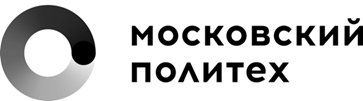
Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(МОСКОВСКИЙ ПОЛИТЕХ)



ЛАБОРАТОРНАЯ РАБОТА 1

**Вводная часть. Основные понятия и определения.**

По курсу

**Проектирование Веб-сервисов**

ВыполнилКилеев С.И**.**

**студент группы 211-321**

Проверил

Пардаев А.А.

Москва, 2024

**Лабораторная работа 1**

**Цель работы:** Изучить основные архитектурные концепции веб-приложений.

**Задачи:**

1. Ознакомиться с Основными определениями и понятиями  
2. Выполнить тест  
3. Ответить на вопросы на Контрольные вопросы. Ответы на Контрольные вопросы написать в отчет

4. Привести примеры (ссылки на ресурс, минимум три) веб-сервиса, веб-сайта и веб-приложения.

**Контрольные вопросы по лабораторной работе**

1) Чем отличаются веб-сервис, веб-сайт и веб-приложение?

1. ВЕБ-САЙТ: Это коллекция веб-страниц, объединенных общей темой или целью. Обычно веб-сайты используются для представления информации, предоставления контента и обмена информацией с пользователями. Веб-сайты могут содержать текст, изображения, видео и другие мультимедийные элементы. Зачастую они предоставляют статический контент и не обладают сложными функциональными возможностями.

2. ВЕБ-СЕРВИС: Это программное обеспечение, которое предоставляет определенную функциональность через сеть Интернет. Веб-сервисы позволяют приложениям взаимодействовать и обмениваться данными с помощью стандартных протоколов, таких как HTTP. Они могут предлагать различные возможности, например, обработку платежей, интеграцию социальных сетей, поиск информации и многое другое. Веб-сервисы могут работать как самостоятельные сущности или быть частью веб-приложений.

3. ВЕБ-ПРИЛОЖЕНИЕ: Это интерактивный веб-сайт или программное обеспечение, которое позволяет пользователям выполнять определенные задачи, используя веб-интерфейс. Веб-приложения имеют возможность взаимодействия с пользователями, обрабатывать ввод данных и предоставлять динамический контент. Они могут включать в себя функциональность, такую как онлайн-магазины, социальные сети, электронные почты и управление бизнес-процессами.

2) В чем разница между SOAP и REST?

REST и SOAP на самом деле не сопоставимы. REST — это архитектурный стиль. SOAP — это формат обмена сообщениями.

* Пример реализации RESTful: JSON через HTTP
* Пример реализации SOAP: XML поверх SOAP через HTTP

3) Какие виды API существуют?

### REST API (Representational State Transfer)

REST API является наиболее распространенным типом API в клиент-серверной архитектуре и имеет ряд свойств — например, отсутствие хранения состояния и кэшируемость. REST API построен на стандартных протоколах HTTP и обычно возвращает данные в формате JSON, хотя может поддерживать и другие форматы.

REST API использует стандартные HTTP-методы для взаимодействия, включая GET, POST, PUT, DELETE и другие. Каждый метод соответствует определенному типу действий, которые могут быть выполнены над ресурсами API.

Простота и гибкость REST API сделали его лучшим выбором для многих разработчиков, но при работе с большими объемами данных он может быть неэффективными, так как для получения всех необходимых данных может требоваться несколько запросов.

### SOAP API (Simple Object Access Protocol)

SOAP — это протокол для обмена структурированной информацией в веб-сервисах с использованием XML. Он обладает высокой расширяемостью и позволяет осуществлять обмен данными через несколько различных транспортных протоколов, включая HTTP, SMTP и другие.

API SOAP известны своей надежностью и часто используются в энтерпрайз-разработке. Они предлагают встроенную обработку ошибок и их можно использовать с различными сетевыми протоколами. Однако зависимость SOAP от XML может приводить к большим объемам передаваемых данных, что делает его менее эффективным по сравнению с другими API в каких-то сценариях использования.

### WebSocket API

WebSocket API обеспечивает постоянный, полнодуплексный канал связи между клиентом и сервером. В отличие от REST и SOAP, которые придерживаются формата «запрос-ответ», WebSocket сохраняет соединение открытым, что позволяет передавать данные в режиме реального времени. Это делает WebSocket API идеальным для приложений, требующих функциональности в реальном времени, таких как чат-приложения, онлайн-игры и системы отслеживания в реальном времени.

Несмотря на свою мощь, WebSocket API могут быть более сложными в реализации и требуют больше ресурсов для поддержания открытых соединений.

### gRPC API (Google Remote Procedure Call)

gRPC — это высокопроизводительная платформа с открытым исходным кодом, разработанная компанией Google. Он использует протокол HTTP/2 для передачи данных и Protocol Buffers (protobuf), высокопроизводительный формат двоичных данных, в качестве языка определения интерфейса.

gRPC поддерживает четыре типа взаимодействия: унарный (стандартный запрос-ответ), потоковая передача данных с сервера, потоковая передача данных от клиента и двунаправленная потоковая передача. Эти возможности в сочетании с эффективностью делают gRPC идеальным решением для микросервисной архитектуры.

### GraphQL API

Разработанный компанией Facebook, GraphQL представляет собой язык запросов к API и рабочую среду для выполнения этих запросов. В отличие от REST API, где вам нужно делать запросы к различным эндпоинтам, чтобы получить соответствующие данные, GraphQL позволяет сделать один единственный запрос, чтобы получить именно те данные, которые вам нужны.

4) Какие принципы REST архитектуры?

Иерархия клиент-сервер. Обеспечивает разделение обязанностей между клиентом и сервером, что упрощает разработку и поддержку приложений.

Отсутствие состояния. Позволяет серверу считаться функционально независимым от клиента, что облегчает масштабирование и улучшает производительность.

Кэширование. Позволяет сохранять копии ответов сервера для уменьшения нагрузки на сеть и повышения скорости работы приложения.

Единообразие интерфейса. Обеспечивает однозначное и простое взаимодействие между клиентом и сервером, что упрощает разработку и интеграцию систем.

Ограничение связности. Позволяет уменьшить зависимость между компонентами системы, что улучшает ее гибкость и модульность.

5) Что такое WSDL?

WSDL (язык описания веб-служб) — это язык на основе XML, используемый для описания функций, операций и интерфейсов веб-службы.

Он предоставляет стандартизированный формат для определения структуры и поведения веб-сервисов, позволяя клиентам понять, как взаимодействовать с сервисом.

Ключевые компоненты WSDL включают в себя:

* определение службы,
* типы,
* сообщения,
* операции,
* типы портов,
* привязка,
* служба.

6) Как обеспечить безопасность веб-сервиса?

Для обеспечения безопасности веб-сервиса можно использовать следующие меры:

1. Контроль целостности данных через скрипты.
2. Правильная структура данных в базе данных.
3. Поиск и исправление ошибок в ПО (профилактика, обслуживание).
4. Защита от DOS-атак.
5. Нагрузочное тестирование.
6. Резервный сервер.
7. Мониторинг доступности и оповещения.

7) Какие методы аутентификации и авторизации могут использоваться для веб-сервисов?

Для аутентификации и авторизации в веб-сервисах могут использоваться следующие методы:

1. Аутентификация по паролю. Пользователь предоставляет username и password для идентификации и аутентификации в системе.
2. Использование сертификата для аутентификации. Во время аутентификации сервер выполняет проверку сертификата на основании правил, таких как проверка цепочки сертификатов, проверка срока действия и проверка списков исключения.
3. Аутентификация по одноразовым паролям. Этот метод обычно применяется дополнительно к аутентификации по паролям для реализации двухфакторной аутентификации (2FA).
4. Аутентификация по ключам доступа. Этот способ чаще всего используется для аутентификации устройств, сервисов или других приложений при обращении к веб-сервисам.

8) Каким образом можно обеспечить масштабируемость и доступность веб-сервисов?

Для обеспечения масштабируемости и доступности веб-сервисов можно использовать облачные технологии:

1. Автоматическое масштабирование. Позволяет динамически изменять количество ресурсов в зависимости от текущей нагрузки. Это обеспечивает стабильную работу сайта даже при резких скачках трафика.
2. Балансировка нагрузки. Распределяет трафик между несколькими серверами, что улучшает производительность и отказоустойчивость.

9) Какие инструменты и технологии широко применяются для разработки и проектирования веб-сервисов?

Для разработки и проектирования веб-сервисов широко применяются следующие инструменты:

1. Системы управления контентом (CMS): WordPress, Joomla, Drupal.
2. Инструменты для создания схем и прототипов: Balsamiq, Axure, Sketch и Figma.
3. Инструменты для дизайна и верстки: Adobe Photoshop, Adobe Illustrator, Sketch и Canva.
4. Редакторы кода: Visual Studio Code, Sublime Text, Atom и Brackets.
5. Инструменты для кросс-браузерного тестирования: BrowserStack, Sauce Labs и CrossBrowserTesting.
6. Инструменты оптимизации производительности: GTmetrix, Pingdom и Google PageSpeed Insights.
7. Системы контроля версий: обеспечивают централизованное хранилище для кода и других файлов проекта.

10) Зачем нужен мониторинг и аналитика веб-сервиса?

Веб-аналитика — это набор инструментов, который позволяет оценить и проанализировать, насколько эффективно работает продвижение. Она помогает определить целевую аудиторию ресурса, собрать и исследовать данные о пользователях и их поведении на сайте. Впоследствии, опираясь на этот анализ, можно предпринять результативные шаги по оптимизации и усовершенствованию сайта, его функциональности, улучшить качество дизайна, юзабилити и контента.

Мониторинг веб-сервиса нужен, чтобы наблюдать за его функционированием, в том числе за:

* доступностью сайта,
* скоростью загрузки страниц,
* работой сервисов и приложений на сайте,
* наличием ошибок и сбоев.

Мониторинг позволяет оперативно реагировать на проблемы и устранять их до того, как они повлияют на пользователей и бизнес-процессы.

**Отчёт по выполнению:**

Веб-сайты:

1. <https://climbiotech.ru/>
2. <https://4pda.to/>
3. <https://www.rbc.ru/>

Веб-сервисы:

1. [https://mail.google.com](https://mail.google.com/mail/u/0/#inbox)
2. <https://yookassa.ru>
3. <https://robokassa.com>

Веб-приложения:

1. <https://miro.com/ru/>
2. <https://www.youtube.com/>
3. <https://www.figma.com>