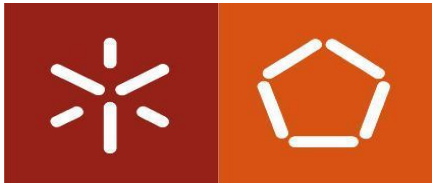
Informatics-Social-Network



Universidade do Minho
Departamento de Informática

Grupo 2

Processamento de Linguagens e Conhecimento

Universidade do Minho, Mestrado Integrado em Engenharia Informática,
4º Ano, 1º Semestre, Janeiro 2020

Estrutura da Apresentação

- Gramática
- Aplicação Web
- Demonstração

Gramática

Objetivo

- Processar e extrair informação do ficheiro de registo do utilizador
- Construir JSON com dados do utilizador
- *Visitor Pattern*

Ficheiro de Registo

```
------------------------------------------ Registo ------------------------------------------

Para efetuar o registo devidamente, preencha os campos substituando os espacos para o efeito.
No final faca upload do ficheiro na plataforma.

------------------------------------------ Registo ------------------------------------------

Nome: _____________________________________________________

Email: _____________________________________________________

Cidade: _____________________________________________________

Curso: _____________________________________________________

Password: _____________________________________________________

Tipo de Utilizador (aluno ou docente): _____________________

Genero (masculino ou feminino): _____________________________
```

Gramática

```
parser grammar RegisterParser;

options {
    tokenVocab=RegisterLexer;
}
registo
    : cabecalho info
    ;
cabecalho
    : BLOCK TEXT BLOCK
    ;
info
    : ENUNCIADO NOME ENUNCIADO EMAIL ENUNCIADO NOME ENUNCIADO NOME ENUNCIADO PASSWORD ENUNCIADO NOME ENUNCIADO NOME
    ;
```

```
lexer grammar RegisterLexer;

BLOCK       :   [\-]+' '[a-zA-Z]+' '[\-]+                       ;
TEXT        :   [^\-]                                           ;
NOME        :   ([a-zA-Z]+)(' '[a-zA-Z]+)*                      ;
EMAIL       :   ([a-zA-Z]|[\-_.]|[0-9])+[@]([a-z]+)[.]([a-z]+)  ;
PASSWORD    :   ([a-zA-Z]|[\-_@#!*]|[0-9])+                     ;
ENUNCIADO   :   ([a-zA-Z]|' '|[(),'])+[:]                       ;
WS          :   [ \r\n\t] -> skip                               ;
```

Visitor

```
public class Visitor extends  RegisterParserBaseVisitor<Integer>{
    private String res;

    public Visitor() { this.res = "{ \"nome\": " ; }

    public String getJson() { return this.res; }

    @Override
    public Integer visitRegisto(RegisterParser.RegistoContext ctx){
        return visit(ctx.info());
    }
    @Override
    public Integer visitInfo(RegisterParser.InfoContext ctx) {
        this.res += "\"" + ctx.NOME( i: 0) + "\", \"email\": ";
        this.res += "\"" + ctx.EMAIL() + "\", \"cidade\": ";
        this.res += "\"" + ctx.NOME( i: 1) + "\", \"curso\": ";
        this.res += "\"" + ctx.NOME( i: 2) + "\", \"password\": ";
        this.res += "\"" + ctx.PASSWORD() + "\", \"tipo\": ";
        this.res += "\"" + ctx.NOME( i: 3) + "\", \"genero\": ";
        this.res += "\"" + ctx.NOME( i: 4) + "\" }";

        return 1;
    }
}
```

Input

```
------------------------------------------ Registo ------------------------------------------

Para efetuar o registo devidamente, preencha os campos substituando os espacos para o efeito.
No final faca upload do ficheiro na plataforma.

------------------------------------------ Registo ------------------------------------------

Nome: Ricardo Pereira

Email: ricardo@hotmail.com

Cidade: Braga

Curso: MIEI

Password: sdf*A342-_@Dad!#

Tipo de Utilizador (aluno ou docente): aluno

Genero (masculino ou feminino): masculino
```

Resultado


```
{
    "nome": "Ricardo Pereira",
    "email": "ricardo@hotmail.com",
    "cidade": "Braga", "curso": "MIEI",
    "password": "sdf*A342-_@Dad!#",
    "tipo": "aluno",
    "genero": "masculino"
}
```

Utilização

- Conversão da Gramática em Java para NodeJS
  - antlr4 -Dlanguage=JavaScript -visitor <Lexer e Parser>.g4

- Implementação no Registo do Utilizador
  - Download do ficheiro de registo
  - Preenchimento
  - Upload
  - Inserção do Utilizador

Aplicação Web

Funcionalidades

- Alunos e docentes
  - Registar eventos/publicações
  - Atualizar/remover eventos/publicações criados por si
  - Pesquisar eventos/publicações
  - Atualizar e exportar dados do perfil
- Admin
  - Acesso a toda a informação da base de dados
  - Ativar/desativar utilizadores
  - Remover publicações/eventos

Arquitetura da Solução

- Modelos
- Controladores
- Rotas
- Vistas

Modelos

- Evento
- Publicação
- Utilizador

Modelos - Evento

```
var EventoSchema = new mongoose.Schema(
    {
        tipo: {type: String, required: true},
        titulo: {type: String, required: true},
        data: {type: String, required: true},
        local: {type: String, required: true},
        descricao: {type: String, required: true},
        uc: {type: String},
        duracao: {type: String},
        hora: {type: String, required: true},
        email_utilizador: {type: String, required: true},
        id_utilizador: {type: String, required: true},
        anexos:[{type: FicheiroSchema}],
        visivilidade: {type: Number, required: true},
        utilizadores: [{type: UserSchema}]
    }
```

Modelos - Publicação

```
var PublicacaoSchema = new mongoose.Schema(
    {
        email_utilizador: {type: String, required: true},
        id_utilizador: {type: String, required: true},
        titulo: {type: String, required: true},
        curso: {type: String, required: true},
        data: {type: String, required: true},
        descricao: {type: String, required: true},
        gostos: [{type: String}],
        comentarios: [{type: ComentarioSchema}],
        anexos:[{type: FicheiroSchema}],
        visivilidade: {type: Number, required: true},
    }
)
```

Modelos - Utilizador

```
var UtilizadorSchema = new mongoose.Schema(
    {
        nome: {type: String, required: true},
        curso: {type: String, required: true},
        local: {type: String, required: true},
        email: {type: String, required: true},
        tipoUtilizador: {type: String, required: true},
        genero: {type: String, required: true},
        password: {type: String, required: true},
        ativo: {type: Number },
        fotografia: {type: FicheiroSchema},
        anexos:[{type: FicheiroSchema}]
    }
)
```

Modelos - Esquemas auxiliares

```
var FicheiroSchema = new mongoose.Schema(
    {
        data: {type: String, required: true},
        name: {type: String, required: true},
        path: {type: String, required: true},
        mimetype: {type: String, required: true},
        size: {type: Number, required: true},
    }
)
```

```
var ComentarioSchema = new mongoose.Schema(
    {
        data: {type: String, required: true},
        descricao: {type: String, required: true},
        id_utilizador: {type: String, required: true},
        email_utilizador: {type: String, required: true},
    }
)
```

```
var UserSchema = new mongoose.Schema(
    {   email_utilizador: {type: String, required: true},
        id_utilizador: {type: String, required: true},
    },
    {   _id: false    }
)
```

Controladores

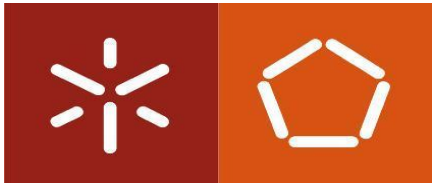
- Evento
- Publicação
- Utilizador

Rotas

- Rotas da API
  - Utilizam os controllers para obter informação da base de dados
- Rotas do Front-End
  - Fazem a ligação entre a API e o Front-End

Demonstração

Informatics-Social-Network



Universidade do Minho
Departamento de Informática

Grupo 2

Processamento de Linguagens e Conhecimento

Universidade do Minho, Mestrado Integrado em Engenharia Informática,
4º Ano, 1º Semestre, Janeiro 2020