Министерство сельского хозяйства Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Пермский государственный аграрно-технологический университет

имени академика Д.Н. Прянишникова»

Кафедра Информационных технологий

и программной инженерии

**ОТЧЁТ ПО УЧЕБНОЙ ПРАКТИКЕ**

(проектно-технологическая практика)

на тему: «Разработка информационной системы

для автоматизации бизнес-процесса»

Выполнил:

студент группы ПИб-2019

направления подготовки

09.03.03 Прикладная информатика

Солдатова Ирина Олеговна

Проверил:

доцент кафедры ИТиПИ, к.т.н., доцент

Беляков Андрей Юрьевич

2022

Содержание

[Постановка задачи на проектирование 3](#_Toc105536466)

[Анализ технологий проектирования 4](#_Toc105536467)

[Реализация функционала информационной системы 9](#_Toc105536468)

[Заключение 12](#_Toc105536469)

[Список источников 13](#_Toc105536470)

[ПРИЛОЖЕНИЕ А 14](#_Toc105536471)

[ПРИЛОЖЕНИЕ Б 20](#_Toc105536472)

[РЕЦЕНЗИЯ 26](#_Toc105536473)

# Постановка задачи на проектирование

Разработать программную реализацию бизнес процесса обратной связи от пользователей и клиентов.

Программа разрабатывается для оптимизации бизнес-процесса. Важная составная часть его это обратная связь от клиентов о продукции, об услугах, об оказанном сервисе.

**Отзывы клиентов** — обратная связь от пользователей на ваши продукты или услуги и обязательный атрибут бизнеса. Однако компании не всегда понимают их значение: не собирают, реагируют только на положительные отклики и не работают с негативными. Такой подход — ошибка.

Отзывы формируют лояльность клиента, информируют пользователей о продукте нативно, помогают выявить недостатки, улучшить продукт и принести компании увеличение продаж.

Подавляющее большинство покупателей читают отзывы перед покупкой, поэтому игнорировать обратную связь от клиентов не стоит.

Отзывы формируют доверие клиента. Продукт без отзывов вызывает опасения. Продукт исключительно с положительными отзывами - недоверие. Если компания на своём сайте публикуют только положительные отзывы, то пользователи уходят на независимые площадки, типа Otzovik.com или Яндекс.Маркет, просматривают соцсети или ролики видео-блоггеров.

Недобросовестные компании игнорируют и удаляют негативные отзывы, но такой подход только раздражает клиентов.

Отзывы покупателей — это ещё и увеличение объёма продаж. Девять из десяти заказов онлайн делаются людьми после изучения отзывов. Хороший подробный комментарий убедительнее любой рекламной кампании. Он помогает клиентам понять, подойдёт ли им товар. Исследование «AliExpress Россия» показало, что с такой целью к чужому мнению обращаются 65% покупателей. Через рассказы пользователей люди находят ту информацию, которой нет в стандартном описании. Например, пользовательский опыт эксплуатации продукта.

В основе сервисов бронирования отелей и посуточного жилья именно отзывы. Никакие описания и красивые фотографии не спасут, если кто-то напишет о тараканах в номере.

Программная реализация должна: содержать файл в катором храняться отзывы клиентов и отражаться на web-странице, записывать новый отзыв клиента.

Интрефейс web-приложения должен быть простым и интуетивно понятен для пользователя и всегда иметь возможность вернуться на шаг назад.

Техническое обеспечение системы должно максимально и наиболее эффективно использовать существующее в отделе автоматизации оборудование:

* процессор – 2х Intel Xeon 3.7 ГГц;
* оперативная память – 32 ГБ;
* дисковая система – 2 х 1ТБ;
* сетевой адаптер – 1 Гб/с.

# Анализ технологий проектирования

**SQLite** — это быстрая и легкая встраиваемая однофайловая СУБД на языке C, которая не имеет сервера и позволяет хранить всю базу локально на одном устройстве. Для работы SQLite не нужны сторонние библиотеки или службы.

База данных — это набор структурированной информации. Для ее изменения требуются системы управления — СУБД. Как и любая СУБД, SQLite позволяет записывать новую и запрашивать существующую информацию, изменять ее, настраивать доступ.

Благодаря свойствам SQLite применяется:

* на сайтах с низким и средним трафиком;
* в локальных однопользовательских, мобильных приложениях или играх, не предназначенных для масштабирования;
* в программах, которые часто выполняют прямые операции чтения/записи на диск;
* в приложениях для тестирования бизнес-логики.

SQLite не требует администрирования и работает на мобильных устройствах, игровых приставках, телевизорах, беспилотных летательных аппаратах, камерах, автомобильных мультимедийных системах и т.д. СУБД использует множество программ: Firefox, Chrome, Safari, Skype, XnView, AIMP, Dropbox, Viber и другие.

Большинство СУБД используют клиент-серверную архитектуру: данные хранятся и обрабатываются на сервере, а запросы к нему посылает клиент. «Клиент» — это часть программы, с которой взаимодействует пользователь. «Сервером» может быть и отдельный процесс на том же компьютере (так называемый демон), и стороннее устройство, как в случае с сайтами.

SQLite устроена иначе и не имеет сервера. Это значит, что все данные программное обеспечение хранит на одном устройстве. СУБД встраивается в приложение и работает как его составная часть. Если установить на компьютер программу, использующую SQLite, то база данных тоже будет храниться на нем же. Формат базы — один текстовый файл, который можно прочитать на любой платформе. Такой подход повышает производительность и скорость работы.

**MySQL** — это реляционная система управления базами данных (СУБД), которая распространяется как свободное программное обеспечение. Является одной из наиболее популярных, так как отличается гибкостью, легкостью, удобством в использовании.

**Кто пользуется MySQL**

* Бэкенд-разработчики, задача которых — управлять базой данной на серверной стороне сайта или приложения.
* Разработчики десктопных и мобильных приложений, в которых есть база данных для хранения информации.
* Администраторы баз данных — специалисты, которые занимаются построением и настройкой базы.

В основном MySQL используется для работы со средними и небольшими проектами. Она имеет открытый исходный код, поэтому любой разработчик может с ним ознакомиться. Сейчас систему поддерживают известная компания Oracle и независимое сообщество разработчиков-энтузиастов.

**PostgreSQL** — это объектно-реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых СУБД в мире. Имеет открытый исходный код и является альтернативой коммерческим базам данных.

**Кто пользуется PostgreSQL**

* Бэкенд-разработчики, которым приходится взаимодействовать с базами данных при работе с «серверной» частью сайта.
* Администраторы и разработчики баз данных — специалисты, основная задача которых заключается в обслуживании и поддержке работоспособности базы.

DevOps-инженеры, обслуживающие рабочую инфраструктуру проекта.

**Для чего нужна PostgreSQL**

* Гибкий доступ к базам данных, их организация и хранение.
* Управление записями в базах данных: создание, редактирование и удаление, обновление версий и так далее.
* Просмотр нужной информации из базы по запросу, например, для ее отправки на сайт или в интерфейс приложения.
* Отправка транзакций, последовательных запросов, собранных в подобие скрипта.
* Настройка и контроль доступа к той или иной информации, группировка пользователей по уровню прав.
* Контроль версий и организация одновременного доступа к базе из разных источников так, чтобы предотвратить сбои.
* Защита информации от возможных утечек и потерь.
* Контроль состояния базы в целом.

**С# («Си Шарп»)** – один из наиболее быстро растущих, востребованных и при этом «удобных» языков программирования. Это модификация фундаментального языка С от компании Microsoft, призванная создать наиболее универсальное средство для разработки программного обеспечения для большого количества устройств и операционных систем.

С# – это объектно-ориентированный язык программирования. Он был создан в период с 1998 по 2002 год командой инженеров Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота.

Язык входит в семью С-подобных языков. Синтаксис приближен к Java и C++. Его особенности:

* статистическая типизация,
* поддерживается полиморфизм,
* поддерживается перегрузка операторов,
* доступна делегация, атрибуты, события, обобщенные типы и анонимные функции.

**Основные преимущества языка**

С# популярен за счет своей «простоты». Простоты для современных программистов и больших команд разработчиков, чтобы те могли в сжатые сроки создавать функциональные и производительные приложения. Этому способствуют нетипичные конструкции языка и специфичный синтаксис, помогающий максимально органично реализовать намеченные функции.

Понятный синтаксис C# заметно упрощает не только разработку как таковую, но и другие важные аспекты совместной работы, например, чтение чужого кода. Это упрощает процесс рефакторинга и исправления ошибок при работе над приложениями в больших командах.

Также нельзя не упомянуть низкий порог вхождения. С# – популярная и достаточно простая в освоении технология.

**Что разрабатывают с помощью С#**

Язык C# практически универсален. Можно использовать его для создания любого ПО: продвинутых бизнес-приложений, видеоигр, функциональных веб-приложений, приложений для Windows, macOS, мобильных программ для iOS и Android.

**Python** (произносится как «пайтон», но многие говорят «питон») – это язык программирования общего назначения, который широко применяется в различных областях: от создания банальных веб-страниц до систем управления роверами на других планетах.

Язык скриптовый, он универсален и является самым популярным языком программирования в мире (по данным издания [Tiobe](https://timeweb.com/go?url=https%3A%2F%2Ftiobe.com&hash=97466ae372597e81d32e0afaa910831091d2e5d7" \t "_blank), составляющего топы наиболее востребованных языков).

**Что пишут на Python?**

Что вздумается. Любой программный продукт, независимо от сложности и специфики, можно создать на Python и одном из сотен фреймворков, разработанных для него.

**Веб-разработка**

Для создания веб-приложений и сайтов используются сторонние библиотеки в духе Pyromid, Flask, Django и десятков других, помогающих заменить PHP и JavaScript.

На Python даже полноценные системы управления данными пишут, которые впоследствии становятся аналогами мастодонтов наподобие WordPress и Drupal. Также на Python создают вспомогательные инструменты для работы с вебом, например, скрапперы, собирающие информацию с чужих веб-страниц.

**Node.js** – это кроссплатформенная среда выполнения с открытым исходным кодом для разработки серверных и сетевых приложений. Приложения Node.js написаны на JavaScript и могут запускаться в среде выполнения Node.js в OS X, Microsoft Windows и Linux.

Node.js можно установить на сервер (так же, как Python) и исполнять на нём ваш код, отдавая результат исполнения пользователям. На нём же можно делать отдельные приложения, используя дополнительные фреймворки.

Node.js также предоставляет богатую библиотеку различных модулей JavaScript, которая значительно упрощает разработку веб-приложений с использованием Node.js.

**Особенности Node.js**

Ниже приведены некоторые важные функции, которые делают Node.js лучшим выбором для архитекторов программного обеспечения.

* **Асинхронный и управляемый событиями** – все API библиотеки Node.js являются асинхронными, то есть неблокирующими. По сути, это означает, что сервер на базе Node.js никогда не ждет, пока API вернет данные. Сервер переходит к следующему API после его вызова, а механизм уведомления о событиях Node.js помогает серверу получить ответ от предыдущего вызова API.
* **Очень быстрая** – библиотека Node.js, созданная на основе движка Google Chrome V8 JavaScript, очень быстро выполняет код.
* **Однопоточный, но очень масштабируемый** – Node.js использует однопотоковую модель с зацикливанием событий. Механизм событий помогает серверу реагировать неблокирующим образом и обеспечивает высокую масштабируемость сервера по сравнению с традиционными серверами, которые создают ограниченные потоки для обработки запросов. Node.js использует однопотоковую программу, и та же самая программа может обслуживать гораздо большее количество запросов, чем традиционные серверы, такие как Apache HTTP Server.
* **Без буферизации** – приложения Node.js никогда не буферизуют данные. Эти приложения просто выводят данные кусками.
* **Лицензия** – Node.js выпущен под лицензией MIT .

**Асинхронный и управляемый событиями** – все API библиотеки Node.js являются асинхронными, то есть неблокирующими. По сути, это означает, что сервер на базе Node.js никогда не ждет, пока API вернет данные. Сервер переходит к следующему API после его вызова, а механизм уведомления о событиях Node.js помогает серверу получить ответ от предыдущего вызова API.

**Очень быстрая** – библиотека Node.js, созданная на основе движка Google Chrome V8 JavaScript, очень быстро выполняет код.

**Однопоточный, но очень масштабируемый** – Node.js использует однопотоковую модель с зацикливанием событий. Механизм событий помогает серверу реагировать неблокирующим образом и обеспечивает высокую масштабируемость сервера по сравнению с традиционными серверами, которые создают ограниченные потоки для обработки запросов. Node.js использует однопотоковую программу, и та же самая программа может обслуживать гораздо большее количество запросов, чем традиционные серверы, такие как Apache HTTP Server.

**Без буферизации** – приложения Node.js никогда не буферизуют данные. Эти приложения просто выводят данные кусками.

**Лицензия** – Node.js выпущен под лицензией MIT .

# Реализация функционала информационной системы

В рамках практики будет разработано web-приложение "**Отзывы на продукт**".

**Используемые технологии:**

|  |  |
| --- | --- |
| * Nodejs * Express * ejs * MVC * csv * json | * re * SQLite * html * css * lodash |

**Архитектура приложения** основана на паттерне проектирования **MVC** (Model-View-Controller).

Составные части паттерна:

**model** - модель - это часть архитектуры приложения:

- включает в себя бизнес-логику приложения (классы, методы, функции обработки данных);

- модель "не знает" о контроллерах и представлениях, не ориентируется на них;

- данные модели могут быть представлены таблицами базы данных или файлами XML, JSON, csv;

- это может быть просто набор объектов или функций для реализации логика приложения.

**view** - представление - это часть архитектуры приложения (примеры представления: HTML-страница, WPF форма, Windows Form):

- отвечает за визуальное отображение данных, полученных от модели

- представление может читать данные, но не может записывать или изменять их

- представление может иметь программный код, который реализует логику отображения данных

**controller** - контроллер - это часть архитектуры приложения:

- совокупность обработчиков событий представления

- принимает события от представления

- отдаёт событие на обработку в модель

- возвращает пользователю обновлённое представление

- или выбирает какое именно представление должно быть отображено в ответ.

**Интерфейс пользователя**

На рис.1 представлено представление главной формы с кнопкой «Оставить новый отзыв», при нажатии который открывается второстепенная форма для заполнения отзыва, и с таблицей, отражающей оставленные ранее пользователями отзывы.

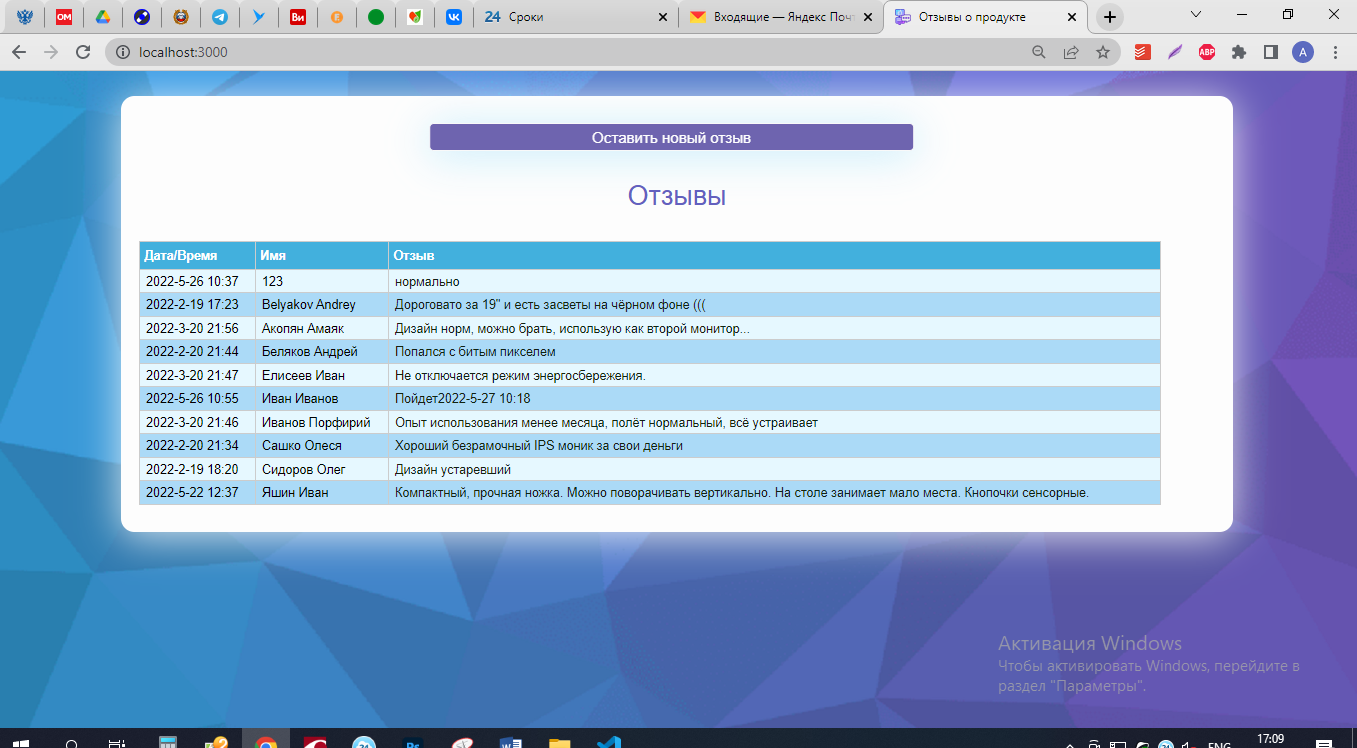


Рисунок 1 – Главная форма страницы с отображением оставленных пользователями отзывов

На рис. 2 можем увидеть отображение формы с пустой таблицей отзывов пользователей.

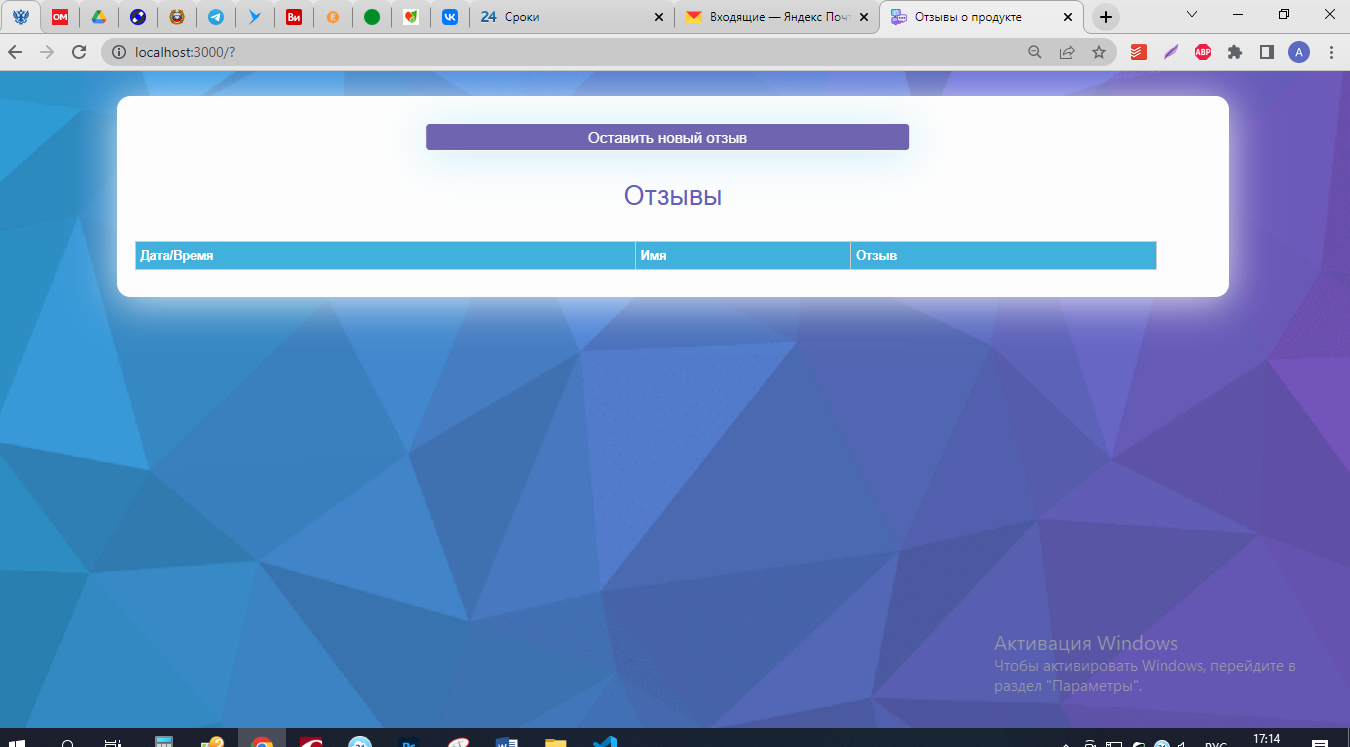


Рисунок 2– Главная форма страницы, пустая

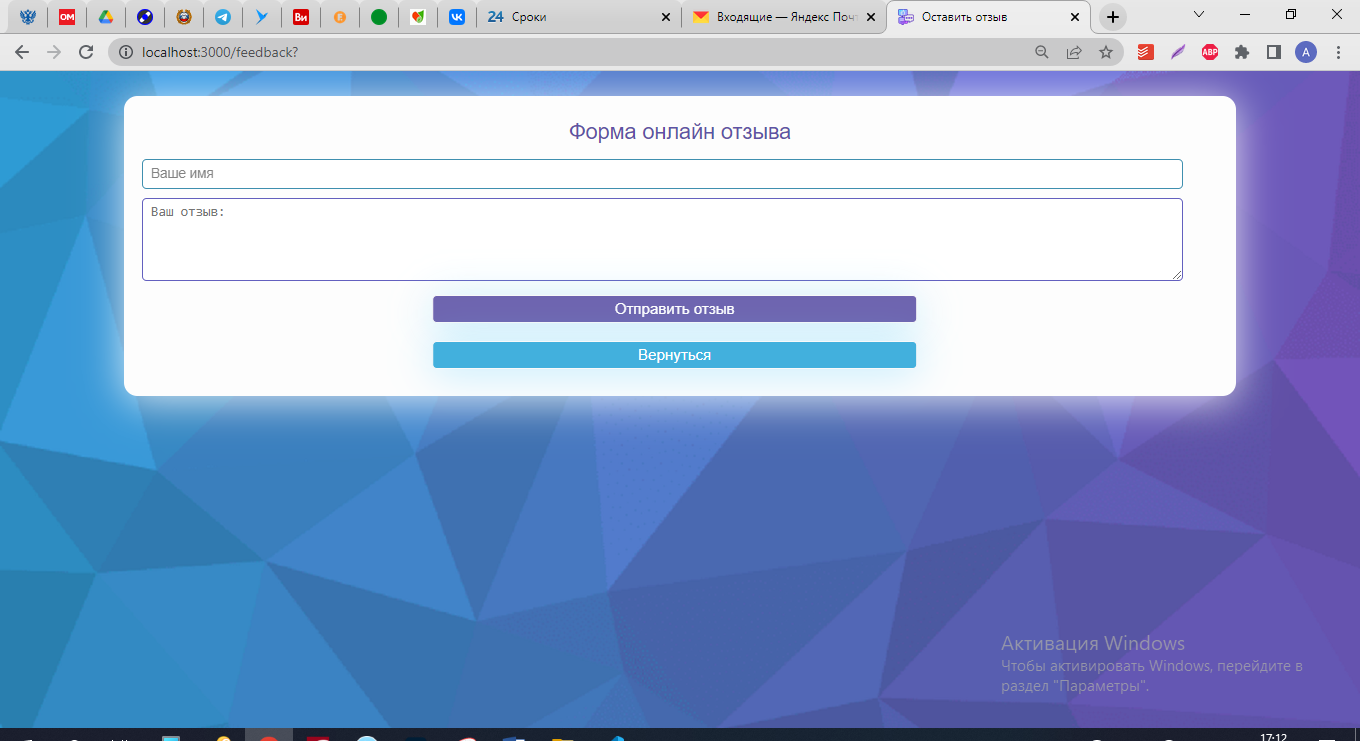


Рисунок 3 – Второстепенная форма для заполнения отзыва

На рис.3 можем видеть второстепенную форму для заполнения отзыва пользователем с кнопками «Отправить отзыв» и «Вернуться», и с полями «Ваше имя» и «Ваш отзыв:»

При нажатии кнопки «Отправить отзыв» данные с заполненных полей отправляются в таблицу с оставленными ранее отзывами.

При нажатии кнопки «Вернуться» мы можем вернуться на главную форму.

# Заключение

На учебной проектно-технологическая практике я изучила и получила опты работы с технологиями технологии: Nodejs, Express, ejs, MVC, сsv, json, html, css. Изучила и научилась создавать архитектуру приложения на паттерне проектирования **MVC** (Model-View-Controller).

Изучила и получила опыт работы с языками программирования C#, **Python**, а также с кроссплатформенной средой выполнения с открытым исходным кодом для разработки серверных и сетевых приложений **- Node.js**.

В рамках практики было разработано web-приложение "**Отзывы на продукт**" Для оптимизации бизнес-процесса. Важная составная часть его это обратная связь от клиентов о продукции, об услугах, об оказанном сервисе. Подобные разработки web-приложений имеют большие перспективы в будущем ведь они оптимизируют и ускоряют рабочий процесс. Приносят пользу как пользователям, так и компаниям по отслеживанию качества продукции, услуг, оказанном сервисе.

# Список источников

1. ГОСТ 19.401-78 Текст программы. требования к содержанию и оформлению.
2. ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы.
3. ГОСТ 19.701-90 Схемы алгоритмов, программ данных и систем.
4. Алгоритм хеширования MD5. [Электронный ресурс] – Электрон. текстовые дан. Режим доступа: https://ru.wikipedia.org/wiki/MD5
5. Заяц, А.М. Проектирование и разработка WEB-приложений/ А.М. Заяц, Н.П. Васильев- Санкт-Петербург: Лань, 2021.-120с.
6. МЕТОДИЧЕСКОЕ ПОСОБИЕ по выполнению курсового проекта по дисциплине Объектно-ориентированное программирование. Режим доступа: <https://pcoding.ru/pdf/CourseProject.pdf>
7. Git Repositories Documentation [Электронный ресурс] – Электрон. текстовые дан. – Microsoft, Режим доступа: https://docs.microsoft.com/ru-ru/azure/devops/repos/git, свободный.
8. MSDN. Техническая документация. MD5 Класс. [Электронный ресурс] Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/api/system.security.cryptography.md5?view=netframework-4.8
9. MSDN. Техническая документация. HashAlgorithm.ComputeHash Метод. [Электронный ресурс] Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/api/system.security.cryptography.hashalgorithm.computehash?view=netframework-4.8>

ПРИЛОЖЕНИЕ А

наименование организации – разработчика ТЗ на АС

УТВЕРЖДАЮ

Руководитель

(должность, наименование предприятия – заказчика АС)

Личная подпись Расшифровка подписи

(печать)

Дата

УТВЕРЖДАЮ

Руководитель

(должность, наименование предприятия – разработчик АС)

Личная подпись Расшифровка подписи

(печать)

Дата

наименование вида АС

наименование объекта автоматизации

сокращённое наименование АС

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

На листах

Действует с

2022

1. Общие сведения
2. Наименование системы

Полное наименование разрабатываемой системы – «web-приложение «Отзывы на продукт».

Краткое наименование – «Модуль «Отзывы на продукт».

1. Наименование заказчика и исполнителя

Организация: ФГБОУ ВО Пермский ГАТУ.

Адрес: ул. Петропавловская, 23.

Телефон: +7 (342) 217-90-66;

Исполнитель: Солдатова Ирина Олеговна

1. Плановые сроки начала и окончания работ

Дата начала работ: 24.05.2022.

Дата окончания работ: 07.06.2022.

1. Назначение и цели создания системы

К целям создания модуля «Отзывы на продукт» можно отнести:

* улучшение оперативного взаимодействия и интеграция модулей;
* автоматизировать опрос сотрудников и студентов университета.

Достижение целей приведёт к следующим положительным результатам:

* руководитель сможет быстрее оценивать оперативную картину видя оставленные отзывы;
* благодаря возможности отставить отзыв разработчики смогут получать реальную оценку работы портала;

1. Характеристика объекта автоматизации

Объектом автоматизации является подсистема организационной коммуникации корпоративного портала ФГБОУ ВО Пермский ГАТУ в виде оставленных отзывов.

1. Требования к системе

Общие требования к модулю «Отзывы на продукт» являются:

* надёжность и работоспособность;
* интуитивно понятный интерфейс;
* лицензионная чистота – применение средств в рамках общего лицензионного соглашения касательно корпоративного портала;
* соблюдение информационной безопасности.

1. Требования к способам и средствам связи для информационного обмена между компонентами

Для обеспечения информационного обмена компоненты подсистемы должны взаимодействовать с объединённой информационной базой данных. Благодаря хранению данных в различных схемах, таких как мероприятия или поручения модуль «Отзывы на продукт» может объединить эти данные представив их как единый информационный поток.

1. Перспективы развития, модернизация системы

Дальнейшим развитием модуля «Отзывы на продукт» может быть объединение иных подсистем корпоративного портала.

1. Требования к квалификации персонала и режиму его работы

Для эксплуатации модуля «Отзывы на продукт» определены следующие роли:

* системный администратор – должен быть квалифицированным специалистом с практическим опытом выполнения работ по администрированию программных и технических средств. В обязанности входит: установка, модернизация, настройка программного обеспечения, ведение учётных записей портала;
* пользователь портала – должен иметь опыт работы с персональным компьютером на уровне опытного пользователя и свободно осуществлять базовые действия с корпоративным порталом посредством браузера с доступом в интернет.

1. Требования к надёжности технических средств и программного обеспечения

Надёжность по отношению к техническим средствам должна обеспечиваться использованием в системе средств повышенной отказоустойчивости и их резервированием.

Надёжность программного комплекса обеспечивается использованием сертифицированных операционных систем, общесистемных программных средств и инструментальных программных систем, используемых при разработке программного обеспечения. Само программное обеспечение должно обеспечивать защиту от некорректных действий пользователей и ошибочных исходных данных.

1. Требования к безопасности

Разрабатываемый информационный модуль «Отзывы на продукт» должен обеспечивать безопасный доступ к данным, предотвращая несанкционированный доступ или модифицирование данных.

1. Требования по эргономике и технической эстетике

Модуль должен иметь удобный и интуитивно понятный графический пользовательский интерфейс. Диалоговый интерфейс должен соблюдать контекст подсистемы организационной коммуникации и управления в целом, тем самым действия конечного пользователя должны быть ясны и знакомы.

1. Требования к программному обеспечению

При проектировании информационного модуля «Отзывы на продукт» необходимо эффективно использовать используемое на данный момент в проекте корпоративного портала применяется веб-фреймворк Vue.js, в качестве серверного окружения используется программная платформа Node.js.

1. Требования к техническому обеспечению

Техническое обеспечение системы должно максимально и наиболее эффективно использовать существующее в отделе автоматизации оборудование:

* процессор – 2х Intel Xeon 3.7 ГГц;
* оперативная память – 32 ГБ;
* дисковая система – 2 х 1ТБ;
* сетевой адаптер – 1 Гб/с.

1. Порядок контроля и приёмки системы

Приёмо-сдаточные испытания системы проводятся с привлечением сотрудников отдела автоматизации. По результатам опытной эксплуатации оформляется акт о приёме работ. Акт содержит заключение о соответствии системы техническому заданию.

1. Требования к составу и содержанию работ подготовки объекта автоматизации к вводу системы в действие

При подготовке к вводу в эксплуатацию модуля «Отзывы на продукт» отдел управления информатизации должен обеспечить выполнение следующих работ:

* определить подразделение и ответственных должностных лиц для внедрения информационного модуля;
* обеспечить пользователей руководством, которое поможет быстрее освоить внедрённый модуль;
* провести опытную эксплуатацию модуля «Отзывы на продукт».

# ПРИЛОЖЕНИЕ Б

**Листинг кода формы авторизации – FormAutho.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace word\_blank

{

**Листинг кода app.js**

// подключение зависимостей

const express = require('express');

const tools = require('./private/tools');

const params = require('./private/config.json').debug;

// настройка приложения

const app = express();

const htmlParser = express.urlencoded({extended: false});

app.use('/images', express.static(\_\_dirname + '/images'));

app.use('/css', express.static(\_\_dirname + '/css'));

app.set('view engine', 'ejs'); // npm i ejs

/\* паттерн MVC \*/

// models - модели данных

const md = require('./models/model');

// controllers - обработчики событий

app.get('/', function (req, res) { // главная страница

let file\_records = './private/feeds.csv'; // файл с отзывами

md.md\_index.feeds = md.get\_records(file\_records); // получить из csv-файла

// md.md\_index.feeds = md.md\_index.feeds.reverse(); // новые записи вверху списка

md.md\_index.feeds = md.md\_index.feeds.sort((a,b) => a.name>b.name? +1: -1);

// md.md\_index.feeds = md.md\_index.feeds.sort((a,b) => a.date\_time>b.date\_time? -1: +1);

res.render('index', md.md\_index); // render view

});

app.get('/feedback', (req, res) => {

res.render("feedback.ejs", md.md\_feed); // render view

});

app.post('/feedback', htmlParser, (req, res) => {

// if (!req.body) return res.sendStatus(400);

tools.append\_record(req.body.name, req.body.feed); // находим по имени в шаблоне

res.redirect("/"); // возвращаемся на главную

});

// запуск приложения

app.listen(params.port, params.hostname, () => {

console.log(`>>> ${params.hostname}:${params.port}/\n>>> to stop: Ctrl+C`);

});

…**Листинг кода CSS– Feedback.cs**

body {

background: url(/images/bg.png) repeat ;

background: url(/images/bg.png)repeat , url(/images/bg1.png)repeat ;

font: normal normal 14px/20px Arial, Georgia, serif;

background-position: center center;/\*центруем картинку\*/

background-size: cover;/\*адаптивная картинка\*/

background-attachment: fixed;/\*фон зафиксирован\*/

}

h2 {

color: #5E53A6;/\*заголовок\*/

font-size: 24px;

font-weight: lighter;

position: relative;

/\*transform: translate(38%, 38%);\*/

text-align: center;

}

h3 {

color: #5E53A6;/\*заголовок\*/

font-size: 20px;

font-weight: lighter;

position: relative;

text-align: center;;

}

h4 {

color: #6360BF;/\*заголовок Отзывы\*/

font-size: 30px;

font-weight: lighter;

position: relative;

text-align: center;;

}

a {

text-decoration: none;

font-weight: lighter;

color: #5E53A6;

font-size: 20px;

position: relative;

text-align: center;;

}

a:hover {

color: #bb2;

}

.feed {

color: #121;

font-size: 14px;

font-weight: normal;

font: serif, Georgia;

}

table {

width: 95%;

border-collapse: collapse;

}

td, th {

border: 1px solid #CCC;

padding: 3px 7px 2px 7px;

}

th {

text-align: left;

padding: 5px;

background-color: #42B0DD;/\* верхняя 1 строка таблицы \*/

color: #fff;

}

tr:nth-child(odd) {

background-color: #abdaf7; /\* не четные строки стаблицы таблицы \*/

}

tr:nth-child(even) {

background-color: #E6F8FF; /\* четные строки стаблицы таблицы \*/

}

/\* Форма обратной связи \*/

#inline {

margin-left:30px;

width:80%;

margin:0 auto; /\* отступы \*/

background:rgb(253, 253, 253); /\* фон форм \*/

padding:10px 20px;

border-radius:15px;/\* скругление \*/

opacity:1; /\*прозрачность \*/

box-shadow: 0 5px 45px rgba(255, 255, 255, 0.8) ;

}

.txt {

display:inline-block;

color:#676767;

width:95%;

margin-bottom:10px;

border:1px solid rgb(41,131,166);/\* ободок поля ваше имя форма онлайн отзыва \*/

padding:5px 9px;

font-size:15px!important;

line-height:1.4em;

border-radius:5px;

opacity:0.90;

}

.txtarea {

display:inline-block;

color:#676767;/\* цвет текста при вводе в блоке ваш отзыв\*/

resize: vertical;

width:95%;

height:80px;

margin-bottom:10px;

border:1px solid #6360BF;/\* ободок поля ваш отзыв форма онлайн отзыва \*/

padding:5px 9px;

font-size:15px!important;

line-height:1.4em;

border-radius:5px;

}

.txt:focus, .txtarea:focus {

border-style:solid;

border-color:#BABABA;

color:#444;

}

input.error, textarea.error {

border-color:#973D3D;

border-style:solid;

background:#F0BEBE;

color:#A35959;

}

input.error:focus, textarea.error:focus {

border-color:#973D3D;

color:#A35959;

}

#send {

color:#FFFFFF;/\*кнопка оставить отзыв\*/

display:block;

cursor:pointer;

padding:5px 11px;

font-size:1.2em;

border:solid 1px #F9F9F9;

border-radius:5px;

background:#5E53A6;

width:45%;

margin-bottom:20px;

opacity:0.90;

position: relative;

left: 45%;

transform: translate(-40%, 0);

box-shadow: 0 5px 45px rgba(112, 211, 250, 0.466) ;

}

#send:hover {

background:#979797;

}

#back {

color:#FFFFFF; /\*кнопка вернуться\*/

display:block;

cursor:pointer;

padding:5px 11px;

font-size:1.2em;

border:solid 1px #F9F9F9;

border-radius:5px;

background:#42B0DD;

width:45%;

margin-bottom:20px;

position: relative;

left: 45%;

transform: translate(-40%, 0);

box-shadow: 0 5px 45px rgba(112, 211, 250, 0.466) ;

}

#back:hover {

background:#979797;

}

**Листинг кода Model– model.cs**

const fs = require('fs');

const csvjson = require('csvjson'); // npm i csvjson

const md\_index = {

title: "Отзывы о продукте",

heading: "Отзывы",

button: "Оставить новый отзыв",

feeds: []

};

const md\_feed = {

title: "Оставить отзыв",

heading: "Форма онлайн отзыва",

button: "Отправить отзыв"

};

// из csv файла в массив json - csv\_to\_json для заполнения md\_index.feeds

const get\_records = (nameFile, del='|') => {

let textCSV = fs.readFileSync(nameFile, { encoding: 'utf8' });

return csvjson.toObject(textCSV, { delimiter: del });

}

module.exports = {

md\_index,

md\_feed,

get\_records

}

**Листинг кода Private– config.json**

{

"debug": {

"port": 3000,

"hostname": "localhost"

},

"deploy": {

"port": 42011,

"hostname": "coding.1gb.ru"

},

"deployp": {

"port": 42025,

"hostname": "pcoding.1gb.ru"

}

}

**Листинг кода Private– feed.csv**

date\_time|name|feed

2022-2-19 17:23|Belyakov Andrey|Дороговато за 19" и есть засветы на чёрном фоне (((

2022-2-19 18:20|Сидоров Олег|Дизайн устаревший

2022-2-20 21:34|Сашко Олеся|Хороший безрамочный IPS моник за свои деньги

2022-2-20 21:44|Беляков Андрей|Попался с битым пикселем

2022-3-20 21:46|Иванов Порфирий|Опыт использования менее месяца, полёт нормальный, всё устраивает

2022-3-20 21:47|Елисеев Иван|Не отключается режим энергосбережения.

2022-3-20 21:56|Акопян Амаяк|Дизайн норм, можно брать, использую как второй монитор...

2022-5-22 12:37|Яшин Иван|Компактный, прочная ножка. Можно поворачивать вертикально. На столе занимает мало места. Кнопочки сенсорные.

2022-5-26 10:37|123|нормально

2022-5-26 10:55|Иван Иванов|Пойдет2022-5-27 10:18||

**Листинг кода Private– tools.js**

const fs = require('fs');

// добавим данные в файл

const append\_record = (nameUser, feed) => {

let date = new Date();

let year = date.getFullYear();

let month = date.getMonth()+1;

let day = date.getDate();

let hour = date.getHours();

let min = date.getMinutes();

let date\_time = `${year}-${month}-${day} ${hour}:${min}`;

feed = feed.replace(/(\r\n|\n|\r)/gm, ' ');

let record = `${date\_time}|${nameUser}|${feed}\n`;

fs.appendFile("./private/feeds.csv", record, (err) => {

if (err) return console.error(err);

}

)};

module.exports = {

append\_record

}

**Листинг кода views– feedback.ejs**

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title><%=title%></title>

<link rel="stylesheet" type="text/css" href="/css/feedback.css" />

<link rel="shortcut icon" href="/images/favicon .ico" />

</head>

<body>

<br>

<div id="inline">

<h2><%=heading%></h2>

<form action="/feedback" method="post"> <!-- вызов события /feedback -->

<input id="name" class="txt" name="name" type="name" placeholder="Ваше имя">

<textarea id="feed" class="txtarea" name="feed" placeholder="Ваш отзыв:"></textarea>

<button id="send"><%=button%></button>

</form>

<form> <!-- вызов главной страницы - событие / -->

<button id="back" type="submit" formaction="/">Вернуться</button>

</form>

</div>

</body>

</html>

**Листинг кода views – index.js**

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title><%=title%></title>

<link rel="stylesheet" type="text/css" href="/css/feedback.css" />

<link rel="shortcut icon" href="/images/favicon .ico" />

</head>

<body>

<br>

<div id="inline">

<br>

<form action="/feedback" method="get"> <!-- вызов события /feedback -->

<button id="send"><%=button%></button>

</form>

<h4><%=heading%></h4>

<table>

<tr>

<th>Дата/Время</th>

<th>Имя</th>

<th>Отзыв</th>

</tr>

<% for (let feed of feeds) {%>

<tr>

<td class="name"><%=feed.date\_time%></td>

<td class="name"><%=feed.name%></td>

<td class="feed"><%=feed.feed%></td>

</tr>

<%} %>

</table>

<br>

</div>

</body>

</html>

# РЕЦЕНЗИЯ

на курсовую работу \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

доцент кафедры ИТиПИ, к.т.н.,

доцент Беляков Андрей Юрьевич