****

**INSTITUTO POLITÉCNICO DE LEIRIA**

Carbuddy

RELATÓRIO DE projeto em sistemas de informação

Relatório de Projeto em Sistemas de Informação para cumprimento dos requisitos necessários à realização da prova de apresentação de projeto do Curso Técnico Superior Profissional (TeSP) de **Programação de Sistemas de Informação** realizado sob a orientação de Marco Vicente

Andreia Agostinho Batista nº 2202415

diogo Rafael Cunha Pereira nº2201126

gonçalo santos ferreira nº2201131

Programação em Sistemas de informação

DATA

**DECLARAÇÃO**

Declaro que este Relatório se encontra em condições de ser apreciada (o) pelo júri a designar.

Andreia Agostinho Batista,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Leiria, .... de Fevereiro de 2022

Relatório de Projeto em Sistemas de Informação para cumprimento dos requisitos necessários à realização da prova de apresentação de projeto do Curso Técnico Superior Profissional (TeSP) de **Programação de Sistemas de Informação** realizado sob a orientação de Marco Vicente

**DECLARAÇÃO**

Declaro que este Relatório se encontra em condições de ser apreciada (o) pelo júri a designar.

Diogo Rafael Cunha Pereira,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Leiria, .... de Fevereiro de 2022

Relatório de Projeto em Sistemas de Informação para cumprimento dos requisitos necessários à realização da prova de apresentação de projeto do Curso Técnico Superior Profissional (TeSP) de **Programação de Sistemas de Informação** realizado sob a orientação de Marco Vicente

**DECLARAÇÃO**

Declaro que este Relatório se encontra em condições de ser apreciada (o) pelo júri a designar.

Gonçalo Santos Ferreira,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Leiria, .... de Fevereiro de 2022

# Agradecimentos

Para que todo o desenvolvimento deste projeto fosse possível, queríamos deixar um claro agradecimento aos docentes das unidades curriculares do presente ano letivo, sendo eles, Mário Viana, Mário Fernandes e Marco Vicente. Aos restantes elementos envolvidos no projeto, um muito obrigado pelo apoio e ajuda.

Sem os elementos referidos anteriormente, não seria possível desenvolver este projeto com tanta estabilidade e calma. Um muito obrigado a todos eles.

# Resumo

**RELATÓRIO DE PROJETO - CarBuddy**

**Andreia Agostinho Batista**

**Diogo Rafael Cunha Pereira**

**Gonçalo Santos Ferreira**

Em contexto das cadeiras de Plataformas de Sistemas de Informação, Serviços e Interoperabilidade de Sistemas, Acesso Móvel a Sistemas de Informação e Projeto em Sistemas de Informação, foi proposto aos alunos que desenvolvessem um projeto em grupo, projeto esse que irá ser orientado e gerido em concordância entre estas quatro.

O projeto é de tema único que engloba todas e este mesmo é de livre escolha, tal como também se pretende que o desenvolvimento do mesmo utilize uma metodologia ágil.

Todo o projeto será desenvolvido com a separação de tarefas em sprints a serem geridas pelo grupo.

Pode-se referir também o facto de que o projeto tem como principal objetivo a consolidação e aplicação dos conhecimentos adquiridos ao longo do curso.

Por fim, relativamente ao tema do nosso projeto, decidimos implementar um conceito de sistema de gestão e reparação de veículos, que compreende uma combinação de três componentes, sendo eles um website, onde é feita toda a gestão de contas de utilizador, onde vão ser vistos e/ou registados os veículos, o seu estado e as suas reparações, uma aplicação, onde cada cliente, iniciando sessão, tem listadas todas as reparações, dados dos respetivos veículos e o colaborador/mecânico, tal como quando inserindo o VIN de um veículo, acede ao histórico de reparações e aos dados do respetivo veículo. Uma API, para ser possível fazer uma interligação de dados entre ambos os componentes referidos anteriormente.

O objetivo primordial, é que mesmo a nível internacional, se possam registar e verificar o histórico de reparações, utilizando o VIN.

PALAVRAS-CHAVE: gestão, veículos, website, aplicação, API.

# Índices

Índice

[Agradecimentos v](#_Toc95140335)

[Resumo vi](#_Toc95140336)

[Índices vii](#_Toc95140337)

[1. Introdução 2](#_Toc95140338)

[2. Metodologia 4](#_Toc95140339)

[3. Arquitetura do Sistema 8](#_Toc95140340)

[4. Gestão do Projeto 13](#_Toc95140341)

[5. Análise 14](#_Toc95140342)

[6. Desenho 23](#_Toc95140343)

[7. Implementação 24](#_Toc95140344)

[8. Testes 26](#_Toc95140345)

[9. Conclusão e trabalho futuro 31](#_Toc95140346)

[10. Bibliografia e Webgrafia 32](#_Toc95140347)

[11. Anexos 33](#_Toc95140348)

Índice de Ilustrações

[Figura 1 - Divisão de tarefas no Jira 5](#_Toc95140349)

[Figura 2 - Exemplo do Git (Commit's) 6](#_Toc95140350)

[Figura 3 - Calendarização do Projeto 7](#_Toc95140351)

[Figura 4 - Modelo Relacional da Base de Dados 8](#_Toc95140352)

[Figura 5 - Exemplo da UI da Aplicação Móvel 9](#_Toc95140353)

[Figura 6 - Exemplo da UI da Plataforma WEB 10](#_Toc95140354)

[Figura 7 - Exemplo de EndPoint da API 12](#_Toc95140355)

[Figura 8 - Configuração dos Testes Unitários 27](#_Toc95140356)

[Figura 9 - Desenvolvimento de um Teste Unitário 27](#_Toc95140357)

[Figura 10 - Configuração dos Testes Funcionais 28](#_Toc95140358)

[Figura 11 - Desenvolvimento de um Teste Funcional 28](#_Toc95140359)

[Figura 12 - Configuração dos Testes de Aceitação 29](#_Toc95140360)

[Figura 13 - Desenvolvimento de um Teste de Aceitação 29](#_Toc95140361)

[Figura 14 - Execução dos Testes 30](#_Toc95140362)

[Figura 15 - Exemplo de Erro nos Testes 30](#_Toc95140363)

Índice de Tabelas

[Tabela 1 - Tabela de Roles do RBAC 11](#_Toc95140364)

[Tabela 2 - Tabela de Rules do RBAC 11](#_Toc95140365)

[Tabela 3 - Requisitos Funcionais 14](#_Toc95140366)

[Tabela 4 - Requisitos não Funcionais de Usabilidade 15](#_Toc95140367)

[Tabela 5 - Requisitos não Funcionais de Fiabilidade 16](#_Toc95140368)

[Tabela 6 - Requisitos não Funcionais de Segurança 17](#_Toc95140369)

[Tabela 7 - Requisitos não Funcionais de Eficiência 18](#_Toc95140370)

[Tabela 8 - Requisitos não Funcionais de Disponibilidade 19](#_Toc95140371)

[Tabela 9 - Requisitos não Funcionais de Ambiente 20](#_Toc95140372)

[Tabela 10 - Requisitos não Funcionais de Desenvolvimento 21](#_Toc95140373)

Índice de Anexos

[Anexo 1 - Relatório de PlatSI 33](#_Toc95140374)

[Anexo 2 - Relatório de SIS 34](#_Toc95140375)

[Anexo 3 - Relatório de AMSI 35](#_Toc95140376)

# Introdução

Apresentação do trabalho

Em contexto das cadeiras de Plataformas de Sistemas de Informação, Serviços e Interoperabilidade de Sistemas, Acesso Móvel a Sistemas de Informação e Projeto em Sistemas de Informação, foi proposto aos alunos que desenvolvessem um projeto em grupo, projeto esse que irá ser orientado e gerido em concordância entre estas quatro.

O projeto é de tema único que engloba todas e este mesmo é de livre escolha, tal como também se pretende que o desenvolvimento do mesmo utilize uma metodologia ágil.

Todo o projeto será desenvolvido com a separação de tarefas em sprints a serem geridas pelo grupo.

Pode-se referir também o facto de que o projeto tem como principal objetivo a consolidação e aplicação dos conhecimentos adquiridos ao longo do curso.

Relativamente ao tema do nosso projeto, decidimos implementar um conceito de sistema de gestão e reparação de veículos, que compreende uma combinação de três componentes, sendo eles um website, onde é feita toda a gestão de contas de utilizador, onde vão ser vistos e/ou registados os veículos, o seu estado e as suas reparações, uma aplicação, onde cada cliente, iniciando sessão, tem listadas todas as reparações, dados dos respetivos veículos e o colaborador/mecânico, tal como quando inserindo o VIN de um veículo, acede ao histórico de reparações e aos dados do respetivo veículo. Uma API, para ser possível fazer uma interligação de dados entre ambos os componentes referidos anteriormente.

O objetivo primordial, é que mesmo a nível internacional, se possam registar e verificar o histórico de reparações, utilizando o VIN.

Estrutura do Relatório

Relativamente à estrutura deste documento, podemos observar que este está dividido em 9 pontos, sendo eles a Introdução, a Metodologia, a Arquitetura do Sistema, Gestão do Projeto, Análise, Desenho, Implementação, Testes, e por fim, Conclusão e Trabalho Futuro.

Quanto à Metodologia, podemos referir que falamos a cerca da nossa metodologia de trabalho e de como distribuímos as tarefas entre o grupo.

Na Arquitetura do Sistema trabalhamos os conceitos de componentes tecnológicas utilizadas.

Relativamente à Gestão do Projeto, vamos explicar o nosso método de funcionamento, onde foi adotada uma metodologia ágil, sendo bastante flexível, permitindo contornar eventuais contratempos.

No ponto referente à Análise, tal como está implícito no nome deste mesmo ponto, é feita uma análise pormenorizada dos requisitos e funcionalidades desenvolvidas, bem como dos principais objetivos do projeto.

Apresentamos os Wireframes e Mockups realizados pela equipa no ponto de Desenho, em que também referimos os detalhes e fundamentamos as opções tomadas e as alterações que realizámos para que pudéssemos ao máximo aprimorar o conceito do projeto que escolhemos. Neste ponto também é feita uma comparação entre os desenhos referidos anteriormente, com o resultado final das plataformas desenvolvidas.

Recorremos ao ponto da Implementação para explicar todo o processo de desenvolvimento do software.

No tema dos testes, vamos mostrar todos os testes desenvolvidos ao sistema, de modo a procurar erros e anomalias no código, visando a melhoria do código.

# Metodologia

Durante o período de análise da temática do projeto, o grupo decidiu pensar na melhor forma de abordar o desenvolvimento programático do projeto. Assim, decidimos abordar uma metodologia ágil, que por sua vez é mais flexível.

A metodologia ágil destaca-se pelos seguintes elementos:

* Desenvolvimento incremental e respetiva melhoria contínua,
* Cooperação entre equipa e cliente (ciclo de [feedback](https://neilpatel.com/br/blog/feedback-o-que-e/) constante),
* Entregas rápidas e de alta qualidade,
* Flexibilidade de desenvolvimento do projeto,
* Adaptabilidade às mudanças;

Tendo em conta as vantagens da metodologia ágil discriminadas anteriormente, o grupo destaca um acontecimento real durante o período de desenvolvimento do projeto. Durante essa fase importantíssima do projeto, dois dos elementos do grupo testaram positivo ao covid-19, dificultando o desenvolvimento do projeto, por várias variantes, por causas motoras, físicas e mentais. Felizmente, ambos os casos foram assintomáticos, tornando mais fácil contornar a situação. No entanto, foi importante adotar esta metodologia, pois se molda a situações imprevisíveis.

Ao adotar esta metodologia, o grupo achou, por bem, criar reuniões todos os dias úteis de desenvolvimento, no inicio e no fim do dia. Decidimos também adotar o Git, para a partilhar de código e respetiva documentação e ainda o Jira, para a criação e gestão de todo o tempo útil de desenvolvimento e as respetivas funcionalidades a desenvolver.

Gestão de Tarefas

Para gerir as tarefas a realizar, utilizámos um repositório do Jira Software. Torna-se mais fácil gerir o projeto, sendo que podemos ver o que falta fazer, o que está em andamento e o que está concluído. Assim fizemos uma divisão de tarefas e distribuímo-las pelos elementos do grupo.

Como podemos ver na seguinte imagem, as tarefas encontram-se listadas e divididas entre todos os membros do grupo no repositório Jira.

Uma imagem com texto

Descrição gerada automaticamente

Figura 1 - Divisão de tarefas no Jira

GIT

Como podemos ver na figura abaixo, podemos observar que, todos os membros do grupo fizeram commit’s regularmente, podendo dizer assim, que todos os membros do grupo aproveitaram da melhor forma o seu tempo útil de desenvolvimento.

Foi com esta organização e divisão de tarefas que foi possível desenvolver um projeto com estas três vertentes, sendo elas, API, APP e WEB.

Uma imagem com texto

Descrição gerada automaticamente

Figura 2 - Exemplo do Git (Commit's)

Calendarização do Projeto

A nível de calendarização estava previsto e foi cumprido, que o grupo durante o mês de outubro analise o problema, faça uma análise de mercado e faça a realização de mockup’s estruturais do projeto. Em novembro e dezembro o objetivo passa pelo desenvolvimento programático do projeto, complementando-se com correções de erros e bugs e com a escrita do relatório do projeto.

Por fim, em janeiro, o objetivo passa pela entrega do respetivo projeto, com toda a sua documentação.

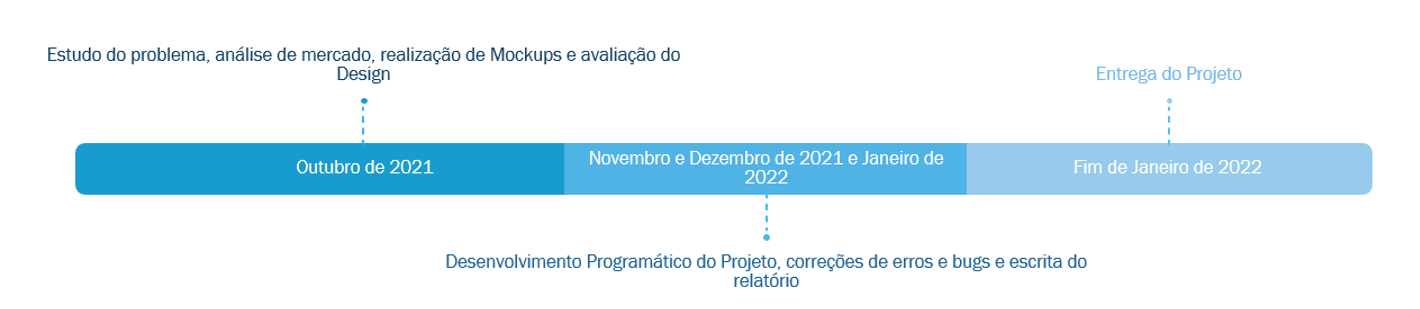


Figura 3 - Calendarização do Projeto

# Arquitetura do Sistema

Base de Dados

Para que fosse possível armazenar os dados de uma forma centralizada, foi essencial e extremamente importante planear o modelo de uma base de dados. Assim o nosso modelo é composto por 11 entidades. 5 dessas entidades, criadas para o sistema RBAC. As restantes 6 entidades foram criadas e idealizadas pelo grupo, com o intuito de guardar dados. É de salientar que a base de dados foi desenvolvida em MySQL, planeada e estruturada de acordo com as suas normas.

A base de dados, podemos afirmar que, das 6 entidades desenvolvidas pelo grupo, apresenta uma estrutura adequada e organizada. Assim, temos a tabela de utilizadores, a tabelas carros que se associa a um utilizador, a tabela reparações que se associa a um carro e a um contribuidor, a tabela de agendamentos que se associa a um carro e a uma empresa e a tabela contribuidores que se associa a um utilizador e a uma empresa.

Uma imagem com mesa

Descrição gerada automaticamente

Figura 4 - Modelo Relacional da Base de Dados

Aplicação Móvel

A aplicação móvel desenvolvida é exclusivamente dedicada e direcionada ao cliente e interage diretamente com a API do projeto, que por sua vez interage com a base de dados do projeto.

A aplicação móvel foi desenvolvida em Java no Android Studio. Para a interação entre a aplicação e a API, foi utilizada a tecnologia Volley, que se trata de uma biblioteca HTTP que facilita a criação de redes para apps para Android de maneira mais rápida. Utilizando o volley, conseguimos facilmente aceder à API criada para este propósito e garantir a transmissão de dados entre a base de dados da API e o singleton da aplicação móvel.

Com o fluxo de dados constante, foi necessário utilizar a tecnologia singleton, que é um padrão de desenvolvimento de software. Este padrão garante a existência de apenas uma instância de uma classe, mantendo um ponto global de acesso ao seu objeto. Utilizando o singleton, garantimos a coesão dos dados e a eficiência da aplicação móvel.

Uma imagem com texto, eletrónica, preto

Descrição gerada automaticamente

Figura 5 - Exemplo da UI da Aplicação Móvel

Plataforma web

A plataforma WEB desenvolvida é dedicada ao administrador no back-office e dedicada a clientes e colaboradores no front-office. A plataforma foi desenvolvida em PHP utilizando a framework YII, que demonstrou ter os essenciais componentes ao desenvolvimento de uma plataforma WEB. Este componente do projeto interage diretamente com a base de dados do projeto. Cada operação de CRUD, em qualquer tabela, é feita diretamente na base de dados, sem passar por algum endpoint da API.

É ainda importante salientar um importante componente deste componente do projeto, o RBAC, que permite gerir todos os acessos a determinadas funcionalidades da plataforma.

O RBAC (Role-based Acess Control) é uma abordagem de segurança para restringir o acesso de utilizadores a uma dada funcionalidade. Assim o RBAC permite ou não a um utilizador aceder a uma dada página da nossa plataforma. Assim sendo, realizamos duas tabelas, uma para as Roles e outra para as Rules da configuração do nosso RBAC. Podemos também afirmar que, os administradores e colaboradores têm acesso a funcionalidades exclusivas, no entanto tanto os administradores, os colaboradores e os clientes têm acesso às funcionalidades do cliente.

Uma imagem com texto, eletrónica

Descrição gerada automaticamente

Figura 6 - Exemplo da UI da Plataforma WEB

Tabela 1 - Tabela de Roles do RBAC

|  |  |  |
| --- | --- | --- |
| Nome | Descrição | ID |
| admin | Tipo de utilizador utilizado para o Administrador | **1** |
| collaborator | Tipo de utilizador utilizado para o Colaborador | **2** |
| client | Tipo de utilizador utilizado para o Cliente | **3** |

Tabela 2 - Tabela de Rules do RBAC

|  |  |
| --- | --- |
| Nome | Descrição |
| backendCrudCompany | Rule associada ao Administrador, que permite aceder a todo o CRUD de empresas do backend |
| backendCrudContributor | Rule associada ao Administrador, que permite aceder a todo o CRUD de contribuidores do backend |
| backendCrudUser | Rule associada ao Administrador, que permite aceder a todo o CRUD de utilizadores do backend |
| frontendCrudVehicle | Rule associada ao Cliente, que permite aceder a todo o CRUD de veículos do frontend |
| frontendCrudRepair | Rule associada ao Colaborador, que permite aceder a todo o CRUD de reparações do frontend |
| frontendCurdScheculesCollaborator | Rule associada ao Colaborador, que permite aceder a todo o CRUD de gestão de agendamentos do frontend |
| frontendCurdScheculesClient | Rule associada ao Cliente, que permite aceder a todo o CRUD de agendamentos do frontend |
| frontendReadRepair | Rule associada ao Cliente, que permite apenas ler as reparações no frontend |

Api

Este é, provavelmente, o componente mais importante do projeto e é aquele que o cliente não sabe e não nota que existe. A API foi também ela desenvolvida em PHP utilizando a framework YII, que demonstrou ter os essenciais componentes ao desenvolvimento de uma API. É de salientar que a API se encontra dentro de um modulo do front-office da plataforma WEB. O objetivo deste componente é, essencialmente, transportar dados para a aplicação móvel, receber informações, atualização e todo o tipo de CRUD disponível na aplicação móvel.

Neste componente, é importante salientar que foram criadas muitas ações customizáveis para tornar a interação entre a aplicação móvel e o utilizador mais fácil. Assim, podemos dar o exemplo de uma ação customizada, quando o cliente adiciona um carro. O cliente ao adicionar o seu carro, teria de preencher na entidade carro o campo da chave estrangeira entre a ligação de entidades carro e utilizador. Assim, com uma ação customizada, esse processo é automático, facilitando a ação ao cliente e garantido a congruência dos dados para o programador.

Uma imagem com texto

Descrição gerada automaticamente

Figura 7 - Exemplo de EndPoint da API

# Gestão do Projeto

Neste capítulo deve ser apresentado o plano do Projeto (Sugestão: utilizar a ferramenta de Gestão de projetos Redmine):

* Gantt Chart;
* Identificação das principais actividades;
* Deliverables;
* Milestones;
* Equipa de Projeto e funções respetivas.

# Análise

Para proceder ao desenvolvimento do projeto, foi essencial realizar um estudo de requisitos do sistema. Assim, foi realizado um estudo com a finalidade de chegar às funcionalidades essenciais do sistema a desenvolver.

Assim, para esse efeito, foram desenvolvidas tabelas, subdivididas em requisitos funcionais e não funcionais.

Tabela 3 - Requisitos Funcionais

|  |  |  |
| --- | --- | --- |
| # ID | Descrição | Prioridade |
| RF-01 | O sistema deverá obter os seus dados pela API desenvolvida em SIS e pela base de dados | **Alta** |
| RF-02 | O sistema deverá ter Login para todos os utilizadores | **Alta** |
| RF-03 | O sistema deverá ter uma ferramenta de gestão de veículos para os colaboradores | **Alta** |
| RF-04 | O sistema deverá ter uma página onde conste os detalhes de um veiculo selecionado | **Alta** |
| RF-05 | O cliente deverá ter acesso a um activity onde conste os detalhes de um veículo selecionado e as respetivas manutenções e revisões | **Alta** |
| RF-06 | O sistema deverá confirmar todas as ações antes de as executar | **Média** |
| RF-07 | O sistema deverá ter uma resposta de no máximo 2 segundos por cada ação | **Alta** |
| RF-08 | O design do sistema deverá ser simples, agradável e intuitivo | **Alta** |
| RF-09 | O design do sistema deverá ser idêntico ao design do sistema desenvolvido para a unidade curricular de PlatSI | **Média** |

**Requisitos Não Funcionais de Usabilidade (Usability)** – foca-se na perspetiva da interação com o utilizador (facilidade de uso).

Tabela 4 - Requisitos não Funcionais de Usabilidade

|  |  |  |
| --- | --- | --- |
| # ID | Descrição | Prioridade |
| RNF-USA-01 | Atividades - A aplicação deverá ter, no mínimo, 6 atividades e um máximo recomendado de 10 atividades; | **Alta** |
| RNF-USA-02 | Todas as páginas da aplicação devem ser consistentes - os elementos devem ser organizados de forma semelhante entre páginas; | **Média** |
| RNF-USA-03 | Todas as páginas da aplicação devem ser user-friendly – Devem ser fáceis de usar e apresentar a informação com o número de cliques mínimo; | **Alta** |
| RNF-USA-04 | O layout das páginas da aplicação deve ser intuitivo – O layout deve ser apresentado de forma a dar a entender ao utilizador as ações disponíveis através de elementos como botões, mensagens ou erros; | **Média** |
| RNF-USA-05 | O conteúdo deve ser mostrado de uma forma lógica e natural; | **Alta** |
| RNF-USA-06 | Fragmentos - Algumas funcionalidades têm de ser implementadas/apresentadas através de fragmentos; | **Alta** |
| RNF-USA-07 | CRUD - No mínimo, 1 das atividades implementadas deve permitir introdução de dados pelo utilizador; | **Alta** |
| RNF-USA-08 | Os utilizadores devem conseguir navegar sempre para a homepage de qualquer parte da aplicação; | **Média** |

**Requisitos Não Funcionais de Fiabilidade (Reliability)** – Descreve-se como a habilidade do software se comportar de forma consistente e aceitável para o utilizador, sendo também a probabilidade e percentagem do software funcionar sem falhas para um período específico ou uso.

Tabela 5 - Requisitos não Funcionais de Fiabilidade

|  |  |  |
| --- | --- | --- |
| # ID | Descrição | Prioridade |
| RNF-FIA-01 | A aplicação deve ser o mais modular possível, para que seja fácil a adição de novas funcionalidades; | **Média** |
| RNF-FIA-02 | A aplicação deve suportar atualizações; | **Alta** |
| RNF-FIA-03 | A aplicação deve estar apta a adaptar-se ao aumento do uso (usage) e a lidar com mais dados à medida que o tempo progride; | **Alta** |
| RNF-FIA-04 | CRUD - No mínimo, 4 das funcionalidades implementadas têm de aceder ao webservice para realização das operações CRUD sobre os dados; | **Alta** |
| RNF-FIA-05 | A aplicação deve ter um tempo de resposta baixo (até 3 segundos) e todas as ações devem ser processadas instantaneamente; | **Alta** |
| RNF-FIA-06 | Acesso - O acesso aos recursos do webservice não pode bloquear a interface gráfica; | **Alta** |

**Requisitos Não Funcionais de Segurança (Security)** – Tais requisitos definem como o sistema se encontra protegido ou como deve ser efetuado o acesso ao sistema.

Tabela 6 - Requisitos não Funcionais de Segurança

|  |  |  |
| --- | --- | --- |
| # ID | Descrição | Prioridade |
| RNF-SEG-01 | Comunicações - Tornar todas as comunicações o mais seguras possível; | **Alta** |
| RNF-SEG-02 | Anti-hacking – A aplicação deve estar protegida contra ataques exteriores; | **Alta** |
| RNF-SEG-03 | Cópia de segurança – A aplicação deve ser capaz de guardar uma cópia local das mensagens e ficheiros enviados e recebidos pelos clientes; | **Alta** |

**Requisitos Não Funcionais de Eficiência (Performance)** – definem o quão bem o sistema lida com capacidade, taxa de transferência e tempo de resposta (desempenho, espaço, tempo).

Tabela 7 - Requisitos não Funcionais de Eficiência

|  |  |  |
| --- | --- | --- |
| # ID | Descrição | Prioridade |
| RNF-EFI-01 | Taxa de resposta de envio – Deve conseguir apresentar de imediato as comunicações enviadas pelo cliente; | **Alta** |
| RNF-EFI-02 | Taxa de resposta de receção – Deve conseguir apresentar as comunicações recebidas numa fração de segundos; | **Alta** |
| RNF-EFI-03 | Registo de Dados - É obrigatório o registo de dados em ficheiros locais e em base de dados SQL local; | **Alta** |
| RNF-EFI-04 | CRUD - No mínimo, 1 das atividades implementadas deve permitir introdução de dados pelo utilizador; | **Alta** |

**Requisitos Não Funcionais de Disponibilidade (Availability)** – Estes definem o tempo de funcionamento do sistema, o tempo que leva para reparar uma falha e o tempo entre ciclos.

Tabela 8 - Requisitos não Funcionais de Disponibilidade

|  |  |  |
| --- | --- | --- |
| # ID | Descrição | Prioridade |
| RNF-DIS-01 | Atualizações - A aplicação deve estar sempre pronta a obedecer ao seu normal funcionamento, mesmo na ocorrência de atualizações; | **Alta** |
| RNF-DIS-02 | Erros e Bugs – De maneira a evitar erros e bugs a equipa deve conseguir editar a aplicação a qualquer momento sem grandes oportunos à utilização normal por parte do utilizador; | **Alta** |

**Requisitos Não Funcionais de Ambiente (Portability)** – especificam o ambiente operacional do sistema (capacidade de adaptação).

Tabela 9 - Requisitos não Funcionais de Ambiente

|  |  |  |
| --- | --- | --- |
| # ID | Descrição | Prioridade |
| RNF-AMB-01 | Ambiente I - A aplicação deve correr em sistema operativo Android; | **Alta** |
| RNF-AMB-02 | Ambiente II - Uma aplicação Web composta por Front-office e BackOffice; | **Alta** |
| RNF-AMB-03 | Ambiente III – A aplicação deve correr em sistemas compatíveis com todos os sistemas operativos; | **Alta** |
| RNF-AMB-04 | Base de Dados - Uma base de dados relacional ou NoSQL; | **Alta** |
| RNF-AMB-04 | API - Uma API de serviços HTTP; | **Alta** |

**Requisitos Não Funcionais de Desenvolvimento (Implementation)** – indicam restrições ao **uso de ferramentas** ou linguagens de programação; ou do desenvolvimento do projeto em si; standards.

Tabela 10 - Requisitos não Funcionais de Desenvolvimento

|  |  |  |
| --- | --- | --- |
| # ID | Descrição | Prioridade |
| RNF-DES-01 | **Equipa -** Será valorizada a utilização de planeamento, regras para coordenação entre elementos do grupo e implementação que siga as boas práticas; | **Alta** |
| RNF-DES-02 | **Indentação -** Todos os ficheiros da aplicação devem conter o código devidamente indentado; | **Alta** |
| RNF-DES-03 | **Deadline -** O tempo de desenvolvimento do projeto não deve ultrapassar a data de 24 de Janeiro de 2022; | **Alta** |
| RNF-DES-04 | **Linguagem -** As tecnologias obrigatórias devem consistir em Java, PHP, HMTL; | **Alta** |
| RNF-DES-05 | **Relatório -** No final do projeto deverá ser entregue o respetivo relatório com todos os requisitos desenvolvidos; | **Alta** |
| RNF-DES-06 | **Entrega final I -** No final do projeto deverá ser entregue a aplicação a funcionar devidamente; | **Alta** |
| RNF-DES-07 | **Entrega Final II -** No final do projeto deverá ser entregue o website a funcionar devidamente; | **Alta** |
| RNF-DES-08 | **Entrega Final II -** No final do projeto deverá ser entregue a API a funcionar devidamente. | **Alta** |

Temática e Objetivos Principais

O CarBuddy tem como foco a gestão e reparações de qualquer tipo de veículos a nível internacional. Para que tal aconteça, iremos desenvolver uma plataforma segura onde os clientes poderão agendar os serviços nas empresas registadas e poderão gerir todos os seus veículos próprios. Para que isto tenha efeito, pretende-se desenvolver uma aplicação para facilitar a gestão de todo o tipo de veículos e serviços.

Na aplicação o cliente pode agendar um serviço numa empresa registada na plataforma, indicando a sua data e a sua hora. O cliente na aplicação pode adicionar todos os seus veículos, podendo adicionar de qualquer tipo, tais como motas e carros por exemplo, e de qualquer país. Para além disso, pode agendar uma reparação com uma empresa que esteja registada na nossa plataforma. O cliente pode ainda gerir todos os seus veículos, ver os seus dados e o seu histórico de reparações.

É pretendido uma aplicação na qual os clientes, irão fazer o registo dos seguintes dados de utilizador:

* Nome
* Data de Nascimento
* NIF (que deve ser único)
* Password
* E-mail
* Telemóvel.

Estas informações irão ser armazenadas na sua totalidade na base de dados da aplicação gerida pela empresa, de modo a facilitar o pedido de serviços mais do que uma vez.

É de salientar que existem vários tipos de utilizador, sendo eles o administrador, o cliente e o colaborador, sendo que na aplicação móvel todos os utilizadores são considerados clientes.

# Desenho

Especificar Protótipos, Casos de Uso, Modelo de Dados, Mockup’s, etc.

# Implementação

No decorrer do desenvolvimento do projeto, devem ser especificados o tipo de requisitos e justificar a conformidade com os requisitos obrigatórios estabelecidos na proposta de projeto fornecida pelo docente da cadeira.

Ao analisarmos os objetivos propostos pela proposta de projeto e pelos objetivos práticos que o desenvolvimento da aplicação requer, deparámo-nos com os seguintes requisitos:

* Autenticar-se no servidor fornecendo as credenciais;
* Receber ligações de cliente;
* Guardar a chave pública do cliente;
* Autenticar um utilizador já registado no sistema;
* Validar as assinaturas do cliente;
* Um cliente para dispositivo móvel;
* Uma aplicação Web composta por Front-office e BackOffice;
* Uma base de dados relacional ou NoSQL de suporte às restantes componentes;
* A aplicação deverá ter, no mínimo, 6 atividades (Activity e/ou Fragment) e um máximo recomendado de 10 atividades (Activity e/ou Fragment);
* Algumas funcionalidades têm de ser implementadas/apresentadas através de fragmentos;
* No mínimo, 4 das funcionalidades implementadas têm de aceder ao webservice para realização das operações CRUD (inserir, consultar, alterar e remover) sobre os dados;
* No mínimo, 1 das atividades implementadas deve permitir introdução de dados pelo utilizador;
* No mínimo, 1 das atividades implementadas deve ser construída de forma dinâmica, com os dados obtidos do webservice;
* É obrigatório o registo de dados em ficheiros locais (shared preferences) e em base de dados SQL local (ex.: cache de informação);
* O acesso aos recursos do webservice não pode bloquear a interface gráfica;
* Será valorizada a utilização de planeamento, regras para coordenação entre elementos do grupo e implementação que siga as boas práticas (utilização de padrão MVC, comentários, etc.).

Enquanto grupo, também tivemos algumas ideias, que não são requisitos, mas achamos pertinentes listar, sendo elas:

* Desenvolver a App e o Site com design idênticos,
* Pensar na utilidade da App para o cliente,
* Pensar no futuro e nas valências da App,
* Prever, organizar e dividir as tarefas entre os elementos do grupo,
* Ajudar qualquer um dos membros do grupo, no caso de surgir alguma questão ou dúvida,
* Organizar as ideias de forma sucinta para evitar que surjam conflitos de funcionalidades,
* Realizar pontos de situação todas as semanas,
* Desenvolver o projeto com reuniões semanais,
* Utilização do GitFlow;

Uma vez analisados todos os requisitos do projeto e as suas respetivas componentes, Web, APP e API, o processo de desenvolvimento foi incremental.

Assim, utilizando a tecnologia GitFlow, eram cridos vários branch’s no Git do projeto que visavam uma funcionalidade especifica, não havendo perigo de no momento de fazer merge no repositório houvesse erros.

Para além disso, o grupo reunia-se de todas as manhas e no final de todos os dias, para verificar que todos os membros com as suas respetivas tarefas estavam a ser desenvolvidas consoante o tempo a elas especulado e dedicado.

Por fim, o grupo tentou prever quais as funcionalidades que poderiam causar um eventual atraso e decidiu reforçar o tempo dedicado a essas funcionalidades.

# Testes

Durante o desenvolvimento de um projeto, consoante o desenvolvimento de uma funcionalidade, o programador tem tendência a testar e executar o código desenvolvido. Por vezes, existem erros facilmente detetados e corrigidos e por outras, os erros passam despercebidos.

Neste capitulo, vamos mostrar como foram desenvolvidos todos os testes no projeto, com especial foco na componente Web. Na componente da APP e da API, conforme íamos desenvolvendo o código, íamos testando intensivamente o código. Podemos testar até cerca de 10 vezes a mesma funcionalidade. Na componente web, tratando-se de um requisito do projeto, foi utilizada a framework codeception.

Assim, para a componente Web, foram desenvolvidos três tipos de teste, sendo eles, unitários, funcionais e aceitação.

Testes Unitários

Um teste unitário, visa validar ou invalidar dados inseridos por um dado utilizador. Assim, quando um dado for inválido conseguimos restringir o preenchimento desse campo e avisar o utilizador do erro em questão. Para este efeito, foi necessário configurar os testes unitários da framework codeception, onde foram definidos os componentes utilizados nos testes e a respetiva base de dados.

Uma imagem com texto

Descrição gerada automaticamente

Figura 8 - Configuração dos Testes Unitários

Por fim, passamos ao desenvolvimento de um teste, onde vemos o que é aceito ou não num dado campo e onde fazemos testes de integração à base de dados, verificando assim a interligação entre o componente Web e a base de dados local.

Uma imagem com texto

Descrição gerada automaticamente

Figura 9 - Desenvolvimento de um Teste Unitário

Testes Funcionais

Um teste funcional, visa testar uma funcionalidade de uma respetiva componente, garantindo a qualidade da funcionalidade oferecida ao cliente. Para este efeito, foi necessário configurar os testes funcionais da framework codeception, onde foram definidos os componentes utilizados nos testes e a respetiva base de dados.

Uma imagem com texto

Descrição gerada automaticamente

Figura 10 - Configuração dos Testes Funcionais

Por fim, passamos ao desenvolvimento de um teste, onde verificamos se uma dada funcionalidade é executada corretamente. Este teste pode também ser utilizado para performance, verificando se uma funcionalidade demora muito tempo a ser executada.

Uma imagem com texto

Descrição gerada automaticamente

Figura 11 - Desenvolvimento de um Teste Funcional

Testes de Aceitação

Um teste de aceitação, visa testar uma funcionalidade de uma respetiva componente, garantindo a qualidade da funcionalidade oferecida ao cliente antes do processo de disponibilização do projeto ao público. Para este efeito, foi necessário configurar os testes funcionais da framework codeception, onde foram definidos os componentes utilizados nos testes e a respetiva base de dados. Foi também necessário configurar o componente phpBrowser, o componente dedicado à execução do teste.

Uma imagem com texto

Descrição gerada automaticamente

Figura 12 - Configuração dos Testes de Aceitação

Por fim, passamos ao desenvolvimento de um teste, onde verificamos se uma dada funcionalidade é executada corretamente. Este teste pode também ser utilizado para performance, verificando se uma funcionalidade demora muito tempo a ser executada.

Uma imagem com texto

Descrição gerada automaticamente

Figura 13 - Desenvolvimento de um Teste de Aceitação

Execução dos testeS, Resultados obetidos e correção de erros

Após a execução dos testes realizados, verificamos se os testes passam ou não, e o seu respetivo tempo de execução.

Uma imagem com texto

Descrição gerada automaticamente

Figura 14 - Execução dos Testes

Por fim, alguns testes podem evidenciar um erro. Inicialmente o erro pode vir por falta de configuração de um dado teste, mas numa altura mais avançada dos testes, será mais provável haver algum problema com um dos testes em especifico. A seguinte figura, mostra-nos um exemplo de um erro de um teste unitário, onde na tabela de utilizadores, estava a ser inserida uma chave única em duplicado no campo NIF, erro este resolvido e encontrado através dos testes.

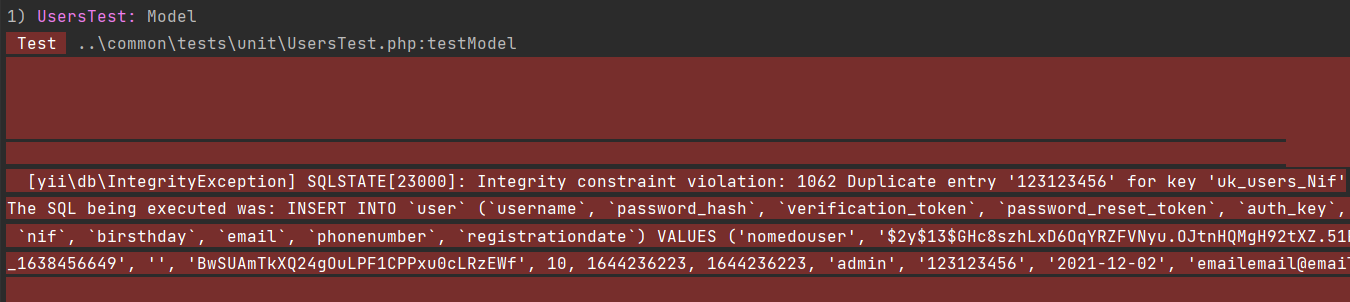


Figura 15 - Exemplo de Erro nos Testes

# Conclusão e trabalho futuro

A primeira parte de um projeto que está a começar é elaborar um plano para o mesmo. Este plano poderá indicar as diversas etapas que são necessárias passar para que este seja concluído com sucesso. No entanto, existe um elemento que não devemos esquecer no momento de elaborar um plano: todos os dias são diferentes. Isto quer dizer que a motivação não será sempre a mesma, que outras atividades poderão surgir, que podem existir contratempos das mais diversas ordens ou até mesmo os nossos estados emocionais não serão sempre os mesmos todos os dias. Por estes motivos, devemos fazer planos flexíveis e que não causem frustração quando alguma das etapas não for cumprida.

Assim, o nosso tema de projeto não foi escolhido sem ser bem pensado. Pensamos num tema onde todos os elementos do grupo tivessem uma relação positiva e que gostassem desse mesmo tema. Foi fácil manter a motivação, porque, de facto, estávamos a desenvolver algo que queríamos. Por vezes, temos projetos que nos desmotivam devido ao seu tema e não devido ao trabalho e tempo que exigem.

O grupo foi sempre organizado, fluido e dinâmico, o que proporcionou pouco stress no desenvolvimento do projeto e trouxe calmaria no momento do desenvolvimento.

Por fim, é extremamente importante que, no fim do projeto, todos os membros do grupo sintam que deram o seu melhor, e que no fundo, tenham adquirido os conhecimentos relacionados com o projeto. Na atualidade, um programador estende o seu leque de competências com este tipo de projeto, motivando assim o grupo a aprender e dar o seu melhor contributo.

Enquanto grupo, o trabalho foi sempre divido e muito bem gerido entre nós, podendo assim, dizer que ambos os três elementos do grupo trabalharam de igual modo. O grupo fica contente por conseguir concluir assim um projeto com esta grandeza, pois sentimos evolução pessoal e profissional ao realizar este projeto.

A nível de trabalho futuro, o desenvolvimento deste projeto impulsionou-nos no desenvolvimento programático. Será dado, certamente, seguimento a este projeto, devido ao nosso empenho, dedicação e consistência do projeto desenvolvido.

# Bibliografia e Webgrafia

Para que o desenvolvimento do projeto fosse possível, foi necessário muito trabalho autodidata, sendo, por vezes, aplicadas tecnologias que não foram dadas nas unidades curriculares. Assim, é de salientar a importância da pesquisa autodidata, pelo que, podemos indicar abaixo a nossa bibliografia.

**AMSI**

* Documentação disponível na Plataforma da unidade curricular,
* O seguinte URL, [Stack Overflow em Português](https://pt.stackoverflow.com/),
* O seguinte URL, [Desenvolvedores Android  |  Android Developers](https://developer.android.com/),
* O seguinte URL, [Documentation  |  Desenvolvedores Android  |  Android Developers](https://developer.android.com/docs),

**PlatSI**

* Documentação disponível na Plataforma da unidade curricular,
* O seguinte URL, [Stack Overflow em Português](https://pt.stackoverflow.com/),
* O seguinte URL, [Yii PHP Framework (yiiframework.com)](https://www.yiiframework.com/),
* O seguinte URL, [Wiki | Yii PHP Framework (yiiframework.com)](https://www.yiiframework.com/wiki);

**SIS**

* Documentação disponível na Plataforma da unidade curricular,
* O seguinte URL, [Stack Overflow em Português](https://pt.stackoverflow.com/),
* O seguinte URL, [Yii PHP Framework (yiiframework.com)](https://www.yiiframework.com/),
* O seguinte URL, [Wiki | Yii PHP Framework (yiiframework.com)](https://www.yiiframework.com/wiki);

# Anexos

Para melhor contextualização do projeto como um todo e de como foram aplicadas as três vertentes do projeto, serão anexados a este relatório, os respetivos relatórios dos projetos de Plataformas de Sistemas de Informação, Serviços e Interoperabilidade de Sistemas, Acesso Móvel a Sistemas de Informação e Projeto em Sistemas de Informação.

Para mais informações relativamente ao projeto de Plataformas de Sistemas de Informação, são relatados no respetivo relatório, assuntos como RBAC, especificação de requisitos e especificação de user stories.

Uma imagem com texto

Descrição gerada automaticamente

Anexo 1 - Relatório de PlatSI

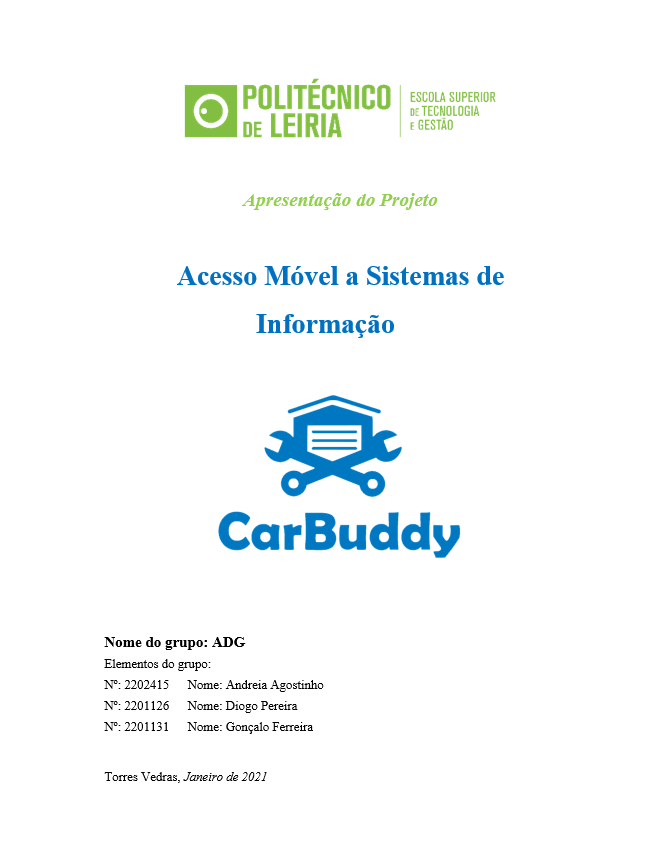
Para mais informações relativamente ao projeto de Serviços e Interoperabilidade de Sistemas, são relatados no respetivo relatório, assuntos como aplicação do serviço MQTT e todos os métodos de implementação de todos os endpoints da API desenvolvida.

Uma imagem com texto

Descrição gerada automaticamente

Anexo 2 - Relatório de SIS

Para mais informações relativamente ao projeto de Acesso Móvel a Sistemas de Informação e Projeto em Sistemas de Informação, são relatados no respetivo relatório, assuntos como especificação de requisitos, tecnologias utilizadas, interface da aplicação e dificuldades esperadas com as respetivas soluções.



Anexo 3 - Relatório de AMSI