

Mejora de accesibilidad

Cabina de autenticación

Grupo 2 EGC Curso 16-17

Contenido

[Documentación del proyecto 1](#_Toc470606697)

[Descripción 1](#_Toc470606698)

[Equipo de desarrollo 1](#_Toc470606699)

[Roles 1](#_Toc470606700)

[Comunicación 2](#_Toc470606701)

[Comunicación externa 2](#_Toc470606702)

[Comunicación interna 2](#_Toc470606703)

[Gestión de las incidencias 2](#_Toc470606704)

[Gestión del código 3](#_Toc470606705)

[Entorno de desarrollo 4](#_Toc470606706)

[Material de referencia 4](#_Toc470606707)

Documentación del proyecto

# Descripción

Nuestro objetivo se centrará en la parte de autenticación y registro. Para ello, nuestro principal desarrollo será, previsiblemente, en el repositorio de agora-gui-common, el cual está desarrollado principalmente en Javascript con AngularJS.

Usaremos ARIA, la cual define maneras de hacer el contenido Web y las aplicaciones Web (especialmente las desarrolladas con Ajax y JavaScript) para que sean más accesibles para las personas con discapacidad.

# Equipo de desarrollo

Aurora Gómez Medina [aurgommed@alum.us.es](mailto:aurgommed@alum.us.es)  
Eduardo Ger Rodríguez [edugerrod@alum.us.es](mailto:edugerrod@alum.us.es)  
Jorge Ramos Rivas [jorramriv@alum.us.es](mailto:jorramriv@alum.us.es)  
José Luis Maya Quirós [josmayqui@alum.us.es](mailto:josmayqui@alum.us.es)  
Sara Chávez García [sarchagar1@alum.us.es](mailto:sarchagar1@alum.us.es)

# Roles

Eduardo Ger Rodríguez (Coordinador/Desarrollador)  
Sara Chávez García (Subcoordinador/Desarrollador)  
Aurora Gómez Medina (Desarrollador)  
José Luis Maya Quirós (Desarrollador)  
Jorge Ramos Rivas (Desarrollador)

# Comunicación

## Comunicación externa

Las reuniones con el equipo de Ágora Voting han tenido lugar usando el editor de textos colaborativo TitanPad: <https://titanpad.com/KxlHuhjhcZ>

Y mediante el uso de [Mumble](http://www.mumble.com/), gestor de salas de conferencias online.

## Comunicación interna

Usaremos WhatsApp como herramienta principal y Teamviewer, además de las reuniones presenciales.

# Gestión de las incidencias

Para la gestión de incidencias nuestro grupo ha decidido gestionarlas a través de la creación de issues con GitHub. A cada incidencia se le asignará una prioridad, que podrá ser alta, neutra o baja. El proceso para la creación de una incidencia es el siguiente:

* Un miembro del grupo crea una incidencia con el estado inicial "Pendiente". Dicho miembro debe subir adjunto el archivo más adecuado para describir la incidencia. Podrán usarse imágenes, texto, enlaces a código GitHub, etc.
* El miembro que ha creado la incidencia comunica al grupo mediante Whatsapp que acaba de crear una incidencia.
* Otro miembro del grupo, cuando ha sido notificado de la incidencia, cambia el estado de la misma a "En proceso" y se pone a trabajar sobre ella.
* Una vez que el miembro ha resuelto los problemas que definían la incidencia, cambia de nuevo su estado a "Resuelta".
* Antes de cambiar finalmente el estado a "Cerrada", al menos un miembro del grupo debe verificar que la incidencia se ha resuelto correctamente.
* Eventualmente, si una incidencia ha pasado a "Cerrada" y aún quedan problemas de esa incidencia por solventar, podrá pasar a un estado intermedio "Reabierta" para luego volver a pasar al estado "Pendiente".
* Una incidencia que esté en estado "Pendiente" puede pasar directamente al estado "Cerrada", bien porque se haya abierto erróneamente, bien porque el problema que ha ocasionado la incidencia no sea suficientemente grave como para crear una incidencia, o bien porque el problema es imposible de resolver.
* De igual manera, una incidencia que está "En proceso" puede pasar al estado "Cerrada" también directamente, por ejemplo porque una vez que alguien del grupo se ha puesto a trabajar sobre esa incidencia, no ha podido conseguir solucionar los problemas de esa incidencia.

# Gestión del código

Se ha realizado un fork de cada uno de los repositorios de AgoraVoting en GitHub a nuestra organización. Cada miembro del grupo debe hacer un git clone de los repositorios a su máquina, de manera que cada uno trabaja en su máquina en local, haciendo commit cada vez que sea necesario. Si el miembro logra obtener cambios estables, notificará al grupo mediante Whatsapp que va a realizar un git push, pero antes de hacerlo, al menos un miembro del grupo debe revisar que el cambio funciona correctamente. Una vez que se ha hecho git push con el cambio, hay que volver a desplegar AgoraVoting con dicho cambio. Cada vez que se haga un push, se deben documentar y describir los nuevos cambios. Estos documentos generados de los cambios se subirán tanto a la wiki del grupo como a nuestro Google Drive.

# Entorno de desarrollo

En nuestro entorno de desarrollo estamos usando una máquina virtual basada en Ubuntu, versión 16.04 LTS, que es gestionada por el gestor de máquinas virtuales VMware. La máquina virtual ha sido creada en VMware a partir de una imagen del sistema operativo mencionado en formato .iso, el cual hemos descargado desde [aquí](https://atlas.hashicorp.com/).

Con la máquina creada, se ha procedido a desplegar la aplicación AgoraVoting, lo cual es necesario para realizar los cambios. Para ello, hemos seguido los pasos que indica el equipo de AgoraVoting [aquí](https://github.com/agoravoting/agora-dev-box/blob/next/doc/devel_deployment.md), aunque con versiones diferentes que las indicadas por ellos. Concretamente, hemos usado las siguientes versiones de cada componente:

* Oracle VirtualBox v5.1.10
* Vagrant v1.9.0
* Ansible v2.1.3.0

Después de seguir el proceso indicado anteriormente, dentro de VirtualBox se crean tres máquinas virtuales Ubuntu 14.03 que actúan como servidores, necesarias para el despliegue de la aplicación. El resultado es entonces una máquina virtual Ubuntu en VMware, y dentro de esta tres máquinas virtuales más de Ubuntu en VirtualBox.

# Material de referencia

* [Complete nVotes deployment system](https://github.com/agoravoting/agora-dev-box/blob/next/doc/devel_deployment.md)
* [Atlas by HashiCorp](https://atlas.hashicorp.com/)