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-21

направления подготовки

09.03.03 Прикладная информатика

Николаев Егор Антонович

Проверил:

доцент кафедры ИТиПИ, к.т.н., доцент

Беляков Андрей Юрьевич

Пермь – 2022

Оглавление

[Постановка задачи 3](#_Toc108645504)

[Анализ технологии проектирования 5](#_Toc108645505)

[Реализация функционала информационной системы 7](#_Toc108645506)

[Заключение 10](#_Toc108645507)

[Список источников 11](#_Toc108645508)

[Приложение А 12](#_Toc108645509)

[Приложение Б 13](#_Toc108645510)

[Приложение B 14](#_Toc108645511)

[Програмный код 15](#_Toc108645512)

## Постановка задачи

В рамках практики разработать web-приложение "**Отзыв 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

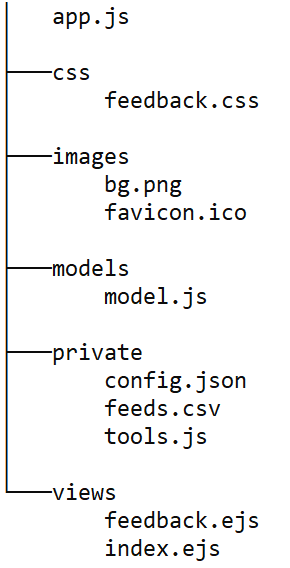
- принимает события от представления

- отдаёт событие на обработку в модель

- возвращает пользователю обновлённое представление

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

**Дерево директорий** приложения:

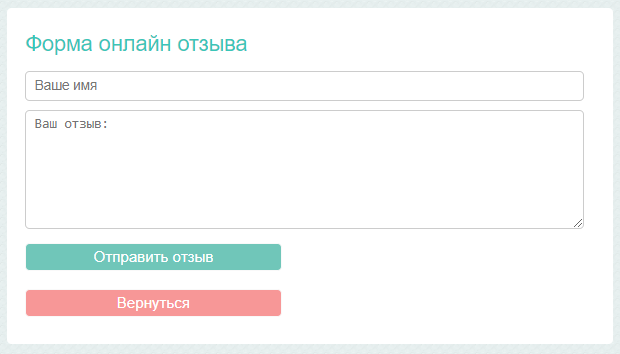


**Организация интерфейса пользователя** и функционал визуальных форм приложения.

Главная форма приложения:



Форма отзыва:



## Анализ технологии проектирования

SQLite — компактная встраиваемая СУБД. Исходный код библиотеки передан в общественное достояние. В 2005 году проект получил награду Google-O’Reilly Open Source Awards. Слово «встраиваемый» (embedded) означает, что SQLite не использует парадигмы клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а представляет собой библиотеку, с которой программа компонуется, и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется; ACID-функции достигаются в том числе за счёт создания файла журнала. Несколько процессов или потоков могут одновременно без каких-либо проблем читать данные из одной базы. Запись в базу можно осуществить только в том случае, если никаких других запросов в данный момент не обслуживается; в противном случае попытка записи оканчивается неудачей, и в программу возвращается код ошибки. Другим вариантом развития событий является автоматическое повторение попыток записи в течение заданного интервала времени. SQLite поддерживает динамическое типизирование данных. Возможные типы значений: INTEGER, REAL, TEXT и BLOB. Также поддерживается специальное значение NULL. Размеры значений типа TEXT и BLOB не ограничены ничем, кроме константы SQLITE\_MAX\_LENGTH в исходниках sqlite, равной миллиарду (109). Каждое значение в любом поле любой записи может быть любого из этих типов, независимо от типа, указанного при объявлении полей таблицы. Указанный при объявлении поля тип хранится для справки в его исходном написании, и используется в качестве основы для выбора предпочтений (так называемое «type affinity»: это подход, редко встречающийся в других СУБД) при выполнении неявных преобразований типов на основании похожести этого названия типа на что-либо, знакомое SQLite. В этот алгоритм зашит обширный перечень практикуемых в других СУБД вариантов названий типов данных. Если безопасного преобразования записываемого значения в предпочитаемый тип не получается, SQLite записывает значение в его исходном виде. Для получения значений из базы есть ряд функций для каждого из типов, и если тип хранимого значения не соответствует запрашиваемому, оно тоже, по возможности, преобразуется.

Node или Node.js — программная платформа, основанная на движке V8 (компилирующем JavaScript в машинный код), превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API, написанный на C++, подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на Node.js и десктопные оконные приложения (при помощи NW.js, AppJS или Electron для Linux, Windows и macOS) и даже программировать микроконтроллеры (например, tessel, low.js и espruino). В основе Node.js лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом.

## Реализация функционала информационной системы

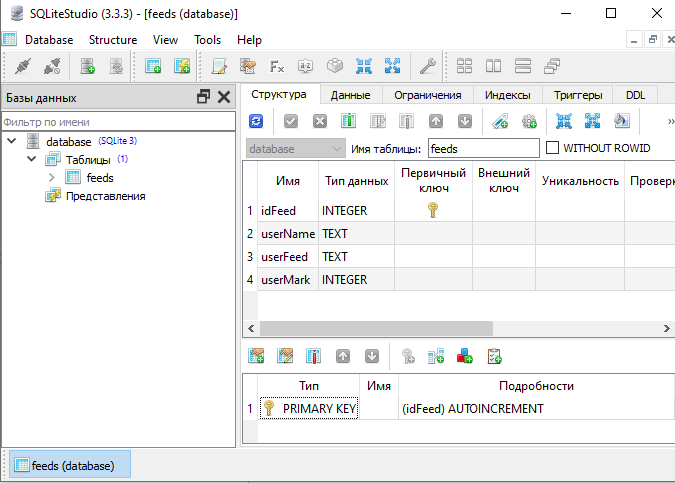


Рисунок 1. Графический интерфейс приложения SQLiteStudio.

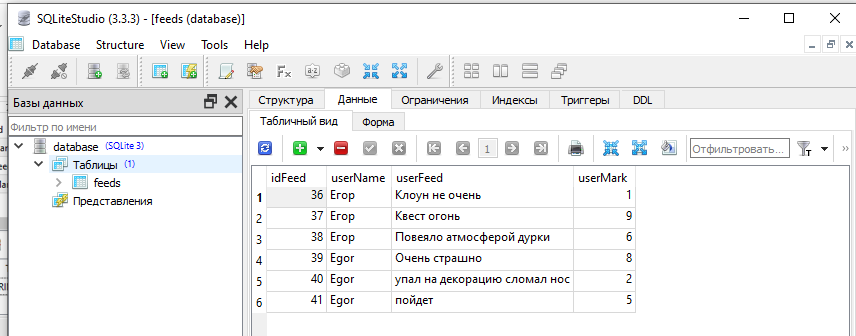


Рисунок 2. SQLiteStudio с данными.

 Рисунок 3. SQLiteStudio с данными.

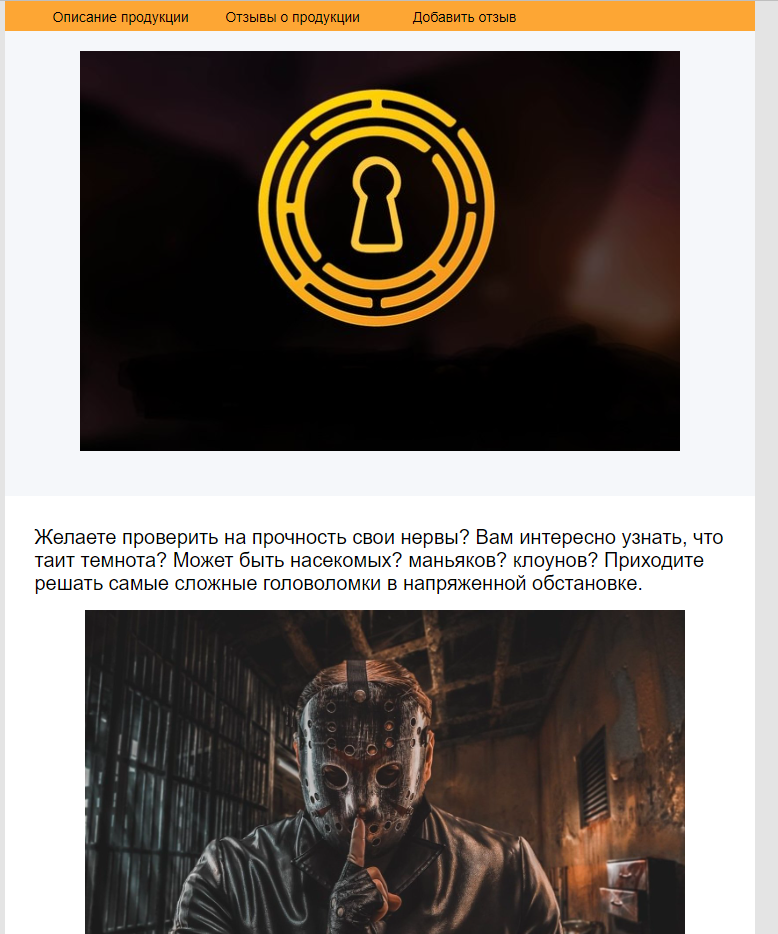


Рисунок 4. Интро.

## Заключение

В ходе практической работы были проанализированы несколько языков программирования(python, C#, node.js) и СУБД(MySQL, SQLite) для создания программного продукта. На первом этапе реализовали базу данных и программу, которая выводит из бд отзывы о квест руме. Далее создали фильтрацию отзывов в бд и внесения оценки пользователя. Автоматизация для программы не требуется , так как данные от человека получается и автоматически обрабатываются для вывода на вкладку отзывов, поэтому надобность в присутствие человека отсуцтвует.

В дальнейшем можно реализовать функцию аккаунтов, чтобы человек мог написать только один отзыв со своего аккаутна. Также можно создать вкладку с фотографиями после походов в квест. Еще можно добавить страницу с кратким описанием локаций.

## Список источников

1. SQLite Forensics (2018) Author: Paul Sanderson Publisher: Amazon
2. Беляев С.А. Разработка игр на языке JavaScript. Учебное пособие (2016)
3. Как работают веб-приложения [Электронный ресурс] Режим доступа:

## Приложение А

1. Добавить поле "Оценка продукта" (добавить отдельный, самый правый столбец таблицы)

* Добавить в таблицу БД и в шаблон html-страницы с отзывами и в обработчик события
* На шаблоне страницы по вводу отзыва добавить поле input типа number для ввода оценки и установить ограничения от 1 до 10 (10 - это высокая оценка)

1. Сделать вывод на страницу с отзывами отсортированным с обратном поряде по полю "Оценка продукта"

* Вверху таблицы отзывы с высокой оценкой, внизу - с низкой

1. Изменить на свой вкус дизайн меню, фона, кнопок, шрифтов, цветовое оформление...
2. Выбрать для себя тему работы - тот продукт (фирму, фильм, игру и т.д.), о котором будут отзывы

* На стартовой странице поменять логотип и текст описания
* Заменить отзывы, которые были в БД с примерами работы на свои (согласно выбранной теме, ~ штук 10 отзывов)

## Приложение Б

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

## Приложение B

РЕЦЕНЗИЯ

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

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

доцент кафедры ИТиПИ, к.т.н., доцент

Беляков Андрей Юрьевич Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Програмный код

