



UNIVERSIDADE ESTADUAL DE CAMPINAS

---

## Tópico 01: Atividade

---

Atividade da disciplina Programação Web SI401B  
submetido em Setembro de 2020.

São Paulo  
2020

## Informações Gerais do Projeto

---

- Título do projeto: Tópico 01: Atividade.
- Estudantes responsáveis: Enzo Juniti Fujimoto ([e233930@dac.unicamp.br](mailto:e233930@dac.unicamp.br)).
- Número de matrícula: 233930.
- Docente responsável: Guilherme Palermo Coelho (301143).
- Disciplina: SI401 B - Programação Web.
- Instituição sede do projeto<sup>1</sup>: Faculdade de Tecnologia (FT) da Universidade Estadual de Campinas.
- Período de vigência proposto: 17 de Setembro de 2020 a 24 de Setembro de 2020.

---

<sup>1</sup> O projeto concentrou-se remotamente devido ao período emergencial proporcionado pela a pandemia do Coronavírus.

# Sumário

---

<b>1. Informações Gerais do Projeto</b>	<b>1</b>
2. Introdução	3
3. Apresentação dos Exercícios	4
4. Respostas dos exercícios	5
5. Conclusões	26
6. Referências Bibliográficas	27

## Introdução

---

Na aula do tópico 01, referente à introdução da matéria, ministrada no dia 17 de setembro de 2020, o docente solicitou aos estudantes uma atividade que comportasse todas as informações explicadas, de tal forma que fosse entregue um relatório conciso na semana seguinte (23/09/2020). Assim, essa atividade deveria ser realizada individualmente no formato de um relatório, conforme explicitada como pré-requisito.

Outras exigências foram:

- O relatório deve ser claro e completo (ou seja, a resposta deve estar correta e completa e o texto deve poder ser sempre entendido por uma pessoas que não nunca viram a atividade);
- O arquivo deve ser entregue em formato PDF;
- Esse documento em PDF será submetido na plataforma Moodle.

Todo o restante do relatório se caracteriza em quatro segmentos necessários: **i) Introdução**, onde existe a breve exposição do hodierno trabalho e sua metodologia; **ii) Apresentação dos Exercícios**, parte que constarão os exercícios propostos pelo o docente; **iii) Resposta dos Exercícios**, em que se terá as respostas dos exercícios; **iv) Conclusões**, onde se espera trazer inferências sobre o tema que fora estudado e, por último **v) Referências Bibliográficas**, segmento aberto às citações oriundas, dos trabalhos que serviram de base.

## Apresentação dos Exercícios

---

1. Apresente (e explique brevemente) todos os códigos de status numéricos, com as respectivas descrições, que podem ser retornados em uma resposta HTTP/1.1.
2. O Chrome DevTools (ou “Ferramentas do Desenvolvedor”) é um conjunto de ferramentas de autoria e depuração nativamente incorporado ao navegador Chrome. Consulte a documentação do Chrome DevTools [1] e apresente uma descrição de todos os painéis disponíveis pela ferramenta.
3. Com o auxílio das Chrome DevTools, acesse o endereço <http://www.ft.unicamp.br> e responda:
  - a. Quantas requisições são feitas ao servidor, até que a página seja exibida na íntegra?
  - b. Todas as requisições retornam status 200 (OK)? Caso não, indique qual requisição retorna um código diferente, qual é o código em questão e qual o seu significado.
  - c. Apresente todos os campos dos cabeçalhos (de requisição e resposta) que são trocados na requisição que retorna o documento principal da página.
  - d. Há alguma mensagem de erro ou de warning exibida no console? Se sim, quais?

## Respostas dos exercícios

---

1. Segue-se a tabela, cuja característica é apresentar os códigos e suas respectivas descrições dos erros[2]:

Status dos Códigos	Definições
100 Continue	Essa é uma resposta provisória e indica que tudo ocorreu bem até o momento e que o cliente deve continuar com a requisição ou ignorar se já conseguiu o que desejava.
101 Switching Protocols	Esse código é enviado em resposta a um cabeçalho de requisição Upgrade pelo o cliente, e indica o protocolo que o servidor está alternando.
200 OK	É bem comum de acontecer, mas os usuários não sabem que ocorreu. De um modo geral, significa que a requisição foi bem sucedida. O significado do sucesso varia de acordo com o método HTTP: <b>GET</b> : O recurso foi buscado e transmitido no corpo da mensagem. <b>HEAD</b> : Os cabeçalhos da entidade estão no corpo da mensagem. <b>POST</b> : O recurso descrevendo o resultado da ação e transmitido no corpo da mensagem. <b>TRACE</b> : O corpo da mensagem contém a mensagem de requisição recebida pelo servidor.
201 Created	A requisição foi bem sucedida e um novo recurso foi criado como resultado. Essa é uma típica resposta enviada após uma requisição POST.
202 Accepted	A requisição foi recebida, no entanto não houve nenhuma ação tomada sobre ela. Isto é uma requisição não-comprometedora, o que significa que não há nenhuma maneira no HTTP para enviar uma resposta assíncrona indicando o resultado do processamento da solicitação. Isto é indicado para casos onde outro processo

	ou servidor lida com a requisição, ou para processamento em lote.
203 Non-Authoritative Information	Esse código de resposta significa que o conjunto de meta-informações retornadas não é o conjunto exato disponível no servidor de origem, mas coletado de uma cópia local ou de terceiros. Exceto nessa condição, a resposta de 200 OK deve ser preferida em vez dessa resposta.
204 No Content	Não há conteúdo para enviar para esta solicitação, mas os cabeçalhos podem ser úteis. O user-agent pode atualizar seus cabeçalhos em cache para este recurso com os novos.
206 Partial Content	Esta resposta é usada por causa do cabeçalho de intervalo enviado pelo cliente para separar o download em vários fluxos.
300 Multiple Choices	A requisição tem mais de uma resposta possível. User-agent ou o user deve escolher uma delas. Não há maneira padrão para escolher uma das respostas.
301 Moved Permanently	Esse código de resposta significa que a URI do recurso requerido mudou. Provavelmente, a nova URI será especificada na resposta.
302 Found	Esse código de resposta significa que a URI do recurso requerido foi mudada temporariamente. Novas mudanças na URI poderão ser feitas no futuro. Portanto, a mesma URI deve ser usada pelo cliente em requisições futuras.
303 See Other	O servidor manda essa resposta para instruir ao cliente buscar o recurso requisitado em outra URI com uma requisição GET.
304 Not Modified	Essa resposta é usada para questões de cache. Diz ao cliente que a resposta não foi modificada. Portanto, o cliente pode usar a mesma versão em cache da resposta. Indica que não há necessidade de retransmitir a requisição de recursos.
305 Use Proxy	Foi definida em uma versão anterior da especificação HTTP para indicar que uma resposta deve ser acessada por um proxy. Foi

	depreciada por questões de segurança em respeito a configuração em banda de um proxy.
306 (Unused)	Esse código de resposta não é mais utilizado, portanto encontra-se reservado. Foi usado numa versão anterior da especificação HTTP 1.1.
307 Temporary Redirect	O servidor mandou essa resposta direcionando o cliente a buscar o recurso requisitado em outra URI com o mesmo método que foi utilizado na requisição original. Tem a mesma semântica do código 302 Found, com a exceção de que o user-agent não deve mudar o método HTTP utilizado: se um POST foi utilizado na primeira requisição, um POST deve ser utilizado na segunda.
400 Bad Request	Essa resposta significa que o servidor não entendeu a requisição pois está com uma sintaxe inválida.
401 Unauthorized	Embora o padrão HTTP especifique "unauthorized", semanticamente, essa resposta significa "unauthenticated". Ou seja, o cliente deve se autenticar para obter a resposta solicitada.
402 Payment Required	Este código de resposta está reservado para uso futuro (está despadronizado ainda). O objetivo inicial da criação deste código era usá-lo para sistemas digitais de pagamento, porém ele não está sendo usado atualmente.
403 Forbidden	O cliente não tem direitos de acesso ao conteúdo portanto o servidor está rejeitando dar a resposta. Diferente do código 401, aqui a identidade do cliente é conhecida. Indica que o servidor entendeu o pedido, mas se recusa a autorizá-lo.
404 Not Found	O servidor não pôde encontrar o recurso requisitado. Este código de resposta talvez seja o mais famoso devido à frequência com que acontece na web.
405 Method Not Allowed	O método de solicitação é conhecido pelo servidor, mas foi desativado e não pode ser usado. Os dois métodos obrigatórios, GET e



	HEAD, nunca devem ser desabilitados e não devem retornar este código de erro.
406 Not Acceptable	Essa resposta é enviada quando o servidor da Web após realizar a negociação de conteúdo orientada pelo servidor, não encontra nenhum conteúdo seguindo os critérios fornecidos pelo agente do usuário.
407 Proxy Authentication Required	Semelhante ao 401 porém é necessário que a autenticação seja feita por um proxy.
408 Request Timeout	Esta resposta é enviada por alguns servidores em uma conexão ociosa, mesmo sem qualquer requisição prévia pelo cliente. Ela significa que o servidor gostaria de derrubar esta conexão em desuso. Esta resposta é muito usada já que alguns navegadores, como Chrome, Firefox 27+, ou IE9, usam mecanismos HTTP de pré-conexão para acelerar a navegação. Note também que alguns servidores meramente derrubam a conexão sem enviar esta mensagem.
409 Conflict	Esta resposta será enviada quando uma requisição conflitar com o estado corrente do servidor.
410 Gone	Esta resposta será enviada quando o conteúdo requisitado foi deletado do servidor.
411 Length Required	O servidor rejeitou a requisição porque o campo Content-Length do cabeçalho não está definido e o servidor o requer.
412 Precondition Failed	O cliente indicou nos seus cabeçalhos pré-condições que o servidor não atende.
413 Request Entity Too Large	A entidade requisição é maior do que os limites definidos pelo servidor; o servidor pode fechar a conexão ou retornar um campo de cabeçalho Retry-After.
414 Request-URI Too Long	A URI requisitada pelo cliente é maior do que o servidor aceita para interpretar.
415 Unsupported Media Type	O formato de mídia dos dados requisitados não é suportado pelo servidor, então o servidor rejeita a requisição.

416 Requested Range Not Satisfiable	O trecho especificado pelo campo Range do cabeçalho na requisição não pode ser preenchido; é possível que o trecho esteja fora do tamanho dos dados da URI alvo.
417 Expectation Failed	Este código de resposta significa que a expectativa indicada pelo campo Expect do cabeçalho da requisição não pode ser satisfeita pelo servidor.
500 Internal Server Error	O servidor encontrou uma situação com a qual não sabe lidar.
501 Not Implemented	O método da requisição não é suportado pelo servidor e não pode ser manipulado. Os únicos métodos exigidos que servidores suportem (e portanto não devem retornar este código) são GET e HEAD.
502 Bad Gateway	Esta resposta de erro significa que o servidor, ao trabalhar como um gateway a fim de obter uma resposta necessária para manipular a requisição, obteve uma resposta inválida.
503 Service Unavailable	O servidor não está pronto para manipular a requisição. Causas comuns são um servidor em manutenção ou sobrecarregado. Note que junto a esta resposta, uma página amigável explicando o problema deveria ser enviada. Estas respostas devem ser usadas para condições temporárias e o cabeçalho HTTP Retry-After: deverá, se possível, conter o tempo estimado para recuperação do serviço. O webmaster deve também tomar cuidado com os cabeçalhos relacionados com o cache que são enviados com esta resposta, já que estas respostas de condições temporárias normalmente não deveriam ser postas em cache.
504 Gateway Timeout	Esta resposta de erro é dada quando o servidor está atuando como um gateway e não obtém uma resposta à tempo.
505 HTTP Version Not Supported	A versão HTTP usada na requisição não é suportada pelo servidor.

2. Segundo a documentação do Chrome DevTools [1], *“ele é um conjunto de ferramentas de autoria e depuração de Web incorporado ao Google Chrome. Use o DevTools para iterar, depurar e criar o perfil do seu site”*. Destarte, são apresentados a seguir, os principais conceitos atrelados a essa ferramenta. Além disso, vamos utilizar o site da UNICAMP [3] para oferecer exemplos e guiar nosso conteúdo.

**Device Mode:** é utilizado com o objetivo de visualizar de que forma a página se vai estruturar e como funcionará em um dispositivo móvel. É Um modo de emulação. Inclusive, não são necessárias execuções de código. Simula-se a interação humano-computador (notebook ou desktop), com uma usabilidade acessível e até gerenciável. Mas, nem sempre é possível simular todos os aspectos. Um desses exemplos é a arquitetura das CPUs de dispositivos móveis que são muito diferentes das arquiteturas das CPUs de um computador. É possível pensar no Device Mode como uma aproximação de como sua página estará em um dispositivo móvel. Basicamente, o Device Mode é o nome que referencia o conjunto das ferramentas avulsas do Chrome DevTools que auxilia na simulações de dispositivos móveis. Os recursos simulados são:

- ❑ **Simulação de uma janela de visualização móvel**, que está muito atrelada à Modo de janela de visualização responsiva (torna a janela de visualização dimensionável e atraente, além, das alças grandes). Veja abaixo:



**Figura 01 - Visualização de um dispositivo móvel**

❑ Controle de rede, em que é possível acessar o painel **Network** e eleger

**Fast 3G** ou **Slow 3G** na lista **Throttle**. Veja um exemplo:

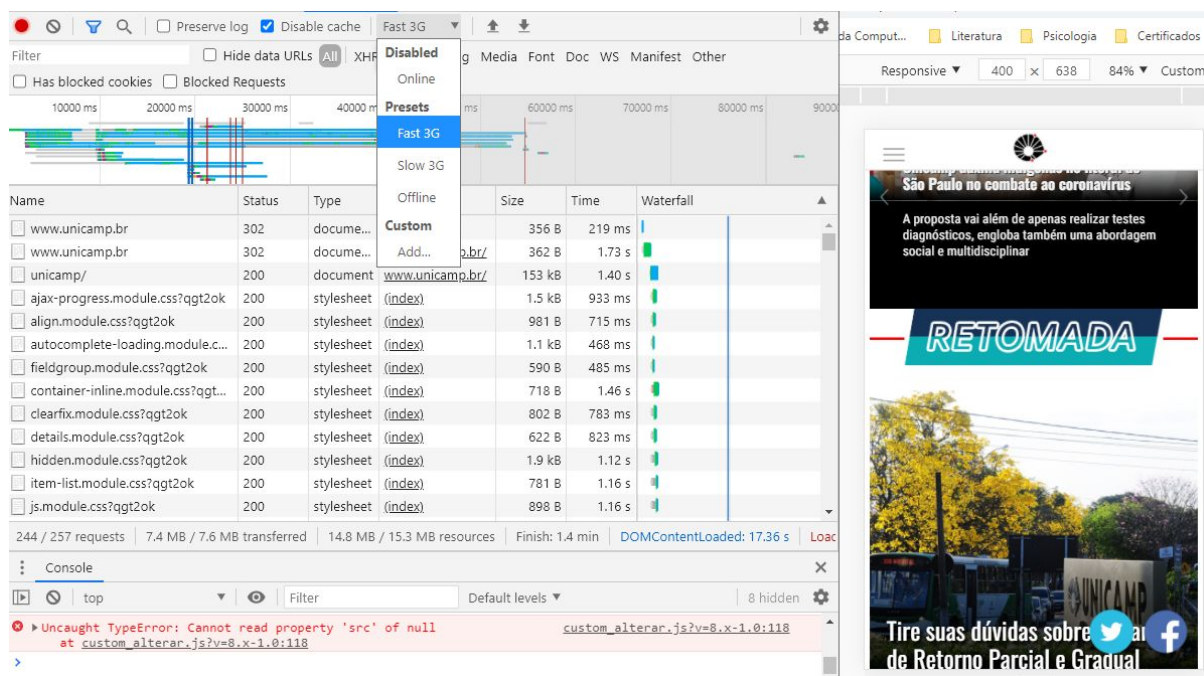


Figura 02 - Controle de Rede

❑ Controle de CPU, que serve para limitar somente a CPU e não a rede.

Acesse o painel **Performance**, clique em **Capture Settings** e selecione **4x slowdown** ou **6x slowdown** na lista **CPU**.

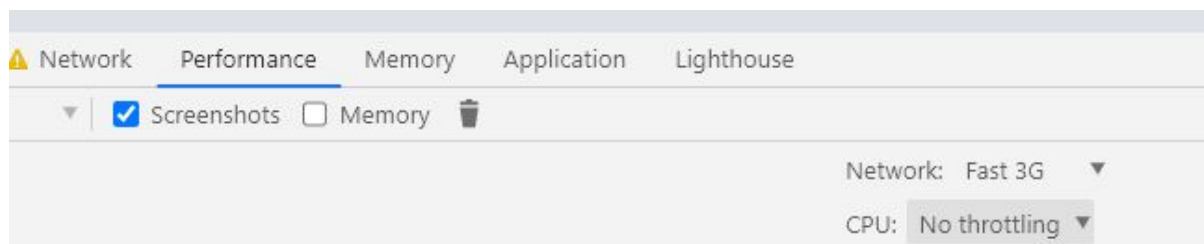
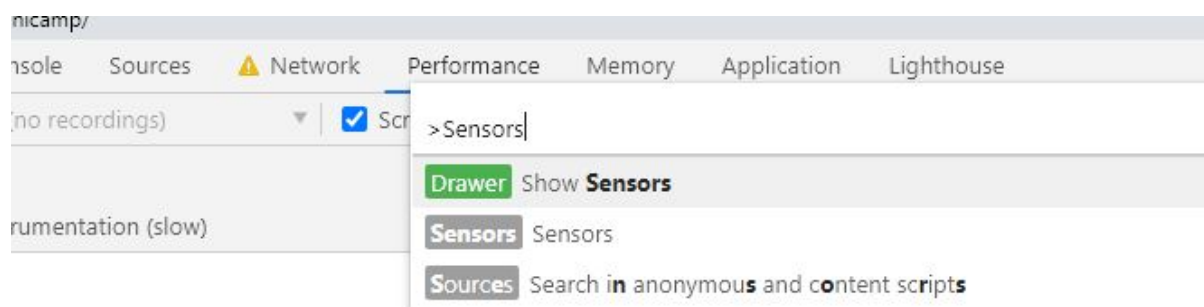
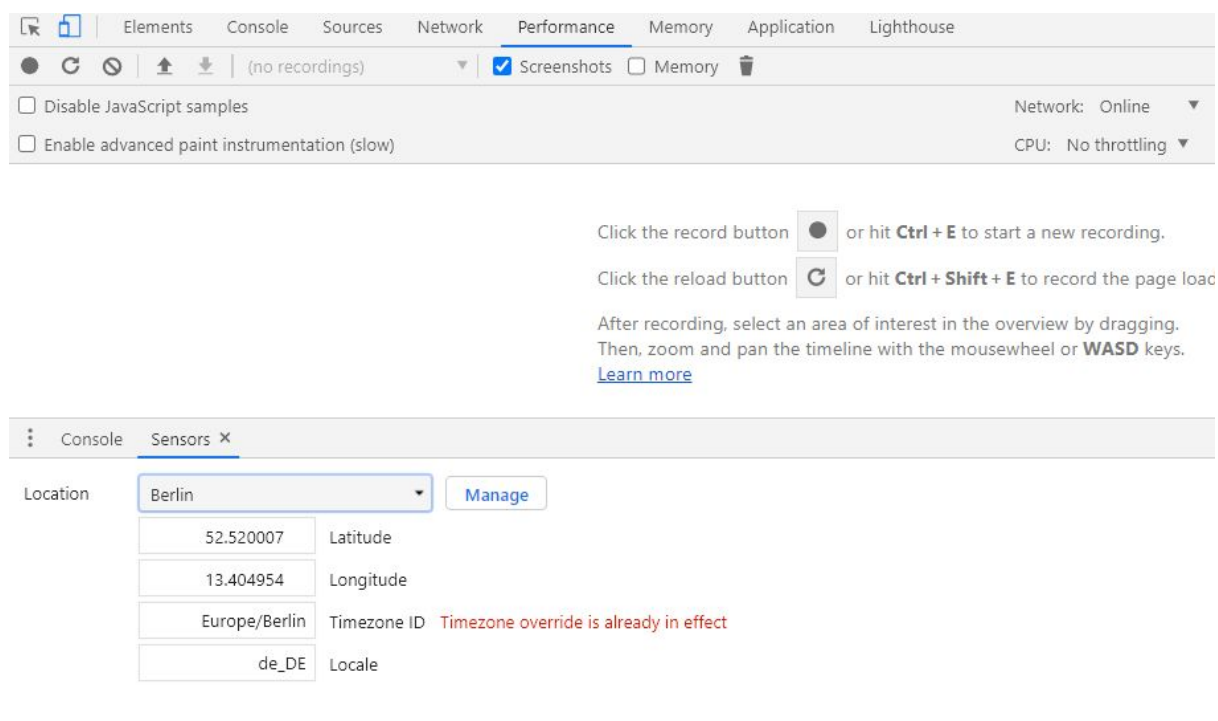


Figura 03 - Controle de CPU

- ❑ Simulação de geolocalização, em que gerencia a localização. Para abrir a IU de modificação da geolocalização, clique em **Customize and control DevTools** e selecione **More tools > Sensors**.

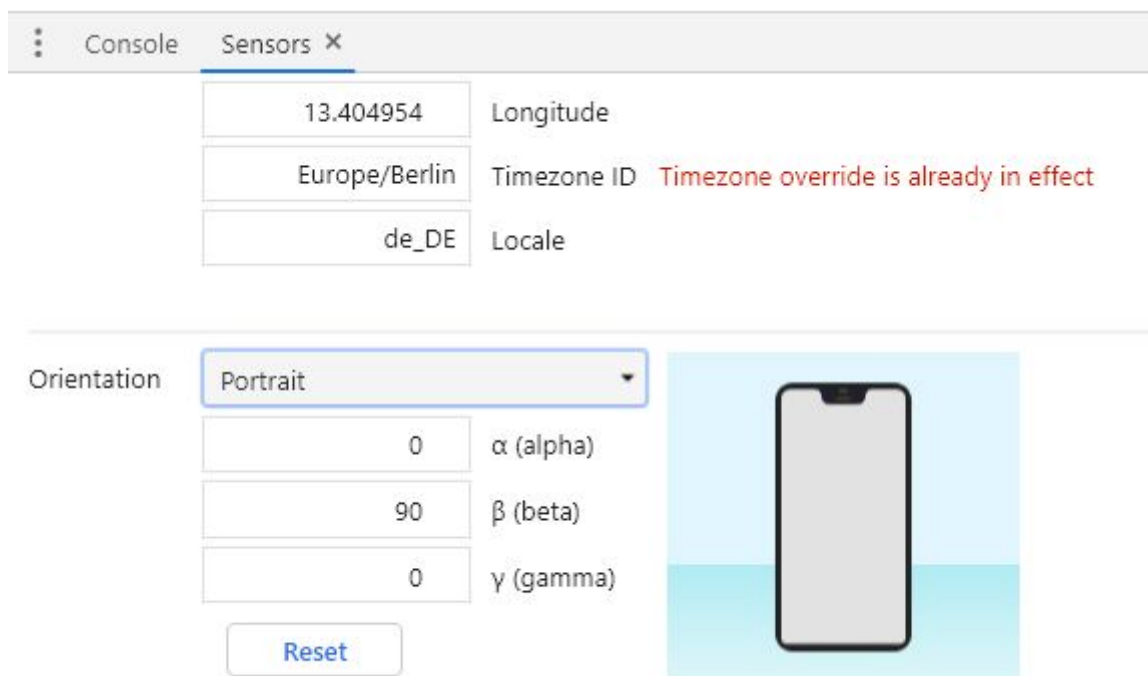


**Figura 04 - Ícone Sensors**



**Figura 05 - Painel de Geolocalização**

- ❑ Definição de orientação, em que define o formato da tela. Para abrir a IU de orientação, clique em **Customize and control DevTools**. Depois, em inline-icon e selecione **More tools > Sensors**.



**Figura 06 - Definição de Orientação**

Assim, encerra-se o ambiente Device Mode. Vamos às explicações dos outros elementos dentro do Chrome DevTools:

Elements: É usado para inspecionar e editar em tempo real o HTML e o CSS de uma página usando o painel Elements. É interessante analisar esse ambiente, pois permite fazer mudanças em tempo real na página, sem alterar o código do servidor web. Então, é viável brincar por aqui! Sendo possível também visualizar e alterar as regras de CSS aplicadas a qualquer elemento no painel Styles.



Inclusive, podemos visualizar e editar o modelo de caixa de um elemento selecionado no painel Computed e por fim, visualizar todas as mudanças feitas na sua página localmente no painel Sources. Há uma série de atividades a serem feitas por aqui. Um exemplo é na página da UNICAMP:

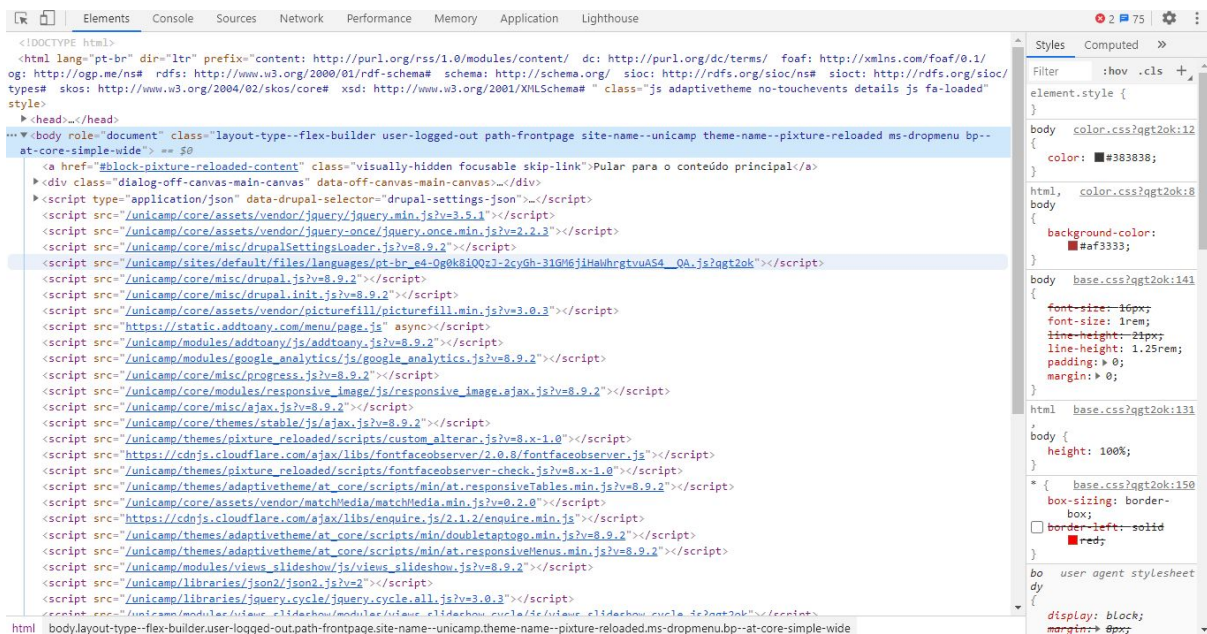
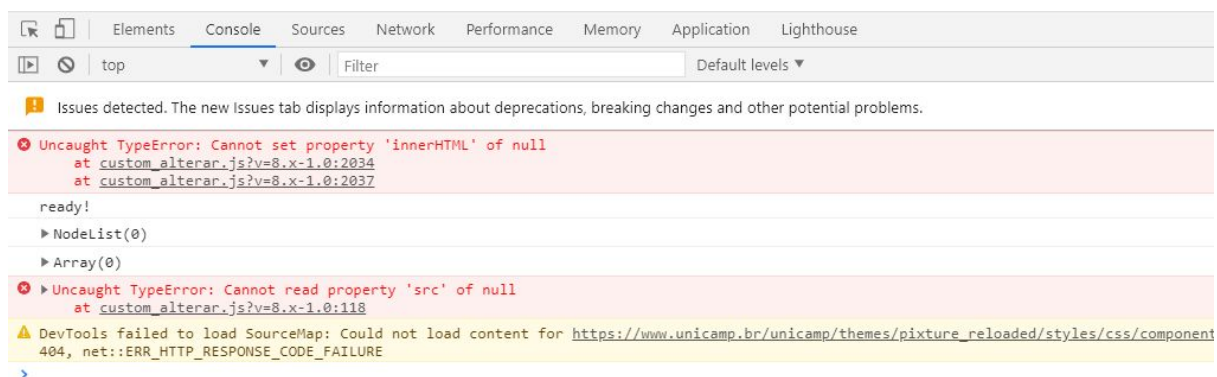


Figura 07 - Elements

**Console:** É usado principalmente com o objeto de registrar as informações durante o desenvolvimento, sendo um diagnóstico da página. Assim como é responsável por evidenciar as mensagens redundantes ou exibi-las nas próprias linhas, apagar ou forçar uma saída ou salvá-la em um arquivo, filtrar a saída e acessar configurações adicionais do Console. O console também permite criar algumas notificações de alerta no navegador e até brincar com o comando console.log. Vejamos:

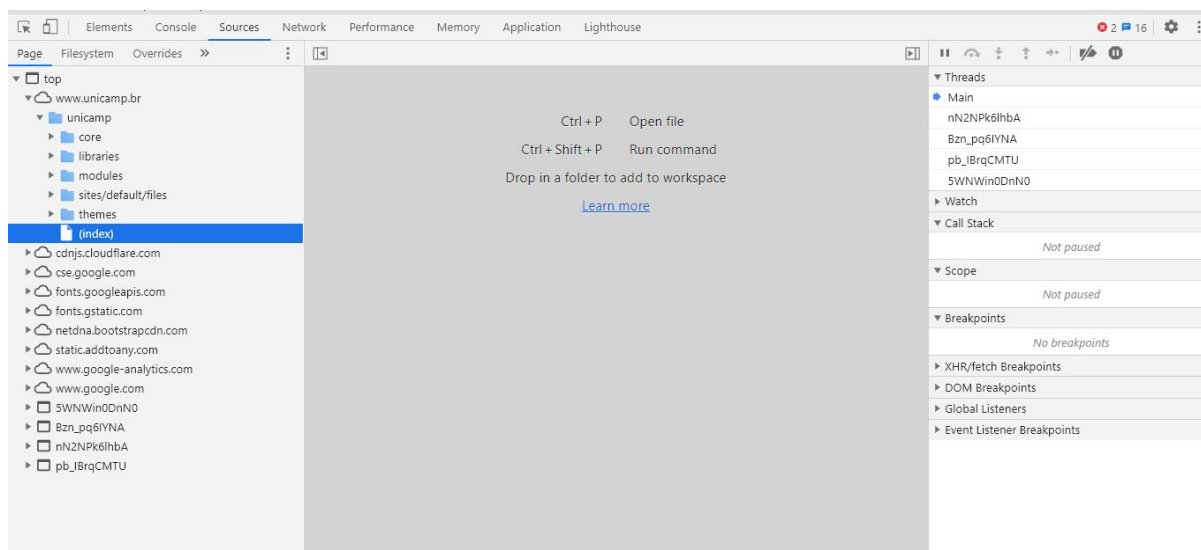




**Figura 08 - Console**

**Sources:** Neste painel é possível depurar o JavaScript usando pontos de interrupção no painel Sources ou conectar os arquivos locais por meio dos locais de trabalho para usar o editor em tempo real do DevTools. Portanto, é possível fazer alterações no DevTools dentro da página. Vemos que nesse espaço é possível verificar como os diretórios estão subdivididos, além do servidor web.

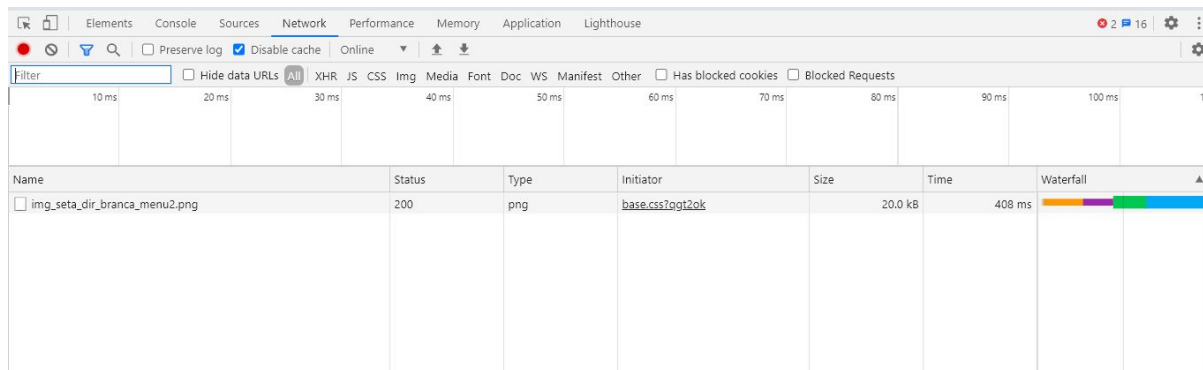
Veja-se:



**Figura 09 - Sources**

**Network:** É um indicador de desempenho da rede do site usando o painel network. Visualiza-se em tempo real o registro de informações sobre cada

operação de rede em uma página, incluindo dados detalhados de tempo, solicitação HTTP e cabeçalhos de resposta, cookies e muito mais. Podemos ver as imagens, por exemplo, carregando e o status atrelado essa figura. Vejamos:



**Figura 10 - Network**

**Timeline:** É usada para melhorar o desempenho da página, registra e analisa todas as atividades no seu aplicativo enquanto ele está em execução. O painel Timeline é composto por quatro seções: Controls, Overview, Flame Chart, Details.

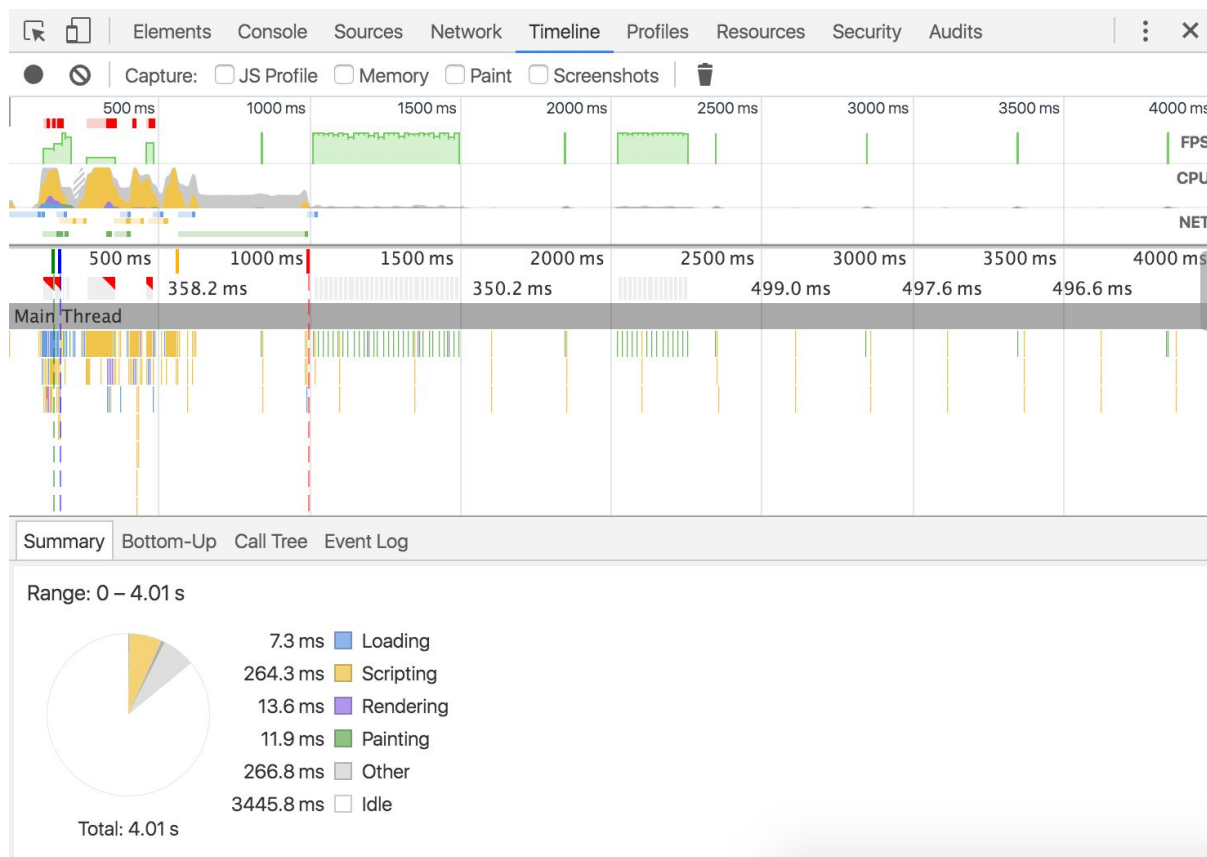


Figura 11 - Timeline[4]

Profiles/Memory: Identifica funções pesadas usando o criador de perfis de CPU do Chrome. Registra exatamente funções que foram chamadas e quanto tempo cada uma levou com o criador de perfis de CPU. Visualiza seus perfis como um diagrama de chamadas.

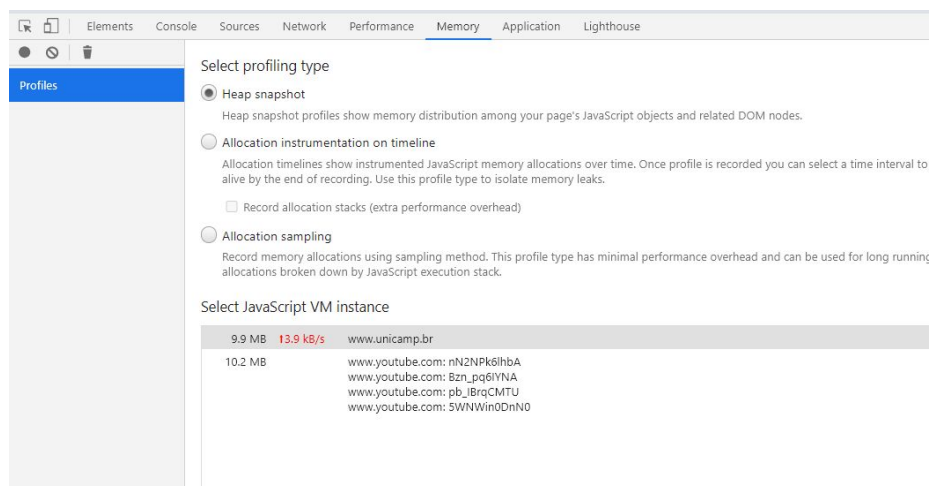
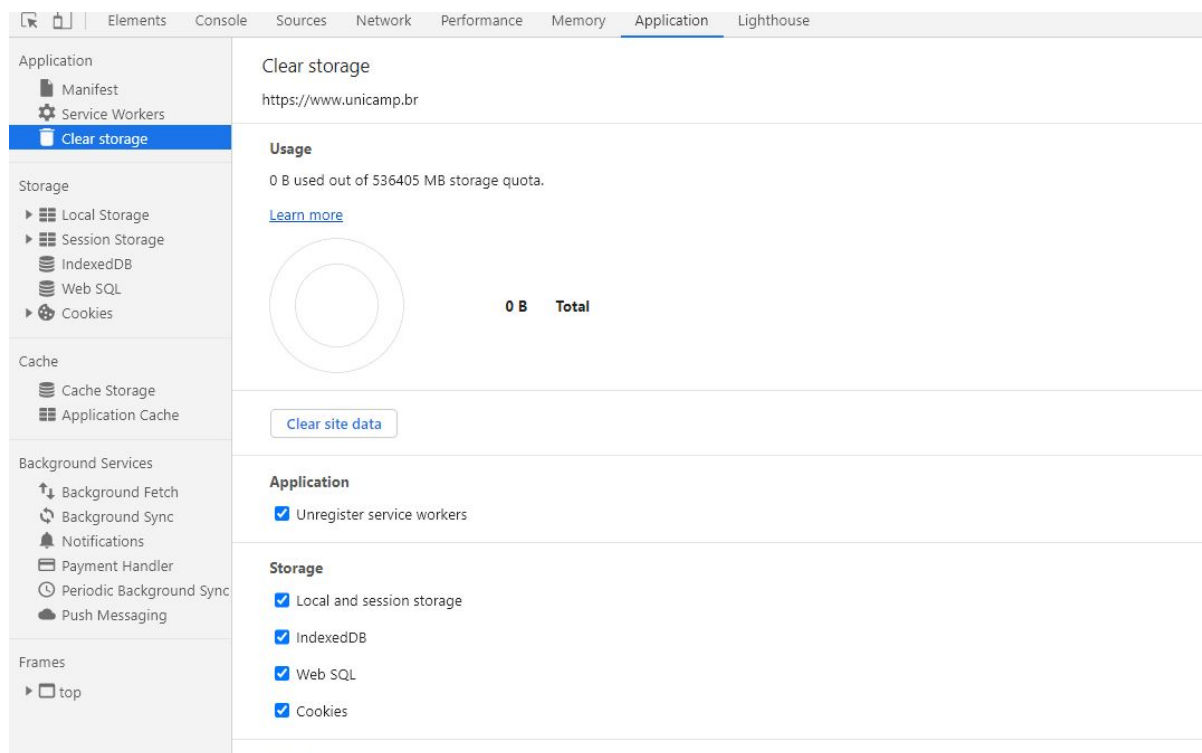


Figura 12 - Profiles

	Self	Total	Function
Profiles	4447.3 ms	4447.3 ms	(idle)
CPU PROFILES	2162.6 ms 6.61 %	2165.4 ms 6.62 %	▶ montReduce crypto.js:583
Profile 1 Save	1951.8 ms 5.97 %	1951.8 ms 5.97 %	(garbage collector)
	1643.9 ms 5.02 %	1652.8 ms 5.05 %	▶ lin_solve navier-stokes.js:152
	1476.7 ms 4.51 %	1964.1 ms 6.00 %	▶ Scheduler.schedule richards.js:188
	1271.8 ms 3.89 %	1271.8 ms 3.89 %	(program)
	1170.8 ms 3.58 %	1172.0 ms 3.58 %	▶ bnpSquareTo crypto.js:431
	987.9 ms 3.02 %	1081.7 ms 3.31 %	▶ GeneratePayloadTree splay.js:50
	884.5 ms 2.70 %	2269.5 ms 6.94 %	▶ a8 (program):1
	763.5 ms 2.33 %	837.0 ms 2.56 %	▶ one_way_unify1_nboyer earley-boyer.js:3635
	720.7 ms 2.20 %	720.7 ms 2.20 %	▶ a6 (program):1
	682.6 ms 2.09 %	1577.2 ms 4.82 %	▶ rewrite_nboyer earley-boyer.js:3604
	624.5 ms 1.91 %	624.5 ms 1.91 %	▶ SplayTree.splay_ splay.js:322
	619.2 ms 1.89 %	846.0 ms 2.59 %	▶ Exec regexp.js:1
	558.0 ms 1.71 %	795.0 ms 2.43 %	▶ (anonymous function) code-load.js:1518
	540.0 ms 1.65 %	540.4 ms 1.65 %	▶ Plan.execute deltablue.js:776
	517.8 ms 1.58 %	799.7 ms 2.44 %	▶ (anonymous function) code-load.js:1541
	458.2 ms 1.40 %	1348.3 ms 4.12 %	▶ loop2 earley-boyer.js:4272
	402.6 ms 1.23 %	402.8 ms 1.23 %	▶ HandlerTask.run richards.js:465
	320.8 ms 0.98 %	582.2 ms 1.78 %	▶ Constraint.satisfy deltablue.js:175
	312.6 ms 0.96 %	1333.4 ms 4.08 %	▶ loop3 earley-boyer.js:4286
	301.8 ms 0.92 %	1348.7 ms 4.12 %	▶ sc_loop1_98 earley-boyer.js:4258
	274.2 ms 0.84 %	1349.6 ms 4.12 %	▶ deriv_trees earley-boyer.js:4254

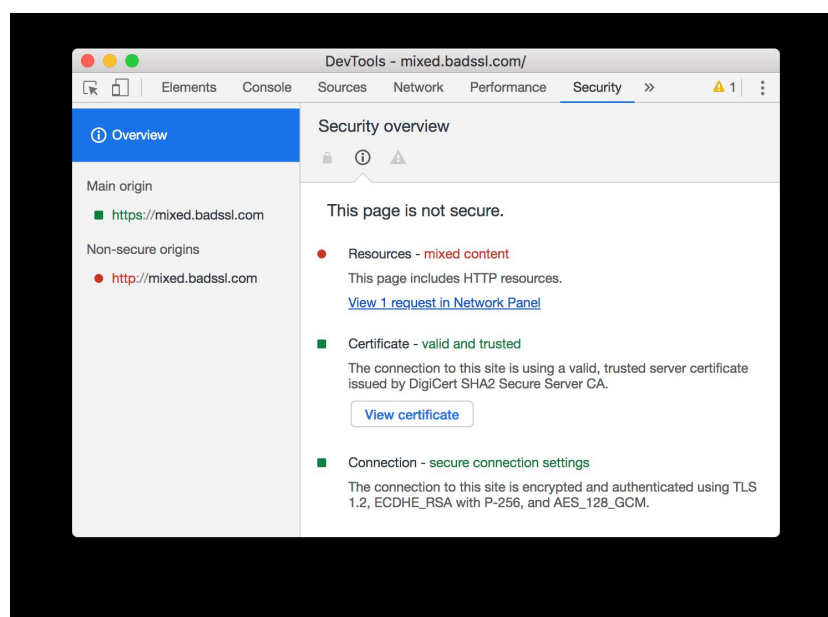
Figura 13 - Exemplo do DevTools[5]

**Application:** O painel Resources é usado para inspecionar todos os recursos carregados, incluindo bancos de dados Web SQL ou IndexedDB, armazenamento local e da sessão, cookies, cache do aplicativo, imagens, fontes e folhas de estilo. Podemos ir buscar todos os elementos que quisermos, como nos Frames, um documento em HTML ou algum Banco de Dados.



**Figura 14 - Application**

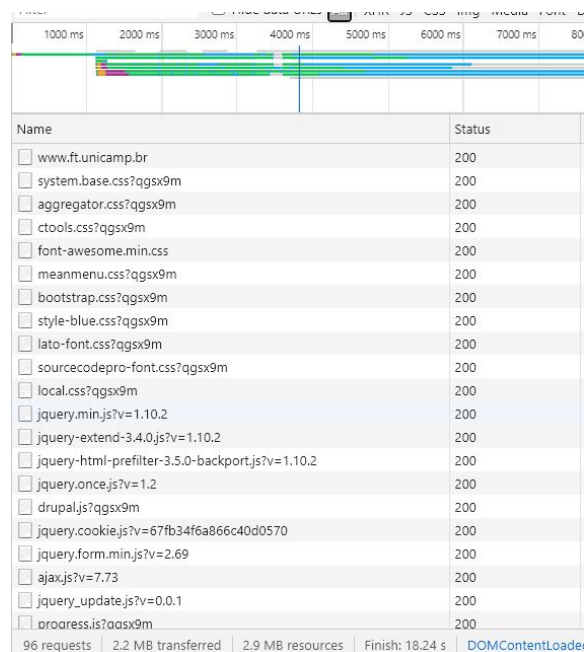
**Security:** Usado para garantir que o HTTPS seja implementado corretamente em uma página. O HTTPS é importante para fornecer segurança e integridade de dados fundamentais para o site e as informações pessoais dos usuários. É possível ver o exemplo abaixo:



**Figura 15 - Security no DevTools [6]**

3. Todas as respostas estão abaixo:

- a. Foram feitas 96 requisições, no Google Chrome, no Sistema Operacional Windows 10, conforme a imagem abaixo:



**Figura 16 - Requisições no Chrome**

Entretanto, a fim de teste, busquei as requisições também através do Sistema Operacional Ubuntu, no Firefox. Retornaram-se 98 requisições:



Style sheet could not be

Inspetor






















Filter URLs

Status	Method	D
200	GET	
200	GET	
304	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
200	GET	
304	GET	
200	GET	

98 requests 3,08 MB

**Figura 16 - Requisições no Firefox**

- b. Sim, todas obtiveram status OK no Google Chrome.

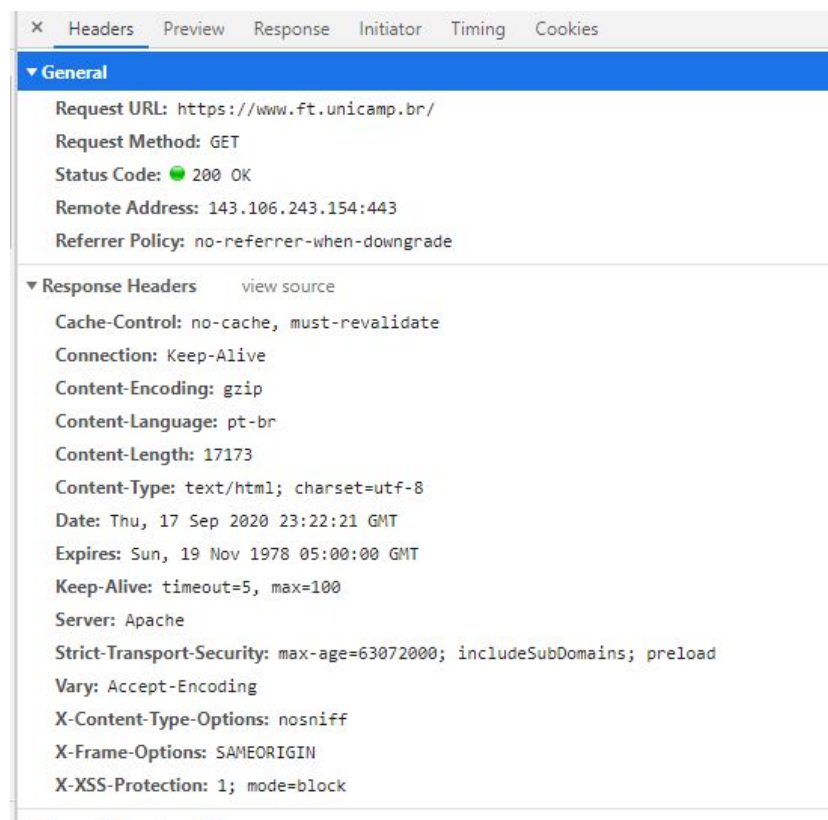
Name	Status
 www.ft.unicamp.br	200
 system.base.css?qgsx9m	200
 aggregator.css?qgsx9m	200
 ctools.css?qgsx9m	200
 font-awesome.min.css	200
 meanmenu.css?qgsx9m	200
 bootstrap.css?qgsx9m	200
 style-blue.css?qgsx9m	200
 lato-font.css?qgsx9m	200
 sourcecodepro-font.css?qgsx9m	200
 local.css?qgsx9m	200
 jquery.min.js?v=1.10.2	200
 jquery-extend-3.4.0.js?v=1.10.2	200
 jquery-html-prefilter-3.5.0-backport.js?v=1.10.2	200
 jquery.once.js?v=1.2	200
 drupal.js?qgsx9m	200
 jquery.cookie.js?v=67fb34f6a866c40d0570	200
 jquery.form.min.js?v=2.69	200
 ajax.js?v=7.73	200
 jquery_update.js?v=0.0.1	200
 progress.js?qgsx9m	200

**Figura 17 - Status no Google Chrome**

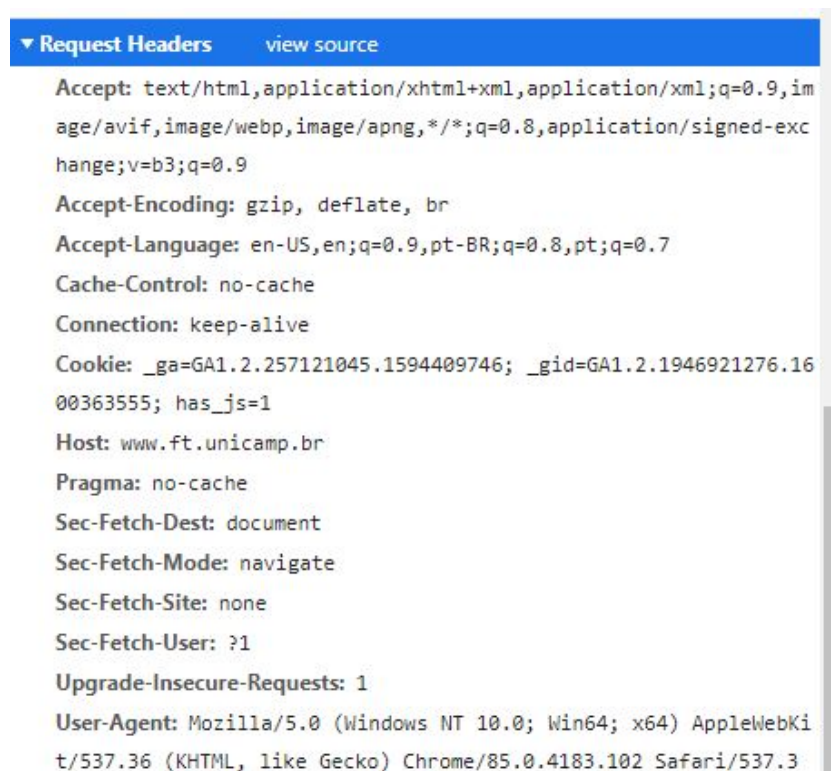
Porém, no Firefox, obtiveram alguns com Status 304. Ao todo, foram 32 requisições com o status 304, que, segundo o site Developer Mozilla [7], “indica que não há necessidade de retransmitir a requisição de recursos. É um redirecionamento implícito para o recurso em cache. Isto ocorre quando o método de requisição é safe, assim como uma requisição GET ou HEAD, ou quando a requisição é condicional e usa um cabeçalho If-None-Match ou If-Modified-Since”.







**Figura 19 - Response Headers**



**Figura 20 - Request Headers**

d. Sim, existem mensagens de erro no Console:

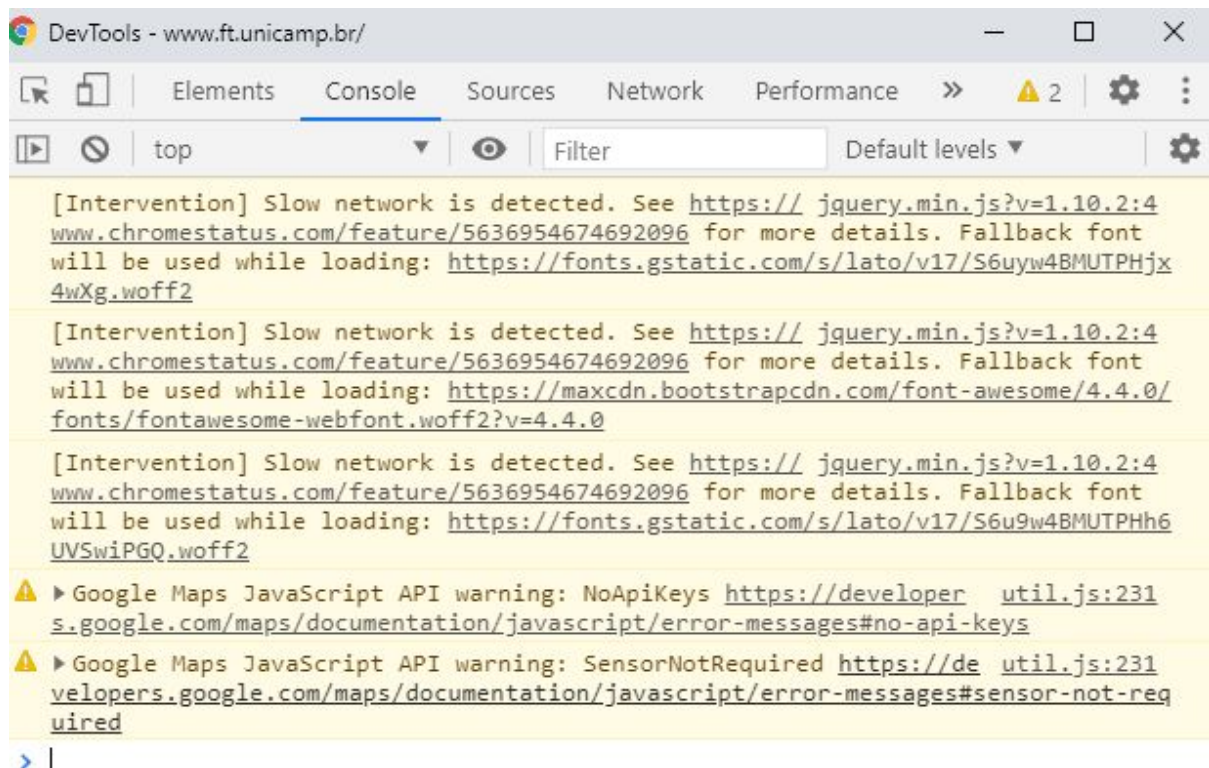


Figura 21 - Warnings

## Conclusões

---

Foi uma atividade interessante, pois incentivou a(o) estudante a buscar alguns dos principais status de erro HTTP que lhe são acometidos durante a navegação web e, portanto, conhecer alguns elementos das páginas, e como estão demonstradas, por “baixo dos panos”, foi muito instigante.

## Referências Bibliográficas

---

- [1]. **Chrome DevTools | Google Developers.** Disponível em: <<https://developers.google.com/web/tools/chrome-devtools/>>. Acesso em: 17 set. 2020.
- [2]. **RFC 7231 - Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content.** Disponível em: <<https://tools.ietf.org/html/rfc7231#section-6.5.1>>. Acesso em: 17 set. 2020.
- [3]. **Portal Unicamp | Unicamp.** Disponível em: <<https://www.unicamp.br/unicamp/>>. Acesso em: 17 set. 2020.
- [4]. **Como usar a ferramenta Timeline | Chrome DevTools | Google Developers.** Disponível em: <<https://developers.google.com/web/tools/chrome-devtools/evaluate-performance/timeline-tool?hl=pt-br>>. Acesso em: 20 set. 2020.
- [5]. **Profiles Panel - Google Chrome.** Disponível em: <<https://developer.chrome.com/devtools/docs/profiles>>. Acesso em: 20 set. 2020.
- [6]. **Understand Security Issues With Chrome DevTools | Google Developers.** Disponível em: <<https://developers.google.com/web/tools/chrome-devtools/security>>. Acesso em: 20 set. 2020.

- [7]. **304 Not Modified - HTTP | MDN.** Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Status/304>>. Acesso em: 20 set. 2020.