AGORA VOTING

GRUPO 1 – CREACIÓN Y ADMINISTRACIÓN DE VOTACIONES

|  |  |
| --- | --- |
| Fecha: 26 Diciembre 2015 | Versión: 1.0 |

Autorización:

|  |  |  |  |
| --- | --- | --- | --- |
| Roles | Nombre | Firma(s) | Fecha entrega |
| Sponsor | -David Benavides Cuevas  -Pablo Neira Ayuso |  | 26 Diciembre 2015 |
| Jefe del Proyecto | -Aaron Valverde Villalón |  | 26 Diciembre 2015 |
| Coordinador del equipo | -David Gutierro Gallego  -Adrián Romero Núñez |  | 26 Diciembre 2015 |
| Equipo de proyecto (Desarrolladores) | -Adrián Romero  -David Gutierro Gallego  -Aaron Alejandro Valverde Villalón  -Daniel Pérez Rodríguez |  | 26 Diciembre 2015 |

Contenido del documento

[Autorización 1](#_Toc437912361)

[1. Resumen del proyecto 3](#_Toc437912362)

[2. Introducción 4](#_Toc437912363)

# Resumen del proyecto

El objetivo general del trabajo es que el grupo ponga en práctica y observe cómo se ponen en funcionamiento en un proyecto real todos los conceptos teórico-prácticos que se vean en la asignatura y profundice en ellos todo lo que su motivación lo lleve.

El grupo debe elegir un proyecto de los desarrolladores del curso anterior para su estudio, análisis, documentación y toma de decisiones. Esto es, el grupo debe proponer cuáles son los cambios que se van a introducir al proyecto. Cabe destacar que estos cambios implican añadir nuevas funcionalidades, cambio del lenguaje de programación, corrección de errores etc.

Para ser más exactos, los alumnos de 4º curso de Ingeniería Informática del Software, de la asignatura “Evolución y Gestión de la Configuración”, trabajaran sobre una aplicación para realizar votaciones, muy similar a “Agora Voting”, realiza por los alumnos del curso pasado (2014/2015). En ella deberán aplicar los posibles cambios, nombrados, sin limitaciones.

El contenido de este documento recogerá todo el contenido que se pide en la sección “4.1 Documento del proyecto”, el cual es un documento que se entrega al principio del curso y que sirve de guía. Estará compuesto por todos los aspectos que el grupo ha llevado a cabo para, como el nombre de la propia asignatura indica, la evolución y gestión de la configuración.

Cabe destacar que el subsistema elegido en la primera acta (creación de grupos) ha sido “Creación y administración de votaciones”.

Como apartados más importantes que debemos recoger en este documento son la ***gestión de código fuente***, en el cual se explicarán los procesos, técnicas y herramientas utilizadas (en nuestro caso utilizaremos Github), tanto entre los miembros del subgrupo como para integrar con el resto de subsistemas. La **gestión de la construcción e integración continua**, donde se construye e integra el proyecto a partir del código fuente (mediante Eclipse y Maven para la construcción) y Gradle para la integración continua. Utilizaremos la misma herramienta para la integración externa con el resto de subsistemas.

Para la **gestión de incidencias**, usamos Github y se explica más detalladamente a medida que transcurre este documento.

En cuanto a la **gestión de despliegue**, se llevó a cabo una propuesta con los demás subsistemas, en la que un coordinador de cada uno llevará a cabo la integración.

Para finalizar el documento, redactaremos una serie de lecciones aprendidas y conclusiones en las que se explique todo lo aprendido, los problemas que hemos tenido y sus posibles soluciones.

# Introducción

Como ya hemos mencionado antes, hemos optado por la opción de trabajar sobre un proyecto real realizado por alumnos del curso anterior(Agora-Voting), en nuestro caso el proyecto(subsistema) elegido, ha sido “Creación y administración de votaciones”.

“Agora Voting” es un sistema de software libre cuyo fin dar soluciones electorales de voto electrónico de forma segura, donde los clientes pueden votar tanto en: primarias y elecciones de candidatos, presupuestos participativos, referéndums y consultas, elecciones profesionales y sindicales y en universidades. Trabajan con clientes como: “PODEMOS”, “Concello de Lugo”, “UNED” y “COMISIONES OBRERAS”.

Gracias a la colaboración de nuestros profesores con los desarrolladores de “Agora-Voting” y así realizarse una charla por parte de uno de sus desarrolladores a los alumnos, pudimos ponernos en contexto con la aplicación y tener un mejor entendimiento sobre ella a la hora de trabajar sobre el proyecto realizado por el curso anterior.

El subsistema elegido por nosotros “Creación y administración de votaciones” está relacionado con los subsistemas de:

* Autenticación: para saber si un usuario está autenticado o no.
* Creación/Administración de censos: cuando se realiza una votación se muestra el identificador, nombre y fechas de la votación. Se termina de añadir un censo y al guardar se almacena la relación con la votación. En la lista de votaciones creadas, también se puede editar su censo.
* Cabina de votaciones: Cuando quieren responder a una votación, redireccionan a la URL: "vote/survey.do/id=XXX". Para devolver un JSON con la siguiente información: {"id":"XXX","title":"XXX","description":"XXX","startDate":"XXX","endDate":"XXX","questions":[{"id":"XXX","text":"XXX"},{"id":"XXX","text":"XXX"]}
* Visualización: Cuando quieren realizar visualizaciones, redireccionan a la URL: "vote/allSurveys.do/". Enviamos un JSON con la lista de todas las votaciones: {{"id":"XXX","title":"XXX","description":"XXX","startDate":"XXX","endDate":"XXX","questions":[{"id":"XXX","text":"XXX"},{"id":"XXX","text":"XXX"]}, {"id":"XXX","title":"XXX","description":"XXX","startDate":"XXX","endDate":"XXX","questions":[{"id":"XXX","text":"XXX"},{"id":"XXX","text":"XXX"]},...}
* Deliberación: cuando se confirma que una votación ha sido creada, redireccionamos a una URL transmitiendo el nombre de la votación.
* Verificación: cada votación tendrá una clave pública y otra privada.

Para la gestión de código fuente, hemos definido un proceso en el cual se muestra paso a paso la forma de gestionar el código de nuestro subsistema. Hemos usado directamente los repositorios “Git” en vez de usar un repositorio “SVN” en ProjETSII ya que Git dispone de un modelo distribuido, permitiendo realizar “commits” en local sin necesidad de tener un repositorio central.

Además, existe una plataforma más rápida y con menor peso de los repositorios (GitHub). Por esto, creamos un repositorio en Github en el cual subimos el código fuente de nuestro proyecto, a parte de los documentos que creíamos oportunos, haciendo “commits” de los cambios que realizamos. En la gestión de ramas, al crear un repositorio, también se crea una rama principal o “master” donde se encontrará la versión más fiable del software y creamos otra rama secundaria para subir los demás archivos. Hicimos diferentes pruebas en la implementación en otras ramas, y si estás eran satisfactorias se unirían a la rama de la que surgió.

El repositorio creado es compartido por nuestro grupo de trabajo (Creación y administración de votaciones) y por los grupos de los demás subsistemas. La URL del repositorio es: URL

Por otro lado, para la gestión de la construcción desde una primera reunión acordamos usar Maven para construir nuestro proyecto. Además de haberla usado anteriormente en otras asignaturas de la carrera y así facilitarnos su uso, podemos generar los archivos .jar y .war a través de “maven install” y “maven compile”, lo cual facilita las tareas de construcción.

Para la construcción del código hemos usado Eclipse Luna. Posteriormente, se pensó en utilizar la herramienta Gradle, permitiéndonos esta realizar la construcción/integración del proyecto. Gradle es gratuito y es una herramienta de las más usadas para el modelo de integración continua. Se basa en el uso de tareas, en las cuales podemos gestionar scripts para que se ejecuten de manera programada. Estas tareas pueden ir desde la construcción automática del código a la ejecución de pruebas unitarias. Hemos elegido esta herramienta porque es la que vimos durante las clases de prácticas y ya tenemos ciertos conocimientos sobre su funcionamiento.

Para la gestión de cambio, incidencias y depuración nuestro grupo de trabajo se comunica vía “Whatsapp” o en horario de clase para identificar posibles mejoras del proyecto, añadir funcionalidades, detección de errores, incidencias, etc.

A la hora de gestionar una incidencia, accedemos a github y añadimos las incidencias que nos han ido surgiendo en el apartado “issues” al usuario encargcado de resolverla, este la edita poniendo una descripción y la cierra (Closed).

En cuanto a lo referido a la gestión de liberaciones ya que no se ha dado en clase sobre ello tuvimos que investigar un poco acerca de este proceso. La gestión de liberaciones se ocupa de hacer una planificación y programación adecuada, controlar la construcción, probar y desplegar las liberaciones y entregar nuevas funcionalidades que el negocio requiere.

Una vez que todos los subsistemas han finalizado las mejoras de su código y su posterior integración ha sido probada, sin fallos, se pasa a desplegar la aplicación. Para que el despliegue se realice sin problemas, se hará sobre una máquina virtual de Windows XP que, en teoría, todos los alumnos de todos los subsistemas tenemos, ya que fue entregada por el profesorado de otra asignatura. En esta máquina se encuentran programas como Maven, Tomcat 7, Java openjdk 7, MySqlServer y Eclipse Java entre otras, explicado más a fondo en el apartado X del documento.

Como propuesta a la gestión de variabilidad (cambios en la personalización del sistema) a nuestro proyecto se acordó mantener un aspecto de la aplicación similiar al del trabajo del año anterior, nos centraríamos más en mejorar el código, añadir funcionalidades, etc.

En este documento también se recoge un mapa de herramientas con la lista de herramientas utilizadas en el proyecto juntos con una breve descripción de capa una de ellas y sus funcionalidades.

Por último, se recogerán las conclusiones que hemos sacado del proyecto y las lecciones aprendidas, ya que no teníamos experiencia en esto y hemos aprendido muchos conceptos importantes a la hora de gestionar un proyecto.

Como objetivo final del proyecto se pretende proporcionar una serie de entregables compuestos por:

* Este documento, de forma que sintetice los aspectos del proyecto elegido para su desarrollo con respecto a los temas vistos en clase. Debe ser un documento técnico y práctico. Documento presentado de manera profesional con los apartados que el índice describe.
* El enunciado de al menos un ejercicio de cada apartado. Por ejemplo, crear una rama sobre el tronco de un proyecto, junto con su solución.
* Una o varias máquinas virtuales que contengan todos los elementos necesarios para realizar los ejercicios de los apartados anteriores además del proyecto en cuestión y todas sus herramientas
* Un diario del equipo en el que se vayan reflejando todas las decisiones importantes del mismo

# Descripción del sitema

# Elementos de control

# Entorno de desarrollo

### Para el entorno de desarrollo hemos utilizado Eclipse, concretamente la versión “Luna”, con lenguaje “Java” por su previo conocimiento. Integramos este entorno con el contenedor de Servlets Tomcat de Apache para realizar el despliegue de la aplicación. Utilizamos este entorno de desarrollo porque nos permite instalar muchas de las herramientas necesarias para realizar el desarrollo de la aplicación. Utiliza JDK que proporciona herramientas muy útiles para la creación de aplicaciones Java. En concreto en nuestro proyecto hemos utilizado la versión [Java SE Development Kit 8](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html).

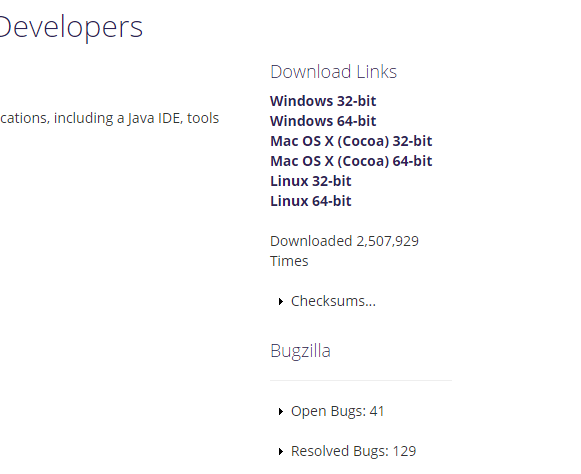
Hemos utilizado Maven para construir nuestro proyecto, ya que permite describir las dependecias con el resto de módulos en los que se ha dividido la aplicación. Lo hemos integrado en el proyecto instalando el plugin M2E-WTP12, el cual describiremos con más detalle en el apartado 11 – Mapa de herramientas.

Hemos utilizado los frameworks Angular JS y Spring para facilitar la visualización de la aplicación. Estas herramientas también se describirán con más detalle en el apartado 11.

Otra dato importante a destacar es que hemos utilizado la herramienta MySQL para conectar nuestra aplicación a al servidor de base de datos, así como el uso de Hibernate para convertir las entidades de la base de datos en los objetos JAVA que vamos a utilizar en el proyecto.

A continuación se presenta una guía que podremos seguir para poder instalar todas las herramientas necesarias para hacer funcionar nuestro sistema al completo:

1. En primer lugar descargaremos el entorno de desarrollo Eclipse Luna desde la página <https://eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/lunasr2> haciendo click en los enlaces que se nos muestra a la izquierda dependiendo del sistema operativo donde vayamos a trabajar.



1. Posteriormente haremos click en “download” y descargaremos nuestro programa.
2. Extraemos el .zip descargado y ya tendremos disponible nuestro entorno de desarrollo. Esta versión de Eclipse ya trae instalado el plugin M2E-WTP12 para integrar Maven, por lo que no será necesario descargarla.
3. A continuación descargaremos la herramienta Apache Tomcat, en concreto la versión 7.0. Para ello lo descargaremos de la página <http://tomcat.apache.org/>, iremos a la sección Tomcat 7.0.67 Released y haremos clic en “download”. Posteriormente haremos clic en los enlaces que se muestran en la sección Core dependiendo de cual sea nuestro sistema operativo.

