|  |  |
| --- | --- |
| Plano | Resultado de imagem para governo de portugal educaÃ§Ã£o  **PAP**  Projeto de Aptidão Profissional    **Second World RP**  Duarte Santos, Simão Teixeira  22358, 22313    **Ano Letivo 2020/2021**  Curso cofinanciado    eu_ce_vert_grdeu_ce_vert_grdeu_ce_vert_grdeu_ce_vert_grd  **[DOC GPSI\_PAP\_206]** |

**Índice**

[1. Introdução 3](#_Toc58623145)

[1.1 Âmbito 3](#_Toc58623146)

[1.2 Enquadramento 3](#_Toc58623147)

[1.4 Suporte técnico 3](#_Toc58623148)

[1.4 Desenvolvimento 4](#_Toc58623149)

[2. Modelo Concetual 6](#_Toc58623150)

[2.1 Diagramas de Uso de Caso 6](#_Toc58623151)

[2.2 Desenho Comportamental do Sistema 12](#_Toc58623152)

[2.2.1 Como jogar no servidor 13](#_Toc58623153)

[2.2.2 Sistema de Garagens/Carros 14](#_Toc58623154)

[2.2.3 Sistema de Vida 14](#_Toc58623155)

[2.2.4 Sistema de inventários 15](#_Toc58623156)

[2.2.5 Sistema de Logs do Discord (Administradores) 15](#_Toc58623157)

[2.2.6 Sistemas de Dinheiro Sujo 16](#_Toc58623158)

[3. Modelo Lógico 17](#_Toc58623159)

[3.1 Diagrama Entidade-Relação 17](#_Toc58623160)

[3.2 Modelo relacional 18](#_Toc58623161)

[4 Modelo Físico 19](#_Toc58623162)

[4.1 Dicionário de dados 19](#_Toc58623163)

[4.2 Modelo físico de Implementação da Base de Dados 23](#_Toc58623164)

[5. Protótipo 31](#_Toc58623165)

[6. Análise crítica 37](#_Toc58623166)

[7. Fontes Bibliografia | Webgrafia 38](#_Toc58623167)

[8. Reflexão final 39](#_Toc58623168)

1. Introdução

1.1 Âmbito

A nossa PAP tem como finalidade o entretenimento de todas as pessoas que já conhecem o jogo GTA V e pretendem ter uma experiência diferente dentro dele. O nosso projeto consubstancia-se no de uma camada para o jogo ajustada à realidade, onde os utilizadores poderão simular o seu quotidiano como arranjar um emprego, conviver com outros jogadores, comprar carros, entre muitas outras coisas que podem ser feitas tal como na vida real. Alem dis desenvolvimento so tencionamos construir uma comunidade para o servidor usando o Discord como meio de comunicação.

1.2 Enquadramento

O GTA V é um jogo de Ação e Aventura com Mundo Aberto. Foi lançado em 2013 pela empresa Rockstar fazendo parte da saga de GTA’s que começou em 2008 com o GTA 1. É um jogo que é possível ser jogado em 7 modelos de consolas e em computador que coleciona inúmeros recordes desde que foi publicado:

* O jogo que mais cópias vendeu em 24 horas (11 milhões e 210 mil cópias)
* O jogo/produto online com mais lucro em 24 horas (815 milhões de dólares)
* O jogo/produto online que rendeu 1 bilião de dólares mais rápido (4 dias)
* Em fevereiro de 2020 tornou se o jogo com mais vendas de sempre com 120 milhões de cópias vendidas
* Trailer mais visualizado de jogos

Atualmente é um jogo que pesa mais de 90GB e que continua a receber suporte e a realizar grandes eventos como aconteceu em maio de 2020 em que esteve durante 7 dias gratuito na loja online *Epic Games* o que causou uma quebra de resposta dos servidores da loja durante algumas horas devido à quantidade de pessoas a tentar comprar o GTA V.

Iremos utilizar o FiveM que é uma plataforma digital onde estão alojados milhares de servidores comunitários do jogo GTA V.

1.4 Suporte técnico

O Fivem disponibiliza para os programadores várias livrarias para programarmos o nosso servidor. É possível fazer scripts em Lua, C# e JS, mas a nossa escolha irá ser Lua (cliente e servidor), pelo facto de haver muito mais material e suporte na internet para LUA, no Fivem do que as outras duas linguagens.

O FiveM possui algumas micro-frameworks desenvolvidas pela comunidade e nós iremos usar uma chamada ES\_Extended.

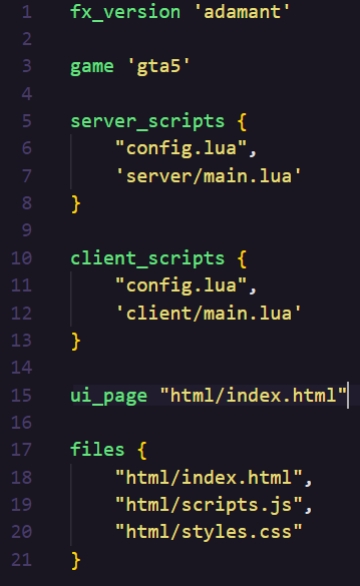
As nossas interfaces irão ser desenvolvidas em HTML, JS e CSS para além do SQL para as consultas a base de dados que será em Maria DB devido ao FiveM apenas aceitar este tipo de base de dados. Vamos usar o Visual Studio Code e o Heidi SQL durante o desenvolvimento do servidor.

Para alem disso iremos usar alguns mapas e carros pré feitos pela comunidade para embelezar um pouco melhor o nosso mapa de jogo.

1.4 Desenvolvimento

Para construir um servidor no FiveM devemos usar .dll’s que o FiveM nos disponibiliza. Com essas .dll´s podemos contruir scripts para modificar o jogo GTA V. Alem dessas .dll´s vamos usar um micro-framework chamada ES\_EXTENDED que nos vai facilitar na performance e gestão de utilizadores do servidor. Também vamos usar uma base de dados MYSQL onde usaremos uma livraria chamada MYSQL-ASYNC para nos ajudar a ter um acesso mais seguro e fácil à base de dados.

Os scripts podem ser feitos com Lua, C# e JavaScript, mas a nossa escolha vai ser Lua devido à vasta quantidade de material e suporte que existe para a criação de scripts em Lua. Cada script contém um ficheiro que é executado no lado do cliente e outro que é executado no lado do servidor, ambos os ficheiros estão ligados através de Eventos. Para o sistema saber qual ficheiro ele deve executar no servidor ou no cliente devemos indicá-los noutro ficheiro chamado fxmanifest.lua ou \_\_resouce.lua. Caso o script precise de uma interface em HTML os seus ficheiros também devem ser indicados no *manifest*.

Esse *manifest* tem a seguinte estrutura:

No lado do cliente podemos ler as várias teclas que o utilizador vai pressionando, criar menus e interfaces gráficas em HTML, CSS e JavaScript, alem disso conseguimos executar animações no personagem do jogador.

No lado do servidor é onde se faz o acesso à base de dados e onde se pode disparar eventos para outros utilizadores.

**EventHandlers:** Função que está associada a um evento criado posteriormente e que é executada quando o seu respetivo evento é disparado, pelo sistema ou através do nosso código.

**Evento:** São uma referência que nos permite comunicar com o servidor ou outros scripts que tenhamos criado. Temos a possibilidade de os disparar e todos os seus *Handlers* são executados, também podemos cancelar o seu efeito.

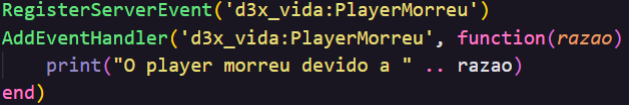
Tanto no lado do cliente como no servidor é possível adicionar *EventHandlers* para os Eventos criados pelo jogo ou até mesmo por nós. É possível criar eventos tanto no lado do servidor como do cliente e é assim que maior parte dos scripts funcionam.

Os eventos criados são mútuos a todos os scripts, ou seja, ao criar o evento "Cair" num script, esse evento poderá ser disparado de outro script completamente diferente, assim como adicionar um *Handler* para esse evento noutro script que não seja o seu criador.

Os eventos, quando disparados, é possível passar-lhes parâmetros de todo o tipo de dados. Existe a possibilidade de passar uma função no parâmetro do evento que depois é executada no *Handler* quando ele bem entender. A isto chamamos de Eventos *Callback*.

**Exemplo:**

**Registo e criação de um *Handler* para um Evento do servidor:**



**Disparar o evento criado no servidor:**



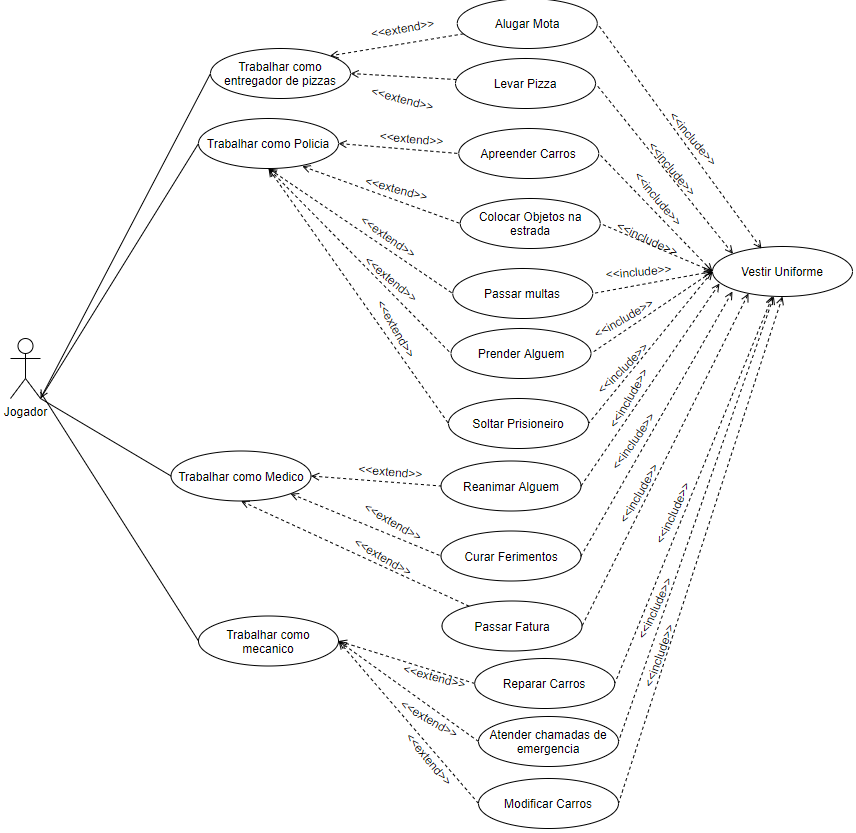
2. Modelo Concetual

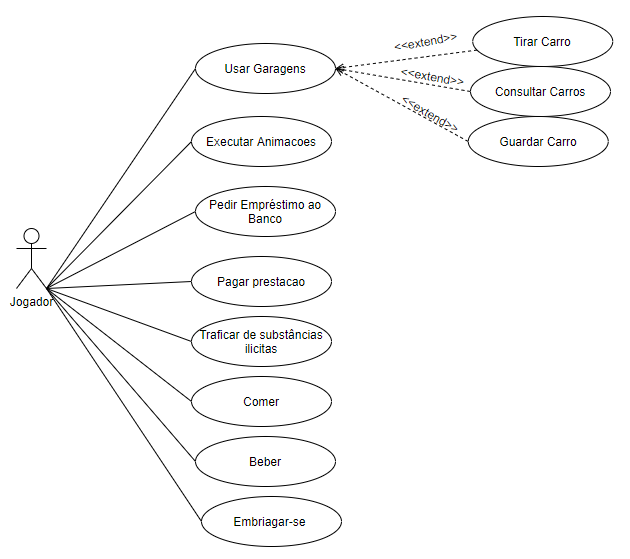
**(Análise de Requisitos e desenho comportamental do sistema)**

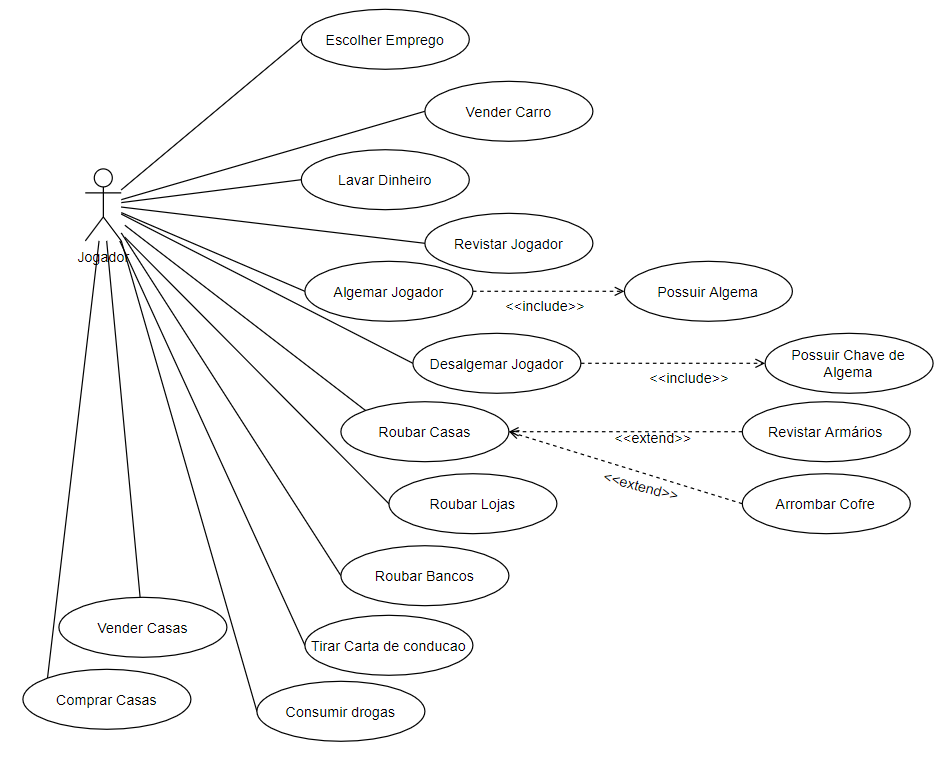
2.1 Diagramas de Uso de Caso



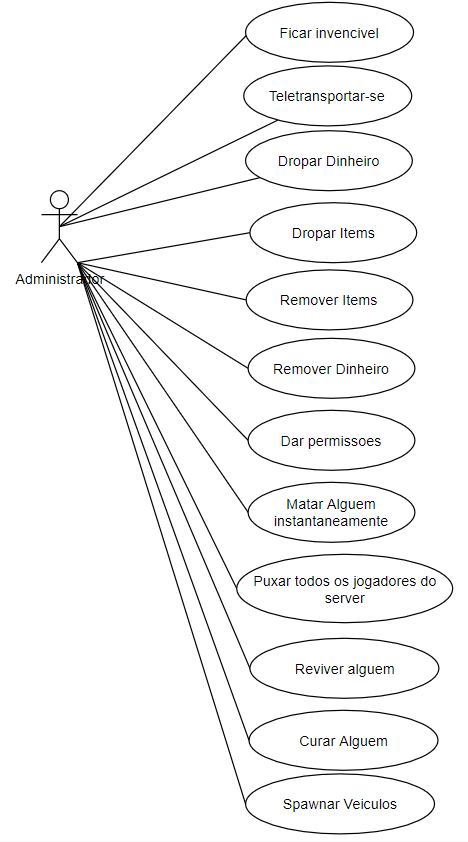




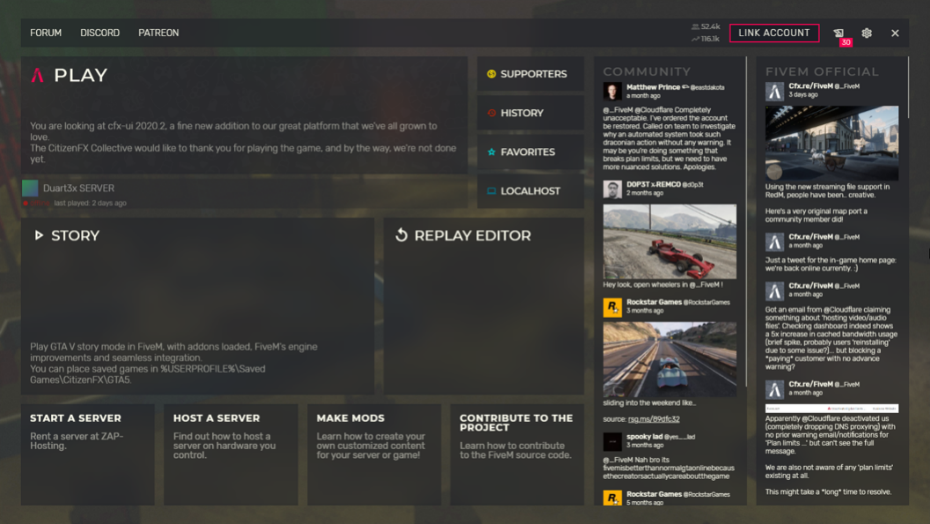






2.2 Desenho Comportamental do Sistema

2.2.1 Como jogar no servidor

Para os jogadores poderem jogar no nosso servidor irão ter de ter instalada versão legal do GTA V e o Launcher FiveM, de seguida abrem o FiveM e pesquisam pelo nome do servidor na barra de pesquisa ou então colocam o IP do servidor e começam a jogar.

Página Inicial FiveM



Página Pesquisa FiveM

Para um melhor controlo dos jogadores que queiram jogar no nosso servidor (FiveM), todos devem ser membros do nosso servidor de Discord, se não a conexão não se irá realizar.

Para fazermos esta validação existe um evento do FiveM que é disparado quando alguém se tenta conectar ao servidor, nesse evento é devolvido um array com as licenças do GTA V da pessoa assim como o seu Id do Discord, a sua licença da Microsoft Store, a sua licença Steam bem como o seu IP. Como é óbvio estes id´s só são recolhidos pelo FiveM caso a pessoa tenha essas aplicações instaladas no seu Computador ou tenha as suas contas associadas a uma conta do FiveM. De seguida com o Id do Discord da pessoa e utilizando a API do mesmo conseguimos saber se aquele ID se encontra ou não no nosso servidor (Discord) e assim barramos-lhe, ou não, a entrada. Depois de estar dentro do servidor é só criar a sua personagem dando-lhe um Nome, Data de nascimento, Sexo e Altura, customizá-la a seu gosto com roupas e outros adereços e, de seguida, o jogador escolhe o seu emprego público ou então espera por fazer uma candidatura para um emprego *whitelisted*.

2.2.2 Sistema de Garagens/Carros

Os jogadores têm a oportunidade de comprar carros no stand, na compra do carro é lhe atribuída uma matrícula que identifica o carro. Quando o jogador não quiser usar mais o carro pode dirigir-se a uma garagem pública devidamente identificada no mapa, nesta garagem poderá guardar o seu carro. Quando precisar dele novamente poderá retirá-lo na mesma garagem.

2.2.3 Sistema de Vida

A personagem de cada jogador tem uma barra de vida, escudo, fome, sede, oxigénio e uma de stress.

O jogador deverá dirigir-se às lojas de comida e comprar comida e água para não morrer de fome ou de sede, pois caso as barras cheguem a 0 o jogador morre.

A barra da vida é recuperada com ligaduras que podem ser aplicadas pelos médicos ou pelo próprio jogador. Caso o jogador fique sem vida o INEM ainda tem a possibilidade de o reanimar, caso contrário o jogador irá ser teletransportado para o Hospital e perde todos os itens que tinha no inventário.

A barra de oxigénio é alterada quando o jogador se encontra de baixo de água.

A barra de stress vai diminuindo ou aumentando consoante as ações que o jogador esteja a desempenhar, se estiver muitas horas acordado o stress vai aumentando, se quiser diminuir o stress deve beber um chá ou tomar algum comprimido.

2.2.4 Sistema de inventários

Cada jogador possui um inventário, bem como as empresas/sociedades. As suas respetivas casas também terão um cofre.

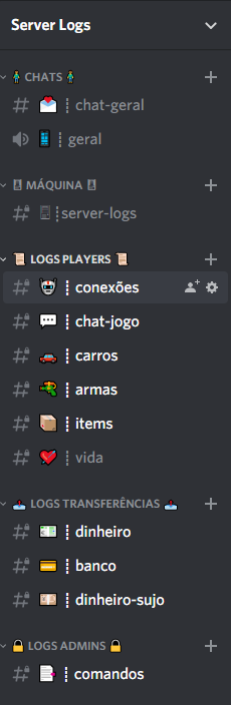
No inventário cada jogador pode guardar Itens como Pão, Água, telemóvel, Algemas etc., bem como armas, dinheiro e dinheiro sujo. Para as armas irá haver espaços para as equipar e usá-las posteriormente.

O inventário de cada jogar é apenas visível para ele mesmo exceto quando este for revistado pela polícia ou assaltado por alguém, já os inventários das empresas/sociedades são públicos para todos os membros das mesmas.

Poderá haver itens no chão e o jogador terá a possibilidade de, ao passar por cima do mesmo, o apanhar e guardar no inventário. Os jogadores conseguem guardar e retirar itens dos inventários das suas casas assim como da empresa a que pertence

2.2.5 Sistema de Logs do Discord (Administradores)

Os administradores têm acesso a todos os inventários assim como às transações que ocorrem entre eles através de um sistema de logs no Discord.

Através da API do Discord conseguimos enviar mensagens do servidor FiveM para salas de texto de um servidor Discord. 

Nesse servidor do Discord iremos ter salas onde irão ser lançadas mensagens sempre que:

* Um jogador se conecta;
* Um jogador se desconecta;
* Um jogador enviar uma mensagem no chat;
* Carros forem spawnados;
* Itens forem transacionados de inventário para inventário;
* Um jogador morre;
* Elevadas quantias forem transacionadas;
* Administradores executarem comandos especiais no chat;
* Existem transações de armas.

2.2.6 **Sistemas de Dinheiro** Sujo

Tal como na vida real existe corrupção e negócios ilegais. Na nossa cidade virtual também não poderia deixar de haver tal coisa, então, para todos aqueles que desejarem fazer esses negócios, irá existir uma "moeda" à parte que é o dinheiro sujo. Como é óbvio o objetivo é que os jogadores evitem essa possibilidade, para isso acontecer a polícia da cidade tem como papel apanhar e prender todos aqueles que tiverem esse tipo de dinheiro ou que sejam apanhados a fazer negócios com ele.

Para um melhor controlo desta situação irá haver uma organização, os *Peaky Blinders* inspirado na série "Peaky Blinders", que irã controlar a corrupção existente na cidade.

O jogador irá ter acesso a esse dinheiro sujo em poucas situações, estas situações irão ser muito raras e restritas a todos os jogadores, das quais se destacam:

* Assaltos a bancos, casas ou lojas em que é roubada uma grande quantia;
* Tráfico de estupefacientes ou qualquer item ilegal aos olhos da lei.

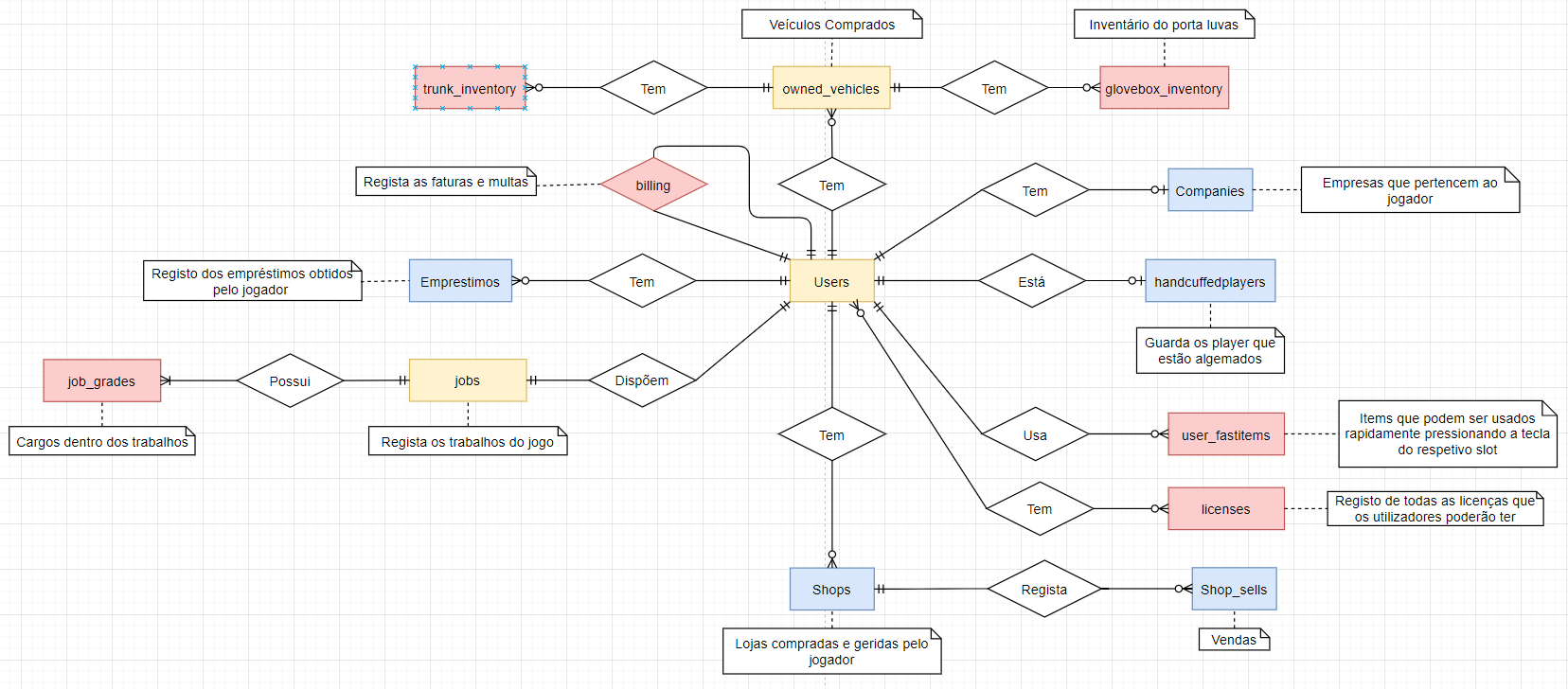
Tal como na vida real teremos meios para prevenir a criminalidade, tal como a Polícia e toda a comunidade de cidadãos que poderão reportar estas situações às autoridades.

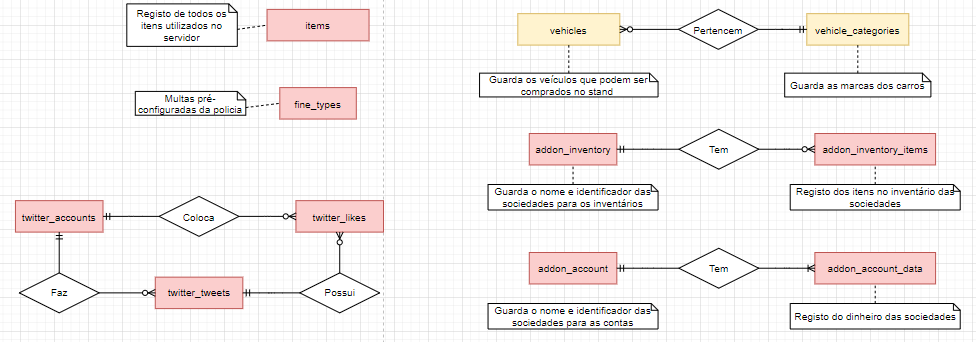
Existe a possibilidade de transformar dinheiro sujo em dinheiro "limpo", para isso irá existir a lavagem de dinheiro, que ainda não decidimos como irá proceder.

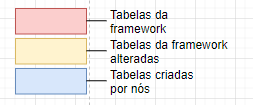
3. Modelo Lógico

**(Estrutura de dados do Sistema de Informação)**

3.1 Diagrama Entidade-Relação







3.2 Modelo relacional

users (identifier<PK>, accounts, group, inventory, ***job***, job\_grade, loadout, position, firstname, lastname, dateofbirth, sex, height, skin, status, is\_dead, phone\_number, createdAt, lastSave)

user\_fastitems (***identifier<FK>***, slot<PK>, weapon)

user\_licenses (id<PK>, ***type<FK>***, ***owner<FK>***)

licenses (type<PK>, label)

vehicles (name, model<PK>, price, ***category<FK>***, imglink)

vehicle\_categories (name<PK>, label)

owned\_vehicles (plate<PK>, ***owner<FK>***, state, vehicle, type, job, stored)

trunk\_inventory (id<PK>, *plate*, data, owned)

glovebox\_inventory (id<PK>, *plate*, data, owned)

jobs (name<PK>, label, whitelisted, description)

job\_grades (id<PK>, ***job\_name<FK>***, grade, name, label, salary, skin\_male, skin\_female)

emprestimos (id<PK>, ***owner<FK>***, descricao, juros, value, prestacao, prestacoes\_totais, n\_prestacoes, timeToEnd, pago, createdAt)

billing (id<PK>, ***identifier<FK>***, ***sender<FK>***, target\_type, target, label, amount)

shops (shopNumber<PK>, ***ownerIdentifier<FK>***, last\_update, shopInventory)

shop\_sells (id<PK>, ***shopNumber<FK>***, buyer, products, value, date)

handcuffedplayers (id<PK>, ***identifier<FK>***)

companies (id<PK>, ***name<FK>***, label, price, owned, ***ownerIdentifier<FK>***, last\_update, private, private\_price)

items (name<PK>, label, weight, rare, can\_remove)

fine\_types (id<PK>, label, amount, category)

twitter\_accounts (id<PK>, username, password, avatar\_url)

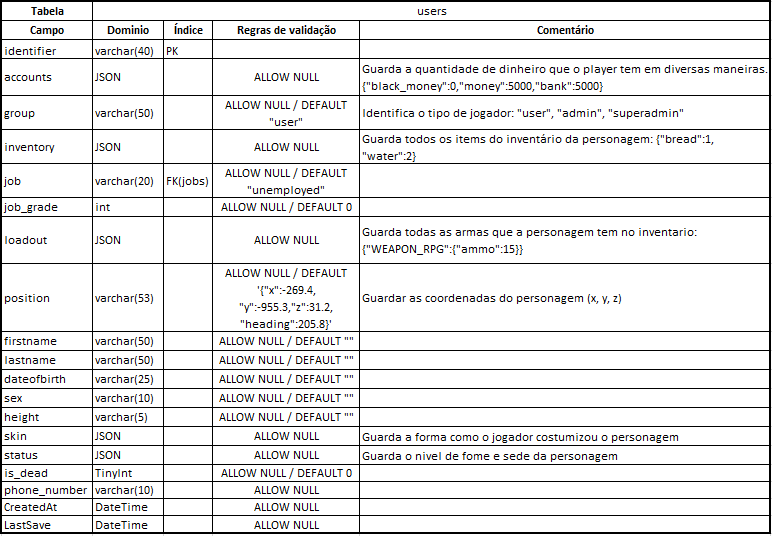
twitter\_likes (id<PK>, ***authorId<FK>***, ***tweetId<FK>***)

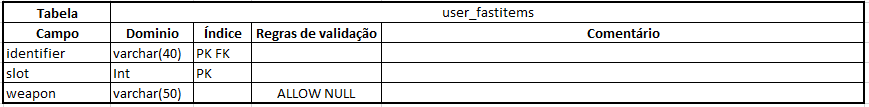
twitter\_tweets (id<PK>, ***authorId<FK>***, realUser, message, time, likes)

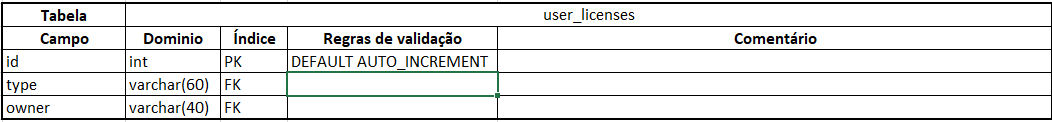
4 Modelo Físico

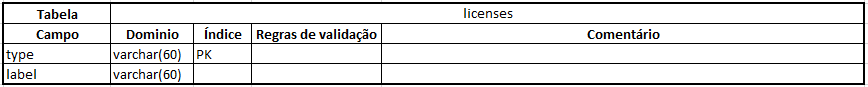
**(Implementação da Base de Dados no SGBD)**

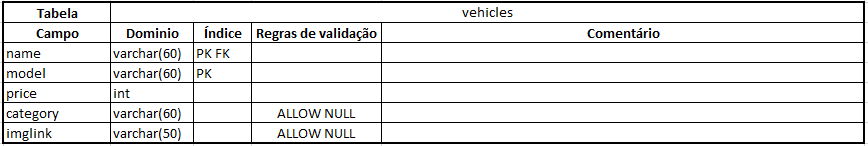
4.1 Dicionário de dados

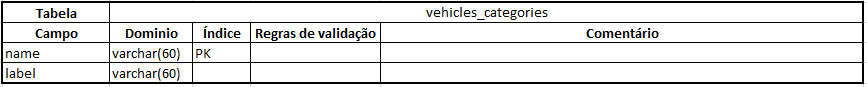


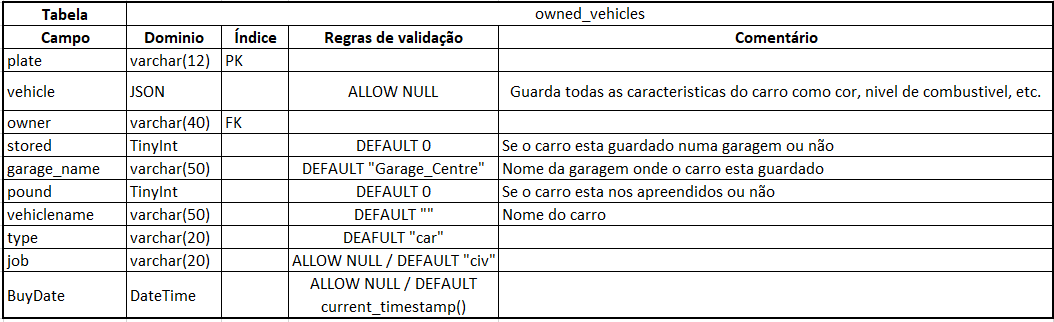


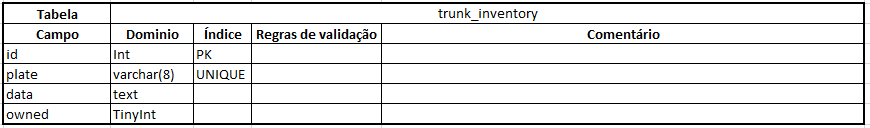




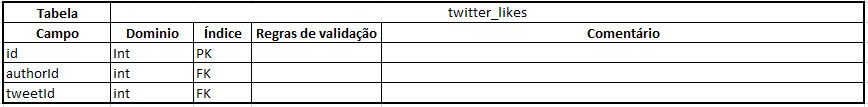


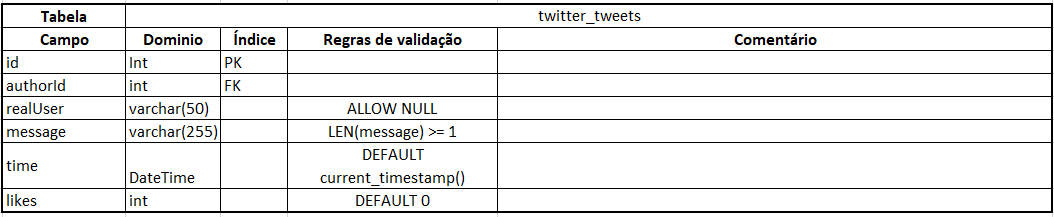


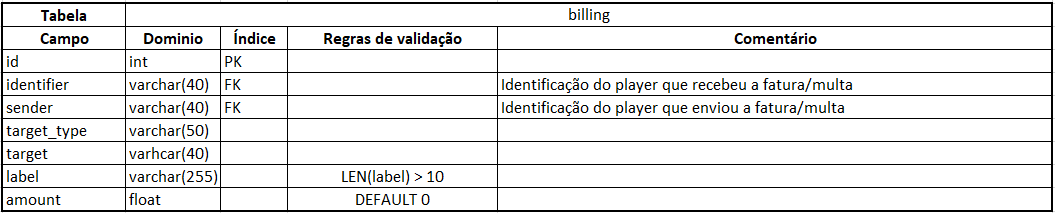


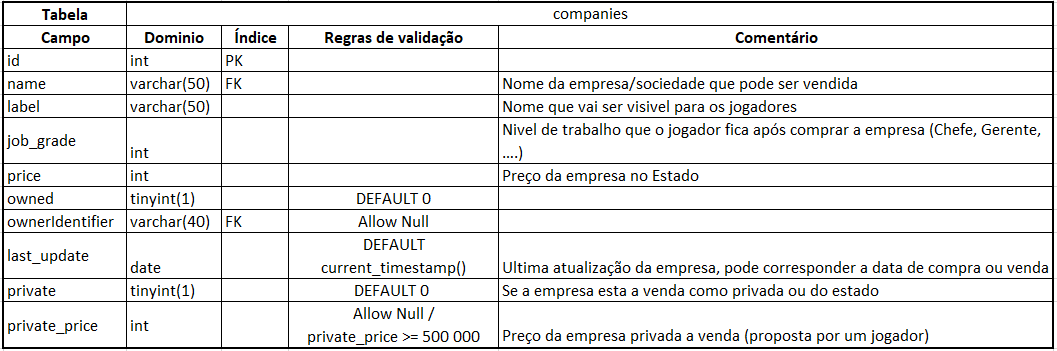


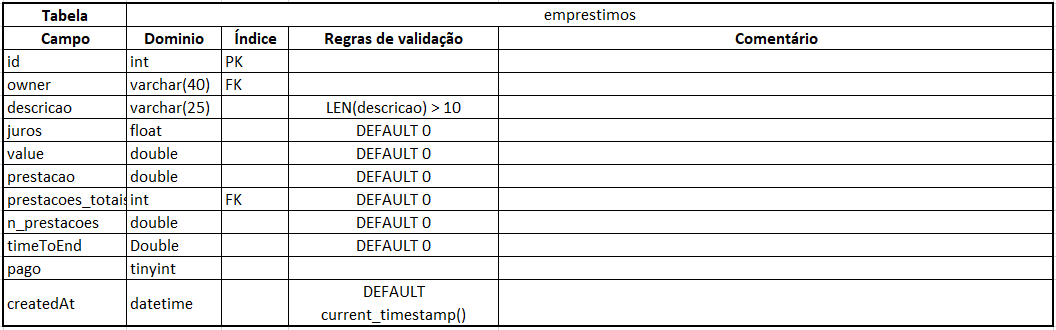


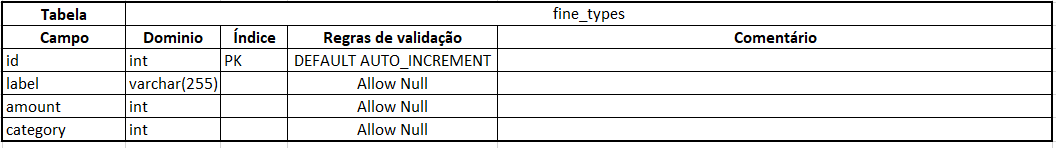


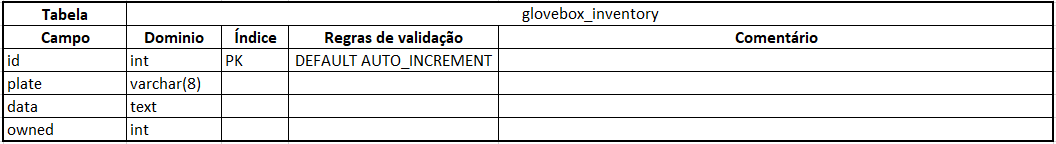


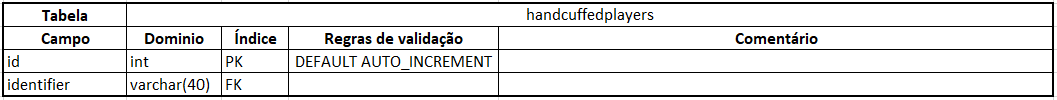


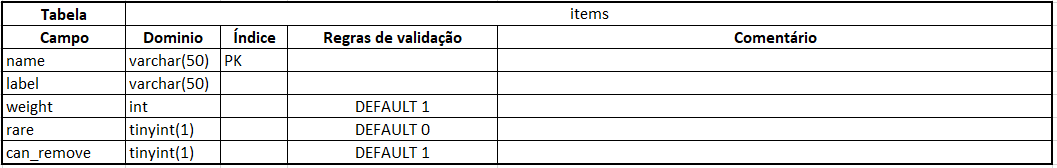


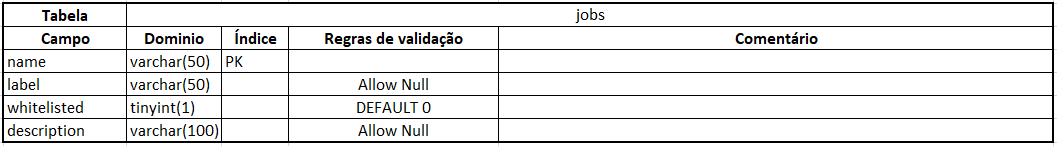


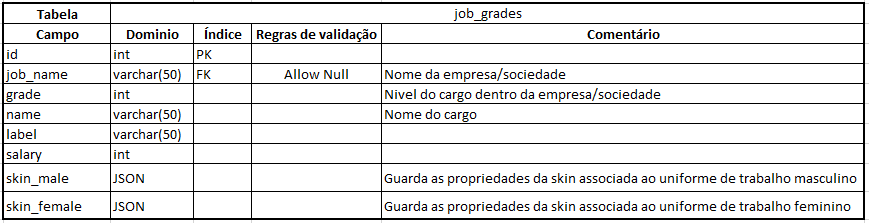


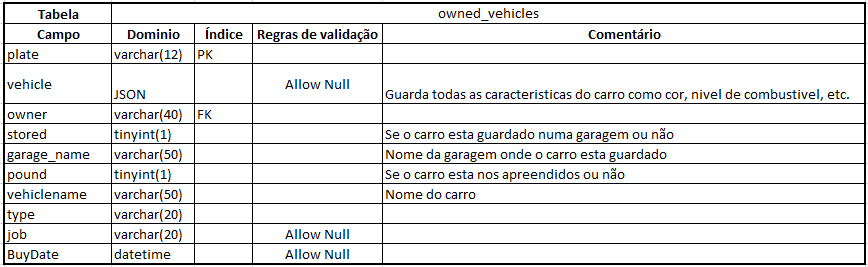


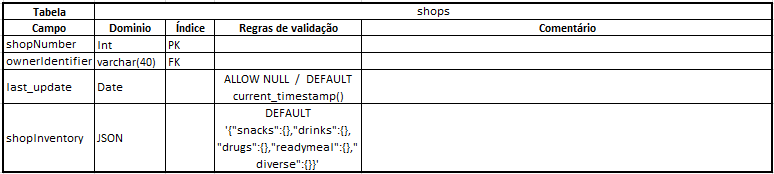












4.2 **Modelo físico de Implementação da Base de** Dados

|  |
| --- |
| **CREATE** **TABLE** `users` (  `identifier` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `accounts` **LONGTEXT** **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `group` **VARCHAR**(50) **NULL** **DEFAULT** 'user' **COLLATE** 'utf8\_general\_ci',  `inventory` **LONGTEXT** **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `job` **VARCHAR**(20) **NULL** **DEFAULT** 'unemployed' **COLLATE** 'utf8\_general\_ci',  `job\_grade` **INT**(11) **NULL** **DEFAULT** '0',  `loadout` **LONGTEXT** **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `position` **VARCHAR**(53) **NULL** **DEFAULT** '{"x":-269.4,"y":-955.3,"z":31.2,"heading":205.8}' **COLLATE** 'utf8\_general\_ci',  `firstname` **VARCHAR**(50) **NULL** **DEFAULT** '' **COLLATE** 'utf8\_general\_ci',  `lastname` **VARCHAR**(50) **NULL** **DEFAULT** '' **COLLATE** 'utf8\_general\_ci',  `dateofbirth` **VARCHAR**(25) **NULL** **DEFAULT** '' **COLLATE** 'utf8\_general\_ci',  `sex` **VARCHAR**(10) **NULL** **DEFAULT** '' **COLLATE** 'utf8\_general\_ci',  `height` **VARCHAR**(5) **NULL** **DEFAULT** '' **COLLATE** 'utf8\_general\_ci',  `skin` **LONGTEXT** **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `status` **LONGTEXT** **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `is\_dead` **TINYINT**(1) **NULL** **DEFAULT** '0',  `phone\_number` **VARCHAR**(10) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `last\_property` **VARCHAR**(255) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`identifier`) **USING** **BTREE**,  **INDEX** `FK\_userjob` (`job`) **USING** **BTREE**,  **CONSTRAINT** `FK\_userjob` **FOREIGN** **KEY** (`job`) **REFERENCES** `es\_extended`.`jobs` (`name`) **ON** **UPDATE** **SET** **NULL** **ON** **DELETE** **SET** **NULL**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `user\_fastitems` (  `identifier` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `slot` **INT**(11) **NOT** **NULL**,  `weapon` **VARCHAR**(255) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`identifier`, `slot`) **USING** **BTREE**,  **CONSTRAINT** `FK\_user\_fastitem` **FOREIGN** **KEY** (`identifier`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **CASCADE**  )  **COLLATE**='utf8mb4\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `user\_licenses` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `type` **VARCHAR**(60) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `owner` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_licensetype` (`type`) **USING** **BTREE**,  **INDEX** `FK\_licenseowner` (`owner`) **USING** **BTREE**,  **CONSTRAINT** `FK\_licenseowner` **FOREIGN** **KEY** (`owner`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**,  **CONSTRAINT** `FK\_licensetype` **FOREIGN** **KEY** (`type`) **REFERENCES** `es\_extended`.`licenses` (`type`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**; |
| **CREATE** **TABLE** `licenses` (  `type` **VARCHAR**(60) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `label` **VARCHAR**(60) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`type`) **USING** **BTREE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `vehicles` (  `name` **VARCHAR**(60) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `model` **VARCHAR**(60) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `price` **INT**(11) **NOT** **NULL**,  `category` **VARCHAR**(60) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `imglink` **VARCHAR**(50) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`model`) **USING** **BTREE**,  **INDEX** `FK\_vehicles\_categories` (`category`) **USING** **BTREE**,  **CONSTRAINT** `FK\_vehicles\_categories` **FOREIGN** **KEY** (`category`) **REFERENCES** `es\_extended`.`vehicle\_categories` (`name`) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **RESTRICT**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `vehicle\_categories` (  `name` **VARCHAR**(60) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `label` **VARCHAR**(60) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`name`) **USING** **BTREE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `owned\_vehicles` (  `plate` **VARCHAR**(12) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `vehicle` **LONGTEXT** **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `owner` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `stored` **TINYINT**(1) **NOT** **NULL** **DEFAULT** '0',  `garage\_name` **VARCHAR**(50) **NOT** **NULL** **DEFAULT** 'Garage\_Centre' **COLLATE** 'utf8\_general\_ci',  `pound` **TINYINT**(4) **NOT** **NULL** **DEFAULT** '0',  `vehiclename` **VARCHAR**(50) **NOT** **NULL** **DEFAULT** '0' **COLLATE** 'utf8\_general\_ci',  `type` **VARCHAR**(20) **NOT** **NULL** **DEFAULT** 'car' **COLLATE** 'utf8\_general\_ci',  `job` **VARCHAR**(20) **NULL** **DEFAULT** 'civ' **COLLATE** 'utf8\_general\_ci',  `BuyDate` **DATETIME** **NULL** **DEFAULT** **current\_timestamp**(),  **PRIMARY** **KEY** (`plate`) **USING** **BTREE**,  **INDEX** `FK\_owner\_identifier` (`owner`) **USING** **BTREE**,  **CONSTRAINT** `FK\_owner\_identifier` **FOREIGN** **KEY** (`owner`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**; |
| **CREATE** **TABLE** `trunk\_inventory` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `plate` **VARCHAR**(8) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `data` **TEXT** **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `owned` **INT**(11) **NOT** **NULL**,  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **UNIQUE** **INDEX** `plate` (`plate`) **USING** **BTREE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=14  ; |

|  |
| --- |
| **CREATE** **TABLE** `glovebox\_inventory` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `plate` **VARCHAR**(8) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `data` **TEXT** **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `owned` **INT**(11) **NOT** **NULL**,  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **UNIQUE** **INDEX** `plate` (`plate`) **USING** **BTREE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=20  ; |

|  |
| --- |
| **CREATE** **TABLE** `jobs` (  `name` **VARCHAR**(50) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `label` **VARCHAR**(50) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `whitelisted` **TINYINT**(1) **NOT** **NULL** **DEFAULT** '0',  `description` **VARCHAR**(200) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`name`) **USING** **BTREE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `job\_grades` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `job\_name` **VARCHAR**(50) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `grade` **INT**(11) **NOT** **NULL**,  `name` **VARCHAR**(50) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `label` **VARCHAR**(50) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `salary` **INT**(11) **NOT** **NULL**,  `skin\_male` **LONGTEXT** **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `skin\_female` **LONGTEXT** **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_jobname` (`job\_name`) **USING** **BTREE**,  **CONSTRAINT** `FK\_jobname` **FOREIGN** **KEY** (`job\_name`) **REFERENCES** `es\_extended`.`jobs` (`name`) **ON** **UPDATE** **NO** **ACTION** **ON** **DELETE** **CASCADE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=57; |
| **CREATE** **TABLE** `billing` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `identifier` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `sender` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `target\_type` **VARCHAR**(50) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `target` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `label` **VARCHAR**(255) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `amount` **FLOAT** **NOT** **NULL** **DEFAULT** '0',  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_bills\_user` (`identifier`) **USING** **BTREE**,  **INDEX** `FK\_\_bills\_sender` (`sender`) **USING** **BTREE**,  **CONSTRAINT** `FK\_\_bills\_sender` **FOREIGN** **KEY** (`sender`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **NO** **ACTION** **ON** **DELETE** **NO** **ACTION**,  **CONSTRAINT** `FK\_bills\_user` **FOREIGN** **KEY** (`identifier`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **NO** **ACTION** **ON** **DELETE** **NO** **ACTION**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `twitter\_accounts` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `username` **VARCHAR**(50) **NOT** **NULL** **DEFAULT** '0' **COLLATE** 'utf8\_general\_ci',  `password` **VARCHAR**(64) **NOT** **NULL** **DEFAULT** '0' **COLLATE** 'utf8mb4\_bin',  `avatar\_url` **VARCHAR**(255) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8mb4\_bin',  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **UNIQUE** **INDEX** `username` (`username`) **USING** **BTREE**  )  **COLLATE**='utf8mb4\_bin'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=39  ; |

|  |
| --- |
| **CREATE** **TABLE** `twitter\_likes` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `authorId` **INT**(11) **NULL** **DEFAULT** **NULL**,  `tweetId` **INT**(11) **NULL** **DEFAULT** **NULL**,  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_twitter\_likes\_twitter\_accounts` (`authorId`) **USING** **BTREE**,  **INDEX** `FK\_twitter\_likes\_twitter\_tweets` (`tweetId`) **USING** **BTREE**,  **CONSTRAINT** `FK\_twitter\_likes\_twitter\_accounts` **FOREIGN** **KEY** (`authorId`) **REFERENCES** `es\_extended`.`twitter\_accounts` (`id`) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **RESTRICT**,  **CONSTRAINT** `FK\_twitter\_likes\_twitter\_tweets` **FOREIGN** **KEY** (`tweetId`) **REFERENCES** `es\_extended`.`twitter\_tweets` (`id`) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **CASCADE**  )  **COLLATE**='utf8mb4\_bin'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=138  ; |

|  |
| --- |
| **CREATE** **TABLE** `twitter\_tweets` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `authorId` **INT**(11) **NOT** **NULL**,  `realUser` **VARCHAR**(50) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8mb4\_unicode\_ci',  `message` **VARCHAR**(256) **NOT** **NULL** **COLLATE** 'utf8mb4\_unicode\_ci',  `time` **TIMESTAMP** **NOT** **NULL** **DEFAULT** **current\_timestamp**(),  `likes` **INT**(11) **NOT** **NULL** **DEFAULT** '0',  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_twitter\_tweets\_twitter\_accounts` (`authorId`) **USING** **BTREE**,  **CONSTRAINT** `FK\_twitter\_tweets\_twitter\_accounts` **FOREIGN** **KEY** (`authorId`) **REFERENCES** `es\_extended`.`twitter\_accounts` (`id`) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **RESTRICT**  )  **COLLATE**='utf8mb4\_unicode\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=174  ; |
| **CREATE** **TABLE** `companies` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `name` **VARCHAR**(50) **NOT** **NULL** **COMMENT** 'O mesmo da tabela jobs' **COLLATE** 'utf8\_general\_ci',  `label` **VARCHAR**(50) **NOT** **NULL** **COMMENT** 'O que vai aparecer na lista' **COLLATE** 'utf8mb4\_general\_ci',  `job\_grade` **INT**(11) **NOT** **NULL**,  `price` **INT**(11) **NOT** **NULL** **COMMENT** 'Preço da empresa no Estado',  `owned` **TINYINT**(1) **NOT** **NULL** **DEFAULT** '0',  `ownerIdentifier` **VARCHAR**(40) **NULL** **DEFAULT** **NULL** **COMMENT** 'Identifier da tabela users' **COLLATE** 'utf8\_general\_ci',  `last\_update` **DATE** **NOT** **NULL** **DEFAULT** **current\_timestamp**() **COMMENT** 'Ultima atualização da empresa, pode corresponder a data de compra ou venda',  `private` **TINYINT**(1) **NOT** **NULL** **DEFAULT** '0' **COMMENT** 'Se a empresa esta a venda como privada ou do estado',  `private\_price` **INT**(11) **NULL** **DEFAULT** **NULL** **COMMENT** 'Preço da empresa proposta por um jogador',  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_name` (`name`) **USING** **BTREE**,  **INDEX** `FK\_identifier` (`ownerIdentifier`) **USING** **BTREE**,  **CONSTRAINT** `FK\_identifier` **FOREIGN** **KEY** (`ownerIdentifier`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**,  **CONSTRAINT** `FK\_name` **FOREIGN** **KEY** (`name`) **REFERENCES** `es\_extended`.`jobs` (`name`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**  )  **COMMENT**='Empresas que podem ser vendidas no servidor'  **COLLATE**='utf8mb4\_general\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=3  ; |

|  |
| --- |
| **CREATE** **TABLE** `emprestimos` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `owner` **VARCHAR**(40) **NOT** **NULL** **DEFAULT** '' **COLLATE** 'utf8\_general\_ci',  `descricao` **VARCHAR**(25) **NOT** **NULL** **DEFAULT** '0' **COLLATE** 'utf8\_general\_ci',  `juros` **FLOAT** **NOT** **NULL** **DEFAULT** '0',  `value` **DOUBLE** **NOT** **NULL** **DEFAULT** '0',  `prestacao` **DOUBLE** **NOT** **NULL** **DEFAULT** '0',  `prestacoes\_totais` **INT**(11) **NOT** **NULL** **DEFAULT** '0',  `n\_prestacoes` **DOUBLE** **NOT** **NULL** **DEFAULT** '0',  `timeToEnd` **DOUBLE** **NOT** **NULL** **DEFAULT** '0',  `pago` **TINYINT**(4) **NOT** **NULL**,  `createdAt` **DATETIME** **NOT** **NULL** **DEFAULT** **current\_timestamp**(),  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_emprestimo\_owner` (`owner`) **USING** **BTREE**,  **CONSTRAINT** `FK\_emprestimo\_owner` **FOREIGN** **KEY** (`owner`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**  )  **COLLATE**='utf8mb4\_general\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=22  ; |
| **CREATE** **TABLE** `fine\_types` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `label` **VARCHAR**(255) **NULL** **DEFAULT** **NULL** **COLLATE** 'utf8\_general\_ci',  `amount` **INT**(11) **NULL** **DEFAULT** **NULL**,  `category` **INT**(11) **NULL** **DEFAULT** **NULL**,  **PRIMARY** **KEY** (`id`) **USING** **BTREE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=105  ; |

|  |
| --- |
| **CREATE** **TABLE** `handcuffedplayers` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `identifier` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_CuffedID` (`identifier`) **USING** **BTREE**,  **CONSTRAINT** `FK\_CuffedID` **FOREIGN** **KEY** (`identifier`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**  )  **COMMENT**='Jogadores que estão algemados no jogo'  **COLLATE**='utf8mb4\_general\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=6  ; |

|  |
| --- |
| **CREATE** **TABLE** `items` (  `name` **VARCHAR**(50) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `label` **VARCHAR**(50) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `weight` **INT**(11) **NOT** **NULL** **DEFAULT** '1',  `rare` **TINYINT**(1) **NOT** **NULL** **DEFAULT** '0',  `can\_remove` **TINYINT**(1) **NOT** **NULL** **DEFAULT** '1',  **PRIMARY** **KEY** (`name`) **USING** **BTREE**  )  **COLLATE**='utf8\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `shops` (  `shopNumber` **INT**(2) **NOT** **NULL**,  `ownerIdentifier` **VARCHAR**(40) **NOT** **NULL** **COLLATE** 'utf8\_general\_ci',  `last\_update` **DATE** **NULL** **DEFAULT** **current\_timestamp**(),  `shopInventory` **LONGTEXT** **NOT** **NULL** **DEFAULT** '{"snacks":{},"drinks":{},"drugs":{},"readymeal":{},"diverse":{}}\'' **COLLATE** 'utf8mb4\_bin',  **PRIMARY** **KEY** (`shopNumber`) **USING** **BTREE**,  **INDEX** `FK\_IdentifierShops` (`ownerIdentifier`) **USING** **BTREE**,  **CONSTRAINT** `FK\_IdentifierShops` **FOREIGN** **KEY** (`ownerIdentifier`) **REFERENCES** `es\_extended`.`users` (`identifier`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**  )  **COMMENT**='Lojas do mapa com dono'  **COLLATE**='utf8mb4\_general\_ci'  **ENGINE**=**InnoDB**  ; |

|  |
| --- |
| **CREATE** **TABLE** `shop\_sells` (  `id` **INT**(11) **NOT** **NULL** **AUTO\_INCREMENT**,  `shopNumber` **INT**(11) **NOT** **NULL**,  `buyer` **VARCHAR**(50) **NOT** **NULL** **COLLATE** 'utf8mb4\_general\_ci',  `products` **LONGTEXT** **NOT** **NULL** **DEFAULT** '{}' **COLLATE** 'utf8mb4\_bin',  `value` **DOUBLE**(22,0) **NOT** **NULL**,  `date` **DATE** **NOT** **NULL** **DEFAULT** **current\_timestamp**(),  **PRIMARY** **KEY** (`id`) **USING** **BTREE**,  **INDEX** `FK\_ShopSells` (`shopNumber`) **USING** **BTREE**,  **CONSTRAINT** `FK\_ShopSells` **FOREIGN** **KEY** (`shopNumber`) **REFERENCES** `es\_extended`.`shops` (`shopNumber`) **ON** **UPDATE** **CASCADE** **ON** **DELETE** **CASCADE**  )  **COMMENT**='Histórico de vendas da loja visivel para o dono da mesma'  **COLLATE**='utf8mb4\_general\_ci'  **ENGINE**=**InnoDB**  **AUTO\_INCREMENT**=5  ; |

5. Protótipo

**(Desenho abstrato da aplicação na ótica do utilizador)**

****

Figura 1- Página dos empréstimos (fundo transparente)

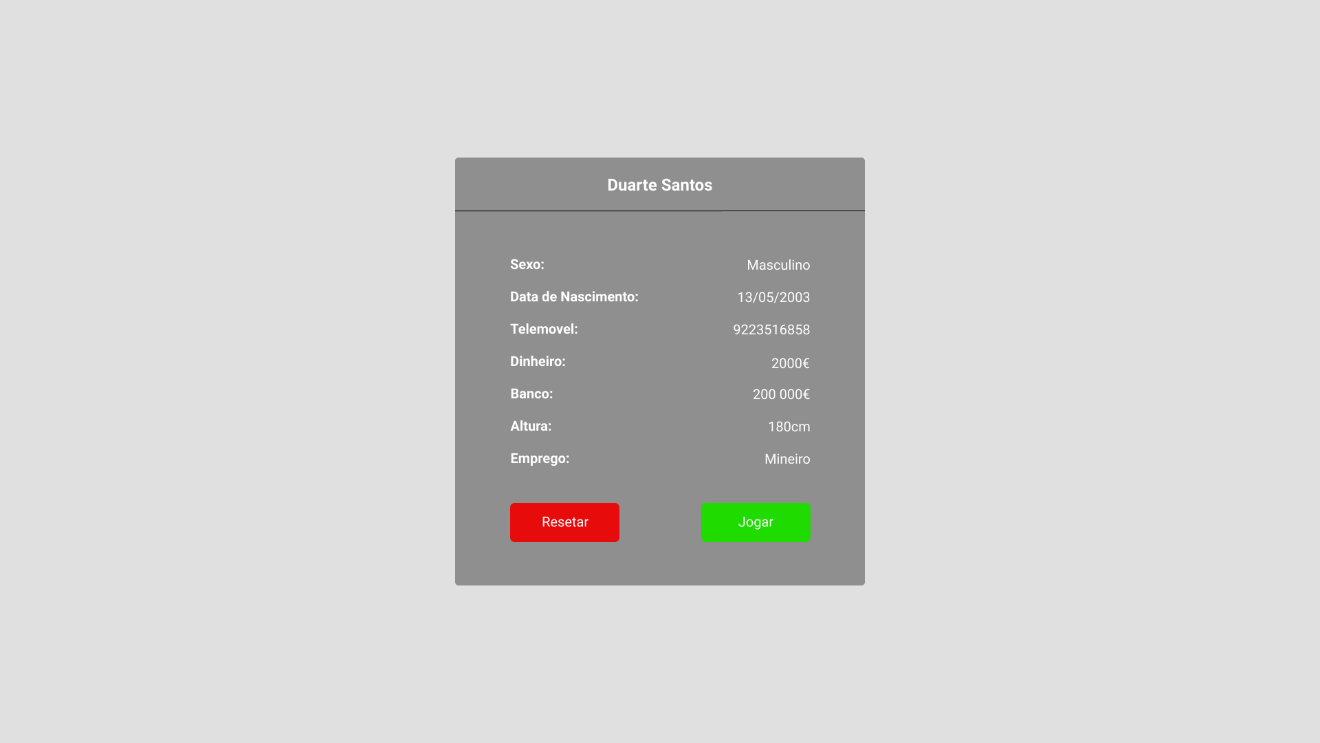


Figura 2-Página de entrada no servidor (fundo transparente)



Figura 3-Página para pedir o reset da personagem (fundo transparente)



Figura 4-Página para escolher um emprego (fundo transparente)

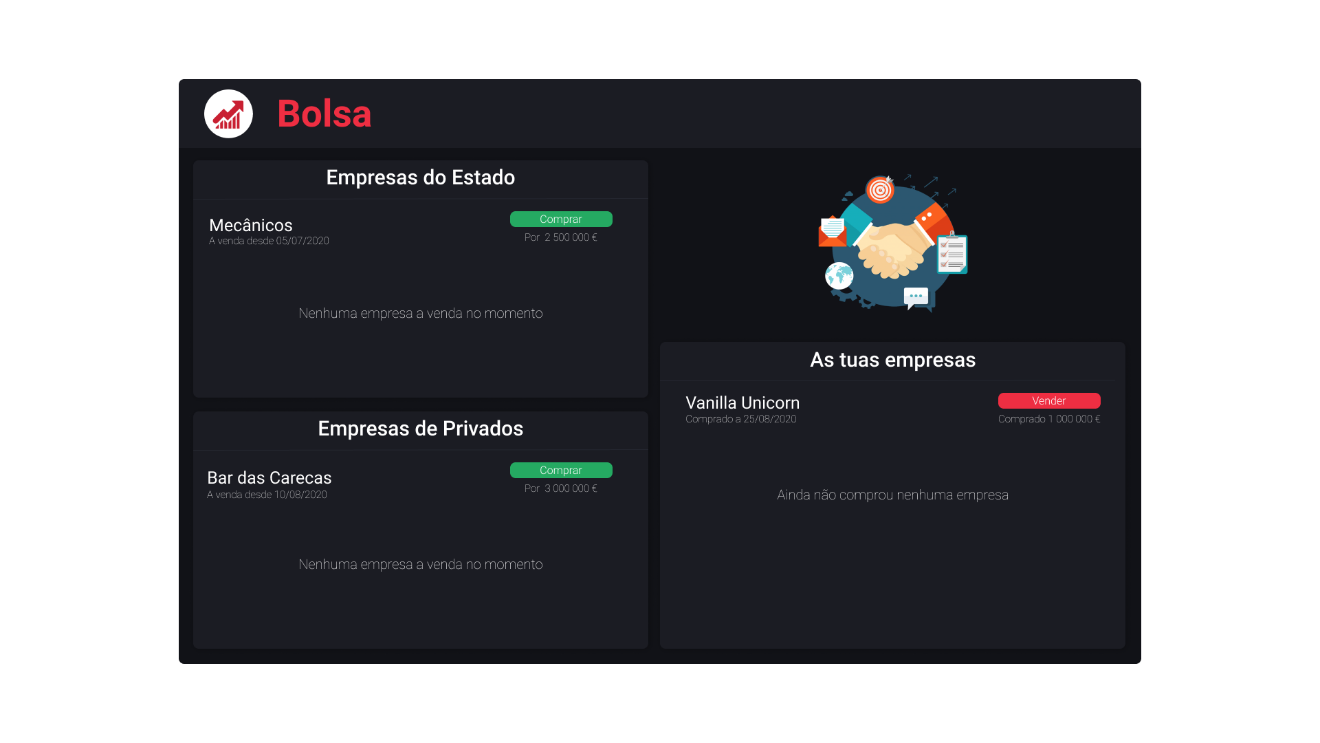


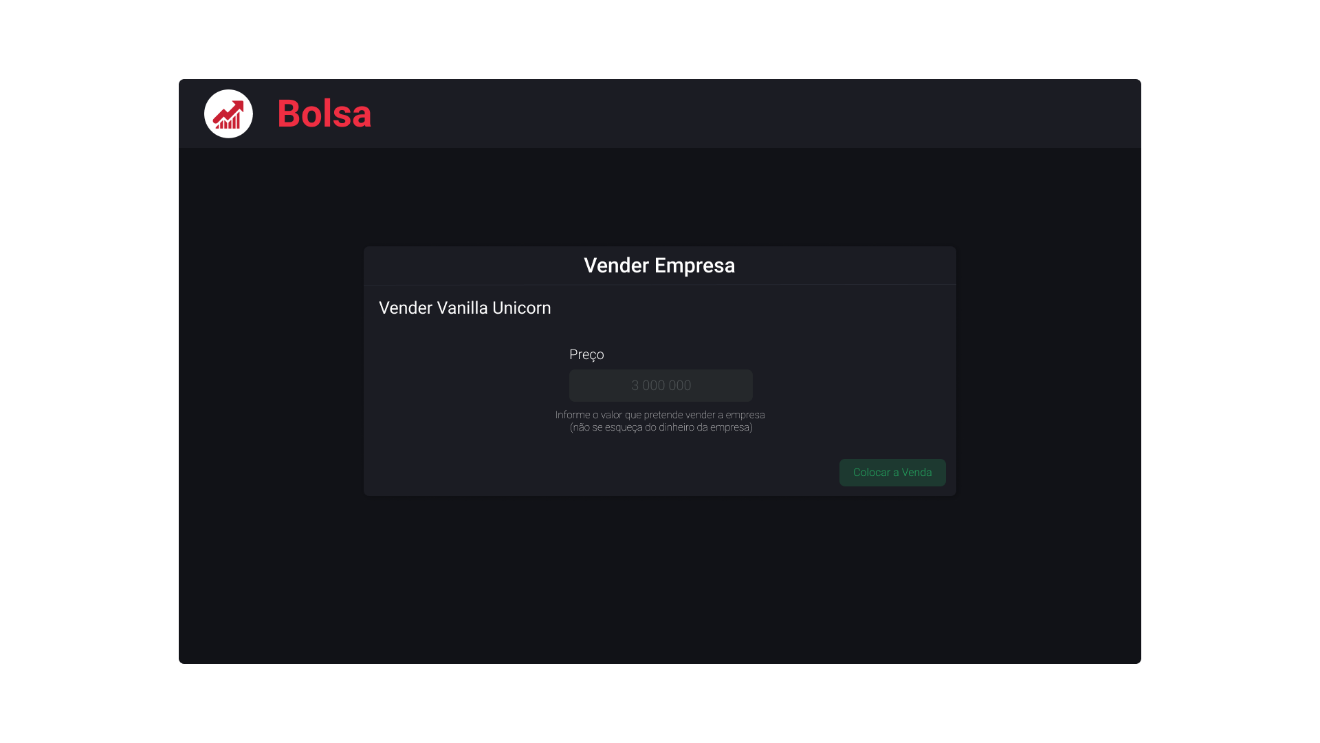
Figura 5-Página inicial Bolsa (Fundo transparente)

Figura 6-Pagina de venda empresas (Fundo transparente)

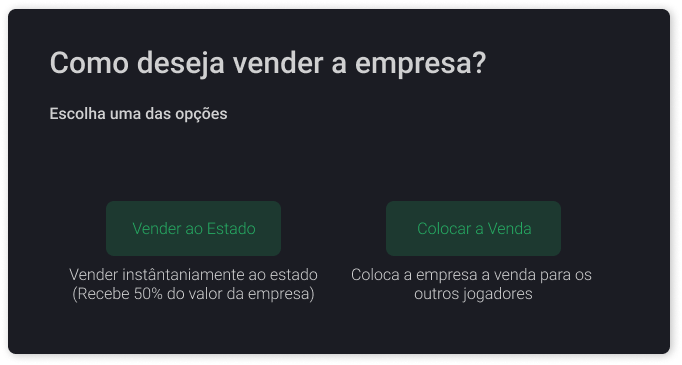


Figura 7-Modal venda empresa

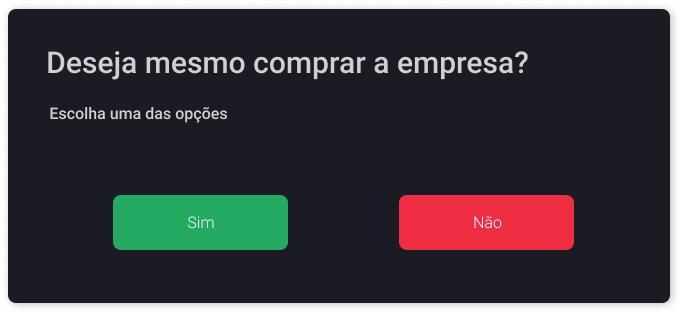


Figura 8-Modal compra empresa

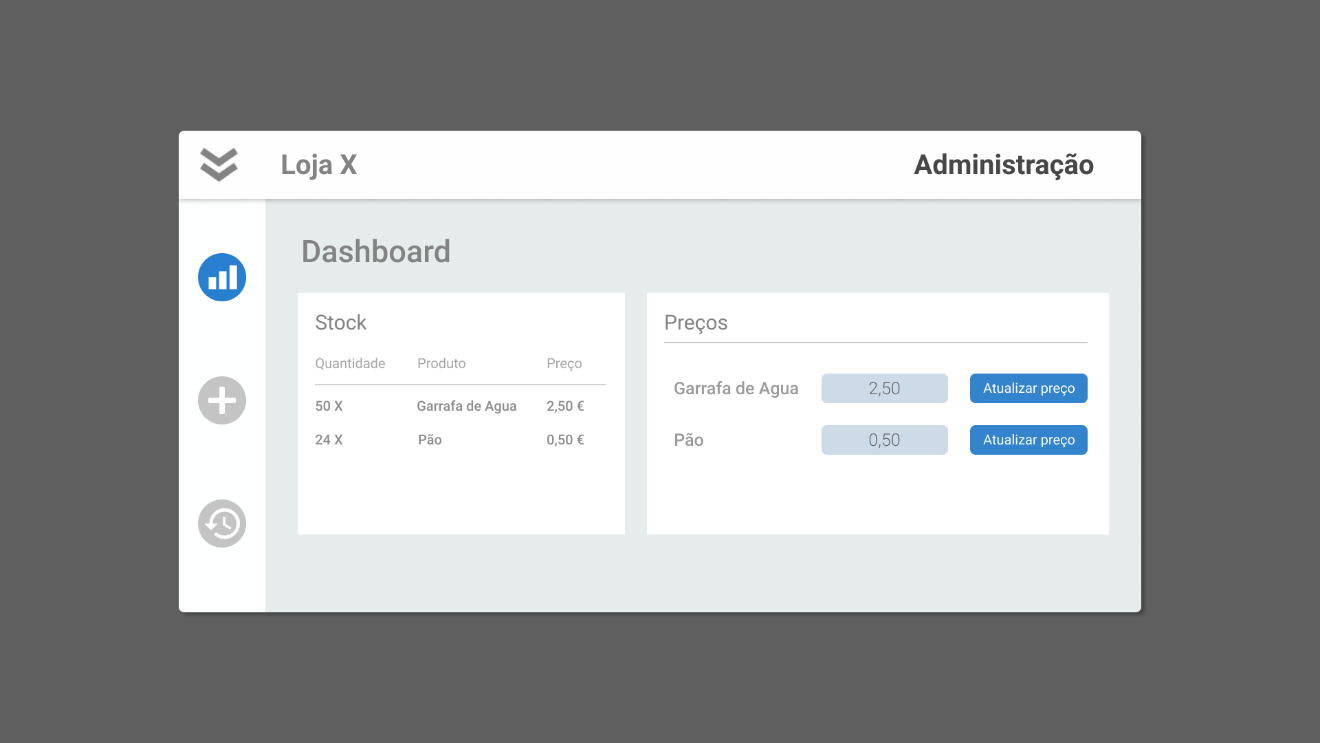


Figura 9-Gestão Stock Lojas (fundo transparente)

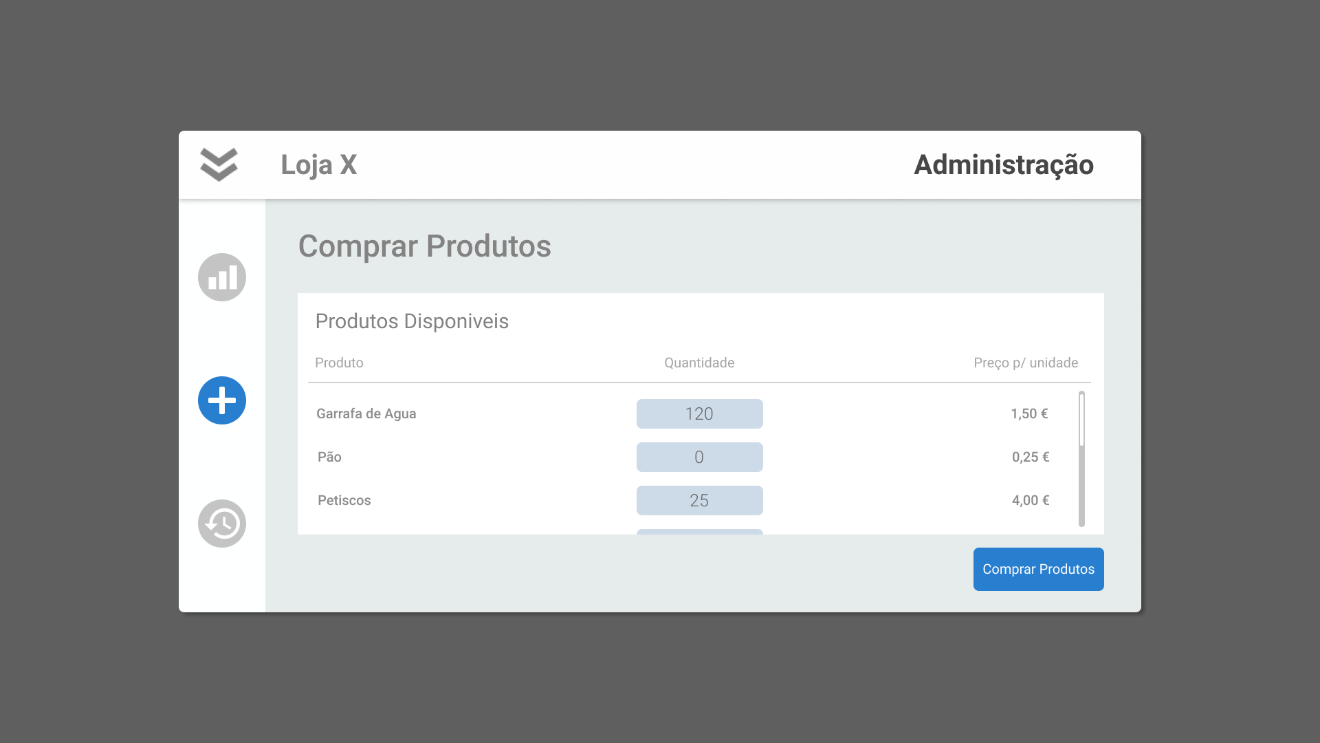


Figura 10-Comprar produtos para vender na loja (fundo transparente)

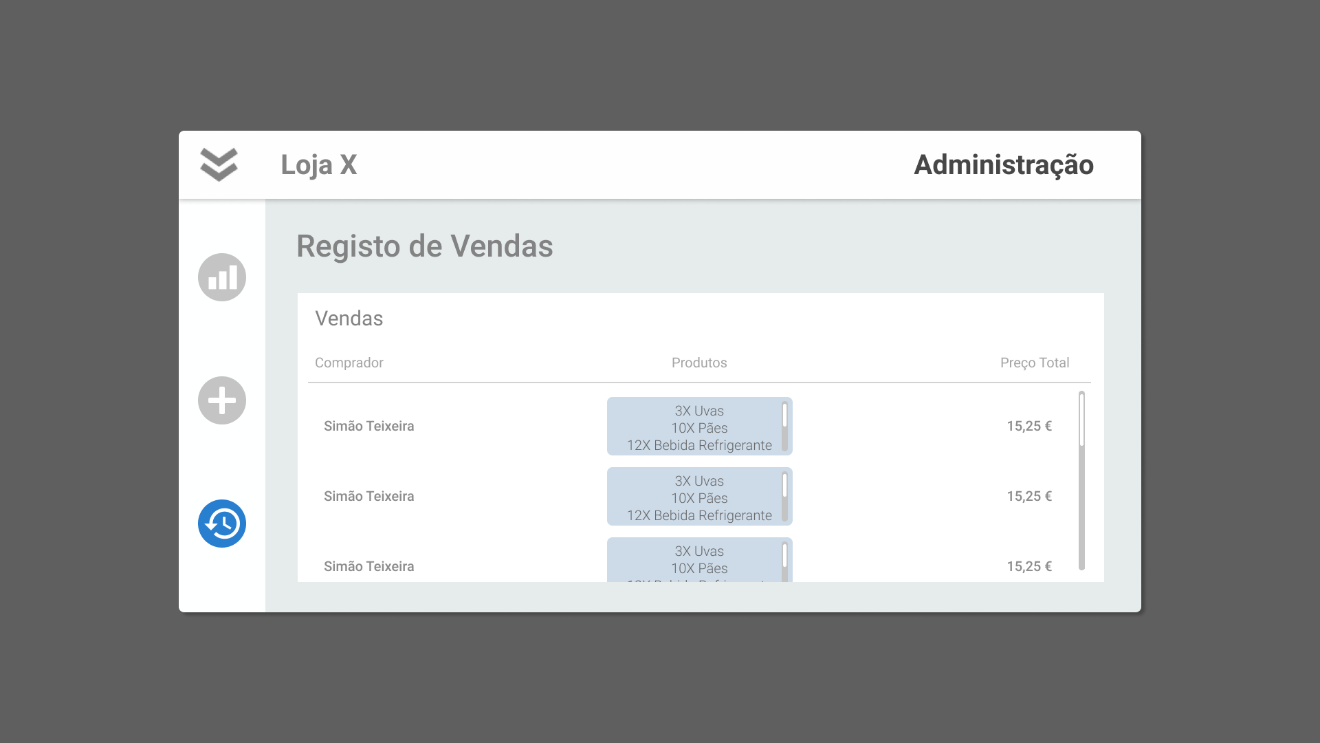


Figura 11-Histórico de vendas da loja (fundo transparente)

6. Análise crítica

**(Levantamento de necessidades)**

Até ao momento conseguimos cumprir todas as tarefas dentro dos prazos pedidos assim como o cronograma planeado no início do ano.

Estamos a utilizar a plataforma [Trello](https://trello.com/) para nos ajudar a aplicar a metodologia Agile através do seguinte modelo:

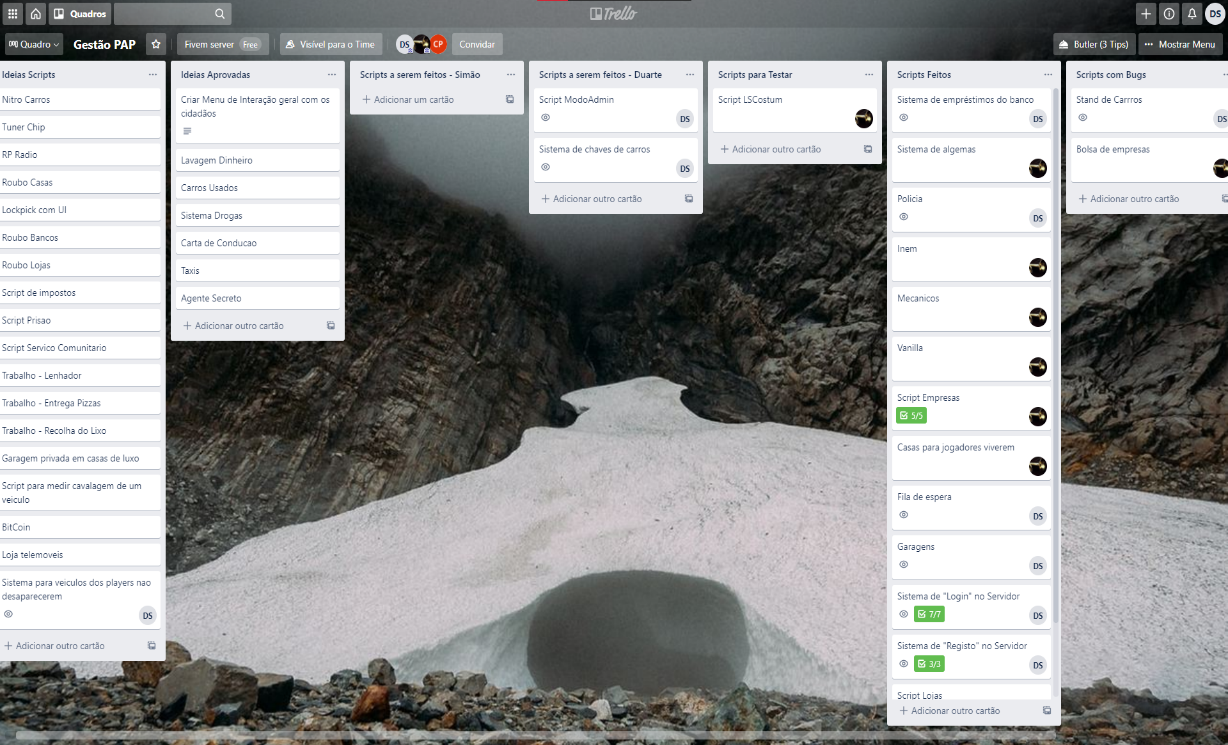


Figura 12-Modelo organização no Trello sobre a metodologia Agile

Alem disso estamos a usar um repositório privado no GitHub para a partilha dos scripts entre nós os dois.

Até agora desenvolvemos alguns scripts, tais como:

* o script que gere a entrada dos jogadores no servidor e o seu registo
* alguns scripts de empregos como Polícia, Médico e Mecânicos
* o script que permite algemar e desalgemar jogadores
* um script para compra e venda de empresas
* um script para simular um stand
* um script para simular a compra e venda de casas
* um script para o inventario.

Durante o desenvolvimento dos scripts temos convidado colegas e amigos para os experimentar juntamente connosco para encontrarmos e resolvermos bugs.

Na base de dados tentámos implementar validações através de ‘*check*s’ nos scripts de criação das tabelas, mas não nos serviram de muito pois ao inserir dados os ‘*checks’* não eram aplicados nos dados, sendo assim, todas as validações são efetuadas no lado do servidor nos respetivos scripts.

7. Fontes Bibliografia | Webgrafia

[FiveM](https://fivem.net/) – Plataforma que dá suporte aos servidores comunitários do GTA V

[Documentação Fivem](https://docs.fivem.net/docs/) – Documentação para usar as livrarias do Fivem

[Documentação ESX Framework](https://esx-framework.github.io/) – Documentação para usar as livrarias da framework ESX

[Fórum FiveM](https://forum.cfx.re/) – Fórum do Fivem onde podemos esclarecer

[FiveM native reference](https://runtime.fivem.net/doc/natives/)  – Documentação para as livrarias do GTA V

[Linguagem de programação LUA](https://www.lua.org/) – Documentação para a linguagem LUA

[Figma](https://www.figma.com/) – Programa usado para desenvolvimento de mockups

[HeidiSQL](https://www.heidisql.com/) – Programa usado para visualizar a Base de Dados e facilitar o acesso à mesma

[ESX Framework (github.com)](https://github.com/esx-framework) – Repositório da framework ESX

[GTA5-Mods](https://www.gta5-mods.com/) – Site usado para fazer download de modelos 3D criados pela comunidade

[Stack Overflow](https://stackoverflow.com/) – Plataforma usada para tirar dúvidas

[Discord](https://discord.com/) – Plataforma usada para comunicação entre jogadores do servidor

[7 great achievements that GTA 5 broke the Guinness record - Somag News](https://www.somagnews.com/7-great-achievements-gta-5-broke-guinness-record/) – Site que mostra os recordes do GTA V

8. Reflexão final

O desenvolvimento do projeto tem estado a correr como planeado e sem grandes dificuldades ou impasses.

A curva de aprendizagem na plataforma foi difícil e complexa pois é usada uma arquitetura muito especifica na criação dos scripts e a linguagem de programação LUA não é orientada a objetos como estamos habituados. Depois de termos entendido como a arquitetura funciona e todos os scripts se interligam o tempo que demorávamos a desenvolver um script passou a ser menor e a resolução de erros também.

Com este projeto, até à data, conseguimos desenvolver várias competências em diversas áreas, temas e tecnologias como por exemplo: programação *multi threading*, base de dados MYSQL, programação com API’s externas, utilização de *jquery* e versionamento de código com o GIT.

No segundo período pretendemos finalizar a maior parte dos scripts que temos em mente desenvolver e abrir o servidor durante algumas semanas ao público para testes com um maior número de pessoas possível.

Se continuarmos assim com este ritmo de trabalho e organização iremos conseguir acabar o projeto a tempo e já com o servidor aberto ao público em junho de 2021.