

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS (ISPTEC)

DEPARTAMENTO DE ENGENHARIA E TECNOLOGIA

LICENCIATURA EM ENGENHARIA INFORMÁTICA

**RELATÓRIO PROJECTO FINAL**

Docente:

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

Judson Paiva

Luanda, 28 de Janeiro de 2024



INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS (ISPTEC)

DEPARTAMENTO DE ENGENHARIA E TECNOLOGIA

LICENCIATURA EM ENGENHARIA INFORMÁTICA

**RELATÓRIO PROJECTO FINAL**

Darilton Mário – 20211852.

Eugelice Yuye – 20210765.

Josué Dosidiana – 20210709.

Otília Marques – 20211958.

Grupo Nº: 6.

Turma: EINF5 M1. Docente:

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

Judson Paiva

Luanda, 28 de Janeiro de 2024

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS (ISPTEC)

DEPARTAMENTO DE ENGENHARIA E TECNOLOGIA

LICENCIATURA EM ENGENHARIA INFORMÁTICA

**RELATÓRIO PROJECTO FINAL**

Trabalho de Conclusão de Semestre, apresentado ao Curso de Engenharia Informática do Departamento de Engenharias e Tecnologias (DET), do Instituto Superior Politécnico de Tecnologia e Ciências (ISPTEC), como Requisito para Obtenção da nota do Exame Final em Engenharia de Software.

**Docente:** Judson Paiva.

Luanda, 28 de Janeiro de 2024

**ÍNDICE**

[DEDICATÓRIA 6](#_Toc158126913)

[AGRADECIMENTO 7](#_Toc158126916)

[RESUMO 8](#_Toc158126923)

[ABSTRACT 9](#_Toc158126924)

[LISTA DE FIGURAS 10](#_Toc158126925)

[LISTA DE ABREVIATURAS E SIGLAS 11](#_Toc158126926)

[1. INTRODUÇÃO 12](#_Toc158126927)

[2. DESENVOLVIMENTO 13](#_Toc158126928)

[**a.** METODOLOGIA ADOTADA 13](#_Toc158126929)

[Requisitos Funcionais 15](#_Toc158126930)

[Requisitos Não Funcionais 16](#_Toc158126931)

[**b.** **ANÁLISE DE CASOS DE USO** 18](#_Toc158126932)

[**c.** **DIAGRAMA DE CLASSES** 19](#_Toc158126933)

[**d.** **DIAGRAMAS DE PACOTES** 20](#_Toc158126934)

[**e.** **DIAGRAMAS DE SEQUENCIA** 21](#_Toc158126935)

[3. CONCLUSÃO 22](#_Toc158126936)

[4. REFERÊNCIAS 23](#_Toc158126937)

# DEDICATÓRIA

# *Dedicamos este trabalho a nós porque nós Merecemos,*

# *lutamos muito para chegar até aqui.*

# AGRADECIMENTO

# Gostariamos de expressar os nossos sinceros agradecimentos a todas as pessoas que estiveram ao meu lado durante esta caminhada académica (3ºAno). Sem o apoio e suporte de cada um deles, esta conquista não teria sido possível.

# À nossa família, pais, irmãos e demais familiares, nosso profundo agradecimento.

# Ao nosso professor Judson Paiva, um verdadeiro mestre, que dedicou o seu tempo e conhecimento para nos ensinar e orientar, expressamos a nossa gratidão.

# Aos nossos colegas de curso, verdadeiros companheiros de jornada, agradecemos por compartilharmos momentos de estudo, discussões e superação.

# Por fim, queremos expressar a nossa gratidão a todos aqueles que acreditaram em nós, que nos encorajaram nos momentos de dúvida e que nos deram suporte emocional e motivacional. Seja através de palavras de incentivo, gestos de apoio ou simplesmente por estarem presentes, vocês fizeram a diferença nas nossas vidas.

# Agradecemos também o ISPTEC por ser uma universidade de excelência no ensino.

# RESUMO

No presente relatório, a nossa tarefa é explorar e aplicar conceitos fundamentais de engenharia de software, centrando-se em requisitos funcionais e não funcionais, diagramas de casos de uso, especificação de casos de uso e prototipagem para o desenvolvimento do Sistema de Gestão de Imobiliária (SGI).

A gestão de uma imobiliária envolve uma série de atividades cruciais, desde a aquisição e avaliação de propriedades até a conclusão de transações. Para otimizar esse processo, estratégias eficazes de captação de imóveis, marketing imobiliário e negociação são essenciais.

Além disso, a implementação de tecnologias inovadoras, como plataformas online e sistemas de gestão imobiliária, torna-se cada vez mais importante para melhorar a eficiência operacional.

Estamos usando práticas do processo de desenvolvimento ágil Scrum. Foi feita uma breve análise do mercado de locação imobiliária em Luanda. O desenvolvimento do sistema envolveu desde o levantamento de requisitos e modelagem de casos de uso e de classes.

No entanto, desafios como a flutuação do mercado e questões regulatórias exigem uma abordagem adaptativa por parte dos gestores. Este relatório visa fornecer uma visão abrangente dessas considerações, visando contribuir para a excelência na gestão de imobiliárias.

Palavras-chave: Sistema de Gestão de Imobiliária, Desenvolvimento Ágil.

# ABSTRACT

In this report, our task is to explore and apply fundamental concepts of software engineering, focusing on functional and non-functional requirements, use case diagrams, use case specification, and prototyping for the development of a Real Estate Management System (REMS).

Real estate management involves a series of crucial activities, from property acquisition and evaluation to transaction completion. To streamline this process, effective strategies for property acquisition, real estate marketing, and negotiation are essential.

Additionally, the implementation of innovative technologies such as online platforms and real estate management systems is increasingly important for improving operational efficiency.

We are using Agile development process practices, specifically Scrum. A brief analysis of the real estate rental market in Luanda was conducted. The system development included requirements gathering and modeling of use cases and classes.

However, challenges such as market fluctuations and regulatory issues require an adaptive approach from managers. This report aims to provide a comprehensive overview of these considerations, aiming to contribute to excellence in real estate management.

Keywords: Real Estate Management System, Agile development.

# LISTA DE FIGURAS

# LISTA DE ABREVIATURAS E SIGLAS

# INTRODUÇÃO

A gestão eficiente de uma imobiliária desempenha um papel crucial no setor imobiliário, onde a complexidade e a dinâmica do mercado exigem abordagens estratégicas e operacionais bem definidas. Neste relatório, exploraremos os principais aspectos relacionados à gestão de uma imobiliária, abrangendo desde a captação de imóveis até a conclusão de transações bem-sucedidas. Analisaremos as práticas recomendadas, desafios enfrentados pelo setor e as oportunidades emergentes que moldam o cenário atual da gestão imobiliária.

Este relatório apresenta a concepção e os principais aspetos do projeto desenvolvido. O desafio que nos foi proposto consiste na criação de um sistema abrangente para a gestão eficiente de uma imobiliária diversificada. O objetivo central é aprimorar ou enfrentar desafios significativos na administração e controle do portfólio de imóveis, notadamente nas áreas de gestão de locações, vendas e manutenções.

Como resposta a esse desafio da "Casa Eficiente" consideramos a implementação de um Sistema de Gestão de Imóveis (SGI). E para isso usamos os fundamentos sólidos de programação orientada a objetos, o mesmo teve integração com base de dados e interface gráfica.

O projeto está estruturado em pacotes e classes, refletindo a organização e encapsulamento de responsabilidades. Os demais pacotes “Empreiteiro”, "Cliente" e "Corrector” apresentam subclasses específicas que compõem o projecto.

Este relatório serve como um guia abrangente para a compreensão do projeto, destacando seus principais componentes e funcionalidades. A seguir, detalharemos cada aspeto do sistema, desde os casos de uso até as classes fundamentais, visando uma compreensão completa do trabalho.

# DESENVOLVIMENTO

## **METODOLOGIA ADOTADA**

Metodologia SCRUM

O uso da metodologia ágil Scrum para o desenvolvimento do Sistema de Gestão Imobiliária (SGI) foi mais vantajoso por várias razões, considerando os requisitos e características do projeto. Aqui estão alguns motivos pelos quais o Scrum pode ser adequado para o SGI:

Adaptabilidade a Mudanças: O Scrum é conhecido por sua capacidade de se adaptar a mudanças nos requisitos de forma flexível.

Entrega Incremental: O Scrum enfatiza a entrega incremental de funcionalidades a cada iteração (sprint). Isso significa que partes do sistema podem ser entregues e testadas regularmente, permitindo que os usuários obtenham valor mais rapidamente.

Colaboração Intensa: O Scrum promove uma colaboração intensa entre os membros da equipe, incluindo desenvolvedores. Isso é crucial para garantir que o sistema atenda às expectativas e necessidades de todos os stakeholders.

Feedback Contínuo dos Usuários: A abordagem iterativa do Scrum permite a obtenção de feedback contínuo dos usuários finais.

Priorização de Funcionalidades: O Scrum utiliza uma lista de prioridades (Product Backlog) que permite aos stakeholders e à equipe priorizar as funcionalidades mais importantes. Isso é útil quando há uma variedade de requisitos, como definir trajetos, solicitação de manutenção e adicionar veículos.

Transparência e Visibilidade: O Scrum promove a transparência e a visibilidade do progresso do projeto por meio de reuniões regulares (reuniões diárias, revisões de sprint). Isso ajuda na identificação rápida de problemas e no alinhamento contínuo com as metas do projeto.

Aprimoramento Contínuo: O Scrum incentiva a retrospetiva ao final de cada sprint, promovendo a melhoria contínua. Isso é valioso para ajustar processos e práticas com base no feedback e nas lições aprendidas.

No entanto, é importante notar que a escolha do Scrum ou de qualquer outra metodologia deve ser baseada em uma avaliação cuidadosa do contexto do projeto, das necessidades da equipe e dos stakeholders.

**REQUISITOS FUNCIONAIS E NÃO FUNCIONAIS**

### Requisitos Funcionais

Requisitos funcionais são funcionalidades que descrevem o que o sistema deve fazer, ou seja, suas funcionalidades e comportamentos principais.

* **RF01** – **Pesquisar, Alocar, Avaliar e Comprar Imóveis:**
  + - **Ator:** Cliente
    - **Descrição:** permite ao cliente fazer todas essas operações.
* **RF02 –** **Gerir pedidos e gerar histórico de manutencões de imóveis**
  + - **Ator:** Empreiteiro
    - **Descrição** permite ao empreiteiro fazer todas essas operações.
* **RF03 – Vender, Registar, Actualizar, Listar Imóveis e Gerar Contratos:**
  + - **Ator:** Corrector
    - **Descrição:** possibilita ao corrector realizar todas as funções básicas.
* **RF03 – Eliminar Cliente, Registar, Actualizar, Listar, e Terinar Sessão:**

### Ator: Administrador.

* **Descrição**: possiblita ao admin fazer algumas funções do programa

### Requisitos Não Funcionais

Requisitos não funcionais são funcionalidades que estabelecem as características de qualidade e restrições do sistema.

* **RNF01 - Desempenho**

O sistema deverá ser capaz de gerir um número elevado de imoveis simultaneamente, sem atrasos significativos e responder de maneira eficiente, especialmente durante a procura dos imoveis.

* **RNF02 - Segurança**

O sistema deverá garantir que as informações pessoais dos usuários, como nome, endereço e número de telefone, sejam mantiteca das em sigilo e protegidas contra hackers e outros ataques cibernéticos.

**Acesso restrito:** cada usuário deve ter permissões específicas de acordo com seu papel (cliente, propreitário, corrector, empreiteiro etc).

* **RNF03 - Portabilidade**

O sistema deverá ser capaz de ser executado em diferentes plataformas como desktops, laptops, tablets e smartphones.

* **RNF04 - Confiabilidade**

O sistema deve ser capaz de lidar com falhas de hardware e software sem interromper o serviço para os usuários.

* **RNF05 – Escalabilidade**

O Sistema deverá ser capaz de lidar com um aumento no número de usuários e imoveis sem diminuir o desempenho.

* **RNF06 - Manutenção**

O código fonte deve ser organizado e documentado para facilitar manutenções futuras,

atualizações do sistema devem ser implementadas de forma a minimizar impactos nas operações.

* **RNF07 - Compatibilidade**

O sistema deve ser compatível com diferentes sistemas operacionais e navegadores para garantir a acessibilidade.

* **RNF08 - Análise de Dados**

O sistema deve suportar ferramentas de análise de dados para extrair insights significativos das avaliações e feedback dos clientes.

* **RNF10 – Histórico detalhado**

A funcionalidade de registo histórico deve ser capaz de armazenar detalhes completos das manutenções realizadas, proporcionando um histórico detalhado e acessível.

## **ANÁLISE DE CASOS DE USO**

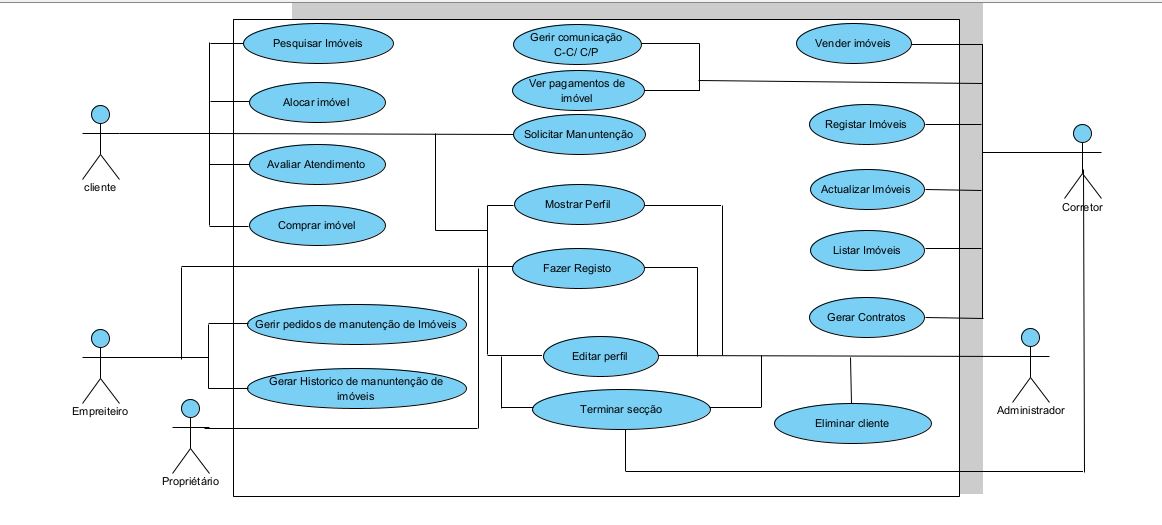
O Diagrama de casos de uso é uma ferramenta fundamental na modelagem de sistemas, destacando as interações entre actores externos e o sistema. Os diagramas de casos de uso desempenham um papel fundamental na engenharia de software, proporcionando uma visão visual e clara da interação entre um sistema e seus usuários.

Esses diagramas fazem parte da linguagem de modelagem unificada (UML) e são utilizados para capturar e descrever os requisitos funcionais de um sistema, destacando as diferentes maneiras pelas quais os usuários interagem com ele.

Ao representar as funcionalidades principais e cenários de uso, os diagramas de casos de uso auxiliam no entendimento e na comunicação entre desenvolvedores, analistas e clientes, facilitando a identificação e análise de requisitos essenciais.

Ao longo do ciclo de vida do projeto, o Diagrama de casos de uso é flexível e pode ser atualizado para refletir mudanças nos requisitos, tornando-se uma ferramenta vital para a modelagem e comunicação eficazes no desenvolvimento de software.

Então baseado nisso foi construído o Diagrama de Casos de Uso para o Sistema de Gestão Imobiliária.

****

## **DIAGRAMA DE CLASSES**

Um diagrama de classes é uma representação visual das classes de um sistema de software, seus atributos, métodos e os relacionamentos entre elas. Ele descreve a estrutura estática do sistema, mostrando como as classes estão relacionadas umas com as outras e quais são seus papéis no contexto do sistema. Esse tipo de diagrama é amplamente utilizado na modelagem orientada a objetos para visualizar a estrutura do sistema e facilitar a comunicação entre os membros da equipe de desenvolvimento.

Abaixo temos as nossas principais classes:

* **CasaEficiente:**
* **Pacote:** Encapsula as classes essenciais para a gestão da Casa Eficiente.
* **Algumas Classes que temos:**
  + - Imovel.
    - Cliente.
    - Usuário.
    - Conta.
    - Administrador.
    - Empreiteiro.
    - Corrector.
    - Manutencao.
    - Etc.

## **DIAGRAMAS DE PACOTES**

Diagramas de pacotes são diagrama estruturais comumente usados para simplificar os diagramas de classe complexos e organizar as classes em pacotes. Um pacote é uma coleção de elementos relacionados, incluindo diagramas, documentos, classes e pacotes de eventos. Além disso, o diagrama de pacotes oferece uma valiosa visibilidade de alto nível para grandes projetos e sistemas.

## **DIAGRAMAS DE SEQUENCIA**

Um diagrama de sequência é um tipo de diagrama de interação em UML (Unified Modeling Language) que descreve como os objetos em um sistema interagem em uma determinada sequência de eventos ao longo do tempo. Ele mostra a troca de mensagens entre os objetos, destacando a ordem em que essas mensagens são enviadas e recebidas durante a execução de um determinado cenário ou funcionalidade do sistema.

Os diagramas de sequência são úteis para visualizar o fluxo de controle e a colaboração entre os objetos em diferentes etapas de um processo ou operação dentro do sistema. Eles ajudam a entender melhor o comportamento dinâmico do sistema e são frequentemente usados durante o processo de análise e design de sistemas de software.

# CONCLUSÃO

A presente documentação descreve o Sistema de Gestão de uma Imobiliária e foi desenvolvido pelo grupo acima apresentado, cujo principal objetivo foi aprimorar a eficiência operacional e enfrentar obstáculos para manter um registro preciso e atualizado das propriedades disponíveis, enfrentando assim os obstáculos na gestão de locações, vendas e manutenções, incorporando conceitos fundamentais de programação orientada a objetos.

Em resumo, neste projeto desenvolvi um software para a gestão de uma imobiliária, que permite a locação de imóveis, comparativo de propriedades e gerenciamento de manutenção, entre outras funcionalidades.

Este software foi projetado para otimizar os processos da imobiliária, proporcionando uma gestão mais eficiente e facilitando as operações diárias relacionadas à administração de propriedades.

# REFERÊNCIAS

Beck, Kent (2001) “Manifesto Ágil”, http://www.manifestoagil.com.br/, Janeiro 2015.

Hansson, H. David (2003),”Ruby on Rails”, http://rubyonrails.org, Fevereiro 2015.

Henry, Orion. (2007),”Heroko”, https://www.heroku.com , Junho 2015.

Neighman, Daniel (2009) “Gem Divise”, https://github.com/plataformatec/devise, Junho 2015. Schwaber, K. & Sutherland, J. (2013). “Guia do Scrum - Um guia definitivo para o Scrum: As regras do jogo”. http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf, Junho 2015.

Thoughtbot. (2009) “HighVoltage”, https://github.com/thoughtbot/high\_voltage, Janeiro 2015.

Twitter. (2010), “Bootstrap”, http://getbootstrap.com, Maio 2015. Smaldone, Javier (2008) “RailRoady”, https://github.com/preston/railroady, junho 2015

Sommerville, Ian. (2011). "Engenharia de Software." 9ª edição. Addison-Wesley. Este livro oferece uma abordagem abrangente sobre os princípios da engenharia de software, fornecendo fundamentos teóricos e práticos que foram aplicados no desenvolvimento do Sistema de Gestão.

Booch, Grady, Rumbaugh, James, & Jacobson, Ivar. (2005). "UML - Guia do Usuário." 2ª edição. Bookman. Esta referência é essencial para compreender os conceitos e a aplicação prática da Linguagem de Modelagem Unificada (UML), utilizada na elaboração dos diagramas de casos de uso e na modelagem do sistema.

Gamma, Erich, Helm, Richard, Johnson, Ralph, & Vlissides, John. (1995). "Design Patterns: Elements of Reusable Object-Oriented Software." Addison-Wesley. Este livro é uma fonte valiosa para entender e aplicar padrões de projeto, sendo uma referência útil para a criação da estrutura modular e organizada do Sistema de Gestão.