**Что я должен написать ?**

**- в практической части писать о PHP, скрипты, БД, логику программы, выбор среды разработки.**

**- в теории написать о бэкэндэ, роли БД в информационных системах и образовательных учреждениях, можно чуток затронуть верстку**

# ВВЕДЕНИЕ

Backend-программирование является одной из самых важных областей разработки программного обеспечения. Это критическая компонента, которая обеспечивает работу веб-сайтов и мобильных приложений, позволяя пользователям получать доступ к данным и выполнять действия на стороне сервера.

Backend-разработчик занимается созданием и поддержкой серверной части программного обеспечения. Это включает в себя работу с базами данных, настройку серверов, написание API и других интерфейсов для обмена данными между клиентской и серверной частями. Backend-разработчик также заботится о безопасности и оптимизации работы приложений.

Важным аспектом backend-разработки является выбор языка программирования. Существует множество языков, которые могут использоваться для написания серверной части, таких как PHP, Python, Ruby, Java и другие. Каждый из них имеет свои преимущества и недостатки, и выбор языка зависит от конкретных требований проекта.

Один из наиболее распространенных языков для backend-разработки является PHP. Это скриптовый язык программирования, который используется для создания динамических веб-страниц и приложений. Он наиболее распространен на серверной стороне web-разработки. PHP позволяет программистам создавать интерактивные веб-страницы, обрабатывать формы, управлять базами данных, создавать и отправлять электронные письма и многое другое. Он предоставляет множество библиотек и фреймворков для разработки веб-приложений.

Backend-разработка также включает в себя работу с базами данных. Существует множество типов баз данных, таких как реляционные, NoSQL и графовые базы данных. Реляционные базы данных, такие как MySQL и PostgreSQL, широко используются в веб-приложениях и обеспечивают хранение структурированных данных. NoSQL базы данных, такие как MongoDB и Cassandra, могут хранить неструктурированные данные и обеспечивать высокую производительность при обработке больших объемов данных. Графовые базы данных, такие как Neo4j, используются для хранения и обработки данных, которые имеют связи между собой.

Backend-разработчик также должен обеспечивать оптимизацию работы приложения. Это может быть достигнуто путем использования кэширования данных, минимизации количества запросов к базе данных и оптимизации кода. Оптимизация является критически важной для обеспечения быстрой работы приложения и удовлетворения пользовательских потребностей.

В заключение, backend-программирование является критически важной областью разработки программного обеспечения. Backend-разработчик заботится о создании и поддержке серверной части приложения, работе с базами данных, обеспечении безопасности и оптимизации работы приложения. Выбор языка программирования зависит от требований проекта, а безопасность и оптимизация являются критически важными для обеспечения быстрой и безопасной работы приложения.

Цель выпускной квалификационной работы состоит в написании web-сайта по упрощению образовательного процесса, значительной частью которого является backend-разработка.

Для реализации backend части web-сайта необходимо решить следующие задачи:

1. Выбрать язык программирования backend разработки.

2. Разработать архитектуру backend приложения, определить, какие компоненты будут использоваться (например, базы данных, кэширование, веб-серверы и т.д.) и как они будут взаимодействовать между собой.

3. Написать код backend приложения, включая обработчики запросов, механизмы аутентификации и авторизации пользователей, обработку ошибок и т.д.

4. Разработать механизмы безопасности, такие как защита от SQL-инъекций, XSS-атак и CSRF-атак.

5. Настроить взаимодействие с базой данных, написать SQL-запросы и хранимые процедуры.

6. Разработать API для взаимодействия с frontend частью сайта или другими приложениями.

7. Настроить веб-сервер и оптимизировать его работу для обеспечения высокой производительности и масштабируемости.

8. Написать тесты для backend приложения, чтобы убедиться в его работоспособности и безопасности.

9. Настроить систему логирования, чтобы можно было отслеживать ошибки и проблемы в работе backend приложения.

10. Разработать механизмы кэширования, чтобы ускорить работу backend приложения и уменьшить нагрузку на сервер.

# Основные функции backend части сайта

## Обработка запросов и взаимодействие пользователей с серверной частью сайта

Обработка запросов и взаимодействие пользователей с серверной частью сайта происходит следующим образом:

1. Пользователь отправляет запрос на сервер с помощью веб-браузера. Запрос может быть отправлен при нажатии на кнопку, переходе по ссылке или заполнении формы.

2. Запрос попадает на сервер и обрабатывается серверной частью сайта. Серверная часть получает данные от пользователя, проверяет их на корректность и обрабатывает запрос.

3. Если запрос требует доступа к базе данных, серверная часть взаимодействует с базой данных, обрабатывает запрос, получает необходимые данные и возвращает их пользователю.

4. Серверная часть обрабатывает запрос и генерирует ответ, который отправляется обратно пользователю.

5. Веб-браузер получает ответ от сервера и отображает его пользователю. Результат может быть отображен на странице, загружен новый контент, произведен редирект на другую страницу или произведена другая операция, в зависимости от того, что требуется.

6. Если запрос содержит ошибки или некорректные данные, серверная часть обрабатывает ошибку и отправляет пользователю соответствующий ответ.

7. Пользователь может взаимодействовать с серверной частью сайта, отправляя новые запросы, заполняя формы, загружая файлы или производя другие действия, которые требуют взаимодействия с сервером.

Таким образом, обработка запросов и взаимодействие пользователей с серверной частью сайта - это процесс, который позволяет пользователям взаимодействовать с сайтом, отправлять запросы на сервер, получать ответы и результаты, обрабатывать ошибки и многое другое. Этот процесс является необходимым для работы любого веб-сайта и позволяет пользователям получать необходимую информацию и выполнять необходимые действия на сайте.

## Взаимодействие с базой данных

Backend-часть сайта взаимодействует с базой данных для получения, сохранения и обработки данных. Для этого используется язык запросов SQL (Structured Query Language), который позволяет взаимодействовать с базой данных и выполнять запросы к ней.

В первую очередь происходит установка соединения с базой данных. Backend-часть сайта использует специальные драйверы для взаимодействия с базой данных. Для установки соединения необходимо указать параметры подключения, такие как имя хоста, имя пользователя, пароль и название базы данных.

Далее происходит выполнение запросов к базе данных. Backend-часть сайта может выполнять различные запросы к базе данных, такие как запросы на чтение и запись данных, запросы на создание и удаление таблиц, запросы на изменение структуры таблиц и т.д. Для выполнения запросов используется язык SQL.

Backend-часть сайта получает результаты запросов, которые могут быть представлены в виде таблиц или наборов данных. Эти результаты могут быть обработаны и переданы на frontend-часть сайта для отображения пользователю.

Для оптимизации работы с базой данных используются различные механизмы, такие как кэширование запросов, использование индексов, разделение базы данных на несколько таблиц и т.д.

После завершения работы с базой данных соединение с ней должно быть закрыто, чтобы освободить ресурсы сервера и предотвратить утечки данных.

Таким образом, взаимодействие backend-части сайта с базой данных является важной частью работы веб-приложений. Оно позволяет получать, сохранять и обрабатывать данные, предоставлять доступ к информации для пользователей и выполнять множество других задач, связанных с работой сайта.

## Аутентификация и авторизация

Аутентификация - это процесс проверки подлинности пользователя. Аутентификация позволяет убедиться в том, что пользователь является тем, за кого он себя выдает. Для аутентификации пользователей в backend-части сайта используются различные механизмы, такие как проверка логина и пароля, использование токенов, проверка сертификатов и т.д.

Авторизация - это процесс проверки прав доступа пользователя. Авторизация позволяет определить, какие действия пользователь может выполнять в приложении и какие ресурсы ему доступны. Для авторизации пользователей в backend-части сайта используются различные механизмы, такие как проверка прав доступа к определенным ресурсам, использование ролей и правил, проверка токенов и т.д.

Процесс аутентификации и авторизации в backend-части сайта может быть описан следующим образом:

Пользователь отправляет запрос на сервер, который требует аутентификации и/или авторизации. Серверный код проверяет данные пользователя, такие как логин и пароль, и определяет, является ли пользователь действительно тем, за кого он себя выдает. Если пользователь успешно прошел аутентификацию, серверный код проверяет права доступа пользователя к ресурсам сайта. Если пользователь имеет необходимые права доступа, серверный код предоставляет ему доступ к ресурсам. Если пользователь не имеет необходимых прав доступа, серверный код возвращает пользователю ошибку доступа и предоставляет ему возможность войти под другим аккаунтом или зарегистрироваться, чтобы получить доступ к ресурсам.

По завершении работы с ресурсами сайта, серверный код закрывает сессию пользователя и завершает процесс работы с сервером.

Таким образом, аутентификация и авторизация в backend-части сайта позволяют обеспечить безопасность веб-приложений и контролировать доступ пользователей к ресурсам сайта. Они используются для защиты данных пользователей, предотвращения несанкционированного доступа к сайту и обеспечения безопасности работы приложения.

## Работа с файлами

Работа с файлами в backend-части сайта может использоваться для различных задач, таких как загрузка и хранение изображений, аудио и видео файлов, обработка текстовых файлов, скачивание документов и многое другое. Этот процесс является важной частью веб-приложений и позволяет пользователям работать с файлами на сайте.

Процесс работы с файлами в backend-части сайта может быть описан следующим образом: Пользователь отправляет файл на сервер через специальную форму. Backend-часть сайта получает файл и сохраняет его на сервере. Backend-часть сайта может обрабатывать эти файлы, выполнять различные операции с ними, например, изменять размер изображений, конвертировать файлы в другой формат, обрабатывать текстовые файлы и так далее. Файлы сайта хранятся на сервере или в специальных облачных хранилищах. Для хранения файлов на сервере может использоваться специальная директория, доступная для записи.

Backend-часть сайта также может отправлять файлы пользователю по запросу. Например, пользователь может нажать на кнопку «Скачать», чтобы скачать файл с сервера. Backend-часть сайта может удалять файлы, которые больше не нужны. Например, файлы, которые были загружены на сервер, но не были использованы, могут быть удалены, чтобы освободить место на сервере.

## Обработка ошибок

Обработка ошибок в backend-части сайта позволяет обеспечить безопасность и стабильную работу приложения. Она помогает предотвратить сбои в работе сайта, защитить данные пользователей и своевременно реагировать на проблемы в работе приложения.

Backend-часть сайта может обнаружить ошибки в процессе выполнения операций, таких как доступ к базе данных, обработка файлов, отправка электронной почты и других операций. Она также должна логировать (делать логи) ошибки, чтобы можно было отследить их происхождение и причины возникновения. Логирование ошибок помогает быстро найти причину проблемы и исправить ее. Чтобы уведомить программиста об ошибке, происходит отправка уведомления администраторам сайта о возникшей ошибке, например в консоль браузера. Это позволяет быстро реагировать на проблему и исправлять ее до того, как она приведет к серьезным последствиям. Backend-часть сайта должна обрабатывать ошибки, чтобы предотвратить сбои в работе приложения. Обработка ошибок может включать в себя такие действия, как вывод сообщения об ошибке пользователю, перенаправление на другую страницу, отправка уведомления администраторам сайта и т.д.

# 2 Базы данных в backend-разработке

## 2.1 Роль баз данных в информационных системах

Базы данных играют важную роль в информационных системах, поскольку позволяют хранить, организовывать и обрабатывать большие объемы данных.

Информационные системы предприятий используются для хранения и обработки различных данных, таких как информация о клиентах, заказах, продуктах, финансовых показателях и других. Важно, чтобы эти данные хранились в надежном и безопасном месте, а также были структурированы и эффективно обрабатывалась.

Роль баз данных в информационных системах заключается в том, что они позволяют хранить данные в определенной структуре и обеспечивают доступ к этим данным для пользователей.

Одним из основных преимуществ баз данных является возможность сократить время доступа к данным и упростить процесс их обработки. Например, если предприятие хранит информацию о клиентах в базе данных, то оно может быстро найти нужные данные и обрабатывать их, не тратя время на поиск нужной информации в большом объеме документов.

Базы данных также позволяют предприятиям сохранять информацию на протяжении длительного времени и использовать ее для анализа и прогнозирования тенденций в бизнесе. Например, предприятие может использовать базу данных, чтобы определить, какие товары или услуги наиболее востребованы у клиентов, какие регионы являются наиболее перспективными для развития бизнеса и т.д. Кроме того, базы данных позволяют предприятиям улучшить свою деятельность и увеличить свою прибыль. Например, предприятие может использовать базу данных для определения наиболее эффективных рекламных кампаний, для улучшения качества обслуживания клиентов и т.д.

Таким образом, роль баз данных в информационных системах состоит в том, что они позволяют предприятиям хранить, обрабатывать и организовывать большие объемы данных. Базы данных позволяют улучшить работу предприятий, повысить эффективность деятельности и увеличить прибыль. Они также обеспечивают безопасность хранения данных и контроль доступа к ним. В целом, базы данных являются неотъемлемой частью информационных систем предприятий и выполняют важную роль в их успешной работе.

## 2.2 Типы баз данных

Существует несколько типов баз данных, каждый из которых имеет свои особенности, преимущества и недостатки. Рассмотрим наиболее распространенные типы баз данных:

1. Реляционные базы данных - это наиболее распространенный тип баз данных, который использует таблицы, связанные друг с другом через ключи. Реляционные базы данных хорошо подходят для хранения структурированных данных, таких как данные о клиентах, заказах, продуктах и т.д. Они обеспечивают высокую точность и надежность данных, а также позволяют легко добавлять, удалять и изменять данные. Однако реляционные базы данных могут быть медленными при обработке больших объемов данных и могут требовать дополнительной работы при проектировании и нормализации структуры базы данных.

2. Иерархические базы данных - это тип баз данных, в котором данные организованы в виде древовидной структуры. Иерархические базы данных хорошо подходят для хранения данных, которые имеют явную иерархическую структуру, например, данные о семейных отношениях, организационной структуре предприятия и т.д. Однако иерархические базы данных могут быть ограничены в использовании, так как они могут быть трудными для изменения и обработки.

3. Объектно-ориентированные базы данных - это тип баз данных, который использует объектно-ориентированный подход к хранению данных. Объектно-ориентированные базы данных хорошо подходят для хранения сложных иерархических структур данных, которые могут быть трудными для хранения в реляционных базах данных. Они обеспечивают более гибкий и быстрый доступ к данным, а также позволяют использовать наследование, полиморфизм и другие особенности объектно-ориентированного программирования. Однако объектно-ориентированные базы данных могут быть сложными в использовании и требовать большого количества времени и ресурсов для их разработки и поддержки.

4. NoSQL базы данных - это тип баз данных, который не использует традиционную реляционную модель для хранения данных. NoSQL базы данных хорошо подходят для хранения неструктурированных данных, таких как данные о социальных сетях, текстовые данные, данные машинного обучения и т.д. Они обеспечивают более гибкую и масштабируемую структуру данных, а также позволяют легко добавлять и изменять данные. Однако NoSQL базы данных могут быть менее надежными и требовать более сложной обработки данных, чем реляционные базы данных.

В заключение, каждый тип баз данных имеет свои преимущества и ограничения, и выбор типа базы данных должен основываться на конкретных потребностях и требованиях предприятия. Реляционные базы данных обеспечивают высокую точность и надежность данных, иерархические базы данных хорошо подходят для хранения данных с явной иерархической структурой, объектно-ориентированные базы данных обеспечивают более гибкий и быстрый доступ к данным, а NoSQL базы данных хорошо подходят для хранения неструктурированных данных.

## 2.3 Работа с базами данных

Работа с базами данных является важным аспектом при backend разработке. Backend-разработчики используют базы данных для хранения, организации и обработки данных, необходимых для работы приложения. Рассмотрим, как происходит работа с базами данных при backend разработке.

На этапе разработки backend-части приложения необходимо выбрать наиболее подходящую базу данных для хранения данных. Разработчик должен учитывать требования к производительности, масштабируемости, безопасности и другим параметрам, чтобы определить, какая база данных наилучшим образом подходит для конкретного проекта. После выбора базы данных необходимо разработать структуру базы данных. Разработчик должен определить таблицы, поля, связи между таблицами и другие параметры, которые будут использоваться для хранения данных. После проектирования структуры базы данных необходимо создать саму базу данных на сервере. Это может быть выполнено с помощью специальных инструментов управления базами данных, таких как MySQL, PostgreSQL, MongoDB, Redis и другие. После создания базы данных необходимо настроить соединение между приложением и базой данных. Это может быть выполнено с помощью программного интерфейса для работы с базой данных, таких как JDBC, ORM (Object-Relational Mapping), PDO и т.д. После подключения к базе данных можно выполнять различные операции с данными, такие как добавление, удаление, обновление, выборка данных и другие. Операции с базой данных могут быть выполнены с помощью языка SQL или специальных библиотек для работы с базами данных, таких как Hibernate, Doctrine и другие. При работе с базами данных возможны ошибки, такие как недоступность базы данных, ошибки при выполнении запросов и другие. Разработчик должен предусмотреть обработку ошибок, чтобы приложение не прекращало работу и не потеряло данные. При работе с базами данных важно оптимизировать производительность, чтобы минимизировать время доступа к данным и улучшить работу приложения. Разработчик должен учитывать такие факторы, как индексирование таблиц, кэширование данных, оптимизация запросов и другие.

Таким образом, работа с базами данных при backend разработке включает в себя выбор наиболее подходящей базы данных, проектирование структуры базы данных, создание базы данных, настройку соединения, выполнение операций с данными, обработку ошибок и оптимизацию производительности. Она является важным аспектом при разработке приложений и позволяет обеспечить эффективное хранение, организацию и обработку данных.

# 3 Разработка backend части web-сайта

## 3.1 Выбор языка программирования

В качестве серверного языка программирования для backend разработки web-сайта был выбран PHP. Он является одним из наиболее популярных языков программирования, используемых для backend разработки, и есть несколько причин, почему для разработки backend-части приложения был выбран PHP:

1. Простота использования. PHP имеет простой и понятный синтаксис, который делает его легким для изучения и использования.

2. Широкое распространение. PHP является одним из наиболее распространенных языков программирования для веб-разработки, и на рынке существует множество инструментов и ресурсов, которые помогут разработчикам работать с PHP.

3. Открытый исходный код. PHP является языком с открытым исходным кодом, что означает, что разработчики могут использовать множество библиотек, фреймворков и других инструментов для разработки приложений.

4. Поддержка баз данных. PHP имеет встроенную поддержку для многих баз данных, включая MySQL, PostgreSQL, Oracle и другие.

5. Большое сообщество. PHP имеет большое сообщество разработчиков, которые создают множество библиотек, фреймворков и других инструментов для упрощения разработки приложений.

6. Гибкость. PHP можно использовать для создания многих типов веб-приложений, включая сайты, блоги, электронную коммерцию, социальные сети и другие.

7. Высокая производительность. PHP обладает высокой производительностью и может легко масштабироваться для обработки больших объемов данных.

В целом, PHP является популярным выбором для backend разработки благодаря своей простоте, широкому распространению, открытому исходному коду, поддержке баз данных, большому сообществу разработчиков, гибкости и высокой производительности.