Саранский высший гуманитарно-технический колледж

им. Абая Кунанбаева

Специальность 1304000 – «Программное обеспечение»

# **Задание на производственное обучение по Web программирование и Internet технологии**

Студента Жиганова Валерия Дмитриевича

Группы ПВТ-9-20 технологического отделения

Сроки практики: с 18.04.2023 г. по 07.09.2022 г.

Содержание отчета:

**Введение**

1.1 Особенности IP-протоколов. IP-туннели. Обзор браузеров.

1.2 Клиент-серверные технологии Web.

1.3 Планирование Web-сайта.

1.4 Элементы Web-страниц.

2.1 Определение целей веб-сайта и его позиционирование

2.2 Создание концепта сайта

2.3 Верстка сайта.

2.4 Работа с базами данных MySQL

2.5 Наполнение сайта информацией.

2.6 Тестирование сайта

Заключение

Список используемой литературы

Приложения

Задание выдано 18.04.2023 г.

Срок сдачи отчета 07.09.2022 г.

Зам. директора по ПО Теленков С.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Председатель П(Ц)К Авдеева И.О. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель УП Коломоец Д.Л. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

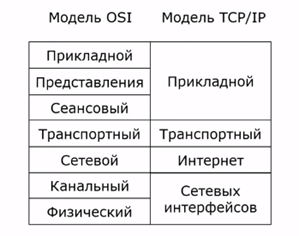
## Задание на учебную практику получил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (Ф.И.О.)

**Введение**

* 1. Особенности IP-протоколов. IP-туннели. Обзор браузеров.

Стек протоколов TCP/IP – это альфа и омега Интернета, и нужно не только знать, но также понимать модель и принцип работы стека. Модель TCP/IP Изначально данный стек создавался для объединения больших компьютеров в университетах по телефонным линиям связи соединения «точка-точка». Но когда появились новые технологии, широковещательные (Ethernet) и спутниковые, возникла необходимость адаптировать TCP/IP, что оказалось непростой задачей. Именно поэтому наряду с OSI появилась модель TCP/IP. Через модель описывается, как необходимо строить сети на базе различных технологий, чтобы в них работал стек протоколов TCP/IP.



В таблице представлено сравнение моделей OSI и TCP/IP. Последняя включает в себя 4 уровня:

1. Самый нижний, уровень сетевых интерфейсов, обеспечивает взаимодействие с сетевыми технологиями (Ethernet, Wi-Fi и т. д.). Это объединение функций канального и физического уровней OSI.
2. Уровень интернет стоит выше, и по задачам перекликается с сетевым уровнем модели OSI. Он обеспечивает поиск оптимального маршрута, включая выявление неполадок в сети. Именно на этом уровне работает маршрутизатор.
3. Транспортный отвечает за связь между процессами на разных компьютерах, а также за доставку переданной информации без дублирования, потерь и ошибок, в необходимой последовательности.
4. Прикладной объединил в себе 3 уровня модели OSI: сеансовый, представления и прикладной. То есть он выполняет такие функции, как поддержка сеанса связи, преобразование протоколов и информации, а также взаимодействие пользователя и сети.

Структуру стека протоколов TCP/IP можно разделить на четыре уровня. Самый нижний — уровень сетевого интерфейса (уровень IV) -соответствует физическому и канальному уровню модели OSI. В стеке протоколов TCP/IP этот уровень не регламентирован. Уровень сетевого интерфейса отвечает за прием дейтаграмм и передачу их по конкретной сети. Интерфейс с сетью может быть реализован драйвером устройства или сложной системой, которая использует свой протокол канального уровня (коммутатор, маршрутизатор). Он поддерживает стандарты физического и канального уровня популярных локальных сетей: Ethernet, Token Pang, FDDI и т.д. Для распределенных сетей поддерживаются проколы соединений РРР и SLIP, а для глобальных сетей — протокол Х.25. Предусмотрена поддержка использования развивающейся технологии коммутации ячеек — ATM. Обычной практикой стало включение в стек протоколов TCP/IP новых технологий локальных или распределенных сетей и регламентация их новыми документами RFC. Сетевой уровень (уровень III) — это уровень межсетевого взаимодействия. Уровень управляет взаимодействием между пользователями в сети. Он принимает от транспортного уровня запрос на посылку пакета от отправителя вместе с указанием адреса получателя. Уровень инкапсулирует пакет в дейтаграмму, заполняет ее заголовок и при необходимости использует алгоритм маршрутизации. Уровень обрабатывает приходящие дейтаграммы и проверяет правильность поступившей информации. На стороне получателя программное обеспечение сетевого уровня удаляет заголовок и определяет, какой из транспортных протоколов будет обрабатывать пакет. В качестве основного протокола сетевого уровня в стеке TCP/IP используется протокол IP, который и создавался с целью передачи информации в распределенных сетях. Достоинством протокола IP является возможность его эффективной работы в сетях со сложной топологией. При этом протокол рационально использует пропускную способность низкоскоростных линий связи. В основе протокола IP заложен дейтаграммный метод, который не гарантирует доставку пакета, но направлен на ее осуществление. К этому уровню относятся все протоколы, которые создают, поддерживают и обновляют таблицы маршрутизации. Кроме того, на этом уровне функционирует протокол обмена информацией об ошибках между маршрутизаторами в сети и отправителями. Следующий уровень — транспортный (уровень II). Основной его задачей является обеспечение взаимодействия между прикладными программами. Транспортный уровень управляет потоком информации с обеспечением надежной передачи. Для этого использован механизм подтверждения правильного приема с дублированием передачи утерянных или пришедших с ошибками пакетов. Транспортный уровень принимает данные от нескольких прикладных программ и посылает их более низкому уровню. При этом он добавляет дополнительную информацию к каждому пакету, в том числе и значение вычисленной контрольной суммы. На этом уровне функционирует протокол управления передачей данных TCP (Transmission Control Protocol) и протокол передачи прикладных пакетов дейтаграммным методом UDP (User DatagramProtocol). Протокол TCP обеспечивает гарантированную доставку данных за счет образования логических соединений между удаленными прикладными процессами. Работа протокола UDP аналогична работе протокола IP, но основной его задачей является выполнение функций связующего звена между сетевым протоколом и различными приложениями. Самый верхний уровень (уровень I) — прикладной. На нем реализованы широко используемые сервисы прикладного уровня. К ним относятся: протокол передачи файлов между удаленными системами, протокол эмуляции удаленного терминала, почтовые протоколы и т.д.

* 1. Клиент-серверные технологии Web.

HTTP (HyperText Transfer Protocol - RFC 1945, RFC 2616) - протокол прикладного уровня для передачи гипертекста.

Центральным объектом в HTTP является ресурс, на который указывает URL в запросе клиента. Обычно такими ресурсами являются хранящиеся на сервере файлы. Особенностью протокола HTTP является возможность указать в запросе и ответе способ представления одного и того же ресурса по различным параметрам: формату, кодировке, языку и т. д. Именно благодаря возможности указания способа кодирования сообщения клиент и сервер могут обмениваться двоичными данными, хотя изначально данный протокол предназначен для передачи символьной информации. На первый взгляд это может показаться излишней тратой ресурсов. Действительно, данные в символьном виде занимают больше памяти, сообщения создают дополнительную нагрузку на каналы связи, однако подобный формат имеет много преимуществ. Сообщения, передаваемые по сети, удобочитаемы, и, проанализировав полученные данные, системный администратор может легко найти ошибку и устранить ее. При необходимости роль одного из взаимодействующих приложений может выполнять человек, вручную вводя сообщения в требуемом формате.

В отличие от многих других протоколов, HTTP является протоколом без памяти. Это означает, что протокол не хранит информацию о предыдущих запросах клиентов и ответах сервера. Компоненты, использующие HTTP, могут самостоятельно осуществлять сохранение информации о состоянии, связанной с последними запросами и ответами. Например, клиентское веб-приложение, посылающее запросы, может отслеживать задержки ответов, а веб-сервер может хранить IP-адреса и заголовки запросов последних клиентов.

Все программное обеспечение для работы с протоколом HTTP разделяется на три основные категории:

Серверы - поставщики услуг хранения и обработки информации (обработка запросов).

Клиенты - конечные потребители услуг сервера (отправка запросов).

Прокси-серверы для поддержки работы транспортных служб.

Основными клиентами являются браузеры, например: Internet Explorer, Opera, Mozilla Firefox, Netscape Navigator и другие. Наиболее популярными реализациями веб-серверов являются: Internet Information Services (IIS), Apache, lighttpd, nginx. Наиболее известные реализации прокси-серверов: Squid, UserGate, Multiproxy, Naviscope.

"Классическая" схема HTTP-сеанса выглядит так.

1.Установление TCP-соединения.

2.Запрос клиента.

3.Ответ сервера.

4.Разрыв TCP-соединения.

Таким образом, клиент посылает серверу запрос, получает от него ответ, после чего взаимодействие прекращается. Обычно запрос клиента представляет собой требование передать HTML-документ или какой-нибудь другой ресурс, а ответ сервера содержит код этого ресурса.

В состав HTTP-запроса, передаваемого клиентом серверу, входят следующие компоненты.

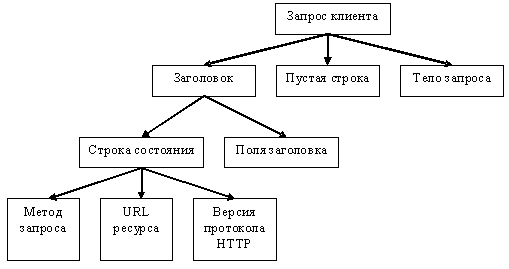
Строка состояния (иногда для ее обозначения используют также термины строка-статус, или строка запроса).

Поля заголовка.

Пустая строка.

Тело запроса.

Строку состояния вместе с полями заголовка иногда называют также заголовком запроса.



Строка состояния имеет следующий формат:

метод\_запроса   URL\_pecypca       версия\_протокола\_НТТР

Рассмотрим компоненты строки состояния, при этом особое внимание уделим методам запроса.

Метод, указанный в строке состояния, определяет способ воздействия на ресурс, URL которого задан в той же строке. Метод может принимать значения GET, POST, HEAD, PUT, DELETE и т.д. Несмотря на обилие методов, для веб-программиста по-настоящему важны лишь два из них: GET и POST.

GET. Согласно формальному определению, метод GET предназначается для получения ресурса с указанным URL. Получив запрос GET, сервер должен прочитать указанный ресурс и включить код ресурса в состав ответа клиенту. Ресурс, URL которого передается в составе запроса, не обязательно должен представлять собой HTML-страницу, файл с изображением или другие данные. URL ресурса может указывать на исполняемый код программы, который, при соблюдении определенных условий, должен быть запущен на сервере. В этом случае клиенту возвращается не код программы, а данные, сгенерированные в процессе ее выполнения. Несмотря на то что, по определению, метод GET предназначен для получения информации, он может применяться и в других целях. Метод GET вполне подходит для передачи небольших фрагментов данных на сервер.

POST. Согласно тому же формальному определению, основное назначение метода POST - передача данных на сервер. Однако, подобно методу GET, метод POST может применяться по-разному и нередко используется для получения информации с сервера. Как и в случае с методом GET, URL, заданный в строке состояния, указывает на конкретный ресурс. Метод POST также может использоваться для запуска процесса.

Методы HEAD и PUT являются модификациями методов GET и POST.

* 1. Планирование Web-сайта.

1. Язык разработки.

Первое- для разработки был использован один из самых вариативных и стремительно набирающих популярность фреймворков – Flutter.

Особенностью данного фреймворка является уникальный стиль написания сайтов. Основное — это применение концепций ООП в его написании.

Языком данного фреймворка является «Dart». Язык построенный на удобстве С++ и надёжности JVM.

1. Интеграция с БД

Для данного проекта был выбранa достаточно популярная БД MySQL

В неё занесены комментарий, а также данные для подключения к интегрированным сервисам(нет)

1. Интерфейс

Минимализм и красота – слоган, которым руководствуется разработчик при написание сайта

Основные цвета, использованные в разработке:

 static const Color primary = Color(0xFF0B1E2D);

  static const Color secondary = Color(0xFF152A3A);

  static const Color blue = Color(0xFF22A2BD);

  static const Color white = Color(0xFFFFFFFF);

  static const Color red = Color(0xFFEB5757);

  static const Color grey = Color(0xFF5B6975);

  static const Color green = Color(0xFF43D049);

  static const Color transparent = Colors.transparent;

  static const Color violet = Color(0xFF8d32e3);

  static const Color black = Color.fromARGB(255, 0, 0, 0);

  static const Color transparentpol = Color.fromARGB(150, 0, 0, 0);

* 1. Элементы Web-страниц.

2.1 Определение целей веб-сайта и его позиционирование

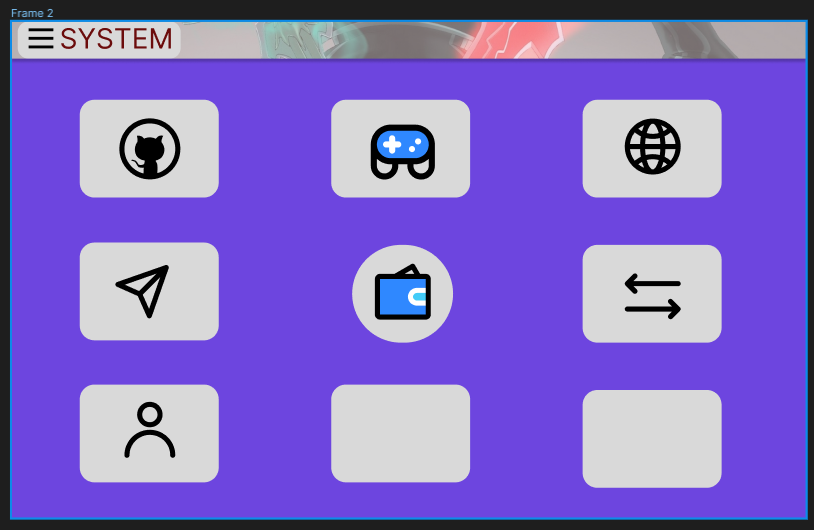
Цели преследуемые в разработке сайта

* Использование любого и языков WEB разработки
* Интеграция с базой данных (далее БД)
* «User-friendly» интерфейс
* Емкость проекта
* Использование актуальной идеи
* Красивая подача сайта
* (защита практики )

Сайт позиционирует себя как Сайт-визитная карточка

2.2 Создание концепта сайта

Концепт сайта был разработан на платформе Figma и дошел до итогового результата практически без изменений



Так же, интерфейс адаптирован под мобильные устройства, а так же, в пару кликов мыши, благодаря использованный при написаний сайта технологий, может стать приложением под любую из 6 доступных платформ.

2.3 Верстка сайта.

На сайте представлена всего одна страница(одностраничный сайт). Все переходы оставляют вас на том же самом сайте, просто меня его внешний вид

При открытие сайт пользователя встречает заголовок с просьбой не рассматривать этот проект как коммерческое предложение в ввиду того что в данном проекте было первое ознакомление с той или иной библиотекой и всем фреймворком в целом. Далее при прокручивание в низ, появляется 10 блоков

Каждый из блоков отвечает за смену(или редирект ) на внешний сервис

А десятый блок, отвечает за оставление комментариев

На разработку сайта было потрачено 87 часов 17 минут(ну это так, к слову)

2.4 Работа с базами данных MySQL

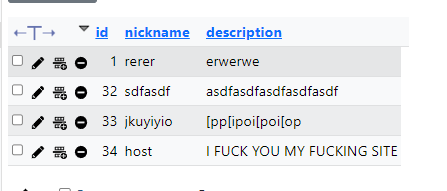
Бд была организована с помощью OSPanel, MySQL, PHP

Серверная часть кода представлена так же в проекте

Столбец id отвечает за автоматическое присваивание пользователю определённого номера

Столбец Nickname отвечает за идентификацию пользователю (не уникален, не охота было написать 4 строчки{знаю, ленивый })

Столбец description разумеется отвечает за текст комментария



2.5 Наполнение сайта информацией.

Так как сайт является визитной карточкой, наполнение информацией происходило контактами для связи(TG, DS) с Создателем, какими то личным проектами(github.com)

2.6 Тестирование сайта

Тестирование сайта производилось методом белого ящика так как результат выполнения заранее был известен, и в большей части случаев совпадал с полученным в ходе тестирования

Заключение

В заключение можно сказать что оно того стоило, проделанная работа была выполнена почти полностью, но при желании можно доделать данный проект и разместить его на каком-нибудь хостинге и распространять как свою визитную карточку

Список используемой литературы

* Google.com
* Chat.openai.com
* Stackoverflow.ru
* Stackoverflow.com

Приложение

