



UNIVERSIDADE da MADEIRA

Faculdade de Ciências Exatas e da Engenharia

Relatório de Estágio Asseco PST Funchal - Backoffice USSD

Professor Orientador:

Filipe Quintal

Orientador na Empresa:

João Drummond Sousa

Estudante:

João Araújo, 2045919

Índice

1. Introdução.....	2
2. Problema.....	4
3. Solução.....	5
4. Revisão de Literatura.....	6
5. Implementação.....	7
6. Avaliação.....	19
7. Discussão.....	20
8. Conclusão.....	21
9. Referências.....	22

1. Introdução

A Assecó PST(Portuguese Speaking Territories) é uma empresa de tecnologias de informação, especializada no desenvolvimento de software bancário e um referencial na criação de soluções tecnológicas diferenciadoras e de conhecimento em todos os mercados onde atua. Além do desenvolvimento de software bancário, a empresa assegura uma atividade relevante ao nível da prestação de serviços, em quatro áreas: IT Infrastructure & Security, Consulting, Development e Training.[1]

Este relatório destina-se ao relato do meu trabalho na componente Projeto/Estágio da Licenciatura em Engenharia Informática da Universidade da Madeira(UMa) onde tenho como professor orientador o professor Filipe Quintal.

Esta componente possibilitou-me estagiar na Assecó PST Funchal introduzido numa das suas equipas de desenvolvimento, a equipa MDC(Madeira Development Center), equipa de inovação e consultoria interna, especializada em blockchain[2], machine learning[3], consultoria em desenvolvimento frontend e backend[4], etc, na qual tive como orientador por parte da empresa o responsável pela equipa João Drumond Sousa, tal como o suporte dos outros membros dessa mesma equipa para qualquer dúvida que tivesse durante o período de estágio, de 7 de março até 5 de maio, concluindo as 210 horas de estágio na empresa.

Neste período de estágio fiquei encarregado de desenvolver um módulo BackOffice[5] USSD e três CRUD[6] APIs[7], menu e resource bundle[8] no BackOffice[5] de USSD para facilitar a criação de menus para o utilizador final por parte dos desenvolvedores.

O USSD (Unstructured Supplementary Service Data) é um protocolo do Sistema Global para Comunicações Móveis (GSM) que pode ser usado para enviar mensagens de texto e comunicar com operadores de rede móvel. É semelhante ao SMS (Short Message Service), sendo ambos padrões de GSM, no entanto, em contraste com o SMS, uma mensagem USSD cria uma conexão em tempo real durante uma sessão USSD, o que torna um serviço que usa USSD mais responsivo do que um que usa SMS. A comunicação é estabelecida entre o dispositivo móvel e outro dispositivo, normalmente uma rede ou servidor.[9]

O USSD pode ser usado para navegação no Protocolo de Aplicação Wireless (WAP), serviços de pagamento móveis, serviços de retorno de chamadas pré-pagos, serviços de informação baseados em menus e serviços de conteúdo baseados em localização, assim um utilizador interage diretamente do seu dispositivo móvel selecionando opções através de diversos menus, ou seja, o USSD permite a comunicação bidirecional de informação, desde que a linha de comunicação permaneça aberta. Desta forma, as queries e respostas são quase instantâneas.[9]

No geral, o USSD é uma tecnologia rápida, interativa e económica para uma ampla gama de aplicações, tornando-se uma ferramenta importante para empresas, fornecedores de serviços e utilizadores de dispositivos móveis.

No caso deste projeto em específico a ligação é começada por parte do utilizador final que envia o código correspondente à conexão com o banco pretendido para a operadora de rede móvel, depois a operadora irá encaminhar a mensagem para o servidor USSD que então irá fazer a ligação ao core bancário, de modo a criar um canal de comunicação responsivo onde o utilizador continua a inserir caracteres correspondentes ao serviço desejado recebendo resposta com o menu designado proveniente do USSD para seleção de opções.

No desenvolvimento destes projetos pode existir a necessidade de criação de ferramentas adicionais para facilitar a criação de ambientes personalizados para o utilizador final, como é o caso deste em que houve a necessidade de criar uma ferramenta para criação e manutenção dos menus de escolha por onde o utilizador irá navegar para realizar as operações desejadas depois de ativada a aplicação, que sem esta ferramenta seria realizada diretamente na base de dados o que obviamente não é prático no seu desenvolvimento.

Este projeto será de uso direcionado aos desenvolvedores do projeto principal USSD na equipa MDC, de modo a facilitar a criação dos menus para o utilizador final visualizar no seu dispositivo móvel, uma das vantagens do USSD é o facto de não necessitar de conexão à internet[10], apenas à operadora de rede, o que é vantajoso em países em que o uso de smartphones não é tão comum ou em que o método mais popular de realizar operações bancárias seja o USSD como em vários países africanos.

2. Problema

O USSD destina-se a possibilitar operações bancárias em dispositivos móveis através da rede operadora, comunicando com o USSD e então com o core bancário. Ao desenvolver o USSD, na testagem dos serviços dos menus e formato dos mesmos foi feita a criação da base dos menus diretamente na base de dados, no entanto, ao chegar ao formato desejado esta forma de trabalhar torna-se indesejada/imprática para testagem, criação e manutenção dos diferentes menus, sendo que a listagem na base de dados nem sempre é a mais organizada em comparação a uma tabela de valores, quando o processo poderia ser agilizado através de uma ferramenta de uso específico para o caso, daí surge uma necessidade de criação de tal ferramenta para não só agilizar mas também simplificar e organizar o processo de criação e manutenção desses menus, os quais serão depois mostrados ao utilizador final.

Na criação de objetos e variáveis na base de dados, no caso deste projeto no MongoDB[11], é sempre necessário escolher o tipo de dado a ser inserido e o processo de colocar objetos dentro de objetos ou arrays[12] tem toda uma especificidade requerida na base de dados, no entanto, numa ferramenta esses formatos já estão programados automaticamente e apenas é necessária a inserção dos valores a ser inseridos na base de dados em formulários mais práticos e claros para manuseamento por parte do desenvolvedor, sem grandes complexidades na operação, tanto a criação como manutenção podem ser feitas mais facilmente apenas inserindo valores e pressionando um botão, o que se repete diversas vezes na testagem do USSD, daí a necessidade de não estar sempre a realizar a mesma tarefa de forma tão dificultada quando o processo pode ser simplificado.

Sendo assim, o meu estágio destina-se a criar a ferramenta para criação e manutenção de menus para o USSD, solucionando o problema e facilitando o desenvolvimento do USSD.

3. Solução

Dado o problema o meu projeto destinou-se a solucioná-lo, através do desenvolvimento de uma ferramenta de criação e manutenção dos menus na base de dados utilizando a arquitetura de referência[13] da empresa no Frontend[4], tornando a criação e manutenção dos menus no desenvolvimento do produto mais simples, de modo a que se possam criar os menus desejados de forma mais organizada, mais rápida e menos complexa, para além de obter uma interface mais apelativa ao desenvolvedor.

Na solução, foram desenvolvidos dois programas em separado:

- Front(ou Frontend[4]), o programa que cria a visualização disponibilizada para o utilizador (User Interface) com características de filtragem e visualização clara do que são os dados através de grelhas e outras secções, como necessário em diversas partes tendo em conta o formato na base de dados e ainda envio de informação quando requeridos dados ou criação ou alteração dos mesmos.
- Middleware[4], o programa que recebe os pedidos e informação do Front e busca ou faz as alterações necessárias na base de dados de acordo com o formato dos modelos da base de dados.

Ao correr os dois programas é então possível alcançar a solução desejada do projeto através da cooperação do Front com o Middleware para uma visualização simples dos dados e alteração dos mesmos caso necessário, de modo a criar os menus desejados para expôr ao utilizador final.

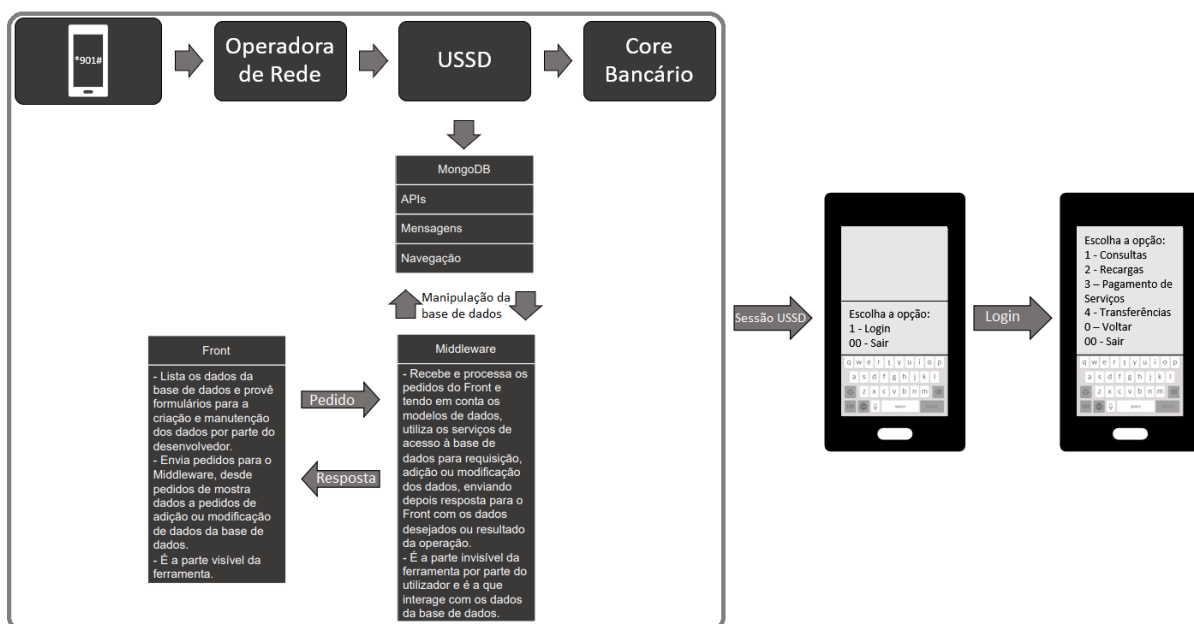


Fig. 1 Diagrama de ilustração da comunicação entre os componentes

4. Revisão de Literatura

Neste projeto foi utilizada majoritariamente a linguagem Javascript[14], mas mais especificamente no Front foi utilizada a tecnologia React[15] juntamente com a arquitetura de referência[13] da empresa, de modo a ficar com um formato desejado e personalizado da empresa, para além de facilitar o desenvolvimento do Front de aplicações, ao possibilitar a concentração do desenvolvedor mais no desenvolvimento de novas funcionalidades e menos na personalização de formatos e cores de diversos componentes, já personalizados e disponíveis ao utilizador na página tendo já um formato pré concebido para certos projetos na empresa.

Quanto ao Middleware, foi utilizada a biblioteca MongoDB[11] para comunicação com a base de dados e manipulação e formatação dos mesmos, foi também utilizado o ambiente Node.js[16] e a framework[17] Express.js[18] para o desenvolvimento deste programa. Quanto à testagem do Middleware e visualização dos seus outputs foi usado o conjunto de ferramentas open-source Swagger[19], um software que eu nunca tinha utilizado e que apesar de ter de ser construído um formato para mostra e testagem neste software, acabou por demonstrar ser uma mais valia na testagem de funções ligadas à base de dados, devido à visualização das funções ao serem ativadas sem necessidade do ambiente no Front estar desenvolvido.

Acredito que estas tecnologias foram bem seleccionadas e provaram-se úteis no desenvolvimento do projeto num todo, no entanto, não discarto a possibilidade do projeto ser desenvolvido utilizando outras tecnologias como Angular[20] ou só Javascript[14] sem frameworks[17], no entanto, dado o projeto a escolha foi bem realizada.

5. Implementação

5.1. Middleware

Na implementação deste projeto comecei por desenvolver os serviços base do middleware também denominados de CRUD[6], para isso criei um programa base conectado ao browser destinado a receber pedidos do Front[4] para então pedir a informação à base de dados, neste caso o MongoDB[11], através de queries[21].

Para que estas queries[21] funcionem de forma correta é primeiro necessário conectar o programa à base de dados e criar modelos de objetos para inserção e reconhecimento dos dados por parte da base de dados, neste caso existem 3 modelos principais (APIs[7], mensagens e navegações), os quais podem ter outros modelos de dados inseridos na sua constituição, caso necessário. As APIs[7] têm caminhos dos serviços a chamar quando uma opção é selecionada para obter os dados a serem ilustrados no menu, as navegações têm os caminhos de cada menu, ou seja, ao que corresponde cada resposta por parte do utilizador e o que deve ser ativado nessa seleção, passando de navegação para navegação, e as mensagens têm o texto que será mostrado ao utilizador nas opções dos menus e resultados das operações juntamente com os dados recebidos do serviço chamado pela API[7] relacionada.

Criados os modelos de dados e as queries[21] iniciais procedi à configuração do Swagger[19] onde escrevi o código correspondente à simulação de formulário característico do programa, onde criei campos equivalentes ao que estará no Front para simular a manipulação de dados sem necessidade de ter o Front funcional, assim verificando a funcionalidade das queries[21] e formato dos dados como ilustrado na Fig. 2.

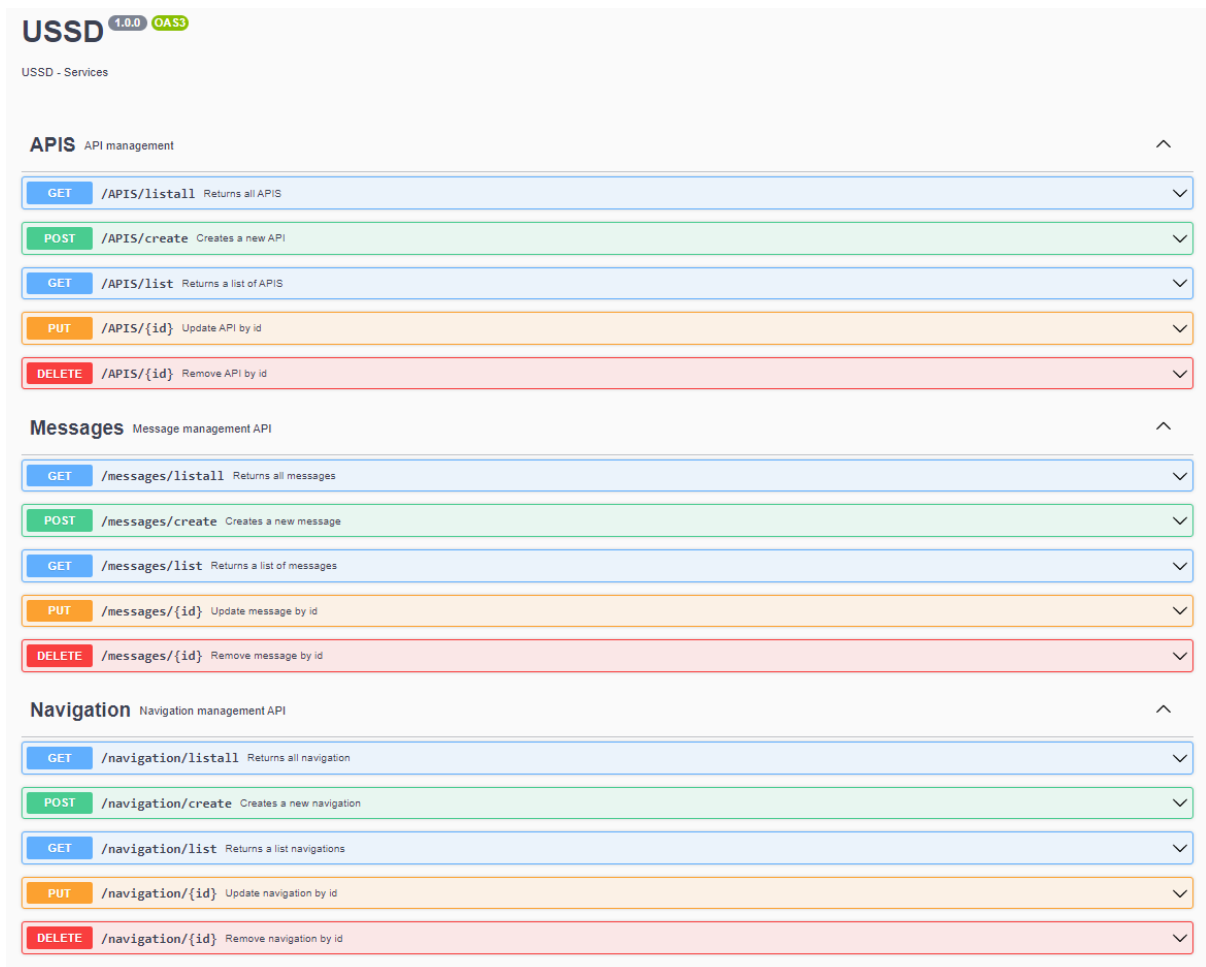


Fig. 2 Serviços base USSD no swagger

Tendo o Middleware base avancei para o Front, alterando o Middleware de acordo com as necessidades do Front no seu desenvolvimento.

5.2. Listagem dos dados

No Front comecei por criar as páginas de consulta das APIs[7], Mensagens e Navegações, utilizando a estrutura de referência da empresa como base, nas quais primeiramente fiz uma lista de todos os dados (recebidos das pastas na base de dados) de cada página. Após verificar o funcionamento completo das listas nas páginas procedi ao planejamento do aspeto que essas páginas iniciais teriam, como ilustrado na Fig. 3, o qual incluía um dashboard[22] vertical já incorporado na estrutura de referência, onde inseri as 3 abas dentro de uma aba já existente chamada “Consultas”, um método de filtragem, no caso por nome, um botão de criar novos objetos na base de dados, dependendo da página, e a lista já antes incorporada com paginação já existente da estrutura de referência à qual adaptei as queries[21] do Middleware para respeitar o formato.

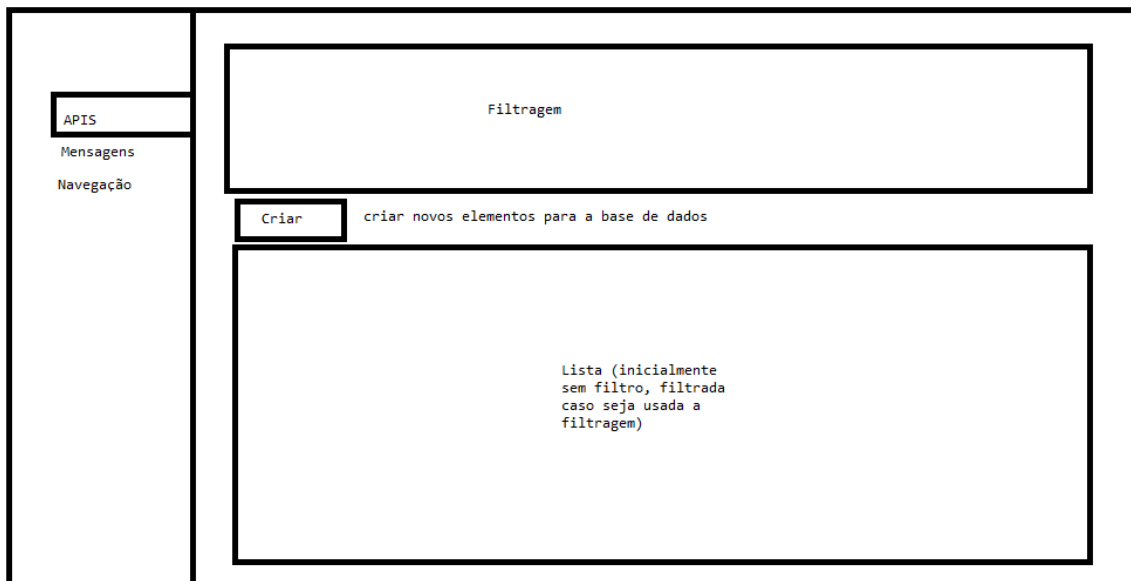


Fig. 3 Esboço base das páginas iniciais

Até este ponto o objetivo seria apenas começar por apresentar as informações presentes na base de dados na lista da página.

Ao consultar o meu orientador da empresa de modo a obter feedback sobre o formato da página inicial, ele expressou que eu deveria organizar a informação em arrays[12] e objetos dentro de uma secção em que se possam visualizar detalhes, os quais seriam essas listas e objetos, para além de modificar os IDs presente na lista para os nomes presente nos reais objetos com os IDs correspondentes como podemos ver na Fig. 4.

ID	Nome	Tipo	API	Entradas	Saldos	Mensagem de Sucesso	Mensagem de Erro	SubApis
6184c034789a0b8d9ca05da	username_session	POST		Array	Array	6184dad8789a0b8d9ca0812	Object	Array
6186437781ab4890b58ababab	account	GET	/v/accounts	Array	Array	6183d521446e7e6c2185a2bae	Object	Array
61968882789a0b8d9ca0578	last 10 movements	GET	/v/accounts/{accountNumber}/movements	Array	Array	618cdce340669039a59ad24a	Object	Array
618257dc446e7e6c2185a252	language_session	POST		Array	Array	6183b462446e7e6c2185a29b	Object	Array
61a4f4da135f0cb9e72c3f1	set account number	GET	/v/accounts	Array	Array		Object	Array
617833fd924aa977da392b1c	detail_account	GET	/v/accounts/{accountNumber}	Array	Array	618508c2d93ca3637809e32e	Object	Array
61a445349eb78739c5f8c96	Account Number	GET	/v/accounts/{accountNumber}	Array	Array	61a459849eb78739c5f8c9d	Object	Array
618f02a53234b4779a8f839	login	POST	/v/login/password	Array	Array	618f597a446e7e6c2185a23b	Object	Array

Fig. 4 Screenshot do projeto com os problemas seleccionados a vermelho

Depois de criar uma forma de visualizar detalhes, no caso expandindo para baixo, voltei a ser lembrado de que teria de desenvolver a lista de modo a que ao

invés de mostrar apenas diretamente o que está na base de dados como na Fig. 5, alguns dados como por exemplo o resource bundle[8], próxima navegação e API[7] nas navegações, ao invés de mostrar diretamente o ID do objeto que é o que estava presente na base de dados, teria de fazer uma procura na base de dados com o ID do objeto de modo a que constasse o nome do objeto correspondente ao invés do ID que não seria prático na identificação do objeto por parte de um utilizador, logo para solucionar este problema fui ao Middleware e criei um método de procura através do ID, desta forma poderia utilizar esse método no Front para pedir informação enviando o ID presente no objeto da lista e recebendo a informação correspondente, neste caso selecionado o nome.

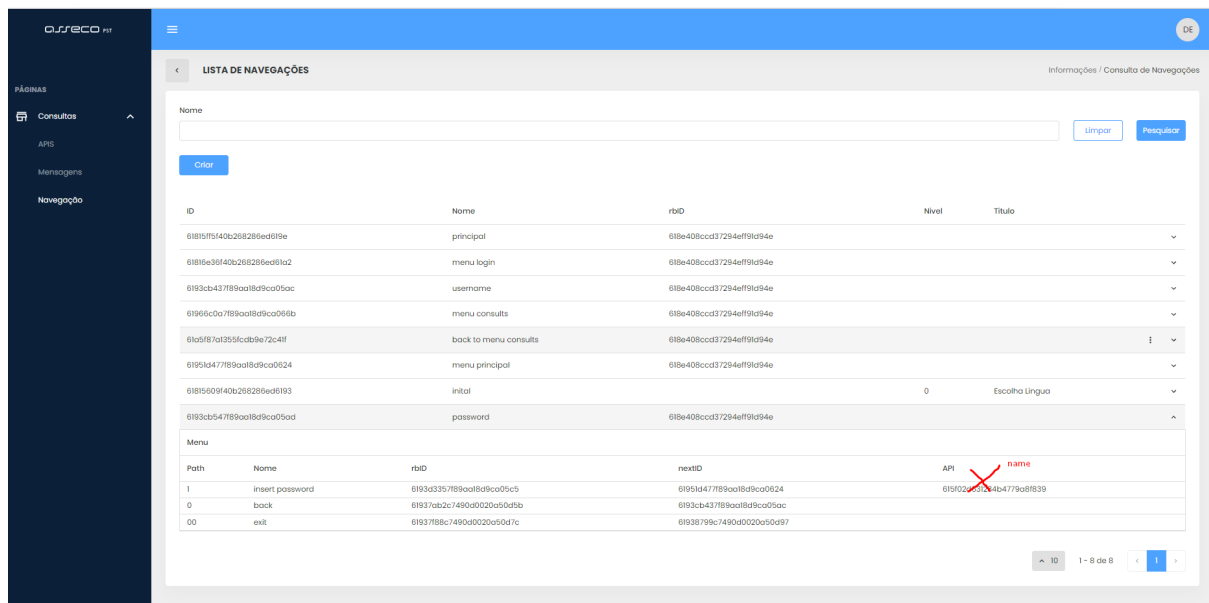


Fig. 5 Screenshot da página com a função de expandir e IDs

Após solucionados os problemas acima descritos dos detalhes e IDs decidi modificar a visualização dos detalhes pois haveriam mais listas e objetos nas navegações e APIs[7] o que fariam a expansão para além de imprática, confusa e problemática. Logo os detalhes foram modificados para abrir uma nova página, a página de detalhes que varia nos conteúdos entre as APIs[7] e navegações, a consulta de mensagens é a única página sem detalhes por ter apenas algumas variáveis dentro de cada mensagem, o que não requer nada a mais para além da lista de dados.

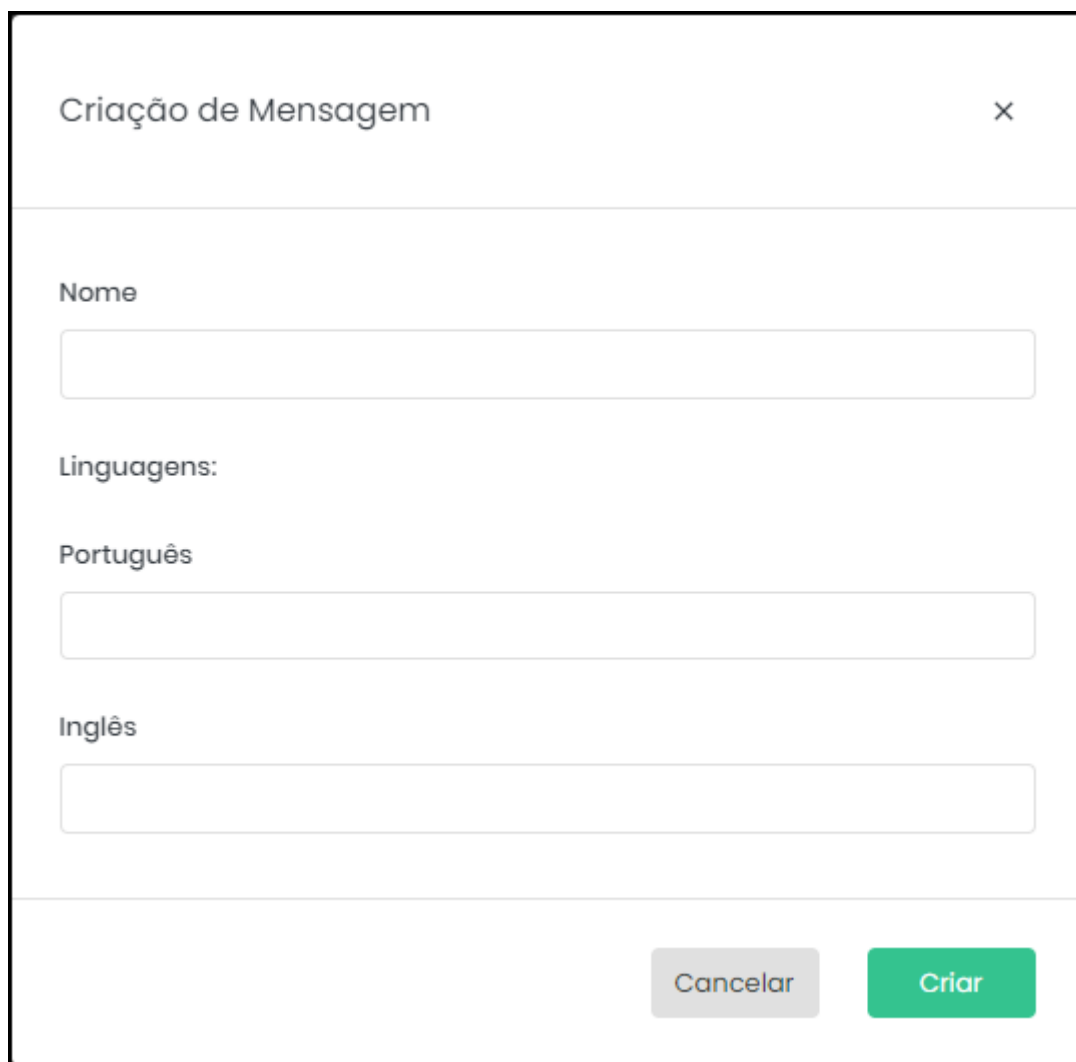
5.3. Mensagens

Devido à página de mensagens ser a mais simples, de modo a servir de base para as outras páginas, comecei na manutenção da mensagens, começando por criar uma nova página de criação de mensagens que se abre ao clicar no botão “Criar” a azul existente em cada página. No entanto, devido ao facto das mensagens apenas terem 3 campos de texto para introduzir dados e criar uma mensagem não achei que justificasse uma página inteira, para além de não ser esteticamente muito

agradável, logo procurei outra opção e depois de consultar um colega da equipa MDC em que estava inserido encontrei uma solução, que seria a utilização de uma simples caixa de diálogo, a qual já estaria previamente implementada na base da estrutura de referência que eu estava a utilizar mas que no momento era apenas uma caixa com uma pequena mensagem a perguntar se gostaria de eliminar um item, ainda sem nenhuma real funcionalidade, apenas estética.

Revi o formato da caixa de diálogo e importei o formulário que tinha feito anteriormente na página de criação de mensagem inicial como podemos ver na Fig. 6, no entanto tive de modificar alguns elementos e adaptar o funcionamento à caixa de diálogo porque apesar de esteticamente mais agradável, a caixa de diálogo tinha algumas complexidades a mais para funcionar.

Criado o novo formato de criação de mensagens consultei o meu orientador da empresa para decidir qual seria o melhor formato e ele concordou que as caixas de diálogo ficariam melhores, assim sendo, retirei a página de criação e mantive apenas a caixa de diálogo, que então reutilizei para a edição e eliminação de mensagens que aparecem ao clicar com o botão direito no objeto da lista ou clicar em 3 pontinho que aparecem quando o rato está em cima do objeto. A edição abre uma caixa de diálogo igual à da criação mas importando os dados para os campos do formulário assim facilitando na visualização dos dados a editar por parte de um utilizador, já a eliminação é como a edição, mostrando os campos do formulário preenchidos com a diferença em que todos os campos estão desabilitados, não podendo ser modificados e servindo apenas para visualização/verificação dos dados quando um utilizador deseja eliminar uma mensagem. Além disso nos formulários também criei verificação dos campos, de modo a que caso o que for preenchido não for o que foi requisitado o valor não será aceite, por exemplo se o valor requisitado for um número então esse campo apenas aceita valores numéricos, fiz esta verificação e também a importação de dados como nos casos da edição e eliminação utilizando os métodos de validação do “react-hook-form”[23] e “yup”[24].



A modal dialog box titled "Criação de Mensagem" with a close button (X) in the top right corner. The form contains three input fields: "Nome", "Português", and "Inglês". The "Linguagens:" label is positioned above the "Português" and "Inglês" fields. At the bottom right, there are two buttons: "Cancelar" (grey) and "Criar" (green).

Criação de Mensagem

Nome

Linguagens:

Português

Inglês

Cancelar Criar

Fig. 6 Caixa de diálogo com formulário para criação de mensagens

Concluída a manutenção de mensagens a página de mensagens está então completa, antes de passar para a próxima página modifiquei a dashboard[22], retirando a aba expansível de consulta e deixei apenas as 3 abas necessárias, adicionando símbolos característicos que importei da biblioteca “@mui”[25], como podemos ver na Fig. 7.

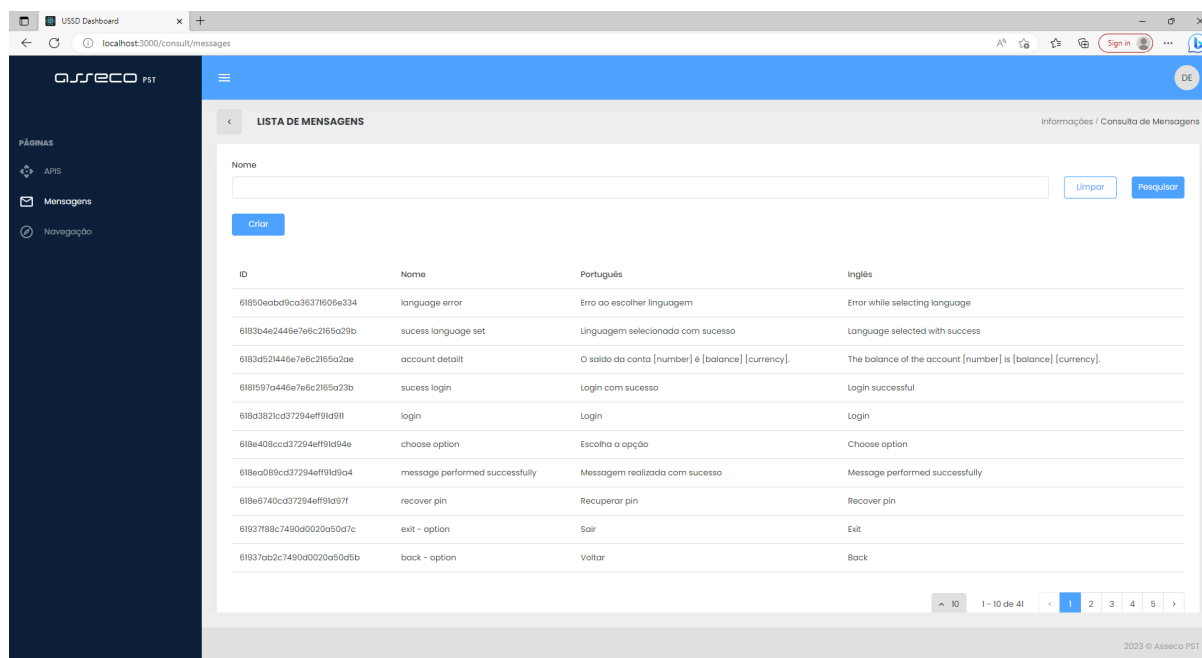


Fig. 7 Página de consulta de mensagens concluída

5.4. Navegação

Assim passei para a próxima página, a página de navegação, onde comecei por fazer as caixas de diálogo para a criação, edição e eliminação de navegações da mesma forma que foi feito para as mensagens mas apenas para os dados presentes na página inicial das navegações (id, nome e resource bundle[8]), procedendo depois à página de detalhes, onde comecei por criar um cabeçalho(header) onde coloquei os dados do objeto selecionado presentes na lista da página principal de modo a não confundir o utilizador com apenas os dados dos detalhes sem a origem do que foi selecionado e então abaixo criei uma aba com os objetos presentes dentro do array[12] Menu, estando então a lista dos objetos que caracterizam a navegação no USSD, utilizando as mensagens (resource bundle[8]) como descrição para o utilizador. Aproveitei também para dar uso a uns botões que encontrei na base da estrutura de referência e personalizei as cores e simbolos dos botões de modo a fazer sentido no contexto, tendo um botão para criar um novo objeto no menu, um botão de editar para editar a navegação em si (o que está no cabeçalho), tendo a mesma função que a da página inicial só que dentro do objeto aberto que depois de editado irá atualizar a informação na página de detalhe, um botão de eliminar a navegação que irá eliminar e levar o utilizador de volta à página inicial e um botão de imprimir os dados que deixei por achar que poderia ser útil a imprimir uma lista dos dados, achei que ficaria esteticamente mais agradável, mas depois de consultar com um colega de equipa e com o meu orientador da empresa decidimos modificar um pouco o display de modo a ficar mais dentro dos padrões de estilo da empresa e ficando os botões menos confusos, porque não fazia sentido o botão de criar um objeto no menu ficar acima onde os botões de edição da navegação em si ficavam e o de imprimir não seria utilizado. Assim sendo, teria de

passar o botão de criação para baixo para dentro do menu e criar 3 pontinhos no cabeçalho que ao abrir deveriam também mostrar as opções de edição e eliminação da navegação, ilustrado na Fig. 8.

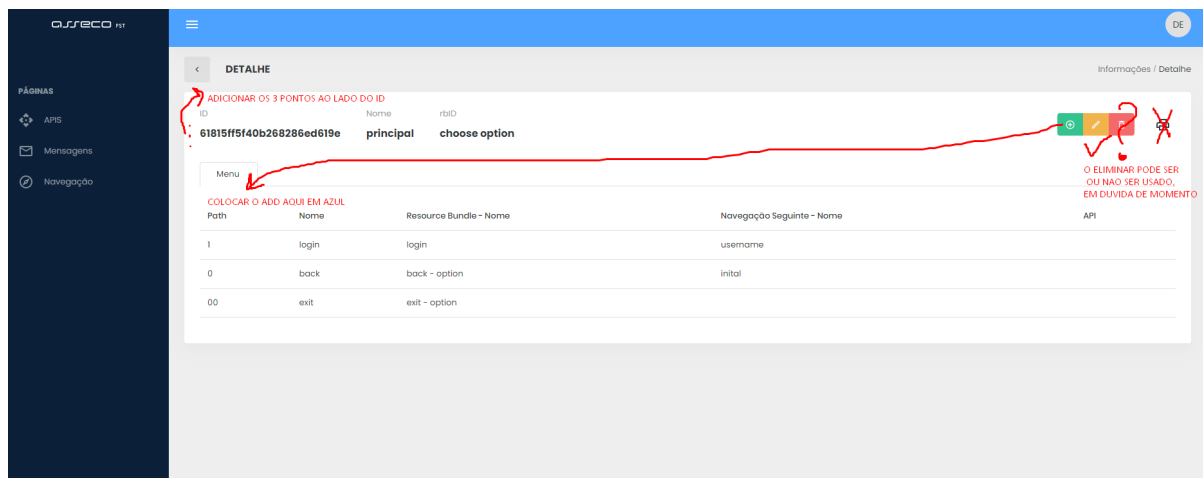


Fig. 8 Ilustração das mudanças a serem feitas na página de detalhe da navegação

Depois destas mudanças serem concluídas passei então para o formulário da criação, edição e eliminação dos objetos do Menu, onde reutilizei as caixas de diálogo das mensagens e criei um novo formulário para os dados a serem inseridos e ilustrados na criação, edição e eliminação do menu da navegação. Feita a página de detalhe procedi para a página dos parâmetros da API[7] relacionada à navegação que seria mais uma página de detalhe dentro do detalhe que abriria como detalhe depois de clicar no objeto/elemento do menu pretendido, no entanto, após averiguar a situação decidiu-se que uma página não justificaria apenas uma pequena lista que poderia até estar vazia, logo retirei a página para os parâmetros da API[7] e ao invés de uma página reutilizei a ideia que tive no início do projeto de expandir o objeto da lista, desta vez fazendo sentido como seria uma pequena lista de máximo de 5 elementos no momento, como podemos ver na Fig. 9. De modo a criar um formulário dinâmico capaz de se adequar ao tamanho da lista pretendido, criei um método de detecção de elementos da lista, a qual o formulário irá utilizar para se expandir ou restringir de acordo com o valor e método usado, por exemplo, os botões “Add” para adicionar e “Apagar” para apagar um parâmetro só são visíveis nos modos de criação e edição, ilustrado na Fig. 10, não estando presentes no método de eliminação a qual o formulário é apenas utilizado para visualização e verificação dos valores existente.

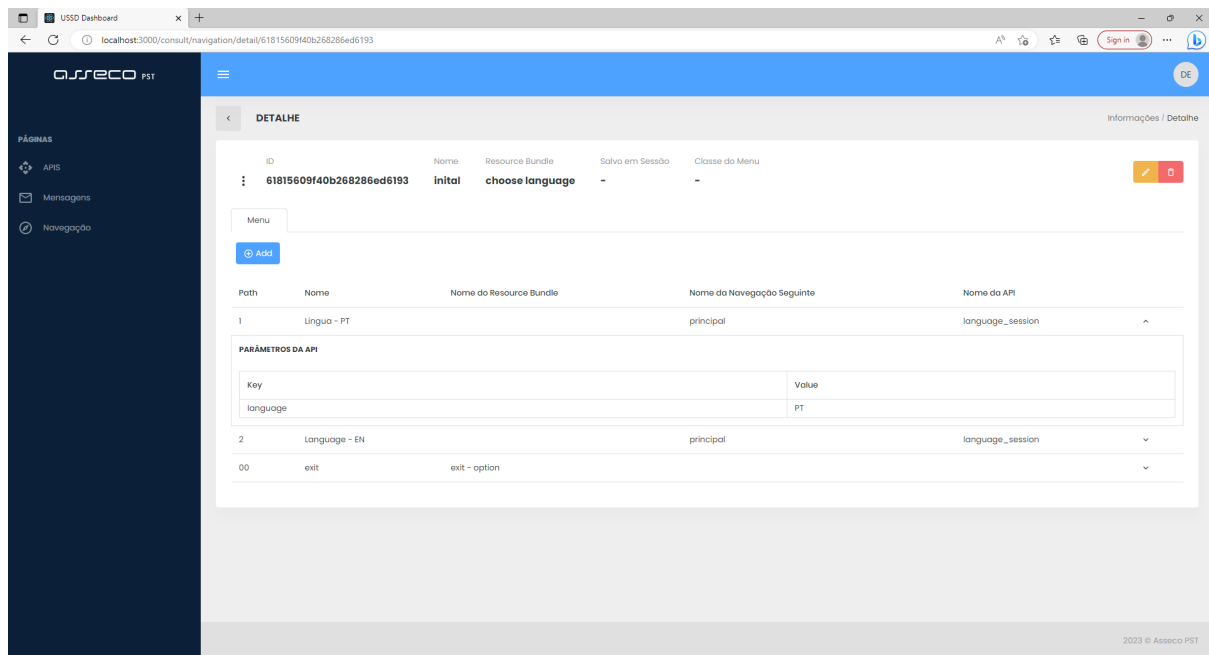


Fig. 9 Versão final da página de detalhe de navegação

Assim ficou concluída a página de detalhe de navegação, estando as navegações concluídas, ficando apenas a faltar a página de APIs[7] e os seus detalhes.

Edição de Elemento do Menu

×

Path

1

Nome

Lingua - PT

Nome do Resource Bundle

Nome da Navegação Seguinte

principal

Nome da API

language_session

PARÂMETROS DA API

Add

Key

Value

language

PT

Apagar

Cancelar

Guardar

Fig. 10 Caixa de diálogo com formulário de edição de um elemento de um menu de navegação

5.5. APIs

Na página principal das APIs[7] já tendo a lista e filtragem concebidas procedi à manutenção das APIs[7], ativando as caixas de diálogo com o formulário de API[7]

ao clicar na opção desejada, como presente na Fig. 11, ficando assim a página inicial das APIs[7] concluída.

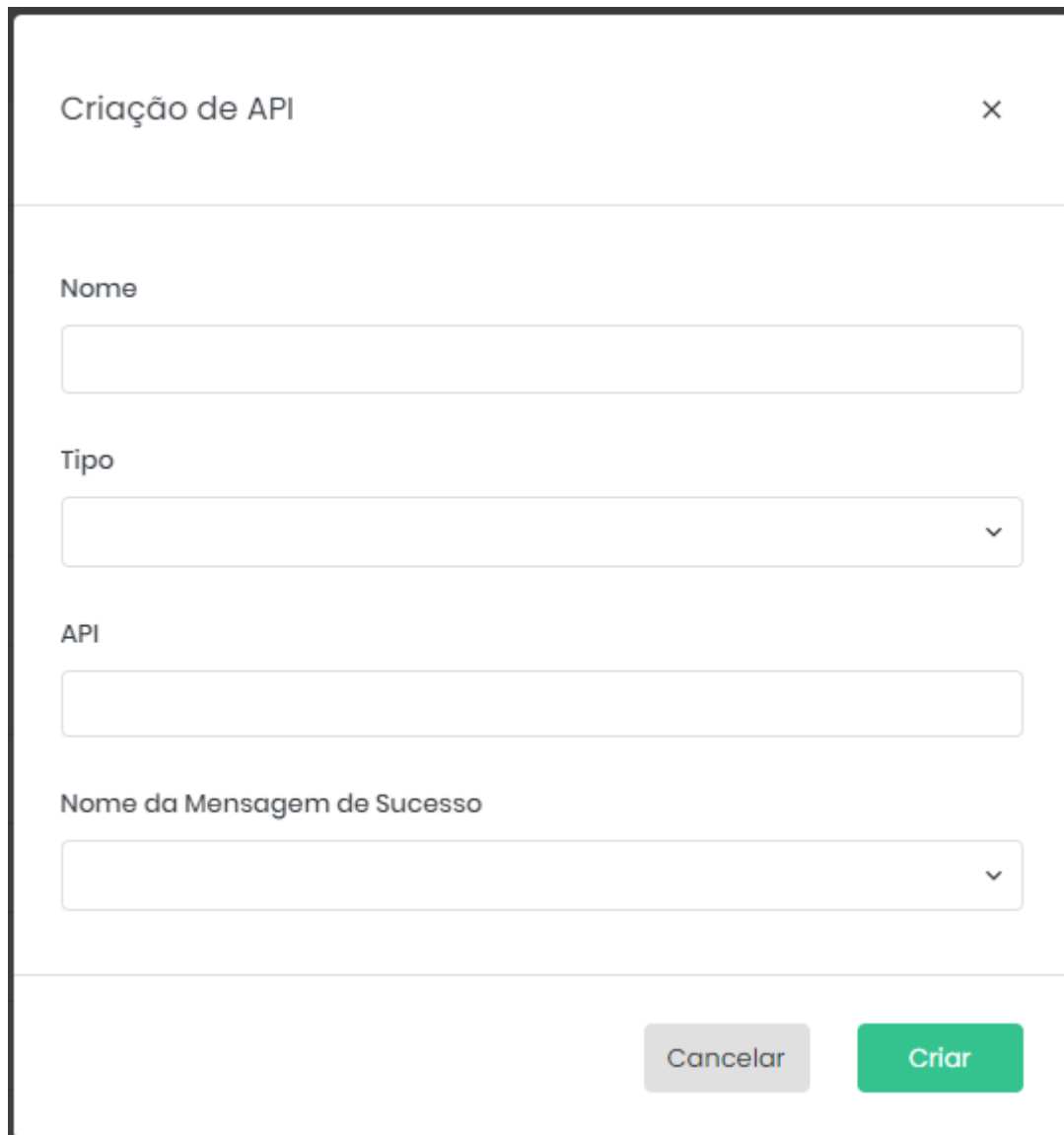
A caixa de diálogo intitulada "Criação de API" possui um botão de fechar "X" no canto superior direito. O formulário contém quatro campos: "Nome" (campo de texto), "Tipo" (menu suspenso), "API" (campo de texto) e "Nome da Mensagem de Sucesso" (menu suspenso). No rodapé, há dois botões: "Cancelar" (cinza) e "Criar" (verde).

Fig. 11 Caixa de diálogo com formulário de criação de API

Passando à página de detalhe de uma API[7] segui o mesmo formato que usei previamente no detalhe de navegação, colocando um cabeçalho com a informação presente na página principal e as opções correspondentes. Quanto à lista de dados ao invés de só utilizar uma aba como no caso do detalhe da navegação, utilizei 4, pois as APIs[7] têm mais listas de dados necessários, os Inputs, Outputs, Mensagens de Erro e ChildApis.

Os Inputs foram o mais simples sendo apenas uma lista simples de dados e respetivo formulário, os Outputs o mesmo mas com a diferença de que alguns dados poderiam ser do tipo array[12], os quais no caso teriam uma opção de detalhe junto da edição e eliminação que iria abrir uma nova página com a lista de dados nesse array permitindo modificar a lista, as Mensagens de Erro poderiam

existir ou não, e se existem podem ter códigos de erro associados, logo fiz de que forma a que quando um dado da mensagem de erro fosse inserido então aparecesse uma lista para inserção de códigos de erro caso seja o que o utilizador pretenda pois os códigos estão associados a uma mensagem de erro, como visualizado na Fig. 12, e finalmente quanto às ChildApis que são APIs[7] dependentes da API[7] principal de onde foi aberta a página detalhe, foi feita uma lista simples com uma caixa de diálogo, no entanto, as APIs[7] são selecionadas na caixa de diálogo da lista de APIs[7] como já feito em outros formulários neste projeto apesar de ter dado alguns problemas no Middleware devido à composição que os dados necessitariam de ter na base de dados.

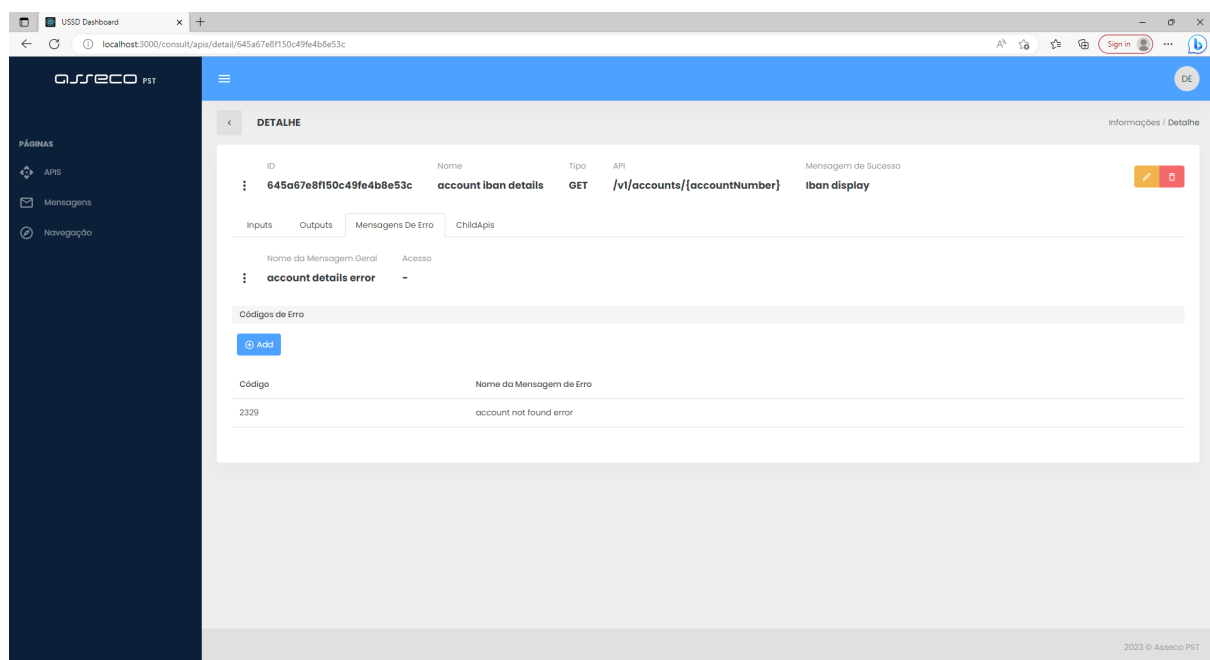


Fig. 12 Página de detalhe de uma API na aba de Mensagens de Erro

5.6. Notificações

Depois de estar acabado o principal do projeto que eram as páginas decidi criar uma forma de dizer ao utilizador o que se passa sempre que houver algum problema ou então quando uma ação foi bem sucedida, para isso criei um método de verificação que recebe a resposta do Middleware de modo a saber se a operação foi bem sucedida, o que em caso afirmativo usa uma notificação com mensagem a dizer que a ação foi bem sucedida, e caso contrário imprime a mensagem de erro personalizada do erro proveniente da operação no Middleware na notificação, estas notificações foram possíveis utilizando o “react-toastify”[26], assim acabando o desenvolvimento do projeto tanto no Front como no Middleware, satisfazendo todos os requisitos necessários para este projeto.

6. Avaliação

Depois de vários testes ao decorrer do desenvolvimento do projeto e no fim do mesmo, posso afirmar que o projeto está completamente funcional e que teve uma excelente performance sem lag[27] ou qualquer atraso ou impedimento proveniente do código, no entanto, as notificações infelizmente não funcionam nas listas das páginas de detalhe, o que depois de confirmar com colegas da equipa tratava-se de um detalhe da forma como as listas estavam personalizadas na estrutura de referência que impede o aparecimento da mesma apesar de estar funcional mas o problema já estaria a ser solucionado pelos desenvolvedores da empresa.

7. Discussão

Dado o resultado final, podemos compreender que o problema para o qual este projeto foi desenvolvido para solucionar foi resolvido com sucesso, ainda com funcionalidades extra com objetivo de melhorar a visualização e compreensão dos dados por parte do utilizador, assim tendo todos os requisitos e funcionalidades corretamente implementados e funcionais. Também podemos concluir que a escolha das tecnologias foi correta, dado o resultado final do projeto e a sua boa performance.

Esta ferramenta para visualização dos dados e criação e manutenção dos menus será imprescindível para a criação dos diferentes menus por parte do desenvolvedor, simplificando a tarefa e criando um ambiente mais fácil de usar para o mesmo.

O desenvolvimento desta ferramenta não teve grandes percalços nem nenhum grande problema que demorasse dias a resolver, no entanto, por não ter muita experiência com Middleware, precisei de mais algum tempo no início do projeto a pesquisar sobre modelos de dados do MongoDB[11], enquanto os criava, e ao decorrer do projeto tive também alguns erros derivados de incompatibilidades de dados, onde tive que por exemplo aprender a manipular IDs de objetos, objetos e arrays[12] do MongoDB[11], tanto na base de dados como no middleware em que nalgumas situações foi necessário realizar a organização e formatação de dados no Middleware para então enviar corretamente para a base de dados, pois a mesma se recebesse os dados no formato original ficaria com formatos mais complexos, o que seria indesejado.

Assim concluído o projeto e estágio, posso afirmar que foi uma mais valia para a minha experiência profissional, porque aprendi sobre novos formatos e tecnologias que de outra forma poderia não ter aprendido ou até aprendido mas com mais dificuldade e sem suporte de profissionais na área.

8. Conclusão

Este estágio foi uma oportunidade para aprender sobre novas tecnologias e aperfeiçoar os meus conhecimentos na Asseco PST Funchal. Posso afirmar que foi uma boa experiência e que o projeto desenvolvido neste estágio teve vários pontos interessantes e de aprendizado no seu desenvolvimento, desde aprender sobre USSD, aprender a utilizar o Swagger[19], criar os serviços CRUD[6], aperfeiçoar os meus conhecimentos em Javascript[14] e na sua framework[17] React[15], criar formas de visualizar dados e pensar sobre qual a melhor apresentação para o utilizador da ferramenta, aperfeiçoar os meus conhecimentos em formulários e formatos dos mesmos como no caso das caixas de diálogo que se provaram ser uma solução interessante e agradável, treinar a minha capacidade de resolução de problemas quando havia alguma incompatibilidade de formato de dados que tinha de ser resolvida para a base de dados ficar exatamente como pretendido e pude também trabalhar o meu trabalho em contexto de equipa num ambiente profissional.

No fim do estágio pude avaliar que o projeto estava funcional, sem erros, atrasos ou requisitos não implementados. Com mais tempo poderia ter adicionado ainda outra funcionalidade extra em que se conseguiria criar uma mensagem no formulário das navegações ou APIs[7], de modo a que não houvesse necessidade de estar sempre a voltar atrás e a ir à página de mensagens para criar a mesma, ou começar por criar uma mensagem sem intenção clara de como ficaria uma navegação ou API[7], assim facilitando a criação de dados sem ter de estar sempre a passar de página em página como os dados estão todos interligados.

Para concluir, foi uma boa experiência em contexto de trabalho tanto na empresa com o meu orientador João Drummond Sousa e a sua equipa a guiar-me no projeto sempre disponíveis a ajudar, como por parte da universidade com o meu professor orientador Filipe Quintal também sempre disponível para dúvidas e a acompanhar o meu progresso na empresa.

9. Referências

- [1] «Asseco PST- member of Asseco Group», *Asseco PST*.
<https://www.pst.asseco.com> (acedido 14 de julho de 2023).
- [2] «What is Blockchain Technology? - IBM Blockchain | IBM». <https://www.ibm.com/topics/blockchain> (acedido 17 de julho de 2023).
- [3] «What is Machine Learning? | IBM». <https://www.ibm.com/topics/machine-learning> (acedido 17 de julho de 2023).
- [4] «Tech Recruiting: Frontend, Backend, Middleware, Full-stack, Server Side and Client Side Programming». <https://www.linkedin.com/pulse/tech-recruiting-frontend-backend-middleware-server-side-pandey> (acedido 14 de julho de 2023).
- [5] «Back Office Application», *Techopedia*, 24 de outubro de 2012.
<https://www.techopedia.com/definition/1406/back-office-application> (acedido 14 de julho de 2023).
- [6] «What is CRUD?», *Codecademy*.
<https://www.codecademy.com/article/what-is-crud> (acedido 14 de julho de 2023).
- [7] «What is an API?». <https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces> (acedido 14 de julho de 2023).
- [8] baeldung, «A Guide to the ResourceBundle | Baeldung», 11 de março de 2018.
<https://www.baeldung.com/java-resourcebundle> (acedido 14 de julho de 2023).
- [9] «What is USSD (Unstructured Supplementary Service Data)?», *Networking*.
<https://www.techtarget.com/searchnetworking/definition/USSD> (acedido 14 de julho de 2023).
- [10] «SMS & USSD Banking - INM». <http://www.inm.pt/pt/produtos/sms-ussd-banking> (acedido 14 de julho de 2023).
- [11] «O Que É O MongoDB?», *MongoDB*.
<https://www.mongodb.com/pt-br/what-is-mongodb> (acedido 14 de julho de 2023).
- [12] «What is Array?», *GeeksforGeeks*, 27 de outubro de 2017.
<https://www.geeksforgeeks.org/what-is-array/> (acedido 14 de julho de 2023).
- [13] «Arquiteturas de Referência e sua importância». <https://pt.linkedin.com/pulse/arquiteturas-de-refer%C3%A2ncia-e-sua-import%C3%A2ncia-carlos-mattos> (acedido 14 de julho de 2023).
- [14] «What is JavaScript? - Learn web development | MDN», 3 de julho de 2023.
https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript (acedido 14 de julho de 2023).
- [15] «React – A JavaScript library for building user interfaces». <https://legacy.reactjs.org/> (acedido 14 de julho de 2023).
- [16] «About», *Node.js*. <https://nodejs.org/en/about> (acedido 14 de julho de 2023).
- [17] C. Team, «What Is a Framework?», *Codecademy Blog*, 23 de setembro de 2021. <https://www.codecademy.com/resources/blog/what-is-a-framework/> (acedido 14 de julho de 2023).
- [18] «Express - Node.js web application framework». <https://expressjs.com/> (acedido 14 de julho de 2023).
- [19] «What is Swagger». <https://swagger.io/docs/specification/2-0/what-is-swagger/> (acedido 14 de julho de 2023).
- [20] «Angular - What is Angular?». <https://angular.io/guide/what-is-angular>

- (acedido 14 de julho de 2023).
- [21] «What is a query? Explore database queries and more». <https://www.techtarget.com/searchdatamanagement/definition/query> (acedido 14 de julho de 2023).
- [22] «What is a dashboard? Definitions and uses | Adjust». <https://www.adjust.com/glossary/dashboard/> (acedido 14 de julho de 2023).
- [23] «React Hook Form - A Complete Guide», *Hygraph*, 27 de setembro de 2022. <https://hygraph.com/blog/react-hook-form> (acedido 14 de julho de 2023).
- [24] B. Hick, «yup: Validações no React de uma forma muito simples», *Medium*, 4 de setembro de 2021. <https://bradhick.medium.com/yup-valida%C3%A7%C3%B5es-no-react-de-uma-forma-muito-simples-700c039114e3> (acedido 14 de julho de 2023).
- [25] «Overview - Material UI». <https://mui.com/material-ui/getting-started/> (acedido 14 de julho de 2023).
- [26] K. Lodha, «React Toastify», *Scaler Topics*, 3 de março de 2023. <https://www.scaler.com/topics/react/react-toastify/> (acedido 14 de julho de 2023).
- [27] «Lag», *Techopedia*, 8 de novembro de 2011. <https://www.techopedia.com/definition/17182/lag-gaming> (acedido 14 de julho de 2023).