# **ОБЩИЕ СВЕДЕНИЯ**

Полное наименование системы и её обозначение:

Наименование программы: Система управления рейтингом и отзывами клиентов об автосалоне.

Шифр темы: ПМ.03 Сопровождение и обслуживание программного обеспечения компьютерных систем

Наименование предприятия, разработчика и заказчика:

Задание на разработку программы было выдано в Энгельсском технологическом институте (филиал) Саратовского Государственного Технического Университета.

Перечень документов, на основании которых создается система:

* Техническое задание
* Руководство пользователя

1. **НАЗНАЧЕНИЕ И ЦЕЛИ СОЗДАНИЯ (РАЗВИТИЯ) СИСТЕМЫ**
   1. Назначение системы

Система предназначена для сбора, обработки и анализа отзывов клиентов об автосалоне.

1. **ХАРАКТЕРИСТИКА ОБЪЕКТОВ АВТОМАТИЗАЦИИ**

Краткие сведения об объекте автоматизации или ссылка на документы, содержащие такую информацию.

Документы, на основании которых ведётся разработка:

* Техническое задание
* Руководство пользователя

1. **ТРЕБОВАНИЯ К СИСТЕМЕ**
   1. Требования к системе в целом

4.1.1. Требования к структуре и функционалу системы

Программа должна обеспечивать выполнение следующих функций:

* Разработка главной страницы
* Добавления отзыва
* Выбор категории отзыва
* Авторизация и регистрация
* Создание базы данных

4.1.2. Требования к численности и квалификации персонала

Для нормального функционирования системы требуется минимум 2 сотрудника, обладающих следующими навыками:

* Уверенное владение ПК;
* Базовые знания в области работы с удаленными базами данных (SQL).

4.1.3. Показатели назначения

Целевые показатели внедрения системы:

* + Приятное визуальное оформление;
  + Возможность добавления новых отзывов;
* Выбор категории для создания отзыва;
* Авторизация и регистрация;
* Создание базы данных под систему.

Результатом внедрения программы является выполнение всех указанных функций, обеспечивающих удобство работы пользователей и администрации.

4.1.4. Требования к надежности

Надежная работа системы обеспечивается соответствием платформы следующим минимальным требованиям:

* Процессор с тактовой частотой от 1.66 ГГц;
* Оперативная память не менее 2 ГБ;
* Видеоускоритель с поддержкой DirectX 9;
* Операционная система Windows 10;
* Разрешение экрана от 800x600;
* Наличие стандартных устройств ввода (клавиатура, мышь).

4.1.5. Требования к безопасности

* Предотвращение аппаратных сбоев;
* Защита данных от несанкционированного доступа и утечек.

4.1.6. Требования к транспортабельности

Для данной системы требования к транспортабельности отсутствуют.

4.1.7. Требования к эксплуатации, обслуживанию и хранению

Эксплуатация системы должна соответствовать требованиям СанПиН 2.2.2/2.4.1340-03:

* Организация рабочего места с учетом гигиенических норм;
* Регулярное техническое обслуживание серверного оборудования и баз данных.

4.1.8. Требования к защите от влияния внешней среды

* При перегрузке системы должно быть предусмотрено автоматическое отключение для предотвращения повреждения данных и оборудования;
* Соответствие аппаратных средств допустимым уровням влажности и температур, установленным производителем.

4.2. Требования к функциям, выполняемым системой

1. **Категории** — создание отзыва, с выбором доступных категорий.
2. **Добавление отзыва** — создание отзыва, с заполнением полей.
3. **Внешний вид главной страницы** — удобный и современный дизайн главной страницы.
4. **Авторизация и регистрация пользователя** — регистрация нового аккаунт и авторизация в данный аккаунт.
5. **Навигация** — навигация по сайту.

4.3. Требования к видам обеспечения

Таблица 1. Функции системы

|  |  |  |
| --- | --- | --- |
| **Название операции** | **Функции** | **Действие** |
| Создание отзыва с категориями | Создание новых отзывов с выбором категории | Пользователь имеет возможность выбрать категорию для отзыва. |
| Навигация на главной страницы | Возможность навигации по отзывам | Пользователь имеет возможность навигации по сайту |
| Регистрация и авторизация | Возможность создания и входа в аккаунт | Пользователь имеет возможность зарегистрировать новый или войти в уже существующий аккаунт |

**СОСТАВ И СОДЕРЖАНИЕ РАБОТ ПО СОЗДАНИЮ СИСТЕМЫ**

Перечень стадий и этапов работы по созданию системы, сроки выполнения:

Таблица 2. Этапы разработки

|  |  |  |  |
| --- | --- | --- | --- |
| **Название этапа** | **Планируемая дата исполнения** | **Фактическая дата исполнения** | **Подпись исполнителя** |
| Ознакомление с основной нормативной документацией |  |  | **Водопьянов Н.А.** |
| Сформировать отчет об анализе предметной области |  |  | **Водопьянов Н.А.** |
| Заполнение информационных таблиц |  |  | **Водопьянов Н.А.** |
| Создание форм, оформление интерфейса |  |  | **Водопьянов Н.А.** |
| Написание кода |  |  | **Водопьянов Н.А.** |
| Тестирование работоспособности |  |  | **Водопьянов Н.А.** |
| Пробный запуск |  |  | **Водопьянов Н.А.** |

**АННОТАЦИЯ**

Настоящий документ представляет собой руководство пользователя (далее Руководство) веб-приложения Motora.

Сайт позволяет пользователям оставлять отзывы, выставлять рейтинг и просматривать мнения других клиентов.

Руководство кратко описывает программу, а также определяет порядок установки на компьютер, дальнейшей эксплуатации и порядок действий при возникновении ошибок.

**УСЛОВИЯ ПРИМЕНЕНИЯ**

**Минимальные системные требования для клиентского приложения:**

* + Операционная система: Windows 10 или выше, macOS 10.15 или выше, современные дистрибутивы Linux.
  + Процессор: Двухъядерный процессор с тактовой частотой 1.8 ГГц или выше.
  + Оперативная память: 4 ГБ ОЗУ.
  + Хранилище: 500 МБ свободного места на диске для кеширования данных.
  + Экран: Разрешение не менее 1024x768.
  + Подключение к интернету: Стабильное соединение со скоростью 5 Мбит/с или выше.
  + Браузер: Современные версии Google Chrome, Mozilla Firefox, Microsoft Edge или Safari с поддержкой HTML5 и CSS3.

**Минимальные программные требования для клиентского приложения:**

* + Поддержка JavaScript и Cookies в браузере.
  + Наличие установленного браузера с актуальными обновлениями безопасности.
  + При использовании мобильных устройств – ОС Android 9.0 или выше, iOS 13.0 или выше.

**Минимальные системные требования для серверной части (рассчитано до 500 пользователей):**

* Операционная система: Windows Server 2019 или выше.
* Процессор: Четырехъядерный процессор с тактовой частотой 2.4 ГГц или выше.
* Оперативная память: 8 ГБ ОЗУ.
* Хранилище:
* 50 ГБ свободного места на диске (SSD рекомендуется) для базы данных и файлового хранилища.
* Сеть: Скорость соединения 10 Мбит/с или выше, с пропускной способностью, обеспечивающей обслуживание 500 одновременных пользователей.
* Программное обеспечение:
* Веб-сервер: Apache 2.4 или выше, либо Nginx 1.18 или выше.
* Интерпретатор: PHP 7.4 или выше с необходимыми модулями.
* СУБД: Po (в зависимости от выбранной архитектуры).
* Средства резервного копирования данных.

**КРАТКОЕ ОПИСАНИЕ ВОЗМОЖНОСТЕЙ**

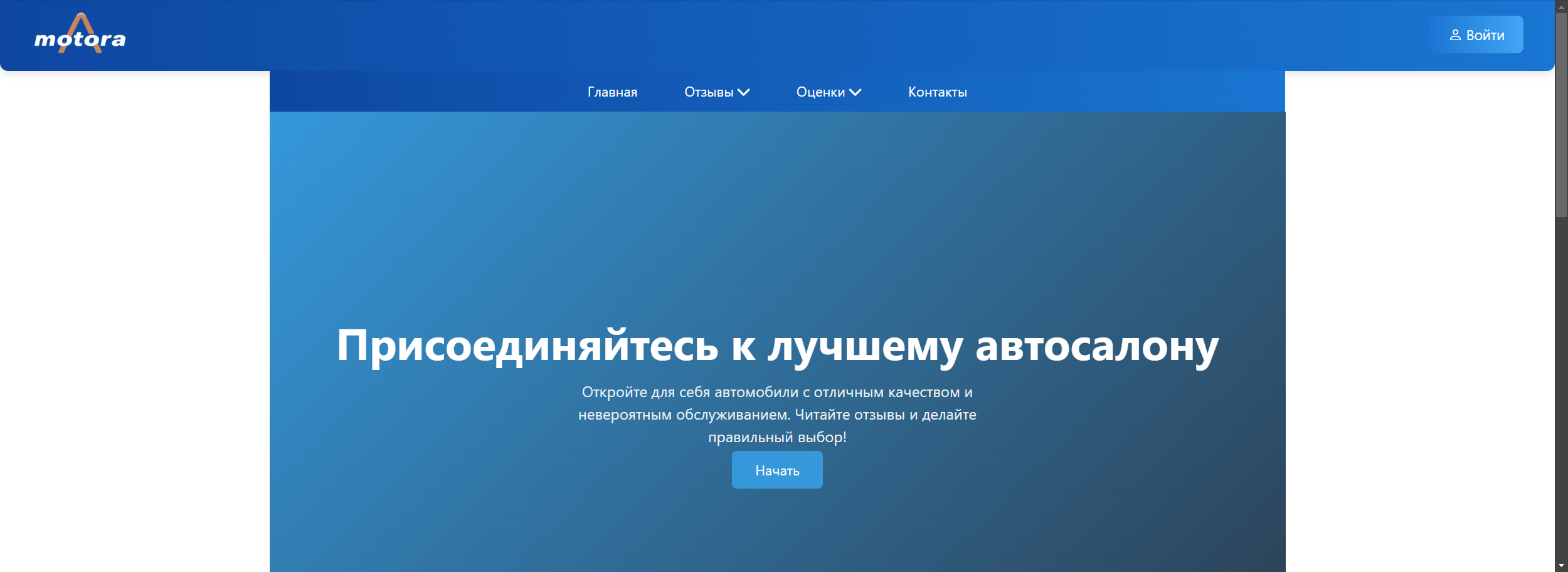
Система предназначена для сбора, обработки и анализа отзывов клиентов об автосалоне. Она предоставляет пользователям удобный интерфейс для оставления отзывов, выставления рейтингов и просмотра мнений других клиентов.

Благодаря анализу собранных данных, автосалон получает ценную обратную связь, что позволяет улучшить качество обслуживания, выявить слабые места и повысить уровень удовлетворенности клиентов.

Кроме того, наличие отзывов и рейтингов способствует повышению доверия со стороны потенциальных покупателей, что положительно сказывается на привлечении новых клиентов и росте репутации компании.

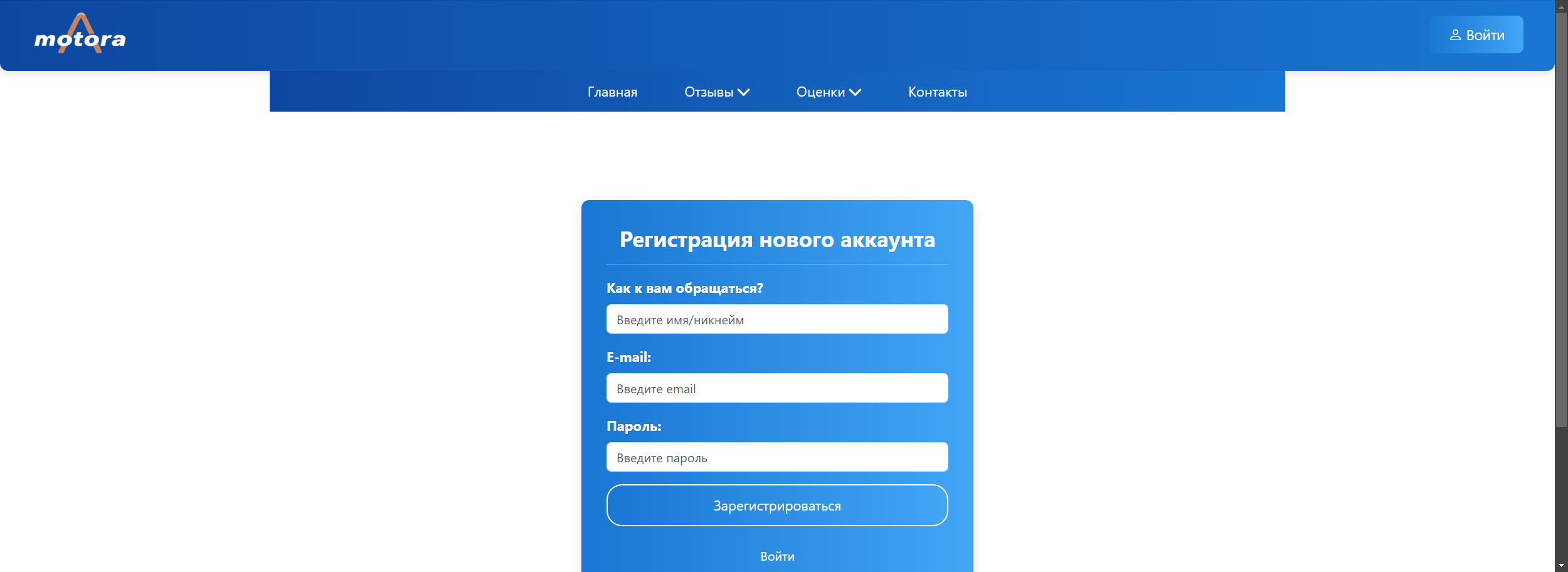
**ЭКСПЛУАТАЦИЯ**

Пользователь имеет право использовать навигацию сайта для удобного поиска информации и быстрого перехода между разделами. Навигация включает главную, отзывы, оценки и контакты.

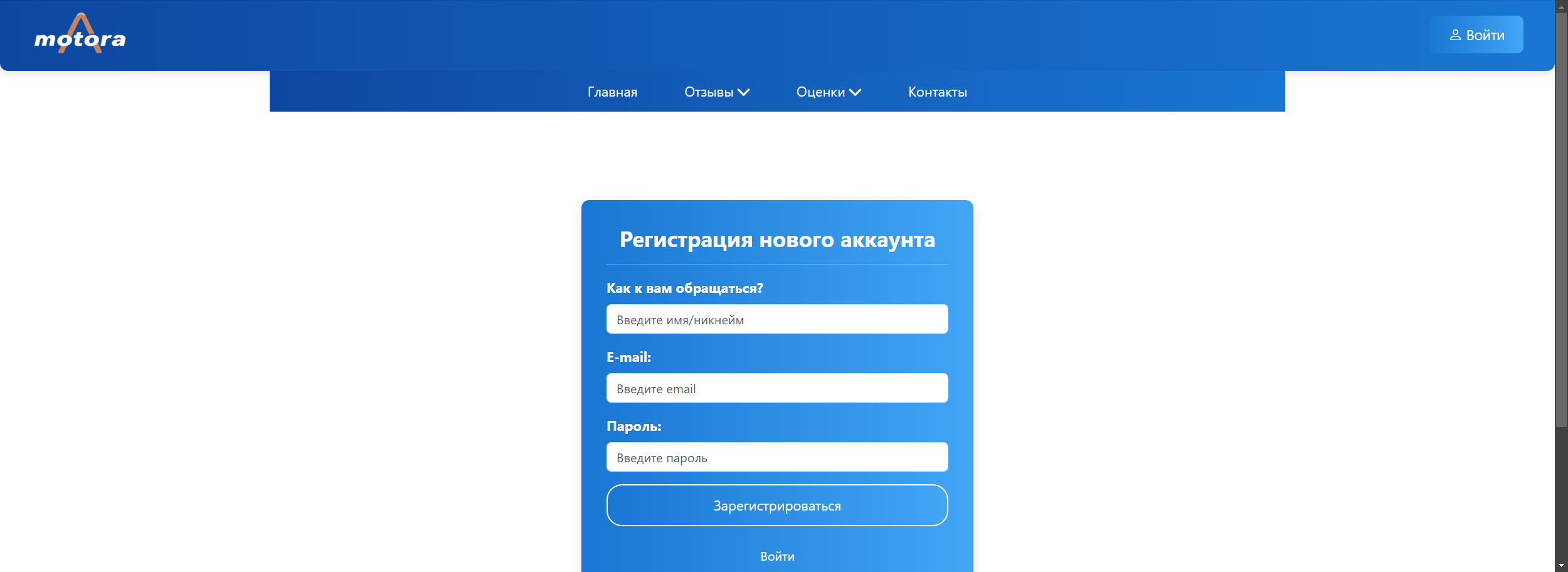


**Рисунок 1. Главная страница**

Перед тем, как оставить отзыв, пользователь должен зайти (Рисунок 2) или зарегистрировать новый аккаунт (Рисунок 3).

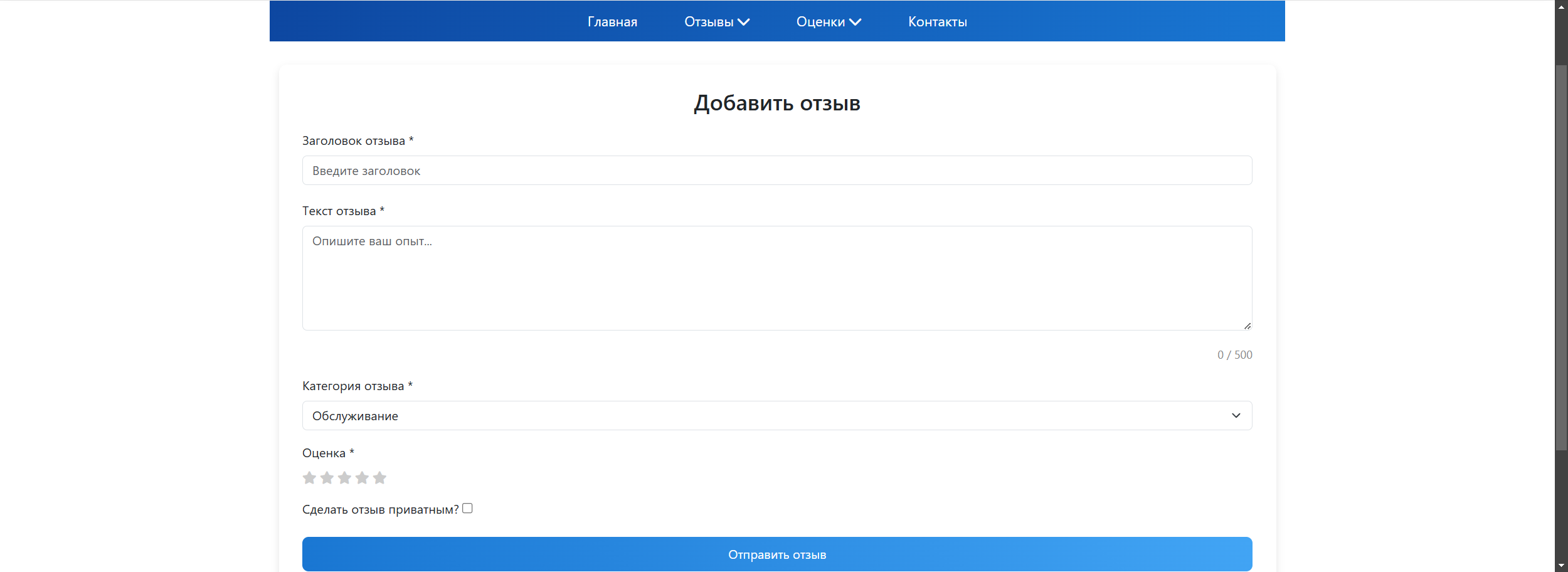


**Рисунок 2. Авторизация**



**Рисунок 3. Регистрация**

Для добавления отзыва, пользователь должен заполнить основные данные для корректной отправки (Рисунок 4). По желанию пользователь, может остаться анонимным или нет.



**Рисунок 4. Добавить отзыв**

**УСТАНОВКА ПРОГРАММЫ**

1. Нажать два раза правой кнопкой мыши по браузеру.
2. Заходим на сайт <https://motora.somee.com/>.
3. Установка завершена.

**ОШИБКИ И ПУТИ УСТРАНЕНИЯ**

Таблица 1. Ошибки и пути устранения

|  |  |  |  |
| --- | --- | --- | --- |
| Номер ошибки | Обозначение | Описание | Пути решения |
| 0001 | Не верный логин или пароль | Введены неверные логин или пароль | 1) Введите другие логин или пароль  2) Обратитесь к администратору для восстановления |
| 0002 | Не заполнены обязательные поля | Нужные поля, не заполнены. | Заполните все пустующие поля |

using Microsoft.AspNetCore.Mvc;

using Motora.Service.Discussion;

namespace Motora.Controllers

{

public class DiscussionController : Controller

{

private readonly ReviewsService \_reviewsService;

// Единый конструктор

public DiscussionController(ReviewsService reviewsService)

{

\_reviewsService = reviewsService ?? throw new ArgumentNullException(nameof(reviewsService));

}

public IActionResult Reviews()

{

return View();

}

[HttpPost]

public async Task<IActionResult> reviews(string title, string text, string category, string estimation, bool privates)

{

var user\_id = HttpContext.Session.GetInt32("ID");

var user = await \_reviewsService.ReviewsAsync(user\_id, title, text, category, estimation, privates);

if (user == null)

{

ModelState.AddModelError("", "Пользователь с таким email уже существует.");

return View();

}

return RedirectToAction("Login");

}

public IActionResult AllReviews()

{

return View();

}

public IActionResult Estimation()

{

return View();

}

public IActionResult AllEstimation()

{

return View();

}

}

}

using Supabase.Gotrue;

namespace Motora.Service.Discussion

{

public class ReviewsService

{

private readonly Supabase.Client \_client;

public ReviewsService(SupabaseClientService clientService)

{

\_client = clientService.Client;

}

public async Task<Reviews?> ReviewsAsync(int? user\_id, string title, string text, string category,string estimation, bool privates)

{

try

{

var existingUser = await \_client.From<Reviews>()

.Filter("user\_id", Supabase.Postgrest.Constants.Operator.Equals, user\_id)

.Single();

if (existingUser != null)

{

return null;

}

var reviews = new Reviews

{

user\_id = user\_id,

title = title,

text = text,

category = category,

estimation = estimation,

privates = privates

};

var response = await \_client.From<Reviews>().Insert(reviews);

return response.Models.FirstOrDefault();

}

catch (Exception ex)

{

return null;

}

}

}

}

using Supabase.Gotrue;

public class AccountService

{

private readonly Supabase.Client \_client;

public AccountService(SupabaseClientService clientService)

{

\_client = clientService.Client;

}

//Авторизация

public async Task<User?> LoginAsync(string email, string password)

{

try

{

var user = await \_client.From<User>()

.Filter("email", Supabase.Postgrest.Constants.Operator.Equals, email)

.Single();

if (user == null || user.password != password)

{

return null; // Неверный email или пароль

}

return user;

}

catch (Exception ex)

{

Console.WriteLine($"Error in LoginAsync: {ex.Message}");

return null;

}

}

// Регистрация

public async Task<User?> RegisterAsync(string email, string password, string your\_name)

{

try

{

var existingUser = await \_client.From<User>()

.Filter("email", Supabase.Postgrest.Constants.Operator.Equals, email)

.Single();

if (existingUser != null)

{

return null;

}

var newUser = new User

{

your\_name = your\_name,

email = email,

password = password,

};

var response = await \_client.From<User>().Insert(newUser);

return response.Models.FirstOrDefault();

}

catch (Exception ex)

{

return null;

}

}

}

using Microsoft.AspNetCore.Authentication.Cookies;

using Microsoft.AspNetCore.Authentication;

using Microsoft.AspNetCore.Mvc;

namespace Motora.Controllers

{

public class ProfileController : Controller

{

private readonly AccountService \_authService;

// Единый конструктор

public ProfileController(AccountService authService)

{

\_authService = authService ?? throw new ArgumentNullException(nameof(authService));

}

public IActionResult Login()

{

return View();

}

[HttpPost]

public async Task<IActionResult> login(string email, string password)

{

if (string.IsNullOrWhiteSpace(email) || string.IsNullOrWhiteSpace(password))

{

ModelState.AddModelError("", "Email и пароль обязательны.");

return View();

}

try

{

var user = await \_authService.LoginAsync(email, password);

if (user == null)

{

ModelState.AddModelError("", "Неверный email или пароль.");

return View();

}

HttpContext.Session.SetInt32("ID", user.id);

HttpContext.Session.SetString("Email", user.email);

HttpContext.Session.SetString("YourName", user.your\_name);

// Редирект на главную страницу

return RedirectToAction("Index", "Home");

}

catch (Exception ex)

{

// Логирование ошибки

ModelState.AddModelError("", "Произошла ошибка. Попробуйте позже.");

return View();

}

}

public IActionResult Register()

{

return View();

}

[HttpPost]

public async Task<IActionResult> register(string email, string password, string your\_name)

{

var user = await \_authService.RegisterAsync(email, password, your\_name);

if (user == null)

{

ModelState.AddModelError("", "Пользователь с таким email уже существует.");

return View();

}

return RedirectToAction("Login");

}

[HttpPost]

public async Task<IActionResult> Logout()

{

await HttpContext.SignOutAsync(CookieAuthenticationDefaults.AuthenticationScheme);

HttpContext.Session.Clear();

return RedirectToAction("Index", "Home");

}

}

}