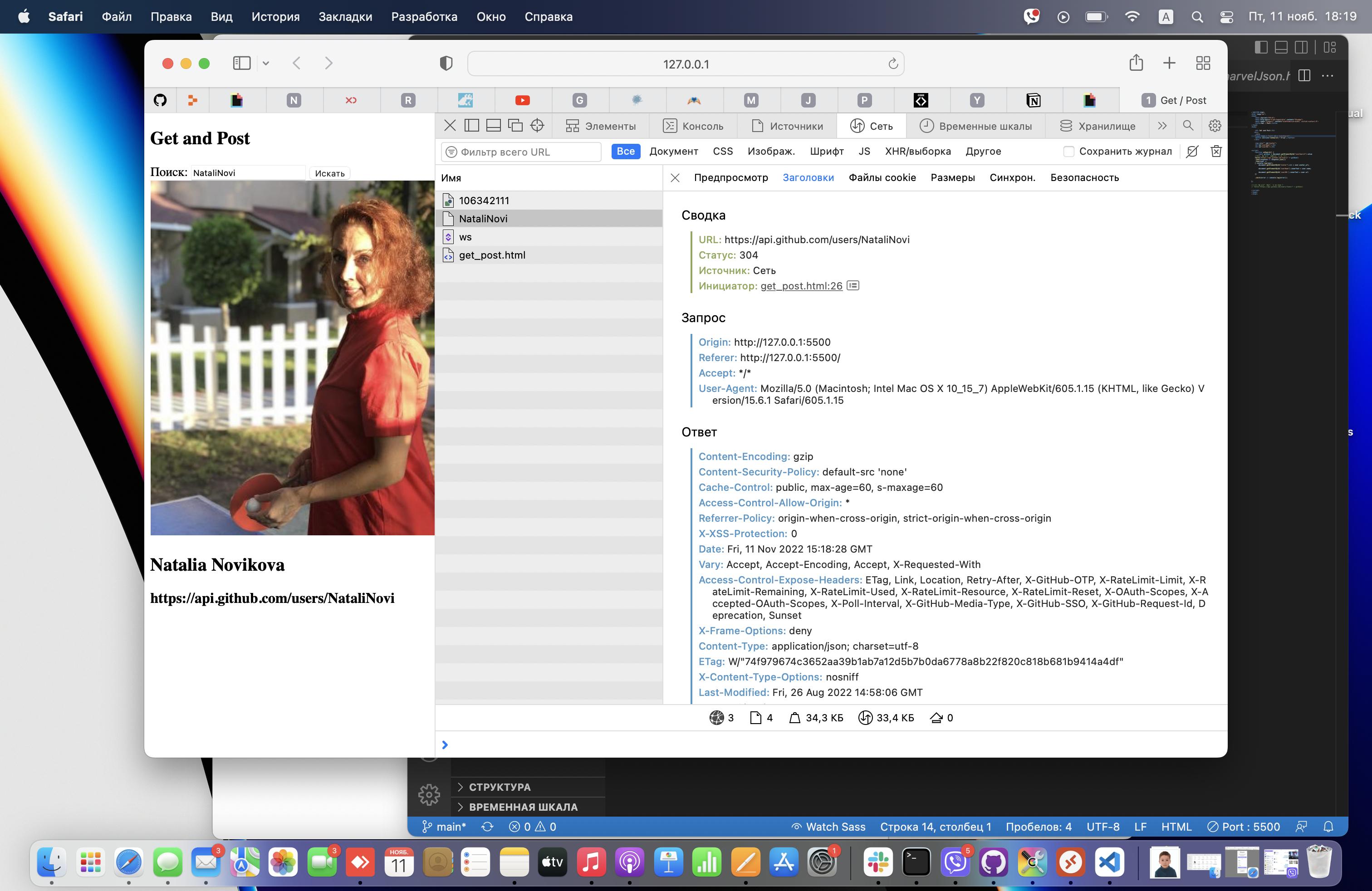
Week 21  
  
1.

<https://www.21vek.by/cushioned_furniture/?filter%5Bproducer%5D%5B%5D=mio_tesoro&filter%5B30199%5D%5B%5D=403183&filter%5Bsa%5D=>

<https://catalog.onliner.by/mobile?mfr%5B0%5D=apple>

2. GET-запросы используются для отправки и получения не секретной информации небольшого объема. Гет-запрос отправляется в виде адреса URL и содержит в себе параметр имя-ключ. Гет запрос сохраняется в памяти (истории просмотров).  
POST-запросы передают информацию не открыто, а внутри заголовков, поэтому могут быть использованы для передачи и получения конфиденциальной информации большого объема. Пост-запросы не сохраняются в памяти, поэтому каждый раз отправляя запрос на одного и того же пользователя, мы будет получать разные ответы (данные).

3. GET  
  
4. POST  
  
5. api\_key - API-ключ, получаемый при регистрации  
 query - текст поискового запроса  
 limit - макс количество возвращаемых результатов

Lang - язык  
6. Через панель разработчика - сеть - заголовки

НО У МЕНЯ ВОПРОС:  
Что конкретно из этого заголовки?  
  
  
ЭТО?  
Access-Control-Expose-Headers: ETag, Link, Location, Retry-After, X-GitHub-OTP, X-RateLimit-Limit, X-RateLimit-Remaining, X-RateLimit-Used, X-RateLimit-Resource, X-RateLimit-Reset, X-OAuth-Scopes, X-Accepted-OAuth-Scopes, X-Poll-Interval, X-GitHub-Media-Type, X-GitHub-SSO, X-GitHub-Request-Id, Deprecation, Sunset

7. Cross-Origin Resource Sharing ([CORS](https://developer.mozilla.org/ru/docs/Glossary/CORS)) — механизм, использующий дополнительные [HTTP](https://developer.mozilla.org/ru/docs/Glossary/HTTP)-заголовки, чтобы дать возможность [агенту пользователя](https://developer.mozilla.org/ru/docs/Glossary/User_agent) получать разрешения на доступ к выбранным ресурсам с сервера на источнике (домене), отличном от того, что сайт использует в данный момент. Говорят, что агент пользователя делает запрос с другого источника **(cross-origin HTTP request),** если источник текущего документа отличается от запрашиваемого ресурса доменом, протоколом или портом.

Пример cross-origin запроса: HTML страница, обслуживаемая сервером с http://domain-a.com, запрашивает [<img> src](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img#attributes) по адресу http://domain-b.com/image.jpg. Сегодня многие страницы загружают ресурсы вроде CSS-стилей, изображений и скриптов с разных доменов, соответствующих разным сетям доставки контента (Content delivery networks, CDNs).

В целях безопасности браузеры ограничивают cross-origin запросы, инициируемые скриптами. Например, [XMLHttpRequest](https://developer.mozilla.org/ru/docs/Web/API/XMLHttpRequest) и [Fetch API](https://developer.mozilla.org/ru/docs/Web/API/Fetch_API) следуют политике одного источника ([same-origin policy](https://developer.mozilla.org/ru/docs/Web/Security/Same-origin_policy)). Это значит, что web-приложения, использующие такие API, могут запрашивать HTTP-ресурсы только с того домена, с которого были загружены, пока не будут использованы CORS-заголовки.  
  
8. **Медиа тип** (так же известный как **Multipurpose Internet Mail Extensions** или **MIME тип)** является стандартом, который описывает природу и формат документа, файла или набора байтов. Он определён и стандартизирован в спецификации [RFC 6838](https://datatracker.ietf.org/doc/html/rfc6838) .

Организация [Internet Assigned Numbers Authority (IANA)](https://www.iana.org/) является ответственной за все официально признанные MIME типы, и вы можете найти самый последний и полный лист MIME типов на их странице [Медиа Типов](https://www.iana.org/assignments/media-types/media-types.xhtml).

**Предупреждение:** **Важно:** Для принятия решения о том, как обрабатывать URL, браузеры используют MIME типы, а не расширения файлов, так что серверам необходимо отправлять правильные MIME типы в [Content-Type](https://developer.mozilla.org/ru/docs/Web/HTTP/Headers/Content-Type) заголовке ответа. При неточном задавании этого заголовка, браузеры с большой вероятностью будут неправильно интерпретировать и обрабатывать содержание файлов, из-за чего сайт будет работать неверно.

Структура MIME типа

Простейший MIME тип состоит из типа и подтипа — двух строк разделённых наклонной чертой (/), без использования пробелов.

тип/подтип

**Тип** представляет общую категорию, в которой находится тип данных, например video или text. **Подтип** же строго отождествляется с отдельным типом данных, представляемых данным MIME типом. Например, для MIME типа text, подтипы могут быть plain (простой текст), html ([HTML](https://developer.mozilla.org/ru/docs/Glossary/HTML) source code) или calendar (для iCalendar .ics).

Необязательный **параметр** может быть добавлен для указания дополнительных деталей

тип/подтип;параметр=значение

Например, для MIME типов категории text, необязательный параметр charset может быть задан для уточнения кодировки, используемой в документе. Для объявления, что пересылаемый файл имеет кодировку UTF-8, необходимо использовать MIME тип text/plain;charset=UTF-8. При не указании параметра charset, его значение автоматически будет задано, как [ASCII](https://developer.mozilla.org/ru/docs/Glossary/ASCII) (US-ASCII), если в настройках браузера не будет определено иначе.

MIME типы являются нечувствительными к регистру, но традиционно их пишут строчными буквами, за исключением значений параметров.

9. let user = {

breed: "Beagle",

size: "large",

color: "orange",

age: 6 };

let response = await fetch('/localhost/pets/add', {

method: 'POST',

headers: {

'Content-Type': 'application/json;charset=utf-8'

},

body: JSON.stringify(user)

});

let result = await response.json();

alert(result.message);

</script>

10. При вызове промиса запускается функция, которая приводит либо к выполнению действия, либо к ошибке. Просим - связь между функцией и ожидаем ее результата.  
Что-то похожее было в get-запросах.

7.