y la acepte, se cerrará la tarea y esta pasará a estado 'Done'

Todo cambio realizado en el proyecto que tenga que ver con una tarea en concreto, se referenciará a ésta tarea con el objetivo de poder hacer un seguimiento de los cambios que han afectado al proyecto debido a esa tarea.

Para las incidencias externas, el coordinador de otro grupo vendrá a nuestro grupo con la susodicha incidencia. Entonces se meterá la issue en nuestro tablero y se asignará al coordinador del otro grupo y al miembro de nuestro equipo que se vaya a encargar de la incidencia.

Como herramienta que da soporte a estos procesos usaremos Github. Para el manejo de las incidencias, usaremos las issues de Github junto con los proyectos de Github, lo que nos dará una combinación que nos permita saber todas nuestras incidencias y el estado de ella en un tablero Kanban que también será el de GitHub, creando las columnas previamente dichas.

Para los mecanismos de depuración usamos los que nos ofrece PyCharm, que es el IDE que utilizamos para el proyecto.

Evidencias

Ejemplo de issues

<input type="checkbox"/>	6 Open ✓ 45 Closed	Author ▾	Labels ▾	Projects ▾	Milestones ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	<div><div>Mejorar el captcha</div><div>feature</div></div> <div>#51 opened 10 minutes ago by Garbancito</div> <div>Milestone 4</div>						
<input type="checkbox"/>	<div><div>Redactar documento: Gestión del cambio, incidencias y depuración</div><div>documentation</div></div> <div>#49 opened 5 hours ago by caolin333</div> <div>Milestone 4</div>						2
<input type="checkbox"/>	<div><div>Redactar documento: Introducción y contexto</div><div>documentation</div></div> <div>#47 opened 2 days ago by tomasruan</div> <div>Milestone 4</div>						
<input type="checkbox"/>	<div><div>Redactar documento: Resumen</div><div>documentation</div></div> <div>#46 opened 2 days ago by tomasruan</div> <div>Milestone 4</div>						
<input type="checkbox"/>	<div><div>Redactar documento: Gestión de liberaciones, despliegue y entregas</div><div>documentation</div></div> <div>#45 opened 3 days ago by frasanvel</div>						1
<input type="checkbox"/>	<div><div>Redactar documento: Gestión de la construcción y CI</div><div>documentation</div></div> <div>#44 opened 3 days ago by frasanvel</div>						1

Ejemplo de issue cerrada

Generar cuenta para captcha en el login #42

Edit New issue

Closed caolin333 opened this issue 3 days ago · 1 comment

caolin333 commented 3 days ago

Member

+ 😊 ✎

Nueva característica para que, al logearse, el usuario se le pida un captcha antes de ser redirigido a la página del listado de votaciones. En esta issue se generará la cuenta aleatoria que se le mostrará al usuario.

caolin333 added the feature label 3 days ago

caolin333 added this to the **Milestone 4** milestone 3 days ago

caolin333 self-assigned this 3 days ago

caolin333 added this to **To do in Login** via automation 3 days ago

caolin333 added a commit that referenced this issue 2 days ago

Añade generación de operación del captcha en el método GET y POST de1... 5c42cc7

caolin333 added a commit that referenced this issue 2 days ago

Añade campos para la operación del captcha en el método GET y POST de... 6719b12

manuelgomezsuares moved this from **To do** to **Need Revision** in **Login** 2 days ago

alejandrohalla commented 3 hours ago

Owner

+ 😊 ✎ ✕

Integrado en develop

alejandrohalla closed this 3 hours ago

Start timer

Assignees

caolin333

Labels

feature

Projects

Done in Login

Milestone

Milestone 4

Notifications

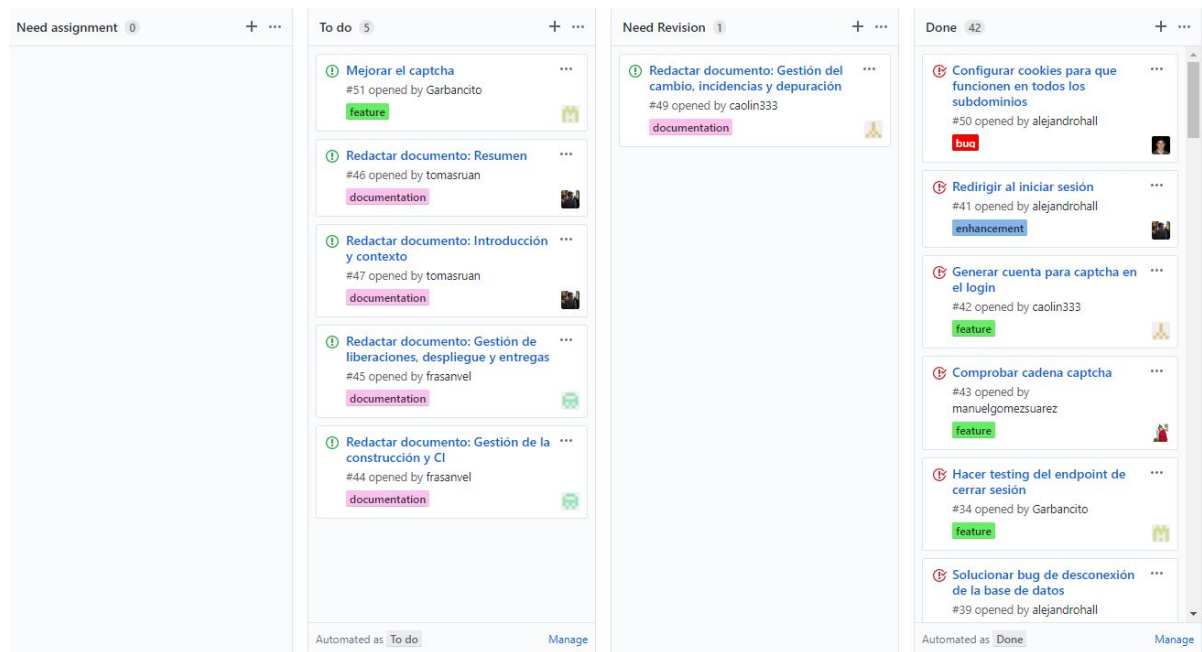
Unsubscribe

You're receiving notifications because you were assigned.

2 participants

Lock conversation

Tablero KanBan



Gestión del código fuente

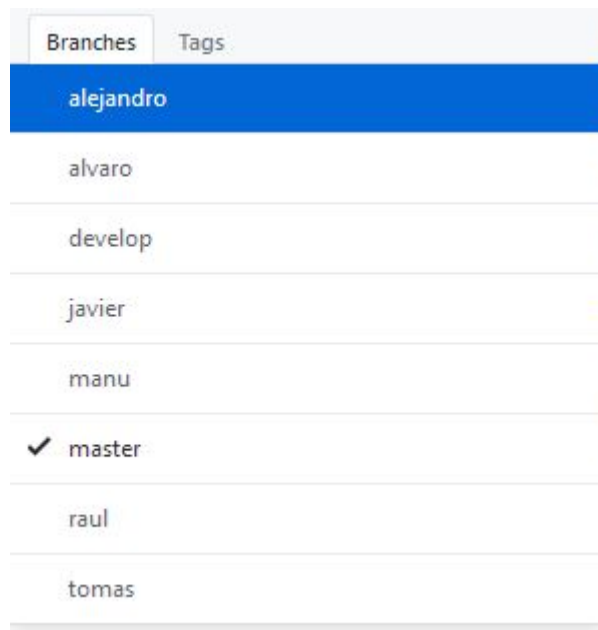
Gestión de ramas

Tal como propone Gitflow, nuestro proyecto tiene las ramas Master y Develop.

En la rama Master, siempre debe existir una versión estable lista para subir a producción.

En la rama Develop, se encuentra el código que conformará la siguiente versión planificada del proyecto.

Nuestro proyecto no sigue el modelo de flujo de trabajo propuesto por Gitflow para las ramas auxiliares. Esto se debe a que los miembros del grupo no tienen una amplia experiencia con el manejo de Git, por tanto cada miembro del equipo tendrá su propia rama en la cuál desarrollará el código y de esta forma pueda tener un mejor control de su trabajo.



Lecciones aprendidas

Este modelo de flujo de trabajo nos ha funcionado muy bien y no hemos tenido problemas ni necesidad de cambiarlo, el equipo ha adquirido suficiente experiencia con el manejo de las ramas y en un proyecto futuro estaríamos preparados para adoptar por completo el flujo de trabajo propuesto por Gitflow.

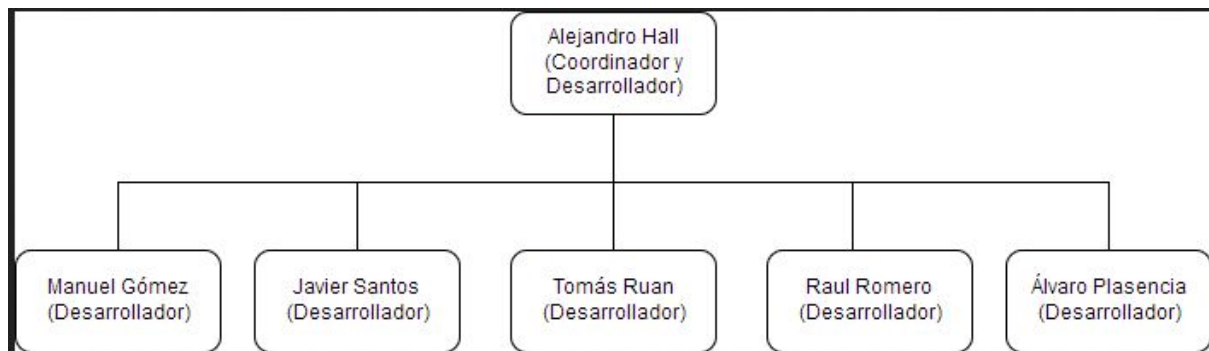
Roles en la gestión del código

Nuestro grupo está conformado por 6 miembros, de los cuales uno es Coordinador del equipo y desarrollador (Alejandro Hall) y los demás desarrolladores.

Esto es debido a que el coordinador del equipo tiene más experiencia en proyectos con Git que el resto, siendo por tanto el mejor candidato para organizar al grupo.

El coordinador es el encargado de organizar las tareas para cada milestone, revisar el código y dar su aprobación para que este sea integrado en develop y decidir cuando la versión de develop está lista para integrarse a master.

Además el coordinador es el que ha llevado las relaciones con los demás grupos de trabajo. Los desarrolladores son los encargados de realizar las tareas asignadas en cada Milestone, estos también pueden crear nuevas tareas, ya sea para añadir nuevas características o para mejorar algo ya hecho.



Lecciones aprendidas

Los miembros del grupo, han ido mejorando su manejo con Git sacando provecho de los conocimientos del coordinador, por tanto nos parece una buena forma de organizarse. Por contrapartida, podemos decir que todas las dudas y cuestiones por parte de los miembros del grupo se dirigen a una sola persona, en este caso al coordinador, el cual puede llegar a saturarse y afectar al desarrollo del proyecto y a su duración, cosa que por fortuna no ha sucedido.

Por tanto creemos conveniente que en futuros proyectos podríamos repartir mas las responsabilidades sobre varios miembros del equipo. Por ejemplo podríamos haber asignado la responsabilidad de comunicación con otros grupos a otro miembro del equipo.

Gestión del código en las ramas

Cuando se asigna una tarea a un usuario, este puede empezar a trabajar el código en su propia rama.

Cada vez que realice un cambio lógico deberá hacer un commit, para ello debe acompañar al commit de un mensaje lo suficientemente descriptivo para saber que contiene dicho commit sin tener que abrirlo.

Los commits deberán seguir el siguiente formato:

Verbo en imperativo + descripción cambio lógico

Cada commit que realizado tiene una causa, la cual es la Issue o tarea que el usuario tiene asignada, por tanto además de la descripción del cambio lógico, se debe añadir en el mensaje del commit la issue referenciada, por ejemplo:

Corrije archivo para construir imagen de docker. Ref #30

De esta forma, Github se encarga de realizar la asociación entre la Issue y el commit realizado.

Una vez un usuario termina su tarea en su rama, es el coordinador del equipo el que se encargará de revisar el trabajo realizado y decidir si el cambio se aprueba.

Cuando el coordinador da el visto bueno, los cambios se pasan a la rama develop.

Una vez se quiera lanzar la versión de la rama develop a producción, se incluirán los cambios de esta a la rama Master.

Lecciones aprendidas

Esta forma de gestionar el código facilita mucho la revisión por parte del coordinador del equipo y también a los demás miembros del equipo cuando estos necesitan consultar antiguos commits o código de otros miembros.

Gestión de la construcción e integración continua

La construcción se lleva a cabo una vez se hayan aplicado nuevos cambios al proyecto y en un entorno de desarrollo aislado al resto de componentes del equipo, para así evitar posibles conflictos entre ellos.

Para automatizar el proceso de construcción usamos las siguientes herramientas, detalladas a continuación:

La gestión de la construcción está integrada directamente en Travis junto a la integración continua. Travis será el encargado de automatizar toda la construcción cuando se produzcan nuevos cambios sobre el proyecto.

Cada vez que un cambio se sube a la rama develop y master el código es construido por Travis usando como base de datos MariaDB y la versión 3.5 de Python. Dicha configuración se especifica en el archivo `.travis.yml` de la raíz del proyecto y especifica los pasos que Travis realizará internamente para llevar a cabo la construcción del proyecto. Las librerías necesarias para la compilación del proyecto están especificadas en el archivo `requirements.txt`.

Travis además nos avisa cuando la construcción y las pruebas del proyecto se han realizado y pasado correctamente. En caso positivo y estar en la rama master pasaría al

despliegue. En caso negativo podemos acceder al enlace de la build en Travis, donde aparecerá el problema que ha realizado que la construcción del proyecto fallara.

Ejemplo de integración continua de la rama develop

Proyecto-EGC-G1 / Autenticacion-EGC-G1  build passing

Current Branches Build History Pull Requests > Build #41 More options

✓ develop Arregla fallo sobre las cookies a través de los dominios -> #41 passed

Commit 71b9879 [↗](#) Ran for 1 min 43 sec
Compare cf391bb..71b9879 [↗](#) about 17 hours ago
Branch develop [↗](#)
Alejandro Hall authored and committed

Job log View config

```
1 Worker information worker_info
6 mode of '/usr/local/clang-5.0.0/bin' changed from 0777 (rwxrwxrwx) to 0775 (rwxrwxr-x)
7 Build system information system_info
414
415 removed '/etc/apt/sources.list.d/basho-riak.list'
416 Executing: /tmp/tmp.tbDvx7S0mK/gpg.1.sh --keyserver
417 hkp://keyserver.ubuntu.com:80
```

Ejemplo de la cronología de las dos ramas con integración continua

Proyecto-EGC-G1 / Autenticacion-EGC-G1  build passing

Current Branches Build History Pull Requests More options

Default Branch

✓ master	# 42 passed	c72547d ↗	✓	✓	✓	✓	✓
6 builds	about 17 hours ago	Alejandro Hall					

Active Branches

✓ develop	# 41 passed	71b9879 ↗	✓	✓	✓	✓	✓
29 builds	about 17 hours ago	Alejandro Hall					

Gestión de la liberaciones, despliegues y entregas

Todo lo relacionado con el despliegue y entregables da comienzo una vez que el código ha sido lo suficientemente testeado y aprobado para su liberación siguiendo una serie de requisitos que se definen en el proceso. A su vez, el despliegue se dará por concluido una vez haya sido correctamente distribuido en un entorno de producción.

Requisitos para comenzar el despliegue:

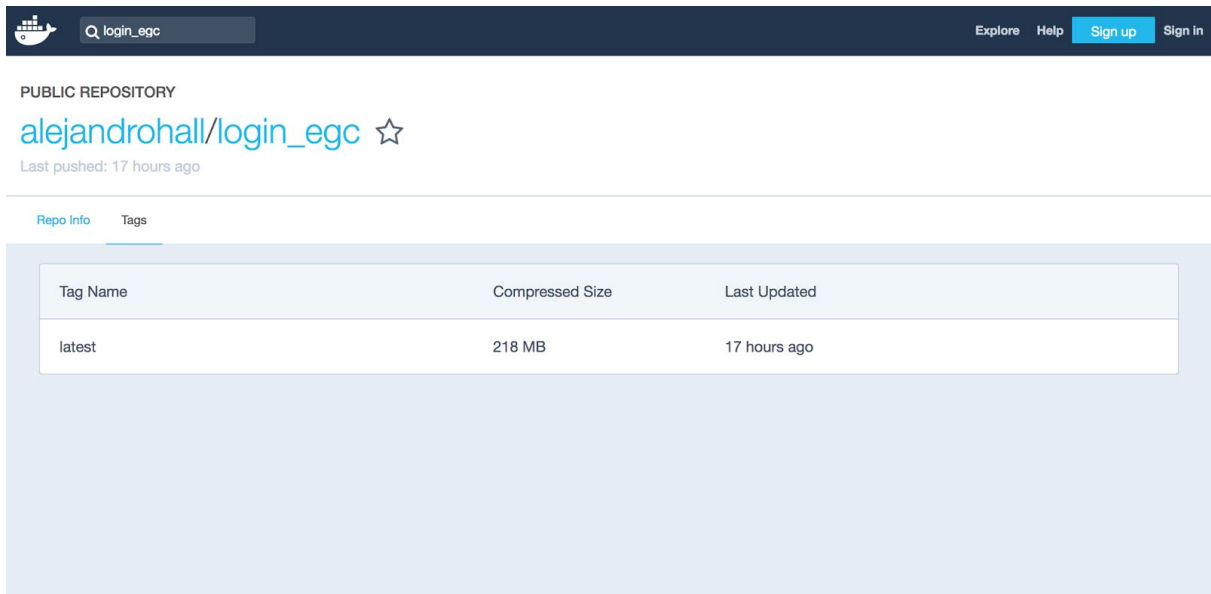
- El proyecto no genera error en la compilación.
- Las pruebas se han realizado y ejecutado correctamente.
- La imagen Docker se ha generado correctamente.

Las herramientas que usamos para el proceso se explican a continuación:

En cada milestone se genera una nueva release con su tag en Git, lo que nos permite conocer el estado del proyecto en cada Milestone pasada. A su vez, para el despliegue del proyecto, es necesario que Travis lo haya construido y pasado los tests. En caso positivo, y siempre que los cambios se hayan realizado sobre master, se ejecuta en fichero `build_and_upload_docker_image.sh` que construye la imagen de Docker desde el repositorio.

Una vez se construye la imagen Docker con los últimos cambios se sube automáticamente a Docker Hub y finaliza la entrega. Se puede ver la imagen publicada en Docker Hub en la siguiente dirección: https://hub.docker.com/r/alejandrohalla/login_egc/tags/

Además, a través de una API que se encarga de, una vez que un grupo ha terminado de generar la imagen de Docker y subirla a Docker Hub, se le pasa a ese endpoint el nombre del grupo y Docker se encarga de tirar el contenedor específico y levantarlo con la imagen nueva generada anteriormente.



The screenshot shows a GitHub repository page for 'login_egc' by user 'alejandrohalla'. The repository is a public repository, last pushed 17 hours ago. The 'Tags' tab is selected, showing a table with one tag named 'latest', which has a compressed size of 218 MB and was updated 17 hours ago.

Tag Name	Compressed Size	Last Updated
latest	218 MB	17 hours ago

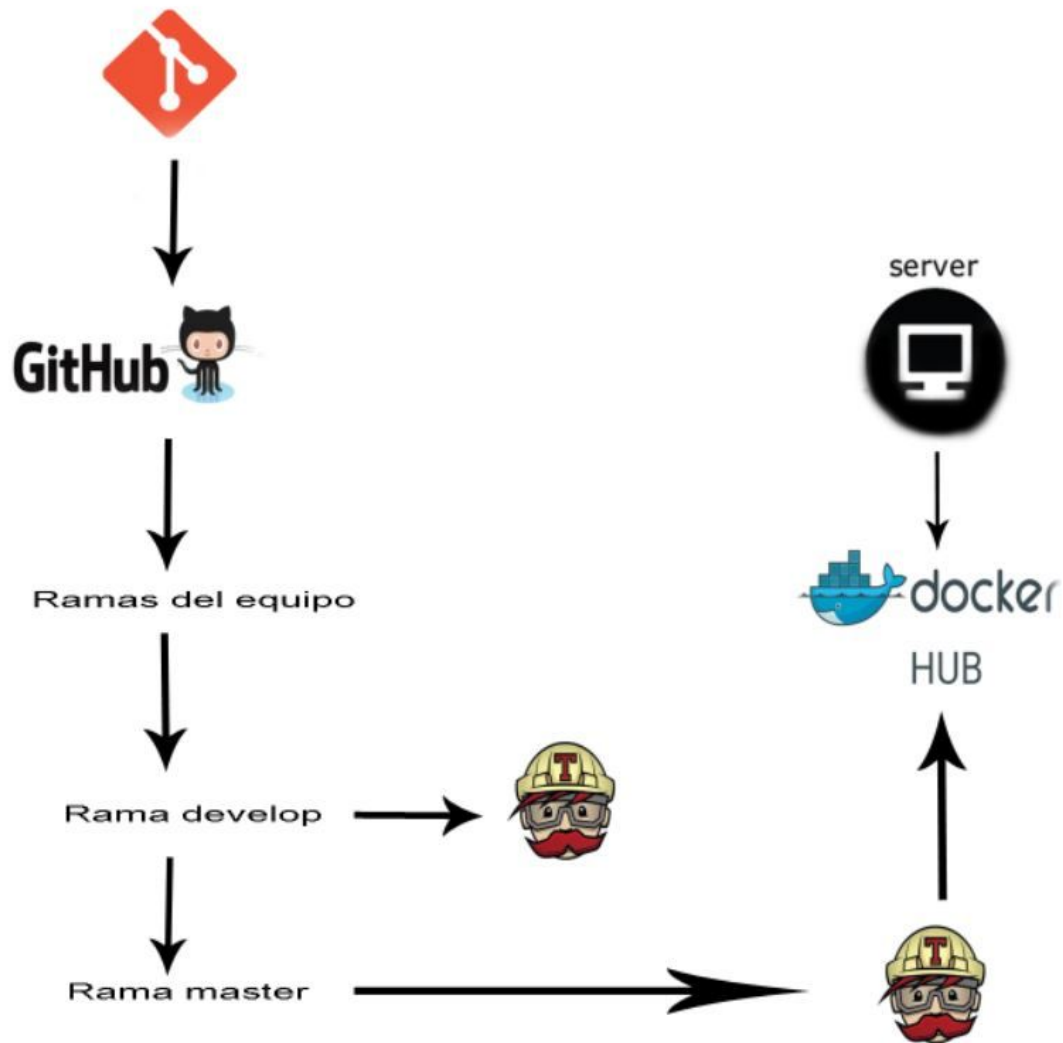
Mapas de herramientas

Hemos utilizado Git y GitHub para realizar la gestión de incidencias y el control de versiones. Travis como integración continua y DockerHub para alojar las imágenes de Docker generadas por la integración continua.

Nuestro flujo de trabajo es el siguiente:

1. El equipo trabaja en su propia rama y cuando se unen los cambios a la rama develop se comprueban los test con Travis CI.
2. Una vez se terminen de integrar todos los cambios a develop y no hay errores se hace un merge en la rama master.
3. Los cambios de la rama master se comprueban con Travis CI para asegurar que todo está correcto y si no hay errores se crea la imagen Docker que se sube a DockerHub. El servidor es el encargado de comprobar si existen imágenes nuevas del sistema y desplegarlas.

Herramientas de gestión de la configuración

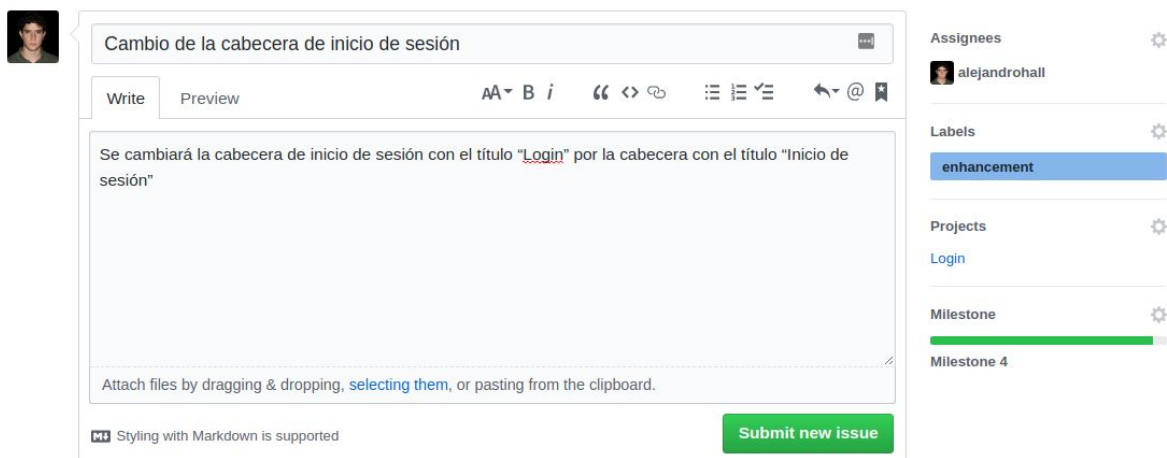


Ejercicio de propuesta de cambio

El ejercicio de cambio será simplemente cambiar una cabecera en el inicio de sesión, se cambiará la cabecera “Login” por la cabecera “Inicio de sesión”

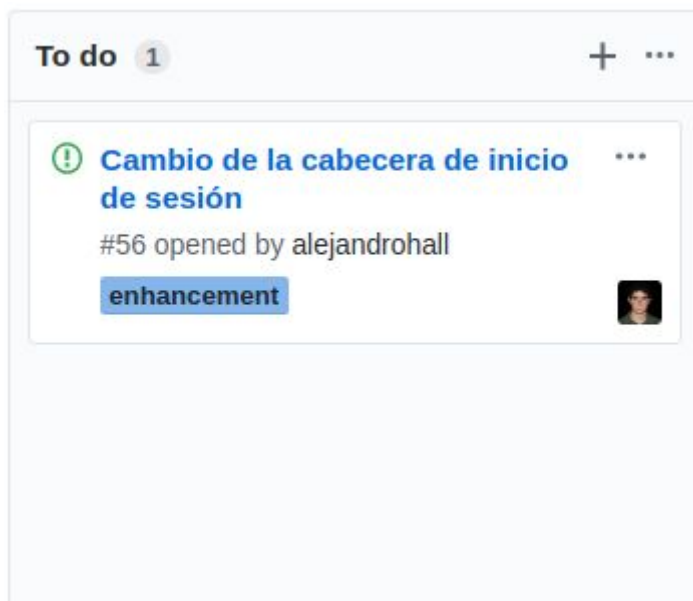
En primer lugar creamos la issue siguiendo el flujo que comentábamos arriba, con la siguiente información

- Título: Cambio de la cabecera de inicio de sesión
- Descripción: Se cambiará la cabecera de inicio de sesión con el título “Login” por la cabecera con el título “Inicio de sesión”
- Asignación: Alejandrohall
- Etiqueta: Enhancement (Mejor)
- Proyecto: Login
- Milestone: Milestone 4



The screenshot shows a Jira issue creation interface. On the left is a user profile picture. The main form has a title field containing 'Cambio de la cabecera de inicio de sesión'. Below the title are tabs for 'Write' and 'Preview'. The 'Write' tab is active, showing a rich text editor with the description: 'Se cambiará la cabecera de inicio de sesión con el título "Login" por la cabecera con el título "Inicio de sesión"'. The word 'Login' is underlined in red. Below the text area is a note: 'Attach files by dragging & dropping, selecting them, or pasting from the clipboard.' At the bottom left of the form is a small icon and the text 'Styling with Markdown is supported'. At the bottom right is a green button labeled 'Submit new issue'. To the right of the form is a sidebar with several sections: 'Assignees' with a user icon and name 'alejandrohalla'; 'Labels' with a blue button labeled 'enhancement'; 'Projects' with a blue link labeled 'Login'; and 'Milestone' with a green progress bar and the text 'Milestone 4'.

Una vez creada la issue y asignada a mi, se establece en la columna de “To do”, para señalar que esa tarea se está haciendo y tiene asignado una persona para ello.



Una vez que la persona se dedica a realizar el cambio, siempre tiene que trabajar sobre su rama. Después de que el cambio sea ejecutado ésta persona subirá éste a su rama, referenciando en el mensaje del commit a la issue anteriormente creada y siguiendo el formato del commit.

```
[alejandrohall@AntergosPC egc_login]$ git status
En la rama alejandro
Cambios no rastreados para el commit:
  (usa "git add <archivo>..." para actualizar lo que será confirmado)
  (usa "git checkout -- <archivo>..." para descartar los cambios en el directorio de trabajo)

    modificado:    login/templates/login.html

sin cambios agregados a la confirmación (usa "git add" y/o "git commit -a")
[alejandrohall@AntergosPC egc_login]$ git add -A
[alejandrohall@AntergosPC egc_login]$ git commit -m "Cambia cabecera de login. Ref #56"
[alejandro 0109576] Cambia cabecera de login. Ref #56
 1 file changed, 1 insertion(+), 1 deletion(-)
[alejandrohall@AntergosPC egc_login]$ git push origin alejandro
Contando objetos: 5, listo.
Delta compression using up to 4 threads.
Comprimiendo objetos: 100% (4/4), listo.
Escribiendo objetos: 100% (5/5), 441 bytes | 441.00 KiB/s, listo.
Total 5 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:Proyecto-EGC-G1/Autenticacion-EGC-G1.git
 71b9879..0109576  alejandro -> alejandro
[alejandrohall@AntergosPC egc_login]$
```

Cambio de la cabecera de inicio de sesión #56

Open alejandrohall opened this issue 7 minutes ago · 0 comments



alejandrohall commented 7 minutes ago

Owner



Se cambiará la cabecera de inicio de sesión con el título "Login" por la cabecera con el título "Inicio de sesión"



alejandrohall added the **enhancement** label 7 minutes ago



alejandrohall added this to the **Milestone 4** milestone 7 minutes ago



alejandrohall self-assigned this 7 minutes ago



alejandrohall added this to **To do** in **Login** via **automation** 7 minutes ago



alejandrohall added a commit that referenced this issue just now



Cambia cabecera de login. Ref #56

0109576

Con el cambio ejecutado, la persona será responsable de mover su tarea de "To Do" a "Need revision", con el objetivo de que el coordinador sepa que hay un cambio nuevo a revisar.

Need Revision 2 + ...

Cambio de la cabecera de inicio de sesión ...
#56 opened by alejandrohall
enhancement

Mejorar el captcha ...
#51 opened by Garbancito
feature

El coordinador al percatarse que hay una tarea que necesita revisión se encargará de revisar el código, si el código no es aceptado, éste dará sus razones y propondrá la corrección del mismo. Si el cambio es aceptado, se situará en la rama develop con el objetivo de integrar el código de la rama de la persona que ha realizado el cambio en la rama develop.

Si surgiera algún tipo de conflicto se solucionarían, sino se subiría a la rama de develop.

```
[alejandrohalla@AntergosPC egc_login]$ git checkout develop
Cambiado a rama 'develop'
[alejandrohalla@AntergosPC egc_login]$ git merge alejandro
Actualizando 71b9879..0109576
Fast-forward
 login/templates/login.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
[alejandrohalla@AntergosPC egc_login]$ git push origin develop
Total 0 (delta 0), reused 0 (delta 0)
To github.com:Proyecto-EGC-G1/Autenticacion-EGC-G1.git
 71b9879..0109576 develop -> develop
[alejandrohalla@AntergosPC egc_login]$
```

Al subirlo a la rama develop, la integración continua saltará, ejecutando los tests en remoto. Si los tests han fallado Travis avisa al correo, y se investiga para ver qué ha generado el fallo y volver a solucionarlo, sino falla todo sigue para adelante.

Una vez subido a develop, la issue pasa a estado 'Done'

Proyecto-EGC-G1 / Autenticacion-EGC-G1 build passing

[Current](#) [Branches](#) [Build History](#) [Pull Requests](#)

 **develop** Cambia cabecera de login. Ref #56

 #43 started

 Commit 0109576 

 Running for 22 sec

 Compare 71b9879..0109576 

 Branch develop 

 Alejandro Hall authored and committed

 This job is running on our [Trusty](#) environment, which has been updated on Tuesday, December 12th, 2017. Read all about this [on our blog](#) and take [.travis.yml](#) file to use the previous image version.

[Job log](#)

[View config](#)

```
▶ 1 Worker information
6 mode of '/usr/local/clang-5.0.0/bin' changed from 0777 (rwxrwxrwx) to 0775 (rwxrwxr-x)
▶ 7 Build system information
414
415 removed '/etc/apt/sources.list.d/basho_riak.list'
416 Executing: /tmp/tmp.xweUCw8yim/gpg.1.sh --keyserver
417 hkp://keyserver.ubuntu.com:80
418 --recv
419 0C49F3730359A14518585931BC711F99BA15703C6
420 gpg: requesting key A15703C6 from hkp server keyserver.ubuntu.com
421 gpg: key A15703C6: "MongoDB 3.4 Release Signing Key <packaging@mongodb.com>" 1 new signature
422 gpg: Total number processed: 1
423 gpg:          new signatures: 1
```


Proyecto-EGC-G1 / Autenticacion-EGC-G1 build passing

[Current](#) [Branches](#) [Build History](#) [Pull Requests](#)

✓ **develop** Cambia cabecera de login. Ref #56

👉 #43 passed

➡ Commit 0109576 

🕒 Ran for 1 min 20 sec

🔗 Compare 71b9879..0109576 

📅 less than a minute ago

🌿 Branch develop 

👤 Alejandro Hall authored and committed

[Job log](#)

[View config](#)

```
1 Worker information
6 mode of '/usr/local/clang-5.0.0/bin' changed from 0777 (rwxrwxrwx) to 0775 (rwxrwxr-x)
7 Build system information
414
415 removed '/etc/apt/sources.list.d/basho_riak.list'
416 Executing: /tmp/tmp.xweUCW8yIm/gpg.1.sh --keyserver
417 hkp://keyserver.ubuntu.com:80
418 --recv
419 0C49F3730359A14518585931BC711F9BA15703C6
420 gpg: requesting key A15703C6 from hkp server keyserver.ubuntu.com
421 gpg: key A15703C6: "MongoDB 3.4 Release Signing Key <packaging@mongodb.com>" 1 new signature
422 gpg: Total number processed: 1
423 gpg:      new signatures: 1
424 W: http://ppa.launchpad.net/couchdb/stable/ubuntu/dists/trusty/Release.gpg: Signature by key 15866BAFD9BCC4F3C1E0DFC7D69548E1C17EAB57 uses weak digest
425
426 127.0.0.1 localhost nettuno travis vagrant
427 127.0.1.1 travis-job-8c6f1eab-8bd1-4f18-9049-bfa9a0947944 travis-job-8c6f1eab-8bd1-4f18-9049-bfa9a0947944 ip4-loopback trusty64
428
429 3.5 is not installed; attempting download
430 Downloading archive: https://s3.amazonaws.com/travis-python-archives/binaries/ubuntu/14.04/x86_64/python-3.5.tar.bz2
431 $ sudo tar xjf python-3.5.tar.bz2 --directory /
432 $ git clone --depth=50 --branch=develop https://github.com/Proyecto-EGC-G1/Autenticacion-EGC-G1.git Proyecto-EGC-G1/Autenticacion-EGC-G1
433 $ sudo service mariadb start
443
```

Cuando ocurran una serie de cambios y acabe el milestone, todo lo integrado en develop se pasará a master y se subirá al repositorio.

```
[alejandrohalla@AntergosPC egc_login]$ git checkout master
Cambiado a rama 'master'
Tu rama está actualizada con 'origin/master'.
[alejandrohalla@AntergosPC egc_login]$ git merge develop
Merge made by the 'recursive' strategy.
 login/templates/login.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
[alejandrohalla@AntergosPC egc_login]$ git push origin master
Contando objetos: 2, listo.
Delta compression using up to 4 threads.
Comprimiendo objetos: 100% (2/2), listo.
Escribiendo objetos: 100% (2/2), 290 bytes | 290.00 KiB/s, listo.
Total 2 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:Proyecto-EGC-G1/Autenticacion-EGC-G1.git
   c72547d..ffacae6  master -> master
[alejandrohalla@AntergosPC egc_login]$
```

Una vez subido todo a master la integración continua volverá a saltar, pero ésta vez generará la imagen de Docker.

```
1262 Login Succeeded
1263 The push refers to a repository [docker.io/alejandrohalla/login_egc]
1264 b9551ea6aaec: Preparing
1265 53633e3d1326: Preparing
1266 ae0047294fd8: Preparing
1267 a55a876d47c2: Preparing
1268 d063213f0854: Preparing
1269 9ca4cadaa149: Preparing
1270 6fabdb937ee9: Preparing
1271 24294e86c343: Preparing
1272 f17fc24fb8d0: Preparing
1273 6458f770d435: Preparing
1274 5a876f8f1a3d: Preparing
1275 d2f8c05d353b: Preparing
1276 48e0baf45d4d: Preparing
1277 9ca4cadaa149: Waiting
1278 6fabdb937ee9: Waiting
1279 24294e86c343: Waiting
1280 f17fc24fb8d0: Waiting
1281 6458f770d435: Waiting
1282 5a876f8f1a3d: Waiting
1283 d2f8c05d353b: Waiting
1284 48e0baf45d4d: Waiting
1285 a55a876d47c2: Pushed
1286 9ca4cadaa149: Pushed
1287 b9551ea6aaec: Pushed
1288 6fabdb937ee9: Pushed
1289 f17fc24fb8d0: Layer already exists
1290 53633e3d1326: Pushed
1291 6458f770d435: Layer already exists
1292 d2f8c05d353b: Layer already exists
1293 5a876f8f1a3d: Layer already exists
1294 48e0baf45d4d: Layer already exists
1295 24294e86c343: Pushed
1296 d063213f0854: Pushed
1297 ae0047294fd8: Pushed
1298 latest: digest: sha256:0bba222077b958d640a47fbd429a65b56f871b5bd0d47006bdfc1bc2a5137d99 size: 3037
1299
1300
1301 The command "bash build_and_upload_docker_image.sh" exited with 0.
1302
1303 Done. Your build exited with 0.
```

PUBLIC REPOSITORY

alejandrohalla/login_egc ☆

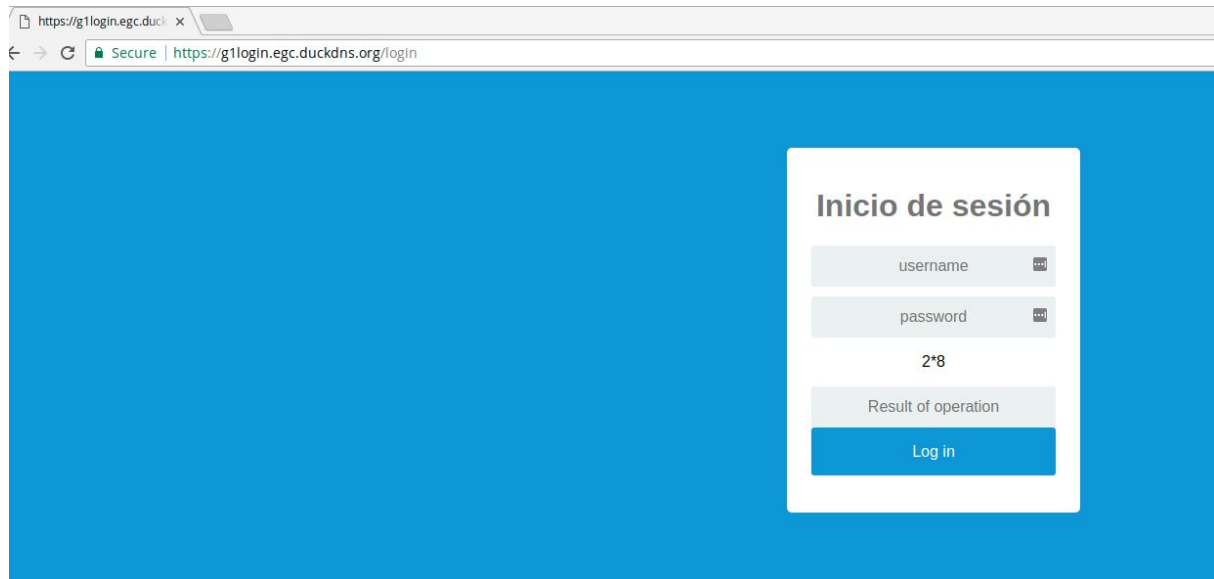
Last pushed: a few seconds ago

Repo Info

Tags

Tag Name	Compressed Size	Last Updated
latest	218 MB	a few seconds ago

Una vez que el servidor detecte que hay una imagen nueva de nuestro subsistema, éste se la descargará y desplegará automáticamente.



Conclusiones y trabajo futuro

Conclusiones

En general el trabajo ha salido bien pues a pesar de los problemas de organización que hubo al principio se ha hablado por los grupos de telegram y realizado reuniones para aclarar las relaciones entre los sistemas de cada grupo.

Creemos que nuestra parte de Autenticación del proyecto-EGC-G1 ha ido bastante bien pues hemos terminado todas nuestras tareas en las fases tempranas del proyecto e incluso hemos propuesto mejoras en el proyecto como la generación del id de la cookie segura para que no fuese un número de escaso tamaño.

Mejoras

Algunas de las posibles mejoras que podrían realizarse para el próximo curso serían las siguientes:

- Que los grupos dispongan de una máquina virtual con todos los programas necesarios ya instalados pues permitiría ahorrar muchísimo tiempo a todos.
- Cambiar el captcha por uno de google ya que es más cómodo pues solo necesita un click.
- Invertir más tiempo en la planificación de como va a estar estructurado el sistema con clases y relaciones, pues se ha visto que algunos grupos no sabían como interactuaba su sistema con otros.
- Añadir más test de forma que se cubran todas las funcionalidades.