Licenciatura em Engenharia Informática e de Computadores

Queuality

Sistema de Gestão de Filas de Espera

Joana Campos, A44792 a44792@alunos.isel.pt
Nuno Cardeal, A44863 a44863@alunos.isel.pt
Carolina Couto, A44871 a44871@alunos.isel.pt

Orientador: Paulo Pereira paulo.pereira@isel.pt

Junho 2021

Índice

Lista de Figuras

1 Introdução

Numa sociedade onde existe cada vez mais adesão a *multitasking*, estar preso numa fila de espera, parado sem fazer nada, pode ser considerado uma perda de tempo. Tempo esse que poderia ser usado para trabalhar ou para "ir beber um café". Principalmente nestes tempos de pandemia estar em contacto com outras pessoas deve ser evitado.

Para contornar esta situação, uma solução viável seria um sistema de gestão de filas de espera que, assumindo que o cliente se encontra próximo do local onde espera ser servido, o notifica quando estiver próximo da sua vez, evitando assim que o cliente tenha de estar presencialmente na fila, podendo ir fazer outras coisas enquanto aguarda, melhorando assim a sua "qualidade de vida".

Neste projeto pretende-se criar um sistema de gestão de filas de espera. O sistema incluirá a autenticação dos funcionários para que possam realizar a gestão das diferentes filas. Terá ainda de admitir a existência de pelo menos um administrador que estará encarregue de editar as filas de espera. Os clientes terão a possibilidade de verem a senha atual de todas as filas, o número de senhas à sua frente, retirar uma senha ou fazer marcação, e ainda, serão notificados quando a sua vez estiver próxima.

2 Caracterização da Solução

A solução deste projeto dividir-se-á em quatro partes, uma base de dados que irá guardar a informação de cada utilizador, uma *Web API*, uma aplicação *web* dirigida para os funcionários do serviço e uma aplicação móvel dirigida para os clientes.

A *Web API* divide-se em duas partes, a implementação das funcionalidades da aplicação e o contrato público que disponibiliza as mesmas para as aplicações clientes. Como é possível ver na Figura 1 a Web API será o elo de ligação entre as funcionalidades da nossa solução e a aplicação de cliente e web. A implementação das funcionalidades irá ainda comunicar com a base de dados.

A aplicação *web* será destinada aos funcionários, sejam eles os que realizam o atendimento aos clientes ou os que administram o sistema. Estes dois papéis são distintos uma vez que o primeiro estará encarregue de avançar na fila de espera, e o último terá permissões para alterar o sistema das mesmas.

Para os clientes será disponibilizada uma aplicação móvel que permitirá obter a senha, conhecer quantas senhas se encontram à sua frente, assim como receber notificações quando faltar um determinado número de senhas para a sua vez.

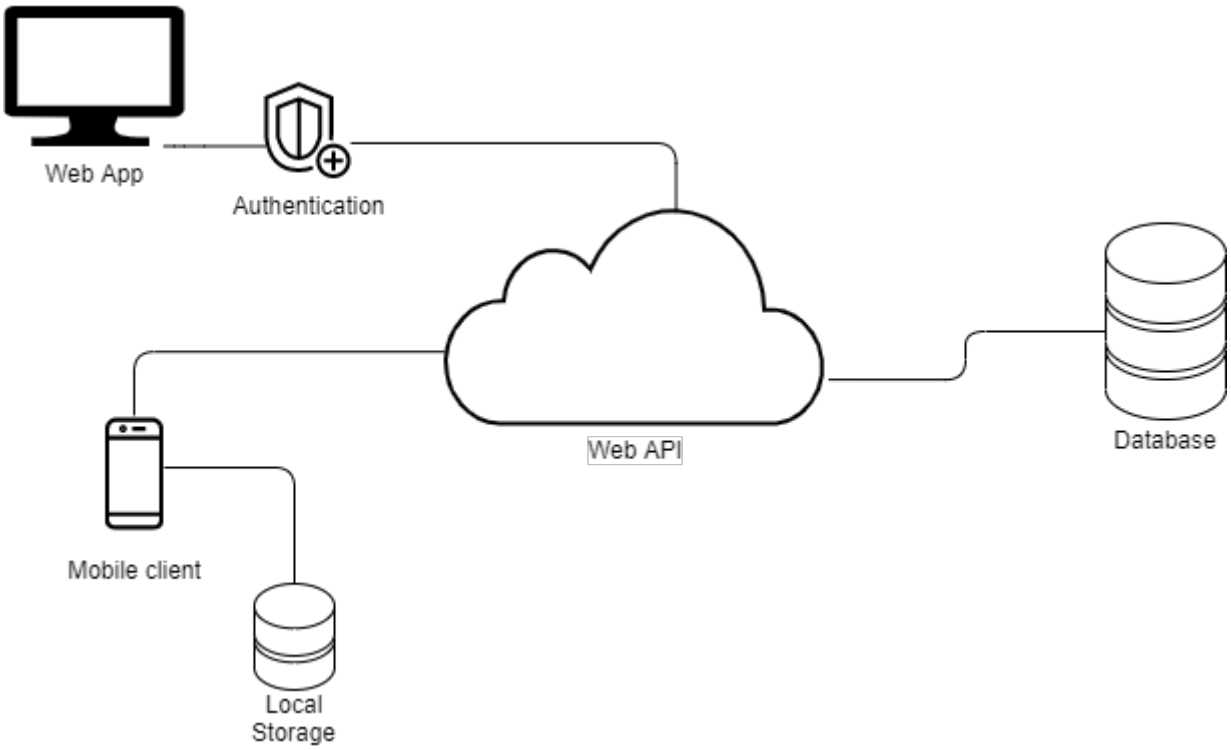


Figura 1: Diagrama do projeto Queuality.

3 Requisitos Funcionais

3.1 Base de Dados

Nesta secção é apresentada a proposta de solução para a base de dados utilizada. Esta estará responsável por armazenar a informação da Web API. No projeto presente a base de dados é não relacional, neste caso do tipo *Document Store* utilizando a tecnologia *MongoDB*[1]. A escolha deste tipo de base de dados deu-se ao facto de ser algo não muito explorado no curso e algo bastante usado na indústria. Base de dados do tipo documento tem também como vantagem guardar os dados de forma bastante similhante aos objetos usados na aplicação em questão, reduzindo assim a necessidade de haver tradução dos dados como estariam guardados numa base de dados relacional para os dados que a aplicação recebe. Na figura 2 encontra-se o modelo de dados utilizado neste projeto. Foi com base neste modelo que se criou as coleções e campos necessários para poder guardar a informação necessária.

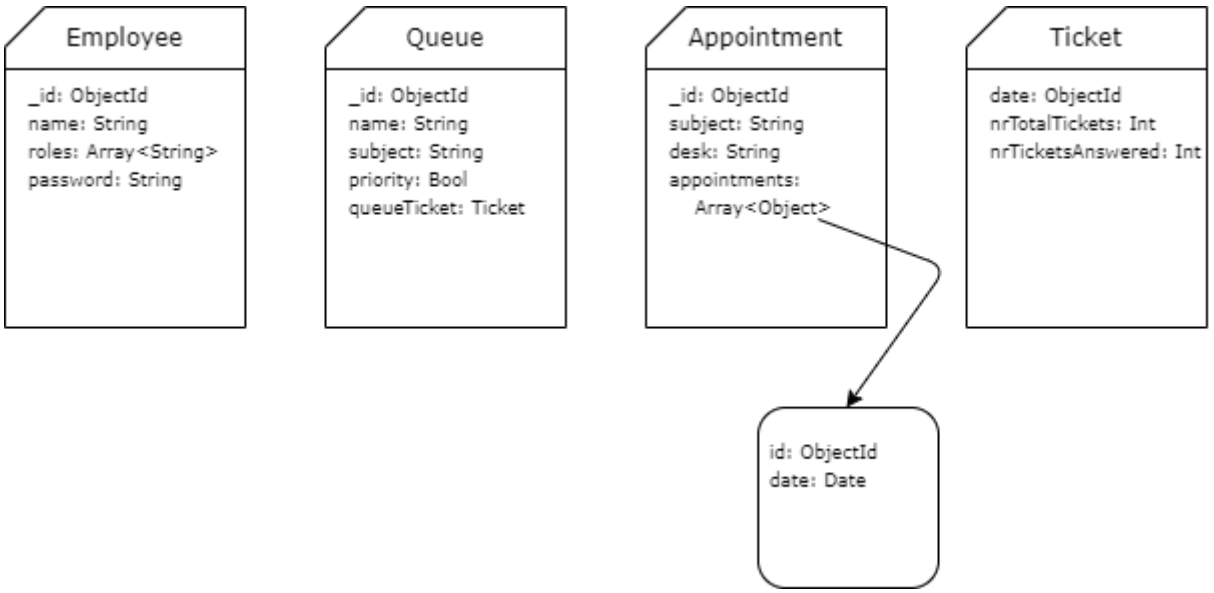


Figura 2: Modelo de Dados do Queuality.

Começou-se por definir as coleções necessários para lidar com um sistema de gestão de senhas. Estas entidades ainda estão abertas para melhoramentos ou adição de campos que se considerem necessários guardar. Assim, definiu-se a coleção *Employee* 2 ilustrada na figura que irá conter a informação necessária para identificar um funcionário assim como os seus roles como funcionário da empresa em questão. Seguidamente, foi defenida a coleção *Queue* ilustrada na figura 2 que irá conter o nome da fila, o assunto que irá ser tratado nessa fila e também irá ser guardada a informação das senhas da fila em questão. A informação sobre as senhas de cada fila irá ser guardado num objecto definido por *Ticket* em que irá conter o número de pessoas

que foram atendidas daquela fila, o número total de pessoas que tiraram senha naquela fila e finalmente a data que irá servir para reiniciar todos os dias a contagem das senhas a zero.

3.2 Aplicação *Web*

Nesta secção é apresentada a proposta de solução para a aplicação *web*. A aplicação web foi realizada com o propósito de ser utilizada pelos funcionários da empresa que usufruir do sistema em que irão ser englobadas diferentes ações a realizar, gerir as filas, gerir marcações realizadas e avançar nas senhas. A aplicação irá ter um serviço de autenticação *OpenId*[7] para que apenas os funcionários autenticados possam realizar ditas ações.

Apesar de ainda não decidido, possivelmente os diferentes *roles* que irão existir não poderão ter acesso a todos os links de navegação que irão estar disponíveis na aplicação *web*.

Na aplicação escolheu-se usar a tecnologia *React*[2] com *TypeScript*[3]. A escolha de *React*[2] para *frontend* sucedeu-se visto que é uma tecnologia cada vez mais usada na indústria, é bastante simples de compilar e fácil e apresenta qualidade na *user interface* o que é bastante importante para quem usufruir da aplicação. Para a linguagem estava-se indeciso entre *JavaScript*[4] ou *TypeScript*[3] acabando-se por escolher a última visto que usa tipos o que consegues corrigir erros que poderia facilmente acontecer em *JavaScript*[4] que é uma linguagem fracamente tipificada.

Neste momento todos os funcionários autenticados irão ter acesso à aplicação toda.

Na figura 3 está presente o diagrama de navegação utilizado para definir as diferentes páginas que a aplicação irá conter.

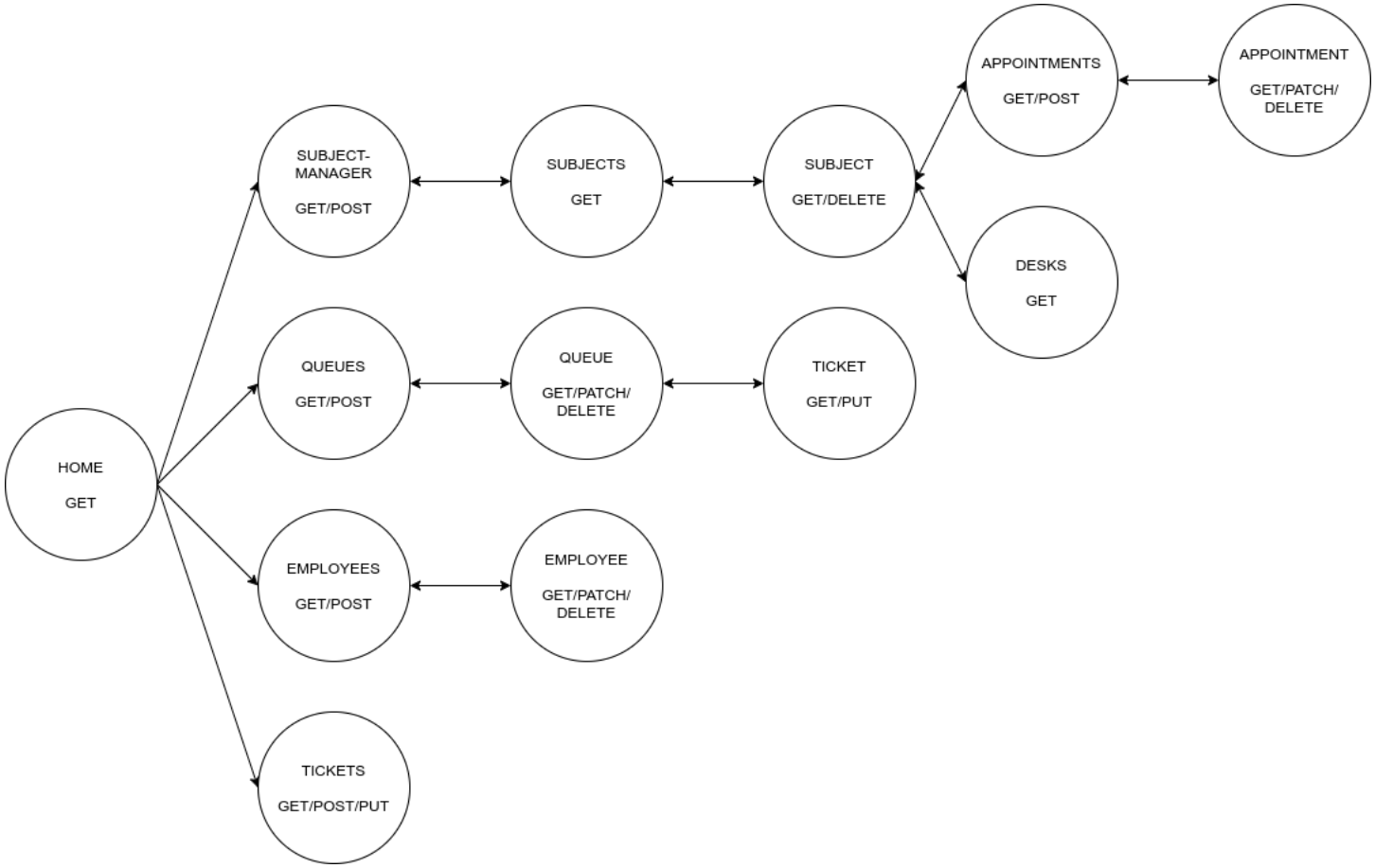


Figura 3: Diagrama de navegação da Aplicação *Web*

De seguida, desenhou-se a página de gestão das filas (Figura 4) em serão apresentados o nome das filas, o assunto das filas, a prioridade e onde será possível editar, apagar e adicionar filas. Estas ações apenas aos funcionários que o papel designado para tal.

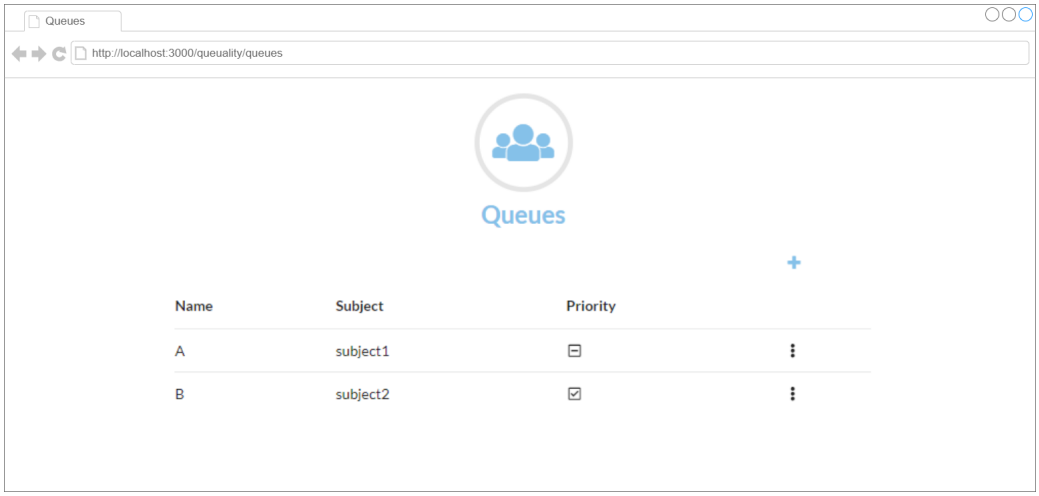


Figura 4: Mock da Aplicação *Web*: Gestão das filas

Já na página dos tickets é possível visualizar as próximas cinco senhas a serem chamadas assim como o funcionário ter um botão em que lhe será possível avançar na fila. Um *mock* desta página encontra-se na figura (Figura 5).

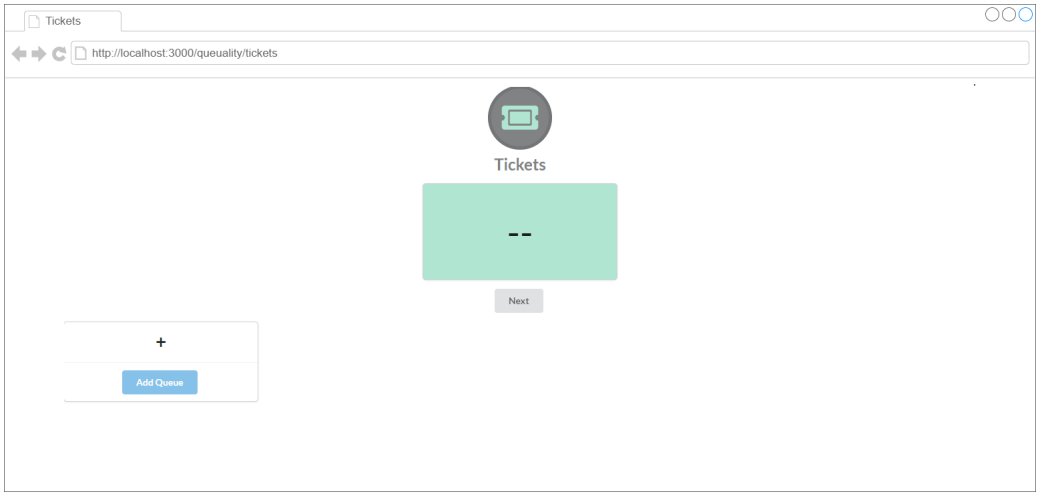


Figura 5: Mock da Aplicação *Web*: Gestão das senhas

3.3 *Web API*

Nesta secção é descrita a solução realizada neste projeto para a construção da *Web API*. A *API* foi escrita com recurso à linguagem *JavaScript*[4]. Esta linguagem é bastante utilizada no mundo de *web development* devido à sua simplicidade de aprendizagem, semelhança com outras liguagens, compactação do código e velocidade de execução. No entanto, devido à não existência de um ambiente virtual de execução nativo da linguagem *JavaScript*[4] é necessário ainda a utilização de um ambiente de execução que possa tirar o maior partido da linguagem e facilite a execução da *API* em *background* para tal foi utilizado o ambiente *Node.js*[8] para a execução da *API*. As próximas subsecções explicam em detalhe os aspetos mais importantes da *API*.

3.3.1 Express

O módulo Express[9] foi utilizado para a implementação da *Web API*, visto existir uma maior confortabilidade com este módulo devido à sua utilização na cadeira Programação na Internet. Este módulo é *open-source*, por isso desenvolvido pela comunidade que utiliza *Node.js*[8] como seu ambiente virtual de execução. Esta característica torna o módulo bastante completo pois quando a comunidade sente a necessidade de alguma funcionalidade esta é implementada muito mais rapidamente, tornando também o modulo bastante versátil.

As principais vantagens da utilização deste módulo para além das descritas acima são ainda a facilidade de incorporação com outros módulos, a criação automática de um servidor HTTP para as suas aplicações, livre-arbítrio nas diferentes decisões de arquitetura da aplicação e o suporte integrado para a criação de uma *REST API*.

3.3.2 Acesso à Base de Dados

Devido à escolha de MongoDB[1] para o alojamento da base de dados como referido da secção 3.1 optou-se também por utilizar o módulo desenvolvido pela mesma empresa, visto este apresentar um suporte e documentação mais completos para todas as funcionalidades disponibilizadas. Para a implementação deste acesso foi criado um módulo que realiza a comunicação com a base de dados e outros quatro módulos que servem como repositórios para cada coleção. É através destes repositórios que são realizadas todas as verificações necessárias para as operações realizadas na *API*. Existe ainda os módulos de serviço onde são implementadas todas as funcionalidades da *API* que necessitem de acesso à base de dados.

3.3.3 Autenticação

Tendo em atenção que o Queuality pretende ser um projeto que possa ser adotado por qualquer empresa optou-se por utilizar um sistema de autenticação aberto seguindo o protocolo *OpenID*[7] em vez de ser criado um sistema de autenticação fechado. Desta forma qualquer empresa que adote o projeto pode facilmente utilizar o seu sistema de autenticação para permitir

aos seus trabalhadores acesso ao sistema de gestão de base de dados, acrescentando apenas algumas variáveis de ambiente ao seu servidor.

Para efeitos de teste e demonstração foi escolhido o sistema de autenticação da Google, visto o grupo já ter tido experiência com este sistema na cadeira Segurança Informática.

3.4 Aplicação Móvel

A aplicação móvel deste sistema foi desenvolvida em *TypeScript*[3] juntamente com *React*[2] com o auxílio da *framework Ionic*[5]. Esta foi escolhida sobre outras opções uma vez que é possível desenvolver uma aplicação *Android* e uma aplicação *iOS* com o mesmo código. Outro ponto que nos fez escolher *Ionic*[5] em vez de *React Native*[6] foi o facto de *Ionic*[5] pedir apenas conceitos básicos de *React*[2] enquanto que *React Native*[6] pede conhecimentos avançados sobre o mesmo.

A aplicação móvel destina-se exclusivamente aos clientes da empresa utilizadora deste serviço. A aplicação desenvolvida divide-se em três módulos principais: O módulo de filas, o módulo de senhas e o módulo de marcações, sendo estes separados por abas. A figura 6 ilustra as funcionalidades presentes na aplicação móvel.

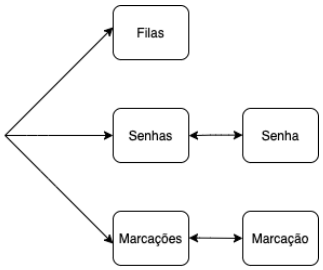
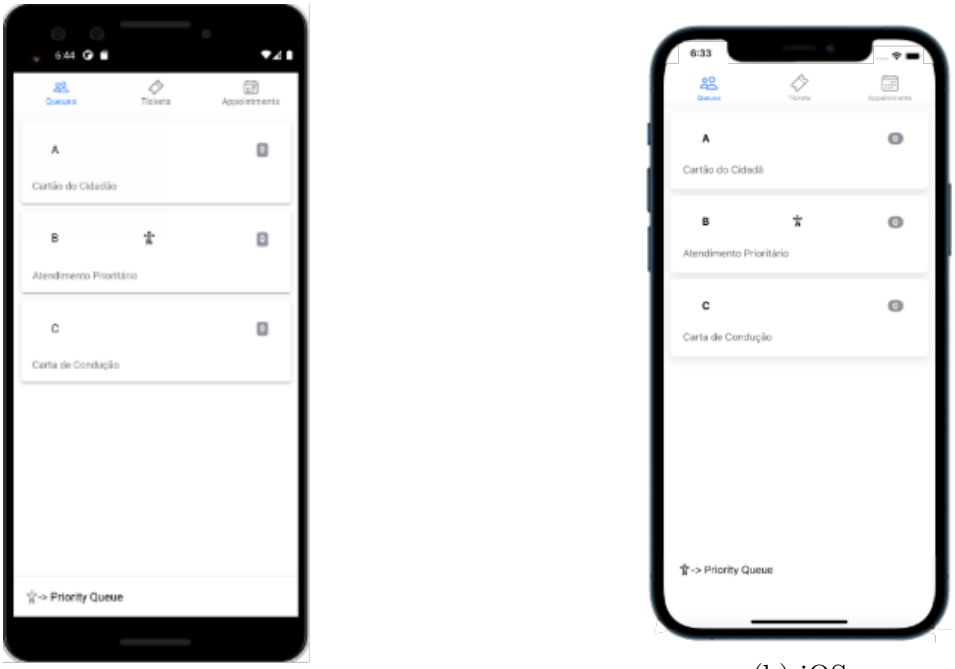


Figura 6: Diagrama de navegação da Aplicação Móvel

A primeira página a ser desenvolvida foi a página do módulo das filas. O principal objetivo desta página é apresentar ao cliente todas as filas existentes no sistema (nome, assunto e se é uma fila prioritária ou não), em que número vai cada fila e, ao carregar numa fila, dar a possbilidade ao cliente de tirar uma senha para a mesma. Nas figuras 7 é possível observar *mocks* da página de filas tanto para *Android* como para *iOS*.

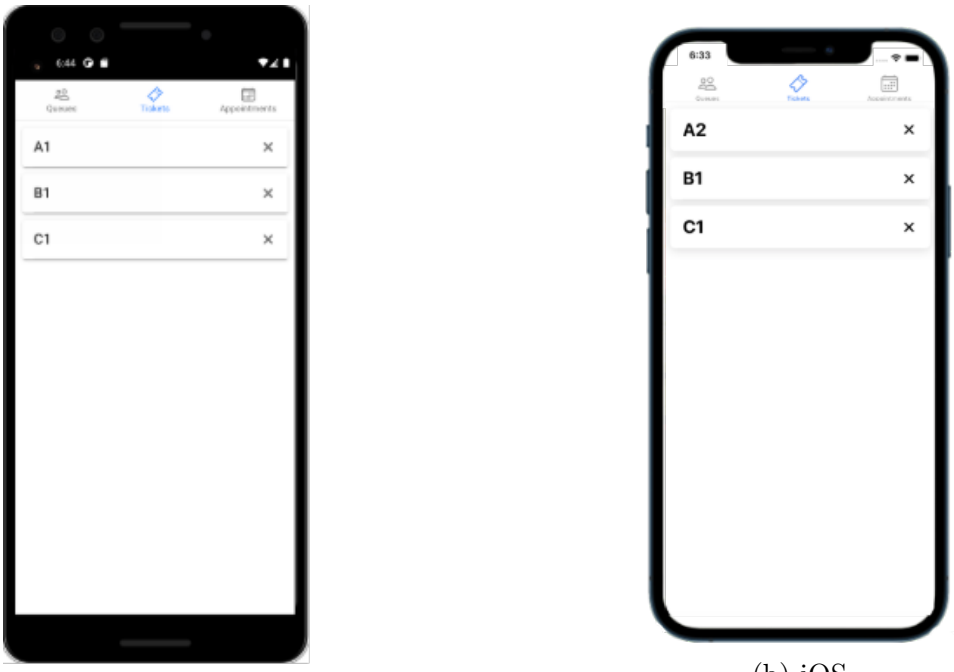


(a) Android

(b) iOS

Figura 7: Mock da página das filas

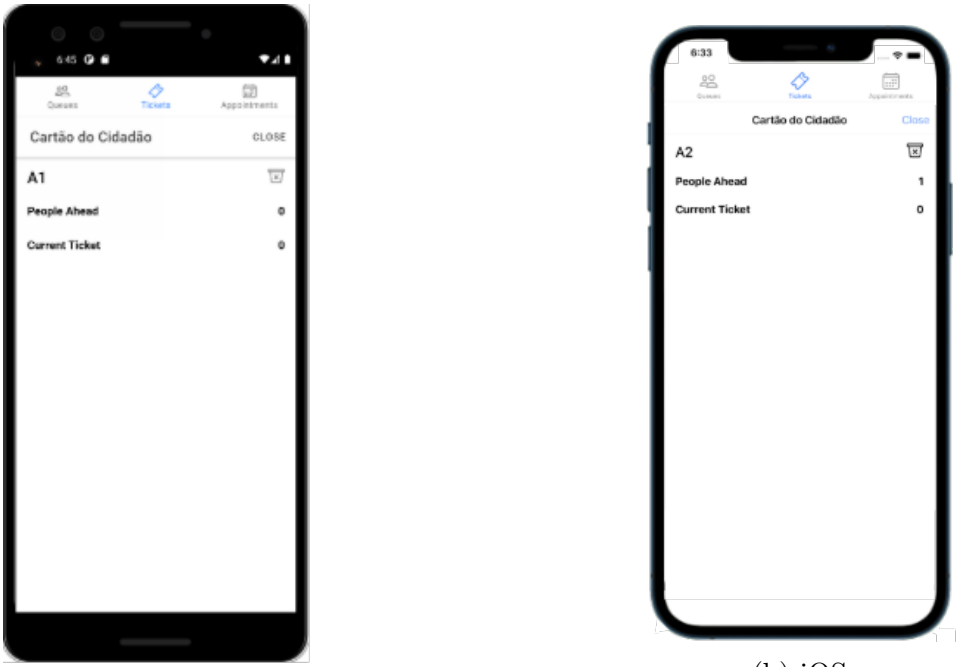
Na página das senhas é apresentada ao cliente uma lista com as suas senhas (com um máximo de uma senha por fila) e, carregando na senha, o cliente será redirecionado para outra página com os detalhes da mesma, entre as quais, o seu número, o assunto da fila, quantas pessoas estão à sua frente e em que número vai a fila para a qual tirou senha. Terá também a possbilidade de cancelar a senha em ambas as páginas deste módulo.



(a) Android

(b) iOS

Figura 8: Mock da página das senhas



(a) Android

(b) iOS

Figura 9: Mock da página dos detalhes de uma senha

4 Requisitos Não Funcionais

Para a solução deste projeto os recursos serão alocados numa base de dados não relacional, para dar oportunidade ao aprofundamento dos conhecimentos sobre a mesma. Esta base de dados irá guardar os utilizadores autenticados e as suas respetivas funções. Cada utilizador anónimo terá uma base de dados local que guardará marcações e/ou senhas juntamente com a informação adicional que for necessária no decorrer do projeto. Tenciona-se realizar *deployment* na *cloud* de modo que o projeto tenha oportunidade de ser distribuído a potenciais interessados. Terá a possibilidade de receber relatórios de análise de forma a ser possível monitorizar os potenciais problemas que possam vir a ocorrer no sistema.

5 Estado atual do Projeto

Neste momento, a componente servidora e a base de dados encontram-se implementadas, no entanto sempre prontas para melhoramentos. Em relação ao *frontend*, na aplicação móvel, a atividade das filas encontra-se completa e a de o utilizador poder tirar uma senha também, ficando a faltar na sua totalidade as atividade das marcações. Em relação a aplicação web falta implementar a parte da autenticação, a página da informação dos funcionários e a das marcações. Encontra-se completamente implementada a página das filas e a página das senhas encontra-se parcialmente implementada faltando apenas tomar algumas decisões finais.

Referências

[1] MongoDB,
`https://www.mongodb.com/`. Consultado a 15/04/2021

[2] *React*,
`https://reactjs.org/`. Consultado a 20/04/2021

[3] TypeScript,
`https://www.typescriptlang.org/`. Consultado a 20/04/2021

[4] JavaScript,
`https://www.typescriptlang.org/`. Consultado a 12/04/2021

[5] Ionic,
`https://ionicframework.com/`. Consultado a 06/06/2021

[6] React Native
`https://reactnative.dev/`. Consultado a 12/04/2021

[7] OpenID,
`https://openid.net/`. Consultado a 23/05/2021

[8] NodeJs,
`https://nodejs.org/`. Consultado a 02/04/2021

[9] Express,
`https://expressjs.com/`. Consultado a 02/04/2021

A Documentação das Rotas da API

O URL base desta API é /api. As próximas secções irão descrever cada endpoint da API.

A.1 Módulo de Filas

O Módulo de Filas contem 3 rotas distintas às quais são feitos 5 pedidos diferentes. Nos próximos pontos iremos descrever cada um desses pedidos

A.1.1 GET /queues

Este pedido retorna a lista das filas existentes em sistema. Tanto o administrador como os clientes deste sistema tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:


```
{
    "class": ["Queues"],
    "properties": [
        {
            "_id": "60b3e2830e97c070513f24a6",
            "name": "A",
            "priority": false,
            "queueTicket": {
                "nrTicketsAnswered": 0,
                "nrTotalTickets": 1,
                "date": "Sun Jun 13 2021"
            },
            "subject": "Cartão do Cidadão"
        },
        {
            "_id": "60b3e2f20e97c070513f2512",
            "name": "B",
            "priority": true,
            "queueTicket": {
                "nrTicketsAnswered": 0,
```

```
            "nrTotalTickets": 1,
            "date": "Sun Jun 13 2021"
        },
        "subject": "Atendimento Prioritário"
    },
    {
        "_id": "60bbf1490fa7216722109854",
        "name": "C",
        "priority": false,
        "queueTicket": {
            "nrTicketsAnswered": 0,
            "nrTotalTickets": 1,
            "date": "Sun Jun 13 2021"
        },
        "subject": "Carta de Condução"
    }
],
"entities": [
    {
        "rel": ["/rel/queue"],
        "properties": {
            "_id": "60b3e2830e97c070513f24a6",
            "name": "A",
            "priority": false,
            "queueTicket": {
                "nrTicketsAnswered": 0,
                "nrTotalTickets": 1,
                "date": "Sun Jun 13 2021"
            },
            "subject": "Cartão do Cidadão"
        },
        "class": ["Queue"],
        "links": [
            {
                "rel": ["self"],
                "href": "/api/queues/60b3e2830e97c070513f24a6"
            },
            {
                "rel": ["/rel/current-ticket"],
                "href":"/api/queues/60b3e2830e97c070513f24a6/current-ticket"
```

```
            }
        ],
        "actions": [
            {
                "name": "update-queue",
                "title": "Update a Queue",
                "method": "PATCH",
                "href": "/api/queues/60b3e2830e97c070513f24a6",
                "type": "application/vnd.siren+json",
                "fields": [
                    {
                        "name": "priority",
                        "type": "boolean"
                    },
                    {
                        "name": "subject",
                        "type": "text"
                    }
                ]
            },
            {
                "name": "delete-queue",
                "title": "Delete a Queue",
                "method": "DELETE",
                "href": "/api/queues/60b3e2830e97c070513f24a6",
                "type": "application/vnd.siren+json"
            }
        ],
        "title": "Get Queue"
    },
    {
        "rel": ["/rel/queue"],
        "properties": {
            "_id": "60b3e2f20e97c070513f2512",
            "name": "B",
            "priority": true,
            "queueTicket": {
                "nrTicketsAnswered": 0,
                "nrTotalTickets": 1,
                "date": "Sun Jun 13 2021"
```

```
    },
    "subject": "Atendimento Prioritário"
},
"class": ["Queue"],
"links": [
    {
        "rel": ["self"],
        "href": "/api/queues/60b3e2f20e97c070513f2512"
    },
    {
        "rel": ["/rel/current-ticket"],
        "href":"/api/queues/60b3e2f20e97c070513f2512/current-ticket"
    }
],
"actions": [
    {
        "name": "update-queue",
        "title": "Update a Queue",
        "method": "PATCH",
        "href": "/api/queues/60b3e2f20e97c070513f2512",
        "type": "application/vnd.siren+json",
        "fields": [
            {
                "name": "priority",
                "type": "boolean"
            },
            {
                "name": "subject",
                "type": "text"
            }
        ]
    },
    {
        "name": "delete-queue",
        "title": "Delete a Queue",
        "method": "DELETE",
        "href": "/api/queues/60b3e2f20e97c070513f2512",
        "type": "application/vnd.siren+json"
    }
],
```

```
    "title": "Get Queue"
},
{
    "rel": ["/rel/queue"],
    "properties": {
        "_id": "60bbf1490fa7216722109854",
        "name": "C",
        "priority": false,
        "queueTicket": {
            "nrTicketsAnswered": 0,
            "nrTotalTickets": 1,
            "date": "Sun Jun 13 2021"
        },
        "subject": "Carta de Condução"
    },
    "class": ["Queue"],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/queues/60bbf1490fa7216722109854"
        },
        {
            "rel": ["/rel/current-ticket"],
            "href":"/api/queues/60bbf1490fa7216722109854/current-ticket"
        }
    ],
    "actions": [
        {
            "name": "update-queue",
            "title": "Update a Queue",
            "method": "PATCH",
            "href": "/api/queues/60b3e2f20e97c070513f2512",
            "type": "application/vnd.siren+json",
            "fields": [
                {
                    "name": "priority",
                    "type": "boolean"
                },
                {
                    "name": "subject",
```

```
                        "type": "text"
                    }
                ]
            },
            {
                "name": "delete-queue",
                "title": "Delete a Queue",
                "method": "DELETE",
                "href": "/api/queues/60b3e2f20e97c070513f2512",
                "type": "application/vnd.siren+json"
            }
        ],
        "title": "Get Queue"
    }
],
"links": [
    {
        "rel": ["self"],
        "href": "/api/queues"
    }
],
"actions": [
    {
        "name": "add-queue",
        "title": "Add a Queue",
        "method": "POST",
        "href": "/api/queues",
        "type": "application/vnd.siren+json",
        "fields": [
            {
                "name": "name",
                "type": "text"
            },
            {
                "name": "priority",
                "type": "boolean"
            },
            {
                "name": "subject",
                "type": "text"
```

```
                }
            ]
        }
    ]
}
```

A.1.2 POST /queues

Este pedido adiciona uma fila ao sistema. Apenas o administrador tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json
Body:


```
{
    "name": "C",
    "priority": false,
    "subject": "Carta de Condução"
}
```


Resposta
Status Code : 201
Headers : content-type: aplication/json
Body:


```
 {
    "class": ["Queue"],
    "properties": {
        "_id": "60c655d50fa72167221187b0",
        "name": "C",
        "priority": false,
        "queueTicket": {
            "nrTotalTickets": 0,
            "nrTicketsAnswered": 0,
            "date": "Sun Jun 13 2021"
        },
        "subject": "Carta de Condução"
    },
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/queues"
        }
    ],
    "actions": []
}
```

A.1.3 PATCH /queues/:queueId

Este pedido edita o assunto e/ou a prioridade de uma fila do sistema. Apenas o administrador tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json
Path Parameter : queueId - o identificador da fila
Body:

```
{
    "subject": "finanças",
    "priority": true
}
```

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:

```
{
    "class": ["Queue"],
    "properties": {
        "id": "60c655d50fa72167221187b0",
        "name": "C",
        "priority": true,
        "queueTicket": {
            "nrTotalTickets": 0,
            "nrTicketsAnswered": 0,
            "date": "Sun Jun 13 2021"
        },
        "subject": "finanças"
    },
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/queues/60c655d50fa72167221187b0"
        }
    ],
    "actions": []
}
```

A.1.4 DELETE /queues/:queueId

Este pedido apaga uma fila do sistema. Apenas o administrador tem acesso a este pedido.

Pedido
Headers : content-type: aplication/json

Resposta
Status Code : 200
Headers : content-type: aplication/json
Body:


```
{
    "class": ["Queue"],
    "properties": {},
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/queues/60c655d50fa72167221187b0"
        }
    ],
    "actions": []
}
```

A.1.5 PUT /queues/:queueId/current-ticket

Este pedido incrementa o número de senhas atedidas de uma fila do sistema. Tanto o administrador como o funcionário têm acesso a este pedido.

Pedido
Headers: content-type: aplication/json

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body :


```
{
    "class": ["Current Ticket"],
    "properties": "C1",
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/queues/60c655d50fa72167221187b0/current-ticket"
        }
    ],
    "actions": []
}
```

A.2 Módulo de Senhas

O Módulo de Senhas contem 2 rotas distintas às quais são feitos 4 pedidos diferentes. Nos próximos pontos iremos descrever cada um desses pedidos

A.2.1 GET /tickets/waiting-tickets

Este pedido retorna o número de senhas em espera. Tanto o administrador como os funcionários deste sistema tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:

```
{
    "class": ["Tickets"],
    "properties": 1,
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/tickets/waiting-tickets"
        }
    ],
    "actions": []
}
```

A.2.2 GET /tickets

Este pedido retorna a lista de senhas em espera. Tanto o administrador como os funcionários deste sistema tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:


```
{
    "class": ["Tickets"],
    "properties": [
        {
            "_id": "60b3e2830e97c070513f24a6",
            "ticketNumber": "A2",
            "subject": "Cartão do Cidadão",
            "priority": false
        },
    ],
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/tickets"
        },
        {
            "rel": ["ticket-queues"],
            "href": "/api/queues"
        }
    ],
    "actions": [
        {
            "name": "add-ticket",
            "title": "Add a Ticket",
            "method": "POST",
            "href": "/api/tickets",
            "type": "application/vnd.siren+json",
            "fields": [
```

```
                {
                    "name": "queueId",
                    "type": "text"
                }
            ]
        },
        {
            "name": "delete-ticket",
            "title": "Delete a Ticket",
            "method": "PUT",
            "href": "/api/tickets",
            "type": "application/vnd.siren+json"
        }
    ]
}
```

A.2.3 POST /tickets

Este pedido adiciona uma senha à lista de espera. Apenas os clientes deste sistema têm acesso a este pedido.

Pedido
Headers: content-type: aplication/json
Body:

```
 {
     "queueId": "60b3e2830e97c070513f24a6"
}
```

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:

```
{
    "class": ["Tickets"],
    "properties": "A2",
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/tickets"
        }
    ],
    "actions": []
}
```

A.2.4 PUT /tickets

Este pedido remove uma senha da lista de espera. Apenas os clientes deste sistema têm acesso a este pedido.

Pedido
Headers: content-type: aplication/json
Body:

```
{
    "ticket": "A1"
}
```

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body :

```
{
    "class": ["Tickets"],
    "properties": {},
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/tickets"
        }
    ],
    "actions": []
}
```

A.3 Módulo de Funcionários

O Módulo de Funcionários contem 2 rotas distintas às quais são feitos 5 pedidos diferentes. Nos próximos pontos iremos descrever cada um desses pedidos

A.3.1 GET /employees

Este pedido retorna a lista dos funcionários. Apenas o administrador tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:


```
{
    "class": ["Employees"],
    "properties": [
        {
            "_id": "60941a400e97c070512c05cd",
            "name": "Nome",
            "password":"$2b$10$Js0Kck5FqJjz0curcH36x.CUIhTzLM280/S/yL.2lRrJfOS4rXG/i",
            "roles":["admin"]
        },
        {
            "_id": "609421240e97c070512c069a",
            "name": "Nome2",
            "password":"$2b$10$EoK66j5LuYmUjSh4sQNb5OOS/uroG12/UXv6mWiR6rTR9hRyxSDLO",
            "roles": ["func"]
        }
    ],
    "entities": [
        {
            "rel": ["/rel/employee"],
            "properties": {},
            "class": ["Employee"],
            "links": [
                {
                    "rel": ["self"],
```

```
                "href": "/api/employees/60941a400e97c070512c05cd"
            }
        ],
        "actions": [],
        "title": ""
    },
    {
        "rel": ["/rel/employee"],
        "properties": {},
        "class": ["Employee"],
        "links": [
            {
                "rel": ["self"],
                "href": "/api/queues/609421240e97c070512c069a"
            }
        ],
        "actions": [],
        "title": ""
    }
],
"links": [
    {
        "rel": ["self"],
        "href": "/api/employees"
    }
],
"actions": [
    {
        "name": "add-employee",
        "title": "Add a Employee",
        "method": "POST",
        "href": "/api/employees",
        "type": "application/vnd.siren+json",
        "fields": [
            {
                "name": "name",
                "type": "text"
            },
            {
                "name": "roles",
```

```
                    "type": "object"
                }
            ]
        }
    ]
}
```

A.3.2 POST /employees

Este pedido retorna a adiciona um funcionário. Apenas o administrador tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json
Body:

```
{
    "name": "Nome3",
    "password" : "password",
    "roles": ["admin"]
}
```

Resposta
Status Code: 201
Headers: content-type: aplication/json
Body:

```
{
    "class": ["Employees"],
    "properties": {
        "_id": "60c671090fa72167221189a4",
        "name": "Nome3",
        "roles": ["admin"]
    },
    "entities": [],
    "links": [
        {
            "rel": [ "self"],
            "href": "/api/employees"
        },
        {
            "rel": ["/rel/employee"],
            "href": "/api/employees/60c671090fa72167221189a4"
        }
    ],
    "actions": []
}
```

A.3.3 GET /employees/:employeeId

Este pedido retorna o funcionário com o identificador especificado no parametro do url. Apenas o administrador tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json
Path Parameter: employeeId- o identifacor do funcionário

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:


```
{
    "class": ["Employee"],
    "properties": {
        "_id": "609421240e97c070512c069a",
        "name": "Nome",
        "roles": ["func"]
    },
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/employees/609421240e97c070512c069a"
        }
    ],
    "actions": [
        {
            "name": "update-employee",
            "title": "Update an Employee",
            "method": "PATCH",
            "href": "/api/employees/609421240e97c070512c069a",
            "type": "application/vnd.siren+json",
            "fields": [
                {
                    "name": "roles",
                    "type": "object"
                }
            ]
        },
```

```
        {
            "name": "delete-employee",
            "title": "Delete an Employee",
            "method": "DELETE",
            "href": "/api/employees/undefined",
            "type": "application/vnd.siren+json"
        }
    ]
}
```

A.3.4 PATCH /employees/:employeeId

Este pedido edita a password e/ou as funções do funcionário com o identificador especificado no parametro do url. Apenas o administrador tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json
Path Parameter: employeeId- o identifacor do funcionário
Body:

```
{
    "oldPassword": "password",
    "newPassword" : "pass",
    "roles": ["func"]
}
```

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:

```
{
    "class": ["Employee"],
    "properties": {},
    "entities": [],
    "links": [
        {
            "rel": ["self"],
            "href": "/api/employees/609421240e97c070512c069a"
        },
        {
            "rel": [ "/rel/employees"],
            "href": "/api/employees"
        }
    ],
    "actions": []
}
```

A.3.5 DELETE /employees/:employeeId

Este pedido apaga o funcionário com o identificador especificado no parametro do url. Apenas o administrador tem acesso a este pedido.

Pedido
Headers: content-type: aplication/json
Path Parameter: employeeId- o identifacor do funcionário

Resposta
Status Code: 200
Headers: content-type: aplication/json
Body:

```
{
    "class": ["Employee"],
    "properties": {},
    "entities": [],
    "links": [
        {
            "rel": [ "/rel/employees"],
            "href": "/api/employees"
        }
    ],
    "actions": []
}
```