

АННОТАЦИЯ

на выпускную квалификационную работу бакалавра
специальности Управление информационными ресурсами

Зокиров Бахромджон Б. на тему:

“Оценка и развитие информационной системы переписи населения:
Разработка серверной части”

Система переписи населения является ключевым инструментом для сбора и анализа данных о населении определенной страны или региона. Ее основной целью является получение точной и полной информации о численности населения, его составе, социально-экономических характеристиках и других важных параметрах. Принципы переписи населения включают в себя объективность, анонимность, добровольность участия и конфиденциальность данных.

Первый принцип, объективность, требует сбора данных без каких-либо искажений или предвзятости, чтобы полученные результаты отражали реальное положение дел. Анонимность гарантирует, что переписываемые лица не идентифицируются по их ответам, что способствует более открытому и честному предоставлению информации. Добровольность участия подразумевает, что никто не должен быть вынужден участвовать в переписи, что способствует доверию и качественному сбору данных. И, наконец, конфиденциальность данных гарантирует, что информация, предоставленная участниками, используется исключительно для статистических целей и не раскрывается третьим лицам.

краткое содержание о
чем идет речь
в работе

Цель ВКР заключается в проведении всесторонней оценки текущего состояния информационной системы переписи населения и разработке рекомендаций по ее усовершенствованию. Мы планируем не только выявить существующие проблемы и слабые места системы, но и предложить конкретные шаги по их устранению и дальнейшему развитию системы в соответствии с современными требованиями и ожиданиями общества. Это важный шаг к созданию более эффективной и надежной системы переписи населения, способной точно отражать реальные демографические и социальные процессы и служить основой для принятия обоснованных решений на уровне государственной и общественной политики.

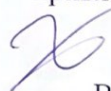
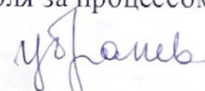
Здесь не все пункты введения
Смотрите в положении
и требования

неясность в использовании результатов предыдущих переписей могут сдерживать граждан от активного участия в этом процессе.

Сложность сбора данных - еще одна проблема, с которой сталкивается система переписи населения. Трудоемкость, высокая стоимость и сложность доступа к респондентам могут замедлить процесс сбора необходимой информации, а иногда и снизить ее качество.

Перспективы включают в себя ряд мер, направленных на улучшение ситуации. В частности, повышение осведомленности населения о переписи через информационные кампании и использование доступных форматов для представления информации может способствовать увеличению участия граждан.

Для укрепления доверия к власти и системе переписи важно обеспечить конфиденциальность данных, прозрачность в использовании результатов и привлечение независимых экспертов для контроля за процессом.

 _____ 

В этой главе рассматриваются основы и принципы системы переписи населения, включая описание основных принципов, этапов проведения, методов сбора данных и международных стандартов. Также обсуждаются проблемы и перспективы развития данной системы. Глава предоставляет введение в систему переписи населения, выделяя ее важность для сбора и анализа данных о населении. Описанные принципы, этапы и методы помогают понять процесс переписи населения как ключевой механизм для получения надежной информации о населении. Международные стандарты играют важную роль в обеспечении сопоставимости данных между различными странами. Наконец, рассмотрение проблем и перспектив развития подчеркивает необходимость постоянного совершенствования системы переписи населения для эффективного использования данных в планировании и управлении социально-экономическим развитием.

2.2 Проектирование архитектуры серверной части:

При принятии решения о выборе архитектурного стиля для вашего проекта, сервисный монолит представляет собой привлекательную альтернативу, которая сочетает в себе преимущества обеих моделей.

Сервисный монолит представляет собой гибрид монолитной и микросервисной архитектур, где приложение организовано как монолит, но разделено на отдельные компоненты или сервисы по функциональным областям. Сервисный монолит обеспечивает простоту в разработке и развертывании, аналогично традиционной монолитной архитектуре. Это означает, что команда разработки может быстро начать работу над проектом без необходимости организации сложной инфраструктуры для микросервисов.

В то же время, сервисный монолит предоставляет определенную гибкость и изоляцию функциональности, подобную микросервисам. Каждый сервис может быть разработан и масштабирован независимо от других, что улучшает общую поддержку приложения.

Однако стоит учитывать, что сервисный монолит также не лишен недостатков. Вместе с упрощением разработки и развертывания могут возникнуть сложности при поддержке при росте размера и сложности приложения, подобные тем, которые характерны для традиционной монолитной архитектуры.

Тем не менее, при удачном выборе и проектировании сервисного монолита можно минимизировать эти риски и обеспечить гибкость и масштабируемость, характерные для микросервисов, сохраняя при этом простоту управления и развертывания, типичные для монолитных решений.

1. Выбор протоколов взаимодействия:

↓ су сажирааи
нав ?


```

        if (connection.State != ConnectionState.Closed)
        {
            await connection.CloseAsync();
        }
        throw new InvalidOperationException(ex.Message);
    }
}

```

Данный метод предназначен для обработки HTTP GET запросов по маршруту "GetInfo". Он асинхронно извлекает информацию из базы данных и возвращает объект типа Data в формате JSON.

Пошаговое описание метода:

1. Устанавливается подключение к базе данных с использованием строки подключения "DefaultConnection".
2. Формируется SQL-запрос для извлечения данных.
3. Выполняется запрос к базе данных, и результаты читаются с помощью SqlDataReader.
4. Если в результате есть строки, данные извлекаются и десериализуются в объект типа Data с использованием библиотеки Newtonsoft.Json.
5. Полученный объект Data возвращается в качестве ответа на запрос.
6. В случае возникновения исключения, происходит корректное закрытие подключения к базе данных, исключение пробрасывается дальше для обработки на уровне выше.

Этот метод обеспечивает получение данных из базы данных и предоставляет их для использования в интерфейсе или других целях.

Листинг 2. SQL запрос для отображения данных об уровне образования.

```

select EducationalLevel.[Name] as 'EducationalLevelName',
Gender.[Name] as 'Gender', COUNT(*) 'Count' from List
inner join EducationalLevel on List.EducationalLevelId =
EducationalLevel.Id
inner join Gender on List.GenderId = Gender.Id

```

человека в таблице List будет найдена соответствующая запись о его поле в таблице Gender.

3. Выборка и группировка данных:

Запрос выбирает следующие столбцы:

EducationalLevel.Name с псевдонимом EducationalLevelName: Это название уровня образования, к которому относится человек.

Gender.Name с псевдонимом Gender: Это пол человека.

COUNT(*) с псевдонимом Count: Это количество людей в каждой группе, полученной в результате объединения и группировки данных.

Запрос группирует данные по трем полям:

EducationalLevel.Id: Это гарантирует, что результаты будут разделены по отдельным уровням образования.

EducationalLevel.Name: Это обеспечивает дальнейшую разбивку внутри каждого уровня образования по его фактическому названию.

Gender.Name: Это позволяет дифференцировать количество людей по полу в каждой группе.

4. Сортировка результатов:

Наконец, запрос сортирует результаты по EducationalLevel.Id. Это гарантирует, что данные будут представлены в порядке уровней образования, с разбивкой по полу в каждой категории.

5. Анализ результатов:

В целом, этот запрос позволяет получить подробную информацию о том, как люди распределены по уровню образования, полу и региону.

Пример использования:

Предположим, вам нужно узнать, сколько мужчин и женщин обучаются на каждом уровне образования в вашей стране.

Данный SQL-запрос может предоставить вам эту информацию, позволяя вам анализировать такие данные, как:

Уровень образования с наибольшим количеством учащихся: Это может помочь вам определить, на какие уровни образования следует направить больше ресурсов.

Распределение полов на каждом уровне образования: Это может выявить возможные гендерные диспропорции в образовании.

Регионы с наибольшим количеством учащихся на определенном уровне образования: Это может быть полезно для планирования распределения образовательных ресурсов по регионам.

Листинг 3. SQL-запрос для отображения данных об источниках доходов населения.

```
select      FinancialSources.Id,      FinancialSources.[Name],  
COUNT(FinancialSources.Id) as 'Count' from List  
inner join FinancialSources on List.FinancialSourcesId =  
FinancialSources.Id  
group by FinancialSources.id, FinancialSources.[Name]
```

Цель данного запроса: Отображение данных об источниках доходов населения.

Структура:

SELECT: Определяет столбцы, которые будут отображаться в результирующем наборе.

FinancialSources.Id: ID источника дохода.

FinancialSources.Name: Название источника дохода.

COUNT(FinancialSources.Id) as 'Count': Количество записей, связанных с каждым источником дохода.

FROM: Указывает таблицу, из которой будут извлекаться данные.

по центру!

← ГЛАВА 3. АПРОБАЦИЯ И ОПИСАНИЕ ИСПОЛЬЗОВАНИЯ ПРИЛОЖЕНИЯ

В данной главе представлены результаты апробации разработанного приложения, а также его описание и рекомендации по использованию. Апробация играет ключевую роль в процессе разработки программного обеспечения, поскольку она позволяет оценить эффективность приложения в реальных условиях использования и выявить потенциальные проблемы или недочеты.

3.1. Публикация приложения

Публикация приложения в IIS и использование Ngrok для доступа извне:

Публикация приложения в IIS:

После завершения разработки приложения и его тестирования локально, мы приступили к его публикации на сервере.

Для этого был выбран сервер с операционной системой Windows, на котором был установлен IIS.

Затем мы создали новый сайт в IIS и сконфигурировали его для размещения нашего приложения. Это включало указание пути к файлам приложения, настройку параметров безопасности и другие необходимые настройки.

Использование Ngrok для доступа извне:

Для того чтобы позволить внешним пользователям обращаться к нашему приложению, мы воспользовались сервисом Ngrok.

После установки Ngrok на нашем сервере или локальном компьютере, мы создали туннель к нашему серверу IIS, указав порт, на котором работало наше приложение.

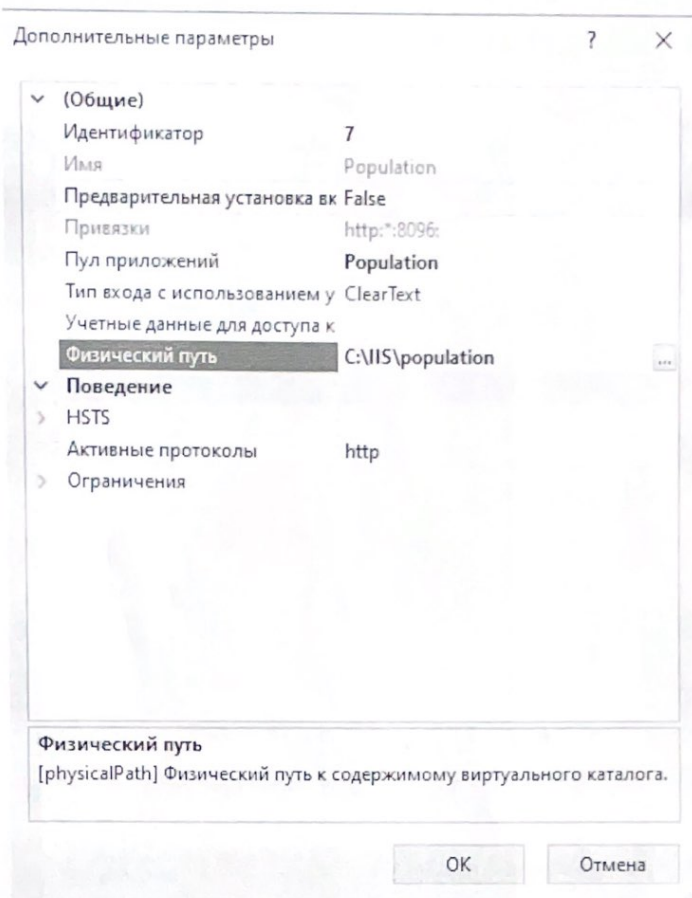


Рис. 2. Интерфейс дополнительные параметры сайта

На этом рисунке показано, что серверная часть приложения опубликована по протоколу http, а путь к проекту находится на диске C в папке iis.

Замечание

Ngrok сгенерировал уникальный URL, через который стало возможным доступ к нашему приложению из интернета.

Проверка доступности приложения:

После завершения настройки мы проверили доступность приложения, сначала в локальной сети для убеждения в его корректной работе, а затем через предоставленный Ngrok URL для проверки доступа извне.

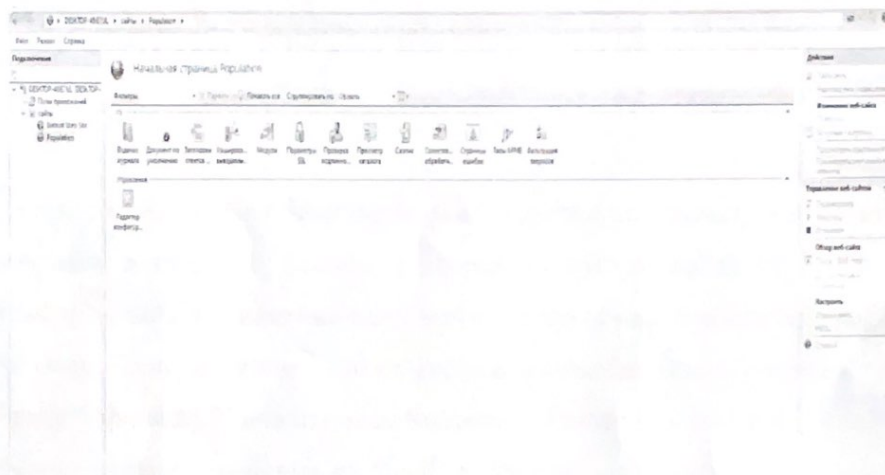


Рис. 1. Главный интерфейс ИИС

На этом рисунке показан iis-интерфейс и опубликована серверная часть программы, имя сайта population на порт 8096.

заполните!

На данном рисунке отображается интерфейс мониторинга запросов в программе Ngrok. Здесь представлена информация о запросах, поступающих на сервер через созданный туннель. Этот мониторинг обеспечивает видимость активности, позволяя отслеживать количество запросов, время ответа и другие параметры. Анализ этой информации помогает в оценке производительности сервера и выявлении потенциальных проблемных моментов, что в свою очередь способствует оптимизации работы приложения и обеспечивает более высокий уровень обслуживания пользователей.

3.3. Анализ результатов апробации

В процессе анализа обнаружились следующие проблемы. Первая ошибка заключалась в том, что фронтенд отправлял пустое значение ("") в поле "nationalityName", а серверная часть приложения проверяла наличие значения в этом поле. Если значение было не пустым, оно добавлялось как новая нация в таблицу "nationality", иначе - не добавлялось. Решением этой проблемы стало изменение пустого значения на "null" в фронтендной части, что позволило корректно обрабатывать этот случай на сервере.

Вторая проблема возникала во вкладке "Анкеты", где требовалось отправить данные для выбора вариантов. Изначально фронтенд должен был отобразить только пять вариантов, однако после добавления новых записей, таких как нация и религия, варианты стали отображаться все записи. Для решения этой проблемы была внесена изменения в запрос таким образом, чтобы фронтенд получал только пять записей, что позволило соблюсти требуемые условия отображения.

В этой главе были представлены результаты апробации разработанного приложения, а также описание процесса его использования. В ходе апробации были проанализированы различные аспекты функционирования приложения,

надо добавить
info решение
добавить!
можно вставить
рисунок

начиная от публикации на сервере с использованием IIS и Ngrok для доступа извне, и заканчивая анализом результатов тестирования и обнаружением ошибок.

Основные результаты данной главы включают в себя успешное развертывание приложения на сервере с использованием IIS и Ngrok, что обеспечило его доступность как в локальной сети, так и через интернет. Были также выявлены и решены различные проблемы, обнаруженные в ходе апробации, такие как некорректная обработка пустых значений и ошибки в отображении данных в пользовательском интерфейсе.

Общий вывод об этой главе подчеркивает важность апробации в процессе разработки программного обеспечения для выявления и исправления потенциальных проблем и обеспечения надежного функционирования приложения перед его внедрением.

Заключение
Список источников

В тексте не вижу []

Ссылка на источники.