

## **IView – Aplicação de Entrevistas e Currículos**

36210, Rui Pedro Gama Franco, 933941557, 36210@alunos.isel.ipl.pt

40278, Diogo Mendes Aires, 962573900, 40278@alunos.isel.ipl.pt

### **Orientadores**

Paula Graça, ISEL, [mgraca@deetc.isel.pt](mailto:mgraca@deetc.isel.pt)

Frederico Ferreira, DoItLean, [frederico.ferreira@doitlean.com](mailto:frederico.ferreira@doitlean.com)

Relatório de progresso realizado no âmbito de Projeto e Seminário,  
do curso de licenciatura em Engenharia Informática e de Computadores  
Semestre de Verão 2017/2018

Maio de 2018



# **Instituto Superior de Engenharia de Lisboa**

Licenciatura em Engenharia Informática e de Computadores

## **IView – Aplicação de Entrevistas e Currículos**

40278 Diogo Mendes Aires

36210 Rui Pedro Gama Franco

---

Orientadores: Paula Graça, ISEL  
Frederico Ferreira, DoItLean

---

---

Relatório de versão beta realizado no âmbito de Projeto e Seminário,  
do curso de licenciatura em Engenharia Informática e de Computadores  
Semestre de Verão 2017/2018

Maio de 2018

# Resumo

O projeto IView tem um objetivo principal, o desenvolvimento de uma aplicação de entrevistas que seja uma mais valia para uma empresa da área de informática e tecnologias e que permita ao candidato, ter um papel mais envolvido e simplificado.

Para a empresa, a aplicação irá não só garantir uma administração simples de entrevistas, candidatos, projetos, cliente e vagas, como também uma forma de se apresentar aos candidatos, demonstrando o que mais valida nos mesmos e como estes mais facilmente se podem apresentar. Por outro lado, irá oferecer aos candidatos, uma forma simples, interativa e apelativa de administrar as suas informações e realizar aplicações a vagas, que conseguira visualizar como também pesquisar tendo em conta os seus interesses.

Apesar do projeto ser focado no desenvolvimento de uma aplicação *web*, existirá também uma aplicação *mobile* que servirá como um suporte para os candidatos.

Durante o processo de desenvolvimento desta aplicação também iremos colocar os nossos conhecimentos da arquitetura de desenvolvimento OutSystems em prática, arquitetura cujo o conhecimento é cada vez mais requisitado no mercado informático e tecnológico Português e estrangeiro, o que torna qualquer experiência sobre a mesma indispensável para o futuro.

**Palavras-chave:** OutSystems; *Widgets*; Candidatos; Entrevistas; Currículos; App;



# Abstract

Abstract text (1 page).

**Keywords:** sorted keyword list, delimited by;.



# Índice

RESUMO .....	IV
ABSTRACT .....	VI
LISTA DE FIGURAS .....	X
LISTA DE TABELAS .....	XII
<b>1. INTRODUÇÃO .....</b>	<b>1</b>
1.1 ENQUADRAMENTO .....	1
1.2 OBJETIVOS .....	1
1.3 ORGANIZAÇÃO DO DOCUMENTO .....	2
<b>2. FORMULAÇÃO DO PROBLEMA .....</b>	<b>3</b>
<b>2.1 ESTADO DA ARTE .....</b>	<b>3</b>
<b>2.2 DESCRIÇÃO DO PROJETO .....</b>	<b>5</b>
2.2.1 APLICAÇÃO WEB.....	6
2.2.2 APLICAÇÃO MOBILE .....	10
2.2.3 REQUISITOS NÃO FUNCIONAIS .....	11
2.3 DESCRIÇÃO DA PLATAFORMA, MODELOS DE DESENVOLVIMENTOS.....	11
2.3.1 MODELOS DE DESENVOLVIMENTO .....	13
<b>3 SOLUÇÃO PROPOSTA .....</b>	<b>16</b>
3.1 ARQUITETURA DO PROJETO.....	16
3.2 MODELO ENTIDADE-ASSOCIAÇÃO DA BASE DE DADOS .....	17
3.2.1 <i>Utilizadores</i> .....	17
3.1.1 <i>Vagas</i> .....	20
3.1.2 <i>Eventos</i> .....	22
3.2 WIREFRAMES DO PROJETO .....	23
3.3 DESENVOLVIMENTO WEB.....	41
3.4 DESENVOLVIMENTO MOBILE .....	56
<b>4. AVALIAÇÃO EXPERIMENTAL .....</b>	<b>59</b>
<b>5. CONCLUSÕES.....</b>	<b>60</b>
REFERÊNCIAS .....	61
<b>A.1 DIAGRAMAS DA APLICAÇÃO .....</b>	<b>62</b>
<b>A.2 MODELOS DE DADOS .....</b>	<b>64</b>





# Lista de Figuras

Figura 1 – Visão geral do IView.....	6
Figura 2 - Casos de utilização, Utilizadores Não Registados. ....	7
Figura 3 - Casos de utilização, Colaborador 1.....	8
Figura 4 - Casos de utilização, Colaborador 2.....	9
Figura 5 - Casos de utilização, Candidato. ....	10
Figura 6 - Casos de utilização, Candidato Mobile.....	11
Figura 7 - Arquitetura OutSystems [5]. ....	12
Figura 8 - As Layers de Desenvolvimento Outsystems.....	13
Figura 9 - Estrutura 4 Layer Canvas.....	16
Figura 10 - Modelo EA, Candidatos.....	18
Figura 11 - Modelo EA, Vagas.....	20
Figura 12 - Modelo EA, Eventos.....	22
Figura 13 - Frames, Home Page .....	23
Figura 14 - Frames, Menu de Colaboradores. ....	24
Figura 15 - Frames, Menu de Candidatos.....	24
Figura 16 - Frame, MoreInfo Values Tab.....	25
Figura 17 - Frame, MoreInfo Contacts Tab.....	25
Figura 18 - Frame, MoreInfo Patnerships Tab. ....	25
Figura 19 - Frame, MoreInfo Personnel Tab. ....	26
Figura 20 - Frame, MoreInfo Client Tab. ....	26
Figura 21 - My Curriculum, Show.....	27
Figura 22 - Frame, MyCurriculum, Edit.....	27
Figura 23 - Frames, MyProfile. ....	29
Figura 24 - Frames, MyProfile PopUp. ....	30
Figura 25 - Frames, Forms.....	31
Figura 26 - Frames, Applications. ....	32
Figura 27 - Frames, Vacancies. ....	33
Figura 28 - Frames, Events.....	33
Figura 29 - Frames, CheckCV.....	34
Figura 30 - Frames, Candidate.....	35
Figura 31 - Frames, Event. ....	36
Figura 32 - Frames, Application.....	37
Figura 33 - Frames, Vacancy General Information Tab.....	38
Figura 34 - Frames, Vacancy Steps Tab.....	38
Figura 35 - Frames, Vacancy Tools and Languages Tab.....	38

Figura 36 - Frames, FormEditAdd.....	39
Figura 37 - Frames, VacancyAdd Form Chosse PopUp.....	39
Figura 38 - Frames, Vacacncy Add Main Tab.....	40
Figura 39 - Frames, Vacacncy Add Sterps Tab. ....	40
Figura 40 - Frames, Vacacncy Add Tools Tab.....	40
Figura 41 - Frames, Vacacncy Add Languages Tab.....	41
Figura 42 - Paginação .....	42
Figura 43 – Pesquisa.....	44
Figura 44 - Adicionar Availability.....	45
Figura 45 - Adicionar ao Profile .....	47
Figura 46 - Remover do Profile .....	47
Figura 47 – Aceitar Currículo.....	48
Figura 48 – Formação dos Events para um mês .....	49
Figura 49 – Alteração do calendário de mês para semana.....	50
Figura 50 - Formação dos Events para uma semana.....	51
Figura 51 – Inicio de criação de um Event, ou escolha de um Event existente .....	51
Figura 52 – Adição de um novo Event .....	53
Figura 53 – Adição de Vacancy.....	54
Figura 54 – Geração de percentagens para o Pie Chart .....	55
Figura 55 – Escolha de secção do Pie Chart.....	56
Figura 56 - Registro de utilizador na OneSignal .....	57
Figura 57 - Sincronismo Read-Only .....	58

# Lista de Tabelas

Tabela 1 - Application .....	66
Tabela 2 - ApplicationCurrentStep .....	66
Tabela 3 – ApplicationInterview .....	66
Tabela 4 – CandidateAcademics.....	66
Tabela 5 – CandidateAppDeveloped .....	66
Tabela 6 - CandidateAvailability.....	67
Tabela 7 – CandidateCurriculum.....	67
Tabela 8 - CandidateFormations.....	68
Tabela 9 – CandidateFrameworks .....	68
Tabela 10 - CandidateIDE .....	68
Tabela 11 – CandidateLanguage.....	68
Tabela 12 - CandidateTechnology.....	69
Tabela 13 - CandidateWorkExp .....	69
Tabela 14 – Capacity .....	69
Tabela 15 – Client.....	69
Tabela 16 – Discussion.....	69
Tabela 17 – Employee .....	70
Tabela 18 - Event.....	70
Tabela 19 - EventGroup.....	70
Tabela 20 - EventGroupMember .....	70
Tabela 21 - EventType.....	70
Tabela 22 - Form .....	71
Tabela 23 - FormQuestion .....	71
Tabela 24 - Interview .....	71
Tabela 25 - InterviewAnswer .....	71
Tabela 26 - JobTitle .....	71
Tabela 27 – Language.....	71
Tabela 28 - Location.....	72
Tabela 29 - Photo.....	72
Tabela 30 –Project .....	72
Tabela 31 - ProjectClient.....	72
Tabela 32 –ProjectResponsible.....	72
Tabela 33 - SpontaneousCurriculum .....	72
Tabela 34 - SpontaneousCurriculumFile.....	73

Tabela 35 – StepState .....	73
Tabela 36 - StepType.....	73
Tabela 37 – User.....	73
Tabela 38 – Tools .....	73
Tabela 39 – TypeLanguage.....	74
Tabela 40 – Vacancy .....	74
Tabela 41 - VacancyLanguages .....	74
Tabela 42 - VacancyStep .....	74
Tabela 43 –VacancyTools .....	74



# 1. Introdução

Neste capítulo iremos introduzir algumas ideias fundamentais sobre o projeto IView, inclui algumas inspirações e enquadramento no mercado de informática atual, os objetos gerais, a serem aprofundados em capítulos seguintes da aplicação e a organização do relatório.

## 1.1 Enquadramento

Muitas empresas de tecnologia portuguesas, têm um processo de entrevista bastante arcaico e limitado, não colocando os seus conhecimentos tecnológicos em utilização. Normalmente este processo envolve a transferência de ficheiros importantes, como por exemplo currículos e dossiês de competências<sup>1</sup> por correio eletrónico. Neste processo, qualquer marcação de entrevista é realizada por telemóvel, tendo como consequência a dependência do candidato num colaborador da empresa, que caso se esqueça, ou não o informe de forma correta, a entrevista poder-se-á perder por falha de comunicação.

Uma das raras exceções que não se limita ao processo acima descrito, é a consultoria Mind Source, que fornece o “Portal de Emprego” [1]. Contudo, este portal oferece pouco mais que a capacidade de guardar ficheiros e realizar candidaturas a cargos disponíveis. Outro exemplo é a aplicação “My Profile” [2] desenvolvida pela empresa de recursos humanos Randstad, que disponibiliza um formato simples e interativo para descrever o currículo e dossiê de capacidades, mas nada mais.

Atendendo às limitações consideradas, nasceu o projeto IView, que tem como objetivo principal, não só expandir as funcionalidades das duas aplicações referidas, como também incluir outras funcionalidades uteis para as empresas que o utilizem.

O desenvolvimento do IView utilizou como base uma empresa de tecnologia fictícia a que chamámos PS Tec, servindo de enquadramento para outras possíveis empresas.

## 1.2 Objetivos

O projeto IView foca-se no desenvolvimento de duas aplicações, *web* e *mobile*, que funcionam em conjunto para garantir a comunicação entre candidatos e entrevistadores de uma forma simples e acessível, desde o momento em que uma nova oferta para um cargo na empresa é criada, até ao momento final da eventual contratação.

A aplicação *web*, aplicação central onde estão desenvolvidas grande parte das funcionalidades, garante acessos de leitura e escrita à base de dados onde serão guardados, entre outros dados:

---

<sup>1</sup> Ficheiro em que um candidato consegue demonstrar as suas competências em tecnologias, idiomas e onde, normalmente, pode incluir aplicações que desenvolveu

- Currículos, dossiês de competências e disponibilidades<sup>2</sup> de candidatos;
- Informações gerais dos colaboradores que participam na aplicação;
- Clientes e projetos com os mesmos;
- Vagas a novos cargos na empresa - Cada vaga inclui um processo de entrevista;
- Aplicações a vagas, incluindo a situação da mesma no processo de entrevista;
- Entrevistas e outros eventos marcados;
- Resultados de entrevistas que já ocorreram;
- Informações gerais da empresa.

A aplicação *mobile*, serve como apoio ao candidato para interagir com a empresa e para aceder ao seu processo de candidatura.

### **1.3 Organização do documento**

O restante relatório tem a seguinte composição de capítulos:

- Formulação do Problema – contém informação sobre da aplicação IView, incluído referências para algumas aplicações semelhantes, detalhe dos requisitos funcionais e plataforma tecnológica onde o projeto é desenvolvido;
- Solução Proposta - neste capítulo está incluída a arquitetura da aplicação, o modelo geral da base de dados, algumas ideias gerais da interface gráfica e alguns dos principais fluxos de implementação.

---

<sup>2</sup> Por disponibilidades, referimos a janelas de tempo associados a dias de semana que o candidato considerar fiáveis para participar numa entrevista



## 2. Formulação do Problema

Neste capítulo, na secção 2.1, é verificada a utilidade e exemplos de aplicações semelhantes, incluindo algumas limitações de tais aplicações e como a IView as complementa. A secção 2.2 inclui uma breve descrição do projeto, apresentando as várias funcionalidades do IView. Por último, na secção 2.3, é descrita a plataforma de desenvolvimento OutSystems, com a qual o projeto foi implementado.

### 2.1 Estado da arte

Existe um considerável número de empresas, pelo menos as da área das tecnologias, que utilizam uma aplicação semelhante ao IView, mas com algumas limitações que pretendemos colmatar no IView.

A mais marcante é a ausência de interação entre o candidato e o colaborador da empresa. As aplicações que encontramos, servem normalmente para a empresa gerir candidatos, os seus currículos e a partilha dos mesmos com clientes, não incluindo qualquer tipo de comunicação entre as duas partes. Quando o candidato necessita de realizar qualquer alteração ao seu currículo e/ou dossiê de capacidades, tem que indicar tais alterações a um colaborador por *email*, o que não garante que os ficheiros atualizados sejam utilizados em próximas partilhas com clientes.

Por outro lado, as aplicações acima referidas, funcionam unicamente com dossiês de capacidades em formato PDF. Assim, não só é dificultada a atualização do dossiê, como não é incluído qualquer mecanismo para filtrar informação desnecessária ou repetida, nem é indicado que competências tecnológicas a empresa procura, ou que considera importantes. Muitas vezes as bases do dossiê (*template* seguido na criação de um dossiê), não expressam que informações podem ser importantes para incluir no documento. Normalmente é requerido que o candidato expresse as suas capacidades em níveis, mas não é estabelecido o que cada nível implica.

Mesmo que esta falta de informação não pareça importante, deve ser tomado em conta que o candidato possa ser alguém que esteja a entrar pela primeira vez no mercado de trabalho, e que pode ainda não saber como tratar essa situação.

As aplicações acima referidas, também não incluem qualquer tipo de funcionalidade de marcação de entrevistas, embora algumas utilizem o Google Calendar. Aquelas que não utilizam nenhum mecanismo semelhante, arriscam que na ocorrência de qualquer falha de comunicação entre as duas partes interessadas, possa causar a não realização da entrevista. Aquelas que utilizam o Google Calendar, conseguem ultrapassar esta limitação, mas o candidato fica completamente dependente de receber um convite para a entrevista, o qual pode não chegar. Um exemplo real que se passou com um dos alunos que se encontra a realizar este projeto, demonstra esse problema. Foi marcada uma entrevista, mas o convite nunca chegou ao próprio, e só depois de conseguir

entrar em contacto com o colaborador da empresa que geria o processo, é que foi encontrado o erro, ou seja, o convite tinha sido enviado para o *email* de outro indivíduo.

Na pesquisa que efetuámos para o desenvolvimento do IView, encontrámos, contudo, duas aplicações que permitem a comunicação entre candidato e colaborador. A primeira, já referida anteriormente, o “Portal de Emprego” [1] da Mind Source, permite aos candidatos duas funcionalidades principais:

- Gerir ficheiros, currículos e dossiês, quebrando assim a necessidade de comunicar com um colaborador cada vez que se deseja realizar uma atualização de um dos ficheiros;
- Verificar as vagas disponíveis e aplicar-se às mesmas.

Contudo, estas funcionalidades têm limitações importantes, como por exemplo, não é indicado quais os ficheiros que o candidato deve incluir, bem como a sua estrutura. Já para as aplicações, a informação fornecida sobre as mesmas é bastante limitada, sendo que só indica que aplicações estão a ocorrer, mas mais nada.

Outro exemplo é uma aplicação de gestão de perfil produzida pela Randstad [2], a qual não inclui qualquer capacidade de verificação de vagas e candidaturas as mesmas, estando limitado unicamente à construção e gestão do perfil do candidato. Esta aplicação permite, contudo, o desenvolvimento e gestão do currículo e dossiê de capacidades de uma forma bastante fácil.

De todas as formas, a Randstad não é uma empresa na área das tecnologias, incluindo por isso o perfil do candidato, um extenso raio de informações que torna a sua formação um pouco mais complexa. Pensando na adequação do perfil para a área de informática, a aplicação da Randstad, não inclui no dossiê de capacidades, a lista de aplicações desenvolvidas pelo candidato, que reflete de alguma forma, a sua experiência e tipo de responsabilidades que tem vindo a assumir na sua experiência profissional.

A aplicação não inclui também qualquer tipo de informação sobre vagas e candidaturas as mesmas, o que torna o início do processo de entrevista só possível pela parte da empresa, ou através do recurso do candidato a outras fontes, para dessa forma descobrir eventuais vagas.

Apesar das limitações acima enumeradas, a aplicação da Randstad proporcionou-nos parte da inspiração gráfica para o projeto, sendo que, apesar de não termos a certeza, acreditamos que esta aplicação foi desenvolvida utilizando a plataforma OutSystems.

Existem ainda duas outras aplicações que estão de alguma forma, relacionadas com a IView, mas ao contrário das referidas anteriormente, estas não são limitadas a uma só empresa e são mais adequadas para a partilha de informação sobre perfis e vagas na área.

A primeira a que nos referimos, trata-se da rede social LinkedIn [3], que permite gerir um perfil análogamente à aplicação da Randstad, mas de uma forma menos interativa. Esta aplicação, também não inclui qualquer tipo de informação sobre vagas, aplicações ou entrevistas, mas em si

é uma aplicação que permite a partilha fácil de informação de perfis adequados a empresas variadas.

Quase numa vertente completamente oposta, a aplicação ITJobs [4], fornece uma extensa fonte de informação sobre empregos, formações e eventos na área de tecnologias e informática, não inclui qualquer possibilidade de gestão de perfis, não permitindo também o estabelecimento de registos de utilizador, ou qualquer processo de entrevista. As informações são registadas pela empresa que proporciona a vaga, não sendo acessíveis pela ITJobs.

Tal como referido no início desta secção, as aplicações semelhantes ao IView, mesmo que limitadas, são bastante comuns em empresas na área da informática e tecnologias. Tal facto é indicativo que o IView pode ser bastante útil para esta área, procurando ir mais além na gestão da informação dos candidatos, bem como promovendo a facilidade de interação destes com as empresas. Por exemplo, utilizando de novo a experiência pessoal de um dos autores do projeto, em algumas entrevistas na área, este constatou que:

- Um extenso número de empresas da área de informática e tecnologias, já dispõem de aplicações do tipo da IView;
- A possibilidade dos candidatos conseguirem interagir com as empresas através dessas aplicações, é uma funcionalidade que não foi considerada, mas que seria desejável - normalmente os entrevistadores referem que no futuro a empresa para a qual trabalham, deverá iniciar um projeto interno para criar tal funcionalidade.

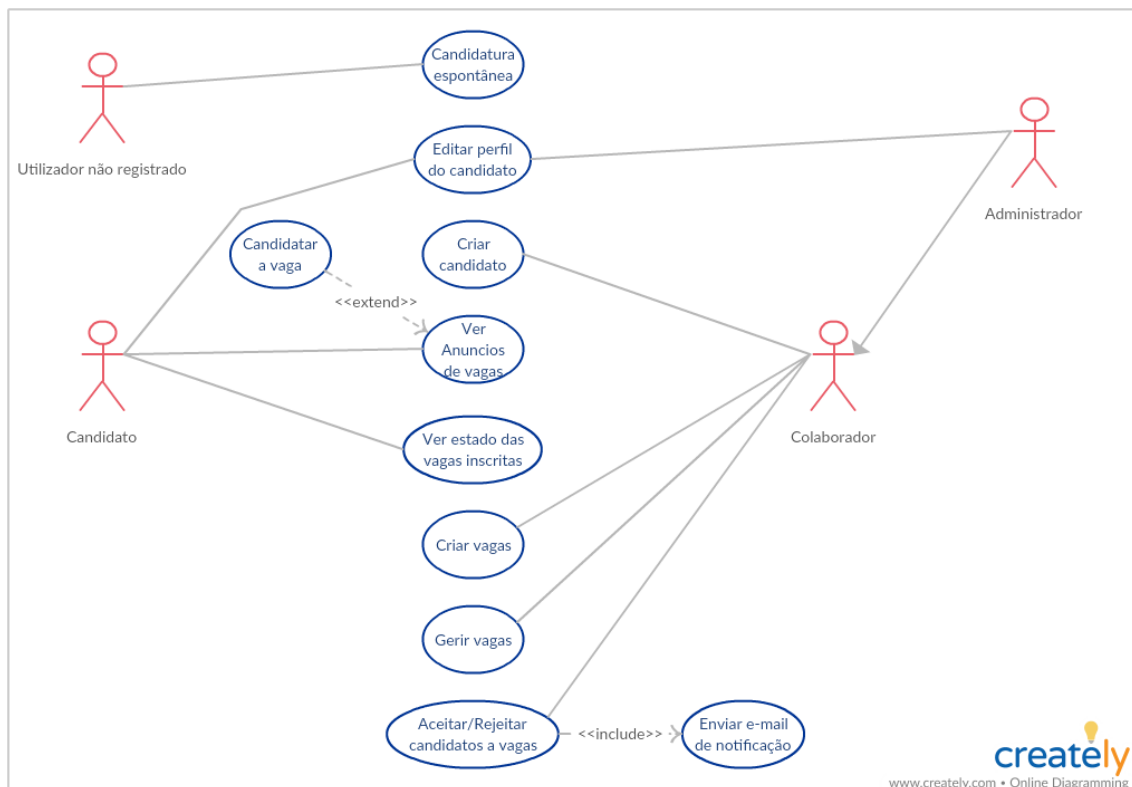
## **2.2 Descrição do Projeto**

O projeto IView está dividido em duas aplicações, tendo cada aplicação um conjunto individual de requisitos funcionais, de acordo com o seu objetivo específico.

Na Figura 1 está representado um caso de utilização que descreve de forma geral as funcionalidades das aplicações para os vários perfis de utilizador.

Nesta figura é possível verificar a existência de quatro perfis de utilizador:

- Utilizador Não Registrado – o nível mais baixo que existe, com funcionalidades limitadas;
- Candidato – representa o candidato às vagas disponibilizadas pelas empresas, sendo os seus acessos limitados a leitura e escrita sobre a informação a si associada, com a exceção dos anúncios das vagas, que são públicos;
- Colaborador – representa um colaborador na empresa, possuindo um maior conjunto de acesso, não limitado a instâncias associadas ao próprio;
- Administrador – este nível inclui todas as funcionalidades do colaborador, mais alguns extras que possibilitam a administração de utilizadores



**Figura 1 – Visão geral do IView.**

Nas restantes 3 subsecções, descrevemos as funcionalidades de cada aplicação (*web* e *móvel*), apresentando casos de utilização, seguidos de uma ideia geral dos mesmos. Descrevemos também, algumas interações *back-end* que ocorrerão no decorrer das funcionalidades, referindo as entidades da base de dados, que podem ser verificadas na secção 3.2 que apresenta o respetivo modelo entidade-associação.

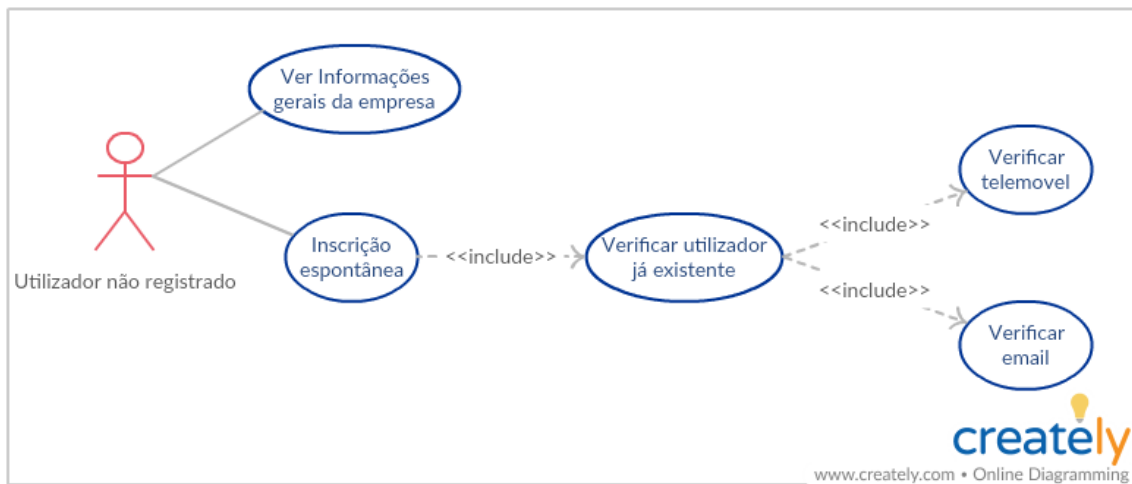
### **2.2.1 Aplicação web**

A vertente *web* do projeto, tem como foco a disponibilização de diferentes funcionalidades aos dois perfis principais participantes na aplicação: candidatos e empresa.

Também inclui, a publicação de informação geral da empresa, incluindo:

- Contactos;
- Missão e valores da empresa;
- Anúncios para novas colocações;
- Informação geral sobre os colaboradores;
- Possibilitar a realização de candidaturas espontâneas.

Estas funcionalidades, as funcionalidades dos utilizadores não registrados, podem ser verificadas na Figura 2.

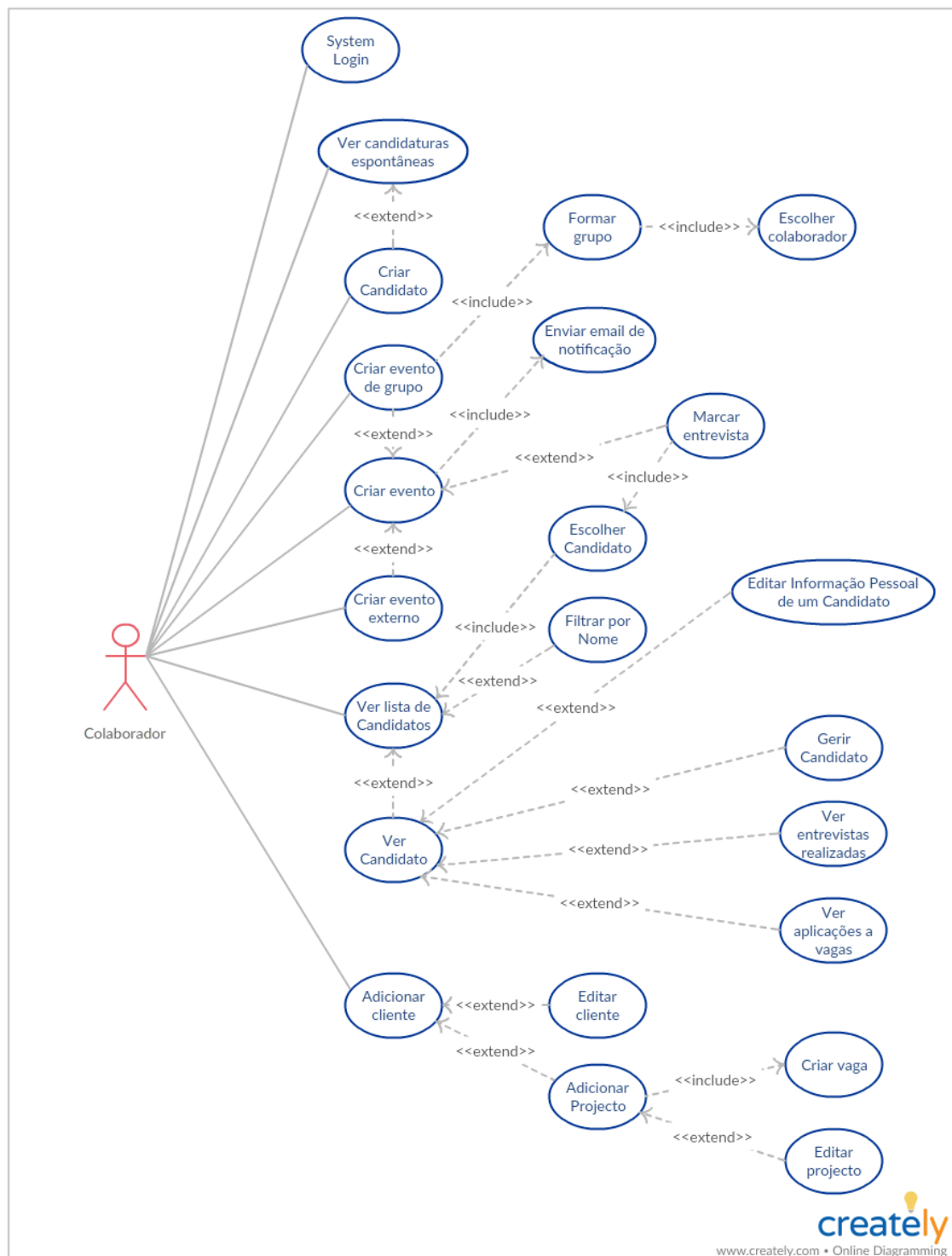


**Figura 2 - Casos de utilização, Utilizadores Não Registrados.**

Os colaboradores da empresa, que participem no processo de entrevista são capazes de:

- Estabelecer novos candidatos em função de candidaturas espontâneas;
- Formar os processos de entrevista para novos cargos na empresa como também os formulários para as entrevistas;
- Gerir os vários passos que compõem um processo de candidatura e os respetivos estados de acordo com a sua progressão;
- Verificar candidaturas a novos cargos, permitindo promover um candidato para o próximo passo do processo, ou terminar, sendo o candidato informado automaticamente, por *email*;
- Marcar novas entrevistas e verificar entrevistas já marcadas;
- Guardar entrevistas, cuja informação poderá ser utilizada em futuras considerações.

A Figura 3 e a Figura 4 ilustram as funcionalidades acima enumeradas, de uma forma mais abrangente e detalhada, mostrando a segunda figura as funcionalidades mais centradas nas entrevistas e outros eventos, sendo as restantes mostradas pela primeira.



**Figura 3 - Casos de utilização, Colaborador 1.**

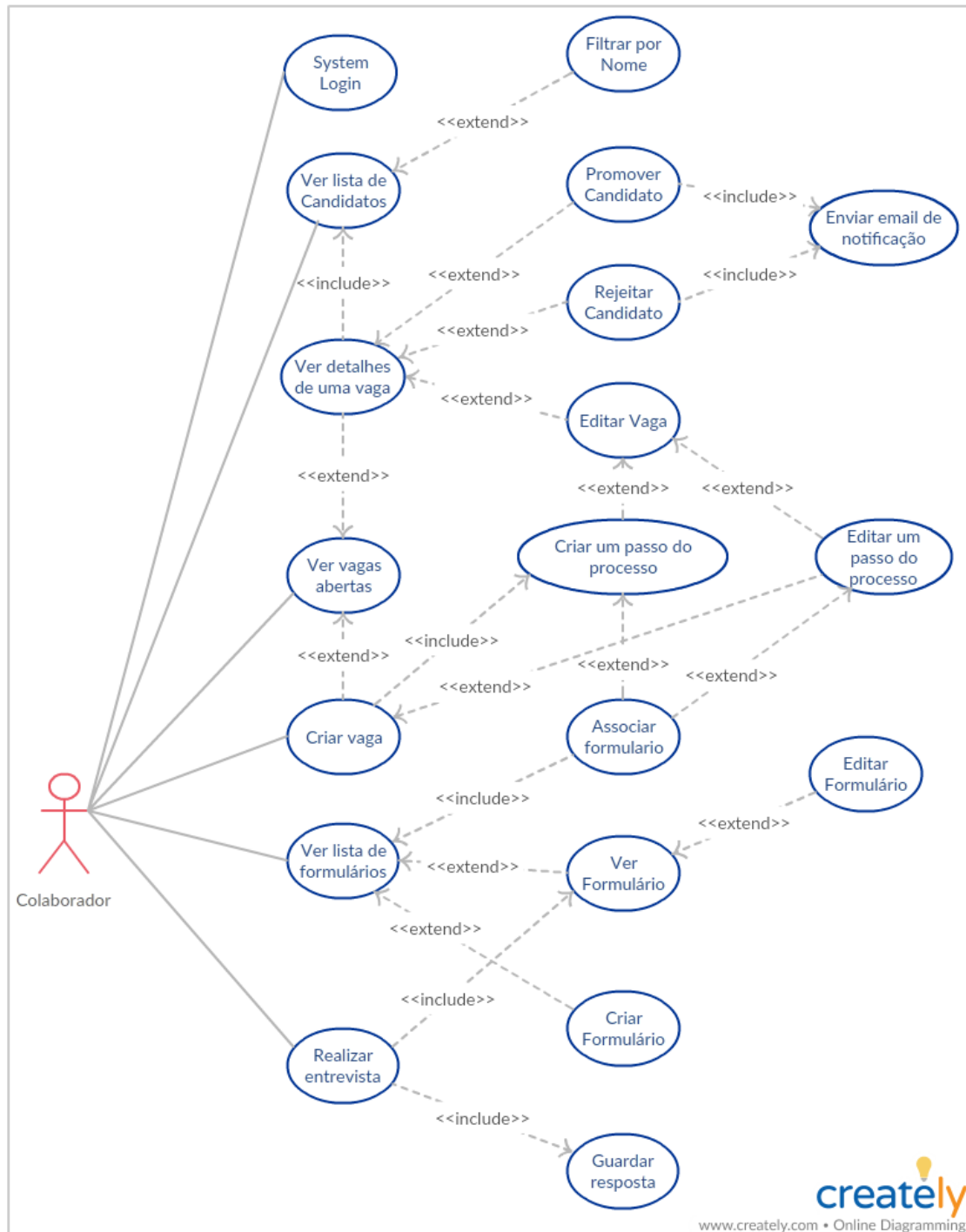


Figura 4 - Casos de utilização, Colaborador 2.

Os candidatos a novos cargos na empresa são capazes de:

- Gerir o seu currículo e dossiê de capacidades, com limitações e indicações estabelecidas pela empresa - o dossiê poderá incluir informação como: experiência com tecnologias (linguagens), *frameworks*, formação e educação na área, etc.;
- Verificar/procurar anúncios, aos quais se podem candidatar;
- Averiguar o estado de processos de entrevistas em que participam no momento;
- Verificar entrevistas marcadas, incluindo informações gerais da entrevista.

As funcionalidades referidas na aplicação *web*, podem ser verificadas na Figura 5.

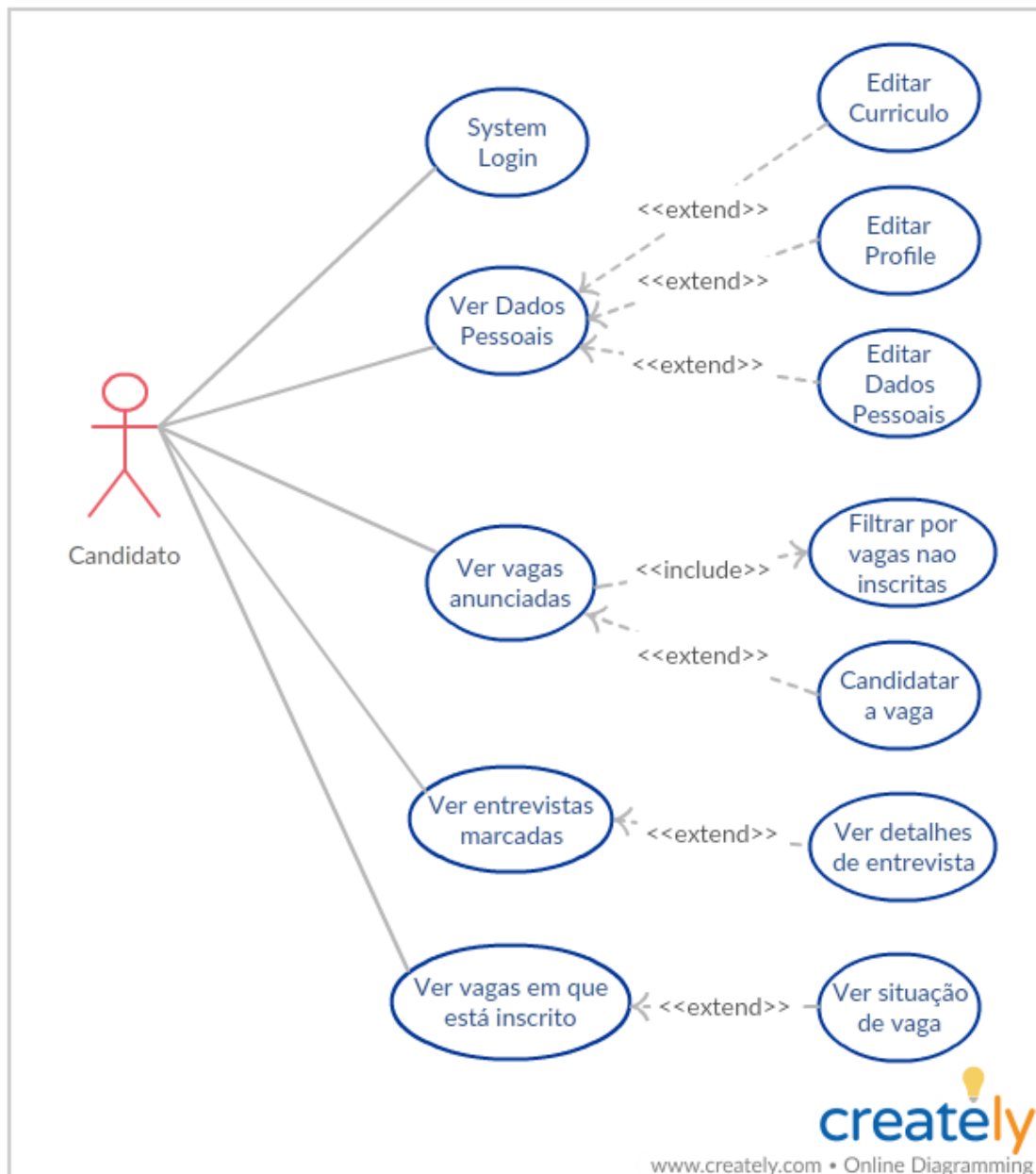


Figura 5 - Casos de utilização, Candidato.

### 2.2.2 Aplicação *mobile*

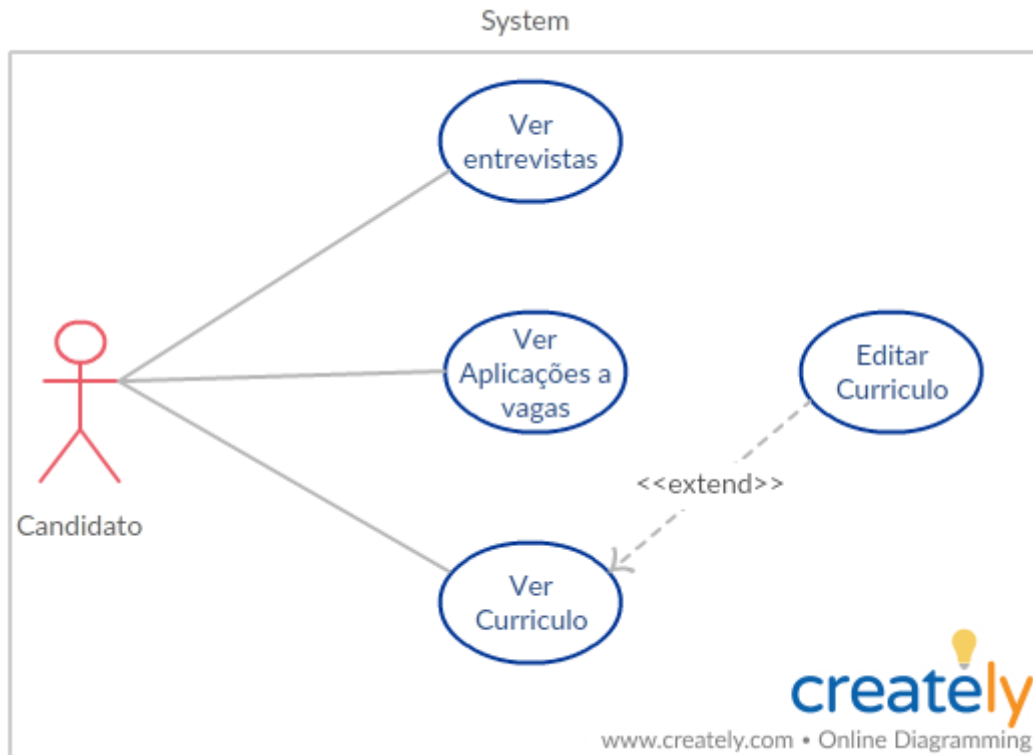
A vertente *mobile* da aplicação, tem como objetivo ser uma fonte de informação conveniente e *lightweight* para os candidatos que utilizam o IView, optando-se assim por manter nesta vertente, as funcionalidades estritamente necessárias a este fim. Através desta aplicação, os candidatos são capazes de:

- Verificar e editar a informação no seu currículo;
- Ver o estado de processos aos quais se candidataram - receber notificações quando tais processos sofrem alteração, ou seja, se foram rejeitados ou passaram à próxima fase;



- Ver entrevistas marcadas e respectivas informações.

Estas funcionalidades, podem ser verificadas Figura 6.



**Figura 6 - Casos de utilização, Candidato Mobile.**

A necessidade de uma aplicação mais leve deve-se, em parte, ao problema de sincronismo de aplicações moveis desenvolvidas na OutSystems, ao manter só alguns acessos de leitura essenciais e ainda menos acessos de escrita, que em si são restringidos a informações acessíveis unicamente a um candidato, a aplicação desenvolvida pode tomar partido dos padrões de sincronização mais simples, este problema e as suas possíveis soluções são abordadas, numa forma simplificada, na subsecção 2.3.

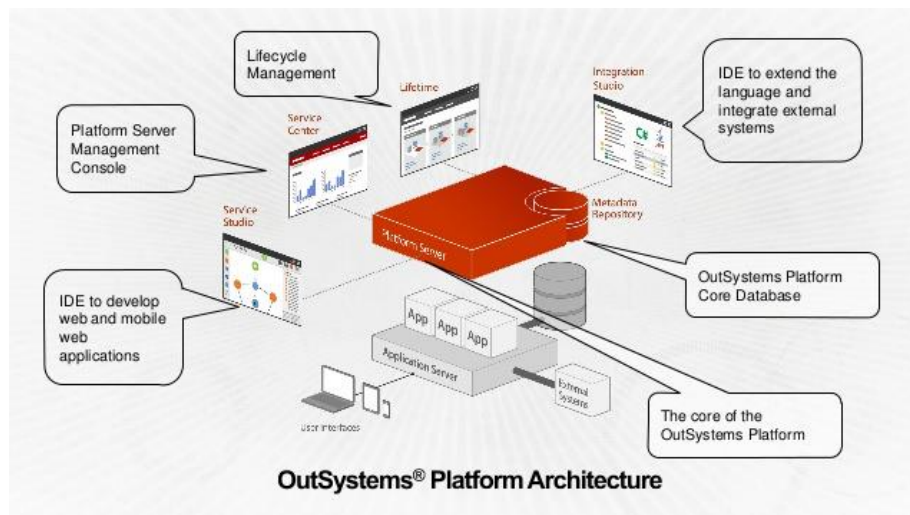
### **2.2.3 Requisitos Não Funcionais**

Para futuras melhorias, são ponderadas as seguintes funcionalidades:

- Permitir que a aplicação funcione com várias empresas;
- Usar a *API* do *Google Calendar* para marcação das entrevistas;
- Utilizar o *Google Maps* para demonstrar a localização da empresa;
- Utilizar uma base de dados *SQL* externa.

## **2.3 Descrição da plataforma, modelos de desenvolvimentos**

Para a implementação de ambas as aplicações do IView, foi utilizada a plataforma de desenvolvimento OutSystems, cuja arquitetura se encontra ilustrada na Figura 7.



**Figura 7 - Arquitetura OutSystems [5].**

A escolha desta plataforma foi devido a duas razões fundamentais:

- Sendo capaz de desenvolver as duas vertentes (*web* e *mobile*), é possível aproximar as duas interfaces gráficas;
- Permite o desenvolvimento duma interface visualmente agradável e simples de utilizar, sem ter de se colocar muito tempo no desenvolvimento da mesma.

A plataforma OutSystems permite um desenvolvimento rápido das aplicações através de um modo visual quer na aplicação *web* quer na aplicação *mobile*. As aplicações são publicadas utilizando a *framework* .Net (linguagem C#). No *back-end*, utiliza uma base de dados SQL Server e padrões JavaScript para produzir o *front-end*.

Para tal desenvolvimento simples e rápido, a OutSystems inclui três servidores fundamentais e especializados numa área diferente. O primeiro dos três servidores é o *Code Generator* que essencialmente realiza uma leitura da interface gráfica de desenvolvimento e a partir de informação da mesma:

- Verifica dependências externas e aplica otimizações a aplicação;
- Gera código nativo;
- Gerar *proxies* de integração;
- Agrupar as capacidades de gerir de sessão, autenticação e configurações para a aplicação;
- Adicionar recursos de registo e monitoramento a aplicação.

No *Deployment Service* os componentes gerados de uma aplicação são aplicados num server de aplicações, garantido que a mesma é instalada consistentemente em cada servidor *front-end* da *farm* de servidores da organização dona da aplicação

Por ultimo existe o *Application Service* que fornece dois serviços importantes;

- O *Scheduler Service*, que administra a execução de *threads* planeadas;

- O *Log Service*, responsável por gerir erros, auditorias e o desempenho de uma aplicação.

### 2.3.1 Modelos de Desenvolvimento

A OutSystems *Service Studio*, onde é possível desenvolver tanto aplicações *web* como *mobile*, divide o desenvolvimento em três *layers*, cada uma dividida num conjunto de componentes. Estes *layers* são verificados na Figura 8, note-se que no caso desta figura os *layers* são os de desenvolvimento de aplicações *mobile*.

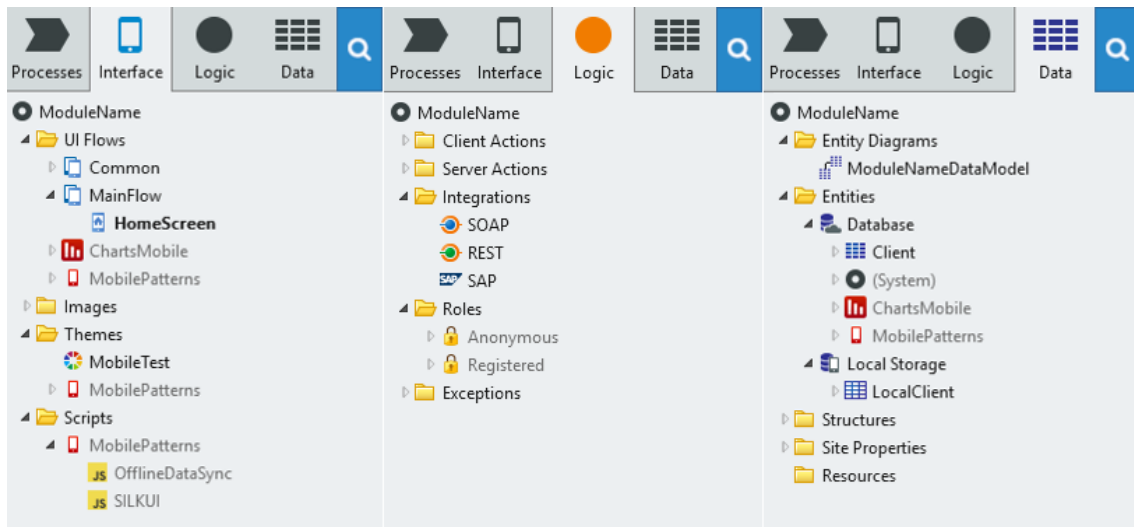


Figura 8 - As Layers de Desenvolvimento Outsystems.

Primeiro temos a *Interface Layer*, que se centra na interface gráfica da aplicação e que é composta pelos seguintes componentes:

- *UI Flows* – onde estão incluídos tanto os ecrãs que compõem a interface gráfica da aplicação desenvolvida, como também padrões utilizados;
- *Images* – uma pasta das imagens, gráficos e *icons* que podem ser utilizadas como imagens estáticas nos ecrãs;
- *Themes* – temas que estrutura a interface gráfica da aplicação;
- *Scripts* – recursos JavaScript scripts, único nas aplicações *mobile*;
- *Multilingual Locals* – permite gerir *locales* que em si garante a tradução da aplicação em varias línguas, único nas aplicações *Web*.

Depois temos a *Logic Layer*, responsável pelo desenvolvimento de ações que podem correr durante o processo da aplicação:

- *Client Actions* – ações que ocorrem no lado do cliente;
- *Functions* – ações que ocorrem no lado do servidor, existe unicamente no desenvolvimento *mobile*;
- *Integrations* – permite configurar integrações que a aplicação pode utilizar, permite integrações em servidores *SOAP*, *REST* e *SAP*;

- *Roles* – onde é possível incluir papéis de segurança;
- *Exceptions* – onde é incluído as excepções desenvolvidas para a aplicação.

Por ultimo temos a *Data Layer*, que permite estabelecer data importante para a aplicação, esta é composta por:

- *Entity Diagrams* – diagramas das entidades que servem como uma representação viável das entidades que compõem a base de dados;
- *Database* – inclui as entidades que guardam dados, tanto online, como offline, no caso de desenvolvimento *mobile*;
- *Structures* – é possível gerir e guardar estruturas de apoio neste componente da *Data Layer*;
- *Session Variables* – variáveis de secções estabelecidas, existe unicamente no desenvolvimento de aplicações *mobile*;
- *Site Properties* – propriedades do site;
- *Resources* – recursos disponíveis na aplicação.

**Web LifeCycle**, o ciclo de vida de uma página *Web* desenvolvida é normalmente composto por três passos, com o primeiro sendo opcional:

- *Preparação* – onde é obtido e gerido dados demonstrados na página, este passo é opcional sendo que se a informação da página pode ser completamente estática;
- *Composição* – a estrutura da página e os seus elementos *HTML* são geridos, também são obtidos recursos como *CSS*, *JS* e *IMG*;
- *Interação* – a partir deste ponto o utilizador pode interagir com a página, incluindo invocar ações *Screen* e/ou *Server*.

**Mobile LifeCycle**, apesar do ciclo de vida dum ecrã *mobile* poder ser visto como também dividido nos três passos do ciclo de vida anterior, este ciclo é algo mais complexo. Sendo que não é incluída a ação *Preparation*, em vez disso, ecrãs incluem *fetches* e *event handlers*.

O primeiro permite obter os dados a demonstrar no ecrã, sendo possível obter os dados da base de dados, do armazenamento local ou até de fontes terceiras, estes *fetches* são realizados de forma assíncrona e paralela a outros *fetches* do mesmo ecrã.

Em termos de *event handlers* cada ecrã tem a possibilidade de incluir ações associadas a quatro eventos, pelos quatro *handlers*:

- *OnInitialize* - ação ocorre depois de se verificar a possibilidade de acesso pelo utilizador, mas antes de qualquer *fetch*, permite inicializar o ecrã, definindo valores *default*;
- *OnReady* – ação ocorre quando ecrã está pronto, mas antes do *render* de o mesmo começar, manipula o *DOM* do ecrã;

- *OnRender* – ação para correr quando o ecrã termina o seu *render*, ou cada vez que um elemento do ecrã muda;
- *OnDestroy* – ação ocorre antes de um ecrã ser destruído.

**Sincronismo**, desenvolvimento sobre a plataforma *mobile* regularmente envolve a utilização de duas fontes de informação, a base de dados, *online*, e o armazenamento local, *offline*. Mas estas fontes devem sempre, ou pelo menos sempre que possível, sincronizadas para garantir que informação da segunda fonte é a mais atualizada possível.

Por esta razão a OutSystems fornece uma *framework* de sincronismo, alguns dos exemplos de sincronismo possíveis são:

- *Read-Only Data* – utilizável para quando utilizadores precisam unicamente de ler informação e quando a dimensão da data a sincronizar é baixa;
- *Read-Only Data Optimized* – semelhante ao *Read-Only Data*, mas para dimensões de data mais extensiva;
- *Read/Write Data Last Write Wins* – este padrão é útil quando existe alteração de data, modo *offline*, mas em que não existe conflito de escrita entre vários utilizadores;
- *Read/Write Data with Conflict Detection* – uma versão mais aprofundada do padrão anterior, permitindo a resolução de conflitos de escrita de múltiplos utilizadores;
- *Read/Write Data One-to-Many* – continuação dos dois padrões anteriores, este padrão permite que vários utilizadores alterem a mesma informação.

A aplicação *mobile* desenvolvida neste projeto utilizará os padrões *Read-Only Data Optimized* e *Read/Write Data Last Write Wins*.

O primeiro para ecrãs onde será verificado unicamente informação que não se pode alterar, exemplo entrevista e aplicações a vagas, que podem incluir uma extensa quantidade de informação. Já o segundo padrão será utilizado em qualquer alteração realizada na aplicação, sendo que as informações a alterar, currículo e possivelmente dossiê de capacidades, são únicos acessíveis a um candidato e por isso não existirá concorrência.

## 3 Solução Proposta

Neste capítulo é demonstrada a arquitetura do projeto em 3.1, de seguida é descrita a base de dados e os seus modelos relacionais no 3.2 e por fim são mostrados vários exemplos de páginas da aplicação no subcapítulo 3.2.

O foco deste capítulo será a proposta do projeto gerado, referindo a estrutura da aplicação, que segue a arquitetura 4 *Layers Canvas* estabelecida pela OutSystems, a base de dados da aplicação, os *wireframes* que demonstram alguns exemplos de páginas que compõem a aplicação *web* e o por ultimo alguma lógica para cumprir certas funcionalidades importantes de notar.

### 3.1 Arquitetura do projeto

Como qualquer aplicação desenvolvida na arquitetura OutSystems, IView é estruturada seguindo o padrão de 4 *Layer Canvas*, esta promove a abstração correta de serviços reutilizáveis, o isolamento correto de módulos funcionais distintos e a partilha dos mesmos por varias aplicações desenvolvidas em conjunto.

A Figura 9 demonstra uma breve representação e descrição desta estrutura, descrevendo cada uma das camadas.

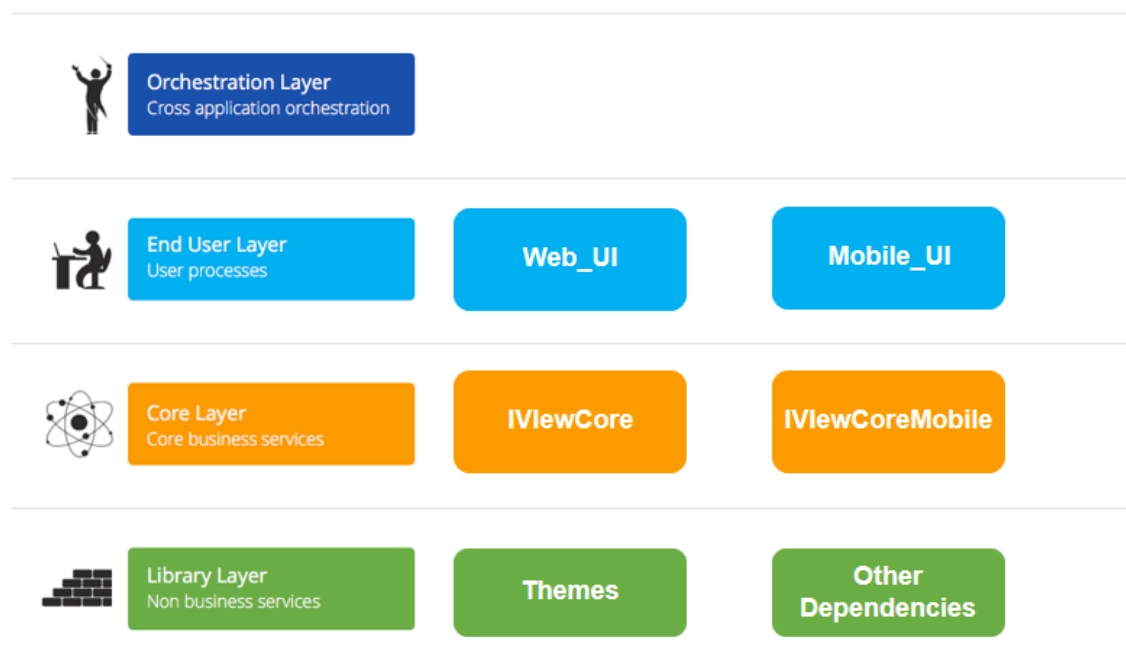


Figura 9 - Estrutura 4 Layer Canvas.

Em termos da arquitetura da IView as quatro *layers* são:

- Orchestration Layer – este projeto não inclui qualquer módulo ou elemento que pertença a esta layer;
- End User Layer – sendo que o projeto é composto por duas aplicações, *web* e *mobile*, o projeto utiliza a *Web\_UI* e *App\_UI*;

- *Core Layer* – esta *layer* será composta por dois módulos:
  - *IViewCore* – onde é desenvolvido a base de dados e algumas estruturas partilhadas;
  - *IViewMobileCore* – base de dados local.
- *Library Layer* – para este projeto utilizamos como dependências:
  - *Liverpool Template*;
  - *Silk UI*;
  - *FileSystem*;
  - *Html2PdfConverter*;
  - *FullCalendar2*;
  - *ardoJSON*;
  - *Common Plugin*;
  - *File Plugin*;
  - *File Viewer Plugin*;
  - *Google Maps Mobile*;
  - *OneSignal Plugin*.

### 3.2 Modelo entidade-associação da base de dados

Para facilitar a demonstração do modelo criado para o projeto, o modelo foi dividido em três partes, como esta secção:

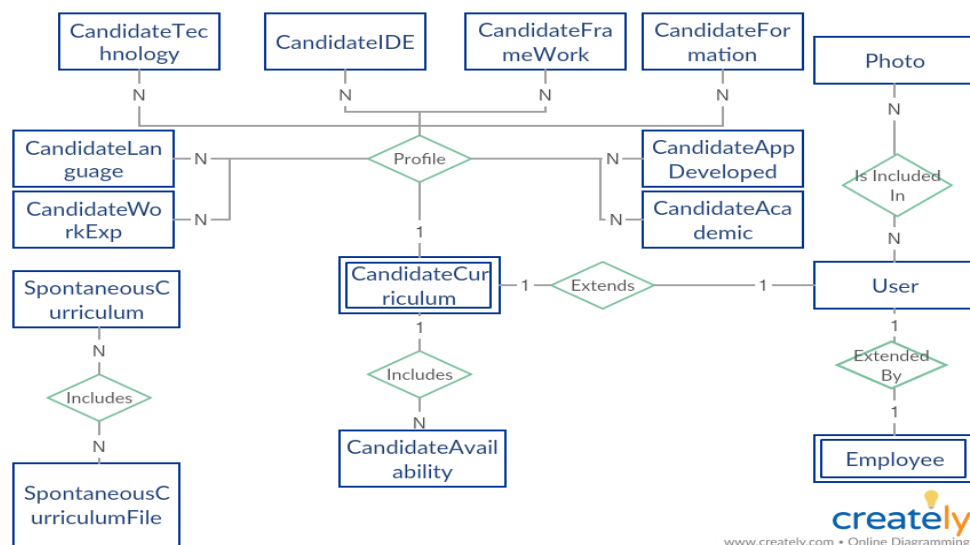
- *Utilizadores* - onde são verificadas as entidades que incluem informação dos Candidatos e Empregados, os utilizadores;
- *Vagas* - inclui as entidades sobre as vagas, projetos e aplicações a vagas;

*Eventos* - contém todas as entidades relacionadas com eventos e entrevistas. Nesta secção só ira ser verificado numa forma muito geral as entidades que compõem a base de dados, para verificar as mesmas numa forma mais abrangente temos no anexo a secção Modelos de Dado Em cada subsecção também será incluído um conjunto de notas sobre o desenvolvimento da base de dados.

#### 3.2.1 Utilizadores

Para esta componente do modelo estão incluídas todas as tabelas que estendem a tabela *User*, tabela incluída por padrão pela *OutSystems*, juntando informação importante, ou sobre o cargo do Candidato, ou sobre o currículo e capacidades do Candidato.

A Figura 10 demonstra o modelo entidade-associação desta componente do modelo.



**Figura 10 - Modelo EA, Candidatos.**

As entidades da Figura 10 são:

- User, uma entidade padrão da OutSystems, representa os utilizadores registrados na aplicação;
- Photo, todos os utilizadores podem incluir uma fotografia, que é representada por esta entidade;
- SpontaneousCurriculum, informações sobre candidatura espontânea;
- SpontaneousCurriculumFile, o currículo em si da candidatura espontânea;
- Employee, as instancias desta entidade adicionam informação ao User, tal informação sendo relacionada com a posição e funções do utilizador na PS Tec;
- CandidateCurriculum, uma das entidades que adiciona informação a utilizadores candidatos, representando o currículo do mesmo;
- CandidateAvailability, instancias desta entidade representam a disponibilidade que o candidato tem para entrevistas, associando a um dia de semana uma janela de tempo;
- CandidateTechnology, uma entidade que faz parte do dossiê de capacidades de candidatos, representa a capacidade que um candidato tem com uma linguagem de tecnologia, reconhecida pela aplicação;
- CandidateFramework, uma entidade que faz parte do dossiê de capacidades de candidatos, representa a capacidade que um candidato tem com uma *framework*, reconhecida pela aplicação;
- CandidateIDE, uma entidade que faz parte do dossiê de capacidades de candidatos, representa a capacidade que um candidato tem com uma IDE, reconhecida pela aplicação;



- *CandidateFormation*, uma entidade que faz parte do dossiê de capacidades de candidatos, representa uma formação oficial que o candidato tenha obtido, considerando as linguagens tecnológicas, frameworks e IDEs que a aplicação considera importantes;
- *CandidateLanguage*, uma entidade que faz parte do dossiê de capacidades de candidatos, representa a capacidade que um candidato tem com uma linguagem, reconhecida pela aplicação;
- *CandidateWorkExperience*, uma entidade que faz parte do dossiê de capacidades de candidatos, representa experiencia laboral que o candidato considere importante de notar;
- *CandidateAcademic*, uma entidade que faz parte do dossiê de capacidades de candidatos, representa curso académicos que o candidato considere importante de notar;

*CandidateAppDeveloped*, uma entidade que faz parte do dossiê de capacidades de candidatos, representa aplicações desenvolvidas pelo candidato que o candidato considere importante de notar. Para garantir a limitação de acesso a utilizadores estabelecidos, existente devido a divisão entre Candidatos e Colaboradores, é utilizado a capacidade de OutSystems *Roles*, que permite estabelecer papeis que pode ser utilizado para garanti tais limitações. Os Roles criados e utilizados são *CandidateIView* e *EmpolyeeIView*. Assim qualquer pagina que necessita de registo só permite acesso a um utilizador com um dos dois *Roles*.

Originalmente entidades como *CandidateLanguage*, entidades que representam informações de currículo do Candidato estavam associadas ao User em vez do *CandidateCurriculum*. Mas esta relação levantava dois problemas simples, primeiro não limitava estas informações a Candidatos, e a transmissão de Candidato para Colaborador tornava-se mais complexa.

Na relação original quando um Candidato era tornado num Colaborador era necessário obter todas as informações para as depois remover, sendo que o User não devia ser removido, sendo necessário para estabelecer o Colaborador. Na relação atual estas informações podem ser facilmente terminadas, simplesmente terminando o *CandidateCurriculum* do Candidato a transmitir.

Para representar as fotografias foi originalmente considerado incluir as fotografias nas entidades *CandidateCurriculum* e *Empolyee*, diferenciado as fotografias dos candidatos e dos colaboradores. E apesar deste modo não introduzir muitos incómodos, garante que em qualquer situação em que a fotografia a demonstra possa ser tanto do candidato como do colaborador, seja necessário o cuidado extra de obter uma entidade diferente. Ao ter uma só entidade para as fotografias este pequeno incomodo não ocorre.

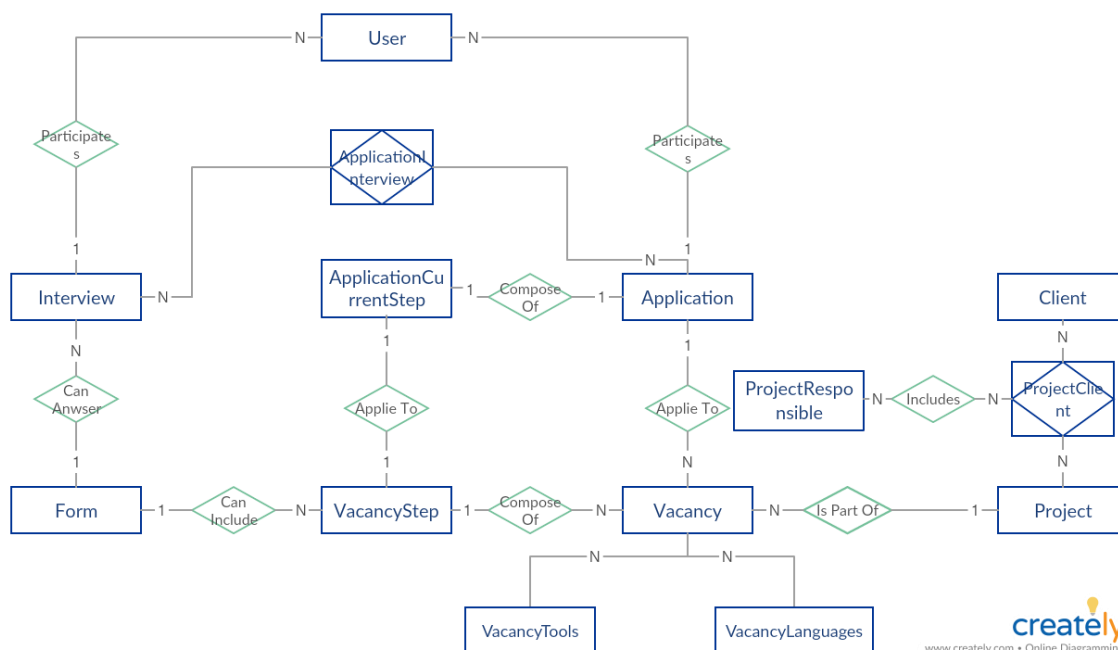
A razão pela divisão de SpontaneousCurriculum e SpontaneousCurriculumFile deve-se ao facto que o acesso principal a estas entidades envolver obter varais candidaturas espontâneas para decidir que candidaturas resultam no estabelecimento de um novo candidato e que candidaturas não resultam.

Tendo em conta as possíveis dimensões das candidaturas fornecidas, tal acesso poderá envolver um peso significativo a aplicação, peso desnecessário quando se considera que o ficheiro em si só é necessário para realizar o download do mesmo, que pode ser obtido só com o requisito do mesmo.

### 3.1.1 Vagas

Nesta componente da base de dados estão incluídas todas as entidades que permitem estabelecer vagas para candidatos se aplicarem, como também essas mesmas aplicações de candidatos.

A Figura 11 demonstra estas entidades.



**Figura 11 - Modelo EA, Vagas.**

As entidades incluídas na Figura 11 são:

- Client, nesta entidade é possível incluir informações dos clientes da PS Tec, que podem iniciar projetos com a PS Tec;
- Project, projetos a serem realizados pela PS Tec, por estes projetos é possível estabelecer vagas para candidatos;
- ProjectClient, com esta entidade é possível associar vários clientes a um projeto, e claro um cliente a vários projetos;
- ProjectResponsible, cada cliente a participar no projeto pode incluir um ou mais responsáveis a que informações dos candidatos escolhidos para vagas serão enviados,

para que os mesmos possam decidir quem deve continuar no processo de entrevista, as instancias desta entidade representam tal responsável;

- Vacancy, as vagas a preencher em si, podem ou não ser associadas a projetos;
- VacancyTool, com esta entidade é possível associar varias ferramentas reconhecidas pela aplicação a uma vaga, ferramentas cujo conhecimento pode ser visto como essencial para preencher a vaga;
- VacancyLanguage, cada instancia desta entidade demonstra uma linguagem, reconhecida pela aplicação, cujo o conhecimento é visto como indispensável para preencher uma vaga;
- VacancyStep, cada vaga deve incluir um conjunto de passos do processo de entrevista, tais passos são representados por instancias desta entidade;
- Form, sendo que um passo no processo de entrevista pode ser uma entrevista em si, o mesmo pode, não obrigatoriamente referir um Form, que em si é um formulário para uma entrevista;
- Application, quando uma vaga é criada qualquer candidato da aplicação pode-se candidatar à mesma, tal candidatura a vaga é representada por uma instancia desta entidade;
- ApplicationCurrentStep, esta entidade representa o passo atual duma candidatura;
- ApplicationInterview por instancias desta entidade uma entrevista pode ser associada a uma candidatura, permitindo assim que uma entrevista possa ser utilizada para varias candidaturas;

Interview, uma entrevista que tenha ocorrido/ ainda para ocorrer com um utilizador, que refere. Possivelmente devido a um passo duma candidatura que também refere. Também refere um Form, o formulário utilizado da entrevista Um fator importante a notar das entidades ApplicationCurrentStep e ApplicationInterview é que originalmente existiam como uma só entidade, chamada ApplicationStep. A ideia desta entidade antiga seria que representaria um passo duma candidatura, tanto os finalizados como também o passo presente.

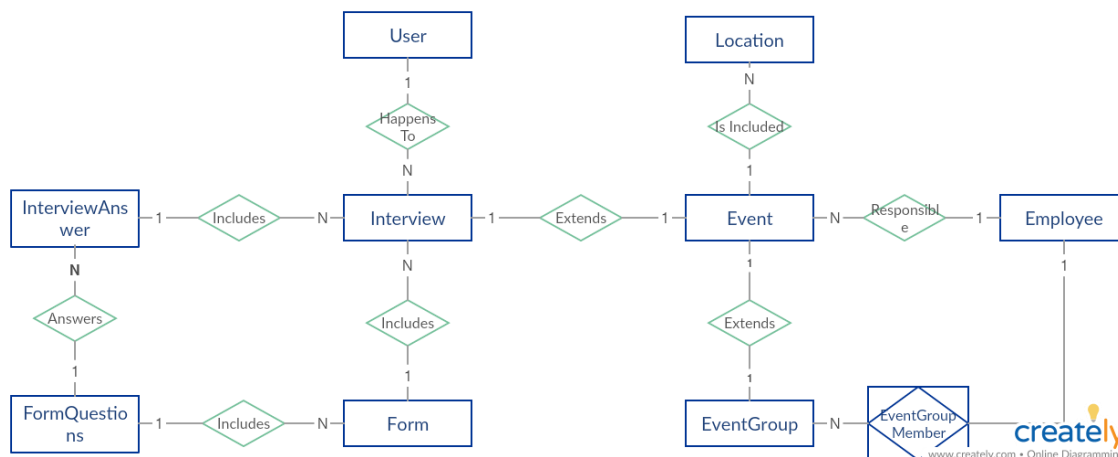
O problema deste sistema é que adicionaria peso desnecessário na base de dados, sendo que a única informação importante de passos anteriores é em si as entrevistas, que são agora representadas pelas ApplicationInterview. Este problema de peso é ainda mais notável quando uma entrevista que já ocorreu é adicionada a uma candidatura, assim as outras informações da ApplicationStep eram completamente desnecessários e ocupavam espaço.

Com a entrevista guardada na ApplicationInterview, as outras informações de ApplicationStep tornaram-se desnecessárias e por isso podiam ser facilmente transmitidas para uma só entidade que é atualizada com cada passo.

### 3.1.2 Eventos

Neste componente do modelo EA são demonstradas as entidades relacionadas com eventos como também uma continuação das entidades Interview e Form.

Estas entidades podem ser verificadas na Figura 12 que se segue.



**Figura 12 - Modelo EA, Eventos.**

Na Figura 12 pode ser verificado as seguintes entidades:

- Event, esta entidade representa os vários eventos que podem ocorrer que envolvam pelo menos um colaborador, que refere;
- EventGroup, grupo de colaboradores para um evento em grupo,
- EventGroupMember, membro individual de um grupo, para um evento,
- Form, formulário para uma entrevista;
- Interview, uma entrevista que tenha ocorrido/ ainda para ocorrer com um utilizador, que refere. Deve referir o formulário utilizado;
- FormQuestion, cada instancia desta entidade corresponde a uma questão de um formulário da Form;
- InterviewAnswer, na mesma forma que um formulário (Form) é composto por várias questões (FormQuestions), uma entrevista (Interview) é composta pelas várias respostas às questões, sendo estas representadas por instancias desta entidade;
- Location, uma localização reconhecida pela aplicação, normalmente representa localidades onde um Event poder ocorrer.

Uma nota, durante o desenvolvimento desta entidade considerou-se não incluir a referencia ao Form, sendo que Interview normalmente estará associada a um VacancyStep, que já refere o Form. Mas sendo que nos queremos manter uma entrevista, mesmo depois de uma vaga, e por isso os seus passos, terminar, o formulário deve ser referido pela entrevista para a mesma ainda fazer sentido.

### 3.2 Wireframes do projeto

Para esta secção serão demonstradas algumas *frames* para páginas importantes, em que algumas são únicas na aplicação e outras servem de exemplo que demonstram o formato habitual de várias páginas, também são utilizados estes exemplos para demonstrar alguns dos elementos mais comuns da aplicação, tanto *widgets* (*rich* e normais) estabelecidos pela OutSystems como também elementos mais estruturados especificamente para a aplicação.

Os elementos mais comuns da aplicação são o *header* e o menu de utilizador, que aparecem sempre no topo das páginas, como se pode ver na **Erro! A origem da referência não foi encontrada.**, acompanhada pela Figura 14 e Figura 15 onde é demonstrado os dois possíveis menus de utilizador.



Figura 13 - Frames, Home Page

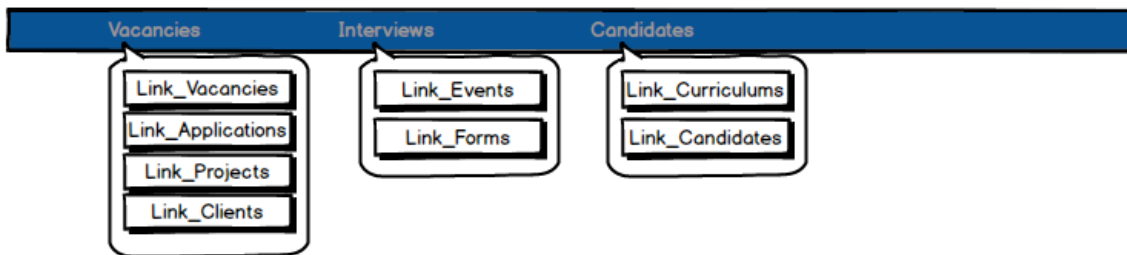


Figura 14 - Frames, Menu de Colaboradores.

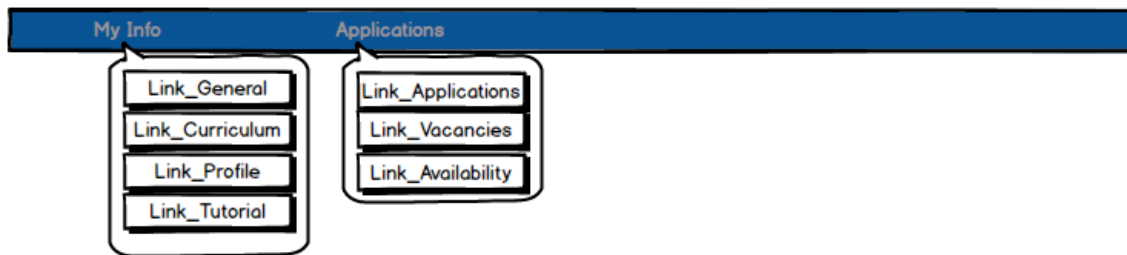


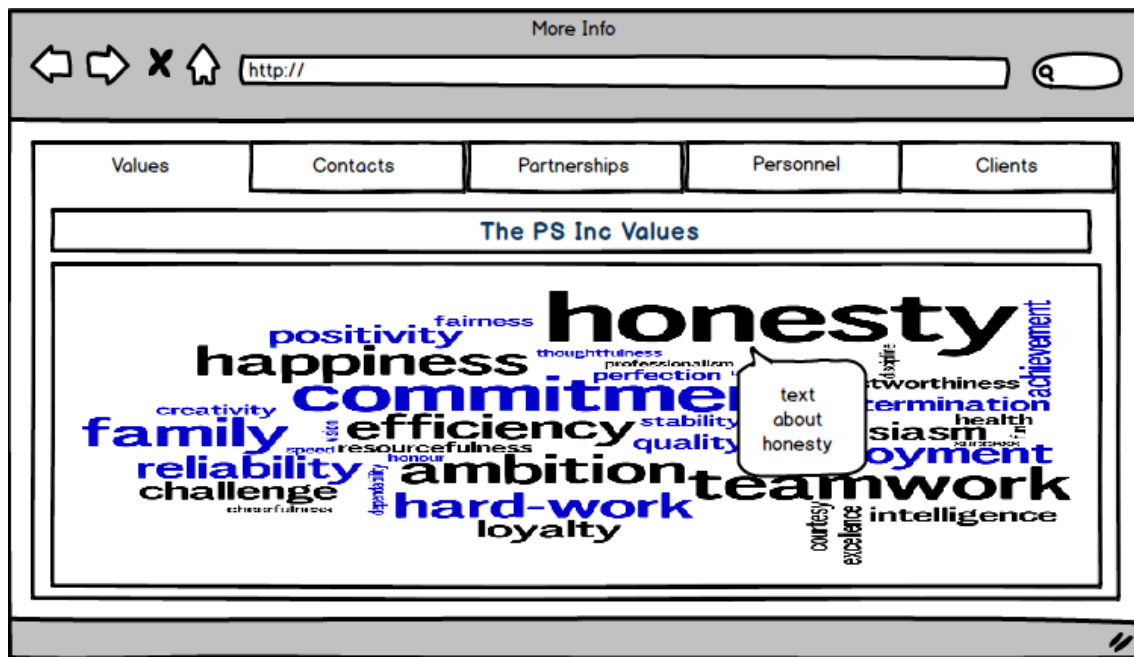
Figura 15 - Frames, Menu de Candidatos.

Começando com o *header*, este inclui sempre um *link* para a *Home Page* (**Erro! A origem da referência não foi encontrada.**) à esquerda, mais à direita aparece ou um *link* para a página de *login* ou a possibilidade de abrir um menu onde é possível aceder a página de informação geral do utilizador, ou a página de *logout*. A escolha das duas possibilidades é feita em função do utilizador ser não registrado (primeira possibilidade) ou estar registrado (segunda possibilidade).

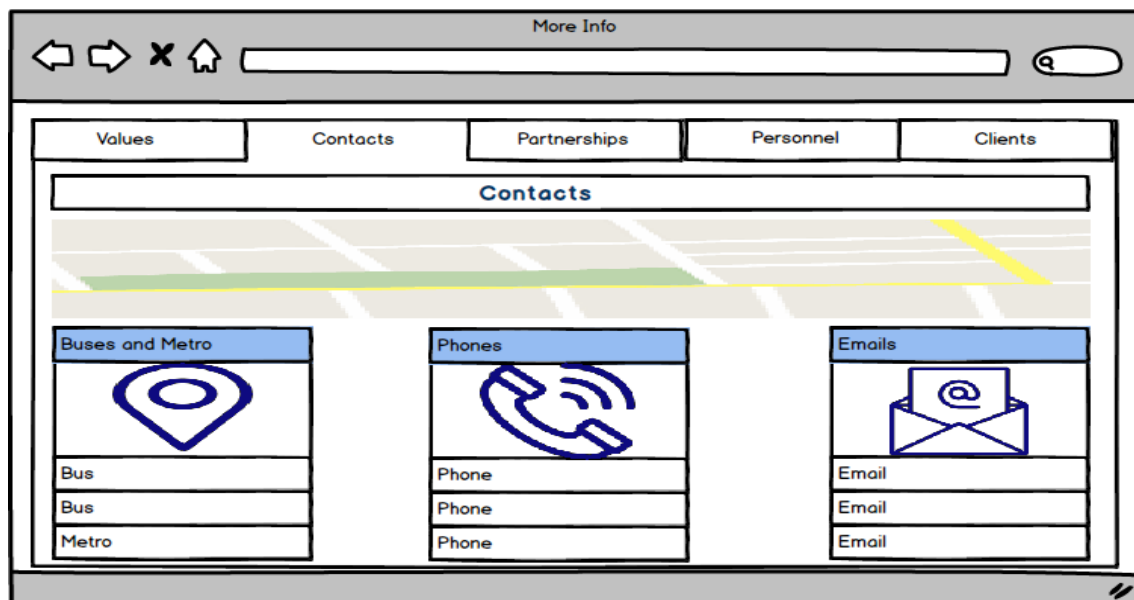
Como se pode verificar, pela Figura 14 e a Figura 15, existem dois menus diferentes, o primeiro demonstrado é o menu de colaboradores e o segundo o de candidatos. Deve ser notado que os menus só aparecem para utilizadores já registrados. Cada menu é composto por um conjunto de *ButtonDropDowns* que abram e fecham um *div* associado, onde é incluindo um conjunto de links para paginas consideradas importantes.

Originalmente foi considerado utilizar um *Ballon* em vez do *ButtonDropDowns*, mas entre algumas limitações visuais como também uma má interação entre o *Ballon* e vários *plugins* da *Forge*, como por exemplo *FullCalendar2* que é utilizado, o *ButtonDropDown* tornou-se uma opção mais viável.

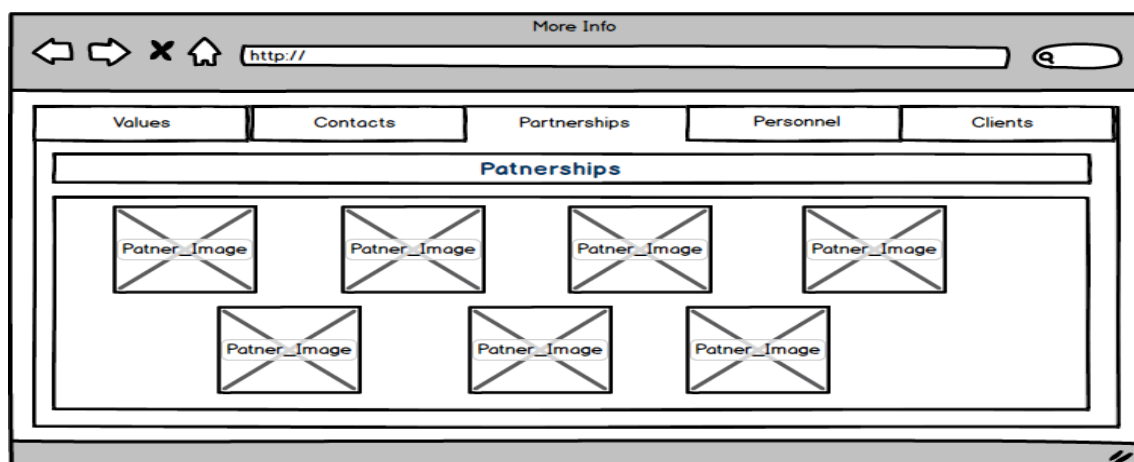
Na Figura 16, Figura 17, Figura 18, Figura 19 e Figura 20 é possível verificar a página de informação geral da PS Tec que introduz dois elementos visuais fundamentais à aplicação, mais outro menos utilizável, note-se que as cinco figuras representam uma só pagina, mas cada demonstra uma *Tab* diferente da pagina.



**Figura 16 - Frame, MoreInfo Values Tab.**



**Figura 17 - Frame, MoreInfo Contacts Tab.**



**Figura 18 - Frame, MoreInfo Patnerships Tab.**

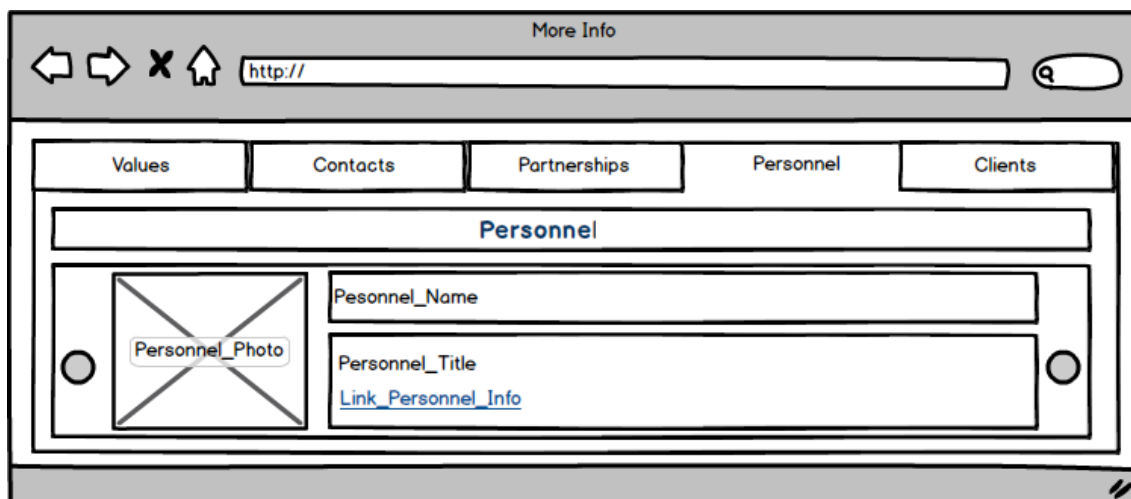


Figura 19 - Frame, MoreInfo Personnel Tab.

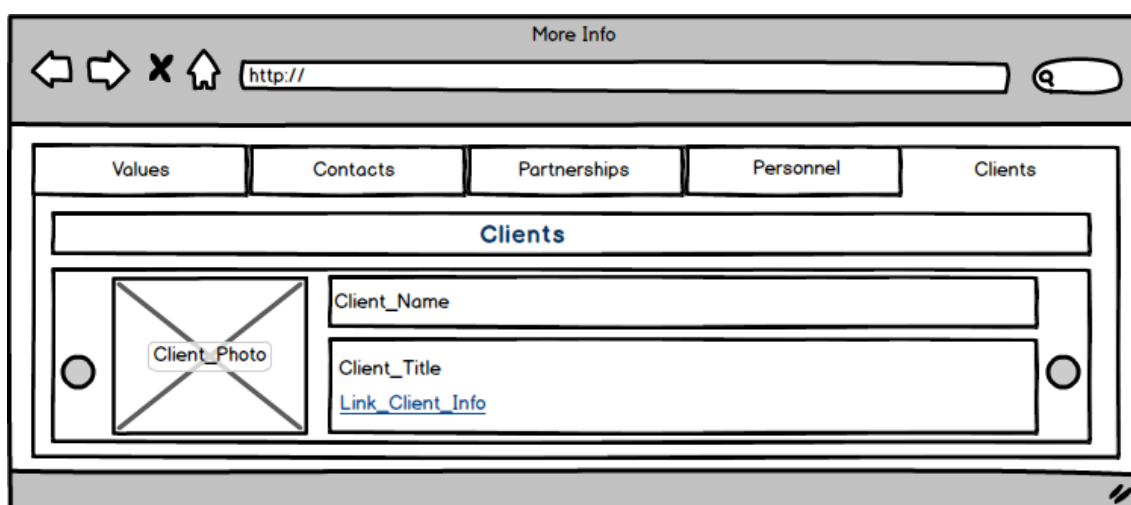


Figura 20 - Frame, MoreInfo Client Tab.

Esta página demonstra:

- Valores e ambições da PS Tec;
- Contactos;
- Parcerias importantes;
- Colaboradores registrados na aplicação;
- Clientes já registrados.

Como referido anteriormente existem dois elementos importantes na aplicação que podem ser verificados nesta página, o primeiro desses é o *rich widget* Tabs, que pode ser verificado no topo da página. Este elemento permite dividir uma página até cinco partes, com uma sempre visualizada e as outras quatro escondidas, assim quando uma página inclui muita informação, em vez de ser extensa verticalmente, as informações estão divididas em Tabs.

O outro elemento utilizado nesta página a notar é o *widget* List Records, que permite demonstrar vários elementos obtidos na base de dados, numa estrutura bastante livre, não forçando um formato tipo tabela, a List Records é normalmente utilizada em situações em que se



quer demonstrar informação que possa ocupar mais que uma linha ou inclua pelo menos uma imagem.

No caso da Figura 19 e da Figura 20, esta *widget* é utilizada com a *rich widget* Carroussel para demonstrar os clientes e colaboradores num formato simples. Cada um dos colaboradores e clientes demonstrados são representados por uma instancia do *rich widget* CardLeftImage, que permite demonstra uma figura a esquerda e informação geral a direita, este *rich widget* é utilizado varias vezes, principalmente em listas de Users e clientes.

Outra característica notável desta pagina é que varias palavras da Tab demonstrada na Figura 16 abre um Ballon com algum texto sobre o valor em si.

Na secção **Modelo entidade-associação da base de dados** foi estabelecido a existência de um currículo que um candidato pode verificar e alterar, estas duas capacidades são possíveis na página representada pela Figura 21, verificar, e pela Figura 22, editar.

Complete Name	Candidate Name
Date Of Birth	Candidate Date Of Birth
Native Tongue	Candidate Native Tongue
Locality	Candidate Locality
Accessibility	Candidate Accessibility

Figura 21 - My Curriculum, Show.

Complete Name	Name_Input
Date Of Birth	Date_Of_Birth_Chosser
Native Tongue	Native_Tongue_ComboBox
Locality	Locality_Input
Accessibility	Accessibility_Input

Cancel Save

Figura 22 - Frame, MyCurriculum, Edit.

Esta página serve como exemplo da utilização de alguns *inputs* da aplicação como também de dois *widgets* muitas vezes utilizados para demonstrar e alterar informação, tais *widgets* são

Show Record e Edit Record, ambos partilham o formato tabela e por isso são bastante utilizados em conjunto e às vezes separados. O formato apesar de simples combina visualmente com qualquer página e por isso tem uma utilidade regular quando se quer demonstrar ou introduzir informação extensa.

Nesta página são utilizados dois *inputs* que irão aparecer algumas vezes, primeiro é o Combo Box, verificado a frente de “Native Tongue”, da Figura 22, este *input* permite associar a uma variável um valor duma lista estabelecida, assim limitando o valor escolhido a valores na lista. No exemplo da Figura 22, a língua nativa do currículo é limitada a línguas reconhecidas pela aplicação, instancias da entidade Languages. O outro *input* é na verdade a combinação de um *input* simples, mais o *rich widget* Input Calendar, que faz aparecer um calendário onde se pode escolher uma data cada vez que o *input* normal é pressionado, com a data escolhida sendo demonstrada nesse mesmo *input*.

Como também foi referido na secção **Modelo entidade-associação da base de dados**, um candidato consegue controlar o seu dossiê de capacidades que é composto de várias entidades associadas ao colaborador. A Figura 23 demonstra esta página que introduz alguns elementos notáveis e regularmente utilizados na aplicação.

The screenshot displays the 'My Profile' web application interface. At the top, there is a navigation bar with links: Technologies, Frameworks, IDEs, Languages, Academics, Formations, Work Experience, and Applications Developed. The main content area is divided into sections for different entities. The 'Technologies' section is currently active, showing a form with the following fields: 'Technologie' (a dropdown menu), 'Practical Experience' (a numeric input field with a value of 3), 'Academic Experience' (a numeric input field with a value of 3), and 'Capacity' (a dropdown menu). Below the form are buttons for 'Save', 'Cancel', and 'Delete', and an 'Add Technology' link. Similar forms are visible for 'Frameworks' and 'IDEs'. The 'Academics' section is partially visible at the bottom, showing fields for 'Course', 'University', 'Complete' (checkbox), 'Average' (input), 'Start Date' (calendar), and 'End Date' (calendar).

☐ Save
☐ Cancel
☐ Delete

☐ Add Academic

### Formations

Tool	Provider	Date
Tool_Chosse	Provider_Input	//

☐ Save
☐ Cancel
☐ Delete

☐ Add Formation

### Work Experience

Position	Company	Start	End
Position_Input	Company_Input	//	//

☐ Save
☐ Cancel
☐ Delete

☐ Add Work Experience

### Applications Developed

Name	Name_Input
Start Date	//
End Date	//
Responsibilities	Responsibilities_Input
Technologies	Technologies_Input
Accomplishments	Accomplishments_Input
Summarization	Summarization_Input
	<input type="button" value="Delete"/> <input type="button" value="Edit"/>
	<input type="button" value="Add"/>

#### Add Application Developed

Name	Name_Input
Start Date	//
End Date	//
Responsibilities	Responsibilities_Input
Technologies	Technologies_Input
Accomplishments	Accomplishments_Input
Summarization	Summarization_Input
	<input type="button" value="Cancel"/> <input type="button" value="Add"/>

Figura 23 - Frames, MyProfile.

O primeiro elemento a notar é a Editable Table, uma tabela que permite adicionar elementos à mesma, como também alterar elementos já existentes, estas tabelas incluem sempre um botão que adiciona um vazio a tabela, que pode ser alterado. Quando um elemento desta tabela é selecionado é possível alterar o mesmo, sendo disponível três botões, uma cancela qualquer

29

alteração, outro remove a instancia selecionada e último guarda as alterações à mesma. No caso desta página, o remover e guardar elementos não só altera a tabela como também altera diretamente a base de dados.

A única entidade que compõe o dossiê que não é alterado por uma Editable Table é a entidade CandidateAppDeveloped, sendo que mesma envolve informação extensa de mais para uma só linha de texto, por isso é utilizado uma ListRecord, com cada elemento utilizando um EditRecord para alterar a aplicação. Sendo que ListRecord não permite a utilização direta de EditRecord, o mesmo é na verdade parte de um WebBlock. Esta combinação de ListRecord e WebBlock é utilizada varias vezes para listar elementos com informação extensa.

Já para adicionar uma nova instancia a CandidateAppDeveloped é utilizado o *PopUp* Editor<sup>3</sup>. No caso da MyProfile, quando o botão “Add” é pressionado um *PopUp* abre que permite introduzir as informações do projeto desenvolvido, normalmente *PopUps* não são utilizados para introduzir novos valores a uma entidade, mas sim para facilitar a escolha de instancias duma tabela.

Este *PopUp* pode ser verificado na Figura 24.

**Figura 24 - Frames, MyProfile *PopUp*.**

A estrutura desta pagina passou por algumas alterações, a utilização das tabelas e lista foi sempre comum, mas pequenas alterações a volta destes elementos ocorreram durante o desenvolvimento.

Originalmente a pagina era composta simplesmente por um conjunto de Editables, mas esta estrutura tornava a pagina muito grande. Para tornar a pagina mais curta considerou-se incluir a paginação das tabelas, mas sendo que a List\_Navigation não funciona corretamente com uma Eidtable, seria necessário utilizar o sistema original dos botões “Prev” e “Next”. Mas tal envolveria um conjunto extenso de código, envolvendo um *array* para as varavies *curr*, *needsNext*

<sup>3</sup> Rich widget que permite criar uma página tipo *popup* a partir de uma página já estabelecida na aplicação.

e *needPrev*. Já a Screen Action associada aos botões necessitaria de um *Switch* para escolher que tabela é paginada.

Percebendo que paginação não seria uma hipótese, procurou-se outra forma de navegação, daí ser introduzido o *rich widget* *SectionIndex* que permite centrar a pagina na secção escolhida, permitindo uma navegação em função das secções, onde fui incluído as tabelas que compõem a pagina. Para facilita ainda mais a visualização a *Section* utilizada para manter a tabela é do tipo *Expandable*, o que garante que a tabela só é demonstrada quando for selecionada, com a possibilidade de desseleccionar a tabela.

Uma funcionalidade regular das páginas da aplicação *web* da *IView* é de listar um conjunto de instancias estabelecidas duma entidade da base de dados. Um exemplo simples destas páginas é o demonstrado na Figura 25, que mostra todas as instancias da entidade *Form*.

The screenshot shows a web browser window with the title 'Forms'. The address bar contains 'http://'. The main content area has a header 'Forms' and an 'Add Form' button. Below the header is a search bar with a text input 'Search\_Form\_Title\_Input', a 'Search' button, and a 'Reset' button. Underneath the search bar is a table with three columns: 'Title', 'Number Of Questions', and 'Created'. The table contains one row with the following data: 'Form\_Title\_Link', 'Form\_Number\_Of\_Questions', and 'Form\_Date\_Of\_Creation'. At the bottom of the table is a pagination bar with buttons for 'Previous', '1', '2', '3', an ellipsis, and 'Next'.

Title	Number Of Questions	Created
Form_Title_Link	Form_Number_Of_Questions	Form_Date_Of_Creation

Figura 25 - Frames, Forms.

Esta página apesar de simples mostra os componentes mais comuns das páginas que demonstram as instancia duma entidade, a primeira é a utilização da *Table Records* para demonstrar as instancias em si. Esta *widget* é semelhante a *List Records*, ambas demonstram instancias duma lista, mas ao contrario da *List* a *Tabel* demonstra a informação no formato de tabela. Esta *widget* serve principalmente para demonstrar instancias com informação pouco extensa e por isso que possam ser demonstradas numa só linha de texto. Normalmente os elementos desta tabela incluem um *link* que permite aceder a página onde a instancia pode ser demonstrada numa forma mais extensa, onde também é possível alterar ou destruir a instancia.

Esta tabela é formada para demonstrar um conjunto limitado de instancias da base de dados, por causa desta limitação é necessário utilizar outro elemento habitual deste tipo de páginas a lista de botões de navegação, esta navegação é realizada com o *rich widget* *List\_Navigation*, que produz a lista de botões de navegação como necessário tendo a conta a tabela, sem de precisar de código extra da parte da aplicação.

Outro elemento bastante comum nas páginas de instancias é um formulário de pesquisa, que normalmente é composto por um *input* e dois botões, no *input* o utilizador pode introduzir um valor de pesquisa, normalmente um valor texto da entidade demonstrada na página, sobre os

botões um permite pesquisa reexecutando o Aggregate que obtém a informação da tabela, com o valor da *input*, outro permite restabelecer a tabela, com o valor de pesquisa *null*.

Por último temos o botão de adição, no caso da Figura 25, o botão “Add Form”, este botão não é utilizado sempre neste tipo de páginas, mas aparece em grande parte e serve sempre para navegar para uma página onde é possível adicionar um novo elemento.

Existem algumas exceções nas páginas de listagem das instancias, algumas introduzindo um formato alternativo a pesquisa, outros demonstram a informação numa forma bastante diferente e por últimos outros introduzem mais ações possíveis por cada instancia da entidade listada, um exemplo dessas exceções pode ser visto na Figura 26, onde é verificada a página de listagem de aplicações a vagas existentes.

Figura 26 - Frames, Applications.

Como se pode ver entre a Figura 25 e a Figura 26 existem três diferenças notáveis, a primeira é a falta do botão de adição, o que faz sentido sendo que uma candidatura só pode ser adicionada por um colaborador, utilizador que não tem acesso a página da Figura 26.

A tabela também é bastante diferente sendo utilizado um List Record em vez de uma Table Record, com o elemento demonstrado sendo uma combinação dos *widgets* CardLeftImage e Show Record.

Por ultimo o formulário de pesquisa inclui a escolha de pesquisa por nome do candidato ou titulo da vaga, utilizando um Check Box para escolher entre um e outro, este quando é alterado força a pesquisa com o valor na *input*.

Outra exceção do formulário de pesquisa encontra-se na página de listagem de vagas, que pode ser verificada na Figura 27.

Figura 27 - Frames, Vacancies.

Como se pode ver a pesquisa de vagas é realizado sem nenhum *input* de texto, sendo realizado unicamente por escolha de um valor de uma Combo Box, que demonstra as instancias de JobTitle e pela escolha de ferramentas e /ou linguagens, este formulário, como se pode notar não inclui qualquer botão de pesquisa, qualquer alteração ao Combo Box ou da seleção resulta numa pesquisa. Também deve-se notar que o botão “Smart Search” altera a visualização do formulário, e caso o botão torne o formulário invisível, então pesquisa é cancelada e as vagas demonstradas não são limitadas pelos valores no formulário.

Outra exceção a notar é a página de listagem de eventos que pode ser verificada na Figura 28.

Figura 28 - Frames, Events.

Uma alteração notável desta página quando comparada as outras páginas já referidas é que a paginação de eventos é realizada em função do utilizador atual. Os eventos listados na página são os eventos em que o utilizador participa, como responsável ou como participante, em eventos em grupo, também pode ser notado que a pesquisa é realizada, não com um *input* simples, mas com um Input Calendar e que os eventos listados não são só os eventos do dia da pesquisa, mas também os próximos quatro dias, não incluindo fim-de-semana.

Mas o fator mais notável de exceção é que os eventos são listados com um bloco que ocupa a janela de tempo entre o início e o fim do evento, na data correta, algo que é possível, não com a utilização de uma Table Records, mas uma List Records onde cada elemento listado resulta num bloco ocupado ou livre.

A ultima exceção a notara é verificada na Figura 29, que não envolve o formulário de pesquisa e adiciona capacidades a cada instancia da entidade. Esta figura demonstra a listagem de todas as candidaturas espontâneas ainda não tratadas.

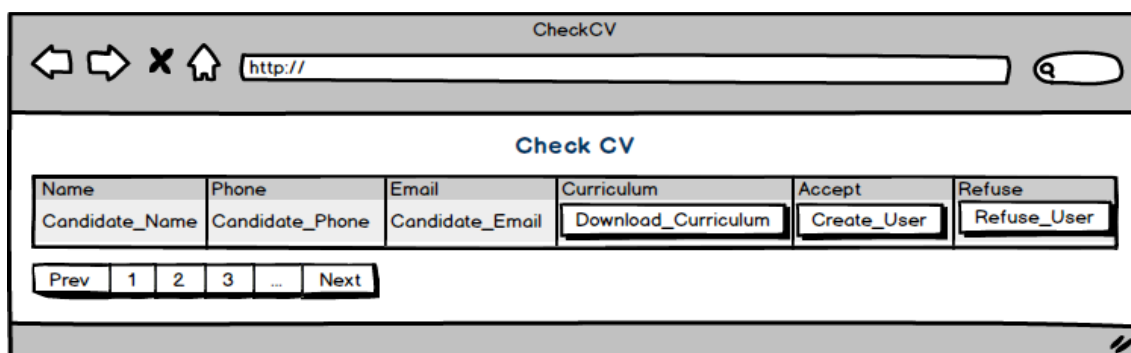


Figura 29 - Frames, CheckCV.

Como se pode ver cada instancia da tabela da Figura 29, inclui três botões com cada tendo a sua funcionalidade:

- *Download*, faz download do currículo enviado na candidatura;
- *Accept*, aceita a candidatura e é criado um utilizador, candidato, com a informação da candidatura, remove a candidatura da base de dados;
- *Refuse*, recusa a candidatura, remove a candidatura da base de dados.

Anteriormente foi estabelecido o facto que as páginas de listagem normalmente incluem um *link* nas instancia listadas, para uma página que demonstra a informação da instancia mais detalhada. Um exemplo simples deste tipo de página pode ser verificado na Figura 30, que demonstra a página de um candidato.



Candidate

http://

Candidate

Name	Candidate_Name
Email	Candidate_Email
Phone	Candidate_PhoneNumber
Curriculum	<a href="#">Candidate_Curriculum_Link</a>
Profile	<a href="#">Candidate_Profile_Link</a>
Interviews	<a href="#">Candidate_Interviews_Link</a>

Candidate\_Photo

Figura 30 - Frames, Candidate.

Esta página, como muitas outras que demonstram uma instancia, utiliza o Show Record para demonstrar a informação da instancia em si, muitas vezes as páginas deste tipo não são tão simples, sendo que algumas instancias em si estão associadas a vários elementos de outra entidade, nesses casos a página também demonstra essas instancias de outra entidade, como se pode ver na Figura 31, onde se pode verificar a página que demonstra alguma informação de um evento, mais os seus participantes.

Event

http://

Event

Date : Event\_Date Start: Event\_Start End: Event\_End

Responsible

Name	Responsible_Name
Email	Responsible_Email
Phone	Responsible_Phone

Responsible\_Photo

Participant

Name	Responsible_Name
Email	Responsible_Email
Phone	Responsible_Phone

Participant\_Photo

Group

Participant

Name	Responsible_Name
Email	Responsible_Email
Phone	Responsible_Phone

Participant\_Photo

35

Event Comment

Figura 31 - Frames, Event.

Algo a notar na página é que a mesma inclui sempre os blocos Responsible e Event Comment, podendo incluir o bloco Participant ou Group, caso for uma entrevista ou evento de grupo. No primeiro caso de um dos blocos possíveis, é simplesmente a utilização duma imagem e um Show Record para demonstrar o participante, mas no caso de grupo para demonstra um elemento individual de tal grupo, também é utilizado a combinação, mas o mesmo está numa List Record que lista todos os elementos de um grupo para um evento.

Às vezes a listagem dos elementos pode ser mais complexa que a verificada na Figura 31, sendo que em certas listagens o que é demonstrado depende de vários fatores da instancia a ser listada, um bom exemplo de uma página com esta complexidade é a página da Figura 32, que demonstra a página de uma candidatura, o que envolve a listagem dos passos dessa candidatura que já ocorreram e o passo atualmente a correr.

Application

End

Name	<a href="#">Candidate Name Link To Candidate Profile</a>
Email	Candidate_Email
Phone	Candidate_Phone
Vacancy	<a href="#">Vacany Title Link To Application Vacancy</a>
Current Step	Application_Current_Step_Number

Steps

Step Number : Step\_Number

Type : Step\_Type

If_Step_State_ToSchedule	Schedule	
If_Step_State_Scheduled	Check Event	
If_Step_State_ToProcess	If_Step_Is_Interview	If_Step_Is_Contract
	Check Info	Hire
	If_Step_Not_Interview	If_Step_Not_Contract
	Check Event	Continue
If_Step_State_Finished	If_Step_Is_Interview	
	Check Info	
	If_Step_Not_Interview	
	Check Event	

**Figura 32 - Frames, Application.**

Neste caso o valor na lista depende do estado do passo e possivelmente no tipo do passo, com cada estado permitindo executar uma ação diferente sobre o passo. Deve ser notado que os textos “If\_Step\_State\_...” e “If\_Step\_Is/Not” não aparecem na página, só servem na frame para indicar que estado do passo demonstra o conteúdo relacionado ao texto, neste caso o estado da candidatura demonstra um botão diferente que permite uma ou mais opções.

- Estado ToSchedule, como o passo está para marcar, a opção permite marcar a entrevista/evento;
- Estado Scheduled, a candidatura tem uma entrevista marcada, a opção permite demonstrar essa entrevista;
- Estado ToProcess, o processo pode ser continuado para o próximo passo, neste caso há varias opções:
  - O passo é entrevista – é possível verificar a opção;
  - O passo não é entrevista – permite demonstrar informação geral do candidato;
  - O passo é contracto – o candidato é contratado;
  - O passo não é contracto – o processo pode ser continuado para o próximo passo.
- Estado Finished, o passo já terminou, nesse caso existem duas possibilidades:
  - O passo é entrevista – é possível verificar a opção;
  - O passo não é entrevista – permite demonstrar informação geral do candidato.

Deve ser notado que a candidatura pode ser terminada, utilizando o Botão “End” no canto superior direito.

As vezes estes tipos de páginas demonstram informação bastante extensa, então para facilitar a visualização e organização da página são utilizadas Tabs para dividir blocos de informação.

Um exemplo da utilização das Tabs nestas páginas pode ser verificado na Figura 33, Figura 34 e Figura 35 onde se pode ver a página que demonstra uma vaga, que inclui a informação geral da mesma, os vários passos para a vaga e por últimos a lista das ferramentas e linguagens consideradas importantes para a vaga.

Vacancy

http://

**Vacancy**

General Information	Steps	Tools And Languages
Title	Vacancy_Title	
Label	Vacancy_Job_Title	
Project	<a href="#">Vacancy_Project_Link</a>	
Requests	Vacancy_Requests	

Figura 33 - Frames, Vacancy General Information Tab.

Vacancy

http://

**Vacancy**

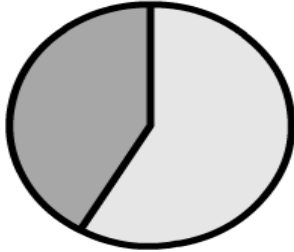
General Information	Steps	Tools And Languages								
										
<div style="display: flex; justify-content: space-around;"> <span>Step</span> <span>Step</span> </div> <table border="1"> <thead> <tr> <th>Step</th> <th>Type</th> <th>Form</th> <th>Applications</th> </tr> </thead> <tbody> <tr> <td>Step_Number</td> <td>Step_Type</td> <td><a href="#">Link_To_Step_Form</a></td> <td><a href="#">Link_To_Step_Applications</a></td> </tr> </tbody> </table>			Step	Type	Form	Applications	Step_Number	Step_Type	<a href="#">Link_To_Step_Form</a>	<a href="#">Link_To_Step_Applications</a>
Step	Type	Form	Applications							
Step_Number	Step_Type	<a href="#">Link_To_Step_Form</a>	<a href="#">Link_To_Step_Applications</a>							

Figura 34 - Frames, Vacancy Steps Tab.

Vacancy

http://

**Vacancy**

General Information	Steps	Tools And Languages	
<p><b>Tools</b></p> <table border="1"> <tr> <td>Vacancy_Tool</td> </tr> </table>			Vacancy_Tool
Vacancy_Tool			
<p><b>Languages</b></p> <table border="1"> <tr> <td>Vacancy_Language</td> </tr> </table>			Vacancy_Language
Vacancy_Language			

Figura 35 - Frames, Vacancy Tools and Languages Tab.

O único outro fator a notar da página da Figura 34 é o facto que é utilizado um Pie Chart para demonstrar a divisão por passos (em percentagem) dos candidatos participantes da vaga. Este *widget* não é utilizado em qualquer outra página, mas deve ser notado para esta.

Por ultimo deve ser verificado páginas que servem para introduzir novas instancias das entidades da base de dados, um exemplo bastante simples destas páginas é a verificada na Figura 36, que demonstra a página que permite introduzir um formulário.

Figura 36 - Frames, FormEditAdd.

A página da Figura 36, demonstra os dois *widgets* mais comuns a páginas para adicionar instancias, o primeiro sendo o Edit Record, este *widget* é normalmente utilizado quando se quer introduzir a informação que compõe a instancia a introduzir, o outro *widget* regular é o Edit Table, que é utilizado regularmente para introduzir outras instancias sobre uma entidade relacionada com a instancia central a adicionar. No exemplo da Figura 36, o Editable Table serve para introduzir as questões (FormQuestions) que compõem o formulário. Outros dois elementos regulares destas páginas são os botões “Create” e “Cancel”, com o primeiro servindo para criar a instancia formulada na página e o segundo para cancelar a criação da instancia.

Em algumas situações uma página simples não será suficiente para estruturar uma instancia a adicionar a base de dados, nesses casos uma de duas soluções são utilizadas, a utilização de Tabs ou de *PopUps*, o exemplo de uma *PopUp* utilizada neste tipo de paginas pode ser verificada na Figura 37.

Figura 37 - Frames, VacancyAdd Form Chosse *PopUp*.

The screenshot shows a web browser window titled "VacancyAdd". The address bar contains "http://". The page has a header with the title "Vacancy Add" and a "Create" button. Below the header is a tabbed interface with four tabs: "Main", "Steps", "Tools", and "Languages". The "Main" tab is active. It contains a form with the following fields: "Title" (text input), "Number of People" (number input), "Position" (dropdown menu), "Requests" (text input), and "Project" (dropdown menu). A "Chosse Project" button is located next to the "Project" dropdown.

Figura 38 - Frames, Vacacncy Add Main Tab.

The screenshot shows the "Steps" tab of the "VacancyAdd" interface. It contains a form with two main sections: "Step Type" and "Form". The "Step Type" section has a dropdown menu labeled "Step\_Type\_Chosse". The "Form" section has a dropdown menu labeled "Step\_Form\_Chosse" and a "Chosse Form" button. Below these sections are three radio buttons: "Add Step", "Save", and "Delete".

Figura 39 - Frames, Vacacncy Add Sterps Tab.

The screenshot shows the "Tools" tab of the "VacancyAdd" interface. It contains a form with a single checkbox labeled "Tool\_Chosse".

Figura 40 - Frames, Vacacncy Add Tools Tab.

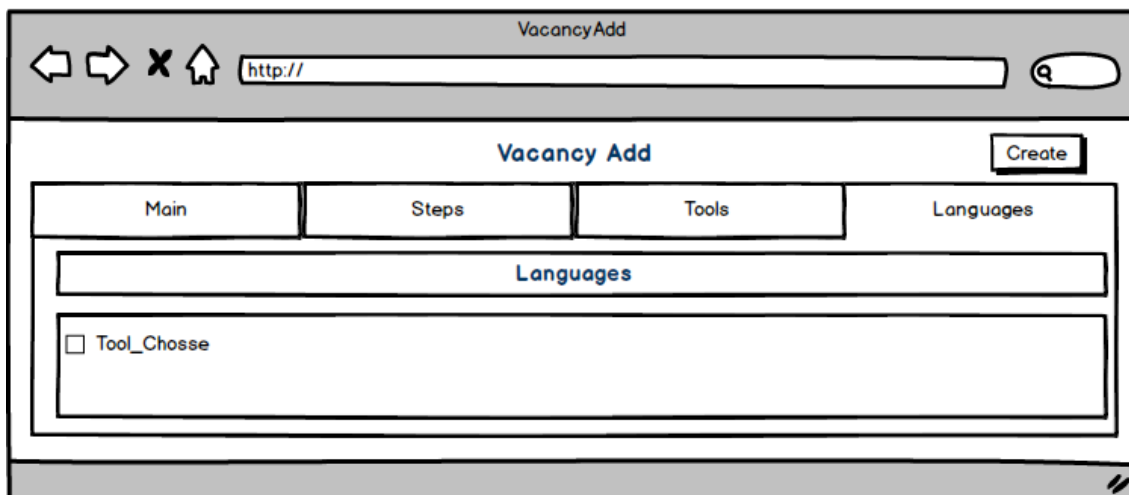


Figura 41 - Frames, Vacacncy Add Languages Tab.

No caso das figuras anteriores, o *PopUp* serve para escolher o Form para cada passo da vaga, demonstrando todos os Forms existentes, como também disponibilizando uma pesquisa dos Forms pelo nome.

### 3.3 Desenvolvimento Web

Para facilitar a demonstração das implementações mais notáveis, esta secção será dividida num conjunto de subsecções, com as primeiras sendo centradas na utilização de certos *widgets* e outros elementos comuns nas paginas, e as seguintes subsecções verificando paginas particulares, que incluem alguma logica na preparação ou ação que realiza. Para facilitar a descrição ira ser utilizado o termo *action* quando se esta a referir a uma *Screen Action* e *function* uma *Function*.

#### 3.3.1 Tabs

Algumas paginas, aquelas que demonstram um conjunto extenso de informações algo separadas, incluem este *rich widget* para dividir a pagina em blocos de informação, tornando a visualização de tal informação mais simples.

Mas este *rich widget* não foi a escolha original para realizar esta divisão de informação, originalmente foram consideradas duas possibilidades, ambas com limitações notáveis.

A primeira possibilidade foi adicionar um novo menu, abaixo do menu verificado na Figura 14 e na Figura 15. Este menu teria incluindo um conjunto de botões que quando pressionados demonstrava um bloco diferente de informação. Já a segunda escolha continuaria a primeira, transmitindo todos estes botões para um Balloon que podia ser invocado por um botão que apareceria mais a direita no topo da pagina.

Ambas possibilidades envolviam algum código extra da parte da aplicação, como por exemplo garantir que um só bloco era demonstrado envolvia utilizar uma variável cujo valor definia que bloco era visível (utilizando a característica *Visible*).

Os botões pressionados executariam uma *function* que alteraria tal variável para um valor específico, o valor que demonstra o bloco de informação associado ao botão, seguido de um *Ajax Refresh* para alterar a visibilidade dos blocos.

Por isso a utilização destas duas possibilidades envolve sempre incluir uma variável extra, como também manter um *function* só para a alteração da visibilidade.

Outra limitação destas possibilidades é que qualquer alteração, tanto introduzir um novo bloco, remover um bloco ou até alterar a ordem de blocos, envolveria sempre um pouco de mais trabalho, garantindo que a *function* altera como necessário.

Também pode ser notado que a Tabs tem uma desvantagem notável, o facto que este *rich widget* só pode incluir, no máximo, cinco blocos de informação, garantindo assim um numero limite de blocos que uma pagina pode incluir, utilizando este *rich widget*. No momento tal limitação impôs nenhum problema, por isso a decisão é bastante simples.

### 3.3.2 Paginação

Grande parte das paginas que incluem uma, ou mais, lista(s) de informação incluem o *rich widget* List\_Navigation, para realizar a paginação de tal lista. Este *widgte* envolve alguma logica sendo que para ser corretamente utilizado deve ser definido quatro características do mesmo:

- *ListWidgetId* : o identificador da lista a paginar
- *LinCount*: a *LineCount* (numero de linhas) a lista a paginar
- *TotalRowCount*: a dimensão (numero completo) da lista a paginar
- *OnNotify* : a *function* invocada quando os botões da paginação são utilizados

No caso da ação *OnNotify*, a mesma é bastante simples, sendo que unicamente inclui realizar o *Aggregate* que obteve a lista a paginar, seguido de um *Ajax Refresh* para demonstrar a lista na pagina escolhida. Um exemplo desta ação pode ser verificada na Figura 42, ação de paginação para a pagina que demonstra os candidatos.

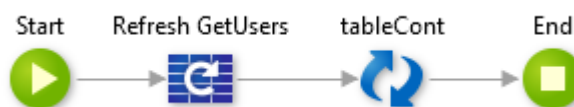


Figura 42 - Paginação

Mas este sistema não foi sempre utilizado, originalmente era utilizado um sistema bastante mais complexo que envolvia um quanto código na aplicação. Originalmente as tabelas e lista incluíam dois botões, o “*Next*” e “*Prev*”, que permitiam alterar a pagina de elementos demonstrados. De facto, tais botões eram incluídos em todas as paginas que incluíam uma lista de instancias.

Estes botões envolviam um quanto código extra da parte da aplicação, primeiro era utilizado quatro variáveis, o *current* que representava a página da lista verificada, *nRows* o numero de elemento por cada página da lista, *needsNext* e *needsPrev* que demonstram a necessidade do botão



“*Next*” e o botão “*Prev*”, estas duas ultimas variáveis seriam utilizadas para definir as características de visibilidade e habilitar dos botões.

Para verificar a necessidade do *Prev* era bastante simples, sendo que desde que *current* fosse maior que zero então havia pelo menos uma página anterior a mostrar.

Já a necessidade de *Next* era mais complexo, primeiro a informação para a listagem teria de ocorrer com um *SQLQuery* em vez de um *Aggregate* (como é feito atualmente), e nesta *query* era obtida *nRows* mais 1 instancia da entidade. Se o resultado dessa *query* for um numero de instancias igual a *nRows* mais 1 então é necessário o *Next*, para no mínimo obter a ultima instancia.

### 3.3.3 Menus

Na Figura 14 e na Figura 15 foi verificado os menus existentes, com o menu dos colaboradores incluindo links para:

- Vacancies, vagas já estabelecidas, permite introduzir uma nova vaga,
- Applications, lista de aplicações de candidatos a vagas já estabelecidas,
- Projects, projetos iniciados, permite iniciar um novo projeto,
  - Nota: é possível criar vagas a partir de um projeto.
- Clients, clientes da PS Tec, permite adicionar um cliente,
  - Nota: é possível criar projetos a partir de um cliente.
- Events, demonstra os eventos em que o utilizador participa, permite introduzir um novo evento,
- Forms, lista de formulários existentes, permite adicionar e alterar formulários,
- Curriculum, demonstra as candidaturas espontâneas que ainda não foram recusadas ou aceites,
- Candidates, lista dos candidatos já existentes,
  - Nota: cada candidato demonstrado permite ver as informações gerais, o currículo, dossiê de capacidades, aplicações e até entrevista já realizadas.

Já o menu de candidatos tem links para as páginas:

- General, informação geral do candidato,
- Curriculum, currículo do candidato, permite alterar o mesmo,
- Profile, dossiê de capacidades do candidato, permite alterar o mesmo,
- Tutorial, uma página de tutorial que explica algumas características do currículo e dossiê,
- Applications, aplicações a vagas existentes que o candidato esta aplicado,
- Vacancies, vagas disponíveis que o candidato ainda não se aplicou,
  - Nota: a página permite uma pesquisa inteligente de vagas.

- Availability, com esta página o candidato pode demonstrar a sua disponibilidade para entrevista.

Originalmente os menus eram mais simples, mas também menus eficazes, sendo que nas primeiras versões eram compostos unicamente por botões que serviam de *links*, alinhados horizontalmente na página.

Tal estrutura levantou um problema notável, o espaço ocupado por tantos botões. Cada link a adicionar necessitaria de um novo botão o que ocuparia mais espaço, por exemplo tendo em conta o numero de link atuais, 7, o menu de candidatos ocuparia duas linhas completamente, qualquer link extra e seria 3 linhas.

Uma possibilidade considerada para estes menus foi a utilização do *rich widget* Accordion, que também permite fazer algo semelhante ao DropDownButton. Mas este *widget* introduz uma limitação visual não introduzida pelo DropDownButton, quando um Accordion é aberto o menu é expandido horizontalmente, não só a o Accordion selecionado.

### 3.3.4 Pesquisa

Varias paginas que envolvem uma listagem de vários elementos inclui uma capacidade de pesquisa, permitindo filtrar os valores da listagem em função de um campo do tipo texto, como por exemplo nome para candidatos ou titulo para formulários.

Esta pesquisa inclui sempre duas capacidades, a da pesquisa em si como também a possibilidade de reiniciar a tabela, colocando a mesma do estado antes de uma pesquisa. Para tal é realizada uma ação semelhante a da Figura 43, a ação de pesquisa na pagina de candidatos.

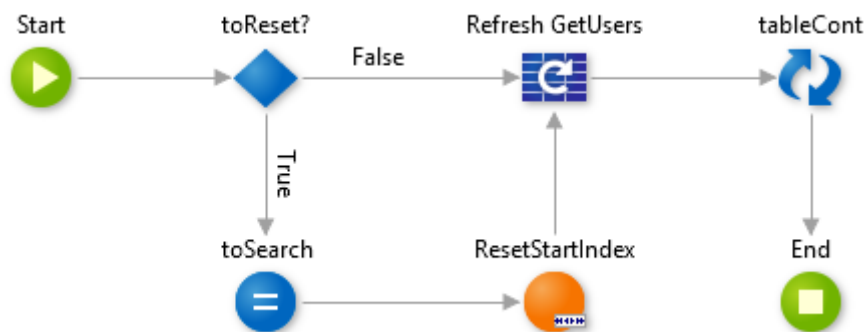


Figura 43 – Pesquisa

Como se pode verificar as pesquisas incluem uma variável para realizar, ou não, o reinício. Caso o mesmo tiver de ocorrer a variável de pesquisa assume o valor *null* e ocorre o *ResetStartIndex*, para colocar o índice da tabela a zero. Ocorra ou não o reinício é realizada a pesquisa, chamando o *Aggregate* (ou *SQL Query*) que obteve a tabela inicial, seguido de um *Ajax refresh* sobre a tabela.

### 3.3.5 Candidato – Availability

A pagina de introdução e demonstração de disponibilidade de um candidato não inclui uma preparação particularmente notável, sendo que na mesma é só obtido as instancias de CandidateAvailability associados ao candidato mais a lista de dias da semana e horas que compõem um dia de trabalho. Para as duas ultimas listas é utilizado a *function* DaysList e HoursBlock.

Mas a introdução duma nova instancia de CandidateAvailability, não é tão simples como criar uma nova instancia e adiciona-la a base de dados, como se pode verificar na Figura 44.

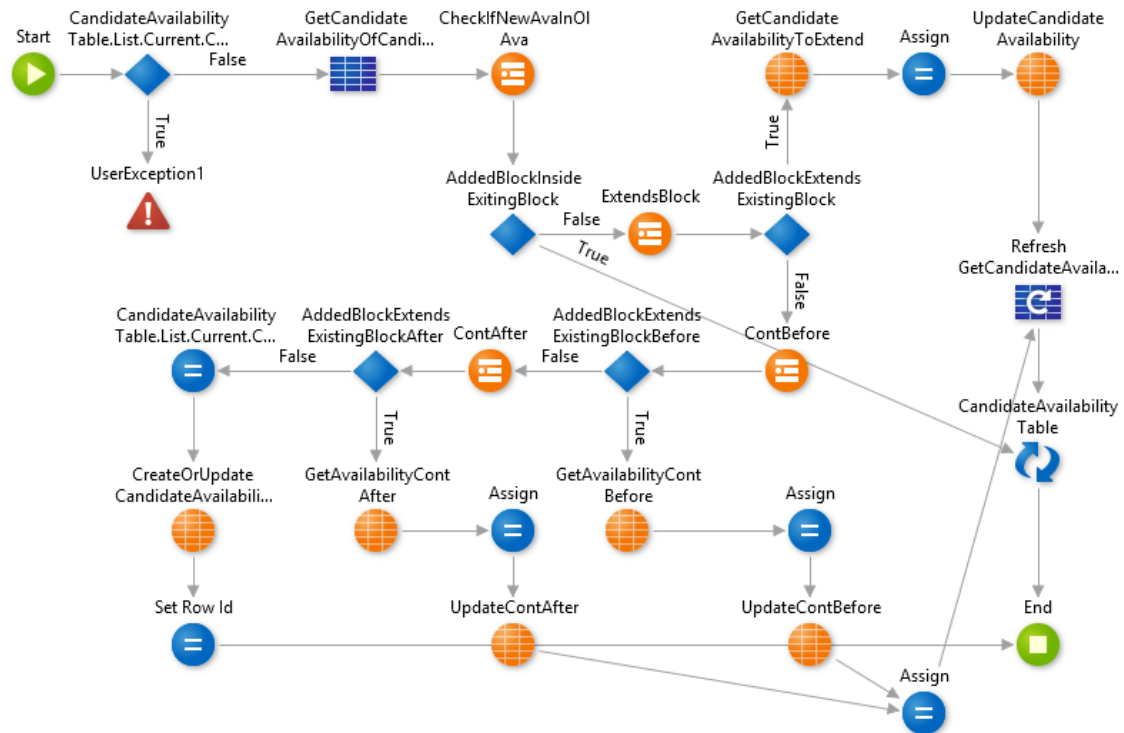


Figura 44 - Adicionar Availability

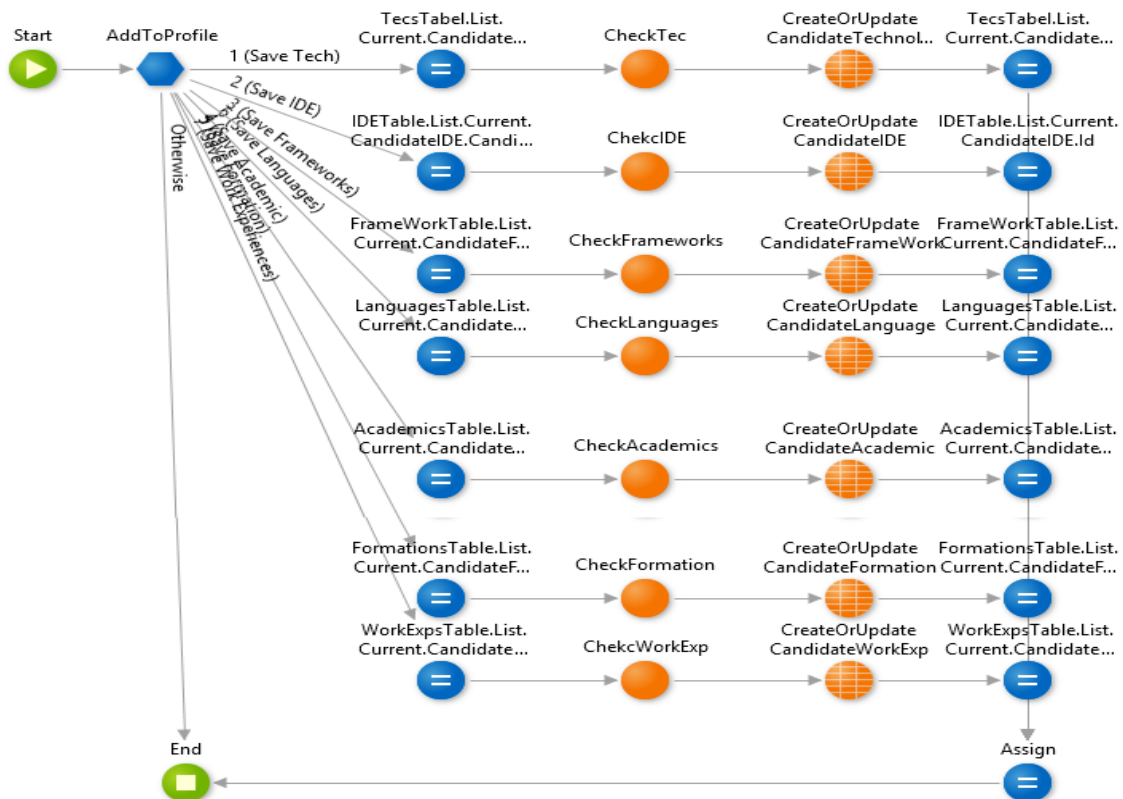
A complexidade da introdução deve-se ao fato que quando um candidato introduz um novo bloco de disponibilidade, existe três formas de interferir com blocos já existentes. Por causa disto, qualquer adição de bloco temporal, começa por verificar se o bloco interfere com qualquer bloco existente, e dependentemente de qual interseção acontecer é realizada uma alteração diferente a base de dados.

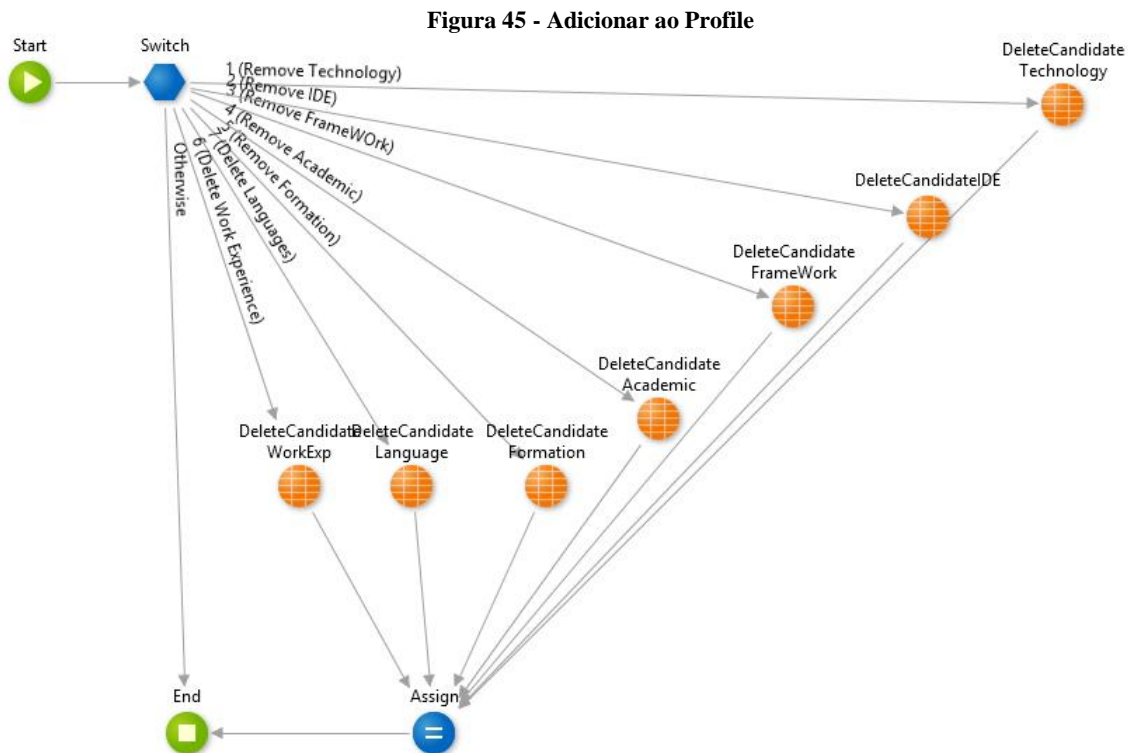
- O bloco a adicionar ocorre dentro de um bloco já estabelecido
  - O inicio do bloco a adicionar acontece depois do inicio do bloco estabelecido e o fim do bloco a adicionar acontece antes do fim do bloco estabelecido
  - Neste caso o bloco a adicionar é ignorado e não é realizado qualquer alteração a base de dados.
- O bloco a adicionar começa antes de um bloco já existente, e acaba no meio de tal bloco

- O início do bloco a adicionar acontece antes do início do bloco estabelecido e o fim do bloco novo acontece depois do fim do bloco estabelecido
- Neste caso o início do bloco estabelecido é alterado para o início do bloco a adicionar
- O bloco a adicionar começa no meio de um bloco já existente, e acaba depois de tal bloco
  - O início do bloco a adicionar acontece depois do início do bloco estabelecido e o fim do bloco a adicionar acontece depois do fim do bloco estabelecido
  - Neste caso o fim do bloco estabelecido é alterado para o início do a adicionar
- Não acontece qualquer intercessão
  - Não ocorrendo qualquer intercessão, o a adicionar novo pode ser simplesmente adicionado a base de dados

### 3.3.6 Candiato – Profile

Na Figura 23 foi demonstrado a página de dossier de capacidades, que como se pode verificar inclui todas as capacidades do candidato. Estas, em grande parte, são demonstradas por um conjunto de Editable Tables que permite adicionar elementos a mesma, mas não inclui qualquer mecanismo próprio para escrever ou remover o elemento a base de dados. Para tal é utilizado dois *function* um para adicionar e outro, Figura 45 e Figura 46, para remover, com cada utilizando um Switch que escolhe as ações a realizar tendo em conta uma input que recebe.





**Figura 46 - Remover do Profile**

No adicionar é sempre verificado se é possível adicionar a instancia, garantindo que não existe repetições, utilizado uma *function* diferente. Numa desta é verificado fatores como repetições e data corretas se é incluindo uma data de inicio e outra de fim é garantido que a primeira ocorre antes da segunda. Caso não for levantado qualquer exceção, o que indica que o elemento a adicionar não é repetido, então é criado ou atualizado o objeto a escrever.

Já o remover não inclui qualquer verificação, simplesmente remove o elemento da entidade indicada.

O único elemento do dossier de capacidades que não é demonstrado por uma Editable Tabel é as aplicações desenvolvidas, sendo que cada instancia desta entidade é muito extensa para uma só linha de texto. Em vez disso as aplicações são demonstradas com ListRecord, em que cada elemento é composto por um WebBlock.

Tal WebBlock demonstra um Edit Record ou um Show Record dependente do bloco estar no modo de edição ou não. Para mudar de estado cada bloco inclui um botão “Edit” que altera o estado e, salva qualquer alteração. Outro botão incluído neste bloco é o botão “Delete” que simplesmente notifica a pagina do dossier. O notificar alerta a pagina do “Delete” e por isso não só remove o projeto escolhido, como também realiza um *refresh* na lista de aplicações desenvolvidas para demonstra a remoção do projeto.

Já para adicionar um novo projeto, o botão “Add” invoca uma *PopUp* onde é possível adicionar as informações do novo projeto. O *PopUp* inclui dois botões, o “Cancel” simplesmente

fecha o *PopUp*, já o “Add” adiciona o projeto a base de dados, notifica a pagina do dossier e fecha a *PopUp*. Quando a pagina do dossier é notificada da adição dum projeto realiza um *refresh* a lista de aplicações para demonstra o novo projeto.

### 3.3.7 Colaborador – Curriculum

Por esta pagina um colaborador consegue verificar currículos enviados para possíveis candidatos, para tal a pagina inclui uma lista de todas as instancias da entidade SpontaneousCurriculum, com cada entidade incluindo dois botões que executam duas ações opostas. A primeira ação é aceitar currículo, onde é criado um novo candidato utilizando a informação do SpontaneousCurriculum. Já a segunda é negar em que o mesmo é simplesmente removido da base de dados. Esta segunda ação é bastante simples sendo que simplesmente utiliza a função delete para remover a instancia de SpontaneousCurriculum que foi negado.

Já a primeira ação é um pouco mais complexa, como se pode verificar na Figura 47. Esta começa com a criação e encriptação de uma password. A geração é bastante simples, sendo que é utilizado a função *built-in* GeneratePassowrd que gera (no caso desta ação) uma sequencia de 15 caracteres (números e/ou letras) aleatórios. Já para realizar a encriptação é necessário primeiro produzir o *username* do novo utilizador.

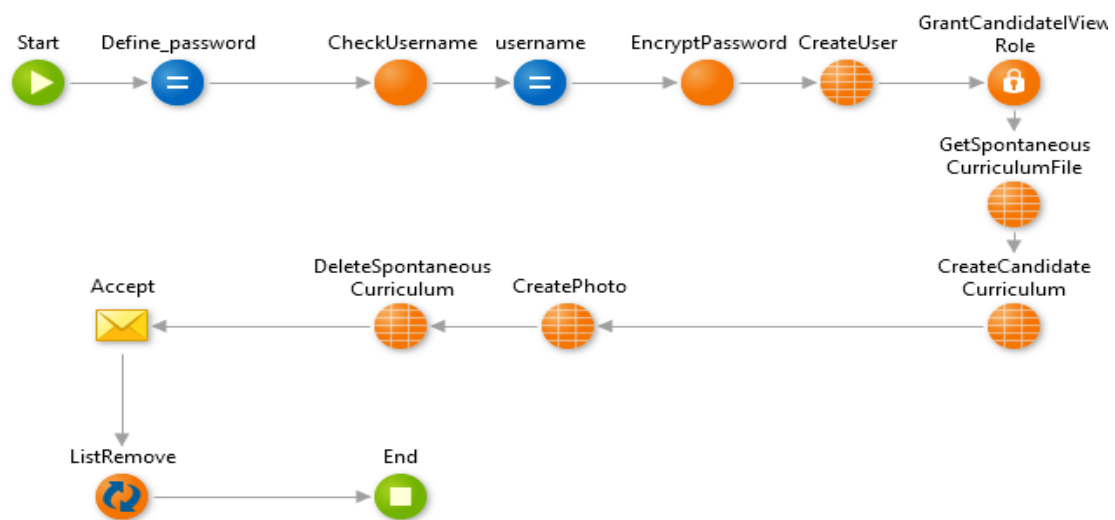


Figura 47 – Aceitar Currículo

Para tal é primeiro criado um *username* que é simplesmente o nome inteiro fornecido pelo candidato, com os espaços substituídos por um ponto. Mas sendo que nada garante que tal *username* não é repetido, é obtido a lista de todos os utilizadores que tenham o *username* igual ao *username* produzido. Sé a lista não for vazia então é adicionado ao *username* a dimensão de tal lista.

Com a password e o *username*, a encriptação é bastante simples, sendo simplesmente realizado *function* *EncryptPassword* com a password e *username* produzido. O produto de tal função mais a informação fornecida pelo candidato (na sua candidatura) são utilizados para produzir o novo utilizador. Também removido o SpontaneousCurriculum aceitado e enviado um

email ao novo utilizador a informar do novo utilizador, mais particularmente candidato e password.

### 3.3.8 Colaborador – Events

Nesta pagina um colaborador consegue visualizar o seu calendário de eventos, tanto ao mês como por dias da semana, com cada sendo demonstrado com um calendário diferente. Para esta pagina é utilizado o componente *FullCalendar2*, uma aplicação pulicada na *Forge* que permite desenvolver um calendário que pode ser alterado, como permite realizar varias interações, com todas estas sendo capturadas pela ação *GetNotifyCalendarCallback*.

Para começar a preparação da pagina envolve a *function* *FormEventsMonth* que gera os *Events*<sup>4</sup> que demonstram a estado de ocupação dos dias do mês atual. Esta *function* recebe o identificador do utilizador cujo calendário esta a ser formado, o mês a demonstrar no calendário e uma data de inicio para o calendário. Esta *function* pode ser verificada na Figura 48.

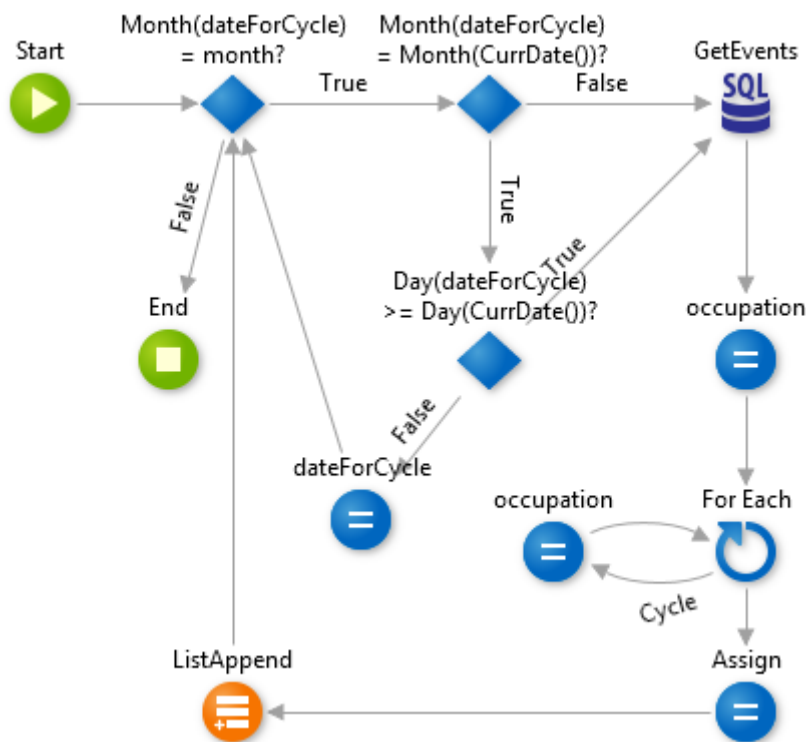


Figura 48 – Formação dos Events para um mês

A *function* em si foca-se num ciclo em que uma data, *dateForCycle*, incrementa por um dia, até o mesmo ultrapassar o mês a demonstrar. Dentro do ciclo é realizado um *SQL Query* que essencialmente obtém todos os *Events*, a instancia da base de dados não a estrutura da *FullCalendar2*, que o utilizador atual participa, tanto como responsável como simples participante. O tempo que cada *Event* obtido é acumulado numa variável, gerando assim o valor de ocupação, para gerar a ocupação também fui considerado simplesmente utilizar o numero de eventos, mas tal levantava o problema que desta forma um dia com *Event* que ocupa 4 horas, por

<sup>4</sup> Estrutura de *FullCalendar2* que representa um evento

exemplo, seria visto tendo uma ocupação menor que um dia com 8 Events, com cada ocupando 30 minutos.

Tendo a ocupação gerada é gerado o Event, a estrutura a adicionar, que demonstra a ocupação do dia, com cada variando em duas características, background-color, cor de fundo para demonstra o estado, e titulo, onde é referido o estado do dia. As três possibilidades são:

- Com ocupação menor ou igual a 2 horas,  $\frac{1}{4}$  do dia do trabalho, o titulo é “Empty” e a background-color é verde
- Com ocupação entre 2 a 6 hora, não incluindo qualquer das duas, o titulo é “Occupied” e cor é azul
- Com 6 ou mais horas o titulo é “Full” e a cor é vermelha

Tanto a preparação desta pagina como a ação que permite mudar o mês demonstrado no calendário, utiliza a FormEventsMonth para produzir a lista de Events, que depois é serializada para JSON, utilizando OutSystems2JSON, serialização que funciona como fonte de informação para o calendário a demonstrar.

Outra funcionalidade desta pagina é permitir introduzir novos eventos no calendário do utilizador atual, para tal quando uma data é seleccionada no calendário o mesmo demonstra a semana do dia escolhido, que permite introduzir o novo evento. Este mesmo calendário permite também verificar qualquer evento já estabelecido que envolve o utilizador atual.

Para tal é associado uma *action* a OnNotify do calendário de mês, que pode ser verificada na Figura 49. Esta *action* começa com a utilização do GetNotifyCalendarCallback, para verificar a interação que ocorreu entre o utilizador e o calendário, mais particularmente o dia que foi seleccionado pelo utilizador. Com a informação do dia seleccionado é executada a *Function* FormEventsWeek que devolve a lista de Events, a estrutura, que representam todos os eventos que o utilizador atual ira participar na semana seleccionada. O resultado desta *function* é depois serializada com OutSystems2JSON, servindo como fonte de informação ao calendário da semana.

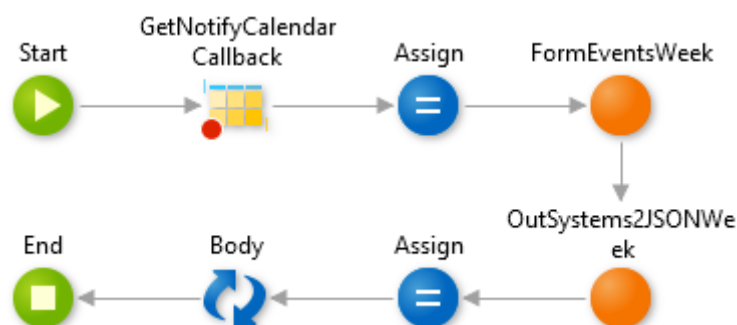


Figura 49 – Alteração do calendário de mês para semana

Esta *action* é bastante simples, mas a FormEventsWeek, Figura 50, utilizada merece alguma verificação. A mesma, é focada num ciclo para criar Events, que representam os eventos em que o utilizador atual participa uma semana, para tal a *action* começa por criar dois valores dateStart, inicio da semana e dateEnd o fim da semana.



O primeiro é igual a subtrair a data que a *action* recebe, a data selecionada no calendário, ao numero do dia da semana menos um, essencialmente devolvendo sempre a data do inicio da semana, já a segunda data só envolve adicionar 5 dias a data inicial.

Tendo as duas datas, é obtidos todas estancias de Events em que o utilizador atual participa, dentro das duas datas, incluindo as duas. Para cada é criado evento obtido, uma estrutura Event que inclui o identificador do Event, objeto da base de dados, o titulo, fim e inicio do mesmo. Esta estrutura é depois adicionada a lista de Events que é um input da *action*.

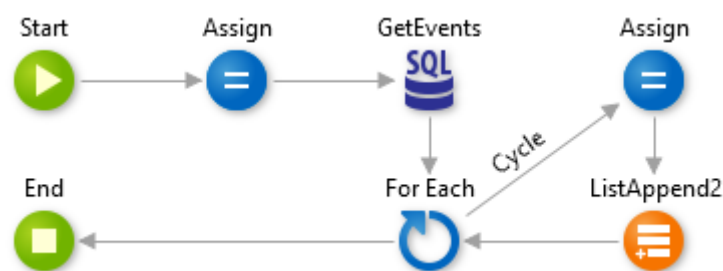


Figura 50 - Formação dos Events para uma semana

Por ultimo deve ser verificado o que ocorre quando um utilizador interage com o calendário da semana. Para tal existe a *action* da Figura 51, que esta associada a característica OnNotify do calendário da semana.

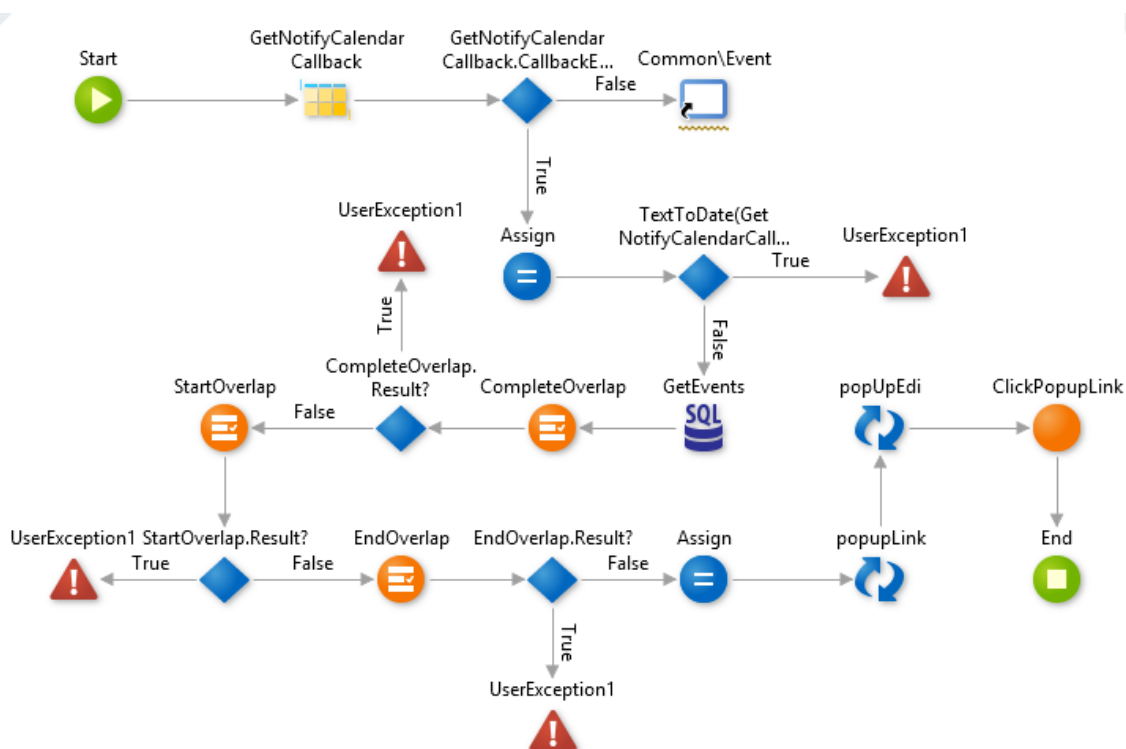


Figura 51 – Inicio de criação de um Event, ou escolha de um Event existente

Esta *action* começa por verificar que interação ocorreu entre utilizador e calendário. Se um evento já estabelecido foi pressionado, então é aberta a pagina que demonstra o evento numa forma mais detalhada.

Caso o utilizador criou um bloco por *click and drag*, um Event é para ser criado baseado no bloco criado. Mas primeiro é verificado se o bloco pode ser produzido, sendo que o bloco pode ser criado num dia que já passou ou no dia atual, mas numa hora que já passou, caso um destes casos acontecer então é levantado uma exceção.

Caso o Event não aconteça num momento já passado é verificado três possibilidades de interferência entre o novo evento e eventos já estabelecidos. Para tal é obtido todos os Events que ocorrem no dia do evento a adicionar. Desta lista é formada outras três que se uma só incluir um único elemento, então existe interferência entre o evento a adicionar e um Event já estabelecido. As listas são:

- Todos os Events cujo inicio ocorre depois do inicio do evento a adicionar e cujo fim acontece antes do fim do evento a adicionar
- Todos os Events cujo inicio ocorre depois do inicio do evento a adicionar e cujo fim acontece depois do inicio do evento a adicionar
- Todos os Events cujo inicio ocorre antes do fim do evento a adicionar e cujo fim acontece antes do fim do evento a adicionar

Caso nenhuma interferência ocorrer então é utilizado uma RunJavaScript para forçar a ativação do link *PopUpLink*, que em si abre um *PopUp* que demonstra a data e horas seleccionadas para criar o evento, permitindo realizar tal criação, direccionado o utilizador para a pagina que será verificada na próxima subsecção.

### 3.1.1 Colaborador – AddEvent

Por esta pagina um colaborador conseguira introduzir um novo Event, qualquer tipo de Event, que ocorrera na data e tempos que a pagina recebe como paramentos de entrada.

A pagina inclui bastantes *actions* bastante simples, com uma *action* para paginação, uma para pesquisa e outra para reiniciar a tabelas demonstradas, todas estas são bastante simples, sendo semelhantes as *actions* normais que realizam estas funcionalidades, simplesmente alterando o alvo da funcionalidade tendo em conta a EventType.

A única *action* a notar e a associada a criação do Event, que em si só é notável devido a sua utilização de um *PopUp*. Quando um colaborador já cumpriu um numero de definições, já a verificar, o mesmo pode pressionar o botão de *Add* que enfoca um *PopUp* onde é possível seleccionar uma localização, a única característica comum a todos os Events ainda a definir. Quando o mesmo é seleccionado a ação verificada na Figura 52.

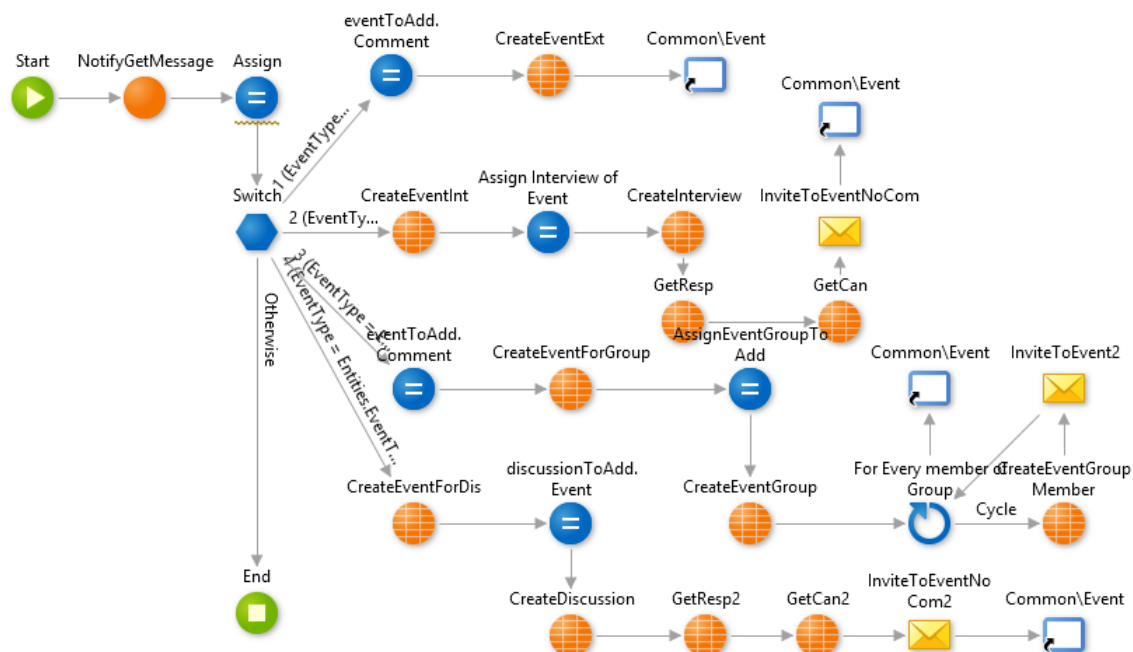


Figura 52 – Adição de um novo Event

Como se pode verificar a ação começa por obter a mensagem do *PopUp*, que em si representa o identificador da localidade escolhida, que é utilizado para definir o campo *field* da instancia de Event a adicionar. Com tal localização definida o Event a adicionar é completamente definido, em função das informações da pagina e o Event é criado. Caso o Event criado envolve outros participantes, sem considerar o responsável, tais participantes são informados com o enviar o email. Independentemente do Event criado esta ação acaba sempre direcionado o utilizador atual para a pagina do Event criado.

Outra utilização de um *PopUp* nesta pagina é na listagem de Form a adicionar a um Interview, sendo que cada elemento inclui a possibilidade de demonstrar o formulário por inteiro por um *PopUp*.

Como referido anteriormente a adição do Event só pode ocorrer depois de cumpriu um numero de definições baseado no tipo de Event a adicionar, tais definições são:

- External – nenhuma definição é necessária
- Discussion – um candidato foi escolhido
- Interview – um candidato e um formulário foram escolhidos
- Group – o grupo do Event tem pelo menos um utilizador

### 3.1.2 Colaborador – VacancyAdd

Esta pagina permite um colaborador introduzir uma nova vaga, definindo os diferentes passos do mesmo, incluindo o formulário de cada e as varias ferramentas e linguagens cujo conhecimento é indispensável para o cargo anunciado pela vaga. A única ação a notar desta pagina é a que permite adicionar a nova vaga em si, que pode ser verificada na Figura 53.

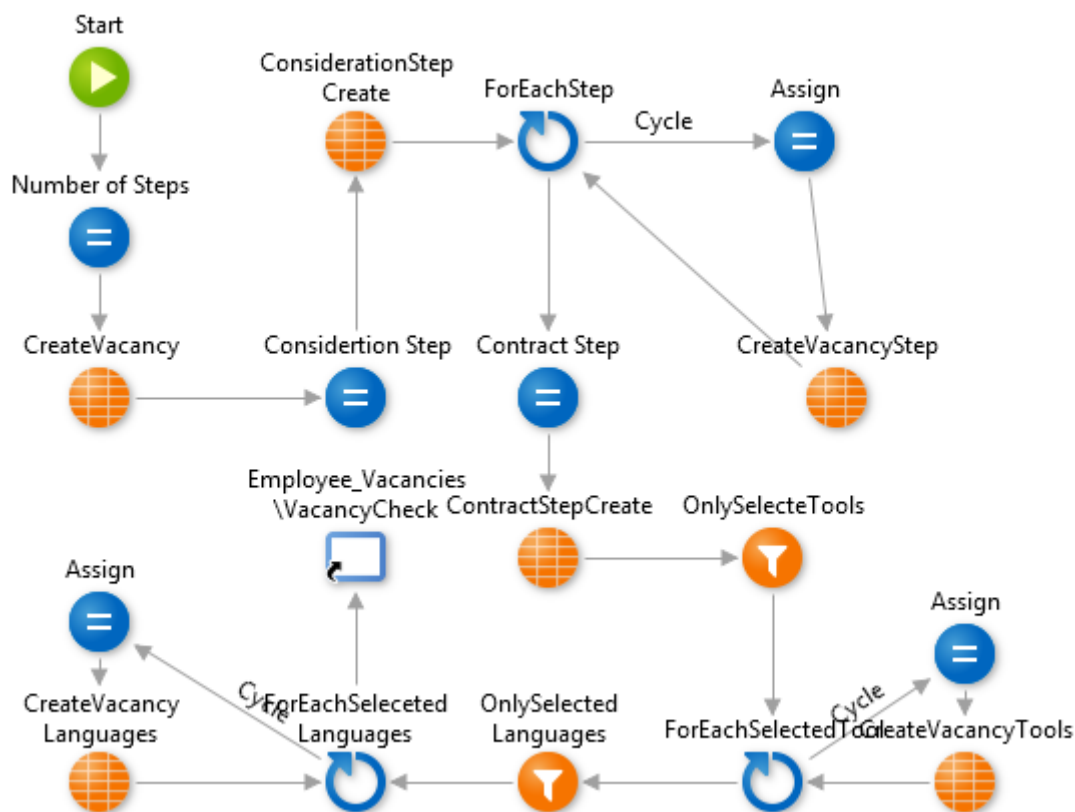


Figura 53 – Adição de Vacancy

Nesta ação, depois de ser criada a instancia de Vacancy que representa a vaga a adicionar, é executado três ciclos diferentes:

- No primeiro acontece para a lista de passos adicionados, sendo adicionado um VacancyStep para cada passo
- Depois todas as ferramentas que foram selecionadas, criando uma instancia de VacancyTools para cada ferramenta selecionada
  - o OnlySelectedTools filtra a lista de ferramentas possíveis, para demonstra unicamente as selecionadas
- Por ultimo é realizado um ciclo semelhante ao anterior, mas em vez de ferramentas é linguagens selecionadas que são utilizadas para produzir instancias da entidade VacancyLanguages
  - o OnlySelectedLanguages tem uma funcionalidade idêntica a OnlySelectedTools, mas para as linguagens em vez das ferramentas

Também deve ser notado que nesta pagina são demonstradas todas as ferramentas e linguagens reconhecidas pela aplicação, sendo associada a cada um booleano que demonstra se a mesma foi selecionada ou não. Para tal é utilizado duas estruturas que essencialmente associam o nome e identificador de uma ferramenta ou linguagem a um booleano.

### 3.1.3 Colaborador – VacancyCheck

Com esta pagina um colaborador pode verificar uma vaga, mais a candidaturas a mesma, com tais candidaturas sendo demonstradas num Pie Chart, como se pode verificar na figura Figura 34. A *action* demonstrada na Figura 54 forma as informações deste Pie Chart.

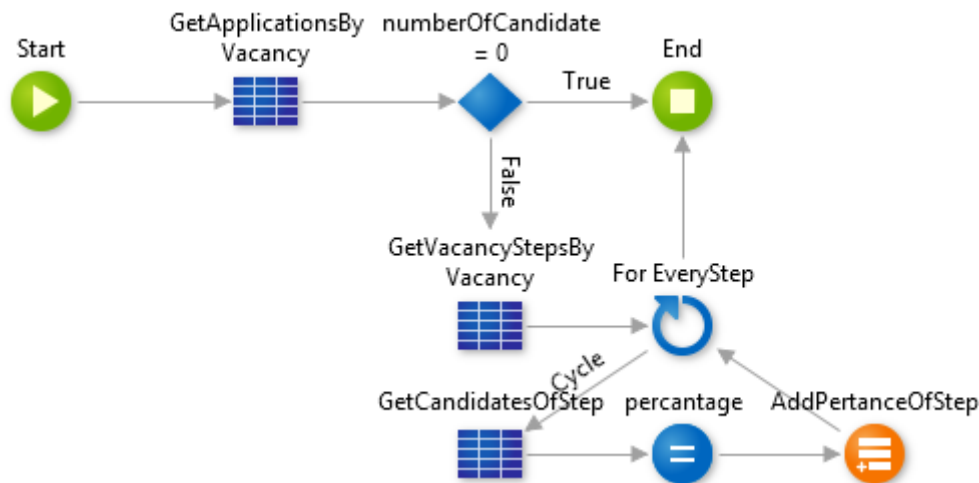


Figura 54 – Geração de percentagens para o Pie Chart

Para formar tais informações a *action* começa por obter todas as candidaturas a vaga a demonstrar no Pie Chart. Se não houver nenhuma candidatura então a *action* acaba de seguida, se houver pelo menos uma candidatura é realizado um ciclo para cada passo que compõem uma vaga.

Para cada passo é obtido o numero de candidaturas nesse mesmo passo que em conjunto com o numero de candidaturas obtidas no inicio, permite obter a percentagem de candidaturas nesse passo. Esta percentagem mais um *label* são depois adicionadas a lista que fornecem a informação para a Pie Chart. A *label* referida é sempre composta pelo o tipo de passo da vaga mais o nome do formulário utilizado, no caso das entrevistas técnicas.

O único outro fator a notar desta pagina é que o Pie Chart não só serve para demonstrar a percentagem de colaboradores em cada passo, mas também serve de link para a pagina que demonstra o paço numa forma mais notável.

Para tal é associado a *OnClick destination* a *action* que pode ser verificada na Figura 55, que utiliza o *DataPoint\_GetClicked* para obter a *label* da secção seleccionada. Desta é obtido o *type* que igual a *label* menos a ultima palavra, se a *label* for uma só palavra então a *type* é a *label* por inteiro.

Com o *type* é obtido o passo pressionado, na vaga que a pagina demonstra, se houver um só passo obtido então o identificador do mesmo é utilizado para direccionar o utilizador para a pagina do passo. Na situação de houver mais que um passo resultante, exemplo existe várias entrevistas do tipo técnica, então o identificador é obtido, utilizando a ultima palavra da *label* para obter o passo cujo o titulo do formulário é a ultima palavra.

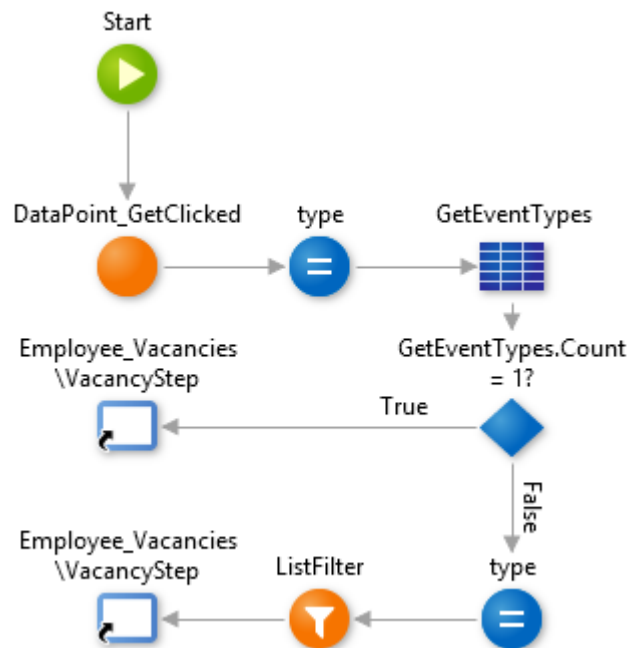


Figura 55 – Escolha de secção do Pie Chart

### 3.4 Desenvolvimento Mobile

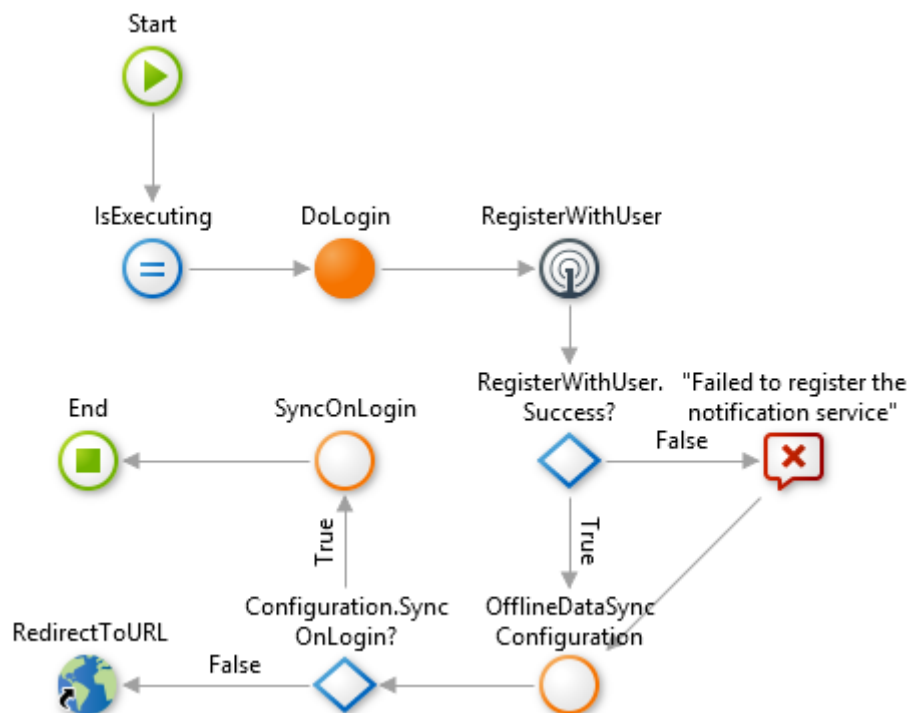
Como a secção anterior, esta secção ira demonstrar as implementações que podem ser consideradas como mais notáveis no desenvolvimento da componente *mobile* do projeto. Estas implementações são restringidas as notificações e sincronismo dos dados entre a base de dados local e remota.

#### 3.1.1 Notificações

Como já fui estabelecido, a aplicação *mobile* inclui nas suas funcionalidades a visualização de notificações que irão demonstrar informações importantes. Para realizar esta funcionalidade foi escolhido o *pluggin* da *Forge OneSignal* que permite realiza as notificações desejadas com pouca adição de logica ou peso a aplicação.

Os únicos requisitos para utilizar esta *pluggin* são o facto que a aplicação tem de ser registada nos serviços da *OneSignal*, o que ocorre fora da aplicação e registrar cada utilizador e o seu dispositivo no mesmo servidor, associando os mesmos a aplicação já registrada.

Por isso a ação de *login* que ocorre na aplicação *mobile*, que pode ser averiguada na Figura 56 , inclui a utilização da ação *RegisterWithUser* que realiza a registo do utilizador nos servidores de *OneSignal* utilizando o identificador da aplicação *mobile* desenvolvida, obtida no registo da mesma no servidor da *OneSignal*.



**Figura 56 - Registro de utilizador na OneSignal**

Também se pode verificar que o é incluindo uma verificação de erro depois do registro, garantindo assim que se o registro não ocorrer, o utilizador é informado da situação, utilizando o uma mensagem de error.

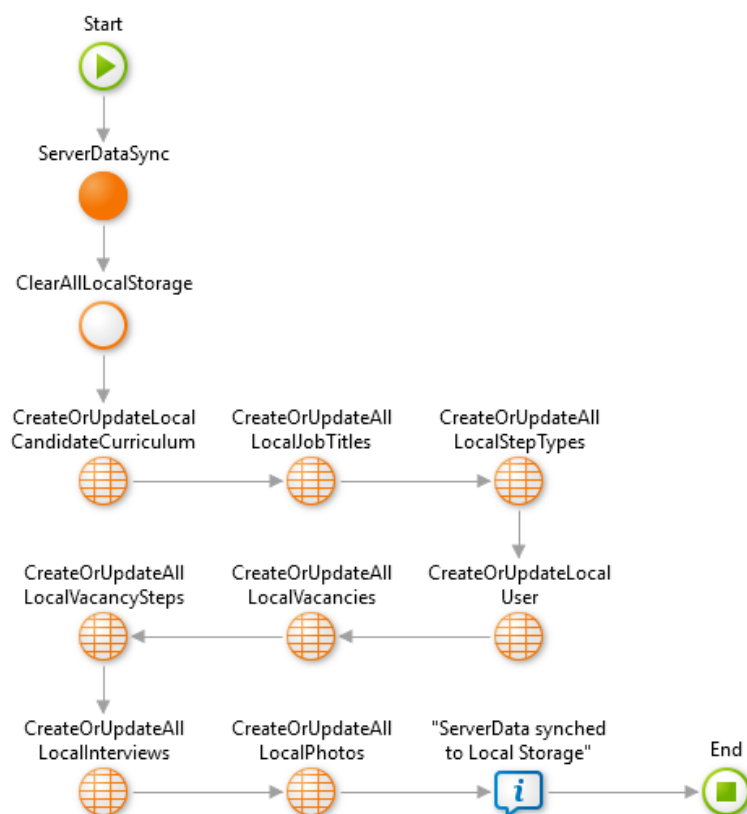
Esta funcionalidade ainda se encontra no seu estado inicial, sendo que as criações das notificações ainda não foram implementadas, estas irão em principio ocorrer três situações:

- Quando um processo de entrevista que um candidato participa acabou ou fui passou para outro passo
- Quando um evento que inclui o candidato é criado
- Quando existe um evento em o candidato participa próximo da data atual

### **3.1.2 Sincronismo**

Na secção 2.3 fui referido o conceito de sincronismo que é será aplicado na vertente *mobile* do projeto, sendo a mesma bastante simples e leve. Esta sincronização ocorrera em dois padrões de sincronização, a primeira e mais simples das duas, *Read-Only*, ocorre pela ação *SyncOnLogin*, que ocorre no *login*, como se pode verificar na Figura 56.

Esta ação, que pode ser verificada na Figura 57, começa por utilizar a ação *ServerDataSync* que essencialmente obtém todos os dados que a aplicação acede a base de dados ou a *local storage* já estabelecida, caso não exista a um acesso a internet. Tendo as informações a sincronizar, os dados da *local storage* são limpos para de seguida serem restabelecidos com as várias ações *CreateOrUpdate* que seguem.



**Figura 57 - Sincronismo Read-Only**

A única situação, já estabelecida, que não envolve um sincronismo do tipo *Read-Only*, ocorre quando um colaborador que alterar o seu currículo, que sendo uma escrita em que não existe o conflito, é utilizado o padrão de sincronismo *Read/Write Data Last Write Wins*. Como se pode notar o currículo é um dos vários valores adicionados ao *local storage*, por isso a alteração do currículo não só ocorre na base de dados mas também ocorre no *local storage*.



## **4. Avaliação Experimental**

## **5. Conclusões**

# Referências

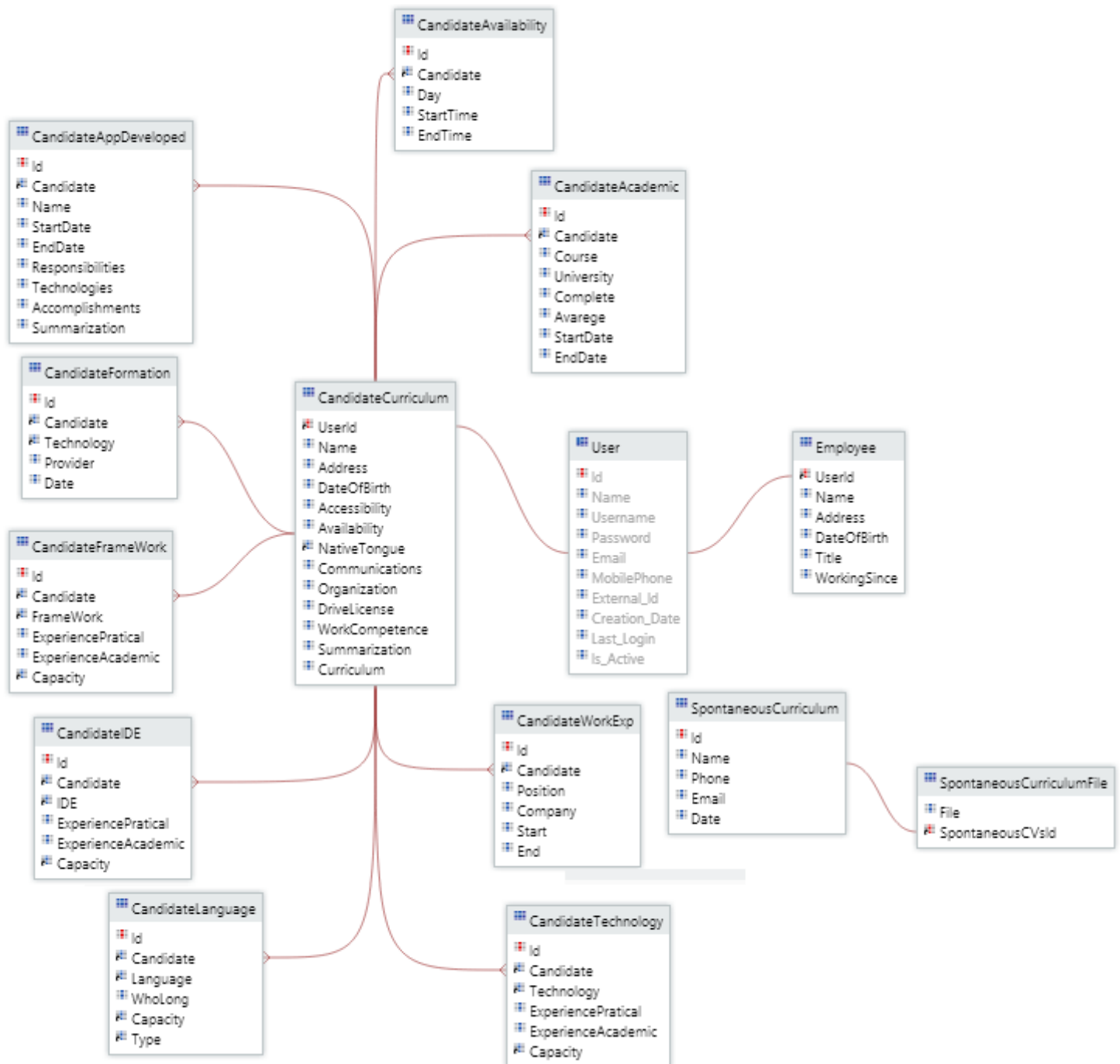
- [1] Mind Source, “Portal de Emprego,” Mind Source, [Online]. Available: <https://emprego.mindsources.pt/>. [Acedido em Abril 2018].
- [2] Randstad, “My Profile,” Randstad, [Online]. Available: <https://myprofile.randstad.pt/>. [Acedido em Abril 2018].
- [3] LinkedIn Corporation, “LinkedIn,” LinkedIn Corporation, 05 Maio 2003. [Online]. Available: <https://pt.linkedin.com/>. [Acedido em Abril 2018].
- [4] ITJobs Portugal, “ITJobs,” ITJobs Portugal, [Online]. Available: <https://www.itjobs.pt/>. [Acedido em Abril 2018].
- [5] OutSystems, “OutSystems Architecture,” OutSystems, [Online]. Available: [https://success.outsystems.com/Evaluation/Architecture/2\\_OutSystems\\_Platform\\_architecture](https://success.outsystems.com/Evaluation/Architecture/2_OutSystems_Platform_architecture). [Acedido em Abril 2018].

## **A.1 Diagramas da Aplicação**



## A.2 Modelos de dados

Sendo que a base de dados desenvolvida para este projeto inclui um notável numero de entidades, o modelo de dados será demonstrado em três submodelos, com a divisão das entidades sendo igual a verificada na secção 3.2.



Modelos De Dados 1 - Users



Tabela 1 - Application

Atributo	Tipo	Descrição
Application	Application Identifier	Identificador do Application a que o passo pertence
VacancyStep	VacancyStep Identifier	Passo da vaga que a instancia tenta cumprir
State	State Identifier	Estado do passo
Event	Event Identifier	Evento que podera ocorrer no passo

Tabela 2 - ApplicationCurrentStep

Atributo	Tipo	Descrição
Id	Identifier	Identificador do ApplicationInterview
Application	Application Identifier	Aplicação a que a entrevista é associada
Interview	Interview Identifier	Entrevista a associar

Tabela 3 – ApplicationInterview

Atributo	Tipo	Descrição
Id	Identifier	Identificador da CandidateAcademic
Candidate	User Identifier	Identificador do candidato
Course	Text(50)	Nome do curso da entidade
University	Text(50)	Nome da universidade fornecedor do curso
Complete	Boolean	Espressa se o curso já foi terminado ou não
Avarege	Integer	Média final do curso
StartDate	Date	Data de inicio do curso
EndDate	Date	Data de fim do curso, caso o curso ainda não foi obtido, este é null

Tabela 4 – CandidateAcademics

Atributo	Tipo	Descrição
Id	Identifier	Identificador da CandidateAppDeveloped
Candidate	User Identifier	Identificador do candidato
StartDate	Date	Data de iniciação do projecto
EndDate	Date	Data de finalização do projecto
Responsibilities	Text(150)	Responsibilidades tomadas pelo candidato durante o desenvolvimento do projecto
Technologies	Text(150)	Tecnologias utilizadas no desenvolvimento do projecto
Accomplishments	Text(150)	Objectivos do projecto cumpridos
Summarization	Text(250)	Breve explicação da aplicação, funcionalidades

Tabela 5 – CandidateAppDeveloped



<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador da CandidateAvailability
Candidate	User Identifier	Identificador do candidato
Day	Int	Dia da semana para disponibilidade
StartHour	Time	Início do bloco de disponibilidade
EndHour	Time	Fim do bloco de disponibilidade

**Tabela 6 - CandidateAvailability**

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
UserId	User Identifier	Identificador de CandidateCurriculum
Name	Text(256)	Nome do candidato
Address	Text(150)	Morada do candidato
DateOfBirth	Date	Data de nascimento do candidato
Accessiblity	Text(100)	Disponibilidade geográfica do candidato
DriveLicense	Text(50)	Licença de condução do candidato
NativeTongue	Identificador Languages	Linguagem nativa do candidato, é limitada as languages já estabelecidas na aplicação
Availability	Text(100)	Acessibilidade do candidato, tempo inteiro tempo parcial, diurno, pós-laboral, imediato ou não
WorkCompetence	Text(150)	Ideia geral das experiencia na área de tecnologias, quantos anos, tipo de cargos, preferencias por cargos
Communications	Text(150)	Ideia geral das capacidades de comunicação do candidato. A sua capaciade de se expressar em grandes grupos, se está disponível a falar em diferentes languages
Organization	Text(150)	Ideia geral das capacidades de organização do candidato. Trabalha melhor em grupo ou individualmente, prefere posições de liderança ou não
Summarization	Text(350)	Simples sumarização do candidato, fatores extras a ter em conta em futuras considerações, como razões por escolher a área de tecnologias
Curriculum	Binary Data	Ficheiro pdf, curriculo original

**Tabela 7 – CandidateCurriculum**

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador da CandidateFormation
Candidate	User Identifier	Identificador do candidato
Technology	Technologies Identifier	Tecnologia foco da formação
Provider	Text(50)	Fornecedor da formação
Date	Date	Data de obtenção da formação

Tabela 8 - CandidateFormations

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador da CandidateFramework
Candidate	User Identifier	Identificador do candidato
ExperiencePractical	Integer	Anos de experiencia pratica com a framework
ExperienceAcademic	Integer	Anos de experiencia academcia com a framework
Framework	Framework Identifier	Framework com que o candidato têm experiencia
Capacity	Capacity Identifier	Capacidade com a framework

Tabela 9 – CandidateFrameworks

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador da CandidateIDE
Candidate	User Identifier	Identificador do candidato
IDE	IDE Identifier	IDE com que o candidato têm experiencia
ExperiencePractical	Integer	Anos de experiencia pratica com a IDE
ExperienceAcademic	Integer	Anos de experiencia academcia com a IDE
Capacity	Capacity Identifier	Capacidade com a IDE

Tabela 10 - CandidateIDE

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador da CandidateLanguage
Candidate	User Identifier	Identificador do candidato
Language	Language Identifier	Linguagem com que o candidato têm experiencia
Capacity	Capacity Identifier	Capacidade com a linguagem
Type	TypeLanguage Identifier	Tipo de expressão que o candidato é capaz
WhoLong	Integer	Numero de anos que o candidato utiliza a linguagem

Tabela 11 – CandidateLanguage

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador da CandidateTechnology
Candidate	User Identifier	Identificador do candidato
Technology	Technology Identifier	Linguagem tecnológica com que o candidato têm experiencia
ExperiencePratical	Integer	Anos de experiencia pratica com a linguagem tecnológica
ExperienceAcademic	Integer	Anos de experiencia academcia com a linguagem tecnológica
Capacity	Capacity Identifier	Capacidade com a linguagem tecnológica

Tabela 12 - CandidateTechnology

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identficator da CandidateWorkExp
Candidate	User Identifier	Identificador do candidato
Position	Text(50)	Posição ocupada pelo candidato
Company	Text(50)	Empresa onde candidato trabalho
Start	Date	Data do inico do cargo
End	Date	Data do fim do cargo

Tabela 13 - CandidateWorkExp

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador da Capacity
Label	Text(50)	Descrição do nivel
Order	Integer	Garante que os niveis seguem a ordem correcta
Knowlegment	Text(50)	Conhecimento esperado para o nivel
StandardOfWork	Text(50)	Padrão de trabalho esperado para o nivel

Tabela 14 – Capacity

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador de Client
Name	Text(50)	Nome do cliente
Locality	Text(50)	Localidade da sede do cliente
Photo	Binary Data	Fotografia/logotipo do cliente
Information	Text(150)	Informação geral sobre o cliente a considerar
Since	Date	Quando o cliente se tornou cliente da empresa

Tabela 15 – Client

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Event	Event Identifier	Identificador de Event e Discussion
Candidate	CandidateCurriculum Identifier	Identificador do candidato participante

Tabela 16 – Discussion

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
UserId	User Identifier	Identificador de Employee
Name	Text(250)	Nome inteiro do colaborador
Address	Text(150)	Morada do colaborador
DateOfBirth	Date	Data de nascimento do colaborador
Title	Text(50)	Titulo do cargo do colaborador
WorkingSince	Date	Data de entrada do colaborador na empresa

Tabela 17 – Employee

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador de Event
Date	Date	Data em que o evento ocorrer
Start	Time	Tempo a que o evento começa
End	Time	Tempo, esperado, a que o evento acaba
Responsible	User Identifier	Colaborador responsavel pelo evento
EventType	EventType Identifier	Tipo do evento
Comment	Text(100)	Comentario que o responsavel pode formar, pode servir para responsavel estabelecer uma informação geral do evento

Tabela 18 - Event

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Event	Event Identifier	Evento para o grupo fui criado, serve como identificador do EventGroup
Number	Integer	Numero de individuos no grupo

Tabela 19 - EventGroup

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador de EventGroupMember
Group	Group Identifier	Grupo a que membro pertence
User	User Identifier	Membro do grupo

Tabela 20 - EventGroupMember

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador do EventType
Label	Text(50)	Tipo de evento em si

Tabela 21 - EventType

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador do Form
NumberOfQuestions	Integer	Numero de perguntas que compõem o formulário
Title	Text(100)	Titulo do formulário, normalmente o tipo de entrevista a realizar ou o nome de um cargo
Create	Date	Data de criação do formulário
isGeneral	Boolean	Fomulario é utilizado para entrevistas gerais

Tabela 22 - Form

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador Da FormQuestions
Form	Form Identifier	O formulário a que a questão pertence
Question	Text(150)	Questão em si

Tabela 23 - FormQuestion

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador de Interview
Candidate	User Identifier	O candidato que participa na entrevista
Form	Form Identifier	Formulário para entrevista
Date	Date	Quando ocorreu a entrevista
IsComplete	Boolean	Se a entrevista foi completa ou não
Event	Event Identifier	Evento em que ocorreu a entrevista

Tabela 24 - Interview

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador da InterviewAnswer
Interview	Interview Identifier	Entrevista a que a resposta pertence
FormQuestion	FormQuestions Identifier	Pergunta que a resposta responde
Answer	Text(100)	Resposta em si

Tabela 25 - InterviewAnswer

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identificador	Identificador da JobTitle
Label	Text(50)	Nome do cargo de trabalho

Tabela 26 - JobTitle

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identificador	Identificador da Language
Label	Text(50)	Nome da linguagem representada

Tabela 27 – Language

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identificador	Identificador da Location
Name	Text(50)	Nome associado a localização
Local	Text(50)	Endereço da localização

Tabela 28 - Location

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
UserId	User Identifier	Identificador do User, cuja a fotografia pertence
File	Binary Data	Fotografia em si

Tabela 29 - Photo

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador de Project
Name	Text(50)	Nome do projecto
Summary	Text(200)	Sumarização do projecto

Tabela 30 –Project

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador de ProjectClient
Project	Project Identifier	Identificador do Projecto
Client	Client Identifier	Identificador do Cliente que participa no Projecto
Location	Text(100)	Localização onde será realizado a parte do Projecto relacionado com o Cliente da instancia

Tabela 31 - ProjectClient

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador de ProjectResponsible
ProjectClient	ProjectClient Identifier	Identificador do ProjectClient
Responsible	Text(50)	Nome do responsável
Email	Email	Email do responsável
Phone	Phone	Numero de telemovel do responsável

Tabela 32 –ProjectResponsible

<b>Atributo</b>	<b>Tipo</b>	<b>Descrição</b>
Id	Identifier	Identificador do SpontaneousCurriculum
Name	Text(256)	Nome do candidato
Phone	Phone Number	Numero de telemovel do candidato
Email	Email	Email do candidato
Date	Date	Data de ocorrenento da candidatura

Tabela 33 - SpontaneousCurriculum



Atributo	Tipo	Descrição
SpontaneousCV	SpontaneousCurriculum Identifier	Identificador da candidatura
File	Binary Date	O ficheiro pdf da candidatura

Tabela 34 - SpontaneousCurriculumFile

Atributo	Tipo	Descrição
Id	Identifier	Identificador de StepState
Label	Text(50)	Designação do estado do passo

Tabela 35 – StepState

Atributo	Tipo	Descrição
Id	Identifier	Identificador de StepType
Label	Text(50)	Nome do passo
GeneralInfo	Text(250)	Informação geral do passo, utilizada para informar candidatos o que podem esperar
IsInterview	Boolean	Se o passo é uma entrevista ou não, note-se que entrevistas com clientes não contam, sendo que as mesmas ocorrem fora da aplicação

Tabela 36 - StepType

Atributo	Tipo	Descrição
Id	Identifier	Identificador do User
Name	Text(256)	Nome do utilizador
Username	Text(250)	Username para identificação
Email	Email(250)	Email do utilizador
MobilePhone	PhoneNumber(20)	Numero de Telefone do utilizador
Password	Text(256)	Password para autenticação

Tabela 37 – User

Atributo	Tipo	Descrição
Id	Identifier	Identificador da Technology
Label	Text(50)	Tecnologia representada pela entidade
ToolType	Integer	O tipo de ferramenta, 0 para Tecnologias 1 para IDEs e 2 para Frameworks

Tabela 38 – Tools

Atributo	Tipo	Descrição
Id	Identifier	Identificador da TypeLanguage
Label	Text(50)	Descrição do tipo de comunicação
Order	Integer	Garante que os tipos seguem a ordem correcta

Tabela 39 – TypeLanguage

Atributo	Tipo	Descrição
Id	Identifier	Identificador de Vacancies
Title	Text(50)	Nome do novo cargo da vaga para que a vaga foi aberta
NumberOfSteps	Integer	O numero de passos que compõem o processo de contratação
Requests	Text(250)	Requisitos para o novo cargo para que a vaga foi aberta
NumberOfPeople	Integer	Numero de pessoas para completar a vaga
Client	Text(50)	Quando a vaga é para um projecto de um cliente, Client representa tal cliente

Tabela 40 – Vacancy

Atributo	Tipo	Descrição
Id	Identifier	Identificador da VacanciesStep
Language	Language Identifier	Linguagem da vaga
Vacancy	Vacany Identifier	Vaga a que a language pertence

Tabela 41 - VacancyLanguages

Atributo	Tipo	Descrição
Id	Identifier	Identificador da VacanciesStep
Step	Integer	Numero do passo no processo
Vacancy	Vacany Identifier	Vaga a que o passo pertence
StepType	StepType Identifier	Tipo do passo
Form	Form Identifier	Formulário para entrevistas

Tabela 42 - VacancyStep

Atributo	Tipo	Descrição
Id	Identifier	Identificador da VacanciesStep
Tool	Tool Identifier	Ferramenta da vaga
Vacancy	Vacany Identifier	Vaga a que a ferramenta pertence

Tabela 43 –VacancyTools