- - - - - - Лабораторная 20 - - - - - -

# 1. Поясните понятие **Шаблонизация**.

Позволяет использовать html-шаблоны для генерации конечных html-страниц

# 2. Поясните понятие **Шаблон (template).**

**Шаблоны** обычные файлы HTML в формате handlebars, в которых с помощью специального синтаксиса выводятся передаваемые данные

# 3. Поясните понятие **Контекст шаблона (template context).**

**Контекст шаблона** содержит все переменные шаблона

# 4. Поясните понятие **Макет (layout).**

**Макет** оформление, которое пишем в отдельный файл и указываем, куда вставлять

# 5. Поясните понятие **Частичное представление (partial-view).**

**Частичные представления**, - разделяемые общие элементы, которые можно добавлять на обычные представления.

# 6. Поясните понятие **Хелпер** (**helper)**.

**Хелперы** функции, которые возвращают некоторую строку. После получения эту строку можно добавить в любое место представления

# 7. Перечислите встроенные хелперы Handlebars.

**If** Вывод блока по условию. Если аргумент этого хелпера вернет значение false, то блок внутри хелпера не будет выполнен

**Unless** Хелпер обратный хелперу if. Блок будет выведен, если аргумент вернет значение false.

**Each** Хелпер для перебора списков

**With** Позволяет перепривязать контекст

# 8. Поясните понятие **Статический файл**.

**Статика** - это файлы, которые лежат на сервере и отдаются пользователю в том виде, в каком есть (HTML, CSS, JS, изображения, другие файлы).

# 9. Поясните назначение http-заголовков: **Cache-control, ETag, Last-Modified**, **If**-**Modified-Since, If-None-Match, Expires.**

**Cache-Control** задания инструкций кэширования для запросов и ответов

**ETag** идентификатором специфической версии ресурса позволяет более эффективно использовать кэш, позволяя серверу отправлять не весь ответ, если содержимое не изменилось.

**Last-Modified** в ответе HTTP содержит время, когда запрашиваемый ресурс был изменен

**If-Modified-Since** указывает время, за которое браузер впервые загрузил ресурс с сервера

**If-None-Match** используется для обновления тегов сущностей на сервере

**Expires** содержит дату/время, по истечении которой ответ сервера считается устаревшим

# 10. Поясните свойство **\_\_dirname** объекта **global**.

**\_\_dirname** возвращает путь к папке, в которой находится текущий файл JavaScript.

# 11. Поясните назначение и принцип работы метода **response.download().**

Функция **res.download()** браузеры предлагают пользователю загрузить переданный файл

# 12. Поясните назначение и принцип работы метода **response.attachment().**

**«Content-Type»** будет автоматически установлен на основе расширения файла

заголовок **«Content-Disposition»** будет установлен на «filename= filename».

# 13. Поясните назначение http-заголовка: **Content-Disposition: attachment.**

указывает, что файл должен быть загружен; браузеры предоставляют диалоговое окно «Сохранить как», предварительно заполненное со значением параметров filename если они есть

- - - - - - Лабораторная 21 - - - - - -

**Passport** — это промежуточное ПО для проверки подлинности, которое используется для управления сессиями.

**Идентификация** — это заявление о том, кем вы являетесь. В зависимости от ситуации, это может быть имя, адрес электронной почты, номер учетной записи и т.д.

**Аутентификация** — предоставление доказательств, что вы на самом деле есть тот, кем идентифицировались (от слова authentic — истинный, подлинный).

**Авторизация** — проверка, что вам разрешен доступ к запрашиваемому ресурсу.

**BASIC**

Первое промежуточное ПО используется для проверки аутентификации клиента, когда сервер запускается и клиент вводит адрес локального хоста. Изначально req.headers.authorization не определено, и функция обратного вызова next() возвращает код состояния 401, несанкционированный доступ к браузеру. Клиент заполняет учетные данные, а учетные данные зашифрованы в формате base64. После этого он расшифровывает данные формата base64, которые содержат имя пользователя и пароль, затем, после проверки правильности имени пользователя и пароля, метод next() вызывает следующее промежуточное программное обеспечение, которое упоминается ниже промежуточного программного обеспечения аутентификации, в противном случае форма аутентификации появляется снова и снова. .

# **1. Перечислите все известные виды аутентификации. Какие из них описаны в RFC?**

**OAuth 2.0** (официально указан [RFC 6749](http://tools.ietf.org/html/rfc6749)) предоставляет структуру авторизации, которая позволяет пользователям авторизовать доступ к сторонним приложениям. После авторизации приложению выдается токен для использования в качестве учетных данных аутентификации.

**OAuth** (официально указан [RFC 5849](http://tools.ietf.org/html/rfc5849)) дает пользователям возможность предоставлять сторонним приложениям доступ к своим данным, не раскрывая свой пароль этим приложениям.

**Basic и Digest** (указаны в [RFC 2617](http://tools.ietf.org/html/rfc2617))

**Аутентификация по паролю** - пользователь должен предоставить username и password для успешной идентификации и аутентификации в системе

**Аутентификация по сертификатам** - cертификат представляет собой набор атрибутов, идентифицирующих владельца, подписанный certificate authority (CA).

**Аутентификация по одноразовым паролям** - обычно применяется дополнительно к аутентификации по паролям для реализации two-factor authentication (2FA).

**Аутентификация по ключам доступа** - этот способ чаще всего используется для аутентификации устройств

**Аутентификация по токенам** - такой способ аутентификации чаще всего применяется при построении распределенных систем Single Sign-On (SSO), где одно приложение делегирует функцию аутентификации пользователей другому приложению

# **2. Опишите схему BASIC-аутентификации.**

**Basic** — наиболее простая схема, при которой username и password пользователя передаются в заголовке Authorization в незашифрованном виде (base64-encoded). Однако при использовании HTTPS (HTTP over SSL) протокола, является относительно безопасной.

# **3. Опишите схему DIGEST-аутентификации.**

**дайджест-аутентификация** - при которой хэш информации о клиентах отправляется по каналу связи, что делает его более безопасным по сравнению с обычной аутентификацией.

**Digest** — сервер посылает уникальное значение nonce, а браузер передает MD5(хеширование с ключом для аутентификации сообщения) хэш пароля пользователя, вычисленный с использованием указанного nonce. Более безопасная альтернатив Basic схемы при незащищенных соединениях, но подвержена man-in-the-middle attacks (с заменой схемы на basic). Кроме того, использование этой схемы не позволяет применить современные хэш-функции для хранения паролей пользователей на сервере.

# **4. Почему в чистом виде http-аутентификация не является надежной?**

Данные пользователя(username, password) хранятся в незашифрованном виде.