МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Белгородский государственный технологический университет
имени В.Г. Шухова

Институт ИЭИТУС
Кафедра Информационных технологий

Курсовой проект
по дисциплине информационные технологии на тему:

«Разработка веб-приложения для управления арендой теннисных кортов «VDTENCOURT»

Выполнил студент группы ИТ- 32
Давыдов В.О.
Проверил
доц. Шаптала В.В.
ст.преп. Чернова С.Б.

Белгород 2019

Оглавление

Введение

С наступлением информационной эпохи самым актуальным и влиятельным стал, несомненно, фактор информации и ее обмена. В связи с этим большое значение придается сети Интернет. Он стал наиболее эффективным средством рекламы и продвижения и является одним из важных элементов современной цивилизации. Интернет может удовлетворить все потребности современного человека, такие как покупки, общение, заключение деловых отношений, поиск клиентов и так далее.

Присутствие в информационном поле, благодаря сети Интернет является сегодня не только критическим фактором успеха, но и необходимым условием нормальной работы любой организации. В настоящее время происходит бурное развитие информационных технологий, которые с каждым годом всё больше и больше входят в жизнь людей, делая её более комфортной. Всё больше людей начинают использовать Интернет для своих нужд. Сфера услуг не отстаёт в этом вопросе и предлагает пользователям и потребителям быстрый и удобный доступ к её ресурсам за счёт использования веб-серверов и веб-приложений.

Количество сайтов в сети Интернет растет астрономическими темпами. Очевидно, что Интернет становится катализатором экономического роста, открывая бескрайние просторы для экономической активности, освоения новых рынков и аудиторий, как для бизнеса, так и для некоммерческого сектора. То есть, справедливо, что вопросы влияния Интернет среды на хозяйственную деятельность экономических субъектов в настоящее время являются крайне актуальными. Можно утверждать, что количество природных ресурсов ограничено, но нет никаких границ для роста рынков информационно - коммуникационных услуг.

1. Постановка задачи и основных требований к разрабатываемому программному обеспечению «VDTENCOURT»

1.1. Постановка задачи

Основание для проведения разработки является задание по курсовому проектированию, которое предполагает разработку программного средства, согласно своему варианту:

«*Разработать веб-приложение по предметной области «Аренда кортов»*».

1.2. Основания для разработки

Программное средство разрабатывается на основе учебного плана кафедры «Информационные технологии» для специальности 09.03.02 «Информационные системы и технологии» по дисциплине «Информационные технологии».

1.3. Назначение программного средства «VDTENCOURT»

Программа представляет собой веб-приложение и предназначена для предоставления пользователям быстрого и комфортного доступа к ресурсам компании.

1.4. Требования к программному средству «VDTENCOURT»

1.4.1. Требования к функциональным характеристикам

Программа должна обеспечивать возможность выполнения следующих функций:

- Для пользователей:
  - поддержка современных браузеров;
  - предоставление информации о компании;
  - предоставление информации о свободном для бронирования времени;
  - бронирование времени для игры;

- выбор партнёра для игры;
- проведение онлайн оплаты;
- сохранение квитанции об оплате и о забронированном времени;

- Для администратора:
  - просмотр и редактирование информации о клиентах, бронированиях, тренерах, свободном времени;
  - сохранение таблицы базы данных о бронированиях на локальный компьютер для дальнейшего использования;

Исходные данные:

- дата аренды;
- время аренды;
- необходимость в предоставлении тренерских услуг;
- необходимость в предоставлении спортивного оборудования;
- имя пользователя;
- фамилия пользователя;
- номер карты пользователя;
- проверочный код карты пользователя;
- срок действия карты пользователя;

1.4.2. Требования к надежности

Предусмотреть контроль вводимой пользователем информации, возможные неполадки при соединении с веб-сервером. Программа должна функционировать корректно, в соответствии с разработанным алгоритмом.

1.4.3. Требования к составу и параметрам технических средств

- Для администратора

  Сервер IIS с платформой .NET Framework 4.7.2. Минимальное количество оперативной и постоянной памяти — 1024Мб. Наличие устройств ввода (клавиатура и указывающее устройство), а также устройства вывода (монитор).

- Для пользователей

  Наличие устройств ввода — клавиатуры (физической или экранной), а также указывающего устройства и устройства вывода (монитор). Приложение не адаптировано для мобильных устройств — возможны визуальные дефекты.

1.5. Требования к условиям эксплуатации

- Для администратора

  Необходима постоянная работа приложения, а также бесперебойный доступ к сети Интернет и электричеству.

- Для пользователей

  Желательно наличие современных Интернет браузеров, таких как Mozilla Firefox, Opera, Microsoft Edge, Google Chrome и его производных. При использовании других веб-браузеров возможны незначительные визуальные дефекты. Правильно выставленное время на компьютере.

1.6. Требования к программной документации

Необходимы спецификации программ, содержащие описание функций, констант, глобальных переменных, таблиц базы данных, представлений и блок-схемы.

2. Проектирование программного средства и реализация программы «VDTENCOURT»

2.1. Разработка структурной схемы базы данных

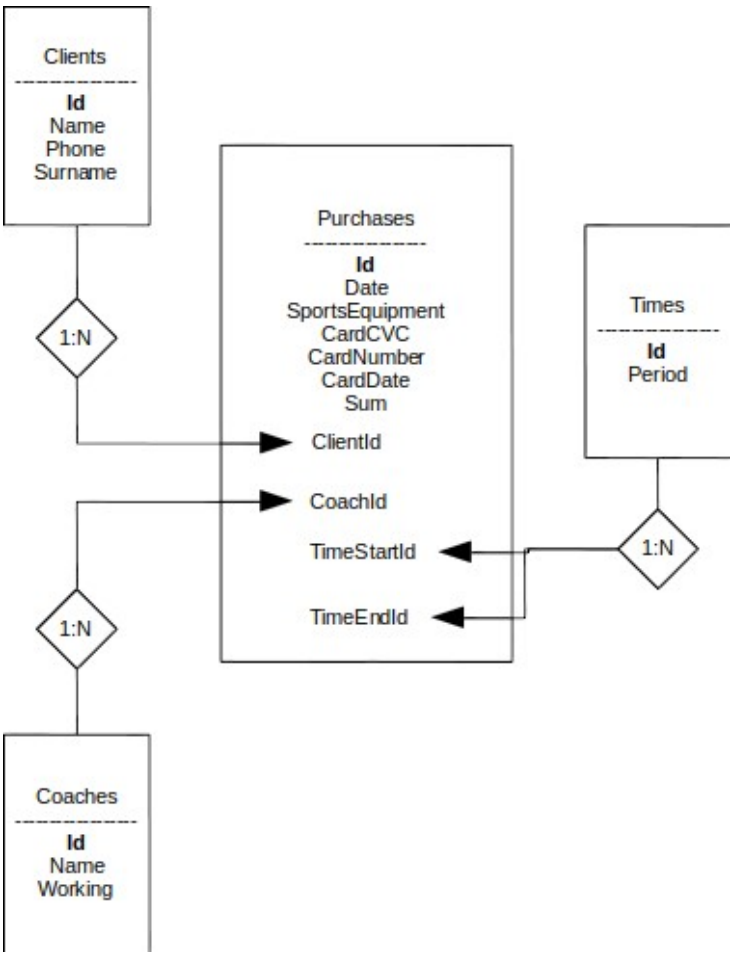

Рис. 1: Структурная схема базы данных

2.2. Разработка алгоритмов программ

2.2.1. Контроллер HomeController

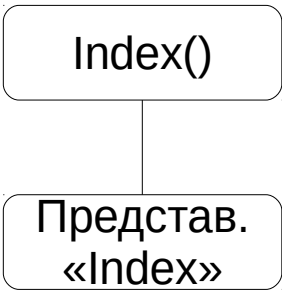

Рис. 2: Функция ознакомления с компанией

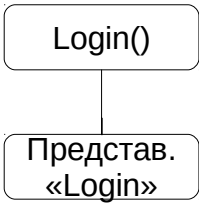

Рис. 3: Функция входа в кабинет администратора

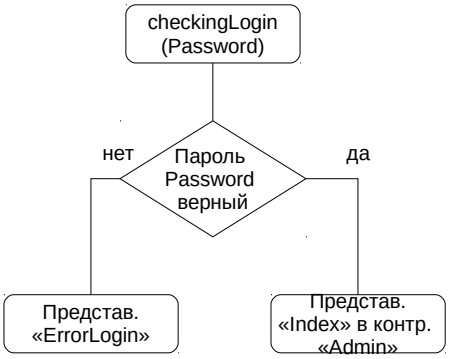

Рис. 4: Функция, проверяющая пароль для входа в кабинет администратора

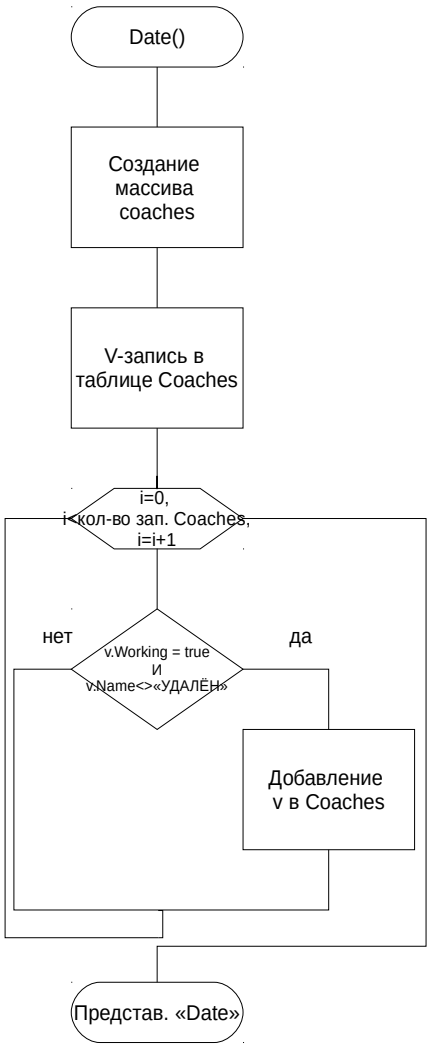

Рис. 5: Функция, предоставляющая тренеров и даты

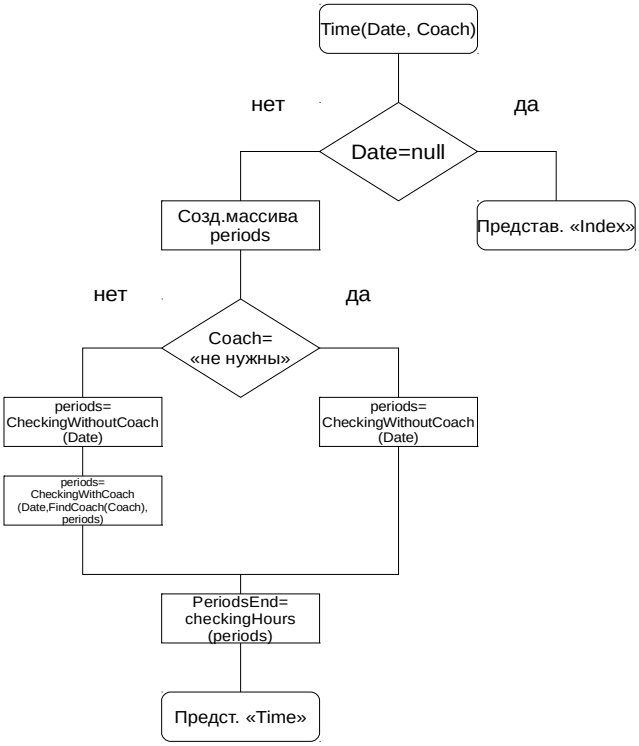

Рис. 6: Функция, предоставляющая свободное время

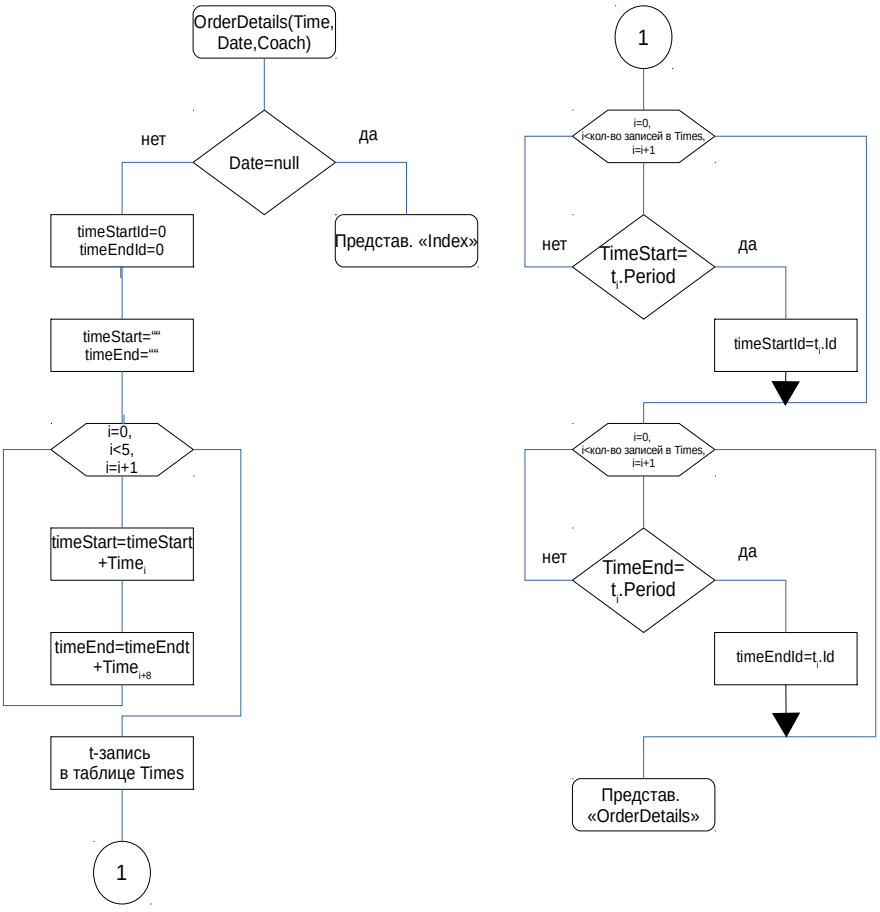

Рис. 7: Функция, предоставляющая страницу заполнения данных

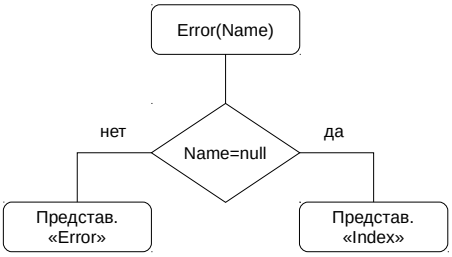

Рис. 8: Функция, сообщающая об ошибке при оформлении заказа

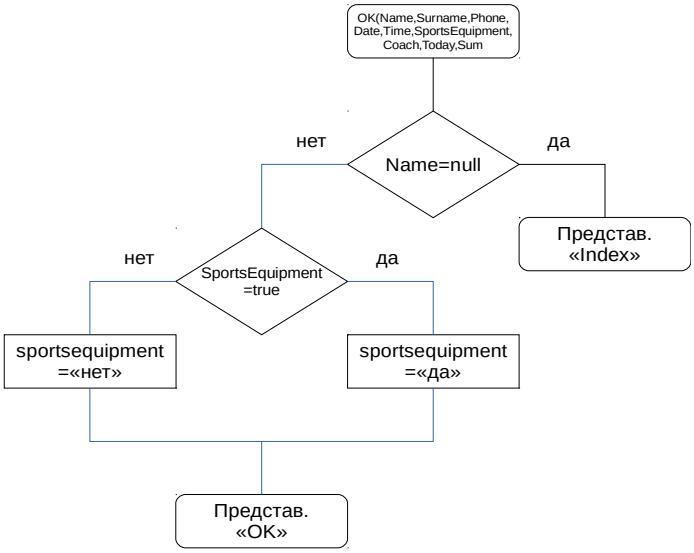

Рис. 9: Функция, подтверждающая заказ

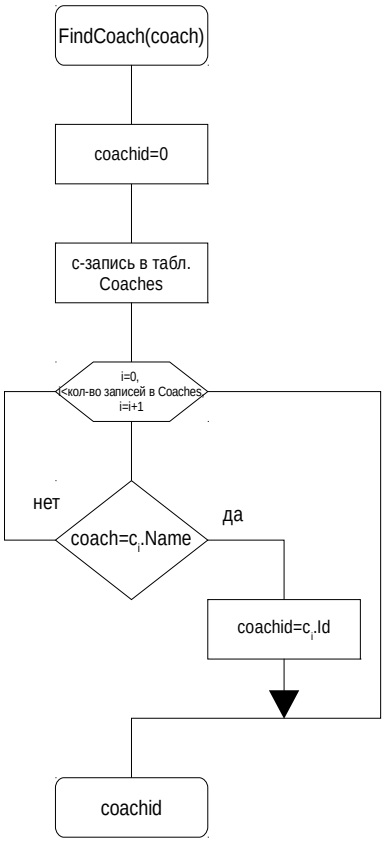

Рис. 10: Функция, ищущая тренера по ФИО

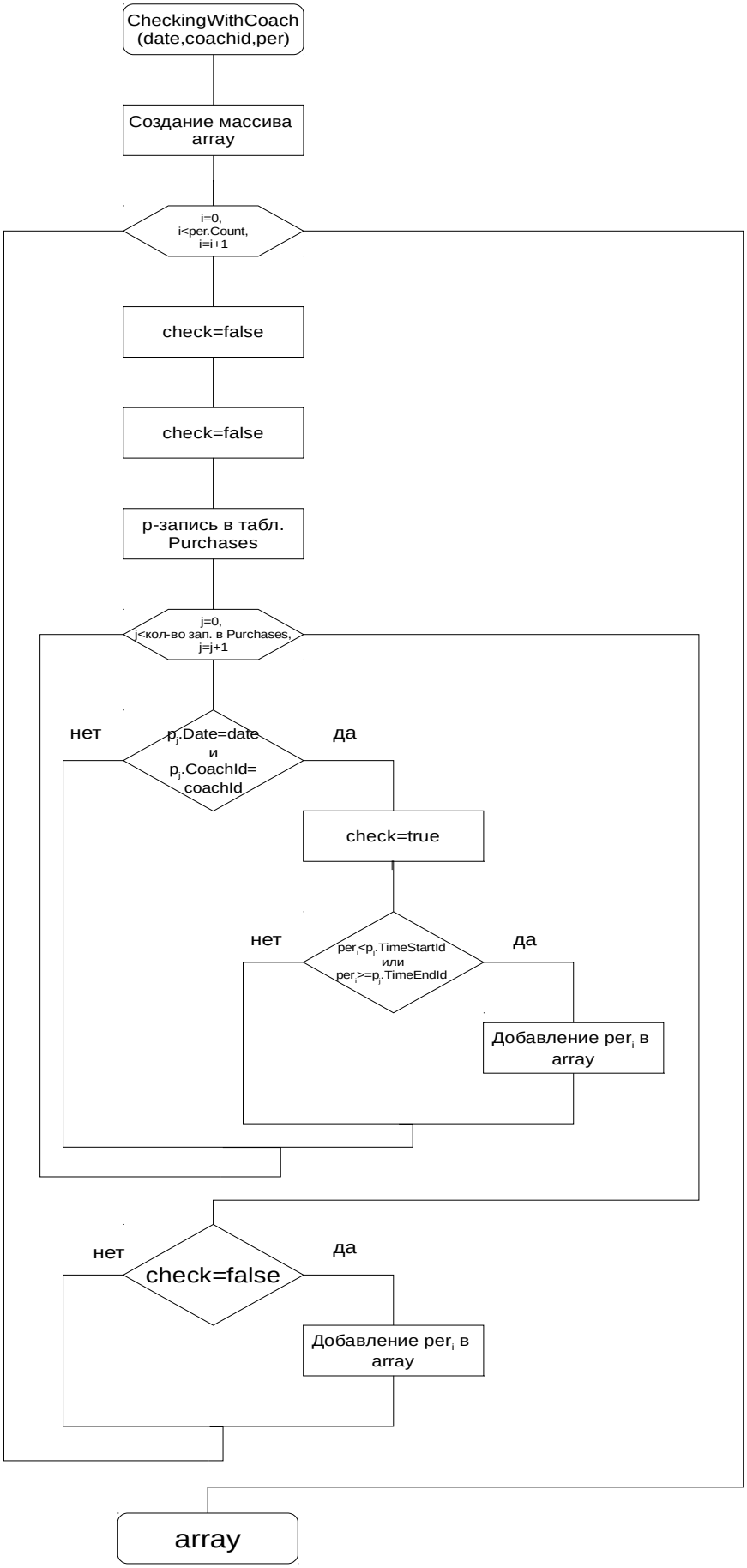

Рис. 11: Функция, проверяющая свободное время с тренерскими услугами

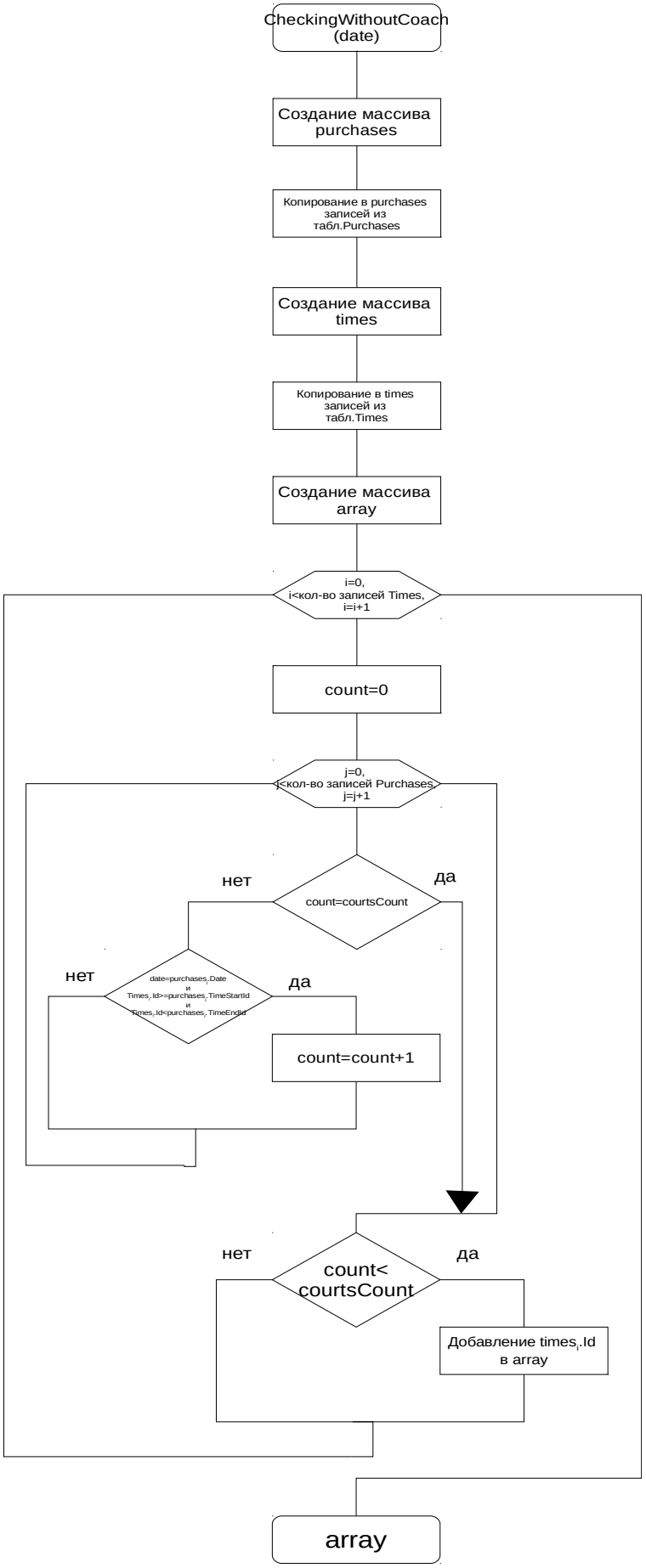

Рис. 12: Функция, проверяющая свободное время без тренерских услуг

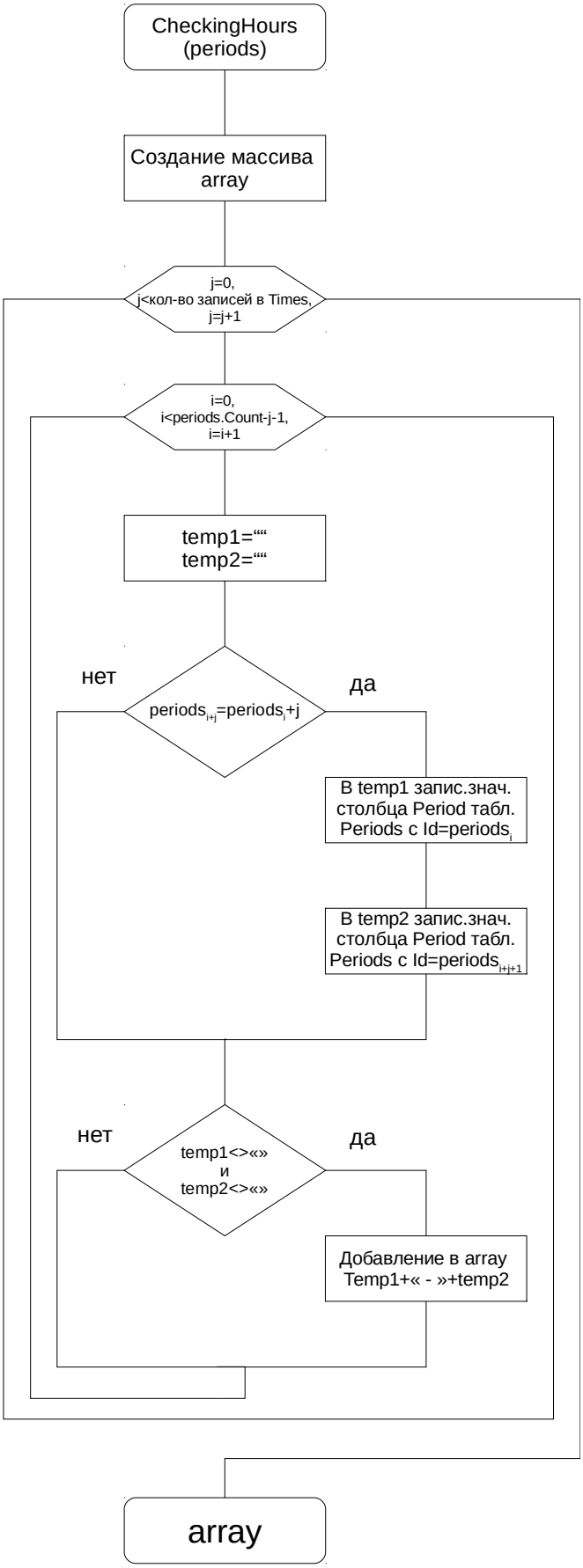

Рис. 13: Функция, проверяющая все возможные свободные промежутки времени

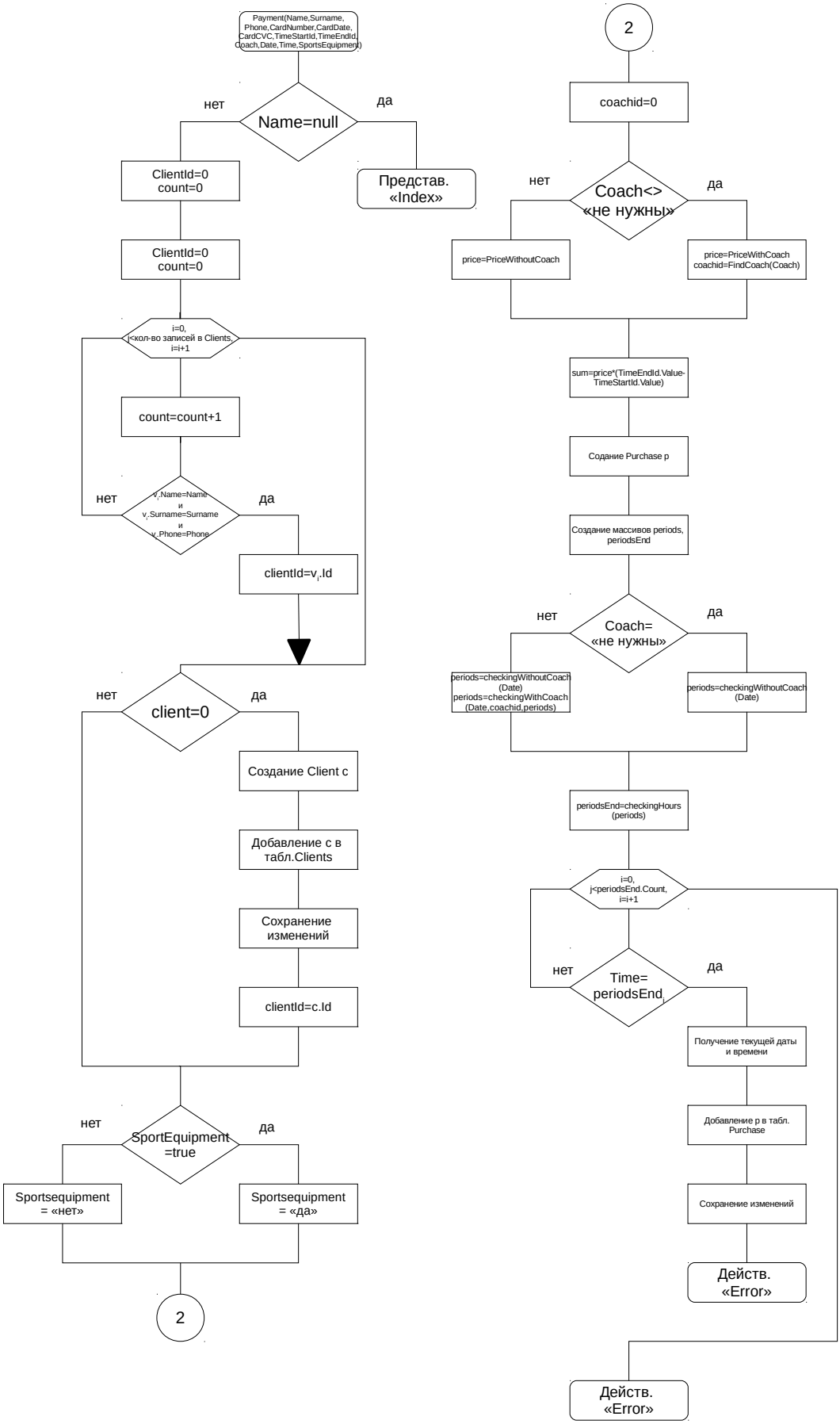

Рис. 14: Функция, подтверждающая платёж и аренду

2.2.2. Контроллер AdminController

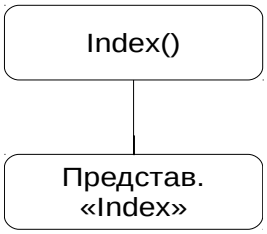

Рис. 15: Функция главной страницы администратора

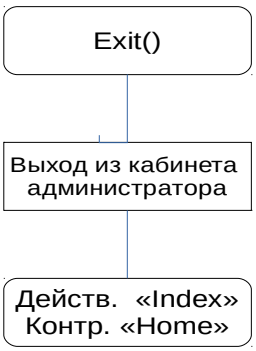

Рис. 16: Функция, осуществляющая выход из кабинета администратора

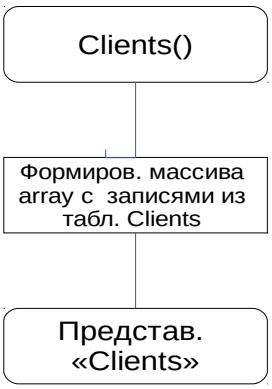

Рис. 17: Функция, формирующая массив с клиентами

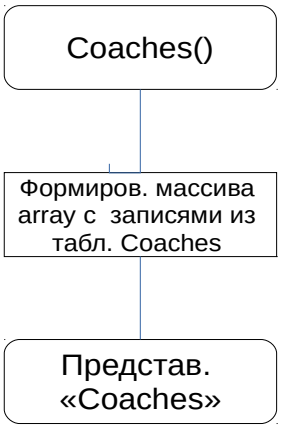

Рис. 18: Функция, формирующая массив с тренерами

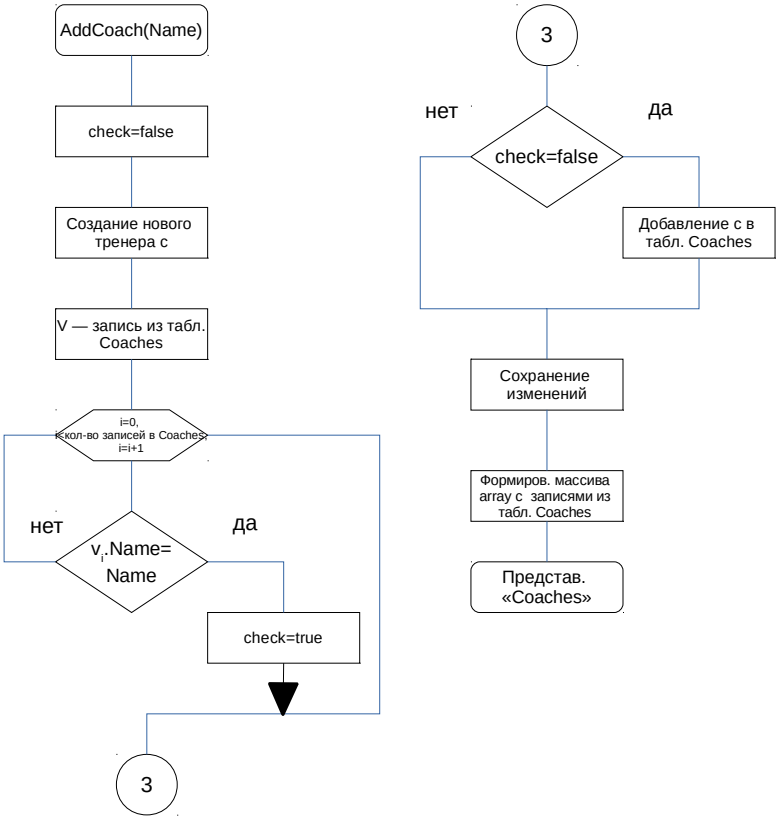

Рис. 19: Функция добавления нового тренера

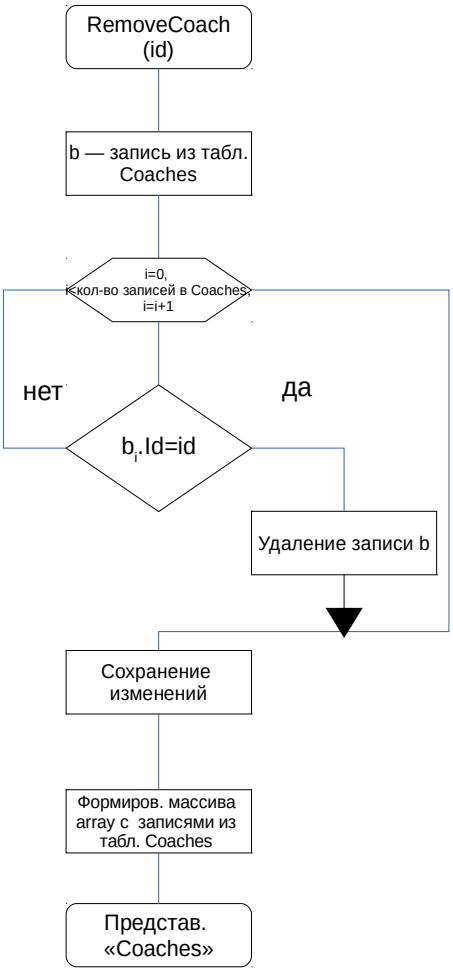

Рис. 20: Функция удаления тренера

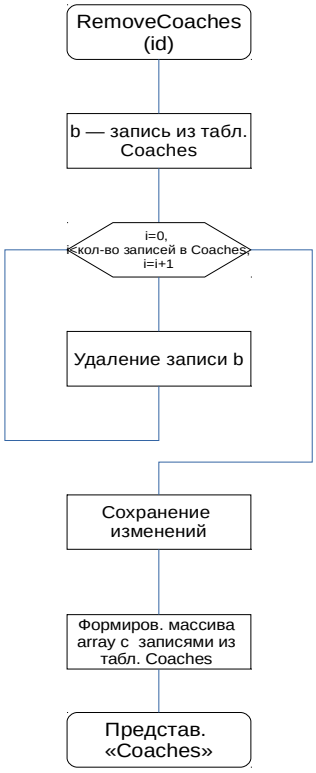

Рис. 21: Функция, удаляющая всех тренеров

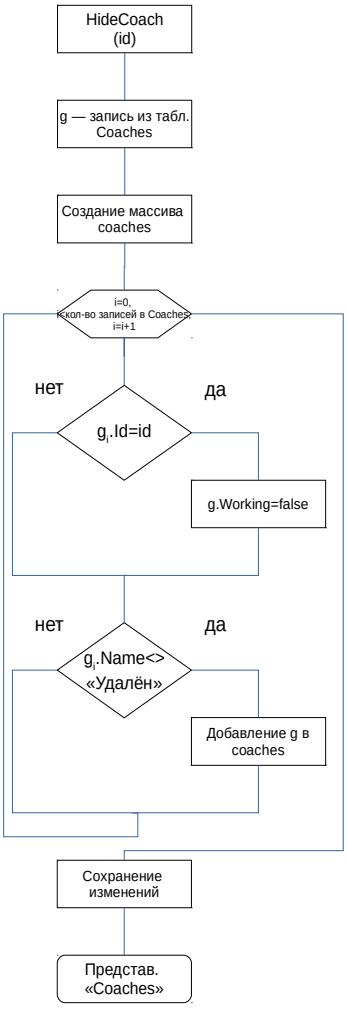

Рис. 22: Функция, отправляющая тренера в отпуск

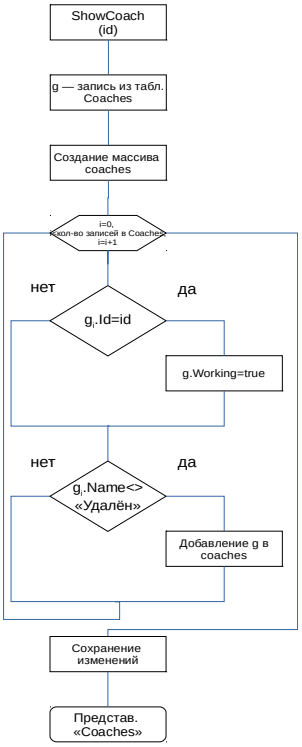

Рис. 23: Функция, возвращающая тренера на работу

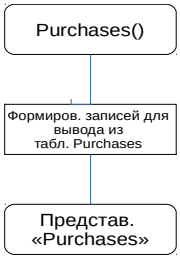

Рис. 24: Функция, предоставляющая данные об аренде

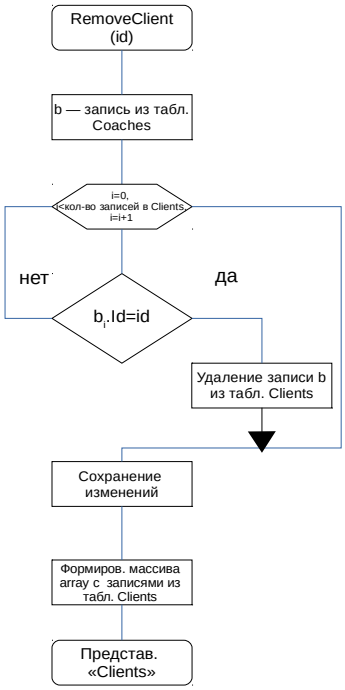

Рис. 25: Функция, удаляющая клиента

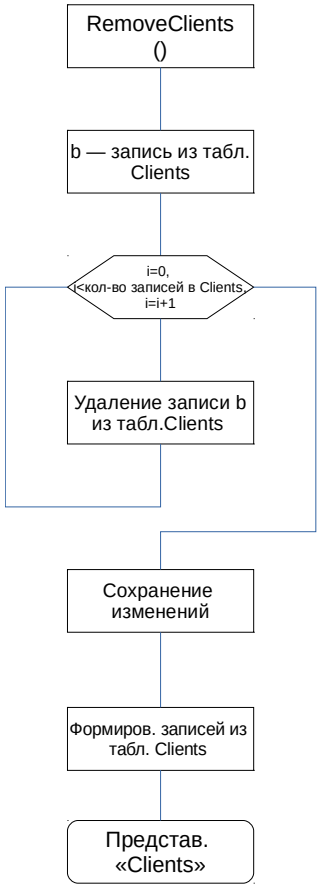

Рис. 26: Функция, удаляющая всех клиентов

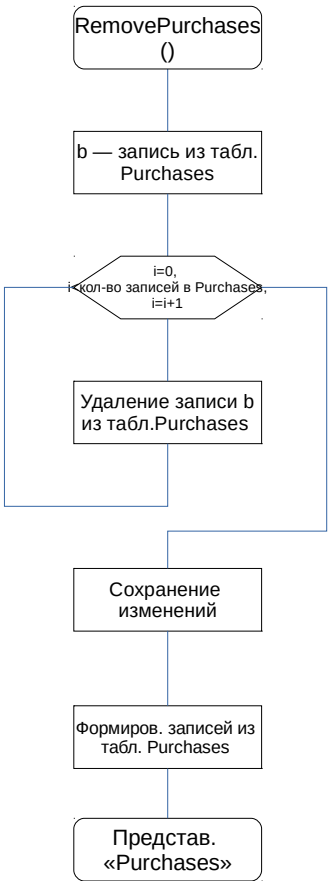

Рис. 27: Функция, удаляющая все аренды

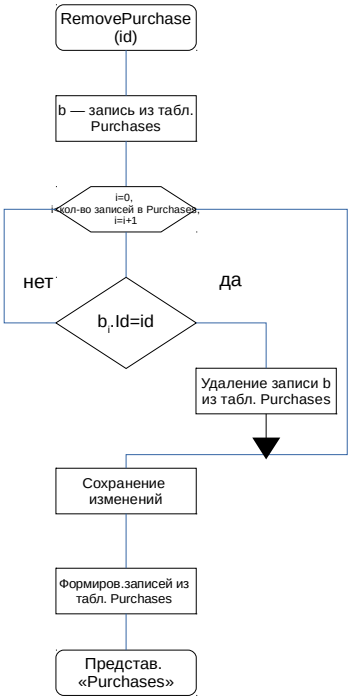

Рис. 28: Функция, удаляющая аренду

Рис. 29: Функция, полностью очищающая базу данных

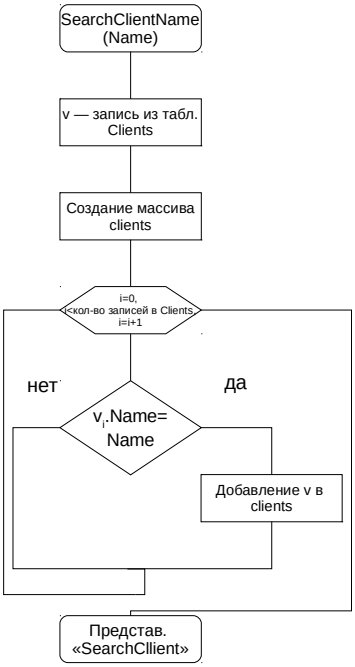

Рис. 30: Функция поиска клиента по его имени

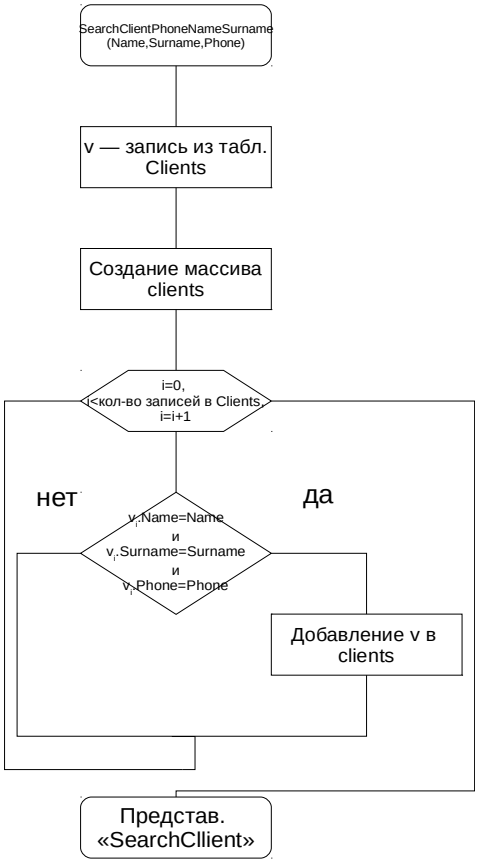

Рис. 31: Функция поиска клиента по имени, фамилии и номеру телефона

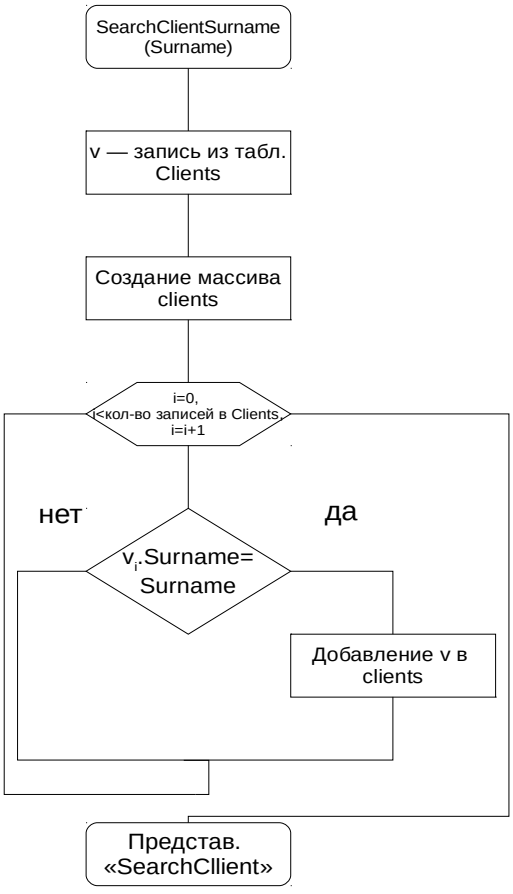

Рис. 32: Функция поиска клиента по фамилии

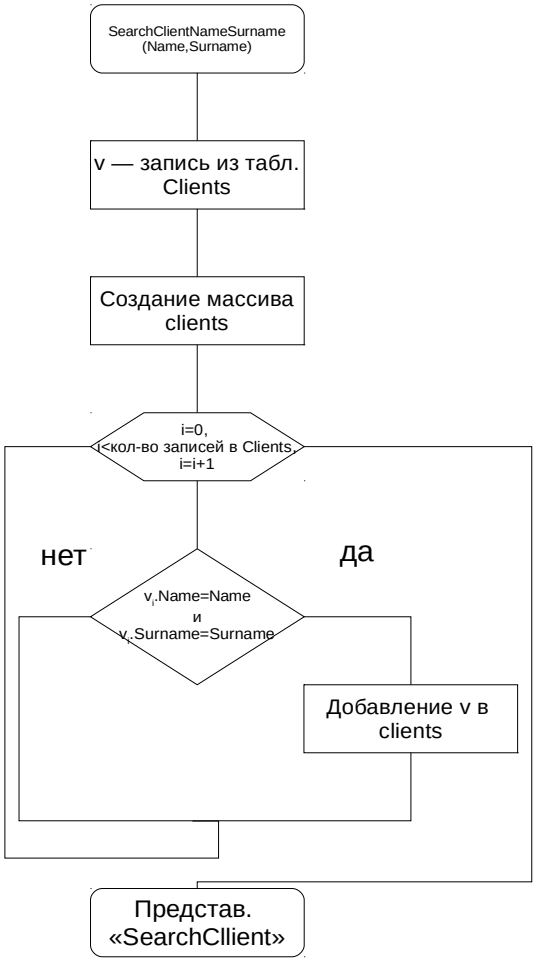

Рис. 33: Функция поиска клиента по имени и фамилии

Рис. 34: Функция поиска клиента по номеру телефона

Рис. 35: Функция поиска аренды по имени и фамилии клиента

Рис. 36: Функция поиска аренды по ФИО тренера

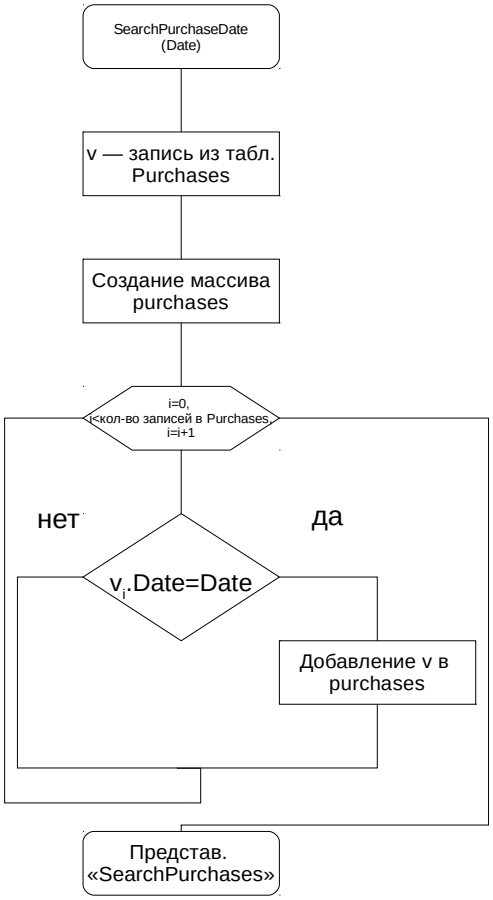

Рис. 37: Функция поиска аренды по дате

Рис. 38: Функция поиска аренды по времени

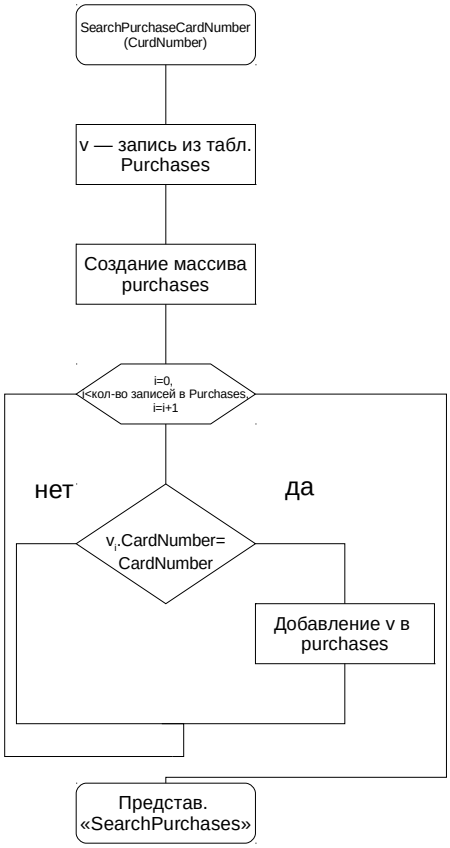

Рис. 39: Функция поиска аренды по номеру карты клиента

Рис. 40: Функция поиска аренды по времени и дате

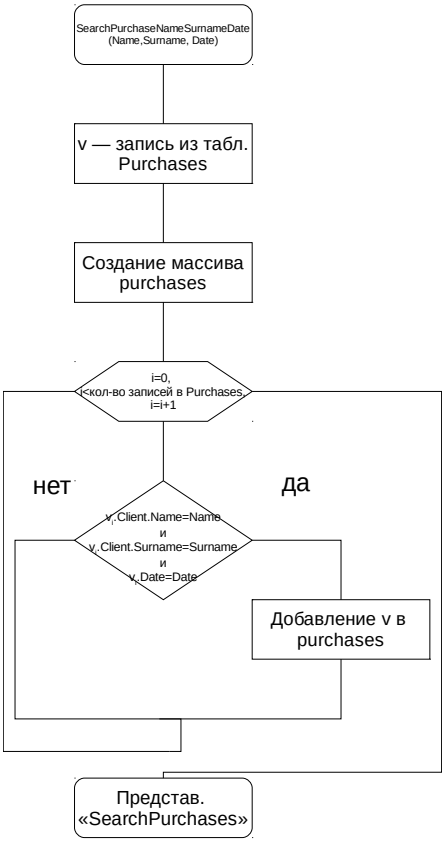

Рис. 41: Функция поиска аренды по имени, фамилии клиента и дате

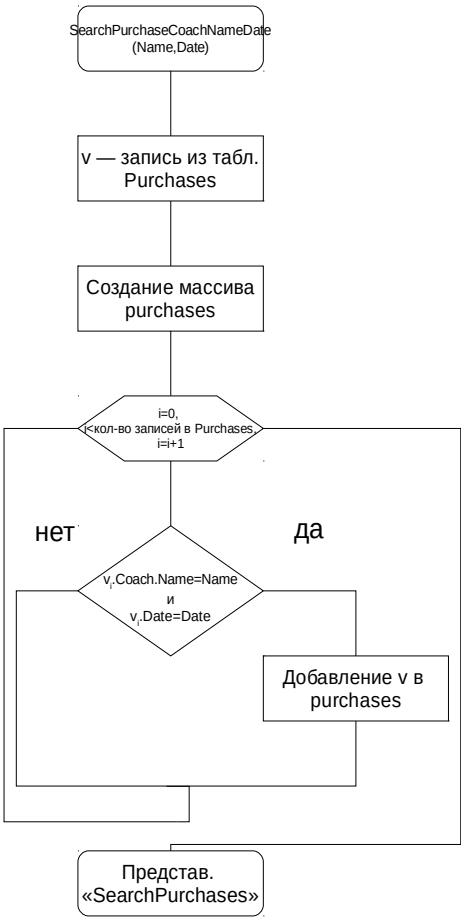

Рис. 42: Функция поиска аренды по ФИО тренера и дате

Рис. 43: Функция загрузки файла с информацией об арендах

2.3. Описание функций, констант, глобальных переменных, таблиц базы данных и представлений

2.3.1. Описание глобальных переменных

- db – объект контекста данных для взаимодействия с базой данных;

2.3.2. Описание констант

- courtsCount — количество кортов;
- PriceWithoutCoach — цена аренды без использования тренерских услуг;
- PriceWithCoach – цена аренды с использованием тренерских услуг;

2.3.3. Описание таблиц базы данных

- Clients – хранение идентификационного номера, имени, фамилии и телефонного номера клиентов;
- Coaches – хранение идентификационного номера, статуса работы, а также ФИО тренера;
- Purchases – хранение идентификационного номера аренды, идентификационного номера тренера, идентификационного номера клиента, даты и времени совершения покупки, необходиомости в спортивном оборудовании, идентификационных номеров времени начала и конца аренды, номера карты, срок действия карты, проверочного кода карты;
- Times – хранение идентификационного номера и времени (с промежутком в 1 час, начиная с 6 утра и заканчивая 9 часами вечера в 24х часовом формате);

2.3.4. Описание представлений

- Контроллер «HomeController»
  - Date – страница выбора даты аренды и тренера, если это необходимо;
  - Error – страница, появляющаяся, если другой клиент уже успел забронировать данное время;
  - ErrorLogin – страница, появляющаяся при неверном пароле от администраторского кабинета;
  - Index – главная страница, содержит информацию о компании, а также ссылку на начало процесса оформления аренды;
  - Login – страница входа в администраторскую учётную запись;
  - OK — страница, подтверждающая успешное завершение процесса арренды, а также содержащая детальную информацию об аренде;
  - OrderDetails – страница заполнения личных данных клиента для возможности совершить онлайн оплату и оформление заказа;
  - Time – страница с выбором доступного в раннее выбранный день времени для аренды;
- Контроллер «AdminController»
  - Clients – страница с информацией обо всех клиентах, хотя бы раз совершавших аренду;
  - Coaches – страница с информацией обо всех тренерах;
  - Index – главная страница администратора, содержит ссылки на необходимые действия:
  - Purchases – страница с информацией обо всех арендах;
  - SearchClient – страница отображения результатов поиска клиентов;
  - SearchPurchase – страница отображения результатов поиска аренд;

2.3.5. Описание функций

- Контроллер «HomeController»
  - Index()

Возвращает представление «Index», в котором содержится информация о компании.

- Login()

  Возвращает представление «Login», в котором пользователь может войти в администраторский кабинет при наличии пароля;

- CheckingLogin(string Password)

  Проверяет правильность ввода пароля для входа в кабинет администратора. Если пароль правильный, то предоставляется доступ в кабинет, иначе возвращается представление «ErrorLogin» с сообщением о неправильном пароле.

- Date()

  Формирует список работающих тренеров и передаёт их в представление «Date», которое и возвращает.

- Time(string Date, string Coach)

  Принимает выбранную дату и тренера. Если дата пустая, значит пользователь перешёл по этой ссылке случайно. В таком случае пользователь перенаправляется на главную страницу сайта. Если же всё корректно, то функция формирует массив со списком свободных для периодов времени, в зависимости от того, нужны ли тренерские услуги или нет. После формирования массива возвращается представление «Time», в которое передаётся выбранная дата, тренер и время аренды.

- OrderDetails(string Time, string Date, string Coach)

  Принимает выбранную дату, время и тренера. Если дата пустая, значит пользователь перешёл по этой ссылке случайно. В таком случае пользователь перенаправляется на главную страницу сайта. Если же всё корректно, то функция находит идентификационный номер времени начала и окончания аренды, а затем возвращает представление «OrderDetails», в которое передаёт выбранное время, дату, ФИО тренера и идентификационные номера начала и окончания времени аренды;

- Payment(string Name, string Surname, string Phone, string CardNumber, string CardDate, string CardCVC, int? TimeStartId, int? TimeEndId, string Coach, string Date, string Time, bool SportsEquipment = false)

  Принимает имя и фамилию клиена, его телефонный номер, проверочный код карты, срок действия карты, номер карты, ФИО тренера, идентификационные номера промежутка начала и конца времени аренды, строковое представление перида времени аренды, дату аренды и необходимость в спортивном оборудовании. Если имя клиента пустое, значит пользователь перешёл по этой ссылке случайно. В таком случае пользователь перенаправляется на главную страницу сайта. Если же всё корректно, то первым делом фунцкия наход идентификационный номер тренера. Затем осуществляется поиск клиента в базе данных. В случае успешного нахождения возвращается идентификационный номер клиента, иначе функция формирует нового клиента и добавляет его в базу данных. После этого вычисляется стоимость аренды, в зависимости от того, пользуется ли клиент тренерскими услугами или нет. В конце ещё раз осуществляется поиск на то, не занял ли кто-то другой данное время, пока пользователь вводил всю необходимую информацию. Если время ещё свободно, то функция формирует и добавляет в базу данных запись о новой аренде и возвращает представление «OK», иначе пользователь перенаправляется на страницу с ошибкой.
- Error(string Name)

  Принимает имя клиента. Если оно пустое, значит пользователь перешёл по этой ссылке случайно. В таком случае пользователь перенаправляется на главную страницу сайта. Если же всё корректно, то функция возращает представление «Error».
- OK(string Name, string Surname, string Phone, string Date, string Time, bool SportsEquipment, string Coach, string Today, string Sum)

Принимает имя клиента. Если оно пустое, значит пользователь перешёл по этой ссылке случайно. В таком случае пользователь перенаправляется на главную страницу сайта. Если же всё корректно, то функция возвращает представление «OK», в которое передаёт всю выбранную и введённую раннее пользователем информацию.

- checkingHours(ArrayList periods)

  Принимает массив с идентификационными номерами периодов времени, формирует из него и возвращает новый строковый массив, в котором комбинируются сочетания всех возможных данных периодов.

- checkingWithoutCoach(string date)

  Принимает выбранную пользователем дату аренды. Просматривает все записи таблицы «Purchases» и возвращает массив с идентификационными номерами всех доступных для аренды начал промежутков времени с учётом количества свободных кортов.

- checkingWithCoach(string date, int coachid, ArrayList per)

  Принимает выбранную пользователем дату аренды, идентификационный номер тренера и массив всех свободных периодов времени на данную дату. Просматривает все записи таблицы «Purchases» и возвращает массив с идентификационными номерами всех доступных для аренды начал промежутков времени с учётом количества свободных кортов, а также с учётом графика тренеров.

- FindCoach(string coach)

  Функция поиска тренера по его ФИО. На вход принимается ФИО тренера. Осуществляется поиск по всем записям в таблицы базы данных Coaches. При совпадении возвращается идентификационный номер найденного тренера;

- Контроллер «AdminController»
  - Index()

    Функция, возвращающая главную страницу кабинета администратора.

- Exit()

  Функция выхода из кабинета администратора. Перенаправляет пользователя на главную страницу пользователя.
- Clients()

  Функция, выводящая список всех клиентов.
- Coaches()

  Функция, выводящая список всех трненеров.
- AddCoach(string Name)

  Функция добавления тренера в базу данных. Принимает ФИО тренера. Первым делом осуществляется проверка на наличие данного тренера в базе данных. Если тренер отсутствует, то происходит добавление тренера. Возвращает представление «Coaches», которое содержит список всех тренеров.
- RemoveCoach(int id)

  Функция удаления тренера из базы данных. Принимает идентификационный номер тренера для удаления. После удаления возвращает представление «Coaches», которое содержит список всех тренеров.
- RemoveCoaches()

  Функция удаления всех тренеров из базы данных.
- HideCoach(int id)

  Функция скрытия тренера (для больничного или отпуска — при скрытии пользователь не может выбрать данного тренера для тренировки). Принимает идентификационный номер тренера для скрытия. После этого возвращает представление «Coaches», которое содержит список всех тренеров.
- ShowCoach(int id)

  Функция восстановления тренера (после больничного или отпуска — после восстановления пользователь снова может выбрать данного

тренера для тренировки). Принимает идентификационный номер тренера для восстановления. После этого возвращает представление «Coaches», которое содержит список всех тренеров.

- Purchases()

  Функция, выводящая список всех бронирований. Использует связи между таблицами базы данных для вывода полезной информации вместо идентификационных номеров.

- RemoveClient(int id)

  Функция удаления клиента из базы данных. Принимает идентификационный номер клиента для удаления. После удаления возвращает представление «Clients», которое содержит список всех клиентов.

- RemoveClients()

  Функция удаления всех клиентов из базы данных.

- RemovePurchases()

  Функция удаления всех данных о бронированиях из базы данных.

- RemovePurchase(int id)

  Функция удаления бронирования из базы данных. Принимает идентификационный номер бронирования для удаления. После удаления возвращает представление «Purchases», которое содержит список всех аренд.

- ClearAll()

  Функция очистки всей базы данных — полностью очищает информацию о клиентах, тренерах и бронированиях.

- SearchClientPhoneNameSurname(string Name, string Surname, string Phone)

  Функция, осуществляющая поиск клиента по его имени, фамилии и номеру телефона. Просматривает все записи из таблицы базы данных «Clients» и в случае совпадения добавляет данного клиента в массив.

После просмотра всех записей возвращается представление «SearchClient», в которое передаётся данный массив.

- SearchClientName(string Name)

  Функция, осуществляющая поиск клиента по его имени. Просматривает все записи из таблицы базы данных «Clients» и в случае совпадения добавляет данного клиента в массив. После просмотра всех записей возвращается представление «SearchClient», в которое передаётся данный массив.

- SearchClientSurname(string Surname)

  Функция, осуществляющая поиск клиента по его фамилии. Просматривает все записи из таблицы базы данных «Clients» и в случае совпадения добавляет данного клиента в массив. После просмотра всех записей возвращается представление «SearchClient», в которое передаётся данный массив.

- SearchClientNameSurname(string Name, string Surname)

  Функция, осуществляющая поиск клиента по его имени и фамилии. Просматривает все записи из таблицы базы данных «Clients» и в случае совпадения добавляет данного клиента в массив. После просмотра всех записей возвращается представление «SearchClient», в которое передаётся данный массив.

- SearchClientPhone(string Phone)

  Функция, осуществляющая поиск клиента по его номеру телефона. Просматривает все записи из таблицы базы данных «Clients» и в случае совпадения добавляет данного клиента в массив. После просмотра всех записей возвращается представление «SearchClient», в которое передаётся данный массив.

- SearchPurchaseNameSurname(string Name, string Surname)

  Функция, осуществляющая поиск бронирования по имени и фамилии клиента. Просматривает все записи из таблицы базы данных «Purchases»

и в случае совпадения добавляет данное бронирование в массив. После просмотра всех записей возвращается представление «SearchPurchase», в которое передаётся данный массив.

- SearchPurchaseCoachName(string Name)

  Функция, осуществляющая поиск бронирования по ФИО тренера. Просматривает все записи из таблицы базы данных «Purchases» и в случае совпадения добавляет данное бронирование в массив. После просмотра всех записей возвращается представление «SearchPurchase», в которое передаётся данный массив.
- SearchPurchaseDate(string Date)

  Функция, осуществляющая поиск бронирования по дате аренды. Просматривает все записи из таблицы базы данных «Purchases» и в случае совпадения добавляет данное бронирование в массив. После просмотра всех записей возвращается представление «SearchPurchase», в которое передаётся данный массив.
- SearchPurchaseTime(string Time)

  Функция, осуществляющая поиск бронирования по времени аренды. Просматривает все записи из таблицы базы данных «Purchases» и в случае совпадения добавляет данное бронирование в массив. После просмотра всех записей возвращается представление «SearchPurchase», в которое передаётся данный массив.
- SearchPurchaseDateTime(string Date, string Time)

  Функция, осуществляющая поиск бронирования по дате и времени бронирования. Просматривает все записи из таблицы базы данных «Purchases» и в случае совпадения добавляет данное бронирование в массив. После просмотра всех записей возвращается представление «SearchPurchase», в которое передаётся данный массив.
- SearchPurchaseCardNumber(string CardNUmber)

Функция, осуществляющая поиск бронирования по номеру карты клиента. Просматривает все записи из таблицы базы данных «Purchases» и в случае совпадения добавляет данное бронирование в массив. После просмотра всех записей возвращается представление «SearchPurchase», в которое передаётся данный массив.

- SearchPurchaseNameSurnameDate(string Name, string Surname, string Date)

  Функция, осуществляющая поиск бронирования по имени и фамилии клиента, а также по дате бронирования. Просматривает все записи из таблицы базы данных «Purchases» и в случае совпадения добавляет данное бронирование в массив. После просмотра всех записей возвращается представление «SearchPurchase», в которое передаётся данный массив.

- SearchPurchaseCoachNameDate(string Name, string Date)

  Функция, осуществляющая поиск бронирования по ФИО тренера и дате аренды Просматривает все записи из таблицы базы данных «Purchases» и в случае совпадения добавляет данное бронирование в массив. После просмотра всех записей возвращается представление «SearchPurchase», в которое передаётся данный массив.

- Download()

  Функция, формирующая файл электронной таблицы, в который копируются записи из таблицы базы данных Purchase. После формирования файла функция предлагает выбрать место для сохранения данного файла. После подтверждения пользователем файл начинает автоматически скачиваться.

Заключение

В ходе выполнения данной курсовой работы было написано веб-приложение с использованием платформы ASP.NET MVC.

Концепция MVC (model - view - controller) предполагает разделение приложения на три компонента:

- Контроллер — представляет класс, обеспечивающий связь между пользователем и системой, представлением и хранилищем данных. Он получает вводимые пользователем данные и обрабатывает их и в зависимости от результатов обработки отправляет пользователю определенный вывод, например, в виде представления.
- Представление — это собственно визуальная часть или пользовательский интерфейс приложения. Как правило, html-страница, которую пользователь видит, зайдя на сайт.
- Модель — представляет класс, описывающий логику используемых данных.

Благодаря этому реализуется концепция разделение ответственности, в связи с чем легче построить работу над отдельными компонентами. Кроме того, вследствие этого приложение обладает лучшей тестируемостью.

Разработанное средство в рамках данной курсовой работы получилось очень многофункциональным, а также с дружественным пользовательским интерфейсом, что положительно сказывается на аудитории потребителей. Также в программе производится контроль вводимой пользователем информации, исключены тупиковые ситуации. Можно отметить, что в ходе выполнения курсовой работы был более глубоко изучен язык программирования «C#», а также получено представление о том, как работают веб-приложения. Они работают на сервере и исполняются в рамках веб-сервера. Взаимодействие клиента и сервера осуществляется в рамках схемы "запрос-ответ". Вся логика веб-приложения размещается на сервере, что позволяет легче их разворачивать и обновлять по сравнению с настольными приложениями.

Список использованной литературы

1. ASP.NET MVC Pattern [Электронный ресурс]. URL: https://dotnet.microsoft.com/apps/aspnet/mvc (дата обращения: 13.07.2019).
2. Руководство по ASP.NET MVC 5 [Электронный ресурс]. URL: https://metanit.com/sharp/mvc5/ (дата обращения: 13.07.2019).

Приложение 1 Листинг программы

- Представление «Index.cshtml»

```
@{
    Layout = null;
}

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title> Главная </title>
    <link rel="icon" href="../../Content/favicon.ico" type="image/x-icon">
    <link href="~/Content/index.css" rel="stylesheet" />
    <script src="~/Scripts/jquery-3.3.1.min.js"></script>
    <script type="text/javascript">
                $(function () { $('a[href^="#"]').click(function () { var target = $(this).attr('href'); $('html, body').animate({ scrollTop: $(target).offset().top }, 1500); return false; }); });
    </script>
</head>
<body>
    <div id="center">
        <div id="first_block">
            <div id="logo">
                <img src="../../Content/2000px-Tennis_pictogram.png" alt="Логотип">
                <div id="heading1">
                    БЕЛГОРОДСКИЙ <br> ТЕННИСНЫЙ <br> КЛУБ
                </div>
            </div>
            <div id="menu">
                <ul>
                    <li>
                        <a href="#about">О НАС</a>
                    </li>
                    <li>
                        <a href="#coaches">ТРЕНЕРСКИЙ СОСТАВ</a>
                    </li>
                    <li>
                        <a href="#location">МЕСТОПОЛОЖЕНИЕ</a>
                    </li>
                    <li>
                        <a href="#contacts">КОНТАКТЫ</a>
                    </li>
                </ul>
            </div>
            <div id="heading2">
                ОБРАЗ ЖИЗНИ, <br> КОТОРЫЙ <br> ЗАРЯЖАЕТ <br> ЭНЕРГИЕЙ!
            </div>
            <div id="line">
                <a href="/Admin/Index"></a>
            </div>
            <p>НАЧНИ ПРЯМО <br> СЕЙЧАС! </p>
            <a id="order_court" target="_blank" href="/Home/Date">Заказать корт</a>
        </div>
        <div id="about">
            <img id="nadal" src="../../Content/nadal.png">
            <h1> О НАС </h1>
            <div class="all">
                <input checked type="radio" name="respond" id="desktop">
```

```
                    <article id="slider">
                        <input checked type="radio" name="slider" id="switch1">
                        <input type="radio" name="slider" id="switch2">
                        <input type="radio" name="slider" id="switch3">
                        <input type="radio" name="slider" id="switch4">
                        <input type="radio" name="slider" id="switch5">
                        <div id="slides">
                            <div id="overflow">
                                <div class="image">
                                    <article><img src="../../Content/1.jpg"></article>
                                    <article><img src="../../Content/2.jpg"></article>
                                    <article><img src="../../Content/3.jpg"></article>
                                    <article><img src="../../Content/4.jpg"></article>
                                    <article><img src="../../Content/5.jpg"></article>
                                </div>
                            </div>
                        </div>
                        <div id="controls">
                            <label for="switch1"></label>
                            <label for="switch2"></label>
                            <label for="switch3"></label>
                            <label for="switch4"></label>
                            <label for="switch5"></label>
                        </div>
                        <div id="active">
                            <label for="switch1"></label>
                            <label for="switch2"></label>
                            <label for="switch3"></label>
                            <label for="switch4"></label>
                            <label for="switch5"></label>
                        </div>
                    </article>
                </div>
                <table>
                    <tr>
                        <td> <span>4</span> <br> травяных корта </td>
                        <td> <span>1000+</span><br> посетителей в месяц </td>
                         <td> <img src="../../Content/kisspng-brooklyn-boulders-logo-business-climbing-boulderin-no-cost-
5b100379291336.975284581527761211683.png"> <br> Бесплатная парковка</td>
                          <td> <img src="../../Content/kisspng-brooklyn-boulders-logo-business-climbing-boulderin-no-cost-
5b100379291336.975284581527761211683.png"> <br> Бесплатные спортивные сооружения</td>
                    </tr>
                </table>
            </div>
            <div id="coaches">
                <h1>ТРЕНЕРСКИЙ СОСТАВ</h1>
                <table>
                    <tr>
                        <td>
                            <img src="../../Content/IMG_20170806_152056.jpg" class="photo" alt="Давыдов">
                            Давыдов Вячеслав Олегович
                        </td>
                        <td>
                            <img src="../../Content/IMG_20180626_1401381.jpg" class="photo" alt="Бубнов">
                            Бубнов Сергей Владимирович
                        </td>
                        <td>
                            <img src="../../Content/72353.png" class="photo" alt="Виняр">
                            Виняр Павел Андреевич
                        </td>
                        <td>
```

```
                <img src="../../Content/4325.png" class="photo" alt="Виняр">
                Виняр Елизавета Юрьевна
              </td>
            </tr>
            <tr>
              <td>
                <img src="../../Content/534.Png" class="photo" alt="Кобзарь">
                Кобзарь Егор Олегович
              </td>
              <td>
                <img src="../../Content/2018-09-20-08.24.png" class="photo" alt="Луценко">
                Луценко Нелли Сергеевна
              </td>
              <td>
                <img src="../../Content/пакв1.png" class="photo" alt="Мартыненко">
                Мартыненко Михаил Валентинович
              </td>
              <td>
                <img src="../../Content/coach1.jpg" class="photo" alt="Рябков">
                Рябков Павел Александрович
              </td>
            </tr>
            <tr>
              <td colspan="4">
                <img src="../../Content/0000180485_cdwdmh3r.jpg" class="photo" alt="Жукова">
                Жукова Мария Андреевна
              </td>
            </tr>
          </table>
        </div>
        <div id="contacts">
          <h1>КОНТАКТЫ</h1>
          <div id="container">
            <div id="bubnov">
              <img src="../../Content/IMG_20180626_1401381.jpg" class="photo" alt="Бубнов">
              <div class="information">
                <p> Бубнов Сергей Владимирович<p>
                  <img src="../../Content/phone-802177_12801.png" class="phone" alt="Логотип">
                <p> +7-905-170-64-20<p>
              </div>
            </div>
            <div id="me">
              <img src="../../Content/IMG_20170806_152056.jpg" class="photo" alt="Давыдов">
              <div class="information">
                <p> Давыдов Вячеслав Олегович<p>
                  <img src="../../Content/phone-802177_12801.png" class="phone" alt="Логотип">
                  <img src="../../Content/telegram_icon-icons.com_720551.png" class="phone" alt="Логотип">
                  <img src="../../Content/whatsapp_icon-icons.com_720541.png" class="phone" alt="Логотип">
                  <img src="../../Content/viber_icon-icons.com_720201.png" class="phone" alt="Логотип">
                <p> +7-951-158-68-26<p>
                  <img src="../../Content/logo-1162901_19201.png" id="email_logo" alt="Логотип">
                <p> fck3135@gmail.com<p>
              </div>
            </div>
          </div>
        </div>
        <div id="footer">
          С РАДОСТЬЮ ЖДЁМ ВАС!
          <a id="order_court" target="_blank" href="/Home/Date">Заказать корт</a>
        </div>
      </div>
```

```
</body>
</html>
```

- Представление «Date.cshtml»

```
@{
Layout = null;
}

<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8" />
   <link rel="icon" href="../../Content/favicon.ico" type="image/x-icon">
   <title>Дата</title>
   <script src="~/Scripts/jquery-3.3.1.min.js"></script>
   <link href="~/Content/bootstrap.min.css" rel="stylesheet" />
   <link href="~/Content/bootstrap-datetimepicker.css" rel="stylesheet" />
   <script>
      $(function () { $('#datetimepicker1').datetimepicker({ format: 'DD/MM/YYYY', locale: 'ru' }); });
      $(function () { $('#datetimepicker1').data("DateTimePicker").minDate(moment()); });
                  $(function  ()  {  $('#datetimepicker1').data("DateTimePicker").maxDate(moment('01-11-2020',
'DD.MM.YYYY')); });
   </script>
   <script src="~/Scripts/moment-with-locales.min.js"></script>
   <script src="~/Scripts/bootstrap.min.js"></script>
   <script src="~/Scripts/bootstrap-datetimepicker.min.js"></script>
   <script src="~/Scripts/jquery.maskedinput.min.js"></script>
   <link href="~/Content/date.css" rel="stylesheet" />
   <script>
      $(function () { $("#date").mask("99/99/9999"); });
   </script>
</head>
<body>
   <h1>Давайте начнём!</h1>
   <form method="post" action="/Home/Time">
      <table id="tbl">
         <tr>
            <td class="tab">Дата</td>
            <td>
               <div class="form-group">
                  <div class="input-group" id="datetimepicker1">
                                          <input  type="text"  class="form-control"  name="Date"  id="date"  required
placeholder="XX/XX/XXXX">
                     <span class="input-group-addon">
                        <span class="glyphicon glyphicon-calendar"></span>
                     </span>
                  </div>
               </div>
            </td>
         </tr>
         <tr>
            <td class="tab">Тренерские услуги </td>
            <td>
               <select name="Coach" required>
                  @foreach (var b in ViewBag.Coaches)
                  {
                     <option class="opt">@b.Name</option>
                  }
               </select>
            </td>
         </tr>
```

```
            </table>
            <center>
                <input type="submit" value="Далее">
                <a href="/Home/Index" id="cancel"> Отмена</a>
            </center>
        </form>
    </body>
    </html>
```

- Представление «Error.cshtml»

```
    @{
        Layout = null;
    }

    <!DOCTYPE html>
    <html>
    <head>
        <meta charset="utf-8" />
        <title>Ошибочка</title>
        <link href="~/Content/error.css" rel="stylesheet" />
        <link rel="icon" href="../../Content/favicon.ico" type="image/x-icon">
    </head>
        <body>
            <div>
                    <h1>УПС! Кажется кто-то успел забронировать данное время на несколько секунд раньше Вас(</h1>
                <center>
                    <a href="/Home/Date">Повторить попытку</a>
                    <a href="/Home/Index">Отмена</a>
                </center>
            </div>
        </body>
    </html>
```

- Представление «ErrorLogin.cshtml»

```
    @{
        Layout = null;
    }
    <!DOCTYPE html>
    <html>
        <head>
            <meta charset="utf-8" />
            <title>Ошибка</title>
        </head>
        <body>
            <center>
                <h1>Неверный пароль!</h1>
                <a href="/Home/Login">Повторить попытку</a>
            </center>
        </body>
    </html>
```

- Представление «Login.cshtml»

```
    @{
        Layout = null;
    }

    <!DOCTYPE html>
    <html>
        <head>
            <meta charset="utf-8" />
```

```
        <title>Вход</title>
    </head>
    <body>
        <center>
            <h1>Если Вы зашли на эту страницу случайно, то вернитесь на главную!</h1>
            <h2>
                <a href="/Home/Index">На главную</a>
            </h2>
            <h1>Только для администратора!</h1>
            <form method="post" action="/Home/CheckingLogin">
                <label for="password">Пароль</label>
                <input type="password" autocomplete="off" name="Password" required />
                <input type="submit" value="ВОЙТИ" />
            </form>
        </center>
    </body>
</html>
```

- Представление «OK.cshtml»

```
@{
    Layout = null;
}

<!DOCTYPE html>
<html>
    <head>
        <meta name="viewport" content="width=device-width" />
        <link href="~/Content/ok.css" rel="stylesheet" />
        <title>Ураaa!</title>
        <link rel="icon" href="../../Content/favicon.ico" type="image/x-icon">
    </head>
    <body>
        <h1>Оплата прошла успешно!</h1>
        <h2>Ваши данные</h2>
        <table>
            <tr>
                <td>Имя</td>
                <td>@ViewBag.Name</td>
            </tr>
            <tr>
                <td>Фамилия</td>
                <td>@ViewBag.Surname</td>
            </tr>
            <tr>
                <td>Номер телефона</td>
                <td>@ViewBag.Phone</td>
            </tr>
            <tr>
                <td>Дата аренды</td>
                <td>@ViewBag.Date</td>
            </tr>
            <tr>
                <td>Время аренды</td>
                <td>@ViewBag.Time</td>
            </tr>
            <tr>
                <td>Тренерские услуги</td>
                <td>@ViewBag.Coach</td>
            </tr>
            <tr>
                <td>Стоимость</td>
```

```
                <td>@ViewBag.Sum руб.</td>
            </tr>
            <tr>
                <td>Дата оформления покупки</td>
                <td>@ViewBag.Today</td>
            </tr>
        </table>
        <h2>Пожалуйста, не опаздывайте!</h2>
            <h3>По всем вопросам звоните: <br> +79051704620 - Бубнов Сергей Владимирович <br> +79511586826 - Давыдов Вячеслав Олегович </h3>
        <h1>Спасибо, что Вы с нами!</h1>
        <center>
            <a href="/Home/Index"> На главную</a>
            <a href="javascript:(print());">Распечатать</a>
        </center>
    </body>
</html>
```

- Представление «OrderDetails.cshtml»

```
@{
    Layout = null;
}

<DOCTYPE html>
    <html>
    <head>
        <meta charset="utf-8">
        <link rel="icon" href="../../Content/favicon.ico" type="image/x-icon">
        <link href="~/Content/orderdetails.css" rel="stylesheet" />
        <script src="~/Scripts/jquery-3.3.1.min.js"></script>
        <script src="~/Scripts/jquery.maskedinput.min.js"></script>
        <script>
            $(function () { $("#phone").mask("+7-999-999-99-99"); });
        </script>
        <script>
            $(function () { $("#cardNum").mask("9999 9999 9999 9999"); });
        </script>
        <script>
            $(function () { $("#cardDate").mask("99/99"); });
        </script>
        <script>
            $(function () { $("#cardCVC").mask("999"); });
        </script>
        <title> Оплата </title>
    </head>
    <body>
        <h1>Последний шаг!</h1>
        <P>
            Стоимость аренды корта без тренера - <u>300 руб/час</u>, с тренером - <u>1000 руб/час</u>. Спортивное оборудование предоставляется <u><b>бесплатно</b></u>.
        </P>
        <form method="post" action="/Home/Payment">
            <table id="tbl">
                <tr>
                    <td>Имя </td>
                    <td>
                        <input type="text" name="Name" required placeholder="Ваше имя" class="form">
                    </td>
                </tr>
                <tr>
                    <td>Фамилия </td>
```

```
                    <td>
                        <input type="text" name="Surname" required placeholder="Ваша фамилия" class="form">
                    </td>
                </tr>
                <tr>
                    <td>Номер телефона </td>
                    <td>
                        <input type="text" name="Phone" required id="phone" placeholder="+7-XXX-XXX-XX-XX"
class="form">
                    </td>
                </tr>
                <tr>
                    <td>Спортивное оборудование </td>
                    <td>
                        <input type="checkbox" name="SportsEquipment" value="true" id="cb">
                        <label for="cb"></label>
                    </td>
                </tr>
                <tr>
                    <td>Номер карты</td>
                    <td>
                        <input type="text" name="CardNumber" id="cardNum" required placeholder="XXXX XXXX
XXXX XXXX" autocomplete="off">
                    </td>
                </tr>
                <tr>
                    <td>Срок действия карты</td>
                    <td>
                        <input autocomplete="off" type="text" name="CardDate" required id="cardDate"
placeholder="XX/XX">
                    </td>
                </tr>
                <tr>
                    <td>Проверочный код карты <br> (3 цифры на обороте карты)</td>
                    <td>
                        <input autocomplete="off" type="text" placeholder="XXX" name="CardCVC" required
id="cardCVC" />
                    </td>
                </tr>
            </table>
            <center>
                <input type="submit" value="Оплатить">
                <a href="/Home/Date" id="cancel"> Назад</a>
                <a href="/Home/Index" id="cancel"> Отмена</a>
            </center>
            <input type="text" hidden name="Coach" value="@ViewBag.Coach">
            <input type="text" hidden name="Date" value="@ViewBag.Date">
            <input type="text" hidden name="TimeStartId" value="@ViewBag.TimeStartId">
            <input type="text" hidden name="TimeEndId" value="@ViewBag.TimeEndId">
            <input type="text" hidden name="Time" value="@ViewBag.Time">
        </form>
    </body>
</html>
```

- Представление «Time.cshtml»

```
@{
Layout = null;
}

<!DOCTYPE html>
```

```
<html>
<head>
   <link rel="icon" href="../../Content/favicon.ico" type="image/x-icon">
   <link href="~/Content/date.css" rel="stylesheet" />
   <link href="~/Content/time.css" rel="stylesheet" />
   <meta charset="utf-8">
   <title>Время</title>
</head>
   <body>
      <h1>Осталось ещё чуть-чуть...</h1>
      <p>На <u><b>@ViewBag.Date</b></u> доступно следующее время:</p>
      <form method="post" action="/Home/OrderDetails">
         <center>
            <select name="Time">
               @foreach (var p in ViewBag.Periods)
               {
                  <option>@p</option>
               }
            </select>
         </center>
         <center>
            <input type="submit" value="Далее">
            <a href="/Home/Date" id="cancel"> Назад</a>
            <a href="/Home/Index" id="cancel"> Отмена</a>
         </center>
         <input type="text" name="Date" hidden value="@ViewBag.Date">
         <input type="text" name="Coach" hidden value="@ViewBag.Coach">
      </form>
   </body>
</html>
```

- Файл стилей «time.css»

```
p{
   color: white;
   font-size: 25px;
}

select {
   width: 300px;
}
```

- Файл стилей «orderdetails.css»

```
body {
   background-color: green;
   font-family: Arial, non-serif;
   width: 900px;
   margin: 0 auto;
   user-select: none;
   -webkit-user-select: none;
}

h1 {
   color: yellow;
   font-size: 40px;
   font-weight: bold;
   font-style: italic;
   text-align: center;
}

p {
   color: white;
```

```
    font-size: 25px;
    text-align: center;
    padding: 0 10px;
}

#cancel {
    text-decoration: none;
    margin: 0 auto;
    background-color: red;
    color: white;
    padding: 10px 30px;
    border-radius: 35px;
    font-size: 30px;
    margin: 10px;
}

input[type=text]{
    width: 100%;
    color: black;
    background-color: yellow;
    font-weight: bold;
    font-style: italic;
    margin: 5px 0;
    padding: 10px;
    border: 3px solid yellow;
    font-size: 25px;
    border-radius: 20px;
}

    input[type=text]:hover{
        border: 3px solid blue;
    }

#tbl{
    font-size: 25px;
    color: white;
    padding: 0 20px;
    margin: 0 50px;
}

#cancel:hover {
    background-color: white;
    color: red;
}

input[type=submit] {
    text-decoration: none;
    color: green;
    margin: 0 auto;
    background-color: yellow;
    margin: 10px;
    padding: 10px 30px;
    border-radius: 35px;
    border: 3px solid yellow;
    font-size: 30px;
}

    input[type=submit]:hover {
        background-color: white;
        color: green;
        border: 3px solid white;
```

```
    }

input[type=checkbox]{
    display: none;
}

label:before {
    content: " ";
    width: 35px;
    height: 35px;
    background-color    : yellow;
    display: inline-block;
}

label:hover {
    outline: solid blue 3px;
}

input:checked + label:before {
    content: "X";
    font-weight: bold;
    font-size: 30px;
    display: block;
    text-align: center;
    color: black;
}

input:disabled + label:before {
    background-color: yellow;
}
```

- Файл стилей «ok.css»

```
body {
    background-color: white;
    font-family: Arial, non-serif;
    width: 500px;
    margin: 0 auto;
    user-select: none;
    -webkit-user-select: none;
    margin-bottom: 10px;
}
h1 {
    color: black;
    font-size: 40px;
    font-weight: bold;
    font-style: italic;
    text-align: center;
    margin-bottom: 20px;
}

h2{
    font-size:30px;
    text-align: left;
}

h3 {
    font-size: 20px;
    text-align: left;
}

a{
```

```
    color: blue;
    margin: 10px;
    text-decoration: none;
    font-size: 30px;
}

a:hover{
    color: black;
}

table td{
    font-size: 20px;
    padding: 5px;
    padding-left: 30px;
}

@media print {
    a{
        display: none;
    }
}
```

- Файл стилей «index.css»

```
body {
    margin: 0;
    padding: 0;
    font-family: arial,non-serif;
    min-width: 1300px;
    user-select: none;
    -webkit-user-select: none
}

#center {
    width: 100%;
    margin: 0 auto
}

#first_block {
    background-image: url(../../Content/6836244918_d6fa89a154_o.jpg);
    background-size: 100%;
    background-repeat: no-repeat;
    height: 800px;
    width: 100%
}

#logo {
    padding-top: 10px;
    margin-left: 40px;
    min-width: 470px;
    float: left
}

    #logo img {
        width: 120px;
        height: 120px;
        float: left;
        margin-right: 10px
    }

#heading1, #location h1, #contacts h1, #footer, #coaches h1 {
    color: #ff0;
```

```
    font-size: 40px;
    font-weight: 700;
    font-style: italic;
    margin-left: 0
}

#heading2, #first_block p {
    margin-top: 150px;
    margin-left: 70px;
    margin-bottom: 50px;
    color: #fff;
    font-size: 50px;
    font-weight: 700;
    font-style: italic
}

#first_block p {
    margin-top: 50px
}

#line a {
    display: block;
    border: solid #fff 5px;
    height: 0px;
    width: 100px;
    margin-left: 70px;
    margin-bottom: 50px
}

#order_court {
    text-decoration: none;
    color: #fff;
    padding: 15px;
    width: 100%;
    padding: 15px;
    background-color: green;
    color: #fff;
    width: 190px;
    height: 40px;
    border-radius: 35px;
    margin-left: 70px;
    font-size: 30px
}

    #order_court:hover {
        background-color: #ff0;
        color: green
    }

#menu ul {
    margin: 0;
    padding-top: 10px;
    display: flex;
    list-style-type: none;
    color: #fff;
    font-size: 18px;
    justify-content: space-around
}

#menu li {
    padding: 7px
```

```
}

#menu a {
    text-decoration: none;
    color: #fff
}

    #menu a:hover {
        text-decoration: underline;
        color: #ff0
    }

#location {
    width: 100%;
    background-color: green;
    margin: 0 auto;
}

#location div{
    width: 1300px;
    margin: 0 auto;
}

    #location h1, #contacts h1, #footer, #coaches h1, #about h1 {
        margin: 0;
        padding: 20px;
        text-align: center
    }

    #location p {
        color: #fff;
        font-size: 25px;
        font-style: italic;
        float: left;
        width: 30%;
        padding-left: 20px;
        line-height: 40px;
        margin-top: 100px
    }

    #location iframe {
        margin-bottom: 10px;
        border-radius: 40px;
        width: 750px;
        height: 450px;
        margin-left: 30px
    }

#contacts {
    height: 350px;
    width: 100%;
    background-color: grey;
    margin: 0
}
    #contacts p {
        color: #fff;
        font-style: italic;
        font-size: 25px;
        font-weight: 700
    }
```

```
.photo {
    width: 200px;
    height: 200px;
    border-radius: 150px
}

#bubnov {
    width: 50%;
    float: left
}

#me img, #bubnov img {
    margin: 0 10px;
    float: left
}

#me {
    height: 100%
}

#email_logo {
    width: 30px;
    height: 20px
}

.phone {
    width: 20px;
    height: 25px
}

.information {
    padding-top: 10px
}

#container {
    width: 1200px;
    margin: 0 auto;
    float: none
}

#footer {
    height: 50px;
    background-color: green
}

#coaches {
    height: 100%;
    width: 100%
}

    #coaches img {
        display: block;
        margin: 0 auto
    }

table {
    width: 1000px;
    margin: 0 auto;
    text-align: center;
    font-style: italic;
    font-weight: 700;
```

```
    font-size: 20px
}

#coaches h1 {
    color: #000
}

#footer a {
    background-color: #ff0;
    color: #000
}

    #footer a:hover {
        background-color: #fff;
        color: #000
    }

#about {
    background-color: #90ee90;
    margin: 0;
    height: 580px
}

#nadal {
    position: absolute
}

#about h1 {
    color: #000;
    font-weight: 700;
    font-style: italic
}

#about table {
    width: 60%;
    margin: 0 auto;
    float: left;
    margin-left: 400px
}

    #about table span {
        color: red;
        font-size: 35px;
        font-weight: 700
    }

    #about table img {
        width: 50px;
        height: 50px;
        float: none
    }

#big {
    font-size: 50px;
    font-weight: 700;
    color: red
}

#slider {
    margin: 0 auto;
    margin-left: 400px
```

```
}

#slides article {
   width: 20%;
   float: left
}

#slides .image {
   width: 500%;
   line-height: 0
}

#overflow {
   width: 100%;
   overflow: hidden
}

article img {
   width: 100%
}

#desktop:checked ~ #slider {
   max-width: 800px
}

#switch1:checked ~ #controls label:nth-child(5), #switch2:checked ~ #controls label:nth-child(1), #switch3:checked ~ #controls label:nth-child(2), #switch4:checked ~ #controls label:nth-child(3), #switch5:checked ~ #controls label:nth-child(4) {
   background: url(prev.png) no-repeat;
   float: left;
   margin: 0 0 0 -84px;
   display: block;
   height: 68px;
   width: 68px
}

#switch1:checked ~ #controls label:nth-child(2), #switch2:checked ~ #controls label:nth-child(3), #switch3:checked ~ #controls label:nth-child(4), #switch4:checked ~ #controls label:nth-child(5), #switch5:checked ~ #controls label:nth-child(1) {
   background: url(next.png) no-repeat;
   float: right;
   margin: 0 -84px 0 0;
   display: block;
   height: 68px;
   width: 68px
}

label, a {
   cursor: pointer
}

.all input {
   display: none
}

#switch1:checked ~ #slides .image {
   margin-left: 0
}

#switch2:checked ~ #slides .image {
   margin-left: -100%
```

```
}

#switch3:checked ~ #slides .image {
   margin-left: -200%
}

#switch4:checked ~ #slides .image {
   margin-left: -300%
}

#switch5:checked ~ #slides .image {
   margin-left: -400%
}

#controls {
   margin: -25% 0 0;
   width: 100%;
   height: 50px
}

#active label {
   border-radius: 10px;
   display: inline-block;
   width: 15px;
   height: 15px;
   background: #bbb
}

#active {
   margin: 23% 0 0;
   text-align: center
}

   #active label:hover {
      background: #76c8ff;
      border-color: #777 !important
   }

#switch1:checked ~ #active label:nth-child(1), #switch2:checked ~ #active label:nth-child(2), #switch3:checked ~ #active label:nth-child(3), #switch4:checked ~ #active label:nth-child(4), #switch5:checked ~ #active label:nth-child(5) {
   background: #18a3dd;
   border-color: #18a3dd !important
}

#slides .image {
   transition: all 800ms cubic-bezier(0.770,0.000,0.175,1.000)
}

#controls label:hover {
   opacity: .6
}

#controls label {
   transition: opacity .2s ease-out
}
```

- Файл стилей «error.css»

```
body {
   background-color: red;
   font-family: arial, non-serif;
   min-width: 1000px;
```

```
}

div{
    width: 1000px;
    margin: 0 auto;
}

h1{
    color: white;
    font-weight: bold;
    font-size: 50px;
    font-style: italic;
    text-align: center;
}

a {
    text-decoration: none;
    color: yellow;
    margin: 0 auto;
    background-color: green;
    padding: 20px 30px;
    border-radius: 35px;
    font-size: 30px;
    margin: 10px;
}

a:hover{
    background-color: yellow;
    color: black;
}
```

- Файл стилей «date.css»

```
body {
    background-color: green;
    font-family: Arial, non-serif;
    width: 900px;
    margin: 0 auto;
    user-select: none;
    -webkit-user-select: none;
}

h1 {
    color: yellow;
    font-size: 40px;
    font-weight: bold;
    font-style: italic;
    text-align: center;
    margin-bottom: 20px;
}

#tbl{
    font-size: 25px;
    padding: 0 20px;
    margin: 0 50px;
}

#tbl td {
    font-size: 25px;
    text-align: center;
    padding: 0 10px;
}
```

```
.tab{
   color: white;
}

.form-group, #datetimepicker1 td {
   font-size: 15px;
}

input[type=submit] {
   text-decoration: none;
   color: green;
   margin: 0 auto;
   background-color: yellow;
   margin: 10px;
   padding: 10px 30px;
   border-radius: 35px;
   border: 3px solid yellow;
   font-size: 30px;
}

   input[type=submit]:hover {
      background-color: white;
      color: green;
      border: 3px solid white;
   }

#cancel {
   text-decoration: none;
   color: white;
   margin: 0 auto;
   background-color: red;
   color: white;
   padding: 10px 30px;
   border-radius: 35px;
   font-size: 30px;
   margin: 10px;
}

#cancel:hover {
   background-color: white;
   color: red;
}

input[type=text],select, option {
   width: 100%;
   color: black;
   background-color: yellow;
   font-weight: bold;
   font-style: italic;
   margin: 5px 0;
   padding: 10px;
   border: 3px solid yellow;
   font-size: 25px;
   border-radius: 20px;
}
   input[type=text]:hover {
      border: 3px solid blue;
   }

.form-group {
```

```
    border-radius: 20px;
    width: 100%;
}

.form-control {
    background-color: yellow;
    height: 55px;
    color: black;
}

.input-group-addon {
    color: white;
    background-color: green;
    border: solid green 3px;
    font-size: 30px;
}
```

- Контроллер «AdminController.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using course2.Models;
using System.Web.Security;
using System.Data.Entity;
using System.Collections;
using ClosedXML.Excel;
using System.IO;
using DocumentFormat.OpenXml.Spreadsheet;

namespace course2.Controllers
{
    public class AdminController : Controller
    {
        Context db = new Context();
        public ActionResult Index()
        {
            return View("Index");
        }
        public ActionResult Exit()
        {
            FormsAuthentication.SignOut();
            return RedirectToAction("Index", "Home");
        }
        public ActionResult Clients()
        {
            ArrayList clients = new ArrayList();
            foreach (var c in db.Clients)
            {
                if (c.Name != "---УДАЛЁН---")
                    clients.Add(c);
            }
            ViewBag.Clients = clients;
            return View("Clients");
        }
        public ActionResult Coaches()
        {
            ArrayList coaches = new ArrayList();
            foreach (var c in db.Coaches)
            {
```

```
        if (c.Name != "---УДАЛЁН---")
            coaches.Add(c);
    }
    ViewBag.Coaches = coaches;
    return View("Coaches");
}
public ActionResult AddCoach(string Name)
{
    bool check = false;
    Coach c = new Coach { Name = Name, Working = true };
    foreach (var v in db.Coaches)
    {
        if (v.Name == c.Name)
        {
            check = true;
            break;
        }
    }
    if (!check)
        db.Coaches.Add(c);
    db.SaveChanges();
    ArrayList coaches = new ArrayList();
    foreach (var g in db.Coaches)
    {
        if (g.Name != "---УДАЛЁН---")
            coaches.Add(g);
    }
    ViewBag.Coaches = coaches;
    return View("Coaches");
}
public ActionResult RemoveCoach(int id)
{
    foreach (var b in db.Coaches)
    {
        if (b.Id == id)
        {
            db.Coaches.Remove(b);
            break;
        }
    }
    db.SaveChanges();
    ArrayList coaches = new ArrayList();
    foreach (var b in db.Coaches)
    {
        if (b.Name != "---УДАЛЁН---")
            coaches.Add(b);
    }
    ViewBag.Coaches = coaches;
    return View("Coaches");
}
public ActionResult RemoveCoaches()
{
    foreach (var b in db.Coaches)
    {
        db.Coaches.Remove(b);
        break;
    }
    db.SaveChanges();
    ViewBag.Coaches = db.Coaches;
    return View("Coaches");
}
```

```
        public ActionResult HideCoach(int id)
        {
            ArrayList coaches = new ArrayList();
            foreach (var g in db.Coaches)
            {
                if (g.Id == id)
                {
                    g.Working = false;
                }
                if (g.Name != "---УДАЛЁН---")
                    coaches.Add(g);
            }
            ViewBag.Coaches = coaches;
            db.SaveChanges();
            return View("Coaches");
        }
        public ActionResult ShowCoach(int id)
        {
            ArrayList coaches = new ArrayList();
            foreach (var g in db.Coaches)
            {
                if (g.Id == id)
                {
                    g.Working = true;
                }
                if (g.Name != "---УДАЛЁН---")
                    coaches.Add(g);
            }
            ViewBag.Coaches = coaches;
            db.SaveChanges();
            return View("Coaches");
        }
        public ActionResult Purchases()
        {
                ViewBag.Purchases = db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p =>
p.TimeStart).Include(p => p.TimeEnd);
            return View("Purchases");
        }
        public ActionResult RemoveClient(int id)
        {
            foreach (var b in db.Clients)
            {
                if (b.Id == id)
                {
                    db.Clients.Remove(b);
                    break;
                }
            }
            db.SaveChanges();
            ArrayList clients = new ArrayList();
            foreach (var b in db.Clients)
            {
                if (b.Name != "---УДАЛЁН---")
                    clients.Add(b);
            }
            ViewBag.Clients = clients;
            return View("Clients");
        }
        public ActionResult RemoveClients()
        {
            foreach (var b in db.Clients)
```

```
            {
                db.Clients.Remove(b);
            }
            db.SaveChanges();
            ViewBag.Clients = db.Clients;
            return View("Clients");
        }
        public ActionResult RemovePurchases()
        {
            foreach (var b in db.Purchases)
            {
                db.Purchases.Remove(b);
            }
            db.SaveChanges();
            ViewBag.Purchases = db.Purchases;
            return View("Purchases");
        }
        public ActionResult RemovePurchase(int id)
        {
            foreach (var b in db.Purchases)
            {
                if (b.Id == id)
                {
                    db.Purchases.Remove(b);
                    break;
                }
            }
            db.SaveChanges();
                ViewBag.Purchases = db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p =>
p.TimeStart).Include(p => p.TimeEnd);
            return View("Purchases");
        }
        public ActionResult ClearAll()
        {
            foreach (var g in db.Clients)
            {
                db.Clients.Remove(g);
            }
            foreach (var g in db.Coaches)
            {
                db.Coaches.Remove(g);
            }
            foreach (var g in db.Purchases)
            {
                db.Purchases.Remove(g);
            }
            db.SaveChanges();
            return View("Index");
        }
        public ActionResult SearchClientPhoneNameSurname(string Name, string Surname, string Phone)
        {
            ArrayList clients = new ArrayList();
            foreach (var v in db.Clients)
            {
                if ((v.Name == Name) && (v.Surname == Surname) && (v.Phone == Phone))
                {
                    clients.Add(v);
                }
            }
            ViewBag.Clients = clients;
            return View("SearchClient");
```

```
        }
        public ActionResult SearchClientName(string Name)
        {
            ArrayList clients = new ArrayList();
            foreach (var v in db.Clients)
            {
                if (v.Name == Name)
                {
                    clients.Add(v);
                }
            }
            ViewBag.Clients = clients;
            return View("SearchClient");
        }
        public ActionResult SearchClientSurname(string Surname)
        {
            ArrayList clients = new ArrayList();
            foreach (var v in db.Clients)
            {
                if (v.Surname == Surname)
                {
                    clients.Add(v);
                }
            }
            ViewBag.Clients = clients;
            return View("SearchClient");
        }
        public ActionResult SearchClientNameSurname(string Name, string Surname)
        {
            ArrayList clients = new ArrayList();
            foreach (var v in db.Clients)
            {
                if ((v.Name == Name) && (v.Surname == Surname))
                {
                    clients.Add(v);
                }
            }
            ViewBag.Clients = clients;
            return View("SearchClient");
        }
        public ActionResult SearchClientPhone(string Phone)
        {
            ArrayList clients = new ArrayList();
            foreach (var v in db.Clients)
            {
                if (v.Phone == Phone)
                {
                    clients.Add(v);
                }
            }
            ViewBag.Clients = clients;
            return View("SearchClient");
        }
        public ActionResult SearchPurchaseNameSurname(string Name, string Surname)
        {
            ArrayList purchases = new ArrayList();
                    foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p => p.TimeStart).Include(p => p.TimeEnd))
            {
                if ((v.Client.Name == Name) && (v.Client.Surname == Surname))
                {
```

```
                purchases.Add(v);
            }
        }
        ViewBag.Purchases = purchases;
        return View("SearchPurchase");
    }
    public ActionResult SearchPurchaseCoachName(string Name)
    {
        ArrayList purchases = new ArrayList();
                foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p =>
p.TimeStart).Include(p => p.TimeEnd))
        {
            if (v.Coach.Name == Name)
            {
                purchases.Add(v);
            }
        }
        ViewBag.Purchases = purchases;
        return View("SearchPurchase");
    }
    public ActionResult SearchPurchaseDate(string Date)
    {
        ArrayList purchases = new ArrayList();
                foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p =>
p.TimeStart).Include(p => p.TimeEnd))
        {
            if (v.Date == Date)
            {
                purchases.Add(v);
            }
        }
        ViewBag.Purchases = purchases;
        return View("SearchPurchase");
    }
    public ActionResult SearchPurchaseTime(string Time)
    {
        ArrayList purchases = new ArrayList();
                foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p =>
p.TimeStart).Include(p => p.TimeEnd))
        {
            if (v.TimeStart.Period + " - " + v.TimeEnd.Period == Time)
            {
                purchases.Add(v);
            }
        }
        ViewBag.Purchases = purchases;
        return View("SearchPurchase");
    }
    public ActionResult SearchPurchaseDateTime(string Date, string Time)
    {
        ArrayList purchases = new ArrayList();
                foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p =>
p.TimeStart).Include(p => p.TimeEnd))
        {
            if ((v.TimeStart.Period + " " + v.TimeEnd.Period == Time) && (v.Date == Date))
            {
                purchases.Add(v);
            }
        }
        ViewBag.Purchases = purchases;
        return View("SearchPurchase");
```

```
        }
        public ActionResult SearchPurchaseCardNumber(string CardNUmber)
        {
            ArrayList purchases = new ArrayList();
                    foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p => p.TimeStart).Include(p => p.TimeEnd))
            {
                if (v.CardNumber == CardNUmber)
                {
                    purchases.Add(v);
                }
            }
            ViewBag.Purchases = purchases;
            return View("SearchPurchase");
        }
        public ActionResult SearchPurchaseNameSurnameDate(string Name, string Surname, string Date)
        {
            ArrayList purchases = new ArrayList();
                    foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p => p.TimeStart).Include(p => p.TimeEnd))
            {
                if ((v.Client.Name == Name) && (v.Client.Surname == Surname) && (v.Date == Date))
                {
                    purchases.Add(v);
                }
            }
            ViewBag.Purchases = purchases;
            return View("SearchPurchase");
        }
        public ActionResult SearchPurchaseCoachNameDate(string Name, string Date)
        {
            ArrayList purchases = new ArrayList();
                    foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p => p.TimeStart).Include(p => p.TimeEnd))
            {
                if ((v.Coach.Name == Name) && (v.Date == Date))
                {
                    purchases.Add(v);
                }
            }
            ViewBag.Purchases = purchases;
            return View("SearchPurchase");
        }
        public ActionResult Download()
        {
            using (XLWorkbook workbook = new XLWorkbook(XLEventTracking.Disabled))
            {
                var worksheet = workbook.Worksheets.Add("Корты");
                worksheet.Cell("A1").Value = "Id";
                worksheet.Cell("B1").Value = "Клиент";
                worksheet.Cell("C1").Value = "Тренер";
                worksheet.Cell("D1").Value = "Дата";
                worksheet.Cell("E1").Value = "Время";
                worksheet.Cell("F1").Value = "Спортивное оборудование";
                worksheet.Cell("G1").Value = "Номер карты";
                worksheet.Cell("H1").Value = "Срок действия карты";
                worksheet.Cell("I1").Value = "Проверочный код карты";
                worksheet.Cell("J1").Value = "Сумма, руб.";
                worksheet.Row(1).Style.Font.Bold = true;
                int i = 2;
```

```
                    foreach (var v in db.Purchases.Include(p => p.Coach).Include(p => p.Client).Include(p => p.TimeStart).Include(p => p.TimeEnd))
                    {
                        worksheet.Cell(i,1).Value = v.Id;
                        worksheet.Cell(i, 2).Value = v.Client.Name+" "+v.Client.Surname;
                        worksheet.Cell(i, 3).Value = v.Coach.Name;
                        worksheet.Cell(i, 4).Value = v.Date;
                        worksheet.Cell(i, 5).Value = v.TimeStart.Period+" - "+v.TimeStart.Period;
                        worksheet.Cell(i, 6).Value = v.SportsEquipment;
                        worksheet.Cell(i, 7).Value = v.CardNumber;
                        worksheet.Cell(i, 8).Value = v.CardDate;
                        worksheet.Cell(i, 9).Value = v.CardCVC;
                        worksheet.Cell(i, 10).Value = v.Sum;
                        i++;
                    }
                    worksheet.Columns().AdjustToContents();
                    using (var stream = new MemoryStream())
                    {
                        workbook.SaveAs(stream);
                        stream.Flush();
                                    return new FileContentResult(stream.ToArray(),"application/vnd.openxmlformats-officedocument.spreadsheetml.sheet")
                        {
                            FileDownloadName = "Корты.xlsx"
                        };
                    }
                }
            }
        }
```

- Контроллер «HomeController.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using course2.Models;
using System.Collections;
using System.Web.Security;

namespace course2.Controllers
{
    public class HomeController : Controller
    {
        Context db = new Context();
        const int courtsCount = 4;
        const int PriceWithoutCoach = 300;
        const int PriceWithCoach = 1000;
        public ActionResult Index()
        {
            return View("Index");
        }
        public ActionResult Login()
        {
            return View("Login");
        }
        public ActionResult CheckingLogin(string Password)
        {
            if(FormsAuthentication.Authenticate("admin", Password))
            {
```

```
            FormsAuthentication.SetAuthCookie("admin", false);
            return RedirectToAction("Index", "Admin");
        }
        return View("ErrorLogin");
    }
    public ActionResult Date()
    {
      ArrayList coaches = new ArrayList();
        foreach (var v in db.Coaches)
        {
            if ((v.Working) && (v.Name!="---УДАЛЁН---"))
                coaches.Add(v);
        }
        ViewBag.Coaches = coaches;
        return View("Date");
    }
    public ActionResult Time(string Date, string Coach)
    {
        if (Date == null)
            return View("Index");
        ArrayList periods = new ArrayList();
        ViewBag.Date = Date;
        ViewBag.Coach = Coach;
        if (Coach == "не нужны")
        {
            periods = checkingWithoutCoach(Date);
        }
        else
        {
            periods = checkingWithoutCoach(Date);
            periods = checkingWithCoach(Date, FindCoach(Coach), periods);
        }
        ArrayList periodsEnd = checkingHours(periods);
        ViewBag.Periods = periodsEnd;
        return View("Time");
    }
    public ActionResult OrderDetails(string Time, string Date, string Coach)
    {
        if (Date == null)
            return View("Index");
        ViewBag.Time = Time;
        ViewBag.Date = Date;
        ViewBag.Coach = Coach;
        int timeStartId =0, timeEndId=0;
        string timeStart = "", timeEnd = "";
        for(int i=0; i<5; i++)
        {
            timeStart += Time[i];
            timeEnd += Time[i + 8];
        }
        foreach(Time t in db.Times)
        {
            if(timeStart==t.Period)
            {
                timeStartId = t.Id;
                break;
            }
        }
        foreach (Time t in db.Times)
        {
            if (timeEnd == t.Period)
```

```
                {
                    timeEndId = t.Id;
                    break;
                }
            }
            ViewBag.TimeStartId = timeStartId;
            ViewBag.TimeEndId = timeEndId;
            return View("OrderDetails");
        }
         public ActionResult Payment(string Name, string Surname, string Phone, string CardNumber, string CardDate, string CardCVC, int? TimeStartId, int? TimeEndId, string Coach, string Date, string Time, bool SportsEquipment = false)
        {

            if (Name == null)
                return View("Index");
            int clientId = 0, count = 0;
            foreach (var v in db.Clients)
            {
                count++;
                if ((v.Name == Name) && (v.Surname == Surname) && (v.Phone == Phone))
                {
                    clientId = v.Id;
                    break;
                }
            }
            if (clientId == 0)
            {
                Client c = new Client{ Name = Name, Surname = Surname, Phone = Phone };
                db.Clients.Add(c);
                db.SaveChanges();
                clientId = c.Id;
            }
            string sportsequipment;
            if (SportsEquipment)
            {
                sportsequipment = "да";
            }
            else
            {
                sportsequipment = "нет";
            }
            int coachid = 0, price;
            if (Coach != "не нужны")
            {
                price = PriceWithCoach;
                coachid = FindCoach(Coach);
            }
            else
            {
                price = PriceWithoutCoach;
            }
            int sum = price * (TimeEndId.Value - TimeStartId.Value);
             Purchase p = new Purchase { ClientId = clientId, CoachId = coachid, Date = Date, SportsEquipment = sportsequipment, CardNumber = CardNumber, CardCVC = CardCVC, CardDate = CardDate, Sum = sum, TimeStartId = TimeStartId.Value, TimeEndId = TimeEndId.Value };
            ArrayList periods = new ArrayList(), periodsEnd = new ArrayList();
            if (Coach == "не нужны")
            {
                periods = checkingWithoutCoach(Date);
            }
```

```
                else
                {
                    periods = checkingWithoutCoach(Date);
                    periods = checkingWithCoach(Date, coachid, periods);
                }
                periodsEnd = checkingHours(periods);
                for (int i = 0; i < periodsEnd.Count; i++)
                {
                    if (Time == (string)periodsEnd[i])
                    {
                        string d = DateTime.Now.ToString("dd/MM/yyyy | HH:mm");
                        string s = sum.ToString();
                        db.Purchases.Add(p);
                        db.SaveChanges();
                          return RedirectToAction("OK", new { name = Name, surname = Surname, phone = Phone, date = Date, time = Time, sportsEquipment = SportsEquipment, coach = Coach, today =  d, sum = s});
                    }
                }
                return RedirectToAction("Error", new { name = Name});
            }
            public ActionResult Error(string Name)
            {
                if (Name == null)
                    return View("Index");
                 return View("Error");
                }
            }
                public ActionResult OK(string Name, string Surname, string Phone, string Date, string Time, bool SportsEquipment, string Coach, string Today, string Sum)
            {
                if (Name == null)
                    return View("Index");
                else
                {
                    ViewBag.Name = Name;
                    ViewBag.Surname = Surname;
                    ViewBag.Phone = Phone;
                    ViewBag.Date = Date;
                    ViewBag.Time = Time;
                    ViewBag.Today = Today;
                    ViewBag.Sum = Sum;
                    string sportsequipment;
                    if (SportsEquipment)
                    {
                        sportsequipment = "да";
                    }
                    else
                    {
                        sportsequipment = "нет";
                    }
                    ViewBag.SportsEquipment = sportsequipment;
                    ViewBag.Coach = Coach;
                    return View("OK");
                }
            }
            private ArrayList checkingHours(ArrayList periods)
            {
                ArrayList array = new ArrayList();
                for(int j=0; j<db.Times.Count(); j++)
                {
                    for (int i = 0; i < periods.Count - j - 1; i++)
```

```
                {
                    string temp1 = "", temp2 = "";
                    if ((int)periods[i + j] == (int)periods[i] + j)
                    {
                        temp1 = db.Times.Find((int)periods[i]).Period;
                        temp2 = db.Times.Find((int)periods[i] + j + 1).Period;
                    }
                    if ((temp1 != "") && (temp2 != ""))
                        array.Add(temp1 + " - " + temp2);
                }
            }
            return array;
        }
        private ArrayList checkingWithoutCoach(string date)
        {
            List<Purchase> purchases = new List<Purchase>();
            foreach(Purchase p in db.Purchases)
            {
                purchases.Add(p);
            }
            List<Time> times = new List<Time>();
            foreach (Time t in db.Times)
            {
                times.Add(t);
            }
            ArrayList array = new ArrayList();
            for (int i = 0; i < db.Times.Count(); i++)
            {
                int count = 0;
                for (int j = 0; j < db.Purchases.Count(); j++)
                {
                    if (count == courtsCount)
                        break;
                        if ((date == purchases[j].Date) && ((times[i].Id >= purchases[j].TimeStartId) && (times[i].Id <
purchases[j].TimeEndId)))
                        {
                            count++;
                        }
                }
                if (count < courtsCount)
                {
                    array.Add(times[i].Id);
                }
            }
            return array;
        }
        private ArrayList checkingWithCoach(string date, int coachid, ArrayList per)
        {
            ArrayList array = new ArrayList();
            for (int i=0; i<per.Count; i++)
            {
                bool check = false;
                foreach(Purchase p in db.Purchases)
                {
                    if((p.Date==date) && (p.CoachId==coachid))
                    {
                        check = true;
                        if(((int)per[i]<p.TimeStartId) || ((int)per[i] >= p.TimeEndId))
                        {
                            array.Add(per[i]);
                        }
```

```
                }
            }
            if(!check)
                array.Add(per[i]);
        }
        return array;
    }
    private int FindCoach(string coach)
    {
        int coachid = 0;
        foreach (Coach c in db.Coaches)
        {
            if (coach == c.Name)
            {
                coachid = c.Id;
            }
        }
        return coachid;
    }
  }
}
```

- Модель «Client.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace course2.Models
{
  public class Client
  {
    public int Id { get; set; }
    public string Name { get; set; }
    public string Surname { get; set; }
    public string Phone { get; set; }
  }
}
```

- Модель «Coach.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace course2.Models
{
  public class Coach
  {
    public int Id { get; set; }
    public string Name { get; set; }
    public bool Working { get; set; }
  }
}
```

- Модель «Context.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Data.Entity;
```

```
namespace course2.Models
{
   public class Context: DbContext
   {
      public Context() : base("Tennis") { }
      public DbSet<Client> Clients { get; set; }
      public DbSet<Purchase> Purchases { get; set; }
      public DbSet<Coach> Coaches { get; set; }
      public DbSet<Time> Times { get; set; }

   }
}
```

- Модель «Purchase.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace course2.Models
{
   public class Purchase
   {
      public int Id { get; set; }
      public int ClientId { get; set; }
      public int CoachId { get; set; }
      public string Date { get; set; }
      public string SportsEquipment { get; set; }
      public string CardCVC { get; set; }
      public string CardNumber { get; set; }
      public string CardDate { get; set; }
      public int TimeStartId { get; set; }
      public int TimeEndId { get; set; }
      public int Sum { get; set; }
      public Client Client { get; set; }
      public Coach Coach { get; set; }
      public Time TimeStart { get; set; }
      public Time TimeEnd { get; set; }
   }
}
```

- Модель «Time.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace course2.Models
{
   public class Time
   {
      public int Id { get; set; }
      public string Period { get; set; }
   }
}
```

- Таблица базы данных «Clients»

```
CREATE TABLE [dbo].[Clients] (
   [Id]     INT           IDENTITY (1, 1) NOT NULL,
   [Name]    NVARCHAR (MAX) NOT NULL,
```

```
    [Phone]   NCHAR (16)     NOT NULL,
    [Surname] NVARCHAR (MAX) NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```

- Таблица базы данных «Coaches»

```
CREATE TABLE [dbo].[Coaches] (
    [Id]      INT           IDENTITY (1, 1) NOT NULL,
    [Name]    NVARCHAR (MAX) NOT NULL,
    [Working] BIT           DEFAULT ('True') NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```

- Таблица базы данных «Purchases»

```
CREATE TABLE [dbo].[Times] (
    [Id]     INT       NOT NULL,
    [Period] NCHAR (5) NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```

- Таблица базы данных «Times»

```
CREATE TABLE [dbo].[Purchases] (
    [Id]             INT       IDENTITY (1, 1) NOT NULL,
    [ClientId]       INT       DEFAULT ((26)) NOT NULL,
    [CoachId]        INT       DEFAULT ((24)) NOT NULL,
    [Date]           NCHAR (10) NOT NULL,
    [TimeStartId]    INT       NOT NULL,
    [SportsEquipment] NCHAR (5)  NOT NULL,
    [CardCVC]        NCHAR (3)  NOT NULL,
    [CardNumber]     NCHAR (19) NOT NULL,
    [CardDate]       NCHAR (5)  NOT NULL,
    [TimeEndId]      INT       NOT NULL,
    [Sum]            INT       NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC),
        CONSTRAINT [FK_Purchases_Clients] FOREIGN KEY ([ClientId]) REFERENCES [dbo].[Clients] ([Id]) ON DELETE SET DEFAULT,
      CONSTRAINT [FK_Purchases_Coaches] FOREIGN KEY ([CoachId]) REFERENCES [dbo].[Coaches] ([Id]) ON DELETE SET DEFAULT,
    CONSTRAINT [FK_Purchases_Times] FOREIGN KEY ([TimeStartId]) REFERENCES [dbo].[Times] ([Id]),
    CONSTRAINT [FK_Purchases_ToTimes] FOREIGN KEY ([TimeEndId]) REFERENCES [dbo].[Times] ([Id])
);
```

Приложение 2 Результаты работы программы

Рис. 44: Главная страница, раздел «Главное»

Рис. 45: Главная страница, раздел «О нас»

Рис. 46: Главная страница, раздел «Тренерский состав»

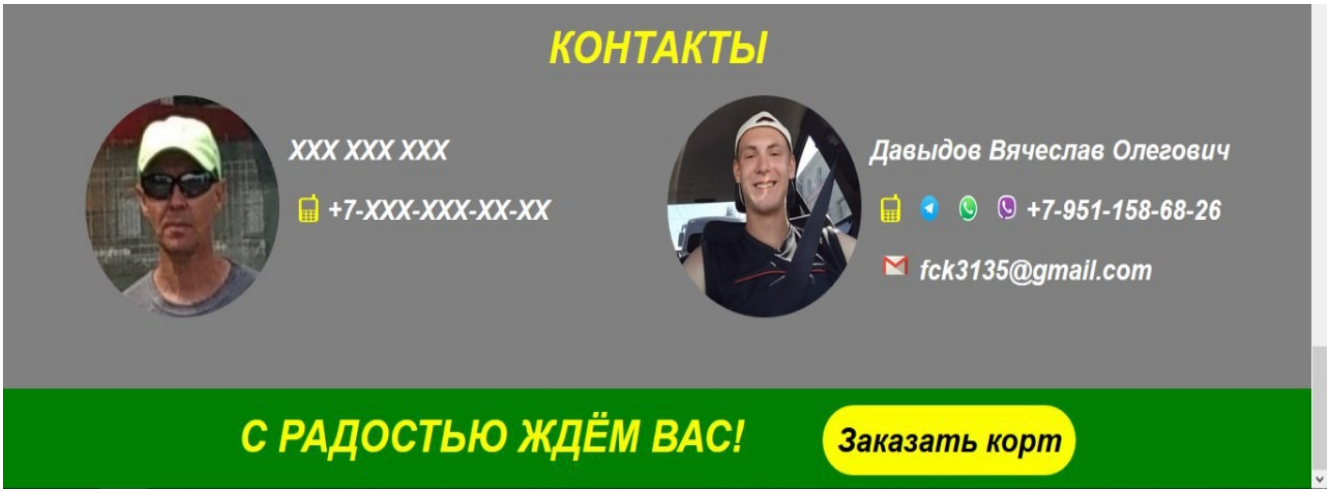

Рис. 47: Главная страница, раздел «Контакты»

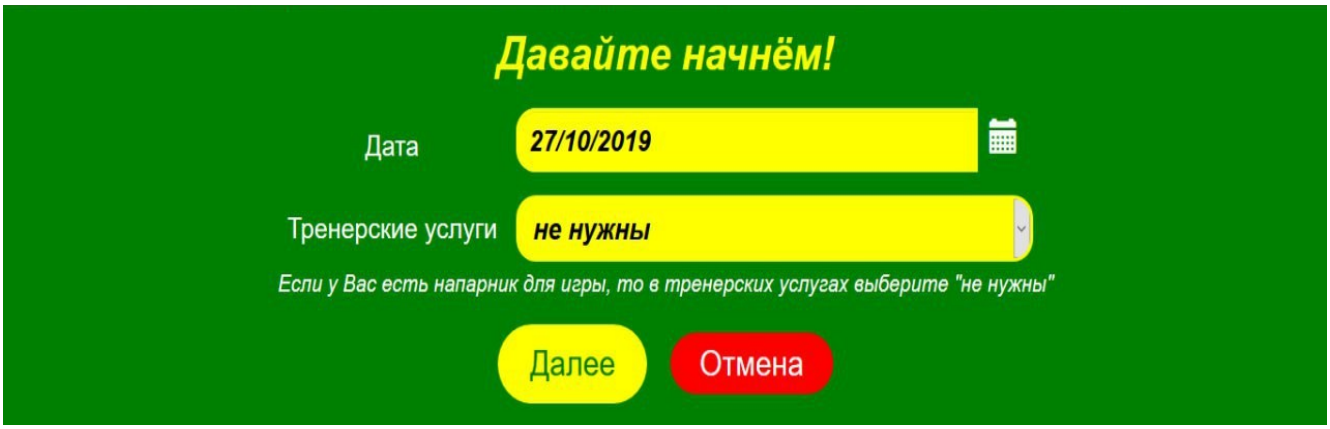

Рис. 48: Страница выбора даты и тренера

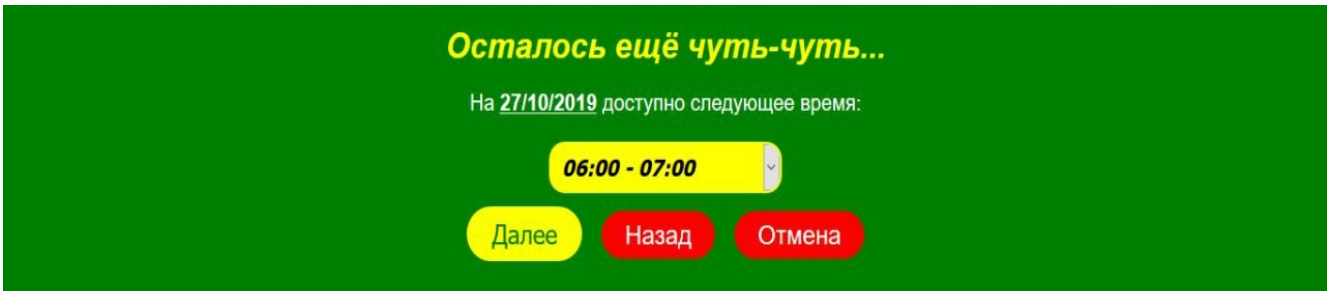

Рис. 49: Страница выбора времени

Рис. 50: Страница оплаты и ввода персональных данных

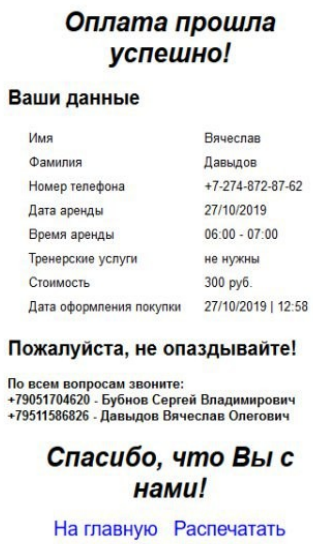

Рис. 51: Страница подтверждения оплаты и забронированного времени

Рис. 52: Кабинет администратора

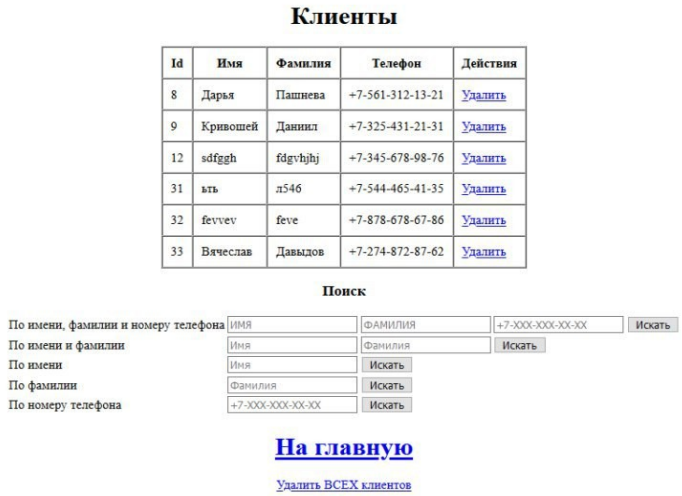

Рис. 53: Страница с данными о клиентах

Тренера

Id	ФИО	Действия	
0	не нужны	Приостановить работу	Удалить
2	Бубнов Сергей Владимирович	Приостановить работу	Удалить
3	Виняр Павел Андреевич	Продолжить работу	Удалить
4	Виняр Елизавета Юрьевна	Продолжить работу	Удалить
5	Кобзарь Егор Олегович	Приостановить работу	Удалить
6	Луценко Нелли Сергеевна	Продолжить работу	Удалить
7	Мартыненко Михаил Валентинович	Приостановить работу	Удалить
8	Рябков Павел Александрович	Приостановить работу	Удалить
23	Жукова Мария Андреевна	Приостановить работу	Удалить
27	Давыдов Вячеслав Олегович	Приостановить работу	Удалить

Добавить тренера ФИО тренера Добавить

На главную

Удалить ВСЕХ тренеров

Рис. 54: Страница с данными о тренерах

Если Вы зашли на эту страницу случайно, то вернитесь на главную!

На главную

Только для администратора!

Пароль ВОЙТИ

Рис. 55: Страница входа в администраторский кабинет

Аренда

Id	Клиент	Тренерские услуги	Дата	Время	Спортивное оборудование	Номер карты	Срок действия карты	Проверочный код карты	Сумма	Действия
9	---УДАЛЁН---	Бубнов Сергей Владимирович	20/07/2019	06:00 - 21:00	да	3333 3333 3333 3333	22/22	113	15000 рублей	Удалить
12	Кривошей Даниил	не нужны	21/07/2019	07:00 - 09:00	нет	2341 3513 2412 4321	23/15	241	600 рублей	Удалить
35	ьть л546	не нужны	21/09/2019	07:00 - 13:00	да	1516 1315 1561 3213	65/32	541	1800 рублей	Удалить
36	fevvev feve	не нужны	20/09/2019	06:00 - 07:00	нет	2523 3563 4643 7337	34/73	777	300 рублей	Удалить
37	Вячеслав Давыдов	не нужны	27/10/2019	06:00 - 07:00	нет	3252 6554 5353 4525	34/53	444	300 рублей	Удалить

Поиск

По имени и фамилии клиента Имя Фамилия Искать
По ФИО тренера Фамилия Имя Отчество Искать
По дате XX/XX/XXXX Искать
По времени XX.XX - XX.XX Искать
По дате и времени XX.XX - XX.XX XX/XX/XXXX Искать
По номеру карты XXXX XXXX XXXX XXXX Искать
По имени и фамилии клиента и дате Имя Фамилия XX/XX/XXXX Искать
По ФИО тренера и дате Фамилия Имя Отчество XX/XX/XXXX Искать

Загрузить данные на компьютер

На главную

Удалить ВСЕ аренды

Рис. 56: Страница с данными о бронированиях