Decide part-rota-3

-Grupo:3

-Curso escolar: 2022/2023

-Asignatura: Evolución y gestión de la configuración

Miembros del grupo

Nombre	Implicación
Alfonso Cadenas Morales	10
Abraham Cobelo Galindo	10
Jose Manuel Martín Luque	10
Jesús Martín Vergara	10
Miguel Ángel Romalde Dorado	10

Enlaces de interés

·Repositorio del grupo: https://github.com/Decide-Part-Rota/decide-part-rota-3

·Repositorio común: https://github.com/Decide-Part-Rota/decide-part-rota-main

·Link de la aplicación desplegada: http://decidepartrota.pythonanywhere.com/

Indicadores del proyecto

Miembro del equipo	Horas	Commits	LoC	Test	Issues	Incremento
Alfonso Cadenas Morales	40	34	302	6	4	Formulario registro y login con Github.
Abraham Cobelo Galindo	40	18	115	5	4	Formulario de registro, vista perfil, editar perfil, algunos tests
Jose Manuel Martín Luque	40	12	469	7	4	Implementación de allauth para autenticar usuarios a través de Oauth. Implementación de Oauth de Google. Formulario de completación de información para los usuarios autenticados por Oauth.
Jesús Martín Vergara	40	17	258	5	4	Formulario de login, login con email, validación registro, tests de registro y login,
Miguel Ángel Romalde Dorado	40	16	208	6	4	Login con email, email de verificación al hacer login, login por facebook con oauth2, tests
TOTAL	200	97	1352	25	8	

Integración con otros equipos

Se ha integrado el proyecto junto a otros dos equipos, por ello es un proyecto 'part'.

Decide-part-rota-1: Integración del módulo de censo.

Decide-part-rota-2: Integración de módulo de votaciones.

Decide-part-rota-3: Integración del módulo de autenticación.

Resumen ejecutivo

Tenemos una aplicación base llamada decide, la cual es un sistema de votación, se han propuesto varías formas de hacer un incremento en dicha aplicación, en concreto en nuestro proyecto hemos elegido el incremento de autenticación el cual se basa en tener un control de usuarios los cuales pueden acceder a unas vistas y votaciones del proyecto, somos un subgrupo llamado decide-part-rota3 de un grupo mayor llamado decide-part-rota en el que estamos con dos subgrupos más, uno se encarga del censo y otro de las votaciones.

Esto es importante dado que necesitamos controlar que no todo el mundo pueda entrar a votar, y que solo voten los usuarios que están registrados en la aplicación por ello hemos hecho un registro y login de usuarios.

El login se puede hacer de varias formas, por ejemplo, puedes registrarte y usar el nombre de usuario y la contraseña, o en vez del nombre de usuario usar el mail correspondiente. También hemos implementado login por varias redes sociales, como son Google, Facebook y Github, hemos ampliado el login normal, dado que hay un desarrollo de internet y de redes sociales últimamente, la gente usa más estas plataformas, y dado que no todo el mundo quiere rellenar un formulario con sus correspondientes datos y no rellenar 10 cuadros, buscan una mayor rapidez y agilidad, esto nos lo da los diferentes logins a la aplicación que hemos implementado, además, al crear el usuario creamos una clase persona para poder tener más datos que usaran nuestros grupos asociados.

Para la implementación de estos logins de las redes sociales hemos utilizado una tecnología llamada allauth que nos permite utilizar oauth que es una solución de administración de identidad y hay que remarcar que, aunque estos logins son muy útiles y rapidos, no nos dan todos los datos que nos piden desde los otros grupos, por ello, la primera vez que nos logeamos en la aplicación mediante dichos logins, nos lleva a un pequeño formulario en el cual le pedimos al usuario que se está registrando que indique los datos necesarios para crear dicha clase persona hablada anteriormente.

También hemos de añadir que, hemos implementado una verificación por email, en la cual, aunque te registres, hasta que no verifiques que eres tú realmente, no puedes entrar a nuestra aplicación.

Nuestra participación es importante dado que, si no se tiene un control de usuarios, es posible que hubiera fraudes en los votos, implementando bots para votar de forma continua y esto es impensable en una plataforma de votos, por eso necesitamos la seguridad y la verificación de saber que es una votación legitima.

·Descripción del sistema:

Se ha trabajado sobre un proyecto base llamado Decide, el mismo es una aplicación diseñada para crear votaciones y que los usuarios puedan participar en las mismas.

Cada equipo de desarrollo ha integrado un módulo que incluye ciertas funcionalidades.

En nuestro caso implementamos el módulo de autenticación. Nuestro paquete, como el del resto de módulos, posee unos archivos principales llamados *'urls'* donde se registran los enlaces y qué funciones se llevarán a cabo si accedemos a ellos, *'views'*, en él se encuentras dichas funciones, así como la implementación de los formularios, que están en 'forms.py 'y una carpeta llamada '*templates'* donde se encuentran todas las plantillas usadas para la visual de la parte que nos corresponde.

Principalmente se ha tenido relación con el equipo encargado de implementar las votaciones, ya que, si ellos implementaban una función para votar, la misma requería poder comprobar que la persona que vote lo estuviese haciendo desde una cuenta verificada en **Decide**. Como caso llevado a cabo fue el de implementar al modelo de persona el campo *'discord account'*, que registra el nombre de una cuenta de *discord*, siguiendo un patrón determinado, en este caso, el patrón correcto de una cuenta de *discord* es: '.+#\d{4}'.

Primero se implementó dicho campo en el formulario de registro, y, posteriormente, en la pestaña 'profile', dado que podían existir cuentas sin ese valor que quisiesen usarlo. Una vez implementado esto, los usuarios pueden efectuar votaciones mediante discord, y que éstas queden asignadas a su cuenta de decide.

Lo primero en implementar fue un formulario de registro de usuarios, que recogía todos los datos necesarios para crear un modelo *'User'* en Django. El formulario se amplió dado que creamos un modelo *'Person'*, que extendía el de *User*, con datos necesarios para nuestro módulo o el de los otros equipos. Dichos modelos y formulario se fueron actualizando a medida que se desarrollaba el proyecto, adaptándose a los requerimientos que resultaran necesarios.

La siguiente funcionalidad importante fue la confirmación de email, esto sería una ampliación del proceso de registro por formulario. Una vez se termina el formulario de registro, se enviará un correo de verificación al correo que aparezca en el formulario, y el usuario no podrá iniciar sesión en decide hasta que dicho correo de verificación haya sido abierto y se haya clicado en verificar.

Se implementaron más formas de iniciar sesión, la primera, usando tu propia cuenta de Google. Aquí no habría que rellenar ningún formulario, y la autenticación se da en el momento en el que inicias la sesión con Google, ya que estás usando una cuenta verificada, luego siempre habría que hacer *log in* con la misma cuenta, sin necesitar "registrarla" en ningún momento. De igual forma, se habilitaron el inicio de sesión mediante el uso de cuentas de *facebook* y de *github.* Por último, para usuarios que hayan iniciado sesión, se creó una vista *'profile'* que muestra los datos de la cuenta

que estén usando, dando la posibilidad de editar tanto la edad como el *nick* de la cuenta de *discord* asociado al usuario.

Además, en el archivo de tests, se han ido añadiendo funciones que comprobaban el correcto funcionamiento de todo lo que se haya ido añadiendo al módulo.

·Visión global del proceso de desarrollo

Para desarrollar el proyecto se ha partido del repositorio de **Decide**, que ya existía. Creamos un repositorio común con los otros equipos en *github* con una copia de este y cada grupo de trabajo hace repositorios aparte para sus propios módulos.

De esta manera se posee un acceso común a la información necesaria para trabajar.

Una vez creado, clonamos dicho repositorio en nuestros ordenadores en local y los editamos haciendo uso del *Vs code studio*.

Una vez configurado el proyecto en local se definen los *issues. E*l equipo asigna a las parejas de miembros para dichas *issues*. Posteriormente, para cada tarea se crea una rama, partiendo de la rama *develop*, donde se lleva a cabo la tarea y se completa.

Una vez terminada la tarea se suben los cambios a la rama online y se crea una *pull request*, a la rama *develop*.

Una vez todas las *issues* están completas se hace una *pull request* desde *develop* a *main* y se crea así la versión final y estable de la aplicación con el módulo implementado.

Para la gestión de cambios el proceso es simple, se crea una tarea que recoja los datos y los cambios a hacer sobre el producto, se asigna a una o varias personas, y se trabaja siguiendo el mismo procedimiento que para cualquier otra tarea. Primero se crea una rama nueva, se trabaja sobre la misma en la tarea, una vez terminada se crea una pull request a develop, corrigiendo los errores que puedan surgir. Por último, los cambios son llevados a la rama main, de donde pasarán al repositorio principal en reuniones con los otros equipos, vigilando que no haya errores ni pérdidas de datos.

·Entorno de desarrollo

Como entorno de desarrollo se ha usado el *Vs Code Studio*, en su última versión, y actualizando siempre que alguna nueva sea lanzada. Para instalar el sistema en local, se debe hacer uso del comando '*git clone*', poniendo la *url* del repositorio que se quiera importar, ya sea rota 1,2,3 o *main*, donde se hayan los tres.

Una vez clonado debemos, teniendo *python* instalado en nuestro equipo, abrir una terminal en la raíz del proyecto y ejecutar el siguiente comando: *'pip install –r requirements.txt*', de esta forma instalaremos todas las librerías necesarias para que funcione la web. Hecho esto, ejecutamos '*cd decide*', para entrar en dicho directorio, ejecutamos el comando 'p*ython manage.py makemigrations*' primero, y luego 'p*ython manage.py migrate*', para crear una base de datos necesaria para la ejecución de Decide. Por último, sin cambiar de directorio, ejecutamos el comando *'python manage.py runserver'*, de forma que Decide empezará a ejecutarse en local, en el puerto marcado en la terminal, en nuestro caso sería: 127.0.0.1:8000 o localhost:8000.

Para docker, hay que irse a la carpeta docker 'cd docker', crear la imagen con 'docker compose build', y lanzar los contenedores con 'docker compose up –force-recreate'. El '—force-recreate' se asegura de borrar los contendores antiguos antes de lanzar los nuevos.

·Ejercicio de propuesta de cambio

El equipo interesado en realizar un cambio que nos corresponde implementar lo notifica mediante el grupo común de whatsapp, y crea una tarea en su tablón de github con los datos del cambio que hay que realizar. Nuestro equipo crea una tarea en el tablón propio de github que especifica los cambios a hacer sobre el producto tomando en cuenta los cambios solicitados por el otro equipo en su tarea y se asigna a una o dos personas. Se crea una rama con el siguiente identificador de la misma correspondiente. Se trabaja sobre esa rama hasta tener lista la tarea. Una vez los cambios están implementados, antes de pasarlos a develop, los que han hecho la tarea se ponen en contacto con el otro equipo para enseñar las novedades, y cambiar algo si es necesario. Una vez se tiene el visto bueno por parte del otro equipo, se crea la pull request a develop, y de esta forma estos cambios serán llevados posteriormente a master y al repositorio común.

·Conclusiones y trabajo futuro

La aplicación ha acabado siendo estable y teniendo todas las funcionalidades implementadas. Sin embargo, como mejora a futuro, el equipo coincide en que lo que más necesita Decide es una mejora visual. Añadir un mínimo de css al proyecto, para que no todo sea un fondo blanco con campos por encima, además, no estaría de más añadir textos de ayuda general para guiar al usuario. La creación de vistas como, por ejemplo, políticas de privacidad, o una página de inicio con más información también son cambios que sumarían mucho al producto final.