	<p align="center">Centro Federal de Educação Tecnológica de Minas Gerais Campus VIII – Varginha Curso Técnico em Informática</p>	
<p><i>Disciplina</i> Lab. Aplicações Móveis</p>	<p align="center">Configurando uma Fila no Backend</p>	<p><i>Professor</i> Lázaro Eduardo da Silva</p>

Nessa aula montar uma fila no backend para envio das mensagens Push. A documentação desta ferramenta está no link abaixo:

<https://github.com/RomainLanz/adonis-bull-queue>

Inicialmente instale a biblioteca:

```
npm install @rlanz/bull-queue
```

Após finalizada a instalação, você precisa executar o script de configuração. Para isso, execute o comando abaixo e preencha os dados de autenticação.

```
node ace configure @rlanz/bull-queue
```

Observe que ao executar a configuração, são criadas 3 variáveis de ambiente: `QUEUE_REDIS_HOST`, `QUEUE_REDIS_PORT` e `QUEUE_REDIS_PASSWORD`. Você deverá incluir estas variáveis no servidor render.

Feito isso, vamos criar um Job, trabalho, que irá executar a tarefa de envio da mensagem para o celular do usuário. Para isso, execute o comando abaixo.

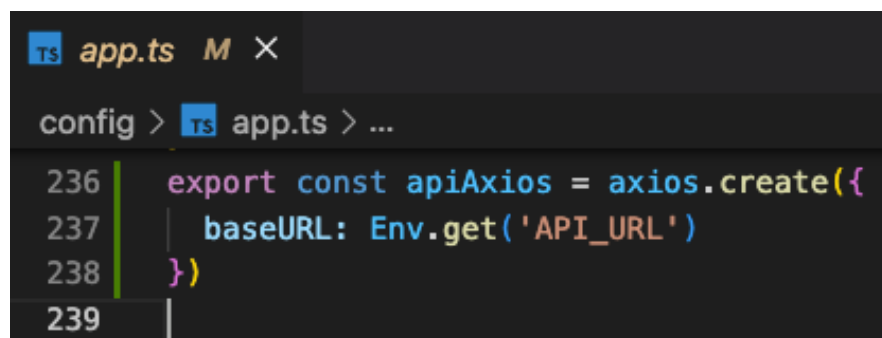
```
node ace make:job SendPush
```

Esse comando irá criar um arquivo na pasta `app/Jobs/SendPush.ts`.

A tarefa do nosso Job será realizar uma requisição para a Notificação do Expo solicitando o envio da mensagem. Para isso, vamos utilizar a biblioteca `axios`. Vamos instalar esta biblioteca no nosso projeto executando o comando:

```
npm install axios
```

Para utilizar esta biblioteca, vamos acrescentar no começo do arquivo `config/app.ts` a sua importação e no final do arquivo o código abaixo.

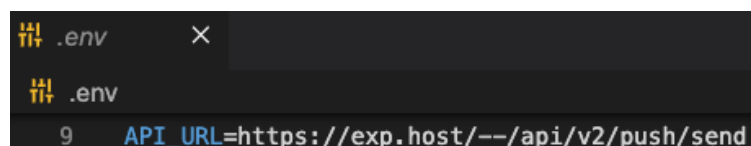


```

TS app.ts M X
config > TS app.ts > ...
236 | export const apiAxios = axios.create({
237 |   |   baseURL: Env.get('API_URL')
238 |   | })
239 |

```


Observe que importamos uma variável `Env`, que é uma variável de ambiente. Com isso precisaremos criá-la no arquivo `.env` e após o deploy, criar na Environment do servidor render. O conteúdo dela deve ser:



```

.env X
.env
9 API_URL=https://exp.host/--api/v2/push/send

```

	<p align="center">Centro Federal de Educação Tecnológica de Minas Gerais Campus VIII – Varginha Curso Técnico em Informática</p>	
<p><i>Disciplina</i> Lab. Aplicações Móveis</p>	<p align="center">Configurando uma Fila no Backend</p>	<p><i>Professor</i> Lázaro Eduardo da Silva</p>

Após realizar todas essas configurações, podemos preencher o arquivo SendPush que está na pasta app/Jobs com o conteúdo abaixo:

```

TS SendPush.ts U X
app > Jobs > TS SendPush.ts > ...
 1  import { apiAxios } from 'Config/app'
 2  import type { JobHandlerContract, Job } from '@ioc:Rlanz/Queue'
 3  import { AxiosError } from 'axios';
 4
 5  export type SendPushPayload = {}
 6
 7  export default class implements JobHandlerContract {
 8      constructor(public job: Job) {
 9          this.job = job
10      }
11
12      /**
13       * Base Entry point
14       */
15      public async handle(payload: SendPushPayload) {
16          try {
17              console.log(payload)
18              await apiAxios.post(
19                  'https://exp.host/--api/v2/push/send',
20                  payload
21              );
22              console.log('foi')
23          } catch (error) {
24              const err = error as AxiosError
25              console.log(err.response?.data)
26          }
27      }
28
29      /**
30       * This is an optional method that gets called if it exists wh
31       */
32      public async failed() { }
33  }

```

Feito isso, vamos colocar o Job para ser executado quando a fila estiver em execução acrescentando o seu nome no arquivo contracts/queue.ts

```
queue.ts M X
contracts > queue.ts > ...
1 | import type { SendPushPayload } from 'App/Jobs/SendPush';
2 | declare module '@ioc:Rlanz/Queue' {
3 |   interface JobsList {
4 |     'App/Jobs/SendPush': SendPushPayload
5 |   }
6 | }
```

Com isso, podemos criar o controller que irá receber a requisição com a mensagem. Para isso, execute o comando abaixo.


```
node ace make:controller SendPush
```

Esse comando irá criar um arquivo na pasta app/Controllers/Http/SendPushesController.ts.

Utilizaremos este arquivo para receber o conteúdo da mensagem e encaminhá-lo para o Job da fila que tem como trabalho encaminhar a mensagem para o celular do usuário que tem o id com Expo Token cadastrado.

```
SendPushesController.ts U X
app > Controllers > Http > SendPushesController.ts > ...
1 | import type { HttpContextContract } from '@ioc:Adonis/Core/HttpContext'
2 | import User from 'App/Models/User'
3 | import { Queue } from '@ioc:Rlanz/Queue';
4 |
5 | export default class SendPushesController {
6 |   public async send({ request }: HttpContextContract) {
7 |     const { id, title, body } = request.all()
8 |     const userDB = await User.findOrFail(id)
9 |     Queue.dispatch('App/Jobs/SendPush', {
10 |       to: userDB.token,
11 |       title,
12 |       body
13 |     });
14 |     return userDB
15 |   }
16 | }
```

Depois vamos criar a rota que chama este controller. Por segurança, vamos permitir o envio da mensagem somente para usuários que estejam autenticados.

	<p align="center">Centro Federal de Educação Tecnológica de Minas Gerais Campus VIII – Varginha Curso Técnico em Informática</p>	
<p><i>Disciplina</i> Lab. Aplicações Móveis</p>	<p align="center">Configurando uma Fila no Backend</p>	<p><i>Professor</i> Lázaro Eduardo da Silva</p>

```

routes.ts M ×
start > routes.ts > ...
8   Route.group(() => {
9     Route.put("/user", "UsersController.update")
10    Route.post("/push", "SendPushesController.send")
11  }).middleware('auth')

```

Assim como fizemos na migrate, vamos criar o script que executa a fila para colocarmos ela para executar no servidor.

```

package.json M ×
package.json > {} scripts > queue
5   "scripts": {
6     "dev": "node ace serve --watch",
7     "build": "node ace build --production",
8     "start": "node build/server.js",
9     "test": "node ace test",
10    "migrate": "node ace migration:run",
11    "queue": "node ace queue:listen"
12  },


```

Outro recurso que precisaremos acrescentar no servidor é o banco de dados REDIS. Ele é um banco de dados que roda na memória do servidor guardando dados estruturados como chave:valor. Para acrescentar, clique em New + e clique em Redis. Dê um nome para o seu servidor e solicite sua inclusão no projeto. Aguarde o status ficar verde e ele já estará disponível para o seu projeto.



Para que sua aplicação utilize este banco de dados você precisará da Internal Redis URL. Ela é composta por `redis://host:port`. Configure estes valores para as variáveis de ambiente `QUEUE_REDIS_HOST` e `QUEUE_REDIS_PORT`.

Faça o commit do seu projeto, acompanhe o deploy e verifique se tudo executa conforme o esperado. Para testar, primeiro acesse o banco de dados no servidor `neon.tech` e verifique os ids dos usuários que estão com o token cadastrado na sua tabela de usuários.

	<p align="center">Centro Federal de Educação Tecnológica de Minas Gerais Campus VIII – Varginha Curso Técnico em Informática</p>	
Disciplina Lab. Aplicações Móveis	Configurando uma Fila no Backend	Professor Lázaro Eduardo da Silva

1 select id, name, token from users			
<div> Run Explain Analyze </div>			
2ms 4 rows			
#	id	name	token
1	1	Lázaro	ExponentPushToken[_ZIP2fKC8Ox1Bvh08NrgKz]
2	6	Breno	ExponentPushToken[fAITbhGsOMCvNDXmh2XE5b]
3	7	Julya	ExponentPushToken[Mbo6VuM8CQH7ICn2xFOlcX]
4	8	Maria Clara	ExponentPushToken[fAITbhGsOMCvNDXmh2XE5b]

Para realizar o teste, você precisará do id do usuário para indicar para quem a mensagem deve ser enviada.

Crie uma nova requisição no arquivo api.rest com o conteúdo abaixo:

```

20 ##### Mensagem Push
   Send Request
21 POST https://apiadonis2023.onrender.com/push HTTP/1.1
22 Content-Type: application/json
23 Authorization: Bearer MzA.YZeMzF9i8KPHdnMc3VrHWhpaQV-RF6Q_vmHPLmaCR_XZbmtkig11YcbNYtSX
24
25 {
26   "id": 1,
27   "title": "Oi",
28   "body": "Tudo e você?"
29 }
```

Altere o endereço do servidor para o seu.

O token na frente do Bearer deve ser o retornado pelo comando de autenticação.

Altere o título e o corpo da mensagem e verifique se ela chega no usuário do celular indicado pelo id.

Bom trabalho!