**Оглавление**

[Введение 3](#_Toc38198942)

[1 Постановка задачи и обзор аналогов 5](#_Toc38198943)

[1.1 Основные технические требования 5](#_Toc38198946)

[1.2 Обзор аналогов 6](#_Toc38198947)

[1.2.1 Приложение LoveTour.by 6](#_Toc38198948)

[1.2.2 Приложение Magellan.by 6](#_Toc38198949)

[1.2.3 Приложение PeopleTravel.by 7](#_Toc38198950)

[1.3 Обзор средств разработки 8](#_Toc38198951)

[1.3.1 Выбор языка программирования 8](#_Toc38198952)

[1.3.2 Выбор средства программирования 9](#_Toc38198953)

[1.3.3 Выбор платформы разработки 10](#_Toc38198954)

[1.4 Обзор используемых технологий и библиотек 11](#_Toc38198955)

[1.4.1 Технология Entity Framework Core для решения задач объектно-реляционного отображения 11](#_Toc38198956)

[1.4.2 Библиотека React для разработки пользовательских интерфейсов 12](#_Toc38198957)

[1.5 Вывод по разделу 13](#_Toc38198958)

[2 Проектирование 14](#_Toc38198959)

[2.1 Архитектура приложения 14](#_Toc38198960)

[2.2 Проектирование базы данных 15](#_Toc38198961)

[2.3 Вывод по разделу 18](#_Toc38198962)

[3 Программная реализация 19](#_Toc38198963)

[Заключение 21](#_Toc38198964)

[Список используемых источников 22](#_Toc38198965)

[ПРИЛОЖЕНИЕ А 23](#_Toc38198966)

Введение

«СООО Системные технологии» **–** компания-разработчик программного обеспечения, специализирующаяся на разработке собственных программных продуктов для автоматизации деятельности банков и предприятий. Компания входит в топ-10 крупнейших производителей ПО по версии белорусского Парка высоких технологий[1].

В линейку продуктов компании входит программное обеспечение для таких отраслей экономики как банковская деятельность, страхование, лизинг и промышленность. Они включают в себя [системы оказания удаленных электронных услуг](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%81%D1%82%D0%B0%D0%BD%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D0%B1%D0%B0%D0%BD%D0%BA%D0%BE%D0%B2%D1%81%D0%BA%D0%BE%D0%B5_%D0%BE%D0%B1%D1%81%D0%BB%D1%83%D0%B6%D0%B8%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), системы кредитного документооборота, автоматизации процессов страхования, нефтегазообеспечения, системы класса [BI](https://ru.wikipedia.org/wiki/Business_Intelligence), [BPM](https://ru.wikipedia.org/wiki/BPM_(%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BA%D0%BE%D0%BD%D1%86%D0%B5%D0%BF%D1%86%D0%B8%D1%8F)) и [CRM](https://ru.wikipedia.org/wiki/CRM), финансовые сервисы, заказную разработку.

Сегодня более 90 % банков Беларуси пользуются программными решениями СООО «Системные технологии»[2]. Клиентами компании выступают ведущие белорусские и зарубежные банки и предприятия, такие как АСБ «Беларусбанк», [ОАО](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BE%D1%80%D0%B1%D0%B0%D0%BD%D0%BA) «[Приорбанк](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BE%D1%80%D0%B1%D0%B0%D0%BD%D0%BA" \o "Приорбанк)», [АО «Райффайзен Банк Аваль](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B9%D1%84%D1%84%D0%B0%D0%B9%D0%B7%D0%B5%D0%BD_%D0%B1%D0%B0%D0%BD%D0%BA_%D0%90%D0%B2%D0%B0%D0%BB%D1%8C)», Центральный банк Азербайджана, ОАО «АЗЕРИГАЗБАНК», ЗАО «Bank Standard» КБ, ОАО «Белвнешэкономбанк», ОАО «БПС-Банк», ОАО «Белагропромбанк», ОАО «Белинвестбанк», РУП "Производственное объединение «Белоруснефть», ОАО «Криница», СООО «Райффайзен-Лизинг» и другие[3].

Организационная структура управления представляет собой совокупность органов управления и существующих между ними должностных и информационных связей. Структура управления компании представлена на рисунке 1.1.

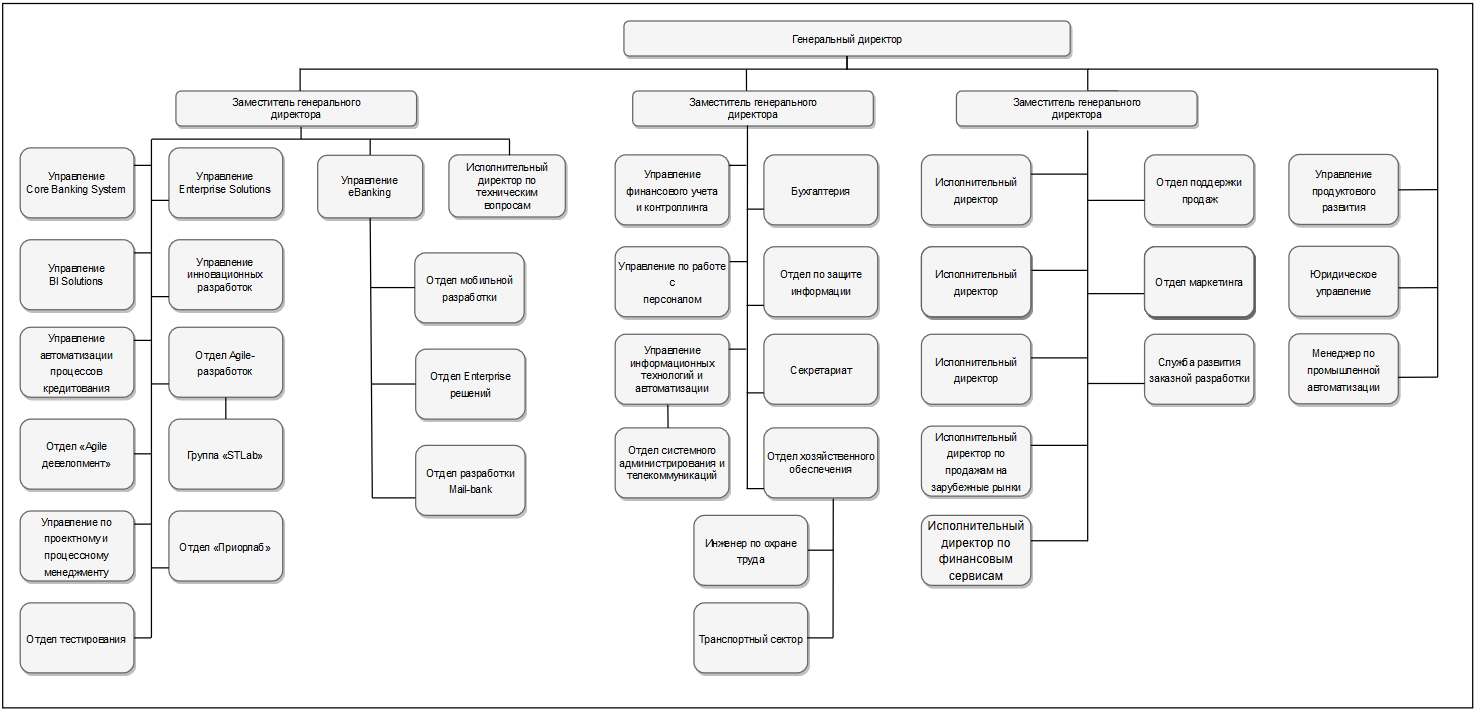


Рисунок 1.1 – Организационная структура СООО «Системные технологии»

Как видно, организационная структура СООО «Системные технологии» по виду передаваемых полномочий является линейной – Во главе каждого подразделения находится руководитель, наделенный всеми полномочиями и осуществляющий единоличное руководство подчиненными ему работниками, сосредотачивающий в своих руках все функции управления. Сам, руководитель находится в непосредственном подчинении руководителя высшего уровня

Главным органом управления предприятия, и соответственно первым уровнем управления, является генеральный директор. Он действует от имени предприятия, представляет его интересы во всех организациях, государственных органах, утверждает штатное расписание, издает приказы по личному составу и основной деятельности предприятия, дает указания и распоряжения, обязательные для исполнения всеми работниками предприятия.

Организационная структура управления предприятием характеризуется большим диапазоном управления и небольшим количеством уровней управления. Полномочия и вся информация сосредотачивается в руках руководителя каждого уровня. Это гарантирует быстроту принятия управленческих решений и возможность легкого контроля за работой исполнителей. К преимуществам данной линейной структуры можно также отнести согласованность действий исполнителей, четкая система взаимных связей между руководителем и подчиненным. Главный недостаток такой структуры – высокий уровень требований к компетентности руководителей при выполнении любых управленческих функций, перегрузка менеджеров высшего уровня, огромное количество информации, множественность контактов с подчиненными и вышестоящими.

# **1 Постановка задачи и обзор аналогов**

## Роль туризма в мировой практике постоянно возрастает. Будучи одной из крупнейших, высокодоходных и наиболее динамичных отраслей, уступая по доходности лишь добыче и переработке нефти, туризм, по данным ЮНВТО, обеспечивает 10% оборота производственно-сервисного рынка планеты. На сферу туризма приходится 6% мирового ВНП, 7% мировых инвестиций, каждое 16-е рабочее место, 11% мировых потребительских расходов, 5% всех налоговых поступлений.

## Следовательно, туризм, испытывающий на себе влияние ряда внешних факторов, сам способен оказывать значительное воздействие на экономическую, политическую, экологическую, социально-культурную ситуацию тех стран и регионов, где он развивается. Причем это воздействие может носить как позитивный, так и негативный характер.

В последнее время в нашей стране стал необычайно актуален вопрос туризма. Связано это с улучшением благосостояния граждан, с расширением открытости границ и стремительным развитием туристического бизнеса в целом. В любой момент человек может обратиться в соответствующую фирму или компанию и заказать тур для поездки в далекие экзотические страны.

Выбор курортов сегодня просто огромен – от далеких таиландских и арабских, до украинских, российских, чешских и других. Вопрос, куда же отправиться, сейчас стоит очень остро, так как в большом количестве предложений легко можно запутаться.

Моё приложение поможет забронировать наиболее подходящие место для туризма. На сайте собрано достаточное количество информации, касающейся различных видов туризма, мест, где можно прекрасно отдохнуть, а также другие данные, обладание которыми сильно поможет Вам, если Вы соберетесь куда-либо отправиться.

Основной задачей дипломного проекта является разработка веб-приложения «Туристическое агентство». Данное приложение должно предоставлять удобный интерфейс пользователю для осуществления online-бронирования туристических билетов.

## 1.1 Основные технические требования

Целью данного дипломного проекта является разработка программного средства, представляющего собой web-приложение. Обеспечивающие удобное бронирование билетов для пассажиров и работу для администраторов, необходимо реализовать серверную и клиентскую часть приложения.

Frontend часть приложения должна:

* обладать удобным интерфейсом;
* предоставлять информацию о турах;
* быть легко поддерживаемым.

Backend часть приложения должна:

* обеспечивать хранение данных;
* основываться на REST-архитектуре;
* обеспечивать легкую поддержку.

## Обзор аналогов

### 1.2.1 Приложение LoveTour.by

В качестве одного из прототипов приложения был выбран сайт LoveTour.by.

Интерфейс приложения представлен на рисунке 1.1[4].

Данный сайт предназначен для ознакомления с возможными турами для последующего бронирования посредством мобильной связи. У него есть возможность выбора автобусного тура или авиатура. В категории спец. предложения можно ознакомиться с уникальными предложениями от туроператора.

Возможности онлайн бронирования данный сайт не предоставляет.

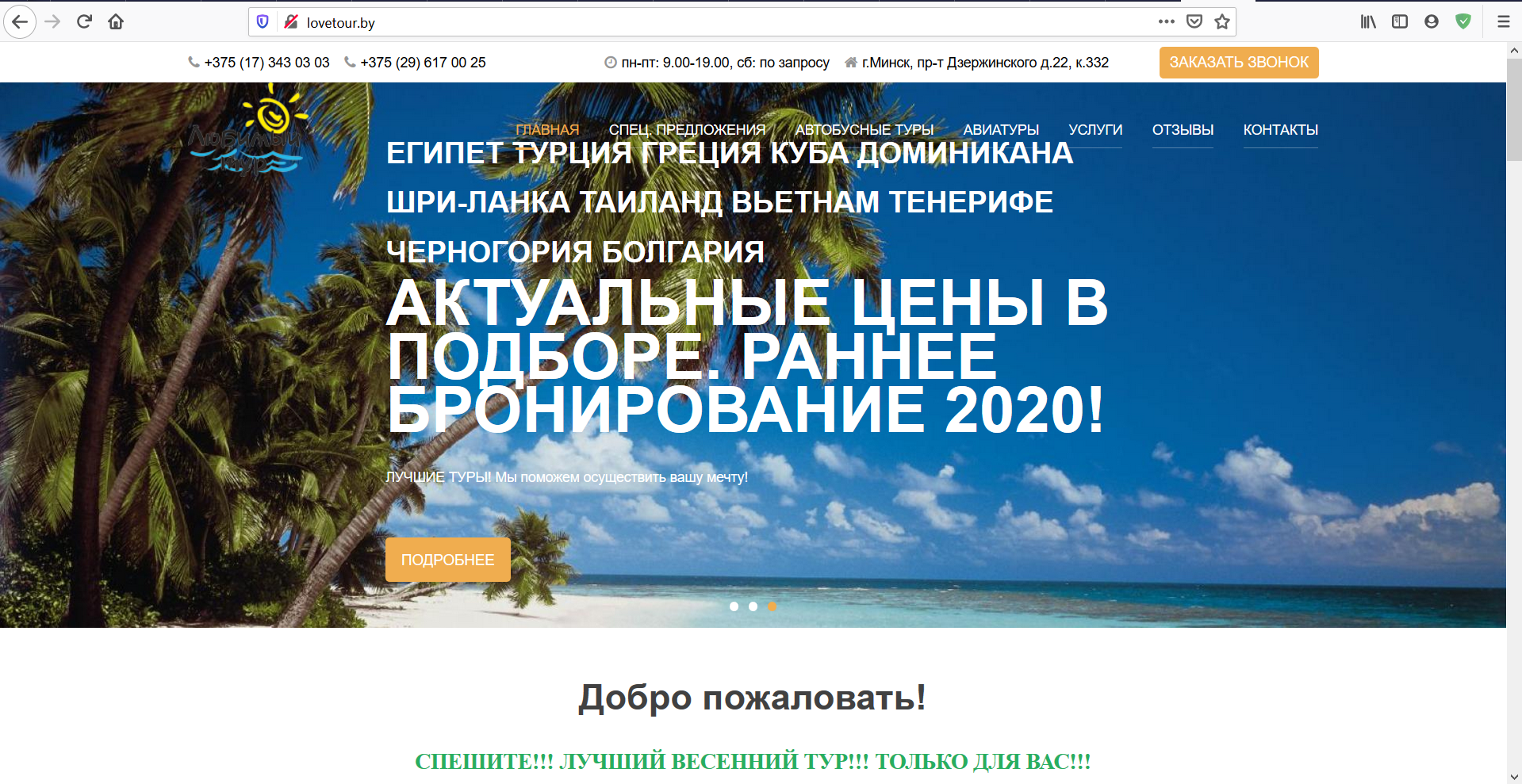


Рисунок 1.1 – Сайт LoveTour.by

Сайт предоставляет удобный интерфейс для взаимодействия с пользователем.

Возможность регистрации аккаунтов не предусмотрена.

### 1.2.2 Приложение Magellan.by

Интерфейс приложения представлен на рисунке 1.2 [5]

Возможности:

* поиск тура между выбранными странами с выбором периода вылета, количеством взрослых и детей, периодом ночей;
* возможность добавления в избранное понравившихся туров;
* наличие online-консультанта;
* услуга «Перезвони мне». Вы указываете в специальной форме свое имя и номер телефона.

Существенным недостатком является не адаптивность данного сайта. Так как в современном мире количество запросов с мобильных устройств преобладает над количеством запросов с персональных компьютеров, очень важно правильное отображение сайта на различных устройствах, динамически подстраивающийся под заданные размеры окна браузера.

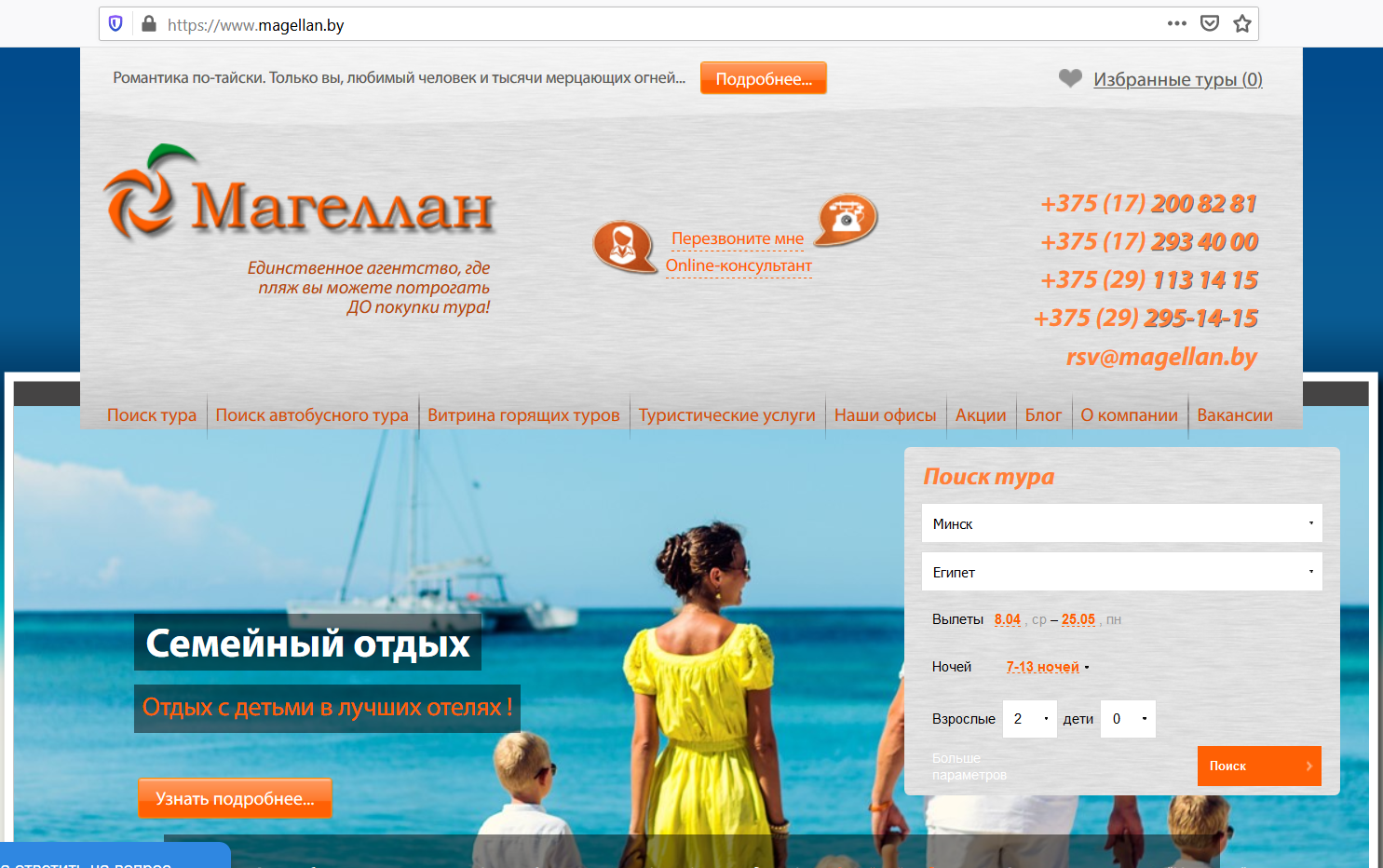


Рисунок 1.2 – Сайт Magellan.by

Приложение имеет ряд ошибок и недостатков.

### 1.2.3 Приложение PeopleTravel.by

Судя по отзывам на сайте, данный сервис имеет хорошую репутацию среди пользователей. Данный сайт предоставляет удобный пользовательский интерфейс и представлен на рисунке 1.3 [6].



Рисунок 1.3 – Сайт PeopleTravel.by

Возможности:

* оставить заявку на подбор тура;
* возможность выбора экскурсионного, авиа или круизного тура;
* наличие туристического календаря;
* есть возможность сравнения туров;
* найти ответы на часто задаваемые вопросы.

Существенным недостатком данного приложения является отсутствие возможности бронирования тура онлайн. Присутствуют незначительные баги с отображением информации.

### Обзор средств разработки

В данном разделе будут рассмотрены различные библиотеки, технологии и подходы, которые будут использованы в разработке приложения.

### Выбор языка программирования

Для разработки программного средства был выбран язык программирования C#. В этом подразделе указаны основные возможности, достоинства и недостатки языка, а также его сравнение с другими языками программирования.

C# – объектно-ориентированный язык программирования. Разработан в 1998-2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота[4] как язык разработки приложений для платформы Microsoft .NET Framework. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

C# разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов C#, которая отражает FCL. Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем. (Однако эта закономерность была нарушена с выходом C# 3.0, представляющим собой расширения языка, не опирающиеся на расширения платформы .NET.) CLR предоставляет C#, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования. Например, сборка мусора не реализована в самом C#, а производится CLR для программ, написанных на C# точно так же, как это делается для программ на VB.NET, J# и др.[7]

Свой синтаксис C# во многом унаследовал от C++ и Java. Но вместе с тем он является во многом новаторским – атрибуты, делегаты и события, вписанные в общую идеологию языка, прочно заняли место в сердцах .NET - разработчиков. Их введение позволило применять принципиально новые приемы программирования

### Выбор средства программирования

Интегрированные среды разработки, среды разработки (Integrated Development Environment – IDE) – совокупность программных средств, с помощью которых программистами производится написание, корректировка, преобразование в машинные коды, наладка и запуск различного программного обеспечения. Интегрированные среды разработки обычно содержат:

* текстовый редактор, предназначенный для ввода, подсвечивания и корректировки текста программы;
* компилятор, с помощью которого программа переводится с языка, на котором она написана, в машинные коды;
* средства отладки и запуска программ;
* общие библиотеки, которые обычно содержат многократно используемые элементы программ;
* справочную систему и другие элементы.

Интегрированная среда разработки Microsoft Visual Studio – это набор инструментов для создания программного обеспечения: от планирования до разработки пользовательского интерфейса, написания кода, тестирования, отладки, анализа качества кода и производительности, развертывания в средах клиентов и сбора данных телеметрии по использованию. Эти инструменты предназначены для максимально эффективной совместной работы. Все они доступны в интегрированной среде разработки Visual Studio.

Visual Studio можно использовать для создания различных типов приложений, от простых приложений для магазина и игр для мобильных клиентов до больших и сложных систем, обслуживающих предприятия и центры обработки данных. Вы можете создавать:

* приложения и игры, которые выполняются не только на платформе Windows, но и на Android и iOS;
* веб-сайты и веб-службы на основе ASP.NET, JQuery, AngularJS и других популярных платформ;
* приложения для самых разных платформ и устройств, включая, но не ограничиваясь: Office, Sharepoint, Hololens, Kinect и "Интернета вещей";
* игры и графические приложения для разных устройств Windows, включая Xbox, с поддержкой DirectX.

По умолчанию Visual Studio обеспечивает поддержку C#, C и C++, JavaScript, F# и Visual Basic. Visual Studio хорошо работает и интегрируется со сторонними приложениями, например, Unity и Apache Cordova, с помощью расширений и средств Visual Studio для Unity и инструментов Visual Studio для Apache Cordova соответственно.

Microsoft Visual Studio – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых компанией Microsoft.

Интерфейс программы состоит:

* меню файл, правка, поиск и тд;
* область выбора файлов проекта для редактирования;
* окна, появляющиеся в режиме отладки: Call Stack, Output, Breakpoints, Watch windows.
* список открытых файлов;
* область выбора файлов проекта для редактирования;
* область написания кода.

Вышеперечисленные возможности этой среды – плагины, удобный интерфейс, хороший отладчик, удобный и быстрый редактор кода, делают Visual Studio одним из лучших средств для разработки программного обеспечения. При этом стоит отметить, что я не перечислил и сотой доли тех возможностей, которые Visual Studio в себя включает.

### Выбор платформы разработки

.NET Core – это открытая универсальная модульная платформа разработки, которая поддерживается корпорацией Майкрософт и сообществом .NET на сайте GitHub. Она является кроссплатформенной, поддерживает Windows, Mac OS и Linux и может использоваться на устройствах, в облаке, во внедренных системах и в сценариях IoT (Интернета вещей). В её основе лежат технологии .NET Framework и Silverlight. Она оптимизирована для мобильных и серверных рабочих нагрузок, поскольку обеспечивает поддержку самодостаточных развёртываний XCOPY[8].

.NET Core обладает следующими характеристиками:

* **кроссплатформенность;**
* согласованность между архитектурами;
* **открытый исходный код;**
* **гибкая разработка;**
* **программы командной строки;**
* **совместимость;**
* **поддержка от Microsoft.**

.NET Core состоит из перечисленных ниже компонентов:

* среда выполнения .NET Core – предоставляет систему типов, функции загрузки сборок, сборщик мусора, собственные функции взаимодействия и другие базовые службы. Библиотеки платформы .NET Core предоставляют примитивные типы данных, типы компоновки приложений и базовые служебные программы;
* среда выполнения ASP.NET Core – это платформа для создания современных облачных приложений, подключенных к сети Интернет: веб-приложений, приложений IoT, серверной части мобильных решений и многого другого;
* .NET Core CLI и компиляторы языков (Roslyn и F#) реализуют возможности разработки .NET Core.

## 1.4 Обзор используемых технологий и библиотек

### 1.4.1 Технология Entity Framework Core для решения задач объектно-реляционного отображения

Entity Framework Core (EF Core) представляет собой объектно-ориентированную, легковесную и расширяемую технологию от компании Microsoft для доступа к данным. EF Core является ORM-инструментом (object-relational mapping – отображения данных на реальные объекты). То есть EF Core позволяет работать c базами данных, но представляет собой более высокий уровень абстракции: EF Core позволяет абстрагироваться от самой базы данных и ее таблиц и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работаем с объектами.

Entity Framework Core поддерживает множество различных систем баз данных. Таким образом, мы можем через EF Core работать с любой СУБД, если для нее имеется нужный провайдер.

По умолчанию на данный момент Microsoft предоставляет ряд встроенных провайдеров: для работы с MS SQL Server, для SQLite, для PostgreSQL. Также имеются провайдеры от сторонних поставщиков, например, для MySQL.

Также стоит отметить, что EF Core предоставляет универсальный API для работы с данными. И если, к примеру, мы решим сменить целевую СУБД, то основные изменения в проекте будут касаться прежде всего конфигурации и настройки подключения к соответствующим провайдерам. А код, который непосредственно работает с данными, получает данные, добавляет их в БД и т.д., останется прежним.

Как технология доступа к данным Entity Framework Core может использоваться на различных платформах стека .NET. Это и стандартные платформы типа Windows Forms, консольные приложения, WPF, UWP и ASP.NET Core. При этом кроссплатформенная природа EF Core позволяет задействовать ее не только на ОС Windows, но и на Linux и Mac OS X.

Центральной концепцией Entity Framework является понятие сущности или entity. Сущность определяет набор данных, которые связаны с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их коллекциями.

Любая сущность, как и любой объект из реального мира, обладает рядом свойств. Например, если сущность описывает человека, то мы можем выделить такие свойства, как имя, фамилия, рост, возраст. Свойства необязательно представляют простые данные типа int или string, но могут также представлять и более комплексные типы данных. И у каждой сущности может быть одно или несколько свойств, которые будут отличать эту сущность от других и будут уникально определять эту сущность. Подобные свойства называют ключами.

При этом сущности могут быть связаны ассоциативной связью один-ко-многим, один-ко-одному и многие-ко-многим, подобно тому, как в реальной базе данных происходит связь через внешние ключи.

Отличительной чертой Entity Framework Core, как технологии ORM, является использование запросов LINQ для выборки данных из БД. С помощью LINQ мы можем создавать различные запросы на выборку объектов, в том числе связанных различными ассоциативными связями. А Entity Framework при выполнении запроса транслирует выражения LINQ в выражения, понятные для конкретной СУБД (как правило, в выражения SQL).

### 1.4.2 Библиотека React для разработки пользовательских интерфейсов

React – популярная библиотека, используемая для создания пользовательских интерфейсов. Она была создана в Facebook с целью решить ряд проблем, связанных с крупномасштабными сайтами, управляемыми данными.

Главная задача React – обеспечение вывода на экран того, что можно видеть на веб-страницах. React значительно облегчает создание интерфейсов благодаря разбиению каждой страницы на небольшие фрагменты. Данные фрагменты называются компонентами.

Компонент React – это участок кода, который представляет часть веб-страницы. Каждый компонент – это JavaScript-функция, которая возвращает кусок кода, представляющего фрагмент страницы.

Для формирования страницы мы вызываем эти функции в определённом порядке, собираем вместе результаты вызовов и показываем их пользователю.

Когда мы пытаемся манипулировать html-элементами с помощью JavaScript, то мы можем столкнуться со снижением производительности, особенно при изменении большого количества элементов. А операции над элементами могут занять некоторое время, что неизбежно скажется на пользовательском опыте. Для решения проблемы производительности появилась концепция виртуального DOM.

Виртуальный DOM (VDOM) – это концепция программирования, в которой идеальное или «виртуальное» представление пользовательского интерфейса хранится в памяти и синхронизируется с «настоящим» DOM с помощью библиотеки, такой как ReactDOM[9].

Данный процесс называется согласованием. Данный подход обеспечивается декларативным React API: вы говорите React, в каком состоянии должен находиться UI, и он гарантирует, что DOM соответствует этому состоянию. Это абстрагирует манипуляции с атрибутами, обработку событий и ручное обновление DOM, которые в противном случае вам пришлось бы использовать для создания приложения.

Еще одна особенность React – использование JSX. JSX представляет комбинацию кода JavaScript и XML и предоставляет простой и интуитивно понятный способ для определения кода визуального интерфейса.

Стоит отметить, что React развивается как открытый проект, и весь исходный код библиотеки доступен на веб-сервисе GitHub, который предназначен для хостинга IT-проектов.

## 1.5 Вывод по разделу

Развитие туризма является одним из приоритетных направлений социально-экономического развития Республики Беларусь.

В связи с этим было решено разработать веб-приложение, предоставляющие удобное бронирование билетов для пользователей.

Клиентская часть приложения должна: обладать удобным интерфейсом, предоставлять информацию о турах, быть легко поддерживаемым.

Серверная часть приложения должна: обеспечивать хранение данных, основываться на REST-архитектуре, обеспечивать легкую поддержку [10].

В качестве аналогов разрабатываемого приложения были рассмотрены следующие веб-приложения: LoveTour.by, Magellan.by и PeopleTravel.by. Были выбраны следующие средства разработки: Visual Studio 2019 для разработки серверной части приложения, Visual Studio Code для клиетской части приложения.

В процессе разработки были использованы:

* свободно-распространяемый кроссплатформенный фреймворк для создания веб-приложений ASP.NET Core;
* ORM технология доступа к данным Entity Framework Core;
* JavaScript-библиотека для разработки пользовательских интерфейсов React.

# **Проектирование**

Разработка архитектуры проекта – одна из важнейших задач в процессе работы над приложением, так как в зависимости от неё определяется уровень зависимости компонентов приложения и насколько легко будут расширяемы её составные части.

## 2.1 Архитектура приложения

Для реализации приложения была использована N-Layer архитектура. Ее схематическое представление продемонстрировано на рисунке 2.1.

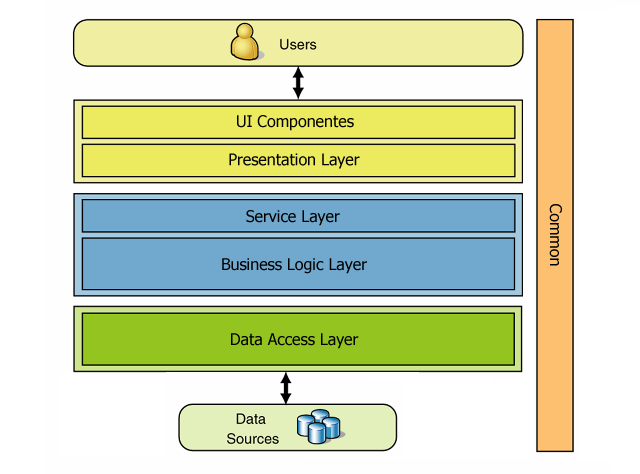


Рисунок 2.1 – Схематическое представление N-Layer архитектуры

Data Access Layer (уровень доступа к данным) содержит сущности, соответствующие таблицам базы данных, а также схемы взаимодействия друг с другом.

Business Logic Layer (уровень бизнес-логики) при тесном взаимодействии с Service Layer (уровень сервисов) являются посредниками между сущностями базы данных и моделями для UI, сочетая в себе сервисы и классы, реализующие основной функционал всего разрабатываемого приложения.

Presentation Layer (уровень представления), поддерживаемый UI Components (User Interface Components – компоненты пользовательского интерфейса), представляет собой генерирующуюся на сервере разметку или получаемые с уровня бизнес-логики данные для REST API в формате JSON, преобразующиеся впоследствии в разметку на стороне клиента.

Уровень представления в данном проекте представлен React приложением.

## 2.2 Проектирование базы данных

База данных является важнейшей частью приложения, так как хранит в себе все данные. К проектированию базы данных требуется подходить с умом, так как в дальнейшем вносить изменения в структуру, будет сложно. Для взаимодействия с базой данных приложению необходим создавать подключение к базе данных.

Использование ORM позволит ускорить процесс разработки, улучшить читаемость кода и избавиться от «чистых» SQL-запросов в коде приложения. Исключение использования «чистых» SQL-запросов из приложения позволяет повысить безопасность приложения. Это достигается за счёт того, что исключает возможность внедрения SQL-инъекции в запущенное приложение. Под SQL-инъекцией понимается возможность получения сервером «чистого» SQL-запроса, который, например, может удалить все данные из таблицы.

Для взаимодействия с базой данных было решено выбрать ORM Entity Framework Core.

Целью Entity Framework Core является освобождение разработчика от значительного объёма сравнительно низкоуровневого программирования при работе в объектно-ориентированных средствах в реляционной базе данных. Разработчик может использовать Entity Framework Core как в процессе проектирования системы классов и таблиц «с нуля», так и для работы с уже существующей [базой данных.](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)

Изначально таблицы проектируемой базы данных будут пустыми, что не позволит пользователям незамедлительно начать работу. Чтобы этого избежать, требуется произвести первоначальную инициализацию.

Для взаимодействия с базой данных для ORM требуется описать сущности базы данных. Каждая сущность представляет собой одну таблицу базы данных, является её отражением на приложение, тем самым позволяя взаимодействовать с данными, хранимыми в базе данных.

На рисунке 2.3 представлен список сущностей приложения, которые располагаются в пакете «Entities» проекта TravelAgency.DAL.

Все сущности, представленные ниже, будут задействованы в приложении.

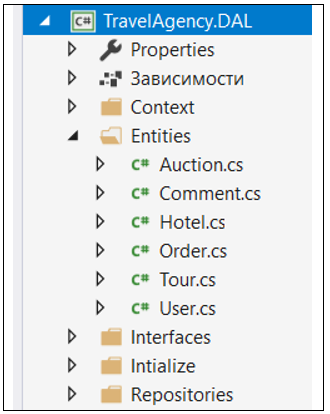


Рисунок 2.3 – Список моделей

В приложении А приведена схема базы данных, где видны все связи между различными таблицами.

Таблица Tours содержит информацию о турах, созданных администратором. Перечень полей таблицы Tours приведен в таблице 2.1.

Таблица 2.1 – Описание полей таблицы Tours

|  |  |
| --- | --- |
| Поле | Типы данных, ограничение целостности |
| TourId | Int, identity primary key |
| TourName | Nvarchar(70) not null |
| Country | Nvarchar(70) not null |
| Duration | Int not null |
| Cost | Int not null |
| ImagePath | Nvarchar(max) null |
| DateStart | Datetime2(7) not null |

Свойство TourId отвечает за идентификацию тура.

Свойство TourName хранит наименование тура.

Свойство Country хранит название страны тура.

Свойство Duration хранит информацию о количестве дней тура.

Свойство Cost хранит в себе информацию о цене предоставляемого тура.

Свойство ImagePath может содержать закодированное изображение тура в формате base64.

Свойство DateStart содержит информацию о начале тура.

Таблица Hotels содержит информацию о отелях, созданных администратором. Для одного тура у администратора имеется возможность создания нескольких отелей. Перечень полей таблицы Hotel приведен в таблице 2.2.

Таблица 2.2 – Описание полей таблицы Hotels

|  |  |
| --- | --- |
| Поле | Типы данных, ограничение целостности |
| HotelId | Int, identity primary key |
| HotelName | Nvarchar(50) not null |
| Cost | Int not null |
| Class | Int not null |
| Description | Nvarchar(500) not null |
| ImagePath | Nvarchar(max) null |
| TourId | Int, foreign key not null |

Свойство HotelId отвечает за идентификацию отеля.

Свойство HotelName отвечает хранит наименование отеля.

Свойство Cost хранит в себе информацию о цене отеля.

Свойство Class определяет количество звезд отеля.

Свойство Description позволяет дать развернутое описание отеля.

Свойство ImagePath может содержать закодированное изображение отеля в формате base64.

Свойство TourId внешний ключ, указывающий на тур.

Таблица Orderrs содержит информацию о заказах пользователей. Перечень полей таблицы приведен ниже в таблице 2.3.

Таблица 2.3 – Описание полей таблицы Orderrs

|  |  |
| --- | --- |
| Поле | Типы данных, ограничение целостности |
| OrderId | Int, identity primary key |
| HotelId | Int, foreign key not null |
| UserId | Nvarchar(450), foreign key not null |
| DateOrder | Datetime2(7) not null |
| Cost | Int not null |

Свойство OrderId отвечает за идентификацию заказа.

Свойство HotelId внешний ключ, указывающий на Hotels.

Свойство UserId внешний ключ, указывающий на AspNetUsers.

Свойство DateOrder хранит дату бронирования тура.

Свойство Cost содержит итоговую сумму стоимости отеля и тура.

Таблица Comments содержит комментарии к туру, оставленные пользователями. Перечень полей таблицы приведен ниже в таблице 2.4.

Таблица 2.4 – Описание полей таблицы Comments

|  |  |
| --- | --- |
| Поле | Типы данных, ограничение целостности |
| CommentId | Int, identity primary key |
| Message | Nvarchar(max) not null |
| DateMessage | Datetime2(7) not null |
| IsRight | Bit not null |
| UserId | Nvarchar(450), foreign key |
| Email | Nvarchar(max) |
| TourId | Int, foreign key |

Свойство CommentId отвечает за идентификацию комментария.

Свойство Message хранит комментарий, оставленный пользователем.

Свойство DateMessage хранит дату создания комментария.

Свойство IsRight определяет будут ли пользователи видеть комментарий. Для отображения комментария администратор должен его подтвердить.

UserId внешний ключ, указывающий на пользователя.

Свойство Email – хранит логин пользователя, оставившего комментарий.

Свойство TourId внешний ключ, указывающий на тур.

Таблица Auctions содержит информацию о проводимых аукционах на тур. Перечень полей таблицы приведен ниже в таблице 2.4.

Таблица 2.5 –Описание полей таблицы Auctions

|  |  |
| --- | --- |
| Поле | Типы данных, ограничение целостности |
| AuctionId | Int, identity primary key |
| ReservePrice | Int not null |
| BuyoutPrice | Int not null |
| CurrentBiddingPrice | Int not null |
| UploadedDate | Datetime2(7) not null |
| AuctionStatus | Int not null |
| EndDate | Datetime2(7) not null |
| TourId | Int, foreign key |

Свойство AuctionId отвечает за идентификацию аукциона.

Свойство ReservePrice хранит стартовую цену.

Свойство BuyoutPrice хранит цену выкупа

Свойство CurrentBiddingPrice хранит цену на текущий момент.

Свойство UploadedDate хранит дату объявления аукциона.

Свойство AuctionStatus хранит текущее состояние аукциона.

Свойство EndDate хранит дату завершения аукциона.

Свойство TourId внешний ключ, указывающий на тур.

## 2.3 Вывод по разделу

Программный продукт включает в себя следующие компоненты:

* Microsoft SQL Server;
* сервер приложения на базе ASP.NET Core;
* клиентская часть приложения, разработанная с помощью библиотеки React.
* данной главе были представлены архитектура программного средства и схема базы данных с подробным описанием. Для серверной части приложения спроектировано 12 таблиц базы данных под управлением Microsoft SQL Server.

# **Программная реализация**

Серверная часть написана на технологии ASP.NET Core Web API, и состоит из следующих контроллеров:

* HotelCcontroller;
* TourController;
* OrderController;
* AuctionController;
* CommentController
* UserController;

Для каждого типа объектов создан свой сервис, который выполняет обработку запросов, связанных с конкретным типом объектов.

Сервисы отвечают за обработку запроса и формирование ответа. Для извлечения данных сервисы используют репозитории, которые работают непосредственно с базой данных через объект контекста базы данных. В классе DatabaseContext осуществляется определение структуры базы данных и связей между таблицами. На рисунке 3.1 продемонстрирован сервис, возвращающий информацию о заказах пользователя.



Рисунок 3.1 – Сервис GetOrderInfo

Аутентификация пользователей осуществляется с использованием технологии Identity. Для использования этого метода аутентификации в классе Startup осуществляется регистрация метода аутентификации.

Клиентская часть написана с помощью библиотеки React.

Структура клиентской части приложения представлена на рисунке 3.2.



Рисунок 3.2 – Структура клиентской части

На рисунке 3.3 представлен скрипт компонента поисковой строки, располагающийся в директории SharedComponents.

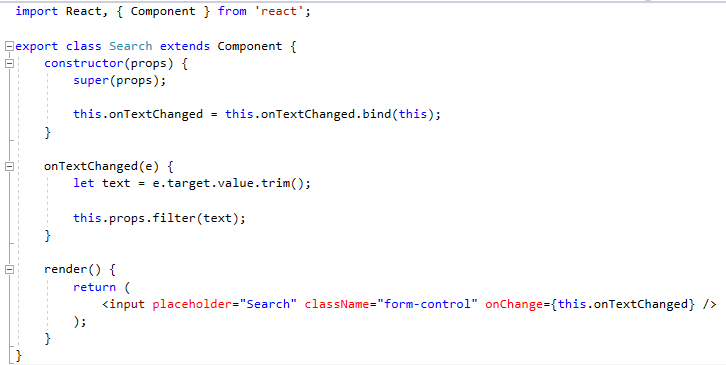


Рисунок 3.3 – скрипт компонента Search

.

Заключение

В ходе прохождения практики был изучен стандарт БГТУ по дипломному проектированию, проведен анализ информационных источников по теме дипломного проекта, изучена структура предприятия.

В ходе проведения анализа требований, разработаны функциональные требования и архитектура дипломного проекта, выбраны технологии, с помощью которых будет производиться разработка:

* система управления базами данных Microsoft SQL Server;
* Entity Framework Core для доступа к базе данных;
* сервер приложения на базе платформы ASP.NET Core;
* клиентское приложение c помощью библиотеки React.

Разработана серверная и клиентская части веб-приложения. Спроектирована структура базы данных.

В ходе преддипломной практики были закреплены, расширены и углублены полученные теоретические знания в области web-разработки, приобретены практические навыки самостоятельной работы, выработаны умения применять их при решении конкретных задач.

# **Список используемых источников**

1. Парк высоких технологий подвел итоги работы в 2016 году [Электронный ресурс]. – Режим доступа: http://www.park.by/post-1380/?lng=ru. – Дата доступа: 17.04.2020.
2. Преимущества [Электронный ресурс]. – Режим доступа: <https://www.st.by/about/advantages/>. – Дата доступа: 17.04.2020.
3. Системные технологии [Электронный ресурс]. – Режим доступа: http://park.by/it/enterprise-21/type-full/. – Дата доступа: 17.04.2020.
4. LoveTour [Электронный ресурс]. – Режим доступа: http://lovetour.by/. – Дата доступа: 11.04.2020.
5. Magellan [Электронный ресурс]. – Режим доступа: https://www.magellan.by/. – Дата доступа: 11.04.2020.
6. PeopleTravel [Электронный ресурс]. – Режим доступа: https://peopletravel.by/. – Дата доступа: 11.04.2020.
7. Microsoft wiki C Sharp [Электронный ресурс] – Режим доступа: <https://microsoft.fandom.com/ru/wiki/C_Sharp/>. – Дата доступа: 18.04.2020.
8. Расставим точки над .NET Core и .NET Standard [Электронный ресурс] – Режим доступа: <https://gosha20777.github.io/code/2018/02/22/dotnetcore/>. – Дата доступа: 17.04.2020.
9. Виртуальный DOM и детали его реализации в React [Электронный ресурс] – Режим доступа: https://ru.reactjs.org/docs/faq-internals.html/. – Дата доступа 17.04.2020.
10. REST // Википедия [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/REST/. – Дата доступа: 17.04.2020.
11. HTTP // Википедия [Электронный ресурс]. – 2020. – Режим доступа: https://ru.wikipedia.org/wiki/HTTP. – Дата доступа: 16.04.2020.

# **ПРИЛОЖЕНИЕ А**

