GlassWindow

Proyecto número 13

Informe del Proyecto Final de las asignaturas Diseño y Programación Orientado a Objeto y Diseño de Interfaces y Prueba

Vicente Samuel Garófalo Jerez

Carlos Daniel Vilaseca Illnait

La Habana, 2022

### **Resumen**

Este proyecto representa el desarrollo de una aplicación de escritorio utilizada para gestionar la bolsa de ubicación laboral, osea, gestionar las ofertas y demandas de empleos de varias empresas y los candidatos. La aplicación está diseñada como una solución multiplataforma, con una interfaz consistente y una arquitectura modular y desacoplada.

### **Índice**

[Resumen 2](#_Toc1)

[Indice 3](#_Toc2)

[Introducción 4](#_Toc3)

[Planificación de Actividades 5](#_Toc4)

[Desarrollo 6](#_Toc5)

[Tarjetas CRC 6](#_Toc6)

[Diagramas de clases UML 8](#_Toc7)

[Propuestas de Reportes 9](#_Toc8)

[Conclusiones 15](#_Toc9)

### **Introducción**

En una agencia empleadora se desea gestionar la bolsa de ubicación laboral. Dicha agencia se encarga de gestionar las ofertas y demandas de empleo de variadas empresas. De cada empresa se conoce: nombre, dirección, teléfono y sector en el que se encuentra ubicada (turismo, salud, educación, entre otros). Una empresa puede realizar solicitudes para tantos puestos de trabajo como necesite, así como solicitar la eliminación de puestos de trabajo que solicitó. De cada oferta se conoce: ID, rama a la que pertenece, salario del puesto y empresa a la que pertenece esta oferta. Al llegar un candidato a empleado a la agencia se le recogen los siguientes datos: carné de identidad, nombre, sexo, dirección, teléfono, nivel de escolaridad, especialidad, rama en la que desea trabajar (directivo, ingeniero, recursos humanos, entre otras) y años de experiencia en su área de especialidad. En dependencia de la rama en la que desee trabajar, se deben incluir otros datos (un candidato que desee trabajar en turismo debe presentar un certificado de idioma, un candidato que desee incorporarse a la plantilla de custodios debe presentar sus calificaciones en las pruebas de eficiencia física, así como su número de historia clínica). Una vez ingresado los datos al sistema, se comprueba si existen actualmente ofertas de empleo de la rama que le interesa. Se debe crear una funcionalidad en cada candidato, en la cual, dada una oferta, se sepa si el candidato puede o no aspirar a ese empleo. De existir alguna oferta de empleo para el candidato, se programa una cita con la empresa interesada. El candidato puede asistir a todas las entrevistas que desee, pero solo puede asistir a una cita al día. De no existir un empleo que cumpla con los requerimientos del candidato, sus datos quedan almacenados en el sistema. Cuando una empresa realice una solicitud de empleo, los candidatos inscritos en el sistema deben ser revisados a ver si existe alguno con las cualidades necesarias para satisfacer dicha solicitud. La agencia empleadora cuenta con un registro mensual de la planificación de entrevista, donde se tiene por día, para cada empleo que solicita una empresa, si hay planificada una entrevista o no.

El objetivo de este proyecto es crear una aplicación de escritorio multiplataforma para gestionar las ofertas y demandas de empleo de varias empresas.

#### Planificación de Actividades

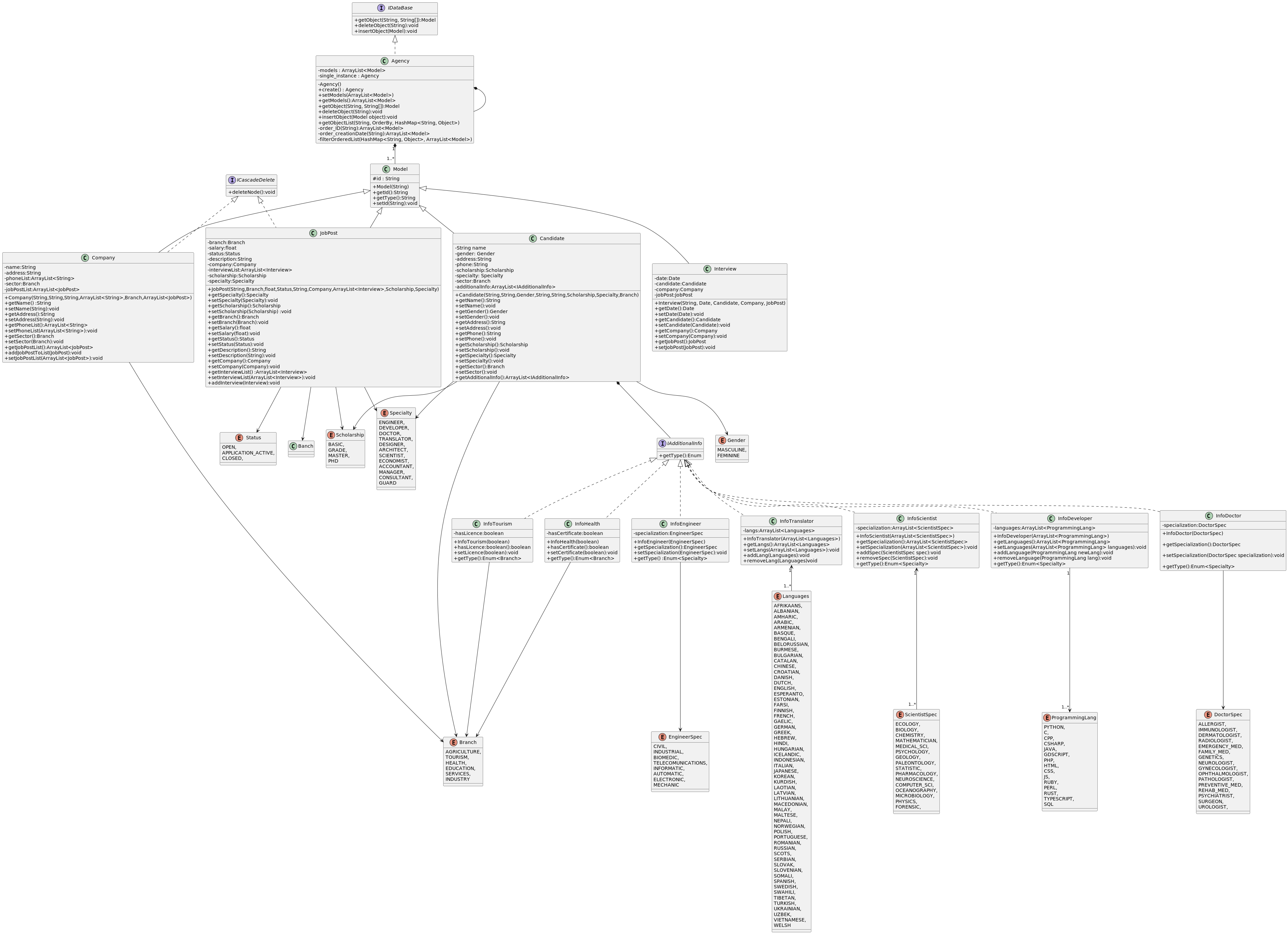
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tareas | Diseñar UML | Elaborar UML | Elaborar CRC | Proponer Reportes | Elaborar Informe del Corte 1 |
| Responsable | Vicente | Carlos Daniel | Carlos Daniel | Vicente y Carlos | Vicente |
| Límite | 2/5/2022 | 9/5/2022 | 13/5/2022 | 12/5/2022 | 13/5/2022 |

### **Desarrollo**

#### Tarjetas CRC

|  |  |
| --- | --- |
| Candidate |  |
| Responsabilidades | Colaboradores |
| Decir su nombre |  |
| Decir su nivel de escolaridad |  |
| Decir la rama en la que desea trabajar |  |
| Decir su carné de identidad |  |
| Decir su dirección |  |
| Decir su teléfono |  |
| Decir su sexo |  |
| Decir su especialidad |  |
| Decir los años de experiencia en su área de especialidad |  |
| Decir dada una oferta de trabajo si puede aplicar o no | JobPost |
|  | |
| Company |  |
| Responsabilidades | Colaboradores |
| Solicitar creación de ofertas de trabajo | Agency |
| Solicitar eliminación de ofertas de trabajo | Agency |
| Decir nombre |  |
| Decir su dirección |  |
| Decir su sector |  |
| Decir su teléfono |  |
| Almacenar los datos de las ofertas de trabajo creadas | JobPost |
|  | |
| JobPost |  |
| Responsabilidades | Colaboradores |
| Decir su ID |  |
| Decir su rama |  |
| Decir su estado |  |
| Decir el salario del puesto |  |
| Decir empresa a la que pertenece |  |
| Decir descripción del puesto |  |
|  | |
| Agency |  |
| Responsabilidades | Colaboradores |
| Decir dada una oferta de trabajo si un candidato puede aplicar | JobPost, Candidate |
| Programar las citas entre empresas y candidatos | Job Offer, Candidate, Company, Enterview |
| Eliminar las ofertas de trabajo solicitadas | Company, JobPost |
| Crear las ofertas de trabajo solicitadas | Company, JobPost |
| Verificar dada una fecha si un candidato puede asistir a una entrevista | Interview, Candidate |
| Dada una empresa decir sus ofertas de trabajo y los candidatos que han aplicado a los mismos en el mes especificado | Company, JobPost, Candidate |
| Dado un día del mes, decir las entrevistas planificadas para ese día | Interview |
| Almacenar los datos de las empresas a las que se les gestiona la bolsa de empleo | Company |
| Almacenar los datos de todos los candidatos registrados | Candidate |
| Almacenar los datos de todas las ofertas de trabajo creadas | Job Post |
| Almacenar los datos de todas las entrevistas planificadas | Interview |
|  |  |
| Interview |  |
| Responsabilidades | Colaboradores |
| Decir la información del candidato | Candidate |
| Decir la información de la oferta de trabajo | Job Post |
| Decir la fecha en la que está planificada | Agency |
| Solicitar la verificación de la fecha de la entrevista | Agency, Candidate |
| Almacenar los datos del candidato que atiende a la entrevista | Candidate |

#### Diagramas de clases UML



#### Propuestas de Reportes

1: Para los candidatos mostrar las ofertas de trabajo que tiene disponibles de acuerdo con sus calificaciones y la rama en la que desea trabajar. Este reporte es importante para servir en la interfaz del candidato solo las ofertas de trabajo que son relevantes para él. En este reporte hay que consultar información de la clase Candidate y usarla para filtrar la lista de JobPost que se encuentra compuesta en la clase Agency.

2: Para una compañía especifica, obtener los nombres de todos los candidatos que han aplicado a sus ofertas de trabajo, con la opción de limitar dicha lista de nombres a los que han aplicado en un mes específico. Vista importante para que las compañías tengan una vista general sobre los candidatos y puedan acceder a rápidamente a sus perfiles para ver sus cualificaciones. Este reporte debe escanear toda la lista de JobPost, para obtener de cada una la lista de Interview, filtrar esa lista por la fecha, y de cada entrevista obtener el Candidate.

3: Tanto para una empresa o un empleado, un reporte que muestre las entrevistas planificadas por cada día del mes. Reporte útil para poblar la vista de calendario. Sería una lista de listas con la siguiente estructura {[Interview1, Interview2, Interview3], [Interview4, Interview5]} donde la posición de índice 0 corresponde al día 1 del mes, y la índice 1 corresponde al día 2.

4: Un reporte que muestre dada una compañía, todas las entrevistas planificadas, agrupadas por oferta de trabajo a la que corresponden. El resultado sería una lista de listas con la siguiente estructura {[Interview1, Interview2], [Interview3], [Interview4]} donde la posición de índice 0 sería la primera JobPost publicada por la empresa dada y así sucesivamente.

Consideraciones para el diseño de interfaces gráficas

#### Descripción de los patrones de Diseño

##### Singleton

El patrón de diseño Singleton (instancia única) está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella.

En este caso el singleton es la Agencia que gestiona la bolsa de empleo.

##### MVC (Model View Controller)

Modelo Vista Controlador (MVC) es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Se trata de un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones, y sobre multitud de lenguajes y plataformas de desarrollo.

* El Modelo que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.
* La Vista, o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos interacción con éste.
* El Controlador, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.

#### Validaciones

Main view (*vista principal*)

* **Search:** sin validación (es un campo opcional).

New Company view (*vista nueva compañía*)

* **Description**: sin validación (campo opcional)
* **Address**: sin validación (campo opcional)
* **Phone**: en caso de usarse, debe ser de 8 dígitos, se hace uso de la excepción *InvalidPhoneException* en caso de que no cumpla con la validación. Es un campo opcional.
* **Sector**: sin validación (es una lista desplegable)

New Candidate view (*vista nuevo candidato*)

* **Candidate ID:** debe ser de 11 dígitos, el tercer y cuarto dígitos que conforman el mes de nacimiento debe ser mayor que 01 y menor que 12, en cuanto al quinto y sexto dígitos que conforman el día del mes de nacimiento, si el mes es 01,03,05,07,08,10 o 12, entonces el día se debe encontrar entre 01 y 31, si el mes es 04,06,09 u 11, entonces el día se debe encontrar entre 01 y 30, en el caso de que sea 02 el mes se validó que el día se encuentre entre 01 y 29, dado que no se puede validar si un año es bisiesto o no con tan solo las dos cifras finales del año, lo mismo ocurre con la cifra 7 del ID, correspondiente al siglo de nacimiento: si nació en el siglo XIX, esté dígito será 9, si el siglo es XX el dígito estará entre 0 y 5, y si es siglo XXI el dígito estará entre 6 y 8, pero de nuevo, no hay como definir el siglo con solo dos cifras del año. Se hace uso de la excepción *InvalidIDException* en caso de que no se cumpla con las validaciones anteriores. En caso de que ya exista un candidato registrado con ese ID se lanza la excepción *DuplicatedIDException*.
* **Full Name:** No debe contener solo espacios, o dejarse el campo vacío, para esta validación se utilizó el método. *isBlank()* que retorna *true* en estos casos, tampoco debe contener números, en caso de ser necesario se deben usar números romanos. Se hace uso de la excepción *InvalidNameException* en caso de que no se cumplan las validaciones.
* **Address:** sin validación (campo opcional).
* **Phone:** en caso de usarse, debe ser de 8 dígitos, se hace uso de la excepción *InvalidPhoneException* en caso de que no cumpla con la validación. Es un campo opcional.
* **Years of Experience:** debe ser un numero mayor que 0, se usa la excepción *InvalidYearsOfExp* en caso de no cumplirse esta validación.
* **Additional Info:** Campo obligatorio en dependencia de la rama y especialidad.

New Job Post view (*vista nueva oferta de trabajo)*

* **Company:** Debe ser una compañía ya registrada, en caso de que no se usa la excepción *ModelNotFoundException*.
* **Description:** sin validación (campo opcional).
* **Salary:** Debe ser un número, mayor que 0, en caso de que no, se hace uso de la excepción *InvalidSalaryException*.
* **Branch:** sin validación (campo opcional).
* **Specialty:** sin validación (campo opcional).

New Company view (*vista nueva compañía*)

* **Description:** sin validación (campo opcional).
* **Address:** sin validación (campo opcional).
* **Phone:** en caso de usarse, debe ser de 8 dígitos, se hace uso de la excepción *InvalidPhoneException* en caso de que no cumpla con la validación. Es un campo opcional.
* **Sector:** sin validación (es una lista desplegable).

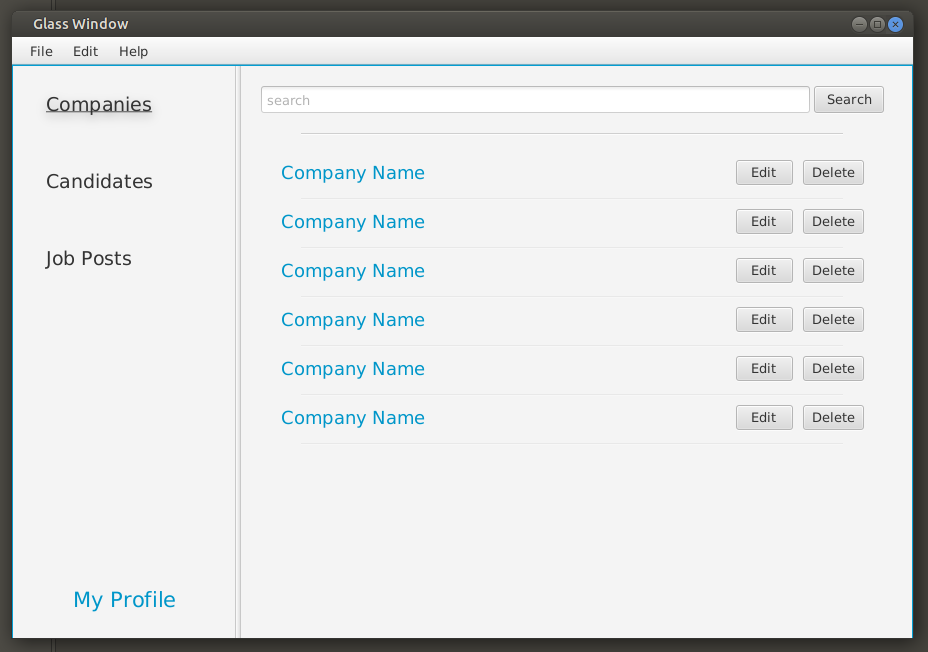
Arrange Interview view (*vista creación de entrevista*)

* **Date:** debe ser una fecha posterior a la fecha actual, en caso de que no se hace uso de la excepción *InvalidDateException*.
* **Time:**
  + **Hour**: debe ser un numero entre 01 y 12.
  + **Minute**: debe ser un numero entre 00 y 59.
* **Candidate**: sin validación (es una lista desplegable).

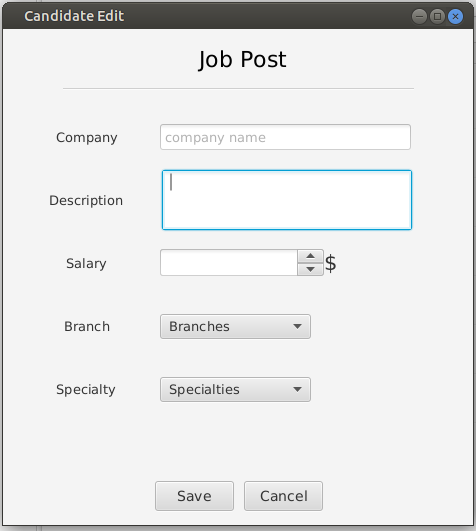
\*Para las vistas de edición de estos modelos se tienen en cuenta las mismas validaciones.

#### Vistas y componentes

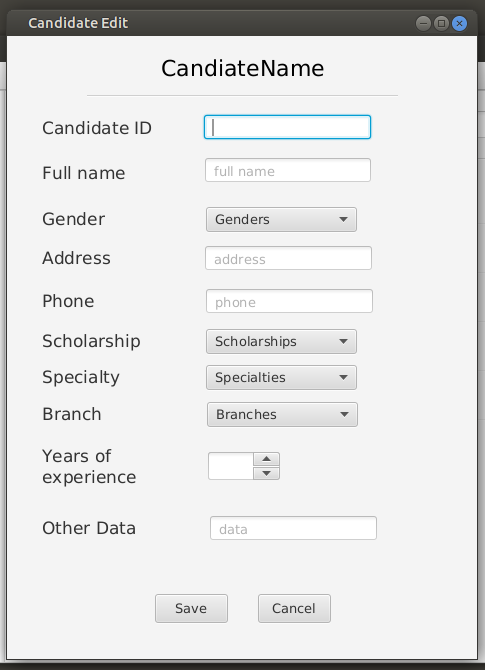
A continuación, unas propuestas de vistas con imágenes de referencia:



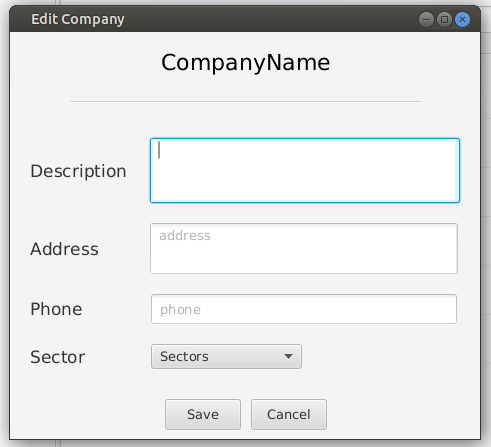
1. Vista principal



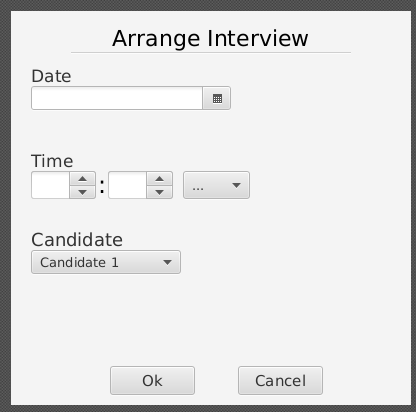
2.Prototipo de Formulario para la creación/edición de JobPosts



3.Prototipo de Formulario para la creación/edición de Candidates



4.Prototipo de Formulario para la creación/edición de Companies



5.Prototipo de Formulario para la creación/edición de Interviews

### **Conclusiones**

Recomendaciones

Bibliografía