1. Шаблонизация - это процесс создания или использования шаблонов для генерации текстового или графического содержимого. Она позволяет разделять структуру документа или приложения от его содержания, что облегчает его создание и поддержку.
2. Шаблон (template) - это файл или часть кода, содержащая заранее определенные элементы, которые могут быть заполнены или изменены в соответствии с конкретными данными или условиями. Шаблоны часто используются в веб-разработке для генерации HTML-страниц на основе данных, полученных из базы данных или других источников.
3. Контекст шаблона (template context) - это набор данных, доступных внутри шаблона для его заполнения. Он содержит переменные, функции и другие объекты, которые могут быть использованы для создания конечного вывода шаблона.
4. Макет (layout) - это основной шаблон, который определяет общую структуру или макет страницы. В него обычно включаются общие элементы, такие как заголовок, навигационное меню, подвал и т. д. Макет обычно используется для обеспечения единообразия в дизайне и структуре веб-страниц.
5. Частичное представление (partial-view) - это отдельный фрагмент шаблона, который может быть включен или подключен к другим шаблонам. Частичные представления используются для повторного использования кода и обеспечения модульности в веб-приложениях.
6. Хелпер (helper) - это функция или метод, который используется в шаблонах для выполнения определенных задач, таких как форматирование данных, выполнение логики или генерация специфического контента. Хелперы обычно предоставляются фреймворком или библиотекой шаблонизации.
7. Встроенные хелперы Handlebars:

* if - для выполнения условных операций.
* each - для итерации по элементам массива или объекта.
* with - для изменения контекста внутри блока.
* unless - для выполнения операции, если условие ложно.
* log - для вывода отладочной информации.

1. Статический файл - это файл, который сервер отправляет клиенту без какой-либо обработки или изменения. Обычно это файлы CSS, JavaScript, изображения и другие ресурсы, которые не требуют динамической генерации на сервере.
2. HTTP-заголовки:

* Cache-Control: указывает, как кешировать ресурс на стороне клиента или прокси-сервера.
* ETag: уникальный идентификатор версии ресурса, используемый для проверки его актуальности.
* Last-Modified: дата и время последнего изменения ресурса.
* If-Modified-Since: условный заголовок, который позволяет клиенту запросить ресурс только в том случае, если он изменился после определенной даты.
* If-None-Match: условный заголовок, который позволяет клиенту запросить ресурс только в том случае, если его ETag не совпадает с указанным значением.
* Expires: указывает дату и время, после которых ресурс считается устаревшим и должен быть перезапрошен.

1. Свойство \_\_dirname объекта global в Node.js содержит абсолютный путь к каталогу, в котором находится текущий исполняемый скрипт.
2. Метод response.download() веб-сервера Node.js отправляет файл клиенту для загрузки. Он отправляет файл как вложение, принимает путь к файлу в качестве аргумента и отправляет его содержимое в ответе.
3. Метод response.attachment() устанавливает заголовок Content-Disposition в значение attachment, что заставляет браузер интерпретировать ответ как файл для скачивания, а не для отображения в браузере.
4. Заголовок Content-Disposition: attachment указывает браузеру интерпретировать содержимое ответа как вложение, которое должно быть скачано, а не отображено в браузере.