Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работная работа по предмету «Проектирование информационных систем» на тему «Построение функциональной модели IDEF0»

Студент: Трубач Д. С.

ФИТ 4 курс 5 группа

Преподаватель: Якубенко К. Д.

Минск 2024

# 1 Постановка задач

Web-приложение должно быть выполнено с использованием асинхронного программирования, взаимодействовать с базой данных, реализовано под разными платформами. Web-приложение должно представлять собой web-приложение с асинхронным UI с использованием фреймворка React. Отображение, бизнес-логика и хранилище данных должны быть максимально независимы друг от друга для возможности расширения. Диаграмму вариантов использования разработать на основе UML, также необходимо разработать логическую схему базы данных и структурную схему приложения. Язык разработки проекта JavaScript, платформа «NodeJS». Развёртывание конечного приложения для последующего использования должно осуществляться с использованием Docker. Web-приложение должно быть логически завершенным. Управление программой должно быть интуитивно понятным и удобным. Листинги проекта должны содержать комментарии.

Функционально web-приложение должно:

* поддерживать роли гостя, пользователя, администратора;
* обеспечить возможность регистрации, авторизации;
* обеспечивать возможностью просмотра пунктов вторсырья;
* обеспечивать возможностью обмена своих накопленных баллов на скидки в различных сервисах;
* обеспечивать возможностями создания, изменения и удаления своих статей;
* обеспечивать возможностью оценки статей;
* обеспечивать возможностью получения информации сколько новой продукции будет сделано из сданных отходов.
* обеспечивать возможностью добавления новых пунктов приема вторсырья, изменение и удаление существующих;
* обеспечивать возможностью изменения и удаления видов вторсырья;
* обеспечивать возможностью добавления скидок, которые пользователи могут использовать за накопленные баллы, изменять и удалять уже существующие скидки;
* обеспечивать возможностью изменения и удаление статей о раздельном сборе отходов.

Основные задачи приложения:

* разработать удобный интерфейс, который позволит пользователям легко ориентироваться и находить информацию о сортировке отходов;
* предоставить функционал для ознакомления с пошаговыми инструкциями по раздельному сбору мусора;
* создать систему накопления баллов за сортировку, с возможностью их обмена на товары, произведенные из переработанных материалов.

# 2 Описание программных средств

Веб-приложение разрабатывалось с использованием ряда современных программных средств, обеспечивающих асинхронное взаимодействие с базой данных и поддержку различных платформ. Основная часть проекта выполнена с использованием языка программирования JavaScript и платформы NodeJS, которая обеспечивает высокую производительность и гибкость для реализации бизнес-логики приложения. Программная платформа NodeJS позволяет создавать кроссплатформенные приложения, поддерживает независимость между бизнес-логикой, пользовательским интерфейсом и хранилищем данных, что способствует масштабируемости и расширяемости проекта.

Для реализации асинхронного пользовательского интерфейса используется библиотека React, которая позволяет быстро обновлять данные на стороне клиента без перезагрузки страницы, обеспечивая более плавное и интерактивное взаимодействие пользователя с приложением. React поддерживает использование виртуальной DOM и компонентов, что ускоряет работу и делает код более поддерживаемым.

Для гарантирования безопасности пользовательских данных используются алгоритмы хэширования, такие как **bcrypt** или **argon2**, которые гарантируют, что даже в случае утечки базы данных пароли останутся защищенными и не будут доступны злоумышленникам в оригинальном виде. Важно также использовать **соль** (случайные данные, добавляемые к паролю перед хэшированием), что предотвращает использование радужных таблиц для дешифровки паролей.

Развёртывание конечного веб-приложения выполняется с использованием Docker, что позволяет создать изолированные контейнеры для различных компонентов системы, обеспечивая их гибкость и независимость при развертывании на различных платформах. Docker предоставляет возможность легко масштабировать приложение и управлять его зависимостями.

Моделирование и проектирование системы выполнено с использованием UML для разработки диаграммы вариантов использования, а также логической и структурной схемы базы данных, что позволяет визуализировать основные компоненты системы и их взаимодействие.

# 3 Описание практического задания

Разработка веб-приложения включала в себя создание функциональной модели, состоящей из нескольких уровней, каждый из которых описывает важные аспекты системы. Все уровни функциональной модели выполнены с использованием схем, которые демонстрируют основные элементы системы, их взаимодействие и потоки данных.

Первым шагом было создание диаграммы вариантов использования с помощью UML, которая визуализирует взаимодействие пользователя с системой. В качестве основных участников были выделены пользователи, администраторы и гости, что позволило определить основные функции приложения, такие как регистрация, авторизация, сдача вторсырья, публикация статей, управление пунктами вторсырья и выдача бонусных баллов.

Следующим этапом была разработка логической схемы базы данных, которая включает основные таблицы и их взаимосвязи. В этой схеме отражены такие ключевые сущности, как пользователи, пункты приема вторсырья, виды вторсырья, статьи и бонусы. Каждая таблица представлена с учетом всех необходимых полей и связей, что обеспечило целостность данных и их правильное хранение.

Также была создана структурная схема приложения, отражающая его архитектуру и взаимодействие компонентов. На этой схеме представлены все основные функциональные блоки приложения, включая клиентскую часть, серверную часть на платформе NodeJS, а также база данных. Особое внимание уделено независимости между блоками отображения, бизнес-логики и хранилища данных, что позволяет легко масштабировать и расширять систему. Асинхронный интерфейс пользователя на React взаимодействует с сервером через API, что обеспечивает быструю передачу данных в реальном времени.

Функциональные блоки системы разделены по задачам, такие как обработка авторизации и регистрации, управление видами и пунктамии приема вторсырья, обработка статей и комментариев. Взаимодействие между блоками реализовано через API, что позволяет выполнять обмен данными в реальном времени. Потоки данных в приложении проходят между пользователем, сервером и базой данных через стандартные HTTP-запросы, SQL-запросы для хранения и извлечения данных.

Таким образом, разработанная функциональная модель обеспечивает наглядное понимание системы, а описание блоков и потоков данных демонстрирует их взаимодействие и роль в общей архитектуре приложения.

# 4 Контекстная диаграмма и диаграмма 1-го уровня декомпозиции

Контекстная диаграмма — это верхнеуровневая диаграмма потоков данных (DFD, Data Flow Diagram), которая представляет всю информационную систему в виде одного процесса и его взаимодействие с внешними сущностями. Моделирование делового процесса начинается с построения контекстной диаграммы. На этой диаграмме отображается только один блок – главная бизнес-функция моделируемой системы. Если речь идет о моделировании целого предприятия, то главная бизнес-функция не может быть сформулирована как, например, «продавать продукцию».

Главная бизнес-функция системы – это «миссия» системы, ее значение в окружающем мире. Нельзя правильно сформулировать главную функцию предприятия, не имея представления о его стратегии. При определении главной бизнес функции необходимо всегда иметь ввиду цель моделирования и точку зрения на модель.

Контекстная диаграмма в функциональной модели еще «фиксирует» границы моделируемой бизнес-системы, определяя то, как моделируемая система взаимодействует со своим окружением. Это достигается за счет описания дуг, соединенных с блоком, представляющим главную бизнес-функцию.

Рассмотрим диаграмму получения информации – рисунок 4.1. Входными данными должны быть данные о количестве сданных отходов, вид сдаваемых отходов и фактор подтверждения. Как и с поиском, нам необходима валидация данных. Механизмы: база данных (сохранение и поиск соответствующей информации), пользователь и администратор приложения, которые обращаются к функции программного средства. На выходе мы получаем информацию о новой продукции, которая будет произведена из сданных отходов.

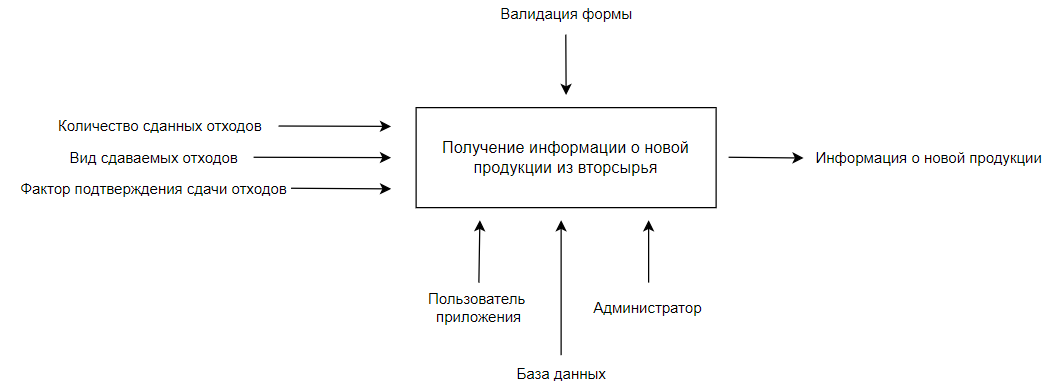


Рисунок 4.1 – Контекстная диаграмма получения информации

Принцип функциональной декомпозиции представляет собой способ моделирования типовой ситуации, когда любое действие, операция, функция могут быть разбиты (декомпозированы) на более простые действия, операции, функции. То есть сложная бизнес-логика представлена в виде совокупности элементарных блоков.

При построении диаграммы 1-го уровня декомпозиции бизнес-функция приложения должна быть представлена в виде совокупности элементарных функций. Количество блоков на диаграмме должно быть не менее двух, но и не более шести. Соблюдение этого принципа приводит к тому, что функциональные процессы, представленные в виде IDEF0 модели, хорошо структурированы, понятны и легко поддаются анализу.

На рисунке 4.2 представлена диаграмма декомпозиции 1-го уровня (А0).

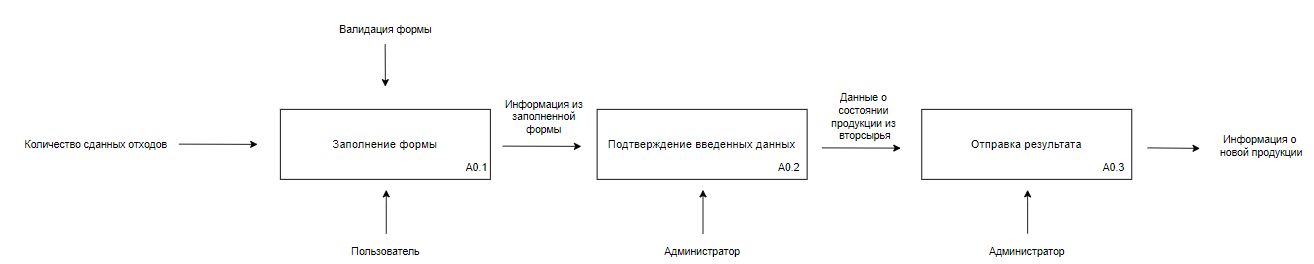


Рисунок 4.2 – Диаграмма А0

На входе в первый блок «Заполнение формы» приходит количество сданных отходов, которые пользователи вписывает в форму, проверяя данные через механизм валидации. На выходе данный блок отдает информации о заполненной форме.

Во второй блок «Подтверждение введенных данных» поступают данные о заполненной форме, которые администратор проверяет на соответствие стандартам или критериям. Администратор подтверждает корректность данных, и на выходе из блока получаются окончательные подтвержденные данные для дальнейшего использования.

Третий блок «Отправка результата» принимает подтвержденные данные о форме и обрабатывает их для отправки информации на перерабатывающие предприятия или в другие системы. Администратор отслеживает состояние отправки данных, и на выходе из блока поступает подтверждение успешной отправки информации в новые процессы или на предприятия.

# 5 Ответы на теоретические вопросы

1. В чем основная сущность структурного подхода?

Сущность структурного подхода к разработке модели состоит в расчленении анализируемой системы на части («черные ящики») и их иерархической организации.

2. Дайте расшифровку терминам DFD, IDEF и SADT.

DFD – Data Flow Diagrams – диаграммы потоков данных.

IDEF – Integration Definition Methodology – объединение методологических понятий.

SADT – Structured Analysis and Design Technique – методология структурного анализа и проектирования.

3. Какие модели строятся с помощью IDEF методологий?

С помощью методологии IDEF строятся функциональные модели, описывая бизнес-функции и контекст поведения.

4. Укажите базовые принципы моделирования в IDEF0.

В IDEF0 реализованы три базовых принципа моделирования процессов:

Принцип функциональной декомпозиции представляет собой способ моделирования типовой ситуации, когда любое действие, операций, функция могут быть разбиты (декомпозированы) на более простые действия, операции, функции. Т.е., сложная бизнес-функция может быть представлена в виде совокупности элементарных функций. Представляя функции графически, в виде блоков, можно «заглянуть внутрь» блока и детально рассмотреть ее структуру и состав.

Принцип ограничения сложности. При работе с IDEF0 диаграммами существенным является условие их разборчивости и удобочитаемости. Суть принципа ограничения сложности состоит в том, что количество блоков на диаграмме должно быть не менее двух и не более шести. Практика показывает, что соблюдение этого принципа приводит к тому, что функциональные процессы, представленные в виде IDEF0 модели, хорошо структурированы, понятны и легко поддаются анализу.

Принцип контекстной диаграммы. Моделирование делового процесса начинается с построения контекстной диаграммы. На этой диаграмме отображается только один блок – главная бизнес-функция моделируемой системы. Если речь идет о моделировании целого предприятия, то главная бизнес-функция не может быть сформулирована как, например, "продавать продукцию". Главная бизнес-функция системы – это "миссия" системы, ее значение в окружающем мире. Нельзя правильно сформулировать главную функцию предприятия, не имея представления о его стратегии. При определении главной бизнесфункции необходимо всегда иметь ввиду цель моделирования и точку зрения на модель.

5. В каких случаях целесообразно применять построение модели «как есть, в а каких «как будет»?

Построение модели “как есть”. Обследование предприятия является обязательной частью любого проекта создания или развития корпоративной информационной системы. Построение функциональной модели “как есть” позволяет четко зафиксировать, какие деловые процессы осуществляются на предприятии, какие информационные объекты используются при выполнении деловых процессов и отдельных операций. Модель “как есть” является отправной точкой для анализа потребностей предприятия, выявления проблем и "узких" мест и разработки проекта совершенствования деловых процессов.

Построение модели “как будет”. Создание и внедрение корпоративной информационной системы приводит к изменению условий выполнения отдельных операций, структуры деловых процессов и предприятия в целом. Это приводит к необходимости изменения системы бизнес-правил, используемых на предприятии, модификации должностных инструкций сотрудников. Модель “как будет” позволяет уже на стадии проектирования будущей информационной системы определить эти изменения. Применение функциональной модели “как будет” позволяет не только сократить сроки внедрения информационной системы, но также снизить риски, связанные с невосприимчивостью персонала к информационным технологиям