

**Universidade do Minho**  
Escola de Engenharia  
Licenciatura em Engenharia Informática

## **Unidade Curricular de Laboratórios de Informática IV**

Ano Letivo de 2024/2025

### **Gestão de uma Linha de Montagem**

**Alex Sá**  
a104257

**Alexandre Dias**  
a103994

**Jorge Pereira**  
a104173

**Paulo Ferreira**  
a96268

**Rafael Fernandes**  
a104271

25 Novembro, 2024

**LI4**

Data da Receção	
Responsável	
Avaliação	
Observações	

## Gestão de uma Linha de Montagem

<b>Alex Sá</b> a104257	<b>Alexandre Dias</b> a103994	<b>Jorge Pereira</b> a104173	<b>Paulo Ferreira</b> a96268	<b>Rafael Fernandes</b> a104271
---------------------------	----------------------------------	---------------------------------	---------------------------------	------------------------------------

25 Novembro, 2024

## Resumo

Este relatório documenta a primeira fase do projeto MineBuilds, que visa criar uma aplicação para otimizar a experiência dos jogadores no Minecraft. O objetivo da aplicação é permitir a gestão de uma linha de montagem, facilitando a criação de construções pré-definidas e promovendo a criatividade dos utilizadores.

Nesta etapa inicial, a equipa concentrou-se na análise e especificação do sistema. Foram definidos os requisitos funcionais e não funcionais, desenvolvidos modelos conceptuais e lógicos, e criados diagramas de atividades e use cases para descrever o comportamento do sistema. Além disso, foi estabelecida uma arquitetura robusta que guiará a implementação futura.

Embora a implementação de código, interface web e base de dados ainda esteja por ser realizada, a base conceptual desenvolvida nesta fase garante uma orientação clara para as próximas etapas. O trabalho futuro incluirá a implementação de backend em .NET, o desenvolvimento da interface web, a criação da base de dados no Microsoft SQL Server e a integração e validação de todos os componentes.

A MineBuilds tem potencial para oferecer uma solução eficaz, que não apenas simplifica a experiência de jogo para novos jogadores, mas também incentiva a exploração criativa, proporcionando benefícios significativos para a comunidade do Minecraft.

**Área de Aplicação:** Desenvolvimento de Software

**Palavras-Chave:** Desenvolvimento de Software, SqlServer, Requisitos

# Índice

<b>1. Introdução</b>	<b>1</b>
1.1. Contextualização	1
1.2. Motivação e Objectivos	1
1.3. Justificação e utilidade do sistema	2
1.4. Estabelecimento da identidade do projeto	2
1.5. Identificação de recursos necessários	3
1.6. Maquete do sistema	4
1.7. Definição de um conjunto de medidas de sucesso	4
1.8. Plano de desenvolvimento (diagrama de GANTT)	5
<b>2. Levantamento e Análise de Requisitos</b>	<b>7</b>
2.1. Apresentação da estratégia e método	7
2.2. Descrição geral dos requisitos (funcionais e não funcionais) levantados	7
2.2.1. Requisitos funcionais	8
2.2.1.1. Autenticação	8
2.2.1.2. Perfil	8
2.2.1.3. Linha de Montagem	9
2.2.1.4. Stock	11
2.2.2. Requisitos não funcionais	11
2.3. Validação dos requisitos estabelecidos	11
<b>3. Especificação e Modelação do <i>Software</i></b>	<b>13</b>
3.1. Apresentação geral da especificação	13
3.2. Aspetos estruturais	15
3.3. Aspetos comportamentais	18
<b>4. Conceção do Sistema de Dados</b>	<b>32</b>
4.1. Apresentação geral da estrutura do sistema de dados	32
4.2. Descrição detalhada dos vários elementos de dados e seus relacionamentos	33
4.2.1. Utilizador	33
4.2.2. Encomenda	33
4.2.3. Bloco	34
4.2.4. Construção	34
4.2.5. Blocos para produzir uma Construção	34
4.2.6. Propriedades do Bloco	34
4.2.7. Propriedades da Construção	35
4.2.8. Relacionamento Utilizador (1,1) para Encomenda (0,N)	35
4.2.9. Relacionamento Utilizador (1,1) para Construção (0,N)	35
4.2.10. Relacionamento Propriedade Construção (1,1) para Construção (1,N)	35
4.2.11. Relacionamento Encomenda (1,1) para Bloco (1,N)	35
4.2.12. Relacionamento Propriedade Bloco (1,1) para Bloco (1,N)	35
4.2.13. Relacionamento Propriedade Bloco (1,N) para Propriedade Construção (1,N)	36
<b>5. Esboço das Interfaces do Sistema</b>	<b>37</b>
5.1. Estrutura geral das interfaces do sistema	37
5.2. Caracterização das interfaces	37

5.2.1. Página Iniciar Sessão	37
5.2.2. Página Criar Conta	38
5.2.3. Página Inicial	38
5.2.4. Perfil	39
5.2.5. Fila de Espera	39
5.2.6. Catálogo	40
5.2.7. Adicionar Construção	41
5.2.8. Linha de Montagem	41
5.2.9. Construção em Produção	42
5.2.10. Construções Acabadas	43
5.2.11. Construção Acabada	43
5.2.12. Stock	44
5.2.13. Encomendar	44
5.2.14. Encomendas	45
5.2.15. Encomenda	46
<b>6. Conclusões e Trabalho Futuro</b>	<b>47</b>
6.1. Conclusões	47
6.2. Trabalho futuro	47
<b>Lista de Definições</b>	<b>48</b>
<b>Lista de Siglas e Acrónimos</b>	<b>49</b>

## Lista de Figuras

Figura 1: Maquete da aplicação a ser desenvolvida.	4
Figura 2: Parte do diagrama de GANTT utilizado.	5
Figura 3: Modelo de Domínio.	14
Figura 4: Modelo de Domínio - Relação (Linha de Montagem, Construção).	14
Figura 5: Modelo de Domínio - Relações agregadas ao bloco.	15
Figura 6: Diagrama de Componentes.	16
Figura 7: Componentes da Lógica de Negócio.	16
Figura 8: Componente Responsável pelo DL.	17
Figura 9: Componente Responsável pela UI.	17
Figura 10: Diagrama de Use Cases.	18
Figura 11: Remover Construção - Diagrama de Atividades.	20
Figura 12: Editar Perfil - Diagrama de Atividades.	22
Figura 13: Funcionamento da Linha de Montagem - Diagrama de Atividades.	23
Figura 14: Adicionar Construção - Diagrama de Atividades.	25
Figura 15: Criar Conta de Utilizador - Diagrama de Atividades.	26
Figura 16: Encomendar Stock - Diagrama de Atividades.	27
Figura 17: Visualizar Catálogo - Diagrama de Atividades.	29
Figura 18: Visualizar Construção Produzida - Diagrama de Atividades.	30
Figura 19: Visualizar Construções Produzidas - Diagrama de Atividades.	31
Figura 20: Esquema do Modelo Lógico.	32
Figura 21: Caracterização da tabela Utilizador.	33
Figura 22: Caracterização da tabela Encomenda.	34
Figura 23: Caracterização da tabela Bloco.	34
Figura 24: Caracterização da tabela Construção.	34
Figura 25: Caracterização da tabela Bloco_Produzir_Construção.	34
Figura 26: Caracterização da tabela PropriedadeBloco.	35
Figura 27: Caracterização da tabela PropriedadeConstrução.	35
Figura 28: Mockup da Página "Iniciar Sessão"	37
Figura 29: Mockup da Página "Criar Conta"	38
Figura 30: Mockup da Página "Inicial"	39
Figura 31: Mockup da Página "Perfil"	39
Figura 32: Mockup da Página "Fila de Espera"	40
Figura 33: Mockup da Página "Catálogo"	40
Figura 34: Mockup da Página "Adicionar Construção"	41
Figura 35: Mockup da Página "Linha de Montagem"	42
Figura 36: Mockup da Página "Construção em Produção"	42
Figura 37: Mockup da Página "Construções Acabadas"	43
Figura 38: Mockup da Página "Construção Acabada"	44
Figura 39: Mockup da Página "Stock"	44
Figura 40: Mockup da Página "Encomendar"	45
Figura 41: Mockup da Página "Encomendas"	46
Figura 42: Mockup da Página "Encomenda"	46

## Lista de Tabelas

Tabela 1: Remover Construção - Use Case.	19
Tabela 2: Editar Perfil - Use Case.	21
Tabela 3: Adicionar Construção - Use Case.	24
Tabela 4: Criar Conta de Utilizador - Use Case.	25
Tabela 5: Encomendar Stock - Use Case.	27
Tabela 6: Visualizar Catálogo - Use Case.	28
Tabela 7: Visualizar Construção Produzida - Use Case.	30
Tabela 8: Visualizar Construções Produzidas - Use Case.	31

# 1. Introdução

## 1.1. Contextualização

Desde o seu lançamento em 18 de novembro de 2011, o Minecraft conquistou milhões de jogadores em todo o mundo, tornando-se um verdadeiro fenômeno cultural. Com mais de 300 milhões de unidades vendidas, a sua popularidade transcendeu fronteiras, abrangendo diversas faixas etárias e demográficas. O jogo não é apenas um simples passatempo; transformou-se também num jogo onde a criatividade e a exploração se unem, permitindo que os jogadores construam mundos inteiros a partir de blocos.

Contudo, a complexidade do jogo e o extenso catálogo de blocos disponíveis podem ser intimidantes para novos jogadores. A variedade de materiais e as mecânicas de *crafting* tornam a experiência inicial desafiadora. Muitos jogadores reportam frustrações ao tentarem compreender como utilizar efetivamente esses recursos para criar construções impressionantes ou ferramentas essenciais. Este desafio pode levar à desistência de muitos que, de outra forma, poderiam aproveitar a riqueza de possibilidades oferecidas pelo jogo.

Neste contexto, a evolução contínua do jogo, com atualizações frequentes e novas mecânicas, só aumenta a necessidade de ferramentas que ajudem os jogadores a navegar por essa vasta experiência. A introdução de plataformas que simplificam o acesso a informações e recursos dentro do jogo poderá não só beneficiar os novatos, mas também reter os jogadores mais experientes, incentivando a partilha e a colaboração.

## 1.2. Motivação e Objectivos

A popularidade do Minecraft e a sua capacidade de fomentar a criatividade geraram uma enorme comunidade de jogadores, mas também expuseram desafios significativos para aqueles que desejam aproveitar ao máximo a experiência do jogo. Muitos novos jogadores, tal como referido anteriormente, enfrentam dificuldades na gestão do vasto catálogo de blocos e nas mecânicas de construção, o que pode levar à frustração e à desistência. Com isso em mente, surge a necessidade de uma solução que não apenas facilite o processo de construção, mas também enriqueça a experiência de jogo. Para isto, a empresa Mojang decidiu contratar um grupo de engenheiros de *Software*, o qual teria como objetivo principal desenvolver uma aplicação capaz de mitigar estes efeitos.

Tendo isto em mente, foram delineados um conjunto de objetivos que a aplicação tem alcançar com a sua implementação:

1. **Facilitar a introdução de novos utilizadores ao jogo;**
2. **Diminuir a taxa de desistência;**
3. **Promover a criatividade e a exploração.**

Com estes objetivos, pretende-se que a aplicação não apenas melhore a experiência de novos jogadores, mas também fortaleça a comunidade de Minecraft como um todo, permitindo que mais pessoas possam desfrutar plenamente das infinitas possibilidades que o jogo tem para oferecer.



### 1.3. Justificação e utilidade do sistema

A aplicação proposta visa proporcionar uma maneira inovadora de aprender a utilizar os blocos deste jogo, permitindo que os jogadores acessem a um conjunto de construções pré-definidas, dependendo do *stock* de blocos que possuam na sua conta. Essa abordagem não só simplifica a experiência de construção, mas também promove a exploração das diversas possibilidades criativas que o jogo oferece.

Através de um conjunto de construções pré-definidas e guias passo a passo, a aplicação permitirá que os novos jogadores compreendam rapidamente como construir e utilizar os diversos blocos disponíveis, reduzindo a curva de aprendizagem. Ao proporcionar um suporte contínuo e acessível, a aplicação visa manter o interesse dos novos jogadores, ajudando-os a superar os desafios iniciais e a desfrutar da experiência completa do Minecraft.

Além de facilitar o acesso ao jogo, o sistema traz várias vantagens económicas para a empresa. Ao criar uma plataforma onde os jogadores podem explorar novas construções e melhorar as suas capacidades dentro do jogo, a empresa pode explorar novos modelos de receita, como:

1. **Conteúdos *Premium*:** A plataforma pode oferecer tutoriais exclusivos ou funcionalidades adicionais para jogadores que estejam dispostos a pagar por um serviço mais completo;
2. **Publicidade Direcionada:** Como o site estará focado em jogadores de Minecraft, é possível vender espaço publicitário a empresas que queiram atingir esse público específico, como vendedores de acessórios para jogadores ou produtos relacionados ao jogo;
3. **Expansão do Catálogo:** A introdução de novas ideias de construção no site também pode incentivar o desenvolvimento de novos pacotes de conteúdo para o jogo, como novos blocos, criando um ciclo de inovação e lucro.

### 1.4. Estabelecimento da identidade do projeto

- **Identificação do Projeto:**
  - **Nome do projeto:** *Minebuilds* – Gestão de Construções do Minecraft;
  - **Descrição:** Aplicação Web para o jogo Minecraft;
  - **Empregador:** Mojang;
  - **Data de Início Prevista:** 15/09/2024;
  - **Data de Conclusão Estimada:** 20/01/2025;
- **Titulares do Projeto:**
  - **Coordenador:** Paulo Ferreira (a96268);
  - **Nomes:** Alex Sá (a104257), Alexandre Dias (a103994), Jorge Pereira (a104173), Paulo Ferreira (a96268), Rafael Fernandes (a104271);
  - **Contato:** a96268@alunos.uminho.pt;
- **Procedimentos e Recursos:**
  - **Recurso Físico:** Computadores;
  - **Recursos Digitais:** Clickup, Typst, draw.io, C#, JavaScript;
  - **Recursos Humanos:** Equipa de desenvolvimento;
- **Descrição do Projeto:** *Minebuilds* será desenvolvido com o intuito de proporcionar uma solução eficaz para a gestão e otimização de construções do Minecraft. Trata-se de uma ferramenta de automação de produção para os jogadores, permitindo o acesso a construções pré-definidas (*pre-builds*) e a gestão de recursos, facilitando a experiência de jogo.

O projeto inclui o desenvolvimento de uma interface web interativa, onde os usuários poderão:

- Selecionar construções pré-definidas;
- Gerir o stock de blocos necessário para cada uma das construções;
- Acompanhar, em tempo real, o progresso da montagem das construções disponíveis;

- **Escopo do projeto:** A aplicação visa otimizar o fluxo do trabalho dos jogadores no Minecraft. O sistema automatizará a cadeia de produção de construções, acompanhando todos os passos desde o *login* até à conclusão das múltiplas construções.

## 1.5. Identificação de recursos necessários

Ao longo do planeamento do projeto, foram identificados vários recursos essenciais para garantir qualidade durante a sua execução e manutenção. Por sua vez, esses recursos podem ser classificados em diferentes tipos, nomeadamente, físicos, digitais e humanos.

A realização deste projeto está a cargo de uma equipa de desenvolvimento composta por cinco elementos, sendo esta a principal responsável pela criação e manutenção de uma aplicação de qualidade. Para alcançar esse sucesso, são utilizados os seguintes recursos:

### Recursos Físicos:

- Computador

### Recursos Digitais:

- ClickUp

O ClickUp é uma aplicação que permite a gestão de projetos, contendo diversas funcionalidades que vão desde listas e quadros de tarefas, quadros brancos em simultâneo, documentos de texto partilhados, diagramas GANTT, entre outras. Esta foi escolhida pela equipa de desenvolvimento, já que não acarreta nenhum custo monetário adicional ao projeto, para além de facilitar a planificação da aplicação.

- Typst

O Typst é um sistema de composição de documentos gratuito que permite a colaboração simultânea de vários utilizadores. Este foi escolhido pela equipa de desenvolvimento, dado que o seu sistema de pré-visualização é superior a outros existentes no mercado, para além da facilidade de utilização.

- draw.io

O draw.io é uma aplicação que disponibiliza aos seus utilizadores várias ferramentas para a criação de diversos tipos de diagramas, com funcionalidades que permitem a colaboração em simultâneo. A escolha desta aplicação pela equipa de desenvolvimento deve-se à familiaridade e experiência obtida em trabalhos anteriores, tendo sido, no presente projeto, desenvolvido o diagrama da maquete do sistema através do draw.io.

- Visual Paradigm

O Visual Paradigm é uma aplicação projetada para auxiliar as equipas de desenvolvimento de *software* na gestão do processo de desenvolvimento de aplicações, permitindo a criação de vários tipos de diagramas. Esta foi utilizada pela equipa de desenvolvimento para a realização de alguns diagramas, como os de Casos de Uso, e para a definição do modelo de domínio.

- SQL Server Management Studio (SSMS)

O SQL Server Management Studio é um programa utilizado para configurar, gerir e administrar os componentes do Microsoft SQL Server. Este foi usufruído para criar as tabelas da futura base de dados, para além de ter permitido à equipa de desenvolvimento a criação do modelo lógico do sistema de dados.

### Recursos Humanos:

- Equipa de desenvolvimento

## 1.6. Maquete do sistema

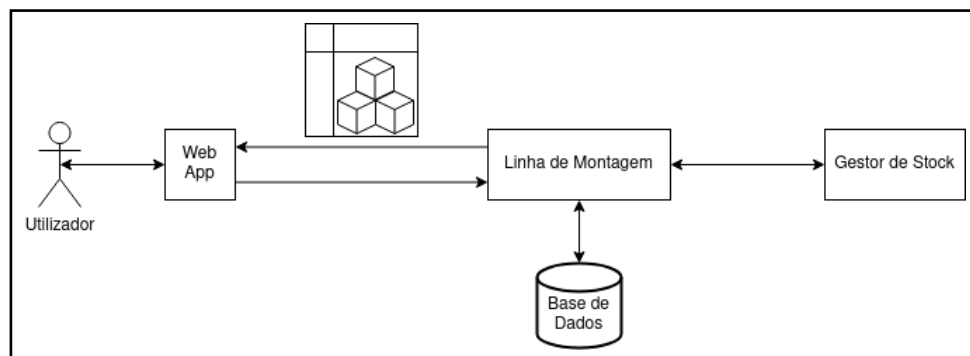


Figura 1: Maquete da aplicação a ser desenvolvida.

A maquete aqui apresentada representa uma visão geral dos componentes chave da arquitetura prevista para o projeto. É importante notar que a mesma foi criada com base em uma análise inicial superficial sobre pontos-chave recolhidos durante reuniões com os *stakeholders*, e que a estrutura final estará dependente de futuras alterações que os requisitos apresentados pelos *stakeholders* requererão.

Esta maquete serve de alicerce conceptual que irá evoluir ao longo do processo da análise do projeto, com o objetivo de facilitar a comunicação entre a equipa de desenvolvimento e os *stakeholders* durante as discussões iniciais do projeto.

### Componentes:

- *Web App*

A Aplicação Web, abreviada como *Web App*, é a interface utilizada pelos Utilizadores para interagir com o sistema final. Espera-se que o Utilizador seja capaz de aceder, visualizar e modificar os dados a que lhe sejam dado acesso. A Aplicação é atualizada em tempo real com base no estado do servidor, sem necessitar de um *reload* explícito do utilizador. Um utilizador apenas é capaz de aceder à sua linha de montagem e stock individuais, necessitando de se autenticar antes de ser possível interagir com os mesmos.

- Linha de Montagem

A Linha de Montagem é o módulo do sistema que produz vários produtos ao consumir e processar stock disponível em armazém, o qual obtém através de pedidos a um Gestor de Stock.

- Base de Dados

A Base de Dados é um sistema de armazenamento de dados persistente que armazene os dados necessários ao funcionamento da *Web App* e do Gestor de Stock.

- Gestor de Stock

O Gestor de Stock é o módulo que gere o armazém das Linhas de Montagem de um Utilizador, responsável por fornecer às mesmas o stock necessário ao seu funcionamento, armazenar a sua produção e repor as matérias-primas necessárias para a manutenção contínua da produção.

## 1.7. Definição de um conjunto de medidas de sucesso

A realização deste projeto baseia-se na avaliação equilibrada entre os aspetos positivos e negativos que este provoca. Ao analisar os mesmos, constata-se que o principal benefício é a união, motivação e o impulsionamento que a comunidade terá sobre o Minecraft, dado que a aplicação a ser criada levará a que os jogadores percebam o jogo de forma mais rápida e ágil, resultando numa redução de 25% na taxa

de desistências e no aumento do compromisso por parte dos utilizadores com o jogo, em cerca de 28%. Dessa forma, o aumento da satisfação do jogador, avaliada em 32% ao longo do primeiro ano, levará a uma maior divulgação do Minecraft, contribuindo para o aumento das vendas da empresa responsável. Adicionalmente, a implementação de conteúdos *premium*, a realização de publicidade e a expansão do catálogo de itens são fatores monetários adicionais que reforçam a realização deste projeto, refletindo um crescimento de 14% em receitas exteriores ao jogo. Em contrapartida, há despesas envolvidas, tais como os salários da equipa de desenvolvimento e a possível necessidade de serem usadas ferramentas pagas a nível digital. No entanto, a análise detalhada reflete que os benefícios associados ao desenvolvimento deste projeto superam os seus custos, concluindo que a aplicação tem tudo para ser um sucesso.

## 1.8. Plano de desenvolvimento (diagrama de GANTT)

Dada a complexidade deste projeto, embora de pequena escala em comparação com um projeto real, foi possível decompor o trabalho em várias etapas menores. Para gerir a distribuição de tarefas e o acompanhamento do progresso, optou-se pela elaboração de um diagrama de GANTT. Este recurso permite ao grupo controlar as atividades já realizadas e as que ainda estão pendentes, além de possibilitar a divisão de tarefas entre os membros do grupo.

Inicialmente, considerou-se a utilização de um *template* de diagrama de GANTT em Microsoft Excel, com o objetivo de agilizar a criação e definição do mesmo. No entanto, após algumas pesquisas, decidiu-se utilizar a aplicação ClickUp. Esta ferramenta, além de permitir a criação de um diagrama deste tipo, oferece também diversas funcionalidades relacionadas com *workspaces*, listas de tarefas e anotações, tal como referido na **Secção 1.5**.

As funcionalidades relativas ao GANTT que tornaram o uso da aplicação apelativo para o grupo foram:

- **Dependência de Tarefas:** Uma tarefa não pode ser considerada iniciada se depender da conclusão de uma tarefa anterior. Isto ajuda o grupo a planear melhor a ordem das atividades, evitando que, por descuido, se inicie algo que depende de uma tarefa ainda não concluída.
- **Atribuição de Membros a Tarefas:** Cada tarefa pode ter um responsável designado, o que, em conjunto com as notificações via e-mail, mantém todos os envolvidos informados sobre o que deve ser feito e os prazos associados.

Para melhor entendimento destes sistemas, a **Figura 2** mostra um exemplo de dependência entre o Debate Inicial e a Contextualização, demonstrando a primeira funcionalidade referida. A outra representada é a distribuição das tarefas pelos elementos e a representação do estado das mesmas, sendo eles, *open*, *in progress* e *close*.

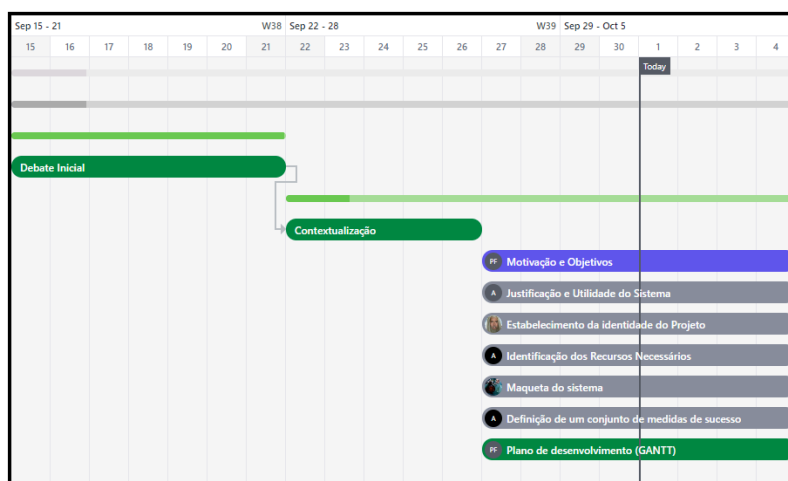


Figura 2: Parte do diagrama de GANTT utilizado.

Essas funcionalidades da aplicação, juntamente com as características inerentes ao diagrama de GANTT, foram fundamentais para o desenvolvimento deste trabalho, permitindo que a distribuição das tarefas e o progresso geral do projeto ocorresse de maneira fluida e natural, sem grandes contratempos.

## 2. Levantamento e Análise de Requisitos

### 2.1. Apresentação da estratégia e método

A fase de levantamento e análise de requisitos é reconhecida por ser uma das mais cruciais no desenvolvimento de um projeto de *software*, sendo que esta poderá ser realizada através de diversas abordagens e metodologias. No caso em questão, a equipa de desenvolvimento optou por inicia-la através de uma reunião colaborativa com a equipa de vendas da Mojang. Durante essa reunião, foram discutidos vários fatores relacionados com o declínio nas vendas do Minecraft, bem como críticas negativas por parte dos jogadores, que proporcionaram à equipa de desenvolvimento a identificação dos motivos principais para a baixa adesão e comprometimento dos jogadores face ao jogo.

Além disso, a equipa de trabalho recorreu a uma abordagem mais direta com os jogadores, através da análise de opiniões em diversos fóruns e da análise de resultados provenientes de um formulário, tendo sido este disponibilizado na página web da Mojang pela equipa de vendas, no qual os utilizadores podiam descrever a sua experiência de jogo até ao momento. Esta interação direta forneceu à equipa uma compreensão mais profunda das necessidades dos jogadores, passando por ser uma das chaves fundamentais para o sucesso da aplicação, já que estes são a entidade mais importante do sistema.

Por último, a avaliação de cenários foi outra técnica utilizada para identificar e detalhar os requisitos funcionais, tendo sido esta a abordagem que proporcionou à equipa de desenvolvimento uma melhor perceção relativamente às necessidades comportamentais e funcionais da aplicação, garantindo que o *software* cumpre com os objetivos estabelecidos e que ofereça um sistema que vá ao encontro das expectativas dos utilizadores.

### 2.2. Descrição geral dos requisitos (funcionais e não funcionais) levantados

Após a recolha dos requisitos usando os métodos previamente referidos, sentiu-se a necessidade de os dividir em requisitos funcionais e não funcionais. Esta prática já não é nova no desenvolvimento de *software*, e é de elevada importância para um bom entendimento entre a equipa de desenvolvimento e os *Stakeholders*, além de ser útil para os primeiros terem uma noção correta, concisa e não ambígua do que é necessário implementar. Para fazer a separação de cada um dos tipos de requisitos, seguiu-se a definição utilizada no livro *Software Engineering*. Segundo o autor:

**“Requisitos funcionais** são declarações de serviços que o sistema deve fornecer, como o sistema deve reagir a determinadas entradas e como o sistema se deve comportar em situações específicas.”

**“Requisitos não funcionais** são restrições aos serviços ou funções oferecidas pelo sistema.”

Nesta secção vamos mostrar uma definição de requisitos utilizando linguagem natural, porém, é possível utilizar outras notações para o mesmo efeito, sendo a notação gráfica ou especificações matemáticas algumas delas. Contudo, vale a pena mencionar que, normalmente, a segunda carece de especificação de requisitos não funcionais, e a terceira não é facilmente entendida pelo cliente.

### 2.2.1. Requisitos funcionais

Dentro destes requisitos foi feito outro agrupamento além daquele aconselhado por *Ian Sommerville*. Através da maqueta foi possível identificar uma separação do *software* em quatro partes, Autenticação, Perfil, Linha de Montagem e Stock. Esta separação acabou por ser útil para distribuir os requisitos por áreas mais específicas, podendo até ser feita a associação destas partes a subsistemas.

#### 2.2.1.1. Autenticação

##### RF01 - Criar conta na aplicação

- Requisitos do utilizador:
  1. A aplicação deve permitir a um utilizador criar uma conta para utilizar esta aplicação.
- Requisitos do sistema:
  1. O sistema deverá solicitar as credenciais ao utilizador, sendo estas um e-mail, um *username* e uma palavra-passe;
  2. O sistema não deverá permitir vários utilizadores terem o mesmo *username* ou e-mail.
  3. O sistema deverá alertar o utilizador caso seja inserido um *username* ou e-mail que já foi utilizado por outro utilizador.

##### RF02 - Iniciar sessão na aplicação

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador aceder à sua conta através das suas credencias.
- Requisitos do sistema:
  1. O sistema deverá validar os dados introduzidos pelo utilizador e fornecer o acesso à sua conta;
  2. O sistema deverá alertar o utilizador e impedir o acesso à conta caso as credenciais sejam inválidas.

##### RF03 - Terminar sessão na aplicação

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador terminar sessão.
- Requisitos do sistema:
  1. O sistema deverá permitir ao utilizador terminar sessão;
  2. O sistema deverá invalidar a sessão ativa assim que a sessão for encerrada;

#### 2.2.1.2. Perfil

##### RF04 - Visualizar perfil

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar a informação do seu perfil.
- Requisitos do sistema:
  1. O sistema deverá ter acesso às informações do utilizador.
  2. O sistema deverá mostrar as informações ao utilizador.

##### RF05 - Editar perfil

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador editar as informações do seu perfil.
- Requisitos do sistema:
  1. O sistema deverá ter acesso às informações do utilizador.
  2. O sistema deverá permitir ao utilizador alterar as suas informações, tais como *username*, e-mail e palavra-passe;
  3. O sistema deverá validar a unicidade dos dados introduzidos, não permitindo que vários utilizadores possuam o mesmo *username* ou e-mail.
  4. O sistema deverá cancelar a operação e alertar o utilizador caso não se comprove a unicidade dos dados.

5. O sistema deverá atualizar o perfil do utilizador.

### **2.2.1.3. Linha de Montagem**

#### **RF06 - Visualizar o catálogo de construções**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar o catálogo com todas as construções.
- Requisitos do sistema:
  1. O sistema deverá ter acesso a uma lista de construções.
  2. O sistema deverá mostrar as construções ao utilizador.
  3. O sistema deverá mostrar, de forma distinta, as construções que o utilizador não possua materiais suficientes para produzir.

#### **RF07 - Visualizar os materiais de uma construção**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar a lista de materiais necessários para produzir uma construção.
- Requisitos do sistema:
  1. O sistema deverá ter acesso aos materiais associados a uma construção.
  2. O sistema deverá mostrar os materiais ao utilizador.

#### **RF08 - Visualizar construção produzida**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar uma construção produzida.
- Requisitos do sistema:
  1. O sistema deverá ter acesso às construções produzidas.
  2. O sistema deverá mostrar ao utilizador as construções produzidas.
  3. O sistema deverá permitir ao utilizador selecionar uma construção produzida.
  4. O sistema deverá permitir ao utilizador visualizar o guião para produzir a construção ou o estado final da construção.

#### **RF09 - Visualizar construção em produção**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar uma construção em produção.
- Requisitos do sistema:
  1. O sistema deverá ter acesso às construções em produção numa linha de montagem.
  2. O sistema deverá mostrar ao utilizador as construções em produção.
  3. O sistema deverá permitir ao utilizador selecionar uma construção em produção.
  4. O sistema deverá mostrar ao utilizador o estado atual da construção.

#### **RF10 - Adicionar construção**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador adicionar uma construção à fila de espera de uma linha de montagem.
- Requisitos do sistema:
  1. O sistema deverá ter acesso aos materiais associados a uma construção.
  2. O sistema deverá validar se o utilizador possui blocos suficientes em stock para iniciar a produção de uma construção.
  3. O sistema deverá alertar o utilizador caso não possua blocos suficientes, mostrando os blocos em falta e não adicionando a construção.
  4. O sistema deverá adicionar a construção à fila de espera de uma linha de montagem.
  5. O sistema deverá alertar o utilizador de que a construção está na fila de espera da linha de montagem.



#### **RF11 - Remover construção**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador remover uma construção da fila de espera de uma linha de montagem.
- Requisitos do sistema:
  1. O sistema deverá permitir ao utilizador visualizar as construções na fila de espera de uma linha de montagem.
  2. O sistema deverá permitir ao utilizador selecionar uma construção da fila de espera.
  3. O sistema deverá remover a construção selecionada, desde que não esteja em produção na linha de montagem.
  4. O sistema deverá alertar o utilizador caso não tenha sido possível remover a construção selecionada.

#### **RF12 - Ver construções na fila de espera**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar uma lista das construções que se encontrem na fila de espera.
- Requisitos do sistema:
  1. O sistema deverá ter acesso às construções em fila de espera.
  2. O sistema deverá permitir ao utilizador visualizar as construções em fila de espera.

#### **RF13 - Ver construções na linha de montagem**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar uma lista das construções que se encontrem na linha de montagem.
- Requisitos do sistema:
  1. O sistema deverá ter acesso à lista de construções que se encontrem na linha de montagem.
  2. O sistema deverá permitir ao utilizador visualizar as construções que se encontrem na linha de montagem.

#### **RF14 - Ver construções produzidas**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar uma lista das construções já produzidas.
- Requisitos do sistema:
  1. O sistema deverá ter acesso às construções que já tenham sido produzidas.
  2. O sistema deverá permitir ao utilizador visualizar as construções que já tenham sido produzidas.

#### **RF15 - Funcionamento da Linha de Montagem**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador adicionar uma construção à linha de montagem, tal como estipulado no requisito funcional número 10 (RF10).
- Requisitos do sistema:
  1. O sistema deverá poder manipular o funcionamento de uma linha de montagem.
  2. O sistema deverá, após adicionar uma construção à fila de espera de uma linha de montagem, avaliar a disponibilidade da primeira estação dessa mesma linha.
  3. O sistema deverá, no caso dessa estação estar vazia, passar a construção da fila de espera para essa estação.
  4. O sistema deverá, no caso dessa estação estar ocupada, manter a construção na fila de espera até que a estação esteja vazia.
  5. O sistema deverá, após o tempo de produção parcial referente à construção ter passado, validar a disponibilidade da próxima estação.
  6. O sistema deverá, no caso da próxima estação estar vazia, passar a construção para essa estação.
  7. O sistema deverá, no caso da próxima estação estar ocupada, manter a construção na estação atual mantendo-a, desta forma, ocupada até que a próxima estação esteja vazia.

8. O sistema deverá repetir o processo descrito nas últimas duas etapas até chegar à última estação de produção da linha de montagem.
9. O sistema deverá, quando a linha de montagem terminar a produção parcial da construção na sua última estação, colocar a construção produzida na lista de construções produzidas.

#### **2.2.1.4. Stock**

##### **RF16 - Visualizar stock**

- Requisitos do utilizador:
  1. A aplicação deve permitir ao utilizador visualizar o seu stock.
- Requisitos do sistema:
  1. O sistema deverá ter acesso ao stock associado a um utilizador.
  2. O sistema deverá mostrar ao utilizador o seu stock.

##### **RF17 - Encomendar stock**

- Requisitos do utilizador:
  1. A aplicação deve permitir a um utilizador encomendar stock.
- Requisitos do sistema:
  1. O sistema deverá ter acesso aos materiais e respetivos tempos de espera de aquisição.
  2. O sistema deverá permitir ao utilizador adquirir materiais para o seu stock.

#### **2.2.2. Requisitos não funcionais**

##### **RNF01 - Uptime entre 95 e 99%**

O sistema desenvolvido precisa de garantir um alto nível de disponibilidade para os seus utilizadores, tendo de assegurar um tempo de atividade entre 95% e 99%, possibilitando, no máximo, a um período de inatividade de 88 a 438 horas por ano. O alto nível exigido permite garantir aos utilizadores uma redução drástica nas consequências durante os períodos de manutenção ou períodos marcados por problemas técnicos, para além de fortalecer uma confiança entre o sistema e os seus utilizadores, através da melhoria da qualidade da experiência dos mesmos.

##### **RNF02 - Interface simples e intuitiva**

A interface da aplicação deverá ser clara e fácil de usar, facilitando o acesso a todas as funcionalidades de maneira intuitiva. A simplicidade da interface permite que os utilizadores realizem tarefas com eficiência, reduzindo o tempo gasto a aprender a utilizar a aplicação. Com estas características, a interface proporciona uma experiência amigável para o utilizador, com navegação fluida e design organizado, levando a um maior nível de produtividade e satisfação.

##### **RNF03 - Utilização de tecnologias como C#, .NET e MySQL**

Para a elaboração do sistema, é de carácter obrigatório o uso da framework .NET, utilizando consequentemente a linguagem de programação C#. Para a gestão da base de dados, deverá ser usado o Microsoft SQL Server, garantindo um alto nível de integração entre as tecnologias. A escolha destas tecnologias oferece uma base sólida, com suporte robusto para o desenvolvimento da aplicação, bem como facilita a manutenção e futuras expansões do sistema.

## **2.3. Validação dos requisitos estabelecidos**

Para garantir que os requisitos estabelecidos, tanto funcionais quanto não funcionais, fossem adequadamente definidos e que pudessem ser corretamente implementados, foi utilizada uma abordagem de validação estruturada, baseada nas seguintes práticas:

- **Validação interna pela equipa de desenvolvimento**

Antes de qualquer validação externa, realizou-se uma revisão interna dos requisitos pela equipa de desenvolvimento, verificando a coerência, viabilidade técnica e conformidade com os objetivos do projeto. Esse processo colaborativo garantiu o alinhamento de todos os membros quanto às especificações e expectativas do *software*.

- **Revisão com Stakeholders**

Após a validação interna, os requisitos foram discutidos e revistos com as partes interessadas, como a equipa de vendas da Mojang. Para garantir a precisão e relevância dos requisitos finais, ocorreram reuniões regulares, onde cada funcionalidade proposta foi avaliada quanto à sua viabilidade e impacto nas metas de negócio.

### 3. Especificação e Modelação do *Software*

Neste capítulo, vão ser apresentados e explicados os diagramas desenvolvidos para melhor compreender os aspetos comportamentais e estruturais da aplicação. Durante o processo de desenvolvimento dos mesmos, foi sempre do interesse da equipa de desenvolvimento seguir as práticas mais corretas relacionadas a desenvolvimento de *software*. Com isto em conta, a modelação do *software* é um passo crucial para garantir que todos os membros da equipa tenham uma visão clara e partilhada do sistema, facilitando a comunicação e evitando erros de interpretação.

Após primeira análise, construiu-se um modelo de domínio cujo propósito é fornecer à equipa de desenvolvimento um entendimento profundo da área de aplicação do *software*. Este modelo permite identificar as principais entidades e relações entre elas, servindo como um guia fundamental para as etapas subsequentes de desenvolvimento. Em seguida, foi elaborado um conjunto de diferentes tipos de diagramas, cada um com um papel específico:

**Diagrama de Componentes:** Este diagrama define a estrutura do *software*, detalhando a organização dos principais componentes e como eles se relacionam. Ele é essencial para compreender a arquitetura do sistema e a interação entre os seus módulos.

**Diagrama de *Use Cases* e Diagrama de Atividades:** Estes diagramas focam-se nos aspetos comportamentais do sistema. O diagrama de *Use Cases* representa as interações entre os atores e o sistema, mostrando detalhadamente os diferentes casos de uso. O diagrama de Atividades, por sua vez, descreve o fluxo das operações através da ilustração das suas sequências e da lógica dos processos.

Seguindo uma filosofia correta de desenvolvimento de *software*, a distinção entre estrutura e comportamento é fundamental para que a equipa compreenda tanto a arquitetura quanto a funcionalidade do *software*, garantido que a implementação atenda adequadamente aos requisitos especificados.

#### 3.1. Apresentação geral da especificação

Nesta secção, apresenta-se uma visão geral da especificação do sistema, com o intuito de comunicar de forma clara os principais elementos e estruturas que compõem o domínio do *software*. O modelo de domínio é por isso um dos principais componentes desta especificação, representando os conceitos e relacionamentos essenciais da área de aplicação. Este modelo atua como uma base de referência para o entendimento conjunto entre os membros da equipa de desenvolvimento, o que enfatiza a ideia de que a comunicação precisa e uma correta documentação facilita o entendimento do sistema.

Logo abaixo, a Figura 3 exibe o modelo de domínio, detalhando as entidades e associações entre elas. Este modelo fornece uma visão de alto nível do sistema, capturando as informações essenciais para apoiar as decisões de design e desenvolvimento.

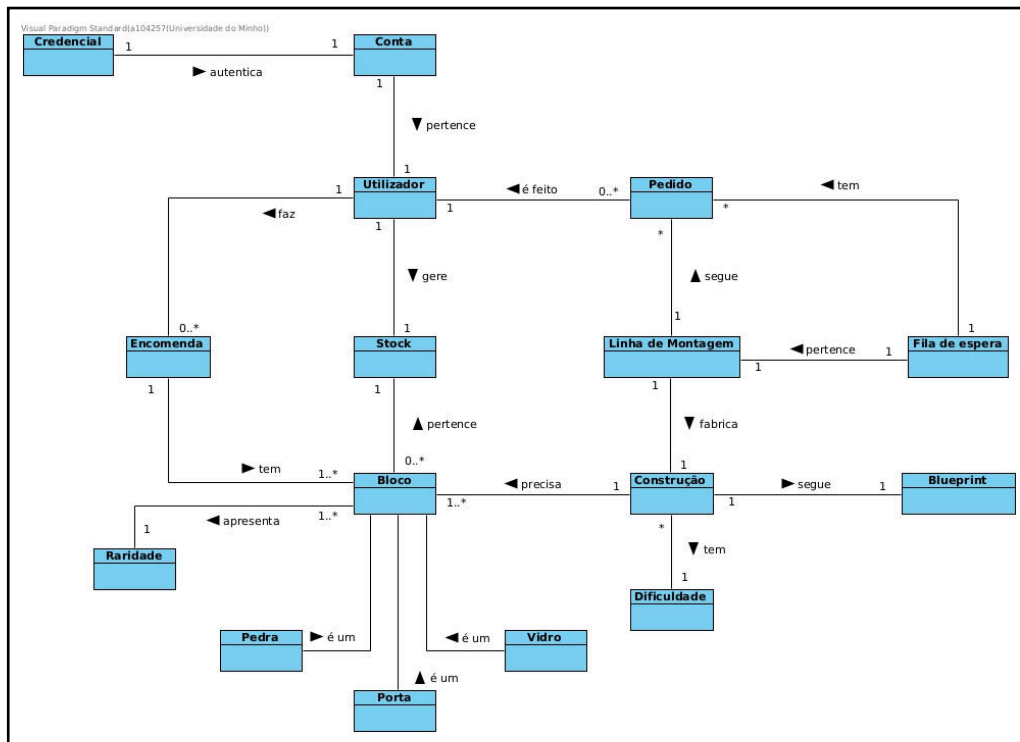


Figura 3: Modelo de Domínio.

Utilizando a Figura 3 como referência vamos agora fazer uma análise mais detalhada de algumas das várias entidades presentes no modelo de domínio e das relações que elas estabelecem. Vamos começar a olhar, por exemplo, para a linha de montagem. No contexto da aplicação, ou seja, no domínio do *software*, é de senso comum que uma linha de montagem irá produzir algum objeto. Isto transcreve-se facilmente no modelo de domínio apresentado na relação: uma Linha de Montagem fabrica uma Construção (isto tendo em conta que a construção é o nosso objeto). Se olharmos para a Figura 4 podemos ver a transcrição direta disto para o modelo de domínio, acrescentando unicamente a noção de que uma construção é fabricada por unicamente uma linha de produção.

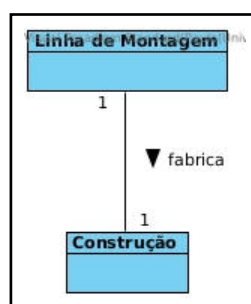


Figura 4: Modelo de Domínio - Relação (Linha de Montagem, Construção).

Para ilustrar uma estrutura mais complexa, começemos por analisar a entidade Bloco, conforme detalhado na Figura 5. Neste contexto, um bloco tem um conjunto de relações específicas, incluindo uma característica de raridade. Além disso, o modelo identifica subtipos de bloco, como Pedra, Porta e Vidro, que representam diferentes tipos de blocos usados nas construções.

As relações que este estabelece com outras entidades ajudam a definir a sua utilidade e integração no sistema:

- Uma construção é composta por diversos blocos, o que revela que estes são fundamentais para o processo de fabricação.
- Todos os blocos pertencem a um *stock*, o que permite identificar uma ideia de armazenamento.
- Uma encomenda contém múltiplos blocos, o que garante a ideia de que é possível adquirir blocos no sistema.

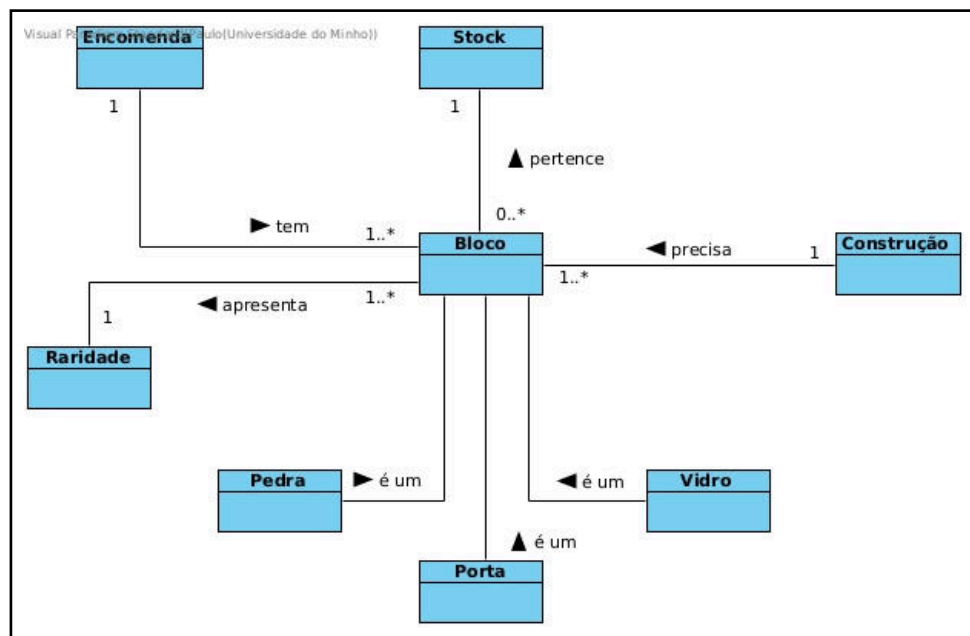


Figura 5: Modelo de Domínio - Relações agregadas ao bloco.

Com estes dois exemplos, podemos concluir que esta estrutura composta por entidades e associações permite que a equipa de desenvolvimento tenha uma visão abstrata e clara das dependências dentro do sistema, o que permite, por sua vez, que as seguintes etapas de desenvolvimento sigam sem grande conflito de opiniões sobre a terminologia usada no domínio e sobre as relações entre as principais entidades.

## 3.2. Aspetos estruturais

Após a definição do modelo de domínio, a equipa de desenvolvimento identificou a necessidade de descrever e documentar os aspetos estruturais do software. Para isso, utilizam-se modelos estruturais, que fornecem uma representação mais próxima da arquitetura real do sistema, detalhando os componentes que o constituem, os subsistemas envolvidos e as interações entre eles. Esses modelos não podem ser transcritos diretamente do modelo de domínio, pois este último foca-se na compreensão do léxico e das interações dos elementos dentro do domínio de aplicação, algo que nem sempre reflete diretamente a estrutura do software.

A descrição desses aspetos pode ser feita utilizando a Unified Modeling Language (UML), por meio de diagramas estruturais, também conhecidos como diagramas estáticos. Embora existam vários tipos de diagramas para esse propósito, como diagramas de classes, diagramas de pacotes e diagramas de *deployment*, a equipa optou por utilizar o diagrama de componentes para ilustrar a estrutura da aplicação, conforme representado na Figura 6. Embora o diagrama de classes também tenha sido considerado, ele exige um nível de detalhe e comprometimento que não se adequava às necessidades e limitações da equipa nesta fase do desenvolvimento.

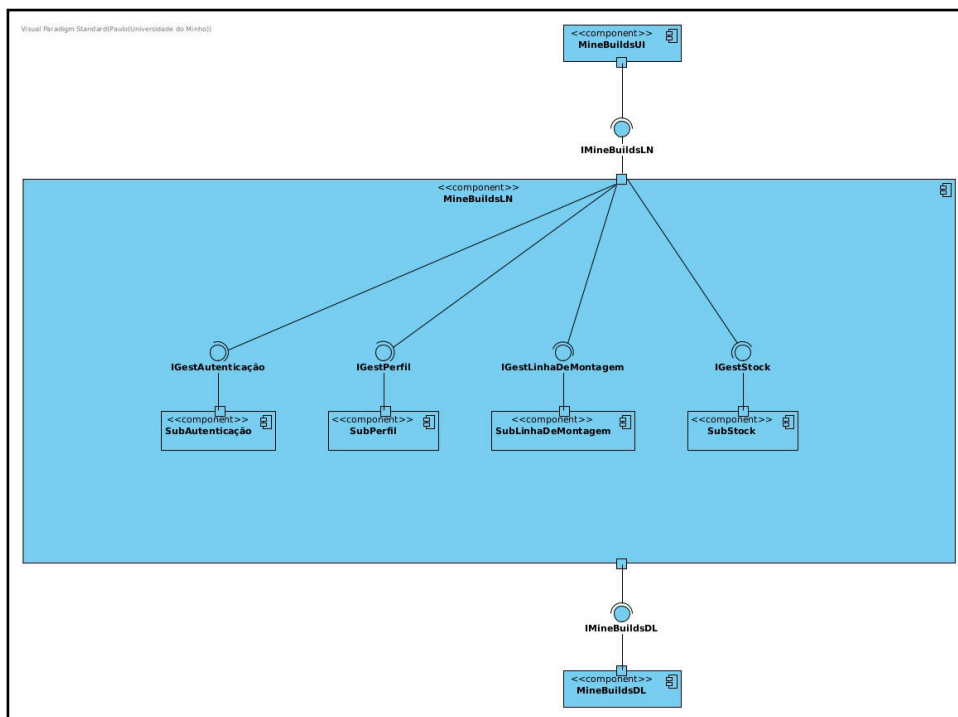


Figura 6: Diagrama de Componentes.

Com o desenvolvimento deste diagrama, foi possível identificar vários subsistemas da aplicação, os quais podem ser diretamente relacionados à descrição dos requisitos na Secção 2.2, dado que esta foi estruturada para separar os requisitos por áreas de aplicação.

O componente mais abrangente, denominado MineBuildsLN, refere-se à lógica de negócio(LN) da aplicação e está representado a maior detalhe na Figura 7. Este componente inclui todos os elementos responsáveis por executar funções ou métodos diretamente relacionados à LN, ou seja, componentes que interagem com os dados da aplicação, realizando operações sobre eles ou executando algoritmos essenciais ao funcionamento do sistema. O componente MineBuildsLN abrange quatro subsistemas fundamentais: o componente de autenticação (**subAutenticação**), o de perfil (**subPerfil**), o da linha de montagem (**subLinhaDeMontagem**) e o de gestão de stock (**subStock**).

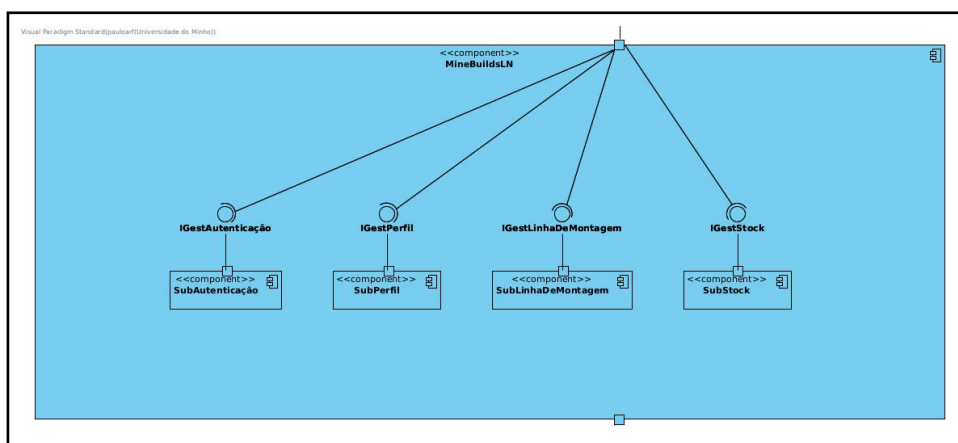


Figura 7: Componentes da Lógica de Negócio.

O componente de **autenticação** é responsável pelos métodos ligados ao processo de autenticação do software, abrangendo desde a criação de contas e o início de sessão até o encerramento da sessão.

O componente de **perfil** gere as operações sobre as informações do utilizador, incluindo alterações a dados como *username*, email, entre outros.

O componente da **linha de montagem** é um dos elementos centrais da aplicação, sendo responsável pela produção das construções e pela gerência das filas de espera associadas.

O componente de **stock** será encarregado de implementar os métodos necessários para a gestão do inventário, possibilitando tanto a encomenda de blocos quanto o controlo da quantidade de blocos disponível para cada utilizador.

Ao observarmos mais atentamente o diagrama, podemos verificar que cada um dos componentes mencionados está associado a um símbolo denominado **IGes<<nome do componente>>**. Isto indica que cada componente deve ter uma interface que facilite a sua gestão. Em outras palavras, para cada componente, deve ser desenvolvida uma interface que permita a interação com componentes externos à LN, funcionando como um *wrapper* para os métodos existentes em cada um deles.

Aproveitando a menção às interfaces, podemos agora abordar os componentes externos à LN, Figura 8 e Figura 9. A equipa de desenvolvimento, depois de alguns debates acerca do necessário para o bom funcionamento da aplicação, identificou o componente **MineBuildsUI**, responsável pela interação com o utilizador, e o **MineBuildsDL**, que gerência a ligação com a base de dados.

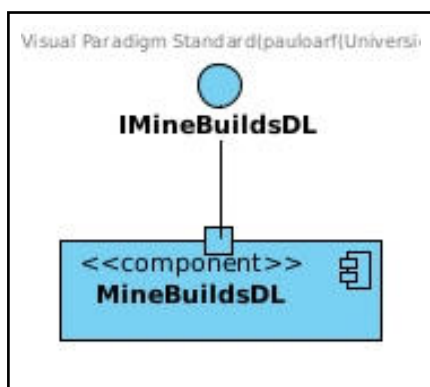


Figura 8: Componente Responsável pelo DL.

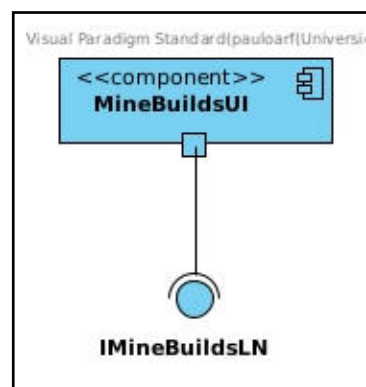


Figura 9: Componente Responsável pela UI.

O componente **MineBuildsUI** é encarregado de todos os métodos relacionados à interface do utilizador (UI - User Interface), enquanto o componente **MineBuildsDL** é responsável pela interação com a base de dados, incluindo a extração, atualização e exclusão das informações necessárias. Embora o **MineBuildsDL** seja descrito como uma interface ligada ao Data Layer (DL), os métodos exatos que serão utilizados ainda não estão completamente definidos, deixando em aberto a possibilidade de utilização de padrões como DAO (Data Access Object), ORM (Object-Relational Mapping), entre outros.

Com a conclusão deste diagrama, a equipa de desenvolvimento obteve uma visão mais clara da estrutura e dos módulos presentes na aplicação, o que facilita também a definição dos aspetos comportamentais, abordados na secção seguinte.



### 3.3. Aspectos comportamentais

Com a definição dos aspectos estruturais do sistema concluída, a equipa de desenvolvimento avançou para a etapa de especificação dos aspectos comportamentais. Similar ao que aconteceu na secção anterior, diversos diagramas poderiam ser utilizados para representar o comportamento do sistema, como o **diagrama de use cases**, **diagramas de atividades**, **diagramas de sequência**, **diagramas de máquinas de estado**, entre outros. No entanto, para este projeto específico, a equipa optou pela utilização dos diagramas de *use cases* e diagramas de atividades como as principais ferramentas para representar os comportamentos do sistema.

A definição dos *use cases* foi realizada através da criação de tabelas que descrevem de maneira clara e estruturada o comportamento esperado de cada caso de uso, organizadas em tópicos que incluem os **atores envolvidos**, o **fluxo normal**, as **pré-condições**, os **fluxos de exceção** e os **fluxos alternativos**, quando aplicável. Essas tabelas permitem uma visão concisa de como os *use cases* devem ser executados no sistema, facilitando tanto a implementação quanto a validação dos mesmos durante o processo de desenvolvimento.

Além das tabelas, os **diagramas de atividades** foram desenvolvidos para complementar a descrição dos *use cases*. Esses diagramas são particularmente úteis para ilustrar fluxos de controle mais complexos, que não poderiam ser completamente explicados através das tabelas. Eles ajudam a visualizar a sequência de ações, decisões, ciclos e paralelismos que ocorrem durante a execução de um *use case*, oferecendo uma representação gráfica do comportamento dinâmico do sistema. Um ótimo exemplo disso seria a **Figura 10**.

Para cada *use case* e diagrama de atividades, foi realizada uma associação direta ao requisito funcional que originou o caso de uso. Essa associação proporciona uma compreensão mais clara do processo de tomada de decisão da equipa de desenvolvimento, garantindo que todos os requisitos sejam devidamente atendidos no *design* do sistema. Isso também facilita a comunicação entre os membros da equipa e os *stakeholders*, já que cada diagrama pode ser diretamente correlacionado aos requisitos originais.

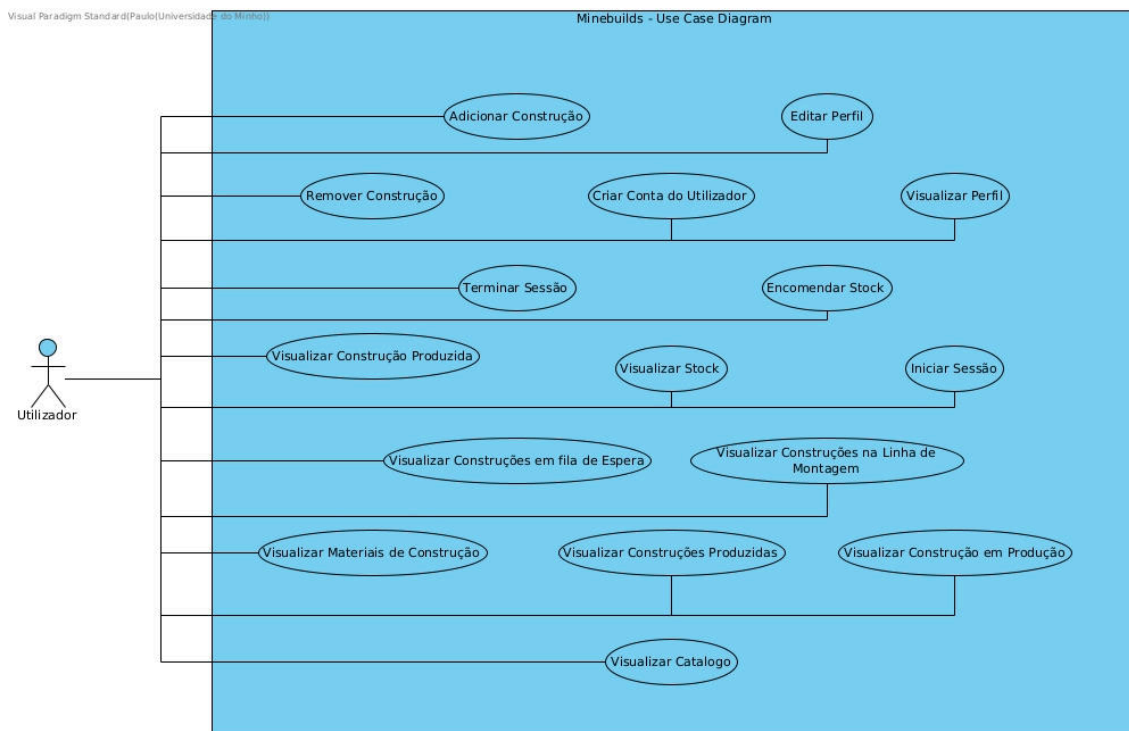


Figura 10: Diagrama de Use Cases.

O diagrama geral, representado na Figura 10, ilustra todos os *use cases* identificados a partir dos requisitos funcionais do sistema. Por exemplo, os requisitos 05, 10, 14 e 17, que correspondem a funcionalidades

como **editar perfil**, **adicionar construção**, **ver construções produzidas** e **encomendar stock**, apresentam uma transcrição direta para *use cases*. Cada um desses *use cases* foi desdobrado em tabelas detalhadas e diagramas de atividades que descrevem o fluxo de execução, incluindo os passos envolvidos e as possíveis exceções ou variações no processo.

No entanto, é importante destacar que nem todos os requisitos funcionais se traduzem diretamente em *use cases*. Alguns, como o requisito RF15 e outros requisitos mais técnicos, envolvem processos internos do sistema que não requerem a interação de um ator específico e, portanto, não geram um *use case*. Estes requisitos podem, por exemplo, referir-se a validações de sistema, regras de negócio ou interações com componentes externos que não envolvem a execução de uma tarefa por parte de um utilizador. Para tais requisitos, embora não sejam criados *use cases* específicos, eles ainda são documentados e considerados no *design* da arquitetura e nos diagramas de atividades quando necessário.

Nas páginas seguintes desta secção, serão explorados em detalhe alguns dos *use cases* e diagramas de atividades mais representativos do sistema, com uma explicação pormenorizada dos fluxos de trabalho, atores envolvidos e possíveis exceções ou alternativas. Para os *use cases* restantes, será apresentada uma representação simplificada, com imagens dos diagramas e explicações mais curtas, a fim de fornecer uma visão geral do comportamento do sistema sem uma análise tão detalhada.

Tabela 1: Remover Construção - Use Case.

USE CASE:	Remover Construção
DESCRIÇÃO:	Ator remove uma construção da fila de espera.
CENÁRIOS:	Cenário 1: Um utilizador seleciona uma construção e remove-a da fila de espera.
PRÉ-CONDIÇÃO:	O Ator tem sessão iniciada e está a visualizar as construções da fila de espera.
PÓS-CONDIÇÃO:	A construção não é produzida.
FLUXO NORMAL:	1. Ator indica que quer remover uma construção da fila de espera. 2. Sistema remove a construção da fila de espera.
FLUXO DE EXCEÇÃO (1):	[A construção selecionada passou a ser produzida.] (Passo 2) 2.1. Sistema informa que a construção passou a ser produzida e que não pode ser removida. 2.2. Sistema cancela a operação.

Olhando, por exemplo, para o Tabela 1, é possível observar como foi estruturada a descrição de um *use case*, começando pelo seu nome, descrição geral e cenários, e acabando no fluxo normal e no fluxo de exceção.

Primeiramente, a **descrição** fornece uma visão geral do *use case*, esclarecendo a sua finalidade e a utilidade no contexto do sistema. Ela explica de forma sucinta o papel do *use case* dentro da aplicação, proporcionando um entendimento inicial do que o sistema deve fazer quando este caso for executado.

O **cenário** descreve um exemplo prático e específico da realização do *use case*, oferecendo um caso real de uso. Este exemplo serve para ilustrar como o *use case* será efetivamente utilizado dentro do sistema, permitindo uma melhor compreensão de sua aplicação no dia a dia do utilizador.

As **pré-condições** e **pós-condições** são dois elementos cruciais que definem os requisitos necessários para a execução do *use case*. A pré-condição especifica os requisitos que devem ser atendidos antes que o *use case* possa ser executado, enquanto a pós-condição descreve o estado esperado do sistema após a execução bem-sucedida do *use case*. Essas condições são fundamentais para garantir que o comportamento do sistema esteja de acordo com as expectativas.

Em seguida, o **fluxo normal** é um dos componentes mais importantes de um *use case*. Ele descreve, de forma detalhada e sequencial, o que o ator deve fazer, assim como as respostas do sistema, assumindo que tudo ocorra conforme o esperado. Esse fluxo é estruturado de maneira a guiar os passos necessários para que o *use case* seja completado com sucesso, considerando as interações entre o utilizador e o sistema em cada etapa.

O **fluxo de exceção** é uma parte crítica do *use case* e descreve o comportamento do sistema quando algo inesperado ocorre durante a execução do fluxo normal. No exemplo apresentado, se, em determinado ponto do fluxo normal, a construção selecionada já foi iniciada para produção, ou seja, “A construção selecionada passou a ser produzida”, o sistema entra em um fluxo de exceção. Isso significa que a ação de remoção da construção da fila de espera não pode mais ser realizada, pois o estado da construção foi alterado e não é mais possível modificar a sua posição na fila. O fluxo de exceção garante que o sistema lida corretamente com este tipo de situação, evitando inconsistências no estado da aplicação.

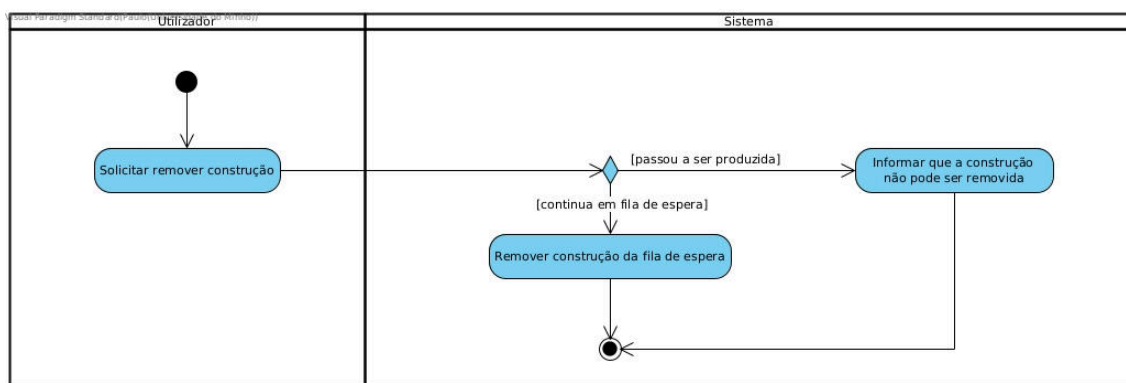


Figura 11: Remover Construção - Diagrama de Atividades.

Após a elaboração da tabela, foi desenvolvido o diagrama de atividades, representado em Figura 11. Este diagrama reflete de forma visual as mesmas ideias expressas na tabela, mas de uma maneira mais dinâmica e sequencial. Ao observar as “caixas” azuis, podemos perceber que elas representam ações específicas derivadas dos passos descritos no fluxo normal. Cada caixa está posicionada na *swimlane* correspondente ao ator que interage com a ação, o que facilita a compreensão das responsabilidades de cada participante. Por exemplo, a partir do passo 1, onde o ator indica a intenção de remover uma construção da fila de espera, podemos associar a caixa azul presente na *swimlane* do **utilizador** que diz “**Solicitar remoção de construção**”, refletindo essa interação.

Além disso, o diagrama contém um **losango**, que indica a presença de uma decisão no fluxo. A partir desse ponto, o diagrama divide-se em dois possíveis fluxos: um em que a construção permanece na fila de espera e outro em que a construção começa a ser produzida. Essa divisão visual no diagrama ilustra claramente a separação entre o **fluxo normal** e o **fluxo de exceção**, sendo que, no segundo caso, a construção não pode mais ser removida da fila, uma vez que entrou em produção. Essa estrutura de decisão no diagrama complementa a tabela, fornecendo uma visão gráfica e de fácil entendimento dos possíveis caminhos que o processo pode seguir.

Por último, as bolas pretas presentes no diagrama representam o início e o fim do fluxo, servindo como marcadores visuais para delimitar claramente o começo e a conclusão do processo. Essas bolas ajudam a entender a direção do fluxo de atividades, fornecendo uma referência clara de onde começa o processo e onde ele termina.

Tabela 2: Editar Perfil - Use Case.

USE CASE:	Editar perfil
DESCRIÇÃO:	Ator edita informações do seu perfil.
CENÁRIOS:	<b>Cenário 1:</b> Um utilizador visualiza as suas informações de perfil e altera o seu <i>username</i> .
PRÉ-CONDIÇÃO:	Ator tem sessão iniciada no sistema.
PÓS-CONDIÇÃO:	Sistema salva as informações de perfil alteradas na base de dados.
FLUXO NORMAL:	1. Ator solicita edição do seu perfil. 2. Sistema obtém as informações de perfil do Ator. 3. Sistema mostra ao Ator as informações do seu perfil. 4. Ator altera as informações que deseja. 5. Ator solicita que as novas informações sejam salvas. 6. Sistema salva as informações de perfil alteradas na base de dados.
FLUXO DE EXCEÇÃO (1):	[Sistema não conseguiu obter as informações de perfil do Ator] (passo 2) 2.1 Sistema mostra ao utilizador uma mensagem a indicar que ocorreu um erro ao obter as suas informações de perfil.
FLUXO DE EXCEÇÃO (2):	[Sistema não conseguiu salvar as novas informações de perfil do Ator] (passo 6) 6.1 Sistema mostra ao utilizador uma mensagem a indicar que ocorreu um erro ao salvar as suas informações de perfil.

Vamos agora analisar outro *use Case*, o editar perfil (**Tabela 2**), esta ação traduz diretamente o requisito funcional RF05, assegurando que o sistema ofereça aos utilizadores a capacidade de gerir as suas informações pessoais de forma segura e eficiente. Tal como no *use case* anterior, este também segue uma estrutura organizada em campos, incluindo descrição, pré-condições, pós-condições, fluxos normal e de exceção. No entanto, este *use case* apresenta duas exceções, detalhadas para lidar com situações específicas que podem ocorrer durante a edição de perfil.

Os requisitos do sistema para o *use case* “Editar Perfil” estão bem definidos e correlacionam-se diretamente com os passos descritos no fluxo normal e nos fluxos de exceção. Abaixo, analisamos cada requisito em detalhe, relacionando-o com os passos do *use case*:

- Requisito:** “O sistema deverá ter acesso às informações do utilizador.”
  - Este requisito é implícito no **passo 2** do fluxo normal, onde o sistema acede às informações do utilizador antes de apresentar os dados disponíveis para edição. Esse acesso assegura que o utilizador visualize informações atualizadas e consistentes.
- Requisito:** “O sistema deverá permitir ao utilizador alterar as suas informações, tais como *username*, e-mail e palavra-passe.”
  - Este requisito está diretamente refletido no **passo 4** do fluxo normal, em que o utilizador edita os campos desejados. O sistema oferece flexibilidade na alteração de múltiplos dados ao mesmo tempo, respeitando as permissões de acesso definidas.
- Requisito:** “O sistema deverá validar a unicidade dos dados introduzidos, não permitindo que vários utilizadores possuam o mesmo *username* ou e-mail.”
  - Embora este requisito esteja implicitamente representado entre os **passos 6 e 6.1**, a equipa de desenvolvimento decidiu enfatizá-lo graficamente no diagrama de atividades. Este detalhe destaca a importância da utilização de ambos os diagramas, complementando-se um ao outro.
- Requisito:** “O sistema deverá cancelar a operação e alertar o utilizador caso não se comprove a unicidade dos dados.”
  - Este cenário é tratado no **fluxo de exceção 6.1**, onde o sistema interrompe a operação ao identificar *username* ou e-mail duplicados, informando o utilizador sobre o erro.
- Requisito:** “O sistema deverá atualizar o perfil do utilizador.”
  - Representado no **passo final do fluxo normal**, este requisito garante que as alterações sejam aplicadas e salvas de forma consistente, refletindo imediatamente no sistema.

O desenvolvimento simultâneo da tabela de *use case* e do diagrama de atividades, Figura 12, foi essencial para identificar lacunas e oportunidades de melhoria no design do sistema. Como exemplo, a validação da unicidade dos dados foi evidenciada com mais clareza no diagrama, permitindo à equipa ajustar o fluxo de trabalho para alinhar com os objetivos do projeto. Essa abordagem iterativa, defendida por Sommerville, fomenta a colaboração entre programadores, analistas e *stakeholders*, resultando num sistema mais robusto e alinhado às expectativas.

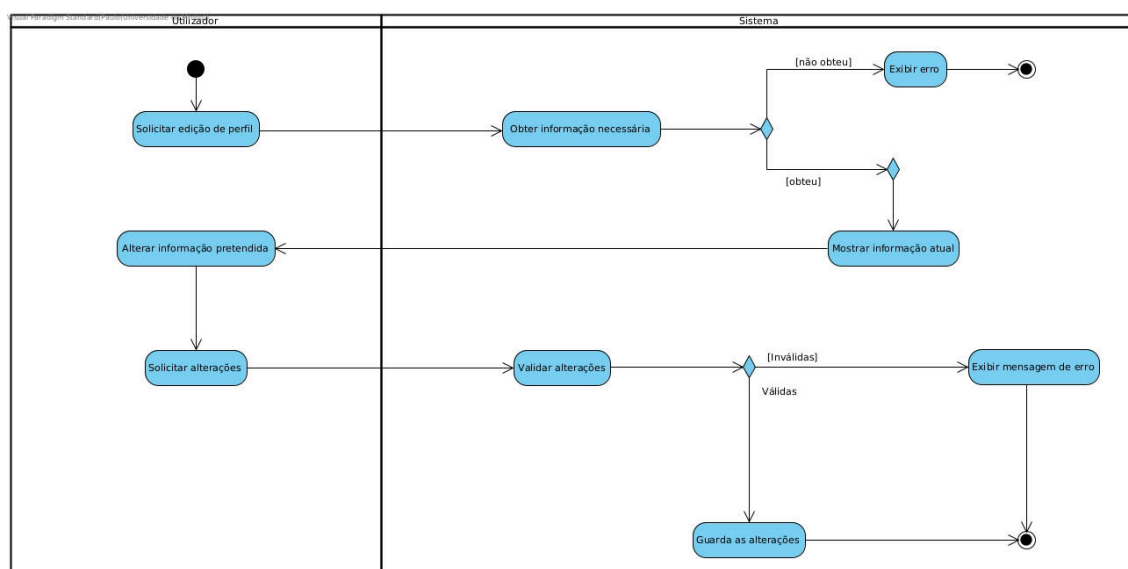


Figura 12: Editar Perfil - Diagrama de Atividades.

Durante o processo de modelação deste *use case*, a equipa de desenvolvimento identificou novos comportamentos relevantes para o sistema. Por exemplo, ao validar a unicidade dos dados, tornou-se evidente a necessidade de um mecanismo eficiente de pesquisa e indexação na base de dados para acelerar as verificações. Essa descoberta foi posteriormente estendida a outros *use cases*, otimizando o desempenho do sistema como um todo.

Além disso, o envolvimento dos *stakeholders*, como a Mojang, permitiu validar e aprovar as alterações propostas, assegurando que o sistema final atenda aos requisitos do cliente e ofereça uma experiência de utilizador aprimorada.

No decorrer do desenvolvimento dos diagramas de *use cases* e de atividades, a equipa identificou que certos requisitos funcionais, como o RF15, não poderiam ser traduzidos diretamente em *use cases*. Estes requisitos representam comportamentos que são executados exclusivamente pelo sistema, frequentemente baseados em algoritmos ou operações internas, sem interação direta com atores externos.

Para abordar esta situação, foi elaborado um diagrama de atividades que modela o fluxo do comportamento esperado para o RF15, relacionado ao funcionamento da linha de montagem. Este diagrama está representado em (Figura 13).

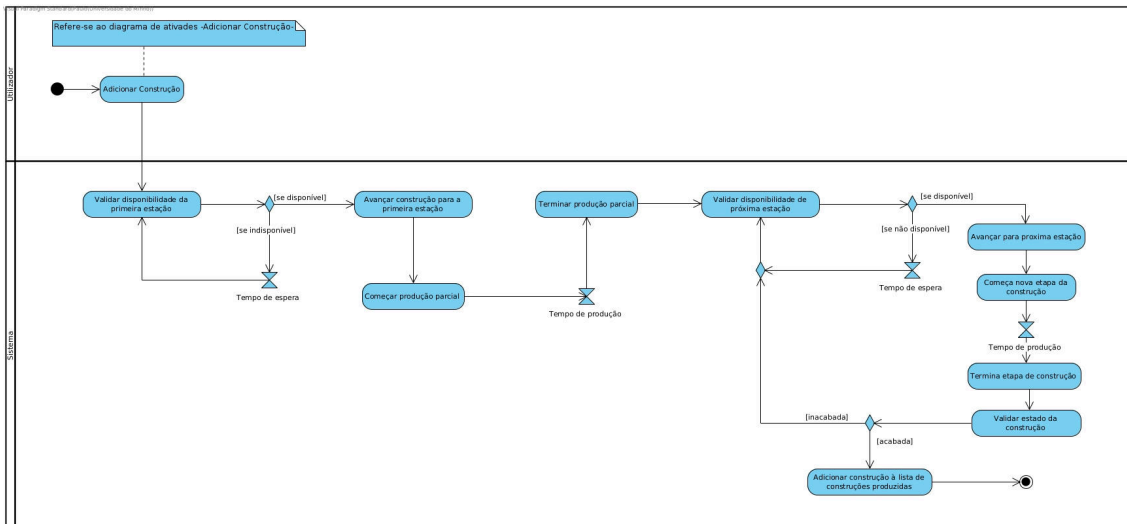


Figura 13: Funcionamento da Linha de Montagem - Diagrama de Atividades.

O requisito funcional 15 descreve como o sistema deve gerir o funcionamento de uma linha de montagem para a produção de construções, assegurando que o processo decorra de forma sequencial e eficiente. Este requisito é representado de forma clara no diagrama fornecido, onde as etapas do fluxo estão bem detalhadas.

O processo começa com o utilizador a iniciar a ação de **adicionar uma construção** à linha de montagem. Em seguida, o sistema procede à validação da disponibilidade da primeira estação. Caso a estação esteja **disponível**, a construção é imediatamente movida da fila de espera para essa estação. Por outro lado, se a estação estiver **ocupada**, a construção permanece na fila de espera até que se liberte.

Quando uma construção ocupa uma estação, o sistema contabiliza o tempo de produção parcial necessário. Após este período, o sistema verifica a disponibilidade da próxima estação. Se a próxima estação estiver livre, a construção avança para ela. Caso contrário, a construção permanece na estação atual até que a seguinte fique disponível.

Este ciclo de verificação de disponibilidade, avanço de construções e, quando necessário, espera, é repetido para todas as estações da linha de montagem, até que a construção chegue à última estação. Quando a produção parcial é concluída na última estação, o sistema move a construção finalizada para a lista de construções produzidas, encerrando o processo.

A modelação deste fluxo no diagrama oferece vários benefícios. Primeiramente, proporciona clareza no funcionamento do sistema, evidenciando as condições e os passos seguidos para a gestão da linha de montagem. Além disso, permite validar que todos os aspetos do requisito funcional 15 foram considerados, incluindo o tratamento de situações em que as estações estão ocupadas. Por fim, esta modelação serve como uma base sólida para a implementação técnica, fornecendo aos desenvolvedores uma referência detalhada para garantir que o sistema se comporta conforme esperado.

A equipa de desenvolvimento seguiu uma abordagem metódica para a modelação dos casos de uso, criando tanto tabelas como diagramas de atividades para cada um deles. Esse processo foi fundamentado na análise dos requisitos funcionais, que foram desdobrados em fluxos de ações, decisões e exceções. Com base nisso, foram desenvolvidos diagramas que ilustram o funcionamento do sistema em diferentes cenários.

Cada caso de uso foi representado de forma sistemática, começando com a identificação dos seus requisitos de origem, seguida da construção da tabela do caso de uso que descreve os atores, pré-condições, fluxo normal, fluxos alternativos e pós-condições. A partir desta tabela, foi desenvolvido um diagrama de atividades correspondente, demonstrando graficamente os passos do fluxo normal e os desvios ocasionados por condições alternativas ou exceções.

Nas próximas páginas, serão apresentados exemplos destes diagramas de atividades, elaborados de forma menos detalhada, mas suficientemente claros para comprovar a linha de raciocínio adotada pela equipe de desenvolvimento. Estes exemplos demonstram como, a partir de um requisito funcional, foi possível estruturar o caso de uso e modelar o fluxo de ações esperado no sistema. Assim, fica evidenciado o rigor do processo, bem como a coerência entre os requisitos levantados e os modelos criados, o que garante uma base sólida para a implementação e validação do sistema.

Tabela 3: Adicionar Construção - Use Case.

<b>USE CASE:</b>	Adicionar Construção
<b>DESCRIÇÃO:</b>	Ator adiciona uma construção.
<b>CENÁRIOS:</b>	<b>Cenário 1:</b> Utilizador adiciona uma construção à fila de espera.
<b>PRÉ-CONDIÇÃO:</b>	Ator tem sessão iniciada no sistema e visualiza o catálogo de construções.
<b>PÓS-CONDIÇÃO:</b>	Construção é adicionada à fila de espera.
<b>FLUXO NORMAL:</b>	1. Ator solicita adição de uma construção.
	2. Sistema verifica se existe stock suficiente para a produção.
	3. Sistema adiciona a construção à fila de espera.
	4. Sistema informa Ator da adição da construção.
<b>FLUXO DE EXCEÇÃO(1):</b>	[Não existe stock suficiente para a produção] (passo 2)
	2.1 Sistema informa o Ator que não existe stock suficiente para a produção da construção.
	2.2 Sistema mostra ao Ator a lista dos blocos em falta.

Com base no requisito funcional **10**, que especifica a funcionalidade de adicionar uma construção à fila de espera, é possível descrever o processo modelado pela equipa de desenvolvimento na **Tabela 3** e **Figura 14**. Este processo garante que todas as condições e comportamentos necessários são considerados, desde a validação do stock até à notificação do utilizador.

O fluxo começa quando o utilizador solicita a adição de uma construção à fila de espera. O sistema verifica então se o utilizador possui blocos suficientes em stock para a produção. Caso o stock seja insuficiente, o sistema informa o utilizador sobre os materiais em falta e não adiciona a construção à fila. Esta validação é essencial para evitar a interrupção do processo produtivo.

Se o stock for suficiente, a construção é adicionada à fila de espera da linha de montagem, e o sistema notifica o utilizador de que a operação foi bem-sucedida. Este processo garante não só o cumprimento dos requisitos do utilizador, mas também uma experiência intuitiva e funcional.

O diagrama de atividades complementa a tabela do caso de uso ao apresentar visualmente as interações entre o utilizador e o sistema, bem como as decisões tomadas com base nas condições de stock. Assim, o diagrama valida o raciocínio utilizado para mapear o requisito funcional 10 em um processo concreto e implementável.

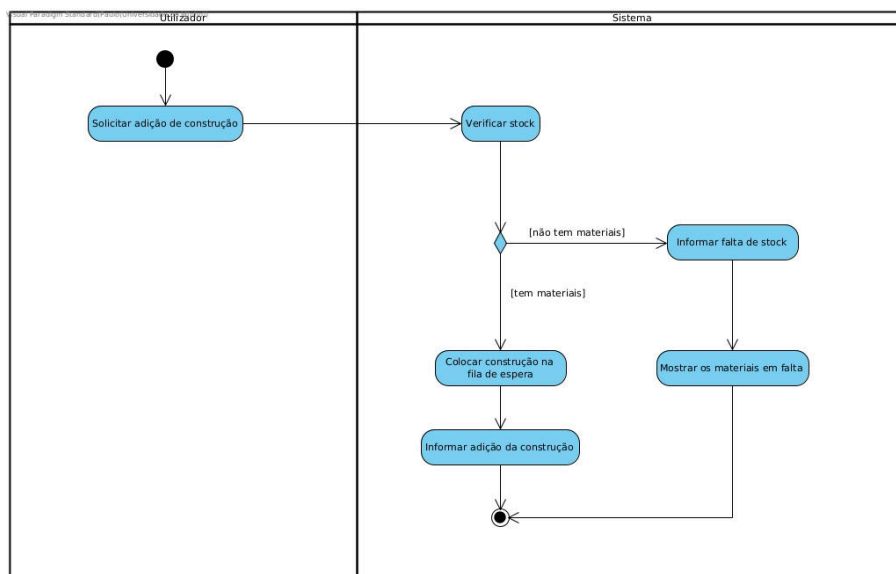


Figura 14: Adicionar Construção - Diagrama de Atividades.

O requisito funcional **01** estabelece os parâmetros necessários para que um utilizador possa criar uma conta na aplicação. O processo modelado na **Tabela 4** e na **Figura 15**, reflete a interação entre o utilizador e o sistema de modo a cumprir este requisito.

O utilizador inicia o processo solicitando a criação de uma conta. O sistema responde ao pedido solicitando as credenciais obrigatórias, que incluem e-mail, *username* e palavra-passe. Após o utilizador introduzir essas informações, o sistema verifica a validade das credenciais. Esta verificação inclui a análise de unicidade do *username* e do e-mail no sistema.

Caso as credenciais já estejam associadas a outra conta, o sistema informa o utilizador do conflito e finaliza o processo sem criar a conta. Por outro lado, se as credenciais forem válidas, o sistema procede à criação da conta e encerra o processo notificando o utilizador do sucesso da operação.

O diagrama de atividades reforça o mapeamento descrito, ao detalhar o fluxo condicional de verificação de credenciais e ao destacar as interações chave que garantem a unicidade dos dados. Este modelo assegura que todos os requisitos do RF01 sejam implementados de forma eficiente e transparente para o utilizador.

Tabela 4: Criar Conta de Utilizador - Use Case.

<b>USE CASE:</b>	Criar conta do utilizador
<b>DESCRIÇÃO:</b>	Ator cria conta no sistema.
<b>CENÁRIOS:</b>	<b>Cenário 1:</b> O Jorge cria uma conta.
<b>PRÉ-CONDIÇÃO:</b>	Ator não tem sessão iniciada
<b>PÓS-CONDIÇÃO:</b>	Sistema cria uma conta para o Ator.
<b>FLUXO NORMAL:</b>	1. Ator solicita a criação de uma conta. 2. Sistema solicita as credenciais (e-mail, <i>username</i> e palavra-passe) para a criação da conta. 3. Ator introduz as credenciais. 4. Sistema valida as credenciais. 5. Sistema cria uma conta para o Ator.
<b>FLUXO DE EXCEÇÃO (1):</b>	[Conta já existe no sistema] (passo 4) 4.1 Sistema informa o Ator que já existe uma conta com o mesmo e-mail.



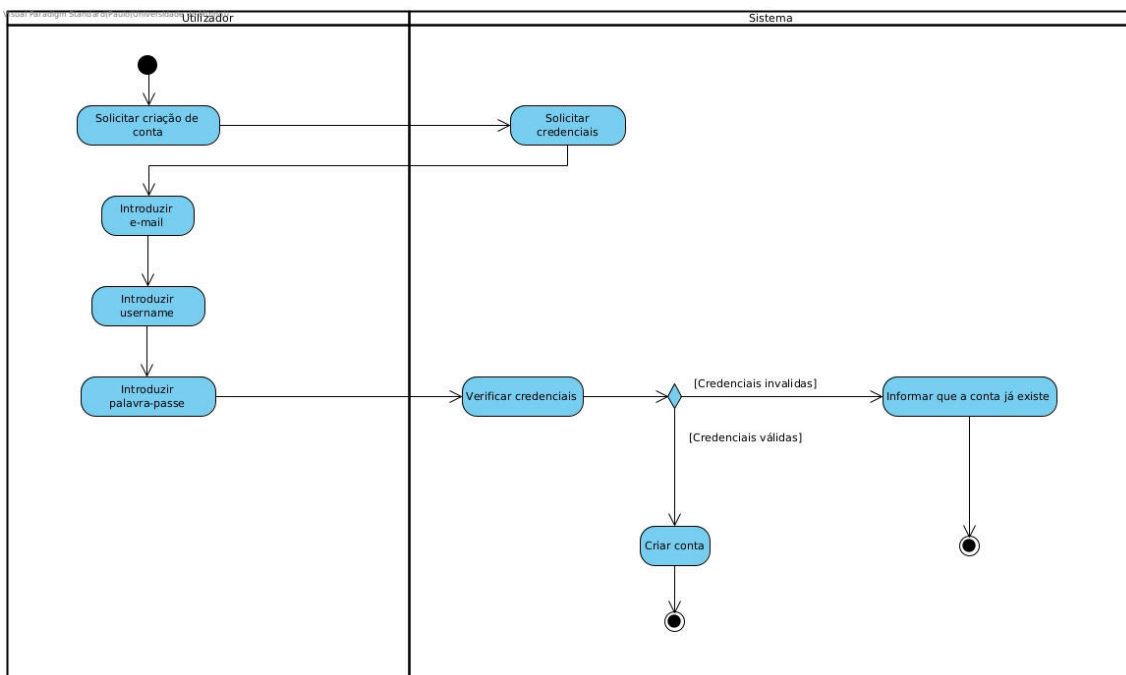


Figura 15: Criar Conta de Utilizador - Diagrama de Atividades.

O requisito funcional **17** descreve o processo necessário para que um utilizador possa encomendar materiais para o seu stock. Este requisito assegura que o sistema recolhe e valida todas as informações necessárias para a encomenda, proporcionando uma experiência clara e informativa para o utilizador.

O processo, demonstrado na **Figura 16** e na **Tabela 5** começa quando o utilizador solicita a criação de uma nova encomenda de stock. Em resposta, o sistema apresenta uma interface que permite ao utilizador selecionar os materiais desejados e especificar as quantidades para cada item. Assim que o utilizador confirma as escolhas, o sistema processa a informação fornecida, criando a encomenda e calculando o tempo estimado para a sua entrega.

Caso o processo seja concluído com sucesso, o sistema regista a encomenda e informa o utilizador tanto sobre o registo da mesma quanto sobre o tempo estimado de espera para que os materiais estejam disponíveis. Este *feedback* garante que o utilizador esteja ciente do progresso da encomenda e dos prazos associados.

O diagrama de atividades complementa a descrição ao ilustrar claramente cada etapa do processo, desde a solicitação inicial até à confirmação final. Este modelo não só mapeia os passos necessários para a criação da encomenda, como também assegura que os requisitos descritos no RF17 sejam implementados de forma eficiente e orientada para o utilizador.

Tabela 5: Encomendar Stock - Use Case.

<b>USE CASE:</b>	Encomendar stock
<b>DESCRIÇÃO:</b>	Ator cria uma encomenda de stock que será entregue futuramente.
<b>CENÁRIOS:</b>	<b>Cenário 1:</b> A Maria encomenda 6 blocos de Madeira para construir as Portas que necessita. <b>Cenário 2:</b> O João encomenda 9 blocos de Pedra, 2 blocos de Vidro, 1 Porta e 12 blocos de Madeira para construir a sua casa. <b>Cenário 3:</b> O Henrique encomenda um bloco de Pedra porque lhe apetece.
<b>PRÉ-CONDIÇÃO:</b>	Ator tem sessão iniciada
<b>PÓS-CONDIÇÃO:</b>	Encomenda fica registada.
<b>FLUXO NORMAL:</b>	1. Ator solicita nova encomenda. 2. Sistema solicita os materiais que o Ator deseja encomendar. 3. Ator escolhe os materiais de uma lista e as suas quantidades. 4. Sistema cria nova encomenda com os materiais e as quantidades fornecidas e calcula um tempo estimado de espera. 5. Sistema informa o Ator do registo da encomenda e do tempo estimado de espera.

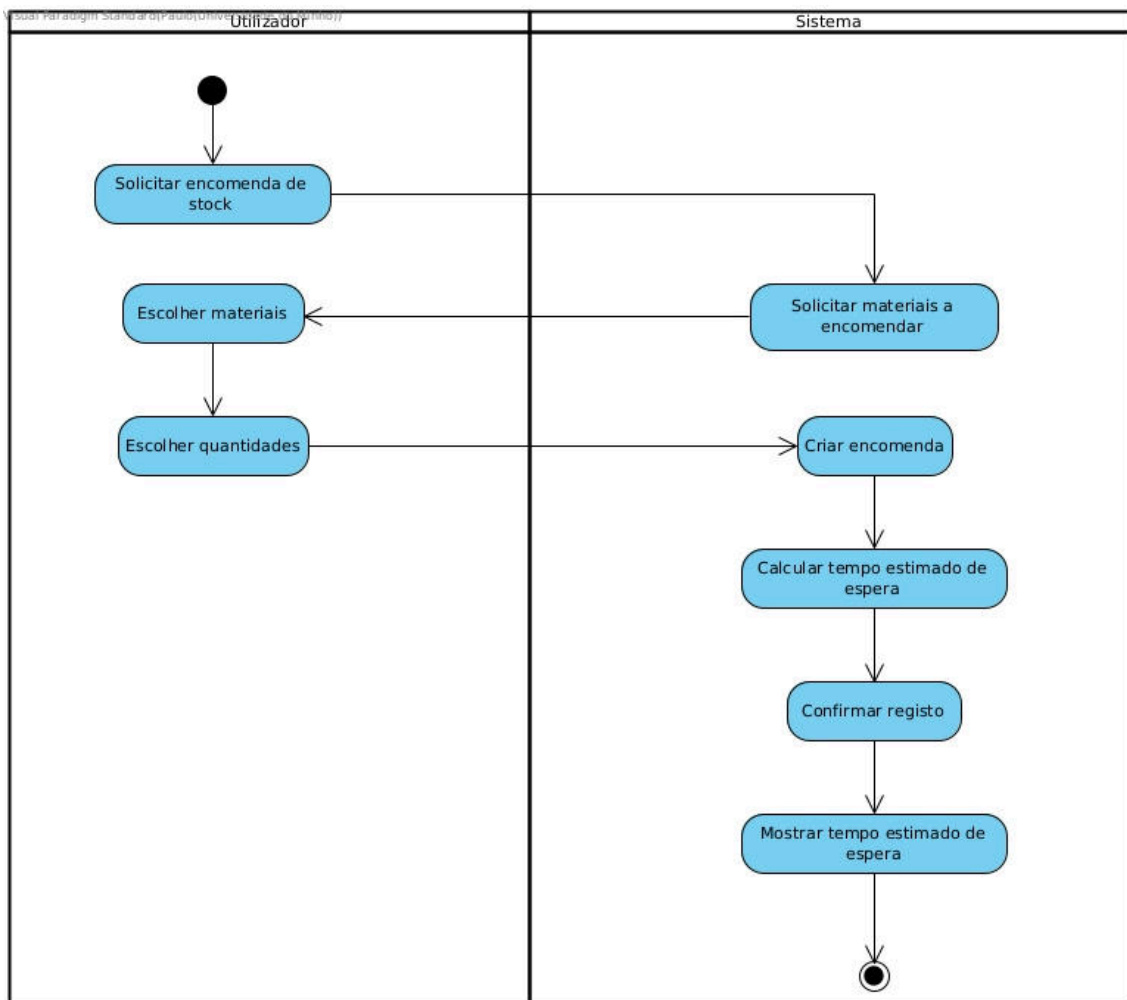


Figura 16: Encomendar Stock - Diagrama de Atividades.

O requisito funcional **06** descreve o processo pelo qual um utilizador pode visualizar o catálogo de construções disponíveis na aplicação. Este catálogo apresenta todas as construções que podem ser produzidas pela linha de montagem, destacando de forma clara as construções para as quais o utilizador não possui materiais suficientes, proporcionando uma experiência informativa e intuitiva.

O processo inicia-se com o utilizador solicitando a consulta do catálogo de construções. Em resposta, o sistema acede à lista de construções suportadas pela linha de montagem, garantindo que todas as opções possíveis são incluídas. Paralelamente, o sistema verifica o stock disponível do utilizador, analisando quais construções podem ser produzidas com os materiais atualmente disponíveis.

Após esta análise, o sistema marca as construções para as quais o utilizador não possui materiais suficientes como indisponíveis. Esta distinção permite que o utilizador identifique facilmente quais as opções viáveis e quais exigem uma reposição de stock. Por fim, o catálogo é apresentado ao utilizador, contendo tanto as construções disponíveis quanto as indisponíveis, devidamente destacadas.

O diagrama de atividades associado detalha o fluxo do processo, evidenciando a interação entre o utilizador e o sistema, bem como os cálculos internos necessários para determinar a disponibilidade das construções. Este modelo garante que os requisitos do RF06 são implementados de forma eficiente, oferecendo uma visão clara e prática do catálogo ao utilizador.

Tabela 6: Visualizar Catálogo - Use Case.

<b>USE CASE:</b>	Visualizar Catálogo
<b>DESCRIÇÃO:</b>	Ator consulta o catálogo de construções.
<b>CENÁRIOS:</b>	<b>Cenário 1:</b> O Barnabás consulta o catálogo da sua linha de montagem.
<b>PRÉ-CONDIÇÃO:</b>	Ator tem sessão iniciada no sistema.
<b>PÓS-CONDIÇÃO:</b>	Sistema apresenta o catálogo ao Ator.
<b>FLUXO NORMAL:</b>	1. Ator solicita consulta do catálogo de construções.
	2. Sistema obtém a lista de construções suportadas pela linha de montagem.
	3. Sistema obtém o stock disponível para o Ator.
	4. Sistema marca todas as construções não cobertas pelo stock como indisponíveis.
	5. Sistema mostra catálogo ao Ator.

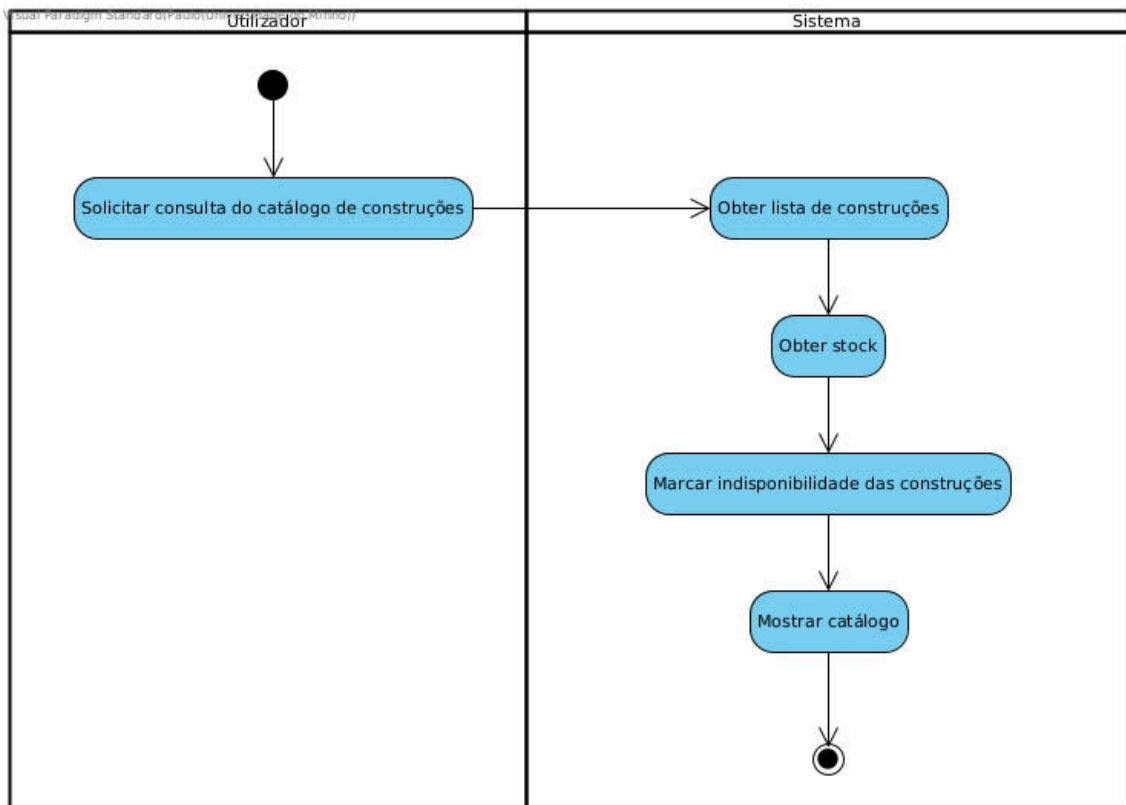


Figura 17: Visualizar Catálogo - Diagrama de Atividades.

O requisito funcional **08** estabelece o processo pelo qual um utilizador pode visualizar os detalhes de uma construção já produzida, incluindo as etapas realizadas durante a produção e os materiais utilizados. Este requisito garante que o utilizador tenha acesso a informações completas e organizadas sobre as construções finalizadas.

O fluxo do caso de uso começa com o utilizador solicitando a visualização dos detalhes de uma construção específica, escolhida a partir da lista de construções produzidas, o que se pode observar facilmente tanto na **Tabela 7** como na **Figura 18**. O sistema, em resposta, recupera da base de dados as informações associadas à construção selecionada, incluindo os materiais utilizados e os passos seguidos durante o processo de produção.

Após recuperar as informações, o sistema apresenta ao utilizador os detalhes completos da construção. Essa apresentação inclui tanto o estado final da construção quanto um guião que documenta as etapas realizadas para produzi-la. Esta abordagem oferece ao utilizador uma visão clara do processo de produção e dos recursos empregados.

O diagrama de atividades reforça este fluxo, evidenciando a interação direta entre o utilizador e o sistema, bem como o processamento interno necessário para garantir a exibição dos dados solicitados. Assim, o modelo implementa os requisitos do RF08 de forma eficiente, assegurando que o utilizador tenha uma experiência informativa e intuitiva ao consultar construções produzidas.

Tabela 7: Visualizar Construção Produzida - Use Case.

<b>USE CASE:</b>	Visualizar construção produzida
<b>DESCRIÇÃO:</b>	Ator visualiza os detalhes de uma construção produzida por completo.
<b>CENÁRIOS:</b>	<b>Cenário 1:</b> O João consulta os passos efetuados para a construção de uma casa.
<b>PRÉ-CONDIÇÃO:</b>	Ator tem a sessão iniciada e visualiza a lista de construções produzidas.
<b>PÓS-CONDIÇÃO:</b>	Sistema apresenta detalhes da construção produzida.
<b>FLUXO NORMAL:</b>	<ol style="list-style-type: none"> <li>1. Ator solicita os detalhes de uma construção na lista de construções produzidas.</li> <li>2. Sistema obtém detalhes da construção selecionada a partir da base de dados.</li> <li>3. Sistema apresenta a lista de materiais utilizados e os passos da construção acabada.</li> </ol>

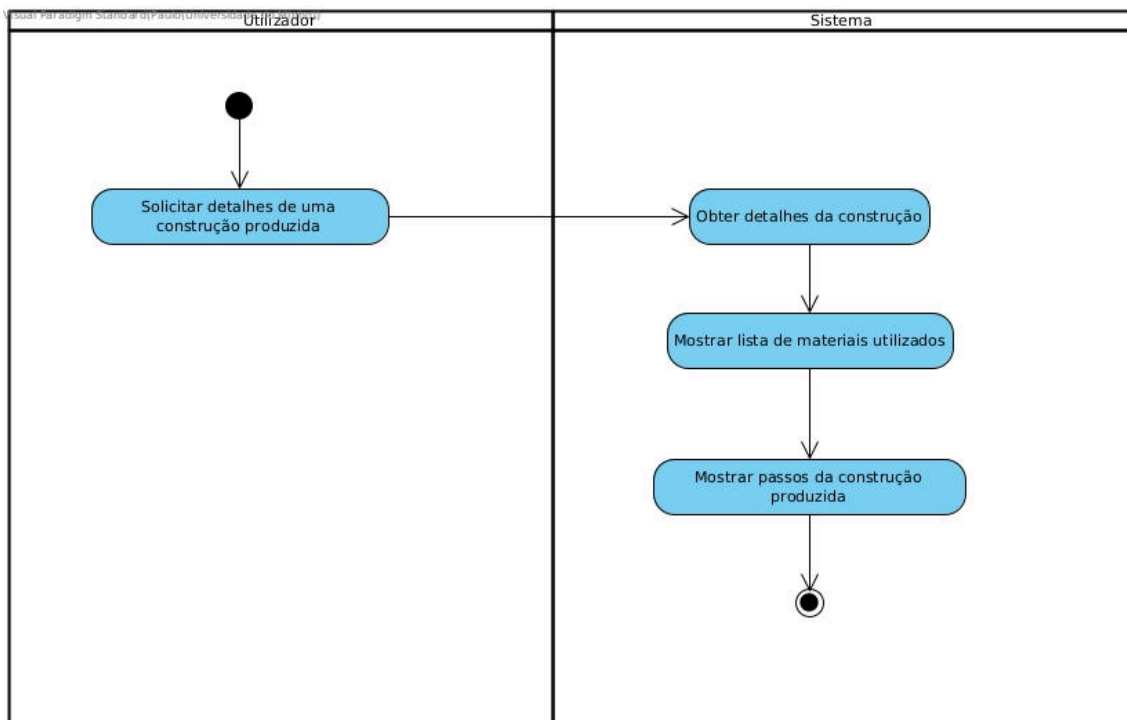


Figura 18: Visualizar Construção Produzida - Diagrama de Atividades.

O requisito funcional **14** define o processo necessário para que um utilizador visualize todas as construções já produzidas na linha de montagem. Este requisito garante que o utilizador tenha acesso a um histórico claro e organizado das produções concluídas, possibilitando o acompanhamento do progresso e das atividades realizadas.

O caso de uso inicia-se com o utilizador solicitando a lista de construções produzidas. O sistema, em resposta, consulta a base de dados para recuperar as informações correspondentes às construções finalizadas. Essas informações incluem todas as construções que já passaram pela linha de montagem e foram concluídas com sucesso.

Após a obtenção dos dados, o sistema apresenta ao utilizador a lista das construções produzidas, de forma clara e organizada, possibilitando que o utilizador visualize rapidamente as produções disponíveis. Esta

funcionalidade é essencial para que o utilizador acompanhe os resultados do processo produtivo e analise as construções concluídas.

O diagrama de atividades, **Tabela 8**, reflete este fluxo de maneira detalhada, destacando a interação direta entre o utilizador e o sistema, bem como o processamento interno necessário para recuperar e exibir as informações. Assim, o modelo assegura que o RF14 seja implementado de forma eficiente, proporcionando ao utilizador uma experiência intuitiva e informativa ao consultar as construções produzidas.

Tabela 8: Visualizar Construções Produzidas - Use Case.

USE CASE:	Visualizar construções produzidas
DESCRIÇÃO:	Ator visualiza uma lista com todas as construções produzidas na linha de montagem.
CENÁRIOS:	<b>Cenário 1:</b> O João consulta as suas últimas 5 construções produzidas.
PRÉ-CONDIÇÃO:	Ator tem a sessão iniciada.
PÓS-CONDIÇÃO:	Ator recebe lista de construções produzidas.
FLUXO NORMAL:	1. Ator solicita a lista de construções produzidas na linha de montagem. 2. Sistema obtém lista de construções produzidas a partir da base de dados. 3. Sistema apresenta a lista de construções produzidas.

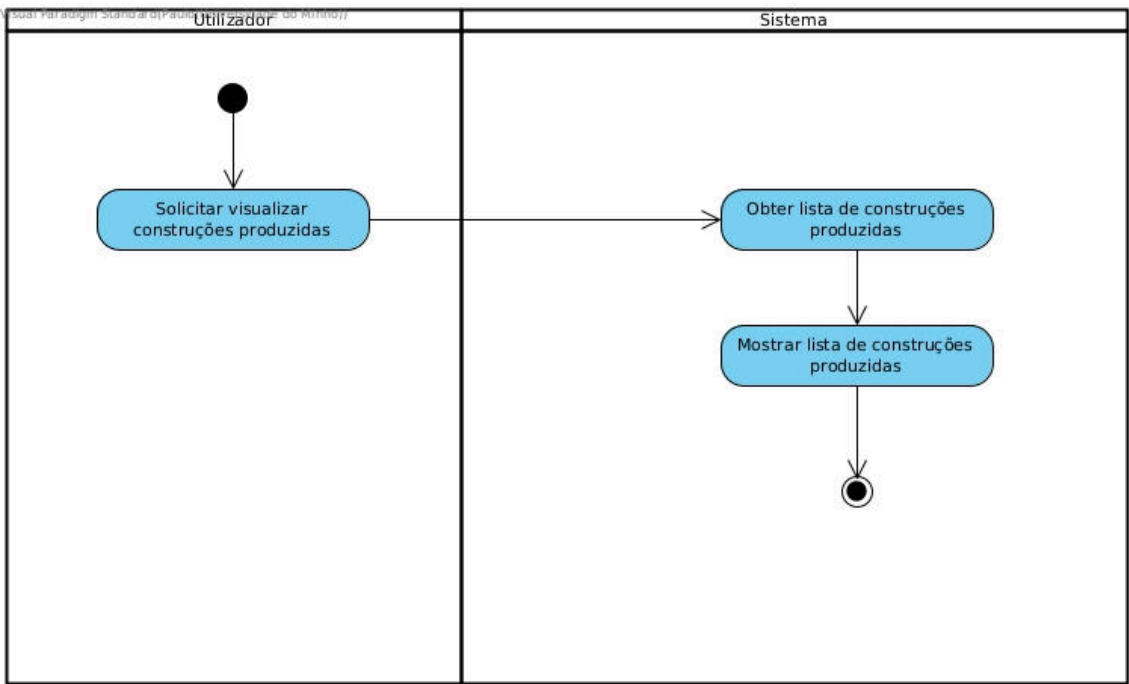


Figura 19: Visualizar Construções Produzidas - Diagrama de Atividades.

## 4. Conceção do Sistema de Dados

### 4.1. Apresentação geral da estrutura do sistema de dados

Através da definição do modelo de domínio, da interiorização por parte de todos os elementos da equipa de desenvolvimento do conhecimento do sistema a concretizar, assim como da clara definição dos requisitos funcionais e não funcionais, a equipa de trabalho elaborou o modelo lógico da base de dados que assegura a capacidade da aplicação suportar as funcionalidades requeridas. Desta forma, o modelo lógico do projeto *MineBuilds* apresenta-se a seguir:

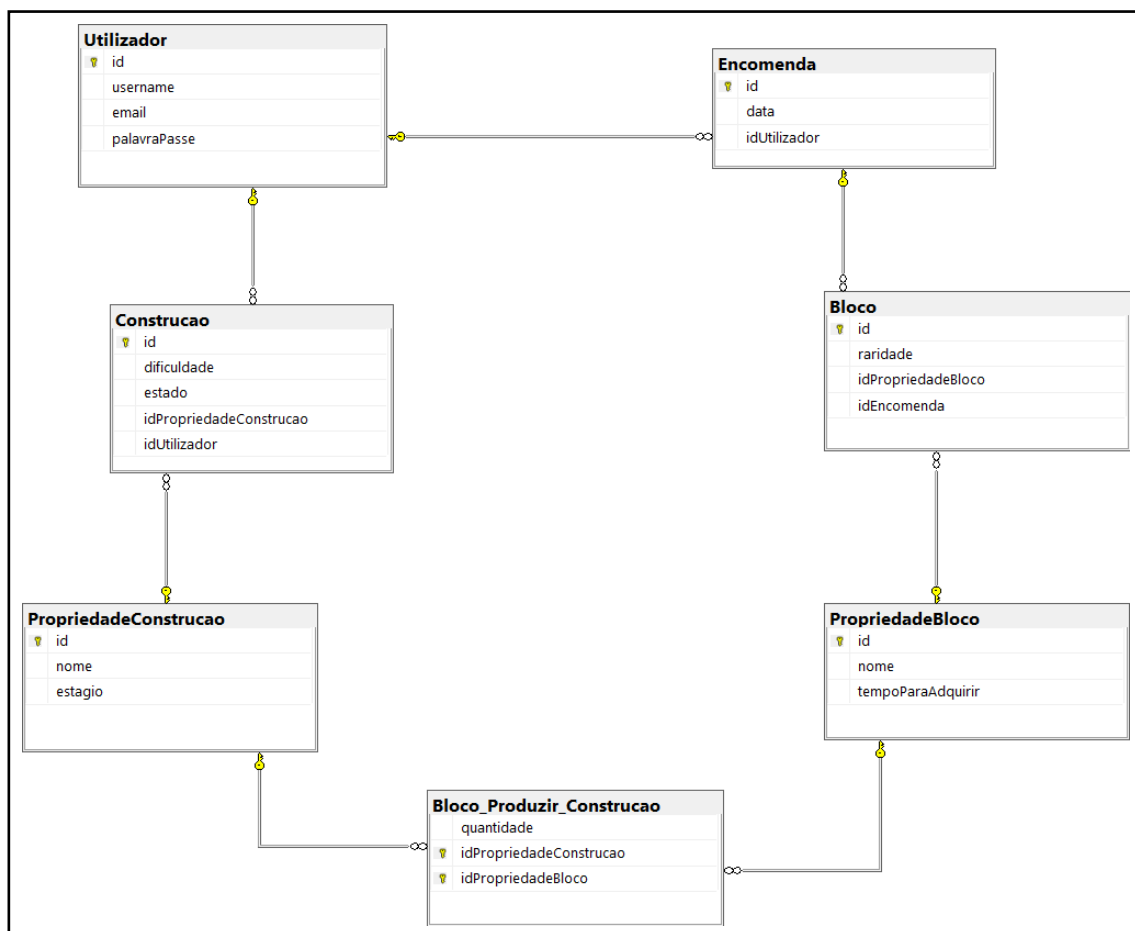


Figura 20: Esquema do Modelo Lógico.

## 4.2. Descrição detalhada dos vários elementos de dados e seus relacionamentos

Para o correto desenvolvimento do modelo lógico, a equipa de desenvolvimento decidiu iniciar o seu planeamento através da criação de um dicionário de dados que caracteriza as relações e as entidades que existem na base de dados. Esta decisão foi crucial para a realização deste modelo, dado que a inexistência de um modelo conceptual, mas sim de um modelo de domínio, poderia levar à criação de relacionamentos e entidades desnecessárias para a base de dados, tendo sido cometidos alguns erros ao longo do desenvolvimento do planeamento do modelo lógico, sendo estes relatados mais abaixo ainda nesta categoria. Assim sendo, a equipa de trabalho começou por identificar as entidades essenciais a serem armazenadas e manipuladas na base de dados, sendo essas: **Utilizador**, **Encomenda**, **Bloco** e **Construção**. Acrescentando a estas entidades, a base de dados possui mais três tabelas que não constavam explicitamente no modelo de domínio, mais concretamente: **Blocos para produzir uma Construção**, **Propriedades do Bloco** e **Propriedades da Construção**. A primeira tabela das três citadas implementa o conceito de **Blueprint** relacionado com a entidade **Construção** no modelo de domínio. Por sua vez, as restantes tabelas foram criadas para proporcionar uma maior eficiência na base de dados e o armazenamento de valores estáticos, tal como será descrito nas respetivas descrições das entidades. Em contrapartida, algumas entidades que existiam no modelo de domínio não justificavam a criação de tabelas no modelo lógico, sendo essas: **Credencial**, **Conta**, **Stock**, **Linha de Montagem**, **Fila de Espera** e **Pedido**. Através de uma análise cuidadosa e retalhada, a equipa de desenvolvimento concluiu que a entidade **Conta** não possui razões suficientes para a criação da sua respetiva tabela na base de dados, porém a entidade **Credencial**, nomeadamente o *username*, e-mail e palavra-passe, passam a ser atributos da entidade **Utilizador**, dado a estas apenas existirem no contexto de uma conta que, por sua vez, apenas existe no contexto de um utilizador, tal como demonstra o modelo de domínio. Já no que diz respeito à entidade **Stock**, esta servia para identificar o utilizador a qual o bloco pertence, sendo um motivo insuficiente para a criação da sua tabela na base de dados, tendo a equipa de desenvolvimento optado por associar um bloco diretamente a um utilizador através das encomendas. Finalmente, as entidades **Linha de Montagem**, **Fila de Espera** e **Pedido** existem no modelo de domínio para caracterizar todo o processo relativo à produção de uma construção, sendo também desnecessária a criação de mais tabelas na base de dados para estas entidades, uma vez que o **Pedido** permitia identificar o utilizador que pediu uma determinada construção, sendo este impasse resolvido com a colocação do identificador do utilizador na construção. Relativamente à **Fila de Espera**, esta proporciona ao utilizador a visualização e remoção de construções que ainda não estejam a ser produzidas, sendo este requisito satisfeito de igual forma através da criação de um atributo, na construção, que permita saber o estado em que esta se encontra. Já a entidade **Linha de Montagem** representa todo o contexto do sistema a desenvolver, não havendo motivos para a sua colocação na base de dados, já que as entidades com quem estava relacionada no modelo de domínio não existem na base de dados. Assim sendo, as entidades mencionadas possuem diferentes tipos de atributos, sendo estas caracterizadas de seguida.

### 4.2.1. Utilizador

A entidade utilizador representa todos os jogadores que usufruem da aplicação desenvolvida. Cada utilizador é identificado por um identificador que se auto-incrementa sempre que um novo jogador se regista na aplicação. O utilizador também possui um *username*, e-mail e palavra-passe que são as suas credenciais para aceder à aplicação, sendo o *username* e e-mail únicos, assim como o seu identificador.

Utilizador				
Atributo	Descrição	Tipo de dados	Máximo espaço ocupado (bytes)	Exemplo
id	Identifica unicamente o utilizador	INT	4	157
username	Identifica unicamente o utilizador	VARCHAR(40)	42	michaeljackson
e-mail	Endereço eletrónico do utilizador	VARCHAR(70)	72	michaeljacksonjr@sapo.pt
palavraPasse	Palavra-passe que permite o acesso à conta do utilizador	VARCHAR(45)	47	yes@im@alive123

Figura 21: Caracterização da tabela Utilizador.

### 4.2.2. Encomenda

A entidade encomenda é a única forma de um jogador obter blocos, possuindo esta um identificador único auto-incremental e a data em que o utilizador fez a encomenda, para além do identificador do utilizador



que a pediu. Esta entidade permite a visualização, por parte do utilizador, de todas as encomendas que este já fez na aplicação.

Encomenda				
Atributo	Descrição	Tipo de dados	Máximo espaço ocupado (bytes)	Exemplo
id	Identifica unicamente a encomenda	INT	4	1
idUtilizador	Identifica o utilizador que fez a encomenda	INT	4	157
data	Data em que o utilizador fez a encomenda	DATETIME	8	2024-03-01 17:59:12

Figura 22: Caracterização da tabela Encomenda.

#### 4.2.3. Bloco

A entidade bloco é uma das mais cruciais no sistema, dado que para uma construção ser produzida será necessário satisfazer os seus requisitos, nomeadamente, possuir blocos. Esta entidade possui um identificador único auto-incremental, o identificador do seu nome, uma raridade associada e o identificador da encomenda em que este foi obtido, que permite obter o utilizador que possui o bloco.

Bloco				
Atributo	Descrição	Tipo de dados	Máximo espaço ocupado (bytes)	Exemplo
id	Identifica unicamente o bloco	INT	4	3
idNomeBloco	Identifica o nome do bloco	INT	4	15
raridade	Raridade do bloco	VARCHAR(7)	9	RARO
idEncomenda	Identifica a encomenda onde o bloco foi requerido	INT	4	1

Figura 23: Caracterização da tabela Bloco.

#### 4.2.4. Construção

A entidade construção é a mais central do sistema, já que todo o processo relativo à sua aquisição e produção refletem a linha de montagem e as suas devidas estações. Esta, tal como as outras entidades, possui um identificador único auto-incremental, um identificador para o seu nome, uma dificuldade associada, o identificador do utilizador que a pediu e um estado de produção associado. A dificuldade permite uma forma de ordenação diferente para mostrar o catálogo de construções ao utilizador, sendo o estado de produção, por outro lado, uma maneira de saber se a construção já foi produzida, se está a ser produzida ou se ainda está em espera, sendo uma condição necessária para a concretização de alguns requisitos funcionais relacionados com a visualização e remoção de construções da linha de montagem.

Construção				
Atributo	Descrição	Tipo de dados	Máximo espaço ocupado (bytes)	Exemplo
id	Identifica unicamente a construção	INT	4	12
idNomeConstrução	Identifica o nome da construção	INT	4	11
dificuldade	Dificuldade de produzir a construção	VARCHAR(6)	8	FÁCIL
estado	Identifica o estado da construção	VARCHAR(9)	11	PRODUZIDA
idUtilizador	Identifica o utilizador que requisitou a construção	INT	4	157

Figura 24: Caracterização da tabela Construção.

#### 4.2.5. Blocos para produzir uma Construção

A tabela blocos para produzir uma construção permite caracterizar o conceito de *blueprint* que está associado a uma construção. Esta tabela contém as quantidades necessárias, de um dado bloco, para construir uma determinada construção, sendo fulcral para mostrar a um utilizador todos os blocos que cada construção precisa para ser produzida, bem como as suas respetivas quantidades. O identificador desta tabela é composto pelos identificadores do nome da construção e do nome do bloco, possuindo também uma quantidade associada.

Bloco_Produzir_Construção				
Atributo	Descrição	Tipo de dados	Máximo espaço ocupado (bytes)	Exemplo
idNomeBloco	Identifica o nome do bloco a ser utilizado	INT	4	15
idNomeConstrução	Identifica o nome da construção onde os blocos são utilizados	INT	4	11
quantidade	Quantidade de blocos a utilizar	INT	4	18

Figura 25: Caracterização da tabela Bloco\_Produzir\_Construção.

#### 4.2.6. Propriedades do Bloco

A tabela propriedades do bloco trata-se de uma tabela de consulta e de um método para otimizar o uso da memória da base de dados. Esta guarda valores estáticos relativos a um bloco, como o seu nome e o tempo que demora a ser adquirido, sendo este valor utilizado, por parte da aplicação, para mostrar ao utilizador o

tempo que demora para obter uma encomenda. Para além disso, a primeira versão criada para o modelo lógico colocava o atributo nome na tabela Bloco, algo que poderia levar à criação de nomes ambíguos e entradas repetidas desnecessariamente. Assim sendo, em vez de guardarmos o nome na tabela Bloco, podendo este ocupar cerca de 42 bytes por cada entrada, decidimos guardar o identificador para a tabela das suas propriedades, ocupando apenas 4 bytes, para além desta nova tabela possuir relativamente um menor número de entradas na base de dados devido à quebra na ambiguidade dos nomes.

PropriedadeBloco				
Atributo	Descrição	Tipo de dados	Máximo espaço ocupado (bytes)	Exemplo
id	Identifica unicamente a propriedade do bloco	INT	4	15
nome	Nome do bloco	VARCHAR(40)	42	Porta de Madeira
tempoParaAdquirir	Tempo que demora a obter o bloco	INT	4	40

Figura 26: Caracterização da tabela PropriedadeBloco.

#### 4.2.7. Propriedades da Construção

A tabela propriedades da construção foi criada pelos mesmos motivos que a tabela propriedades do bloco, porém, neste caso, a construção possui como valor estático a quantidade de estágios para a sua produção, permitindo mostrar ao utilizador em que estágio da linha de montagem se encontra a construção.

PropriedadeConstrução				
Atributo	Descrição	Tipo de dados	Máximo espaço ocupado (bytes)	Exemplo
id	Identifica unicamente a propriedade da construção	INT	4	11
nome	Nome da construção	VARCHAR(50)	52	Casa de Pedra sem janelas
estagio	Identifica a quantidade de estágios para produzir a construção	INT	4	3

Figura 27: Caracterização da tabela PropriedadeConstrucao.

#### 4.2.8. Relacionamento Utilizador (1,1) para Encomenda (0,N)

O utilizador pode fazer várias encomendas ou nenhuma. Porém, uma encomenda deve estar obrigatoriamente associada a um utilizador. Este relacionamento implica o identificador do utilizador estar na tabela Encomenda como chave estrangeira.

#### 4.2.9. Relacionamento Utilizador (1,1) para Construção (0,N)

O utilizador pode ter, ou não, várias construções, contudo cada construção deve estar obrigatoriamente associada ao utilizador que a requisitou. Desta forma, o identificador do utilizador encontra-se na tabela Construção como chave estrangeira.

#### 4.2.10. Relacionamento Propriedade Construção (1,1) para Construção (1,N)

Uma propriedade de construção pode estar associada a uma ou mais construções, e cada construção deve estar obrigatoriamente associada a uma propriedade de construção. Assim sendo, o identificador da tabela de propriedade da construção é uma chave estrangeira na tabela Construção.

#### 4.2.11. Relacionamento Encomenda (1,1) para Bloco (1,N)

Cada encomenda deve ter obrigatoriamente um ou mais blocos associados. Já um bloco não pode existir sem estar relacionado a uma encomenda. Consequentemente, o identificador da tabela Encomenda é uma chave estrangeira na tabela Bloco.

#### 4.2.12. Relacionamento Propriedade Bloco (1,1) para Bloco (1,N)

Uma propriedade de um bloco precisa de estar associada a um ou mais blocos. Porém, cada bloco necessita de estar associado a uma propriedade de bloco, levando a que o identificador da tabela de propriedades do bloco seja uma chave estrangeira na tabela Bloco.

#### **4.2.13. Relacionamento Propriedade Bloco (1,N) para Propriedade Construção (1,N)**

Um conjunto de propriedades de um bloco está correlacionado com as propriedades de uma construção. O mesmo acontece ao contrário, levando a que seja criada a tabela “Bloco Produzir Construção” cuja chave primária é composta pelas chaves primárias das duas tabelas de propriedades.

## 5. Esboço das Interfaces do Sistema

### 5.1. Estrutura geral das interfaces do sistema

Efetuada todo o planeamento necessário ao sistema, desde o levantamento de requisitos à sua arquitetura, foi possível prosseguir para o planeamento dos interfaces do sistema, de modo a prever a vertente gráfica da aplicação. Para isso, foi definido um conjunto essencial de páginas que permitissem responder às necessidades do sistema e desenvolvido para cada uma delas um *mockup*, um esboço de elementos possivelmente relevantes da UI. Os *mockups* são uma parte crucial do processo de *design* da aplicação pois permitem organizar de forma visual elementos relevantes para o funcionamento sistema, possibilitando uma discussão detalhada com o cliente de modo a verificar que o produto vai de encontro ao que tinha imaginado. Assim, foram criados *mockups* para os cenários de: **Página Login, Página Registo, Página Inicial, Perfil, Fila de Espera, Catálogo, Adicionar Construção, Linha de Montagem, Construção em Produção, Construções Acabadas, Construção Acabada, Stock, Encomendas, Encomenda, Encomendar.**

### 5.2. Caracterização das interfaces

#### 5.2.1. Página Iniciar Sessão

A Figura 28 representa a página “Iniciar Sessão”. Nesta página, existem três campos de texto modificáveis, com os nomes “Email”, “Username” e “Palavra Passe”. No canto inferior direito existem dois botões, um para validar os dados e iniciar sessão, com o nome “Confirmar”, e outro que redireciona o utilizador para a página “Criar Conta”, com o nome “Criar Conta”. Para iniciar sessão, o utilizador deverá preencher os campos e clicar no botão “Confirmar”. Se os dados preenchidos forem válidos, a sessão será iniciada e o utilizador será redirecionado para a página “Inicial”. Se os dados preenchidos forem inválidos, um erro a indicar o(s) campo(s) inválidos será apresentado ao utilizador. Este *mockup* satisfaz o use case “Criar Conta do Utilizador”.

O mockup da página "Iniciar Sessão" apresenta uma interface limpa e funcional. No topo, há uma barra de navegação com o nome "MineBuilds" no centro, flanqueado por setas de navegação. Abaixo, o título "Iniciar sessão" está no canto superior esquerdo. No canto superior direito, há um ícone de perfil de usuário. O formulário principal contém três campos de texto rotulados "Email", "Username" e "Palavra Passe". No canto inferior direito, há dois botões: "Criar conta" e "Confirmar".

Figura 28: Mockup da Página "Iniciar Sessão"

### 5.2.2. Página Criar Conta

A Figura 29 representa a página “Criar Conta”. Nesta página, existem três campos de texto modificáveis, com os nomes “Email”, “Username” e “Palavra Passe”. No canto inferior esquerdo existem dois botões, um para validar os dados e criar a conta de utilizador, com o nome “Confirmar” e outro que redireciona o utilizador para a página “Iniciar Sessão”, com o mesmo nome. Para confirmar o registo da sua conta o utilizador deverá preencher os campos e clicar no botão “Confirmar”. Se os dados preenchidos forem válidos, a sua conta de utilizador será criada e terá uma sessão automaticamente iniciada, sendo o utilizador redirecionado para a página “Inicial”. Se os dados preenchidos forem inválidos, um erro a indicar o(s) campo(s) inválidos será apresentado ao utilizador. Este *mockup* satisfaz o use case “Criar Conta do Utilizador”.

O mockup da página "Criar Conta" apresenta uma interface limpa. No topo, há uma barra de navegação com o nome "MineBuilds" no centro. Abaixo, o título "Registar Conta" está centralizado. À esquerda, há três campos de texto rotulados "Email", "Username" e "Palavra Passe". À direita, há um ícone de perfil de utilizador. No canto inferior esquerdo, há dois botões: "Confirmar" e "Iniciar sessão".

Figura 29: Mockup da Página "Criar Conta"

### 5.2.3. Página Inicial

A Figura 30 representa a página “Inicial” da aplicação, disponível apenas com uma sessão válida ativa. Esta página serve como um *hub* para a aplicação, sendo possível aceder a todas as partes da aplicação a partir desta. O conteúdo da página é maioritariamente constituído por botões que redirecionam o utilizador para múltiplas páginas que possibilitam o acesso a diferentes ações dentro da aplicação. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento central encontra-se destacado de modo a informar o utilizador da página na qual ele se encontra. Esta *navbar* é utilizada em todas as subseqüentes *mockups*. Imediatamente abaixo da *navbar*, da esquerda para a direita, estão colocados a foto de perfil do utilizador, o seu nome de utilizador, um botão de redirecionamento para a página “Editar Perfil” imediatamente abaixo do mesmo, um botão de redirecionamento para a página “Construções em Produção”, e um botão de redirecionamento para a página “Stock”. Abaixo dessa secção encontra-se um grupo de três botões de redirecionamento, que redirecionam o utilizador para as páginas “Fila de Espera”, “Catálogo” e “Construções Acabadas”, respetivamente, da esquerda para a direita.

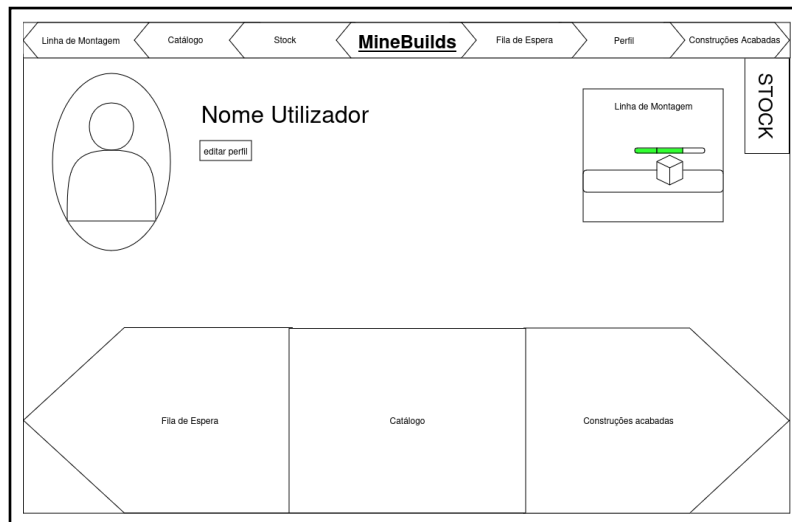


Figura 30: Mockup da Página "Inicial"

#### 5.2.4. Perfil

A Figura 31 representa a página “Perfil”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento com o nome “Perfil” encontra-se destacado de modo a informar o utilizador da página na qual ele se encontra. Nesta página, existem três campos de texto modificáveis, com os nomes “Email”, “Username” e “Palavra Passe”. No canto inferior direito está colocado um botão com o nome “Edit”, que permite ao utilizador modificar os campos sobreditos. Após clicar nesse botão, o botão altera o seu nome para “Salvar”, e um clique subsequente valida os dados atualmente presentes nos campos. Se os dados forem válidos, os mesmos são atualizados na aplicação e os campos são novamente bloqueados e o botão retorna ao seu estado inicial. Se os dados forem inválidos, um erro a indicar o(s) campo(s) inválidos será apresentado ao utilizador. Este *mockup* satisfaz o use case “Visualizar Perfil”.

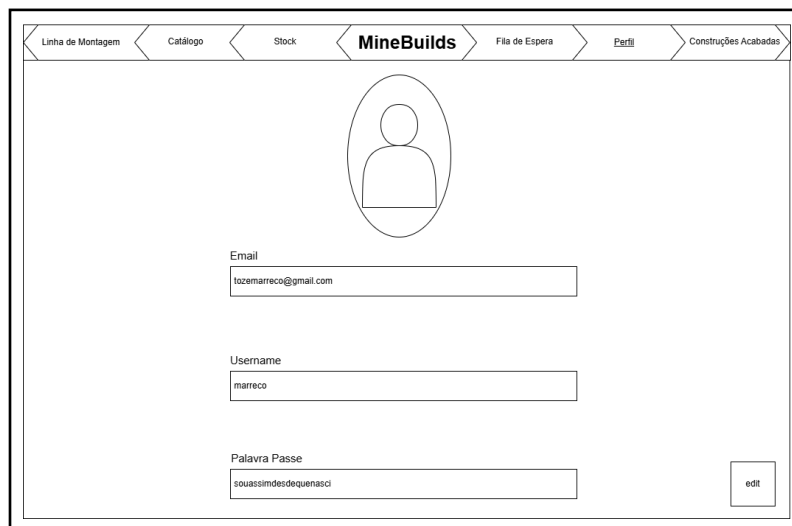


Figura 31: Mockup da Página "Perfil"

#### 5.2.5. Fila de Espera

A Figura 32 representa a página “Fila de espera”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento com o nome “Fila de Espera” encontra-se destacado de modo a informar o utilizador da página na qual ele se encontra. Esta página apresenta a lista de construções em espera para entrar na linha de montagem, organizadas por linhas numa lista. Cada elemento da lista é composto, da

direita para a esquerda, pela imagem da construção, o nome da construção, o número de unidades a serem produzidas nesse lote, um botão com o símbolo “-”, que permite remover uma única unidade do lote, ou o lote inteiro da fila de espera caso o número de unidades seja igual a um, e um botão com o símbolo “x”, que permite remover o lote da fila de espera. Este *mockup* satisfaz o use case “Visualizar Construções Na Fila De Espera”.

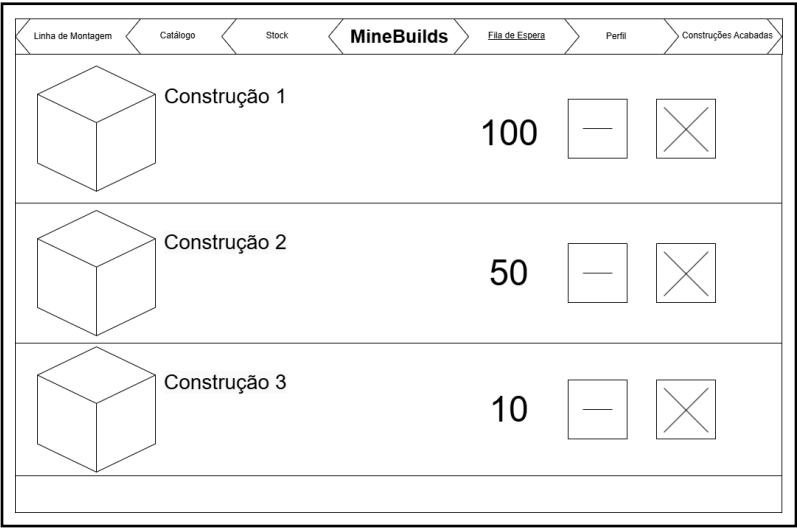


Figura 32: Mockup da Página "Fila de Espera"

5.2.6. Catálogo

A Figura 33 representa a página “Catálogo”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento com o nome “Catálogo” encontra-se destacado de modo a informar o utilizador da página na qual ele se encontra. Nesta página são exibidas as construções que o utilizador pode selecionar para produção. Para cada construção suportada pela aplicação, existe um botão correspondente a essa construção disposto numa grelha composta por linhas de N elementos. Estima-se que cada linha possua 2 elementos, porém o número final está sujeito a uma futura avaliação num protótipo desta interface. Cada elemento da grelha é composto pelo nome da construção, seguido da imagem da construção imediatamente abaixo do mesmo. Cada elemento, após um clique no mesmo, redireciona o utilizador para a página “Adicionar Construção” da construção em questão. Caso não exista stock suficiente para a produção de uma dada construção, o elemento em questão será apresentado com cores sem saturação de modo a indicar que a construção se encontra indisponível. Este *mockup* satisfaz o use case “Visualizar Catálogo”.

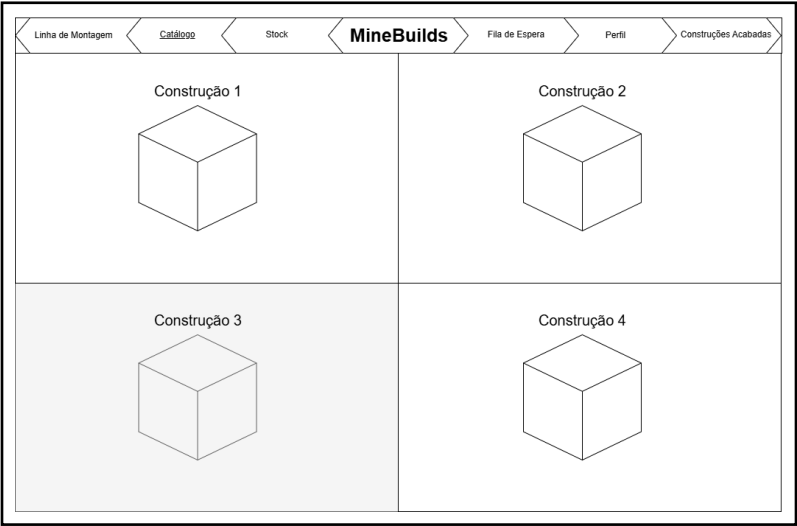


Figura 33: Mockup da Página "Catálogo"

### 5.2.7. Adicionar Construção

A Figura 34 representa a página “Adicionar construção”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. Nenhum elemento se encontra selecionado, dado esta página ser apenas acessível através da página “Catálogo”. Esta página apresenta uma visão geral de uma dada construção selecionada e permite ao utilizador adicionar à fila de espera um lote da construção em questão com tantas unidades quanto existe stock suficiente para a sua produção. Abaixo da *navbar*, da esquerda para a direita, está colocado o nome da construção e um botão que permite retornar à página “Catálogo”. Abaixo dessa secção está colocada uma *preview* da construção, que permite visualizar a construção passo a passo da construção, controlável pela barra de seleção imediatamente abaixo da mesma. À sua direita está colocada uma lista de materiais necessários para a produção do lote. Cada elemento da lista é composto, da esquerda para a direita, pela imagem do material, pelo nome do material e pela quantidade desse material necessária para a produção do lote. Caso a quantidade necessária desse material seja superior à quantidade disponível em stock, o elemento será apresentado em tons de vermelho, de modo a apresentar a indisponibilidade desse material. Abaixo dessa lista, está colocado um campo numérico modificável que determina o número de unidades deste lote. À sua direita encontra-se um botão com o símbolo “+”, que permite incrementar em uma unidade o número de unidades deste lote. Este *mockup* satisfaz o use case “Adicionar Construção” e “Visualizar Materiais de Construção”.

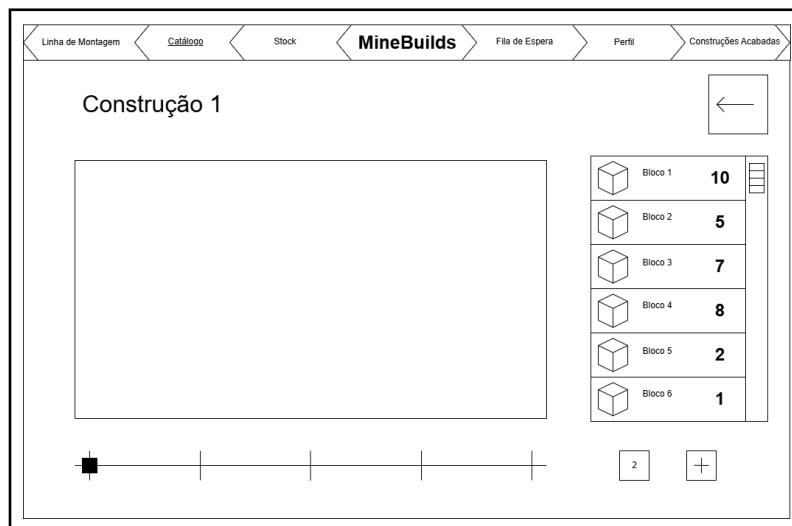


Figura 34: Mockup da Página "Adicionar Construção"

### 5.2.8. Linha de Montagem

A Figura 35 representa a página “Linha de Montagem”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento com o nome “Linha de Montagem” encontra-se destacado de modo a informar o utilizador da página na qual ele se encontra. Esta página apresenta a lista das construções na linha de montagem, onde todos os elementos da lista são um estágio diferente de produção. Cada linha é composta, da esquerda para a direita, pela imagem da construção, pelo nome da construção, pelo estágio de produção na qual a construção se encontra, representado por uma barra de progresso, e o tempo estimado para término da produção. Este *mockup* satisfaz o use case “Visualizar Construções na Linha de Montagem”.



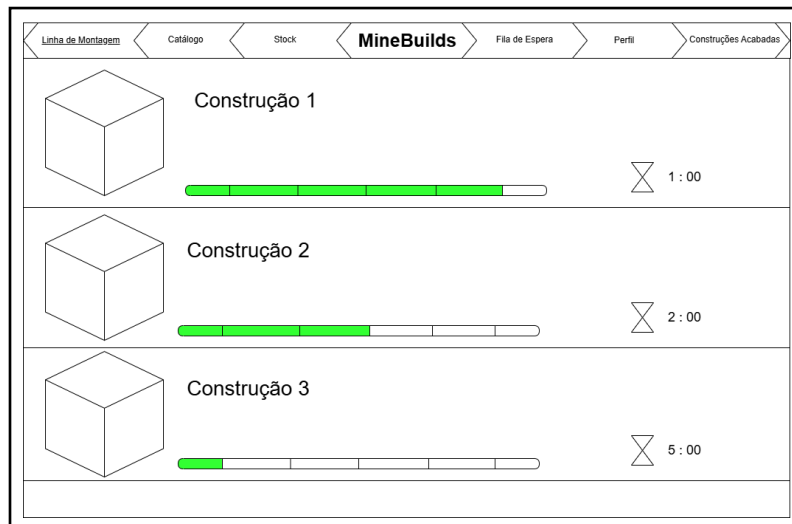


Figura 35: Mockup da Página "Linha de Montagem"

### 5.2.9. Construção em Produção

A Figura 36 representa a página “Construção em Produção”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. Nenhum elemento se encontra selecionado, dado esta página ser apenas acessível através da página “Linha de Montagem”. Esta página apresenta uma estrutura similar à página “Adicionar Construção”, com algumas diferenças: O botão no canto superior direito redireciona o utilizador para a página “Linha de Montagem”. A lista de materiais tem uma estrutura idêntica à sua contraparte, com a diferença de mostrar o número de unidades consumidas seguido do número de unidades alocadas para cada material da construção. Não é possível ocorrer o caso onde um dado material não se encontre disponível, logo não existe para estes elementos um *design* específico para tal caso. Os controlos do número de unidades de um lote não se encontram disponíveis, dada esta página apenas tratar de uma única unidade de uma construção. A barra de controlo da *preview* da construção funciona de modo idêntico à sua contraparte, com a diferença de permitir apenas a seleção dos estágios anteriores e inclusive ao estágio atual da construção na linha de montagem. Este *mockup* satisfaz os use cases “Visualizar Construção em Produção” e “Visualizar Materiais de Construção”.

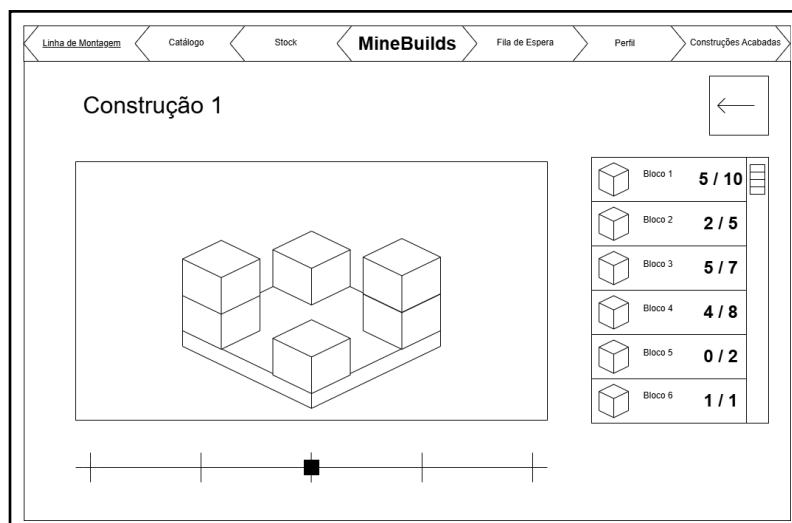


Figura 36: Mockup da Página "Construção em Produção"

### 5.2.10. Construções Acabadas

A Figura 37 representa a página “Construções Acabadas”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento com o nome “Construções Acabadas” encontra-se destacado de modo a informar o utilizador da página na qual ele se encontra. Esta página apresenta uma lista contendo o histórico dos lotes produzidos e atualmente em produção. O conteúdo da página é unicamente uma lista contendo uma linha para cada lote produzido. Cada linha é composta, da esquerda para a direita, pela imagem da construção, pelo nome da construção e por um campo contendo o número de unidades produzidas seguido do número de unidades encomendadas no lote. Cada linha, após um clique na mesma, redireciona o utilizador para a página “Construção” acabada do lote em questão. Este *mockup* satisfaz o use case “Visualizar Construções Produzidas”.

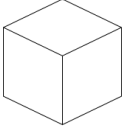
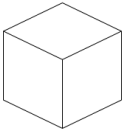
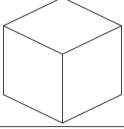
Linha de Montagem	Catálogo	Stock	MineBuilds	Fila de Espera	Perfil	Construções Acabadas
	Construção 1	90/100				
	Construção 2	50/50				
	Construção 3	10/10				

Figura 37: Mockup da Página "Construções Acabadas"

### 5.2.11. Construção Acabada

A Figura 38 representa a página “Construção Acabada”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. Nenhum elemento se encontra selecionado, dado esta página ser apenas acessível através da página “Construção Acabada”. Esta página apresenta uma estrutura similar à página “Adicionar Construção”, com duas únicas diferenças: O botão no canto superior direito redireciona o utilizador para a página “Construções Acabadas”. Os controlos do número de unidades do lote são substituídos por um campo contendo o número de unidades produzidas seguido do número de unidades encomendadas no lote. Este *mockup* satisfaz os use cases “Visualizar Construção Produzida” e “Visualizar Materiais de Construção”.

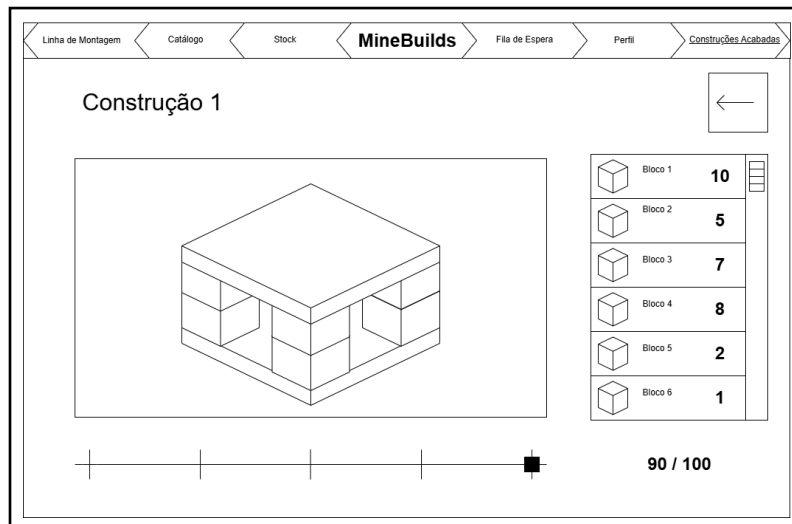


Figura 38: Mockup da Página "Construção Acabada"

### 5.2.12. Stock

A Figura 39 representa a página "Stock". Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento com o nome "Stock" encontra-se destacado de modo a informar o utilizador da página na qual ele se encontra. A página apresenta uma lista de todos os materiais que podem ser utilizados na linha de montagem. Para cada material, existe um elemento correspondente a essa construção disposto numa grelha composta por linhas de N elementos. Estima-se que cada linha possua 3 elementos, porém o número final está sujeito a uma futura avaliação num protótipo desta interface. Cada elemento da grelha é composto pelo nome do material, seguido da imagem do material imediatamente abaixo do mesmo. No canto inferior direito de cada elemento encontra-se o número de unidades disponíveis desse material. Cada elemento, após um clique no mesmo, adiciona uma unidade à guia de encomenda atual. No canto inferior direito da página encontram-se dois botões, com os nomes "Ver Encomendas" e "Encomendar", que redirecionam o utilizador para as páginas "Encomendas" e "Encomendar", respetivamente. Este *mockup* satisfaz o use case "Visualizar Stock".

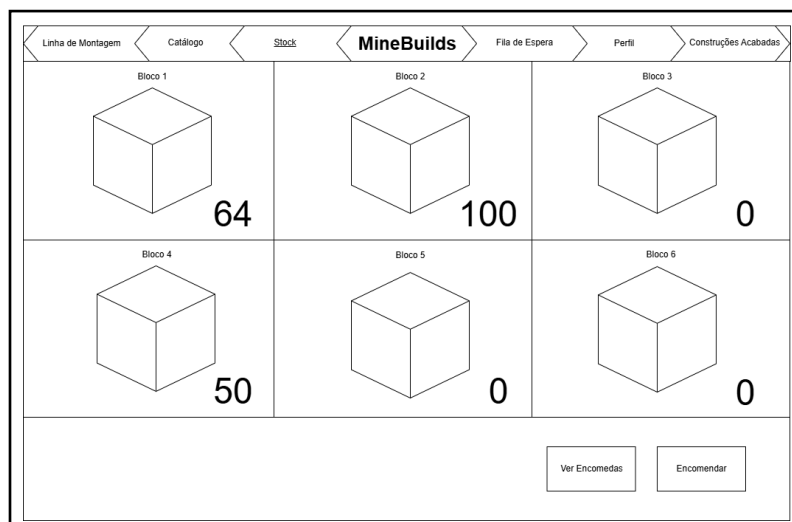


Figura 39: Mockup da Página "Stock"

### 5.2.13. Encomendar

A Figura 40 representa a página "Encomendar". Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas

dentro da aplicação. O elemento com o nome “Stock” encontra-se destacado de modo a informar o utilizador do domínio no qual ele se encontra. A página permite gerir e confirmar a guia de comenda atual. A página apresenta uma lista de todos os materiais a encomendar. Cada elemento é composto, da esquerda para a direita, da imagem do material, do nome do material e dos controlos do material em questão, que por sua vez são compostos, por um campo numérico modificável, que representa o número de unidades a encomendar do material em questão, com dois botões à sua esquerda e direita, com os símbolos “-” e “+”, que incrementam e decrementam o número de unidades desse material, respetivamente. Ancorado ao fundo da página, estão colocados, da esquerda para a direita, um botão que redireciona o utilizador para a página “Stock” e um botão que confirma a guia de encomenda atual e automaticamente cria uma guia nova. Este *mockup* satisfaz o use case “Encomendar Stock”.

O mockup da página "Encomendar" apresenta uma interface com uma barra de navegação superior contendo os seguintes elementos: "Linha de Montagem", "Catálogo", "Stock" (destacado), "MineBuilds", "Fila de Espera", "Perfil" e "Construções Acabadas".

Abaixo da barra de navegação, há uma lista de materiais a encomendar:

- Bloco 1:** Representado por um ícone de bloco, com um campo numérico centralizado no valor "50", precedido por um símbolo "-" e seguido por um símbolo "+".
- Bloco 2:** Representado por um ícone de bloco, com um campo numérico centralizado no valor "45", precedido por um símbolo "-" e seguido por um símbolo "+".
- Bloco 3:** Representado por um ícone de bloco, com um campo numérico centralizado no valor "45", precedido por um símbolo "-" e seguido por um símbolo "+".

Na base da interface, há uma barra contendo um botão com uma seta para a esquerda (para voltar à página "Stock") e um botão rotulado "Confirmar" (para confirmar a encomenda).

Figura 40: Mockup da Página "Encomendar"

#### 5.2.14. Encomendas

A Figura 41 representa a página “Encomendas”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento com o nome “Stock” encontra-se destacado de modo a informar o utilizador do domínio no qual ele se encontra. Esta página apresenta uma lista contendo o histórico de encomendas, incluindo as encomendas não entregues. Cada linha é composta, da esquerda para a direita, e de cima para baixo, pela imagem do material, pelo nome do material e pelo tempo estimado até à entrega da encomenda. Caso a encomenda já tenha sido entregue, tal campo é substituído pela data na qual a encomenda foi entregue. Cada linha, após um clique na mesma, redireciona o utilizador para a página “Encomenda” relativa à encomenda em questão.

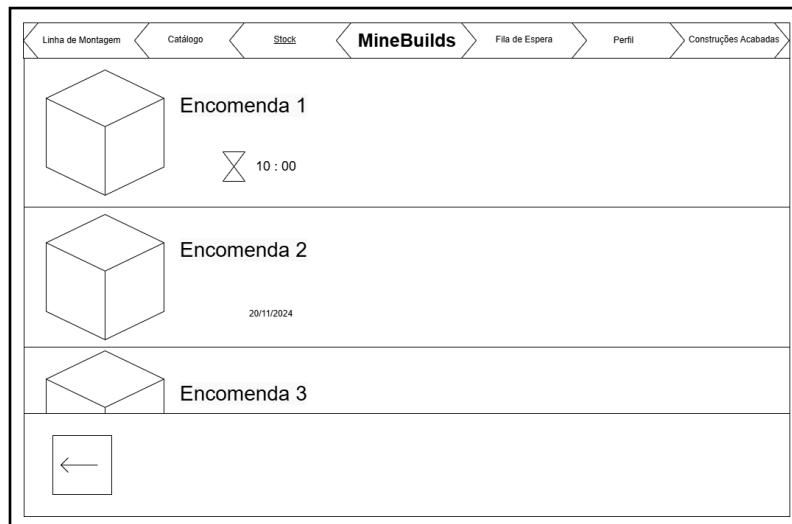


Figura 41: Mockup da Página "Encomendas"

### 5.2.15. Encomenda

A Figura 42 apresenta a página “Encomenda”. Ancorada ao topo da página, ocupando todo o espaço horizontal da mesma, está colocada uma *navbar* que contém elementos de navegação para outras páginas dentro da aplicação. O elemento com o nome “Stock” encontra-se destacado de modo a informar o utilizador do domínio no qual ele se encontra. A página apresenta uma lista de todos os materiais encomendados por essa guia, bem como as quantidades de cada um. Para cada material, existe um elemento correspondente a essa construção disposto numa grelha composta por linhas de N elementos. Estima-se que cada linha possua 3 elementos, porém o número final está sujeito a uma futura avaliação num protótipo desta interface. Cada elemento da grelha é composto pelo nome do material, seguido da imagem do material imediatamente abaixo do mesmo. No canto inferior direito de cada elemento encontra-se o número de unidades encomendadas desse material. No canto inferior direito da página encontra-se um único botão, que redireciona o utilizador para a página “Stock”.

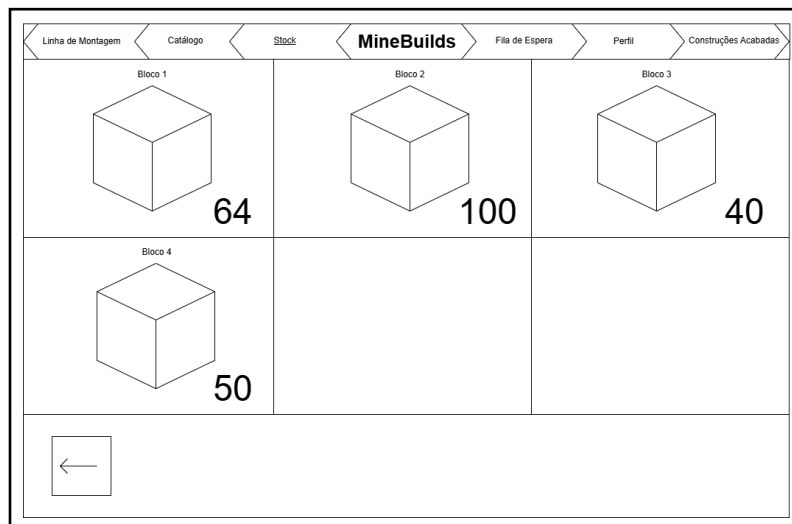


Figura 42: Mockup da Página "Encomenda"

## 6. Conclusões e Trabalho Futuro

### 6.1. Conclusões

Nesta primeira fase do projeto **MineBuilds**, foi realizada uma análise detalhada do domínio do problema e definidos os principais requisitos funcionais e não funcionais. A equipa também desenvolveu modelos conceptuais e lógicos que servirão de base para a implementação futura da aplicação. Este trabalho inicial forneceu um entendimento claro das necessidades dos utilizadores e estabeleceu uma arquitetura sólida para orientar o desenvolvimento.

Entre os principais resultados alcançados nesta etapa, destacam-se:

- A definição de requisitos abrangentes que incluem autenticação, gestão de perfil, gestão de linha de montagem e controle de stock;
- A elaboração de diagramas de atividades e use cases que esclarecem o comportamento esperado do sistema;
- A conceção inicial da estrutura de dados, através de um modelo lógico detalhado, preparado para suportar as funcionalidades planeadas.

### 6.2. Trabalho futuro

O próximo passo no desenvolvimento do projeto envolve a transição para a implementação prática, com foco nas seguintes áreas:

1. Desenvolvimento de Código:
  - Implementação do *backend* utilizando a *framework* .NET com a linguagem C#, com base nos requisitos funcionais detalhados.
2. Criação da Interface Web:
  - Design e desenvolvimento de uma interface intuitiva, garantindo a acessibilidade para diferentes perfis de utilizadores.
3. Implementação da Base de Dados:
  - Criação das tabelas e relações definidas no modelo lógico utilizando o Microsoft SQL Server.
  - Otimização do acesso aos dados para assegurar um desempenho adequado.
4. Integração e Testes:
  - Integração de todos os componentes do sistema (*backend*, *frontend* e base de dados) para garantir a funcionalidade completa.
  - Testes funcionais e não funcionais para validar os requisitos definidos.

À medida que estas tarefas forem concluídas, será possível avaliar a aplicação em funcionamento e recolher feedback para ajustar e melhorar a solução proposta.

## Lista de Definições

**Reload** Processo de reinicialização do conteúdo de uma dada página de uma aplicação web.

**Web App / Aplicação Web** Interface gráfica de Utilizador baseada numa página acessível através de um *browser* de Internet.

**Backend** Parte de uma aplicação que opera nos bastidores, responsável por processar os dados, gerir a lógica de negócio e fornecer respostas ao *frontend* através de interfaces ou outras formas de comunicação.

**Frontend** Parte visível de uma aplicação que interage diretamente com os utilizadores, composta por interfaces gráficas, elementos de design e mecanismos para receber e apresentar informações do *backend*.

**Framework** Estrutura ou conjunto de ferramentas que fornece funcionalidades pré-definidas para facilitar o desenvolvimento de software, promovendo eficiência e padronização.

**Database Management System (DBMS)** Software utilizado para criar, gerir e manter bases de dados, permitindo a interação com os dados de forma eficiente e estruturada.

**GANTT** Diagrama utilizado no planeamento e controlo de projetos, que ilustra graficamente o cronograma de tarefas e as suas dependências.

**Use Case** Técnica utilizada para especificar os comportamentos esperados de um sistema, descrevendo interações entre utilizadores e funcionalidades do software.

**Diagrama de Atividades** Representação gráfica do fluxo de trabalho ou de operações num sistema, evidenciando os passos e decisões necessários para alcançar um objetivo.

**Mockup** Protótipo visual de uma interface de utilizador, que mostra o design e a disposição dos elementos antes da implementação.

**Preview** Representação prévia ou visualização antecipada de um conteúdo ou funcionalidade..

**Navbar / Barra de Navegação** Componente da interface de utilizador que organiza e facilita o acesso a diferentes secções ou funcionalidades de uma aplicação ou site.

## Lista de Siglas e Acrónimos

<b>BD</b>	Base de Dados
<b>DW</b>	Data Warehouse
<b>OLTP</b>	On-Line Analytical Processing
<b>UML</b>	Unified Modeling Language
<b>LN</b>	Lógica de Negócio
<b>UI</b>	User Interface
<b>DL</b>	Data Layer
<b>DAO</b>	Data Access Object
<b>ORM</b>	Object-Relational Mapping
<b>SQL</b>	Structured Query Language
<b>DBMS</b>	Database Management System