



Instituto Superior de Engenharia de Lisboa

Engenharia de Software

Mestrado Engenharia Informática e Multimédia

Adchime

**Plataforma de envio de
mensagens publicitárias em massa**

Semestre de Inverno, 2022/2023

Nome: Gonçalo Fonseca | Número: A50185

Índice

1	Introdução	1
1.1	Âmbito	1
1.2	Definições, Abreviaturas e Acrónimos	1
2	Análise de requisitos	2
2.1	Visão	2
2.1.1	Descrição do problema	2
2.1.2	Descrição Geral da Solução	2
2.1.2.1	Resumo de Características	2
2.1.2.2	Dependências	4
2.1.2.3	Custo e Preço	4
2.1.3	Workflow	5
2.2	Especificação de Requisitos	5
2.2.1	Casos de Utilização	6
2.2.1.1	Modelo de Casos de Utilização relativa à aba CONTACTS	6
2.2.1.2	Modelo de Casos de Utilização relativa à aba LISTS	9
2.2.1.3	Modelo de Casos de Utilização relativa à aba TEMPLATES	10
2.2.1.4	Modelo de Casos de Utilização relativa à aba CAMPAIGNS	12
2.2.1.5	Modelo de Casos de Utilização relativa à aba ADMIN	14
2.2.2	Especificação Suplementar	16
3	Projecto de arquitectura da aplicação	17
3.1	Arquitetura lógica	17
3.1.1	Modelo de Domínio	18
3.1.2	Realização de Casos de Utilização	18
3.1.2.1	Criação de um contacto	19
3.1.2.2	Criação de um <i>template</i> de mensagem	20
3.1.2.3	Criação de uma Campanha	21
3.1.3	Arquitetura de mecanismos	22
3.1.3.1	Criação de um contacto	23
3.1.3.2	Criação de um <i>template</i> de mensagem	23
3.1.3.3	Criação de uma Campanha	24
3.1.4	Arquitetura geral da solução	25
3.2	Arquitetura detalhada	30
3.2.1	Modelo de dinâmica	31
3.2.2	Arquitetura do protótipo de teste	31
3.3	Arquitetura do protótipo aplicacional	33

4	Implementação do protótipo de teste	35
5	Implementação do protótipo aplicativo	36
5.1	Criação de um contacto	37
5.2	Criação de uma template	39
5.3	Criação de uma campanha	40
6	Análise reflexiva do projeto elaborado	44
7	Conclusão	45
	Referências	46

Índice de Figuras

1	Diagrama do Contexto de utilização	2
2	Maquete da página dos contactos com as várias abas na visão do gestor.	3
3	Maquete da página inicial antes do Login	4
4	Exemplo de preçário	4
5	Workflow do envio das mensagens em massa	5
6	Packages utilizados pelo utilizador e gestor	6
7	Modelo de casos de utilização relativo aos CONTACTS	7
8	Modelo de casos de utilização relativa à aba LISTS	9
9	Modelo de casos de utilização relativa à aba TEXTS	11
10	Modelo de casos de utilização relativa à aba CAMPAIGNS	13
11	Modelo de casos de utilização relativa à aba ADMIN	15
12	Modelo de Domínio	18
13	Diagrama de Sequência - Criação de Contacto	20
14	Diagrama de Sequência - Criação de Mensagem	21
15	Diagrama de Sequência - Criação de Campanha	22
16	Arquitetura de mecanismos para o caso de utilização de criação de um contacto	23
17	Arquitetura de mecanismos para o caso de utilização de criação de uma <i>template</i>	24
18	Arquitetura de mecanismos para o caso de utilização de criação de uma campanha	25
19	Arquitetura de subsistemas	26
20	Subsistema de apresentação - Geral	27
21	Subsistema de apresentação - ADMIN & HOME	27
22	Subsistema de apresentação - CAMPAIGNS & TEXTS	28
23	Subsistema de apresentação - CAMPAIGNS, CONTACTS & LISTS	28
24	Subsistema de domínio	29
25	Subsistema de acesso de dados	30
26	Modelo de Dinâmica	31
27	Arquitetura de teste dos casos de utilização nos testes	32
28	Sistema da aplicação de teste	33
29	Detalhes do sistema da aplicação	33
30	Detalhes do sistema da aplicação	34
31	Modelo de implantação de teste	35
32	Modelo de implantação	36
33	Página de Login	37
34	Página dos Contactos	38
35	Exemplo de criação de um contacto	38
36	Página dos Contactos com o novo contacto	39
37	Página dos Templates de Mensagens	39
38	Exemplo de criação de uma template de mensagem	40
39	Página dos Templates com o novo template de mensagem criado	40

40	Página das Campanhas	41
41	Seleção de uma template para ser usado na campanha	41
42	Página de criação de uma campanha	42
43	Página das Campanhas com a nova campanha criada	42
44	Contacts Controller	43

1 Introdução

Este projeto tem como objetivo aprender e aprimorar técnicas de desenvolvimento de software, de maneira a implementá-las seguindo boas práticas, para garantir que o projeto seja bem estruturado e organizado, facilitando a colaboração e a implementação sem criar complexidade, evitando assim erros e desperdício de tempo. Este também permite uma maior flexibilidade para adaptar serviços ou adicionar funcionalidades no futuro e manter o sistema operando corretamente.

A aplicação **AdChime** consistirá numa aplicação web que permite que utilizadores possam criar envios de mensagens texto (SMS) publicitárias em massa. A plataforma é criada para colmatar a necessidade de enviar mensagens 1 a 1, de forma rápida e eficiente, sendo que guarda a possibilidade de ter todos os contactos existentes numa plataforma só e guardar os registos de todos os envios. Por fim, tem-se por objetivo gerar estatísticas dos envios, para que se possa atingir o maior público possível.

1.1 Âmbito

Este relatório está inserido no âmbito da última entrega do projeto final da Unidade Curricular de Engenharia de Software, do Mestrado de Engenharia Informática e Multimédia do ISEL.

1.2 Definições, Abreviaturas e Acrónimos

API - Application Programming Interface (Interface de Programação de Aplicação)

UML - Unified Modeling Language

SMS - Short Message Service

MVC - Model-View-Controller

CRUD - do inglês Create, Read, Update and Delete

DAO - Data Access Object

SQL - Structured Query Language

EF - Entity Framework

SMS - Short Message Service

SMS - Short Message Service

2 Análise de requisitos

2.1 Visão

A análise de requisitos passa por identificar, analisar e compreender o problema em questão e especificar os requisitos de uma solução possível.

2.1.1 Descrição do problema

Uma empresa pretende utilizar uma plataforma de envio de mensagens publicitárias em massa que permita criar templates de mensagens, enviar mensagens de texto em massa a listas de contactos e consultar num dashboard o sucesso das mensagens publicitárias através da contabilização de cliques em links gerados. Para tal, pretende-se criar o produto **AdChime**, que permite fazer tais envios em massa, em vez de ser necessário o envio individual de mensagens publicitárias.

2.1.2 Descrição Geral da Solução

A plataforma funcionará em ambiente web.

Abaixo é apresentado o diagrama do contexto de utilização na figura 1.

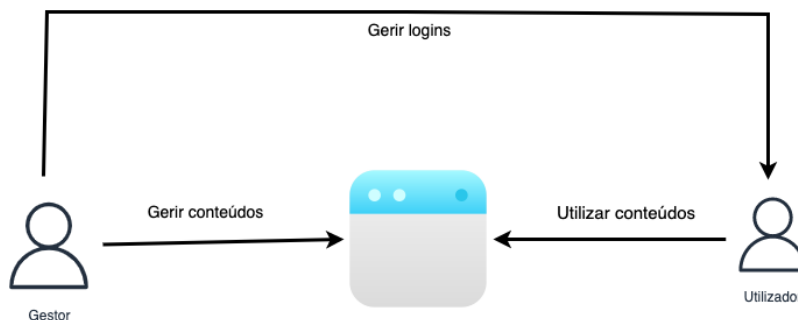


Figura 1: Diagrama do Contexto de utilização

Como mostra a figura, a plataforma difere os utilizadores em Gestor (que gere conteúdos e logins da plataforma) e os Utilizadores (que realizam ações com os conteúdos disponibilizados pelo gestor). É importante realçar que os gestores também podem realizar as mesmas ações que os utilizadores normais, ações essas que serão referidas mais à frente.

2.1.2.1 Resumo de Características

A Adchime - Plataforma de Envios de Mensagens Publicitárias em Massa, apresenta 6 abas principais para que os utilizadores possam realizar os envios das campanhas que pretendem e ver estatísticas sobre os mesmos, passando por alguns passos importantes.

- CONTACTS - página onde é possível criar, editar, visualizar, importar e apagar contactos;
- LISTS - página onde é possível criar, editar e visualizar listas aplicando filtros aos contactos;
- TEMPLATES - página onde é possível criar, editar e visualizar as templates a ser enviadas, sendo que apenas apenas os gestores podem colocar as templates aprovadas;
- CAMPAINGS - página onde é possível criar, editar e visualizar campanhas, que consistem na junção de listas de contactos com templates;
- DASHBOARD - página onde é possível visualizar informação relativa ao sucesso do envio das mensagens publicitárias em massa;
- ADMIN - página onde é possível fazer a gestão de logins de utilizadores e gerir os custos associados à plataforma, em particular a quantidade de mensagens disponíveis para serem enviadas. Apenas o gestor tem acesso a esta aba.

Na figura 3 que se segue, é possível observar um exemplo de um modelo da plataforma web com as abas acima referidas, usando a ferramenta Mockup [2].

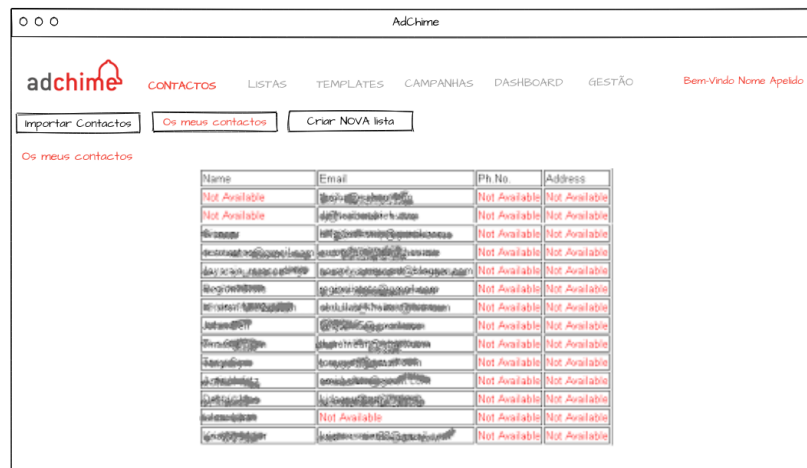


Figura 2: Maquete da página dos contactos com as várias abas na visão do gestor.

Para usar a plataforma, é necessário fazer um login na página inicial, tal como mostra um exemplo de uma página feita com a ferramenta acima supracitada. É com este login que depois se vai diferenciar os menus e as funções que irão ser permitidas tanto ao gestor com ao utilizador normal.

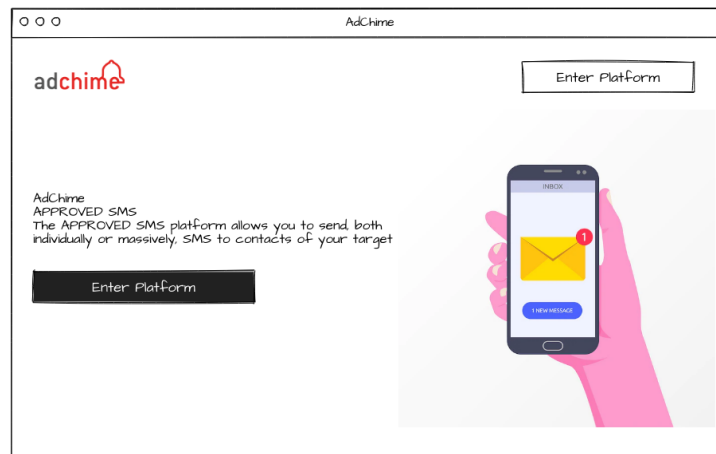


Figura 3: Maquete da página inicial antes do Login

2.1.2.2 Dependências

A plataforma está depende do uso de APIs, tanto para processamento e envio das mensagens, como para a criação dos links para contabilização de cliques.

2.1.2.3 Custo e Preço

A plataforma é disponibilizada de forma gratuita, mas sujeita a pagamento avulso. Ex.: pack de 1000 mensagens por 40 euros como mostra a figura 4.

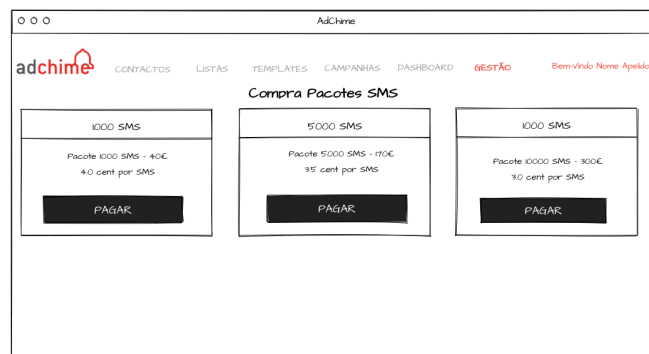


Figura 4: Exemplo de preço

O preço é calculado por SMS, sendo que, para além dos custos da API de envios de mensagem, é necessário ter em conta um certo valor para a manutenção da

plataforma e outros custos associados.

2.1.3 Workflow

A figura 5 representa o workflow geral do projeto para a criação e envio das mensagens em massa.

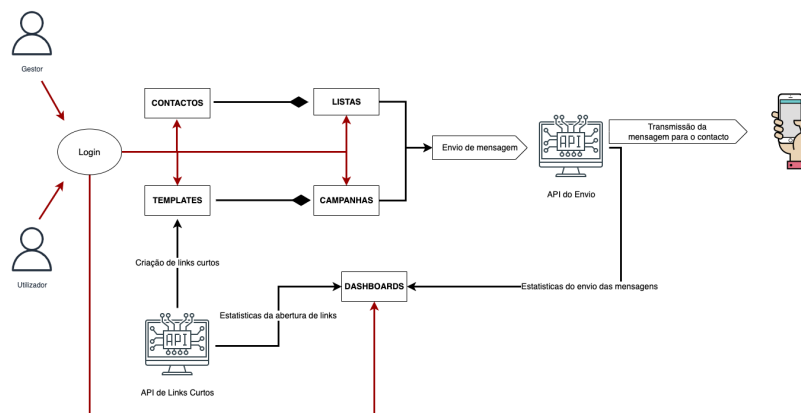


Figura 5: Workflow do envio das mensagens em massa

2.2 Especificação de Requisitos

Neste ponto serão referenciados casos de utilização, na maioria agregados em *packages*, como mostra a figura 6 que se segue. A mesma apenas representa 4 dos 5 *packages* existentes e ainda um caso de utilização que não está ligado a nenhum *package*.

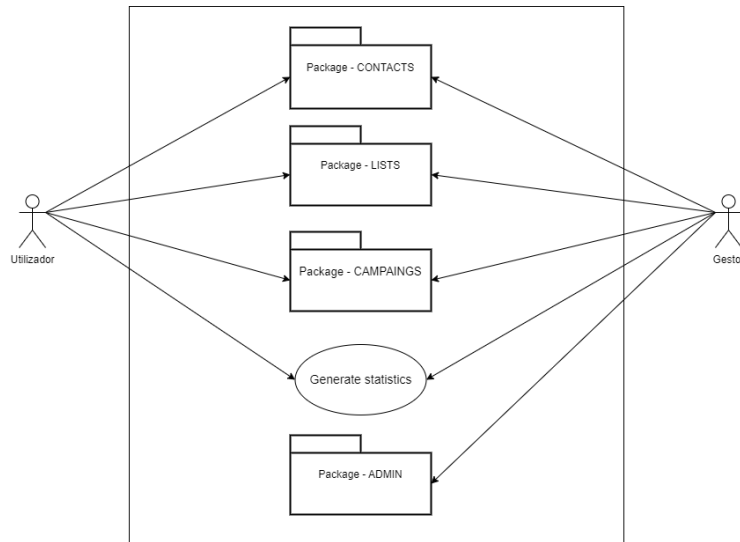


Figura 6: Packages utilizados pelo utilizador e gestor

É importante notar que apenas o package **ADMIN** é usado pelo Gestor e todos os outros são usados por ambos.

2.2.1 Casos de Utilização

Para apresentar alguns exemplos dos fluxos de eventos na aplicação, fluxos esses relacionados a cada um dos três atores supracitados, são usados os casos de utilização que se seguem. Em cada caso, procurar-se-á demonstrar algumas alternativas e desvios do fluxo principal.

2.2.1.1 Modelo de Casos de Utilização relativa à aba CONTACTS

A aba CONTACTS vai permitir que os utilizadores da aplicação tenham todos os contactos num local só, sendo que esta permite que o utilizador adicione contactos tanto de forma manual como através de um ficheiro), veja todas as informações do mesmo, edite essa informação e ainda dá a possibilidade de remover um contacto do conjunto de contactos existentes na plataforma. A figura 7 apresenta, no geral, os casos de utilização nos CONTACTS.

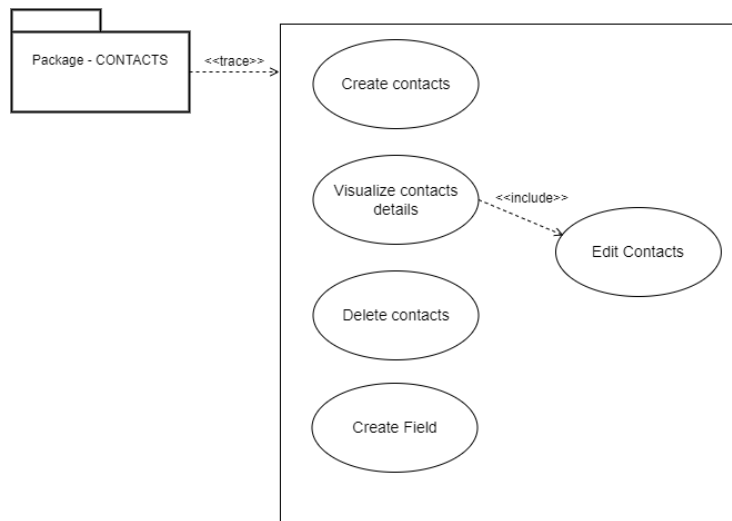


Figura 7: Modelo de casos de utilização relativo aos CONTACTS

A seguir são apresentados os casos de utilização com um resumo e outras informações necessárias para se compreender cada caso.

Caso de Utilização - Carregar Contactos

Resumo: Este caso de utilização permite ao utilizador carregar contactos dentro da plataforma, na aba **CONTACTOS**, usando ficheiro de excel ou inserindo manualmente.

Atores: Todos os utilizadores

Cenário principal:

1. O utilizador abre a página de carregamento no botão IMPORTAR CONTACTOS;
2. Escolhe o ficheiro excel que quer carregar (requisito 1);
3. Liga as colunas do excel às colunas existentes;
4. Clica para proceder ao carregamento.

Cenário alternativo:

1. O utilizador abre uma página para preencher os dados manualmente (requisito 2);
2. Clica para proceder ao carregamento.

Caso de Utilização - Editar Contactos

Resumo: Este caso de utilização permite ao utilizador editar contactos dentro da plataforma, na aba **CONTACTOS**.

Atores: Todos os utilizadores

Cenário principal:

1. Clica no contacto que pretende editar;
2. Altera-se os campos que se pretende atualizar (requisito 2);
3. Clica-se no botão de CONFIRMAR para proceder à atualização.

Cenário alternativo: Através da importação de contactos (ponto 3.1.1), também é possível fazer a atualização dos contactos dentro desse ficheiro, caso existam.

Caso de Utilização - Apagar Contactos

Resumo: Este caso de utilização permite ao utilizador apagar um contacto único, na aba **CONTACTOS**.

Atores: Todos os utilizadores

Cenário principal:

1. O utilizador clica no botão com o símbolo do caixote do lixo assinalado a cor vermelha;
2. O contacto é removido.

Caso de Utilização - Criar campo dos contactos

Resumo: Este caso de utilização permite ao utilizador, dentro da plataforma, na aba **CONTACTOS**, criar um novo campo que ainda não exista.

Atores: Todos os utilizadores

Cenário principal:

1. O utilizador clica no botão ADICIONAR CAMPO;
2. Preenche o nome do campo e o formato (requisito 3);
3. Clica em ADICIONAR para o campo ser adicionado.

Cenário alternativo: Através do carregamento de contactos (ponto 3.1.1), é também possível adicionar um novo campo.

2.2.1.2 Modelo de Casos de Utilização relativa à aba LISTS

A aba LISTS vai permitir os utilizadores da aplicação a criação de listas de contactos, para ao enviar nas campanhas, basta inserir o nome da lista que esta agrega todos os contactos que nela contém.

A figura 8 apresenta no geral os casos de utilização da aba LISTS.

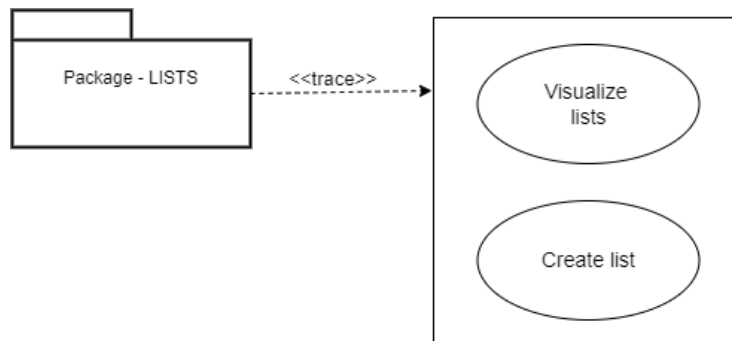


Figura 8: Modelo de casos de utilização relativa à aba LISTS

Posteriormente são apresentados os casos de utilização com um resumo e outras informações necessárias para se compreender cada caso.

Caso de Utilização - Visualizar listas de contactos

Resumo: Este caso de utilização permite ao utilizador ver as listas de contactos existentes, podendo ainda ver o conteúdo de cada uma, ou seja, ver os números que estão na lista selecionada.

Atores: Todos os utilizadores

Cenário principal:

1. O utilizador abre a aba **LISTS**, aparecendo todas as listas de contactos existentes pelo seu nome;
2. Para ver o conteúdo, clica-se no nome de uma lista e é aberta uma janela do lado direito com uma tabela com os contactos dessa lista.

Cenário alternativo: O utilizador pode fazer o mesmo procedimento através da aba **CONTACTOS**.

Caso de Utilização - Criar lista de contactos

Resumo: Este caso de utilização permite ao utilizador criar lista de contactos, filtrando os contactos por um certo campo, para ser usado naquele momento. Caso haja um novo contacto adicionado com aquele campo, o mesmo não será incluído na lista, apenas se for atualizado ou criada uma nova lista.

Atores: Todos os utilizadores

Cenário principal:

1. O utilizador abre a aba **LISTAS**, selecionando o botão de “NOVA LISTA”;
2. Preenche os campos existentes (requisito 4);
3. Clica para proceder ao carregamento.

Cenário alternativo: O utilizador pode fazer o mesmo procedimento através da aba **CONTACTOS**.

2.2.1.3 Modelo de Casos de Utilização relativa à aba TEMPLATES

A aba **TEXTS** vai permitir os utilizadores da aplicação que criem mensagens pré-definidas, que depois serão ligadas às campanhas a serem criadas.

A figura 9 apresenta no geral os casos de utilização da aba **TEXTS**.

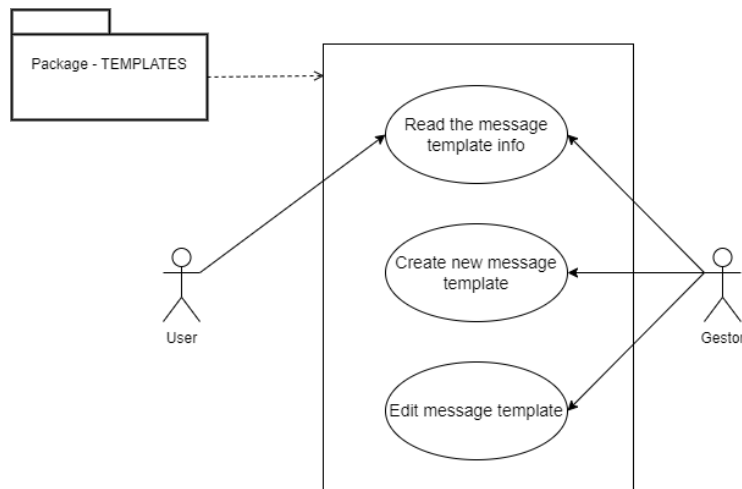


Figura 9: Modelo de casos de utilização relativa à aba TEXTS

Abaixo são apresentados os casos de utilização com um resumo e outras informações necessárias para se compreender cada caso.

Caso de Utilização - Ler mensagens criadas

Resumo: Este caso de utilização permite ao utilizador ler as mensagens criadas pelo gestor (Caso de Utilização 3.2.1).

Atores: Todos os utilizadores

Cenário principal:

1. Na aba dos TEXTS aparecem vários templates;
2. Selecionar um template e uma página abre com os campos preenchidos (Caso de Utilização 3.2.1 e requisito 8). Caso o utilizador seja o gestor, tem acesso ao Caso de Utilização 3.2.2, onde pode alterar os campos. Se não, apenas consegue visualizar sem alterar.

Caso de Utilização - Carregar mensagens

Resumo: Este caso de utilização permite ao gestor criar novas mensagens para serem usadas na plataforma, na aba **TEXTS**.

Atores: Gestor

Cenário principal:

1. Na aba dos TEXTS, seleccionar uma template;
2. De seguida, preenche-se os campos obrigatórios (requisito 8);
3. Submeter os campos preenchidos.

Caso de Utilização - Editar mensagens

Resumo: Este caso de utilização permite ao gestor editar mensagens para serem usadas na plataforma, na aba **TEXTS**.

Atores: Gestor

Cenário principal:

1. Na aba dos TEMPLATES, selecciona uma template;
2. De seguida, pode alterar os campos que tiver necessidade (requisito 8);
3. Submeter os campos preenchidos.

2.2.1.4 Modelo de Casos de Utilização relativa à aba CAMPAIGNS

A aba CAMPAIGNS vai permitir os utilizadores da aplicação criar campanhas para serem enviadas para listas de contactos, sendo ainda possível enviar várias vezes essa mesma campanha, podendo ver os detalhes desses mesmos envios.

A figura 9 apresenta numa forma geral os casos de utilização da aba CAMPAIGNS.

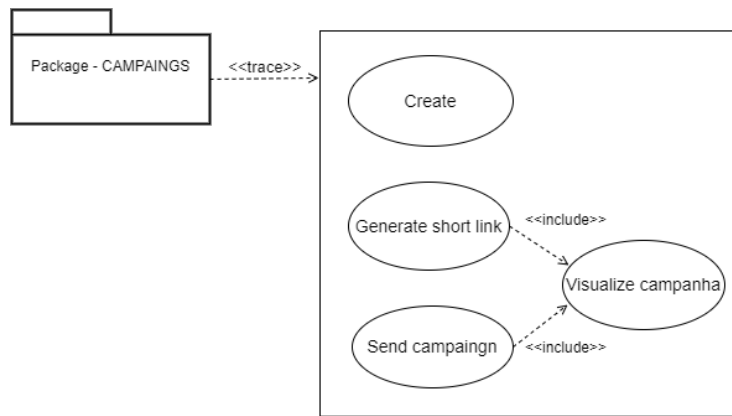


Figura 10: Modelo de casos de utilização relativa à aba CAMPAIGNS

Abaixo são apresentados os casos de utilização com um resumo e outras informações necessárias para se compreender cada caso.

Caso de Utilização - Criar campanha

Resumo: Este caso de utilização permite ao utilizador criar uma campanha de envio de mensagem, dentro da aba **CAMPAIGNS**.

Atores: Todos os utilizadores

Cenário principal:

1. O utilizador clica em CRIAR CAMPANHA;
2. Preenche os campos existentes (requisito 5);
3. Submete para criar a campanha.

Cenário alternativo em modo gestor: Caso o utilizador seja o gestor, no campo de adicionar uma mensagem, pode ainda alterar a mesma.

Caso de Utilização - Enviar campanha

Resumo: Este caso de utilização permite ao utilizador enviar uma campanha de envio de mensagem, dentro da aba **CAMPAIGNS**.

Atores: Todos os utilizadores

Cenário principal:

1. O utilizador clica em ENVIAR na campanha que pretende;
2. Preenche os campos existentes (requisito 10);
3. Submete para criar a campanha.

Cenário alternativo: O utilizador pode preencher um outro campo “Número de teste” para enviar um teste para si.

Cenário alternativo 2: O utilizador pode seleccionar a lista de contactos (requisito 10), mas caso não selecione uma data e hora, o envio é feito no momento.

Caso de Utilização - Criar links curtos

Resumo: Este caso de utilização permite ao utilizador criar links curtos para enviar na mensagem, para ter um maior número de caracteres disponíveis na mensagem (considerando que 1 mensagem equivale a 160 caracteres).

Atores: Todos os utilizadores

Cenário principal:

1. Na página do envio de campanha (caso de utilização 3.1.7), clicar no botão de ADICIONAR LINK CURTO;
2. O utilizador preenche o campo necessário (requisito 6);
3. Depois de concluído, no local do botão será colocado o link gerado.

Cenário alternativo: Na mesma aba, em vez de criar um gerar um link, pode apenas escolher um link já existente.

2.2.1.5 Modelo de Casos de Utilização relativa à aba ADMIN

A aba ADMIN vai permitir os gestores fazer a gestão de logins da equipa e fazer o carregamento de mensagens que serão usadas na plataforma.

A figura 9 apresenta no geral os casos de utilização da aba ADMIN.

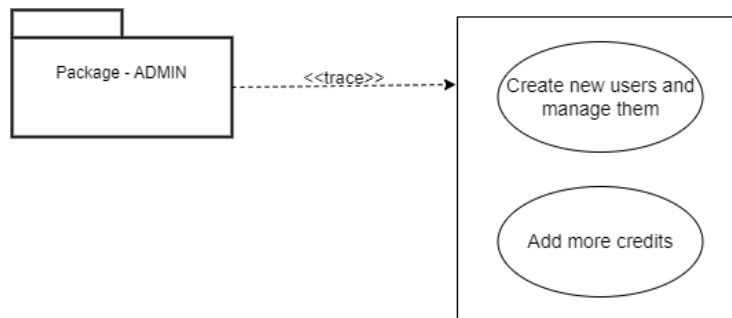


Figura 11: Modelo de casos de utilização relativa à aba ADMIN

Abaixo são apresentados os casos de utilização com um resumo e outras informações necessárias para se compreender cada caso.

Caso de Utilização - Gerir acessos de logins

Resumo: Este caso de utilização permite ao gestor criar contas de utilizadores e aprovar as contas depois do utilizador entrar pela primeira vez. Pode também revogar o acesso.

Atores: Gestor

Cenário principal:

1. Na aba **ADMIN**, abre-se a sub-aba **Manage Logins**;
2. Para criar uma conta, clica-se no botão para tal onde se preenche os campos necessários (requisito 9);
3. Para aprovar ou revogar os acessos, basta clicar nos botões **APPROVE** ou **REVOKE**, respetivamente.

Caso de Utilização - Carregamento de saldo de mensagens

Resumo: Este caso de utilização permite ao gestor carregar mensagens para limitar o número de envio de mensagens, visto estas serem pagas.

Atores: Gestor

Cenário principal:

1. Na aba **ADMIN**, abre-se a sub-aba **Add SMS Balance**;
2. O gestor seleciona o pacote que pretende comprar;
3. Depois do pagamento, as mensagens são carregadas.

2.2.2 Especificação Suplementar

Neste ponto serão apresentados todas as restrições e campos necessários para produzir os casos de utilização.

Os requisitos da tabela 1 estão identificados por uma referência, uma categoria e a descrição da referência. As referências são importantes para que os requisitos sejam identificáveis no texto e as categorias indicam se um requisito é obrigatório ou não.

Tabela 1: Tabela dos requisitos

Ref.	Descrição	Categoria
R1	O utilizador pode inserir ficheiros xlsx, xls, csv e txt.	Obrigatório
R2	Um contacto contém os seguintes campos:	Obrigatório
R2.1	Nome: texto (string)	Obrigatório
R2.2	Código do País: string (texto) com o código do país (p.e. +351 de Portugal)	Obrigatório
R2.3	Número de telefone: int (inteiro)	Obrigatório
R2.4	País: string	Obrigatório
R2.5	Outros: campos extras criados fora dos campos obrigatórios que podem variar o tipo	Opcional
R3	Um novo campo de contactos contém os seguintes campos:	Obrigatório
R3.1	Nome do campo: string	Obrigatório
R3.2	Formato: lista de formatos como por exemplo: string (texto), int (número inteiro), date (data)	Obrigatório
R4	Uma nova lista de contactos contém os seguintes campos:	Obrigatório
R4.1	Nome para a lista: string	Obrigatório
R4.2	Filtro	Obrigatório
R4.2.1	Lista com os campos existentes dos contactos	Obrigatório
R4.2.2	Lista com “é igual” ou “não é igual”	Obrigatório
R4.2.3	Campo que tanto pode ser texto ou uma lista com certos itens, dependendo do requisito R4.2.1	Obrigatório
R5	Uma campanha contém os seguintes campos:	Obrigatório
R5.1	Nome para a campanha: string	Obrigatório
R5.2	Descrição da campanha: string	Obrigatório
R5.3	ID do envio: string - nome que aparece na mensagem de quem recebe	Obrigatório

R5.4	Mensagem: botão para escolher a mensagem a ser enviada	Obrigatório
R6	Uma criação de link curto que apenas contém 1 campo a ser introduzido:	Obrigatório
R6.1	Link: string	Obrigatório
R7	Para gerar estatísticas, é possível introduzir os seguintes campos:	Obrigatório
R7.1	Campanha: lista de campanhas	Obrigatório
R7.2	Data Início: datetime - data e hora inicial para ver as estatísticas	Obrigatório
R7.3	Data Fim: datetime - data e hora final para ver as estatísticas	Obrigatório
R8	Para criar ou editar uma template, existem os seguintes campos:	Obrigatório
R8.1	Nome do template: string	Obrigatório
R8.2	Template aprovado: checkbox - caso esteja selecionado, os templates aparecem aos utilizadores. Caso não esteja, o template não aparecerá aos utilizadores	Obrigatório
R8.3	Mensagem: string	Obrigatório
R9	Para criar uma conta de um utilizador, são necessário preencher os seguintes campos:	Obrigatório
R9.1	Nome: string	Obrigatório
R9.2	Cargo: lista - pode ser um utilizador ou manager	Obrigatório
R9.2	Email: email	Obrigatório
R10	Para enviar uma campanha, é necessário:	Obrigatório
R10.1	Lista de contactos: list - listas já criadas	Obrigatório
R10.2	Data e hora: datetime - data e hora para fazer o agendamento do envio	Opcional
R10.2	Link: botão ADICIONAR LINK CURTO (caso de utilização 3.1.8)	Opcional

3 Projecto de arquitectura da aplicação

3.1 Arquitectura lógica

Atendendo aos requisitos, modelos de casos de utilização e especificação suplementar definidos no capítulo precedente, o próximo passo é processar toda essa informação utilizando modelos genéricos de representação dos mecanismos, es-

truturas e entidades envolvidas no sistema, de um ponto de vista relativamente abrangente.

3.1.1 Modelo de Domínio

O modelo de domínio na figura 12 ilustra as interações entre as várias classes envolvidas no sistema, bem como alguns dos seus atributos.

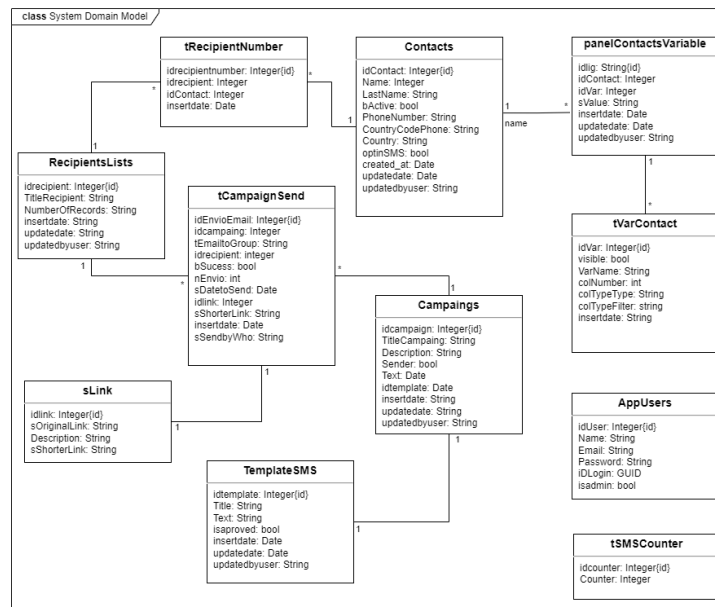


Figura 12: Modelo de Domínio

O modelo de domínio fornece uma visão geral e abrangente da estrutura, entidades e relações entre entidades.

3.1.2 Realização de Casos de Utilização

Para a realização de casos de utilização foi utilizado o padrão MVC (*Model - View - Controller*), pois este ajuda a separar as diferentes preocupações de uma aplicação. O modelo representa os dados, a visão representa a interface do utilizador e o controlador conecta as duas partes. Esta separação facilita a modificação e manutenção das diferentes partes da aplicação de forma independente. Este modelo ajuda também na manutenção fácil de código, promove a reutilização do mesmo e facilita o teste e a escala de uma aplicação.

Para exemplificar alguns casos de utilização, utilizar-se-ão digramas de interação que representarão as operações realizadas pelas diferentes camadas do sistema (de forma causal e temporal) e escolher-se-ão casos de utilização com alguma

relevância para o uso da aplicação. Inicialmente foi pensado em escolher outro caso de uso, também CRUD mas um pouco mais complexo. No entanto, devido ao tempo de desenvolvimento, não foi possível concluir a tempo a implementação e, por isso, foi decidido não incluir o mesmo.

Também é importante referir que não foi implementado nenhum DAO visto que se usou a Entity Framework (será explicada a razão para tal mais à frente). Ou seja, nas figuras dos pontos a seguir, assume-se a ligação à Entity Framework em cada um dos repositórios utilizados.

3.1.2.1 Criação de um contacto

Este caso de utilização serve para, tal como o nome indica, criar um contacto. Para tal, na view *NewContact* são apresentados alguns parâmetros, para que o utilizador da aplicação possa inserir de modo a criar um novo contacto, como “Name”, “Last Name”, entre outros, que são apresentados na entidade *Contacts* no modelo de domínio da figura 12.

Depois de preenchido e clicado no botão “Create”, os dados são enviados do formulário da View para o controlador *ContactsController*, mais especificamente através de *HttpPost*.

De seguida, os dados inseridos neste formulário são enviados para o serviço *ContactsService* através do método *AddContact*, onde são colocados os vários parâmetros. Foi, no entanto, colocado na figura 13 abaixo o parâmetro *fields* dentro do método, fazendo este referência aos vários parâmetros a serem enviados para diminuir a complexidade do diagrama.

Aqui, instanciamos um objeto *Contacts* com os dados vindos do controlador e, depois de estar instanciada, é chamado o método *Add* do repositório *ContactsRepository*, onde é passado como parâmetro a instância criada anteriormente. Como referido no ponto anterior, o repositório faz a ligação ao DAO, através da ferramenta *Entity Framework* referida no final do ponto 3.2, onde estes dados são inseridos na Base de Dados.

Por fim, essa instância é retornada para cada um dos passos anteriores até que, no controlador, este retorna a view *MyContacts*, atualizando a página web para essa mesma view contendo já o contacto criado.

Os vários passos referidos são apresentados no Diagrama de Sequência na figura 13.

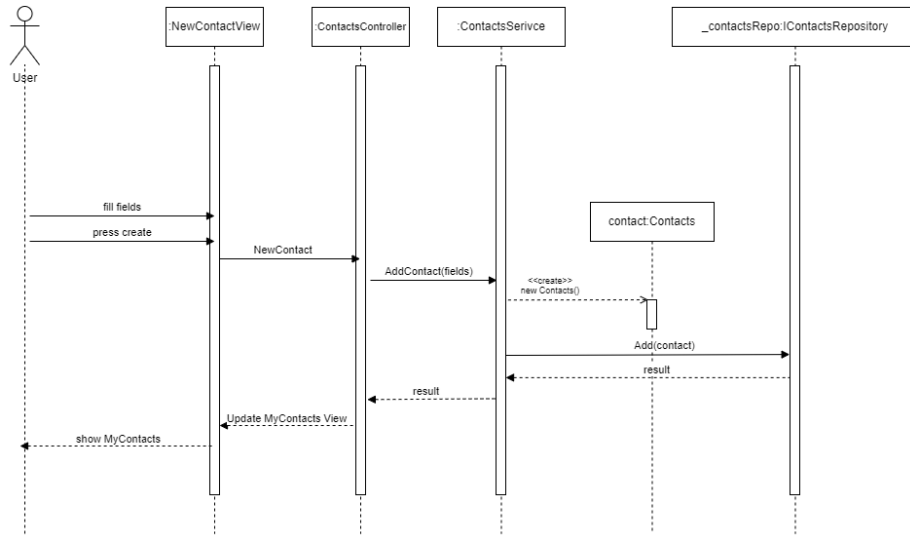


Figura 13: Diagrama de Sequência - Criação de Contacto

Neste diagrama deveria também ser apresentado a ligação aos serviços *Contacts-VariablesService* e *VarContactsService* pois o objetivo final deste caso de utilização é permitir ao utilizador inserir dados também nas variáveis secundárias, mas tal implicaria mais tempo para implementação (que não foi realizada). Então, decidiu-se manter o projeto mais simplificado, mantendo sempre a ideia de mudança para o futuro.

3.1.2.2 Criação de um *template* de mensagem

Este caso de utilização serve para, tal como o nome indica, criar um template de mensagem que depois é utilizado na criação de uma campanha, no ponto seguinte. Para tal, na view *NewText* são apresentados dois parâmetros para que o utilizador da aplicação possa inserir para criar uma template de mensagem. Os parâmetros são “Template Name” que serve de id/referência para ser utilizado mais tarde, e “SMS Text” onde o utilizador insere o texto que quer.

Depois de preenchido e clicado no botão “Submit”, os dados são enviados do formulário da View para o controlador *TextsController*, também através de Http-Post.

De seguida, os dados inseridos neste formulário são enviados para o serviço *TemplateSMSService* através do método *AddTemplate*, onde são colocados os vários parâmetros. Foi, no entanto, também colocado na figura 14 abaixo o parâmetro *fields* dentro do método, fazendo este referência aos vários parâmetros a serem enviados, para diminuir a complexidade do diagrama.

Aqui, instanciamos um objeto *TemplateSMS* com os dados vindos do controlador e depois de estar instanciada é chamado o método *Add* do repositório *TemplateSMSRepository*, onde é passado como parâmetro a instância criada anteriormente.

Por fim, essa instância é retornada para cada um dos passos anteriores até que no controlador, este retorna a view *MyTexts*, atualizando a página web para essa mesma view contento já a template criada.

Os vários passos referidos são apresentados no Diagrama de Sequência na figura 14.

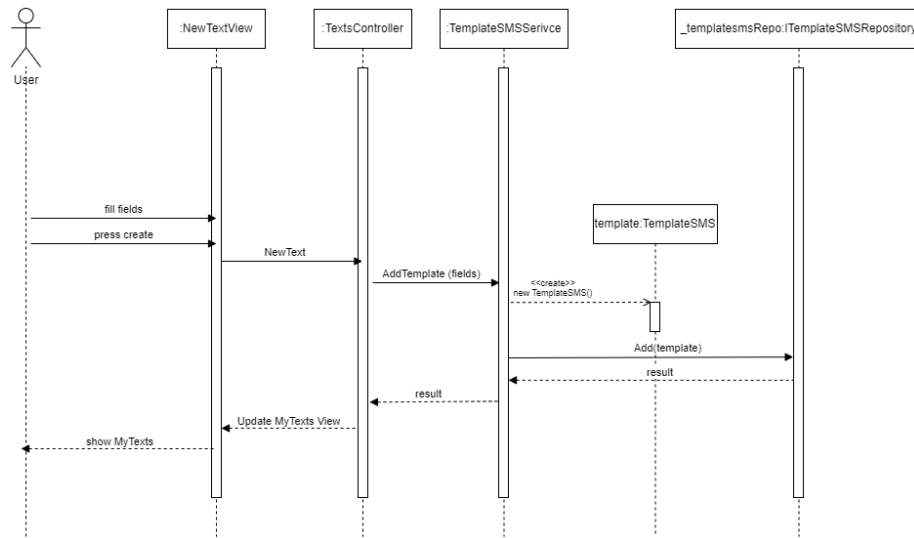


Figura 14: Diagrama de Sequência - Criação de Mensagem

3.1.2.3 Criação de uma Campanha

Este caso de utilização serve para, tal como o nome indica, criar uma campanha para envio da mensagem para um grupo de contactos. Para tal, no processo de abrir a view *NewCampaign*, é necessário obter todas as mensagens existentes. Para isso, usa-se o mesmo procedimento referido no ponto 3.2.3 acima. No entanto, é chamado o método *ChooseSMSText* do *TextsController*, que chama o método *GetAllTemplates* no serviço *TemplateSMSService*, que chama o método *GetAll* no repositório *TemplateSMSRepository*.

Por fim, o resultado deste último é passado para a view que foi aberta. Aqui, são apresentados alguns parâmetros para que o utilizador da aplicação possa criar uma nova campanha, como por exemplo “Campaign Name”, “Description of Campaign”, entre outros que são apresentados na entidade *Campaigns* no

modelo de domínio da figura 12. É apresentada ainda uma caixa para escolher a mensagem da lista de mensagens anteriormente referida, para que esta possa ser adicionada à campanha.

Depois de preenchido e clicado no botão “Submit”, os dados são enviados do formulário da View para o controlador *CampaignsController*, também através de *HttpPost*.

De seguida, os dados inseridos neste formulário são enviados para o serviço *CampaignsService*, através do método *AddCampaign*, onde são colocados os vários parâmetros. Foi, no entanto, também colocado na figura 15 o parâmetro *fields* dentro do método, fazendo este referência aos vários parâmetros a serem enviados para diminuir a complexidade do diagrama.

Aqui, instanciamos um objeto *Campaigns* com os dados vindos do controlador e depois de estar instanciada é chamado o método *Add* do repositório *CampaignsRepository*, onde é passado como parâmetro a instância criada anteriormente.

Por fim, essa instância é retornada para cada um dos passos anteriores, até que, no controlador, este retorna a view *Campaigns*, atualizando a página web para essa mesma view contendo a campanha criada.

Os vários passos referidos são apresentados no Diagrama de Sequência na figura 15.

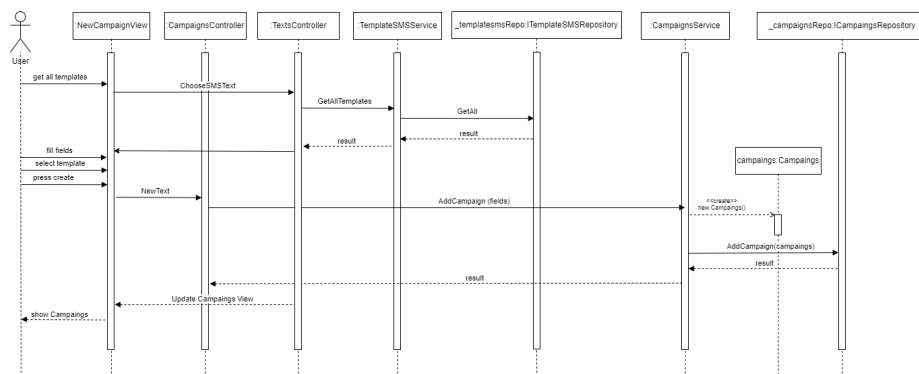


Figura 15: Diagrama de Sequência - Criação de Campanha

3.1.3 Arquitetura de mecanismos

Esta fase serve para obter um esboço dos objetos e entidades que são usados de acordo com o modelo de interação feito no ponto anterior e dos mecanismos entre todas as camadas.

3.1.3.1 Criação de um contacto

A figura 16 ilustra o diagrama de classes do mecanismo deste caso de utilização. A “*ContactData*” contém os parâmetros gerais pedidos na view *NewContact* como referido no ponto 3.2.1.

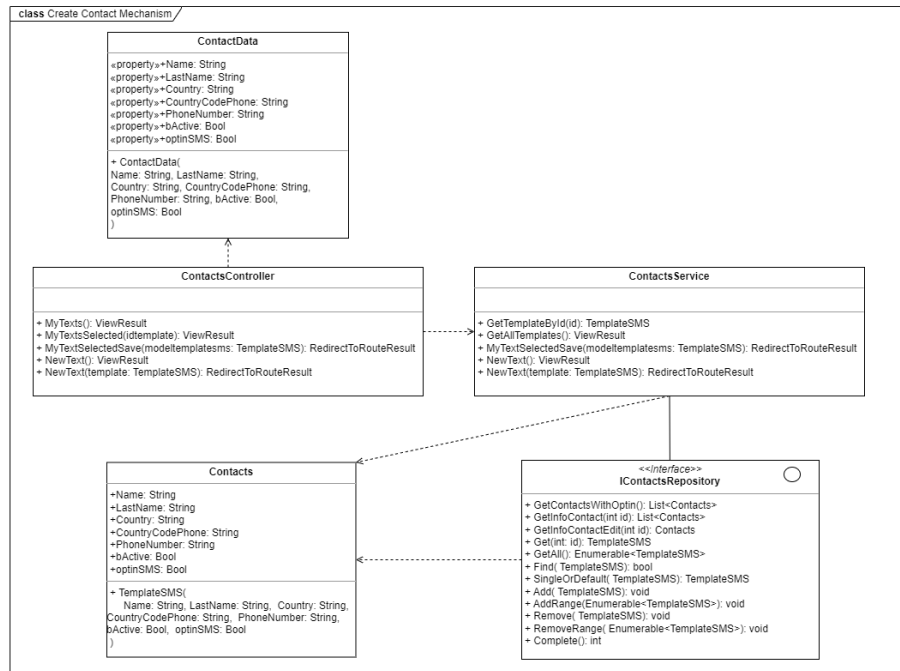


Figura 16: Arquitetura de mecanismos para o caso de utilização de criação de um contacto

3.1.3.2 Criação de um *template* de mensagem

Fez-se o mesmo para este caso de utilização e o mecanismo deste caso de utilização está apresentado na figura 17.

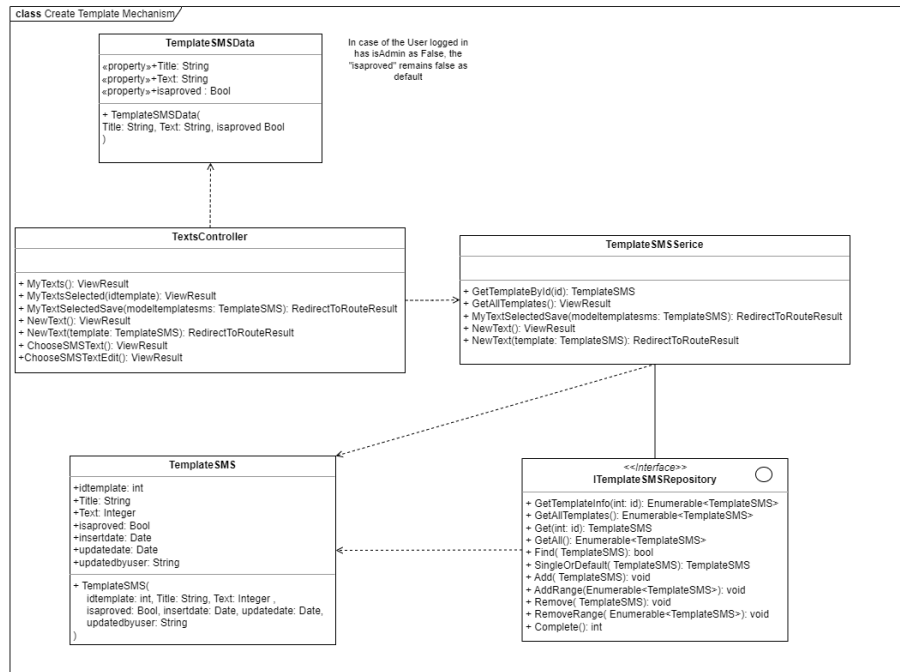


Figura 17: Arquitetura de mecanismos para o caso de utilização de criação de uma *template*

Tal como referido no diagrama acima, se o utilizador autenticado for um gestor, o parâmetro “*isaproved*” pode ser alterado. No entanto, caso o utilizador não seja um gestor, este parâmetro é bloqueado sem ser possível a sua alteração, mantendo o valor de “false”. Tal função é feita pelo controlador quando inicia a view *NewTemplate*.

3.1.3.3 Criação de uma Campanha

Por fim, a figura 18 ilustra o diagrama de classes do mecanismo deste caso de utilização.

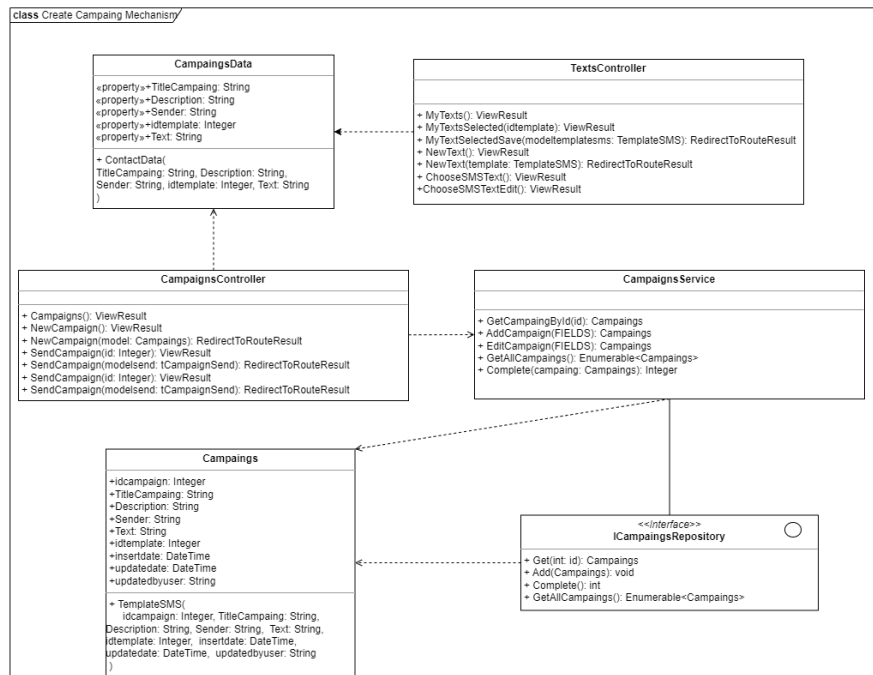


Figura 18: Arquitetura de mecanismos para o caso de utilização de criação de uma campanha

Como referido no ponto 3.2.3, são obtidas todas as templates de mensagem, em que a escolhida será inserida nos dados do formulário representado pela “*CampaignsData*”. Por essa razão, fez-se a ligação com o *TextsController*, visto este conter o método para obtenção das templates. Tendo em conta que todas as entidades e métodos relacionadas com templates de texto estão na figura 17, decidiu-se não apresentar no diagrama acima esses mesmos métodos e entidades, a fim de não haver repetições.

3.1.4 Arquitetura geral da solução

Neste capítulo organiza-se a arquitetura geral da solução, separando todos os componentes em 3 camadas principais, sendo estas a camada de apresentação, de domínio e de acesso a dados, como mostra a figura 19.

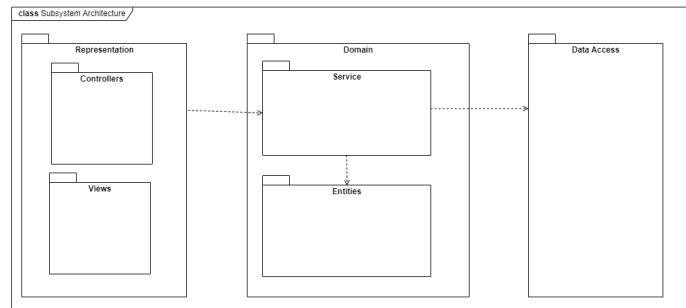


Figura 19: Arquitetura de subsistemas

A proposta de dividir em três níveis é para tornar os componentes entre eles independentes, ou seja, se quisermos modificar a base de dados usada, é possível fazê-lo sem afetar a camada de apresentação (Representation) ou a camada de domínio (Domain). Se quisermos mudar a interface também é possível, basta chamar os mesmos métodos da camada de domínio. Isso facilita a criação de código pois está mais organizado e, se quisermos fazer mudanças futuras, como adicionar recursos, é mais fácil a adaptação.

Outra razão para implementar esse formato é aumentar a coesão, ou seja, agrupar um conjunto de funções que trabalham com algo semelhante, como o acesso à base de dados, tornando o código mais coerente e menos acoplado, ou seja, uma determinada entidade não depende de muitas outras. O problema seria que, ao modificá-la, as que dependem também teriam de ser modificadas, aumentando a complexidade e a possibilidade de cometer erros.

É na camada de domínio que está a lógica da aplicação e os serviços que manipulam os dados obtidos da base de dados para exibí-los na camada de visualização.

As imagens a seguir mostram as relações entre as classes de acordo com a camada à qual pertencem. Decidiu-se apresentar todas as relações pensadas no sistema, para trazer mais robustez ao projeto, observando as várias interações entre os controladores, serviços e entidades.

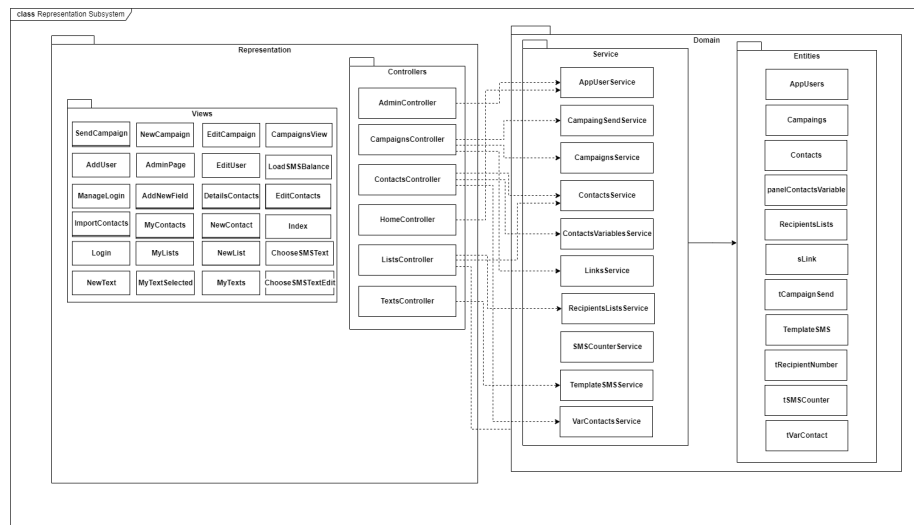


Figura 20: Subsistema de apresentação - Geral

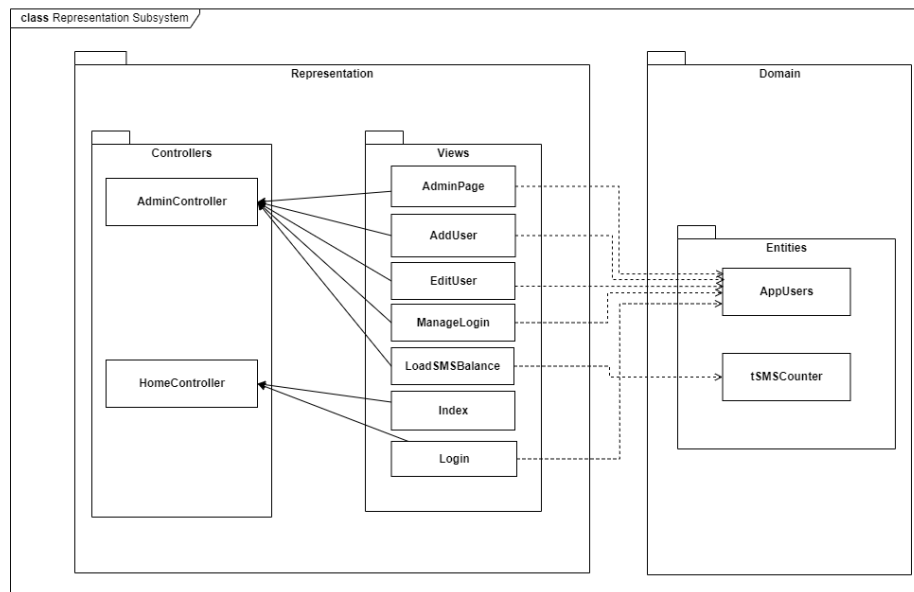


Figura 21: Subsistema de apresentação - ADMIN & HOME

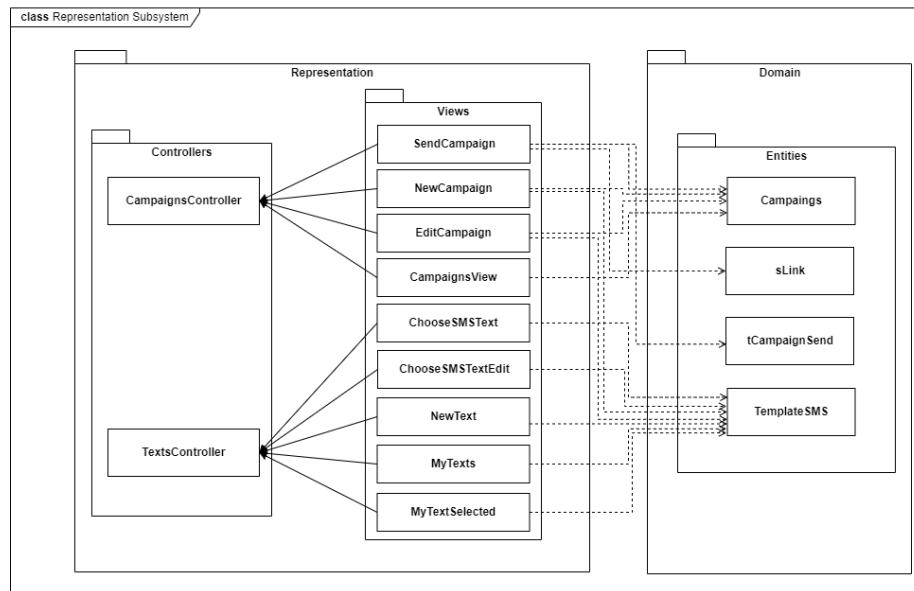


Figura 22: Subsistema de apresentação - CAMPAIGNS & TEXTS

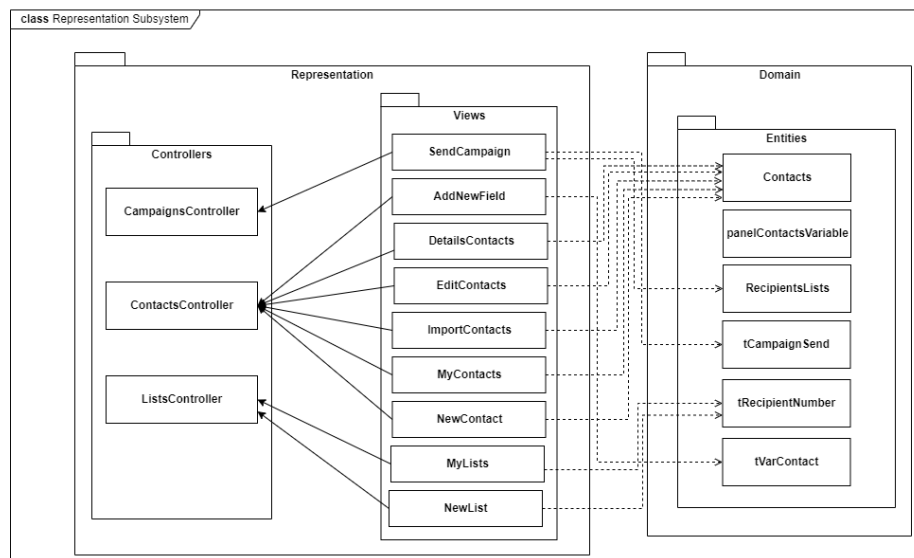


Figura 23: Subsistema de apresentação - CAMPAIGNS, CONTACTS & LISTS

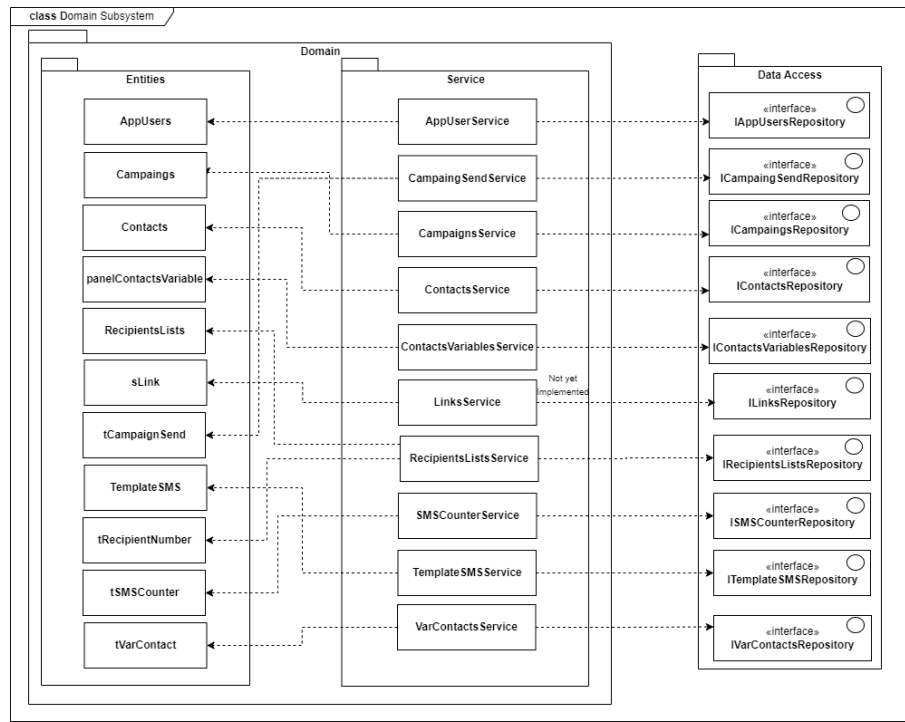


Figura 24: Subsistema de domínio

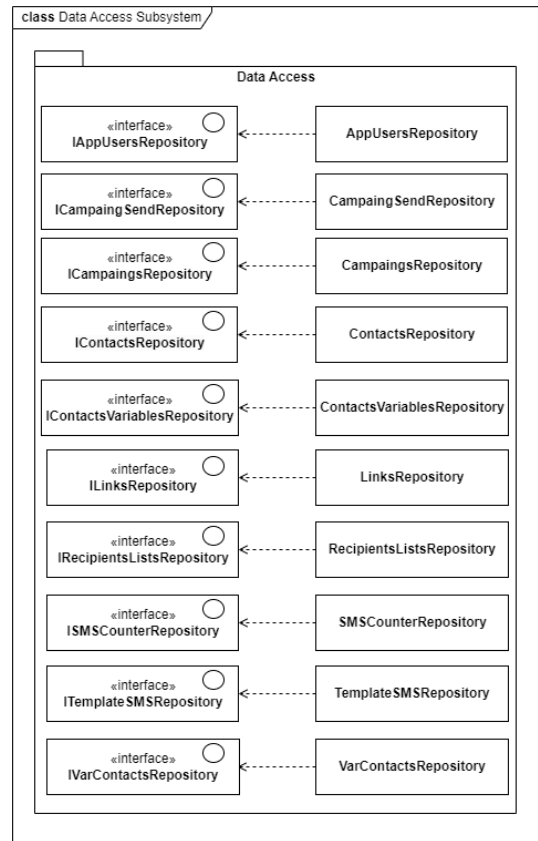


Figura 25: Subsistema de acesso de dados

3.2 Arquitetura detalhada

Nesta fase especifica-se quais as plataformas a serem usadas bem como os dispositivos físicos a ter em conta, planeando com mais detalhe como será feita a implementação e as dependências entre partes. A plataforma que se decidiu usar foi o .Net Framework, que utiliza a linguagem de programação C#. Escolheu-se estas visto que em termos empresariais é a linguagem que mais estou familiarizado para a construção de projetos em MVC (escolhido para este trabalho). Contudo, a versão utilizada é bastante antiga e já existe versões mais recentes. Também se escolheu usar os serviços de base de dados Sql Server pois é o serviço que tenho mais conhecimento, existindo também bastantes referências caso fosse necessário ajuda para corrigir erros. É também bastante suportado pelo .NET Framework e, por isso, acabou por ser uma escolha rápida.

3.2.1 Modelo de dinâmica

O modelo de dinâmica é usado para definir o comportamento de um objeto e as suas restrições.

Na figura 26 é representado um modelo de dinâmica no envio de uma campanha. No caso do Login, se não tiver este feito, vai para o final da ação pois não consegue aceder. Ao enviar uma campanha, é possível enviar a mesma no momento ou programar para mais tarde. Essa programação pode ser enviada ou cancelada antes do envio.

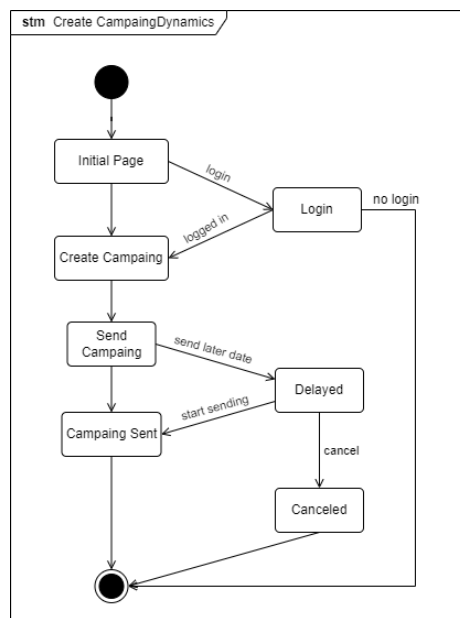


Figura 26: Modelo de Dinâmica

3.2.2 Arquitetura do protótipo de teste

Na arquitetura de teste indicam-se as camadas e os mecanismos para implementar o protótipo de teste. A ideia é conseguir implementar um protótipo de teste que não dependa de questões da plataforma, como a interface, ou de serviços externos, como o acesso à base de dados, com o objetivo de verificar a camada de domínio.

Ao fazer os testes na camada de domínio, diminui-se a chance de erro, visto que se está a garantir que o domínio está completo e funcional.

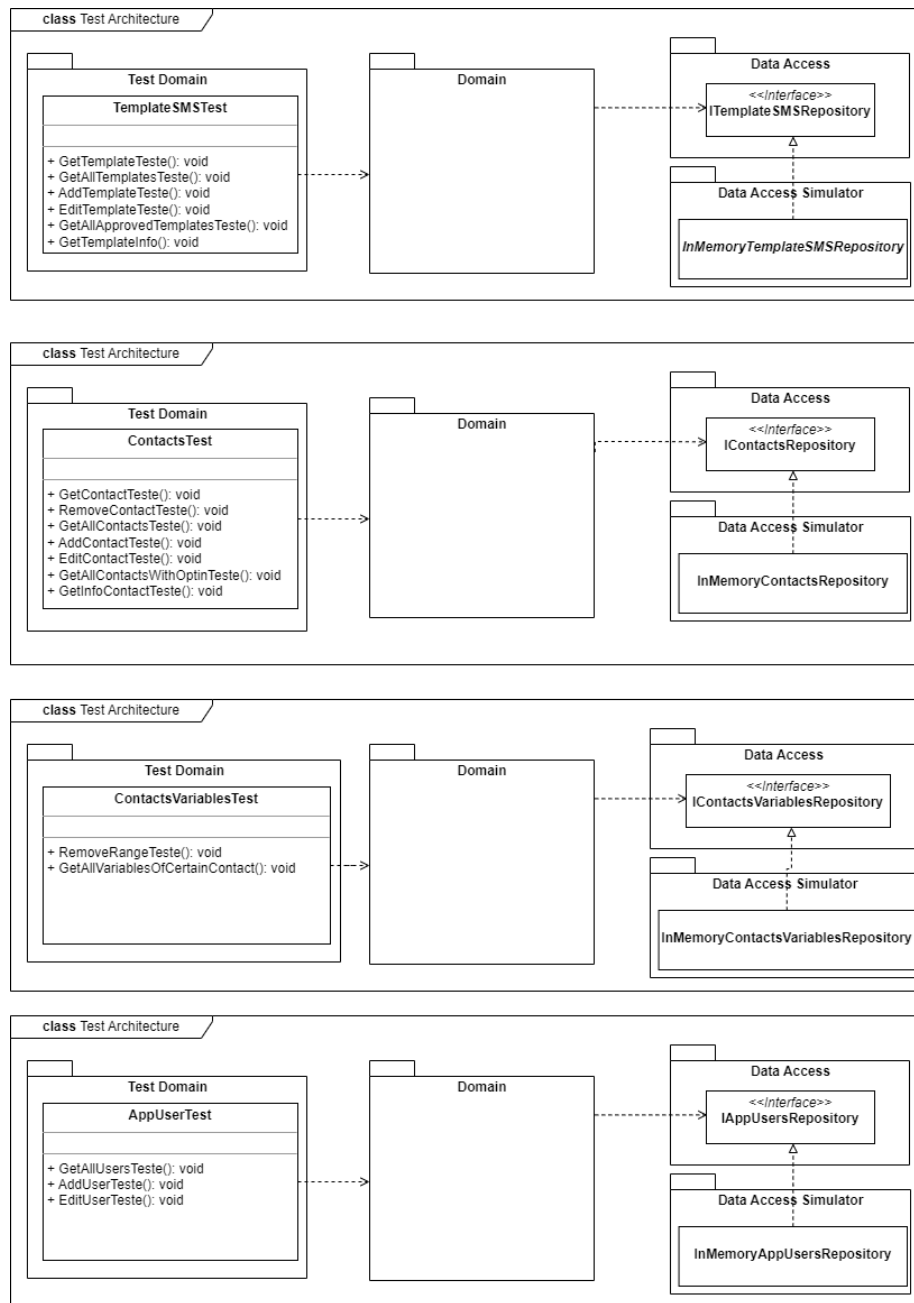


Figura 27: Arquitetura de teste dos casos de utilização nos testes

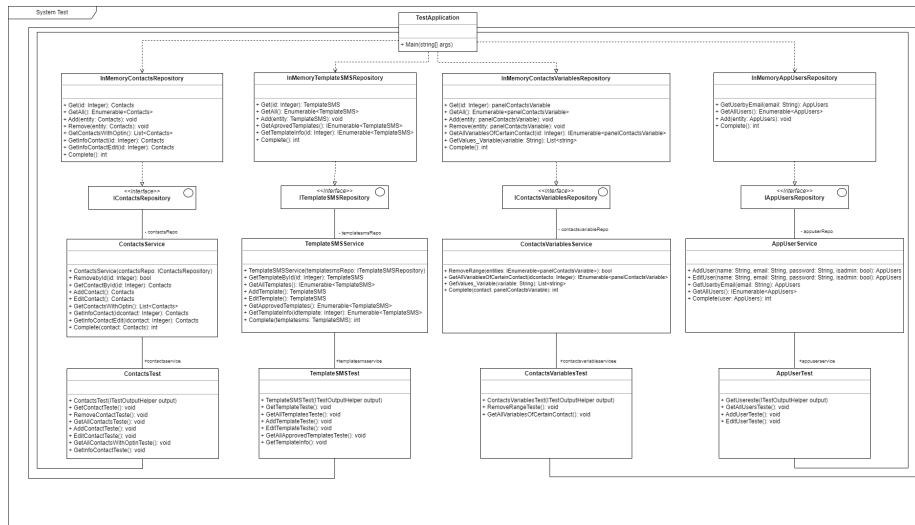


Figura 28: Sistema da aplicação de teste

3.3 Arquitetura do protótipo aplicacional

As figuras 29 e 30 apresentam todas as ligações pensadas para o sistema funcionar na generalidade, podendo faltar alguns pormenores que, devido ao tempo, não foi possível modificar a ideia.

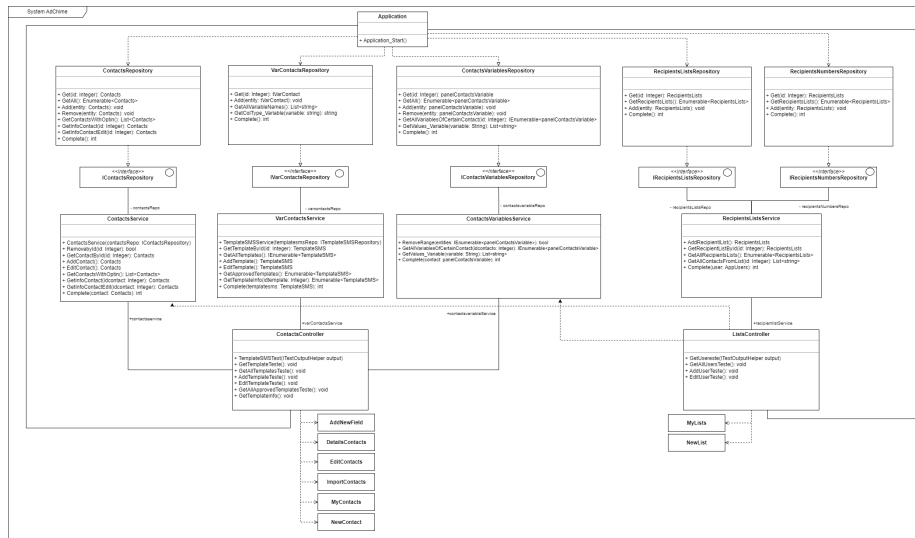


Figura 29: Detalhes do sistema da aplicação

4 Implementação do protótipo de teste

A implementação do protótipo teste visa avaliar a camada de domínio sem qualquer vínculo com a interface ou bibliotecas externas, utilizando apenas a linguagem de programação em modo de linha de comando.

Para isso, os dados são guardados em memória, que é verificado posteriormente se fica corretamente guardado para os quatro casos. No final, se os valores forem iguais, é apresentado o resultado na execução dos testes.

O código foi feito em linguagem C#, usando a framework Xunit para a criação dos testes, criando quatro classes de teste para os quatro casos existentes. Realizou-se um *script* (tests.bat), na pasta principal, para executar o código de maneira a facilitar a apresentação dos resultados para os casos.

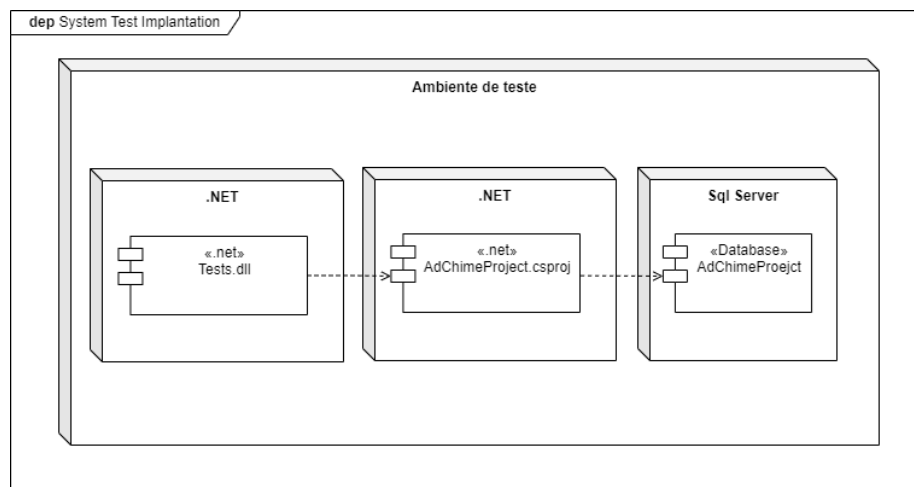


Figura 31: Modelo de implantação de teste

5 Implementação do protótipo aplicativo

Nestas fases implementaram-se todos os casos de utilização referidos no ponto 3.2 - Realização de Casos de Utilização. Para além desses, e para deixar o protótipo aplicativo mais composto, decidiu-se implementar mais alguns casos como por exemplo obter os detalhes de contactos, editar um contacto, criar um novo campo, criação de listas (apenas a página, o método não está funcional), entre outras funcionalidades.

Para fazer tal implementação, como já referido, foi utilizado a linguagem de programação C#, usando a framework .NET Framework para a construção do projeto em MVC. Foi utilizado o serviço SQL Server para a base de dados.

Escolheu-se estas plataformas pois já se tinha conhecimento do uso tanto do C# como dos de serviços de SQL Server. Usou-se também a framework .NET Framework pois esta permite a utilização em ambiente web de forma fácil. A única desvantagem é que esta framework e a base de dados Sql Server apenas correm em sistema Windows e, por isso, não é possível programar em multiplataforma.

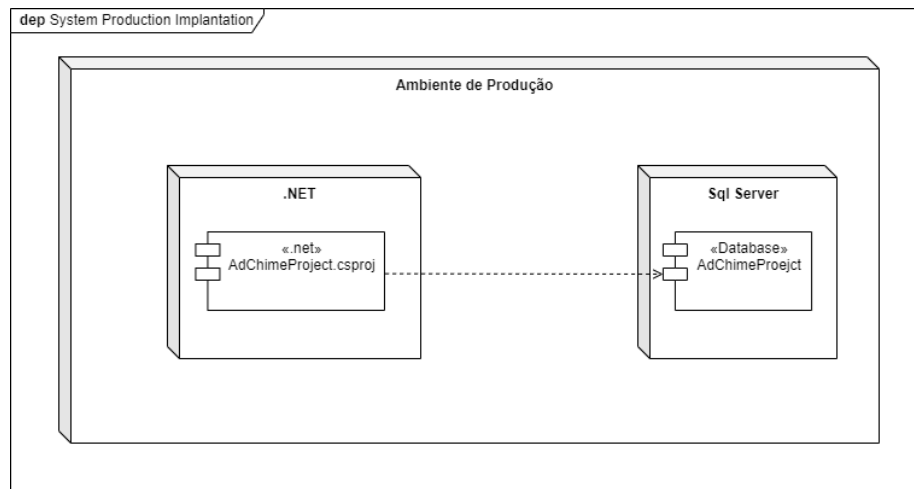


Figura 32: Modelo de implantação

Foi também utilizado a framework *Entity Framework* pois esta é uma camada de abstração que se encontra em cima da base de dados e permite aos desenvolvedores interagir com os dados, usando um conjunto de classes e objetos, em vez de escrever consultas SQL. Isso é uma grande vantagem quando se trata de projetos baseados em dados, pois permite que os desenvolvedores se concentrem na lógica de domínio da aplicação, em vez de gastar tempo com tarefas de acesso a dados de baixo nível. [3]

Para a construção do protótipo, usou-se as classes de domínio criadas anteriormente, já pensadas para o projeto em si. Como referido no ponto 3.4 Arquitetura geral da solução, foram construídas mais algumas funcionalidades do que as 3 mínimas pedidas para deixar o projeto mais robusto. No entanto, algum código foi removido e algumas funcionalidades desabilitadas, devido ao pouco tempo de desenvolvimento do protótipo, mantendo a ideia da existência destas.

Para tal, em todos os casos possíveis, foram construídas interfaces e repositórios para o acesso aos dados, *views* e controladores para a apresentação dos dados e interação com o utilizador, respetivamente. Por fim, foram construídos serviços para cada domínio, para que houvesse uma maior abstração de tarefas e para que não fosse o controlador a ter esse controlo, abstraindo o código para que a sua manutenção seja mais facilitada no futuro.

Também, neste caso nas construções das views, foi criado um ficheiro `_Layout.cshtml` que contém um header e um footer geral, para que não fosse necessário escrevê-los em todos os ficheiros `.cshtml`, fazendo com que se reutilize código. A diferença é apenas na página de Login que não pode usar a mesma estrutura, visto que esta não pode apresentar as várias abas ao utilizador sem Login.

Passando para a demonstração de como se encontram alguns casos de utilização do protótipo, abaixo irá ser apresentado como funciona a dinâmica dos 3 casos de utilização referidos no ponto 3.2.

Antes de ser possível a utilização da plataforma, é necessário que haja um *login*, como mostra a figura 33 que se segue.

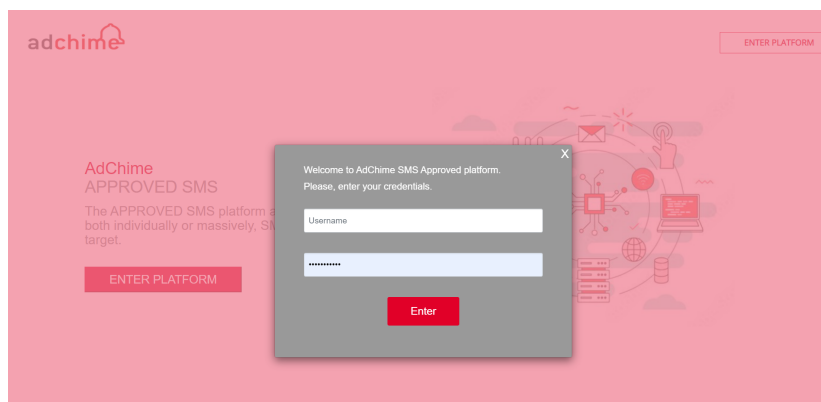
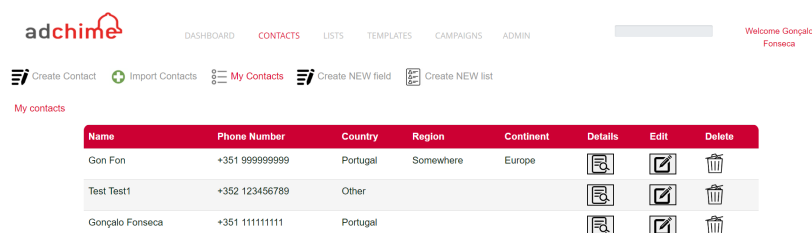


Figura 33: Página de Login

5.1 Criação de um contacto

Para este caso, depois do login feito, a página que abre de seguida é a página que apresenta todos os contactos. Caso não fosse, seria necessário clicar na palavra

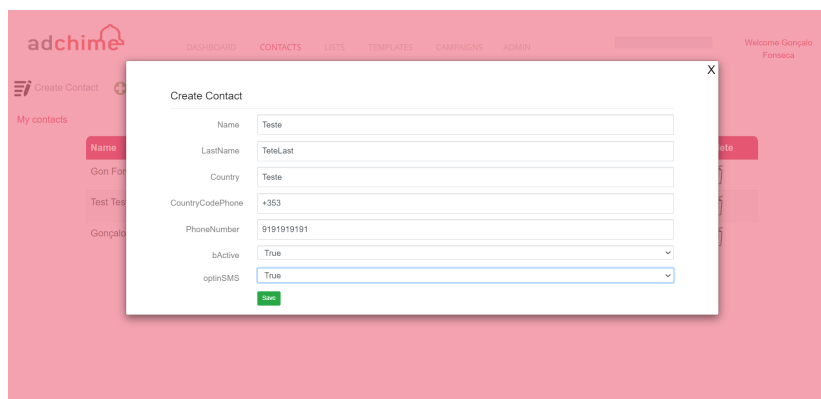
“CONTACTS” nas opções ao lado do logótipo. A figura abaixo apresenta a visualização da mesma.



Name	Phone Number	Country	Region	Continent	Details	Edit	Delete
Gon Fon	+351 999999999	Portugal	Somewhere	Europe			
Test Test1	+352 123456789	Other					
Gonçalo Fonseca	+351 111111111	Portugal					

Figura 34: Página dos Contactos

Ao clicar no botão “Create Contact”, é aberta uma caixa que permite a introdução dos campos pretendidos. De seguida encontra-se um exemplo.



adchime

DASHBOARD CONTACTS LISTS TEMPLATES CAMPAIGNS ADMIN

Welcome Gonçalo Fonseca

Create Contact Import Contacts My Contacts Create NEW field Create NEW list

My contacts

Name

Gon Fon

Test Test1

Gonçalo Fonseca

Create Contact

Name

Teste

TesteLast

Teste

+353

9191919191

True

True

Save

Figura 35: Exemplo de criação de um contacto

Por fim, ao clicar no botão “SAVE”, é atualizada a página dos contactos, já com este criado, como é apresentado de seguida.

Name	Phone Number	Country	Region	Continent	Details	Edit	Delete
Gon Fon	+351 999999999	Portugal	Somewhere	Europe			
Test Test1	+352 123456789	Other					
Gonçalo Fonseca	+351 9919191919	Portugal					
Teste TeleLast	+353 9191919191	Teste					

Figura 36: Página dos Contactos com o novo contacto

Como já referido em outro ponto anterior, faltaria aqui introduzir a possibilidade de preencher as variáveis facultativas. No entanto, por falta de tempo, tal não foi possível.

5.2 Criação de uma template

Para este caso de utilização, primeiramente é necessário abrir a aba “TEXTS”. Ao abrir, tal como na primeira página do ponto anterior, irá abrir uma página que contém todos os templates existentes.

Name	Last Update	Date
Teste 2022 03 11		
Teste 2022 03 16		

Figura 37: Página dos Templates de Mensagens

Ao pressionar o botão “Create NEW SMS template”, uma caixa é aberta que permite a introdução dos campos desejados. Abaixo é mostrado um exemplo.

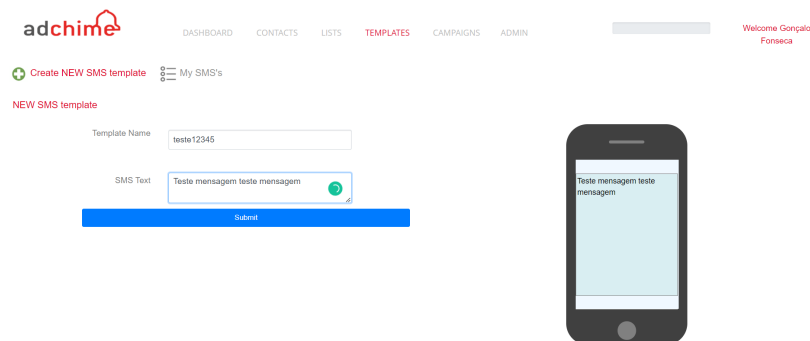


Figura 38: Exemplo de criação de uma template de mensagem

De seguida, ao clicar no botão “SAVE”, é atualizada a página dos templates, já com este criado, como é apresentado de seguida.

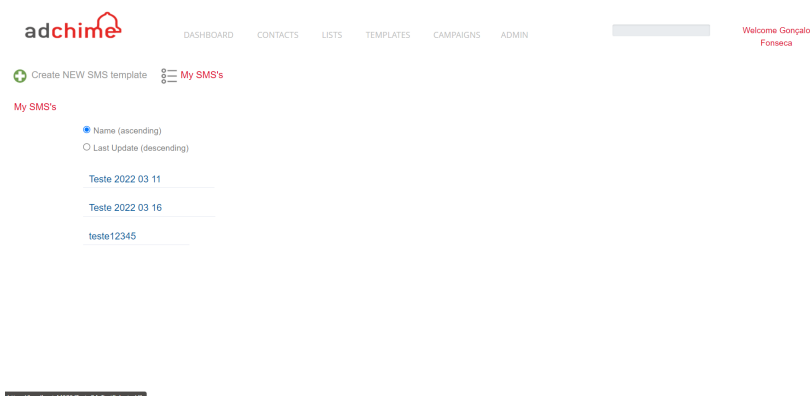


Figura 39: Página dos Templates com o novo template de mensagem criado

5.3 Criação de uma campanha

Para este caso de utilização, primeiramente é necessário abrir a aba “CAMPAIGNS”. Ao abrir, tal como na primeira página do ponto anterior, irá abrir uma página que contém todas as campanhas existentes.

Title Campaign	Description	Sender	Text	
Teste 2	Teste 2 fdfdf df d dfd fd df dfd dfd dfd dfd	Astellas	Mais texto prova que isto tudo é possível, e que, se tudo é mesmo possível, haverá alguém que irá conseguir ainda melhor um desafio do que qualquer outra pesso	Edit Delete Send
TESTE 2	Teste	2L	Mais texto prova que isto tudo é possível, e que, se tudo é mesmo possível, haverá alguém que irá conseguir ainda melhor um desafio do que qualquer outra pesso	Edit Delete Send

Figura 40: Página das Campanhas

Ao pressionar o botão “Create NEW Campaign”, abre-se uma nova página para preencher os vários parâmetros. Um desses parâmetros é um dos textos, em que é necessário clicar no botão “Select Template” para selecionar um texto já preparado, conforme se observa na figura 38.

NEW campaign

Campaign Name

Description of Campaign

SMS From

SMS Text [Select Template](#)

[Submit](#)

The text you select will appear here

Figura 41: Seleção de uma template para ser usado na campanha

Depois de selecionar a template a que se pretende, a janela fecha-se e esse texto é inserido tanto na caixa “SMS Text”, como na figura do telefone. De seguida, preenche-se todos os outros campos. Não existe uma ordem para preenchimento dos campos.

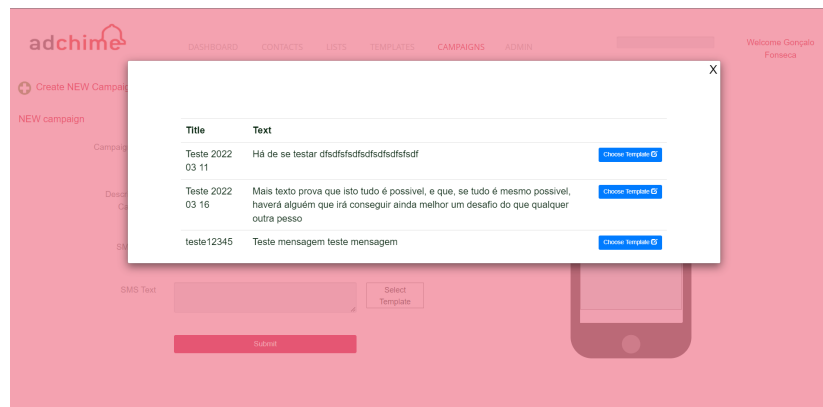


Figura 42: Página de criação de uma campanha

Por fim, ao submeter os dados, a página é redirecionada para a página com todas as campanhas, já com a nova campanha criada incluída.

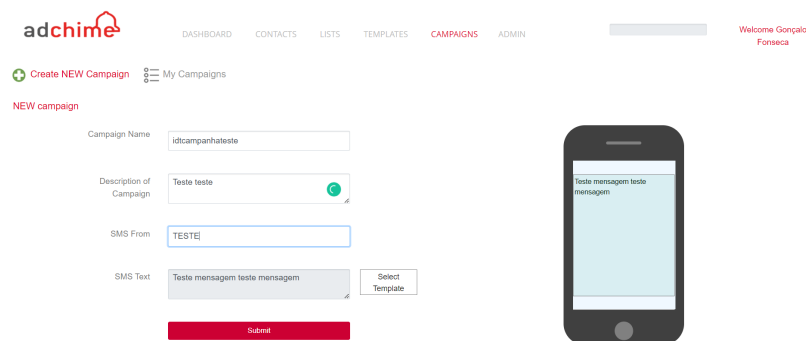


Figura 43: Página das Campanhas com a nova campanha criada

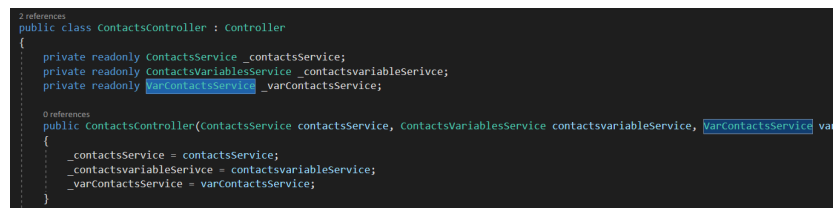
Acima foram apresentados 3 casos de utilização construídos na plataforma. No entanto, ao correr o programa, é possível observar que existem outras funcionalidades disponíveis.

Para correr o programa, é necessário abrir o ficheiro “AdChimeProject.sln” no Visual Studio e iniciar o projeto no botão “IIS Express” presente no topo do programa. Para que este funcione também de forma correta, é necessário ter o Sql Server instalado, com a base de dados AdChimeProeject no servidor local. Caso seja necessário, pode ser alterado o nome da base de dados e a sua conexão no ficheiro “Web.config”.

Por fim, é necessário também esclarecer alguns pontos que não foram debatidos durante o tópico.

Em primeiro lugar, devido à falta de tempo na finalização do projeto, as várias páginas não estão protegidas pelo login. Ou seja, se uma pessoa abrir através do link, irá conseguir aceder. No entanto, essa implementação é rapidamente feita visto já haver uma estrutura de login realizada.

Em segundo lugar, é importante referir que, devido às várias conexões entre serviços em alguns dos pontos (por exemplo, nos CONTACTS era necessário aceder ao serviço **ContactsService**, **ContactsVariablesService** e **VarContactsService**), foram inicializados esses mesmos serviços no Controlador necessário. O que se deveria ter feito era criar um serviço geral (abstrato) que tivesse implementado os serviços referidos. Isto faria com que houvesse uma maior abstração, e uma maior facilidade de manter o código.

A screenshot of a code editor showing the implementation of the ContactsController class. The code is in C# and includes private readonly fields for ContactsService, ContactsVariablesService, and VarContactsService. The constructor initializes these fields. The code is as follows:

```
2 references
public class ContactsController : Controller
{
    private readonly ContactsService _contactsService;
    private readonly ContactsVariablesService _contactsvariableService;
    private readonly VarContactsService _varContactsService;

    0 references
    public ContactsController(ContactsService contactsService, ContactsVariablesService contactsvariableService, VarContactsService var
    {
        _contactsService = contactsService;
        _contactsvariableService = contactsvariableService;
        _varContactsService = varContactsService;
    }
}
```

Figura 44: Contacts Controller

6 Análise reflexiva do projeto elaborado

A elaboração deste projeto consistiu na aprendizagem de técnicas de desenvolvimento de software, seguindo boas práticas, garantindo que o projeto é construído de forma bem estruturada e organizada, permitindo posteriormente a sua alteração sem que haja tantos erros e com uma maior abstração.

A construção dos diagramas serviu, também, para que fosse possível organizar as ideias antes de ter código escrito, permitindo a quem vai ver o código futuramente, possa ter uma ideia do *workflow* do projeto sem ter que analisar o código de forma aprofundada.

Deste modo, com este projeto, foi possível compreender a necessidade do uso de repositórios e serviços para que haja uma maior abstração e, com isso, uma menor chance de ter erros no código, criando também testes para que se garanta que os métodos façam as suas ações com o que é desejado.

7 Conclusão

Ao longo do desenvolvimento deste projeto foi compreendida a relevância de planejar e organizar as etapas para encontrar a solução adequada para o problema apresentado. A importância de minimizar a complexidade do sistema, evitando erros e tempo desperdiçado na resolução dos mesmos, foi também compreendida. Além disso, é fundamental procurar por uma alta coesão e baixo acoplamento entre as partes que compõem o sistema.

Também foi notado o quão útil é a linguagem UML e suas capacidades, pois é uma linguagem que se adapta a qualquer linguagem de programação, podendo ser escalado para qualquer sistema, mantendo organização e a fácil leitura.

Foi constatado também a importância de fazer revisões regulares ao código, o que permite evitar erros que, de outra forma, seriam difíceis de detectar. No fundo, este é o maior desafio na programação de um sistema de software: detectar erros e o tempo perdido para resolvê-los. As abordagens e conceitos aprendidos neste projeto certamente serão aplicados em futuros projetos de software.

Referências

- [1] Documentação de apoio à unidade curricular Engenharia de Software
Luís Morgado, ISEL-DEETC, 1998-2008.
- [2] Obtido de Mockflow - Online Wireframing and Product Design Tool:
<https://www.mockflow.com/>
- [3] MCLEBLANC - Entity Framework Overview - ADO.NET [Em linha]
[Consult. 24 jan. 2023]. Disponível em <https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/ef/overview>.