



**FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA**

Base de Dados(MIEI) – 2017/18

**Relatório de Projeto
Plataforma de LiveStreaming**

Realizado por:

Jorge Pereira – 49771

Tiago Fornelos – 49780

José Leal – 50623

[MIEI]

Grupo 46 (P7)

1 Índice

2	Descrição do Tema -----	2
3	Modelo de Dados -----	3
3.1	Modelo ER-----	3
3.2	Opções de design -----	4
3.3	Esquema Relacional -----	5
3.3.1	Versão 1 -----	5
3.3.2	Versão 2 -----	6
3.4	Criação da Base de Dados -----	7
3.4.1	Criação das Tabelas -----	7
3.4.2	Criação de Sequências-----	11
3.4.3	Criação de Triggers-----	12
3.4.4	Criação de funções -----	18
3.5	Limitações/opções tomadas na implementação da Base de Dados-----	19
4	Descrição da Interface -----	22
5	Manual do Utilizador -----	24
6	Considerações finais-----	34

2 Descrição do Tema

Para o desenvolvimento do projeto, tomámos como inspiração as plataformas de Live Streaming que permitem, a um utilizador (Streamer), partilhar conteúdo de diversos jogos e interagir com o seu público (outros utilizadores que assumem um papel passivo de visualizadores).

Assim, decidimos desenvolver a base de dados da nossa aplicação de streaming GameHub.

Esta permite guardar informação dos Users registados na aplicação, que são identificados por um identificador único. Sobre estes guarda-se ainda o seu e-mail, password, data de nascimento e os canais que este segue. Como dito anteriormente, um User poderá ser identificado como Streamer, podendo realizar transmissões em direto.

Um User poderá ainda optar por pagar uma mensalidade e obter um estatuto Premium, que lhe garante vantagens como o acesso ao Chat da stream, participação nos eventos realizados pelo Streamer, melhor qualidade de imagem, etc. No pagamento desta mensalidade o User poderá optar por qualquer um dos métodos disponíveis no sistema e não é obrigado a pagar todas as mensalidades com o mesmo.

Para aumentar a segurança do User, a aplicação mantém um registo de todos os Logins efetuados.

Cada Stream contém informação dos Users que o seguem, uma lista de reprodução das varias streams realizadas e ainda listas com os melhores momentos das streams de uma dada semana.

Cada Stream terá ainda um chat associado que apenas será utilizado pelos Users Premium. Este Chat será constituído por Mensagens que serão guardadas no sistema.

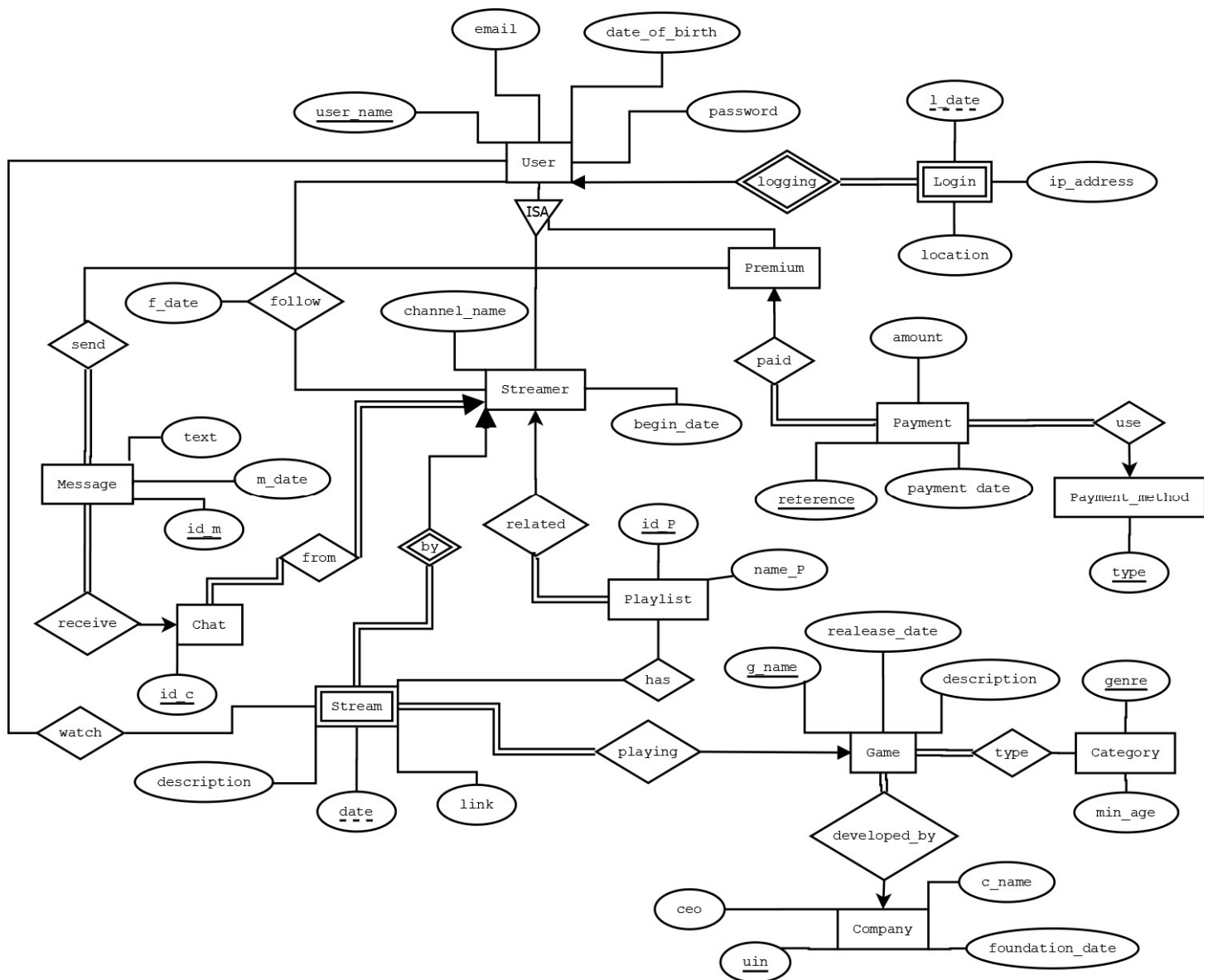
A aplicação guarda informação de cada Game. Estes são identificados univocamente pelo nome, a Company desenvolvedora, as Categories ao qual está associado, uma descrição e a data de lançamento.

Para cada Company, é guardado o seu nome, data de fundação e CEO. Adicionalmente temos também associado à Company o UIN (Unique Identifier Number) que diferencia estas entidades.

Assume-se que, num determinado momento, a transmissão efetuada por um Streamer é composta por apenas um Game, isto é, quando um Streamer muda de Game, é terminada a transmissão anterior e inicia-se uma nova.

3 Modelo de Dados

3.1 Modelo ER



3.2 Opções de design

/*A identidade abstrata foi criada pois as identidades Streamer e Canal são inseparáveis no que diz respeito ao Stream, ou seja, não faz sentido existir um Stream sem Canal e/ou Streamer.*/

Colocámos o Stream com identidade fraca pois pode existir Streams no mesmo dia/hora, mas tem de ser efectuado por diferentes Streamers.

Em relação ao Streamer cada um possui um chat que pode ser utilizado por Utilizadores Premium em que este recebe mensagens dos Premium.

Em relação ao login tivemos de optar por uma identidade fraca porque pode haver logins à mesma data e hora desde que sejam efetuados por utilizadores diferentes e também porque nos permite manter um registo de todos os Logins feitos no sistema.

O tipo de pagamento foi definido como uma entidade em vez de um atributo, pois queríamos evitar redundância e também permitir que os tipos de pagamento só pudessem ser os disponíveis na aplicação.

3.3 Esquema Relacional

3.3.1 Versão 1

Entidades

User(user_name, email, date_of_birth, password)

Channel(channel_id, num_followers)

Streamer(user_name, begin_date, channel_id)

Login(l_date, l_time, ip_adress, location, user_name)

Payment(reference, data, amount, type, user_name)

Payment_method(type)

Stream(channel_id, user_name, date, time, views, desc, link, g_name)

Chat(id_c, channel_id)

Message(id_m, text, m_date, m_time, id_c, user_name)

Playlist(id_p, channel_id)

Game(g_name, release_date, description, uin)

Category(genre, min_age)

Company(uin, ceo, c_name, foundation_date)

Premium(user_name)

Relações

Follow(channel_id, user_name, f_date)

type(genre, g_name)

watch(user_name, channel_id, date, time)

has(id_p, channel_id)

3.3.2 Versão 2

Nesta versão foram eliminadas tabelas de modo a existir mais coerência com o tema do trabalho. E alguns atributos pois podem ser calculados posteriormente através de funções em PL/SQL.

Entidades

Categories(genre, min_age)

Company(uin,ceo,c_name,foundation_date)

Game(g_name,release_date,description,uin)

Users(user_name,email,date_of_birth,password)

Login(l_time,ip_adress,location,user_name)

Premium(user_name)

Streamer(channel_name, user_name, begin_date ,channel_id)

Payment_method(type)

Payment(reference,payment_data,amount,type,user_name)

Playlist(id_p, name_p, streamer_name)

Stream(streamer_name,s_time, views, description, link, g_name)

Chat(id_c,streamer_name)

Message(id_m, text, m_time, id_c, user_name)

Relações

Follow(streamer_name,user_name,f_date)

type(genre,g_name)

watch(user_name,streamer_name,s_time)

has(id_p,streamer_name,s_time)

3.4 Criação da Base de Dados

3.4.1 Criação das Tabelas

Em seguida encontra-se o código referente á criação das tabelas.

--Entidades--

```
CREATE TABLE categories(  
  genre VARCHAR2(15) PRIMARY KEY,  
  min_age NUMBER(2,0)  
);  
  
CREATE TABLE company(  
  uin NUMBER(11) PRIMARY KEY,  
  ceo VARCHAR2(30),  
  c_name VARCHAR2(20),  
  foundation_date DATE  
);  
  
CREATE TABLE game(  
  g_name VARCHAR2(30) PRIMARY KEY,  
  relese_date DATE,  
  description VARCHAR2(150),  
  uin NUMBER(11),  
  FOREIGN KEY(uin) REFERENCES company(uin)  
);  
  
CREATE TABLE users (  
  user_name VARCHAR2(15) PRIMARY KEY,  
  email VARCHAR2(100) UNIQUE NOT NULL ,  
  date_of_birth DATE NOT NULL,  
  password VARCHAR2(255) NOT NULL,  
  CHECK ( REGEXP_LIKE (email, '^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}$'))  
);  
  
CREATE TABLE premium (  
  user_name VARCHAR2(15) PRIMARY KEY,  
  FOREIGN KEY (user_name) REFERENCES users(user_name)  
);
```



```

CREATE TABLE login (
    l_time TIMESTAMP,
    ip_address VARCHAR2(15),
    location VARCHAR2(15),
    user_name VARCHAR2(15),
    PRIMARY KEY (l_time,user_name),
    FOREIGN KEY (user_name) REFERENCES users(user_name),
    CHECK ( REGEXP_LIKE (ip_address,'^([0-9]{1}[0-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]{1}[0-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])$' ))
);

CREATE TABLE streamer (
    user_name VARCHAR2(15) PRIMARY KEY,
    channel_name VARCHAR2(15) UNIQUE NOT NULL,
    begin_date DATE NOT NULL,
    FOREIGN KEY (user_name) REFERENCES users(user_name)
);

CREATE TABLE payment_Method(
    type VARCHAR2(15) PRIMARY KEY
);

CREATE TABLE stream (
    streamer_name VARCHAR2(15) NOT NULL,
    s_time TIMESTAMP NOT NULL,
    g_name VARCHAR2(15) NOT NULL,
    description VARCHAR2(300),
    link VARCHAR2(50) NOT NULL,
    PRIMARY KEY(streamer_name,s_time),
    FOREIGN KEY (streamer_name) REFERENCES streamer(user_name),
    FOREIGN KEY (g_name) REFERENCES game(g_name)
);

CREATE TABLE playlist(
    id_p NUMBER(12,0) PRIMARY KEY,
    name_p VARCHAR2(20) NOT NULL,
    streamer_name VARCHAR2(15) NOT NULL,
    FOREIGN KEY (streamer_name) REFERENCES streamer(user_name)
);

```

```

CREATE TABLE payment(
    reference NUMBER(12,0) PRIMARY KEY,
    payment_date DATE NOT NULL,
    amount NUMBER(5,2),check(amount > 0),
    user_name VARCHAR2(15),
    type VARCHAR2(15),
    FOREIGN KEY (user_name) REFERENCES premium(user_name),
    FOREIGN KEY (type) REFERENCES payment_Method(type)
);

CREATE TABLE chat(
    id_c NUMBER(12,0) PRIMARY KEY,
    streamer_name VARCHAR2(15) NOT NULL,
    FOREIGN KEY (streamer_name) REFERENCES streamer(user_name)
);

CREATE TABLE message(
    id_m NUMBER(12,0) PRIMARY KEY,
    m_time TIMESTAMP NOT NULL,
    id_c NUMBER(12,0) NOT NULL,
    user_name VARCHAR2(15) NOT NULL,
    text VARCHAR(250),
    FOREIGN KEY (id_c) REFERENCES chat(id_c),
    FOREIGN KEY (user_name) REFERENCES premium(user_name)
);

```

--Relações--

```

CREATE TABLE watch(
    user_name VARCHAR2(15) NOT NULL,
    streamer_name VARCHAR2(15) NOT NULL,
    s_time TIMESTAMP NOT NULL,
    PRIMARY KEY (user_name,streamer_name,s_time),
    FOREIGN KEY (user_name) REFERENCES users(user_name),
    FOREIGN KEY (streamer_name,s_time) REFERENCES stream(streamer_name,s_time),
    CHECK (user_name<>streamer_name)
);

```

```

CREATE TABLE has(

```

```

        id_p NUMBER(12,0),
        streamer_name VARCHAR2(15),
        s_time TIMESTAMP,
        PRIMARY KEY (id_p,streamer_name,s_time),
        FOREIGN KEY (id_p) REFERENCES playlist(id_p),
        FOREIGN KEY (streamer_name) REFERENCES streamer(user_name)
    );

CREATE TABLE type(
    g_name VARCHAR2(30),
    genre VARCHAR2(15),
    PRIMARY KEY(g_name,genre),
    FOREIGN KEY (g_name) references game(g_name),
    FOREIGN KEY (genre) references categories(genre)
);

CREATE TABLE follow (
    user_name VARCHAR2(15),
    streamer_name VARCHAR2(15),
    f_date TIMESTAMP NOT NULL,
    PRIMARY KEY (user_name,streamer_name),
    FOREIGN KEY (user_name) REFERENCES users(user_name),
    FOREIGN KEY (streamer_name) REFERENCES streamer(user_name),
    CHECK (user_name<>streamer_name)
);

```

3.4.2 Criação de Sequências

As seguintes sequências são utilizadas para obter os identificadores únicos e incrementais utilizados como chaves primárias nas tabelas company, payment, playlist, chat e message (seq_company, seq_payment, seq_playlist, seq_chat, seq_message respectivamente).

```
DROP SEQUENCE seq_company;  
CREATE SEQUENCE seq_company  
START WITH 1  
INCREMENT BY 1;
```

```
DROP SEQUENCE seq_payment;  
CREATE SEQUENCE seq_payment  
START WITH 1  
INCREMENT BY 1;
```

```
DROP SEQUENCE seq_playlist;  
CREATE SEQUENCE seq_playlist  
START WITH 1  
INCREMENT BY 1;
```

```
DROP SEQUENCE seq_chat;  
CREATE SEQUENCE seq_chat  
START WITH 1  
INCREMENT BY 1;
```

```
DROP SEQUENCE seq_message;  
CREATE SEQUENCE seq_message  
START WITH 1  
INCREMENT BY 1;
```

3.4.3 Criação de Triggers

O trigger ***insert_streamer*** permite fazer a inserção de um streamer na tabela *streamer* sempre que existe uma inserção na *view_streamers*.

```
CREATE OR REPLACE TRIGGER insert_streamer INSTEAD OF INSERT ON view_streamers
FOR EACH ROW
BEGIN
    INSERT INTO streamer VALUES(:new.user_name,:new.channel_name,:new.begin_date);
END;
```

O trigger ***delete_streamer*** permite remover um *streamer* da tabela *streamer* em vez de eliminar da *view_streamers*.

```
CREATE OR REPLACE TRIGGER delete_streamer INSTEAD OF DELETE ON view_streamers
FOR EACH ROW
BEGIN
    DELETE FROM streamer WHERE user_name=:old.user_name;
END;
```

O trigger ***update_streamer*** permite dar update a um *streamer* na tabela *streamer* sempre que existe um update na *view_streamers*

```
CREATE OR REPLACE TRIGGER update_streamer INSTEAD OF UPDATE ON view_streamers
FOR EACH ROW
BEGIN
    UPDATE streamer
    SET
        channel_name=:new.channel_name
    WHERE user_name=:new.user_name;
END;
```

O trigger ***insert_premium*** permite fazer a inserção de um *utilizador premium* na *view_premium*.

```
CREATE OR REPLACE TRIGGER insert_premium INSTEAD OF INSERT ON view_premium
FOR EACH ROW
BEGIN
    INSERT INTO PREMIUM VALUES(:NEW.USER_NAME);
END;
```

O trigger ***delete_premium*** permite remover um *utilizador premium* da *view_premium*.

```
CREATE OR REPLACE TRIGGER delete_premium INSTEAD OF DELETE ON view_premium
FOR EACH ROW
BEGIN
    DELETE FROM premium WHERE user_name=:old.user_name;
END;
```

Nos seguintes quatro triggers assume-se que um utilizador possui um login que expira após um dia (duração 1) pelo que foi necessário criar estes triggers de modo a validar estas 4 operações.

O trigger ***canWatch*** garante que, antes de ocorrer uma inserção na tabela *watch*, o *utilizador* fez login para poder assistir a essa transmissão.

```
CREATE OR REPLACE TRIGGER canWatch BEFORE INSERT ON watch
FOR EACH ROW
DECLARE co NUMBER;last_login TIMESTAMP;
BEGIN
    SELECT count(*) into co
    FROM login
    WHERE user_name=:new.user_name
        SELECT max(l_time) INTO last_login
        FROM login
        WHERE user_name=:new.user_name;
    IF co = 0 or ((TRUNC(:NEW.s_time) - TRUNC(last_login)) >= 1 or (TRUNC(:NEW.s_time) - TRUNC(last_login)) <0)
    THEN Raise_Application_Error (-20100, 'User never logged in or login expired');
    END IF;
END;
```

O trigger **canFollow** garante que, antes de existir uma inserção na tabela *follow*, o *utilizador* tem sessão iniciada para poder seguir um *streamer*.

```
CREATE OR REPLACE TRIGGER canFollow BEFORE INSERT ON follow
FOR EACH ROW
DECLARE co NUMBER;last_login TIMESTAMP;
BEGIN
    SELECT count(*) INTO co
    FROM login
    WHERE user_name=:new.user_name
           select max(l_time) into last_login
    FROM login
    WHERE user_name=:new.user_name;
    IF co = 0 or ((TRUNC(:new.f_date) - TRUNC(last_login)) >= 1 or (TRUNC(:new.f_date) - TRUNC(last_login)) <0)
    THEN Raise_Application_Error (-20100, 'User never logged in or login expired');
    END IF;
END;
```

O trigger **canPay** garante que, antes de existir uma inserção na tabela *payment*, o *utilizador* tem secção iniciada para poder seguir um pagar a mensalidade de modo a obter o estatuto de utilizador *premium*.

```
CREATE OR REPLACE TRIGGER canPay BEFORE INSERT ON payment
FOR EACH ROW
DECLARE co NUMBER;last_login TIMESTAMP;
BEGIN
    SELECT count(*) INTO co
    FROM login
    WHERE user_name=:NEW.user_name;
           SELECT max(l_time) INTO last_login
    FROM login
    WHERE user_name=:NEW.user_name;
    IF co = 0 or ((TRUNC(:NEW.payment_date) - TRUNC(last_login)) >= 1 or (TRUNC(:NEW.payment_date) - TRUNC(last_login)) <0)
    THEN Raise_Application_Error (-20100, 'User never logged in or login expired');
    END IF;
END;
```

O trigger **canSend** garante que, antes de existir uma inserção na tabela *message*, o utilizador tem sessão iniciada para poder enviar uma mensagem no *chat* de uma *stream*.

```
CREATE OR REPLACE TRIGGER canSend BEFORE INSERT ON message
FOR EACH ROW
DECLARE co NUMBER;last_login TIMESTAMP;
BEGIN
    SELECT count(*) INTO co
    FROM login
    WHERE user_name=:new.user_name
        SELECT max(l_time) INTO last_login
        FROM login
        WHERE user_name=:new.user_name;
    IF co = 0 or ((TRUNC(:new.m_time) - TRUNC(last_login)) >= 1 or (TRUNC(:new.m_time) - TRUNC(last_login)) <0)
    THEN Raise_Application_Error (-20100, 'User never logged in or login expired');
    END IF;
END;
```

O trigger **atLeastOneCategory**, após a inserção de um novo jogo, cria uma categoria default, caso não exista, para que possa ser atribuída ao jogo.

```
CREATE OR REPLACE TRIGGER atLeastOneCategory AFTER INSERT ON game
FOR EACH ROW
DECLARE numb NUMBER;
BEGIN
    SELECT count(unique genre) INTO numb
    FROM categories
    WHERE genre = 'DEFAULT';
    IF numb = 0
    THEN INSERT INTO CATEGORIES VALUES ('DEFAULT','0');
    END IF;
    INSERT INTO type VALUES (:new.g_name,'DEFAULT');
END;
```


O trigger **delDefault** elimina a categoria default de um jogo sempre que é associada uma nova categoria a um jogo (ie. Sempre que existe uma inserção na tabela type).

```
CREATE OR REPLACE TRIGGER delDefault BEFORE INSERT ON type
FOR EACH ROW
DECLARE numb NUMBER;
BEGIN
    SELECT count(*) INTO numb
    FROM type
    where g_name= :new.g_name ;
    IF numb = 1
    THEN DELETE FROM type WHERE g_name = :new.g_name and genre = 'DEFAULT';
    END IF;
END;
```

O trigger **firstPremium** verifica se foi a primeira vez que um utilizador pagou a mensalidade, passando assim a *premium*. Se sim, é inserido na tabela dos utilizadores *premium*.

```
CREATE OR REPLACE TRIGGER firstPremium BEFORE INSERT ON payment
FOR EACH ROW
DECLARE id NUMBER;
BEGIN
    SELECT COUNT(user_name) INTO id
    FROM premium
    WHERE user_name = :new.user_name;
    IF id = 0
    THEN INSERT INTO premium VALUES (:new.user_name);
    END IF;
END;
```

O trigger ***checkAge*** verifica se a data introduzida pelo utilizador é válida (isto é, se o utilizador tem no mínimo 13 e no máximo 125, sendo assim uma idade plausível).

```
CREATE OR REPLACE TRIGGER checkAge AFTER INSERT ON users
FOR EACH ROW
BEGIN
    IF ((floor(months_between(SYSDATE, :NEW.date_of_birth) /12)) NOT BETWEEN 13 AND 125)
    THEN Raise_Application_Error (-20100, 'Error: Please Check birthdate ');
    END IF;
END;
```

O trigger ***createChat*** atribui um chat a cada novo *streamer* introduzido no sistema.

```
CREATE OR REPLACE TRIGGER createChat AFTER INSERT ON Streamer
FOR EACH ROW
BEGIN
    INSERT INTO chat VALUES(seq_chat.NEXTVAL, :NEW.user_name);
END;
```

3.4.4 Criação de funções

A função *calc_subs* recebe como argumento o nome do streamer e permite obter o número de followers desse streamer.

```
CREATE OR REPLACE FUNCTION calc_subs(streamer VARCHAR2)
  RETURN NUMBER IS totalsubs NUMBER;
BEGIN
  SELECT COUNT(user_name) INTO totalsubs
  FROM follow f
  WHERE f.streamer_name = streamer;
  RETURN totalsubs;
END calc_subs;
/
```

3.5 Limitações/opções tomadas na implementação da Base de Dados

Foi necessário criar algumas restrições em cada tabela na implementação da base de dados para garantir um bom funcionamento.

Na tabela **categories** declarámos *min_age* como **NOT NULL**, pois são campos obrigatórios. A **chave primária** é *genre*.

Na tabela **company** declarámos *ceo*, *c_name* e *foundation_date* como **NOT NULL** por serem campos obrigatórios. *Uin* é **chave primaria** e **chave estrangeira** que refere *company(uin)*.

Na tabela **game** *release_date* e *uin* como **NOT NULL** por serem campos obrigatórios. A **chave primária** é *g_name*. *Uin* é **chave estrangeira** que refere *company(uin)*.

Na tabela **users** declarámos *email*, *date_of_birth* e *password* como **NOT NULL** por serem campos obrigatórios. Foi usado ainda uma **REGEX** para garantir que o *email* introduzido tinha a forma correta. A **chave primária** é *user_name*.

Na tabela **login** declarámos *ip_adress* e *location*, como **NOT NULL** por serem campos obrigatórios. Usou-se uma **REGEX** para garantir que o *ip_adress* tinha a forma correta. A **chave primária** é (*l_time*, *user_name*). *User_name* é **chave estrangeira** que refere *users(user_name)*.

Na tabela **premium** não foram necessárias restrições pois apenas contém o atributo *user_name*(**chave primaria**) que refere os utilizadores da tabela *users*.

Na tabela **payment_method** não foi necessário criar restrições pois esta tabela apenas contém o atributo *type* que define os vários tipos de pagamento que poderão ser utilizados.

Na tabela **stream** declarámos *g_name* e *link* como **NOT NULL** por serem campos obrigatórios. A **chave primária** é (*streamer_name*, *s_time*), sendo que *streamer_name* é **chave estrangeira** que refere *streamer(user_name)*. Contém ainda *g_name* como **chave estrangeira** que refere *game(g_name)*.

Na tabela **streamer** declaramos *channel_name* e *begin_date* como **NOT NULL** por serem campos obrigatórios. Foi ainda preciso declarar o atributo *channel_name* como **UNIQUE** garantido que o nome do canal de cada *streamer* é único. *User_name* é **chave primária** e **chave estrangeira** que refere *users(user_name)*.

Na tabela **payment** declaramos *date*, *user_name*, *type* como **NOT NULL** por serem campos obrigatórios. Foi preciso usar a constraint **Check** para garantir que o *amount* introduzido seria maior que zero. A **chave primária** é *reference*. *User_name* é **chave estrangeira** que refere *premium(user_name)*. A tabela contém ainda o atributo *type* que é **chave estrangeira** que refere *payment_method(type)*.

Na tabela **playlist** declaramos *name_p* e *streamer_name* como **NOT NULL** por serem campos obrigatórios. A **chave primária** é *id_p*. *Streamer_name* é **chave estrangeira** que refere *streamer(user_name)*.

Na tabela **chat** declaramos *streamer_name* como **NOT NULL** por serem campos obrigatórios. A **chave primária** é *id_c*. *Streamer_name* é **chave estrangeira** que refere *streamer(user_name)*.

Na tabela **message** declaramos *m_time*, *id_c* e *user_name* como **NOT NULL** por serem campos obrigatórios. A **chave primária** é *id_m*. *Id_c* é uma **chave estrangeira** que refere *chat(id_c)*. A tabela contém ainda o atributo *user_name* que é **chave estrangeira** referindo *premium(user_name)*.

Na tabela **watch** não foi necessário declarar os atributos como **NOTNULL** mas foi necessário garantir que o *user_name* era diferente do *streamer_name*, garantindo assim que um **streamer** não vê a própria stream enquanto a realiza. A **chave primária** é (*user_name*, *streamer_name*, *s_time*). *User_name* é uma **chave estrangeira** que refere *users(user_name)*. (*Streamer_name*, *s_time*) é **chave estrangeira** que refere a *stream*.

Na tabela **has** não foi utilizado qualquer tipo de restrições. A **chave primária** é (*id_p*, *streamer_name*, *s_time*). (*Streamer_name*, *s_time*) é **chave estrangeira** que refere a *stream*. A tabela contém ainda a **chave estrangeira** *id_p* referindo *playlist(id_p)*.

Na tabela **type** não foi necessário qualquer tipo de restrições. A **chave primária** é (*g_name*, *genre*). *G_name* é uma **chave estrangeira** que refere *game(g_name)*. *Genre* é **chave estrangeira** que refere *categories(genre)*.

Na tabela ***follow*** declaramos *f_date* como **NOT NULL** por ser um campo obrigatório. Foi necessário usar a constraint **Check** para garantir que *user_name* é diferente de *streamer_name*, garantindo assim que um *streamer* não pode dar *follow* em si mesmo. A **chave primária** é (*user_name*, *streamer_name*). *User_name* é **chave estrangeira** que refere *users(user_name)*. A tabela contém ainda a **chave estrangeira** *streamer_name* que refere *streamer(user_name)*.

4 Descrição da Interface

Ao abrir a nossa interface, encontramos uma página inicial onde é possível aceder às seguintes subpáginas: Chat, Message, Playlist, Users, Company, Games e Stream.

O link para a página chat, redireciona-nos para uma página com a listagem de todos os chat existentes na base de dados.

Na página inicial, temos também acesso à página “message” que contém uma listagem de todas as mensagens enviadas pelos utilizadores premium aos streamers. Existe também um botão create para adicionar novas mensagens ao sistema.

A página inicial contém também um link para a página playlist, que contém uma listagem de todas as playlists de cada canal. Analogamente às outras páginas, existe também um botão create para adicionar novas playlists.

Na página Users podemos consultar todos os utilizadores da base de dados e respetivos atributos, como user_name, email, data de nascimento e password.

No canto inferior direito da página, existem dois botões que nos dão acesso às listas dos utilizadores premium e dos streamers.

No canto superior direito existe um botão “create” que permite adicionar novos utilizadores ao sistema.

Na página dos utilizadores premium existe um botão para adicionar um novo pagamento. Sempre que um utilizador efetua um pagamento pela primeira vez, passa a ser utilizador premium.

Na página dos streamers, podemos adicionar novos streamer e também novos seguidores a cada streamer, através dos botões existentes. Ao clicar no user_name de cada streamer, somos redirecionados para uma página onde obtemos a informação dele, como o número de seguidores, email, password e data de nascimento.

O link company, redireciona-nos para uma página que contém todas as companies existentes na base de dados. Podemos adicionar novas companies através do botão create.

O link game, redireciona-nos para uma página que contém todas os atributos dos jogos existentes na base de dados. Podemos adicionar novos jogos através do botão create.

O link stream, redireciona-nos para uma página que contém todas as streams existentes na base de dados. Podemos adicionar novas streams e alterar os seus dados através do botão create. Este abrirá a página master-detail que nos permite também adicionar novos dados á tabela whatch (ie. Adicionar novos users que visualizaram determinada stream).

5 Manual do Utilizador

Na página inicial encontram-se ligações para as subpáginas da base de dados. Outras vão aparecendo à medida que se vai navegando no menu.



Ao abrir a ligação Users, obtemos os dados dos utilizadores registados no sistema.

The screenshot shows a web interface for managing users. At the top, there is a search bar with a magnifying glass icon, a 'Go' button, an 'Actions' dropdown menu, and a 'Create' button. Below this is a table with the following columns: User Name, Email, Date Of Birth, and Password. The table contains 12 rows of user data. At the bottom right of the table, there is a pagination indicator '1 - 13' and two buttons: 'Go To Premium Users' and 'Go To Streamer User'.

User Name	Email	Date Of Birth	Password
Doloe3SaasssDrs	asesaassd@hotmail.com	31-JAN-00	OLA_ADEUS
Pessoa0	Pessoa0@hotmail.com	31-JAN-00	e3w4sizg
Pessoa1	Pessoa1@hotmail.com	31-JAN-00	b3cpsnhi
Pessoa2	Pessoa2@hotmail.com	31-JAN-00	8ta5k357
Pessoa3	Pessoa3@hotmail.com	31-JAN-00	hssnbnd
Pessoa4	Pessoa4@hotmail.com	31-JAN-00	fk72d8lj
Pessoa5	Pessoa5@hotmail.com	31-JAN-00	plmkyb0e
Pessoa6	Pessoa6@hotmail.com	31-JAN-00	dkkwyzws
Pessoa7	Pessoa7@hotmail.com	31-JAN-00	vhkx672t
Pessoa8	Pessoa8@hotmail.com	31-JAN-00	chjtwpr
Pessoa9	Pessoa9@hotmail.com	31-JAN-00	7m8v9oeb
Pessoa10	Pessoa10@hotmail.com	31-JAN-00	3an1jtpi
Pessoa11	Pessoa11@hotmail.com	31-JAN-00	2homcjm8

No topo direito existe um botão create, que, ao ser clicado, abre uma nova página onde podemos adicionar um novo utilizador ao Sistema.













The screenshot shows a web form titled 'Form on USERS'. It has a 'Cancel' button and a 'Create' button at the top right. The form contains four input fields: 'User Name', '*Email', '*Date Of Birth', and '*Password'. The 'Date Of Birth' field has a calendar icon next to it.

No canto inferior direito existem dois botões que nos permite obter a listagem dos utilizadores premium/streamer.

1 - 13

[Go To Premium Users](#) [Go To Streamer User](#)

Em seguida encontra-se a página dos premium. Quando se quer adicionar um novo premium, é necessário adicionar um novo pagamento. Assim, sempre que é adicionado um novo pagamento de um dado utilizador, este passa a ser premium.

	<input type="text"/>	<input type="button" value="Go"/>	<input type="button" value="Actions"/>
User Name	Email	Date Of Birth	Password
 Pessoa0	Pessoa0@hotmail.com	31-JAN-00	e3w4sizg
 Pessoa1	Pessoa1@hotmail.com	31-JAN-00	b3cpsnhi
 Pessoa2	Pessoa2@hotmail.com	31-JAN-00	8ta5k357
 Pessoa3	Pessoa3@hotmail.com	31-JAN-00	hssnbnd
 Pessoa4	Pessoa4@hotmail.com	31-JAN-00	fk72d8lj
 Pessoa5	Pessoa5@hotmail.com	31-JAN-00	plmkyb0e
 Pessoa6	Pessoa6@hotmail.com	31-JAN-00	dkkwyzws
 Pessoa7	Pessoa7@hotmail.com	31-JAN-00	vhkx672t
 Pessoa8	Pessoa8@hotmail.com	31-JAN-00	chjtwpr
 Pessoa9	Pessoa9@hotmail.com	31-JAN-00	7m8v9oeb
 Pessoa10	Pessoa10@hotmail.com	31-JAN-00	3an1jtpi





1 - 11

Nesta página, adicionamos um pagamento em si. Basta preencher os campos necessários e seleccionar um dos tipos de pagamento disponível no sistema. Notasse que é necessário fazer o login no próprio dia antes de pagar.

Form on PAYMENT		<input type="button" value="Cancel"/>	<input type="button" value="Create"/>
*Payment Date	<input type="text"/>		
Amount	<input type="text"/>		
*User Name	<input type="text" value="Doloe3SaasssDrs"/>		
*Type	<input type="text" value="Apple Pay"/>		

Nesta página podemos visualizar todos os atributos adicionais ao streamer e ainda quantos subscribers o streamer tem, esta última coluna é calculada através de uma função em PL/SQL.

No canto superior direito existe o botão create que permite adicionar um novo streamer.

	<input type="text"/>	<input type="button" value="Go"/>	<input type="button" value="Actions ▼"/>	<input type="button" value="Create"/>
	Begin Date	Username	Channel Id	Numb Of Subscribers
	24-MAY-18	Pessoa10	azeite	1
	24-MAY-18	Pessoa7	+azeite	2
	24-MAY-18	Pessoa2	twitch	0

1 - 3

Aqui podemos escolher um dos user do sistema que queremos adicionar como streamer.

Form on STREAMER

User Name

*Channel Name

*Begin Date

Ao clicar no nome do streamer, somos redirecionados para uma página que mais uma vez mostra o número de subscritores e informação mais detalhada acerca deles.

Streamer

Número de Subs

Numero de Subs 2

1 - 1



	User Name	Email	Date Of Birth	Password
	Pessoa10	Pessoa10@hotmail.com	31-JAN-00	3an1jtpi
	Pessoa5	Pessoa5@hotmail.com	31-JAN-00	plmkyb0e

1 - 2

Caso cliquemos no botão “Add Subscriber to Streamer” é nos mostrada a seguinte form que como o nome do botão indica permite adicionar um novo follower ao Streamer. Notasse que é necessário fazer o login no próprio dia para ser possível seguir um Streamer.

Form on FOLLOW

Cancel Create

User Name

Doloe3SaasssDrs

*Streamer Name

Pessoa10

*F Date

Ao clicarmos em games no menu inicial, podemos ver a lista de todos os jogos presentes no sistema.

Go

Actions

Create

G Name	Realease Date	Description	C Name
Metroid	01-FEB-86	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	Accolade
Mega Man	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	StarControl
Tetris	23-FEB-86	olaaaaaaaaaaaaasdasdasdaasssssdasdasdasdasdasdasdas	StarControl
Super Mario Bros	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	TestDrive
Sonic the Hedgehog	08-SEP-91	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	AccessGames
The Legend of Zelda	07-JUL-88	olaaaaaaaaaaaaasdadlfdskdfdasdas	Pandora
Street Fighter II	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	ACETeam
DOOM	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	AcesStudio
Star Wars: TIE Fighter	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	Acheron
Super Metroid	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	Activision
Final Fantasy VI	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	Crash
Castlevania	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	Spyro
Grim Fandango	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	Tony
TheLegend	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	GuitarHero
Grim	03-FEB-88	olaaaaaaaaaaaaasdadlmcfdmckdfdasdasdas	Skylanders

1 - 15 ▶

a

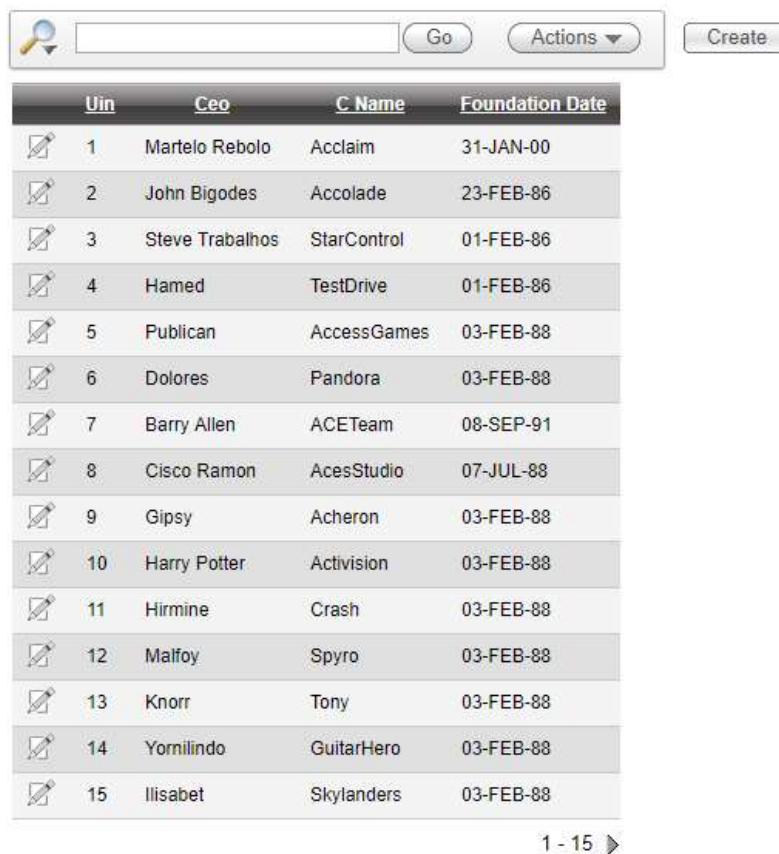
Ao carregar no botão create somos redirecionados para este form que nos permite adicionar um novo jogo ao sistema.



The form is titled "Form on GAME" and has a "Cancel" button and a "Create" button. It contains the following fields:

- G Name**: A text input field.
- Realease Date**: A date picker field.
- Description**: A text input field.
- Uin**: A dropdown menu with "1 Acclaim" selected.

Ao clicarmos em companies no menu inicial, podemos ver a lista de todas as companies presentes no sistema.

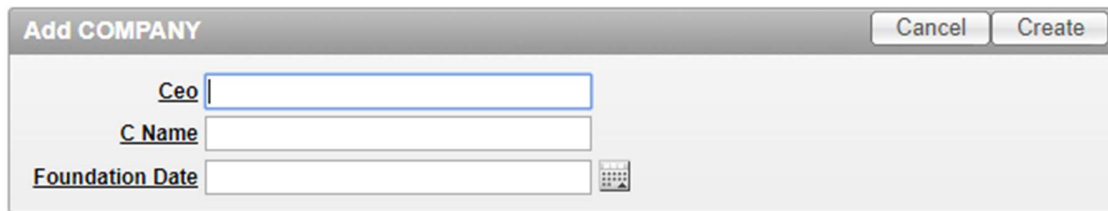


The table displays a list of companies with columns: Uin, Ceo, C Name, and Foundation Date. Each row has a delete icon (scissors) to the left of the Uin. The table is followed by a pagination link "1 - 15 »".

	Uin	Ceo	C Name	Foundation Date
	1	Martelo Rebolo	Acclaim	31-JAN-00
	2	John Bigodes	Accolade	23-FEB-86
	3	Steve Trabalhos	StarControl	01-FEB-86
	4	Hamed	TestDrive	01-FEB-86
	5	Publican	AccessGames	03-FEB-88
	6	Dolores	Pandora	03-FEB-88
	7	Barry Allen	ACETeam	08-SEP-91
	8	Cisco Ramon	AcesStudio	07-JUL-88
	9	Gipsy	Acheron	03-FEB-88
	10	Harry Potter	Activision	03-FEB-88
	11	Hirmine	Crash	03-FEB-88
	12	Malfoy	Spyro	03-FEB-88
	13	Knorr	Tony	03-FEB-88
	14	Yornilindo	GuitarHero	03-FEB-88
	15	Ilisabet	Skylanders	03-FEB-88

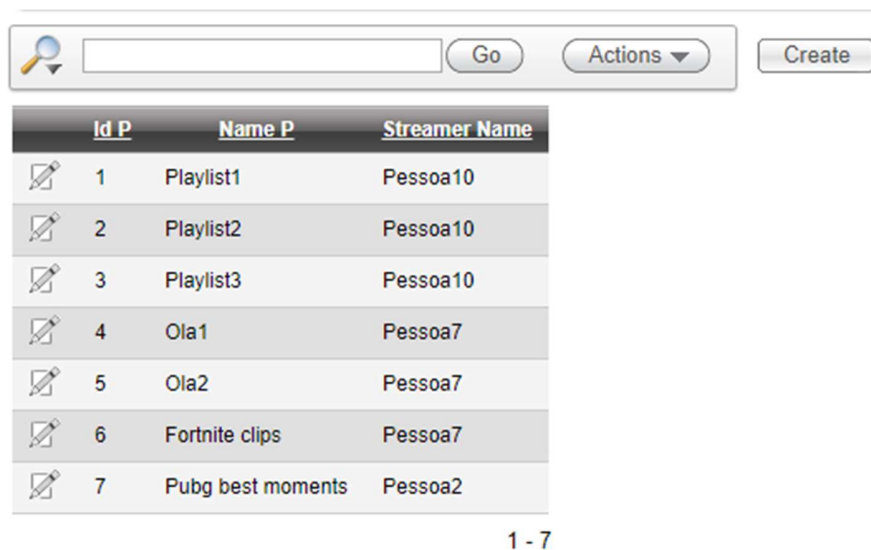
1 - 15 »

Ao carregar no botão create somos redirecionados para este form que nos permite adicionar um novo uma company ao sistema.



The 'Add COMPANY' form contains three input fields: 'Ceo', 'C Name', and 'Foundation Date'. The 'Foundation Date' field includes a calendar icon. At the top right, there are 'Cancel' and 'Create' buttons.

Se no menu iniciar carregarmos em playlist, somos redirecionados para a listagem de todas as playlists.

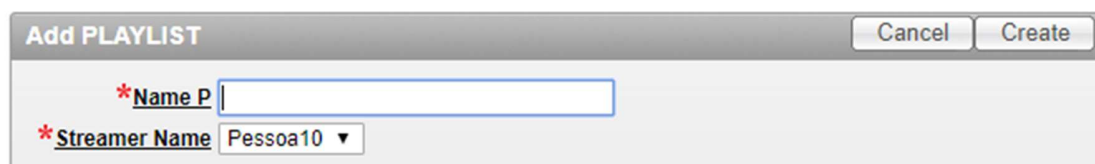


The interface shows a search bar with a magnifying glass icon, a 'Go' button, an 'Actions' dropdown menu, and a 'Create' button. Below is a table of playlists with edit icons in the first column.

	<u>Id P</u>	<u>Name P</u>	<u>Streamer Name</u>
	1	Playlist1	Pessoa10
	2	Playlist2	Pessoa10
	3	Playlist3	Pessoa10
	4	Ola1	Pessoa7
	5	Ola2	Pessoa7
	6	Fortnite clips	Pessoa7
	7	Pubg best moments	Pessoa2





1 - 7

Se carregarmos no botão create, somos redirecionamos para uma form que nos permite adicionar mais uma playlist a um determinado streamer.



The 'Add PLAYLIST' form has two fields: '*Name P' (text input) and '*Streamer Name' (dropdown menu currently showing 'Pessoa10'). 'Cancel' and 'Create' buttons are at the top right.

Se no menu iniciar carregarmos em chat somos redirecionados para a listagem de todos os chats abertos.

	<input type="text"/>	Go	Actions ▼	Create
Id C	Streamer Name			
	1	Pessoa10		
	2	Pessoa7		
	3	Pessoa2		

1 - 3

Se carregarmos no botão create, somos redirecionados para uma form que nos permite adicionar mais um chat a um streamer.

Add CHAT
Cancel
Create

*Streamer Name
Pessoa10 ▼

Ao clicar em message, é possível ver todas as mensagens enviadas pelos users aos streamers e que canal utilizaram para esse fim.



Go

Actions ▼

Create

	<u>Id M</u>	<u>M Time</u>	<u>Id C</u>	<u>User Name</u>	<u>Text</u>
	1	24-MAY-18 01.36.58.000000 PM	1	Pessoa10	Nao gosto da tua stream
	2	24-MAY-18 01.36.58.000000 PM	2	Pessoa10	azeite muito azeite azeite++
	3	24-MAY-18 01.36.58.000000 PM	1	Pessoa2	PogChamp!!
	4	24-MAY-18 01.36.58.000000 PM	1	Pessoa7	CY@
	5	24-MAY-18 01.36.58.000000 PM	2	Pessoa10	azeite pouco azeite azeite--

1 - 5

Ao carregar em create somos redirecionados para um form que nos permite adicionar mais uma mensagem ao sistema, basta preencher devidamente e a seguir validar carregando no botão create. Para mandar uma mensagem é necessário ter feito um login no próprio dia.

No menu início ao carregar em stream possível ver todas as streams registadas e a hora de começo das mesmas.

Edit	S Time	G Name	Description	Link
	28-MAR-12 11.10.00.000000 AM	Tetris	muito mas bastante muito ainda muito azeite	www.twitch/jorge.tv
	21-MAY-18 12.00.00.000000 AM	Tetris	muito mas bastanteg muito ainda muito azeite	www.twitch/jorge.tv

1 - 2

Ao carregar em edit, podemos editar alguns dados relativos à stream, adicionar uma nova stream, bastando apenas preencher devidamente os campos disponíveis.

Stream

Cancel Delete Apply Changes

Streamer Name Pessoa10

S Time 28-MAR-12 11.10.00.000000 AM

*G Name Tetris

Description

*Link www.twitch/jorge.tv

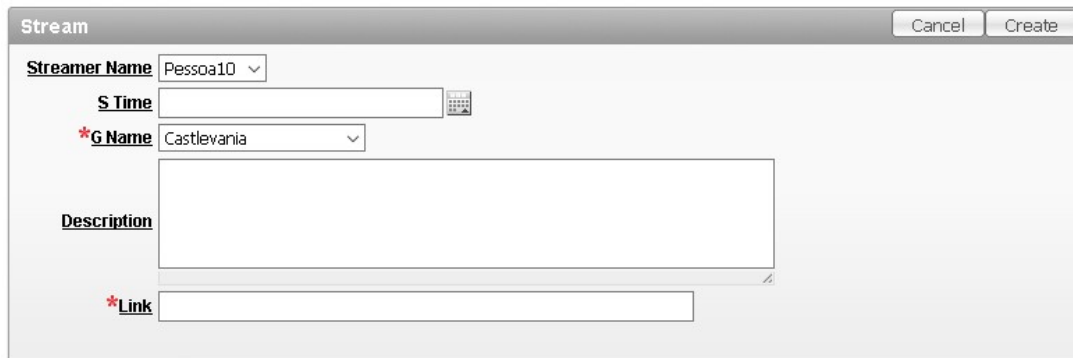
0 of 0

View Watch Detail

Delete Checked Add Row

	User Name	S Time
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>

Para criar uma stream é necessário fornecer um utilizador que seja Streamer, um tempo de início de stream, um jogo que será referenciado no stream, uma descrição e finalmente o link da mesma.



The image shows a web form titled "Stream" with a "Cancel" and "Create" button in the top right corner. The form contains the following fields:

- Streamer Name:** A dropdown menu with "Pessoa10" selected.
- S Time:** A text input field with a calendar icon to its right.
- *G Name:** A dropdown menu with "Castlevania" selected.
- Description:** A large text area.
- *Link:** A text input field.

6 Considerações finais

Esta Base de Dados implementada é uma escala reduzida do que seria esperado no mundo real atualmente. Com este trabalho deu para desenvolver novas capacidades relacionados com a criação de Base Dados e interfaces gráficas relacionadas com a mesma. Para a realização do trabalho foi necessário investigar vários sites de LiveStream para compreender-mos as necessidades e modularmos melhor o projecto.

Para concluir, foi interessante o desenvolvimento deste projeto. Confessamos, no entanto, que o facto de termos escolhido demasiadas datas para atributos das nossas entidades dificultou de algum modo a facilidade do projeto. Mas pensamos ter cumprido e executado fielmente os requisitos a que nos propusemos no início do trabalho.