Instituto Superior de Engenharia do Porto

PROTÓTIPO EUROBOLSO

Diogo Marques Pereira

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas Gráficos e Multimédia

Orientador: Doutora Maria Fátima Coutinho Rodrigues

Júri:

Presidente:

Doutor João Paulo Jorge Pereira, ISEP

Vogais:

Doutor Luís Miguel Moreira Lino Ferreira, ISEP

Doutor Maria de Fátima Coutinho Rodrigues, ISEP

Porto, outubro de 2014

Resumo

O projeto idealizado para a realização da presente tese de mestrado tem como finalidade o desenvolvimento de uma aplicação móvel para o sistema *Android*. Esta aplicação permitirá que, através da passagem do dispositivo móvel por um leitor NFC, seja possível realizar apostas no jogo Euromilhões da Santa Casa, onde os dados ficam gravados numa conta associada a cada jogador. Esta aplicação terá ainda muitas outras funcionalidades que permitirão criar chaves de jogo, gerir o cartão individual de cada jogador, consultar os prémios e outras informações do jogo.

A realização deste projeto está dividida em três fases. A primeira fase consistiu na aquisição de todo o material necessário e estabelecimento da comunicação entre o dispositivo móvel e o *desktop*, por intermédio da tecnologia NFC. A segunda fase centrou-se no desenvolvimento da aplicação móvel e do servidor *web*, onde foram integradas as várias funcionalidades. Estabeleceu-se também a comunicação entre estes dois sistemas. Na terceira e última fase, foi realizada a criação da aplicação *desktop*, capaz de interagir por intermédio da tecnologia NFC, com a aplicação móvel, possibilitando a comunicação entre os dois sistemas.

Palavras-chave: Aplicação móvel, *Android*, *Near Field Communication*, Euromilhões.

Abstract

The project conceived for the realization of this thesis aims to develop a mobile application for Android system. This application will allow, through the passage of the mobile device by an NFC reader, to place bets on the *Euromillions* game of *Santa Casa*, where the data are written to an account associated with each player. This application will also have many other features that will enable the creation of game keys, manage the individual card for each player, check the money prizes and other game information.

This project is divided into three phases. The first phase consisted of the acquisition of all the necessary materials and establishment of communication between the mobile device and the desktop via the NFC technology. The second phase focused on the development of the mobile application and the web server, where several features have been integrated. It also established the communication between these two systems. In the third and final phase, the desktop application was created, which will enable the interaction through NFC technology with the mobile application, allowing communication between the two systems.

Keywords: Mobile Application, Android, Near Field Communication, *Euromillions*.

Agradecimentos

Várias pessoas, de alguma forma, contribuíram para que esta tese de mestrado fosse realizada. Deste modo, não posso deixar de manifestar o meu apreço e sincero agradecimento por toda a ajuda.

Gostaria de dirigir as minhas primeiras palavras de agradecimento à minha orientadora, a Doutora Fátima Rodrigues, pela grande capacidade de trabalho, empenhamento e disponibilidade, tendo sido um grande apoio para a realização da presente tese de mestrado.

A todos os meus amigos e à minha namorada Ana, por todas as palavras de incentivo e pelo apoio em momentos de desânimos e sucessos, pela confiança e pela valorização do meu trabalho.

Por fim, gostaria ainda de deixar um agradecimento especial aos meus pais, por todo o apoio incondicional, incentivo e amizade sempre demonstrados ao longo do meu percurso académico e no meu crescimento a nível pessoal.

Índice

Lista de Figuras

Acrónimos

Lista de Acrónimos

ACS	Advanced Card Systems
ICD	International Data Corporation
JARs	*Java Archives*
JSON	*JavaScript Object Notation*
ISEP	Instituto Superior de Engenharia do Porto
KB/s	Kilobits por segundo
LLCP	*Logical Link Ccontrol Protocol*
MHz	Megahertz
NFC	*Near Field Communication*
NFCF	*Data Exchange Format*
REST	*Representational State Transfer*
RFID	Identificação por Radiofrequência
UL	*Ultralights*
ULC	*Ultralights* Cs
URI	*Uniform Resource Identifiers*
XML	*Extensible Mark-up Language*
3DES	*Triple Data Encryption Standard*
TfL	*Transport for London*
RDBMS	*Relational Database Management System*
REST	*Representational State Transfer*
HTTP	*Hyper Transfer Protocol*
POS	*Point of Sale*

1 Introdução

A realização desta tese surge no âmbito do mestrado em Sistemas Gráficos e Multimédia referente ao curso de Engenharia Informática, ministrado no Instituto Superior de Engenharia do Porto (ISEP), cujo tema assenta no desenvolvimento de uma aplicação móvel capaz de substituir o habitual boletim do jogo Euromilhões.

O presente documento tem como objetivo explicar, de forma simples, clara e concisa, todos os passos e metodologias utilizadas na realização do referido protótipo.

1.1 Enquadramento e caracterização geral do problema

Uma aplicação móvel consiste na criação de um *software* que poderá ser instalado num dispositivo eletrónico móvel, como *tablets*, *smartphones*, entre outros. Os utilizadores podem adquirir estas aplicações através do *download* a partir de lojas *on-line*, *como Google Play, App Store*, etc.

Ao longo do tempo tem-se verificado um grande aumento do número de *downloads* de aplicações móveis, que está diretamente relacionado com a crescente utilização de *smartphones*. Deste modo, a utilização de aplicações móveis no dia-a-dia é uma realidade cada vez mais presente e de importância vital no mercado atual.

O uso de telemóveis no quotidiano é indispensável, como tal, o desenvolvimento de novas ferramentas de carácter pessoal e profissional acarreta inúmeras vantagens para os utilizadores, auxiliando as tarefas do dia-a-dia. Determinar o número total de aplicações que existem no mercado é bastante complexo, contudo, de acordo com a *Canalys* estima-se que, no ano de 2013, as duas maiores lojas *online* (*Apple Store* e *Google Play*) contivessem mais de 800.000 aplicações disponíveis cada.

Existem diferentes tipos de aplicações de acordo com a sua finalidade, podendo-se integrar plataformas de informação, entretenimento ou de prestação de serviços. A criação destas

aplicações tem como principais objetivos a simplicidade, a rapidez e o fácil acesso a todos os utilizadores, criando uma maior proximidade com o público-alvo.

O Euromilhões é o jogo que abrange alguns países da Europa, sendo um dos jogos que mais atrai os portugueses. Para jogar na Lotaria do Euromilhões pode-se optar pelo preenchimento e compra do boletim de jogo *online* ou a um revendedor aderente. Face à crescente utilização de novas tecnologias, torna-se imprescindível haver uma constante inovação e criação de novas funcionalidades vantajosas para os jogadores surgindo, neste contexto, o desenvolvimento da aplicação "EuroBolso", a qual pretende possibilitar jogar no Euromilhões através de uma aplicação móvel.

1.2 Motivação e Objetivos Propostos

A principal motivação para a realização do presente trabalho prende-se com a criação de uma aplicação simples que dê a possibilidade, ao utilizador, de poder jogar na Lotaria do Euromilhões, a partir de uma aplicação móvel, onde ficam registados todos os jogos efetuados.

Com a aplicação "EuroBolso" o utilizador não necessitará de um boletim de jogo impresso pois todos os jogos efetuados ficariam registados numa plataforma *online* e no dispositivo móvel, aumentando a comodidade do utilizador e eliminando o risco de perda do boletim de jogo. Terá ainda a possibilidade de efetuar a consulta de vários dados referentes ao jogo, o que facilitará o seu acesso.

A solução final centrar-se-á na utilização da tecnologia NFC, que permitirá realizar as apostas simplesmente através da passagem do dispositivo móvel no leitor NFC no posto de venda.

1.3 Resultados Esperados

No final deste projeto espera-se que a aplicação móvel seja uma mais-valia para os apostadores do Euromilhões, que utilizem o sistema operativo *Android*. Num mundo onde a tecnologia está em constante crescimento, soluções simples e acessíveis que consigam melhorar a gestão de tempo dos utilizadores é uma demanda importantíssima.

Deste modo, a criação de uma aplicação que permita realizar apostas e consultar inúmeras informações alusivas a este jogo mostra-se bastante conveniente, uma vez que a tecnologia está cada vez mais presente no quotidiano da população em geral. No final, espera-se que a aplicação "Eurobolso" permita realizar o registo da chave de jogo em postos com a tecnologia NFC disponível, sendo esta a característica que diferencia a presente aplicação face às demais existentes.

1.4 Organização do documento

Este documento está dividido em 5 capítulos:

- ✓ **Capitulo 1 – Introdução:** é feito um enquadramento do projeto e são descritas as motivações para a realização da presente tese de mestrado;

- ✓ **Capitulo 2 – Estado da Arte:** é descrita a evolução do sistema operativo *Android* bem como a importância e crescimento das aplicações móveis. É efetuada uma abordagem sobre as aplicações móveis pré-existentes do Euromilhões. É também abordada a tecnologia mais importante para a realização deste projeto, a tecnologia *Near Field Communication*, explorando as diversas formas de funcionamento e de aplicabilidade desta tecnologia;

- ✓ **Capitulo 3 – Desenho Conceptual:** é apresentado o protótipo do projeto bem como todo o *hardware* necessário à sua implementação;

- ✓ **Capitulo 4 – Sistema Desenvolvido:** inicialmente é descrita a arquitetura de todo o sistema e, em seguida, são detalhados os passos da sua implementação. É também apresentado o *layout* final da aplicação Eurobolso juntamente com as suas funcionalidades;

- ✓ **Capitulo 5 – Conclusões e Trabalho Futuro:** é realizado um resumo do processo percorrido ao longo do projeto sendo ainda, apresentadas algumas ideias que foram surgindo que poderão ser trabalhadas no futuro.

2 Estado da Arte

2.1 História e Evolução do Sistema Operativo Android

O *Android* é um sistema operativo baseado em *Linux* e desenvolvido para ser utilizado em dispositivos móveis. Em outubro de 2003 foi fundada, em Palo Alto (Califórnia), a *Android Inc.* por *Cris White, Nick Sears, Rich Miner* e *Andy Rubin*. O principal objetivo da *Android Inc.,* segundo *Andy Rubin*, era a "criação de dispositivos móveis mais inteligentes e cientes da sua localização, que atendam às preferências do seu utilizador".

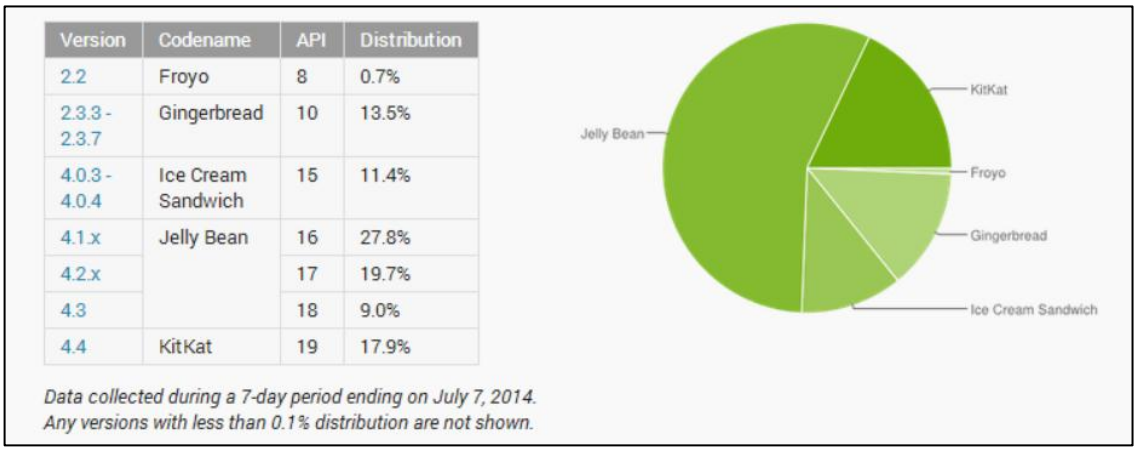
Em junho de 2005 a *Google* compra a *Android Inc.* que, na altura, era apenas uma *startup* bastante promissora. Com esta aquisição, a *Google* pretendia entrar no mercado dos dispositivos móveis, mercado este em forte expansão, de modo a aumentar a divulgação dos seus novos produtos.

Cerca de 3 anos mais tarde, em 2008, com a parceria da *T-Mobile*, empresa alemã responsável pelo fabrico de telemóveis fundada em 2000, é lançado o primeiro *smartphone* com o sistema operativo *Android*, denominado G1.

Hoje em dia, os dispositivos *Android* não se limitam apenas aos *smartphones*. O sistema operativo está presente em *tablets*, televisões que fornecem serviços de Internet, netbooks, relógios e óculos. Os *tablets* têm vindo a ganhar uma especial preponderância no mercado dos dispositivos móveis ao longo dos últimos anos. Sendo considerados um meio-termo entre um computador e um *smartphone*, o número de utilizadores dos dispositivos *Android* aumenta cada vez mais.

Paralelamente ao desenvolvimento dos dispositivos móveis, onde os fabricantes investem milhões de euros na melhoria das suas capacidades, tanto ao nível das capacidades de processamento como de memória, o *Android* foi sofrendo sucessivos melhoramentos com o lançamento de novas versões caracterizadas, à exceção das duas primeiras versões, por terem nomes de sobremesas.

A Figura 1, abaixo representada, demonstra as várias versões do *Android*, bem como a quota de mercado que cada uma representa na atualidade. Na Figura 1, estão apenas representados dados desde a versão 2.2, pois esta versão foi a primeira a incluir a aplicação *Google Play Store*, permitindo assim recolher esta análise. Contudo, em Agosto de 2013 a *Google* apontava para cerca de 1% de dispositivos a recorrer a versões anteriores à 2.2.



Version	Codename	API	Distribution
2.2	Froyo	8	0.7%
2.3.3 - 2.3.7	Gingerbread	10	13.5%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	11.4%
4.1.x	Jelly Bean	16	27.8%
4.2.x		17	19.7%
4.3		18	9.0%
4.4	KitKat	19	17.9%

Data collected during a 7-day period ending on July 7, 2014.
Any versions with less than 0.1% distribution are not shown.

Figura 1 – Cota do mercado das versões *Android* nos dispositivos móveis de 2014

De acordo com um estudo realizado pela *International Data Corporation* (IDC), apenas no primeiro trimestre do ano de 2014 foram vendidos cerca de 287,8 milhões de *smartphones* em todo o mundo, constituindo um aumento de 31,5% face aos 219,0 milhões de unidades vendidas no período homólogo do ano transato.

O mesmo estudo acrescenta ainda que os *smartphones* representam, neste momento, 63,1% do mercado da telefonia móvel no mundo, contrariamente aos 50,7% no ano anterior.

A Figura 2 indica que no primeiro trimestre de 2014 a quota de mercado nos *smartphones* é liderada pelo sistema operativo *Android*, representando 81,1% dos dispositivos, seguido pelo iOS, sistema operativo proprietário da *Apple*, presente nos *iPhones* que ocupa 15,2% do mercado. Pode-se ainda verificar, que o *Windows Phone* sistema operativo da *Microsoft* representa 2,7% seguido pelo *BlackBerry OS*, proprietário da *BlackBerry*, com 0,5%. Os restantes sistemas operativos, como é o caso do *Symbian OS*, entre outros, apenas representam 0,6% do mercado.

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
Q1 2014	81.1%	15.2%	2.7%	0.5%	0.6%
Q1 2013	75.3%	17.1%	3.2%	2.9%	1.5%
Q1 2012	59.2%	23.0%	2.0%	6.3%	9.5%
Q1 2011	36.1%	18.3%	1.2%	13.6%	30.8%

Figura 2 – Quota do mercado dos *smartphones* por sistema operativo

De destacar que, ao longo dos últimos 4 anos, o único sistema operativo móvel a apresentar contantes evoluções na quota de mercado foi o *Android*, passando dos 36,1% no ano de 2011 para os atuais 81,1%.

2.2 A Importância das Aplicações Móveis

As aplicações móveis ganham cada vez mais preponderância na maneira como as pessoas abordam a tecnologia e usam os seus dispositivos móveis.

É evidente que, na atualidade, grande parte da população não passa sem o recurso aos dispositivos móveis pois, por intermédio destes, é possível aceder a todo o tipo de informação em qualquer lugar, à distância de poucos cliques. Esta constante exigência na busca pela informação por parte de utilizadores levou a que empresas e programadores apostassem em força no mercado móvel.

O conceito de "posto de trabalho" deixou de ser suficiente. Um individuo ou empresa requer acesso imediato às suas aplicações pessoais ou profissionais, e à possibilidade de tomar decisões, em qualquer lugar e a qualquer hora, com a máxima flexibilidade e acessibilidade.

Estas premissas fortalecem a ideia de que, na atualidade, as empresas encaram a mobilidade como uma das principais prioridades nos seus negócios. Segundo um estudo da CIONET, sete em cada dez organizações contam já com uma lista definida de iniciativas de mobilidade, uma tendência que deverá acentuar-se.

Segundo dados fornecidos pela *Flurry*, em abril de 2014**,** o uso de aplicações móveis nos *smartphones* ultrapassa largamente o uso dos *browsers* móveis. Com cerca de uma média de 2 horas e 42 minutos de uso diário, os utilizadores gastam 86% do tempo com o recurso a aplicações móveis e apenas 14% no *browser* do dispositivo. Comparativamente com o ano 2013 verificou-se um aumento no uso das aplicações de 6%, uma tendência que deverá continuar a aumentar.

A forte aposta que se tem vindo a sentir no mercado das aplicações móveis, justifica por si só os dados apresentados no estudo anterior. Hoje em dia, a oferta é vasta e abrange as mais diversas áreas contudo, a procura constante por parte dos utilizadores da satisfação das suas necessidades não pára de crescer.

2.3 Aplicações Móveis e o Euromilhões

Longe estão os dias, onde se aguardava a retransmissão do sorteio de jogos da sorte na televisão ou se esperava pelo dia seguinte para verificar quais as chaves sorteadas e, consequentemente, se os apostadores foram os felizes contemplados com algum dos prémios. Hoje em dia, num mundo onde a tecnologia impera e o tempo escasseia, tornou-se

necessário criar automatismos que permitam uma rápida análise sobre os resultados de forma a facilitar o acesso aos dados referentes às apostas.

A criação de aplicações que permitam ao utilizador criar uma chave aleatória, verificar o histórico das suas apostas e compará-las com os resultados dos sorteios, são alternativas viáveis para o aumento do comodismo por parte do apostador, libertando-o de todo o processo de análise, por intermédio de apenas alguns cliques.

No espetro nacional destacam-se duas aplicações que potenciam o que foi descrito anteriormente. A aplicação EuroMilhões criada pela *CodeLineStudios* e a aplicação EuroMilhões criada por Joel Brito. Nos seguintes subcapítulos será feita uma análise e descrição destas aplicações.

2.3.1 Aplicação "Euromilhões" (*CodeLineStudios*)

A aplicação EuroMilhões, criada pela *CodeLineStudios*, possui um *layout* bastante atrativo e uma boa usabilidade para o utilizador (ver Figura 3), apresentado cores atrativas e bons contrastes entre as mesmas, de modo a facilitar a leitura dos dados. Paralelamente, a navegação na aplicação é bastante fluida, simplificando assim o seu uso por parte do utilizador.

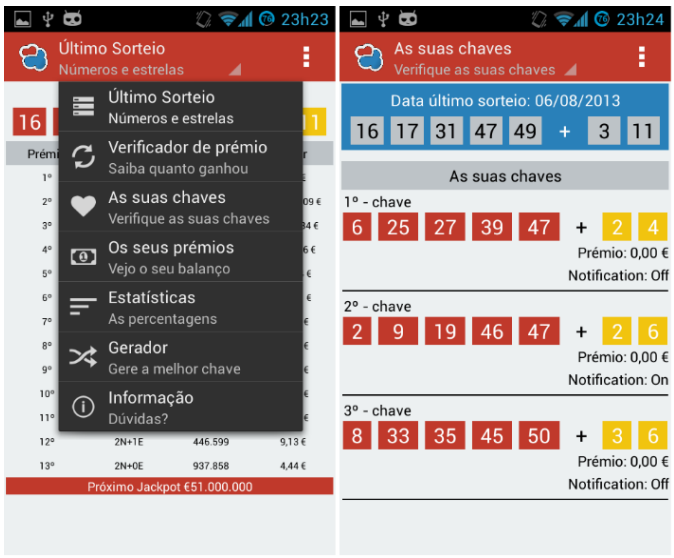


Figura 3 - Aplicação "EuroMilhões" de *CodeLineStudios*

O aplicativo permite ao utilizador usufruir das seguintes funcionalidades:

- ✓ Verificar o último sorteio com tabela de prémios;
- ✓ Verificador de prémio;
- ✓ Guardar as suas chaves favoritas e ativar notificações sobre elas;
- ✓ Verificar quanto já ganhou, no total, no Euromilhões;

- ✓ Visualizar estatísticas dos números e estrelas do último sorteio;
- ✓ Gerador de chaves;
- ✓ Multilíngues;
- ✓ Diferentes tipos de moeda.

Apesar de todos os aspetos positivos acima referidos, a aplicação apresenta algumas lacunas, nomeadamente ao nível das apostas múltiplas, dado que não permite a sua inserção, verifica-se ainda um atraso bastante significativo na atualização da lista de prémios, pois só se encontra disponível no dia seguinte ao do sorteio.

2.3.2 Aplicação "Euromilhões" (Joel Brito)

A aplicação criada por Joel Brito apresenta uma interface carregada bem como uma usabilidade com algum grau de complexidade, não potenciando assim todas as suas funcionalidades.

Entre as várias funcionalidades, muitas delas comuns à aplicação descrita anteriormente, destacam-se as seguintes:

- ✓ Possui lista de prémios em todos os sorteios;
- ✓ Apresenta os resultados de todos os sorteios, desde o início do Euromilhões;
- ✓ Permite consultar estatísticas dos números e estrelas, com possibilidade de ordenação, de todos os sorteios do Euromilhões;
- ✓ Possibilidade de chaves múltiplas.

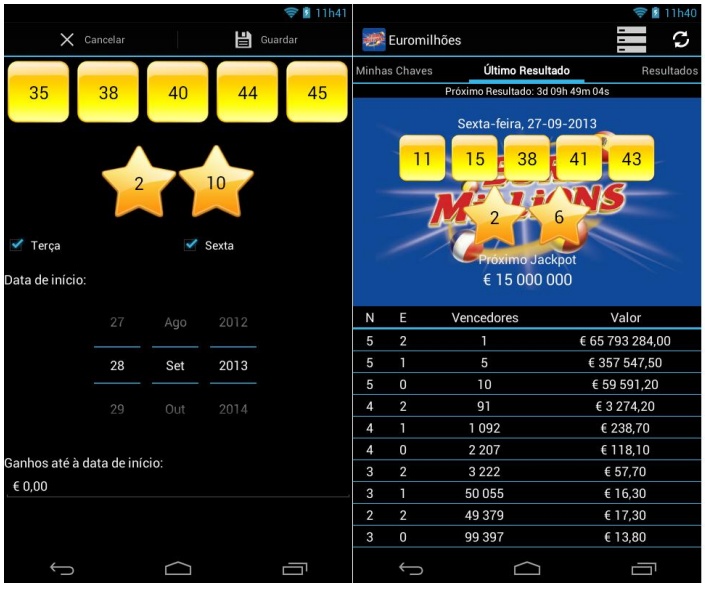


Figura 4 - Aplicação "Euromilhões" de Joel Brito

2.4 Near Field Communication

O nascimento do NFC remonta para o início dos anos 80 sob a forma de identificação por radiofrequência (RFID), mas com um alcance de comunicação mais curto, para fins de segurança. Muitos cientistas e investigadores contribuíram para o nascimento do RFID, mas foi *Charles Walton* que, em 1983, conseguiu patentear o primeiro objeto com recurso a esta tecnologia.

Em 2004, a *Sony*, a *Nokia* e a *Phillips* juntaram-se e criaram o Fórum NFC. O grupo foi crescendo ao longo dos anos, contando agora com mais de 160 membros, mantendo o seu foco em promover a segurança, facilidade de uso e popularidade. Com isto, foi-se educando empresas sobre a tecnologia e criando normas que permitam que o NFC possa funcionar entre dipositivos de marcas e modelos diferentes.

Dois anos depois, surgiu o primeiro conjunto de especificações para uma *tag* NFC. Foi então que apareceram os primeiros "cartazes inteligentes", capazes de armazenar informações. No decorrer do mesmo ano, a *Nokia* lançou o primeiro telemóvel a possuir esta tecnologia, o *Nokia* 6131, na Figura 5.

Figura 5 – *Nokia* 6131

Com o decorrer dos anos, as especificações para esta tecnologia foram crescendo passando originalmente de métodos de pagamento para a possibilidade de partilha de vídeos, músicas, *links* ou mesmo convites para jogos. O primeiro telemóvel com o sistema operativo *Android* a possuir a tecnologia NFC, foi lançado em 2010, o *Samsung Nexus S*.

2.4.1 Modo de Funcionamento

O NFC distingue-se de outras tecnologias sem fios, como o *Bluetooh* ou o *Wi-Fi*, pela simplicidade de utilização bastando a simples aproximação entre dois dispositivos para que a conexão se realize. Depreende-se por dispositivos tudo o que contenha um simples *chip* NFC, desde *smartphone*, *tablets*, bilhetes de metro, cartazes, etc…

Existem diversas gamas de *chips*, cada uma com características próprias. Variam na capacidade de armazenamento de dados, se possuem ou não a capacidade de leitura e escrita e outras características mais específicas, como é o caso de poderem ser instalados no interior ou exterior de um objeto, ou mesmo de serem soldados a um objeto de metal.

Abaixo estão descritos os vários tipos de *chips* NFC, bem como algumas das suas principais características técnicas.

- ✓ ***Ultralights* (UL)** – Informação adequada para um número de telefone ou URL curto (64 *bytes*);
- ✓ ***Ultralight Cs* (ULC)** – Tem um pouco mais de capacidade de armazenamento de dados face ao UL e é bom para um URL longo ou um cartão de contacto pequeno (192 *bytes*);
- ✓ ***Standard 1K*** – Podem armazenar mais dados para um cartão de contacto grande, mas são mais substancialmente dispendiosos (1024 *bytes*);
- ✓ ***NTAG203*** – Semelhante ao ULC, mas é um novo tipo de chip com maior capacidade de resposta (168 *bytes*);
- ✓ ***Anti-Metal Tags*** – Podem ser soldados em superfícies metálicas.

Anti-Metal Tags	Classic 1K (S50) - Anti Metal	Mifare	38 x 25mm	Rectangular	White	1024 Bytes
	Classic 1K (S50) - Anti Metal	Mifare	35mm	Square	White	1024 Bytes
	Ultralight - Anti Metal	Mifare	38 x 25mm	Rectangular	White	64 Bytes
	Ultralight - Anti Metal	Mifare	35mm	Square	White	64 Bytes
Business Cards	Mifare Classic 1K (S50)	Mifare	86 x 54mm	Rectangular	White	1024 Bytes
	Ultralight	Mifare	86 x 54mm	Rectangular	White	64 Bytes
Normal Tags	Classic 1K (S50)	Mifare	25mm	Circular	White	1024 Bytes
	Classic 1K (S50) - "Bullseye"	Mifare	38mm	Circular	White	1024 Bytes
	Classic 1K (S50)	Mifare	45mm	Circular	White	1024 Bytes
	Classic 1K (S50)	Mifare	55 x 17mm	Rectangular	White	1024 Bytes
	Classic 1K (S50)	Mifare	35 x 35mm	Square	White	1024 Bytes
	Classic 1K (S50)	Mifare	50 x 50mm	Square	White	1024 Bytes
	Ultralight	Mifare	25mm	Circular	White	64 Bytes
	Ultralight	Mifare	45mm	Circular	White	64 Bytes
	Ultralight	Mifare	38 x 25mm	Rectangular	White	64 Bytes
	Ultralight	Mifare	35 x 35mm	Square	White	64 Bytes
	Ultralight	Mifare	50 x 50mm	Square	White	64 Bytes
	Ultralight "C"	Mifare	81 x 49mm	Rectangular	White	192 Bytes
	NTAG ULC	NTAG	25mm	Circular	White	192 Bytes
	NTAG ULC	NTAG	38mm	Circular	White	192 Bytes
Key Fobs	Classic 1K (S50)	Mifare	N/A	Teardrop	Red	1024 Bytes
	Classic 1K (S50)	Mifare	N/A	Teardrop	Blue	1024 Bytes
	Ultralight "C"	Mifare	N/A	Teardrop	Red	192 Bytes
	Ultralight "C"	Mifare	N/A	Teardrop	White	192 Bytes
	Ultralight "C" - Jelly	Mifare	N/A	Circular	White	192 Bytes
Wristbands	Mifare Classic 1K (S50)	Mifare	18cm	Circular	White	1024 Bytes
	Mifare Classic 1K (S50)	Mifare	22cm	Circular	White	1024 Bytes
	Ultralight	Mifare	18cm	Circular	Red	64 Bytes
	Ultralight	Mifare	22cm	Circular	Red	64 Bytes

Figura 6 – Tipos de *Chips* NFC

Os tipos de *chips* mais populares são os ULC e NTAG203**.** O ULC é uma evolução do UL, triplicando as capacidades de memória. Para além deste *upgrade*, a segurança foi também um alvo de atenção, incluindo nesta versão Criptografia segundo o padrão 3DES (*Triple Data Encryption Standard*). Apesar de todos estes aspetos positivos, o fator de leitura à distância, que varia de 1 a 10, sofreu um decréscimo, passando do nível 7 para o nível 4.

Os *chips Standard 1k* são aqueles que apresentam as maiores capacidades de memória, apresentando também a possibilidade de Criptografia, neste caso segundo o padrão *Crypto*-1. Embora apresentem enormes taxas de armazenamento, comparativamente com os outros tipos de chips, apresentam um grande défice de incompatibilidade, não podendo ser utilizados em dispositivos móveis e não obedecendo às normas do Fórum NFC T2.

Por fim, os *chips* NTAG203 foram os últimos a ser criados. Apresentam características semelhantes aos ULC, embora não possuam capacidade de Criptografia. Em todo o caso, são os *chips* que apresentam as melhores taxas de performance e as melhores taxas referentes à leitura à distância, atingindo o valor de 9, numa escala de 1 a 10.

	Ultralight	Ultralight C	Standard 1K	NTAG203
Memory Size [1]	64 bytes	192 bytes	1024 bytes	168 bytes
User Memory [2]	46 bytes	137 bytes	716 bytes	137 bytes
URL Length [3]	41 chars	132 chars	256 chars	132 chars
Text Length [4]	39 chars	130 chars	709 chars	130 chars
Mobile Comp. [5]	Yes	Yes	No	Yes
Best Use	Cost effective, short URL, smart poster and general NFC use.	Good capacity, general NFC use.	Recommended for specific high capacity usage and vCards only.	Latest chip, best performance. Good value. RapidNFC recommended.
NFC Forum T2 [6]	Yes	Yes	No	Yes
Serial Number [7]	Yes	Yes	Yes	Yes
Cryptography [8]	No	3DES	Crypto-1	No
ScanStrength [9]	7	4	6	9

Figura 7 – Especificações técnicas dos diferentes tipos de *chips*

A comunicação entre os dispositivos NFC é bastante linear, um dispositivo faz o papel de *Initiator*, responsável por iniciar as comunicações e controlar a troca de dados entre os dois dispositivos e o outro faz o papel de *Target*, devendo responder às solicitações do *Initiator*.

Caso se trate de uma transmissão passiva, então apenas um dos dispositivos (o *Initiator*) gera sinal de radiofrequência, podendo o outro dispositivo (o *Target*) não receber nenhuma carga elétrica. Por sua vez, numa transmissão ativa, ambos os dispositivos geram sinal de rádio.

A comunicação é realizada a uma frequência de 13.56 *Megahertz* (MHz) e a velocidade de transmissão de dados varia entre os 106, 212 e 424 *Kilobits* por segundo (KB/s). A distância máxima entre os dois dispositivos não pode exceder os 10 centímetros, como se pode verificar na Figura 8.

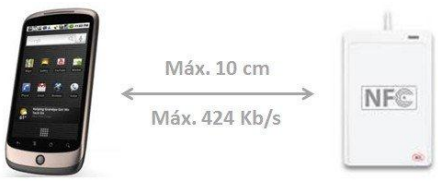


Figura 8 – Alcance/velocidade de transmissão do NFC

A tecnologia NFC permite que os dispositivos operem três modos distintos: a emulação de cartão, emulação de leitor e em modo *Peer-to-Peer*, seguidamente descritos.

Emulação de Cartão - Neste modo de operação, o dispositivo NFC passa por ser um cartão inteligente, de forma a que o dispositivo leitor não consiga distinguir se se trata de um cartão físico, ou um dispositivo eletrónico, a simular um cartão, de acordo com a Figura 9.

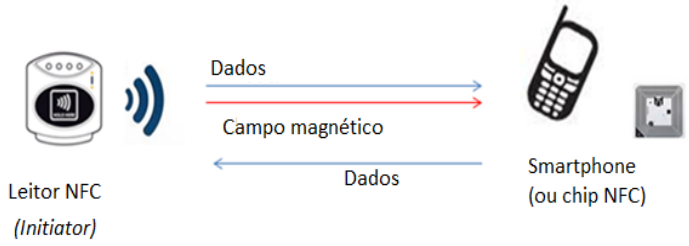


Figura 9 – Modo Emulação de Cartão

Emulação de Leitor - No modo de emulação de leitor, tem-se por base a transmissão passiva onde o dispositivo se transforma num leitor, podendo ler e escrever sobre os dados contidos num outro dispositivo, representado na Figura 10.

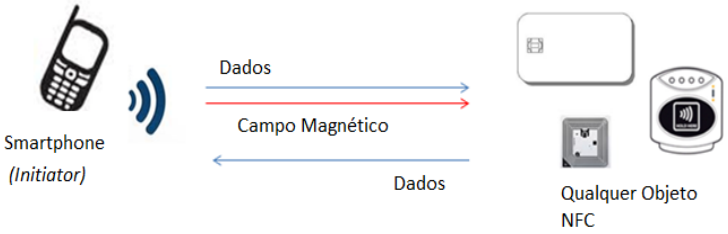


Figura 10 – Modo de Leitura

***Peer-to-Peer* -** O modo de operação *Peer-to-Peer* tem por base a transmissão ativa, pois trata-se de uma troca bidirecional de informação entre dois dispositivos, ou seja, cada um pode receber e/ou enviar dados entre si, como mostra a Figura 11.

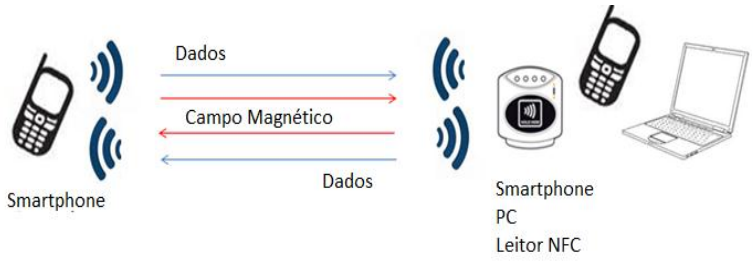


Figura 11 – Modo *Peer-to-Peer*

2.4.2 Protocolos NFC

O NFC Fórum especificou um formato de encapsulamento de mensagens denominado de NFC *Data Exchange Format* (NDEF), para troca de informação entre dispositivos NFC, leitores ou *tags*, desde que estes elementos sigam a especificação do NFC Fórum.

2.4.2.1 NDEF

NDEF é um formato de mensagem binária para troca de carga aplicacional ou *payload*, de qualquer tipo ou tamanho, dentro de uma única mensagem. O *payload* é descrito por um tipo, sendo o seu tamanho um identificador opcional. Os diferentes tipos possíveis são os URI (*Uniform Resource Identifiers*), MIME e tipos específicos do NFC.

O tamanho varia entre um número inteiro negativo e um inteiro positivo, que indica o número de *bytes* do *payload*. Para um *payload* mais pequeno, existe um registo mais pequeno e compacto. Os *payloads* podem incluir mensagens aninhadas ou cadeias de blocos ligados com tamanho desconhecido no momento em que os dados são gerados. O NDEF é apenas um formato de mensagem que não permite guardar informações sobre as ligações ou sobre os circuitos lógicos.

A especificação NDEF define o formato da estrutura de dados NDEF, definindo quais as regras para criar uma mensagem NDEF válida, bem como para uma coleção de registos NDEF continuo, intactos e ordenados. O NDEF assume um protocolo fundamental e confiável e não especifica como é que os dados têm que ser transferidos entre entidades NFC. O tamanho das mensagens é de 2^32-*1Bytes*.

Baseado no NDEF, o NFC Fórum desenvolveu 4 diferentes definições de tipos de registo NFC, nomeadamente o URI, o Texto, o Controlo Genérico e o *Smart Poster*. O intuito é possibilitar que os programadores consigam criar as suas aplicações da forma correta, utilizando um conjunto de regras e formatos gratuitos e disponíveis para todos.

O tipo de dados num registo e o seu tamanho ficam guardados num cabeçalho (*Header)* anexado ao *payload*. O cabeçalho inclui um campo “tipo” que identifica o tipo de *payload*, e o tamanho do *payload* indica o número de octetos. Pode ainda ser utilizado um identificador opcional que permite que aplicações para o utilizador consigam identificar o tipo de *payload* transportado num registo NDEF.

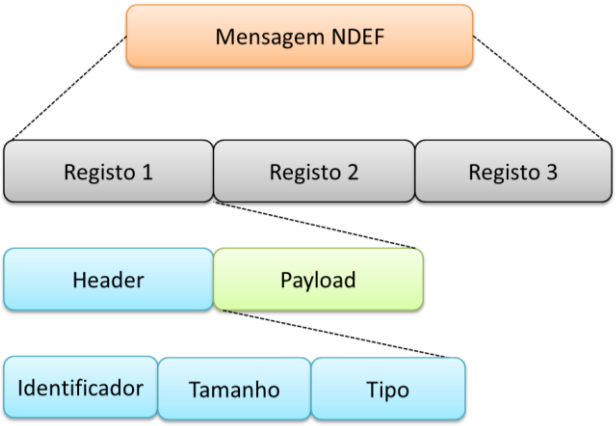


Figura 12 – Mensagem NDEF

2.4.2.2 *Record Type Definition* (RTD)

A Figura 13 representa de forma resumida os diferentes registos NDEF e o tipo de carga que cada registo pode conter.

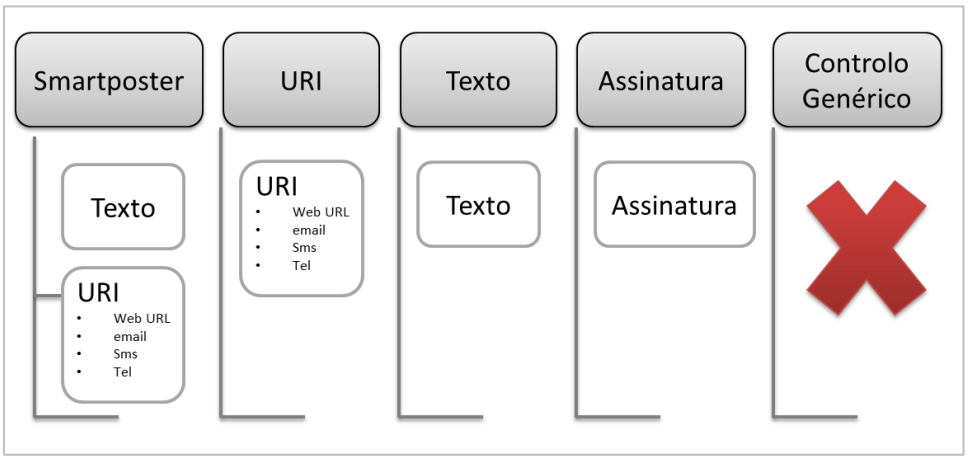


Figura 13 – Tipos de registo NDEF

- ✓ **Texto** - Fornece uma maneira eficiente de armazenar sequências de texto em vários idiomas usando o mecanismo de RTD e formato NDEF;
- ✓ **URI** - Fornece uma maneira eficiente de armazenar URI usando o mecanismo de RTD e formato NDEF;
- ✓ ***Smartposter*** – Define um NFC *Forum Well Known Type* para colocar URLs, SMSs ou números de telefone numa *tag* NFC, ou para transportá-los entre os dispositivos. O *Smart Poster* RTD baseia-se no mecanismo de RTD e formato NDEF, usando a URI RTD e Texto RTD como blocos de construção;
- ✓ **Controlo Genérico** – A especificação foi retirada sem qualquer substituição;
- ✓ **Assinatura** - Especifica o formato usado quando se assina um ou vários registos NDEF. Define os campos de assinatura RTD necessários e opcionais, e também fornece uma

lista de assinatura de algoritmos apropriados e os tipos de certificados que podem ser utilizados para criar a assinatura. Não define nem impõe um *Public Key Infrastructure* (PKI) específico ou sistema de certificação, bem como também não define um novo algoritmo para usar com a *Signature* de IDT. A especificação do processo de certificação está fora do alcance.

2.4.2.3 *Logical Link Control Protocol*

Para melhorar o modo *Peer-to-Peer*, o NFC Fórum especificou um protocolo ao nível da camada de ligação de dados, conhecidos por "*Logical Link Ccontrol Protocol*" (LLCP). Este protocolo fornece funcionalidades adicionais à especificação NFCIP-1/ISO 18092.

O protocolo LLCP introduz duas vias de ligação de dados, permitindo assim que ambos os dispositivos possam enviar e receber dados, utilizando os seguintes métodos para transferência de dados:

- ✓ Transferência de ligação orientada, na qual as transferências de dados são reconhecidas/confirmadas pelos *peers*.
- ✓ Transferência sem ligação, onde as transferências de dados não são reconhecidas.

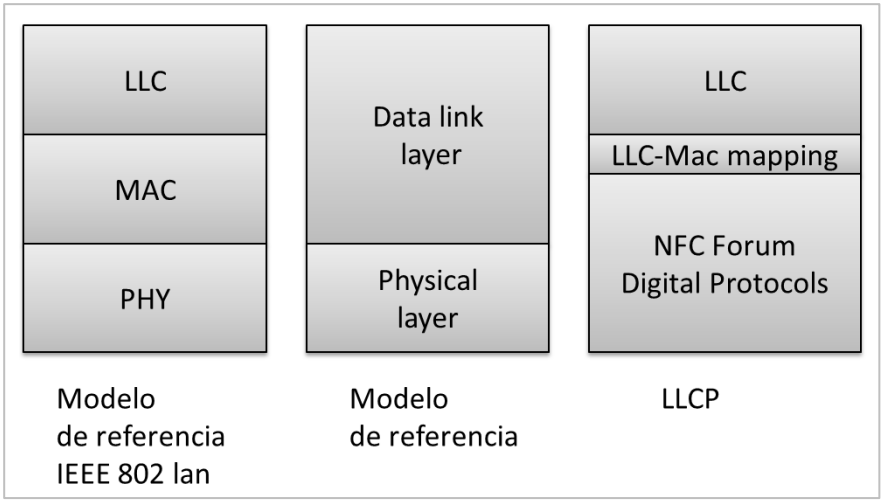


Figura 14 – Modelos de referência LLCP

2.4.3 Exemplos de Utilização

Tem surgido um número crescente de aplicações com recurso a este tipo de tecnologia. Neste momento existem aplicações para o pagamento por contacto onde, para compras pequenas, é apenas necessário aproximar o cartão ou telemóvel. Atualmente, um exemplo desta funcionalidade, salvaguardando ainda algumas limitações no que diz respeito à compatibilidade de dispositivos, é o *Google Wallet*.

Figura 15 – *Google Wallet*

Na área do *marketing* e publicidade, os *chips* NFC, têm sido utilizados em cartazes, canetas, *flyers* entre outros, fornecendo um acesso simples a conteúdos extra, como é um caso de vídeos, musicas ou mesmo endereços w*eb*.

No turismo, podemos encontrar NFC em museus, pontos turísticos, entre outros. O objetivo é possibilitar o utilizador de, apenas com um toque, aceder a toda a informação sobre o objeto ou ponto selecionado.

Outra área com forte crescimento são os transportes. Hoje em dia, em Londres, os transportes metropolitanos são geridos pela tecnologia NFC. O uso de cartões magnéticos estará com os seus dias contados, pois a *Transport for London* (TfL) anunciou que os utilizadores poderão utilizar os seus *smartphones* para fazerem as validações das suas viagens. O processo de implementação, ainda está em estágio inicial, ainda com as companhias de telemóveis, juntamente as empresas de cartões de crédito a desenvolver as aplicações móveis necessárias. Acredita-se que, no final do deste ano, esteja disponível para o público em geral.

2.5 Desafios

Em regra, a utilização de aplicações móveis normalmente não capta a atenção do utilizador na sua totalidade. Frequentemente, durante a utilização de aplicações móveis realizam-se outras tarefas como: ouvir música, ver televisão, ler um jornal ou uma revista, entre outras. Como tal, o *smartphone* torna-se um objeto secundário da atenção dos utilizadores dado que, após a consulta que realiza na aplicação, este retorna para a tarefa que estava a efetuar inicialmente.

A necessidade de uma aplicação possuir uma boa usabilidade bem como uma interface apelativa, são premissas indispensáveis para o sucesso de qualquer aplicação.

O maior desafio desta aplicação está na potencialização da mesma para um novo paradigma de utilização. Com esta aplicação pretende-se dar ao utilizador todo o poder de análise descrito nas aplicações anteriores mas, ao mesmo tempo, utilizar a aplicação e o seu *smartphone* como um cartão de jogador físico e, com isto, salvaguardar todos os interesses do apostador.

Ao longo do dia, a quantidade de papéis que se vão acumulando nos bolsos e carteiras é um problema comum à grande parte da sociedade. Surgem regularmente histórias de perdas do talão premiado do EuroMilhões, deste modo, a necessidade de precaver estes problemas tornou-se indispensável.

Com esta aplicação, o utilizador tem a salvaguarda de, quando efetuar o seu registo, este fique guardado no seu *smartphone* e numa base de dados central. Mesmo que sofra o infortúnio de perder o seu *smartphone*, conseguirá resgatar o seu prémio sem grande transtorno.

3 Desenho Conceptual

O objetivo deste projeto consiste na criação de uma alternativa fidedigna face aos tradicionais sistemas existentes para os jogos da sorte, nomeadamente do EuroMilhões, para utilizadores de dispositivos móveis *Android* com a funcionalidade NFC disponível.

O projeto centraliza-se numa aplicação móvel que, para além de fornecer dados estatísticos e notificações sobre os prémios, funciona simultaneamente como um cartão de jogador e como um boletim de apostas. Foi também criada uma outra aplicação, capaz de ler os dados provenientes dos dispositivos móveis, por intermedio do leitor NFC ligado ao computador, onde o leitor utilizado, foi o ACR 122 U da *Advanced Card Systems* (ACS).

3.1 Conceitos Tecnológicos Utilizados

De forma a facilitar a compreensão do desenvolvimento do protótipo, é relevante citar alguns conceitos relevantes:

- ✓ ***JavaScript Object Notation* (JSON) –** é um formato de dados leve para troca de dados computacionais e, como o próprio nome indica, é um subconjunto de notação de objeto de *Javascprit*. É muitas vezes utilizado como alternativa ao *Extensible Mark-up Language* (XML), sendo mais fácil para escrever e mais percetível para o programador.

- ✓ ***Web Service* -** é uma solução que permite a comunicação entre aplicações de uma maneira independente de sistema operacional e de linguagem de programação. Com esta solução, por exemplo, é possível interagir entre aplicações antigas desenvolvidas num sistema operativo *Linux*, com aplicações recentes desenvolvidas no sistema operativo *Windows*. Os *Web services* permitem que, aplicações independentes da sua "linguagem" enviem e recebam dados numa linguagem universal, XML ou JSON.

- ***Representational State Transfer* (REST) –** surgiu quando o cientista norte-americano Roy Fielding, um dos principais autores da especificação *Hyper Transfer Protocol* (HTTP), realizou a sua tese de doutoramento. O intuito era permitir a comunicação entre duas aplicações, independentemente do sistema operativo ou linguagem, ver Figura 16. O REST utiliza o protocolo HTTP para troca de mensagens utilizando, além dos métodos clássicos POST e GET, métodos menos comuns tais como o PUT e DELETE.

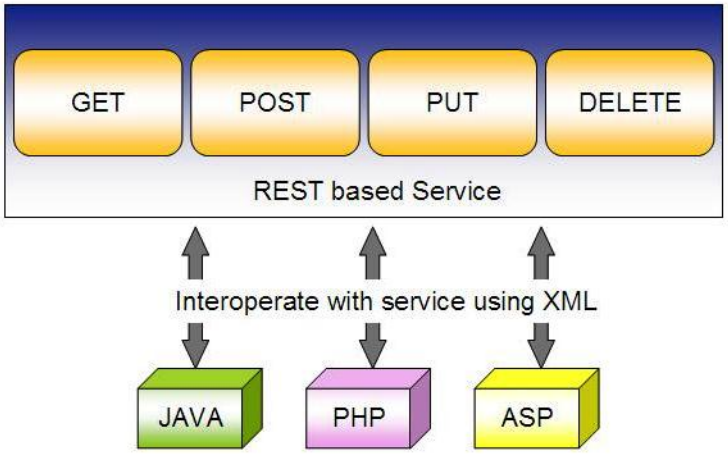


Figura 16 - Operações HTTP para Serviços *Restful*

- **NFC** - é uma tecnologia sem fios de curto alcance que permite a comunicação entre objetos com a mesma tecnologia, a distância inferior a 10cm. O NFC é baseado em padrões de RFID, e é projetado para tornar o dia-a-dia mais simples e proveitoso, melhorando a forma de fazer transações bancárias, troca de conteúdos e conexões entre dispositivos.

3.2 Requisitos da Aplicação Móvel e Aplicação Desktop

A aplicação foi concebida para *smartphones* que apresentem a funcionalidade NFC bem como para dispositivos com versões iguais ou superiores à 4.0 (*Ice Cream Sandwich*), dado que utiliza um recurso apenas inserido a partir desta versão, o *Android Beam*.

A conectividade não é uma premissa indispensável para o uso da aplicação. Ela apenas é necessária para efetuar o registo da aplicação no dispositivo móvel, preenchendo um formulário de registo, para um novo utilizador no sistema, ou um formulário de login, para um utilizador já existente. Os dados dos sorteios serão atualizados sempre que o utilizador pretender, desde que exista conetividade à Internet. Deste modo, o utilizador pode criar as suas chaves e efetuar os registos do EuroMilhões, junto dos *Point of Sale* (POS) sem que, em nenhum destes passos, o utilizador tenha que ter obrigatoriamente acesso à Internet.

Por outro lado, a aplicação *Desktop*, que pretende simular um POS, necessita de estar sempre conectada à Internet, de modo a proceder ao registo dos boletins. Para além disto, o leitor NFC, ACR 122 U (Figura 17) necessita também de estar sempre conectado.

Figura 17 - Leitor NFC ACR 122 U

3.3 Modelo de Conceptual

Antes de desenvolver a qualquer aplicação é essencial o conhecimento de todos os conceitos que esta envolve, bem como da relação existente entre eles. A Figura 18 apresenta o modelo concetual que serve de base à aplicação móvel criada, permitindo visualizar as várias relações existentes entre classes, bem como os atributos específicos de cada classe.

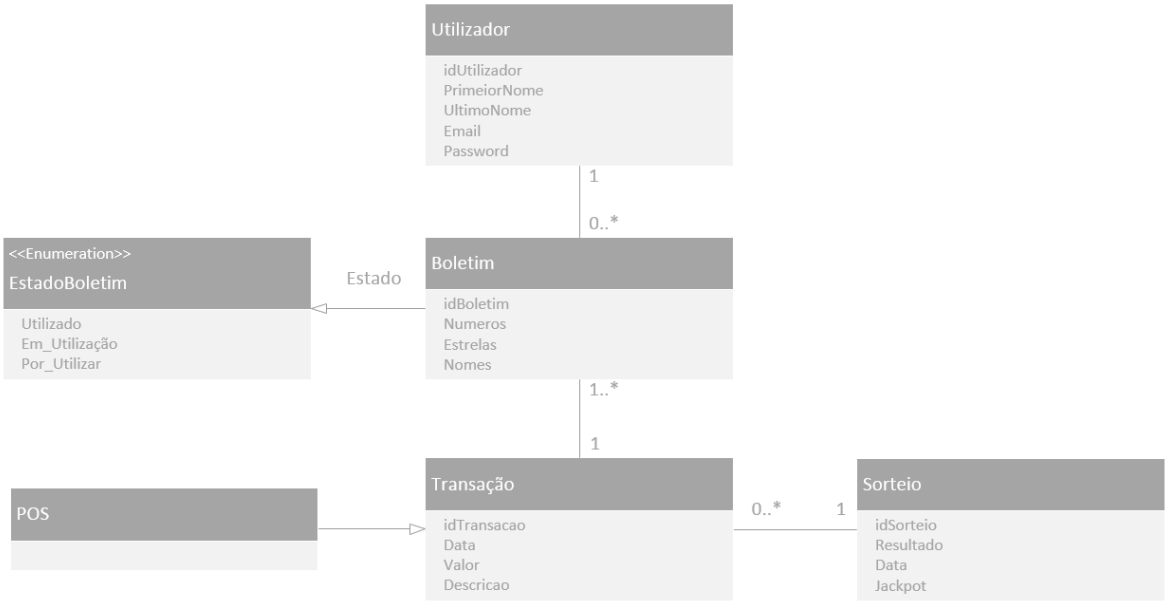


Figura 18 – Modelo Conceptual

3.3.1 Utilizador

Sendo uma aplicação de uso personalizado, é necessário associar toda a informação a um determinado utilizador. O elemento que garante a unicidade do utilizador é o endereço de correio eletrónico pelo que, após o registo, o utilizador não o poderá alterar.

3.3.2 Boletim

Os boletins são os componentes centrais do sistema, sendo através deles que se processam as principais funcionalidades da aplicação. Os boletins têm um estado associado (Utilizado, Em_Utilização, Por_Utilizar), podendo assim o utilizador possuir vários boletins em simultâneo embora apenas um deles possa estar no estado "Em_Utilização". Todos os boletins, independente do seu estado, possuem um conjunto de números, estrelas e um nome únicos.

3.3.3 Transação

Uma transação é uma operação de registo do boletim. Esta operação ocorre invariavelmente num POS e tem associada uma data, um valor e uma descrição associada. Em cada transação é gerado um identificador único facilitando assim, a criação e manutenção de um histórico de transações.

3.3.4 Sorteio

Sorteio como o próprio nome indica, representa o sorteio bissemanal do jogo Euromilhões, onde são extraídos 5 números e 2 estrelas. Cada sorteio tem uma data e um valor (*jackpot*) associado.

3.4 Funcionalidades do Sistema

A Figura 19 representa o diagrama de casos de uso presentes para as principais funcionalidades do sistema.

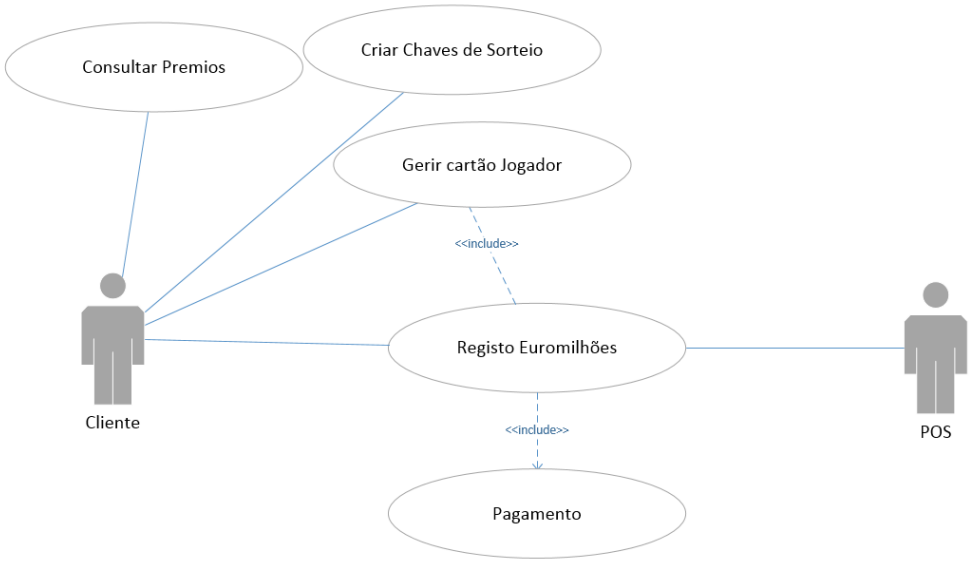


Figura 19 – Diagrama de casos de uso

3.4.1 Consultar Prémios

Esta funcionalidade é utilizada quando o utilizador pretende verificar o resultado do sorteio do Euromilhões. A Figura 20 representa o diagrama de sequência da operação "consultar prémios". Esta operação é descrita pelas seguintes etapas:

1. O utilizador acede à aplicação no menu prémios;
2. A aplicação móvel solicita um pedido de atualização ao serviço *web*. São enviados os dados do cliente;
3. O serviço *web* solicita dados à base de dados;
4. A base de dados retorna os dados para o serviço *web*;
5. O serviço *web* retorna os dados para a aplicação cliente;
6. A aplicação grava os novos dados na base de dados aplicacional para que consultas posteriores possam ser efetuadas em modo *offline*;
7. A base de dados aplicacional retorna aviso quando finalizada a atualização dos prémios.

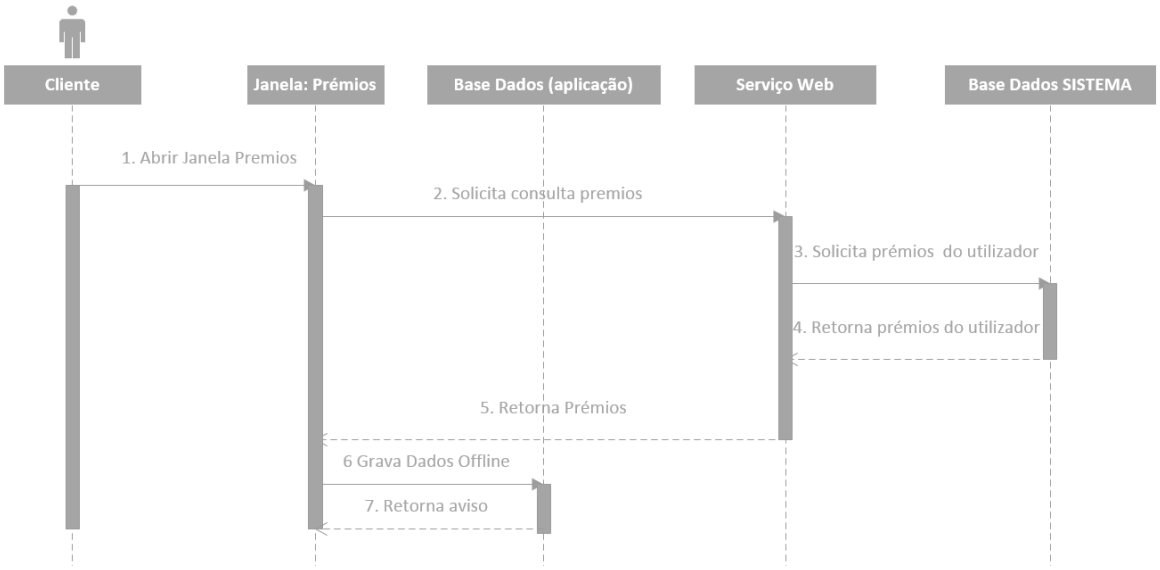


Figura 20 – Diagrama de Sequência ”Consultar Prémios”

3.4.2 Criar Chave Sorteios

Esta funcionalidade é utilizada sempre que o utilizador cria uma nova chave para o sorteio Euromilhões. A Figura 21 representa o diagrama de sequência da operação consultar prémios. Esta operação é descrita pelas seguintes etapas.

1. O utilizador acede ao Menu Chave;
2. É selecionado o modo “Criar Chave”;
3. A aplicação retorna um formulário;
4. O utilizador preenche o formulário com os dados: números, estrelas e nome da chave criada;
5. É realizado um processo de validação do formulário;
6. É criado um novo registo na base de dados da aplicação;
7. A base de dados retorna informação sobre a criação do novo registo.

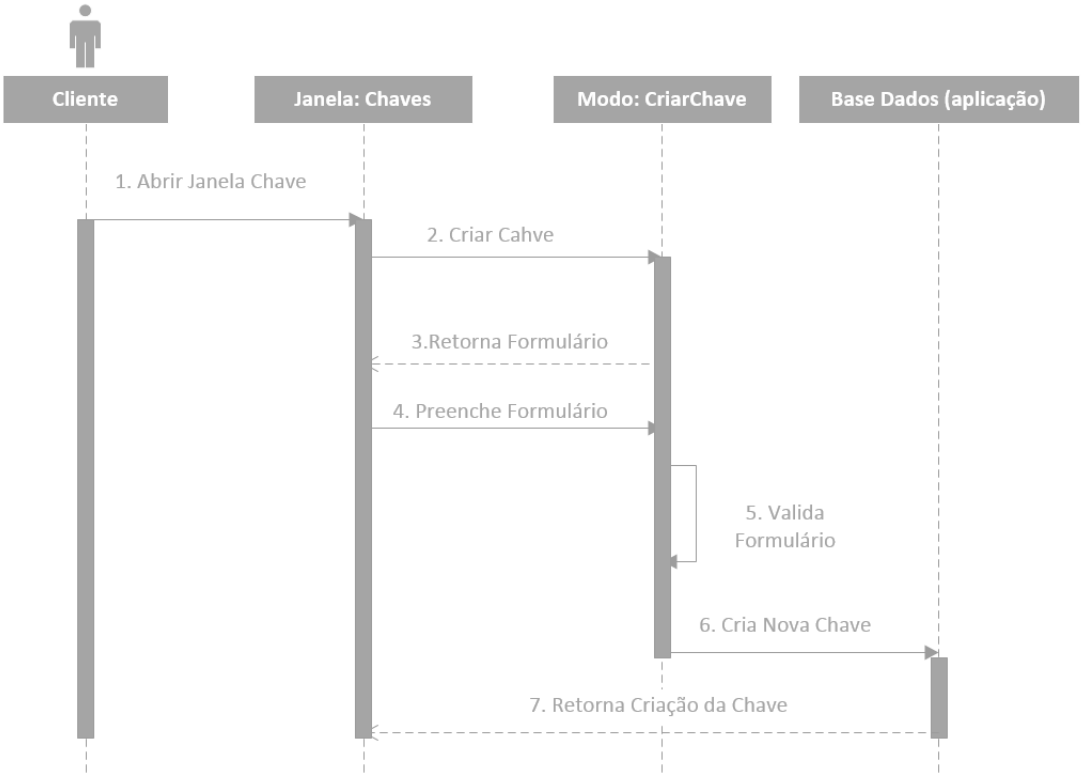


Figura 21 – Diagrama de Sequência ”Criar Chave de Sorteios”

3.4.3 Gerir Cartão Jogador

Esta funcionalidade é utilizada sempre que o utilizador pretende efetuar uma alteração no seu cartão de jogador, nomeadamente alterar a chave com que pretende efetuar o registo para o sorteio. Este caso de uso tem como pré-requisito a elaboração de uma chave (caso de uso 3.4.2).

A Figura 22 representa o diagrama de sequência da operação gerir cartão Jogador. Esta operação é descrita pelas seguintes etapas.

1. O utilizador acede ao “Menu Cartão”;
2. O utilizador escolhe a opção “Editar Cartão Jogador”;
3. A aplicação faz um pedido à base de dados aplicacional de forma a retornar todas as chaves criadas previamente pelo utilizador;
4. A base de dados aplicacional retorna todas as chaves;
5. O utilizador seleciona qual a chave que pretende ter ativa;
6. A aplicação guarda as alterações efetuadas pelo utilizador na base de dados aplicacional;
7. A aplicação informa o utilizador que a operação foi executada.

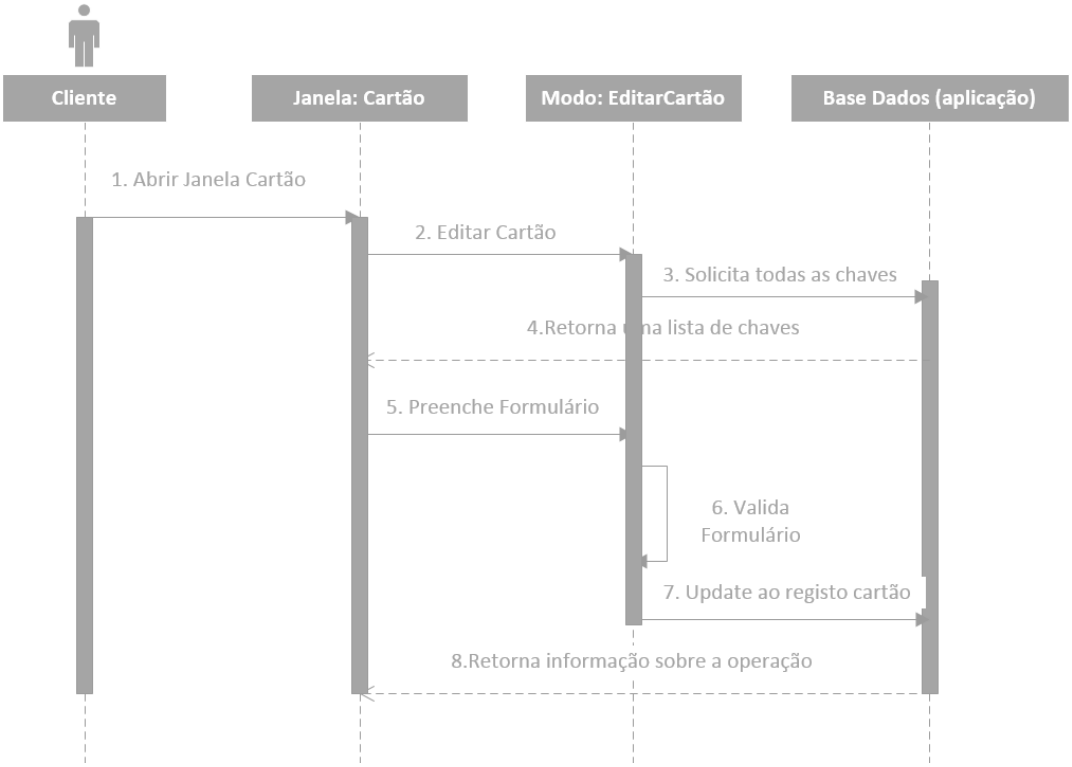


Figura 22 – Diagrama de Sequência ”Gerir Cartão de Jogador”

3.4.4 Registo Euromilhões

Esta funcionalidade é a mais importante de todo o sistema. Neste caso de uso, o utilizador efetua o registo do Euromilhões junto do POS, tendo como pré-requisito a criação de um cartão de jogador e, consequentemente, uma chave (casos de uso 3.4.2 e 3.4.3).

A Figura 23 representa o diagrama de sequência da operação “Registo Euromilhões”. Esta operação é descrita pelas seguintes etapas:

1. O utilizador acede, na aplicação móvel, ao “Menu de Pagamento”,
2. A aplicação móvel envia o boletim virtual para o leitor de cartões, presente no POS, por intermédio da tecnologia NFC;
3. O POS solicita o pedido de um novo registo ao serviço *web;*
4. O serviço *web* valida os dados;
5. O serviço *web* cria um novo registo na base de dados do sistema;
6. A base de dados retorna o registo inserido;
7. O serviço *web* informa o POS que o registo foi concluído com sucesso;
8. O POS efetua os cálculos e gera o valor a pagar por parte do cliente;
9. O POS informa a aplicação móvel de qual o valor a pagar por intermédio do leitor NFC que envia uma mensagem à aplicação.

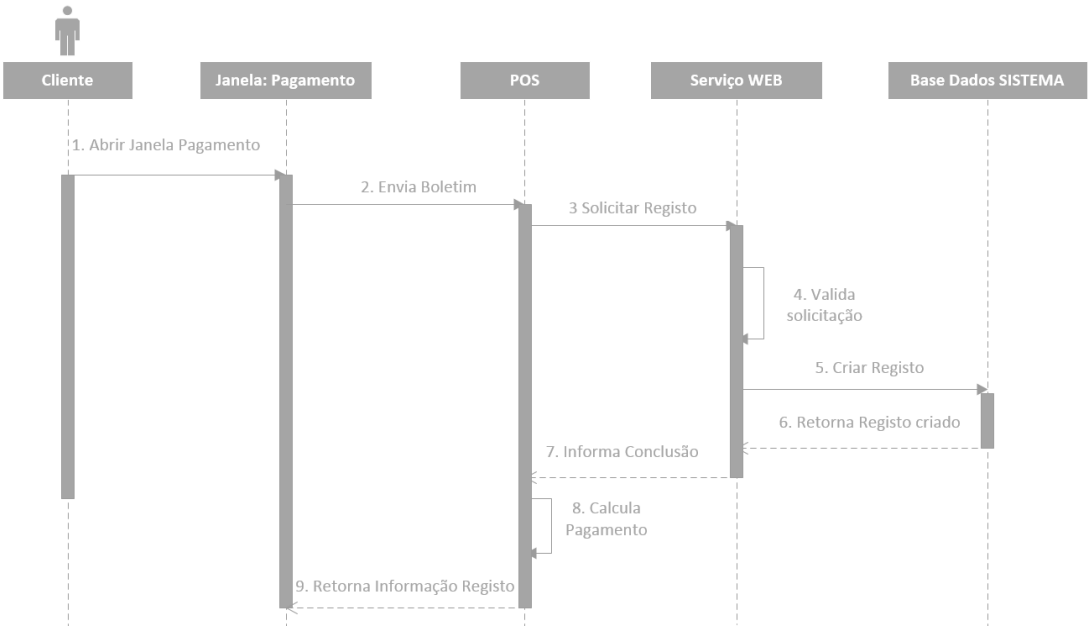


Figura 23 – Diagrama de Sequência ”Registo Euromilhões”

3.5 Interação Utilizador/Eurobolso

Para o funcionamento da aplicação, o utilizador necessita primeiramente de criar uma conta de utilizador. Esta funcionalidade apenas fica disponível na primeira abertura da aplicação “Eurobolso” no dispositivo móvel.

O utilizador quando abre pela primeira vez a aplicação tem a sua disposição a opção para criar uma nova conta de utilizador por intermédio de um formulário específico. O utilizador pode também ao invés de criar uma nova conta, efetuar o Login de uma conta existente.

O aspeto visual do protótipo não foi muito trabalhado, tendo-se dado principal incidência aos automatismos para as comunicações entre os diversos sistemas, nomeadamente entre a aplicação móvel e a aplicação que se pretende que funcione como o POS.

O utilizador após abrir a aplicação, entra no ecrã principal, obtendo inúmeras opções que, quando selecionadas, redirecionam o utilizador para os restantes ecrãs. Entre as funcionalidades desta aplicação, o utilizador poderá criar chaves para os boletins, atribuir chaves para o próximo sorteio, efetuar o registo da chave, visualizar os resultados dos sorteios e verificar quanto ganhou no sorteio.

O processo para o registo da chave do Euromilhões junto do POS é realizado pela simples aproximação do dispositivo móvel ao leitor NFC, tendo o utilizador previamente acedido ao ecrã de pagamento na aplicação.

O ecrã principal centraliza toda a informação, com botões para os restantes ecrãs, funcionando de forma semelhante a um menu. Para além disto, é apresentada informação sobre o estado do próximo sorteio, nomeadamente informação relativa ao tempo restante para efetuar o registo do boletim, qual o prémio sorteado e se já procedeu ou não ao registo.

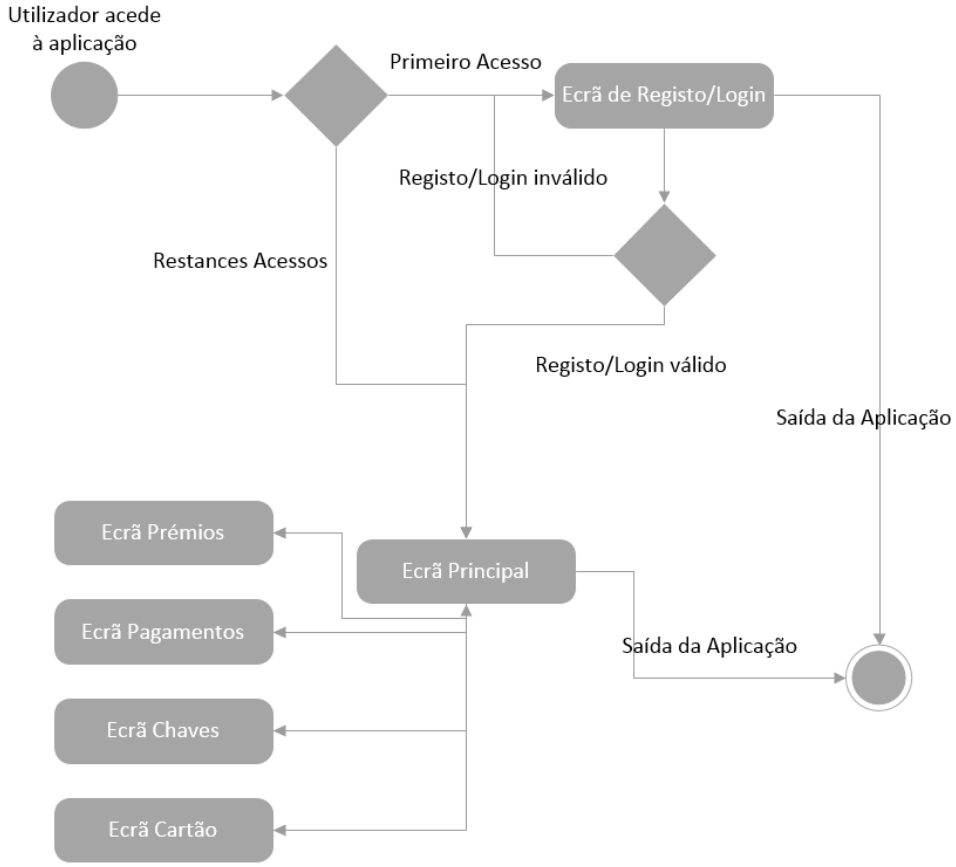


Figura 24 – Diagrama de Estados da Aplicação “Eurobolso”

4 Sistema Desenvolvido

4.1 Arquitetura do Sistema

Neste capítulo é descrita a forma como foi idealizada e desenvolvida a arquitetura de todo o sistema. Inicialmente será abordada a arquitetura do servidor, onde estão centralizados todos os *Web Services* consumidos pelas aplicações, bem como a base de dados central. Em seguida será abordada a aplicação que pretende simular o POS e, por fim, será descrita a aplicação móvel "Eurobolso".

4.1.1 Arquitetura do Servidor

O servidor selecionado para a realização deste protótipo foi o servidor *WampServer*. É um servidor que permite a publicação de aplicações *Web* com *Apache2 e* PHP, possuindo ainda uma base de dados em *MySQL* gerida por intermédio da ferramenta *PhpMyAdmin*.

O servidor é responsável pela resposta aos pedidos tanto da aplicação móvel, como da aplicação que pretende simular o POS, lançando também alguns serviços de atualização. Estão definidos três diretórios, onde os *Web Services* são alocados consoante a sua função. Deste modo, não são misturados os pedidos da aplicação móvel com os pedidos da aplicação que simula o POS, ou mesmo de serviços específicos da atualização de dados.

Tratando-se de um servidor que assenta sobre o sistema operativo *Windows*, foi possível recorrer a uma das suas ferramentas (Programador de Tarefas) de modo a poder agendar tarefas de atualização em dias e horas específicas, facilitando assim a constante atualização dos dados.

O processo de atualização é realizado nos dias dos sorteios do Euromilhões, terças e sextas respetivamente, sendo dividido em duas etapas: atualização da chave e atualização do prémio, de modo a que haja uma correta atualização. Foi necessário criar esta divisão nos

processos de atualização, pois existe sempre um atraso entre a hora do sorteio e a hora em que são publicados os resultados dos prémios. Estas etapas encontram-se representadas de acordo com a Figura 25, sendo seguidamente descritas:

- ✓ **Atualização da Chave –** Nesta etapa, agendada para as 20h30m de terça e sexta-feira é feita uma leitura à página *web* da Santa Casa, onde é extraída a chave sorteada e inserida na base de dados do servidor. Esta leitura é iniciada pelo Programador de Tarefas do *Windows* que, por sua vez, executa um ficheiro *Batch* que invoca o serviço referido anteriormente.

- ✓ **Atualização do Prémio** – Esta segunda etapa está calendarizada de forma diferente em relação à anterior pois, a disponibilização dos resultados finais por parte da Santa Casa, não define uma hora fixa para o seu lançamento. Deste modo, optou-se por fazer carregamentos nos dias dos sorteios entre as 21:00 e as 24:00 horas com uma periocidade de 30 minutos. A leitura da página de resultados da Santa Casa é realizada pelo mesmo princípio, existindo um outro ficheiro *Batch* que invoca o respetivo serviço.

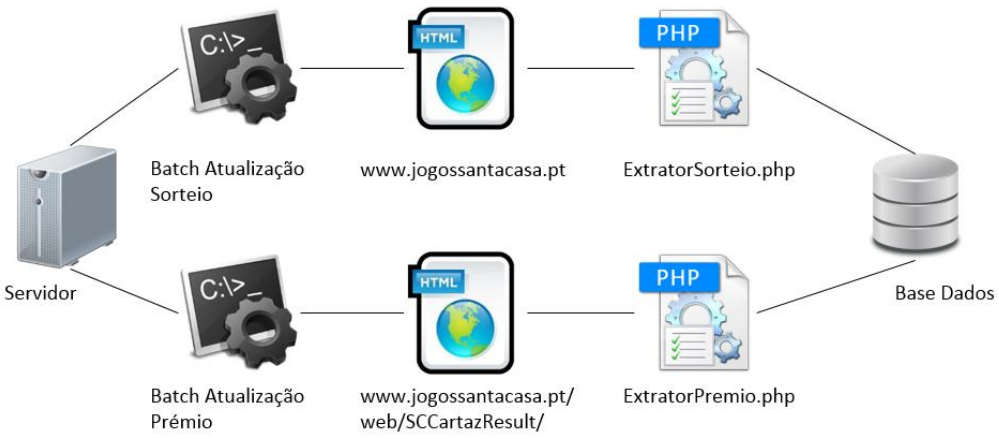


Figura 25 – Esquema de atualização do resultado Euromilhões

A base de dados central do sistema é constituída por seis tabelas, contendo informações sobre os utilizadores e os seus registos do Euromilhões e dados relativos aos sorteios, nomeadamente informação geral sobre o sorteio e informações sobre os prémios de cada sorteio.

A Figura 26, abaixo representada, ilustra como a base de dados se encontra modelada.

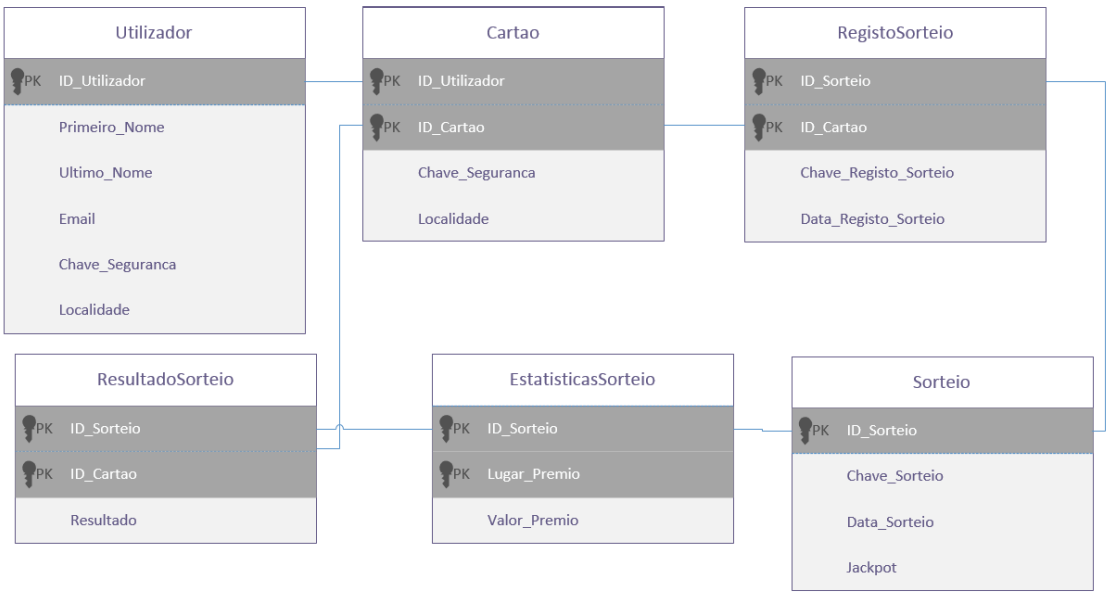


Figura 26 - Modelo de dados do sistema

4.1.2 Arquitetura da Aplicação (POS)

O desenvolvimento da aplicação POS foi efetuado em *Java*. De modo a realizar a comunicação entre esta aplicação e o dispositivo móvel foi necessário recorrer a um conjunto de bibliotecas e ferramentas (*NFCTools*).

O *NFCTools* é um conjunto de quatro bibliotecas (*api, core, ndef e p2p*) desenvolvidas em *Maven*. A biblioteca fornece mecanismos de forma a agilizar a comunicação entre leitores USB ligados ao computador e os dispositivos móveis com o sistema operativo *Android,* permitindo ainda a programação de cartões e *tags NFC*.

4.1.2.1 Estrutura da Aplicação

A estrutura da aplicação POS está estruturada de acordo com a Figura 27.

As quatro primeiras pastas são as bibliotecas da *NFCTools*, sendo necessária importar uma a uma para o seu correto funcionamento. O *Integrated Development Environment* (IDE) para o desenvolvimento foi o *Eclipse*, como tal, foi necessário recorrer ao *plugin "m2eclipse"* de forma a poder realizar a importação da biblioteca desenvolvida na linguagem *Maven*.

A interface da aplicação POS está definida na classe JanelaCompra.java e é lançada por intermedio das classes POS_LeitorCartao e POS_LeitorTelemovel, podendo assim receber dados de um cartão NFC ou de um telemóvel *Android* sem recorrer a duas aplicações distintas.

A leitura de dados de um cartão NFC é realizada por um comando *Application Protocol Data Unit* (APDU). Por sua vez, a comunicação entre o leitor NFC e um telemóvel *Android* é realizada através do *Simple NDEF Exchange Protocol* (SNEP), onde o leitor recebe uma mensagem, denominada NFC *Data Exchange Format* (NDEF) enviada pelo *smartphone*.

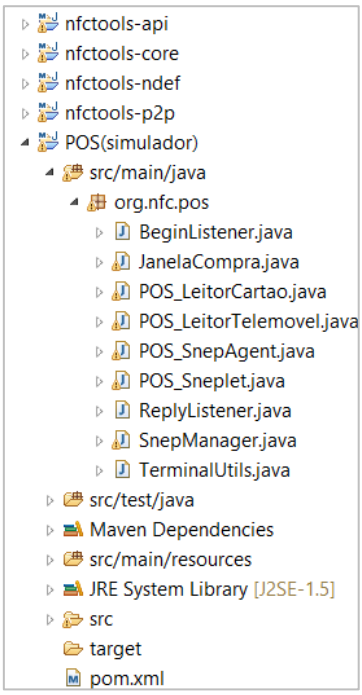


Figura 27 – Estrutura da aplicação POS

4.1.3 Arquitetura da Aplicação Móvel "Eurobolso"

4.1.3.1 Bibliotecas Auxiliares

A aplicação móvel foi construída em *Java* e, por conseguinte, orientada para ser utilizada no sistema operativo *Android*. Contudo, foi necessário recorrer a algumas bibliotecas já implementadas (SQLite e Jackson), de modo a facilitar o processo de desenvolvimento.

- ✓ **SQLite** - É uma pequena biblioteca, desenvolvida em linguagem C no ano de 2000, que implementa um amplo subconjunto do *standard* SQL 92, onde a sua reputação é proveniente da combinação do motor de base de dados com a interface dentro de uma única biblioteca. As aplicações que usam *SQLite* podem ter acesso a uma base de dados racional SQL, sem que ocorram processos RDBMS (*Relational Database Management System*) em separado, e sem grandes *overheads*. O SQLite é atualmente uma base de dados amplamente adotada em dispositivos móveis, suportando até 2 TB de dados e tendo suporte nativo por parte do *Android*.

- ✓ ***Jackson*** - *Parser* de JSON, atualmente reconhecida por ser a biblioteca mais rápida e eficaz. É extremamente útil para receber os dados relativos aos sorteios e para o envio de dados da aplicação móvel para o servidor (ver Figura 28).

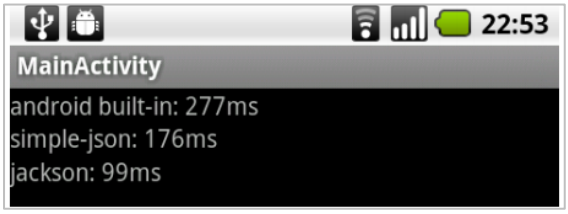


Figura 28 – Eficiência da Biblioteca *Jackson*

4.1.3.2 Estrutura da Aplicação

As aplicações *Android* possuem uma estrutura específica, composta por uma série de pastas e ficheiros, como mostra a Figura 29.

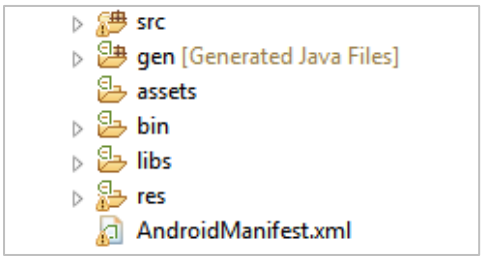


Figura 29 – Estrutura de uma aplicação *Android*

- ✓ **Src** – Código fonte da aplicação, normalmente dividido por *packages*;
- ✓ **Gen** – Código fonte gerado automaticamente;
- ✓ **Assets** – Ficheiros estáticos que serão incluídos na aplicação;
- ✓ **Bin** – Pasta onde, por defeito, os ficheiros compilados são guardados;
- ✓ **Libs** – Pasta onde são guardadas todas as bibliotecas *Java Archives* (JARs) necessárias;
- ✓ **Res** – Pasta onde se encontram recursos como ícones, definição de *layouts* e imagens.

Os recursos da aplicação estão guardados em subpastas, as quais correspondem a um determinado tipo de recurso. Nestas pastas, são guardados os ficheiros XML com a definição de *layouts*, *strings* e cores. Esta organização de informação permite a definição de recursos alternativos para diferentes configurações específicas dos equipamentos, da linguagem e da resolução do ecrã. O acesso aos recursos através do código é realizado através da classe R que é criada no momento da compilação do projeto. A classe R contém subclasses para cada tipo de recursos, desde que exista pelo menos um recurso desse tipo.

Qualquer projeto do sistema *Android* possui um único ficheiro *AndroidManifest.xml* guardado na sua raiz de desenvolvimento. Neste ficheiro ficam definidos a estrutura, os meta-dados e os componentes da aplicação, sendo um elemento crucial de qualquer projeto *Android*. É nele que são declaradas as *Activities* que definem os vários ecrãs que constituem a aplicação, os *Receivers e* os *Services,* a versão SDK e as respetivas permissões que a aplicação terá ao ser instalada nos dispositivos móveis.

Uma *Activity* consiste num ecrã duma aplicação *Android,* sendo implementada como uma única classe que se estende a partir da classe base *Activity*. Essa classe irá mostrar uma interface composta por *Views,* a qual irá responder a eventos por parte do utilizador.

A mudança entre ecrãs é realizada quando se dá início a uma nova *Activity*. Quando um novo ecrã surge, o seu antecessor fica num estado de *standby* e é colocado numa pilha, permitindo assim ao utilizador navegar entre eles. A função *startActivity* responsável pela abertura de *Activities*, inclui como parâmetro uma instância de uma classe denominada *Intent*. Nesta instância, podemos definir parâmetros que sejam necessários para uma correta inicialização da nova *Activity* como por exemplo, o envio de uma *string*, *booleno* ou estrutura.

Um *Receiver* é um componente de qualquer aplicação *Android* que permite responder a notificações específicas enviadas pelo sistema. Quando determinados eventos ocorrem, o *Android* envia mensagens para todo o sistema, de modo a que as aplicações que tenham relevância face às mensagens possam responder. A mensagem enviada pelo sistema *Android* é denominada *Broadcast*, sendo este o componente responsável pela captação desta mensagem, podendo ser denominado *Broadcast Receiver* ou simplesmente *Receiver*.

Outro componente das aplicações Android é o *Service*. O *Service* permite realizar tarefas de longa duração em *background*, não afetando a interface do utilizador. Um determinado componente da aplicação pode iniciar um serviço e o mesmo irá continuar em funcionamento quer o utilizador esteja a utilizar a aplicação ou não.

É ainda importante referir outros conceitos que foram utilizados no decorrer da programação da aplicação:

- ✓ ***Context*** - A classe *Context* permite aceder a recursos e configurações partilhados entre os vários ecrãs (*Activities*) da aplicação. Cada *Activity* possui o seu próprio *Context*.

- ✓ **Interface** - A Interface é um recurso muito utilizado na maioria das linguagens orientadas a objetos. Permite que um determinado grupo de classes possa ter métodos e/ou propriedades em comum num determinado contexto. Em todo o caso, os métodos podem ser implementados de maneira diferente em cada classe.

- ✓ ***View*** - Uma *View* pode ser um componente gráfico de uma aplicação como por exemplo: botões, *checkboxes* e imagens. Para além disto, pode também atuar como

um gerenciador de *layout,* possuindo a função de organizar outras *Views* mais simples (ver Figura 30), como por exemplo os radiobuttongroup.

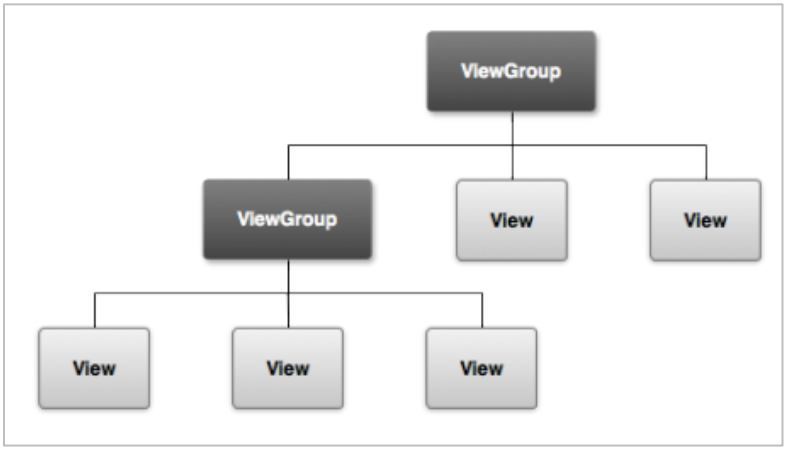


Figura 30 – Hierarquia de uma *View*

- ✓ ***Multitasking*** - É muito comum, nas aplicações mais complexas, ser necessário realizar tarefas mais "dispendiosas" que podem necessitar de um longo tempo de execução até finalizarem. Pode ser uma requisição *Web* ou até uma operação mais complexa na base de dados. Este tipo de tarefas requer uma atenção especial por parte do programador, pois não é recomendável que sejam executadas como parte do processo principal pois, isso impede que o estado da aplicação mude, ou seja, não será possível atualizar o ecrã corrente ou receber nenhum comando de entrada. A aplicação ficará em *standby* durante o processo e, no caso do Android, se isso for frequente, esta poderá encerrar automaticamente.

Parte da solução consiste em utilizar outras *threads* para os diversos processamentos mais complexos e deixar a *thread* principal apenas responsável pelas operações de INPUT/OUTPUT. Em todo o caso, apenas a *thread* que iniciou a UI pode alterá-la, ou seja, não é possível criar uma nova *thread* e obrigá-la a atualizar algo no ecrã. Apesar desta contrariedade, o sistema *Android* fornece os seguintes métodos de forma a contornar o problema:

- ✓ ***Adapter*** - É um padrão de desenho que serve para adaptar duas interfaces incompatíveis. No caso do *Android*, serve para adaptar uma lista de objetos para uma lista de elementos da interface gráfica, como as linhas de uma *ListView* (exibe os componentes em forma de lista). O *Android*, através da classe *ListView* solicita, para cada linha da lista, um novo objeto *View*. A função do *Adapter* é criar um objeto *View* que represente visualmente o objeto da posição específica da lista de objetos.

- ✓ ***AsyncTask*** - Encapsula o processo, por norma complexo, fornecendo métodos para modificar a UI antes, durante e após a execução.

- ✓ ***Shared Preferences*** - A classe *Shared Preferences* fornece uma *Framework* geral que permite salvar e recuperar pares de dados primitivos do tipo chave-valor. Os mesmos continuam armazenados, mesmo após a aplicação ter sido encerrada. Esta classe foi particularmente útil na gestão das sessões dos utilizadores na aplicação.

4.2 Implementação do Sistema

Neste capítulo é abordada a implementação do sistema. Inicialmente, será descrito como foram estruturados e implementados, os *WebServices* existentes no servidor, em seguida será abordada a implementação da aplicação POS e, por fim, a aplicação móvel "Eurobolso".

4.2.1 Implementação do Servidor

A implementação ao nível do servidor está dividida em três áreas como mostra a seguinte Figura 31.

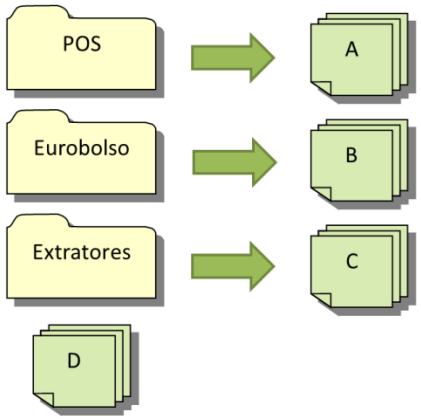


Figura 31 - Estrutura do servidor

Do lado esquerdo da Figura 21, está representada a estrutura de diretórios existentes no servidor, bem como o conjunto de dois ficheiros, representados com a letra D (DB_Config.php e DB_Connect.php), responsáveis por conter as parametrizações e realizar a conexão à Base de Dados, respetivamente. Por sua vez, do lado direito, estão esquematizados os diversos ficheiros existentes para manter as comunicações e atualizações.

O conjunto de ficheiros denominados com as letras A e B são *WebServices* construídos para retornar objetos e *strings* no formato JSON, responsáveis por responder a chamadas efetuadas pelas aplicações POS e Eurobolso.

A Figura 32, representa o *WebService* responsável pelo registo de um novo utilizador no sistema, pertencendo ao conjunto de ficheiros B.

```php
<?php
// array  JSON
$response = array();
// check campos
if (isset($_POST['email']) && isset($_POST['chave']) && isset($_POST['primeiro_nome']) &&
    isset($_POST['ultimo_nome']) && isset($_POST['localidade'])) {
    $email = $_POST['email'];
    $chave = $_POST['chave'];
    $primeiro_nome = $_POST['primeiro_nome'];
    $ultimo_nome = $_POST['ultimo_nome'];
    $localidade = $_POST['localidade'];
    // include db connect
    require_once __DIR__ . '/db_connect.php';
    // connect db
    $db = new DB_CONNECT();
    $result = mysql_query("INSERT INTO utilizador(Email, ChaveSeguranca, PrimeiroNome, UltimoNome, Localidade)
    VALUES('$email', '$chave', '$primeiro_nome', '$ultimo_nome' , '$localidade')");
    // check se foi inserido
    if ($result) {
        // inserido com sucesso
        $response["success"] = 1;
        $response["message"] = "Utilizador inserido";
        //  JSON response
        echo json_encode($response);
    } else {
        // falha row
        $response["success"] = 0;
        $response["message"] = "Oops! Ocorreu um erro!";
        //  JSON response
        echo json_encode($response);
    }
} else {
    // required field is missing
    $response["success"] = -1;
    $response["message"] = "Falta de campos";
    // JSON response
    echo json_encode($response);
}
```

Figura 32 - *CreatUser.php*

Os ficheiros existentes no diretório Extratores são responsáveis por retirarem informação do portal da Santa Casa. Não sendo possível aceder a uma base de dados central para recolher esta informação, optou-se por percorrer duas páginas do Portal de modo a garantir que os dados eram corretamente extraídos.

A leitura das páginas foi efetuada em PHP, onde é feita uma *query* ao código fonte de cada página, enviando como parâmetro, a classe que pretendemos ver retornada. Após o retorno da classe, é utilizada a função *getElementByTagName()* de forma a poder extrair o conteúdo que será manipulado, para que este seja inserido na base de dados.

Na Figura 33, está representado o processo de extração do ID, *Jackpot* e data do próximo sorteio.

```php
$IDSorteio="";
//IDSorteio + data + Jackpot
$classname="banners wideMedium";//div[@class='$class']
$nodes = $finder->query("//*[contains(concat(' ', normalize-space(@class), ' '), ' $classname ')]");
foreach ($nodes as $entry) {
$rows = $entry->getElementsByTagName("span");
$temp1 = substr($rows->item(0)->nodeValue,-10);
$temp3 =  explode("/",$temp1);
$temp4 =str_replace(" ","",substr($temp3[1],-2)."/". substr($temp3[0],-3));
$IDSorteio= $temp4;
$datasorteio="null";
$tttt=$rows->item(1)->nodeValue;
$tttt2 = substr($tttt,0,1);
        if($tttt2=="t" or $tttt2=="T"){
            $temp___2=  explode(",", $rows->item(1)->nodeValue);
            $datasorteio="te, " . substr($temp___2[0],-10);
            echo ("terça");
        }
        else{
            $temp___2=  explode(",", $rows->item(1)->nodeValue);
            $datasorteio="sx, " . substr($temp___2[0],-10);
            echo ("sexta");
        }
 }
 $JACKPOT="";
 $classname="showAwards";
$nodes = $finder->query("//*[contains(concat(' ', normalize-space(@class), ' '), ' $classname ')]");
 foreach ($nodes as $entry) {
  $rows = $entry->getElementsByTagName("span");
  foreach ($rows as $row){
 $JACKPOT= htmlentities($rows->item(1)->nodeValue);
 break;
 }
 break;
}
$result = mysql_query("Insert into sorteioseuro (IDSorteio, DataSorteio, Jackpot)
VALUES ('$IDSorteio','$datasorteio','$JACKPOT')");
```

Figura 33 - Excerto de código do *ExtratorSorteio.php*

4.2.2 Implementação do POS

A aplicação POS possui duas classes fundamentais que permitem a captura de dados por intermédio de um cartão NFC ou de um telemóvel *Android* permitindo, assim, proceder ao registo do EuroMilhões.

O processo de leitura de dados de um cartão NFC encontra-se representado na Figura 34, onde se utiliza um ciclo *While*(*true*) de forma a que esteja constantemente à espera de um cartão.

```
while (true) {
    terminal.waitForCardPresent(0);
    try {
        Card card = terminal.connect("*");
        CardChannel channel = card.getBasicChannel();

        CommandAPDU command = new CommandAPDU(new byte[] { (byte) 0xFF,
                (byte) 0xCA, (byte) 0x00, (byte) 0x00, (byte) 0x04 });
        ResponseAPDU response = channel.transmit(command);

        byte[] byteArray = response.getBytes();
        Thread.sleep(2000);

        String x = bytesToHex(byteArray);
        JanelaCompra sv = new JanelaCompra();
        sv.setVisibleCard(x);

    } catch (CardException e) {
        e.printStackTrace();
    }
}
```

Figura 34 –Leitura Cartão NFC

Como referido no subcapítulo 5.1.2.1, a comunicação entre o leitor e o cartão NFC é realizada através do envio de um comando APDU, por parte do leitor. Este comando desencadeia uma resposta por parte do cartão NFC, retornando um conjunto de *Bytes*. Dependendo do tipo de resposta que se pretende, o comando APDU pode ser alterado, podendo-se aceder inclusive à memória do cartão, se necessário. No âmbito deste projeto, foi utilizado o comando para retornar o código identificativo do cartão (ver Figura 24). O intuito desta solução, passa primeiramente por atribuir uma chave a cada cartão NFC e, consequentemente, aquando da passagem do cartão, o sistema associe a chave correspondente.

A Figura 35, abaixo representada, mostra o resultado após a passagem de dois cartões NFC diferentes.

Figura 35 – Janela de compra em modo leitura cartão

A implementação da leitura para os telemóveis *Android* foi completamente distinta. Neste processo, o telemóvel envia uma mensagem denominada NDEF contendo toda a informação

necessária para efetivar o registo da chave de jogo. Todo este processo de captura da mensagem NDEF é realizado com o auxílio da biblioteca *NFCTools*.

Na Figura 26 está representado o método responsável pela captura da mensagem NDEF enviada pelo *smartphone Android* e pelo lançamento da classe *JanelaCompra.java*, para poder realizar a compra do boletim.

```
public void onNdefMessages(Collection<Record> records) {
    for (Record record : records) {
        log.info(record.toString());
        JanelaCompra  sv = new JanelaCompra();
        sv.setVisiblePhone(record.toString());
    }

    if(replyListener != null)
        replyListener.onReply(records);
}
```

Figura 36 – Metodo onNdefMessages da classe POS_LeitorTelemovel.java

A captura do *Record* é validada e, consequentemente, é lançada a janela com a interface para completar o registo. Este *Record* pode conter dados *Multipurpose Internet Mail Extensions* (MIME) do tipo media, como por exemplo um vídeo, um URL, ou até mesmo a capacidade de efetuar o lançamento de uma aplicação customizada em *Android*.

A Figura 37 ilustra o resultado após a passagem do *smartphone Android* no leitor NFC.

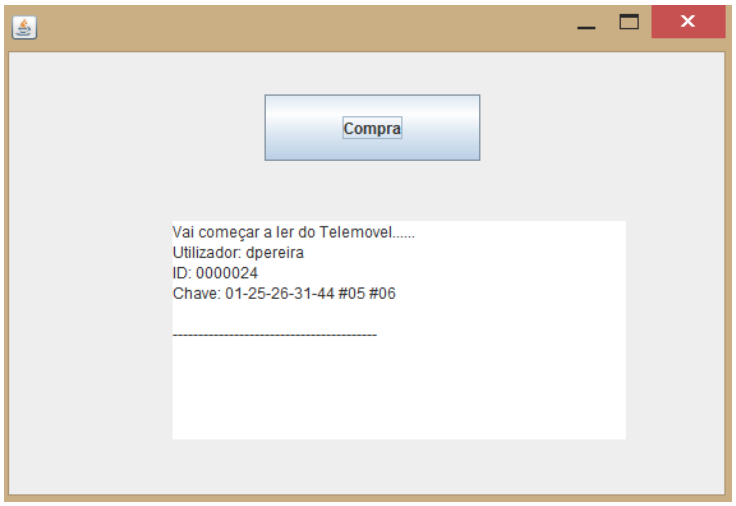


Figura 37 - Janela de compra em modo leitura cartão

4.2.3 Implementação do "Eurobolso"

O código da aplicação "Eurobolso" está subdividido em quatro pacotes. Esta divisão promoveu a organização e estruturação do código em funções específicas. A Figura 38 ilustra como foi processada esta divisão.

Figura 38 - Pacotes da aplicação

- ✓ **Pacote *Activity*** - É constituído por todas as classes que correspondem a janelas existentes na aplicação (*Activitys*).
- ✓ **Pacote *Adapters*** - Encontram-se as classes responsáveis por realizar a interação entre a camada aplicacional e a camada de dados; nomeadamente entre as *Activitys*, a base de dados aplicacional e o sistema *web* (*webservices*).
- ✓ **Pacote BO de *Business Object*** - É composto por todas classes que representam objetos da camada de negócio.
- ✓ **Pacote *Helper*** - Contém todas as classes auxiliares para o bom funcionamento da aplicação, nomeadamente classes auxiliares que contêm, por exemplo, métodos de validação.

4.2.3.1 Chamada das *Activitys*

Um recurso utilizado em toda a aplicação é a mudança de janela. Esta funcionalidade encontra-se presente em todas as janelas da aplicação, ora pela barra de menus presente no topo da janela, ora pelos botões que permitem a mudança de menus e consequentemente a mudança de janelas.

A Figura 39 mostra o excerto de código responsável pela inicialização e abertura de uma nova janela.

```
chaves = (Button) findViewById(R.id.menu_bt_chaves);
chaves.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        Intent paginaChaves = new Intent(
                "com.example.euromillions.Chaves");
        startActivity(paginaChaves);
    }
});
```

Figura 39 - Abrir nova *Activity* "Menu Chaves"

4.2.3.2 Processos *AsyncTask*

A utilização de processos assíncronos foi adotada sempre que foi necessário interagir com sistemas externos, nomeadamente nas chamadas aos *Web Services,* que servem de ligação entre a aplicação móvel e a base de dados central do sistema, ou com a própria base de dados aplicacional.

As classes que estendem *AsyncTask* são compostas por quatro métodos (*onProgressUpdate*, *onPreExecute*, *doInBackground*, *onPostExecute*), sendo a implementação do método *onProgressUpdate* opcional**.**

Embora a sua implementação seja opcional, este método assume bastante relevância pois é o único capaz de atualizar a interface, dando assim ao utilizador um *status* de como a tarefa está a decorrer.

Em cada chamada a uma classe *AsyncTask* optou-se por utilizar o objeto *ProgressDialog*, para indicar ao utilizador as várias fases em que o processo vai passando até ficar concluído. A criação deste objeto é realizada no método *onPreExecute*, sendo a sua atualização, como descrito anteriormente, no método *onProgressUpdate*. Por fim, no método *onPosteExecute*, o objeto *ProessDialog* é fechado.

A Figura 40 representa a criação de um objeto *ProgressDialog*, nomeadamente no processo de criação de um novo utilizador.

```
@Override
protected void onPreExecute() {
    super.onPreExecute();
    pDialog = new ProgressDialog(FormRegisto.this);
    pDialog.setMessage("A criar novo Utilizador");
    pDialog.setCancelable(true);
    pDialog.show();
}
```

Figura 40 - Método *onPreExecute*() da classe *AsyncTask CreatNewUtilizador*

4.2.3.3 Base de Dados aplicacional (*SqlLite*)

A base de dados da aplicação "Eurobolso" é composta por seis tabelas: Utilizador, Chaves, Cartão, Prémios, Estatísticas e Registos.

Na tabela "Utilizador" encontra-se apenas o registo do utilizador do *smartphone* e esta tabela é criada quando o utilizador efetua o registo na aplicação. A tabela "Chaves" é responsável por armazenar todas as chaves que o utilizar cria. O campo responsável pela unicidade desta tabela é o nome. A tabela "Cartão" permite ao utilizador ter mais que um cartão de jogador podendo, deste modo, ter uma chave associada ao cartão A e outra chave associada ao cartão B. As tabelas "Prémios" e "Estatísticas" são tabelas que agrupam informações sobre os

resultados dos vários sorteios. Estas tabelas sofrem atualizações sempre que o utilizador opte por atualizar a aplicação. Por fim, a tabela "Registos", é uma tabela que vai sendo atualizada sempre que o utilizador efetua o registo de um novo Euromilhões no sistema, permitindo assim ter na aplicação um histórico de apostas no Euromilhões.

Cada uma das tabelas referidas representa diretamente uma classe na aplicação com o prefixo *DBAdapter*. Estas classes são as responsáveis pela manipulação dos registos das tabelas, contendo métodos de procura, inserção, atualização e eliminação.

A Figura 41 representa a classe *DBAdapter_Chaves*, responsável pela manipulação da tabela "Chaves". Esta classe possui seis métodos referentes à manipulação dos registos da tabela (*createChave*, *EliminaCjave*, *CursosToEvent*, *getallChaves*, *getChave* e *updateChave*), os métodos *open*() e *close*() comuns a todas as classes *DBAdapter*, responsáveis pela abertura e fecho da conexão à tabela e ainda o construtor da classe (*DBAdapter_Chaves*).

```
public class DBAdapter_Chaves {
    private SQLiteDatabase database;
    private DbHelper dbHelper;
    private String[] allColumns = { DbHelper.IdEuroKey, DbHelper.IDUtilizador,
            DbHelper.datachave, DbHelper.nomechave, DbHelper.valorchave };

    public DBAdapter_Chaves(Context context) {

    public void open() throws SQLException {

    public void close() {

    public Chave createChave(String IDUser, String Datachave, String nomechave,

    public void EliminaChave(int idchave) {

    private Chave cursorToEvent(Cursor cursor) {

    public Cursor getallChaves() {

    public Chave getChave(int idchave) {

    public int updateChave(Chave chave) {

}
```

Figura 41 - DBAdapter_Chaves.java

4.2.3.4 *Web Services*

A chamada dos *Web Services*, como referido no subcapítulo 4.2.3.2, é sempre realizada por intermédio de uma classe que estende a classe *AsyncTask*. Foi ainda criado o método *makeHttpRequest*, que retorna um objeto JSON e consome três parâmetros. A URL onde o serviço está alocado, o tipo de método que se trata (GET ou POST) e uma lista de parâmetros opcional que servem de *input* ao *Web Service*. Deste modo, todas as chamadas aos vários *Web Services* têm por base a chamada deste método.

A Figura 42, representa um excerto do método *makeHttpRequest*, nomeadamente o tratamento entre uma chamada GET ou POST.

```
public JSONObject makeHttpRequest(String url, String method,
        List<NameValuePair> params) {

    try {
        HttpParams httpParameters = new BasicHttpParams();
        HttpConnectionParams.setConnectionTimeout(httpParameters, 10000);
        HttpConnectionParams.setSoTimeout(httpParameters, 10000);

        if(method == "POST"){

            HttpClient httpclient = new DefaultHttpClient();
            HttpPost httpPost = new HttpPost(url);
            httpPost.setParams(httpParameters);
            httpPost.setEntity(new UrlEncodedFormEntity(params));
            HttpResponse httpResponse = httpclient.execute(httpPost);
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        }else if(method == "GET"){

            HttpClient httpclient = new DefaultHttpClient();
            String paramString = URLEncodedUtils.format(params, "utf-8");
            url += "?" + paramString;
            HttpGet httpGet = new HttpGet(url);
            httpGet.setParams(httpParameters);
            HttpResponse httpResponse = httpclient.execute(httpGet);
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();
        }
```

Figura 42 - Excerto do método *makeHttpRequest*

Um exemplo da utilização deste método está representado na Figura 43, abaixo representada, onde os parâmetros introduzidos são um URL a apontar para o *Web Service* criar novo utilizador, uma lista de campos de Utilizador (*Email*, Primeiro Nome, Ultimo Nome, Chave de Segurança e Localidade) e o método POST do protocolo *http*.

```
String url_newUser = "http://XXX.XXX.X.XX/AppEuroMillions/create_user.php"; //URL

List<NameValuePair> params = new ArrayList<NameValuePair>(); //Parametros
params.add(new BasicNameValuePair("email", email));
params.add(new BasicNameValuePair("primeiro_nome", primeiro_nome));
params.add(new BasicNameValuePair("ultimo_nome", ultimo_nome));
params.add(new BasicNameValuePair("chave", chave_seguranca));
params.add(new BasicNameValuePair("localidade", localidade));

JSONObject json = jsonParser.makeHttpRequest(url_newUser,
        "POST", params); //invoca
```

Figura 43 - Chamada ao *makeHttpRequest* para criar novo Utilizador

O retorno do método *makeHttpRequest* retorna um objeto JSON, que posteriormente é convertido num objeto do tipo “Utilizador” e inserido na base de dados aplicacional.

4.2.3.5 Comunicação com o leitor NFC, ACR 122 U

A classe responsável por garantir a interação entre o dispositivo móvel e o leitor NFC ACR 122 U é a *Activity NFC.java*. Esta classe utiliza duas classes auxiliares presentes no pacote aplicacional *Helper*, denominadas *NfcHelper.java* e *NfcManager.java*.

Estas classes fornecem métodos de verificação e validação dos requisitos necessários para se poder realizar a comunicação entre os dois dispositivos, nomeadamente ao nível do *hardware*, analisando se o dispositivo móvel possui a classe *Adapter* NFC e das funcionalidades ativas, validando se o dispositivo está com a funcionalidade NFC ativa. Sem estas duas premissas, não é possível estabelecer comunicação bidirecional entre os dois sistemas.

O processo de criação da mensagem a ser enviada pelo dispositivo móvel ao leitor NFC é ilustrado na Figura 44, onde estão representados dois métodos responsáveis.

```
public String getCartao_Chave() {

    Chave chave = new Chave();
    Cartao card = new Cartao();

    card=datasource_cartao.getCard(myapp.getIdcard());
    chave = datasource_chaves.getChave(Integer.parseInt(card.getChave()));

    String message = "ID: " + card.getIdCartao() + " - Chave: " + chave.getValorchave();
    return message;
}

@Override
public NdefMessage createNdefMessage(NfcEvent event) {
    Log.i(TAG, "createNdefMessage");
    String text = getCartao_Chave();
    NdefMessage msg = new NdefMessage(new NdefRecord[] {
            createMimeRecord("EuroMilhoes phone - ", text) });
    return msg;
}
```

Figura 44 - Criação da Mensagem NDEF

O método *getCartão_Chave* acede à base de dados aplicacional e retorna um texto composto pelo ID do Cartão do Utilizador, concatenado com o valor da chave desse mesmo cartão. Este método é posteriormente utilizado no *creatNdefMessage* para compor a mensagem a enviar pelo dispositivo móvel ao leitor.

O envio é efetuado no método *onCreate*, responsável pela inicialização da *Activity*. A Figura 45 ilustra o método onde primeiramente é efetuada uma validação para identificar se o dispositivo móvel possui o *Adapter* NFC. Esta validação em caso positivo, coloca a mensagem disponível para envio, em caso negativo, retorna uma mensagem no ecrã, indicando que não tem NFC disponível.

```java
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    Log.i(TAG, "onCreate");
    setContentView(R.layout.activity_nfc);
    mInfoText = (TextView) findViewById(R.id.textView);
    mInfoText.setTextColor(Color.parseColor("#008000"));

    // Check do NFC adapter
    mNfcAdapter = NfcAdapter.getDefaultAdapter(this);
    if (mNfcAdapter == null) {
        mInfoText = (TextView) findViewById(R.id.textView);
        mInfoText.setText("Não tem nfc disponivel.");
    }

    if (com.example.euromillions_cgiwork.NfcManager.nfcPresent(this)) {

        mNfcAdapter.setNdefPushMessageCallback(this, this);

        mNfcAdapter.setOnNdefPushCompleteCallback(this, this);
    }
}
```

Figura 45 - Método onCreate da classe NFC.java

4.3 *Layout* do protótipo EuroBolso

Neste subcapítulo são apresentadas algumas imagens relativas à aplicação móvel EuroBolso, ilustrando algumas das suas funcionalidades.

O registo de um novo utilizador na aplicação é um processo que requer validação por parte do servidor, verificando a unicidade do endereço de *e-mail*. Após esta validação, os dados do utilizador são inseridos inicialmente na base de dados do servidor e posteriormente na base de dados aplicacional.

A informação sobre o estado do processo é apresentada ao utilizador como mostra a figura 46. A Figura 46 ilustra três estados que ocorrem neste processo representados da esquerda para a direita pela seguinte ordem: validação do *e-mail* e ligação ao servidor; inserção do novo registo de utilizar e, por fim, criação do registo na base de dados aplicacional.

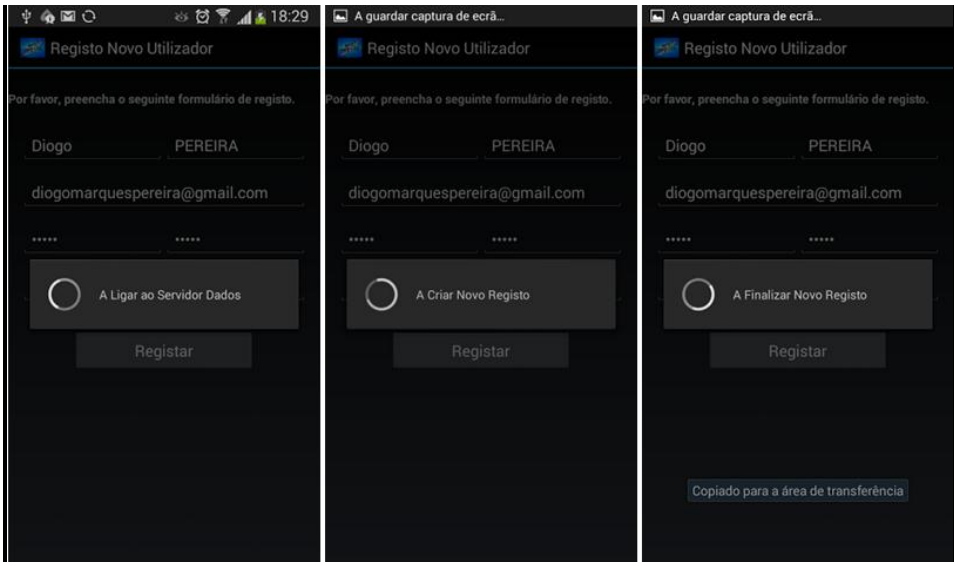


Figura 46 - Estados do ecrã "Registo Utilizador"

Após o registo ser concluído com sucesso, o utilizador é automaticamente redirecionado para um novo ecrã. A Figura 47 representa o menu de entrada da aplicação. No topo desta página, estão dispostos os vários *links* que permitem ao utilizador aceder a outros conteúdos na aplicação. Para além desta informação, são apresentadas três informações referentes ao próximo sorteio do Euromilhões:

- ✓ O valor do próximo *Jackpot;*
- ✓ Uma contagem decrescente, indicando quanto tempo falta até ao início do próximo sorteio;
- ✓ Um estado de Ligado/Desligado indicando se o utilizador já se encontra com o registo efetuado para o próximo sorteio.

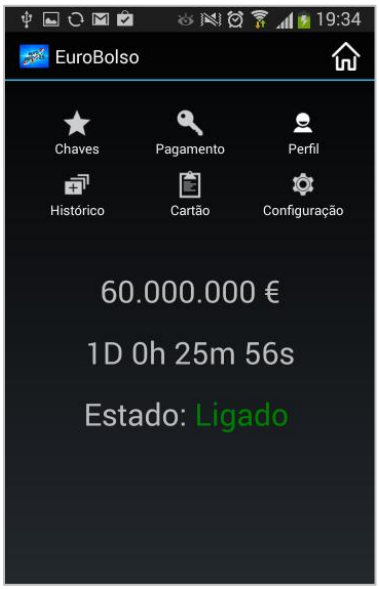


Figura 47 – Ecrã inicial da aplicação

Selecionando a opção “Chaves”, o utilizador acede a uma página onde pode escolher quais os números e estrelas que pretende atribuir à sua nova chave. Este processo encontra-se validado, não permitindo ao utilizador gravar uma nova chave sem atribuir um nome ou selecionar mais do que cinco números ou duas estrelas.

A Figura 48 apresenta do lado esquerdo, o ecrã para a criação de uma nova chave e do lado direito, a mensagem de erro, indicando que o utilizador não pode selecionar mais do que cinco números.

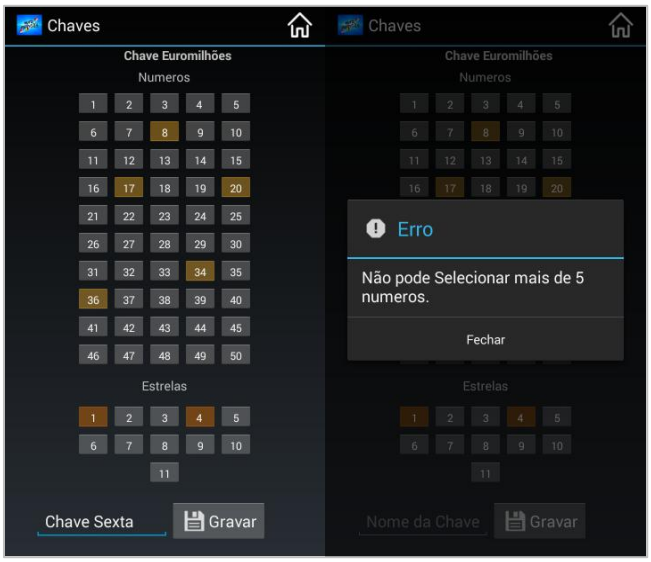


Figura 48 – Ecrã “Criar Chave”

Após a criação de um nova chave, o utilizador pode selecionar a mesma de forma a esta ficar ativa para o próximo registo que efetuar junto de um *POS*. Esta atribuição é efetuada no ecrã "Cartão".

A Figura 49 ilustra este processo de atribuição. O utilizador tem ao seu dispor uma lista de chaves previamente criadas por si, tendo que selecionar uma delas para ela ficar ativa.

Para além desta função, esta página apresenta ainda algumas informações relativas ao utilizador. Uma funcionalidade a acrescentar no futuro é a inclusão de uma "carteira virtual" para os utilizadores, podendo estes efetuar os pagamentos e receber os prémios nesta mesma carteia. Esta funcionalidade encontra-se já em esboço neste ecrã.

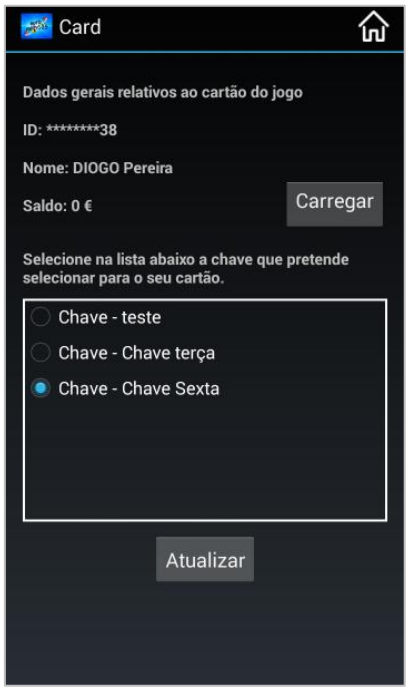


Figura 49 – Ecrã "Cartão"

O registo do boletim é efetuado no ecrã "Pagamento". O utilizador pode aceder a este ecrã em qualquer momento, desde que tenha a funcionalidade NFC ligada no seu dispositivo. A Figura 50 apresenta dois estados deste mesmo ecrã. À esquerda temos o ecrã visível no seu estado normal e à direita temos o ecrã após efetuar um registo junto do POS.

Figura 50 – Ecrãs de “Pagamento NFC”

5 Conclusão

O trabalho que conduziu a esta tese começou por uma fase de análise do problema, seguida de investigação, e depois passando à implementação, onde se exploraram várias hipóteses para certos pontos da especificação e desenvolvimento de uma arquitetura para um sistema de registo de boletins do Euromilhões.

A arquitetura é constituída por três componentes principais, uma aplicação construída para *Android*, um conjunto de serviços *web* desenvolvidos em *php* e uma aplicação responsável pela captura das mensagens enviadas pelo dispositivo móvel desenvolvida em *Java*.

Com o desenvolvimento desta aplicação, pretendeu-se encontrar uma maneira de facilitar a aposta no Euromilhões, bem como um acesso mais rápido e direto a diversas informações alusivas a este jogo.

Os portugueses são os europeus que, atualmente realizam mais apostas neste jogo, pelo que a criação de uma aplicação que facilite as apostas e a consulta de informações se mostra bastante útil. Ao contrário das aplicações já existentes do jogo do Euromilhões, a aplicação Eurobolso desenvolvida neste projeto, permitirá o registo da chave de jogo em postos com a tecnologia NFC disponível.

Esta aplicação por intermédio de todas as funcionalidades que apresenta, poderá ainda abrir oportunidades de implementação em diversas áreas, sendo possível replicar o sistema para a área de restauração, hotelaria, entre outras, onde os cartões com barras magnéticas e/ou chips poderão ser substituídos pelos telemóveis pessoais dos utilizadores.

5.1 Trabalhos Futuros

No futuro e pensando no comodismo do utilizador final, será interessante criar uma carteira de jogo virtual. Deste modo, será possível efetuar o registo e compra do boletim de forma automática. Adicionalmente, ao invés do utilizador ter que se deslocar a um posto de revenda do Euromilhões para levantar o seu prémio, poderá ser dada ao utilizador a possibilidade de este mesmo prémio ser creditado na sua carteia virtual.

Outro aspeto a considerar no futuro, será abranger outros sistemas operativos, nomeadamente o IOS e o *Windows*, aproveitando a tendência que tem sido seguida por parte dos fabricantes de dispositivos móveis ao incluir o NFC com funcionalidade em grande parte dos seus novos dispositivos.

A segurança é sem dúvida outro ponto a ser alvo de trabalho no futuro, podendo utilizar algoritmos de encriptação na camada aplicacional para assim encriptar os dados enviados e garantir a confidencialidade dos registos.

Um outro ponto a melhorar é o *design* da interface. Tratando-se de um protótipo, esta aplicação não é para uso final por parte do utilizador como tal, o seu "aspeto" sofrerá enormes alterações de forma a ir de encontro com as normas e diretrizes para a criação de uma boa interface.

Referências

[Canalys, 2013]	Canalys – Top iOS and Android apps largely absent on Windows Phone and BlackBerry 10, http://www.canalys.com/newsroom/top-ios-and-android-apps-largely-absent-windows-phone-and-blackberry-10#sthash.1Pw4uTN3.dpuf [último acesso: julho 2014]
[Business Insider, 2013]	Business Insider – How Android Grew To Be More Popular Than The iPhone, http://www.businessinsider.com/history-of-android-2013-8?op=1 [último acesso: julho 2014]
[Developers, 2014]	Developers – Dashboards, http://developer.android.com/about/dashboards/index.html [último acesso: julho 2014]
[IDC Analyse the Future, 2013]	IDC – Smartphone OS Market Share Q2, 2014, http://www.idc.com/prodserv/smartphone-os-market-share.jsp [último acesso: julho 2014]
[HIPERSUPER, 2014]	Gonçalves R. – Mobilidade é uma prioridade para 8 em cada 10 empresas, http://www.hipersuper.pt/2014/03/07/mobilidade-e-uma-prioridade-para-8-em-cada-10-empresas/ [último acesso: julho 2014]
[FLURRY, 2014]	Flurry – Apps Solidify Leadership Six Years into the Mobile Revolution, http://www.flurry.com/bid/109749/Apps-Solidify-Leadership-Six-Years-into-the-Mobile-Revolution#.VElmGPnF8Uy [último acesso: agosto 2014]
[NCFNearFieldCommunication, 2014]	NFCNearFieldCommunication – History, http://www.nfcnearfieldcommunication.org/history.html [último acesso: agosto 2014]
[InfoWESTER, 2014]	Alecrim E. – O que é NFC (Near Field Communication), http://www.infowester.com/nfc.php [ultimo acesso: agosto 2014]
[Howstuffworks, 2014]	Strickland J. – How Near Field Communication Works, http://electronics.howstuffworks.com/near-field-communication.htm [ultimo acesso: agosto 2014]
[RapidNFC, 2014]	RapidNFC – Which NFC Chip, http://rapidnfc.com/which_nfc_chip [ultimo acesso: agosto 2014]
[ShowMeTecch, 2012]	Gordilho J. O. – Guia Completo para Near Field Communication, http://showmetech.band.uol.com.br/guia-completo-sobre-nfc/ [ultimo acesso: agosto 2014]
[Android Authority, 2013]	Triggs R. – What is NFC & How Does It Works, http://www.androidauthority.com/what-is-nfc-270730/ [ultimo acesso: agosto 2014]
[NFCForum, 2014]	NFCForum – NFC Forum Technical Specifications, http://members.nfc-forum.org/specs/spec_list/ [ultimo acesso: agosto 2014]
[Microsoft, 2012]	Microsoft – Understanding NFC Data Exchange Format (NDEF) messages, http://developer.nokia.com/community/wiki/Understanding_NFC_Data_Exchange_Format_(NDEF)_messages
[Now.Hear.This., 2014]	Technology Transport – Say goodbye to Oyster cards, smartphones are swiping them out, http://now-here-this.timeout.com/2014/04/24/say-goodbye-to-oyster-cards-

smartphones-are-swiping-them-out/ [ultimo acesso: agosto 2014]

[ECMA, 2014] ECMA – Introducing JSON, http://json.org/ [ultimo acesso: agosto 2014]

[W3schools, 2014] W3schools – Web Services Tutorial, http://www.w3schools.com/Webservices/default.asp [ultimo acesso: agosto 2014]

[DEVMEDIA, 2014] Silva L. G. – Serviços RESTful com Apache CXF e Camel, http://www.devmedia.com.br/servicos-restful-com-apache-cxf-e-camel/26849 [ultimo acesso: agosto 2014]

[LearnREST, 2014] Elkenstein M. – What is REST, http://rest.elkstein.org/ [ultimo acesso: agosto 2014]

[GitHub, 2014] GitHub – Grundid7NFCTools, https://github.com/grundid/nfctools [ultimo acesso: agosto 2014]

[SQLite, 2014] SQLite – Categorical Index Of SQLite Documents, http://www.sqlite.org/docs.html [ultimo acesso: agosto 2014]

[Developers, 2014] Developers – Get The Android SDK, http://developer.android.com/sdk/index.html [ultimo acesso: agosto 2014]