

2015

Instituto Politécnico de Coimbra

INSTITUTO SUPERIOR DE ENGENHARIA DE COIMBRA

IZIPU: Plataforma de Marketing

MESTRADO EM INFORMÁTICA E SISTEMAS

AUTORA | Diana Raquel Tavares Pinheiro

ORIENTADORA | Doutora Teresa Raquel Corga Teixeira da Rocha

Coimbra, setembro 2015



**Departamento
de Engenharia Informática e de Sistemas**

IZIPU: Plataforma de Marketing

Relatório de Estágio apresentado para obtenção do grau de
mestre em Informática e Sistemas

Autor

Diana Raquel Tavares Pinheiro

Orientadores

Doutora Teresa Raquel Corga Teixeira da Rocha

Professora adjunta ISEC

Engenheiro Paulo Jorge da Silva Pinto

Punchline

DEDICATÓRIA

À memória de Teresa Gonçalves Dias, dos meus avô e Padrinho,
à minha família, por todo apoio e dedicação ao longo deste percurso
e
ao Ricardo, por toda a compreensão e motivação.

AGRADECIMENTOS

O presente relatório testemunha o término de mais uma etapa importante na minha vida - curso de Mestrado iniciado há dois anos. Apesar do mérito da conclusão de um curso recair tendencialmente na figura de quem o termina, não é expectável que alguém consiga atingi-lo sozinho pelo que se torna imperativo que agradeça aqueles que mais contribuíram para levar um projeto destes a um bom porto.

O espaço necessariamente limitado impede-me de expressar, de forma exaustiva, o meu agradecimento a quem, direta ou indiretamente me apoiou durante este curso e me impeliu sempre a continuar. Por este motivo, foi imperativo encontrar um critério de seleção que limitasse o número de pessoas citadas, ciente que, qualquer que fosse o critério adotado, não conseguiria evitar a injustiça a que serão sujeitos os que ficam de fora. O critério que me pareceu adequado foi o de citar apenas as pessoas diretamente relacionadas com a realização do estágio curricular na sua dimensão pessoal (família), institucional (ISEC) e empresarial (*Punchline*). Assim sendo, o meu sincero agradecimento:

À **Professora Doutora Teresa Raquel Corga Teixeira da Rocha**, por toda a dedicação e orientação ao longo do estágio. O meu muito obrigado pelo profissionalismo e pela total disponibilidade que sempre demonstrou.

Ao **Engenheiro Paulo Jorge da Silva Pinto**, o meu profundo agradecimento pela orientação e apoio incondicional que não só aumentou o meu conhecimento científico mas que - sem dúvida igualmente importante - estimulou o meu desejo de querer saber sempre mais e a vontade de fazer melhor.

Ao **Doutor Ricardo Miguel Freire Lopes**, o meu sincero agradecimento pela oportunidade que me proporcionou de iniciar e participar em todas as fases do desenvolvimento da plataforma de *marketing* IZIPU. Reconheço com gratidão não só a confiança que me depositou ao longo destes meses mas também, o sentido de responsabilidade que me incutiu em todas as fases deste projeto.

A um grupo selecionado de **Professores do ISEC**, felizmente demasiado grande para listar individualmente mas que foram o espelho da competência, do profissionalismo e do zelo durante o tempo em que frequentei esta instituição. Tenho a certeza que sabem quem são...

Quero ainda agradecer à minha família, em especial **pais e irmã**, que desde o primeiro momento apoiaram incondicionalmente esta etapa. Um enorme obrigado por acreditarem sempre em mim, naquilo que faço e por todos os ensinamentos que me transmitiram ao longo da vida. Espero que esta etapa, agora terminada, possa, de alguma forma, retribuir e compensar todo o carinho, apoio e dedicação que sempre me ofereceram.

Por fim, um agradecimento especial ao **Ricardo** que, com a sua presença tranquilizadora, me manteve motivada nas muitas horas de estudo e não me deixou desesperar nalgumas épocas de exame particularmente aborrecidas. Obrigado pela compreensão do mau humor consequente da fase final desta nova etapa. Sem ti, não teria terminado o mestrado com esta determinação!

RESUMO

A área do *marketing* é um dos sectores que sempre acompanhou os avanços tecnológicos verificados ao longo dos tempos, sendo incontestável que o uso da tecnologia é imprescindível para potenciar o investimento financeiro e temporal de qualquer empresa. Hoje em dia, desenvolver e aplicar *marketing* através da web é primordial para qualquer empresa, razão pela qual é necessário saber adaptá-lo a todas as ferramentas disponíveis num mundo cada vez mais virtual.

A *Punchline* é uma empresa que pretende atuar no mercado do *marketing* e, por esse motivo, o objetivo deste estágio concretizou-se no desenvolvimento da plataforma IZIPU que tem como finalidade permitir o envio de campanhas de *marketing* através de diversos canais (SMS, MMS, *Email*, Voz e Fax) para um elevado número de subscritores, de forma automática, bem como obter estatísticas detalhadas sobre as campanhas enviadas. A plataforma de marketing IZIPU foi desenvolvida em PHP e uma das suas características é ser responsiva, ou seja, adapta-se a diversos tamanhos de ecrãs.

Palavras-chave: plataforma de *marketing*, framework php, Laravel, aplicação *web*.

ABSTRACT

Marketing is undoubtedly one of the areas that could be case studied as early adopters of the technological advances of the last decades. Technological mastery is now unavoidable to get the most out of the available financial and human budget. Creating digital based marketing campaigns is inevitable in the digital age and skills in electronic marketing tools are highly sought after by small and big businesses alike.

Punchline aims to operate in this very competitive market and, for that reason, this internship's goal was to develop IZIPU. IZIPU is a marketing web application that enables customers to create and manage marketing campaigns by SMS, MMS, email, voice and fax to a large base of subscribers effortlessly while providing comprehensive reporting tools to assess market penetration. IZIPU was developed in PHP and its layout is responsive to multiple screen sizes.

Keywords: marketing platform, php framework, Laravel, web application.

ÍNDICE

1 Introdução	1
1.1 Enquadramento	2
1.1.1 Marketing Digital	2
1.1.2 A Punchline	6
1.1.3 Problema e motivação	8
1.2 Estudo de viabilidade do projeto	8
1.3 Objetivos do estágio	10
1.4 Estrutura do documento	11
2 Análise de plataformas de marketing	13
2.1 Plataformas de marketing existentes	13
2.2 Análise comparativa	14
2.3 Política de Spam	18
3 Análise de requisitos	21
3.1 Requisitos funcionais	21
3.2 Requisitos não funcionais	22
3.3 Casos de uso	23
3.3.1 Atores do sistema	23
3.3.2 Diagramas de casos de uso	24
3.3.3 Lista de casos de uso	26
3.3.4 Descrição dos casos de uso	29
3.4 Diagrama de atividade	38
3.5 Prototipagem	40
4 ferramentas de desenvolvimento	41
4.1 Frameworks PHP	41
5 Arquitetura	45
5.1 Modelo de dados	46
5.2 Camada lógica	47
5.3 Camada de apresentação	48
6 Implementação	51
6.1 Controlo de versões	51
6.2 Metodologia de trabalho	52
6.3 Módulos Desenvolvidos	53
6.3.1 Campanhas	54
6.3.2 Envio das campanhas	64
6.3.3 Relatórios	72

6.3.4	Utilizadores	75
6.3.5	Autenticação	77
6.3.6	Subscritores	77
6.3.7	Créditos	80
6.3.8	Conta corrente	84
6.3.9	Remetentes	85
6.3.10	Equipa	88
6.4	website de apresentação (http://www.izipu.com)	90
6.5	Dependências	91
6.5.1	jQuery	91
6.5.2	Bootstrap	92
6.5.3	TinyMCE	92
6.5.4	Responsive File manager	93
6.5.5	Clickatell	93
6.5.6	Twillio	93
6.5.7	PamFax	94
6.5.8	Paypal	95
6.6	Testes	95
7	Conclusões	99
	Bibliografia	103
	Apêndice A – Mockups	107
	Apêndice B – Modelo físico	111

ÍNDICE DE FIGURAS

Figura 1.1: Logotipo da *Punchline*

Figura 2.1: Gráfico da ferramenta *Google Trends* que nos mostra os termos mais procurados nos últimos anos.

Figura 3.1: Diagrama de casos de uso do módulo criar campanha

Figura 3.2: Diagrama de casos de uso do módulo de envio

Figura 3.3: Diagrama de casos de uso do módulo de subscritores

Figura 3.4: Diagrama de casos de uso referente ao módulo de autenticação e configurações de perfil.

Figura 3.5: Diagrama de atividade do caso de uso Criar campanha

Figura 4.1 – Previsão das *frameworks* mais usadas em 2014.

Figura 5.1: Diagrama do modelo MVC

Figura 6.1: *Board* da plataforma de *marketing* IZIPU usada na fase de desenvolvimento.

Figura 6.2: Ecrã que permite ao utilizador importar o seu template

Figura 6.3: Ecrã onde o utilizador pode criar ou editar o conteúdo de uma *newsletter*

Figura 6.4: Ecrã que permite ao utilizador criar uma mensagem de texto

Figura 6.5: Excerto de código referente ao envio de SMS e de *Flash SMS*

Figura 6.6: Ecrã que permite ao utilizador criar uma mensagem de multimédia

Figura 6.7: Captura de ecrã onde se verifica o upload de ficheiros

Figura 6.8: Ecrã que permite ao utilizador criar toda a comunicação entre a máquina e o interlocutor.

Figura 6.9: Captura de ecrã que mostra a criação de uma campanha de voz

Figura 6.10: Esquema que representa o funcionamento do módulo de voz.

Figura 6.11: Excerto do código que permite fazer a chamada de voz através da API Twilio

Figura 6.12: Ecrã da plataforma de *marketing* IZIPU onde o utilizador escolhe o método de envio e para listas de subscritores deseja enviar a campanha.

Figura 6.13: Excerto do código que transmite à API PamFax o subscritor para quem será enviado o fax.

Figura 6.14: Excerto de código que executa a validação e envio do fax

Figura 6.15: Relatório do envio de uma campanha de *email* com diversas estatísticas.

Figura 6.16: Ecrã que permite criar conta no IZIPU

Figura 6.17: Ecrã da plataforma de *marketing* IZIPU onde é possível configurar a assinatura.

Figura 6.18: Excerto de código que permite encriptar a password

Figura 6.19: Ecrã da plataforma de *marketing* IZIPU que permite ao utilizador adicionar diversos campos extra a cada lista de subscritores

Figura 6.20: Ecrã onde o utilizador pode consultar os créditos da conta

Figura 6.21: Ecrã com os meios de pagamento

Figura 6.22: Excerto de código do método que invoca o *PayPal*

Figura 6.23: Ecrã de pagamento no *PayPal*

Figura 6.24: Ecrã onde o utilizador pode consultar os seus movimentos

Figura 6.25: Excerto de código jQuery *DataTables*

Figura 6.26: Excerto de código HTML com o formulário para adicionar remetente

Figura 6.27: Excerto de código *jQuery* que trata o evento do botão 'Adicionar'

Figura 6.28: Excerto de código da função que recolhe os dados enviados no pedido e que cria um registo na tabela 'remetentes'

Figura 6.29: Ecrã onde é possível visualizar e adicionar remetentes

Figura 6.30: Seletor de remetentes aquando da criação da campanha

Figura 6.31: Ecrã que permite ao utilizador convidar outros utilizadores para a sua equipa e atribuir-lhes permissões.

Figura 6.32: Website de apresentação do IZIPU

Figura 6.33: Implementação de uma *tooltip* usando Bootstrap

Figura 6.34: Folha com o resumo do plano de testes da plataforma

Figura 6.35: Folha do plano de testes onde é visível uma *issue*.

ÍNDICE DE TABELAS

TABELA 2.1: Tabela com um resumo dos preços, envios de <i>emails</i> e número de subscritores que cada plataforma permite.	14
TABELA 3.1: Atores do sistema	23
TABELA 3.2: Atores do sistema e respetivos casos de uso do componente de login e menu pessoal.....	27
TABELA 3.3: Atores do sistema e respetivos casos de uso do componente de subscritores	28
TABELA 3.4: Atores do sistema e respetivos casos de uso da componente criação de campanhas	28
TABELA 3.5: Caso de uso - Fazer login.....	29
TABELA 3.6: Caso de uso – Criar conta	30
TABELA 3.7: Caso de uso – Recuperar palavra passe	30
TABELA 3.8: Caso de uso - Confirmar conta	31
TABELA 3.9: Caso de uso – Visualizar termos e condições	31
TABELA 3.10: Caso de uso - Visualizar dashboard.....	31
TABELA 3.11: Caso de uso - Ver menu pessoal.....	32
TABELA 3.12: Caso de uso - Ver dados do perfil.....	32
TABELA 3.13: Caso de uso - Editar dados da conta	33
TABELA 3.14: Caso de uso - Eliminar utilizador.....	33
TABELA 3.15: Caso de uso - Analisar conta corrente	34
TABELA 3.16: Caso de uso - Ver remetentes.....	34
TABELA 3.17: Caso de uso - Configurar assinatura	35
TABELA 3.18: Caso de uso - Eliminar remetente	35
TABELA 3.19: Caso de uso - Gerir equipa	36
TABELA 3.20: Caso de uso - convidar utilizador	36
TABELA 3.21: Caso de uso – criar campanha.....	37
TABELA 3.21: Caso de uso – criar campanha.....	38

ABREVIATURAS

API	Application Programming Interface	78
CMS	Content Management System.....	106
IVR	Interactive Voice Response	26
JS	JavaScript	89
JSON	JavaScript Object Notation	78
MMS	Multimedia Messaging Service	26
MVC	Model View Controller	61
SMS	Short Message Service	26
SVN	Subversion.....	68
SWOT	Strengths Weaknesses Opportunities Threats	24
TTS	Text to Speech.....	26

1 INTRODUÇÃO

A evolução constante que se tem vindo a verificar ao longo dos anos no domínio das tecnologias conduziu a alterações nos modelos de negócio e nos processos de trabalho das empresas de uma forma geral, sendo a área do *marketing* uma das vertentes que sempre acompanhou os avanços tecnológicos.

Ao longo dos últimos anos, a quantidade de informação e conhecimento gerada pela web tem influenciado todos os utilizadores desta ferramenta, provocando determinadas alterações na sociedade, nomeadamente na maneira como as pessoas se relacionam entre si e como utilizam toda a informação disponível [1]. Face a esta evolução, as empresas foram adaptando a sua forma de abordar o mercado. Para desenvolver o seu negócio, estas tiveram que aprender a conviver com os meios tradicionais e com os digitais que, em conjugação, vieram permitir uma comunicação mais eficaz junto das novas gerações de consumidores. De facto, as tecnologias da informação ligaram o mundo e as barreiras de comunicação deixaram de existir [2].

Assistimos assim a um impacto significativo e revolucionário tanto para os consumidores como para as empresas, causado por uma evolução constante nos meios digitais. Atualmente, desenvolver e aplicar *marketing* através da web, é primordial para qualquer empresa e, por esse motivo, é necessário saber adaptar o *marketing* a todas as ferramentas disponíveis num mundo cada vez mais virtual.

Uma das empresas que está atenta a esta tendência e que se quer especializar nesse nicho de mercado é a *Punchline*, onde decorreu o presente estágio curricular, inserido no âmbito da unidade curricular de 'Estágio ou Projeto Industrial' do Mestrado em Informática e Sistemas ministrado no Departamento de Engenharia Informática e de Sistemas do Instituto Superior de Engenharia de Coimbra, tendo sido realizado sob a

orientação do Engenheiro Paulo Jorge da Silva Pinto e da Professora Teresa Raquel Rocha.

1.1 ENQUADRAMENTO

Procede-se seguidamente ao enquadramento do presente estágio, começando por uma introdução à área do *marketing* digital em que este se insere, apresentado em seguida a empresa em que foi realizado e a motivação que esteve na base da sua existência.

1.1.1 Marketing Digital

Marketing é uma palavra proveniente da língua inglesa, apesar de estar intrínseca à cultura mundial. Em inglês, *market* significa mercado e *marketing* pode ser traduzido como mercadologia, um estudo das causas, objetivos e resultados que são gerados através das diferentes formas como se lida com o mercado. Compra, venda e troca de serviços, produtos ou ideias.

Segundo Philip Kotler¹, define-se *marketing* como a ciência e a arte de explorar, criar e entregar valor para satisfazer as necessidades de um mercado-alvo com lucro; identifica necessidades e desejos não realizados; define, mede e quantifica o tamanho do mercado identificado e o potencial lucro; sugere os segmentos que a empresa melhor é capaz de servir e promove os produtos e serviços adequados [3].

Assim sendo, pode afirmar-se que o *marketing* é, no fundo, um conjunto de estratégias, técnicas e práticas que tem como principal objetivo agregar valor às determinadas marcas ou produtos, a fim de lhes atribuir uma maior importância para um determinado público-alvo, ou seja, os consumidores.

¹ Professor universitário e autor de numerosos livros e artigos na área do *marketing* sendo a sua principal obra o "*Marketing Models*", considerado por muitos como a bíblia do *marketing* [47].

Ainda antes do aparecimento do *marketing* digital, a única forma de promoção dos produtos eram os meios tradicionais, que consistiam na divulgação em jornais, televisão, rádios e livros, e que não permitiam uma interação direta com o público. Atualmente existe um esforço para se gerar interatividade e participação com os consumidores sendo neste contexto que entra o uso da internet. De facto, esta interação é estimulada maioritariamente pelo uso do *email* ou das redes sociais, passando assim de um ambiente totalmente passivo proporcionado pelos meios tradicionais, para um contexto onde o público atua de forma ativa, influenciando as decisões das empresas, sendo isto que torna o marketing digital tão poderoso.

Para que se entenda a conceção do *marketing* e dos novos meios de comunicação torna-se necessário que seja melhor compreendida a história da evolução da internet.

O aparecimento do *marketing* digital coincide com o surgimento da internet. Em 1996, a internet já tinha atingido os 40 milhões de utilizadores no mundo e esta rápida evolução desde o seu nascimento deve-se ao facto desta oferecer uma qualidade inatingível que até então não existia na vida moderna em contraste com a alienação provocada pelos antigos media onde a voz do consumidor não tinha tanta relevância [4]. Para que se entenda a conceção do *marketing* e dos novos meios de comunicação torna-se necessário que seja compreendida a história da evolução da internet.

O aparecimento do *marketing* digital coincide com o surgimento da internet. Em 1996, a internet já tinha atingido os 40 milhões de utilizadores no mundo e esta rápida evolução desde o seu nascimento deve-se ao facto desta oferecer uma qualidade que até então não existia na vida moderna, em contraste com a alienação provocada pelos antigos *media* onde a voz do consumidor não tinha tanta relevância [4].

Desde o surgimento da internet, os primeiros e grandes portais que sobreviveram à queda da bolsa de valores, fenómeno este conhecido como o “estouro da bolha da internet”, tais como o Yahoo e MSN, perceberam que simplesmente anunciar com *banners* tradicionais convertidos para os *media*

online não era suficiente e o grande mistério era: “Como converter o tráfego de utilizadores em consumidores? [5]”

O primeiro modelo de negócio rentável na internet foi o mecanismo de buscas GoTo.com, criado por Bill Gross no final dos anos 90. Neste motor de busca, os utilizadores podiam pesquisar sobre assuntos do seu interesse e obtinham misturados, quer os resultados da pesquisa através de palavras-chave relevantes, quer os da pesquisa paga, em que os anunciantes pagavam para aparecerem nos resultados [6]. Este modelo de apresentação foi um grande erro pois os utilizadores na sua maioria confiam mais no que lhes é recomendado do que no que é pago para aparecer. O Google, que hoje é o principal motor de busca, aprimorou este modelo rentável da internet separando os resultados da pesquisa, onde os dados da pesquisa orgânica são explicitamente separados da pesquisa paga. Os utilizadores clicam porque sabem que é uma referência e que o resultado da pesquisa não é propaganda.

A primeira ação de *marketing* digital foi levada a cabo em 1994 pelo escritório de advocacia “Canter e Siegel”, nos Estados Unidos. Esta tentativa de marketing ficou mundialmente conhecida pelo insucesso e má reação obtida, tanto da parte dos utilizadores da rede onde a ação foi realizada, como da parte da imprensa da época. O escritório de advocacia procurou um meio de comunicação barato e enviou um anúncio que oferecia os seus serviços para a obtenção do “*Green Card*”, um cartão de lotaria, a mais de sete mil grupos de discussão, violando uma regra desses grupos que ainda hoje está em vigor em muitos fóruns na internet que é a proibição de publicar campanhas publicitárias. Esta tentativa de marketing digital abalou em muito a reputação do escritório de advocacia [7].

Não é por acaso que o *marketing* digital está a crescer exponencialmente. Para além do número de cibernautas aumentar de dia para dia de forma acelerada, cada vez mais os utilizadores vivem ligados à internet e, por este motivo, este tipo de *marketing* oferece uma série de vantagens:

- Alcance global: Se uma determinada empresa pretende realizar uma campanha que envolve, por exemplo, *outdoors* ou *flyers*, para além de uma limitação geográfica, existe também um impacto ao nível do custo pois são utilizados meios físicos. Por sua vez, no mundo virtual qualquer empresa pode ter um alcance global uma vez que não existem limites geográficos.
- Interatividade: Uma das grandes vantagens do *marketing* digital é o facto da comunicação não ser unilateral permitindo assim a interação com o público. Com isto, para além de conseguir captar mais a atenção dos utilizadores, a empresa consegue ir conhecendo melhor o seu público-alvo e criar ações cada vez mais eficientes.
- Métricas: Na internet quase tudo é mensurável, nomeadamente a quantidade de visitas, performance das campanhas, quantidade de cliques em anúncios, entre outros. Esta quantidade de dados gerados é de extrema importância para que as empresas de *marketing* possam criar campanhas mais eficientes. Esta é de facto uma das melhores vantagens do marketing digital.
- Tempo real: Como tudo na internet hoje em dia, o *marketing* digital pode ser analisado em tempo real. Desta forma, as empresas de *marketing* podem adaptar-se rapidamente de acordo com a performance de uma determinada campanha, evitando assim investir dinheiro em algo que não esteja com o sucesso esperado.
- Segmentação: Esta vantagem está diretamente relacionada com o facto de ser possível obter uma enorme quantidade de dados de utilizadores na internet o que permite que as empresas façam investimentos de *marketing* apenas direcionados ao seu público-alvo aumentando assim a eficiência das campanhas.

Com base no exposto, conclui-se que o *marketing* digital não consiste apenas em anunciar na internet, mas sim falar na linguagem do público-alvo, nomeadamente do que é do seu interesse, e estimular os

clientes a falarem bem da sua marca, recomendando-a de forma espontânea simplesmente porque se sentem satisfeitos.

1.1.2 A Punchline

A *Punchline* define-se como uma empresa multifacetada, especializada em diversas áreas como o *design* gráfico, *design* multimédia e *marketing*. Procura assim desenvolver soluções que respondam às necessidades específicas dos seus clientes, idealizando soluções criativas e inovadoras que contam a história dos seus projetos.

A empresa foi criada em Julho de 2007, em Lisboa, por Ricardo Lopes e outro colaborador, sendo que em 2009 ocorreu uma reestruturação, ficando Ricardo Lopes como Diretor Geral. A sede foi entretanto deslocada para Coimbra, para a Incubadora do Instituto Pedro Nunes, e a agência para a Baixa de Coimbra.

A *Punchline* é uma agência criativa de serviço completo, com uma equipa capaz de dar resposta aos projetos mais ambiciosos, desde projetos ao nível do *design* e multimédia, como a projetos de desenvolvimento de *software*. Esta dedica-se a trabalhos de *design* – publicidade, divulgações e *flyers*, multimédia, desenvolvimento de *software* e *webhosting*. Para tal, a *Punchline* trabalha de “todas as formas e feitios” desde que garantindo os valores éticos e protegendo os interesses dos clientes, nomeadamente confidencialidade e salvaguarda dos dados.

Com efeito, a *Punchline* defende os seguintes valores:

- Desempenho e competência: toda a empresa e os seus colaboradores esforçam-se e dedicam-se inteiramente aos projetos do cliente com o objetivo de o ver integralmente satisfeito;
- Prazer: toda a equipa gosta do que faz, e como referem, “vestimos com gosto a camisola do cliente”. Só deste modo é possível manter acesa a chama da criatividade e obter os melhores resultados;

- Pontualidade: a *Punchline*, considera que os prazos são para cumprir religiosamente, sendo fulcral cumprir o que foi prometido ao cliente;
- Criatividade e inovação: a *Punchline* pretende acabar com estereótipos, inovando em cada trabalho e apresentando sempre novas ideias aos seus clientes.

A *Punchline* tem como objetivo fulcral a satisfação do cliente procurando assim a melhoria contínua da apresentação de serviços disponíveis, tendo por base o envolvimento e empenho de todos os colaboradores da empresa.

Deste modo, a política de qualidade da *Punchline* assenta nos seguintes princípios:

- Surpreender o cliente com propostas inovadoras e criativas;
- Proteger os interesses do cliente quanto à confidencialidade e salvaguarda dos dados dos seus projetos;
- Estabelecer uma relação de proximidade e duradoura com os clientes e fornecedores, garantindo a satisfação de todos os intervenientes;
- Proporcionar a todos os colaboradores a possibilidade de desenvolver as suas capacidades e competências através da participação em formações e eventos relevantes de forma a garantir a sua motivação e valorização;
- Cumprir com rigor toda a legislação relativa à atividade da *Punchline*, as normas de referência e outros requisitos que a empresa subscreva no âmbito da Qualidade e da Segurança e Saúde no trabalho.



Figura 1.1: Logotipo da *Punchline*

1.1.3 Problema e motivação

Encontrando-se a *Punchline* inserida num grupo de empresas, algumas das quais na área da formação com grande necessidade de enviar *newsletters* a promover as suas ações a um elevado número de subscritores, sentia-se a falta de uma ferramenta que facilitasse o envio das mesmas. Surgiu então a ideia inicial de se criar internamente uma aplicação para este efeito.

No entanto, a *Punchline* vislumbrou aqui uma oportunidade de ir mais longe e colocar a ferramenta no mercado das plataformas de *marketing digital*, pelo que se decidiu pelo desenvolvimento de uma plataforma mais completa que possibilitasse a distribuição de *marketing* através de diversos meios digitais

Nesse seguimento surgiu o presente estágio cuja finalidade é a criação de uma plataforma *online* que permita a empresas e a particulares, criar, gerir e difundir campanhas de *marketing* através de um conjunto diversificado de *media* (chamadas telefónicas automatizadas, *Short Message Service*, *Multimedia Messaging Service*, correio eletrónico e mensagens de fax), bem como obter estatísticas detalhadas sobre cada uma delas, de forma a otimizar a comunicação, tanto de empresas como de particulares, com os seus clientes.

A expectativa da *Punchline* relativamente a este estágio é que a plataforma resultante venha a ser comercializada, permitindo assim à empresa aumentar o seu prestígio e a sua referência neste nicho de mercado que é o *marketing digital*.

1.2 ESTUDO DE VIABILIDADE DO PROJETO

Para verificar a viabilidade do projeto foi elaborada uma análise SWOT (*Strengths Weaknesses Opportunities Threats*) que permitiu ver quais os riscos a ter em conta e quais os problemas a resolver (ameaças), assim

como as vantagens e as oportunidades a explorar. Nomeadamente, foram identificados os seguintes aspetos:

Oportunidades:

- Participação num mercado em constante desenvolvimento;
- Mercado com fortes concorrentes, o que permite à empresa ter uma maior competitividade e conseqüentemente um maior empenho para se destacar e elevar ao melhor produto.

Ameaças:

- Entrada de novos concorrentes;
- Dificuldade de integração com algumas APIs;
- Mercado de *marketing* muito disputado.

Forças:

- User Interface e UX (*User Experience*) – plataforma de *marketing* apelativa ao olhar e intuitiva;
- Não possuir mensalidades obrigatórias;
- Disponibilizar funcionalidades essenciais para a gestão dos subscritores.

Fraquezas:

- Não existir informação sobre o projeto *online*.
- Ausência de patrocínios e afiliados.

Apesar das ameaças encontradas e descritas, a *Punchline* concluiu que este seria um projeto viável dando luz verde para que fossem iniciadas as próximas fases do processo de desenvolvimento do mesmo.

1.3 OBJETIVOS DO ESTÁGIO

Tal como referido anteriormente, o âmbito do estágio é a criação de uma plataforma *online* que permita a empresas e a particulares, criar, gerir e difundir campanhas de *marketing* através de um conjunto diversificado de *media*, bem como obter estatísticas sobre as diversas campanhas efetuadas.

Tendo em conta que a plataforma deverá ser totalmente modular, os objetivos principais do estágio consubstanciam-se no desenvolvimento dos seguintes módulos:

- Correio eletrónico – criar mensagens de correio eletrónico personalizadas, tipicamente sob a forma de *newsletters*;
- SMS – enviar ou agendar o envio de uma ou várias SMS personalizadas;
- MMS – criar uma campanha através de uma ou várias mensagens de multimédia. A sua personalização deve compreender todos os *media* suportados por este formato (imagem, som, texto ou vídeo);
- Resposta interativa de voz (IVR) – realizar campanhas através de um sistema IVR, capaz de responder e interagir com o interlocutor através de áudio pré gravado ou gerado dinamicamente com uma solução *text-to-speech* (TTS). As interações podem ser usadas para instruir o interlocutor das ações a tomar e o sistema pode ser enriquecido se suportar o reconhecimento das respostas vocais do interlocutor.
- Fax – criar e enviar campanhas através de fax, quer a preto e branco quer a cores.

Para todas as campanhas criadas e enviadas, independentemente do meio usado, deve ser possível obter um manancial de informação estatística de onde se destaca, por exemplo, e sem detrimento de outra que se revele importante, a seguinte:

- Quais os destinatários que receberam a campanha;
- Quais os destinatários que leram ou ouviram efetivamente a campanha;
- Qual a secção da campanha com mais cliques dos utilizadores;
- Em que momento do dia é que as campanhas têm mais impacto;
- Quais os destinatários que rejeitaram a chamada (para as campanhas de voz);
- Qual a parte de um *email* ou *newsletter* em que um destinatário mais clicou;
- Quais os destinatários que acederam a determinado conteúdo.

Finalmente, a plataforma a desenvolver deverá estar num segmento de mercado que corresponda tanto às pequenas e médias empresas, como a particulares que pretendam promover o seu produto/negócio. Destina-se a um público-alvo que procure o *marketing* digital como estratégia para a gestão de relacionamento com os potenciais clientes; que pretenda reduzir os custos das despesas de comercialização e aquisição de clientes; que deseje explorar e aproveitar o *marketing online* e que pretenda aumentar a eficácia dos esforços de *marketing*.

1.4 ESTRUTURA DO DOCUMENTO

O presente relatório encontra-se dividido nos seguintes capítulos:

1. **Introdução** (presente capítulo): é feito o enquadramento do trabalho de estágio, é apresentado o estudo de viabilidade do projeto a desenvolver e são definidos os principais objetivos do mesmo;

2. **Análise de plataformas de *marketing*:** é realizada uma análise comparativa das principais plataformas de *marketing* digital já existentes no mercado;
3. **Análise de requisitos:** é descrito todo o processo de levantamento de requisitos, são apresentados os diagramas de casos de uso com a descrição dos mais relevantes, e são mostrados os protótipos de interface;
4. **Ferramentas de desenvolvimento:** é feita uma análise a *frameworks PHP* e é definida aquela que servirá de apoio ao desenvolvimento da plataforma de *marketing* ZIPU;
5. **Arquitetura:** é descrita a arquitetura modular do sistema em três camadas;
6. **Implementação:** são detalhados os aspectos relativos à implementação, nomeadamente os módulos desenvolvidos e as várias opções tomadas ao longo do desenvolvimento. São também apresentadas todas as dependências utilizadas ao longo do desenvolvimento da plataforma;

Conclusões: são retiradas as conclusões mais pertinentes e projetado o trabalho futuro a desenvolver.

2 ANÁLISE DE PLATAFORMAS DE MARKETING

Neste capítulo apresenta-se uma análise às principais plataformas de *marketing* existentes no mercado, mostrando as suas funcionalidades mais importantes bem como os seus pontos mais fracos.

2.1 PLATAFORMAS DE MARKETING EXISTENTES

Sendo o mercado das plataformas de *marketing* é bastante amplo e com diversas alternativas, o estudo efetuado focou-se apenas no subconjunto das mais usadas e conhecidas do mercado, nomeadamente Egoi [8], Get Response [9], Mail Chimp [10], Aweber [11] e Klick Mail [12].

A tabela seguinte apresenta um resumo das características de cada ferramenta.

Nome	Preço/mês	Emails	Subscritores	Observações
Egoi	210€	500.000		O E-goi disponibiliza também os mesmos planos com pagamentos semestrais e anuais.
	375€	250.000		
	650€	1.250.000		
	1000€	2.500.000		
Get Response	Preço/mês	Emails	Subscritores	O Get Response ao contrário do E-goi possui um envio ilimitado de <i>emails</i> .
	12€	ilimitados	1.000	
	20€	ilimitados	2.500	
	35€	ilimitados	5.000	
	50€	ilimitados	10.000	
	120€	ilimitados	120.000	

Mail Chimp	Preço/mês	Emails	Subscritores	Para pacotes com elevado número de subscritores, o Mail Chimp disponibiliza ainda de outros pacotes.
	Grátis	12.000	Até 2.000	
	9.28€	Ilimitados	0 – 500	
	13.92€	Ilimitados	501 – 1.000	
	18.56€	Ilimitados	1.001 - 1.500	
	23.19€	Ilimitados	1.501 - 2.000	
Aweber	Preço/mês	Emails	Subscritores	Para um elevado número de subscritores, esta plataforma disponibiliza ainda outros pacotes.
	19€	Ilimitados	0 - 500	
	29€	Ilimitados	501 – 2.500	
	49€	Ilimitados	2.501 – 5.000	
	69€	Ilimitados	5.001 – 10.000	
Klick Mail	Preço/mês	Emails	Subscritores	No primeiro mês a Klick apenas cobra 0.30€ o que permite ao utilizador fazer um elevado número de envios de <i>emails</i> por um baixo custo.
	42€		10.000	
	84€		20.000	
	126€		30.000	
	210		50.000	

Tabela 2.1: Tabela com um resumo dos preços, envios de *emails* e número de subscritores que cada plataforma permite.

2.2 ANÁLISE COMPARATIVA

Através dos testes efetuados a cada uma das plataformas, foi possível observar que cada uma delas possui funcionalidades interessantes e úteis no envio de *marketing* através de variados meios.

Como estas plataformas são todas *web*, para testá-las apenas foi necessário criar registo semelhante a todas as outras plataformas *online* comuns bastando fornecer alguns dados como o nome, *email* e *password*.

As funcionalidades básicas de qualquer plataforma de *marketing* incidem nos módulos de gestão de contactos/subscritores, envios e agendamentos de campanhas de *marketing* e nos meios disponíveis para a produção das campanhas. Por este motivo, são as funcionalidades que poderão ter mais interesse para a *Punchline*.

O E-Goi é uma plataforma de *marketing* desenvolvida por uma equipa portuguesa e, conseqüentemente, está disponível com idioma português, entre outros. Esta plataforma possui um componente intuitivo para a criação de *Landing Pages*²; permite o envio de campanhas de *marketing* através de mensagens de texto, voz e de fax; disponibiliza com alguma frequência *workshops* à distância que instruem sobre o funcionamento de diversas funcionalidades. Uma grande vantagem encontrada nesta plataforma foi o facto de disponibilizar uma API de integração para qualquer aplicação, seja esta uma aplicação *web* desenvolvida pelo utilizador ou uma aplicação *Wordpress* através de *plugin* apropriado.

A plataforma *Get Response* disponibilizou recentemente uma versão em português e incluiu nas suas funcionalidades uma ferramenta de criação de formulários. A plataforma permite a administração de uma conta, ou seja, uma empresa poderá registar-se e associar à mesma conta os colaboradores que desejar, com a vantagem de todos eles partilharem os mesmos dados, inclusivamente o saldo.

O *Mail Chimp* apesar de ser das plataformas mais usadas a nível mundial permite apenas o envio de *marketing* através de um único meio (*newsletter/email*). Das plataformas analisadas, o *Mail Chimp* é a única que

² Página de entrada para uma campanha. Por outras palavras, é a página mostrada ao utilizador quando este clica em partes específicas da *newsletter* enviada por *email*.

permite ao utilizador enviar uma grande quantidade de *emails* (doze mil) por mês de forma gratuita. Mesmo sendo uma plataforma apenas direcionada para o *email marketing*, não significa que seja inferior às outras. Na verdade, possui um *plugin wordpress* para que seja possível integrar o envio de *emails*.

A plataforma *Aweber* apresenta-se apenas no idioma inglês e é também uma plataforma que disponibiliza apenas o envio de *marketing* através de um único meio (*email*). Da experiência tida com esta plataforma constatou-se que esta não é simples de configurar e é pouco intuitiva, presumindo sempre que o utilizador já tem experiência com este tipo de plataformas.

A *Klick Mail* apesar de ser uma plataforma que também só permite o envio através de campanhas de *email marketing* é bastante interessante. Comparada com as plataformas abordadas anteriormente, esta tem a vantagem de permitir ao utilizador efetuar o envio dos *emails* de forma segmentada. Ou seja, permite segmentar uma determinada lista de subscritores em 'sub-listas'. Outra vantagem da *Klick Mail* é permitir uma análise de cada *email* enviado e gerar relatórios estatísticos.

Para além de se terem experimentado as plataformas de *marketing* anteriormente referidas, efetuou-se um pequeno estudo de mercado na tentativa de averiguar qual a plataforma mais procurada. Para esta análise, usou-se uma ferramenta da *Google* bastante conhecida – *Google Trends*, que mostra aos utilizadores os termos mais procurados num determinado intervalo de tempo [13]. Analisando os termos *Aweber*, *Klick Mail*, *E-goi*, *Get Response* e *Mail Chim* num intervalo de dez anos, como se pode verificar na Figura 2.1, a percentagem de pesquisas por estes termos tem vindo de uma maneira geral, a aumentar o que comprova que as aplicações de *marketing* digital têm tido cada vez mais procura.



Figura 2.1: Gráfico da ferramenta *Google Trends* que nos mostra os termos mais procurados nos últimos anos.

Geralmente, a seleção de uma plataforma de *marketing* ou de outros produtos tem por base critérios como o custo, a quantidade de recursos, a facilidade de contratação e os meios de pagamento.

A equipa da *Punchline*, achou que seria interessante que a plataforma de *marketing* ZIPU adotasse a modalidade de adesão/pagamentos a créditos ou preço por *email*, ao invés de mensalidades, visto que é possível e muito provável que o anunciante pague mensalidades e, num dado mês, não realize um único envio.

Como consequência desta análise concluiu-se que iriam ser desenvolvidos módulos comuns a algumas plataformas abordadas, nomeadamente o módulo de construção e envio de campanhas de *emails*, o envio de campanhas através de mensagem de texto e a geração de relatórios com os dados estatísticos dos envios das campanhas.

Para diferenciar a sua plataforma das restantes, a *Punchline* decidiu que a plataforma de *marketing* ZIPU iria ter um módulo de IVR (*Interactive Voice Response*) que para além de permitir enviar apenas uma gravação de

voz normal, também permitirá ao utilizador criar uma chamada interativa onde obterá respostas aos eventos em função da tecla premida.

2.3 POLÍTICA DE SPAM

Hoje em dia, muitos utilizadores fazem uso das plataformas de *marketing* exclusivamente para fazer SPAM. A *Punchline* não pretende que a plataforma de *marketing* ZIPU seja desenvolvida para este efeito e, por este motivo, foi realizado um estudo sobre o tema SPAM e como se pode evitar que as campanhas desenvolvidas sejam consideradas SPAM. Mas afinal o que é o SPAM?

O SPAM é qualquer tipo de comunicação *online* não desejada [14]. Existem diversas formas de fazer SPAM sendo que a mais utilizada é através do *email*. No entanto, é possível fazer SPAM através de mensagens ou de redes sociais.

Segundo a *Out Marketing*, uma empresa especializada em *marketing*, para evitar que os servidores de *email* sejam usados para o envio de spam deve ter em conta as seguintes medidas [15]:

- Evitar comprar ou extrair listas de *emails* na net;
- Eliminar os *emails* que já não existem ou que são inválidos;
- Evitar certas palavras na construção de uma *newsletter*, tais como, promoção, saldo, compre, grátis, clique aqui, última oportunidade, entre outros.
- Evitar o uso de pontos de exclamação e frases com letras maiúsculas no conteúdo da *newsletter*.
- Não escrever o assunto do *email* em letras maiúsculas.
- Realizar o envio dos *emails* com *opt-in*

Segundo as regras básicas da área do *marketing*, para que uma empresa efetue o envio de campanhas de *email marketing* é necessário que a mesma tenha uma base de contactos qualificada e que seja *opt-in*. Ou seja, que todas as pessoas que estão incluídas nas listas de *emails* tenham

autorizado a receção de *emails* com campanhas, comunicados ou promoções de uma determinada empresa [16]. Existem diversas formas de captar os *emails* de pessoas interessadas tais como, por exemplo, a presença de um formulário no site de uma determinada empresa, para que quem estiver interessado em receber a *newsletter* possa inserir os seus dados e autorizar deste modo a empresa a enviar *emails*.

Para além do *opt-in* existe ainda o *double opt-in* que pode descrever-se como um reforço do *opt-in*. Isto é, não basta apenas a pessoa adicionar os seus dados no formulário de inscrição no *website* da empresa manifestando a sua intenção de receber os *emails*. Para que haja o *double opt-in* a empresa, após receber os dados iniciais de uma pessoa que mostrou o interesse em receber as suas campanhas, deve enviar um *email* a este mesmo utilizador com um *link* de confirmação para que seja feita uma dupla confirmação. Para que os dados sejam realmente confirmados é necessário que a pessoa aceda ao seu *email* e clique no *link* de confirmação enviado pela empresa reafirmando assim, pela segunda vez, o seu interesse em receber as campanhas [17].

Assim como existe uma forma de uma pessoa notificar o seu interesse em receber *newsletters*, também tem que existir uma forma de a mesma pessoa cancelar a receção de *newsletters*, procedimento que se designa por *opt-out*. O *opt-out* é o pedido de remoção da subscrição de uma determinada *newsletter*. Um sistema *opt-out* deve ser rápido e simples pois, com a mesma rapidez que o contacto de uma pessoa entra numa empresa, também deve sair. Por norma, em todas as *newsletters* enviadas está presente um *link* que permite ao utilizador remover a subscrição da *newsletter* enviada caso o deseje [18].

3 ANÁLISE DE REQUISITOS

Todo o processo de levantamento de requisitos e acompanhamento do projeto foi realizado através de reuniões semanais/quinzenais, conforme necessário, com as equipas de desenvolvimento e administrativa, as quais avaliavam o progresso do trabalho e sugeriam novas funcionalidades e correções de erros detetados.

Numa primeira reunião com ambas as equipas, foram levantados os principais requisitos a ter em conta. Um dos pré-requisitos seria a implementação de uma plataforma *web* para uso interno da *Punchline* e também para ser comercializada.

Deste modo, nesta secção são apresentados, numa primeira instância, todos os requisitos funcionais e não funcionais da plataforma de *marketing* IZPU. Numa segunda instância são apresentados todos os requisitos através de casos de uso, recorrendo a diagramas e à descrição detalhada dos principais casos de uso. Por fim, para uma visualização gráfica do que se pretende na plataforma são apresentados os protótipos de interface.

3.1 REQUISITOS FUNCIONAIS

Como requisitos funcionais foram identificados os seguintes:

- Implementação de uma plataforma de *marketing online* intuitiva que permita enviar campanhas de *marketing* através de diversos meios:
 - Deve ser possível enviar campanhas através de *newsletters* (*email*).
 - Deve ser possível criar e enviar campanhas através de Mensagens de texto (SMS) e de Mensagens de multimédia (MMS).

- Deve ser possível criar campanhas de voz em que seja possível interagir com o interlocutor. Este tipo de chamadas de voz é designado de IVR (*Interactive Voice Response*).
- Deve ser possível enviar mensagens através de Fax.
- Gestão dos subscritores
 - Deve ser possível agrupar os subscritores por listas (conjuntos de subscritores)
 - Deve ser possível criar, editar e remover subscritores.
 - Deve ser possível criar, editar e remover as listas de subscritores.
- Gestão da conta
 - Deve ser possível editar todos os dados do perfil de cada utilizador;
 - Deve ser possível convidar utilizadores para a conta do utilizador;
 - Deve ser possível configurar uma assinatura pessoal para que os *emails* enviados por convites tenham essa assinatura;
 - Deve ser possível atribuir permissões aos utilizadores convidados.
- Créditos
 - Deve ser possível consultar os créditos disponíveis e permitir a compra de novos créditos;
 - Deve ser possível realizar a compra de créditos através do meio de pagamento Paypal.

3.2 REQUISITOS NÃO FUNCIONAIS

Para além dos requisitos funcionais que definem como é que os módulos devem agir de acordo com diferentes cenários de utilização, também foram identificados diversos requisitos não funcionais que limitam o modo de utilização da plataforma:

- **Segurança:** Não deve existir contaminação dos dados entre as diversas contas de utilizadores;
- **Usabilidade:** A plataforma deve assegurar uma navegação simples e intuitiva para o utilizador.
-

3.3 CASOS DE USO

Nesta secção introduzem-se os vários atores do sistema e que casos de uso poderão executar. São ainda apresentados quatro diagramas de casos de uso mas, devido ao seu elevado número, apenas são detalhados os principais..

3.3.1 Atores do sistema

Na Tabela 3.1 são apresentados os diferentes atores do sistema e são descritos os seus papéis dentro da aplicação.

Nome	Descrição
Utilizador não autenticado	Este utilizador não poderá usar qualquer funcionalidade do sistema à exceção de criar um registo e realizar autenticação.
Utilizador autenticado	Este utilizador poderá utilizar todas as funcionalidades do sistema às quais tiver permissões.
Administrador	Possui acesso a todas as funcionalidades do sistema.
Autor	Este utilizador tem acesso a todas as funcionalidades à exceção do envio de campanhas.
Visualizador	Este utilizador apenas tem permissão para visualizar os relatórios das campanhas enviadas.
Gestor	Possui acesso a todas as funcionalidades à exceção da faturação.

Tabela 3.1: Atores do sistema

3.3.2 Diagramas de casos de uso

De forma a tornar os casos de uso mais legíveis, estes foram repartidos de forma lógica em quatro diagramas:

- Os dois primeiros diagramas representam os casos de uso referentes ao módulo criar campanha;
- O terceiro diagrama representa os casos de uso referentes ao módulo das listas de subscritores;
- O quarto diagrama representa os casos de uso referentes ao módulo de login e configurações de perfil.

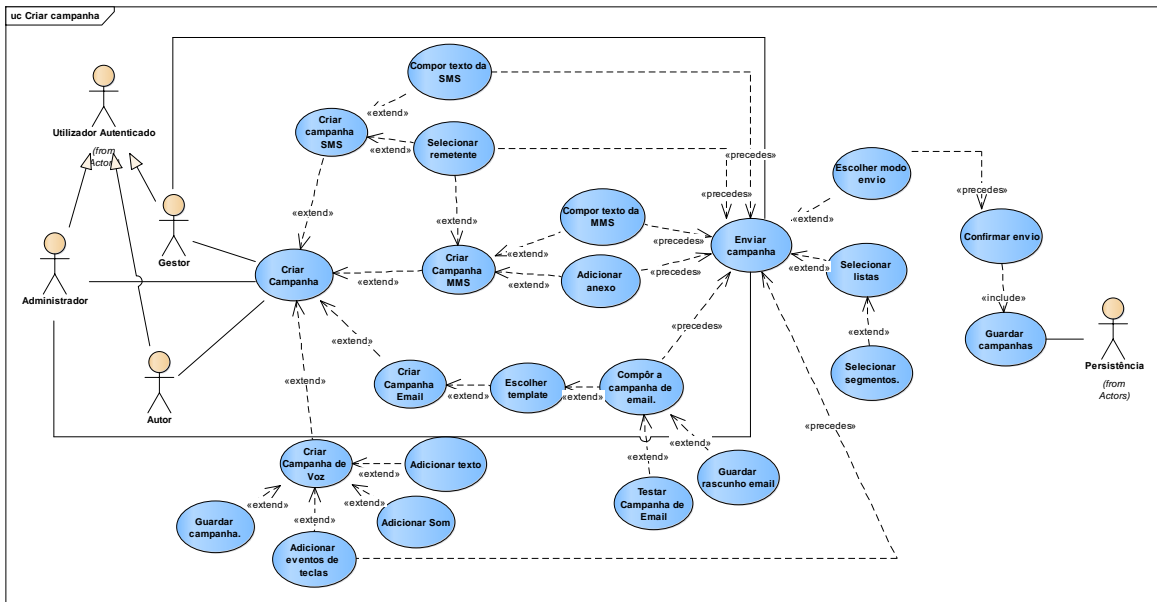


Figura 3.1: Diagrama de casos de uso do módulo criar campanha

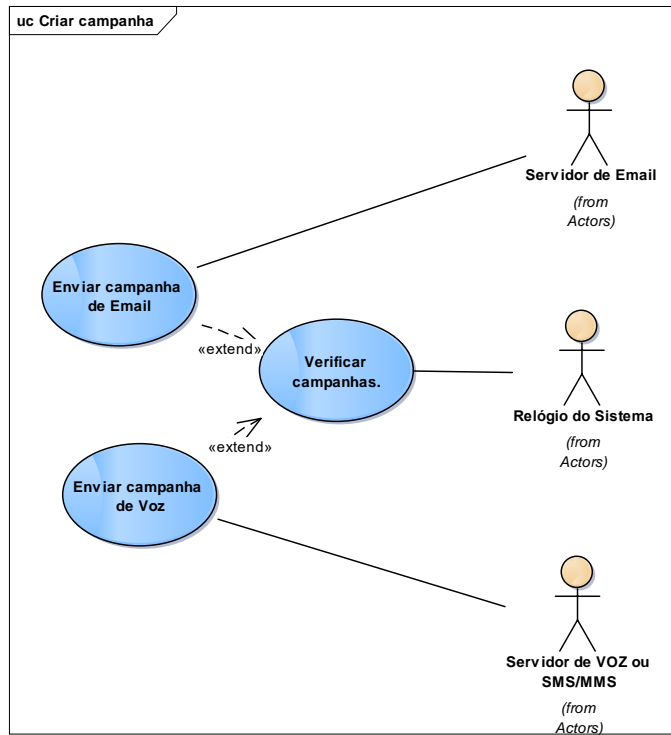


Figura 3.2: Diagrama de casos de uso do módulo de envio

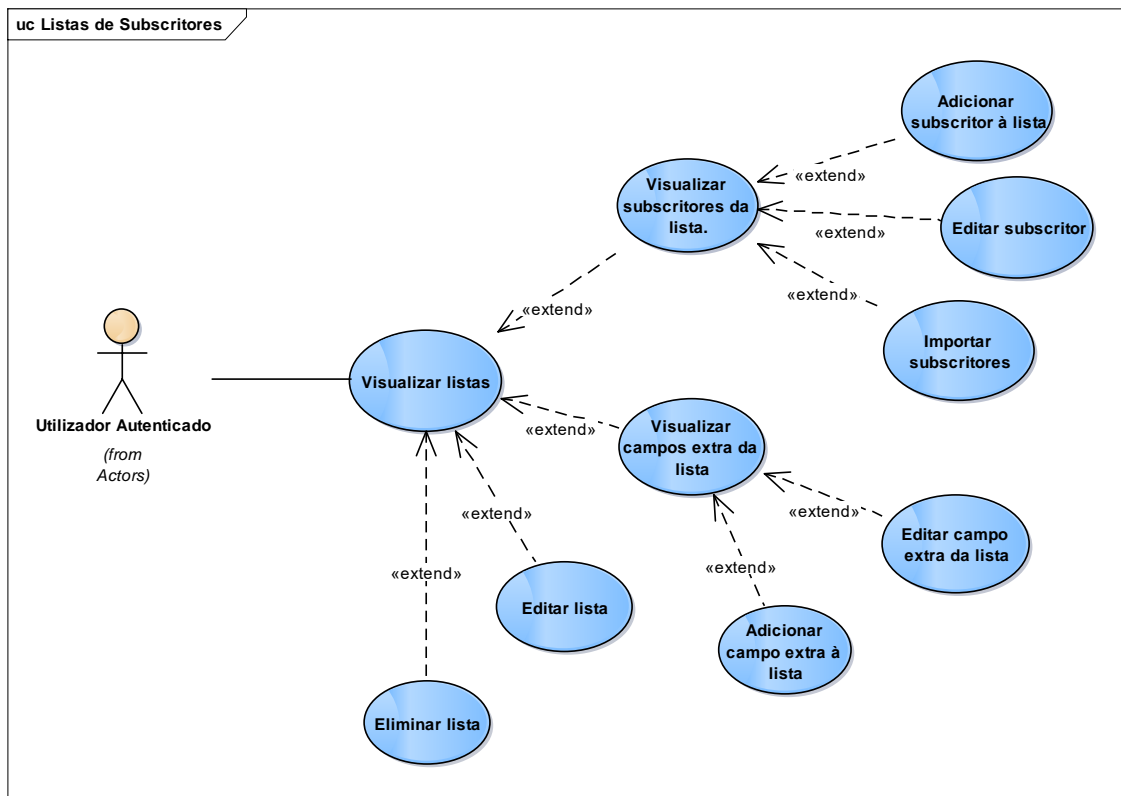


Figura 3.3: Diagrama de casos de uso do módulo de subscritores

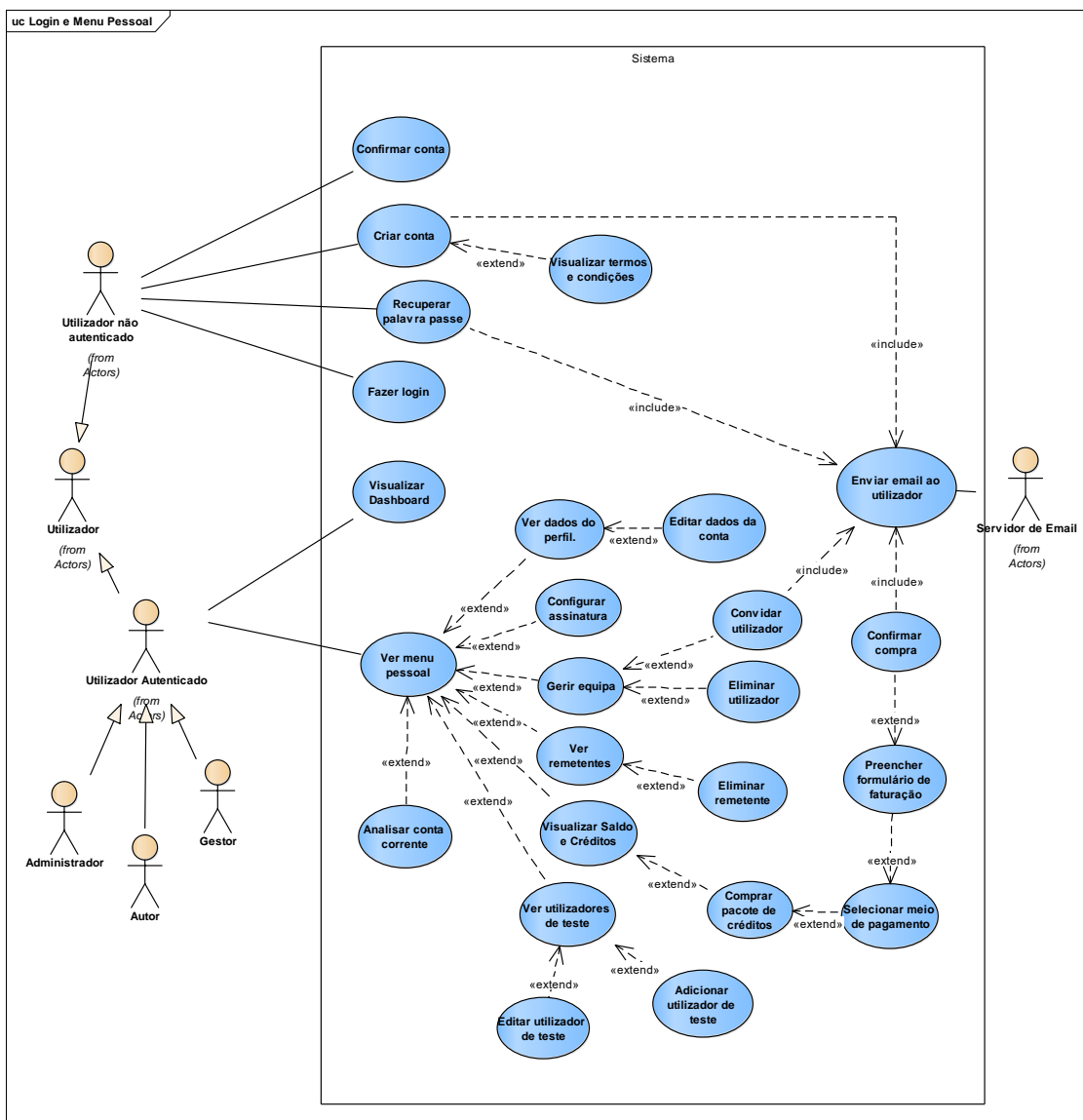


Figura 3.4: Diagrama de casos de uso referente ao módulo de autenticação e configurações de perfil.

3.3.3 Lista de casos de uso

Na Tabela 3.2 são listados os casos de uso dos componentes de login e menu pessoal da plataforma de *marketing* IZIPU, divididos em função dos atores que neles intervêm.

Ator primário	Caso de uso
Utilizador não autenticado	<ol style="list-style-type: none"> 1. Fazer login. 2. Criar conta. 3. Recuperar palavra passe. 4. Confirmar conta. 5. Visualizar termos e condições.
Utilizador autenticado	<ol style="list-style-type: none"> 6. Visualizar Dashboard. 7. Ver menu pessoal. 8. Ver dados do perfil. 9. Editar dados da conta.
Administrador	<ol style="list-style-type: none"> 10. Eliminar utilizador. 11. Analisar conta corrente. 12. Preencher formulário de faturação. 13. Visualizar saldo e créditos. 14. Selecionar meio de pagamento. 15. Comprar pacote de créditos. 16. Confirmar compra.
Administrador, Autor, Gestor	<ol style="list-style-type: none"> 17. Ver remetentes. 18. Configurar assinatura. 19. Eliminar remetente. 20. Adicionar Utilizador de teste.
Administrador, Gestor	<ol style="list-style-type: none"> 21. Gerir Equipa. 22. Convidar utilizador.
Servidor de <i>Email</i>	<ol style="list-style-type: none"> 23. Enviar <i>email</i> ao utilizador.

Tabela 3.2: Atores do sistema e respetivos casos de uso do componente de login e menu pessoal

A Tabela 3.3 lista os casos de uso da componente de subscritores da aplicação web, divididos em função dos atores que neles intervêm.

Ator primário	Caso de uso
Utilizador autenticado	<ol style="list-style-type: none"> 24. Visualizar listas. 25. Eliminar lista. 26. Editar lista. 27. Visualizar campos extra da lista. 28. Adicionar campo extra à lista. 29. Editar campo extra da lista. 30. Visualizar subscritores da lista. 31. Adicionar subscritor à lista.

	32. Editar subscritor. 33. Importar subscritores.
--	--

Tabela 3.3: Atores do sistema e respetivos casos de uso do componente de subscritores

A Tabela 3.4 lista os casos de uso da componente de criação de campanhas da aplicação web, divididos em função dos atores que neles intervêm.

Ator primário	Caso de uso
Administrador, Gestor e Autor	34. Criar campanha. 35. Criar campanha SMS. 36. Compor texto da SMS. 37. Selecionar remetente. 38. Criar campanha de <i>email</i> . 39. Escolher template. 40. Compor a campanha de <i>email</i> . 41. Testar campanha de <i>email</i> . 42. Criar campanha de voz. 43. Adicionar texto. 44. Adicionar Som. 45. Adicionar eventos de teclas. 46. Guardar campanha. 47. Criar campanha MMS. 48. Compor texto da MMS. 49. Adicionar anexo. 50. Selecionar modo de envio. 51. Selecionar listas. 52. Selecionar segmentos. 53. Confirmar envio.
Administrador e Gestor	54. Enviar campanha.
Relógio do sistema	55. Verifica as campanhas.
Servidor de <i>email</i>	56. Verificar campanhas.
Servidor de Voz ou SMS/MMS	57. Enviar campanha.

Tabela 3.4: Atores do sistema e respetivos casos de uso da componente criação de campanhas

3.3.4 Descrição dos casos de uso

Como a plataforma de *marketing* IZIPU é constituída por uma grande quantidade de casos de uso decidiu-se descrever apenas alguns casos. Assim sendo, na presenta secção são descritos os casos de uso 1 a 11 e o caso de uso 17, 18, 19, 21, 22, 34, e 38.

UC 1: Fazer login

ID do caso de uso:	1
Nome do caso de uso:	Fazer login
Atores:	Utilizador não autenticado
Descrição:	Sempre que um utilizador acede à aplicação web tem que fazer login para ser reconhecido pelo servidor.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. O utilizador não autenticado faz login preenchendo o formulário de autenticação e premindo o botão 'Entrar'. Se as credenciais forem reconhecidas o utilizador passa a estar autenticado.2. O sistema apresenta a dashboard da aplicação web.
Fluxos alternativos:	<ol style="list-style-type: none">3. Se as credenciais não forem reconhecidas é mostrada uma mensagem de erro.

Tabela 3.5: Caso de uso - Fazer login

UC 2: Criar conta

ID do caso de uso:	2
Nome do caso de uso:	Criar conta
Atores:	Utilizador não autenticado
Descrição:	Permite ao utilizador criar um registo na aplicação web.
Pré-condições:	Não existem.

Pós-condições:	Caso de uso 'Enviar <i>email</i> ao utilizador'.
Fluxo normal:	<ol style="list-style-type: none"> 1. O utilizador acede à plataforma e prime o botão 'Criar conta'. 2. O sistema apresenta uma página com os campos necessários para a criação da conta. 3. O utilizador preenche todos os campos, aceita os termos e condições e prime o botão 'Criar Conta'.
Fluxos alternativos:	<ol style="list-style-type: none"> 3. O utilizador não aceita os termos e condições. 4. O sistema notifica o utilizador da obrigatoriedade de aceitar os termos e condições da plataforma.

Tabela 3.6: Caso de uso – Criar conta

UC 3: Recuperar palavra passe

ID do caso de uso:	3
Nome do caso de uso:	Recuperar palavra passe
Atores:	Utilizador não autenticado
Descrição:	Permite ao utilizador recuperar a sua palavra passe.
Pré-condições:	Não existem.
Pós-condições:	Caso de uso 'Enviar <i>email</i> ao utilizador'.
Fluxo normal:	<ol style="list-style-type: none"> 1. O utilizador acede à plataforma e prime o link 'Esqueceu-se da sua password?' 2. O sistema apresenta a página de recuperação de password. 3. O utilizador preenche o campo de <i>email</i> e prime o botão 'Enviar'.

Tabela 3.7: Caso de uso – Recuperar palavra passe

UC 4: Confirmar conta

ID do caso de uso:	4
Nome do caso de uso:	Confirmar conta
Atores:	Utilizador não autenticado
Descrição:	O utilizador antes de poder realizar a sua primeira autenticação tem que confirmar o seu registo.
Pré-condições:	Não existem.
Pós-condições:	Não existem.

Fluxo normal:	<ol style="list-style-type: none"> 1. O utilizador acede ao seu <i>email</i> com que se registou na plataforma e prime o link de confirmação. 2. O Sistema apresenta uma página a notificar o utilizador de que já se encontra registado e que pode realizar autenticação.
---------------	--

Tabela 3.8: Caso de uso - Confirmar conta

UC 5: Visualizar termos e condições

ID do caso de uso:	5
Nome do caso de uso:	Visualizar termos e condições
Atores:	Utilizador não autenticado
Descrição:	Para criar uma conta o utilizador tem que ler e aceitar os termos e condições da plataforma.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none"> 1. O utilizador acede à plataforma e prime o botão 'Criar conta'. 2. O Sistema apresenta a página com o formulário necessário para criar a conta. 3. O utilizador deve clicar no link 'Termos e Condições do Izipu'. 4. O sistema apresenta uma dialog com os termos e condições da plataforma.

Tabela 3.9: Caso de uso – Visualizar termos e condições

UC 6: Visualizar dashboard

ID do caso de uso:	6
Nome do caso de uso:	Visualizar <i>dashboard</i>
Atores:	Utilizador autenticado
Descrição:	O utilizador pode visualizar a <i>Dashboard</i> sempre que pretender.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none"> 1. Na plataforma, o utilizador clica no menu lateral esquerdo em 'Dashboard'. 2. O sistema apresenta ao utilizador a Dashboard.

Tabela 3.10: Caso de uso - Visualizar dashboard

UC 7: Ver menu pessoal

ID do caso de uso:	7
Nome do caso de uso:	Ver menu pessoal
Atores:	Utilizador autenticado
Descrição:	Visualização do menu pessoal com todas as acções possíveis.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na plataforma, o utilizador clica no no seu nome no canto superior direito.2. O sistema apresenta ao utilizador o seu menu pessoal com diversas opções.

Tabela 3.11: Caso de uso - Ver menu pessoal

UC 8: Ver dados do perfil

ID do caso de uso:	8
Nome do caso de uso:	Ver dados do perfil
Atores:	Utilizador autenticado
Descrição:	O utilizador visualiza todos os dados do seu perfil.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na plataforma, o utilizador clica no seu nome no canto superior direito.2. O sistema apresenta ao utilizador o seu menu pessoal com diversas opções.3. O utilizador prime 'Dados da conta'.4. O sistema apresenta uma página com todos os dados do seu perfil.

Tabela 3.12: Caso de uso - Ver dados do perfil

UC 9: Editar dados da conta

ID do caso de uso:	9
Nome do caso de uso:	Editar dados da conta
Atores:	Utilizador autenticado
Descrição:	O utilizador edita os dados da sua conta que pretender.
Pré-condições:	Caso de uso 'Ver dados do perfil'.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na página de visualizar os dados de perfil, o utilizador edita os dados que pretender e prime o botão atualizar.2. O sistema notifica o utilizador do resultado da gravação.

Tabela 3.13: Caso de uso - Editar dados da conta

UC 10: Eliminar utilizador

ID do caso de uso:	10
Nome do caso de uso:	Eliminar utilizador
Atores:	Administrador
Descrição:	O utilizador pode eliminar um utilizador que se encontra associado à sua conta.
Pré-condições:	Caso de uso 'Gerir equipa'
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na página da gestão de equipa o utilizador prime o botão 'Eliminar' na linha correspondente ao utilizador que deseja eliminar.2. O sistema apresenta ao utilizador o resultado da eliminação.

Tabela 3.14: Caso de uso - Eliminar utilizador

UC 11: Analisar conta corrente

ID do caso de uso:	11
Nome do caso de uso:	Analisar conta corrente
Atores:	Administrador
Descrição:	O utilizador pode visualizar todos os movimentos da sua conta, isto é, todos os gastos que foram feitos com uma breve descrição.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na plataforma, o utilizador clica no no seu nome no canto superior direito.2. O sistema apresenta ao utilizador o seu menu pessoal com diversas opções.3. O utilizador prime 'Conta corrente'.4. O sistema apresenta uma página com todos os movimentos realizados na presente conta.

Tabela 3.15: Caso de uso - Analisar conta corrente

UC 17: Ver remetentes

ID do caso de uso:	17
Nome do caso de uso:	Ver remetentes
Atores:	Administrador, Autor e Gestor
Descrição:	O utilizador pode visualizar todos os remetentes da sua conta.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na plataforma o utilizador clica no seu nome no campo superior direito.2. O sistema apresenta ao utilizador o seu menu pessoal com diversas opções.3. O utilizador prime 'Remetentes'.4. O sistema apresenta uma página com todos os remetentes que já foram adicionados à sua conta.

Tabela 3.16: Caso de uso - Ver remetentes

UC 18: Configurar assinatura

ID do caso de uso:	18
Nome do caso de uso:	Configurar assinatura
Atores:	Administrador, Autor e Gestor
Descrição:	O utilizador pode editar a assinatura que será anexada aos <i>emails</i> enviados por esta conta.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na plataforma o utilizador clica no seu nome no campo superior direito.2. O sistema apresenta ao utilizador o seu menu pessoal com diversas opções.3. O utilizador prime 'Configurar assinatura'.4. O sistema apresenta uma página com todos os dados da assinatura e uma pré-visualização da mesma.5. O utilizador edita os campos e prime o botão 'Guardar'.6. O sistema apresenta o resultado da gravação.

Tabela 3.17: Caso de uso - Configurar assinatura

UC 19: Eliminar remetente

ID do caso de uso:	13
Nome do caso de uso:	Eliminar remetente
Atores:	Administrador, Autor e Gestor
Descrição:	O utilizador pode eliminar um remetente da sua conta.
Pré-condições:	Caso de uso 'Ver remetentes'.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na página de visualização de remetentes, o utilizador prime o botão 'Eliminar' correspondente ao remetente que deseja eliminar.2. O sistema mostra o resultado da eliminação.

Tabela 3.18: Caso de uso - Eliminar remetente

UC 21: Gerir equipa

ID do caso de uso:	21
Nome do caso de uso:	Gerir equipa
Atores:	Administrador e Gestor
Descrição:	O utilizador pode visualizar toda a equipa de utilizadores que está associado à sua conta.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na plataforma o utilizador clica no seu nome no campo superior direito.2. O sistema apresenta ao utilizador o seu menu pessoal com diversas opções.3. O utilizador prime 'Gerir Equipa'.4. O sistema apresenta uma página com todos os utilizadores que estão associados à sua conta.

Tabela 3.19: Caso de uso - Gerir equipa

UC 22: Convidar utilizador

ID do caso de uso:	22
Nome do caso de uso:	Convidar utilizador
Atores:	Administrador e Gestor
Descrição:	O utilizador pode convidar um utilizador para se associar à sua conta.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. Na página da gestão de equipa o utilizador insere o <i>email</i> do utilizador que pretende convidar, selecciona a permissão que pretende atribuir ao mesmo e prime o botão 'Enviar convite'.2. O sistema apresenta ao utilizador o resultado do envio.

Tabela 3.20: Caso de uso - convidar utilizador

UC 34: Criar campanha

ID do caso de uso:	34
Nome do caso de uso:	Criar campanha
Atores:	Administrador, Gestor e Autor
Descrição:	O utilizador com a permissão de administrador, gestor ou autor pode criar uma campanha.
Pré-condições:	Não existem.
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. No menu principal, o utilizador seleciona a opção 'Campanhas' e o submenu 'Criar'.2. O sistema apresenta ao utilizador um ecrã com os vários tipos de campanhas que este pode criar.

Tabela 3.21: Caso de uso – criar campanha

UC 38: Criar campanha de *email*

ID do caso de uso:	38
Nome do caso de uso:	Criar campanha de <i>email</i>
Atores:	Administrador, Gestor e Autor
Descrição:	O utilizador com a permissão de administrador, gestor ou autor pode criar uma campanha de <i>email</i> .
Pré-condições:	Caso de uso 34
Pós-condições:	Não existem.
Fluxo normal:	<ol style="list-style-type: none">1. O utilizador prime o botão 'Criar' correspondente à campanha de <i>email</i>.2. O sistema apresenta ao utilizador um ecrã com duas tabs: Escolher template e Importar template.3. O utilizador prime 'Escolher template'.4. O sistema apresenta ao utilizador todos os templates disponíveis.5. O utilizador seleciona um template.6. O sistema redireciona o utilizador para um ecrã que permite construir a <i>newsletter</i>.7. O utilizador configura toda a <i>newsletter</i> e prime o botão enviar.

	<ol style="list-style-type: none"> 8. O Sistema redireciona o utilizador para um ecrã que permite escolher as listas de subscritores para quem deseja enviar as campanhas. 9. O utilizador escolhe as listas e os segmentos de subscritores para quem deseja enviar a campanha e prime o botão 'Enviar'.
Fluxos Alternativos	<ol style="list-style-type: none"> 3. O utilizador prime 'Importar template' 4. O sistema mostra um formulário com indicações para realizar o upload do template. 5. O utilizador faz o upload do template 6. O sistema valida o template e mostra ao utilizador o template inserido.

Tabela 3.22: Caso de uso – criar campanha

3.4 DIAGRAMA DE ATIVIDADE

Devido à importância e complexidade de todo o processo para criar campanha, foi criado um diagrama de atividade que engloba todos os casos de uso necessários para o mesmo. Assim, na Figura 3.5 apresenta-se um diagrama de atividade que descreve o processo de Criar uma campanha de *Email*. Este diagrama inclui os casos de uso, Criar campanha de *email*; Guardar campanha, Testar campanha; Selecionar modo de envio; Selecionar segmentos; selecionar listas e Confirmar envio.

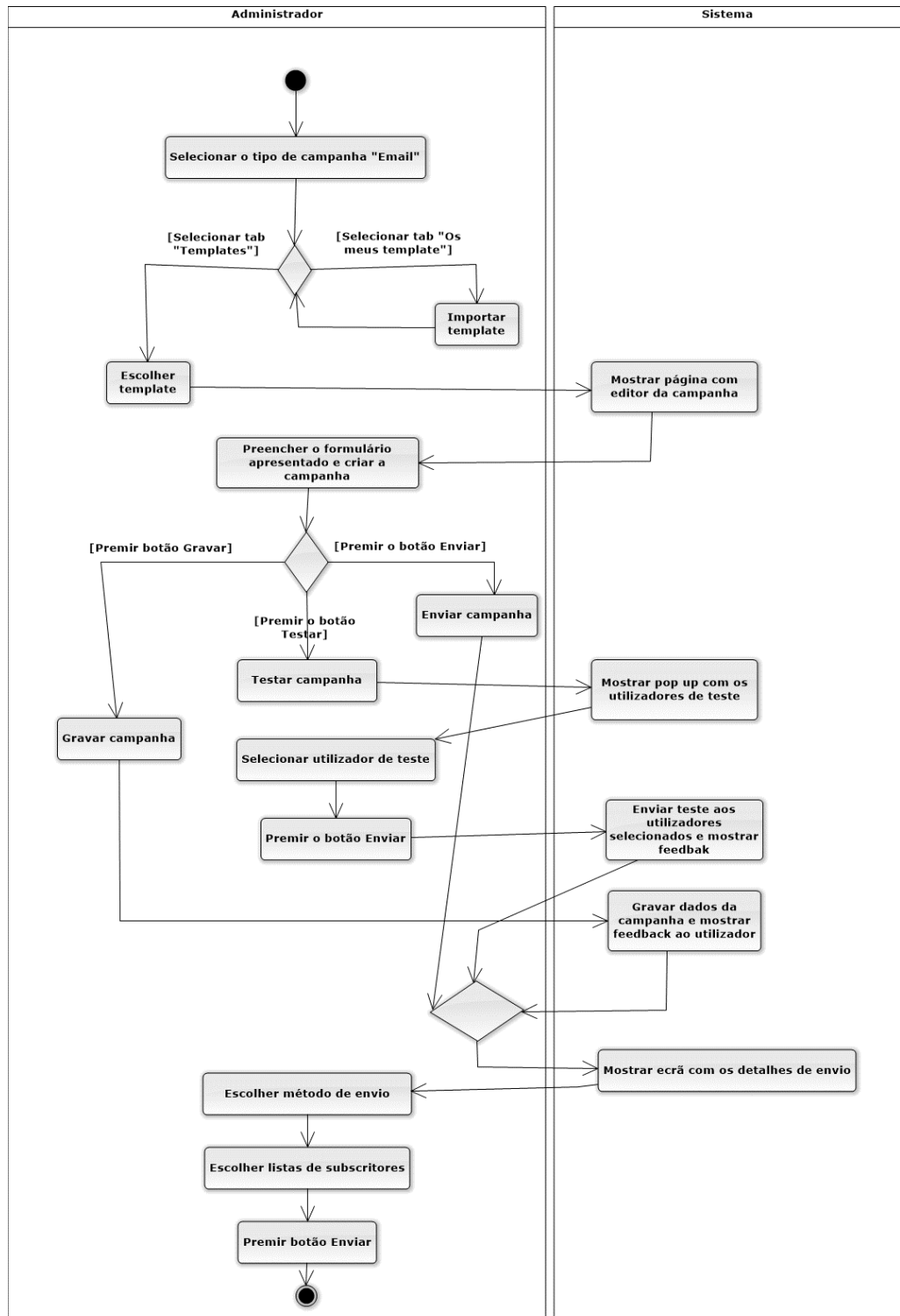


Figura 3.5: Diagrama de atividade do caso de uso Criar campanha

3.5 PROTOTIPAGEM

Para garantir que a plataforma de *marketing* será viável e com uma boa interação com o utilizador, foi necessário um intensivo processo de prototipagem, através de mockups, em conjunto com os potenciais utilizadores da empresa (equipas de desenvolvimento e administrativa).

Desta forma, foi utilizado um *software* de criação de *mockups*³, mais concretamente o *Balsamiq Mockups* [19], através do qual foram estudadas a potencial interface e as principais funcionalidades da aplicação. Foram criadas cerca de seis *wireframes* que representam uma evolução desde um simples esboço até uma simulação do *workflow* da aplicação, evolução essa que se gerou a partir de diversas reuniões com os potenciais clientes já referidos. Após diversas reuniões e alterações de requisitos, chegou-se a um consenso quanto ao *workflow* e interface da aplicação que maximizam a interação com o utilizador.

No Apêndice A – Mockups, estão representadas as *wireframes* referentes aos principais módulos da plataforma de *marketing* IZIPU.

³ Processo de desenho da interface através e papel e lapisou mesmo usando *softwares* apropriados para o efeito, podendo criar uma pré-visualização da interface antes da sua conceção.

4 FERRAMENTAS DE DESENVOLVIMENTO

O PHP é uma linguagem *open source* especialmente utilizada para o desenvolvimento de aplicações web, possuindo uma comunidade muito forte que permite ao programador ter sempre um elevado nível de ajuda quando necessário. Atualmente, o PHP é das linguagens mais famosas do mundo gozando de uma popularidade similar a Java, C++ e C# [20]. Uma das principais vantagens do PHP que levou a *Punchline* a escolhê-la para os seus projetos, é o facto de ser multiplataforma. Ou seja, corre não só em servidores *Windows* mas também em servidores *Linux*.

Como a *Punchline* desenvolve todos os seus projetos em *PHP*, a estagiária focou-se apenas na análise das *frameworks* PHP para o desenvolvimento desta plataforma.

No presente capítulo, para além de ser elaborado um estudo sobre as referidas *frameworks*, é também definida aquela que servirá de apoio ao desenvolvimento da plataforma de *marketing*.

4.1 FRAMEWORKS PHP

Ao nível da análise das ferramentas, o aspeto mais importante é a *framework* a utilizar. Uma *framework* deve ser orientada a objetos e, conseqüentemente, implementar o máximo de padrões de design popularizados por um grupo de autores normalmente conhecido por GoF ou *Gang of Four*. Estes padrões, na sua forma original, mais não são do que a documentação e sistematização de princípios usados, testados e refinados ao longo dos muitos anos de existência da programação orientada a objetos. Os princípios, comumente designados por padrões ou princípios GRASP (do inglês, *General Responsibility Assignment Software Patterns*), são um conjunto de nove orientações para a atribuição de responsabilidades a classes e objetos e não pretendem constituir uma nova maneira de pensar

mas antes de dotar o programador com um conjunto de ferramentas que o auxiliem no *design* de *software* em linguagens orientadas a objetos.

Todos os padrões efetivamente usados respondem a um problema de *software* específico e típico e, na sua generalidade, são transversais à maioria dos projetos, independentemente da sua natureza ou dimensão.

Igualmente importante é o modelo de comunicação entre a *framework* e a camada de persistência dos dados. A maioria das *frameworks* PHP atuais utiliza um conceito recente intitulado *ORM* (*Object Relational Mapping*). Este conceito permite que o programador aceda aos dados armazenados na camada de dados de mais baixo nível, permitindo-lhe abstrair-se dos detalhes da linguagem (tipicamente SQL) que lhe estão normalmente associados.

Numa *framework* sem suporte a *ORM* o programador tem que, de forma manual e contínua, criar e gerir as várias tabelas da base de dados. Se um determinado objeto semântico estiver distribuído por várias tabelas, cabe-lhe a árdua tarefa de organizar as *queries*, muitas vezes complexas, para obter todos os dados.

Com *ORM* o programador cria classes em tudo similares às usadas na programação orientada a objetos. Os dados destas classes são “carregados” através de *queries* escritas automaticamente e que obtém registos provenientes da camada de persistência sem que seja absolutamente necessário haver relação de 1:1 entre as classes criadas e as tabelas usadas. Assim, é possível (e fácil) distribuir uma classe por várias tabelas e, estando o acesso a estes dados abstraído, simplesmente carregar e gravar objetos e ter a segurança de que o sistema os distribui por onde for adequado.

Este modelo de acesso à camada de persistência é mais laborioso na fase inicial do projeto mas tem a vantagem de permitir substanciais poupanças de tempo durante o desenvolvimento, sobretudo quando é necessário modificar os *schemas* das várias tabelas uma vez que, na maioria das *frameworks*, não necessita de reconfiguração.

Em Dezembro de 2013, foi lançado um inquérito com o objetivo de se preverem as *frameworks* mais utilizadas em 2014. Para responder ao inquérito, era um requisito ter experiência em pelo menos duas *frameworks* [21].

Na Figura 4.1 verifica-se que as quatro *frameworks* mais nomeadas pelos programadores e, provavelmente, também as mais usadas, foram o Laravel, Phalcon, Symfony e CodeIgniter.

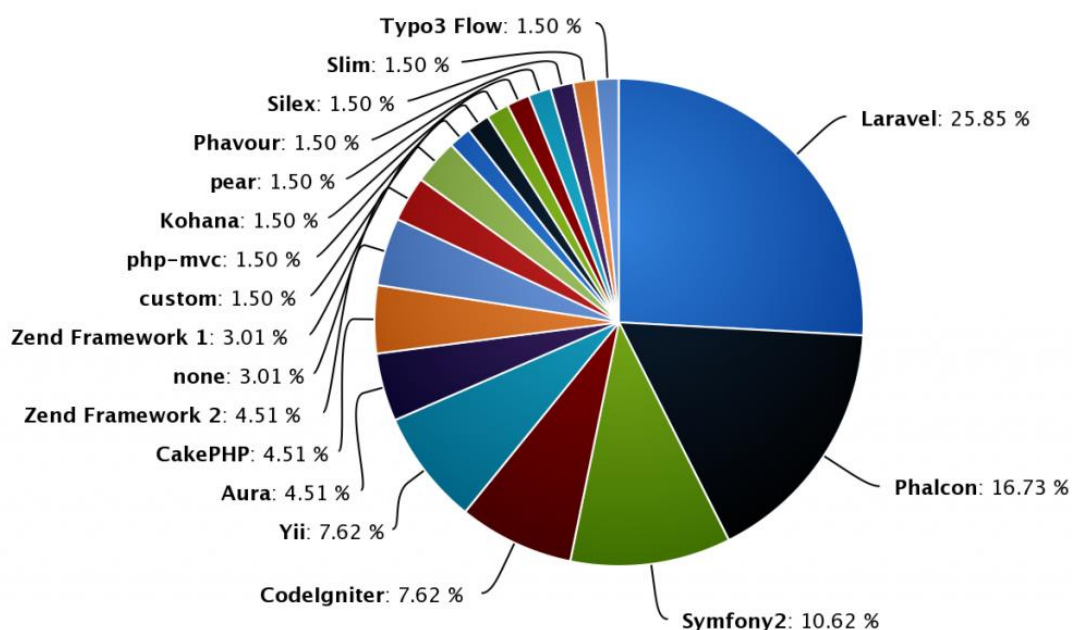


Figura 4.1 – Previsão das *frameworks* mais usadas em 2014.

Devido ao facto de ter uma comunidade grande e bastante ativa, uma documentação bem estruturada e o estagiário ter alguma experiência em trabalhos académicos prévios, o Laravel foi a *framework* escolhida para o desenvolvimento da plataforma de *marketing* IZIPU.

O Laravel é uma *framework open source* para desenvolver aplicações e serviços *web*. Esta faz uso de um instalador de dependências, o *Composer*, que facilita a instalação e atualização de pacotes, um pouco à semelhança do que o *Apache Maven* faz nos projetos desenvolvidos em linguagem Java. Atualmente, com o crescimento das *frameworks*, as bibliotecas vão ficando progressivamente maiores e têm que ser

frequentemente atualizadas. O *Composer* [22] resolve também o problema da desatualização dos pacotes.

Para além do *Composer* a *framework* Laravel disponibiliza ainda uma interface de linha de comandos designada *Artisan* que fornece uma série de comandos úteis para o desenvolvimento de uma aplicação web, tais como a criação de controladores ou a gestão da base de dados. A *framework* permite ainda a criação de novos módulos que implementem uma funcionalidade específica.

O motor de *templates* do Laravel, designado *Blade*, permite reduzir a quantidade de código PHP inserido no meio de HTML, aumentando assim a sua legibilidade e reutilização. Com esta ferramenta, ao invés de se usar a tag “<?php echo \$minhaVariavel; ?>” para colocar um nome no meio de tags de HTML, basta colocar “{!! \$minhaVariavel; !!}” ou “{{ \$minhaVariavel; }}”.

Quanto a validações, o Laravel já tem incluídas imensas e com o texto de erro facilmente editável.

O Laravel conta com uma comunidade oficial designada de Laravel.IO [23]. Sempre que um programador precisar de ajuda em alguma questão, está sempre alguém disponível a ajudar e a dar *feedback* num curto espaço de tempo.

5 ARQUITETURA

Depois da fase de análise em que são definidos os requisitos da aplicação, a concepção do sistema implica a escolha da *framework* arquitetural considerada mais adequada. No caso do presente estágio, seguiu-se o padrão MVC (*Model View Controller*). Com efeito, a *framework* escolhida para o desenvolvimento do IZPU – Laravel - é modular e compreende três camadas: camada de apresentação, camada lógica e camada de dados. Cada uma destas camadas está associada, respetivamente, a *Views*, *Controllers* e *Models* do modelo MVC.

Na Figura 5.1 apresenta-se um diagrama que representa o funcionamento das camadas de apresentação, lógica e de dados da *framework* Laravel.

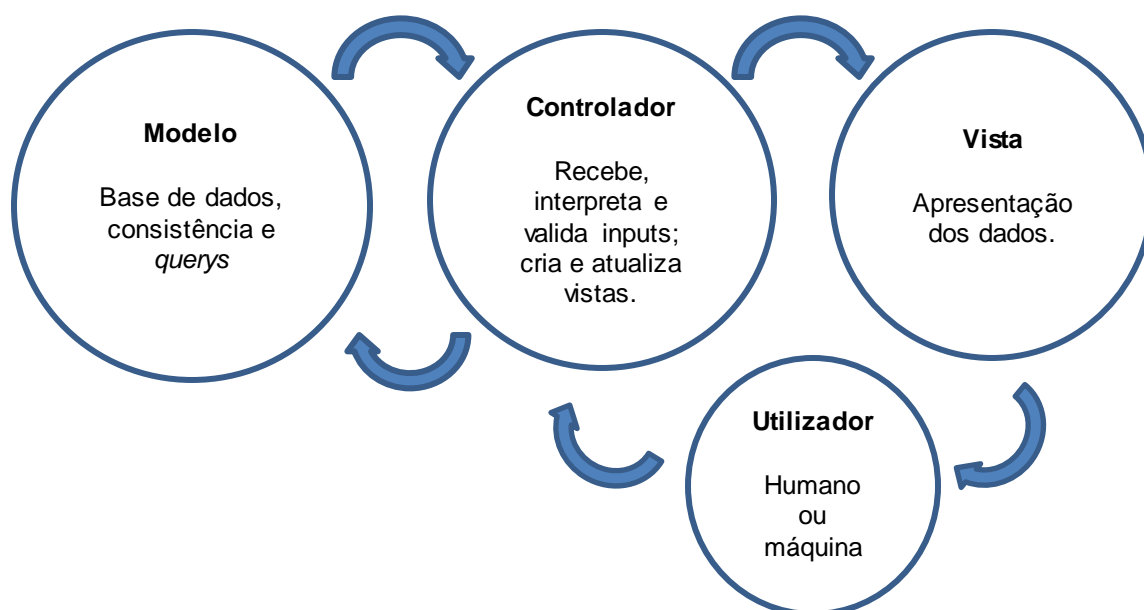


Figura 5.1: Diagrama do modelo MVC

5.1 MODELO DE DADOS

A camada de dados constitui toda a estrutura da base de dados e respetivos *wrappers* de acesso aos mesmos. Nesta camada, são definidos os modelos de dados, representados como classes de PHP que herdam da classe `Illuminate\Database\Eloquent\Model`. Em Laravel as variáveis do modelo são inferidas a partir do *schema* da tabela correspondente na base de dados.

Na *framework Laravel* o *ORM* é designado *Eloquent* e o seu funcionamento é ainda mais simples do que o comumente encontrado em *frameworks* similares. Como é normal acontecer, é necessário criar uma classe que representará o modelo de cada objeto. No entanto, e contrariamente a outros modelos de *ORM* (*Hibernate*, por exemplo), o *Eloquent* tem assunções e extrapola conclusões interessantes acerca da ligação à base de dados.

Em primeiro lugar, o *Eloquent* assume que o nome da tabela é o plural do nome do ficheiro do modelo e que a chave primária se chama 'id'. Se assim não for, é possível especificar nomes diferentes, conforme se pode ver no exemplo seguinte.

```
namespace izipu\Models;
use Illuminate\Database\Eloquent\Model;

class Campanha extends Model {
    protected $table = 'campanha';
    public $primaryKey = 'campanha_id';

    // Estabelece a relação 1:N com a tabela de remetentes
    public function remetente() {
        return $this->belongsTo('izipu\Models\Remetente', 'remetente_id');
    }
}
```

Se a tabela não tiver relações com outras tabelas não é necessário realizar quaisquer configurações adicionais. Contrariamente ao que acontece com *Hibernate*, não é necessário construir *POJOs* completos: o *Eloquent* infere os detalhes do modelo a partir do *schema* da tabela na base de dados.

Como a grande maioria das tabelas das bases de dados está relacionada com outras de alguma forma, é necessário estabelecer relações. No exemplo apresentado foi criada uma relação de 1:N com a tabela de remetentes a que se deu o nome remetente().

O modelo físico da plataforma de *marketing* pode ser consultado no Apêndice B – Modelo físico.

5.2 CAMADA LÓGICA

A camada lógica é responsável por receber o *input* do utilizador, executar as operações lógicas e criar a resposta a ser enviada de volta para o utilizador. Alguns métodos como criar, alterar e apagar objetos do modelo de dados estão presentes nesta camada. Esta camada é considerada o *back-end* da aplicação e é construída depois dos modelos estarem criados.

Estes componentes são invocados aquando da navegação do utilizador de página em página e são os responsáveis por obter os dados, realizar o processamento que se entender necessário e enviá-los para a vista.

O excerto de código abaixo mostra o controlador de campanhas. O primeiro método é o construtor e nele pode ver-se a exigência pela inclusão de um *middleware*, neste caso, o *middleware* responsável pela autenticação. Deste modo garante-se que nenhum utilizador não autenticado pode usar o sistema.

```

class ControladorCampanhas extends Controller {

    public function __construct() {
        $this->middleware('auth');
    }

    public function listarCampanhas() {
        $campanhas = Campanha::all()
            ->with('remetente')
            ->with('campanha_lista.grupo')
            ->with('tipo_envio')
            ->with('tipo_campanha')
            ->with('modo')
            ->with('estado');
        return View::make('campanhas.listar', compact('campanhas'));
    }
}

```

O segundo método, designado *listarCampanhas()* é responsável por obter da base de dados todas as campanhas existentes e de as enviar para a vista. Note-se que as campanhas são carregadas com vários modelos auxiliares, criados na classe do modelo campanhas e representados pela instrução *->with('verbo')*. Por fim o método invoca uma vista, neste caso a vista *listar* da diretoria *campanhas* e envia-lhe os dados obtidos.

5.3 CAMADA DE APRESENTAÇÃO

A camada de apresentação é responsável pela apresentação final, a qual utiliza os dados enviados pela camada lógica e renderiza-os em *templates HTML* pré-definidos para fazerem a exposição dos dados ao utilizador, correspondendo ao *front-end* da aplicação.

A vista é construída com quaisquer tecnologias de apresentação que se pretenda mas tipicamente usa-se *html*, *css* e *javascript*. No exemplo acima a variável *campanhas* é enviada para a vista devidamente preenchida. O acesso a esta variável pode ser feito de várias formas mas a mais confortável é usando o motor de *templates Blade*. O *Blade* permite incluir construções que se assemelham à lógica para aceder às variáveis que lhe forem enviadas. O modo mais simples de acesso é usando os *tokens* `{{ e }}`. Se a variável contiver *tags* de *html* é necessário fazer o seu escape e deve usar-se o *token* `{!! e !!}`. Para além do acesso a variáveis individuais é ainda possível aceder a coleções e avaliar condições. O exemplo abaixo mostra a vista que lista as campanhas.

```

@extends('layouts.master')

@section('css')
@stop

@section('headerPage')
<h1>
    CAMPANHAS
</h1>
<ol class="breadcrumb">
    <li><a href="#"><i class="fa fa-dashboard"></i> Home</a></li>
    <li class="active">Campanhas</li>
</ol>
@stop

@section('conteudo')

<div class="container-fluid campanhas">
    <div class="row">
        <div class="col-sx-12">
            <table id="tabelaCampanhas" class="table table-striped table-bordered
                table-hover">

                <thead>
                    <tr>
                        <th>Nome</th>
                        <th>Remetente</th>
                        <th>Grupos</th>
                        <th>Tipo</th>
                        <th>Modo</th>
                        <th>Estado</th>
                        <th>Editar</th>
                        <th>Duplicar</th>
                        <th>Ver</th>
                        <th>Cancelar envio</th>
                    </tr>
                </thead>
                <tbody>
                    @if(!is_null($campanhas))
                    @foreach($campanhas as $key => $value)
                    <tr>
                        <td>{!! $value->campanha_designacao !!}</td>
                        <td>{!! $value->remetente->remetente_designacao!!}</td>
                        <td>
                            @foreach ($value->campanha_lista as $lista)
                                $lista->grupo->grupo_designacao<br/>
                            @endforeach
                        </td>
                        <td>{!! $value->tipo_campanha->
                            tipo_campanha_designacao !!} </td>
                        <td>{!! $value->tipo_envio->tipo_envio_designacao !!}</td>
                        <td>{!! $value->estado->estado_designacao !!}</td>

                        <td>
                            <a href="{!! url('campanha/editar',
                                $value->campanha_id) !!}">
                                <i class="fa fa-pencil-square-o"></i></a>
                            </td>

                        <td>
                            <a href="{!! url('campanha/duplicar',
                                $value->campanha_id) !!}">
                                <i class="fa fa-files-o"></i></a>
                            </td>

                        <td>
                            <a href="{!! url('campanha/ver',
                                $value->campanha_id) !!}">
                                <i class="fa fa-eye"></i></a>
                            </td>

                        <td>
                            <a href="{!! url('campanha/cancelarenvio',

```

```

                                $value->campanha_id) !!}">
                                <i class="fa fa-power-off"></i></a>
                                </td>
                                </tr>
                                @endforeach
                                @endif
                                </tbody>
                                </table>
                                </div>
                                </div>
                                </div>
                                @stop

                                @section('scripts')
                                <script src="{{ URL::asset('js/campanhas.js')}}" type="text/javascript"></script>
                                @stop

```

As vistas iniciam-se normalmente pela inclusão da *master page*, conseguida com a instrução *@extends*. Em seguida incluem as instruções ou até ficheiros de *css* que se entendam necessários. No final é normal incluir instruções ou ficheiros de *javascript*.

A parte mais interessante da vista está, porém, no acesso aos dados, inicialmente com a instrução *@if (!is_null(\$campanhas))*, para garantir que a variável não é nula e, depois, com o percorrer de uma coleção com a instrução *@foreach(\$campanhas as \$key => \$value)*. Apesar destas instruções serem similares às suas homónimas da lógica não há quebra do padrão MVC por não ser possível aceder a outros dados que não aqueles que o controlador, atempadamente, enviou.

6 IMPLEMENTAÇÃO

Tal como anteriormente referido, de um modo geral, a aplicação a implementar consiste numa plataforma de *marketing* de funcionalidades simples que irá conter diversos módulos, nomeadamente:

- Campanhas
 - de *Email*;
 - de fax;
 - de SMS;
 - de MMS;
 - de Voz.
- Utilizadores
- Subscritores
- Autenticação
- Conta corrente
- Remetentes
- Equipa
- Relatórios

Nesta secção, antes dos detalhes de desenvolvimento de cada um destes módulos, serão abordados a ferramenta utilizada para o controlo de versões e a metodologia de trabalho. Numa segunda instância, será então detalhado cada módulo desenvolvido e as várias opções tomadas. Por fim, serão descritas todas as dependências utilizadas ao longo do desenvolvimento da plataforma.

6.1 CONTROLO DE VERSÕES

Para o desenvolvimento da plataforma de *marketing* *IZIPU* será usada uma ferramenta de controlo de versões. Mesmo que este projeto não seja desenvolvido de forma colaborativa é, mesmo assim, importante utilizar

uma ferramenta que permita aceder a versões prévias do código da aplicação. A adoção desta ferramenta aumenta a segurança e evita catástrofes, no caso de uma perda do código-fonte, ou mesmo anular alguma atualização defeituosa que poderia demorar tempo a reverter.

A ferramenta escolhida foi o SVN dada a experiência de utilização do estagiário. O SVN é uma ferramenta de controlo de versões que permite, além do desenvolvimento colaborativo a partir de um repositório único, *merge* de conteúdo e armazenamento de *logs* [24].

6.2 METODOLOGIA DE TRABALHO

A *Punchline* utiliza uma metodologia própria, a que chama de adaptação de *Scrum*, para a organização do trabalho dos seus colaboradores. Esta metodologia utilizada pela empresa resume-se a *milestones* (metas) com objetivos definidos mas que podem sofrer pequenas alterações ao longo dos períodos. Existem avaliações quinzenais (por vezes podem ser semanais) onde se discute o que foi implementado, o que falta implementar e as dificuldades que o colaborador está a ter nessa *milestone*.

Para garantir uma boa visibilidade do que está a acontecer ao longo do desenvolvimento da plataforma de *marketing*, para priorizar e distribuir tarefas e para transmitir notificações a toda a equipa, a equipa da *Punchline* utiliza uma ferramenta designada de *Trello* [25].

O *Trello* baseia-se no sistema *Kanban*⁴, bastante utilizado no desenvolvimento com *scrum*. Atente-se Figura 6.1 uma ilustração do *Trello* usado na fase de desenvolvimento da plataforma de *marketing*.

⁴ Palavra de origem japonesa que significa “cartão ou registos visíveis”. Apesar da sua origem estar no interior de uma linha de produção, o *Kanban* pode também ser usado como uma ferramenta de gestão de tarefas [46]. Assim como o modelo de *scrum*, o *kanban* é um ótimo método para organizar as tarefas de uma equipa.

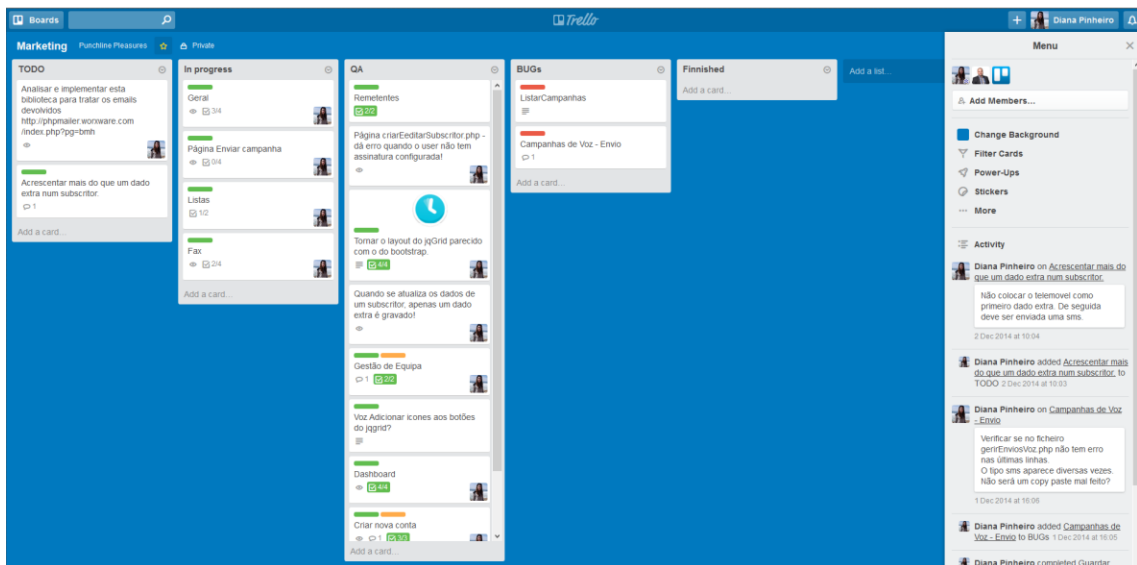


Figura 6.1: Board da plataforma de marketing IZIPU usada na fase de desenvolvimento.

Verifique na imagem que ele é constituído por quadros (*boards*), listas e cartões. Um *board* é o projeto em desenvolvimento podendo assim existir diversos *boards*. Dentro de cada *board* temos as listas que, de uma forma mais simples, são colunas que permitem fazer uma melhor gestão/organização das tarefas. No caso do projeto da plataforma de *marketing* temos as colunas *Todo* onde se colocam os cartões referentes às tarefas que faltam implementar; *In progress* onde estão presentes os cartões com as tarefas em desenvolvimento; *QA* – *Quality Assurance* onde se colocam melhorias a ser implementadas bem como a correção de alguns erros encontrados; *BUGs* onde se colocam todos os erros encontrados na plataforma; e *Finished*, coluna para onde são arrastados os cartões com as tarefas finalizadas.

Deste modo, consegue-se uma melhor comunicação entre a equipa e uma gestão mais simples das tarefas em desenvolvimento.

6.3 MÓDULOS DESENVOLVIDOS

Nesta secção são abordados todos os módulos desenvolvidos e são detalhados os aspetos mais importantes da implementação de cada um.

6.3.1 Campanhas

Uma campanha representa uma forma de promoção enviada para um ou vários subscritores, a qual pode ser efetuada via *email*, SMS, MMS, voz ou fax.

Apenas os utilizadores com permissão de administrador, autor ou gestor podem criar campanhas de *marketing*. De seguida são apresentados os detalhes de implementação de cada campanha. O envio das mesmas será detalhado na secção 6.3.2.

Campanhas de *email*

O *email* é um dos meios pelo qual o utilizador poderá enviar uma campanha desde que tenha a permissão adequada.

A plataforma de *marketing* disponibilizará diversos *templates* de *newsletter* já configurados e prontos a serem editados. No entanto, caso assim o deseje, o utilizador poderá importar o seu *template* ou criar a sua *newsletter* de raiz no editor disponibilizado. Para submeter o seu próprio *template*, o utilizador deve inserir um único ficheiro comprimido (extensão .zip) que deve conter um ficheiro *HTML*, uma pasta com o nome de *images* (no seu interior devem estar todas as imagens usadas no ficheiro *HTML*) e uma imagem de apresentação do *template* que deve ter o nome de *template.png* (este ficheiro é opcional). Na Figura 6.2 verifica-se o layout da página onde o utilizador pode inserir o seu *template*.

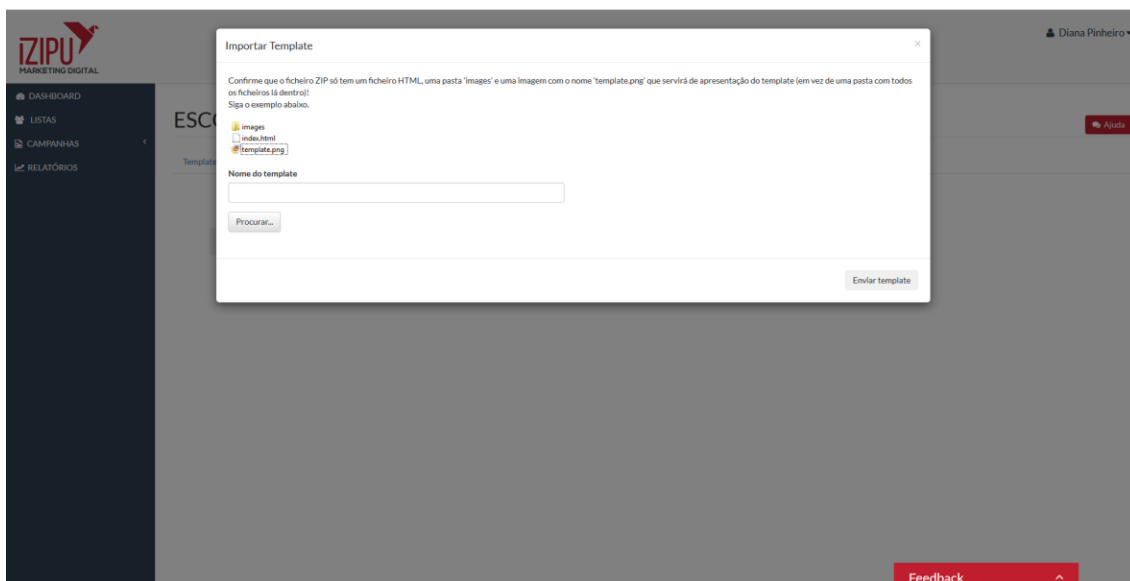


Figura 6.2: Ecrã que permite ao utilizador importar o seu template

Do ponto de vista da programação, nomeadamente da camada lógica, é criada uma pasta de modelos para cada *id* de utilizador existente. Quando o utilizador submete o ficheiro comprimido, a camada da lógica verifica se o ficheiro recebido pela plataforma tem realmente a extensão *.zip*, valida se existe no seu interior um ficheiro *HTML* e uma pasta *images*. Caso não exista esta estrutura, o utilizador é notificado do erro. Se se verificar que a estrutura do ficheiro recebido está correta, é gerada uma *hash* para o *template* e este é adicionado na base de dados

A plataforma disponibiliza ao utilizador um editor com todos os recursos necessários para que este edite o seu conteúdo a seu gosto. Na Figura 6.3 verifica-se o editor implementado é bastante idêntico aos editores comuns. Para implementar este editor foi usada a API *TinyMCE* que será abordada no capítulo das dependências.

Durante a edição da *newsletter*, a qualquer momento o utilizador pode enviar até um máximo de cinco testes.

Internamente, a camada lógica, codifica o conteúdo da *newsletter* (*HTML*) através do algoritmo *base64*⁵, guarda na base de dados um rascunho da campanha e envia o *email* de teste com a *newsletter* usando a biblioteca *PHPMailer*

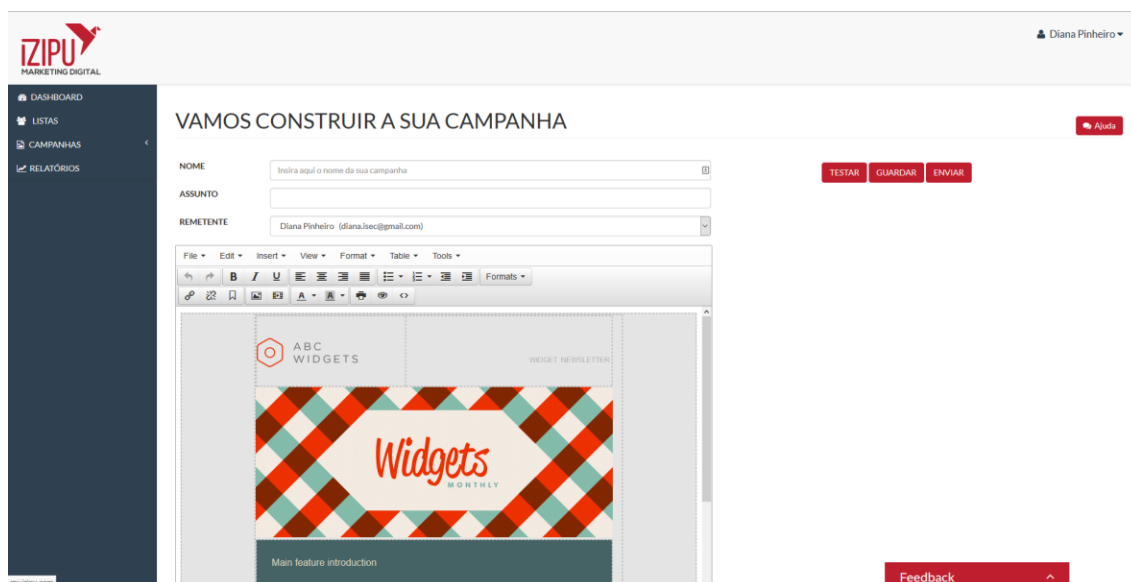


Figura 6.3: Ecrã onde o utilizador pode criar ou editar o conteúdo de uma *newsletter*

Campanhas de SMS

Produzir campanhas através de mensagens de texto também é possível na plataforma de *marketing* desenvolvida. Para criar a SMS, o utilizador deverá preencher um campo denominado por 'Mensagem'. Verifique na Figura 6.4 uma captura de ecrã da página que permite ao utilizador da plataforma criar uma mensagem de texto.

⁵ Método para codificação de dados constituído por 64 caracteres ([A-Z],[a-z],[0-9], "/" e "+"). [42]

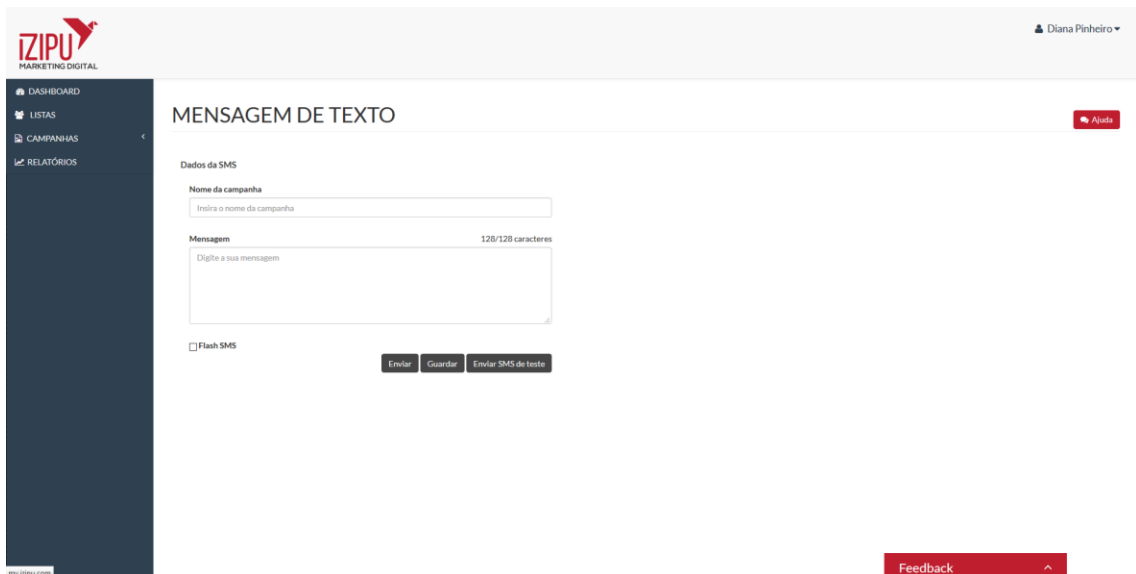


Figura 6.4: Ecrã que permite ai utilizador criar uma mensagem de texto

À semelhança da campanha de *email*, o utilizador também poderá enviar mensagens de teste que, do ponto de vista da programação, terá um procedimento idêntico ou seja, é codificada o conteúdo da mensagem através de *base64*, guardado um rascunho na base de dados e enviada a mensagem.

O utilizador pode enviar dois tipos de mensagem: a normal e a *Flash SMS*. A *flash SMS* [26] é uma mensagem que é automaticamente aberta pelo dispositivo móvel do destinatário como se fosse um “pop-up”.

Do ponto de vista da programação, a diferenciação entre os dois tipos de mensagem é feita através da alteração de um parâmetro do URL usado pela API.

Considere-se o seguinte excerto de código onde é feita essa distinção:

```

if ($smsFlash === true) { // SMS do tipo FLASH
    $url="$baseurl/http/sendmsg?session_id=$sess_id&to=$to&text=$text&from=$from&msg_type=SMS_FLASH";
} else {
    $url="$baseurl/http/sendmsg?session_id=$sess_id&to=$to&text=$text&from=$from";
}

```

Figura 6.5: Excerto de código referente ao envio de SMS e de *Flash SMS*

Campanhas de MMS

As campanhas de MMS são muito idênticas às campanhas de SMS à exceção do facto de poderem ter média. Na Figura 6.6 verifica-se que o ecrã de criação de mensagem de multimédia é muito idêntico ao ecrã de criação de uma mensagem de texto. Difere apenas o botão “Anexar” presente na caixa de texto para permitir ao utilizador inserir anexos de multimédia.

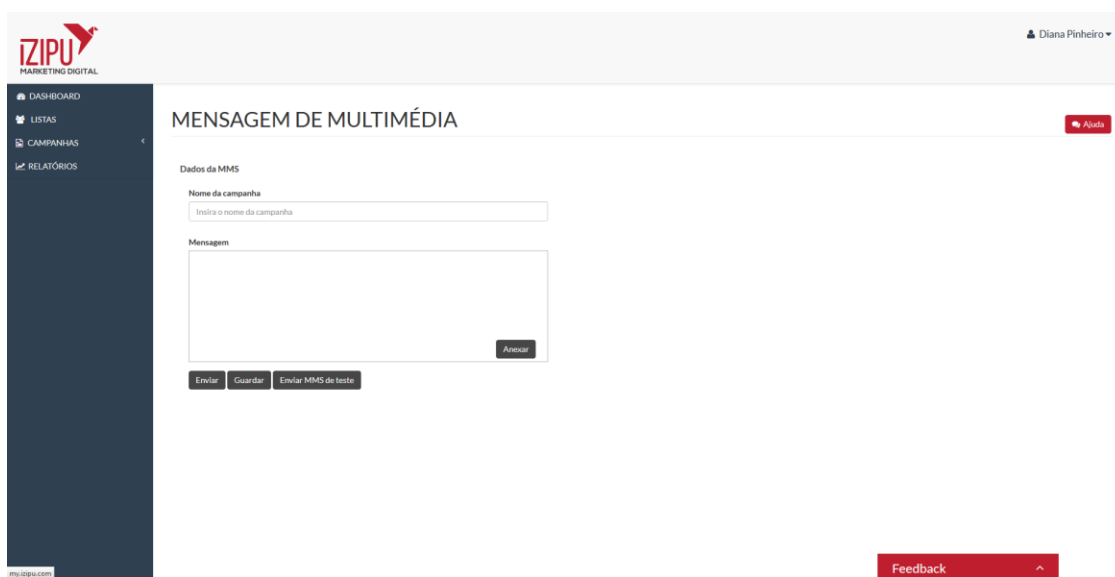


Figura 6.6: Ecrã que permite ao utilizador criar uma mensagem de multimédia

Com efeito, neste tipo de campanha, para além do texto, o utilizador pode anexar ainda um som, uma imagem ou um vídeo. Para anexar os ficheiros foi utilizada a API *Responsive Filemanager* que será abordada no capítulo das dependências. Verifique na Figura 6.7 que esta API permite criar um *uploader* de ficheiro bastante intuitivo. Tem a vantagem de também permitir ao utilizador realizar o *upload* de ficheiros arrastando-os para cima da janela.

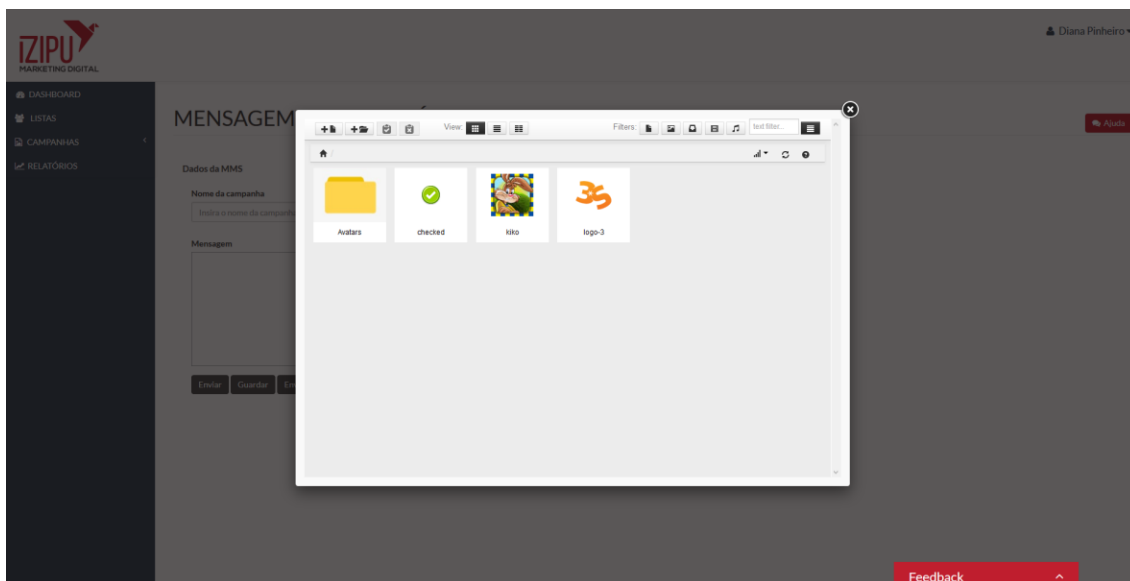


Figura 6.7: Captura de ecrã onde se verifica o upload de ficheiros

Tanto para guardar um rascunho da campanha de MMS, como para enviar a campanha, os anexos inseridos pelo utilizador têm que ser serializados. A serialização consiste em transformar um objeto ou um elemento de algum tipo de dados complexo (array, hash, struct, etc.) numa string que o represente. Através dessa mesma string, será também possível recriar o objeto inicial.

Para o envio das MMS foi usada a mesma API das campanhas de SMS, *Clickatell*.

Campanhas de voz

Um dos requisitos para a plataforma de marketing a desenvolver era existir um módulo IVR (*Interactive Voice Response*). O IVR [27] é um sistema capaz de responder ou interagir com o interlocutor através de áudio pré-gravado ou gerado dinamicamente instruindo o interlocutor sobre as ações a tomar. Este sistema está dotado de reconhecimento de voz para possibilitar o reconhecimento de mensagens vocais.

Na plataforma de marketing IZIPU, este módulo foi um dos mais interessantes e desafiantes de desenvolver pela complexidade, tanto ao

nível da camada lógica como ao nível da camada de apresentação que requer bastante jQuery.

O principal requisito funcional da plataforma de *marketing* era que fosse intuitiva e, assim sendo, todos os interfaces gráficos desenvolvidos seguiram esse princípio. O módulo do IVR, por ser um dos mais complexos, exigiu uma maior preocupação relativamente a esse aspeto.

A Figura 6.8 mostra o ecrã onde o utilizador pode construir a campanha de voz e criar toda a comunicação entre a máquina e o interlocutor.

CAMPANHA DE VOZ

Dados da mensagem de voz

Nome da campanha

Insira o nome da campanha

Mensagem de voz

Arraste para aqui os conteúdos

Personalize a sua mensagem

Texto Som Gravação

Evento teclas Redirecionar chamada

Enviar Guardar

Figura 6.8: Ecrã que permite ao utilizador criar toda a comunicação entre a máquina e o interlocutor.

Para criar a campanha, o utilizador apenas tem que ir arrastando os elementos que constam do lado direito para a caixa onde diz “Arraste para aqui os seus conteúdos” e configurar o que pretende de cada elemento. Em função do que o utilizador arrasta, é acrescentada à caixa de mensagem de voz um novo componente. Na Figura 6.9 verifica-se uma captura de ecrã que mostra a construção de uma campanha de voz. O utilizador arrastou um componente de texto e um de “Evento de teclas” para a caixa “Mensagem de voz”. Automaticamente foram logo criadas uma caixa de texto e uma caixa onde são realizados os eventos de teclas. À medida que o utilizador clica nas teclas que pretende que tenham eventos, vão sendo acrescentadas caixas para a criação do respetivo evento.

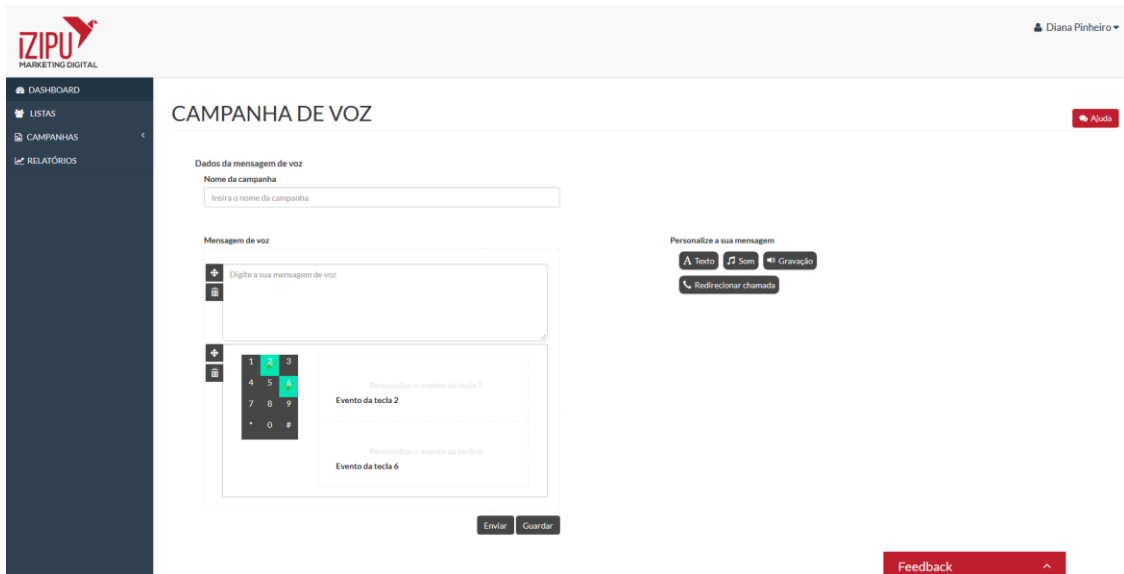


Figura 6.9: Captura de ecrã que mostra a criação de uma campanha de voz

Ao nível da lógica, para criar este módulo foi utilizada a API *Twiml* que é abordada no capítulo das dependências.

Para uma melhor perceção do funcionamento interno deste módulo foi desenhado o diagrama abaixo:

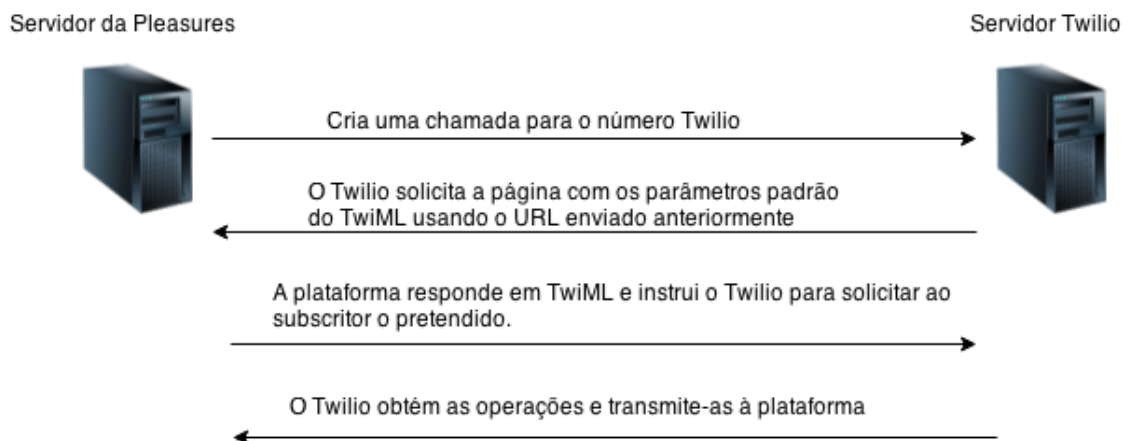


Figura 6.10: Esquema que representa o funcionamento do módulo de voz.

Depois de serem recolhidos todos os dados da construção do IVR pelo utilizador, estes são guardados em formato JSON⁶.

Para iniciar o envio da campanha de voz (IVR), é iniciado o acesso à API *Twilio* com o ID e o *Token* da conta criada previamente e posteriormente é criada uma chamada à API *TwML* com os seguintes parâmetros: número *Twilio* comprado previamente, URL que será invocado quando o subscritor atender a chamada e um array que contém o URL que é invocado em caso de erro durante a chamada e o URL que é invocado quando a chamada é terminada.

Na Figura 6.11 verifica-se o excerto de código que representa o envio da chamada conforme explicado anteriormente.

```
try {
    // Initiate a new outbound call
    $call = $client->account->calls->create(
        $phonenumber, // The number of the phone initiating the call
        $numeroSubscritor, // The number of the phone receiving call

        PUBLIC_ROOT.'envio/realizaChamada/conteudoChamada.php/?m='.$estruturaCampanha
        .'&esid='.$envioSubscritorID, // The URL Twilio will request when the call is
        answered
        array(
            "FallbackUrl" => PUBLIC_ROOT.'envio/realizaChamada/errosChamada.php',
            "StatusCallback" =>
            PUBLIC_ROOT.'envio/realizaChamada/detalhesChamada.php/?esid='.$envioSubscri
            torID,
            "Method" => "GET"
        )
    );
    $resposta = array('sucesso' => true, 'sidChamada' => $call->sid);
} catch (Exception $e) {
    echo 'Error: ' . $e->getMessage();
    $resposta = array('sucesso' => false);
}
```

Figura 6.11: Excerto do código que permite fazer a chamada de voz através da API Twillio

Quando o subscritor atende a chamada, o *Twilio* pede à plataforma online através do URL recebido no pedido anterior a estrutura do IVR. Esta tem que seguir a estrutura *TwML* para que o *Twilio* perceba o que tem que fazer: gravar a chamada, reproduzir uma mensagem para o gravador ou solicitar o subscritor para pressionar dígitos no seu teclado.

⁶ Json é um formato leve de troca de informações/dados entre sistemas. O JSON tem a vantagem de permitir uma maior no transporte dos dados.

Para construir dinamicamente a estrutura em *Twiml* são verificados todos os passos que o utilizador construiu no editor da campanha:

- Caso o utilizador pretenda que o interlocutor oiça um texto digitalizado por si no editor, do ponto de vista da lógica, é acrescentado à estrutura *Twiml* as seguintes *tags*:

```
<Say voice="woman" language="pt-PT">  
// Texto escrito no editor pelo utilizador  
</Say>
```

- Caso o utilizador pretenda que seja transmitindo um som/gravação ao subscritor é acrescentado à estrutura *Twiml* as seguintes *tags*:

```
<Play> //Url do ficheiro som </Play>
```

- Caso o utilizador pretende que o subscritor interaja através do teclado do seu dispositivo e que sejam tomadas ações em função das respetivas teclas pressionadas são acrescentadas as seguintes *tags* à estrutura *Twiml*:

```
<Gather numDigits="1" action="URL que trata dos eventos das  
teclas" method="POST"> </Gather>
```

O URL que consta no código acima é invocado sempre que o subscritor premir uma tecla e é usado para acrescentar à estrutura *Twiml* as *tags* que correspondem ao evento de cada tecla premida pelo subscritor para que o Twilio perceba o que tem que transmitir ao interlocutor.

Imagine-se que ao construir a campanha, o utilizador transmitiu que se o subscritor premir a tecla 1 tem que ouvir um determinado texto.

A plataforma invoca o URL presente no código acima e verifica qual a tecla premida e que *tags* a acrescentar:

```
If ($_REQUEST['Digits'] == 1){  
<Say voice="woman" language="pt-PT">  
// Texto escrito no editor pelo utilizador  
</Say>  
}
```

- Caso o utilizador pretenda que o *Twilio* grave uma mensagem de voz do subscritor acrescenta à estrutura *Twiml* as seguintes *tags*:

```
<Record maxLength="30" transcribe="true"  
transcribeCallback="Url para tradução" action="URL  
para reproduzir gravação"/>
```

Sempre que o interlocutor executa uma gravação, o *Twilio*, para além de gravar, tenta converter a gravação em texto corrido para que fique guardado em modo texto. No entanto, o *Twilio* apenas tem a capacidade de converter uma gravação em texto caso o idioma falado pelo interlocutor seja em inglês, o que torna a plataforma de *marketing* desenvolvida limitada neste aspeto.

6.3.2 Envio das campanhas

Numa primeira fase de envio todas as campanhas executam mesmo passo. Para qualquer tipo de campanha, seja ela de *email*, voz, SMS, MMS ou fax, o utilizador tem que escolher o tipo de envio que pretende e para quem deseja enviar a campanha Na Figura 6.12 pode ver-se um exemplo deste procedimento.

The screenshot shows the 'Campanha de teste' interface. On the left, under 'TIPO DE ENVIO', there are three radio buttons: 'Enviar já' (selected), 'Agendar', and 'Aniversario'. Below this, under 'PARA', there is a checked checkbox for 'Grupo P (4 subscritores)'. Underneath, there are two radio buttons: 'Enviar para toda a lista' and 'Enviar para um segmento' (selected). A dropdown menu labeled 'Escolha um segmento' shows 'Punchline'. Below that, there are three unchecked checkboxes for 'Pleasures (2 subscritores)', 'teste (0 subscritores)', and 'Profiforma (1 subscritores)'. At the bottom left is a 'Voltar' button. On the right, a grey box titled 'RESUMO DA CAMPANHA' contains the following information: 'CUSTO PREVISTO 3', 'NºSUBSCRITORES 3', 'ASSUNTO Campanha de teste', and 'REMETENTE Diana Pinheiro'. At the bottom of this box is an 'ENVIAR' button.

Figura 6.12: Ecrã da plataforma de *marketing* IZIPU onde o utilizador escolhe o método de envio e para listas de subscritores deseja enviar a campanha.

Internamente, o que a plataforma faz é guardar numa tabela de campanhas a enviar a presente campanha e os subscritores para quem esta será enviada. Mesmo que o utilizador selecione a opção ‘*Enviar já*’ a campanha nunca será enviada no preciso momento pois é sempre necessário que o *poller* de envio corra novamente.

Os *pollers* de envio das campanhas foram sem dúvida a parte mais desafiante e mais complexa de desenvolver visto que existiam inúmeras regras que necessitavam de atenção, tais como, o tempo que cada *poller* pode correr, o saldo da conta, o numero de campanhas de *email* que podem ser enviadas por minutos/hora/dia e os envios propriamente ditos onde é necessário um estudo de cada API.

***Poller* de envio das campanhas**

Para enviar as campanhas foram criados quatro *pollers* de envio, um por cada tipo de campanha sendo que as SMS e as MMS estão no mesmo *poller*, que são corridos de cinco em cinco minutos por um *Cron Job*. Um *Cron Job* é um *script bash* comum invocado pela ferramenta *Cron* num determinado momento configurável.

Em termos de algoritmo, todos os *pollers* seguem a mesma estrutura adaptando, no entanto, a codificação ao tipo de campanha. Assim, todos os *pollers* começam por analisar se existem outros *pollers* a correr no sistema,

ou seja, verificam se há outras instâncias de um *poller* do mesmo tipo que ainda esteja ocupado a enviar campanhas e que, por esse motivo, ainda não tenha terminada a sua execução. Caso, algum dos *pollers* ainda esteja a correr há mais de duas horas é considerado 'cancelado' pelo novo *poller*. Da perspetiva do utilizador final, o facto de um *poller* ter sido dado como 'cancelado' não tem qualquer importância pois as eventuais campanhas que tenham ficado por enviar, serão retomadas automaticamente da próxima vez que um *poller* desse tipo for corrido.

Sempre que é iniciado um *Cron Job* todos os dados *Cron Job* atual são registados na base de dados (na tabela de *Crons Job*) nomeadamente a data de início, o estado do *Cron Job* (em desenvolvimento, resolvido e cancelado) e o tipo de *poller* (*email*, *sms*, *voz* ou *fax*).

Posteriormente, o *poller* de envio verifica na base de dados (na tabela de campanhas agendadas) quais as campanhas que possuem um *timestamp* superior ao atual e que tenham um estado 'Por publicar'. O estado destas campanhas é alterado para 'Processar envio' e são analisados os subscritores associados ao envio desta campanha. Antes de avançar para o envio das campanhas propriamente dito, é verificado se ainda existe tempo para continuar o envio das mesmas. A equipa de desenvolvimento decidiu que cada *poller* apenas pode ser executado durante trinta minutos. Caso o *poller* pare a meio de um envio, este será retomado mais tarde não tendo assim qualquer consequência para o utilizador final.

A fase seguinte do algoritmo dos *pollers* de envio consiste em enviar efetivamente as campanhas e a codificação desta fase é, obviamente, diferente para cada tipo de campanha pois são usadas API's e processos diferentes em cada um deles.

Envio de campanhas de *email*

Antes de enviar a campanha, o *poller* de envio substitui todos os *links* da campanha de forma a ser possível, mais tarde, obter estatísticas relevantes, nomeadamente: quem abriu os *links*; a que horas foi acedido

cada *link*; quais os *links* com mais cliques, etc.. A substituição de cada *link* da campanha implica a adição de uma *hash* que permitirá à plataforma identificar o subscritor que fez o clique.

Para além da substituição de todos os *links* da campanha, é ainda acrescentada uma imagem de 1x1 *pixel* e em cuja *source* está um URL com o *ID* da campanha e o *ID* do subscritor. Desta forma, quando o utilizador abre a campanha a plataforma de *marketing* é notificada enriquecendo-se assim a informação estatística disponível.

O passo seguinte do algoritmo assegura o respeito pelas regras de envio de *marketing* e, nesta fase, o *poller* verifica quantos *emails* já enviou no último minuto, na última hora e no último dia, de modo a garantir que não ultrapassa os limites permitidos por cada servidor de *email* (*Gmail*, *Hotmail*, etc). Se esta limitação for ultrapassada os *emails* podem ser classificados como SPAM e toda a campanha ficará comprometida.

Finalmente – e caso exista saldo na conta do utilizador que pretende enviar a campanha – procede-se ao seu envio usando *PHP Mailer*. O *PHP Mailer* é uma classe que fornece uma API para o envio de *emails* via conexão SMTP [28]. O código fonte desta classe é *open source* e pode ser obtido no *GitHub*.

Abaixo apresenta-se o código do método responsável pelo envio de *emails* da plataforma de *marketing*.

```

public static function enviarMail($assunto, $mensagem, $email, $nome, $emailRemetente,
$nomeRemetente)
{
    $mail = new PHPMailer(true); //true retorna excecoes
    $envio = true;
    try
    {
        $mail->IsSMTP(); // Utilização de SMTP
        $mail->Host = config_plataforma::smtp_host; // Servidor SMTP
        $mail->SMTPAuth = true; // Activar autenticação SMTP
        $mail->Username = config_plataforma::smtp_user; // User do servidor SMTP
        $mail->Password = config_plataforma::smtp_pass;
        $mail->SetFrom($emailRemetente, $nomeRemetente); // Nome e Email do Remetente
        $mail->AddAddress($email, $nome);
        $mail->Subject = $assunto; //assunto do e-mail
        $mail->IsHTML(true); // false - O conteúdo da mensagem sera enviado como texto e nao
        HTML
        $mail->Body = '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html;" />
</head>
<body>' . "$mensagem" . '</body>
</html>'; // Conteudo da mensagem

        $mail->CharSet = 'utf-8'; //codificacao da mensagem

        if ($mail->Send())
            return 1;
    } catch (phpmailerException $e) {
        echo $e->errorMessage(); // Erros provenientes do PHPMailer
        error_log("PHP Mailer error >> " . $e->errorMessage());
        return 0;
    } catch (Exception $e) {
        echo $e->getMessage(); // Outros erros
        error_log("ERRO >> " . $e->getMessage());
    }
    error_log("O email não foi enviado");
    return 0;
}

```

Envio de campanhas de SMS/MMS

Antes de enviar a campanha para os respetivos subscritores, o *poller* verifica qual o tipo da campanha a enviar (SMS ou MMS). Caso seja SMS, o *poller* verifica na base de dados se a campanha é *flash* SMS ou SMS normal e invoca o método de envio da SMS adequado, fornecendo como parâmetros o número do subscritor, o texto da mensagem e uma *flag* que indica se se trata de uma *flash* SMS.

Consoante se trate ou não de uma *flash* SMS, o *poller* faz um pedido (através de um URL) à *Clickatell* para que a SMS seja enviada. O método responsável pelo envio das SMS é o seguinte:

```

public static function enviarSMS($numeroSubscritor, $estruturaCampanha, $smsFlash) {
    $message = base64_decode($estruturaCampanha);
    $smsUser = 'XXXX'; // User da API clickatell
    $smsPassword = 'XXXX'; // Password da API
    $remetente_api_id = '3390364';
    $destination = $numeroSubscritor;
    $source = 'Punchline';
    $from = urlencode($source);

    $baseurl = "http://api.clickatell.com";
    $text = urlencode($message);
    $to = urlencode($destination);

    $url = "$baseurl/http/auth?user=$smsUser&password=$smsPassword&api_id=$remetente_api_id";
    $ret = file($url);
    $sess = explode(":", $ret[0]);
    $resposta = null;
    if ($sess[0] == "OK")
    {
        $sess_id = trim($sess[1]);

        if ($smsFlash === true) { // SMS do tipo FLASH
            $url = "$baseurl/http/sendmsg?session_id=$sess_id&to=$to&text=$text&from=$from&msg_type=SMS_FLASH";
        } else {
            $url = "$baseurl/http/sendmsg?session_id=$sess_id&to=$to&text=$text&from=$from";
        }
        $ret = file($url);
        $send = explode(":", $ret[0]);
        error_log('SMS RESPONSE => ' . print_r($ret, 1));
        if ($send[0] == "ID") {
            $envio = true; // Recebeu o ID da mensagem recebida pela API, sucesso
            // Verificar qual o status da mensagem enviada, através do envio: api id, msg_id,
            username, password

            $msg_id = str_replace(" ", "", $send[1]);
            error_log("ID da sms enviada >> " . $msg_id);

            $url_status_msg = "http://api.clickatell.com/http/getmsgcharge?api_id=$remetente_api_id
            &user=$smsUser&password=$smsPassword&apimsgid=$msg_id";

            $ret_status = file($url_status_msg);

            // ... Aqui está presente o código com o estado resultante do envio

        } else {
            $erros = $send = explode(":", $send[1]);
            error_log('ERROS >> ' . print_r($erros, 1));
            $envio = false;
        }
    } else {
        $envio = false;
    }
    return $resposta; // Envia o código do motivo de não envio se existir
}

```

Este método começa por fazer a autenticação na API *Clickatell* e após o sucesso da autenticação verifica qual o tipo de SMS que se pretende enviar sendo criado um URL específico para cada tipo. De seguida, é invocado através desse mesmo URL o envio da mensagem. Posteriormente ao envio, a API disponibiliza um URL para que se possa verificar o *status* da

mensagem que quando invocado retorna um código do estado. Atente-se que no excerto de código acima cortei esta parte do código colocando o comentário "... Aqui está presente o código com o estado resultante do envio" por este ser demasiado extenso e não ser de grande complexidade técnica.

Os estados mais comuns são os seguintes:

- **Status 003:** Mensagem enviada mas sem resposta da operadora;
- **Status 004:** Mensagem recebida;
- **Status 005:** Ocorreu um erro com a mensagem (erro genérico);
- **Status 007:** Ocorreu um erro na entrega da mensagem;
- **Status 008:** Mensagem recebida pela *gateway*;
- **Status 010:** Mensagem expirou;
- **Status 012:** A conta *clickatell* não tem créditos suficientes.

Depois de cada mensagem ser enviada, esta informação é adicionada à base de dados para que seja possível obter estatísticas a cerca da mesma.

Caso se pretenda enviar uma mensagem MMS, o *poller* obtém da base de dados o texto e os anexos correspondentes. O *array* com os anexos é serializado e codificado em formato URL para que possa ser enviado num pedido GET para o módulo que trata do envio da MMS.

Ainda antes de enviar a campanha, o conteúdo da MMS tem que ser codificado num formato próprio com extensão '.mms'. Para codificar o conteúdo em MMS, é usada a class *MMSEncoder* que tem como função adicionar as partes de média (imagens, som, etc) e a parte de texto num só ficheiro e codificá-lo num formato '.mms'.

Depois de codificar o conteúdo na extensão específica e de verificar que há saldo na conta do utilizador que criou a campanha a ser enviada, é invocado o método que permite enviar a MMS passando como parâmetros o

número do subscritor para quem será enviada a campanha, o URL do conteúdo codificado com a extensão '.mms' e o nome da campanha.

Depois de enviar a MMS – e à semelhança do que acontecia no envio das SMS – o *poller* obtém o estado de cada mensagem que são os mesmo do que os já apresentados para as SMS.

Envio de campanhas de fax

Se o *poller* de envio verificar que o tipo de campanha a enviar é fax, verifica na base de dados para que subscritores é que esta irá ser enviada e obtém os dados necessários de cada um deles (nome e número de telefone). Ainda antes de enviar a campanha, o *poller* obtém da base de dados o caminho do ficheiro associado à mesma.

Após já ter o número de telefone do subscritor, o nome do subscritor e o caminho do ficheiro que irá ser enviado, o *poller* verifica se a conta tem saldo suficiente para realizar o envio. Caso estejam reunidas todas as condições é invocado o método *trataFax* que é responsável pelo envio do fax mas não sem antes fazer a autenticação na API *PamFax*. Após a autenticação bem sucedida, é adicionado o subscritor através de um método estático presente na class *FaxJobApi*. Na Figura 6.13 verifica-se o excerto de código do método *trataFax* que permite fazer a adição do subscritor para quem será enviado o fax.

```
// Adiciona o subscritor
$resulta = FaxJobApi::AddRecipient($numFaxSubscritor, $nomeDoSubscritor);
```

Figura 6.13: Excerto do código que transmite à API PamFax o subscritor para quem será enviado o fax.

Depois de se adicionar o subscritor, é necessário indicar qual o ficheiro que se pretende enviar. Para isso, é necessário recorrer ao método estático *AddFile* da classe *FaxJobApi*. Este método apenas necessita de receber como parâmetro o caminho para o ficheiro.

Após adicionar o subscritor e o ficheiro que é para enviar, é executado o xecrto de código da Figura 6.14 que permite avaliar se está tudo conforme e se o fax pode realmente ser enviado.

```
$test = FaxJobApi::GetFaxState();

if (($test instanceof ApiError) || !isset($test['FaxContainer'])) {
// Fax não está pronto para ser enviado
    $state = "editing";
    return false;
} else
    $state = $test['FaxContainer']->state;

// send
if ($state == 'ready_to_send') {
    FaxJobApi::Send(); // Envia o fax
} else {
    FaxJobApi::SendLater();// É enviado automaticamente quando existirem créditos
}
}
```

Figura 6.14: Excerto de código que executa a validação e envio do fax

Através do método estático *GetFaxState* da classe *FaxJobAPI* verifica-se qual o estado do fax sendo que existem dois estados possíveis de retorno: *ready_to_send* e *not_enough_credit*. Caso o fax esteja com o estado *ready_to_send*, é enviado o fax através do método estático *Send* da classe referida anteriormente como se verifica no excerto de código acima. Caso não exista saldo na conta do *PamFax* é devolvido o estado *not_enough_credit* e, conseqüentemente, o envio não é autorizado. Quando a Punchline inserir créditos na conta *PamFax*, os envios pendentes serão realizados automaticamente.

6.3.3 Relatórios

Um dos requisitos mais pedidos relativamente à plataforma quando esta foi abordada com a equipa da *Punchline*, principalmente a equipa da formação que envia muitas *newsletters*, foi a possibilidade de ter uma análise das campanhas. Assim sendo, foi desenvolvido um módulo de relatórios que permite ao utilizador ver as estatísticas das suas campanhas enviadas.

Este componente permite obter muito mais detalhes e estatísticas quando se trata de uma campanha de *email* pois é a única em que o programador consegue ter mais controle. No caso das campanhas com APIs externas, não se consegue perceber, por exemplo, quem é que eliminou a campanha ou quem é que simplesmente não a abriu.

Nas campanhas de *email*, o utilizador pode averiguar que subscritores abriram a *newsletter*, que subscritores clicaram nos *links* presentes na *newsletter* e em que *links* clicaram. Ainda nesta campanha, o utilizador pode verificar em que horas do dia foram realizados os cliques e as aberturas das campanhas através de um gráfico que mostra sempre as últimas 12 horas. Foi desenvolvida também uma tabela que mostra os cliques e aberturas por hora do dia. Através desta tabela o utilizador pode verificar em que altura do dia é que as pessoas mais visualizam o *email*.

Do lado da programação, este componente precisa de uma série de pesquisas na base de dados (cliques realizados, aberturas de campanhas efetuadas e respetivas horas, número de cliques, número de aberturas, subscritores que abriram, etc.). Para desenvolver o *front end* deste componente, para além do HTML 5, CSS e *jQuery*, foi utilizada a biblioteca *Chart.js*.

O *Chart.js* [29] é uma biblioteca em *JavaScript* que auxilia na criação de gráficos utilizando apenas HTML, CSS e JS para desenhar os mesmos (utiliza o elemento *canvas* do HTML5 para desenhar o gráfico).

Para além das funcionalidades abordadas, a plataforma permite ao utilizador exportar o relatório com todos os detalhes em formato *.pdf*. Caso o utilizador pretenda, pode também exportar todos os *emails* dos subscritores que realmente abriram a campanha em formato *.xls*.

Na Figura 6.15 pode ver-se um relatório do envio de uma campanha de *email*.

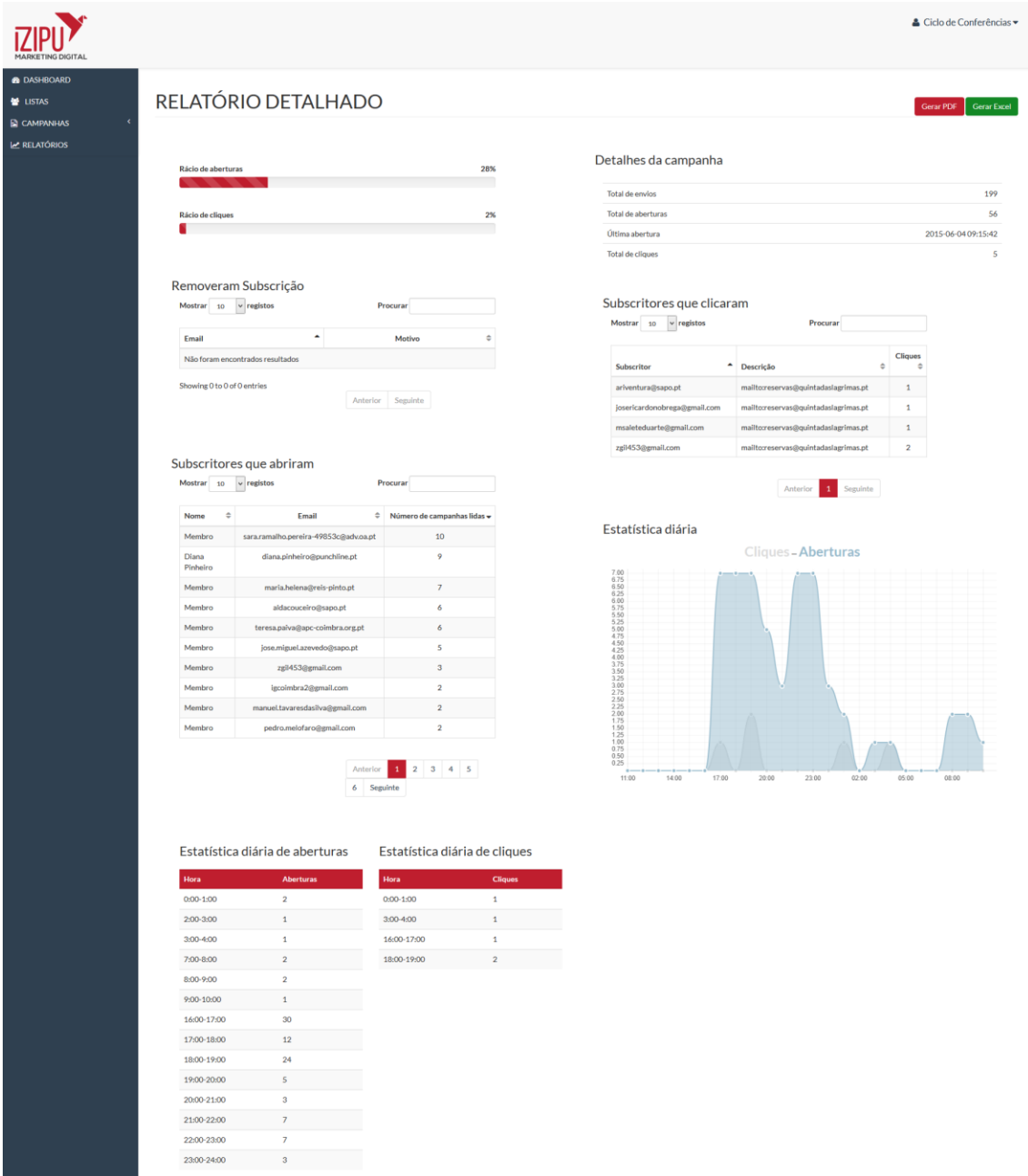


Figura 6.15: Relatório do envio de uma campanha de *email* com diversas estatísticas.

No caso das campanhas de SMS, MMS e voz, os relatórios incluem menos informação. Nas duas primeiras, o relatório indica apenas para quantos subscritores a campanha foi enviada e o número de mensagens que foram entregues. No caso das campanhas de voz, para além de discriminar o número de subscritores para os quais as campanhas foram enviadas, em caso de existência de gravação de voz por parte do interlocutor, o utilizador pode ouvir essas mesmas gravações na presente página. Por ser um

método de envio obsoleto e pouco usado, não foi implementado o módulo de relatórios para a campanha de fax.

6.3.4 Utilizadores

Para usarem a plataforma de *marketing* os utilizadores têm que se registar na mesma. Para isso são-lhes exigidos apenas dois dados obrigatórios, nome e *email*, e a password desejada. Para além destes dados o utilizador tem que concordar com os termos e condições da plataforma IZIPU para que o registo seja possível. Na Figura 6.16 verifica-se o ecrã que permite ao utilizador criar uma conta no IZIPU.

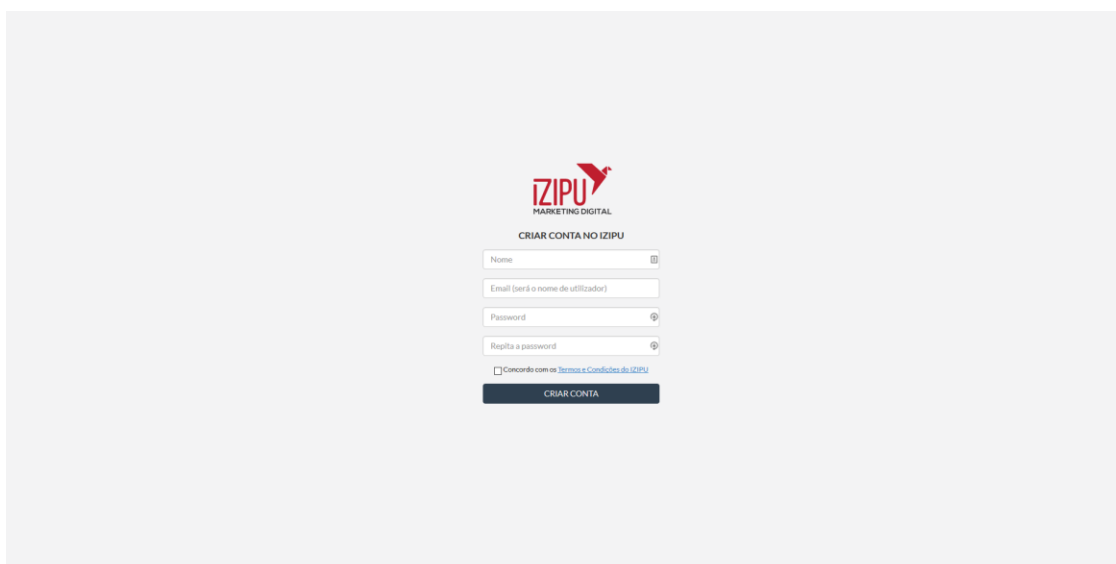
The image shows a web form for creating an account on IZIPU. At the top center is the IZIPU logo, which consists of the word 'IZIPU' in a bold, red, sans-serif font, with a red bird-like icon above the 'I'. Below the logo, the text 'MARKETING DIGITAL' is written in a smaller, black, sans-serif font. Underneath that, the heading 'CRIAR CONTA NO IZIPU' is displayed in a small, black, sans-serif font. The form itself contains five input fields: 'Nome', 'Email (será o nome de utilizador)', 'Password', and 'Repita a password'. Each field has a small icon on the right side. Below the 'Repita a password' field is a checkbox labeled 'Concordo com os Termos e Condições de IZIPU'. At the bottom of the form is a dark grey button with the text 'CRIAR CONTA' in white, uppercase letters.

Figura 6.16: Ecrã que permite criar conta no IZIPU

Internamente, quando os utilizadores se registam, a aplicação verifica na base de dados se o *email* indicado já se encontra registado. Em caso afirmativo, não é autorizada a criação da nova conta. Caso o *email* ainda não se encontre registado, a plataforma insere na base de dados o novo utilizador e envia um *email* para que este confirme o seu registo. O *email* é enviado através da biblioteca *PHPMailer* que será abordada no capítulo das dependências.

Depois do utilizador se autenticar e enquanto a plataforma verificar que nem todos os dados da conta foram preenchidos, é mostrada na *dashboard* uma notificação ao utilizador.

A secção dos dados da conta encontra-se subdividida em quatro secções: informação básica onde contra apenas o nome, *email* e contacto; alteração da *password* onde o utilizador pode escolher uma nova *password*; informação adicional onde consta a morada e a informação relativa à empresa e informação para faturação. Do ponto de vista da programação, a parte mais interessante nesta página é o componente da informação para a faturação. De maneira a simplificar a navegação ao utilizador, a plataforma dá a possibilidade ao utilizador de copiar para este componente os dados já inseridos na informação adicional. Ou seja, caso o utilizador clique no botão ‘Copia dados da informação’, internamente é feita uma pesquisa na base de dados, são obtidos os dados possíveis e são inseridos através de *jQuery* nos componentes HTML.

O utilizador pode ainda realizar outro tipo de configurações na sua conta, nomeadamente a configuração de uma assinatura. Essa assinatura será usada apenas nos *emails* enviados pelo sistema em seu nome como por exemplo quando este convida outro utilizador para fazer parte da sua conta.

CONFIGURAÇÕES

The screenshot displays the 'CONFIGURAÇÕES' (Settings) page of the IZIPU marketing platform. At the top, there are navigation tabs for 'Rodapé', 'Cabeçalho', and 'Textos'. The main content area is divided into two columns. The left column, titled 'Informação básica', contains a form for 'Informação principal' with fields for 'Nome' (filled with 'Diana Pinheiro'), 'Título' (filled with 'Web Deve'), 'Email' (filled with 'diana.pinheiro@punchline.pt'), and 'Contacto' (filled with '+351 919 999 999'). Below this are sections for 'Informação da empresa' and 'Social'. The right column, titled 'Visualizar assinatura', shows a preview of the signature configuration under the heading 'Definições'. It displays the name 'Diana Pinheiro', a phone number '+351 919 999 999', an email 'diana.pinheiro@punchline.pt', and the company address 'Punchline +351 239 899 899 / Rua Mário Pass, Coimbra http://www.punchline.pt'. A 'Ver código HTML' button is located at the bottom right of this section. At the bottom of the page, there are 'Limpar' and 'Guardar' buttons.

Figura 6.17: Ecrã da plataforma de *marketing* IZIPU onde é possível configurar a assinatura.

A componente da edição/construção de uma assinatura foi uma componente bastante interessante de desenvolver do ponto de vista gráfico

pois tem bastante manipulação de *jQuery*. Um dos pontos mais desafiantes em termos de manipulação de *jQuery* foi o facto de quando o utilizador está a editar qualquer campo da sua assinatura poder pré-visualizar, no momento, como esta vai ficar. Na Figura 6.17 verifica-se um exemplo de uma configuração de uma assinatura de um utilizador final da plataforma de *marketing*.

Depois de ter a sua assinatura configurada, o utilizador pode exportar todo o código HTML da mesma para que a consiga inserir noutra local que deseje.

6.3.5 Autenticação

Para desenvolver o módulo da autenticação de um utilizador a *Punchline* decidiu usar um método próprio sem recursos a autenticações externas como *Google* ou *facebook*. Sempre que um utilizador tenta fazer a autenticação, a palavra-passe é encriptada através do algoritmo de *hash* SHA-1⁷ e é verificada na base de dados se existem dados que correspondam aos inseridos de forma a perceber se o utilizador é válido. Na Figura 6.18 verifica-se que para encriptar a *password* basta utilizar o método de PHP *sha1*.

```
// Codifica uma password com SHA-1
function codificaPassword($password) {
    return sha1($password);
}
```

Figura 6.18: Excerto de código que permite encriptar a password

6.3.6 Subscritores

A seguir ao módulo das campanhas, este foi aquele com mais funcionalidades e com alguma complexidade do ponto de vista da programação.

⁷ Algoritmo de um *hash* de 160 bits. O algoritmo SHA-1 gera uma *string* alfanumérica de 40 caracteres [43]

Os subscritores são os destinatários de qualquer campanha desenvolvida pelo utilizador. Estes estão distribuídos por listas, ou seja, antes do utilizador inserir um subscritor tem que seleccionar ou criar uma lista.

Cada lista exige que o utilizador insira dois dados obrigatórios de cada subscritor, nome e *email*. Imagine-se que o utilizador queria criar a lista 'ISEC' onde iria inserir todos os subscritores da instituição mas pretendia que fosse obrigatório saber também o curso de cada subscritor para poder segmentar a lista mais facilmente. A plataforma de *marketing* permite que sejam adicionados campos extra às listas. Ver Figura 6.19.

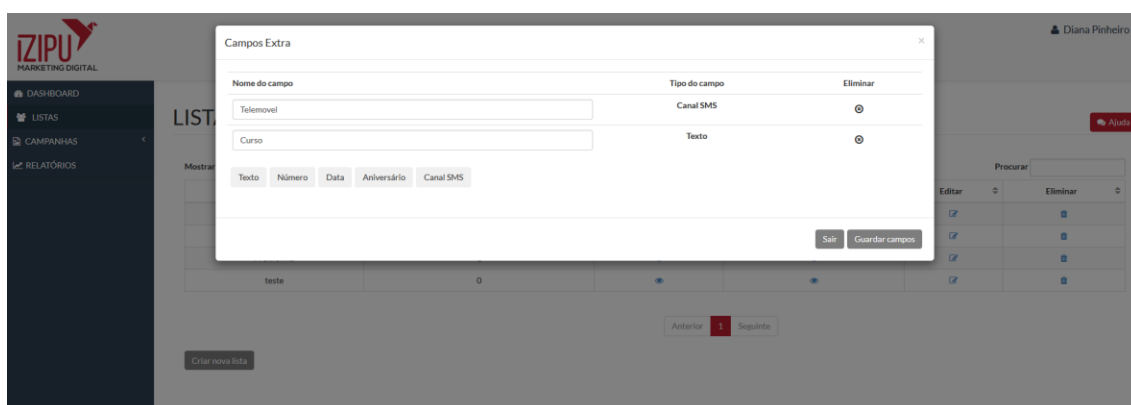


Figura 6.19: Ecrã da plataforma de *marketing* IZIPU que permite ao utilizador adicionar diversos campos extra a cada lista de subscritores

Para adicionar um campo extra, o utilizador tem que seleccionar o tipo de campo que deseja como apresentado na imagem e qual o nome do campo.

Internamente, existem três tabelas na base de dados para dar suporte a este módulo: *lista*, *lista_dadosextra* e *subscritores*. O que a plataforma faz é associar o ID da lista a cada dado extra inserido na tabela.

Para uma utilização mais simplificada, a equipa decidiu que seria interessante dar a possibilidade aos utilizadores de importarem subscritores em grande quantidade. A melhor solução para tal e a mais confortável para o utilizador é que este possa inserir os dados através de um ficheiro editável no Excel. Pelo facto, decidiu-se então usar o .xml.

Para garantir uma margem mínima de erros na importação, a equipa decidiu que a plataforma deveria exportar um *template* que o utilizador editava e voltava a inserir o mesmo *template*. No entanto, surgiu uma dificuldade: como cada lista tem os seus campos extra, como é que o utilizador insere esses dados extra de cada subscritor? Obviamente o processo de inserção resultaria em bastantes erros.

A solução encontrada foi gerar um *template* para cada ID da lista. Internamente, é obtido o ID da lista a que o utilizador pretende adicionar os subscritores, são obtidos todos os campos extra associados a essa lista e estes são inseridos num ficheiro XML e exportados para o utilizador. O nome deste ficheiro é composto por 'subscritores_idDaLista.xml' para que ao importar os subscritores a plataforma consiga validar se realmente os subscritores estão a ser inseridos na lista correcta.

Todos os subscritores são adicionados a uma base de dados e é enviado um *email* para cada um deles a notificar que foram adicionados como subscritores para que estes confirmem se realmente pretendem receber campanhas.

Quando se mostrou este componente às restantes empresas da *Punchline*, nomeadamente às da formação, estas foram da opinião que, como as listas assumiam rapidamente um elevado número de subscritores, havia a necessidade de possibilitar ao utilizador fazer uma segmentação das mesmas.

SUBSCRITORES DA LISTA 'Grupo P'

Subscritores

Mostrar 10 registos

Procurar

Nome	Email	Editar	Eliminar
Bruna Goes	bruna.goes@punchline.pt		
Diana Pinheiro	diana.pinheiro@punchline.pt		
Paulo Pinto	paulo.pinto@pleasures.pt		
Paulo Pinto	paulo.pinto@punchline.pt		

Anterior 1 Seguinte

Filtros Importar subscritores Adicionar Subscritor Voltar

Colaboradores da Punchline

Subscritores que contêm todas condições definidas:

Email contém punchline

Guardar segmento Visualizar segmento

Resultado do segmento	
Nome	Email
Diana Pinheiro	diana.pinheiro@punchline.pt
Paulo Pinto	paulo.pinto@punchline.pt
Bruna Goes	bruna.goes@punchline.pt

A figura acima mostra o exemplo da criação de um segmento na plataforma de *marketing*. O utilizador pode filtrar a lista pelos campos que desejar e pré visualizar o segmento para que perceba se realmente é o que pretende. Na imagem constata-se que a lista é composta por quatro elementos e que o utilizador pretende criar um segmento que filtre apenas os colaboradores da *Punchline*. Para tal seleciona por onde pretende filtrar, neste caso o *email*, e diz que este contém '*Punchline*'. Ao pré-visualizar, verifica-se que apenas são mostrados três utilizadores.

6.3.7 Créditos

De acordo com a campanha enviada, o utilizador gasta determinados créditos. À data da escrita desta secção, os créditos que a *Punchline* definiu para cada tipo de envio de campanha foram os seguintes:

- Campanha de *email*: 1 crédito;
- Campanha de SMS: 3 créditos;
- Campanha de MMS: 4 créditos;
- Campanha de voz: 2 créditos;
- Campanha de fax: 3 créditos.

O utilizador pode, sempre que pretender, consultar os créditos da sua conta e comprar créditos. Na Figura 6.20 verifica-se o ecrã onde é possível consultar os créditos que a conta ainda tem disponíveis.

Para o desenvolvimento desta página foi necessária uma consulta à base de dados à tabela 'pacote' para obter todos os pacotes da plataforma. Os pacotes obtidos são mostrados numa tabela e para cada pacote é adicionado um botão 'Comprar' para que o utilizador possa adicionar créditos à sua conta.

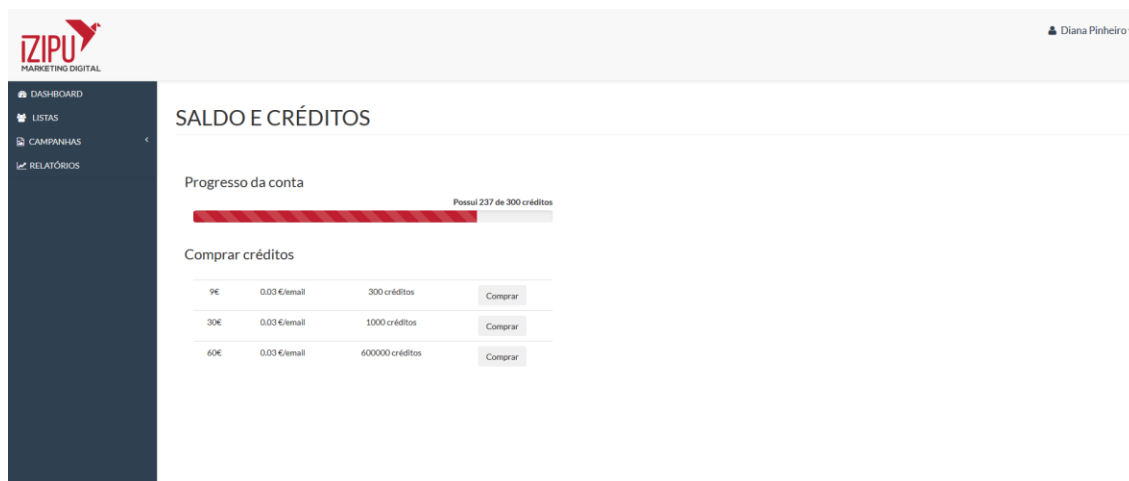


Figura 6.20: Ecrã onde o utilizador pode consultar os créditos da conta

Quando o utilizador prime o botão eliminar são mostrados os três meios de pagamento que a plataforma disponibiliza: cheque, transferência bancária e *PayPal*. Na Figura 6.21 verifica-se o ecrã que é mostrado ao utilizador quando prime o botão comprar.

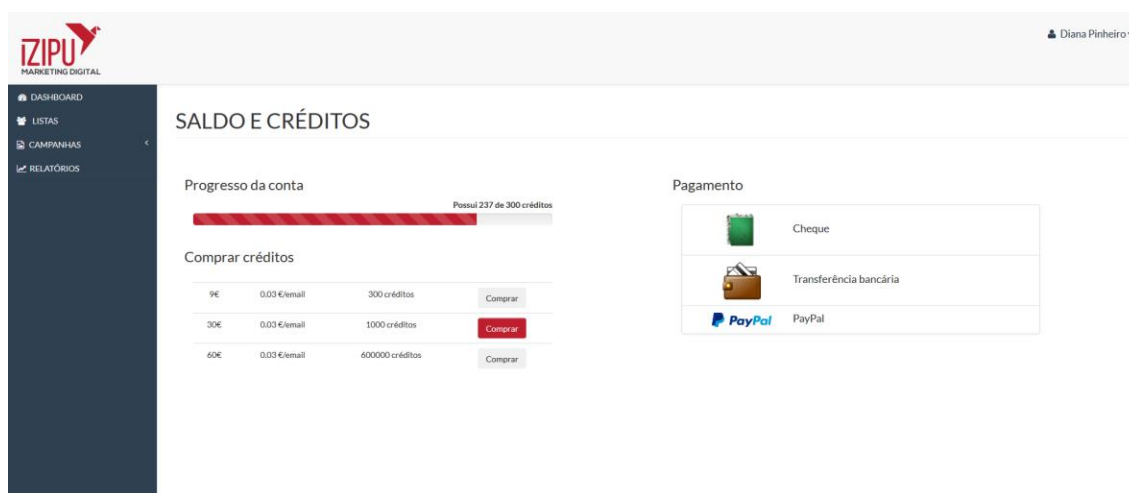


Figura 6.21: Ecrã com os meios de pagamento

Caso o utilizador decida pagar com cheque ou por transferência bancária, este é apenas notificado da morada de entrega do cheque ou do NIB da empresa a quem terá que fazer o pagamento.

O *PayPal* foi, obviamente, o método de pagamento mais interessante de desenvolver pois a estagiária desconhecia a API. Quando o utilizador prime o botão comprar, internamente, é obtido o ID do pacote escolhido e é enviado através de um *POST* para a classe responsável por invocar o *Paypal*. Atente-se na Figura 6.22 um excerto de código do método responsável pelo pagamento através de *PayPal*.

Antes de redirecionar o cliente para o ecrã de pagamento, é necessário transmitir à API todos os dados necessários que são enviados através de um *array*: *username*, *password* e *key* fornecidos pelo *PayPal*; Método '*SetExpressCheckout*' que indica ao *PayPal* que irá ser realizada uma transação; tipo de transação - '*sale*' - que indica que se trata de uma venda; tipo de moeda pretendida; valor do pacote comprado, descrição do produto, URL para redirecionar o utilizador depois de efetuar o pagamento e URL para onde redirecionar em caso de erro.

Posteriormente faz-se a requisição à API do *PayPal* através do método *sendNvpRequest*. Este método retorna um *array* vazio em caso de erro ou um *array* com uma *key* denominada de *ACK* e outra *key* denominada de *token* que será enviado no URL que redireciona o utilizador para o pagamento.

Quando o *paypal* redireciona o utilizador para a janela de pagamento, este visualiza um ecrã que se verifica na Figura 6.23.

```

//Baseado no ambiente, sandbox ou produção, definimos as credenciais
//e URLs da API.
//Credenciais da API para o Sandbox
$username = 'informatica-facilitador_api1.pleasures.pt';
$password = 'J3S98BV4EQBHLYFN';
$key = 'AawMKS9i-nwQsb16b-Gdh7tuxWtRAu71bkFPMQJSaxRGjqJk9Uw3riu9';

//URL da PayPal para redirecionamento
$paypalURL = 'https://www.sandbox.paypal.com/cgi-bin/webscr';

//Campos da requisição da operação SetExpressCheckout, como ilustrado acima.
$dadosPedidos = array(
    'USER' => $username,
    'PWD' => $password,
    'SIGNATURE' => $key,
    'VERSION' => '108.0',
    'METHOD' => 'SetExpressCheckout',
    'PAYMENTREQUEST_0_PAYMENTACTION' => 'SALE',
    'PAYMENTREQUEST_0_AMT' => $valorPacoteComprado,
    'PAYMENTREQUEST_0_CURRENCYCODE' => 'EUR',
    'PAYMENTREQUEST_0_ITEMAMT' => $valorPacoteComprado,
    'L_PAYMENTREQUEST_0_NAME0' => 'Produto',
    'L_PAYMENTREQUEST_0_DESC0' => $descricaoProduto,
    'L_PAYMENTREQUEST_0_AMT0' => $valorPacoteComprado,
    'L_PAYMENTREQUEST_0_QTY0' => '1',
    'HDRIMG' => MEDIA.'images/izipu-logo.png',
    'LOCALECODE' => 'pt_BR',
    'RETURNURL' => PUBLIC_ROOT.'saldoEcreditos/sucessoPagamentoPaypal?idPacote=.'.$idPacoteObtido,
    'CANCELURL' => PUBLIC_ROOT.'/erro.php',
    'BUTTONSOURCE' => 'BR_EC_EMPRESA'
);

//Envia a requisição e obtém a resposta da PayPal
$responseNvp = sendNvpRequest($dadosPedidos, $sandbox);

//Se a operação tiver sido bem sucedida, o cliente é redirecionado para o ambiente de
pagamento.
if (isset($responseNvp['ACK']) && $responseNvp['ACK'] == 'Success') {
    $query = array(
        'cmd' => '_express-checkout',
        'token' => $responseNvp['TOKEN']
    );

    $redirectURL = sprintf('%s?%s', $paypalURL, http_build_query($query));

    header('Location: ' . $redirectURL);

    printf("Location: %s\n", $redirectURL);
}

```

Figura 6.22: Excerto de código do método que invoca o PayPal

Quando o utilizador termina o seu pagamento, o *PayPal* redireciona novamente para a plataforma de *marketing*. De seguida, através do ID do pacote presente no URL, é feita uma pesquisa na BD para obter o número de créditos e estes são adicionados à conta do utilizador autenticado.

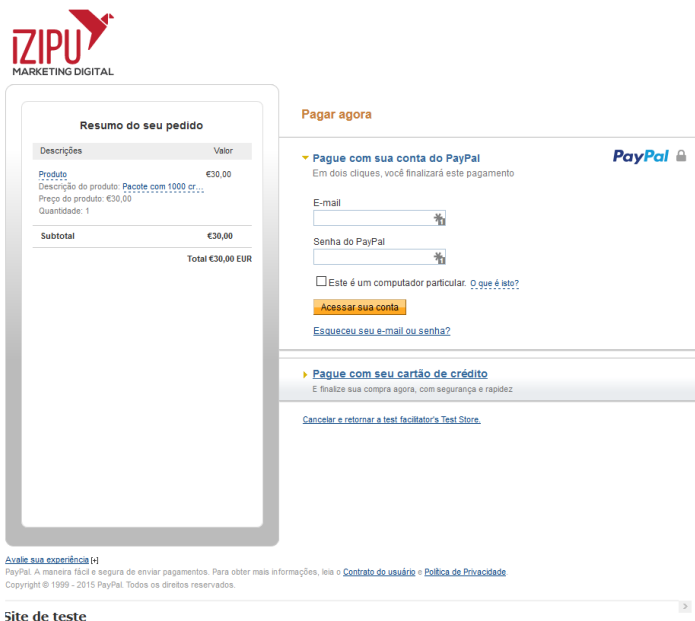


Figura 6.23: Ecrã de pagamento no PayPal

6.3.8 Conta corrente

Como esta é uma plataforma em que o utilizador tem que creditar dinheiro a equipa decidiu que devia ser desenvolvido um módulo para que o utilizador pudesse verificar o número de créditos que foi gasto com uma breve descrição de cada movimento da conta. Na Figura 6.24 verifica-se uma captura de ecrã com os movimentos da conta do utilizador autenticado.

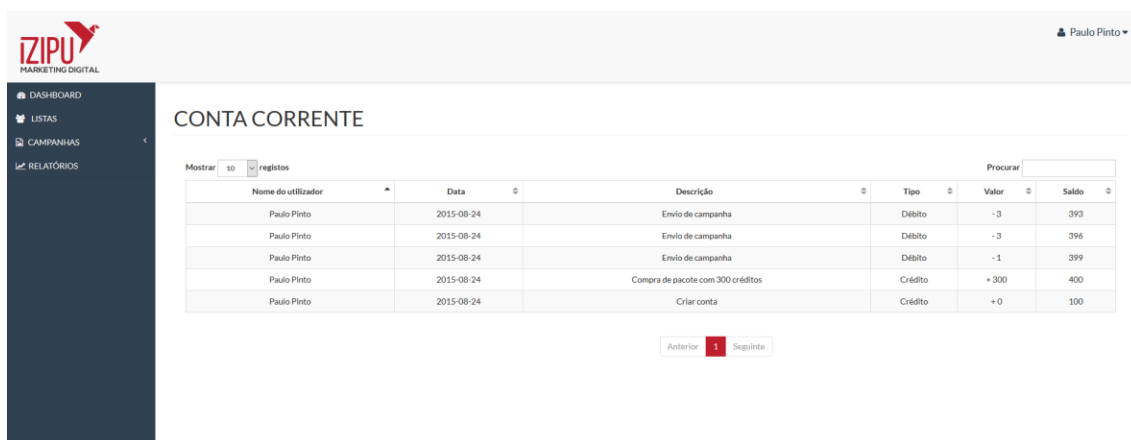


Figura 6.24: Ecrã onde o utilizador pode consultar os seus movimentos

Internamente, sempre que é enviada uma campanha, a plataforma desconta os créditos do saldo da conta e é adicionado à tabela 'utilizador_conta_corrente' o registo do movimento. Para mostrar todos os movimentos ao utilizador, é feita uma pesquisa à base de dados filtrando os registos por *ID* da conta e estes registos são mostrados numa tabela usando a o *plugin* para *jQuery – DataTables*. Este *plugin* é uma ferramenta bastante flexível que permite adicionar funcionalidades avançadas em qualquer tabela HTML de forma simples como pro exemplo a ordenação de colunas, mostrar ou esconder colunas de uma tabela, pesquisar dados de uma tabela, entre outros [30].

Na Figura 6.24 verifica-se que a tabela permite ao utilizador efetuar uma pesquisa, escolher quantos registos este pretende visualizar e ordenar as colunas clicando na setinha que tem em cada cabeçalho de cada coluna. Na Figura 6.25 verifica-se o excerto de código *jQuery* que permite adicionar estas funcionalidades à tabela com os movimentos da conta.

```
$('#tabelaListaMovimentos').dataTable(  
  {  
    "searching": true,  
    "ordering": true,  
    "paging": true,  
    "language": {  
      "sProcessing": "A processar...",  
      "sLengthMenu": "Mostrar _MENU_ registos",  
      "sZeroRecords": "Não foram encontrados resultados",  
      "sInfoFiltered": "(filtrado de _MAX_ registos no total)",  
      "sInfoPostFix": "",  
      "sSearch": "Procurar ",  
      "oPaginate": {  
        "sFirst": "Primeiro",  
        "sPrevious": "Anterior",  
        "sNext": "Seguinte",  
        "sLast": "Último"  
      }  
    }  
  },  
  {}  
});
```

Figura 6.25: Excerto de código *jQuery DataTables*

6.3.9 Remetentes

Para enviar campanhas de *email*, o utilizador tem que primeiramente definir de quem é que vai ser enviada a campanha, ou seja, quais são os remetentes. De um modo mais simples, o remetente é o *email* que o subscritor verá no campo 'From' na sua caixa de correio eletrónico.

Internamente, existe uma tabela designada de 'remetentes' na base de dados, onde é inserido um registo associado ao ID da conta sempre que o utilizador criar um novo remetente.

Na Figura 6.26 verifica-se o excerto de código HTML com o formulário que é apresentado ao utilizador para que este possa adicionar um remetente. Quando o utilizador prime o botão 'Adicionar' é captado o evento do botão através de *jQuery*. Neste evento, é feito um pedido *ajax* para o url onde o remetente será guardado.

```
<div class='row'>
  <div class='col-md-6'>
    <label>Adicionar novo remetente</label>
    <input type='text' class='form-control' id='nomeRemetente' name='nomeRemetente' />
    <input type='email' class='form-control' id='emailRemetente'
name='emailRemetente' />
    <input type='hidden' id='idConta' value="{!!
Session::get('utilizador_conta_id')!!}" />
    <button type='submit' class='btn' id='btnAdicionarRemetente' value="Adicionar" />
  </div>
</div>
```

Figura 6.26: Excerto de código HTML com o formulário para adicionar remetente

Na Figura 6.27 verifica-se o excerto de código *jQuery* que trata o evento do botão que permite adicionar remetente. Neste evento, são obtidos os dados que o utilizador inseriu e é realizado um pedido *ajax* para que os dados sejam guardados.

```
$('#btnAdicionarRemetente').click(function(){
  $.ajax({
    type: "POST",
    url: "{!! url('adicionaRemetente') !!}",
    dataType: 'json',
    data: {
      idConta: $('#idConta').val(),
      nomeRemetente: $('#nomeRemetente'),
      emailRemetente: $('#emailRemetente'),
      tipo: 1,
      remetente_activo: 1
    },
    success: function (data) {
      /*Código que acrescenta o remetente à tabela*/
    }
  });
});
```

Figura 6.27: Excerto de código *jQuery* que trata o evento do botão 'Adicionar'

Do lado do servidor, os dados são recebidos e é adicionado um novo registo na tabela 'remetentes' da base de dados. Na Figura 6.28

verifica-se o excerto do código da função que recebe os dados e cria o registo na base de dados.

```
public function ajaxAdicionarRemetente() {
    $id = Request::get('idConta');
    $tipo = Request::get('tipo');
    $nome = Request::get('nomeRemetente');
    $email = Request::get('emailRemetente');
    $activo = Request::get('remetente_activo');

    Remetente::create([
        'utilizador_conta_id' => $id,
        'tipo_campanha_id' => $tipo,
        'remetente_designacao' => $nome,
        'remetente_email' => $email,
        'remetente_activo' => $activo
    ]);

    /* Aqui está presente o código que verifica se os dados foram guardados com sucesso e
    envia a resposta ao cliente */
}
```

Figura 6.28: Excerto de código da função que recolhe os dados enviados no pedido e que cria um registo na tabela 'remetentes'

Na Figura 6.29 verifica-se uma lista de remetentes de uma conta e um formulário para adicionar um remetente.

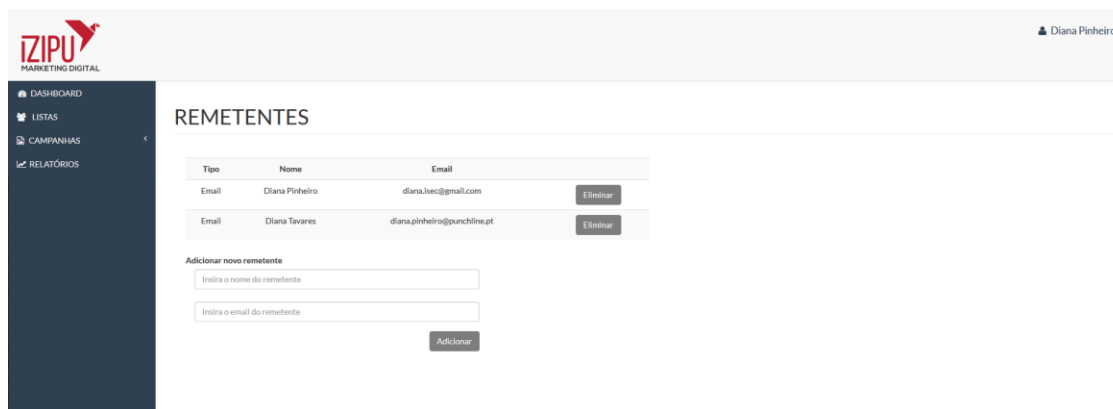


Figura 6.29: Ecrã onde é possível visualizar e adicionar remetentes

Depois de o utilizador ter remetentes criados na sua conta, quando criar uma campanha, e como se verifica na Figura 6.30, existe um seletor para que o utilizador selecione o remetente pretendido para a campanha.

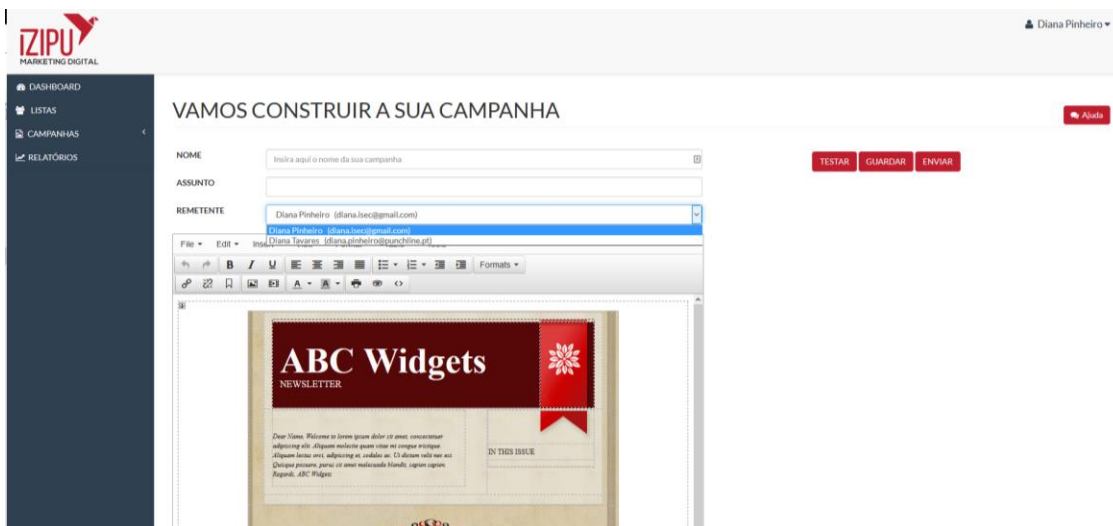


Figura 6.30: Seletor de remetentes aquando da criação da campanha

6.3.10 Equipa

A pensar em empresas, a *Punchline* chegou à conclusão de que seria útil se para uma mesma conta pudessem ser adicionados utilizadores com permissões diferentes. Imagine-se que o diretor da empresa cria uma conta na plataforma mas não tem tempo para criar as campanhas nem para gerir a conta. Poderá convidar outros utilizadores para se juntarem à sua conta com as permissões que este desejar. Por exemplo, pode criar um utilizador apenas com permissão para criar campanhas, poderá convidar um utilizador que apenas pode tratar da faturação, etc. Para uma melhor perceção pode ver-se a Figura 6.31.

Nome de utilizador	Nome	Permissão	Criar SMS	Criar MMS	Criar Voz	Criar email	Ver relatórios	Ver campanhas	Faturação	Estado
diana.pinhoiro@punchline.pt	Diana Pinheiro	Administrador	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Activo

Convidar utilizador para a minha conta

Insira o email do convidado

Permissão do utilizador convidado

Administrador - Acesso a todas as funcionalidades.

Autor - Não tem permissão para enviar campanhas.

Criar campanhas SMS

Criar campanhas MMS

Criar campanhas Email

Criar campanhas Voz

Criar campanhas Fax

Visualizador - Tem permissão apenas para ver os relatórios.

Gestor - Possui acesso a todas as funcionalidade à excepção da faturação.

ENVIAR CONVITE

Figura 6.31: Ecrã que permite ao utilizador convidar outros utilizadores para a sua equipa e atribuir-lhes permissões.

Internamente, sempre que o utilizador se regista na plataforma é criada automaticamente uma conta e este passa a ser dono da conta. Quando o utilizador convida outro utilizador para a sua conta, este tem que se registar na plataforma através do *link* enviado no *email* do convite e ao invés de ser criada automaticamente uma nova conta, o utilizador inserido é associado ao ID da conta do utilizador que convidou. Todos os módulos da plataforma, nomeadamente os de saldos da conta, compra de créditos, movimentos, etc. funcionam em função do ID da conta e não do ID do utilizador autenticado para que todos os utilizadores tenham acessos aos mesmos dados da conta, ou seja, se um determinado utilizador tiver 300 créditos na conta e convidar outro utilizador para a sua conta, este passa a ver na sua conta também 300 créditos.

As Permissões que podem ser atribuídas aos utilizadores convidados da equipa são as seguintes:

- **Administrador:** Possui acesso a todas as funcionalidades da plataforma de *marketing*.
- **Autor:** Não tem permissão para enviar campanhas. No entanto pode criá-las. Ainda dentro desta permissão, o administrador pode escolher que tipos de campanhas é que o utilizador convidado pode criar.

- **Visualizador:** Apenas tem permissão para ver os relatórios de cada campanha enviada com as respetivas estatísticas.
- **Gestor:** Possui acesso a todas as funcionalidades exceto à faturação.

6.4 WEBSITE DE APRESENTAÇÃO ([HTTP://WWW.IZIPU.COM](http://www.izipu.com))

Após o desenvolvimento de toda a plataforma de *marketing* IZIPU surgiu a necessidade desta ser promovida e concluiu-se que a melhor solução seria o desenvolvimento de um website institucional que permitisse ao utilizador averiguar todas as funcionalidades disponibilizadas.

Visto ser um site pouco complexo, este foi construído com recurso ao CMS⁸ *Wordpress*.

O website construído é apenas informativo e dispõe de um formulário de contacto que permite ao utilizador colocar as suas dúvidas e saber mais informações sobre a plataforma. Através deste website, o utilizador pode ainda aceder à plataforma de *marketing* IZIPU.

⁸ Ferramenta que permite a um editor criar, classificar e publicar qualquer tipo de tipo de informação numa página *web*.

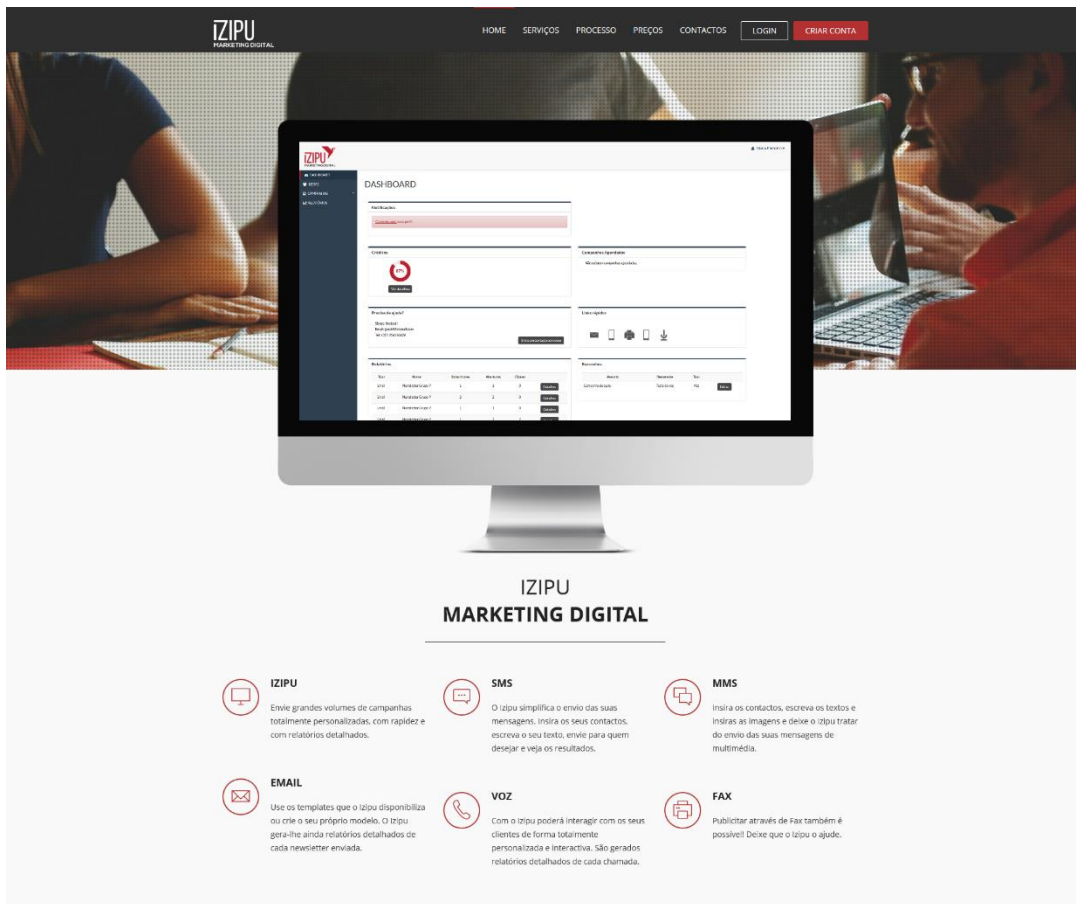


Figura 6.32: Website de apresentação do IZIPU

6.5 DEPENDÊNCIAS

Uma vez que não eram previsíveis as dependências necessárias à implementação da plataforma aquando do início do projeto, estas foram sendo descritas ao longo do desenvolvimento e por esse motivo apenas são abordadas neste capítulo.

6.5.1 jQuery

jQuery é um *wrapper* para as funcionalidades básicas de manipulação de *DOM*, que auxiliam o trabalho do programador ao nível de manipulação assíncrona de interface [31]. Esta biblioteca foi utilizada intensamente no desenvolvimento da plataforma.

6.5.2 Bootstrap

O Bootstrap é uma *framework front-end* que permite aos programadores *web* desenvolver plataformas ou websites responsivos sem ter escrever uma única linha de CSS [32]. O Bootstrap possui uma diversidade de componentes (*plugins*) em *jQuery* que auxiliam o programador na implementação tais como *tooltips*, *modal*, *slideshow*, entre outros. Se um programador não utilizador o Bootstrap, caso pretenda implementar, por exemplo, um componente *tooltip* numa página tem que proceder da seguinte forma:

- Encontrar um *plugin* que permite fazer *tooltips*
- Acrescentar o *script* ao HTML;
- Inicializar o *plugin* através de um *script*;
- Criar a estrutura HTML baseada na documentação do *plugin*.

Usando o Bootstrap, o processo é muito mais simples e o utilizador só tem que adicionar algumas configurações no código. Na Figura 6.33 verifica-se um exemplo de implementação com o auxílio do Bootstrap.

```
1 <a href="#" title="Título que aparece como tooltip" data-toggle="tooltip">Link com Tooltip</a>
2 <script>
3 $(function(){
4   $('[data-toggle="tooltip"]').tooltip();
5 })
6 </script>
```

Figura 6.33: Implementação de uma *tooltip* usando Bootstrap

6.5.3 TinyMCE

O TinyMCE [33] é um editor WYSIWYG (*What You See Is What You Get*) totalmente escrito em *JavaScript*. Esta biblioteca foi utilizada para a criação das campanhas de *email*. Internamente, o que o TinyMCE faz é substituir todos os campos de texto HTML (*textarea*) por um editor visual permitindo assim ao utilizador trabalhar sobre um editor de texto a que está habituado.

6.5.4 Responsive File manager

Responsive File manager é um gestor de ficheiros *open source* que permite aos utilizadores da plataforma fazer *upload* de ficheiros de uma forma elegante e intuitiva. Esta biblioteca pode ser usada como plugin externo do *tinyMCE* sempre que o utilizador pretender inserir imagens no editor de texto.

Esta biblioteca permite ao utilizador uma pré-visualização de todas as imagens que inserir.

6.5.5 Clickatell

A *Clickatell* dispõe de várias APIs mas a utilizada para o envio de SMS/MMS foi a *HTTP/S SMS API*. Segundo a *Clickatell*, esta é a API mais popular entre os programadores [34].

Apresentam-se em seguida algumas funcionalidades da *HTTP/S SMS API*:

- Suporta texto, Unicode e mensagens *flash*;
- Suporta mensagens com grande número de caracteres;
- Converte formatos de *media* para o formato correto;
- Confirma entrega;
- Permite especificar filas de prioridades;
- Permite especificar o tempo de expiração do envio de uma mensagem.

6.5.6 Twilio

O *Twilio* [35] é um serviço baseado na *cloud* para desenvolvimento de aplicações de Voz, SMS e VOIP usando apenas uma das APIs disponibilizadas.

Como a *Punchline* já está bastante ambientada e confortável com a *Clickatell* optou-se por se usar o *Twilio* apenas para Voz.

A API que o *Twilio* disponibiliza para voz é designada por *TwIML*. O *TwiliML* [36] não é mais que um conjunto de instruções que permite ao

programador transmitir ao *Twilio* como deve proceder no caso de receber uma chamada. Quando o *Twilio* recebe uma chamada num número comprado previamente pelo programador, procura o *url* associado a essa mesma chamada e faz um pedido a esse mesmo *url*. A plataforma de *marketing* deve responder ao pedido com as instruções nas *tags* apropriadas e definidas pela *API* para que o *Twilio* determine o que tem a fazer.

Tags disponibilizadas pelo *TwIML* e que são utilizadas no desenvolvimento da plataforma de *marketing*:

- **Say:** Transmite ao *Twilio* para que leia o texto ao interlocutor;
- **Play:** Reproduz um ficheiro de áudio ao interlocutor;
- **Record:** Instrui o *Twilio* para gravar a voz do interlocutor
- **Gather:** Permite invocar a ação da recolha dos dígitos inseridos pelo interlocutor. Esta *tag* é bastante usada quando se quer pedir ao interlocutor para que assinale um dígito em função do que pretende.
- **Redirect:** Intrui o *Twilio* de que deve redirecionar a chamada para outro número.

6.5.7 PamFax

Possível de ser integrado em sistemas Android, Windows, Apples, Linux e Web, o *PamFax* pode ser usado para enviar os documentos que o utilizador desejar através de um 'fax' virtual mediante a cobrança de uma pequena taxa pelo envio de cada fax.

O *PamFax* disponibiliza uma *API* que permite ao programador conceber na sua aplicação todas as funcionalidades que eles disponibilizam nas próprias aplicações. Para além de esta *API* permitir ao programador ter acesso a todas as funcionalidades do portal através do código, a *PamFax API* é ideal para adicionar a funcionalidade de fax à aplicação desenvolvida de forma a aumentar a atratividade da solução [37]

6.5.8 Paypal

O *Paypal* é um sistema que permite a transferência de dinheiro entre empresas/indivíduos usando apenas um endereço de *email* [38]. Uma vantagem de usar o *Paypal* é que todas as informações pessoais como dados bancários e números de cartões de créditos nunca são revelados aos vendedores no ato da compra.

Para realizar os pagamentos da compra dos créditos, a *Punchline* decidiu permitir aos utilizadores que o fizessem através do *Paypal*. Para isso, foi necessário ter recurso à *API Express Checkout* [39]. Esta *API* permite todo o *checkout* da compra na aplicação desenvolvida. Através das chamadas à *API*, é possível montar toda a estrutura de finalização de pagamento na plataforma e redirecionar o comprador para o ambiente *paypal* a que está habituado, para que este se sinta mais confiante e seguro.

6.6 TESTES

Para validar e verificar se o desenvolvimento da plataforma está conforme, a *Punchline* faz o uso de testes de aceitação.

Os testes de aceitação são testes de caixa-preta realizados no sistema antes da sua disponibilização. O objetivo destes testes é garantir que o sistema é capaz de executar a funcionalidade acordada da forma desejada, e consequentemente, garantir a qualidade do produto desenvolvido [40].

Para evitar que a plataforma falhe em termos de usabilidade e simplicidade no *workflow*, a equipa teve em atenção esse aspeto. Com efeito, ao longo do desenvolvimento foram realizados testes com diversos elementos das equipas de outras áreas externas às tecnologias de informação, para se poder ter a noção de quais as dificuldades sentidas por estas pessoas e do que poderia representar vantagens para estes utilizadores comuns.

O plano de testes da plataforma de *marketing* IZIPU está compactado num ficheiro de formato *Excel* e é, naturalmente, demasiado extenso para ser apresentado em pormenor. Ainda assim serão apresentadas figuras com algumas das suas folhas mais representativas.

Na Figura 6.34 pode ver-se a folha de resumo de todos os testes de todos os componentes que integram a plataforma de *marketing* IZIPU. Cada um destes componentes tem direito à sua própria folha.

À data desta captura de ecrã estavam considerados 233 testes, que é necessário repetir sempre que é gerada uma nova atualização de qualquer um dos componentes. Sempre que é criado um novo componente ou quando um qualquer componente ganha novas funcionalidades o plano de testes é revisto e, por norma geral, aumentado.

Páginas Testes	Nº testes aprovados	Nº de testes reprovados	Nº de testes não implementados	Erros de CSS	Total de testes	Sucesso (%)
criar_conta	9	0	0	0	9	100
login	8	0	0	0	8	100
pagina_global	13	0	1	4	18	72
criar_campanha_email	42	1	2	3	48	88
Listas/subscritores/segmentos	46	0	0	1	47	98
Ver_todas_campanhas	8	0	0	0	8	100
dados_da_conta	16	0	0	0	16	100
gerir_equipa	24	0	1	0	25	96
remetentes	11	0	0	0	11	100
saldo_e_creditos	20	0	0	0	20	100
conta_corrente	3	0	0	1	4	75
utilizadores_teste	8	0	0	0	8	100
envio_das_campanhas	11	0	0	0	11	100
Total	219	1	4	9	233	

Figura 6.34: Folha com o resumo do plano de testes da plataforma

Na Figura 6.35 pode ver-se o procedimento a adotar quando é detetado um erro. Para além de se incluir uma breve descrição no próprio ficheiro, era também descrito na plataforma *trello* o erro para que a restante equipa de desenvolvimento tivesse conhecimento.

Sempre que o programador corrigisse o erro tinha que escrever na mesma folha qual era o problema e a solução utilizada.

Plano de testes

Ficheiro Editar Ver Inserir Formatar Dados Ferramentas Suplementos Ajuda Todas as alterações foram guardadas no Drive

Comentários Partilhar

2015/04/23

	A	B	C	D	E	F
20	O utilizador prime o botão "enviar" com o campo assunto preenchido e com o campo "nome da campanha" por preencher.	O campo "nome da campanha" notifica a letras vermelhas de que o utilizador deve inserir um nome da campanha.	OK		2015/04/21	
21	O utilizador preenche todos os campos necessários (assunto e nome da campanha) e prime o botão enviar.	O utilizador é redirecionado para uma página de continuação do envio da campanha.	OK	Como resposta ao clique, o utilizador é redirecionado para uma página em branco.	2015/04/21	Nome da classe Segmento mal escrito. 2015/
22	O utilizador na página de continuação primeo botão "voltar".	O utilizador é redirecionado novamente para a página de edição onde é apresentada a newsletter que o utilizador tinha elaborado.	OK	O utilizador é redirecionado para a página de edição mas o campo de edição da newsletter aparece em branco. O nome da campanha não está a ser preenchido com o nome do assunto.	2015/04/21	O nome do assunto não estava a ser enviado para a página de edição. Ao guardar na tabela, o nome da campanha estava a ser guardado com o nome do assunto. 2015/
23	O utilizador edita a newsletter e prime o botão "enviar" novamente.	O utilizador é redirecionado para uma página de continuação do envio da campanha.	OK		2015/04/21	
24	O utilizador na página de continuação primeo botão "voltar".	O utilizador é redirecionado novamente para a página de edição da newsletter. As últimas alterações realizadas na newsletter ainda são visíveis.	OK		2015/04/21	
25	O utilizador prime o botão "enviar". Na página seguinte, o utilizador prime o botão retroceder do próprio browser.	O utilizador é redirecionado para a página de edição da newsletter. As últimas alterações realizadas na newsletter ainda são visíveis.	!OK	A campanha não é apresentada de forma correta.	2015/04/21	
26	Enviar campanha - Passo 2					
		A página apresenta ao centro os tipos de envio que existem (Enviar já, agendar e anular/parar). Apresenta também as listas				

+ Punchline Elementos_gerais criar_conta login pagina_global criar_campanha_email Listas_subscritores_segmentos Ver_todas_cam

Figura 6.35: Folha do plano de testes onde é visível uma *issue*.

Para além dos testes de aceitação, foram realizados testes de usabilidade que têm por objetivo verificar a facilidade que uma determinada plataforma, ou *software*, possui de ser claramente compreendida ou manipulada pelos utilizadores [41]. As equipas de formação criaram e enviaram *newsletters* e à medida que sentiam dificuldades na utilização da plataforma, notificavam as mesmas à equipa de desenvolvimento com novas sugestões de melhoria.

7 CONCLUSÕES

O desenvolvimento das tecnologias da informação e da comunicação compeliu a alteração dos modelos de negócio e dos processos de trabalho das empresas à qual não foi exceção a área do *marketing*. O *marketing* digital está a autonomizar-se e, por esse motivo, a abandonar o lugar de complemento das técnicas publicitárias tradicionais e a assumir-se como o meio privilegiado de divulgação e comunicação empresarial hodierno.

A plataforma IZIPU nasceu da necessidade de tornar acessível – quer financeira quer tecnicamente – a criação e gestão de campanhas de *marketing* digital de elevado volume, quer a particulares, quer a empresas. Assente em cinco vetores de comunicação, o IZIPU pretende ser *player* líder de mercado, quer na comunicação business-to-business, quer na comunicação com e de particulares.

O presente estágio compreendeu assim as várias fases de desenvolvimento da IZIPU. Numa primeira fase do estágio, foram elicitados todos os requisitos e foi realizado um estudo da viabilidade do projeto com uma análise SWOT. Ao longo desta fase as reuniões foram quase sempre semanais para que o projeto fosse o mais delineado possível.

Posteriormente, foram iniciados os diagramas de casos de uso e, com a ajuda de uma ferramenta já abordada anteriormente, foram desenhados *mockups* de alguns ecrãs da plataforma de *marketing*.

A implementação da plataforma IZIPU iniciou-se ainda durante o primeiro semestre do estágio devido à necessidade de se conseguir lançar uma versão *beta* com o módulo de criação e envio de campanhas de *email* completo, para ser testado internamente na *Punchline*.

No final do estágio ficaram implementados todos os módulos e todas as funcionalidades declaradas nos objetivos, à exceção dos meios de pagamento por cartões de crédito/débito e por multibanco. A equipa da

Punchline decidiu que nesta fase apenas estariam disponíveis os meios de pagamento por cheque, transferência bancário e *PayPal*.

Embora não estivesse previsto no plano de trabalhos, foi desenvolvido um website de apresentação da plataforma de *marketing* IZIPU.

Apesar de já ser colaboradora na *Punchline* desde 2013, esta facultou à estagiária uma oportunidade privilegiada de iniciar o desenvolvimento de um projeto inovador com a presença de diversas tecnologias de ponta que, na sua maioria, lhe eram totalmente desconhecidas.

Além disso, a realização deste estágio permitiu-lhe participar nas várias fases de um projeto de desenvolvimento de *software*, nomeadamente na especificação de requisitos; no desenvolvimento da arquitetura e no desenho detalhado; no desenvolvimento de todos os módulos da plataforma de *marketing*; no desenvolvimento do interface com o utilizador e, finalmente, nos testes ao sistema.

Durante o período de estágio a estagiária esteve também envolvida de forma particularmente ativa nas fases de teste e controlo de qualidade, concebendo e executando planos de teste para as várias componentes que iam sendo construídas.

Ao longo da fase de implementação dos diversos módulos surgiram múltiplos desafios de engenharia em virtude da necessidade de ligação e integração com *APIs* específicas para cada módulo (*Twillio*, *Clickatell*, *PamFax*, etc.), as quais eram desconhecidas, quer pela estagiária, quer pela restante equipa de desenvolvimento. A necessidade de integração de tantas *APIs* diferentes tornaram o projeto não só desafiante como também gratificante, pois proporcionaram-lhe momentos de crescimento enquanto engenheira, com a inevitável aquisição de novos conhecimentos.

Por outro lado, o ambiente profissional proporcionado pela empresa de acolhimento durante o período de estágio revelou-se bastante profícuo, não só do ponto de vista profissional mas também pessoal.

Concluído o estágio, pode constatar-se que à exceção do desenvolvimento dos meios de pagamento Multibanco e cartão de crédito, todos os requisitos do projeto foram cumpridos com sucesso.

O futuro

A plataforma de *marketing* ZIPU é um projeto ao qual é sempre possível acrescentar novos módulos com novas funcionalidades complementares que podem melhorar a experiência dos utilizadores e, por esse motivo, nunca estará verdadeiramente concluída.

Nomeadamente, consideram-se como atualizações importantes a efetuar no futuro, as seguintes:

- Distribuição do envio de *emails* por diversos *IPs*: para evitar que os *emails* sejam considerados SPAM é importante que não sejam enviadas grandes quantidades dos mesmos a partir do mesmo IP. O desenvolvimento desta componente implica a compra de diversos *IPs* para que seja possível distribuir os envios das diversas campanhas de *email* pelos mesmos.
- Autenticação de um remetente: uma das causas que contribui para a eficácia de entrega de um *email* é o remetente (nome e endereço de onde provém esse *email*). Os *emails* enviados a partir da plataforma de *marketing* desenvolvida podem ser enviados com qualquer remetente, mas como o ZIPU faz o envio através do *email* do remetente inserido pelo utilizador na plataforma, os ISP (serviços de *email*) tendem a desconfiar. Por esse motivo, é imprescindível que os remetentes sejam autenticados. As principais formas de autenticação são denominadas por: *SPF*, *SenderID*, *DomainKeys* e *DKIM*.
- Desenvolvimento dos meios de pagamento através de cartões de débito/crédito e multibanco.

BIBLIOGRAFIA

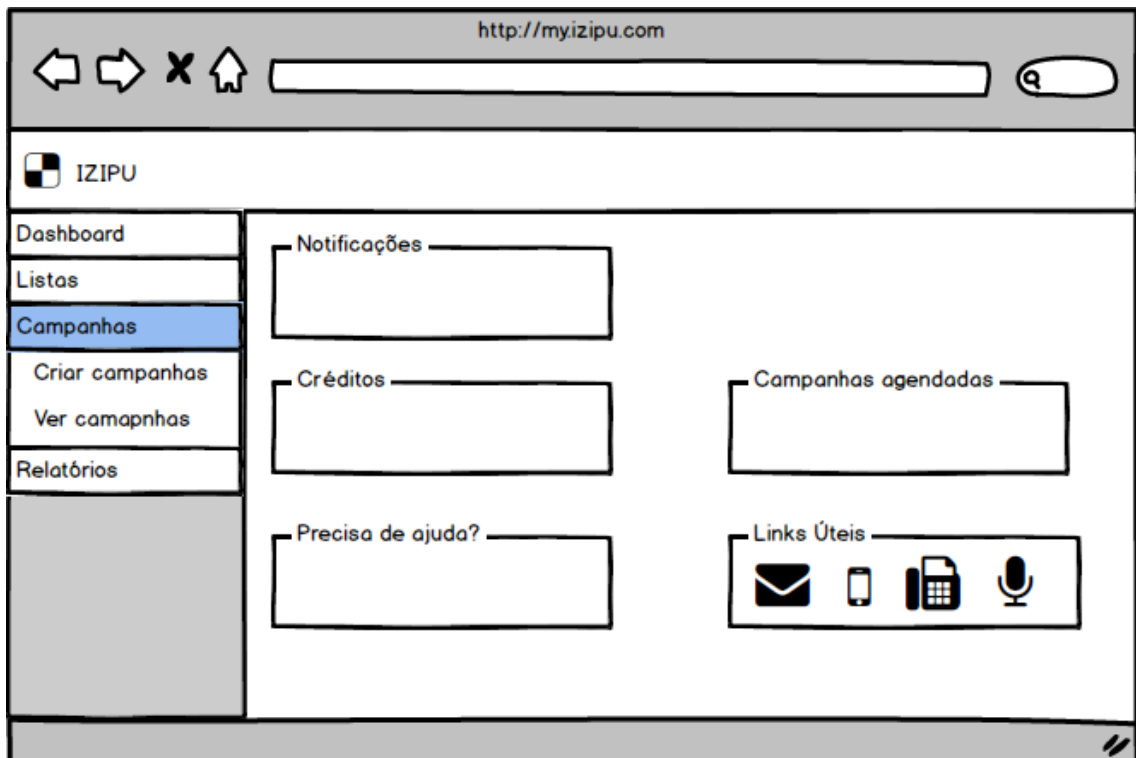
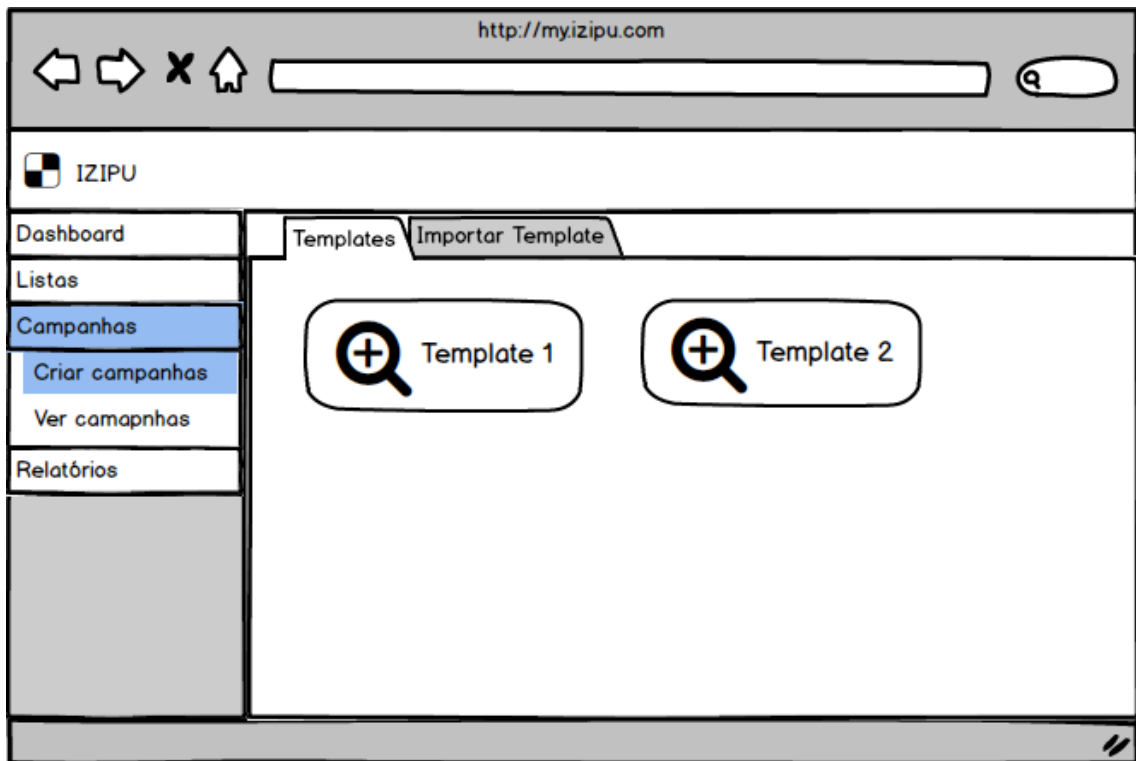
- [1] J. Lovato, "Then and Now: How Much Has Digital Marketing Changed?," 21 Novembro 2014. [Online]. Disponível em: <http://www.mediavisioninteractive.com/blog/digital-marketing-2/digital-marketing-evolution/>. [Acedido em 16 Dezembro 2014].
- [2] T. C. d. Oliveira, "A influência do marketing digital no sector da distribuição de produtos alimentares em Portugal," Viana do castelo, 2015.
- [3] K. M. Group, "Dr. Philip Kotler Answers Your Questions on Marketing," [Online]. Disponível em: http://www.kotlermarketing.com/phil_questions.shtml. [Acedido em 25 Julho 2015].
- [4] Tecmundo, "A história da internet," [Online]. Disponível em: <http://www.tecmundo.com.br/infografico/10054-a-historia-da-internet-a-decada-de-1990-infografico-.htm>. [Acedido em 25 Junho 2015].
- [5] C. Oliveira, "TCC - Marketing Digital: Um estudo exploratório sobre a utilização das mídias digitais como canal de comunicação," 10 Novembro 2011. [Online]. Disponível em: <http://pt.slideshare.net/carllacynthia/tcc-marketing-digital-um-estudo-exploratorio-sobre-a-utilizacao-das-mdias-digitais-como-canal-de-comunicacao>. [Acedido em 25 Junho 2015].
- [6] K. Auletta, "Googled - A história da maior empresa do mundo virtual," 2009. [Online]. Disponível em: <http://tinyurl.com/pahceq9>.
- [7] "Today I Found Out," [Online]. Disponível em: <http://www.todayifoundout.com/index.php/2012/04/this-day-in-history-the-first-mass-commercial-internet-spam-campaign-is-launched/>. [Acedido em 25 Junho 2015].
- [8] e-goi, "e-goi," [Online]. Disponível em: <http://www.e-goi.pt/pt>. [Acedido em 19 Novembro 2014].
- [9] G. Response, "Get Response," [Online]. Disponível em: <http://www.getresponse.pt/>. [Acedido em 18 Novembro 2014].
- [10] M. Chimp, "Mail Chimp," [Online]. Disponível em: <http://mailchimp.com/>. [Acedido em 18 Novembro 2014].
- [11] A. Communications, "Aweber," [Online]. Disponível em: <http://www.aweber.com/>. [Acedido em 18 Novembro 2014].

- [12] “Klick Mail,” [Online]. Disponível em: <https://www.klickmail.com.br/>. [Acedido em 18 Novembro 2014].
- [13] M. Rouse, “Google Trends,” Janeiro 2013. [Online]. Disponível em: <http://whatis.techtarget.com/definition/Google-Trends>. [Acedido em 16 Dezembro 2014].
- [14] T. S. project, “The Definition of Spam,” [Online]. Disponível em: <http://www.spamhaus.org/consumer/definition/>. [Acedido em 21 Janeiro 2015].
- [15] Outmarketing, 16 12 2014. [Online]. Disponível em: <http://www.outmarketing.pt/blog/como-evitar-a-pasta-de-spam-no-email-marketing/>.
- [16] A. Rule, “The importance and benefit of double opt-in *Email* Marketing,” 24 04 2015. [Online]. Disponível em: <https://www.elegantthemes.com/blog/resources/the-importance-and-benefit-of-double-opt-in-email-marketing>. [Acedido em 14 05 2015].
- [17] Y. Starak, “What Is A Double Opt-In *Email* List And Why Is It Important?,” [Online]. Disponível em: <http://www.entrepreneurs-journey.com/509/what-is-a-double-opt-in-email-list-and-why-is-it-important/>. [Acedido em 17 Maio 2015].
- [18] Kevin, “*Email* Opt Out? *Email* Opt In? Which Is Better for You?,” [Online]. Disponível em: <http://emailmarketing.comm100.com/email-marketing-tutorial/email-opt-out.aspx>. [Acedido em 17 05 2015].
- [19] “Balsamiq Mockups,” [Online]. Disponível em: <https://balsamiq.com/products/mockups/>. [Acedido em 23 Junho 2015].
- [20] “TIOBE Index for April 2015,” [Online]. Disponível em: <http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>. [Acedido em 2015 Abril 2015].
- [21] B. Skvorc, “Best PHP *Frameworks* for 2014,” [Online]. Disponível em: <http://www.sitepoint.com/best-php-frameworks-2014/>. [Acedido em 8 Abril 2015].
- [22] J. B. Nils Adermann, “Dependency Manager for PHP,” [Online]. Disponível em: <https://getcomposer.org/>. [Acedido em 09 Abril 2015].
- [23] “Laravel.IO,” [Online]. Disponível em: <http://laravel.io/forum>. [Acedido em 09 Abril 2015].
- [24] “Apache™ Subversion®,” [Online]. Disponível em: <https://subversion.apache.org/>. [Acedido em 23 Junho 2015].
- [25] Trello, Inc., “Trello,” [Online]. Disponível em: <https://trello.com/tour>.

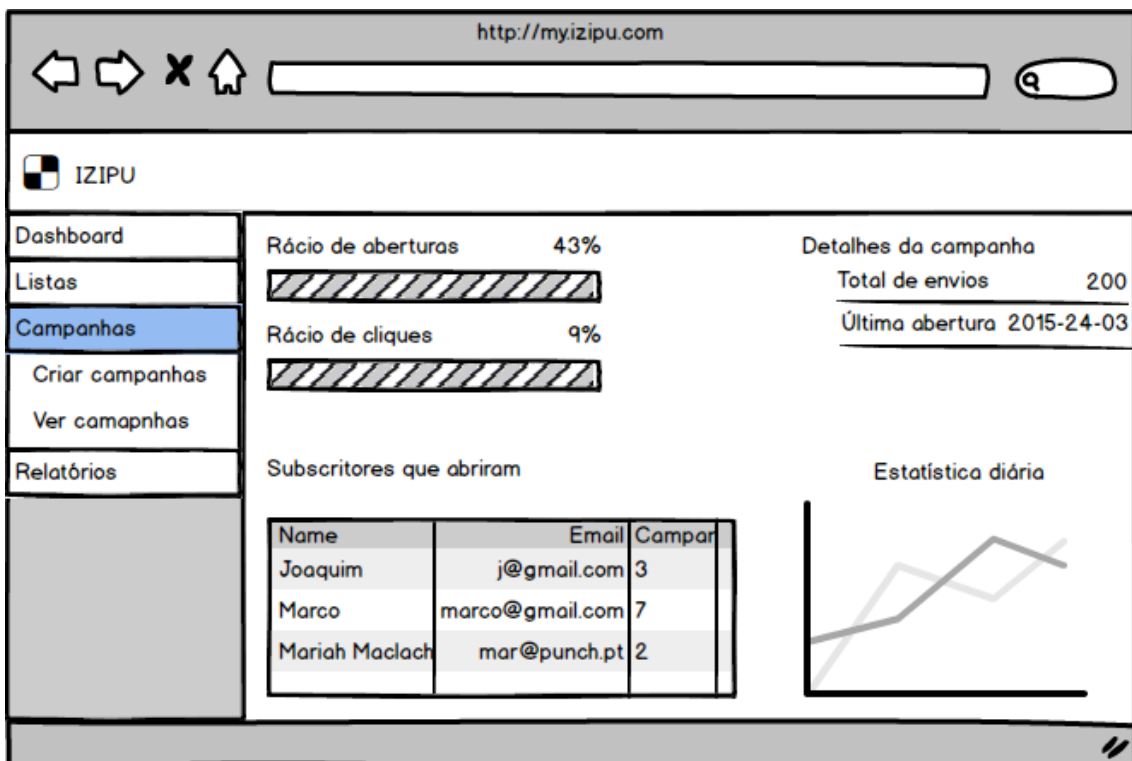
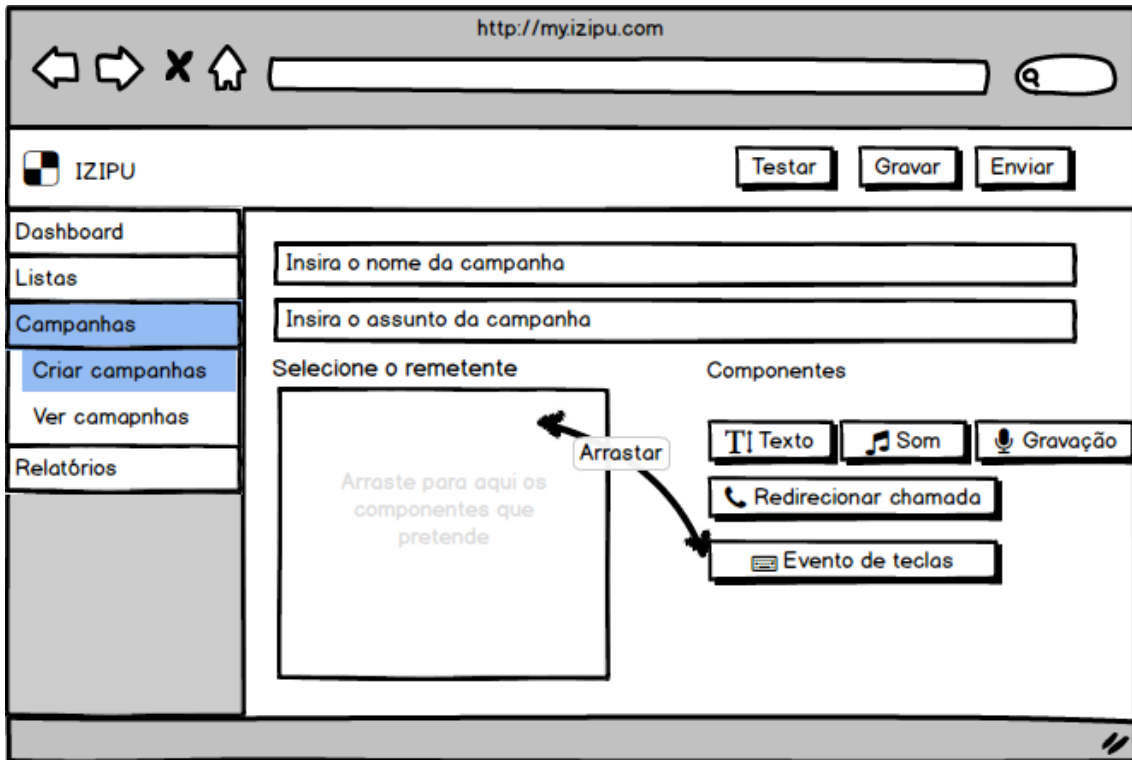
- [Acedido em 17 Junho 2015].
- [26] O. weboffice, “SMS wiki,” [Online]. Disponível em: www.sms-wiki.org/p_28-flash-sms.html. [Acedido em 12 Fevereiro 2015].
- [27] M. Rouse, “TechTarget,” [Online]. Disponível em: <http://searchcrm.techtarget.com/definition/Interactive-Voice-Response>. [Acedido em 23 Abril 2015].
- [28] “Welcome to PHPMailer,” [Online]. Disponível em: <http://phpmailer.worxware.com/>. [Acedido em 23 Junho 2015].
- [29] N. Downie, “Chart.js,” [Online]. Disponível em: <http://www.chartjs.org/>. [Acedido em 6 Maio 2015].
- [30] “DataTables Table plug-in for jQuery,” [Online]. Disponível em: <https://www.datatables.net/>. [Acedido em 2 Julho 2015].
- [31] “jQuery,” [Online]. Disponível em: <https://jquery.com/>. [Acedido em 26 Junho 2015].
- [32] “Bootstrap,” [Online]. Disponível em: <http://getbootstrap.com/>. [Acedido em 23 Junho 2015].
- [33] CacheFly, “tinymce,” [Online]. Disponível em: <http://www.tinymce.com/>. [Acedido em 13 Abril 2015].
- [34] Clickatell, “Clickatell,” [Online]. Disponível em: <https://www.clickatell.com/apis-scripts/apis/http-s/>. [Acedido em 18 06 2015].
- [35] Twilio, “twilio,” [Online]. Disponível em: <https://www.twilio.com/>. [Acedido em 20 Abril 2015].
- [36] Twilio, “Twilio Docs,” [Online]. Disponível em: <https://www.twilio.com/docs/api/twiml>. [Acedido em 20 Abril 2015].
- [37] PamFax, “PamFax,” [Online]. Disponível em: <https://www.pamfax.biz/en/developers/introduction/>. [Acedido em 20 Abril 2015].
- [38] “Paypal,” [Online]. Disponível em: <https://www.paypal.com/pt/webapps/mpp/about>. [Acedido em 20 Abril 2015].
- [39] “Express Checkout API Getting Started Guide,” PayPal, [Online]. Disponível em: https://developer.paypal.com/docs/classic/express-checkout/gs_expresscheckout/. [Acedido em 20 Abril 2015].
- [40] T. d. a. n. d. ágil. [Online]. Disponível em: <http://blog.myscrumhalf.com/2011/10/testes-de-aceitacao-no->

- desenvolvimento-agil/. [Acedido em 19 Junho 2015].
- [41] “Usability Testing,” [Online]. Disponível em: <http://www.usability.gov/how-to-and-tools/methods/usability-testing.html>. [Acedido em 25 Junho 2015].
- [42] I. Galambos, “Inchoo,” [Online]. Disponível em: <http://inchoo.net/magento/what-is-base64-encoding-and-how-can-we-benefit-from-it/>. [Acedido em 22 Janeiro 2015].
- [43] D. H. Yang, “What Is SHA1 Message Digest Algorithm?,” [Online]. Disponível em: <http://www.herongyang.com/Cryptography/SHA1-Message-Digest-What-Is-SHA1.html>. [Acedido em 21 Abril 2015].
- [44] T. L. I. Project, “BSD License Definition,” [Online]. Disponível em: <http://www.linfo.org/bsdlicense.html>. [Acedido em 21 Abril 2015].
- [45] O. Marketing, “Out Marketing,” [Online]. Disponível em: <http://www.outmarketing.pt/>. [Acedido em 16 06 2015].
- [46] LeanKit Inc, “What is Kanban?,” [Online]. Disponível em: <http://leankit.com/kanban/what-is-kanban/>. [Acedido em 17 Junho 2015].
- [47] P. Nunes, “Enciclopédia Temática,” 26 Outubro 2007. [Online]. Disponível em: <http://old.knoow.net/cienceconempr/gestao/kotlerphilip.htm>. [Acedido em 30 Agosto 2015].

APÊNDICE A – MOCKUPS







APÊNDICE B – MODELO FÍSICO

