**Министерство образования и науки Российской Федерации**

федеральное государственное автономное образовательное учреждение

высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа информационных технологий и робототехники

Отделение Информационных технологий

Профиль Геоинформационные системы

**ОТЧЕТ  
по индивидуальному заданию**

Разработка MVC приложения

по дисциплине Управление данными

Выполнил студент группы 8И5Б \_\_\_\_\_\_\_\_\_\_ Смирнов П.О.

\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ 2018 г.

Отчет принят:

Доцент отд. ИCТ \_\_\_\_\_\_\_\_\_\_\_ Шестаков Н.А.

\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ 2018 г.

Томск 2018 г.

1. **Цель работы**

Спроектировать приложение с использованием архитектурного паттерна MVC на ASP.NET на основе спроектированной БД из первой части ИДЗ.

1. **Постановка задачи**

Необходимо написать веб-приложение, реализующее задачи предметной области базы данных. Реализовать выдачу нетривиального запроса пользователю приложения на основе модели, не повторяющую структуру сущности. Реализовать интерфейс использования CRUD операций. Осуществить навигацию пользователя по созданным страницам, реализующим master-detail сценарий.

**Вариант задания**

Автомойка оказывает определенный перечень услуг (с прейскурантом). Клиенты должны предварительно записаться, при этом указав набор желаемых услуг. У клиентов могут быть предпочтения как по сотрудникам, так и по номеру бокса.

1. **Результат работы**

Осуществлено подключение к СУБД SQL Server, для сущностей, используемых в сценарии работы с веб-приложением, на основе модели model-first были созданы классы, описывающие атрибуты сущностей.

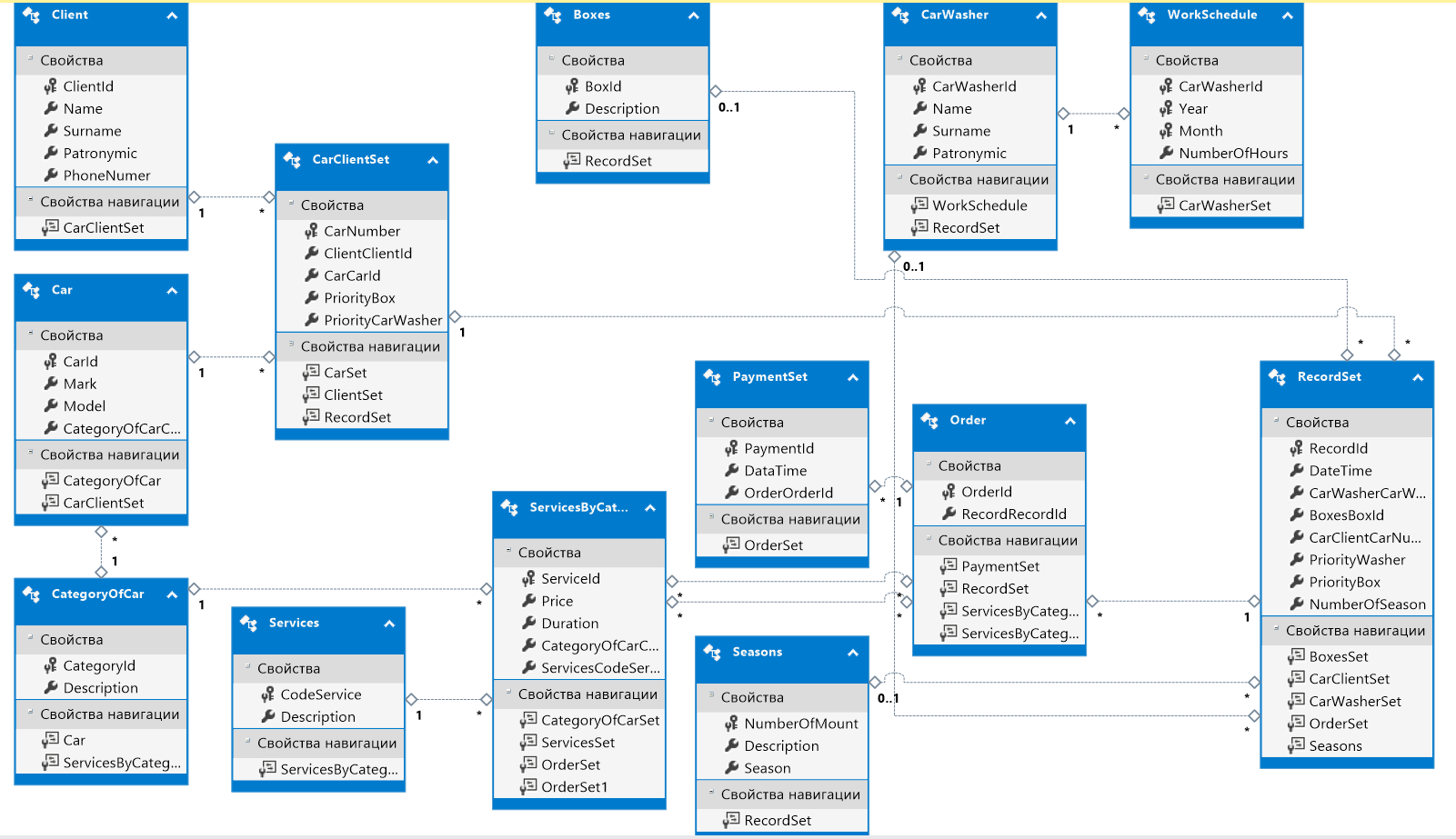


Рисунок 1 – Модель базы данных

Также для взаимодействия модели БД и созданных представлений были созданы ViewModel классы:

public class PriorityInformationViewModel

{

public List<Boxes> Boxes { get; set; }

public List<CarWasher> CarWashers { get; set; }

public CarWasher PriorityCarWasher { get; set; }

public Boxes PriotityBox { get; set; }

public Guid CurrentClient { get; set; }

}

public class RecordViewModel

{

public List<Boxes> Boxes { get; set; }

public List<CarWasher> CarWashers { get; set; }

public string PriorityCarWasher { get; set; }

public int PriotityBox { get; set; }

[Required]

[Display(Name = "Имя")]

public string Name { get; set; }

[Required]

[Display(Name = "Фамилия")]

[StringLength(20, ErrorMessage = "Значение {0} должно содержать не менее {2} символов.", MinimumLength = 6)]

public string Surname { get; set; }

[Display(Name = "Отчество")]

public string Patronymic { get; set; }

[Required]

[DataType(DataType.DateTime)]

[DisplayFormat(DataFormatString = "{0:MM/dd/yy}", ApplyFormatInEditMode = true)]

[Display(Name = "Дата записи")]

public DateTime DateRecord { get; set; }

[Required]

[DataType(DataType.DateTime)]

[DisplayFormat(DataFormatString = "{0:HH:mm}", ApplyFormatInEditMode = true)]

[Display(Name = "Время записи")]

public DateTime TimeRecord { get; set; }

}

public class AutorizedUserRecord

{

public List<Car> UserCars { get; set; }

public Car CurrentCar { get; set; }

public Guid ClientGuid { get; set; }

[Required]

[Display(Name = "Имя")]

public string Name { get; set; }

[Required]

[Display(Name = "Фамилия")]

[StringLength(20, ErrorMessage = "Значение {0} должно содержать не менее {2} символов.", MinimumLength = 6)]

public string Surname { get; set; }

[Display(Name = "Отчество")]

public string Patronymic { get; set; }

[Required]

[Display(Name = "Номер телефона")]

public string Phone { get; set; }

[Required]

[DataType(DataType.DateTime)]

[DisplayFormat(DataFormatString = "{0:MM/dd/yy}", ApplyFormatInEditMode = true)]

[Display(Name = "Дата записи")]

public DateTime DateRecord { get; set; }

[Required]

[DataType(DataType.DateTime)]

[DisplayFormat(DataFormatString = "{0:HH:mm}", ApplyFormatInEditMode = true)]

[Display(Name = "Время записи")]

public DateTime TimeRecord { get; set; }

}

public class CarViewModel

{

public List<Car> Cars { get; set; }

public Guid CurrentModel { get; set; }

public string ClientsCarNumber { get; set; }

public string CurrentMark { get; set; }

public bool HasRecord { get; set; }

}

public class RecordListViewModel

{

public Guid RecordGuid { get; set; }

public List<Services> ListOfServices { get; set; }

public Car CurrentCar { get; set; }

public Services PickedService { get; set; }

public Boxes PriorityBox { get; set; }

public CarWasher PriorityCarWasher { get; set; }

public DateTime DateTime { get; set; }

}

public class AddServiceViewModel

{

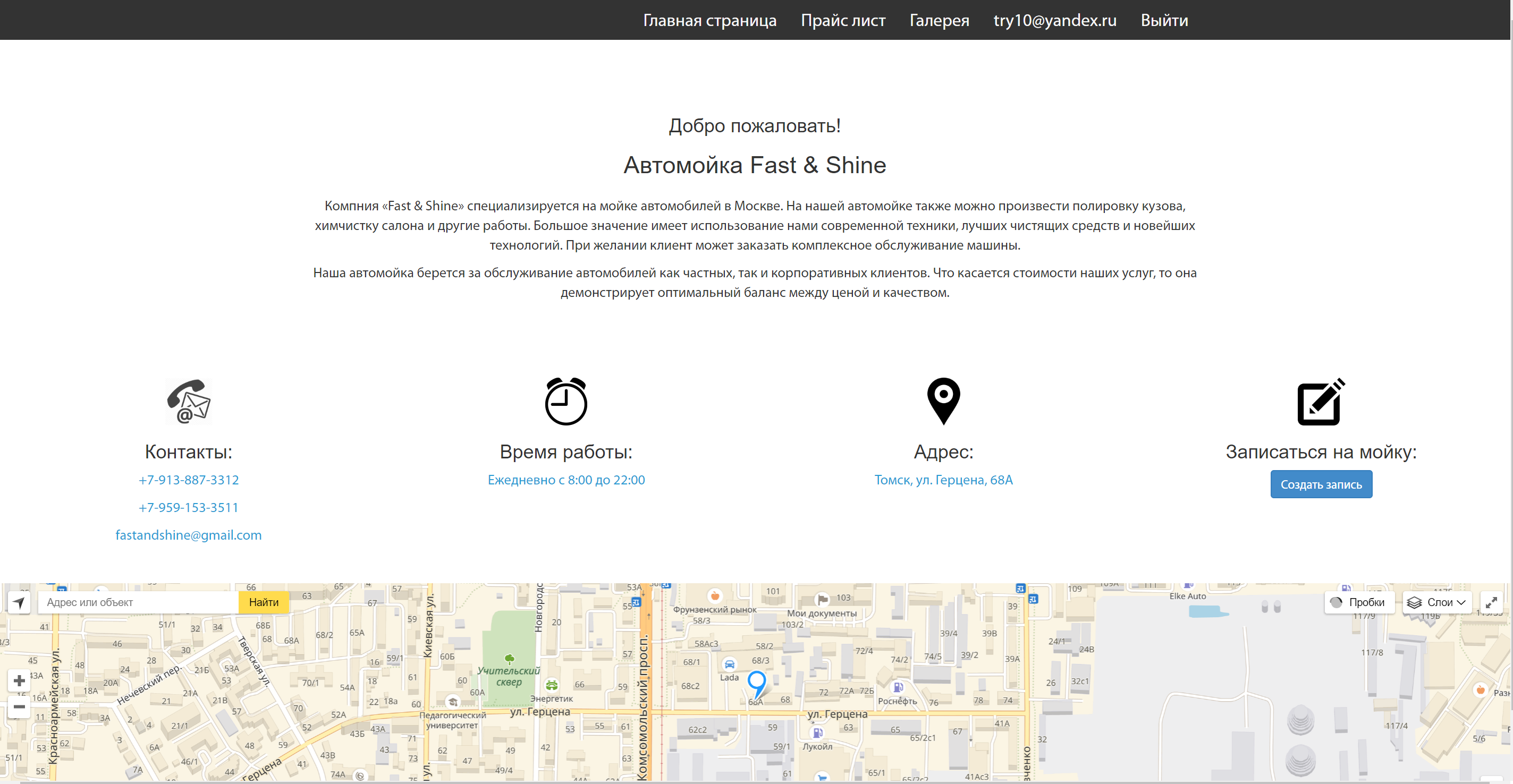
public Guid RecordGuid { get; set; }

public List<Services> listOfServices { get; set; }

public List<ServicesByCategorySet> listOfCategorySet { get; set; }

}

На рисунке 2 изображена главная страница веб-приложения.

Рисунок 2 – Главная страница

В шапке сайта расположен partial-view который привязан к шаблону и применяется ко всем страницам. В навигационной панели при нажатии на вкладке “Прайс лист” в виде таблицы отображается набор услуг и их цены в зависимости от категории. Все данные подгружаются из локальной БД.

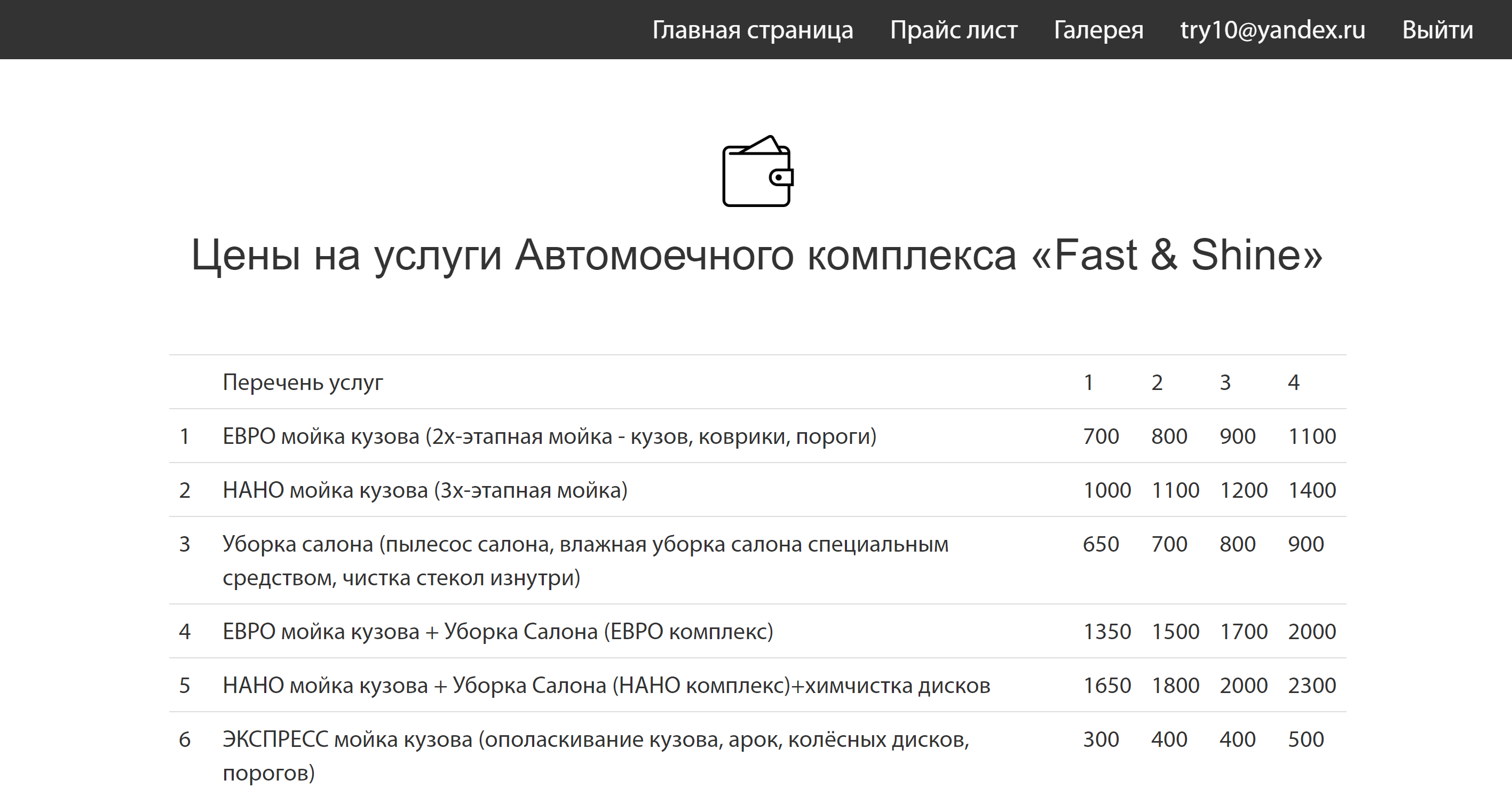


Рисунок 3 – Страница расценок на услуги

Также в веб-приложении реализована функция аутентификации пользователей, позволяющая производить регистрацию и авторизацию пользователей.

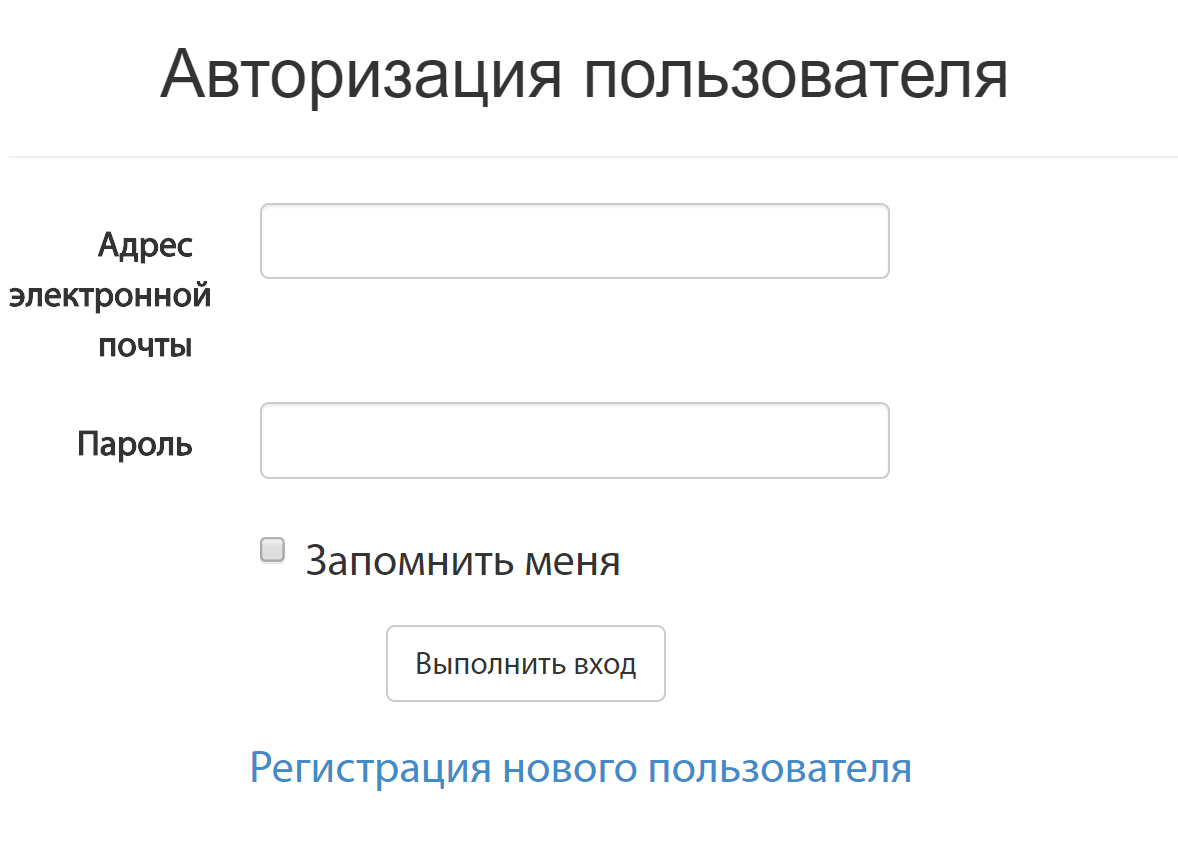


Рисунок 4 – Авторизация пользователя

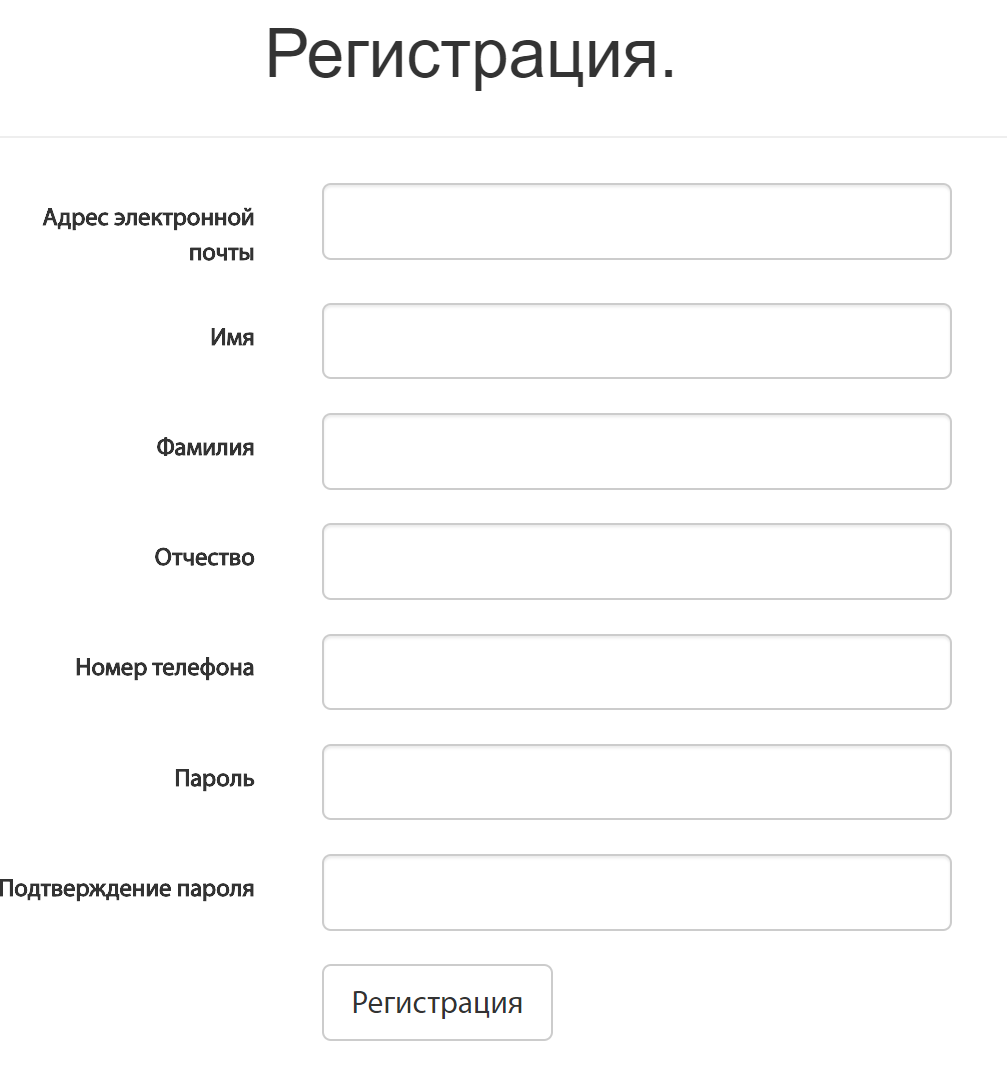


Рисунок 5 – Регистрация пользователя

При выполнении авторизации пользователь может зайти в личный кабинет, где представлена информация о пользователе. Пользователь может указать информацию о автомобилях и приоритетах.

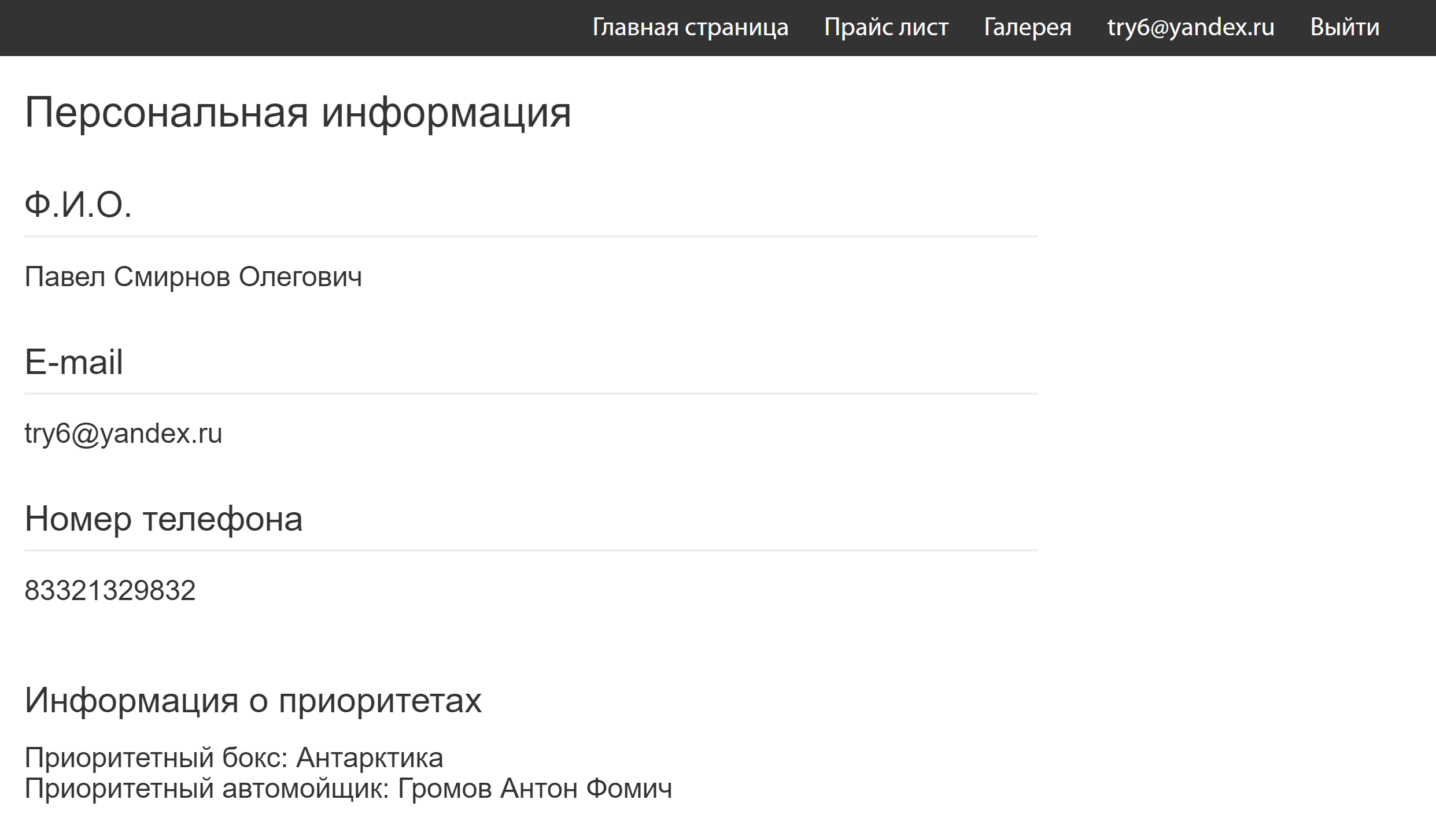


Рисунок 6 – Персональная информация

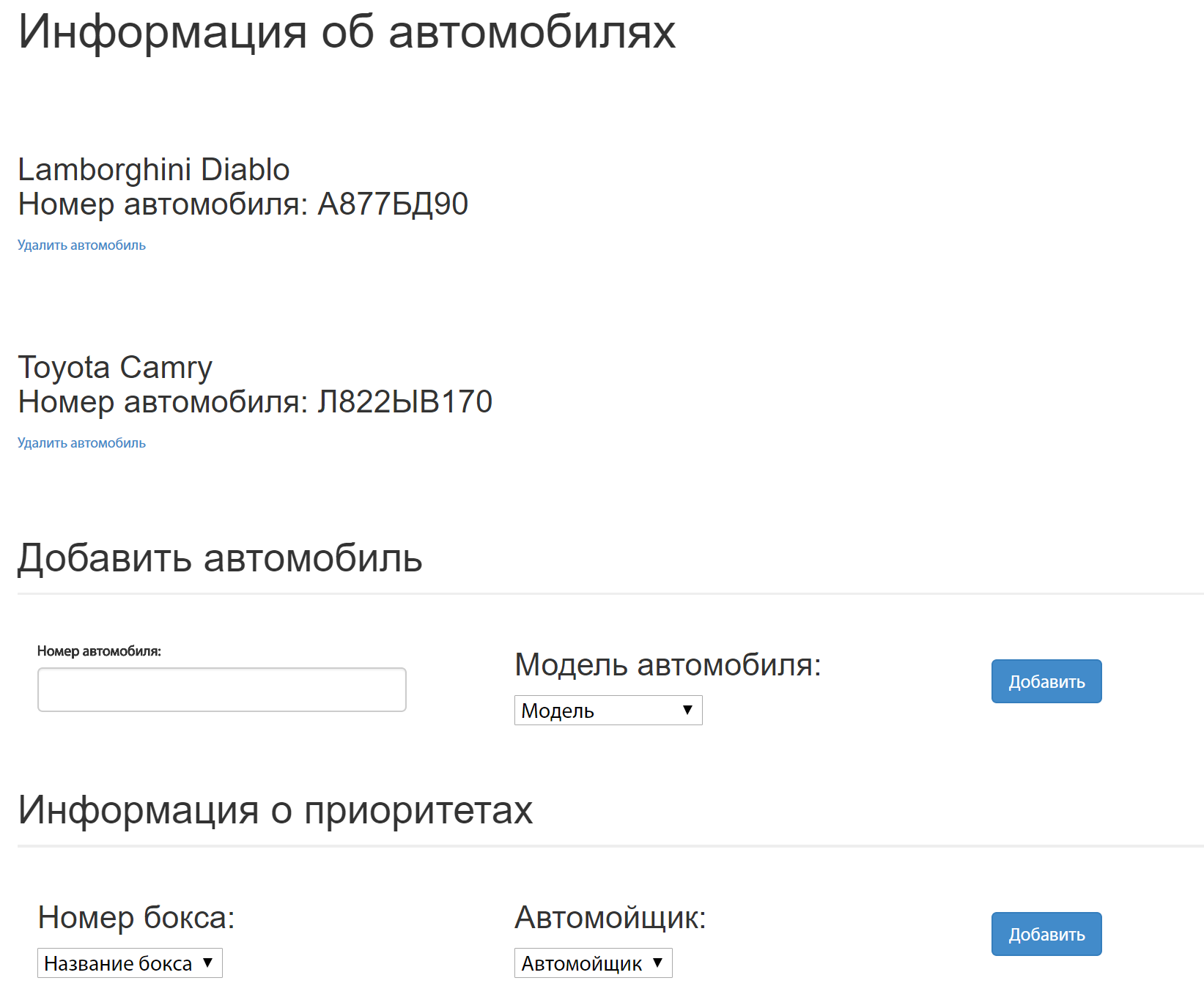


Рисунок 6 – Добавление информации

При нажатии на главной странице на вкладку “Создать запись” происходит создание записи для мойки автомобиля. Если пользователь не авторизирован, то происходит упрощенная авторизация, где пользователь указывает только персональную информацию, дату и время.

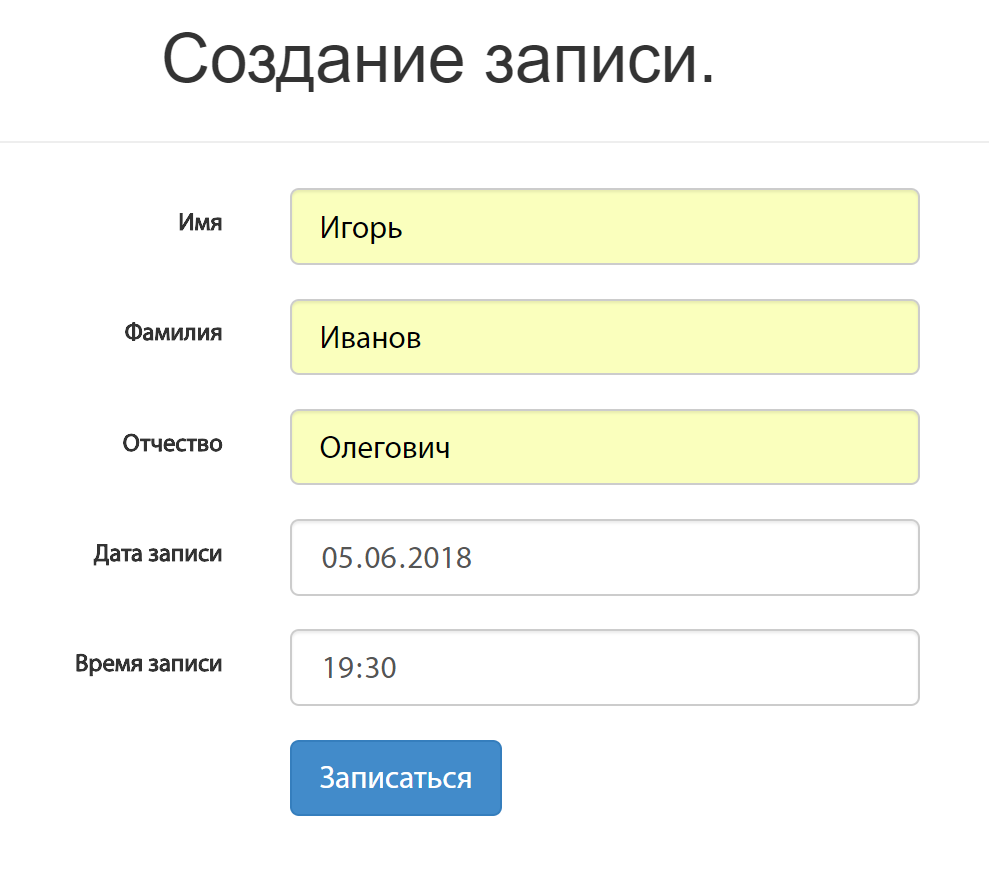


Рисунок 7 – Создание записи неавторизированным пользователем

Если пользователь авторизирован, то в форму авторизации автоматически заносится информация о пользователе, которую он внес при регистрации. В данном окне необходимо указать только дату и автомобиль для которого нужно произвести запись.

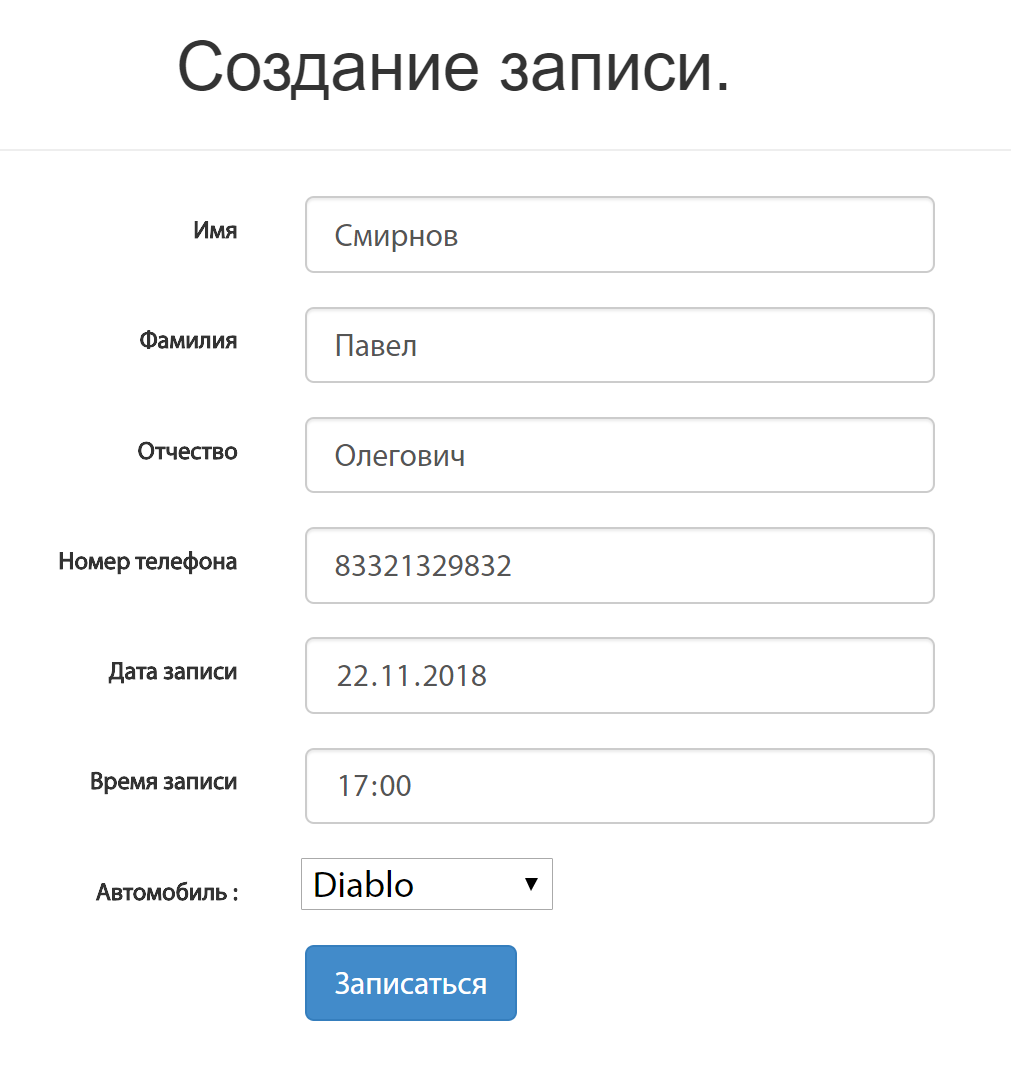


Рисунок 8 – Форма создания записи

Далее необходимо указать услуги, которые нужны пользователю.

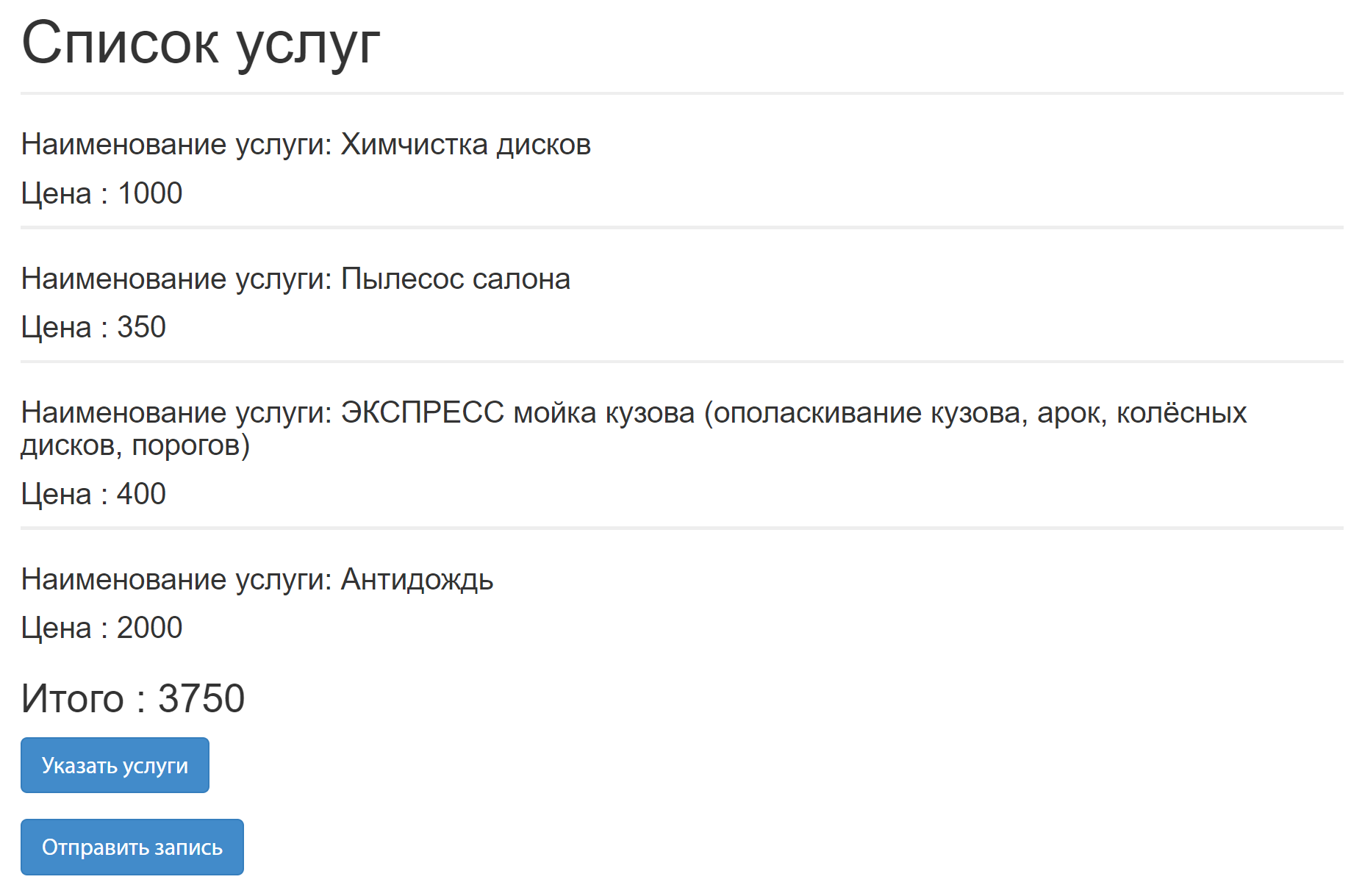


Рисунок 9 – Указание набора услуг

1. **Вывод**

В ходе создания веб-приложения были изучены основы паттерна проектирования MVC. Получены навыки работы со сложными моделями, с платформой ASP.NET.