Universidade de Évora - Escola de Ciências e Tecnologia

Mestrado em Engenharia Informática

Dissertação

Aplicação Móvel para o Sistema de Gestão de Filas Q.track

Tânia Cristiana Martins Bogalho

Orientador(es) | Teresa Cristina de Freitas Goncalves
Vítor Manuel Beires Pinto Nogueira

Évora 2021

Mestrado em Engenharia Informática

Dissertação

Aplicação Móvel para o Sistema de Gestão de Filas Q.track

Tânia Cristiana Martins Bogalho

Orientador(es) | Teresa Cristina de Freitas Goncalves
Vítor Manuel Beires Pinto Nogueira

Évora 2021

A dissertação foi objeto de apreciação e discussão pública pelo seguinte júri nomeado pelo Diretor da Escola de Ciências e Tecnologia:

Presidente	|	Irene Pimenta Rodrigues (Universidade de Évora)
Vogais	|	José Saias (Universidade de Évora) (Arguente)
		Vitor Beires Nogueira (Universidade de Évora) (Orientador)

Évora 2021

Évora 2021

Resumo

Aplicação Móvel para o Sistema de Gestão de Filas Q.track

Hoje em dia os *smartphones* provocam grandes alterações na vida das pessoas. Cada vez mais queremos ser mais produtivos e não desperdiçar tempo no dia-a-dia. Pode-se dizer até que cada vez mais somos mais dependentes dos telemóveis para qualquer atividade diária.

Neste sentido, inicialmente fez-se um estudo sobre a teoria das filas de espera e sobre os diversos modelos existentes. Foi feito um estudo do que são os sistemas de gestão de filas e que exemplos existem atualmente, comparando-os e verificando as lacunas existentes entre eles.

Deste modo, o principal objetivo deste estudo é desenvolver uma aplicação para *smartphone* que é um complemento a um sistema de gestão de filas que tinha esta lacuna, relativamente a outros sistemas de gestão de filas existentes, e que servirá essencialmente, para o utilizador ter acesso ao estado de atendimento dos serviços de uma determinada entidade ou até retirar uma senha para um serviço. A sua caracterização foi feita recorrendo a dados de teste, tendo em conta a hora de chegada a um determinado serviço, a hora de atendimento da senha e a duração do atendimento. Para a realização deste trabalho, recorreu-se a uma base de dados relacional já com dados reais de senhas retiradas e atendimentos efetuados.

Palavras-chave: Aplicações Móveis, Sistemas de gestão de filas, Gestão de filas, filas de espera, serviços.

Évora 2021

Abstract

Mobile Application for the Q.track Queue Management System

Nowadays smartphones cause major changes in people's lives. More and more we want to be more productive and not waste time on a daily basis. It can even be said that we are increasingly dependent on mobile phones for any daily activity.

In this sense, initially a study was made on the theory of queues and on the various existing models. A study of what queue management systems are and what examples exist today, comparing them and checking the problems between them.

Thus, the main objective of this study is to develop a smartphone application that is a complement to a queue management system that had this when compared to other existing queue management systems, and that will essentially serve for the user to have access to the state of attendance of the services of a certain entity or even get a ticket for a service. Its characterization was made using test data, taking into account the time of arrival at a given service, the time for answering the password, the duration of the service. In order to carry out this work, a relational database was already used with real data of withdrawn passwords and calls made.

Évora 2021

Keywords: Mobile Applications, Queue Management Systems, Queue Management, Queuing, Services.

Agradecimentos

A frequência do Mestrado em Engenharia Informática, ministrada pela Universidade de Évora, foi um marco no meu desenvolvimento pessoal e profissional. Foi uma trajetória repleta de inúmeros desafios e percalços pelo caminho, mas foram situações ultrapassadas com a ajuda de um conjunto de pessoas.

Durante este período foram muitos os momentos de aprendizagem e muitas as pessoas com quem pude partilhar conhecimentos e sentires. Quero agradecer à Professora Teresa Gonçalves e professor Vitor Nogueira e todos os professores do Mestrado, por toda a disponibilidade, orientações e saberes que me tentaram transmitir.

Um agradecimento especial aos meus familiares e amigos por todo o apoio, carinho e amizade com que me brindaram neste percurso e a todas as pessoas que contribuíram direta ou indiretamente para a concretização deste trabalho.

A todos, até mesmo os que não mencionei, um muito obrigado.

Évora 2021

Índice Geral

Índice de Tabelas

Índice de Figuras

Évora 2021

1. Introdução

1.1 Motivação

Nos dias de hoje em que o tempo é escasso para tudo, existe, cada vez mais, a necessidade de rentabilizarmos os nossos dias. Para nos ajudar a sermos mais eficazes a organizar o nosso dia e a não perdermos muito tempo em diversos serviços, temos à nossa disposição aplicações para nos ajudar, por exemplo, quando vamos a uma farmácia, a um hospital, Segurança Social, ou até mesmo numa ida a um supermercado. Em todos estes locais temos alguma, senão, muita afluência de pessoas, e o fato de estarmos à espera de sermos atendidos pode ser bastante complicado de gerir.

A formação de filas de espera é uma situação recorrente em locais com afluência de pessoas em grande número. Ocorre frequentemente, por exemplo, em repartições públicas, bancos, aeroportos, supermercados, farmácias, etc.

Para melhorar o serviço de atendimento prestado, foram desenvolvidos sistemas de gestão de filas. Estes sistemas fazem toda a gestão informática da chegada do cliente / utente a uma organização, até o seu atendimento. Desta forma, torna-se possível ter acesso a estatísticas de tempos de atendimento, tempos de espera, entre outros.

Em Portugal, existem diversos sistemas de gestão de filas. Alguns exemplos são: a aplicação Q.Track desenvolvido pela empresa Logicpulse, a aplicação QSystems da empresa Altronix e a aplicação SIGA desenvolvido pela empresa Instituto de Informática.

A motivação principal desta dissertação, surge na necessidade de adicionar novas funcionalidades ao sistema de filas escolhido, o Q.track, nomeadamente, existir uma aplicação móvel que permita ao utilizador, ter uma visão do sistema numa determinada entidade, ou até mesmo retirar uma senha para um determinado serviço.

1.2 Objetivos

Neste trabalho de dissertação pretende-se fazer um estudo dos sistemas de gestão de filas, exemplificar que sistemas de gestão de filas existem, como um sistema de filas funciona, e como um sistema de filas se pode aplicar numa aplicação móvel.

De acordo com o tema da tese “Aplicação Móvel para o Sistema de Gestão de Filas Q.track” o objetivo desta dissertação consistiu em criar uma ferramenta de fácil utilização e que permitisse a um utilizador ter uma visão geral do estado de atendimento numa determinada entidade, ou até ter acesso à possibilidade de retirar uma senha.

1.3 Contribuições

As principais contribuições deste trabalho foram:

- elaborar o estado da arte, onde é possível saber mais acerca de um sistema de gestão de filas e seus exemplos.
- desenvolver uma aplicação para sistemas operativos móveis que permite a um cliente / utente ter acesso a um conjunto de funcionalidades, tais como, verificar o estado de um serviço (senha atual, número de senhas em espera), ou até retirar uma senha se um serviço à sua escolha.

1.4 Organização

A presente dissertação encontra-se subdividida em diferentes capítulos. No capítulo I é feita uma breve contextualização ao tema em estudo, referindo quais os objetivos e a principal motivação. No capítulo II e capítulo III introduz-se a teoria de filas de espera e o sistema de gestão de filas Qtrack, respetivamente. No capítulo IV é apresentado o trabalho desenvolvido. No capítulo V são apresentadas as considerações finais, bem como possível trabalho futuro. No capítulo VI é apresentada a bibliografia utilizada para apoio a esta dissertação.

2. Estado de Arte

Neste capítulo é descrita a teoria das filas de espera, o funcionamento dos sistemas de filas e são dados exemplos de sistemas de gestão de filas na atualidade.

2.1 Teoria das filas de espera

"A teoria das filas é um ramo da probabilidade que estuda a formação de filas, através de análises matemáticas precisas e propriedades mensuráveis das filas [1]." Não é mais do que um ramo da probabilidade que é constituído por um conjunto de modelos que investiga, por exemplo, quais as razões possíveis que fizeram com que as pessoas ficassem à espera de serem atendidas numa loja de cidadão ou farmácia.

A maioria dos modelos de filas de espera que existem baseia-se no processo de chegada de um novo cliente / utente (nascimento) e à partida do mesmo (morte) (Markoviano) [2].

O modelo mais utilizado atualmente é o modelo de *Kendall* GI/G/m, em que GI exprime os tempos de chegada, G os tempos de serviço, como variáveis aleatórias independentes, e m representa o número total de servidores existentes para fornecer o serviço procurado pelos utentes. A este modelo podem ainda ser adicionados outros fatores tais como, por exemplo: a capacidade do serviço, o tamanho da população e a disciplina da fila [3].

Algumas nomenclaturas utilizadas para definir os modelos podem ser encontradas na Tabela 1.

Notação	Definição	Símbolo	Explicação
GI	Distribuição dos tempos de chegada	M	Lei de *Poisson*
		D	Determinística
		EK	Distribuição *Erlang* com parâmetro de forma k
		G	Geral
G	Distribuição dos tempos de serviço	M	Lei de *Poisson*
		D	Determinística
		ED	Distribuição *Erlang* com parâmetro de forma k
		G	Geral
M	Número de Servidores	1, 2, 3, …	

Tabela 1 - Nomenclaturas utilizadas no modelo de Kendall [3].

Mas existem outros modelos propostos por Fogliatti. Alguns exemplos destes modelos são [3]:

- M/M/1/∞/FIFO: apenas um posto de atendimento, não havendo limite para o espaço disponível para a fila de espera, e a ordem de atendimento de utilizadores é a "primeiro a entrar, primeiro a sair" (em inglês, FIFO: First In, First Out);

- M/M/1/K/FIFO: apenas um posto de atendimento, havendo limite para o espaço disponível para a fila de espera, e a ordem de atendimento de utilizadores é a "primeiro a entrar, primeiro a sair" (em inglês, FIFO);

- M/M/C/∞/FIFO: "N" postos de atendimento, não havendo limite para o espaço disponível para a fila de espera, e a ordem de atendimento de utilizadores é a "primeiro a entrar, primeiro a sair" (em inglês, FIFO);

- M/M/C/K/FIFO: vários postos de atendimento, havendo limite para o espaço disponível para a fila de espera, e a ordem de atendimento de utilizadores é a "primeiro a entrar, primeiro a sair" (em inglês, FIFO);

Gerir eficazmente um serviço de atendimento, por exemplo, numa farmácia ou até numa loja de cidadão, torna-se um processo complicado, pois não temos conhecimento do momento exato em que um cliente / utente vai chegar e do tipo e quantidade de recursos que este irá precisar.

2.2 Os sistemas de filas de espera

Um sistema de filas pode ser definido por um conjunto de ações: os clientes / utentes provenientes de uma determinada **população** que precisam de um determinado **serviço**, esperam pelo serviço numa **fila**, caso não sejam atendidos de imediato, seguindo uma determinada regra (**disciplina da fila**), e saírem do sistema após terem sido atendidos.

Em resumo, pode-se dizer que um sistema de filas tem 3 características muito importantes:

- ✓ **Fonte** ou **população**, clientes / utentes que irão comparecer no sistema;
- ✓ **Fila**, composta pelo conjunto de clientes / utentes que se encontram à espera de serem atendidos (não inclui o(s) cliente(s) / utente(s) que já estão a ser atendidos);
- ✓ **Serviço** ou **atendimento**, que pode ser composto por um ou vários postos de atendimento.

A Figura 1 demonstra os elementos principais de um sistema de filas de espera: os clientes / utentes que chegam ao sistema, a fila de espera que é criada e o cliente / utente a ser atendido pelo servidor ou colaborador (quem presta o serviço).

Existem outros elementos importantes para um sistema de filas, tais como, a disciplina da fila, a capacidade do sistema, o número de servidores e o número de fases que compõem o sistema.

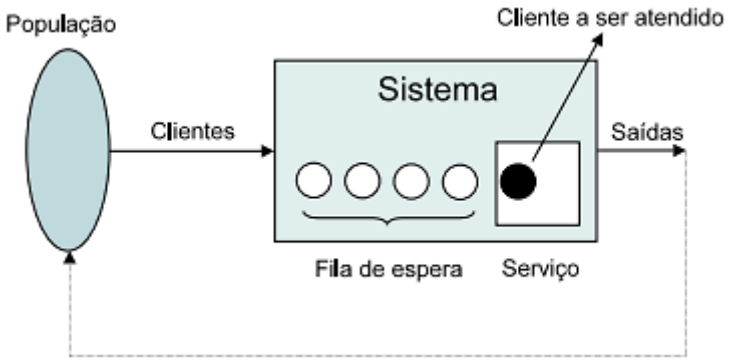

Figura 1 - Componentes de um sistema de filas [4]

Devido ao desconhecimento de quantos clientes / utentes vão chegar, o seu ritmo de chegada e à capacidade de atendimento (tempo necessário para a prestação do serviço), quer por um servidor ou por um colaborador, torna a gestão de um sistema de filas de espera um processo bastante complicado.

2.1.1 Fonte ou população

Os clientes / utentes que chegam ao sistema são oriundos de uma determinada população.

Relativamente à **Fonte** ou **População** deve-se ter em conta algumas das características principais, nomeadamente [5]:

- Tamanho da população

  - ✓ Finita: quando a probabilidade de ocorrer uma nova chegada **é** influenciada pelo número de clientes / utentes que já se encontram no sistema, isto é, quando o número de clientes / utentes admitidos no sistema é limitado.
  - ✓ Infinita: quando a probabilidade de ocorrer uma nova chegada **não é** influenciada pelo número de clientes / utentes que já existem no sistema, o que poderá significar que a chegada de clientes / utentes poderá ultrapassar a capacidade do sistema a qualquer altura, isto é, significa que não existem restrições de chegada;

- Tamanho da chegada: modo de como os clientes / utentes vão chegando a um serviço: clientes / utentes chegam um a um (dimensão da chegada unitária) ou clientes / utentes chegam em grupo)

- Controlo das chegadas
  - ✓ Controláveis (exemplo: agendamento em dias fixos)
  - ✓ Incontroláveis (exemplo: chegada de clientes a uma caixa de supermercado)

- Distribuição das chegadas: a distribuição das chegadas pode ser caracterizado pelo número de chegadas por unidade de tempo (distribuição das chegadas) ou pelo tempo entre duas chegadas seguidas (tempo entre chegadas). Pode-se dizer que esta distribuição pode ser:
  - ✓ constante – quando existem chegadas sucessivas com intervalos de tempo fixos;
  - ✓ aleatório - quando existem chegadas sucessivas com intervalos de tempo entre estas que não podem ser previstos.

- Taxa de chegada (λ): número de clientes / utentes que, em média, chegam ao sistema por cada unidade de tempo. Este valor pode ser:
  - ✓ dependente do estado do sistema;
  - ✓ independente do estado do sistema.

O tempo que decorre entre chegadas seguidas é denominado de *inter-arrival time*. A Figura 2 ilustra este conceito.

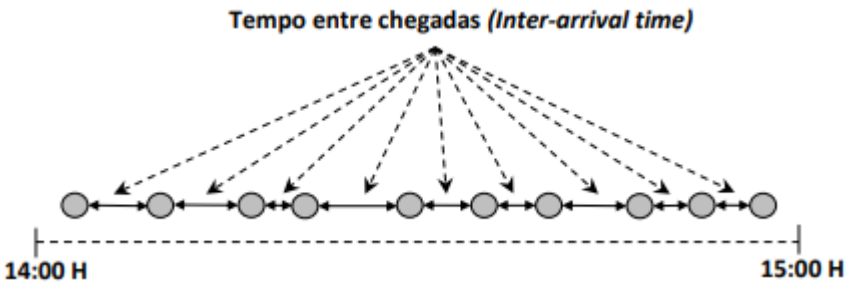

Figura 2 - Medidas de padrões de chegada [5]

Nela, conseguimos verificar que durante uma hora (14:00 – 15:00), existiram 10 chegadas. Se considerarmos que os dados da figura foram verificados noutros horários, pode-se afirmar que a taxa média de chegada é de 10 clientes / utentes por hora. Com esta informação, é possível calcular o tempo médio de espera de cada cliente / utente: se dividirmos a quantidade de clientes / utentes que comparecem num determinado serviço, pelo período de tempo deste exemplo (60 minutos), obtém-se o tempo médio entre as chegadas. Sendo assim, o tempo médio entre chegadas será de $60/10 = 6$ minutos. Concluindo, podemos afirmar que um cliente / utente chega, em média, de 6 em 6 minutos, totalizando o aparecimento de 10 clientes / utentes por hora.

2.1.2 Fila

A **Fila** de espera é formada pelos clientes / utentes que se encontram a aguardar serem atendidos. Esta **Fila** não contempla o(s) cliente / utente(s) que estão já a ser atendidos. Esta pode ser caracterizada quanto ao número, comprimento e disciplina da mesma.

O número de filas (exemplificado na Figura 3) pode ser de dois tipos: fila simples ou fila única quando existe uma única fila mesmo que o servidor tenha disponíveis vários postos de atendimento, ou fila múltipla quando existe uma fila por posto de atendimento e assim, cada conjunto fila/posto de atendimento, significa um sistema separado de fila de espera).

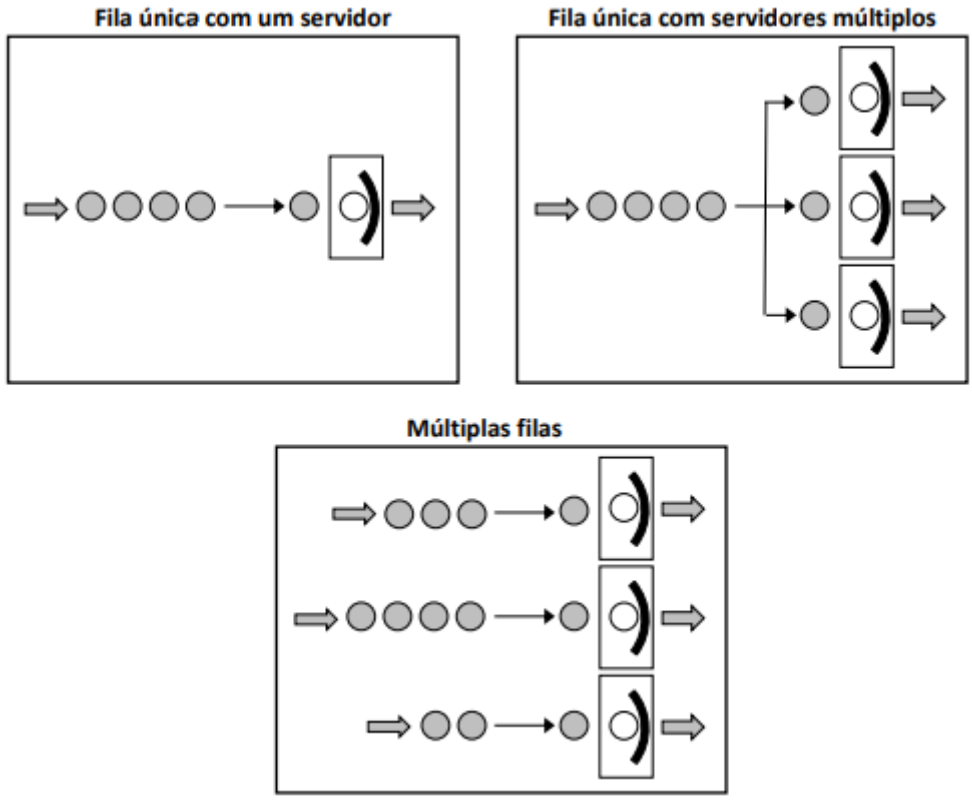

Figura 3 – Tipos de filas [5].

O comprimento da fila significa a capacidade do sistema. Este pode ser infinito quando o comprimento da fila é muito grande relativamente ao número de pessoas que a constituem habitualmente, e finito quando a fila pode ter um número pequeno de clientes / utentes.

A disciplina da fila é o modo de definir a ordem do atendimento dos clientes / utentes que

chegam ao sistema. Pode ser por ordem de chegada (FCFS -> *First Come First Served* ou FIFO *(First In First Out*), através de prioridades ou de forma aleatória. Exemplos:

- Atendimento por ordem de chegada;
- Tempo de processamento mais curto;
- Agendamentos primeiro;
- Prioritários primeiro;
- etc.

2.1.3 Serviço

Relativamente ao serviço (o servidor ou um posto de atendimento) é importante ter conhecimento do tempo que demora a ser efetuado. O tempo de serviço é o tempo decorrido desde que o serviço foi solicitado até que este seja atendido [5].

A configuração do serviço que é o número de fases de atendimento e número de servidores em paralelo (postos de atendimento).

A dimensão do serviço pode ser em grupo (por exemplo, um elevador com capacidade para vários clientes / utentes em simultâneo) ou simples.

A distribuição do tempo de serviço pode ser aleatória ou constante.

A taxa de serviço (μ) é a média de clientes / utentes que podem ser atendidos em cada servidor por unidade de tempo.

$$\frac{1}{\mu} = \text{duração média do serviço}$$

Se a média de clientes / utentes que forem atendidos for menor ou igual à taxa de serviço (μ), pode-se afirmar que acontecem alturas de inatividade.

A taxa de chegada (λ) é a média de clientes / utentes que procuram o serviço por unidade de tempo. Esta taxa pode ser dependente ou independente do estado do sistema. Exemplo de uma taxa de chegada dependente do estado do sistema é o atendimento num serviço de apoio cliente numa operadora que quanto mais depressa atende um cliente / utente, menos fila se forma.

Um sistema de filas de espera pode ser constituído por uma ou várias fases. Quando temos um cliente / utente que tem de passar por mais do que um servido, então esse sistema de filas de espera tem múltiplas fases. Cada fase pode ser formada por um ou vários servidores, criando-se uma única fila ou múltiplas filas [5].

As figuras 4, 5 e 7 representam sistemas de fase única; a figura 6 e 8 representam sistemas de múltiplas fases.

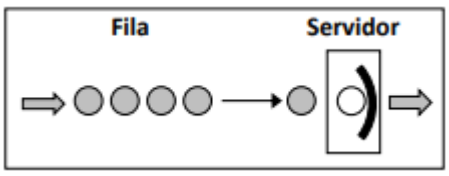

Figura 4 - Serviço ou Posto de atendimento único, fase única. Exemplo: Marcação de consulta. [5]

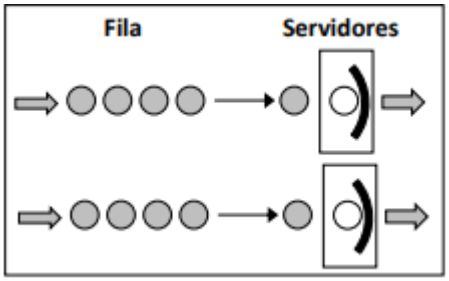

Figura 5 – Múltiplos servidores ou postos de atendimento, uma fila por posto, fase única. [5]

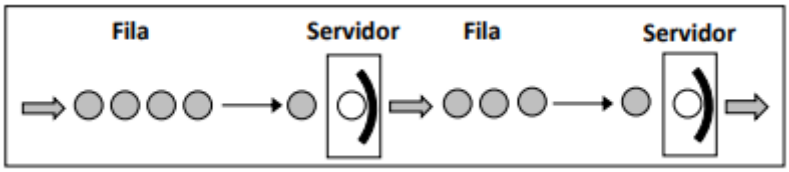

Figura 6 - Servidor ou posto de atendimento único, múltiplas fases (um posto em cada fase). Exemplo: Consulta médica. [5]

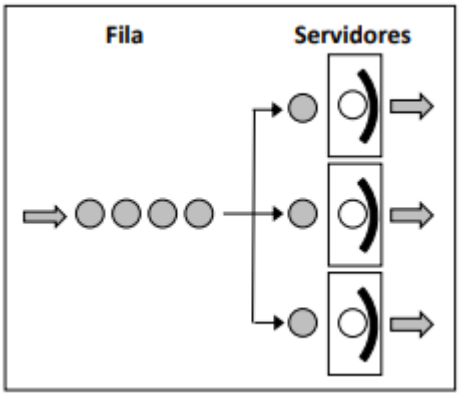

Figura 7- Múltiplos servidores ou postos de atendimento, fase única. Exemplo: Sala de tratamentos de enfermagem. [5]

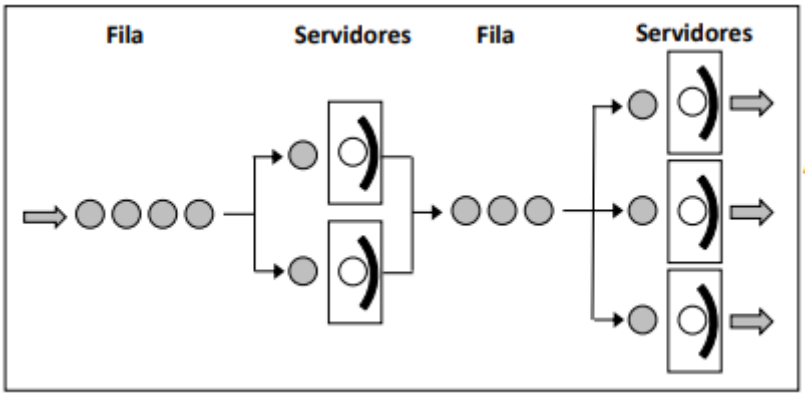

Figura 8 - Múltiplos servidores ou postos de atendimento, múltiplas fases. Exemplo: Serviço de urgência hospitalar com triagem. [5]

2.3 Medidas de desempenho

O desempenho do sistema pode ser medido tendo por base determinadas medidas. A Tabela 2

introduz essas medidas, nomeadamente:

Medidas de desempenho	
Símbolo	**Característica**
L_s	Número médio de clientes / utentes no sistema
L_q	Número médio de clientes / utentes na fila de espera
W_s	Tempo médio que um cliente / utente está no sistema (tempo na fila + tempo de serviço)
W_q	Tempo médio que um cliente / utente está na fila de espera (exclui o tempo que o cliente / utente demora a ser atendido)
P_n	Probabilidade de estarem ***n*** clientes / utentes no sistema
P_0	Probabilidade de estarem **zero** clientes / utentes no sistema
$P(W_q = 0)$	Probabilidade do tempo de espera na fila ser zero
$P(W_q > t)$	Probabilidade do tempo de espera na fila exceder t
$P(W_s > 0)$	Probabilidade do tempo no sistema exceder t

Tabela 2- Terminologia usada nas medidas de desempenho [5].

A letra **L** (*Length*) está relacionada com o **comprimento da fila** e a letra **W** (de *Waiting*) está relacionada com o **tempo de espera**. Assim, usam-se as letras **q** (*queue*) e **s** (*system*) para indicar a fila e o sistema, respetivamente.

2.4 Redes de filas de espera

Até agora foram apenas considerados sistemas de filas de espera com apenas uma localização de atendimento, tenha um ou mais servidores (postos de atendimento). No entanto, existem situações, como por exemplo no atendimento de uma loja de cidadão, onde é possível que um cliente / utente passe por um conjunto de filas de espera, podendo seguir ou não uma determinada ordem, significa que o *output* de algumas filas serão o *input* de outras.

Pode então dizer-se que estamos perante um sistema de múltiplas de filas de espera, denominado por rede de filas de espera. Quando temos esta situação, é importante estudar a rede para ter o conhecimento, para além dos tempos de espera ou o número de clientes / utentes por fase, também o tempo total de espera ou o número total de clientes / utentes no sistema.

Alguns exemplos de soluções que permitem redes de filas de espera são apresentados no capítulo seguinte. Qualquer solução de sistema de gestão de filas de espera tem que ser o mais completo possível, sendo capaz de se "adaptar" e abranger qualquer cenário. Daí torna-se quase obrigatório que tenham a possibilidade de funcionar com redes de filas de espera.

2.5 Exemplos de Sistemas de Gestão de Filas

Existem diversos sistemas de gestão de filas atualmente no mercado. Existem soluções comerciais e de código aberto.

Exemplos de soluções comerciais são:

- Q.track [6]

Produzido pela empresa LogicPulse (Figueira da Foz).

É uma solução que se adapta a qualquer tipo de negócio (retalho, saúde, entidades governamentais, etc.).

- QMagine [7]

Produzido pela empresa PARTTEAM & OEMKIOSKS (Vila Nova de Famalicão.

Ambas as soluções apresentadas anteriormente adaptam-se a qualquer tipo de negócio (retalho, saúde, entidades governamentais, etc.).

Permitem gestão de atendimento de fila única ou de multi-filas. Possuem alertas por SMS, onde os utilizadores recebem mensagens com aviso de proximidade para o seu atendimento, permite recolha de dados estatísticos através de gráficos e tabelas, o que é bastante positivo para um gestor conseguir saber o estado do atendimento, que funcionário está a atender mais pessoas ou que funcionário demora mais nos atendimentos. Possui também chamada de clientes / utentes por voz, o que é algo que faz toda a diferença, porque nos dias de hoje ainda existem pessoas que não sabem ler.

- SIGA [8]

Produzido pelo Instituto de Informática, é uma solução bastante completa, mas que apresenta mais informação que as restantes soluções. Permite filtrar serviços por distrito ou por proximidade em quilómetros.

É uma solução que se encontra em funcionamento em diversas entidades públicas, nomeadamente:

- Instituto da Segurança Social;

- Instituto do Emprego e Formação Profissional;

- Instituto da Mobilidade e dos Transportes, L.P.;

- Agência para a Modernização Administrativa;

- Polícia de Segurança Pública;

- Instituto dos Registos e do Notariado;

- Instituto da Segurança Social dos Açores e Madeira;

- Autoridade para as Condições do Trabalho;

- Autoridade Tributária e Aduaneira;

- etc.

Esta solução apresenta três módulos: sigaBase (atendimento e estatísticas), sigaPlus (portal de marcação de atendimento,ou seja, agendamentos) e sigaApp (aplicação para *Smartphone* e Tablet que permite tirar senhas digitais, evitando assim a deslocação antecipada ao serviço de atendimento).

- QSystems [9]

Produzido pela Altronix (Porto).

É uma solução que se adapta a qualquer tipo de negócio (retalho, saúde, entidades governamentais, etc.).

A solução permite medir o desempenho dos colaboradores, permite o envio de mensagens ou *e-mail*, permite agendamento de um atendimento ou até senhas digitais, permite o acesso a relatórios que podem ser enviados por e-mail. É uma solução que tem uma vertente também muito utilizada nos dias de hoje, a *cloud*.

Todas as soluções apresentadas anteriormente parecem bastante completas e são soluções eficazes para qualquer entidade. Acabam por ser idênticas entre si, pois apresentam os mesmos módulos e mesmas funcionalidades.

Exemplos de soluções de código aberto são:

- Osyes [10]

Produzido pela Osyes. É uma solução extensível e personalizável. É orientado para a Web, pois apenas é necessário um *browser*. Tem alguns requisitos, tais como, o sistema operativo (Linux / Unix), necessita de ter acesso ao Apache 1.3+, PHP 5.3.2 + e o MySQL 4.1 ou superior para funcionar corretamente. Temos a possibilidade de experimentar uma versão de demonstração, mas depois da análise do sistema, é de todo muito mais simples do que as soluções comerciais apresentadas, pois é bastante mais limitado em termos de funcionalidades e em termos visuais.

- FQM [11]

Produzido pela FQM. É uma solução que funciona em vários sistemas operativos (Windows, GNU / Linux e MacOS). É orientado para a Web, pois apenas é necessário um *browser*. Atualmente encontra-se na versão 0.8 Beta. A solução apresenta algumas funcionalidades que o fazem ser melhor que outras soluções de código aberto, tais como, suporta impressoras POS USB, permitindo impressão, permite customizar as interfaces da solução, oferece uma plataforma multimédia, na qual qualquer utilizador consegue enviar vídeos, imagens ou ficheiros de som, para que possam ser incorporados no processo de personalização, permite suporte a anúncios verbais de texto em fala, claros e de alta qualidade, em cinco idiomas diferentes: inglês, árabe, francês, italiano e espanhol e com diferentes preferências e configurações, permite também ter acesso a uma plataforma baseada no utilizador, para que vários utilizadores possam registar, fazer login e monitorar seus próprios atribuídos pelas tarefas administrativas.

3. Sistema de gestão de filas Q.track

Neste capítulo é descrito o sistema de filas utilizado, o Q.track, para o desenvolvimento da componente prática desta dissertação.

Como se pode verificar na Figura 9, o sistema de filas utilizado é composto por um conjunto de elementos, essencialmente, um **quiosque** (módulo de impressão de senhas) para ser possível retirar senhas, uma **televisão** (módulo do Painel de Chamada) que permite aos utentes visualizarem a atividade da organização onde se encontram, nomeadamente, a última senha chamada de cada serviço e em que balcão foi chamada, um **computador ou um servidor** que servirá para alojar a aplicação e a base de dados da empresa. Este computador ou servidor terá de ter rede, pois servirá de ligação entre os elementos mencionados anteriormente.

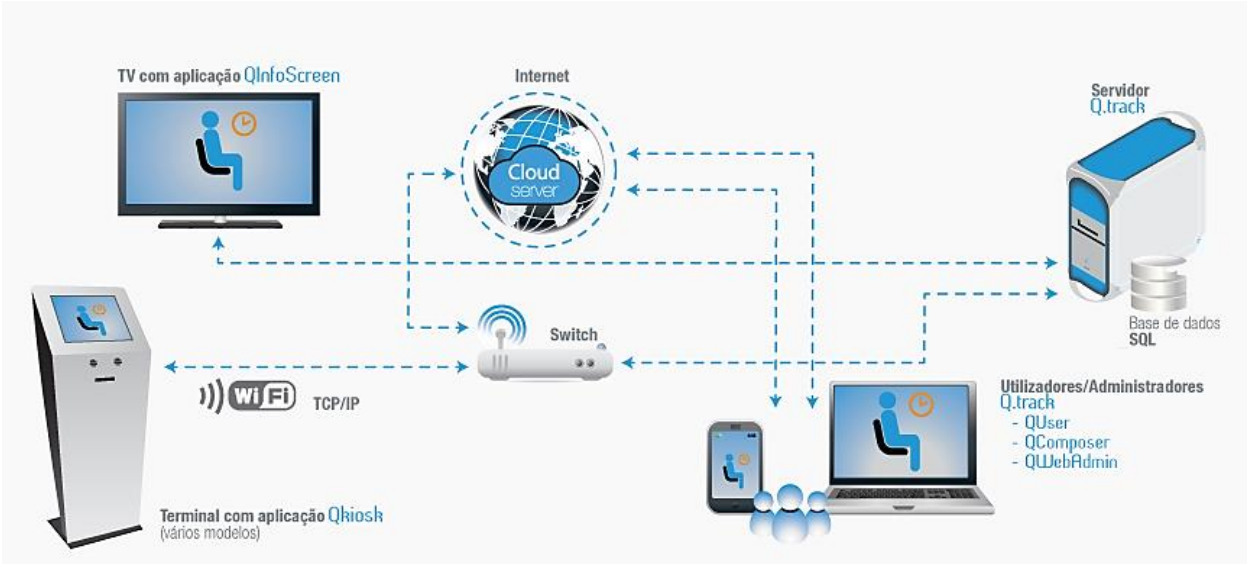

Figura 9 - Esquema de Gestão de Filas / Atendimento [6]

Todos estes elementos são explicados mais em pormenor de seguida.

3.1 Autenticação no sistema (QWeb)

Para um utilizador conseguir gerir o sistema de gestão de filas / autenticação, necessita de efetuar o início de sessão no portal WEB (Figura 10). Para tal existe um utilizador administrador que consegue fazer tudo na aplicação e, por conseguinte, este poderá criar os utilizadores necessários para início de sessão, bem como, criar os utilizadores que irão realizar atendimentos.

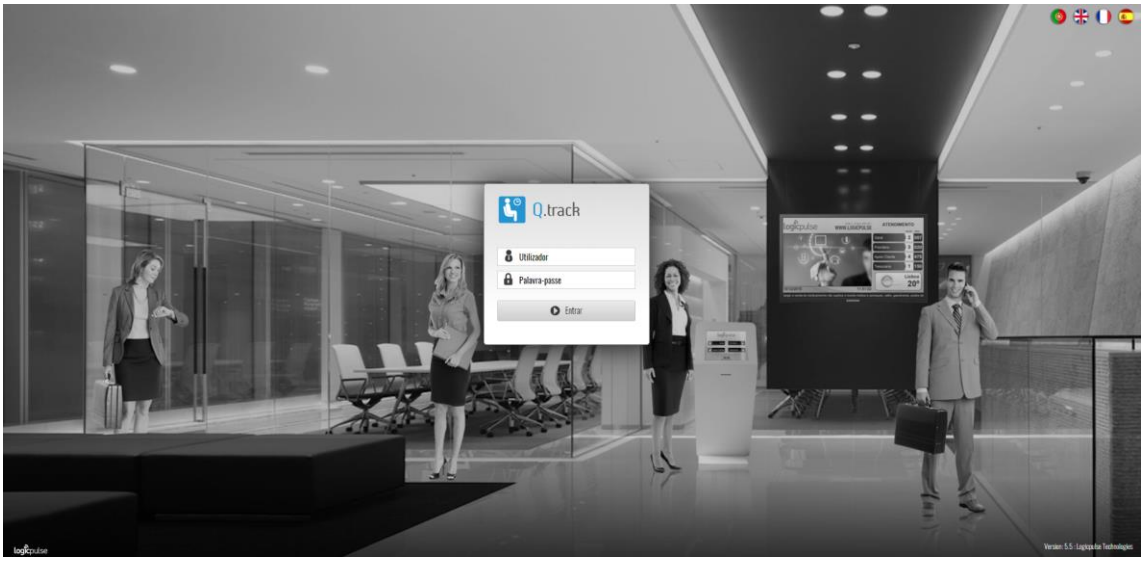

Figura 10 - Página de autenticação da componente WEB.

3.2 Módulo de Impressão de Senha

O módulo de impressão de senhas existe em dois formatos: sistema Android (Figura 11) e sistemas Windows (Figura 12).


Figura 11 - Quiosque Android dispensador de senhas. [6]


Figura 12 - Quiosque Windows dispensador de senhas. [6]

Este módulo permite ao utente retirar a senha que representa a sua vez na fila de espera do sistema de filas.

3.3 Módulo de Atendimento (QUser)

No módulo de atendimento existe a aplicação QUser (Figura 13). Esta aplicação permite realizar um conjunto de tarefas, nomeadamente, atendimento da fila de espera, tendo em conta se existem utentes de algum serviço prioritário ou não, redirecionar senhas para outros serviços ou gabinetes, visualizar o número de senhas em espera por serviço por balcão e por gabinete, imprimir uma senha para um determinado serviço, ou até chamar uma senha de um serviço à escolha.

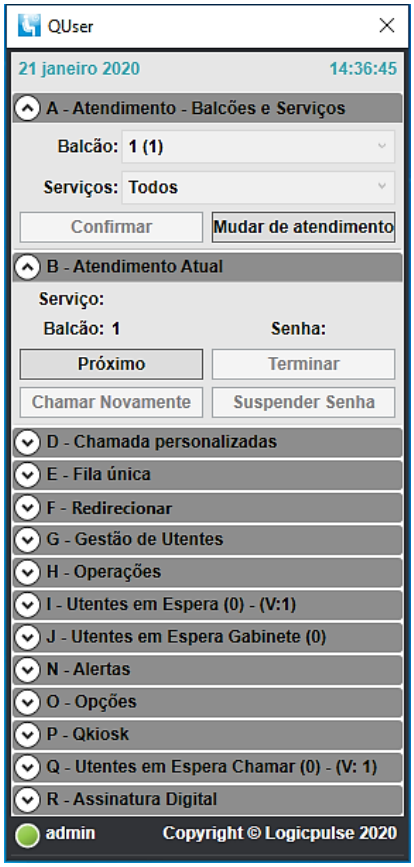

Figura 13 - Aplicação de atendimento de senhas.

3.4 Módulo do Painel de Chamada

O módulo de painel de chamada encontra-se visível através de uma televisão. A aplicação é instalada numa box Android (Figura 14) ou num computador Windows (Figura 15) sendo visível numa televisão (Figura 16).

Figura 14 - Exemplo de uma box Android utilizada. [6]

Figura 15 - Exemplo de uma box Windows utilizada. [6]

Este módulo permite ao utente visualizar, em tempo real, o estado do atendimento.

Recorrendo à Figura 16 é possível visualizar um *template*, que apresenta um conjunto de quatro serviços: Atendimento Geral, Testes, Administração Medicamentos e Encomendas; para serviço é visível a senha atual e o balcão onde foi chamada.

Figura 16 - Exemplo de um *template* visível numa televisão.

3.5 Módulo de criação de templates (QComposer)

No módulo de atendimento existe a aplicação QComposer (Figura 17) que permite a criação de *templates* para os dispensadores de senhas e painel de chamada. Exemplo de um *template* criado neste módulo está representado na Figura 16.

Figura 17 - Aplicação de criação de *templates*.

4. Trabalho Desenvolvido

O trabalho desenvolvido passou por diversas fases.

O primeiro passo foi definir o tema. Posteriormente, foi feito um contacto com a empresa portadora da solução Q.track, a Logicpulse, para autorização da utilização e acesso ao produto para fins desta dissertação. Após o aval por parte da Logicpulse e, tendo por base o facto da mesma já ter pensado e definido o que pretendiam para a sua aplicação para *smartphones*, teve início a construção desta dissertação.

Começou-se por escrever o estado de arte sobre os sistemas de filas, realizando uma pesquisa profunda sobre sistemas de filas e seu funcionamento. Foi necessário ter conhecimento mais aprofundado de como funcionam e que tipos de sistemas existem na atualidade. Foi feita uma pesquisa sobre exemplos de sistemas de gestão de filas existentes no mercado, passando por soluções comerciais e de código aberto.

Após esta fase, passou-se à parte prática, isto é, a criação de um sistema para dispositivos móveis que sirva de complemento a um sistema de gestão de filas existente, o Q.track. Esta aplicação servirá a vertente cliente / utente, pois permite ter acesso a um conjunto de funcionalidades, nomeadamente, visão geral do estado de atendimento ou até retirar uma senha mediante a escolha de um determinado serviço.

Para o desenvolvimento desta aplicação, foram efetuadas reuniões com a empresa portadora do sistema Q.track, a Logicpulse, para definição do que a mesma pretendia com esta aplicação. Ficou decidido que teria de desenvolver uma aplicação para telemóvel (com sistemas operativos IOS e Android) que servirá de complemento ao sistema de gestão de filas que existe atualmente.

Para desenvolver a aplicação foi necessário ter acesso a quatro componentes muito importantes para o sucesso da criação e funcionamento da aplicação:

1. Ter instalado na minha máquina o software Visual Studio 2019 [12];
2. Ter instalado na minha máquina o software Microsoft SQL Server [13];
3. Ter o projeto mais atualizado do *web-service* do Q.track (sistema que funciona como *back-office*);
4. Ter uma Base de Dados para consulta de informação a ser mostrada.

Para a componente prática desta dissertação, foi criado um projeto para a aplicação para dispositivos móveis que permite efetuar um conjunto de tarefas.

A construção da aplicação teve conhecimento e acompanhamento por parte de elementos da empresa Logicpulse, desde o *layout*, tecnologias a utilizar, até ao que pretendiam no seu todo. Foi pedido que fosse uma aplicação simples de usar, apresentasse menus de fácil compreensão, e que permitisse com poucos passos obter o resultado pretendido, isto é, não ter que passar por mil menus até chegar ao que o utilizador pretende. Foram definidas quais as tecnologias a utilizar, tecnologias essas descritas no capítulo 4.1. Foram definidas também, que funcionalidades eram pretendidas, funcionalidades essas que estão descritas no capítulo 4.2.

No Anexo A são apresentadas capturas de ecrã que demonstram as funcionalidades e opções disponibilizadas na aplicação desenvolvida.

4.1 Tecnologias utilizadas

A aplicação foi construída recorrendo às tecnologias DevExpress [14], JavaScript [15], ASP.NET [16] e SQL [18], pois são tecnologias utilizadas atualmente. O principal motivo da utilização destas tecnologias devem-se aos conhecimentos prévios nas mesmas, mas também teve em consideração diretrizes recomendadas pela empresa da solução.

4.1.1 DevExpress

Com o uso mais recorrente de *smartphones* ou *tablets*, criar aplicações para dispositivos que respondem ao toque, tornou-se cada vez mais recorrente. A criação de aplicações pode-se tornar mais fácil com o uso de softwares especializados. Um deles é o DevExpress. É destinada a programadores de softwares para desktop, web e mobile, o DevExpress conta com ferramentas que simulam a interface do utilizador, permitindo um entendimento maior da experiência que será oferecida e tornando o processo de criação mais intuitivo.

A tecnologia DevExpress passa por disponibilizar mais de 500 *widgets* para aplicações, mas também pode ser utilizado em aplicações de *Desktop*. Permite controlos para diversas linguagens de programação. São exemplos:

- ASP.NET (Web Forms, MVC e Core);
- Xamarin;
- HTML (através da DevExtreme);

O DevExtreme pode ser usado com diferentes tecnologias e suporta profunda integração com bibliotecas do lado do cliente. Exemplos são:

- Angular;
- ASP.NET Core;
- ASP.NET MVC;
- jQuery;
- React;
- Vue.

A tecnologia DevExpress apresenta um conjunto de demos para cada *widget*, o que permite obter e experimentar diferentes configurações.

Exemplo do código de um *widget* que adiciona um botão*:*

```
<div id="button1" data-bind="dxButton: { text: 'Guardar', onClick: $root.saveButtonClick}"></div>
```

No código anterior é possível ver um identificador da secção onde se encontra o botão e duas propriedades do botão: o texto a mostrar no botão (`text: 'Guardar'`) e a função de clique do mesmo (`onClick: $root.saveButtonClick`). Mas existem muito mais propriedades, tais como:

- tamanho do botão (comprimento e largura);
- tipo de botão (normal, sucesso, defeito, perigo);
- disponibilidade;
- visibilidade;
- ícone.

Mas como todas as tecnologias, esta apresenta vantagens e desvantagens. A Tabela 3 demostra algumas das vantagens e desvantagens desta tecnologia:

DevExpress 2018	
VANTAGENS	DESVANTAGENS
• Muito versátil • Muitos componentes • Componentes com muitas opções de configuração	• Encontrar informação resolução de problemas encontrados, por vezes, é difícil.

Tabela 3 - Vantagens e Desvantagens da tecnologia DevExpress.

4.1.2 JavaScript

JavaScript ou JS é uma linguagem de programação utilizada para controlar HTML e CSS para manipular comportamentos numa página web. Foi criada em 1995 por Brandan Eich (colaborador na Netscape).

Em conjunto com HTML e CSS, o JavaScript é uma das mais importantes tecnologias da World Wide Web.

Hoje a linguagem JavaScript está presente na maioria dos *browsers*, dispositivos desktops e móveis e sistemas operativos .

Dados de 2016 indicam que o JS é utilizado por mais de 90% dos programadores de sites. Isso revela que, pouco tempo após a sua criação, passou de algo primitivo para ser uma das linguagens mais importantes na internet.

Permite criar páginas web dinâmicas, aplicações para dispositivos móveis, animações, gráficos em três dimensões ou até mapas interativos.

O código JavaScript é representado de várias maneiras possíveis:

1. dentro de dois identificadores <script> e </script>. Assim o *browser* sabe que o código entre estes identificadores é código JavaScript em vez de código HTML.

   Exemplos:

   - referência a um ficheiro JS:

   ```
   <script type="text/javascript" src="../js/ /jquery.min.js"></script>
   ```

   - função

   ```
   <script type="text/javascript">
           function Confirm() {
               alert("Hello World");
           }
   </script>
   ```

2. associando uma ação de um componente HTML a uma função JavaScript.

Exemplo*:*

```
<button type="button" onclick="document.getElementById('demo').innerHTML =
Date()">Mostrar data</button>
```

No código anterior é possível ver associado ao clique do botão uma função JavaScript. Este exemplo usa o método para "encontrar" um elemento HTML (com id = "demo") e altera o valor do elemento (innerHTML) com o resultado da função *Date*() que obtém a data e hora do sistema.

De seguida é mostrado um exemplo do código de uma função JavaScript com o nome *showMessage*. Esta função irá apenas mostrar um alerta com a mensagem "*Hello World*!"*:*

```
<script type="text/javascript">
	function showyMessage()
	{
		alert("Hello World!");
	}
</script>
```

Mas como todas as tecnologias, esta apresenta vantagens e desvantagens. A Tabela 4 demonstra algumas das vantagens e desvantagens:

JavaScript	
VANTAGENS	DESVANTAGENS
• Extremamente versátil • Linguagem bastante simples • Linguagem do lado do cliente • Alta compatibilidade com plataformas, sistemas e *browsers* • É mais rápida que outras linguagens de programação • Permite ter vários efeitos visuais • Permite que as páginas na internet sejam mais dinâmicas e interativas • Permite encontrar e corrigir mais facilmente erros de programação	• Muitas linhas de código • Não permite reaproveitamento de funções • É vulnerável a falhas de segurança nos sistemas, browsers e páginas envolvidas • Pode ser usada para executar programas com más intenções sem que o utilizador se aperceber • Nem sempre compatível com todos os *browsers* (Exemplo: Internet Explorer) e sistemas existentes

Tabela 4 - Vantagens e Desvantagens da tecnologia JavaScript.

4.1.3 ASP.NET

ASP.NET é a *framework* do .NET projetada para o desenvolvimento de páginas web dinâmicas.

Foi lançada pela primeira vez em Janeiro de 2002 com a versão 1.0 do .NET Framework e recentemente foi disponibilizada a versão 4.8 de tal framework.

Podemos descrever a framework .NET como uma combinação de diversas tecnologias que auxiliam os programadores no desenvolvimento de uma multiplicidade de aplicações. Os elementos que

compõe tais tecnologias são os seguintes [17]:

- **linguagens .NET** (C#, Visual Basic,, C++ e F#);

- ***Common Language Runtime*** (CLR): é a componente do *software* que gera o código em tempo de execução, disponibilizando serviços automáticos como gestão de *threads* e gestão de memória enquanto disponibiliza tipos de segurança e verifica se as aplicações podem ser executadas sem erros.

- **Biblioteca de Classes**: fornece aos programadores funcionalidades para serem usadas nas suas aplicações. Habitualmente estas funcionalidades estão agrupadas por conjuntos de tecnologias, por exemplo, *Windows Presentation Foundation* ou WPF (utilizado para construir aplicações para ambientes Windows) ou o ADO.NET (utilizado em aplicações com ligação a base dados).

- **ASP.NET**: alberga as aplicações Web que são criadas pela *framework* .NET e suporta quase todas as características da biblioteca de classes da *framework* .NET. Apresenta também um conjunto de serviços Web, como por exemplo, armazenamento de dados ou autenticação segura.

- **Visual Studio** (VS): ferramenta que permite a construção de aplicações e que dispõe várias opções de programação e *debugging* (processo que permite localizar erros). Esta ferramenta possui a *framework* .NET no seu todo.

No processamento de uma página em ASP.NET, existem três pontos bastante importantes que influenciam o modo como a esta vai ser apresentada e como se vai comportar [17]:

- **Texto estático**: Como por exemplo JavaScript, HTML ou CSS que é escrito numa página e é enviado para o *browser* sem passar pelo processo de *debugging*.

- **ASP.NET *Server Controls***: São utilizados nas páginas aspx e quando são processados, são transformados em HTML.

- **Código de programação**: Código como C# pode ser acionado com base numa ação do utilizador ou pode ser escrito num ficheiro separado e ser executado automaticamente. A execução do código pode influenciar o modo como a página é apresentada.

4.1.4 SQL

SQL (*Structured Query Language* ou Linguagem de Consulta Estruturada) é uma linguagem de pesquisa de acesso a dados utilizada para interagir com base de dados relacionais [18]. Quando se interage com uma base de dados relacional é utilizado SQL para extrair, modificar e atualizar informação. Permite também criar tabelas, colunas, índices ou atribuir permissões a utilizadores.

A linguagem SQL é composta por subconjuntos, cada um com propósitos bem definidos [19]:

- **DQL** (*Data Query* Language ou Linguagem de Consulta de Dados) - comando (SELECT) que permite a consulta de dados armazenados numa ou várias tabelas;

Exemplo:

```
SELECT * FROM Utilizadores;
```

- **DML** (*Data Manipulation* Language ou Linguagem de Manipulação de Dados) - comandos (INSERT, UPDATE e DELETE) que permitem a manipulação de dados nas tabelas;

Exemplos:

```
INSERT INTO Utilizadores (nome, contacto, morada)
VALUES ('Tânia Bogalho', '000000000', 'Travessa das
Flores');

UPDATE Utilizadores
SET nome = 'Tânia Bogalho' WHERE UtilizadorID = 1;

DELETE FROM Utilizadores WHERE UtilizadorID = 1;
```

- **DDL** (*Data Definition* Language ou Linguagem de Definição de Dados) - comandos que permitem criação (*CREATE*) de *views*, índices, tabelas, ou atualização dessas estruturas (*ALTER*), bem como a sua remoção (*DROP*);

- DCL (*Data Control* Language ou Linguagem de Controlo de Dados) - comandos que permitem gerir o acesso às bases de dados, removendo (*REVOKE*) ou adicionando (*GRANT*) permissões de acesso;

- DTL (*Data Transaction* Language ou Linguagem de Transação de Dados) - comandos que permitem controlar as transações que forem efetuadas na base de dados, desde o iniciar (*BEGIN*) uma transação, confirmá-la (*COMMIT*) ou desfazê-la (*ROLLBACK*).

4.2. Funcionalidades implementadas

O tempo de espera em qualquer serviço tem um papel importante na qualidade do mesmo e, por consequência, isso reflete-se no contentamento dos clientes / utentes. A espera pode tornar-se menos exaustiva e cansativa com a disponibilização de informação relevante, por exemplo número de pessoas em espera ou tempo médio de espera de atendimento.

Considerando a questão anterior desenvolveu-se uma aplicação que tem como finalidade melhorar a forma como os serviços são prestados. Esta aplicação fornece informação relevante sobre as senhas em espera, tempo médio de espera, última senha chamada.

Todo o sistema Q.track funciona à volta de uma ou mais localizações. Exemplos de uma localização é o nome da empresa que adquire a solução, ou então no caso de uma empresa com várias filiais, as localizações passam por ser o nome dessas filiais.

De uma forma geral as funcionalidades que a aplicação Q.track Mobile apresenta são:

- apresentar o número de senhas em espera;
- apresentar o tempo médio de espera;
- apresentar a última senha chamada;
- retirar uma senha de um serviço à escolha;
- acesso a um histórico de senhas emitidas para o cliente em questão.

Cada funcionalidade apresentada anteriormente passa obrigatoriamente por uma ligação ao *web-service,* que por sua vez, realiza uma consulta à base de dados (pesquisa ou inserção de dados).

Toda a informação necessária ao funcionamento da aplicação encontra-se, então, numa base de dados. Algumas das tabelas necessárias para o desenvolvimento da aplicação já se encontravam criadas no início do desenvolvimento deste projeto. No entanto somente as tabelas *Services, Location*, *TicketForward*, *Attendance* e *WaitingClients* contêm dados utilizados no funcionamento da aplicação

desenvolvida.

A tabela *Services* contém informações sobre os serviços (nome, serviço ativo ou não, última senha atendida, senha atual retirada, ordem, …).

A tabela *Location* contém informações sobre as diferentes localizações (nome, localização ativa ou não, máximo de senhas que pode atender por dia, hora de fecho do dia, …).

A tabela *TicketForward* contém informações sobre as senhas redirecionadas entre postos de atendimento (identificação do balcão de saída, identificação do balcão de chegada, o número da senha redirecionada, data e hora do início do atendimento, data e hora do final do atendimento, identificador do serviço de origem, identificador do serviço de destino, identificador do atendimento, …).

A tabela *Attendance* contém informações sobre todos os atendimentos efetuados (número da senha atendida, identificação do serviço da mesma, identificação do balcão de atendimento, identificação do utilizador que atendeu a senha, a data e hora da emissão da senha, data e hora do início do atendimento, data e hora do final do atendimento, estado do atendimento, …).

A tabela *WaitingClients* contém informações sobre a lista de senhas em espera (número da senha, data e hora da emissão da senha, data e hora do atendimento da senha, o identificador do serviço da senha, senha chamada ou não, …).

Sendo assim, passa-se a explicar que processo(s) são realizados em cada uma das funcionalidades desenvolvidas:

- apresentar o número de senhas em espera (consulta à tabela *Services*, verifica a diferença entre o valor da coluna da senha atual tirada num dispensador de senhas e o valor da coluna onde está o número da última senha chamada);
- apresentar o tempo médio de espera (consulta à tabela *Services* e *Attendance* onde é calculada a média da diferença entre a data/hora em que foi retirada a senha e a data/ hora em foi atendida cada senha, isto para cada serviço);
- apresentar a última senha chamada (consulta à tabela *Services*, coluna onde está o número da última senha chamada);
- retirar uma senha de um serviço à escolha (consulta do *web-service* onde realiza um conjunto de verificações, entre elas, verificação se o serviço está ativo ou não, se foi atingido o limite se senhas para o serviço ou localização, e, por fim, inserção de informação da senha retirada na tabela *WaitingClients*);
- acesso a um histórico de senhas emitidas para o cliente em questão (consulta da base de dados local do dispositivo onde a aplicação está a funcionar e, caso já tenha sido retirada alguma senha).

Cada vez que o cliente tira uma senha, a informação relativamente a esta também fica guardada localmente, ou seja, numa *local storage* da aplicação no *smartphone* onde esta se encontra em funcionamento, isto é, caso limpem os dados e cache da aplicação, perde-se este "histórico". É através da consulta e listagem desta "base de dados" local que é mostrada a informação do histórico das senhas tiradas.

A aplicação desenvolvida permite existir serviços por níveis, ou seja, permite uma navegação entre serviços, exemplo dessa situação é uma loja do cidadão, onde temos diversos serviços, tais como: IRN (Instituto dos Registos e do Notariado), IMT (Instituto da Mobilidade e dos Transportes), entre outros. Cada um destes serviços tem os chamados serviços "filhos", serviços que pertencem a um outro serviço, por exemplo:

- Instituto dos Registos e do Notariado
    - Cartão do Cidadão (criação, alteração, etc);

- Passaporte (criação, alteração, etc);
- Carta de Condução (criação, alteração, etc).

Cada "ação" na aplicação requer ligação ao *web-service*. Exemplo do código utilizado na aplicação desenvolvida e do *web-service* para retirar uma senha está demonstrado no Anexo B (Código de uma funcionalidade).

5. Conclusão e Trabalho futuro

Tendo em conta toda a pesquisa realizada, percebeu-se que um sistema de filas é mais complexo do que eu tinha ideia. Envolve mais variáveis do que eu tinha alguma vez pensado.

No dia-a-dia, as filas de espera têm uma importância bastante grande, independentemente dos mais variados contextos. É notório que uma gestão bem-feita de um sistema de filas de espera, melhora a qualidade de vida e, consequentemente, a produtividade.

A disponibilização de informações como o tempo de espera é importante para os clientes / utentes no processo de espera e para o próprio serviço a ter uma ideia do bom ou mau funcionamento dos seus serviços. É daqui que é a base desta dissertação, e é a pensar no que que possa trazer de bom para os clientes / utentes que foi construída a aplicação Q.track Mobile.

A aplicação Q.track Mobile foi desenvolvida com o intuito de facilitar a vida dos clientes / utentes de diversas instituições, daí ser uma aplicação com uma interface de fácil utilização, que permite compreensão rápida da mesma por parte dos clientes / utentes. Disponibiliza Apresenta dados importantes, tais como, a senha atual e a última senha a ser chamada.

A aplicação Mobile desenvolvida para a aplicação Q.track, apresenta funcionalidades importantes para um cliente / utente de uma entidade que possua uma solução de gestão de filas.

Neste sentido, os objetivos principais propostos foram cumpridos com sucesso e a aplicação encontra-se funcional e pronta a ser utilizada pelos clientes da empresa Logicpulse.

A principal melhoria que a aplicação Q.track Mobile deverá sofrer a curto prazo é ao nível de agendamento de serviços, isto é, adicionar a possibilidade de o utilizador conseguir agendar o atendimento para um determinado serviço, para uma data e hora que lhe seja mais conveniente.

Assim como já existe o histórico de senhas, irá existir, também, o histórico das senhas que surjam dos agendamentos.

Bibliografia

1. Teoria das filas. Disponível em: https://pt.wikipedia.org/wiki/Teoria_das_filas
2. Caldeira, Helvio - Teoria das filas de espera. Disponível em https://cmtecnologia.com.br/blog/teoria-das-filas/
3. Jorge, Miguel – Gestão de filas de Espera com Recurso à Simulação. Disponível em: https://iconline.ipleiria.pt/bitstream/10400.8/2579/1/Projeto%20-%20MGE%20-%20Miguel%20Jorge.pdf
4. Oliveira, José - Filas de Espera, 1998. Disponível em https://web.fe.up.pt/~mac/ensino/docs/IO20032004/FilasEspera.pdf
5. Carvalho, Mário - Caracterização de uma Fila de Espera de um Serviço Hospitalar – Um Estudo de Caso, 2015. Disponível em https://iconline.ipleiria.pt/bitstream/10400.8/2547/1/M%C3%A1rio%20Carvalho.pdf
6. Gestão de filas Q.track. Disponível em https://q.track.pt/.
7. Gestão de filas QMagine. Disponível em https://site.qmagine.com/pt/.
8. Gestão de filas SIGA. Disponível em http://siga.seg-social.pt/.
9. Gestão de filas QSystems. Disponível em https://www.qsystems.pt/.
10. QMSOS – Gestão de filas Open Source. Disponível em http://www.osyes.net/queue-management-system.html.
11. Gestão de filas FQM. Disponível em https://fqms.github.io/.
12. Visual Studio. Disponível em https://visualstudio.microsoft.com/.
13. Microsoft Sql Server 2019. Disponível em https://www.microsoft.com/pt-pt/sql-server/sql-server-2019.
14. Devexpress. Disponível em https://www.devexpress.com/.
15. JavaScript. Disponível em https://www.javascript.com/.
16. ASP.NET. Disponível em https://www.asp.net/.
17. Veloso, Tiago – Gestão de filas de espera no Serviço de Urgência. Disponível em: https://repositorium.sdum.uminho.pt/bitstream/1822/19963/1/A49992.pdf
18. SQL. Disponível em https://pt.wikipedia.org/wiki/SQL.
19. Guia Completo de SQL. Disponível em https://www.devmedia.com.br/guia/guia-completo-de-sql/38314.

Anexos

Anexo A (Manual da aplicação desenvolvida)

Quando iniciamos a aplicação, surge a opção de escolha entre duas opções: modo Demo e Já sou cliente / utente.

O modo Demo permite visualizar apenas algumas funcionalidades da aplicação, enquanto que a opção de Já sou cliente / utente, permite realizar login na aplicação e usufruir de todas as funcionalidades.

Figura 18 - Ecrã Inicial Q.track Mobile

Se optarmos pela opção "Já sou cliente / utente", passamos para o *Login*:

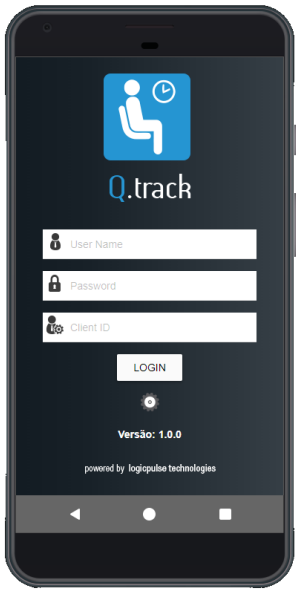

Figura 19 - Ecrã de login

Após login efetuado com sucesso, somos dirigidos para o *Dashboard* onde nos é apresentado o estado de atendimento dos serviços disponíveis no local onde o utilizador se encontra.

Figura 20 - Dashboard

Neste ecrã é possível tirar senha:

Figura 21 - Dashboard

A partir do *Dashboard* temos acesso a um menu com algumas funcionalidades, nomeadamente, Sair, Senhas, Histórico de Senhas, Configurações e Sobre.

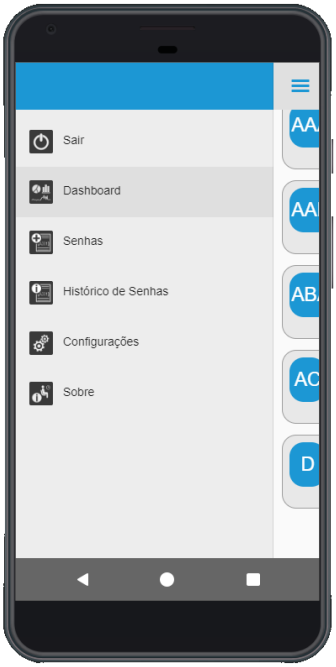

Figura 22 - Opções Menu

Se escolhermos o menu "Senhas", somos levados para o seguinte ecrã:

Figura 23 - Ecrã de Senhas

Neste ecrã é possível percorrer os vários serviços até ao pretendido para retirar a senha. No exemplo da Figura 23, temos serviços de vários níveis, ou seja, temos o Atendimento Geral 1 e o Atendimento Prioritário. Ao clicarmos no primeiro é-nos apresentado os seus serviços "filhos" (exemplo da Figura 24) e por aí adiante até ao final da "cadeia" de serviços (Figura 25).

Figura 24 - Navegação entre níveis de serviços

Figura 25 - Ecrã de tirar senha

Caso o serviço que escolhemos seja um serviço “sem filhos”, é-nos apresentada a opção de retirar senha:

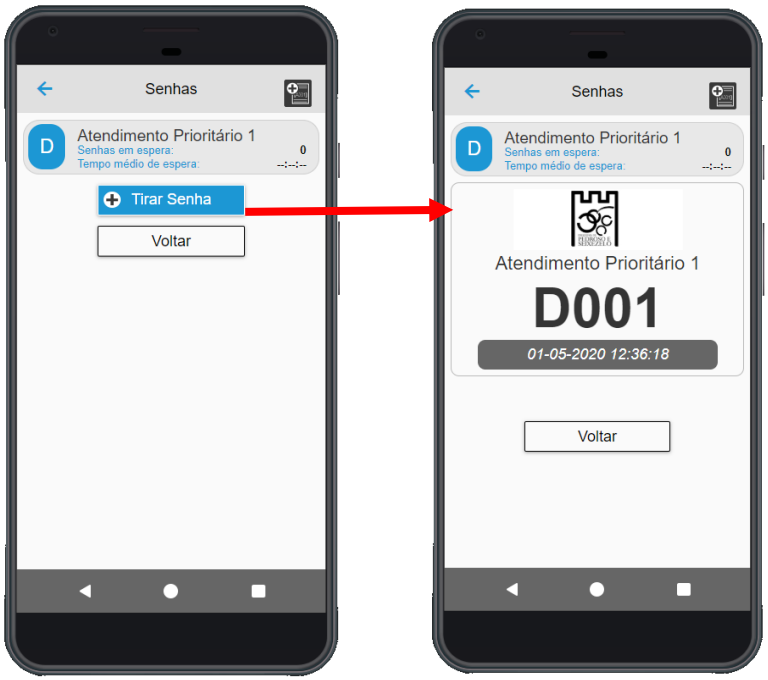

Figura 26 - Ecrã para retirar senha

Se escolhermos o menu "Histórico de Senhas", somos levados para o seguinte ecrã:

Figura 27 - Ecrã Histórico de Senhas

Se escolhermos o menu "Configurações", somos levados para o seguinte ecrã:

Figura 28 - Ecrã de Configurações

Se escolhermos o menu "Sobre", somos levados para o seguinte ecrã:

Figura 29 - Ecrã de Sobre

Anexo B (Código de uma funcionalidade)

Cada "ação" na aplicação que requer ligação ao *web-service*, é efetuada da seguinte maneira:

- aplicação (exemplo da função para retirar senha):

```
function getTicket(idS) {
  loadPanelVisible(true);
  loadPanelMessage(DevExpress.localization.message.format("Loading"));

  QTrack_Mobile.data.getTicket(idS).done(function (result) {
      if (result !== "") {
          if (result[0].serviceName != "MAX_TICKET_LIMIT_REACHED" ||
result[0].serviceName != "SERVICE_INACTIVE") {
                ticketLogo(result[0].ticketLogo);
                ticketID(result[0].idWaitingClient);
                serviceNameT(result[0].serviceName);
                ticketNumberT(result[0].ticketNumber);
                line1T(result[0].line1);
                line2T(result[0].line2);
                line3T(result[0].line3);
                line4T(result[0].line4);
                date(result[0].date);

                $("#ticket").css("display", "block");
                $("#ticketLogo").attr("src", ticketLogo());
                $("#ticketLogo").css("display", "block");
                //$("#l1L").css("display", "block");
                $("#line1T").css("display", "block");
                $("#sNameL").css("display", "block");
                $("#sNameT").css("display", "block");
                $("#nTicketL").css("display", "block");
                $("#tNumberT").css("display", "block");
                $("#l1L").css("display", "block");
                $("#l2L").css("display", "block");
                $("#l3L").css("display", "block");
                $("#l4L").css("display", "block");
                $("#dateL").css("display", "block");
                $("#date").css("display", "block");

                ticketsStore.insert({
                    idTicketHistory: idTicketHistory(),
                    ticketLogo: ticketLogo(),
                    ticketID: ticketID(),
                    idService: idS,
                    serviceName: serviceNameT(),
                    ticketNumber: ticketNumberT(),
                    line1: line1T(),
                    line2: line2T(),
                    line3: line3T(),
                    line4: line4T(),
                    date: date(),
                    idLocation: idLocation(),
                    locationName: locationName()
                });

                idTicketHistory(idTicketHistory() + 1);

                loadPanelVisible(false);
```

```
                    ticketPrint(true);
                }
                else if (result[0].serviceName == "MAX_TICKET_LIMIT_REACHED") {
                    showToast(DevExpress.localization.message.format("messageTo
astMaxTicketLimitReached"), "info");
                    QTrack_Mobile.app.navigate('Dashboard', { root: true });
                    loadPanelVisible(false);
                }
                else if (result[0].serviceName == "SERVICE_INACTIVE") {
                    showToast(DevExpress.localization.message.format("messageTo
astServiceInactive"), "info");
                    QTrack_Mobile.app.navigate('Dashboard', { root: true });
                    loadPanelVisible(false);
                }
            }

        });
    }
```

- aplicação (exemplo da função de ligação ao *web-service* para retirar senha):

```
    function getTicket(idService) {
        var current = null;
        var result = $.Deferred();
        var Done = 0;

        setTimeout(function () {
            if (Done === 0) {
                var properties = [];
                result.resolve(properties);
                return result.promise();
            }
        }, TIMEOUT);

        var url = QTrack_Mobile.app.url +
'qtrackMobile.aspx?type=getTicket&idService=' + idService + '&SESSION_TOKEN=' +
QTrack_Mobile.app.tokenValue;
        $.getJSON(url, function (data) {
            Done++;
            var properties = [];
            $.each(data, function (key, value) {
                properties.push(value);
            });
            result.resolve(properties);
        });
        return result.promise();
}
```

- *web-service*

```
namespace WebServiceQTrack.Mobile
{
    public partial class qtrackMobile : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            Service1 wsService = new Service1();
            ServicesData dbServices = new
ServicesData(ConfigurationManager.AppSettings["MyDbConnSQLServer"]);
```

```
            QueueManage dbQueueManage = new
QueueManage(ConfigurationManager.AppSettings["MyDbConnSQLServer"]);
            LocationData dbLocationData = new
LocationData(ConfigurationManager.AppSettings["MyDbConnSQLServer"]);
            UsersData dbUsersData = new
UsersData(ConfigurationManager.AppSettings["MyDbConnSQLServer"]);
            AIRC_Integration dbAIRC_Integration = new
AIRC_Integration(ConfigurationManager.AppSettings["MyDbConnSQLServer"]);
            AuxTables dbAuxTables = new
AuxTables(ConfigurationManager.AppSettings["MyDbConnSQLServer"]);
            AlertsQuery dbAlertsQuery = new
AlertsQuery(ConfigurationManager.AppSettings["MyDbConnSQLServer"]);

            string json = "{\"name\":\"nok\"}";
            try
            {
                string type = Globals.FilterRequest(Request.QueryString["type"]);
                string _Session = Globals.FilterRequest(Request.QueryString["SESSION_TOKEN"]);

                string result = "[";

                if (type == "checkLicence") …
                else if (type == "getServices") …
                else if (type == "getServicesDetails") …
                 else if (type == "getTicket") …
                 else if (type == "getLocation") …
                 else if (type == "getServicesLocation") …
                 else if (type == "getLogin") …
                 else if (type == "changePasswordFromMobile") …
                 else if (type == "getListWaitingTickets") …
              }
              catch (Exception exx)
              {
                   Parameters.Log.Error("exx: " + exx.Message, exx);
              }
              Response.Clear();
              Response.ContentType = "application/json; charset=utf-8";
              Response.AddHeader("Access-Control-Allow-Origin", "*");
              Response.Write(json);
            }
```