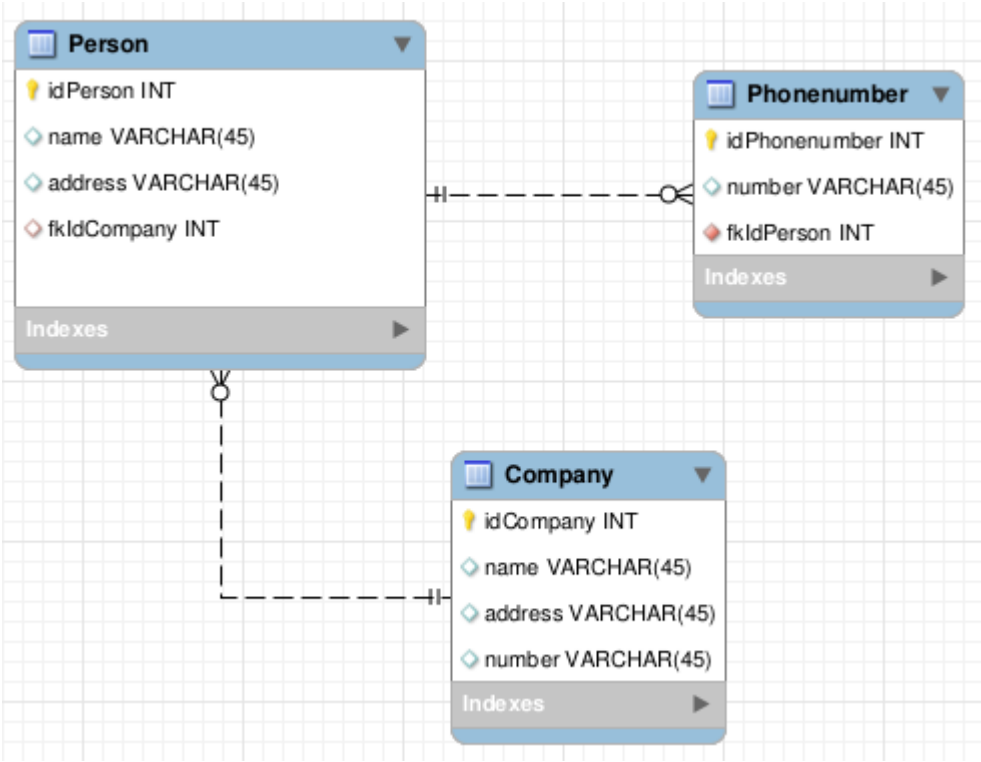
Esquema da base de dados



Webservices criados – Agenda Electrónica

Pessoas:

PUT: http://localhost:10000/ubiwhere/phonebook/person/add

Adiciona uma pessoa.

Request:

```
{
  name:'joaocesar',
  address:'rua do cardal'
}
```

PUT: http://localhost:10000/ubiwhere/phonebook/person/1/edit

Edita os detalhes de uma pessoa.

Request:

```
{
 name:'joaosantoscesar',
 address:'rua do cardal'
}
```

PUT: http://localhost:10000/ubiwhere/phonebook/person/1/number/add

Adiciona um número a uma pessoa.

Request:

```
{
  number:'964343564'
}
```

GET: http://localhost:10000/ubiwhere/phonebook/person/1/details

Lista os detalhes de uma pessoa.

Response:

```
{"idPerson":1,"name":"joaosantoscesar","address":"rua do
cardal","phonenumberCollection":[{"idPhonenumber":1,"number":"964343564"}]}
```

DELETE: http://localhost:10000/ubiwhere/phonebook/person/1/number/1/delete

Remove um número.

PUT: http://localhost:10000/ubiwhere/phonebook/person/1/company/1/link

Associa uma pessoa a uma empresa.

PUT: http://localhost:10000/ubiwhere/phonebook/person/1/company/1/unlink

Desassocia uma pessoa de uma empresa

GET: http://localhost:10000/ubiwhere/phonebook/persons/details

Mostra todas as pessoas existentes na agenda.

Response:

```
[{"idPerson":1,"name":"joaosantoscesar","address":"rua do
cardal","phonenumberCollection":[{"idPhonenumber":1,"number":"964343564"}]}]
```

GET: http://localhost:10000/ubiwhere/phonebook/person/1/delete

Remove uma pessoa da agenda.

Empresas:

PUT: http://localhost:10000/ubiwhere/phonebook/company/add

Adiciona uma empresa à agenda.

Request:

```
{
  name:'empresadojoao',
  address:'rua do cardal n7',
  number:'99993333'
}
```

PUT: http://localhost:10000/ubiwhere/phonebook/company/1/edit

Edita uma empresa da agenda.

Request:

```
{
  name:'empresadojoaocesar',
  address:'rua do cardal n7',
  number:'99993333'
}
```

DELETE: http://localhost:10000/ubiwhere/phonebook/company/1/delete

Remove uma empresa da agenda.

GET: http://localhost:10000/ubiwhere/phonebook/company/1/details

Mostra todas as empresas na agenda.

Response:


```
[{"idCompany":1,"name":"empresadojoao","address":"rua do cardal
n7","number":"99993333","personCollection":[]}]
```

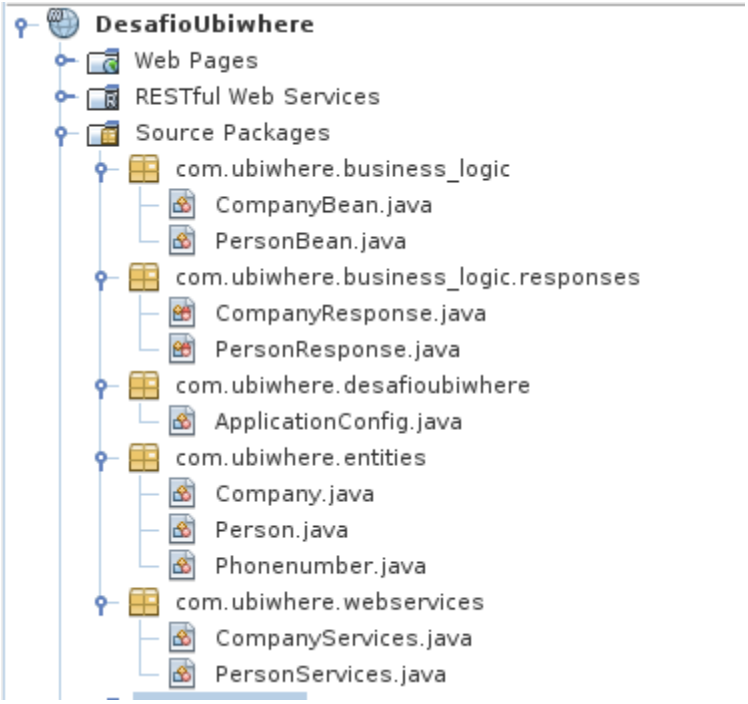
GET: http://localhost:10000/ubiwhere/phonebook/company/details

Mostra todas as empresas na agenda.

Response:


```
[{"idCompany":1,"name":"empresadojoao","address":"rua do cardal
n7","number":"99993333","personCollection":[]}]
```

A solução



A solução desenvolvida é composta por 4 packages principais. O ***com.ubiwhere.business_logic***é o package onde foram criados os Beans e onde estão criados os metódos reponsáveis pelas operações básicas da agenda.

No package ***com.ubiwhere.business_logic.responses*** foram criadas as classes dos enumerados que especificam o tipo de respostas dos métodos dos Beans criados.

O ***com.ubiwhere.entities*** alberga as classes responsáveis pelo mapeamento das tabelas da base de dados que suportam a nossa agenda.

No ***com.ubiwhere.webservices*** foram criadas as classes onde estão declarados os webservices em si e servem como ponto de comunicação para a nossa agenda.

Execução

O projeto pode ser executado através das ferramentas Docker e DockerCompose. Para criar as imagens que suportam os *containers* é necessário primeiro executar o comando docker-compose build, através do terminal, na raiz da pasta do projeto. Depois para executar os containers e mais uma vez na raiz da pasta do projeto, é necessário executar o comando docker-compose up (docker-compose stop é utilizado para parar os containers).

A solução desenvolvida tem 2 containers, um que executa o Glassfish e outro que executa o Mysql.

O ficheiro de configuração (Dockerfile) do container Mysql, está dentro da pasta mysqlcontainer e o ficheiro de contiguração (Dockerfile) do container Glassfish, está na raiz da pasta, onde também está criado o ficheiro docker-compose.yml, onde estão declarados os containers/imagens constituintes.