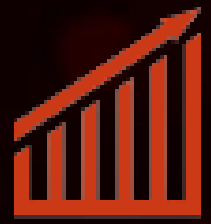
Trabalho TI2 - Sprint 2

Bruno Pena Baêta
Eric Azevedo de Oliveira
Felipe Nepomuceno Coelho



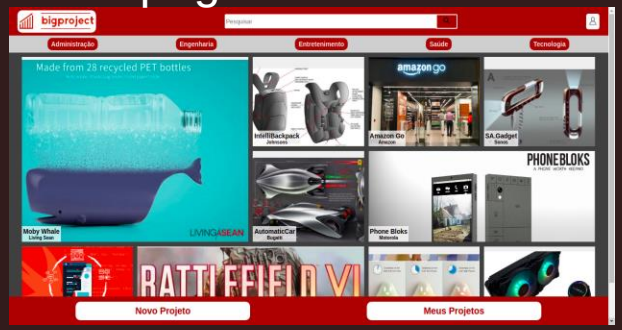
bigproject

Login

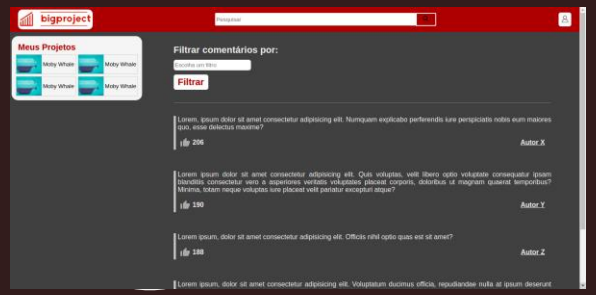
Projeto

Registro

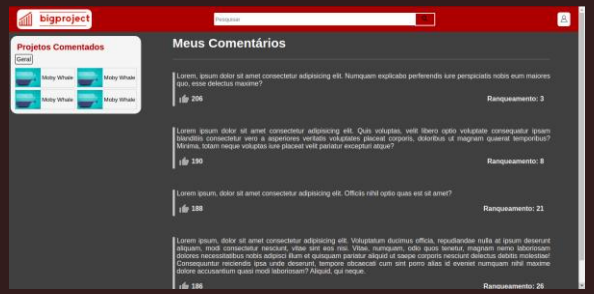
Homepage



Meus Projetos



Meus comentários



Criação do Projeto

CÁLCULO DE REPUTAÇÃO DO USUÁRIO

Níveis

0 - 0 XP
1 - 1000 XP
2 - 2500 XP
3 - 5500 XP
4 - 11500 XP
5 - 23500 XP
6 - 47500 XP
7 - 95500 XP
8 - 191500 XP
9 - 379500 XP
10 - 755500 XP

	Usuário que fez o comentário	Usuário que deu Like
Comentário vencedor	$+(2000 + (Nível * 0{,}7))$	$+(0{,}05 * (2000 + Nível * 0{,}7))$
Demais comentários	$+(200 + (Nível * 0{,}5 + Likes))$	$+(0{,}05 * (200 + (Nível * 0{,}5 + Likes)))$
Comentário lixo	$-(300 + (Nível * 0{,}5))$	$-(100 + (Nível * 0{,}5))$

=> Usuários não podem ver a reputação alheia

CÁLCULO DE QUALIDADE DO COMENTÁRIO

Peso do Like = PL

Quantidade de Likes = QL

Nível do Usuário que deu o Like = UL

Nível do Usuário que fez o comentário = UC

$$PL = UL^2 + 1$$

$$Qualidade\ do\ Comentário = (PL_0 + PL_1 + \ ... + PL_{QL}) * (UC\ + 1)$$

Peso do Like = PL

Qualidade antes do Like = QA

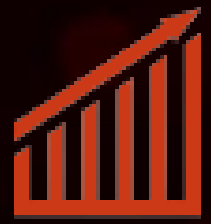
Nível do Usuário que fez o comentário = UC

A cada like novo no comentário, a coluna de Qualidade na tabela COMENTÁRIO, é atualizada. Para atualizar a coluna será usado o seguinte cálculo:

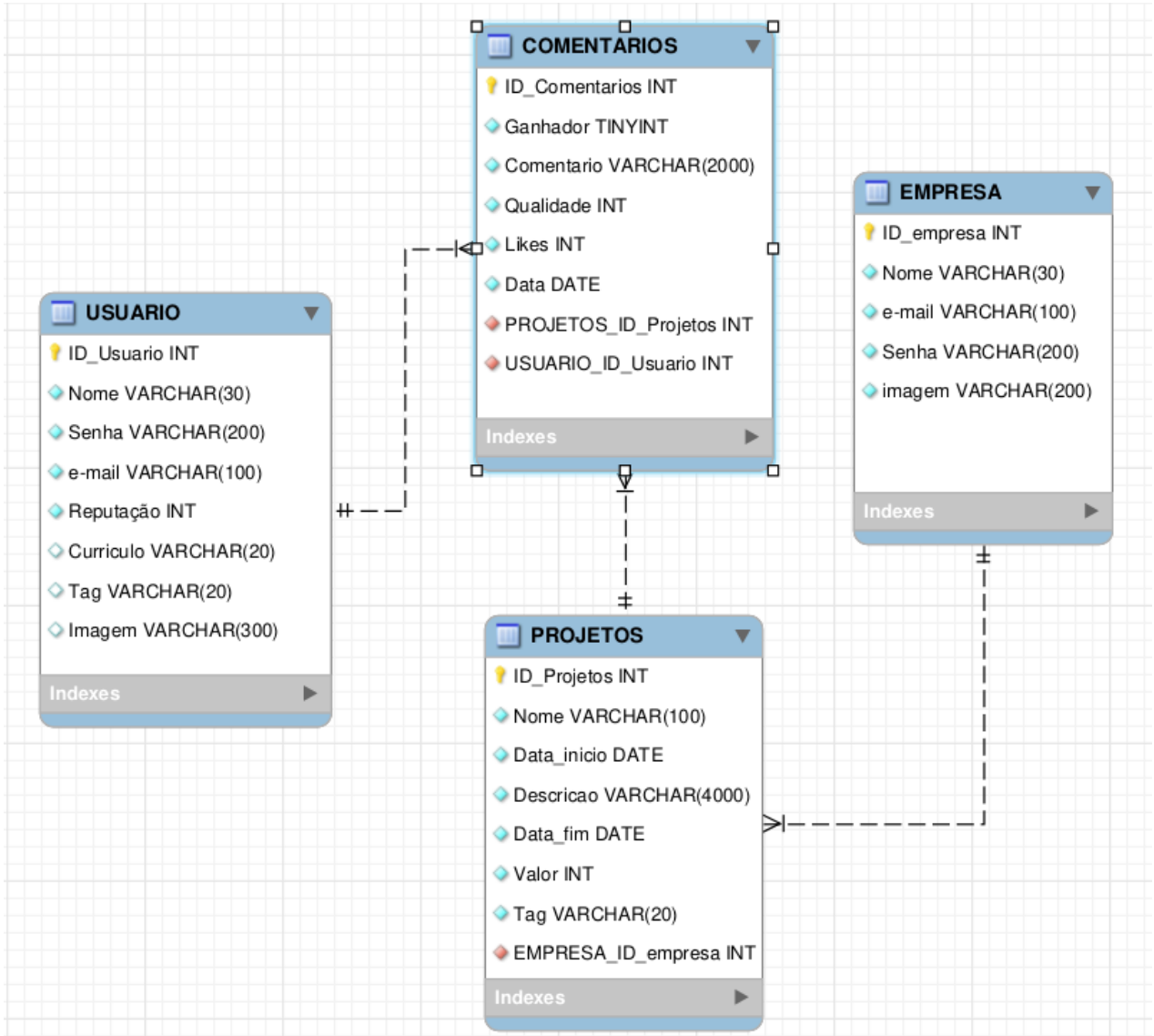
$$Nova\ Qualidade = \left(\left(\frac{QA}{UC+1}\right) + PL\right) * (UC + 1)$$

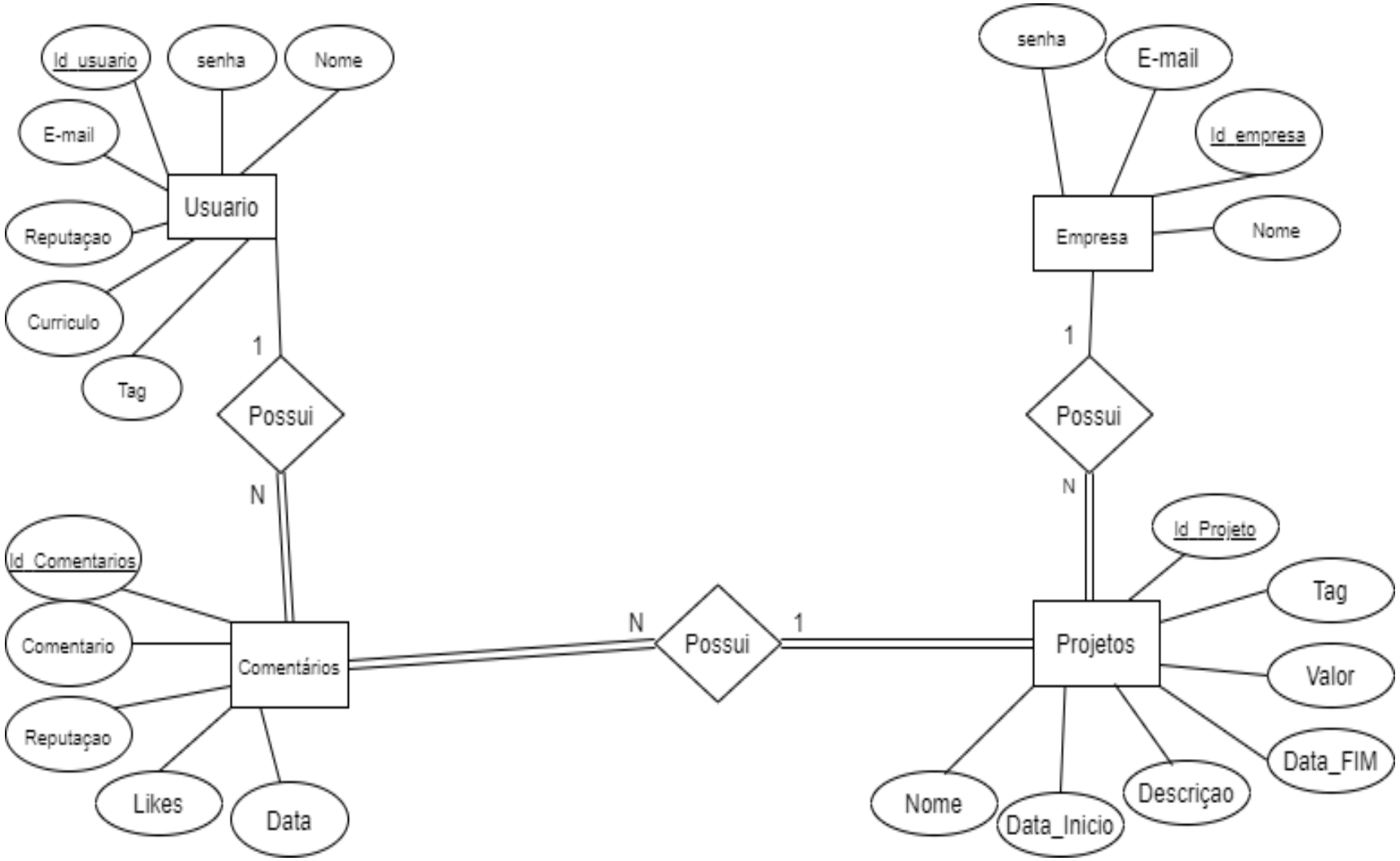
Trabalho TI2 – Sprint 3

Bruno Pena Baêta
Eric Azevedo de Oliveira
Felipe Nepomuceno Coelho



bigproject


COMENTARIOS
ID_Comentarios INT
Ganhador TINYINT
Comentario VARCHAR(2000)
Qualidade INT
Likes INT
Data DATE
PROJETOS_ID_Projetos INT
USUARIO_ID_Usuario INT
Indexes
USUARIO
ID_Usuario INT
Nome VARCHAR(30)
Senha VARCHAR(200)
e-mail VARCHAR(100)
Reputação INT
Curriculo VARCHAR(20)
Tag VARCHAR(20)
Imagem VARCHAR(300)
Indexes
EMPRESA
ID_empresa INT
Nome VARCHAR(30)
e-mail VARCHAR(100)
Senha VARCHAR(200)
imagem VARCHAR(200)
Indexes
PROJETOS
ID_Projetos INT
Nome VARCHAR(100)
Data_inicio DATE
Descricao VARCHAR(4000)
Data_fim DATE
Valor INT
Tag VARCHAR(20)
EMPRESA_ID_empresa INT
Indexes


Id_usuario
senha
Nome
E-mail
Usuario
Reputaçao
Curriculo
Tag
1
Possui
N
Id_Comentarios
Comentario
Comentários
Reputaçao
Likes
Data
N
Possui
1
senha
E-mail
Id_empresa
Empresa
Nome
1
Possui
N
Id_Projeto
Tag
Projetos
Valor
Data_FIM
Nome
Data_Inicio
Descriçao

USUARIO

ID_Usuario	Nome	Senha	E-mail	Reputacao	Curriculo	Tag

COMENTARIOS

ID-Comentario	Comentario	Qualidade	Likes	Data	Ganhador	Projeto_ID	Usuario_ID

PROJETOS

ID_Projeto	Nome	Data-Inicio	Descricao	Data_Fim	Valor	Tag	Empresa_ID

EMPRESA

ID_Empresa	Nome	E-mail	Senha

Conexão backend - frontend

Código em Java

```
BigProjectA conectar = new BigProjectA(); // conectar com nosso banco de dados
conectar.conectarPost();

/* codigos onde iremos colocar a pagina on */
get("/", (req, res) -> mandarSite.renderContent("/home.html"));
get("/login", (req, res) -> mandarSite.renderContent("/login.html"));
get("/register", (req, res) -> mandarSite.renderContent("/register.html"));
get("/myprojects", (req, res) -> mandarSite.renderContent("/myprojects.html"));
get("/creation", (req, res) -> mandarSite.renderContent("/creation.html"));
get("/project", (req, res) -> mandarSite.renderContent("/project.html"));
get("/mycomments", (req, res) -> mandarSite.renderContent("/mycomments.html"));
 }


fetch(`http://localhost:4567/mandarRe?query=${nome},${senha},${email},${file},${tag},${10}`, methodGet)
    .then(res => res.json())
```

Conexão backend – Banco de Dados

Código em Java

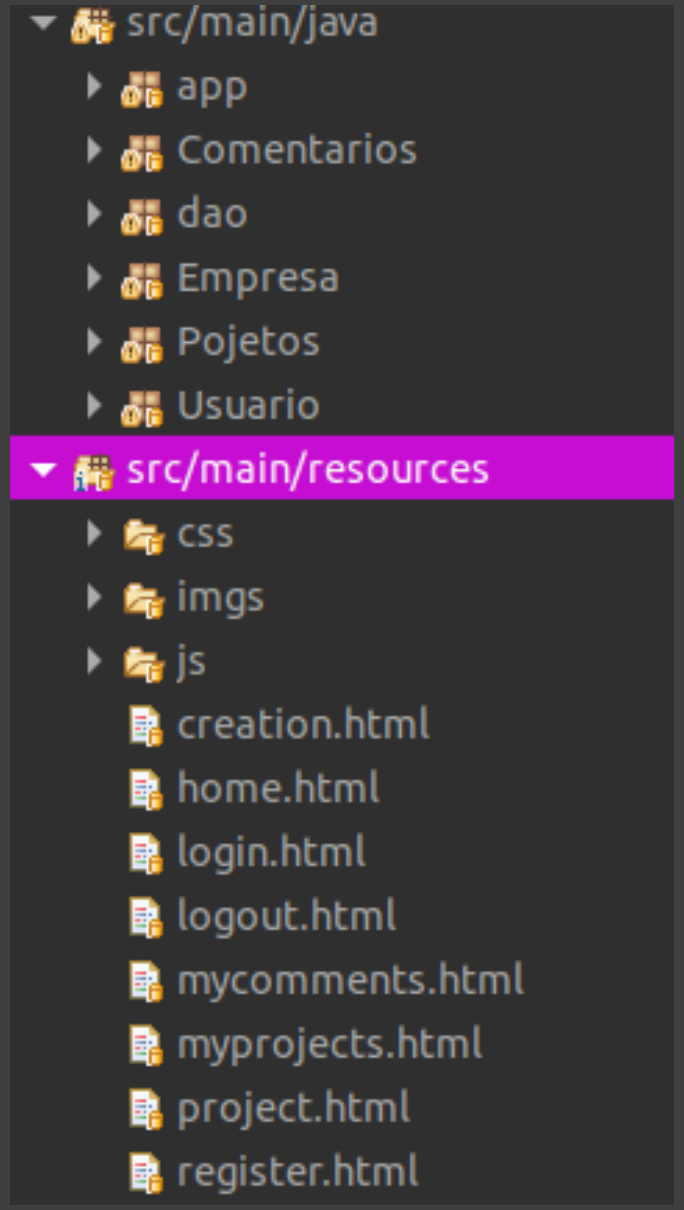
```java
// inseri uma nova empresa em nosso BD
public void inserirEmpresa(Empresa empresa) {
    boolean saberVerdade = false;

    try {
        Statement st = conexao.createStatement();
        st.executeUpdate("INSERT INTO empresa (idempresa,nome,email,senha,imagem)" + "VALUES ("
                + empresa.getIdEmpresa() + ", '" + empresa.getNomeEmpresa() + "', '" + empresa.geteMailEmpresa
                + "', '" + empresa.getSenhaEmpresa() + "', '" + empresa.getImagemEmpresa() + "');");

        st.close();
        saberVerdade = true;
    } catch (SQLException u) {
        throw new RuntimeException(u);
    }

}
```

Pastas do Projeto



Declaração e Metodos Get/Set

Código em Java

```java
public class Projetos {

    private int idProjeto;
    private String nomeProjeto;
    private String descricaoPojeto;
    private String dataInicio;
    private String dataFim;
    private int valorProjeto;
    private String tag;
    private String imagem;
    private String rec;

    public Projetos(int idProjeto1, String nomeProjeto1, String descricaoPojeto1, String dataInicio1, String dataFim1,
            int valorProjeto1, String tag1, String imagem1, String rec) {
        this.idProjeto = idProjeto1;
        this.nomeProjeto = nomeProjeto1;
        this.descricaoPojeto = descricaoPojeto1;
        this.dataInicio = dataInicio1;
        this.dataFim = dataFim1;
        this.valorProjeto = valorProjeto1;
        this.tag = tag1;
        this.imagem = imagem1;
        this.rec = rec;

    }
    public String getRec() {
        return rec;
    }

    public void setRec(String rec) {
        this.rec = rec;
    }
```

```
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`USUARIO`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`USUARIO` (
  `ID_Usuario` INT NOT NULL,
  `Nome` VARCHAR(30) NOT NULL,
  `Senha` VARCHAR(200) NOT NULL,
  `e-mail` VARCHAR(100) NOT NULL,
  `Reputação` INT NOT NULL,
  `Curriculo` VARCHAR(20) NULL,
  `Tag` VARCHAR(20) NULL,
  `Imagem` VARCHAR(300) NULL,
  PRIMARY KEY (`ID_Usuario`),
  UNIQUE INDEX `Nome_UNIQUE` (`Nome` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`EMPRESA`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`EMPRESA` (
  `ID_empresa` INT NOT NULL,
  `Nome` VARCHAR(30) NOT NULL,
  `e-mail` VARCHAR(100) NOT NULL,
  `Senha` VARCHAR(200) NOT NULL,
  `imagem` VARCHAR(200) NOT NULL,
  PRIMARY KEY (`ID_empresa`))
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`PROJETOS`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`PROJETOS` (
  `ID_Projetos` INT NOT NULL,
  `Nome` VARCHAR(100) NOT NULL,
  `Data_inicio` DATE NOT NULL,
  `Descricao` VARCHAR(4000) NOT NULL,
  `Data_fim` DATE NOT NULL,
  `Valor` INT NOT NULL,
  `Tag` VARCHAR(20) NOT NULL,
  `EMPRESA_ID_empresa` INT NOT NULL,
  PRIMARY KEY (`ID_Projetos`),
  INDEX `fk_PROJETOS_EMPRESA1_idx` (`EMPRESA_ID_empresa` ASC) VISIBLE,
  CONSTRAINT `fk_PROJETOS_EMPRESA1`
    FOREIGN KEY (`EMPRESA_ID_empresa`)
    REFERENCES `mydb`.`EMPRESA` (`ID_empresa`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `mydb`.`COMENTARIOS`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`COMENTARIOS` (
  `ID_Comentarios` INT NOT NULL,
  `Ganhador` TINYINT NOT NULL,
  `Comentario` VARCHAR(2000) NOT NULL,
  `Qualidade` INT NOT NULL,
  `Likes` INT NOT NULL,
  `Data` DATE NOT NULL,
  `PROJETOS_ID_Projetos` INT NOT NULL,
  `USUARIO_ID_Usuario` INT NOT NULL,
  PRIMARY KEY (`ID_Comentarios`),
  INDEX `fk_COMENTARIOS_PROJETOS1_idx` (`PROJETOS_ID_Projetos` ASC) VISIBLE,
  INDEX `fk_COMENTARIOS_USUARIO1_idx` (`USUARIO_ID_Usuario` ASC) VISIBLE,
  CONSTRAINT `fk_COMENTARIOS_PROJETOS1`
    FOREIGN KEY (`PROJETOS_ID_Projetos`)
    REFERENCES `mydb`.`PROJETOS` (`ID_Projetos`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_COMENTARIOS_USUARIO1`
    FOREIGN KEY (`USUARIO_ID_Usuario`)
    REFERENCES `mydb`.`USUARIO` (`ID_Usuario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Script do BD

Sistema inteligente

Arquivo Json formatado

```
    "language": "pt",
    "id": "1",
    "text": "Sou um sistema inteligente que busca palavras chaves em textos."
  },
  {
    "language": "pt",
    "id": "2",
    "text": "Existem diversos temas como astrofísica, dinâmica de fluidos entre outros."
  },
  {
    "language": "en",
    "id": "3",
    "text": "it's time to go to bed toby, i won't discuss this anymore."
  }
```

HTTP request

```
POST https://brazilsouth.api.cognitive.microsoft.com/text/analytics/v3.0/keyPhrases HTTP/1.1
Host: brazilsouth.api.cognitive.microsoft.com
Content-Type: application/json
Ocp-Apim-Subscription-Key: ••••••••••••••••••••••••••••••••

{
  "documents": [
    {
      "language": "pt",
      "id": "1",
      "text": "Sou um sistema inteligente que busca palavras chaves em textos."
    },
    {
      "language": "pt",
      "id": "2",
      "text": "Existem diversos temas como astrofísica, dinâmica de fluidos entre outros."
    },
    {
      "language": "en",
      "id": "3",
      "text": "it's time to go to bed toby, i won't discuss this anymore."
    }
  ]
}
```

Sistema inteligente

Resposta da API

```
Transfer-Encoding: chunked
csp-billing-usage: CognitiveServices.TextAnalytics.BatchScoring=3,CognitiveServices.TextAnalytics.TextRecor
ds=3
x-envoy-upstream-service-time: 28
apim-request-id: f1d3a22a-34a1-4d19-99a1-aece1bc4bb85
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
x-content-type-options: nosniff
Date: Sat, 15 May 2021 20:43:50 GMT
Content-Type: application/json; charset=utf-8

{
  "documents": [{
    "id": "1",
    "keyPhrases": ["palavras chaves em textos", "sistema inteligente"],
    "warnings": []
  }, {
    "id": "2",
    "keyPhrases": ["astrofísica", "dinâmica de fluidos", "temas", "outros"],
    "warnings": []
  }, {
    "id": "3",
    "keyPhrases": ["bed toby", "time"],
    "warnings": []
  }],
  "errors": [],
  "modelVersion": "2020-07-01"
}
```


FIM