****

<Título do projeto - Estilo Titulo>

<Nome do Aluno e Número Mecanográfico - Estilo NomeAluno>

<Nome do Aluno e Número Mecanográfico - Estilo NomeAluno>

Trabalho realizado sob a orientação de

<Professor Nome do Orientador - Estilo NomeOrientador>

<Professor Nome do Coorientador - Estilo NomeOrientador >

Licenciatura em Engenharia Informática

2023-2024

<Título do projeto>

Relatório da UC de Projeto  
Licenciatura em Engenharia InformáticaEscola Superior de Tecnologia e Gestão

<Nome1 Sobrenome1 - N. mecanográfico>

<Nome2 Sobrenome2 - N. mecanográfico >

2023-2024

A Escola Superior de Tecnologia e Gestão não se responsabiliza pelas opiniões expressas neste relatório.

Declaro que o trabalho descrito neste relatório é da minha autoria e é da minha vontade que o mesmo seja submetido a avaliação.

Nome do Aluno - Número Mecanográfico

Nome do Aluno - Número Mecanográfico

Dedicatória

(Facultativo) Dedico este trabalho a ...

Agradecimentos

(Facultativo) Agradeço a ...

Resumo

O resumo (no máximo com 250 palavras), permite a avaliação do interesse de um documento e facilita a sua identificação na pesquisa bibliográfica em bases de dados onde o documento se encontre referenciado.

É recomendável que o resumo aborde, de forma sumária:

* Objetivos principais e tema ou motivações para o trabalho;
* Metodologia usada (quando necessário para a compreensão do relatório);
* Resultados, analisados de um ponto de vista global;
* Conclusões e consequências dos resultados, e ligação aos objetivos do trabalho.

Como este modelo de relatório se dirige a trabalhos cujo foco incide, maioritariamente, no desenvolvimento de software, algumas destas componentes podem ser menos enfatizadas, e acrescentada informação sobre análise, projeto e implementação do trabalho.

O resumo não deve conter referências bibliográficas.

**Palavras-chave:** termos (no máximo 4), que descrevem o trabalho.

Abstract

Direct translation (maximum of 250 words) to English of the section “Resumo”.

**Keywords:** direct translation of “Palavras-chave”

Conteúdo

Lista de Tabelas

Tabela 1 - Estrutura Recomendada do Relatório do Projeto 3

Lista de Figuras

Figura 1: Exemplo de imagem PNG. 4

Figura 2 – Exemplo de desenho vetorial feito com o próprio MS Word. 4

Figura 3 – Exemplo de gráfico. 5

Siglas

**ESTiG** Escola Superior de Tecnologia e Gestão.

# 

Introdução

## Enquadramento

## Objetivos

## Estrutura do documento

Este modelo de relatório assume que a maioria dos projetos de fim de curso são centrados no desenvolvimento de uma solução informática para um problema. Sendo esse contexto, é apropriada uma estrutura que descreva a análise, conceção e desenvolvimento da solução implementada. Em particular, em projetos de desenvolvimento de software, espera-se que o relatório documente as principais fases do ciclo de desenvolvimento.

Nos casos em que o trabalho corresponde sobretudo à integração e/ou avaliação de componentes pré-existentes, a estrutura do relatório deverá ser adaptada em conformidade, com ênfase na descrição das tecnologias subjacentes, sua articulação e avaliação.

A estrutura efetivamente adotada para o resto do relatório é, normalmente, clarificada nesta secção, usando texto semelhante a: “O resto do relatório está organizado da seguinte forma: no capítulo 2 descreve-se ...; no capítulo 3 ...; ...; finalmente, o último capítulo apresenta as conclusões e direções de trabalho futuro.”

## Normas de Composição

Para além de uma organização que reflita o percurso seguido, o relatório deve estar bem formatado e ter aspeto sóbrio, convidando à leitura e fazendo jus ao mérito do trabalho descrito. Neste sentido, apresentam-se de seguida algumas das normas a levar em conta.[[1]](#footnote-2)

### Capítulos, seções e subseções

Cada capítulo deve começar com uma linha formata com o estilo “Título 1” ou “Heading 1” – sem qualquer texto adicional. Na linha seguinte colocam o nome do capítulo e aplicam o estilo “Cabeçalho 1 – Texto do capítulo”.

Dentro de cada capítulo podem ter seções e mesmo subseções. As seções devem ser configuradas com o estilo “Heading 2” ou “Título 2”, garantindo assim a correta numeração, a uniformidade da apresentação e que o índice pode ser corretamente construído.

As subseções devem ser formatadas com o estilo “Heading 3” ou “Título 3”, conforme se ilustra a seguir.

Certifiquem-se que os capítulos começam sempre em páginas ímpares. Para tal podem introduzir uma Quebra de Página (ver pasta Inserir), no capítulo anterior – como é feito em todos os capítulos desta template – mas isto só deve ser feito já na fase final da revisão da composição gráfica do documento.

É também de todo recomendável terminar cada capítulo com uma quebra de secção (ver pasta Esquema de página 🡪 Quebras).

### Convenções Tipográficas

Todo o corpo de texto do relatório é baseado no tipo de letra Times New Roman 12, justificado, com espaçamento entre linhas de 1,5. As notas de rodapé devem usar o tipo de letra Times New Roman 9 espaçamento 1,0[[2]](#footnote-3).

Por vezes, opta-se por apresentar as convenções tipográficas seguidas no documento, ou seja, em que circunstâncias se usam texto em *itálico*, **negrito**, ou de espaçamento uniforme (esta última formatação é normalmente usada para apresentar código fonte), bem como quais as fontes tipográficas usadas, respetivas dimensões, etc.

|  |  |
| --- | --- |
| **Nome do capítulo** | **Conteúdo** |
| *Introdução* | Objetivos. O que foi feito e porquê? |
| *Contexto, tecnologias e ferramentas* | Qual o problema tratado?  Qual é a solução proposta para ser implementada?  Quais as ferramentas/técnicas usadas? |
| *Abordagem/Modelação* | Qual é a solução proposta para ser implementada? |
| *Desenvolvimento e Implementação* | Como foi feito o trabalho? Identificar a sequência de operações realizadas |
| *Testes, avaliação e discussão* | O sistema implementado corresponde aos objetivos?  O que é que os resultados significam e porquê? |
| *Conclusões* | E depois? Quais são outras possíveis aplicações e recomendações? E a seguir? |

Tabela 1 - Estrutura recomendada do relatório do Projeto[[3]](#footnote-4)

### Tabelas e Figuras

As tabelas devem ter numeração automática (ver opção Inserir Legenda da pasta Referências do MS Word), ser referenciadas através de Referências Cruzadas (ver pasta Referências do MS Word) e ter um aspeto sóbrio, como ilustrado pela Tabela 1.



Figura 1 – Exemplo de imagem PNG.

Também as figuras devem ter numeração automática, ser referenciadas via Referências Cruzadas e ter um tamanho equilibrado (nem muito grande, nem muito pequeno), como na Figura 1.

Ator 1

Ator 2

Figura 2 – Exemplo de desenho vetorial feito com o próprio MS Word.[[4]](#footnote-5)

Sempre que possível, devem-se usar figuras em formato escalável/vetorial, evitando o uso de figuras com compressão (formatos JPG, PNG, GIF, etc.), como na Figura 1. Podem fazer uso das facilidades disponibilizadas pelo MS Word para fazer desenhos vetoriais simples, conforme ilustrado na Figura 2. Não se esqueçam de agrupar os elementos ou de utilizar uma tela dentro da qual fazem o desenho.

No caso de ser necessário apresentar gráficos, nestes devem ser indicadas, de forma clara, as grandezas associadas a cada eixo, bem como a respetiva legenda – ver exemplo na Figura 3. Adicionalmente, o esquema de cores ou de traços para as linhas, deve ser sóbrio e prevenir ambiguidades na leitura do gráfico. Importante ter em conta que os gráficos devem ser explicados para explicar os pontos principais do mesmo.

Figura 3 – Exemplo de gráfico.

As equações e expressões deverão ser centradas no texto e identificadas por números consecutivos.

Para a introdução de código fonte é usado o tipo de letra Courier New, alinhado à esquerda, com espaçamento entre linhas de 1,0 (espaçamento simples). Ou simplesmente formatar com o estilo Código.

public class Ola {

public static void main(String[] args) {

System.out.println("Ola "+args[0]);

}

}

### Distribuição dos Elementos

A forma como o texto e outros elementos (tabelas, figuras, etc.) se distribui por uma página, deve ser tal que se evitem grandes blocos vazios no final da página. Embora algumas sistemas de composição tendam a garantir isso automaticamente (como o LaTeX), é costume serem necessárias afinações para resolver, manualmente, essas (e outras) desformatações. No entanto, essas afinações devem ser deixadas para a fase pré-impressão, já com o conteúdo do documento estabilizado, de forma a evitar trabalho inconsequente.

### Correcção Ortográfica

Para além da qualidade tipográfica do relatório, é imprescindível minimizar (se possível até, erradicar) erros ortográficos. Qualquer sistema de composição de documentos suporta correção ortográfica (e, muitas vezes, sintática), pelo que não é aceitável a submissão para avaliação de relatórios sem revisão ortográfica prévia.

### Referências Bibliográficas

Ao longo do documento, deve ficar sempre perfeitamente claro o que é escrita original e o que foi baseado (ou até reproduzido de) noutras fontes.

Todas as fontes devem ser descritas na formatação usada na Bibliografia e referenciadas, no texto, pelos seus identificadores únicos. Por exemplo: “Uma solução para o problema em causa deve respeitar as propriedades *x*, *y* e *z*.” [1], ou “Neste trabalho explorou-se a API PThreads [2] com o objetivo de *. . .* ”.

Para efeito de referências bibliográficas, o MS Word possui na pasta Referências meios para adicionar e referenciar a bibliografia. Devem utilizar o estilo IEEE [1].

A reprodução fiel de texto de fontes externas deve ser limitada, surgir entre aspas, ligeiramente destacada, e ter apenas a respetiva referência bibliográfica. Para este efeito podem utilizar o estilo Referência Texto. Por exemplo:

“Recently, the employment of GPU devices is a key to achieve higher performance for computer systems. On those systems, GPUs are used for general calculation but with extreme parallelism.” [1]

### Siglas

Na primeira vez as siglas devem surgir por extenso, sendo resumidas nas vezes seguintes. Por exemplo: “o curso atual de Engenharia Informática da Escola Superior de Tecnologia e Gestão (ESTiG) foi reformulado em 2015 e, a par com o curso de Informática de Gestão, representa o leque de licenciaturas da área de Informática que a ESTiG oferece”.

# 

Contexto e Tecnologias/Ferramentas

Neste capítulo espera-se uma descrição genérica do problema e da área de intervenção: âmbito, conceitos e tecnologia e/ou uma revisão da literatura (“estado da arte”). No caso de um projeto eminentemente prático, devem ser descritas também as ferramentas usadas e a justificação para a sua escolha.

Normalmente, este capítulo é dividido em múltiplas secções, de forma a compartimentar os tópicos abordados, facilitando assim a sua leitura e compreensão.

Tecnologias usadas : css moderno Tailand CSS , next js framework do react

## Contexto e Âmbito do Problema

O principal objetivo deste projeto é impulsionar o desenvolvimento econômico regional ao proporcionar maior visibilidade e acesso a produtos e serviços locais. A plataforma foi concebida para beneficiar produtores regionais, ampliando seu alcance de mercado e facilitando para os consumidores a descoberta e aquisição de produtos locais. Além disso, a plataforma inclui uma funcionalidade de venda de ingressos para eventos e museus em destinos específicos, o que não só estimula o turismo local, mas também enriquece a experiência cultural dos usuários. Com isso, o projeto visa fortalecer a economia local, conectando de forma direta produtores, consumidores e o setor de entretenimento e cultura.

A proposta deste projeto é criar uma plataforma que ofereça maior alcance e oportunidades para pequenos empreendedores locais, superando as limitações frequentemente impostas por grandes sites de comércio eletrônico. Ao dedicar um espaço exclusivo aos produtores regionais, a plataforma abre novas oportunidades de crescimento para esses negócios, permitindo que alcancem um público mais amplo e diversificado. Dessa forma, busca-se fomentar o desenvolvimento econômico local, proporcionando aos pequenos empreendedores a visibilidade e o suporte necessários para competir em um mercado amplamente dominado por grandes sites e-commerce.

Assim, o site foi desenvolvido com o objetivo de atender a essas demandas, tanto no que se refere à expansão econômica de pequenas regiões quanto ao apoio direto aos produtores locais. A plataforma foi projetada para ser uma ferramenta eficaz no crescimento dos negócios regionais, conectando-os a um mercado mais amplo e oferecendo as oportunidades que, muitas vezes, não estão disponíveis em plataformas maiores.

## Revisão da Literatura e Estado da Arte(Ainda não fiz)

2.2.1 Contextualização e Importância do Problema

2.2.2 Estudos e Pesquisas Relacionadas

2.3 Tecnologias e Ferramentas Utilizadas

**2.3.1 Tailwind CSS: Um Framework de CSS Moderno**

O Tailwind CSS é um framework utilitário-first que facilita a criação de interfaces de usuário altamente personalizáveis e responsivas de maneira ágil e eficiente. Diferente de frameworks tradicionais, como Bootstrap ou Foundation, que fornecem componentes pré-construídos, o Tailwind CSS oferece uma extensa coleção de classes utilitárias que podem ser combinadas para construir qualquer design diretamente no HTML. Isso proporciona aos desenvolvedores um controle mais granular sobre o estilo e o layout das aplicações, permitindo uma personalização detalhada sem a necessidade de sobrescrever estilos padrão.

Optou-se por utilizar o Tailwind CSS neste projeto devido à sua flexibilidade, eficiência e capacidade de criar uma interface moderna, responsiva e escalável. Em resumo, a escolha do Tailwind CSS reflete o compromisso em construir uma plataforma que atenda às demandas atuais de design e experiência do usuário, sem comprometer a velocidade e a precisão no desenvolvimento.

**2.3.2 TypeScript: Uma Extensão Poderosa para o Desenvolvimento Web Dinâmico**

TypeScript é uma extensão do JavaScript que adiciona tipagem estática ao desenvolvimento web, proporcionando uma base mais robusta e segura para a criação de aplicações complexas. Embora JavaScript seja fundamental para o desenvolvimento web moderno, TypeScript eleva esse potencial ao permitir q escrevam código eu durante o desenvolvimento da aplicação o código seja mais previsível e de fácil manutenção . Com TypeScript, os elementos da página podem ser atualizados dinamicamente sem a necessidade de recarregar toda a página, oferecendo uma experiência de usuário mais fluida e responsiva, essencial para aplicações web que demandam alta interatividade.

O ecossistema do TypeScript é vasto e inclui diversas bibliotecas e frameworks que aceleram o desenvolvimento de aplicações. Um exemplo proeminente é o React, que, combinado com TypeScript, facilita a construção de interfaces de usuário ao introduzir uma abordagem baseada em componentes reutilizáveis com tipagem forte. Essa modularidade e a segurança adicional proporcionada pela tipagem estática não só melhoram a agilidade no desenvolvimento, como também aumentam a reutilização e a manutenção do código.

Além disso, TypeScript oferece suporte completo para frameworks especializados como o Next.js, que é construído sobre o React e oferece funcionalidades avançadas, como a renderização no lado do servidor e do cliente. A combinação de TypeScript com Next.js permite que a aplicação aproveite essas funcionalidades de forma mais eficiente, garantindo, ao mesmo tempo, a integridade e a robustez do código por meio da tipagem estática.

Portanto, TypeScript não é apenas uma base para criar experiências de usuário interativas, mas também o alicerce que suporta um ecossistema de ferramentas robustas, permitindo o desenvolvimento eficiente, escalável e seguro de aplicações web**.**

**2.3.3 React: Um Framework para Construção de Interfaces de Usuário Modernas**

O React é uma biblioteca JavaScript amplamente utilizada para a construção de interfaces de usuário (UI) modernas e reativas. O React mudou o desenvolvimento frontend ao introduzir uma abordagem baseada em componentes, onde a interface é dividida em partes menores reutilizáveis. Cada componente gerencia seu próprio estado podendo ser integrado a outros componentes para formar interfaces mais complexas de forma modular e eficiente.

Principais Funcionalidades e Benefícios:

- Componentização: A principal característica do React está na sua forma de implementação baseada em componentes. Cada parte da UI é representada como um componente individual promovendo a consistência no design e a manutenção e escalabilidade do código.

- React Hooks: os Hooks permitem no desenvolvimento da aplicação usem estado e outras funcionalidades do React em componentes funcionais, sem necessariamente precisar escrever classes. Simplifica o código e torna mais fácil compartilhar lógica entre componentes.

A escolha do React como o framework principal para o desenvolvimento da interface de usuário deste projeto é justificada por diversas razões. Primeiro, a capacidade do React em criar interfaces interativas e dinâmicas.Assim como , a componentização que permite que o design e a funcionalidade sejam feitos de forma mais modular possível . Em suma, o React foi escolhido por sua robustez, flexibilidade e eficiência na construção de interfaces de usuário modernas e responsivas, alinhada com os objetivos de criar uma solução eficiente, escalável e centrada no usuário.

**2.3.4 Next.js: Um Framework para Aplicações React com Renderização Híbrida**

Next.js é um framework de desenvolvimento baseado na biblioteca React que oferece funcionalidades avançadas para a construção de aplicações web atuais. O Next.js simplifica a criação de aplicações React ao integrar um suporte para renderização no lado do servidor, e renderização do lado do cliente .

Principais Funcionalidades e Benefícios:

- Renderização Híbrida : O Next.js permite que diferentes páginas ou componentes sejam renderizados de maneira mais otimizada, tanto do lado servidor como do lado do no cliente. Permitindo a criação de aplicações mais performáticas, melhorando tanto o tempo de carregamento quanto a otimização para motores de busca.

- Roteamento Automático. O Next.js oferece um sistema de roteamento simplificado, baseado na estrutura de pastas do projeto. Cada arquivo dentro da pasta**pages** é transfrormada em uma rota, eliminando a necessidade de configuração manual e facilitando a navegação entre as páginas da aplicação.

- API Routes: Com o Next.js, torna-se possível criar rotas de API diretamente no diretório da aplicação, facilitando a implementação de backend para funcionalidades como autenticação, processamento de dados, e integração com serviços externos. Permitindo que a aplicação seja desenvolvida em uma única estrutura de código, integrando o frontend e backend de forma mais eficiente.

- Autenticação com NextAuth.js: Next.js se integra com a biblioteca NextAuth.js fornecendo uma solução mais moderna e robusta de autenticação. Na sua versão mais recente do NextAuth.js facilita a implementação de autenticação segura e escalável, suportando múltiplos provedores como Google, Facebook, GitHub, e outros. Além disso, oferece suporte a métodos de autenticação via email e senha, o que proporciona flexibilidade na escolha de estratégias de autenticação. Essa integração é importante para garantir que os usuários possam acessar a plataforma de maneira segura e ao mesmo tempo em que facilita a gestão de autenticação no backend.

- Arquitetura Escalável e Modulada: O Next.js foi desenvolvido para ser escalável. A modularidade do framework integrada a capacidade de renderização híbrida, permitindo que a aplicação seja construída de forma mais otimizada .

A escolha do Next.js como framework para este projeto é justificada pelas suas capacidades de renderização híbrida e flexibilidade. A capacidade de renderização tanto das páginas tanto do lado do servidor quanto no cliente permitiu uma otimização ainda mais precisa durante o desenvolvimento da aplicação, garantindo aos usuários uma experiência mais rápida e responsiva

Em resumo, o Next.js foi escolhido por sua combinação de flexibilidade, performance e suporte robusto a funcionalidades críticas como a autenticação moderna. Esses atributos tornam o Next.js a escolha ideal para o desenvolvimento de uma plataforma que precisa ser tanto poderosa quanto adaptável, cumprindo os objetivos do projeto de criar uma solução moderna, segura e eficiente.

**2.3.5 Firebase: Uma Plataforma Completa para Desenvolvimento de Aplicações**

Firebase é uma plataforma de desenvolvimento de aplicativos móveis e web criada pelo Google. Oferendo uma ampla gama de serviços e ferramentas que auxilia a construir, melhorar, e escalar suas aplicações sem a necessidade de gerenciar a infraestrutura subjacente. Firebase é particularmente valiosa para projetos que precisam de uma solução completa para backend, autenticação, banco de dados e armazenamento.

Principais Funcionalidades e Benefícios:

- Autenticação Simplificada: Firebase Authentication fornece uma solução segura e simplificada de implementação de autenticação. Ele suporta uma variedade de métodos de login, incluindo email/senha, provedores de identidade como Google, Facebook, e GitHub .

- Banco de Dados em Tempo Real e Firestore: Firebase oferece dois tipos principais de bancos de dados: Realtime Database e Firestore. Ambos permitem o armazenamento e a sincronização de dados em tempo real entre clientes e servidores, o que é ideal para aplicações que necessitam de atualizações instantâneas de dados, como chats, notificações, e dashboards interativos.

- Armazenamento Seguro de Arquivos: Firebase Storage permite que os dados armazenem e sirvam arquivos de mídia, como imagens e vídeos, diretamente da nuvem. Ele é altamente integrado com Firebase Authentication, garantindo que o acesso aos arquivos seja seguro de acordo com as permissões definidas .

- Funções na Nuvem (Cloud Functions): Cloud Functions for Firebase permite a execução de códigos backend em resposta a eventos disparados pelo Firebase ou por solicitações HTTP. Adicionando uma camada de lógica de negócio ao backend sem a necessidade de gerenciar servidores .

A escolha do Firebase como a plataforma de backend para este projeto é justificada por várias razões, principalmente pela sua capacidade de fornecer uma solução completa e integrada que cobre todas as necessidades de backend sem a complexidade de gerenciar infraestrutura. Firebase oferece uma implementação simples e escalável de autenticação, o que é importante para garantir a segurança e a privacidade dos usuários da plataforma.

# 

Abordagem/Análise/Modelação

Neste capítulo espera-se uma descrição detalhada do problema e da proposta de solução.

No caso de projetos de desenvolvimento de software, deverá deitar-se mão dos conceitos e ferramentas de Análise/Modelação estudadas durante o curso (por exemplo, diagramas UML ou outra linguagem gráfica). Deve-se indicar explicitamente as tarefas a desempenhar pelo sistema, e os atores que interagem com o mesmo. A descrição deve ter suficientes detalhes para perceber as dificuldades associadas à resolução do problema.

# 

Desenvolvimento/Implementação

Neste capítulo é descrito o trabalho de implementação, salientando os pontos mais relevantes da mesma, dificuldades encontradas ou soluções técnicas inovadoras desenvolvidas ou aplicadas. Em particular, se foi usado código desenvolvido por terceiros (por exemplo, código *open-source*), deve ser facilmente distinguível quais as funcionalidades originais do mesmo e o que foi necessário implementar para obter as funcionalidades desejadas.

# 

Testes/Avaliação/Discussão

Este capítulo apresenta os testes realizados para verificar que o projeto desenvolvido cumpre os objetivos assumidos e resolve, de facto, o problema descrito na Análise/Modelação.

Para uma melhor compreensão, os resultados de cada teste devem ser precedidos de uma descrição, mesmo que resumida, do teste realizado e dos resultados esperados.

Os resultados do trabalho são comentados, acrescentando-lhe valor:

* O que é que se pode inferir ou conjeturar dos resultados obtidos?
* O que poderia/deveria ter sido feito de forma diferente?
* Onde se foi além dos objetivos iniciais?
* Quais os objetivos que ficaram por cumprir e porquê?

# 

Conclusões

As conclusões devem sintetizar e proporcionar uma perspetiva unificadora ao trabalho efetuado. Poderá ser feita uma breve referência a trabalhos de outros com semelhanças ao efetuado e ao conhecimento que resultou do trabalho efetuado, bem como sugestões de trabalho futuro. A coerência do documento implica que as conclusões devem ser coerentes com as ideias expostas na introdução.

Bibliografia

|  |  |
| --- | --- |
| [1] | S. O. T. N. e. S. M. R. Aoki, “Hybrid opencl: Enhancing opencl,” em *IEEE Ninth International Symposium on Parallel and Distributed Processing with Applications*, Washington, 2011. |
| [2] | E. e. B. F. Monteiro, Engenharia de Redes Informáticas., FCA – Editora de Informática, LDA, 2000. |



Proposta Original do Projeto

NOTA: Apêndice X deve ser formatado com o estilo “Apêndice 1” e o título do apêndice com o estilo “Cabeçalho 1 – Texto do capítulo”.

1. <Proposta de Projeto>

Reprodução integral da proposta original do projeto.

<Cada Nome de Anexo tem o estilo Anexo 1>

Todo o corpo de texto do relatório é baseado no tipo de letra Times New Roman 12, justificado, com espaçamento entre linhas de 1,5.

* 1. <Sub-Anexo>

* + 1. <Sub-Sub-Anexo>



Outro(s) Apêndice(s)

1. <Nome do Anexo>

<Cada Nome de Anexo tem o estilo Anexo 1>

Todo o corpo de texto do relatório é baseado no tipo de letra Times New Roman 12, justificado, com espaçamento entre linhas de 1,5.

* 1. <Sub-Anexo>
     1. <Sub-Sub-Anexo>

1. Estas normas, assim como este modelo de documento, são compatíveis com o preconizado no "Regulamento da Unidade Curricular de Projecto das Licenciaturas", designadamente no que diz respeito ao relatório do projeto. [↑](#footnote-ref-2)
2. Nota de rodapé. [↑](#footnote-ref-3)
3. Aplicar estilo Leg1 nas legendas das tabelas. [↑](#footnote-ref-4)
4. Aplicar estilo Leg1 nas legendas das figuras. [↑](#footnote-ref-5)