

Ournote

Artur Queiroz

Luis Albuquerque

Conteúdo

Introdução	3
Modelo	3
Users	3
Groups	3
Esquema	4
API	5
GET	5
POST	5
PUT	5
DELETE	5
Interface	5
GET	5
POST	5
PUT	6
DELETE	6
Cliente	6
Conclusão	6
Trabalho futuro	6

Introdução

Já existem muitas redes sociais, umas mais voltadas para partilhar fotos, outras para partilhar o estado das pessoas, e outras até mais focadas no mundo de trabalho, no entanto não existem assim tantas focadas em desenvolvimento e partilha de notas entre alunos. Com isso em conta, desenvolvemos Ournote, que é uma rede social focada em alunos, organizada em grupos e em páginas, onde o conteúdo pode ser organizado como num caderno, mas com a vantagem de aqui ser permitido facilmente trocar a ordem dos textos, imagens, ficheiros, etc. Os grupos são constituídos por páginas, e uma página esta organizada por uma meta linguagem, desenvolvida por nós, que permite distinguir títulos de diferentes importâncias, criar paragrafos, listas de paragrafos. imagens, eventos, pdfs, e até ficheiros distintos para fazer download.

Modelo

No nosso modelo os nossos modelos foram distinguidos em duas classes, users e groups.

Users

```
name: String,
email : {
  type: String,
  unique: true,
  lowercase: true,
  required: true
},
password: {
  type: String,
  required: true
},
favourite: [ String ],
```

Os *favourites* são um conjunto de caminhos de grupos

Groups

Cada evento tem um título e uma data.

```
var eventSchema = new mongoose.Schema({
  title : String,
  data : String
});
```

Uma *card* é um objeto de uma página pode ser um parágrafo, um título uma imagem, etc.

```

var card_Schema = new mongoose.Schema({
  p: String,
  img: String,
  pdf: String,
  h1: String,
  h2: String,
  h3: String,
  a: String,
  file : String,
  list : [String],
  event : eventSchema,

```

Tudo acima, dentro de *card* são os objectos que uma página pode ter.

```

  comment : [String],
  tags : [String]

```

Lista de comentários e uma lista de *tags*.

```

});

```

```

var PATH = String;
var EMAIL = String;

```

Um *group* é algo que tem uma página com várias *cards* e outros sub-grupos, com *tags* e umas permissões, tanto de leitura como de escrita.

```

var groupSchema = new mongoose.Schema({
  path : {
    type :String,
    unique: true,
    required: true
  },
  id_creator: EMAIL,
  name: {
    type: String,
    required: true
  },
  tags : [String],
  sub_groups : [ PATH ],
  read_perm : [ EMAIL ],
  write_perm : [ EMAIL ],
  page : [ card_Schema ],
});

```

Esquema

Nós dividimos o trabalho em dois servidores (API e Interface) e clientes.

API

GET

<code>/*?token=TOKEN</code>	--- dados do grupo
<code>/profile?token=TOKEN&email=EMAIL</code>	--- favoritos de um utilizador
<code>/profile?token=TOKEN&email=EMAIL&tag=TAG</code>	--- Sitios que têm associado uma determinada tag
<code>/user/:email?token=TOKEN</code>	--- dados de um utilizador

POST

<code>/*?token=TOKEN&update=comment</code>	--- adiciona um comentario a um objeto
<code>/*?token=TOKEN&update=add</code>	--- adiciona premissões a um ou mais utilizadores
<code>/*?token=TOKEN&update=remove</code>	--- remove premissões a um ou mais utilizadores
<code>/*?token=TOKEN&type=file</code>	--- upload de um ficheiro
<code>/*?token=TOKEN</code>	--- adiciona um novo grupo
<code>/login?token=TOKEN&email=EMAIL</code>	--- faz login
<code>/favourite?token=TOKEN&email=EMAIL&path=PATH</code>	--- adiciona uma página aos favoritos
<code>/register</code>	--- regista um utilizador

PUT

<code>/*?token=TOKEN&type=TYPE</code>	--- acrescenta um objeto à página
---	-----------------------------------

DELETE

<code>/*?token=TOKEN</code>	--- remove um objeto da página
-----------------------------	--------------------------------

Interface

GET

<code>/</code>	--- responde com a página de login
<code>/register</code>	--- responde com a página para registar
<code>/profile?tag=TAG</code>	--- procura objectos que tenham a tag TAG
<code>/logout</code>	--- termina a sessão de um utilizador
<code>/*</code>	--- página de um grupo
<code>/*?json=true</code>	--- grupo em json

POST

<code>/add_favourite?path=PATH</code>	--- adiciona aos favoritos
<code>/register</code>	--- regista um utilizador
<code>/login</code>	--- faz login
<code>root/*?update=add</code>	--- acrescenta premissões
<code>root/*?update=remove</code>	--- remove premissões
<code>root/*?update=comment</code>	--- acrescenta um comentario

`root/*?type=TYPE`

--- acrescenta um ficheiro

PUT

`root/*`

--- acrescenta um objeto

DELETE

`root/*`

--- elimina um ou mais objetos

Cliente

A página de um grupo é constituída por 3 partes, a da esquerda é uma barra de navegação com a parte de baixo mais focada no utilizador (logout, profile, adicionar favorito) e ao fim de todos os subgrupos existe um botão para acrescentar um grupo com as permissões desejadas. Uma parte central com o conteúdo da página, em que na parte de cima tem operações para trocar, eliminar e introduzir objectos, no lado direito de cada objecto encontra-se uma checkbox e um icon para ver os comentários daquele objecto. E por último uma parte direita que contém, um botão para adicionar um evento, para adicionar pessoas ao grupo, bem como remover (estas operações são feitas apenas alterando as permissões de escrita e leitura do grupo), tem também uma área com todos os eventos de um grupo, índice, ficheiros e imagens, onde se clicar em cada uma dessas vai para o sitio referido na página. A página do profile, contém todas as páginas adicionada como favoritas, e um campo onde é possível fazer pesquisa de tags em todos os grupos que o utilizador tem permissão de leitura.

Conclusão

Ao longo desta jornada, encontramos diversos desafios, quer na discussão e esquematização da solução, como na própria implementação, como decidir qual seria a nossa abordagem e ao facto de termos usado javascript, e isso nos trazer muitos problemas na hora de debugging, uma vez que os erros muitas vezes ficam perdidos entre a API e a Interface, mas tendo o postman podemos testar individualmente a API, e com isso identificar mais facilmente a razão do problema. No fim de contas, conseguimos apresentar uma rede social funcional, com uma proposta bem original, que acreditamos que consiga ser útil para estudantes organizarem seus trabalhos, e a sua vida escolar.

Trabalho futuro

No nosso trabalho, usamos uma meta linguagem para representar as páginas dos grupos, uma boa adição, seria adicionarmos elementos recursivas e outras funcionalidades, como canvas e suporte de videos, etc. Podiamos criar um parser, e com ajuda do pandoc convertermos as páginas em pdf, powerpoint, word, epub,

etc. Neste momento, é possível acrescentar favoritos, mas não eliminá-los, isso seria uma boa adição.