INGENIERÍA WEB

Documentación

José Antonio Adriano Muñoz
Gonzalo Lamas Quintero

E1 - DOCUMENTO DE PLANIFICACIÓN DE PROYECTO

1. Introducción

Este documento trata la planificación, el planteamiento y el principio de un proyecto al que hemos denominado "FaceJobs" y con el que pretendemos facilitar la búsqueda de trabajo y fomentar las salidas laborales. Este documento ha sido elaborado y aceptado por Jose Antonio Adriano y Gonzalo Lamas Quintero, como únicos miembros del grupo de trabajo que llevará a cabo este proyecto. Sirva este documento, como base para explicar qué vamos a realizar, cómo lo vamos a hacer y qué tiempo tardaremos. También, detallaremos los roles que compartimos cada uno de los miembros, los usuarios/clientes de nuestro proyecto y las herramientas que hemos usado para realizar el mismo, todo ello cumpliendo con los requisitos pedidos en la rúbrica de la asignatura Ingeniería Web de la Universidad de Cádiz, durante el actual curso 2015/2016.

2. Miembros del proyecto y clientes

Los miembros que llevarán a cabo este proyecto son:

- José Antonio Adriano
- Gonzalo Lamas Quintero

Los usuarios/clientes de este proyecto serán:

- Profesor don Juan Manuel Dodero Beardo.
- Profesor don Iván Ruiz Rube.

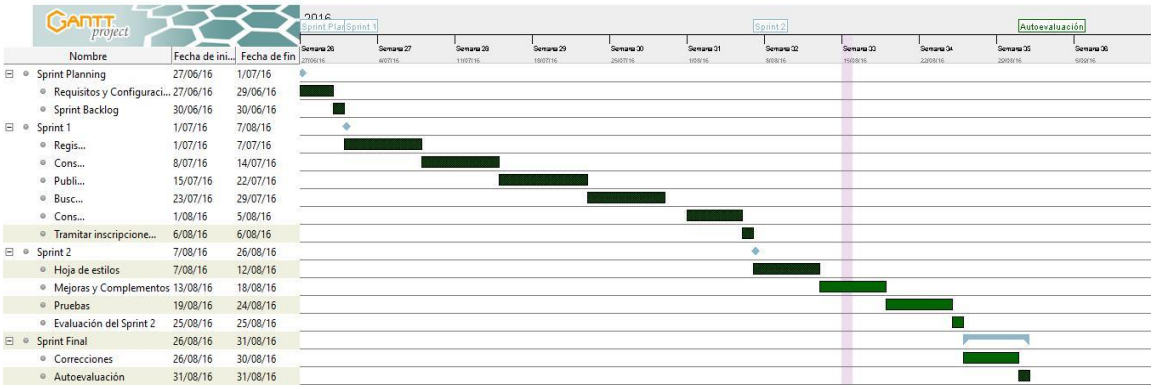
3. Roles

Los miembros del proyecto "FaceJobs", Gonzalo Lamas y José Antonio Adriano, tras la primera reunión realizada, acuerdan realizar los siguientes roles:

Roles\Miembros	José Antonio Adriano	Gonzalo Lamas Quintero
Jefe de Proyecto	✓	
Arquitecto Software	✓	
Diseñador Interfaz de Usuario		✓
Pruebas		✓
Administrador de BBDD	✓	
Administrador de Sistemas	✓	
Evaluador		✓

4. Planificación

El método que se seguirá para la realización del proyecto es un método ágil. Concretamente seguiremos el método SCRUM, en el que se marcan unos objetivos (hitos) y se planifican unos plazos o etapas (sprint) para uno o varios hitos. En estos sprint o etapas realizamos las tareas necesarias para finalizar el proyecto de la manera más ágil, eficiente y correcta posible. En nuestro caso, al tener solo 2 meses para realizar un trabajo de una duración aproximada de 4 meses, realizamos sprints más intensivos de lo normal, quedando la planificación de la siguiente manera:



Nombre	Fecha de ini...	Fecha de fin
Sprint Planning	27/06/16	1/07/16
Requisitos y Configuraci...	27/06/16	29/06/16
Sprint Backlog	30/06/16	30/06/16
Sprint 1	1/07/16	7/08/16
Regis...	1/07/16	7/07/16
Cons...	8/07/16	14/07/16
Publi...	15/07/16	22/07/16
Busc...	23/07/16	29/07/16
Cons...	1/08/16	5/08/16
Tramitar inscripcione...	6/08/16	6/08/16
Sprint 2	7/08/16	26/08/16
Hoja de estilos	7/08/16	12/08/16
Mejoras y Complementos	13/08/16	18/08/16
Pruebas	19/08/16	24/08/16
Evaluación del Sprint 2	25/08/16	25/08/16
Sprint Final	26/08/16	31/08/16
Correcciones	26/08/16	30/08/16
Autoevaluación	31/08/16	31/08/16

5. Herramientas a utilizar

En esta sección pasamos a detallar todas las herramientas que vamos a utilizar para producir y elaborar nuestro proyecto.

- **Spring Tool Site**. Entorno de desarrollo software basado en Eclipse, incluyendo extensiones que nos facilitarán la programación y unificación del proyecto.
- **Apache Maven**. Herramienta, que configurada en Spring Tool Site (STS) nos permitirá una mejor gestión de los paquetes en java y del proyecto en general.
- **MySQL**. Sistema de gestión de bases de datos relacional que usaremos para tratar con las bases de datos en nuestro proyecto.
- **PHPMyAdmin**. Herramienta local de gestión de bases de datos, que nos facilitará el manejo y uso de las bases de datos.
- **Java Enterprise Edition (Java EE)**. Será la plataforma que utilizaremos para desarrollar y ejecutar software de aplicaciones java.
- **Spring Roo/gvNIX**. Es una extensión para Spring Tool Site que usaremos para agilizar el proceso de desarrollo rápido de aplicaciones, que nos permitirá generar automáticamente el código de la misma..
- **Apache Tomcat**. Servidor usado para el desarrollo de la aplicación, además implementa las especificaciones de JavaServer Pages(JSP).
- **Junit**. Son las bibliotecas que utilizaremos para hacer pruebas unitarias en las aplicaciones Java.
- **Selenium**. Es un entorno de pruebas software que será utilizado para las pruebas de nuestro proyecto web.

- **Github**. Será la forja utilizada para unificar la aplicación. En el incluiremos un enlace a los documentos, que serán realizados en GoogleDrive.
- **GoogleDrive**. Es la plataforma para desarrollo de documentos de texto que será utilizada, tanto como para realizar los documentos, como para hacerlos públicos.

Para el desarrollo de la aplicación web serán utilizadas todas las tecnologías aquí mencionadas. Pero hay que decir que para poder desplegar la aplicación no serán necesarias todas, ya que cada una cumple una función en concreto. Los datos sobre el uso de la aplicación vendrán reflejados en el documento de implantación.

E2 - REQUISITOS DE LA APLICACIÓN

2.1- Introducción

Tras haber realizado el Sprint Planning, en el que planteamos nuestro proyecto, y haber realizado el Sprint Backlog, donde dividimos el trabajo en plazos, en este documento incluimos la explicación, detallada, de cada uno de los requisitos tanto funcionales como no funcionales, además de los requisitos de información necesarios para nuestro proyecto.

2.2- Requisitos de información

Las entidades principales y las relaciones que existen entre ellas, representan el modelo de negocio que se pretende construir teniendo como base nuestro sistema y, en nuestro caso, el dominio es el de las ofertas y demandas de empleo que se ofertan y gestionan.

RI-01→ Información de las empresas que ofertan un puesto de trabajo

Las empresas son las encargadas de registrar las ofertas de empleo en nuestro sistema y deberán identificarse con su nombre, CIF, correo, teléfono de contacto, dirección (puede tener varias sedes u oficinas), descripción de la empresa, número de empleados con los que cuenta y otros datos adicionales como complemento a los principales.

RI-02 → Información sobre las ofertas de trabajo

Las ofertas de empleo son registradas por una empresa para cubrir una de sus plazas vacantes o porque se necesita personal. Una oferta de empleo cuenta con información detallada sobre el puesto de trabajo ofertado, tipo de contrato (jornada parcial o completa), sueldo bruto, la fecha aproximada de inicio de las actividades, y del final de las mismas, fechas de inicio y fin de pre-elección para obtener la oferta de trabajo, el número de plazas ofertadas y el perfil requerido para la misma (por experiencia laboral o curriculum). Las ofertas de empleo pasan a estar en diferentes estados:

- **En espera**: la oferta es dada de alta pero todavía no podemos realizar inscripciones.

- **Activa:** la oferta es dada de alta en nuestro sistema y además, los demandantes de empleo pueden acceder a la información sobre la misma.

- **Detenida**: se cierra temporalmente la oferta de trabajo hasta volver a estar disponible, ya que se ha llegado al final del plazo para preinscribirnos en la oferta.

- **Cancelada**: la oferta es anulada por la empresa por un motivo concreto.

- **En trámite**: comienza al fase de tramitación de las inscripciones realizadas por los demandantes de empleo.

- **Resuelta**: la oferta de empleo está resuelta a favor de uno o varios demandantes de empleo que han sido preseleccionados.

RI-03 → Información sobre demandantes de empleo

Los demandantes de empleo son todas aquellas personas que se han inscrito en nuestro sistema buscando un empleo y deberán estar documentados con su nombre y apellido/s, DNI, fecha de nacimiento, dirección domiciliaria, correo electrónico, sexo y teléfono de contacto (móvil o fijo). Los demandantes pueden inscribirse en las ofertas y, en tiempo real, conocer en qué estado se encuentra su inscripción:

- **Recibida**: el demandante de empleo se inscribe en una determinada oferta .
- **Retirada**: el demandante de empleo retira su candidatura a un determinado puesto de trabajo.
- **En estudio:** el encargado de las ofertas de empleo de la empresa (normalmente recursos humanos), consulta el perfil del demandante de empleo en cuestión.
- **Descartada**: el encargado de las ofertas de empleo de la empresa (normalmente recursos humanos) decide no seleccionar al demandante de empleo en cuestión..
- **Aceptada**: el encargado de las ofertas de empleo de la empresa (normalmente recursos humanos) selecciona al demandante en cuestión como candidato para ocupar la plaza de trabajo.

RI-04 → Editar perfil del demandante.

Los demandantes de empleo, además de sus datos personales, también podrán añadir un breve resumen que trate de su experiencia profesional, sus aptitudes, por qué creen que merecen esa plaza de trabajo y una foto. El Curriculum Vitae (CV) incluye los datos sobre la formación académica y a la experiencia profesional, que serán registrados en nuestro sistema.

2.3- Requisitos funcionales

RF-01 → Registrar empresas

El sistema permite a los usuarios autorizados registrar nuevas empresas y editar los datos asociados a las mismas.

RF-02 → Publicar ofertas de trabajo

El sistema permite a los usuarios autorizados registrar nuevas ofertas de trabajo y editar sus datos, además de cambiar el estado de la oferta de trabajo, en tiempo real.

RF-03 → Registrar demandantes de empleo

El sistema permite a los usuarios registrarse como demandantes de empleo en la plataforma.

RF-04 → Editar perfil de demandante

El sistema permite a los usuarios autorizados a editar su perfil de demandante, así como gestionar todos los datos que componen su perfil.

RF-05 → Buscar candidatos

El sistema permite a los usuarios autorizados consultar entre los posibles candidatos para las ofertas de empleo, según determinados criterios de búsqueda, que podrían ser los definidos en la propia oferta de trabajo.

RF-06 → Consultar perfil de demandante

El sistema permite a los usuarios autorizados consultar el perfil completo de los demandantes de empleo.

RF-07 → Buscar ofertas de trabajo

El sistema permite a los usuarios autorizados localizar ofertas de trabajo según determinados criterios de búsqueda, tales como empresa, titulación requerida, puesto de trabajo, localización, etc.

RF-08 → Consultar características del trabajo ofertado

El sistema permite a los usuarios autorizados poder acceder a las características y condiciones del puesto de trabajo ofertado por una empresa, así como los otros datos sobre la misma.

RF-09 → Inscribir en ofertas

El sistema permite a los usuarios autorizados inscribirse en ofertas de empleo. Esta oferta puede aparecer por una búsqueda de ofertas por parte del usuario o que a este usuario le llegue una notificación automática.

RF-10 → Tramitar inscripciones

El sistema permite a los usuarios autorizados aceptar o descartar la candidatura de los demandantes a las ofertas de empleo.

RF-11 → Enviar notificaciones

El sistema deberá realizar notificaciones mediante email según determinados eventos. Por ejemplo: cuando el gestor busque y proponga posibles candidatos para la oferta, para informar a los que tienen una inscripción en curso que la misma se encuentra en estudio, para notificar cuando su candidatura ha sido descartada o cuando esta haya sido la finalmente aceptada. Asimismo, cuando el gestor cambie el estado de las ofertas de empleo se enviarán notificaciones automáticas a los implicados.

2.4- Requisitos no funcionales

RNF-01 → SEGURIDAD

a. Los datos de la aplicación solo podrán ser accedidos y modificados por aquellas personas autorizadas para ello (usuario demandante, gestor empresa).

b. El usuario demandante de empleo puede editar su perfil y añadir información que considere adecuada para mejorar sus posibilidades de encontrar empleo. Además, podrá inscribirse en las ofertas de trabajo que considere de interés.

c. El usuario gestor de empresa puede publicar ofertas de trabajo, buscar candidatos, consultar el perfil de los demandantes de empleo y tramitar las inscripciones de estos.

d. El administrador tendrá acceso a todas las operaciones de mantenimiento (altas, bajas, modificaciones y consultas) de todas las entidades y relaciones del modelo de la aplicación, así como cambiar todos los parámetros que sean configurables.

RNF-02 → INTEROPERABILIDAD ¡!MIRAR BIEN!!

a. La aplicación deberá seguir una Arquitectura Orientada a Servicios, de forma que todas las funcionalidades de la aplicación deberán estar accesibles, además de a través de la interfaz de usuario, mediante llamadas a servicios web que cumplan con la arquitectura ReST.

b. Se valorará la utilización de los servicios web desde la propia aplicación o bien, desde aplicaciones externas. En el primer caso, sería necesario incluir elementos de interacción asíncrona (AJAX) en la propia interfaz del cliente de modo que, por ejemplo, al rellenar la provincia o ciudad del demandante de empleo, automáticamente se filtren los valores aplicables conforme el usuario los teclea en el campo oportuno. En el segundo caso, se podrían recuperar datos de la plataforma para que aparezcan en otras aplicaciones web móviles de interfaz reducida.

c. También será valorable el hacer uso de datos procedentes de sistemas externos a la hora de poblar la base de datos del sistema.

RNF-03 → FIABILIDAD

a. Se realiza un correcto tratamiento de los errores originados por el usuario.

b. Se controlan los posibles fallos que puedan producirse ante un fallo de los sistemas software o hardware con los que se interactúan.

c. Las peticiones concurrentes de acceso a la base de datos deben permitir siempre que la aplicación siga en un estado consistente.

RNF-04 → PORTABILIDAD

a. Es recomendable que el sitio web sea multi-idioma, presentando los textos de la interfaz de usuario en castellano y en inglés. Asimismo, el tratamiento de fechas y números se hace de forma adecuada.

b. El código fuente no presenta instrucciones dependientes de los entornos que puedan, por ejemplo, acceder de manera no controlada a comandos del sistema operativo o a ficheros en el servidor.

c. Se evitan problemas de codificación al usar el juego de caracteres estándar UTF-8 en el código fuente del proyecto.

d. La aplicación deberá cumplir con los estándares marcados por el WWW Consortium (HTML 4.0 o superior, CSS 2.0, etc.), para maximizar su correcto uso en distintos navegadores.

RNF-05 → EFICIENCIA

a. Las consultas a la base de datos deberán estar optimizadas para evitar, por ejemplo, el problema de las N+1 consultas que surge con el uso de los motores ORM sin configurar. La base de datos deberá disponer de un pool de conexiones configurables.

b. Se utilizan soluciones estándar o ad-hoc para cacheo de datos que permitan mejorar el rendimiento del sistema.

RNF-06 → OPERABILIDAD

a. El sitio web deberá tener una estructura clara, ordenando el contenido y las funciones de la aplicación en pestañas o apartados que abarquen todas las funcionalidades disponibles, según el perfil de seguridad del usuario conectado.

b. El sitio web de la aplicación deberá estar lo más personalizado posible para ir en consonancia con la imagen corporativa de la empresa de empleo temporal y deberá poderse explotarse y administrarse empleando cualquier navegador web.

c. Es deseable que se incorporen en el sistema elementos visuales enriquecidos que permitan una mejor interacción para el usuario, tales como detalles, componentes visuales geográficos, etc.

d. El sitio web deberá cumplir con las normas de accesibilidad para aplicaciones web (WAI 2.0) definidas por el WWW Consortium, debiendo cumplir como mínimo el nivel A.

RNF-07 → MANTENIBILIDAD

a. En la implementación de las clases del modelo se respetan los principios clásicos de diseño (alta cohesión, bajo acoplamiento, etc.), y se aplican patrones de diseño habituales, como el Active Record o el Data Access Object.

b. Los elementos de la vista de la aplicación respetan siempre el patrón Decorator, ViewHelper a través del uso de layouts, sin repetir código en las vistas u otras malas prácticas de programación.

c. Los controladores del sistema delegan las peticiones de lógica de negocio sobre el modelo y envían los datos al usuario a través de las vistas de la aplicación, sin incluir código de acceso a datos o de presentación en los propios controladores.

E3- DOCUMENTO DE ANÁLISIS

3.1. Introducción

El objetivo del presente documento, es servir como medio de comunicación entre ingenieros de requisitos y desarrolladores. En este tercer entregable, se va a realizar el análisis del sistema. En él mostraremos los diagramas de casos de uso, que ya fueron descritos en el documento de requisitos, así como los diagramas de robustez. La realización de estos modelos permite a los analistas identificar problemas en los requisitos y dar el primer paso hacia el diseño y la implementación del sistema.

3.2. Descripción del modelo de casos de uso

Para describir cada uno de los casos de uso a realizar en nuestro proyecto, usaremos UML, lenguaje de modelado unificado, en este documento para realizar el análisis de nuestro proyecto.

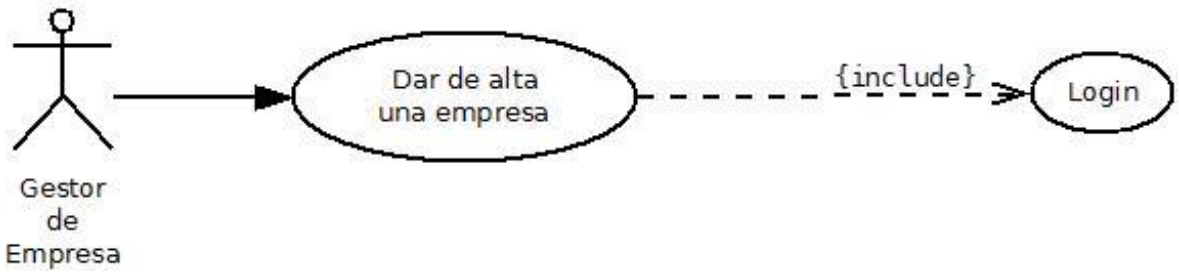
Dentro de UML, realizaremos un diagrama de casos de uso, que es un tipo de diagrama que especifica un comportamiento de una acción que se realiza sobre el sistema que estamos llevando a cabo. Cada caso de uso, es una descripción de las actividades que se llevarán a cabo en los procesos. En Además de describir las funcionalidades de nuestro sistema, vamos a especificar los roles que intervienen en cada de una de estas funcionalidades.

CU-01: REGISTRAR DEMANDANTE DE EMPLEO	
DESCRIPCIÓN	El sistema permite que cualquier usuario se dé de alta en la plataforma como demandante de empleo.
ACTORES	Demandante
PRECONDICIÓN	El usuario no debe estar registrado en el sistema con anterioridad.
FLUJO BÁSICO	1. Introducir los datos del demandante.
FLUJO ALTERNATIVO	El demandante de empleo podrá editar su perfil en cualquier momento una vez esté dado de alta en nuestro sistema.

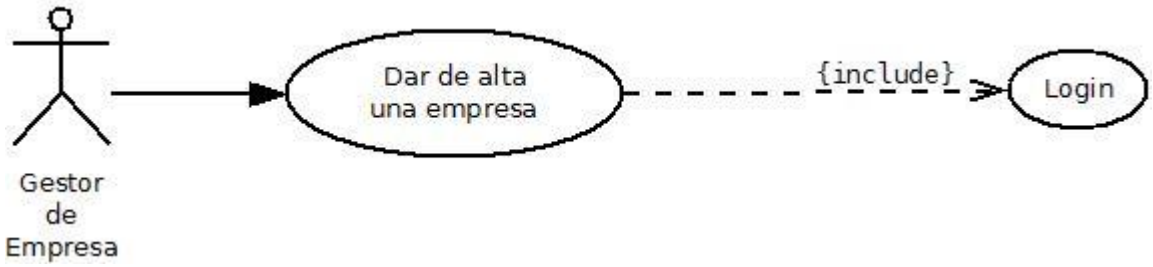
CU-02: EDITAR PERFIL DEL DEMANDANTE	
DESCRIPCIÓN	El sistema permitirá que los usuarios demandantes accedan a su perfil para añadir experiencia y modificar otros datos, etc
ACTORES	Demandante
PRECONDICIÓN	El usuario debe estar registrado en el sistema.
FLUJO BÁSICO	1. El usuario se identifica en nuestro sistema 2. El usuario accede a su perfil y lo modifica
FLUJO ALTERNATIVO	El mismo caso de uso ya es una opción para editar el perfil del demandante. No hay flujo alternativo.

CU-03: CONSULTAR PERFIL DEL DEMANDANTE	
DESCRIPCIÓN	El sistema permitirá a los usuarios autorizados consultar el perfil completo de los demandantes.
ACTORES	GestorEmpresa, Demandante
PRECONDICIÓN	El usuario no debe estar registrado en el sistema con anterioridad.
FLUJO BÁSICO	1. Identificarse en la plataforma. 2. Acceder al perfil de un demandante cualquiera.
FLUJO ALTERNATIVO	Se puede realizar la consulta mediante una búsqueda de los demandantes según diversos criterios.

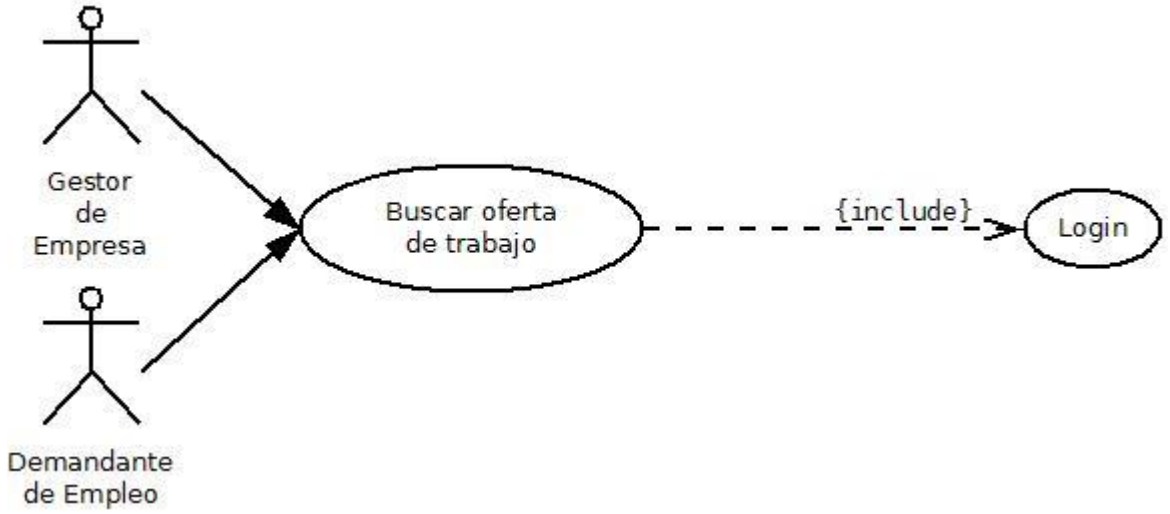
CU-04: ALTA DE EMPRESA	
DESCRIPCIÓN	Se da de alta una empresa en la plataforma.
ACTORES	GestorEmpresa
PRECONDICIÓN	1. La empresa no debe existir previamente en la plataforma. 2. El usuario debe estar registrado en el sistema.
FLUJO BÁSICO	1. Identificarse en la plataforma. 2. Introducir y dar de alta la empresa en la plataforma.
FLUJO ALTERNATIVO	Si la empresa ya existe en el sistema, se avisa al usuario de tal eventualidad y se le permite que revise los datos.



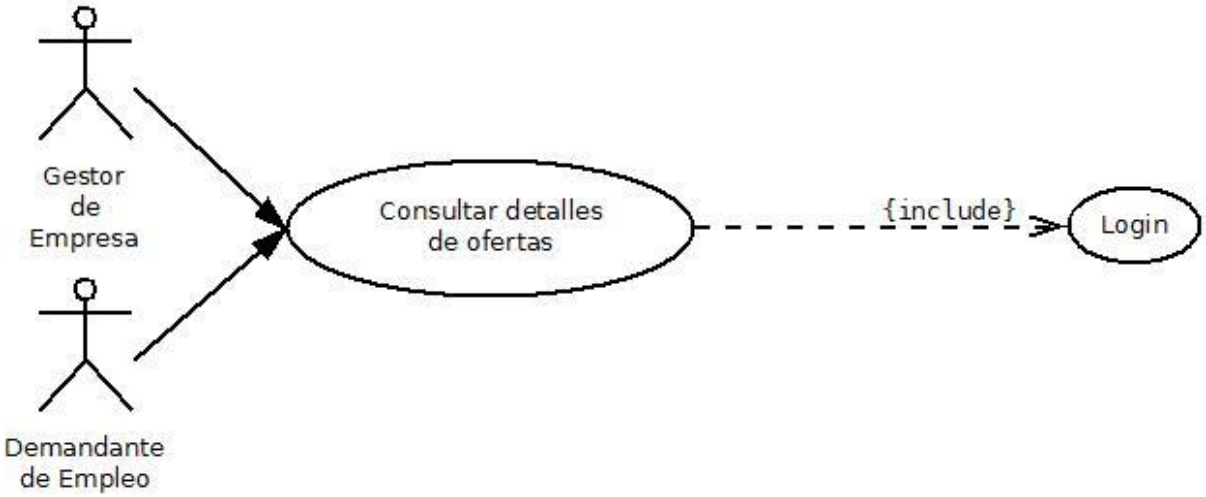
CU-05: ALTA DE OFERTAS DE TRABAJO	
DESCRIPCIÓN	Se da de alta una oferta de trabajo asociada a la sede de una empresa de la plataforma.
ACTORES	GestorEmpresa
PRECONDICIÓN	1. La oferta de trabajo no debe existir con anterioridad en la plataforma. 2. El usuario debe estar registrado en el sistema.
FLUJO BÁSICO	1. El usuario se identifica en la plataforma. 2. Se introducen los datos de la oferta y se da de alta.
FLUJO ALTERNATIVO	Desde una notificación de un anuncio/oferta de empleo



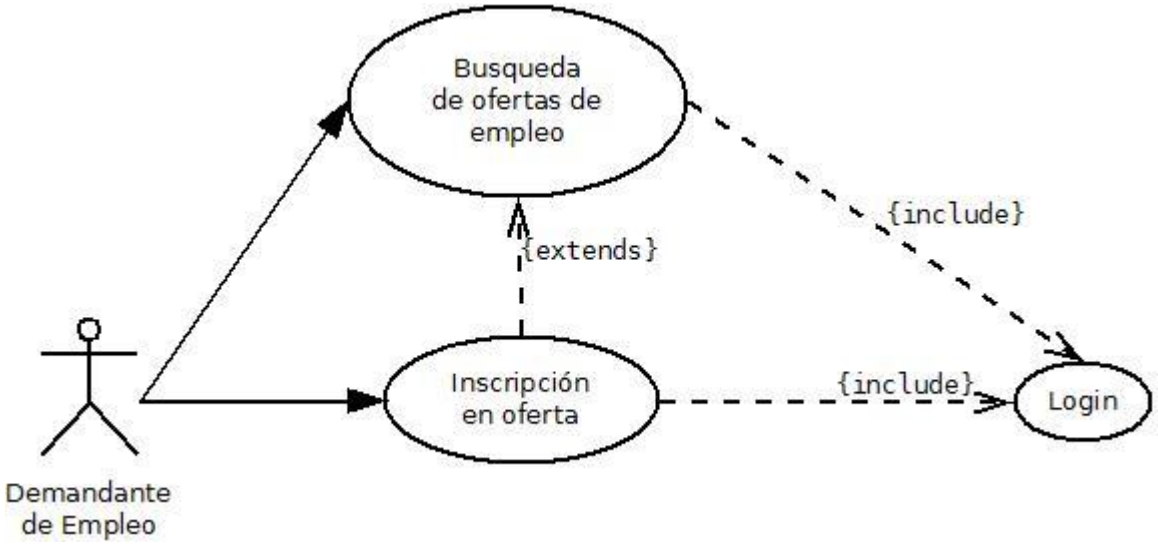
CU-06: BUSCAR OFERTAS DE TRABAJO	
DESCRIPCIÓN	El sistema permitirá a los usuarios autorizados realizar cualquier tipo de consulta sobre las ofertas de trabajo utilizando ciertos criterios como la localización, la empresa, titulación requerida, etc.
ACTORES	Demandante de Empleo, GestorEmpresa
PRECONDICIÓN	Los usuarios que realizan las búsquedas deben estar registrados en el sistema.
FLUJO BÁSICO	1. Identificarse en la plataforma. 2. Realizar una búsqueda. 3. Consultar oferta.
FLUJO ALTERNATIVO	No hay flujo alternativo.



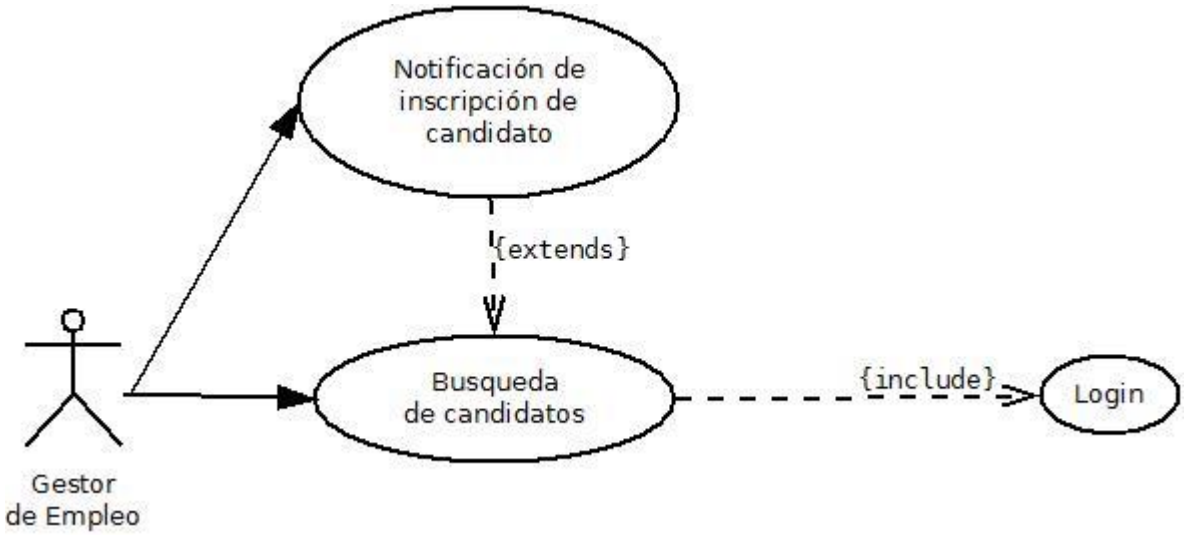
CU-07: CONSULTAR CARACTERÍSTICAS DEL TRABAJO OFERTADO	
DESCRIPCIÓN	El sistema permitirá a los usuarios autorizados visualizar las características y las condiciones particulares del puesto de trabajo ofertado, así como los datos de interés de la empresa.
ACTORES	Demandante, GestorEmpresa
PRECONDICIÓN	El usuario debe estar registrado en el sistema.
FLUJO BÁSICO	1. Identificarse en el sistema. 2. Visualizar oferta.
FLUJO ALTERNATIVO	1. Los usuarios autorizados pueden consultar una oferta a través de una búsqueda con ciertos criterios concretos. 2. Los usuarios podrán consultar los datos de la empresa que oferta.



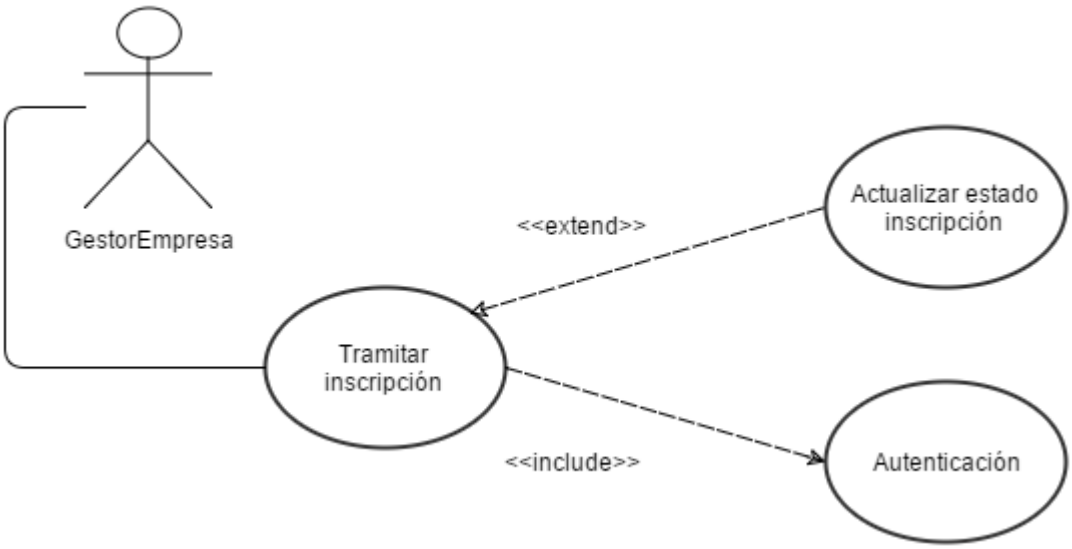
CU-08: INSCRIBIR EN OFERTA DE TRABAJO	
DESCRIPCIÓN	Los demandantes de empleo que se hayan dado de alta en la plataforma, podrán inscribirse en todas las ofertas de trabajo que consideren oportunas según su perfil.
ACTORES	Demandante de empleo
PRECONDICIÓN	1. Es la primera vez que el demandante solicita la inscripción. 2. El demandante es un usuario registrado.
FLUJO BÁSICO	1. Identificarse en la plataforma. 2. Inscribirse en la oferta de trabajo.
FLUJO ALTERNATIVO	1. Los demandantes usan una búsqueda de ofertas de trabajo para inscribirse. 2. Los demandantes pueden inscribirse en la oferta seleccionada de la lista de ofertas buscadas.



CU-09: BÚSQUEDA DE CANDIDATOS	
DESCRIPCIÓN	El sistema permitirá que se lleven a cabo búsquedas por parte de los usuarios autorizados (empresas y gestor uJob) mediante una serie de criterios (experiencia, titulación, etc).
ACTORES	Gestor de Empresa
PRECONDICIÓN	El usuario debe ser un usuario autorizado.
FLUJO BÁSICO	1. Identificarse en la plataforma. 2. Realizar una búsqueda por criterios.
FLUJO ALTERNATIVO	1. Al gestor de empresa le llega una notificación cada vez que un demandante solicita una de sus ofertas de empleo. 2. Realizar una búsqueda de la lista de demandantes de empleo inscritos en una oferta de empleo.



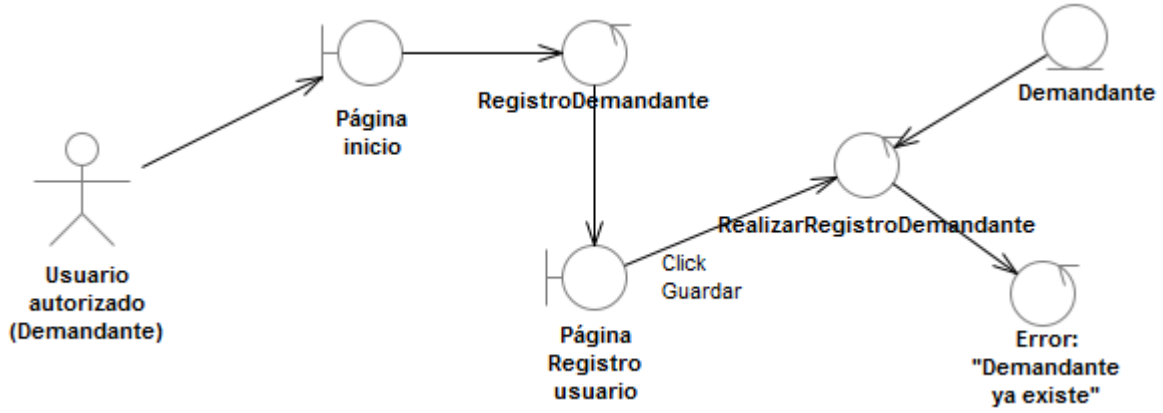
CU-10: TRAMITAR INSCRIPCIONES.	
DESCRIPCIÓN	El sistema permitirá a los usuarios autorizados descartar o aceptar las candidaturas de los demandantes a las ofertas de empleo.
ACTORES	GestorEmpresa
PRECONDICIÓN	1. El usuario debe estar registrado en el sistema
FLUJO BÁSICO	1. Identificarse en la plataforma. 2. Acceder a las inscripciones pendientes de tramitar.
FLUJO ALTERNATIVO	Si el usuario autorizado así lo decide, modificará el estado del candidato y a este se le enviará una notificación con el resultado.



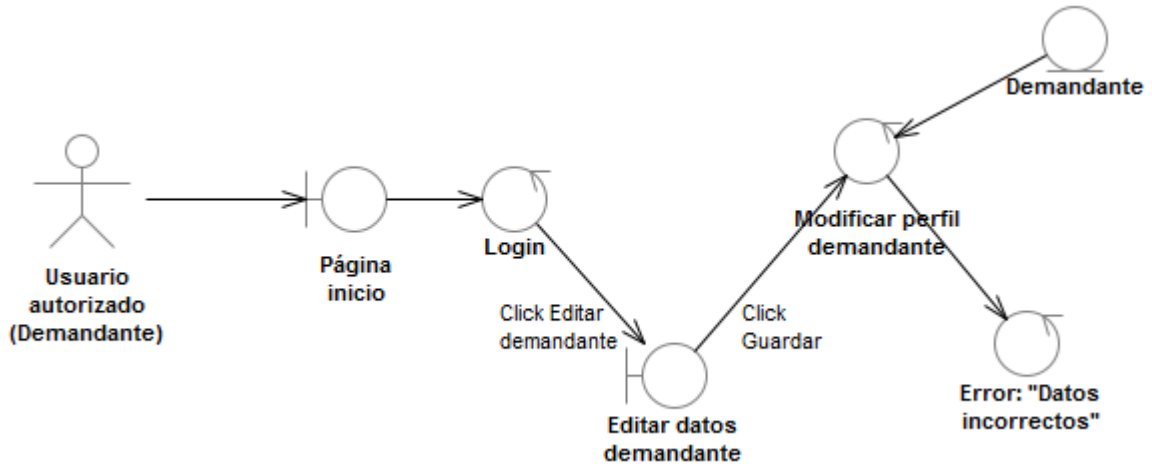
3.2. Descripción de los diagramas de robustez

Un diagrama de robustez es una herramienta que nos permitirá capturar el qué hacer y luego nos ayudará a decidir cómo hacerlo.

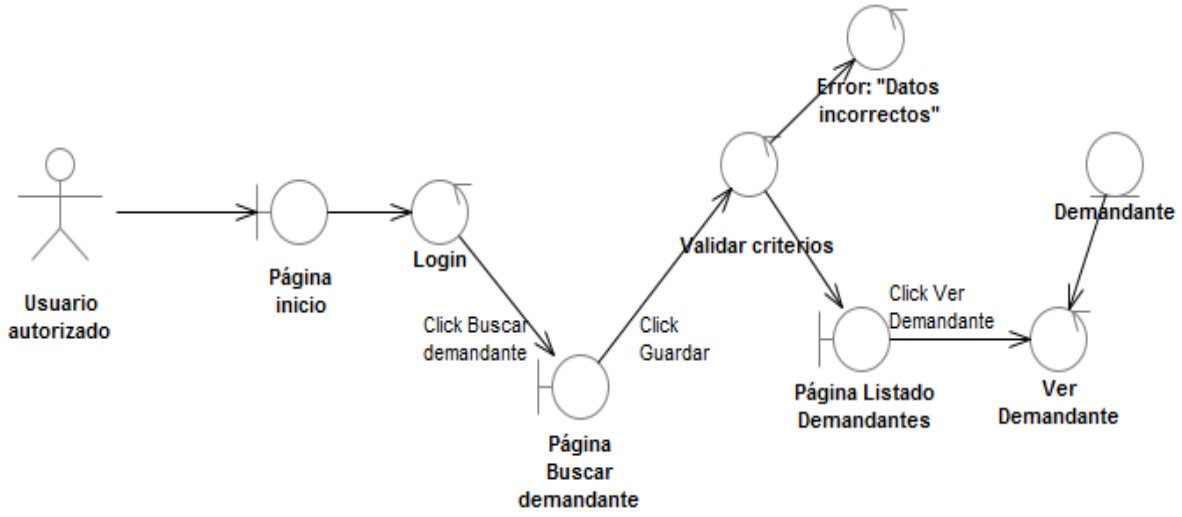
3.2.1 Diagrama Registrar demandante de empleo



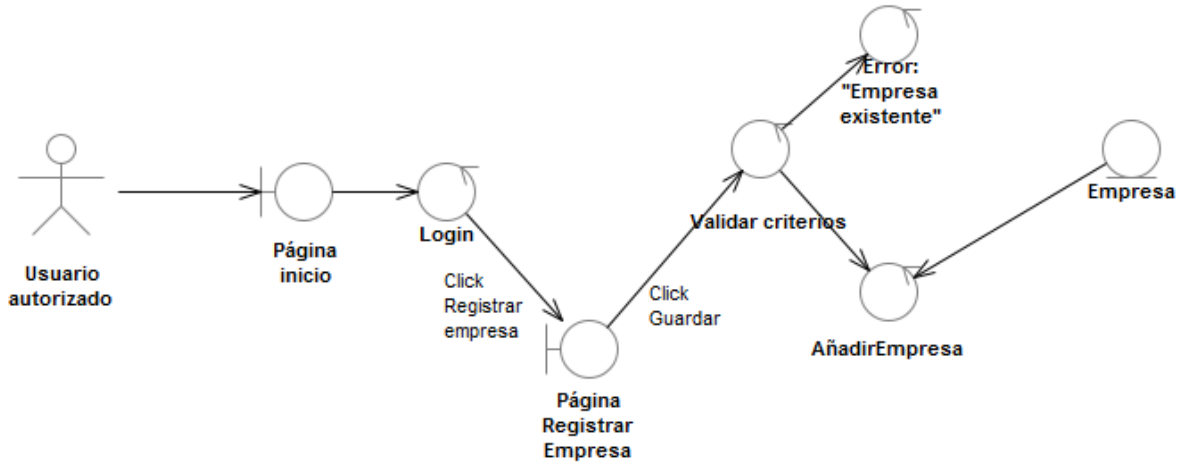
3.2.2 Diagrama Editar perfil del demandante



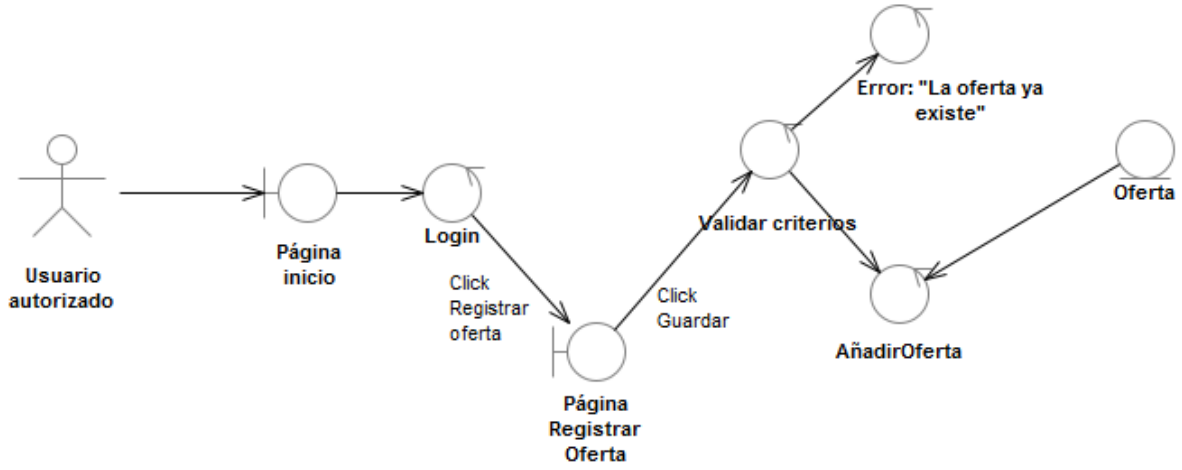
3.2.3. Diagrama Consultar perfil del demandante



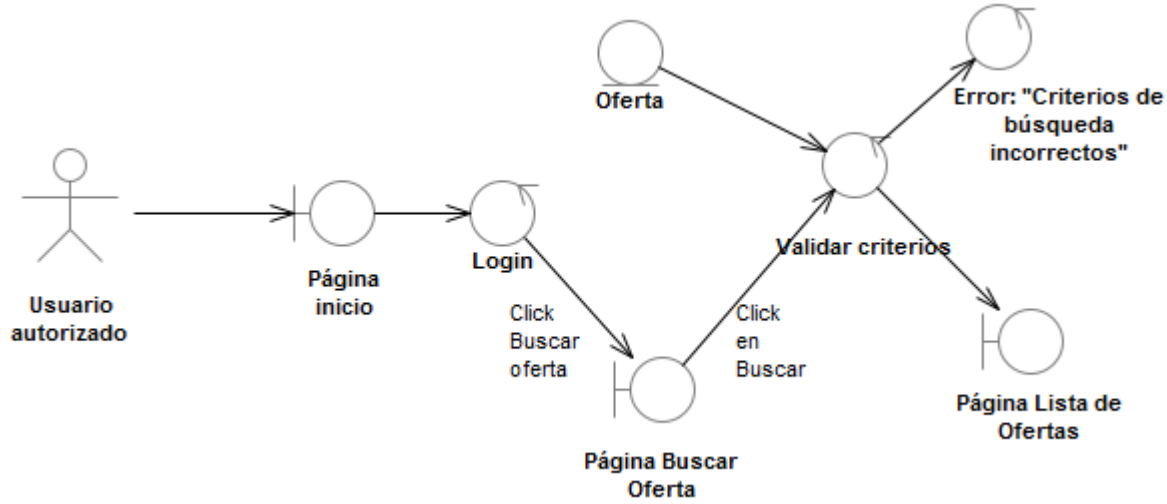
3.2.4. Diagrama *Alta de empresa*



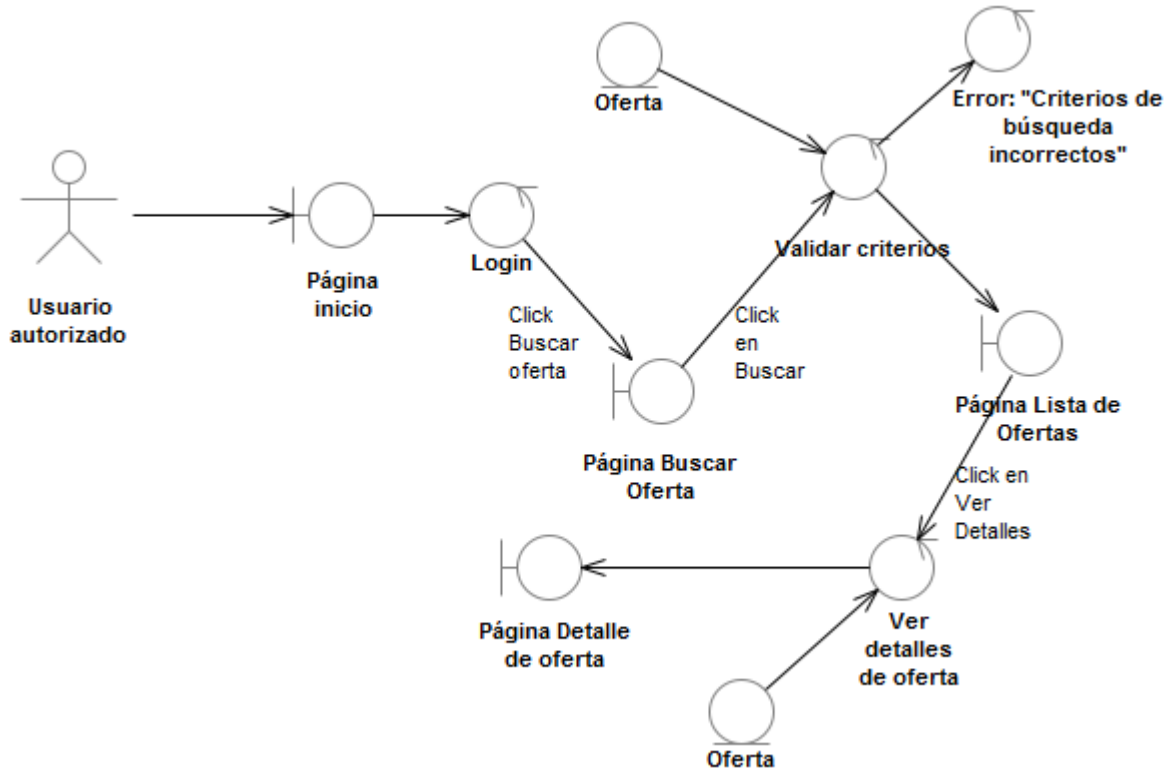
3.2.5 Diagrama Alta de ofertas de trabajo



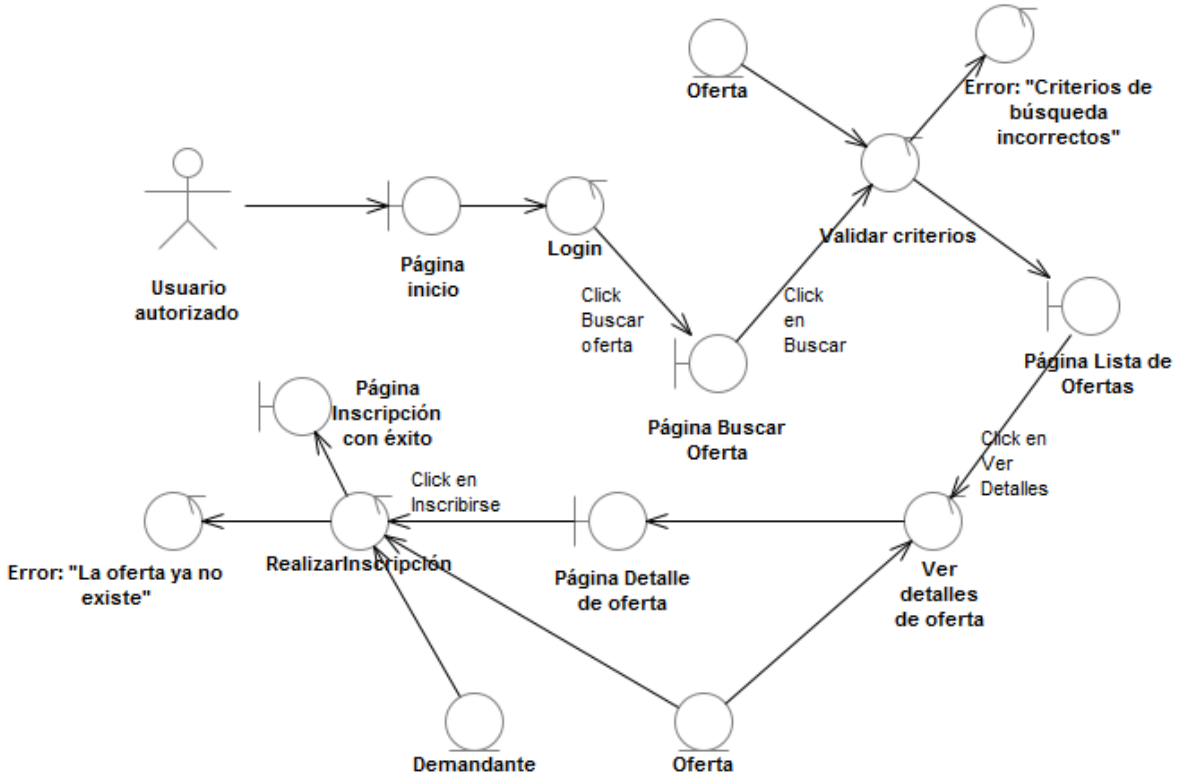
3.2.6. Diagrama Buscar ofertas de trabajo



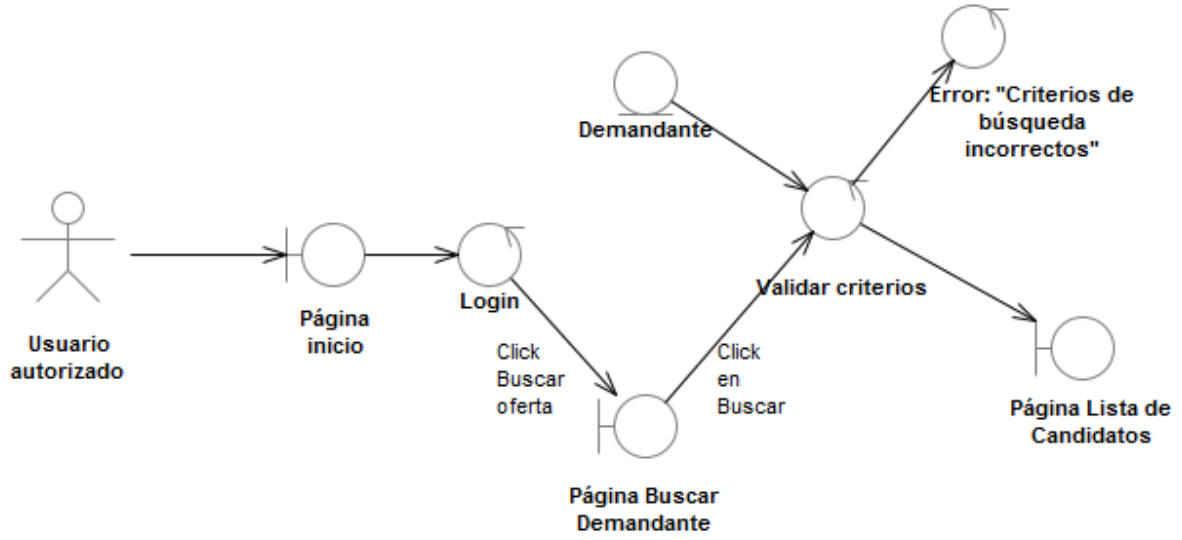
3.2.7. Diagrama Consultar características del trabajo ofertado



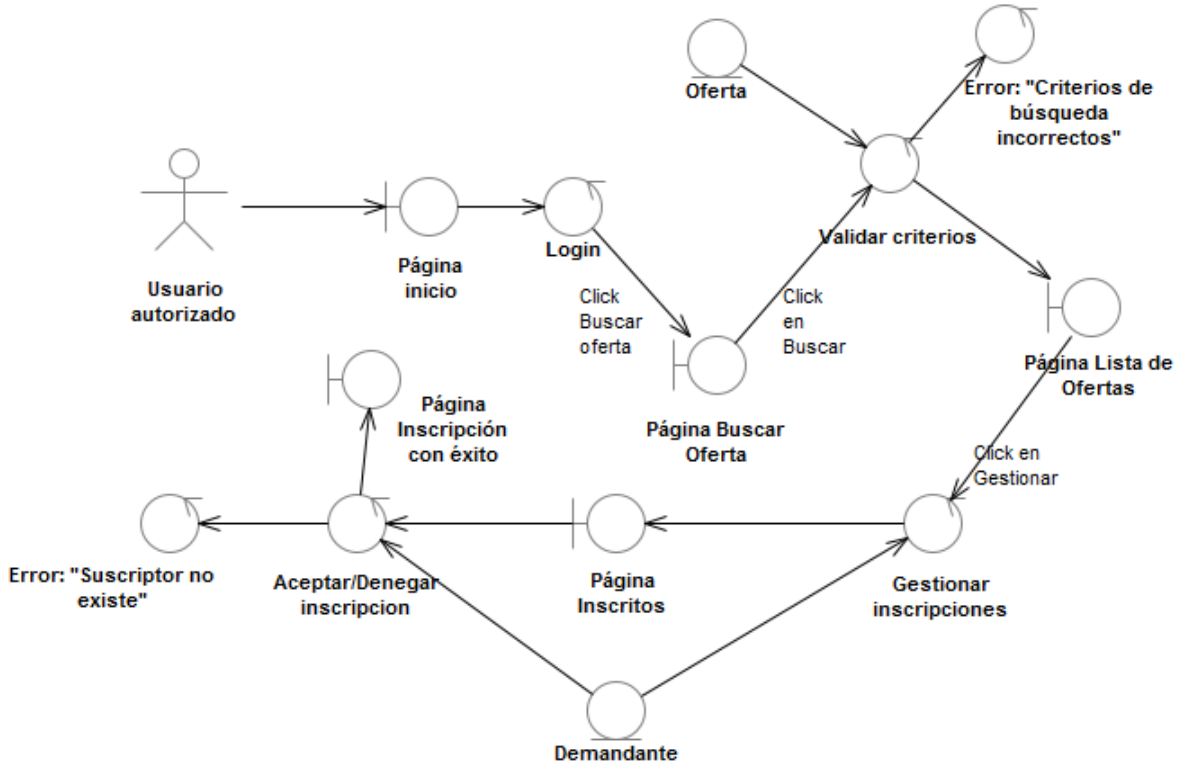
3.2.8. Diagrama Inscribir en ofertas de trabajo



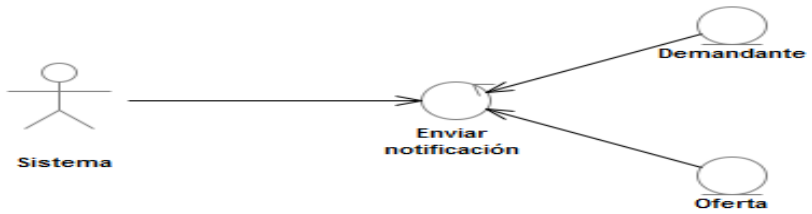
3.2.9. Diagrama Búsqueda de candidatos



3.2.10. Diagrama Tramitar inscripciones



3.2.11. Diagrama Enviar notificaciones

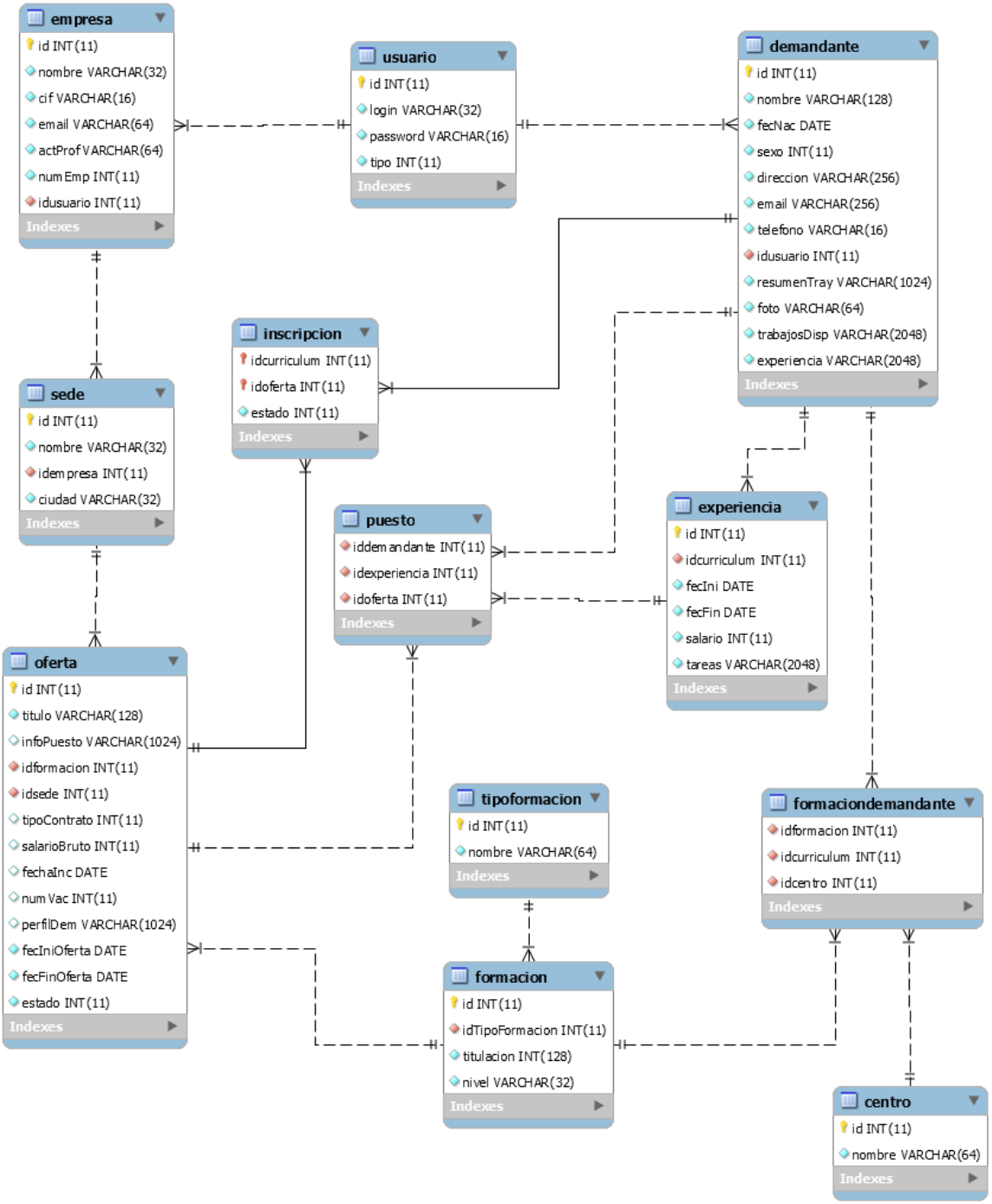


E4 - DOCUMENTO DE DISEÑO

4.1. Introducción

En el siguiente apartado, vamos a incluir el diagrama de clases de diseño que utilizamos como la base para realizar la aplicación solicitada por nuestros clientes.

4.2. Diagrama de clases de diseño



4.3. Descripción del diagrama de clases de diseño

Clase Curriculum: la clase currículum representa la formación educativa y profesional que posee un cierto demandante. Se compone de los siguientes campos:

- Id: Entero que almacena el identificador del currículum.
- Descripción: Cadena de caracteres que almacena información de un demandante.
- Formación: Cadena de caracteres que almacena información sobre los cursos u otros tipos de aprendizajes realizados por el demandante.

Clase Demandante: la clase Demandante representa la información de los demandantes de empleo de nuestro sistema.

Se compone de los siguientes campos.

- ID: entero que almacena el identificador del demandante de empleo.(11 caracteres)
- Dni: Cadena de caracteres que almacena el D.N.I del demandante (9 caracteres).
- Nombre: Cadena de caracteres que almacena el nombre completo del demandante.
- Sexo: Cadena de caracteres que almacena la sexualidad del demandante.
- Fec_Nac: Fecha en la que ha nacido el demandante.
- Dirección: Cadena de caracteres que almacena el domicilio del demandante.
- Email: Cadena de caracteres que almacena la dirección de correo electrónico.
- Teléfono: Cadena de caracteres que almacena el número de teléfono (9 caracteres).
- idusuario: entero que almacena el identificador de usuario en nuestro sistema, para un demandante de empleo registrado en el mismo. Derivado de la clase usuario.
- resumenTray: breve resumen de la trayectoria del demandante de empleo.
- foto: imagen descriptiva del demandante de empleo.
- trabajosDisp: listado de trabajos a los que opta el demandante de empleo.
- Experiencia: Cadena de caracteres que recoge la experiencia previa del demandante.

Clase Formación: la clase Formación representa los títulos académicos y/o profesionales que posee un cierto demandante. Se compone de los siguientes campos:

- Id: Entero que almacena el identificador de titulación.
- titulación: entero que almacena el código de la titulación realizada y obtenida.
- nivel: cadena de caracteres que recoge el nivel de formación del usuario.
- IdTipoFormación: entero que almacena el identificador de la clase tipoInformacion.

Clase Inscripción: la clase inscripción recogerá las inscripciones realizadas en la aplicación donde intervienen demandante y oferta. Se compone de los siguientes campos:

- Idcurriculum: Entero que almacena el identificador del curriculum (clave foránea).
- estado: entero que almacena el estado de la inscripción.
- idoferta: clave foránea de oferta para su relación.

Clase Oferta: la clase oferta representa las características de las ofertas publicadas en la aplicación por parte de las empresas. Se compone de los siguientes campos:

- Id: Entero que almacena el identificador de la oferta.

- titulo: texto que describe brevemente información titular sobre la oferta.
- infoPuesto: texto que detalla la información sobre la plaza de empleo ofertada.
- estado: Valor enumerado que almacena el estado de la oferta.
- FecFinOferta: Fecha que indica el día final respecto a la duración de la oferta.
- fecIniOferta: Fecha que indica el día que la oferta se propuso.
- fechaInc: Fecha que indica el día que se incoporaría el demandante de empleo seleccionado.
- numVac: Entero que almacena el número de plazas que ofrece la oferta.
- perfilDem: Cadena de caracteres que almacena el perfil buscado por la oferta.
- salarioBruto: entero que almacena el sueldo al que se aspira en dicha oferta.
- tipoContrato: entero que indica si la el contrato es parcial o completo.
- idformacion: clave foránea de la clase formación.
- idsede: clave foránea con información sobre la sede de la empresa que oferta el empleo.

Clase Sede: la clase sede recoge información necesaria acerca de una sede de una empresa en la cual se oferta una plaza de trabajo. Se compone de los siguientes campos:

- Id: entero que almacena el identificador de una sede de una empresa.
- nombre: cadena de caracteres que almacena la dirección de la empresa o una de sus sedes.
- ciudad: cadena de caracteres que representa la ciudad donde se ubica una sede.
- idEmpresa: clave foránea de empresa para su relación.

Clase Empresa: la clase empresa trata la información de las empresas registradas en la aplicación. Se compone de los siguientes campos:

- id: entero que almacena el identificador de una empresa.
- cif: cadena de caracteres que almacena el C.I.F de la empresa.
- nombre: cadena de caracteres que almacena el nombre completo de la empresa.
- email: cadena de caracteres que almacena la dirección de correo electrónico de la empresa.
- actProf: cadena de caracteres que detalla qué tipo de actividad profesional realiza una empresa.
- numEmp: entero que almacena la cantidad de empleados activos en la empresa.
- idUsuario: clave foránea de empresa para su relación.

Clase Usuario: la clase usuario trata toda la información necesaria del usuario registrado en nuestro sistema, para garantizarle una serie de permisos acorde a si es demandante o empresa.

Se compone de los siguientes campos:

- id: entero que almacena el identificador de un usuario registrado en nuestro sistema.
- login: cadena de caracteres que almacena el nombre del usuario en el sistema.
- password: cadena de caracteres que almacena la contraseña del usuario en el sistema.
- tipo: entero que marca el tipo de usuario (demandante de empleo o empresa).

Clase Puesto: relaciona el puesto de trabajo de una oferta concreta con toda la información referente al demandante de empleo que se ofrece como candidato. Esta clase está formada al completo por claves foráneas:

- iddemandante: clave foránea relacionada con la información del demandante de empleo.

- idexperiencia: clave foránea referente a la experiencia del demandante de empleo.
- idoferta: clave foránea que incluye el identificador de una oferta de empleo.

E5 - DOCUMENTO DE IMPLEMENTACIÓN

5.1. Introducción

Aquí vamos a incluir el enlace de GitHub con los ficheros del proyecto y los plugins y otras herramientas extra que hemos usado en dicho proyecto.

5.2. Ficheros del proyecto

Todos los ficheros necesarios para el proyecto, tanto los creados gracias al uso del framework *SpringRoo*, como sus comandos, como los ficheros de bases de datos, etc.. se encuentran en la siguiente dirección, dentro del repositorio de GitHub:

https://github.com/IngenieriaWebUCA/iw2016-facejobs

A continuación, tenemos que descargarnos los archivos del proyecto con los comandos propios de Git, que son los siguientes:

Para poder descargarse todos los archivos del repositorio el paso a seguir es el de clonar el repositorio a su equipo, mediante el siguiente comando:

git clone https://github.com/IngenieriaWebUCA/iw2016-facejobs

5.3. Herramientas del proyecto

Además del uso de alguna distribución GNU-Linux y lo mencionado en el apartado de planificación y configuración, hemos hecho uso de algunas herramientas y plugins para dar funcionalidad extra a nuestro proyecto. Las herramientas y plugins empleados en el proyecto han sido:

- Spring Security, para restringir el acceso a usuarios autorizados.
- Spring Roo, como plugin del STS.
- GitHub, donde se aloja el repositorio de la aplicación.

5.4. La base de nuestro sistema

A continuación, detallaremos la implementación y el desarrollo de nuestro sistema durante la creación de este proyecto. Siguiendo una serie de reglas, realizamos la conexión de nuestro sistema con la base de datos, incluyendo entidades, relaciones y atributos, contando con la seguridad necesaria y desde una aplicación web.

Siendo acorde a los requisitos del proyecto, los desarrolladores de nuestro proyecto describen los pasos realizados de la siguiente manera:

 # Crear conexión con la Base de Datos

jpa setup --provider HIBERNATE --database MYSQL --databaseName facejobs --hostName localhost

 # Creación de la entidad

entity jpa --class ~.domain.Entidad --testAutomatically

 # Creación de atributos de las entidades

Ejemplo: definimos un atributo de tipo string, el cuál no puede ser nulo, además de ser único y con un tamaño comprendido de caracteres entre 3 y 64.

field string --fieldName nombreAtributo --notNull --unique --sizeMin 3 --sizeMax 64

Si nos encontramos fuera de la Entidad sobre la que queremos añadir los atributos correspondientes,

únicamente debemos utilizar el siguiente comando para situarnos en la Entidad que necesitemos.

focus --class ~.domain.Entidad

 # Definir referencias entre atributos de las entidades

Podemos definir dos tipos.

- Usando *reference*:
  field reference --fieldName nombreReferencia --type ~.domain.EntidadReferencia --class ~.domain.Entidad2

- Usando *set*:
  field set --fieldName formacion --type ~.domain.EntidadReferencia--class

  ~.domain.Entidad2

 # Generado de capa web

web mvc setup
web mvc all --package ~.web
web mvc scaffold --class ~.web.EntidadController --backingType ~.domain.Entidad

 # Ámbito de seguridad para posterior configuración

roo > security setup

Añadido dicho comando, posteriormente definiremos los roles y caparemos el acceso dependiendo del rol de usuario, todo ello se puede hacer mediante la etiqueta <sec: authorize> y modificando el fichero de configuración donde se definen además los roles a utilizar en nuestra aplicación. Dicho fichero será *applicationContext-security.xml*

5.4.1. Base de datos:

Base de Datos: https://drive.google.com/open?id=0B_aXIjS6_M-jUUh6ckpxS3JTV0U

5.4.2. Restricción por vista de botones de acción (Ejemplo) :

```
<c:if test="${delete}">
        <td class="utilbox">
          <spring:url value="${path}/${itemId}" var="delete_form_url" />
          <spring:url value="/resources/images/delete.png" var="delete_image_url" />
          <form:form action="${delete_form_url}" method="DELETE">
            <spring:message arguments="${typeName}" code="entity_delete" var="delete_label"
htmlEscape="false" />
            <c:set var="delete_confirm_msg">
              <spring:escapeBody javaScriptEscape="true">
                <spring:message code="entity_delete_confirm" />
              </spring:escapeBody>
            </c:set>
            <input alt="${fn:escapeXml(delete_label)}" class="image" src="${delete_image_url}"
title="${fn:escapeXml(delete_label)}" type="image" value="${fn:escapeXml(delete_label)}"
onclick="return confirm('${delete_confirm_msg}');" />
            <c:if test="${not empty param.page}">
              <input name="page" type="hidden" value="1" />
            </c:if>
            <c:if test="${not empty param.size}">
              <input name="size" type="hidden" value="${fn:escapeXml(param.size)}" />
            </c:if>
          </form:form>
        </td>
      </c:if>
```

 # Generador de buscadores por criterios concretos (finders)

Debemos ejecutar la orden siguiente en la consola de Roo:

finder list

Lo cual nos devolverá un listado de finders o buscadores basados en los atributos de la Entidad en la que estemos situados. Por defecto Roo devuelve finders de profundidad uno, no devolverá finders por criterios múltiples del tipo findBySexoLikeAndSueldoGreaterThan()

sino finders del tipo findBySueldoGreaterThan, findBySexoLike, findByNombreLike...etc.

Observamos la lista que nos devuelve y bajo nuestro criterio, elegiremos uno de ellos, lo cuál se le debe indicar para que lo añada mediante la siguiente orden:

finder add nombreDelFinder

En nuestro caso, realizaremos finders bajo múltiples criterios, lo cual quedará modificado bajo nuestra responsabilidad y no la de SpringRoo, ya que no maneja buscadores por más de un criterio.

5.4.5. Idiomas (i18n)

Se han añadido soporte para el idioma inglés, lo cual ha conllevado realizar una traducción del fichero en español al otro idioma.

5.4.6. Fichero de registro de comandos utilizados (log.roo):

```
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-20 12:40:30
project --topLevelPackage es.uca.iw --projectName iw2016-facejobs --java 6 --packaging WAR
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-20 12:40:31
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-20 12:40:35
jpa setup --provider HIBERNATE --database MYSQL
// [failed] entity jpa --class ~.domain.centro
// [failed] field number --fieldName id --type java.lang.Integer --notNull --class ~.domain.centro
// [failed] entity jpa --class ~.domain.centro
// [failed] field number --fieldName id --type java.lang.Integer --notNull --class ~.domain.Centro
// [failed] field number --fieldName id --type java.lang.Integer --notNull --class ~.domain.centro
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-20 12:53:43
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-20 12:54:06
jpa setup --provider HIBERNATE --database MYSQL
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-20 12:54:44
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-20 12:55:15
entity jpa --class ~.domain.centro
field number --fieldName id --type java.lang.Integer --notNull --class ~.domain.centro
field string --fieldName nombre --class ~.domain.centro
finder add --finderName findId --class ~.domain.centro
finder add --finderName findIdLike --class ~.domain.centro
```

```
web mvc setup



web mvc all --package ~.web



entity jpa --class ~.domain.empresa
field number --fieldName id --type java.lang.Integer --notNull --class
~.domain.empresa
field string --fieldName nombre --class ~.domain.empresa
field string --fieldName cif --class ~.domain.empresa
field string --fieldName email --class ~.domain.empresa
// [failed] field string --fieldName email --class ~.domain.empresa
field string --fieldName actProf --class ~.domain.empresa
field number --fieldName numEmp --type java.lang.Integer --class
~.domain.empresa
field number --fieldName idUsuario --type java.lang.Integer --class
~.domain.empresa
web mvc all --package ~.web
entity jpa --class ~.domain.usuario
field number --fieldName id --type java.lang.Integer --notNull --class
~.domain.usuario
field string --fieldName login --class ~.domain.usuario
field string --fieldName password --class ~.domain.usuario
field number --fieldName tipo --type java.lang.Integer --class
~.domain.usuario
web mvc all --package ~.web
entity jpa --class ~.domain.demandante
field number --fieldName id --type java.lang.Integer --notNull --class
~.domain.demandante
field string --fieldName nombre --class ~.domain.demandante
field date --fieldName facNac --type java.util.Date --class
~.domain.demandante
field number --fieldName sexo --type java.lang.Integer --class
~.domain.demandante
field string --fieldName direccion --class ~.domain.demandante
field string --fieldName email --class ~.domain.demandante
field string --fieldName telefono --class ~.domain.demandante
field number --fieldName idusuario --type java.lang.Integer --class
~.domain.demandante
field string --fieldName resumenTray --class ~.domain.demandante
field string --fieldName foto --class ~.domain.demandante
field string --fieldName trabajosDisp --class ~.domain.demandante
field string --fieldName experiencia --class ~.domain.demandante
web mvc all --package ~.web
```

```
entity jpa --class ~.domain.inscripcion
field number --fieldName idcurriculum --type java.lang.Integer --notNull
--class ~.domain.inscripcion
field number --fieldName idoferta --type java.lang.Integer --notNull --
class ~.domain.inscripcion
field number --fieldName estado --type java.lang.Integer --notNull --
class ~.domain.inscripcion
web mvc all --package ~.web
entity jpa --class ~.domain.sede
field number --fieldName id --type java.lang.Integer --notNull --class
~.domain.sede
field string --fieldName nombre --class ~.domain.sede
field number --fieldName idempresa --type java.lang.Integer --notNull --
class ~.domain.sede
field string --fieldName ciudad --class ~.domain.sede
web mvc all --package ~.web
entity jpa --class ~.domain.puesto
field number --fieldName iddemandante --type java.lang.Integer --notNull
--class ~.domain.puesto
field number --fieldName idexperiencia --type java.lang.Integer --
notNull --class ~.domain.puesto
field number --fieldName idoferta --type java.lang.Integer --notNull --
class ~.domain.puesto
web mvc all --package ~.web
entity jpa --class ~.domain.experiencia
field number --fieldName id --type java.lang.Integer --notNull --class
~.domain.experiencia
field number --fieldName idcurriculum --type java.lang.Integer --notNull
--class ~.domain.experiencia
field date --fieldName fecini --type java.util.Date --class
~.domain.experiencia
field date --fieldName fecfin --type java.util.Date --class
~.domain.experiencia
field number --fieldName salario --type java.lang.Integer --class
~.domain.experiencia
field string --fieldName tareas --class ~.domain.experiencia
web mvc all --package ~.web
// [failed] field number --fieldName id --type java.lang.Integer --
notNull --class ~.domain.oferta
entity jpa --class ~.domain.oferta
field number --fieldName id --type java.lang.Integer --notNull --class
~.domain.oferta
field string --fieldName titulo --class ~.domain.oferta
field string --fieldName infoPuesto --class ~.domain.oferta
field number --fieldName idFormacion --type java.lang.Integer --class
```

```
~.domain.oferta
field number --fieldName idSede --type java.lang.Integer --class
~.domain.oferta
field number --fieldName tipoContrato --type java.lang.Integer --class
~.domain.oferta
field number --fieldName salarioBruto --type java.lang.Integer --class
~.domain.oferta
field date --fieldName fechaInc --type java.util.Date --class
~.domain.oferta
field number --fieldName numVac --type java.lang.Integer --class
~.domain.oferta
field string --fieldName perfilDem --class ~.domain.oferta
field date --fieldName fecIniOferta --type java.util.Date --class
~.domain.oferta
field date --fieldName fecFinOferta --type java.util.Date --class
~.domain.oferta
field number --fieldName estado --type java.lang.Integer --class
~.domain.oferta
web mvc all --package ~.web
entity jpa --class ~.domain.tipoformacion
field number --fieldName id --type java.lang.Integer --notNull --class
~.domain.tipoformacion
field string --fieldName nombre --class ~.domain.tipoformacion
web mvc all --package ~.web
entity jpa --class ~.domain.formaciondemandante
field number --fieldName idformacion --type java.lang.Integer --notNull
--class ~.domain.formaciondemandante
field number --fieldName idcurriculum --type java.lang.Integer --notNull
--class ~.domain.formaciondemandante
field number --fieldName idcentro --type java.lang.Integer --notNull --
class ~.domain.formaciondemandante
web mvc all --package ~.web
entity jpa --class ~.domain.formacion
field number --fieldName id --type java.lang.Integer --notNull --class
~.domain.formacion
field number --fieldName idTipoFormacion --type java.lang.Integer --
notNull --class ~.domain.formacion
field number --fieldName titulacion --type java.lang.Integer --class
~.domain.formacion
field string --fieldName nivel --class ~.domain.formacion
web mvc all --package ~.web
field reference --fieldName idusuarioDem --type ~.domain.usuario --class
~.domain.demandante
field reference --fieldName idusuarioEmp --type ~.domain.usuario --class
~.domain.empresa
field reference --fieldName idDemandante --type ~.domain.demandante --
```

```
class ~.domain.inscripcion
field reference --fieldName idOferta --type ~.domain.oferta --class
~.domain.inscripcion
// [failed] field reference --fieldName idEmpresa --type
~.domain.empresa --class ~.domain.sede --cardinality ONE_TO_MANY
field reference --fieldName idEmpresa --type ~.domain.empresa --class
~.domain.sede --cardinality MANY_TO_ONE
// [failed] field reference --fieldName idSede --type ~.domain.sede --
class ~.domain.oferta --cardinality MANY_TO_ONE
field reference --fieldName idSedeOferta --type ~.domain.sede --class
~.domain.oferta --cardinality MANY_TO_ONE
field reference --fieldName idInscripcionOferta --type
~.domain.inscripcion --class ~.domain.oferta --cardinality MANY_TO_ONE
field reference --fieldName idInscripcionDemandante --type
~.domain.inscripcion --class ~.domain.demandante --cardinality
MANY_TO_ONE
field reference --fieldName idExperienciaDemandante --type
~.domain.demandante --class ~.domain.demandante --cardinality
MANY_TO_ONE
// [failed] field reference --fieldName idFormacionDemandanteFormacion -
-type ~.domain.formaciondemandante --class ~.domain.formacion --
cardinality MANY_TO_MANY
// [failed] field reference --fieldName idFormacionTipoFormacion --type
~.domain.formacion --class ~.domain.tipoFormacion --cardinality
MANY_TO_ONE
field reference --fieldName idFormacionTipoFormacion --type
~.domain.formacion --class ~.domain.tipoformacion --cardinality
MANY_TO_ONE
field reference --fieldName idOfertaFormacion --type ~.domain.oferta --
class ~.domain.formacion --cardinality MANY_TO_ONE
web mvc all --package ~.web
field reference --fieldName idPuestoOferta --type ~.domain.puesto --
class ~.domain.oferta --cardinality MANY_TO_ONE
field reference --fieldName idPuestoDemandante --type ~.domain.puesto --
class ~.domain.demandante --cardinality MANY_TO_ONE
field reference --fieldName idPuestoExperiencia --type ~.domain.puesto -
-class ~.domain.experiencia --cardinality MANY_TO_ONE
// [failed] field reference --fieldName idFormacionDemandanteFormacion -
-type ~.domain.formaciondemandante --class ~.domain.formacion --
cardinality MANY_TO_MANY
field reference --fieldName idFormacionDemandanteFormacion --type
~.domain.formaciondemandante --class ~.domain.formacion --cardinality
MANY_TO_ONE
field reference --fieldName idFormacionDemandanteDemandante --type
~.domain.formaciondemandante --class ~.domain.demandante --cardinality
MANY_TO_ONE
```

```
field reference --fieldName idFormacionDemandanteCentro --type
~.domain.formaciondemandante --class ~.domain.centro --cardinality
MANY_TO_ONE
// [failed] field reference --fieldName idPuestoOferta --type
~.domain.puesto --class ~.domain.oferta --cardinality MANY_TO_ONE
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-20
20:52:13
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-21
18:32:16
security setup
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-21
20:00:44
web mvc all --package ~.web



web mvc language --code en
web mvc language --code es
finder add --finderName findOfertasByEstado --class ~.domain.oferta
web mvc finder all
finder add --finderName findOfertasBySede --class ~.domain.oferta
finder add --finderName findOfertasByidSede --class ~.domain.oferta
finder add --finderName findOfertasByIdSede --class ~.domain.oferta
finder add --finderName findOfertasByTipoContrato --class
~.domain.oferta
finder add --finderName findOfertasByTitulo --class ~.domain.oferta
finder add --finderName findOfertasBySede --class ~.domain.oferta
finder add --finderName findOfertasByTituloLike --class ~.domain.oferta
finder add --finderName findEmpresasByNombreLike --class
~.domain.empresa
finder add --finderName findPerfilsByTrabajosDisp --class
~.domain.demandante
finder add --finderName findDemandantesByTrabajosDisp --class
~.domain.demandante
web mvc finder all
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-21
22:29:13
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-22
19:42:11
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-22
20:28:12
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-23
19:35:22
web mvc language --code de
enum type --class ~.inscripcion.estado
enum type --class ~.reference.estadoIns
```

```
enum constant --name Espera
enum constant --name Activada
enum constant --name Detenida
enum constant --name Cancelada
enum constant --name Tramite
enum constant --name Resuelta
focus --class  ~.domain.inscripcion
field enum --fieldName estado --type ~reference.estadoIns --notNull
web mvc all --package ~.web
enum type --class ~.reference.tipoUsu
enum constant --name Empresa
enum constant --name Demandante
focus --class  ~.domain.usuario
field enum --fieldName idTipoUsu --type ~reference.tipoUsu --notNull
enum type --class ~.reference.estadoOfer
enum constant --name Abierta
enum constant --name Cerrada
enum constant --name Cancelada
focus --class  ~.domain.oferta
field enum --fieldName estadoOferta --type ~reference.estadoOfer --
notNull
web mvc finder all
web mvc all --package ~.web
// [failed] entity jpa --class ~.domain.demandante --sequenceName DE_SEQ
--testAutomatically
// [failed] entity jpa --class ~.domain.demandantE --sequenceName
DEE_SEQ --testAutomatically
enum type --class ~.reference.tipoFormacion
enum constant --name ESO
enum constant --name Bachillerato
enum constant --name CFGM
enum constant --name CFGS
enum constant --name Universidad
enum constant --name Master
enum constant --name Doctorado
focus --class  ~.domain.formacion
field enum --fieldName tipoForma --type ~reference.tipoForma --notNull
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-23
21:19:30
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-26
20:47:58
enum type --class ~.reference.tipoSexo
enum constant --name Masculino
enum constant --name Femenino
focus --class  ~.domain.demandante
```

```
field enum --fieldName tipoSexo --type ~reference.tipoSx --notNull
web mvc all --package ~.web
web mvc all --package ~.web
enum type --class ~.reference.tipoContrato
enum constant --name Indefinido
enum constant --name Temporal
web mvc all --package ~.web
web mvc all --package ~.web
test mock --entity ~.domain.Centro
test mock --entity ~.domain.Demandante
test mock --entity ~.domain.Empresa
test mock --entity ~.domain.Experiencia
test mock --entity ~.domain.Formacion
test mock --entity ~.domain.Formaciondemandante
test mock --entity ~.domain.Inscripcion
test mock --entity ~.domain.Oferta
test mock --entity ~.domain.Puesto
test mock --entity ~.domain.Sede
test mock --entity ~.domain.Tipoformacion
test mock --entity ~.domain.Usuario
test integration --entity ~.domain.Centro
test integration --entity ~.domain.Demandante
test integration --entity ~.domain.Empresa
test integration --entity ~.domain.Experiencia
test integration --entity ~.domain.Formacion
test integration --entity ~.domain.Formaciondemandante
test integration --entity ~.domain.Inscripcion
test integration --entity ~.domain.Oferta
test integration --entity ~.domain.Puesto
test integration --entity ~.domain.Sede
test integration --entity ~.domain.Tipoformacion
test integration --entity ~.domain.Usuario
selenium test --controller ~.web.CentroController
selenium test --controller ~.web.DemandanteController
selenium test --controller ~.web.EmpresaController
selenium test --controller ~.web.ExperienciaController
selenium test --controller ~.web.FormacionController
selenium test --controller ~.web.FormaciondemandanteController
selenium test --controller ~.web.InscripcionController
selenium test --controller ~.web.OfertaController
selenium test --controller ~.web.PuestoController
selenium test --controller ~.web.SedeController
selenium test --controller ~.web.TipoformacionController
selenium test --controller ~.web.UsuarioController
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-26
```

	22:09:40
	// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-27 14:16:32
	web mvc all --package ~.web
	web mvc all --package ~.web
	// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-27 14:23:33

E6 - DOCUMENTO DE PRUEBAS

6.1. Introducción

En nuestro documento de pruebas, vamos a detallar todos los casos de prueba realizados en la implementación de las clases del modelo en las funcionalidades que tiene la aplicación (pruebas unitarias y pruebas funcionales) y en la integración de componentes (pruebas de integración) y fichero pom.xml preparado para el lanzamiento de todas las pruebas.

6.2. Generación de casos de prueba

Los casos de prueba mencionados, son automáticamente generados por el framework "*SpringRoo*", pero para que se creen debemos introducir unos comandos, dependiendo del tipo de pruebas que queramos obtener.

6.2.1. Realización de pruebas unitarias

En este tipo de pruebas, como hemos mencionado anteriormente, son casos de prueba aplicados a la implementación de las clases del modelo. Se ejecutan para cada clase, y se realizan con el siguiente comando para todas las entidades:

test mock --entity ~.domain.Entidad

6.2.2. Realización de pruebas funcionales

Este otro tipo de pruebas, son casos de prueba aplicados a las funcionalidades de la aplicación. Estas también se ejecutan para cada clase, y se realizan con el siguiente comando para todas las entidades:

test integration --entity ~.domain.Entidad

6.2.3. Realización de pruebas de integración

Las pruebas de integración son test aplicados a la integración de los componentes, por lo que no se ejecutan en nuestro entorno, sino en otro distinto. Nosotros en nuestro caso, vamos a realizar estas pruebas con la herramienta Selenium. Las pruebas de integración se ejecutan para el controlador de cada entidad, y se realizan de la siguiente manera para todos los controladores:

selenium test --controller ~.web.EntidadController

6.3. Ejecución de los casos de prueba

6.3.1. Pruebas unitarias y funcionales

Ambos tipos de pruebas se ejecutan de la misma manera, escribiendo el siguiente comando:

mvn test

6.3.2. Pruebas de integración

En cambio, las pruebas de integración, se realizan escribiendo el siguiente comando:
mvn selenium: selenese

Fichero de registros para comandos usados en las pruebas:

```
// Spring Roo 1.3.2.RELEASE [rev 8387857] log opened at 2016-08-20 12:40:30
```

```
test mock --entity ~.domain.Centro
test mock --entity ~.domain.Demandante
test mock --entity ~.domain.Empresa
test mock --entity ~.domain.Experiencia
test mock --entity ~.domain.Formacion
test mock --entity ~.domain.Formaciondemandante
test mock --entity ~.domain.Inscripcion
test mock --entity ~.domain.Oferta
test mock --entity ~.domain.Puesto
test mock --entity ~.domain.Sede
test mock --entity ~.domain.Tipoformacion
test mock --entity ~.domain.Usuario
```

```
test integration --entity ~.domain.Centro
test integration --entity ~.domain.Demandante
test integration --entity ~.domain.Empresa
test integration --entity ~.domain.Experiencia
test integration --entity ~.domain.Formacion
test integration --entity ~.domain.Formaciondemandante
test integration --entity ~.domain.Inscripcion
test integration --entity ~.domain.Oferta
test integration --entity ~.domain.Puesto
test integration --entity ~.domain.Sede
test integration --entity ~.domain.Tipoformacion
test integration --entity ~.domain.Usuario
```

```
selenium test --controller ~.web.CentroController
selenium test --controller ~.web.DemandanteController
selenium test --controller ~.web.EmpresaController
```

```
selenium test --controller ~.web.ExperienciaController
selenium test --controller ~.web.FormacionController
selenium test --controller ~.web.FormaciondemandanteController
selenium test --controller ~.web.InscripcionController
selenium test --controller ~.web.OfertaController
selenium test --controller ~.web.PuestoController
selenium test --controller ~.web.SedeController
selenium test --controller ~.web.TipoformacionController
selenium test --controller ~.web.UsuarioController
```

```
// Spring Roo 1.3.2.RELEASE [rev 8387857] log closed at 2016-08-26 22:09:40
```

E7 - DOCUMENTO DE IMPLANTACIÓN

7.1. Software necesario

- S.O. GNU/Linux basado en Debian o Windows (a partir de Windows 7).
- Gestor de control de versiones Git.
- Gestor de base de datos MySQL.
- Java JDK 7.
- Apache Tomcat 7.0.
- Apache Maven.

Para instalar el software necesario, ejecutamos en nuestra línea de comandos la siguiente orden:

sudo apt-get install -y git mysql-client mysql-server openjdk-7-jre openjdk-7-jdk tomcat7 maven2

Si nos lo pide, introducimos nuestra contraseña de Super Usuario del sistema.

7.2. Instalación y despliegue del sistema

Para poder instalar satisfactoriamente la aplicación web tendremos que ejecutar las siguientes órdenes:

git clone https://github.com/IngenieriaWebUCA/iw2016-facejobs.git

cd iw2016-facejobs

sudo chmod 777 ./facejobs.sh

./facejobs.sh

Al ejecutar el script (.sh) y una vez termine la configuración sobre la Base de Datos, se iniciará otro script desde el cual se lanza la aplicación web en el puerto 8080.

7.3. Scripts de configuración

facejobs.sh

```
#! /bin/bash

echo “Usuario MySQL:”

read user

echo  -n  “Contraseña de $user de MySQL: “

read CLAVE

echo -n “Creando base de datos…”

if mysql -u root   - - password=”$CLAVE”  <  iw2016facejobs.sql;   then

echo  “ *** Base de datos creada con éxito *** ”
```

```
else
echo "*** Se ha producido un error creando la base de datos ***"
fi
echo "Desplegando la aplicación... "
mvn clean compile tomcat7:run
```

7.4. Acceso a la aplicación web

Para acceder a la aplicación web tendrá que entrar introducir la siguiente URL en su navegador preferido.

http://localhost:8080/iw2016-facejobs

Roles predefinidos (MODIFICAR)

Usuario	**Contraseña**	**Rol**
dem	dem	Demandante
emp	emp	GestorEmpresa