МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование кафедры)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

Технология программирования
(направленность (профиль)/специализация)

БАКАЛАВРСКАЯ РАБОТА

на тему «Разработка web-приложения оценки командировочных расходов на основе технологии СОА»

Студент А.И. Вагапов _____
(И.О. Фамилия) (личная подпись)

Руководитель С.В. Мкртычев _____
(И.О. Фамилия) (личная подпись)

Допустить к защите

Заведующий кафедрой к.тех.н, доцент, А.В. Очеповский _____
(ученая степень, звание, И.О. Фамилия) (личная подпись)

«_____» _____ 2018 г.

Тольятти 2018

АННОТАЦИЯ

Тема: Разработка web-приложения оценки командировочных расходов на основе технологии СОА.

Ключевые слова: WEB - ПРИЛОЖЕНИЕ, ОЦЕНКА КОМАНДИРОВОЧНЫХ РАСХОДОВ, ТЕХНОЛОГИЯ СОА.

Объект исследования бакалаврской работы – оценка командировочных расходов. Предмет исследования бакалаврской работы – web-приложение оценки командировочных расходов. Цель бакалаврской работы - разработка web-приложения оценки командировочных расходов на основе технологии СОА.

Задачами бакалаврской работы являются:

− анализ известных ИТ-решений для оценки командировочных расходов и обосновать целесообразной разработки нового web-приложения;

− разработка логической модели web-приложения для оценки командировочных расходов;

− выбор средства разработки web-приложения для оценки командировочных расходов;

− реализация web-приложения и оценка его эффективности.

Методы исследования: методы оценки командировочных расходов, объектно-ориентированное программирование, технология СОА.

Выполнена постановка задачи и разработано техническое задание на проектирование web-приложения. Разработана логическая модель web-приложения на основе технологии СОА. Выполнена реализация web-приложения и подтверждена его функциональная эффективность.

Структура бакалаврской работы: страниц 52 с приложением, рисунков 15, таблиц 11 , источников 26 .

ABSTRACT

The topic of bachelor's work is “Developing a web application for estimating travel expenses based on SOA technology”.

Key words: WEB APPLICATION, ESTIMATION OF TRAVEL EXPENSES, SOA TECHNOLOGY.

The object of the study of bachelor's work is the estimation of travel expenses. The subject of the study of bachelor's work is a web application for estimating travel expenses. The aim of the bachelor's work is the development of a web app for the estimation of travel expenses on the basis of SOA technology.

The tasks of bachelor's work are:

- analysis of well-known IT solutions for estimating travel expenses and justify the expedient development of a new web application;

- development of a logical model of a web application for the assessment of travel expenses;

- selection of a web application development tool for estimating travel expenses;

- implementation of the web application and evaluation of its effectiveness.

Research methods: methods for estimating travel expenses, object-oriented programming, SOA technology.

The task was set and a technical task was developed for designing a web application. A logical model of a web application based on SOA technology is developed. Implemented the implementation of the web application and confirmed its functional effectiveness.

The structure of the bachelor's work: pages 52 with the application, figures 15, tables 11, sources 26.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В условиях кризиса для обеспечения эффективной работы предприятия необходимо оптимизировать его расходы, в том числе на производственные нужды.

В полной мере это относится и к командировочным расходам.

Совершенно очевидно, что помимо чисто управленческих решений, направленных на сокращение той или иной статьи затрат, успешное решение данной проблемы невозможно без автоматизации процесса оценки командировочных расходов.

Необходимо отметить, что в условиях постоянно меняющихся курсов валют, цен на авиабилеты и аренду жилья, для определения реальной стоимости командировочных расходов необходима актуальная информация по каждому из их видов. Это позволит существенно повысить эффективность управления соответствующей статьей затрат предприятия.

Данную проблему можно решить с помощью современных web-технологий и подходов к реализации программного обеспечения (ПО).

В последнее время для повышения гибкости программного обеспечения и снижения затрат на его разработку успешно применяются новые интеграционные технологии на основе сервисно-ориентированной архитектуры (СОА), обеспечивающие взаимодействие различных аппаратных и технологических платформ, языков программирования, операционных систем и коммуникаций.

Применение СОА обеспечит следующие преимущества:

- быструю адаптацию к изменяющимся условиям рынка;
- сокращение времени ввода новых продуктов;
- увеличение доходов компании;
- смещение акцента с технической составляющей на результат, доступность информационной системы для бизнеса, концентрацию на его основных задачах.

Таким образом, **актуальность** бакалаврской работы обусловлена необходимостью разработки web-приложения оценки командировочных расходов на основе технологии СОА для повышения эффективности управления соответствующими затратами предприятия.

Объект исследования бакалаврской работы – оценка командировочных расходов.

Предмет исследования бакалаврской работы – web-приложение оценки командировочных расходов.

Цель бакалаврской работы - разработка web-приложения оценки командировочных расходов на основе технологии СОА.

Для достижения поставленной цели необходимо решить следующие задачи:

− произвести анализ известных ИТ-решений для оценки командировочных расходов и обосновать целесообразной разработки нового web-приложения;

− разработать логическую модель web-приложения для оценки командировочных расходов;

− выбрать средства разработки web-приложения для оценки командировочных расходов;

− реализовать web-приложение и оценить его эффективность.

Методы исследования: технология СОА**,** метод объектно-ориентированного анализа и проектирования.

Практическая значимость работы заключается в разработке нового web-приложения для оценки командировочных расходов.

Соответствие содержания бакалаврской работы профессиональным компетенциям по видам профессиональной деятельности выпускника:

- научно-исследовательская деятельность:

− готовность к использованию метода системного моделирования при исследовании и проектировании программных систем (ПК-1).

Представленная бакалаврская работа состоит из введения, трех глав, заключения, приложения и списка литературы.

Во введение обозначается тема работы и ее актуальность, описывается объект и предмет исследования, цели и задачи, которые необходимо решить в данной работе.

В первой главе представлены обзор и анализ известных ИТ-решений, предназначенных для оценки командировочных расходов, и обоснована целесообразность разработки нового web-приложения на основе технологии СОА.

Вторая глава посвящена разработке логической модели web-приложения на основе объектно-ориентированного подхода.

В третьей главе описан процесс реализации web-приложения и оценена его эффективность.

В заключении представлены результаты выполнения бакалаврской работы.

В приложении приведены фрагменты кода web-приложения.

Глава 1 АНАЛИЗ СОСТОЯНИЯ ПРОБЛЕМЫ И ПОСТАНОВКА ЗАДАЧИ НА РАЗРАБОТКУ WEB-ПРИЛОЖЕНИЯ ОЦЕНКИ КОМАНДИРОВОЧНЫХ РАСХОДОВ

1. 1 Разработка требований к web-приложению оценки командировочных расходов

Проектируемое web-приложение предназначено для оценки командировочных расходов, представляющие собой денежные затраты работника, связанные непосредственно со служебной поездкой, в которую он направляется по распоряжению руководителя [14].

Для разработки требований к web-приложению для оценки командировочных расходов (далее - web-приложения) используем технологию FURPS+ [26].

Название FURPS+ происходит от аббревиатуры, представляющей собой усовершенствованную модель для классификации атрибутов качества программного обеспечения (функциональных и нефункциональных требований).

Данная технология широко применяется в программной индустрии в настоящее время.

Ниже представлены требования к разрабатываемому web-приложения, сформулированные на основе данной технологии:

1) Functionality, функциональность:
- регистрация пользователей;
- наличие личного кабинет;
- оценка командировочных расходов в онлайновом режиме;
- формирование отчетности;
2) Usability, удобство использования:
- наличие справочной информации;
- отсутствие функциональной избыточности;
3) Reliability, надежность:

- обеспечение восстановления после сбоев средствами системы;
- режим работы 7/24/365.

4) Performance, производительность:

- количество транзакций в секунду: 300;
- допустимое количество одновременно работающих пользователей: 20;
- время формирования отчетов: до 1 мин.

5) Supportability, поддерживаемость:

- возможность масштабирования;
- простота адаптации;
- дистанционное администрирование;
- время устранения критических проблем: 3 часа.

6) Проектные ограничения:

- применение web-технологий;
- применение технологии СОА;
- применение средств реализации ПО по усмотрению разработчика.

Новое web-приложение должно отвечать вышеперечисленным требованиям.

1. 2 Обзор и анализ ИТ-решений для оценки командировочных расходов

Рассмотрим известные ИТ-решения для оценки командировочных расходов на предмет соответствия установленным требованиям.

1.2.1 Облачное приложение «Командировочные расходы»

Облачное приложение «Командировочные расходы» предназначено для управления бизнес-поездками и, по мнению разработчиков, обеспечивает полную прозрачность всего процесса, а также предоставляет комплексную аналитику данных и детальную отчетность [11].

Всё это позволяет максимально оптимизировать расходы по командировкам и авансовым отчетам (рисунок 1.1).

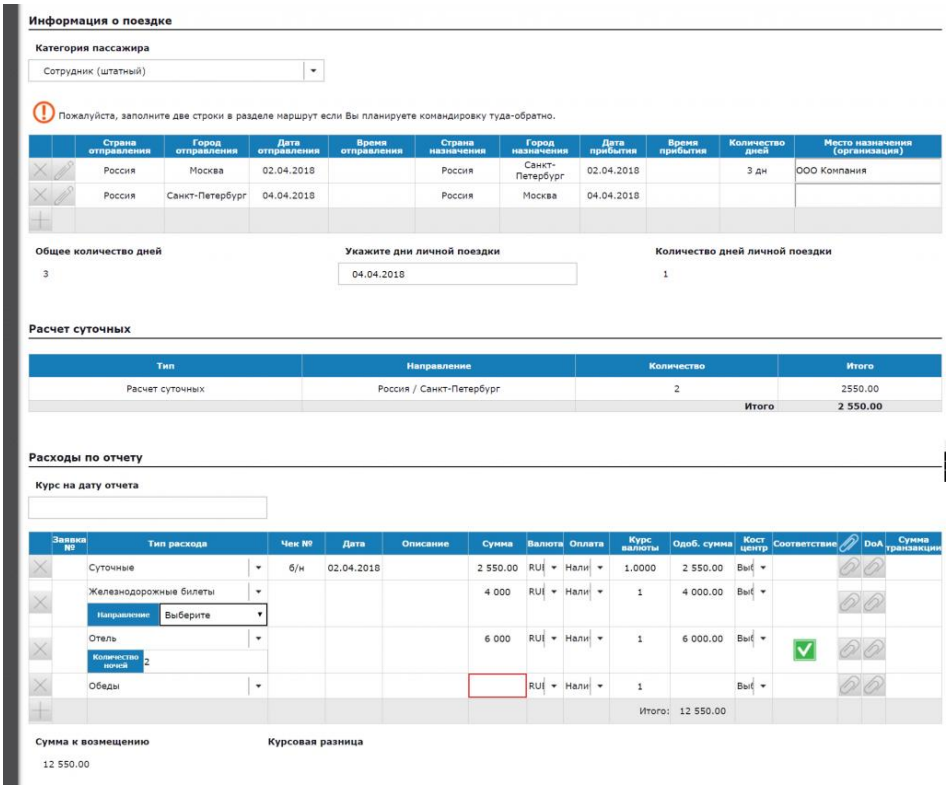

Рисунок 1.1 – Окно облачного приложения «Командировочные расходы»

Основные возможности приложения:

– простой доступ к выписанным билетам, ваучерам на гостиницу и прочим услугам, а также быстрый запрос опций по поездке у туристических агентств;

– возможность как самостоятельно бронировать опции по поездке, так и делать это через утвержденные туристические агентства или Travel-координатора;

– гибкая настройка процесса согласования бизнес-поездок в соответствии с правилами компании;

– обширная отчётность для анализа и оптимизации расходов и другие.

Приложение разработано на базе SaaS-решения или облачной платформы.

Облачная высокопроизводительная платформа Hamilton Apps Hub построена на базе открытых технологий и состоит из различных функциональных модулей. Все приложения Hamilton размещены и функционируют на этой платформе, используя 70% общего клиентского кода.

Все специфические настройки по приложениям, а также настройки под бизнес-процессы клиента делаются через дополнительные параметры самих приложений.

Решения Hamilton Apps используют мультиарендную архитектуру - это значит, что все пользователи приложений одновременно используют единую, общую, централизованную инфраструктуру и базу кода.

Данный облачный сервис – платный.

1.2.2 Онлайн-сервис для планирования командировок OneTwoTrip for Business

OneTwoTrip for Business — это платный сервис для организации деловых поездок. Он помогает бизнесу упростить организацию командировок и сократить расходы с помощью автоматизации процессов [12].

С помощью сервиса можно выбрать направления и даты и приобрести билеты с банковской карты или депозита компании. В том числе персональные или групповые авиабилеты. Данные о покупке появятся в отчётности для компании. Доступна и интеграция с системами корпоративного документооборота. Сотрудникам можно выдавать разные уровни доступа и установить тревел-политику (рисунок 1.2).

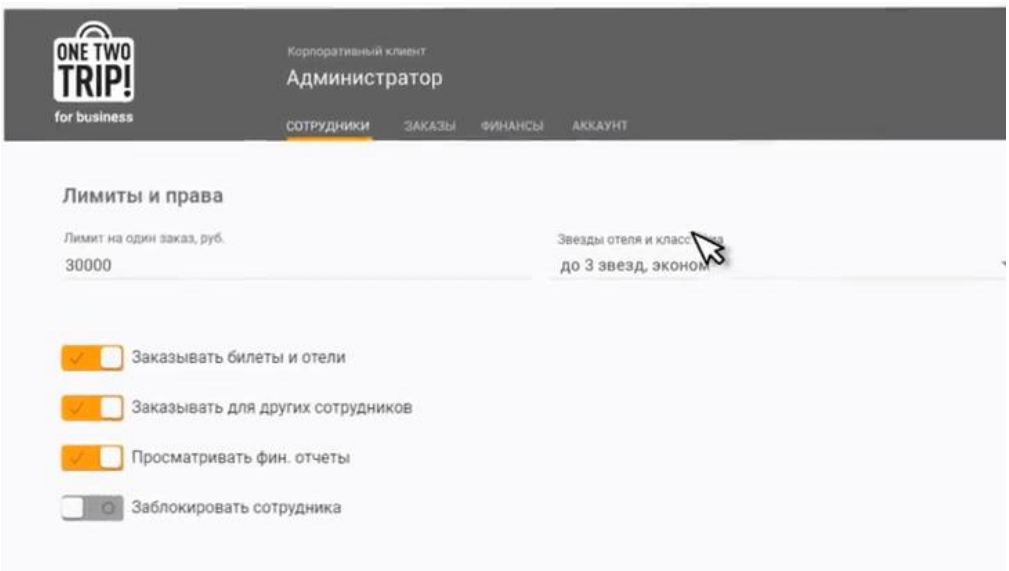

Рисунок 1.2 – Окно онлайн-сервиса OneTwoTrip for Business

Основные характеристики онлайн-сервиса OneTwoTrip:

− платформа: web – приложение;

− развёртывание: облако;

− доступные языки: русский;

− поиск и фильтры.

Безопасность и конфиденциальность обеспечивает благодаря доступу по протоколу HTTPS.

Возможно интеграция и доработка под требования заказчика.

Сервис доступен только в России.

1.2.3 Онлайн-система бронирования деловых поездок TopClient

TopClient - это платный сервис по заказу услуг для оптимизации организации деловых поездок [13].

Возможности TopClient:

− самостоятельное бронирование или через менеджера;

− контроль тревел политики;

− бухгалтерские документы;

− консультация специалистов;

− весь спектр туристических услуг;

− контроль расходов компании;

− статистика поездок за любой период;

− отчеты по каждому сотруднику, отделу или компании в целом.

Основные характеристики онлайн-сервиса TopClient:

− платформа: web – приложение;

− развёртывание: облако;

− доступные языки: русский;

− поиск и фильтры.

Система призвана помочь компаниям в планировании и организации командировок сотрудников, в сокращении временных затрат, в контроле расходов (рисунок 1.3).

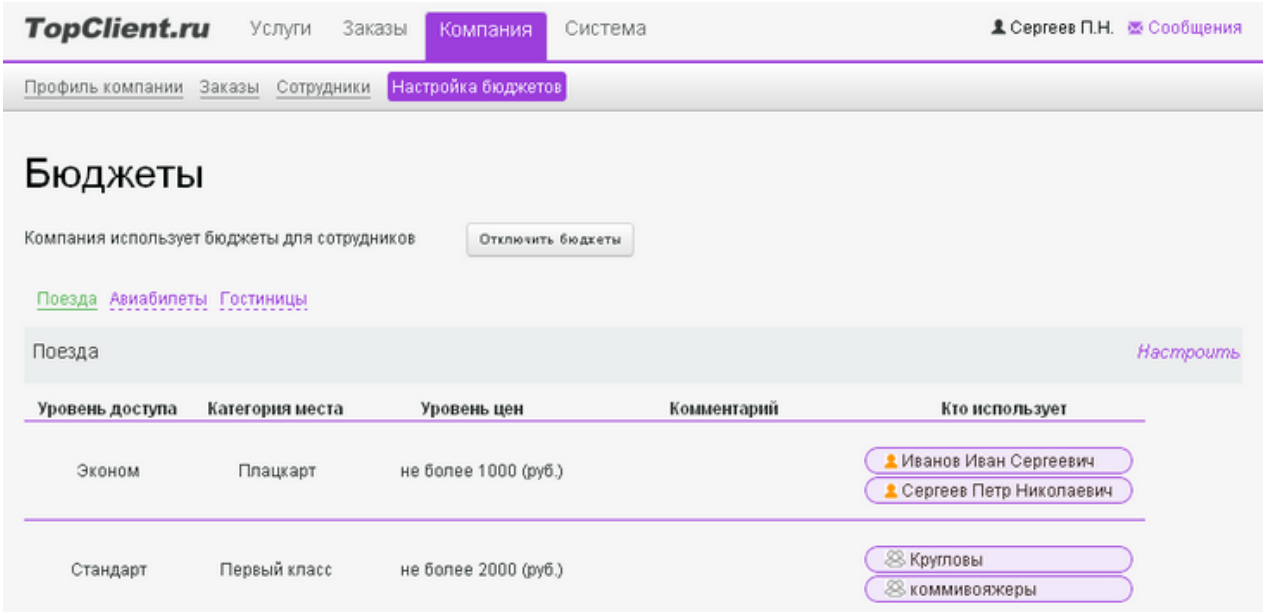

Рисунок 1.3 – Окно онлайн-системы TopClient

Возможно интеграция и доработка онлайн-системы под требования заказчика.

Для проведения сравнительного анализа ИТ-решений все ключевые критерии сравнения ведены в таблицу 1.1.

Таблица 1.1 - Сравнительный анализ аналогов web-приложения оценки командировочных расходов

Критерий /Аналог	Облачное приложение «Командировочные расходы»	Онлайн-сервис OneTwoTrip for Business	Онлайн-система TopClient
оценка командировочных расходов в онлайновом режиме	+	+	+
функциональная избыточность	-	-	-
использование web – технологий	+	+	+

Продолжение таблицы 1.1

использование технологии СОА	+	+	+
простота адаптации	-	-	-
Итого	3	3	3

Как следует из таблицы 1.1, представленные аналоги не отвечают всем требованиям, предъявляемым к web-приложению, прежде всего, требованию по обеспечению простоты адаптации.

В этой связи принято решение о разработке нового web-приложения для оценки командировочных расходов.

1.3 Разработка технического задания на создание web-приложения оценки командировочных расходов

В соответствии с сформулированными требованиями было рпзработано техническое задание на создание web-приложения оценки командировочных расходов (таблица 1.2).

Таблица 1.2 - ТЗ на создание web-приложения оценки командировочных расходов

Раздел	**Содержание**
1. Общие сведения	Наименование: web-приложения оценки командировочных расходов. Основание для проведения работ: бакалаврская работа. Разработчик: Вагапов А.И.
2. Назначение и цели создания системы	Оценки командировочных расходов на основе технологии СОА для повышения эффективности управления соответствующими затратами предприятия

Продолжение таблицы 1.2

3. Характеристика объектов автоматизации	Объектом автоматизации является процедура оценка командировочных расходов. Пользователя: канцелярия, сотрудники.
4. Требования к системе	Архитектура: клиент-сервер, трехзвенная. Технология разработки ПО: СОА Протоколы взаимодействия между компонентами: TCP/IP, SOAP. Источник данных: БД web-приложения. Режим функционирования: 7/24/365. Профилактические работы: не предусматриваются. Ответственное лицо: администратор сайта. Надежность и защита web-приложения должна обеспечиваться за счет применения программно-технических средств хостинговой компании. Требования по видам обеспечения: - требования к математическому обеспечению: нет; - требование к информационному обеспечению: классификаторы и справочники по видам командировочных расходов; - требование к лингвистическому обеспечению: бесплатная среда web-программирования; - требования к ПО: ОС Windows 7/8/10, бесплатная СУБД; -требования к ТО: компьютер-клиент: CPU Intel Core 2,3 Gz, RAM: 4Gb, 500 Gb; сервер: CPU 6 (12 core); RAM: 32 Gb; HDD: 1000 Gb.

Продолжение таблицы 1.2

5. Состав и содержание работ по созданию системы	Работы по созданию web-приложения выполняются в 2 этапа: проектирование и реализация.
6. Порядок контроля и приемки системы	Опытная эксплуатация.
7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие	Разработка логической модели и модели данных web-приложения.
8. Требования к документированию	Разработка инструкции пользователя.
9. Источники разработки	Задание на выполнение бакалаврской работы.

ТЗ составлено на основе ГОСТ 34.602-89. Техническое задание на создание автоматизированной системы [2].

Выводы к главе 1

1) Для разработки требований к web-приложению для оценки командировочных расходов принято решение использовать технологию FURPS+, позволяющую классифицировать требования по признакам, отражающим основные критерии выбора ИТ-решений для оценки командировочных расходов.

2) Рассмотренные ИТ-решения для оценки командировочных расходов не отвечают всем сформулированным требованиям к web-приложению.

Поэтому принято решение о разработке нового web-приложения для оценки командировочных расходов и разработано ТЗ на создание данного приложения согласно ГОСТу.

Глава 2 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ WEB-ПРИЛОЖЕНИЯ ОЦЕНКИ КОМАНДИРОВОЧНЫХ РАСХОДОВ

2.1 Выбор технологии СОА для разработки web-приложения

СОА (SOA, Service-Oriented Architecture) уже давно признана наиболее подходящим подходом для решения проблем создания приложений, легко адаптируемых к изменениям в бизнес, благодаря присущие ей гибкости и повторному использованию атрибутов этой архитектуры [5].

Основным компонентов СОА является web-сервис, под которым понимается:

а) метод взаимодействия между двумя устройствами в Интернете;

б) программная система стандартизированного межмашинного взаимодействия по сети.

Следует выделить следующие технологии СОА [20]:

1. Сервисная шина предприятия (ESB). Как вариация сервера приложений или платформы EAI, ESB организует связь между сервисами, а также с пользователем и любыми источниками.

2. Сервисный реестр или репозиторий. Эта система баз данных отслеживает различные компоненты сервиса, доступные для повторного использования, и публикует доступные сервисы для бизнес-аналитиков и внешних партнеров. В отличие от традиционной базы данных, реестр в виде метаданных также должен сохранять контекст сервиса, который определяет, когда и как он должен использоваться.

3. Основные системы управления данными и хранилища метаданных. Эти две категории продуктов помогают предприятиям управлять своими данными одним и тем же распределенным способом управления бизнес-логикой через сервисы СОА. Это гарантирует, что назначение данных понимается всеми сервиса, которые их используют, поэтому никакие скрытые предположения не

приводят к повреждению результатов, которые предоставляют сервисы при использовании или генерировании данных.

4. Управление пользовательским интерфейсом. Поскольку сервисы охватывают области от веб-порталов до офисных инструментов, управление пользовательским интерфейсом будет требовать сервис-ориентированного подхода.

Описанные технологии используют известные протоколы (наборы операций) WDSL и SOAP.

В последнее время широкое распространения получила технология REST API.

REST API (Representational State Transfer), или веб-службы RESTful, — что это? REST в переводе с английского «репрезентативная передача состояния». Это способ обеспечения взаимодействия между компьютерными системами в Интернете. REST-совместимые веб-службы, позволяющие запрашивающим системам получать доступ к текстовым представлениям веб-ресурсов и управлять ими, используя единый и предопределенный набор операций (рисунок 2.1) [18].

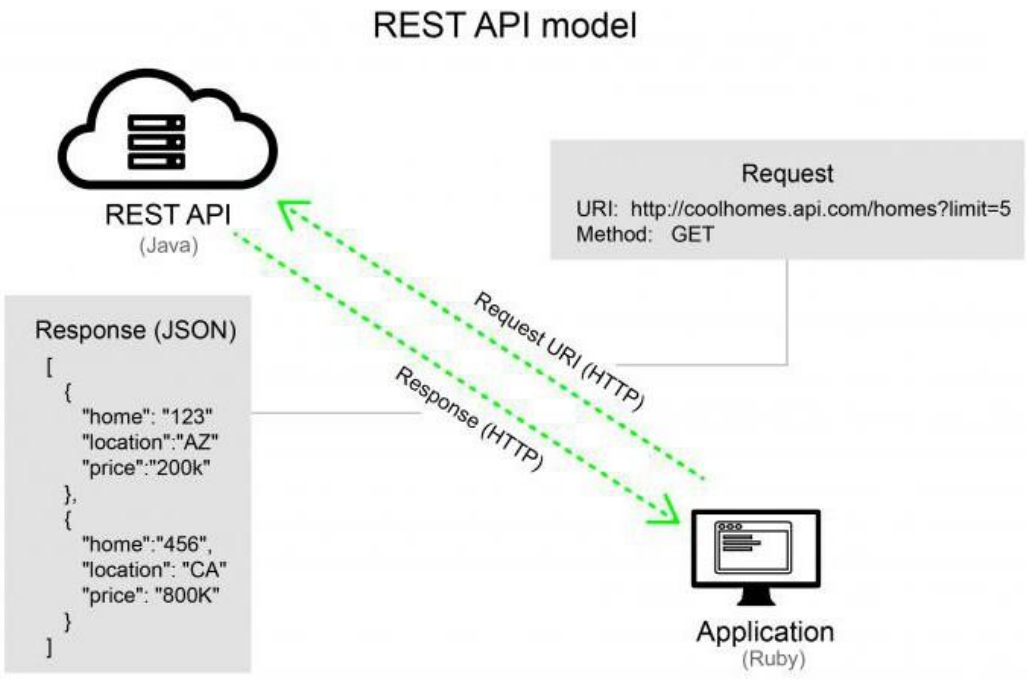

Рисунок 2.1 – Пример модели REST API

Главные преимущества технологии REST API:

− производительность - взаимодействие компонентов является доминирующим свойством в восприятии пользователями производительности и результативности сети;

− масштабируемость для поддержки максимального количества компонентов, тестирования REST API и взаимодействия между ними;

− простота единого интерфейса и авторизации REST API;

− модифицируемость компонентов для удовлетворения меняющихся потребностей (даже во время работы приложения);

− видимость связи между составляющими и сервисными агентами;

− возможность переносить компоненты, перемещая их программный код с данными;

− надежность — высокая отказоустойчивость при наличии сбоев в составе, разъемов или данных.

Следует учесть, что данная технология предполагает взаимодействие только «клиент-сервер» и не применим для разработки распределенных систем, что не противоречит проектным ограничениям на разработку web-приложения.

Другим недостатком технологии является то, что REST не предоставляет встроенной поддержки безопасности. Это очень важно при проектировании web-сервисов REST - требования безопасности и проектирования выполняются заранее. Рекомендуется при разработке требований безопасности для таких web-сервисов учитывать данные моменты.

Однако эта особенность не критична для потенциальных пользователей web-приложения оценки командировочных расходов, поэтому принято решение использовать технологию REST API для его разработки.

2.2 Разработка логическая модели web-приложения оценки командировочных расходов

Для разработки логической модели web-приложения используем языквизуального моделирования UML.

UML (Unified Modeling Language) - это язык и метод визуализации программного обеспечения с помощью набора диаграмм, построенных на основе объектно-ориентированной нотации [15].

На практике для логического проектирования программного приложения достаточно создать разработать диаграммы, отражающие ее основные аспекты, а именно: диаграмму вариантов использования, диаграмму классов и диаграмму последовательности.

2.2.1 Диаграмма вариантов использования web-приложения оценки командировочных расходов

Диаграммы вариантов использования моделируют функциональность системы с использованием актеров и прецедентов.

Варианты использования - это набор действий, служб и функций, которые должна выполнять система. В рассматриваемом контексте – это web-приложение оценки командировочных расходов.

«Актеры» - это люди, организации и внешние системы, которые работают под определенными ролями внутри системы.

Диаграммы вариантов использования помогают выявлять любые внутренние или внешние факторы, которые могут влиять на систему.

Для этого опишем в технологии RUP функции, которые должна реализовать web-приложение для каждого из нижеперечисленных актеров [9]:

Пользователь web-приложения.

Web-сервис.

Опишем прецеденты в табличной форме (таблицы 2.1-2.4).

Таблица 2.1 – Прецеденты

Прецеденты	Краткое описание
1. Регистрация	Регистрация новых пользователей с разграничением прав доступа
2. Оценка командировочных расходов	Оценка затрат по каждому виду командировочных расходов
3. Предоставление онлайн-ресурса	Предоставление онлайн-ресурса по каждому виду командировочных расходов

Таблица 2.2 - Описание прецедента: Регистрация

Прецедент: регистрация
ID: 1
Краткое описание: регистрация новых пользователей
Главный актер: Пользователь
Второстепенные актеры: нет
Предусловие: прецедент начинается по инициативе Пользователя
Основной поток: Пользователь регистрируется в базе данных (БД) web-приложения
Постусловие: Пользователь зарегистрирован в БД с созданием личного кабинета
Альтернативные потоки: нет

Таблица 2.3 - Описание прецедента: оценка командировочных расходов

Прецедент: оценка командировочных расходов
ID: 2
Краткое описание: оценка затрат по каждому виду командировочных расходов

Продолжение таблицы 2.3

Главный актер: Пользователь активизирует web-сервис, связанный с конкретным видом командировочных расходов и выбирает приемлемый вариант
Второстепенные актеры: нет
Предусловие: прецедент начинается по инициативе Пользователя
Основной поток: Пользователь
Постусловие: заполненная позиция затрат по конкретному виду командировочных расходов
Альтернативные потоки: нет

Таблица 2.4 - Описание прецедента: предоставление ресурса

Прецедент: предоставление онлайн-ресурса
ID: 3
Краткое описание: предоставление онлайн-ресурса по каждому виду командировочных расходов
Главный актер: Web-сервис
Второстепенные актеры: нет
Предусловие: прецедент начинается по Пользователя
Основной поток: Web-сервис обеспечивает подключение к онлайн-ресурсу
Постусловие: Пользователь получает доступ к онлайн-ресурсу
Альтернативные потоки: нет

На основании представленных таблиц построена диаграмма вариантов использования web-приложения (рисунок 2.2).

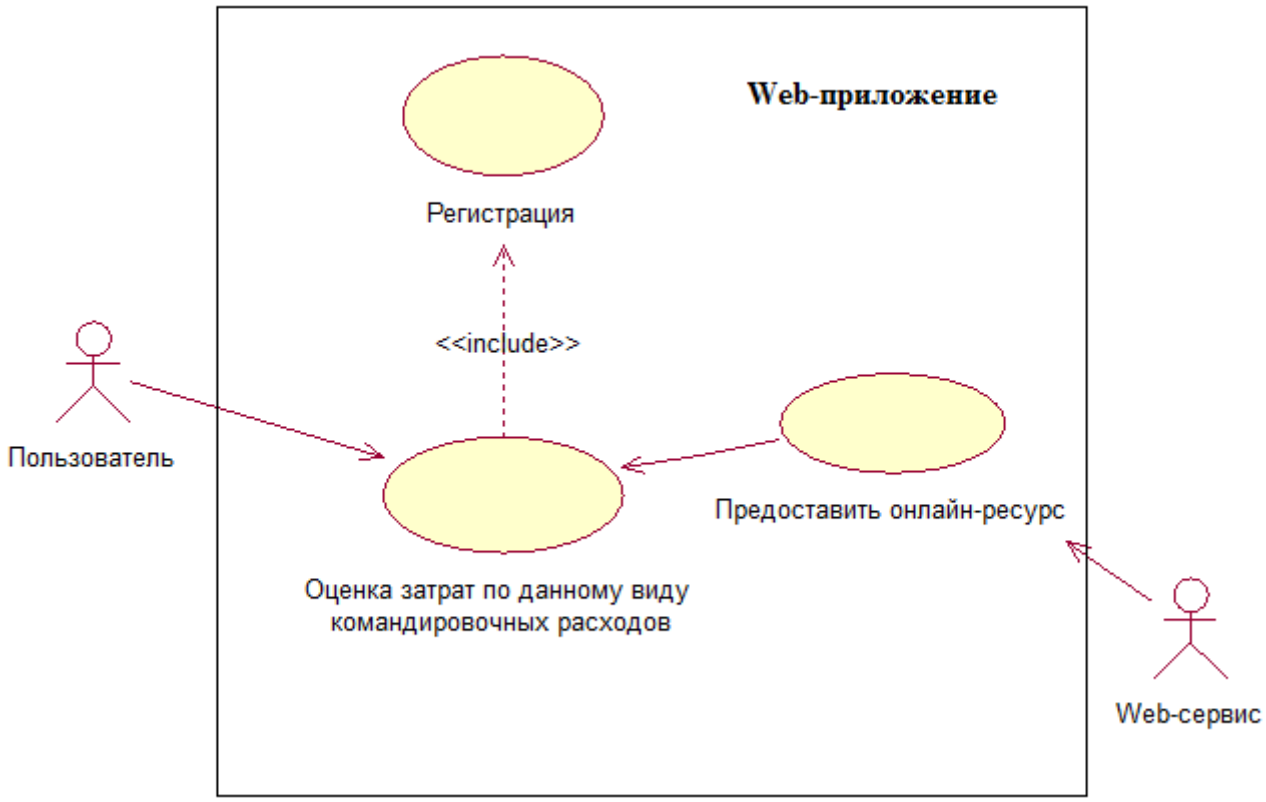

Рисунок 2.2 – Диаграмма вариантов использования web-приложения оценки командировочных расходов

Данная диаграмма отражает функциональный аспект web-приложения оценки командировочных расходов.

2.1. 2 Диаграмма классов web-приложения оценки командировочных расходов

Диаграмма классов моделирует отношения между классами, объектами, атрибутами и операциями.

Классы представляют собой абстракцию объектов с общими характеристиками. Связи между ними представляют собой отношения между классами.

Помимо диаграмм классов рекомендуется использовать диаграммы объектов, которые описывают статическую структуру системы в определенное время.

Они могут использоваться для проверки диаграмм классов для точности.

Рассмотрим диаграмму классов web-приложения оценки командировочных расходов (рисунок 2.3).

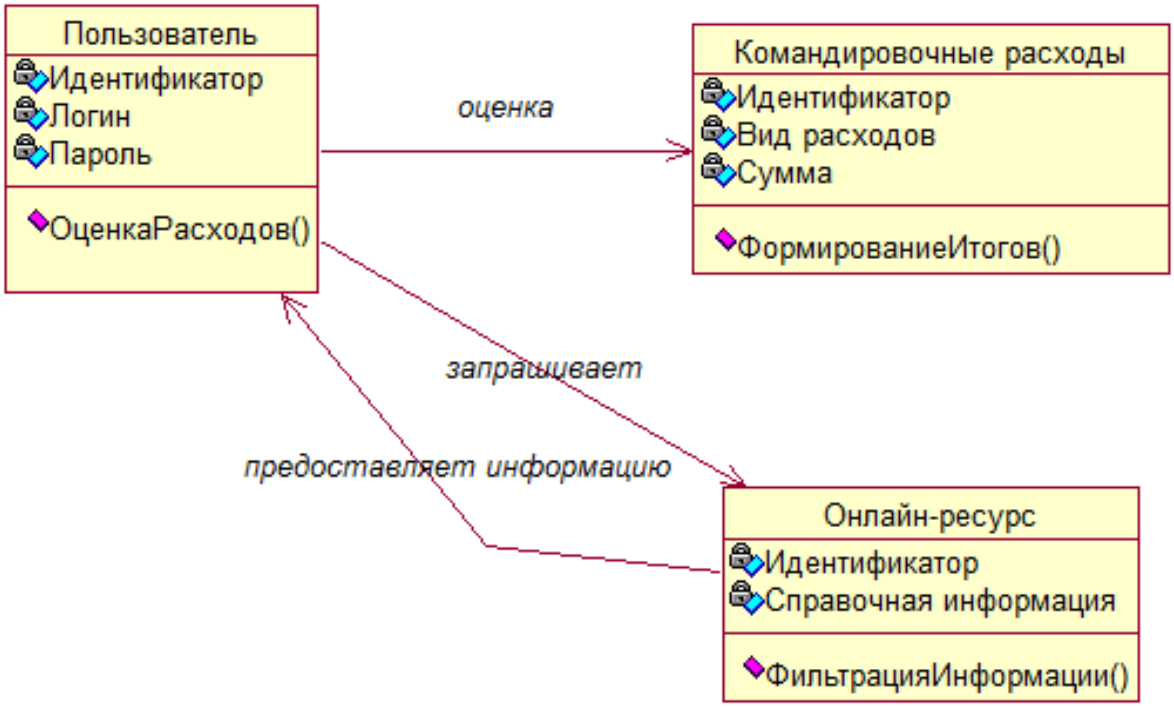

Рисунок 2.3 - Диаграмма классов web-приложения оценки командировочных расходов

Спецификация диаграммы классов web-приложения оценки командировочных расходов:

«Пользователь» - класс объектов - физических лиц, которые оценивают командировочные расходы.

«Командировочные расходы» - класс объектов – записей в таблице командировочных расходов БД web-приложения.

«Онлайн-ресурс» – класс объектов – интернет-ресурсов, предоставляющих справочную или иную информацию по конкретному виду командировочных расходов.

Связи между классами – именованные ассоциации.

На рисунке 2.4 изображена диаграмма объектов REST API.

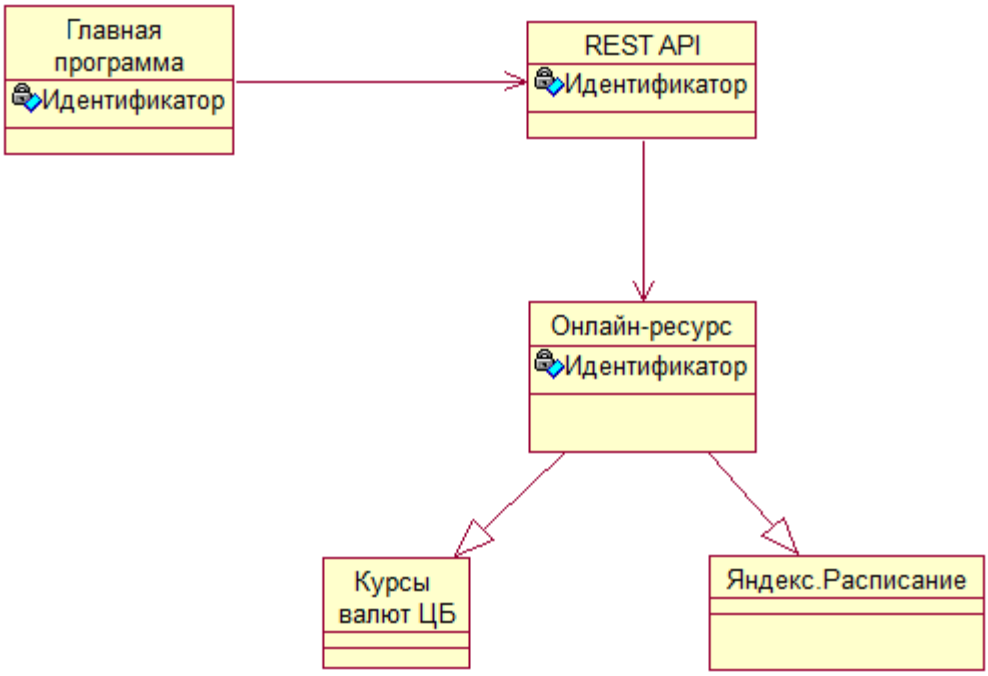

Рисунок 2.4 – Диаграмма объектов web-приложения оценки командировочных расходов

Здесь «REST API» - класс объектов – web-сервисов, построенных по технологии REST API.

«Курсы валют ЦБ» и «Яндекс.Расписание» - объекты-наследники класса «Онлайн-ресурс».

2.3 Диаграмма последовательности оценки командировочных расходов

Диаграмма последовательности является хорошим способом визуализации и проверки различных сценариев выполнения вариантов использования. Они могут помочь предсказать, как система будет себя вести и обнаружить задачи, которые класс, возможно, должен иметь в процессе моделирования новой системы.

На рисунке 2.5 представлена диаграмма последовательности сценария оценки командировочных расходов.

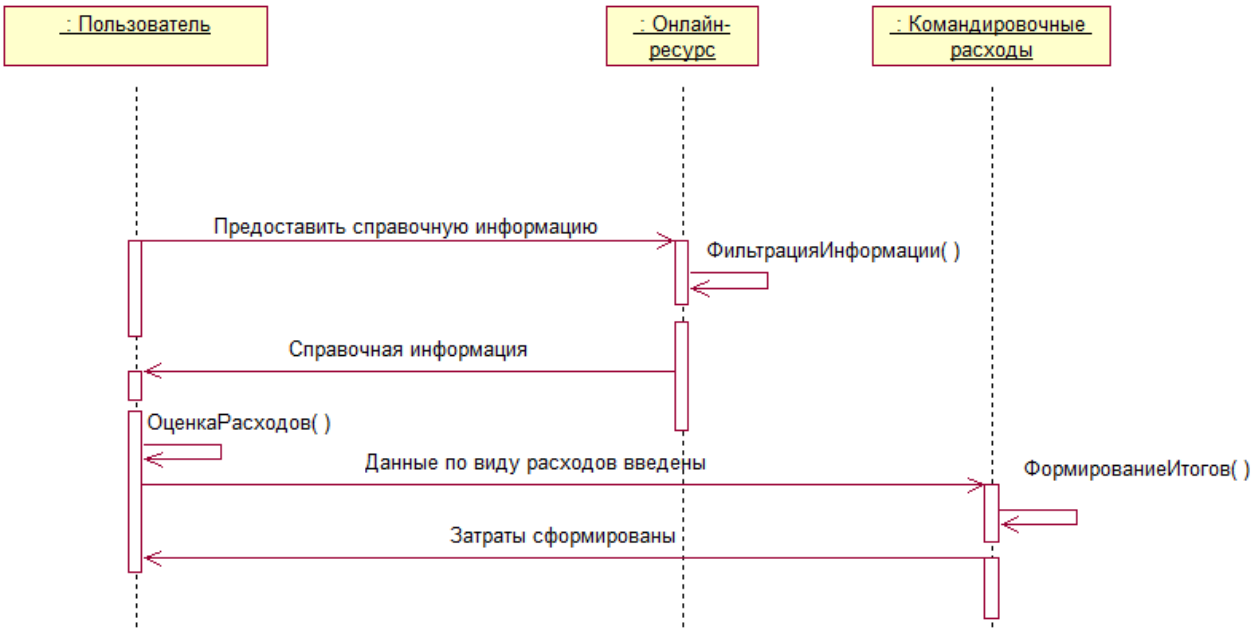

Рисунок 2.5 – Диаграмма последовательности сценария оценки командировочных расходов

Процесс инициализируется объектом Пользователь.

Объект Пользователь запрашивает справочную информацию у объекта Онлайн-ресурс по конкретному виду командировочных расходов.

Объект Онлайн-ресурс выполняет фильтрацию справочной информации и возвращает объекту Пользователь запрашиваемую справочную информацию.

Объект Пользователь выполняет оценку затрат по конкретному виду расходов и передает данные объекту Командировочные расходы для формирования итогов

Объект Командировочные расходы формирует итоги по данному виду командировочных расходов и выдает результат объекту Пользователь.

Процесс оценки командировочных расходов завершается.

Выводы к главе 2

1) Такое преимущество технологии REST API, как простота единого интерфейса и авторизации, обеспечивает возможность ее использования для разработки web-приложения.

2) Для логического проектирования web-приложения достаточно создать разработать диаграммы UML, относящиеся к ядру языка и отражающие основные аспекты приложения.

3) Диаграмма вариантов использования является основой логической модели web-приложения и отображает его функциональный аспект.

4) Диаграмма объектов служит для отражения статической структуры web-приложения и модели наследования его классов.

5) Диаграмма последовательности web-приложения обеспечивает возможность представления взаимодействующих объектов и сообщений между ними в динамике.

Глава 3 РЕАЛИЗАЦИЯ WEB-ПРИЛОЖЕНИЯ ОЦЕНКИ КОМАНДИРОВОЧНЫХ РАСХОДОВ

3.1 Выбор среды для реализации web-приложения оценки командировочных расходов

Для сравнения рассмотрим наиболее известные отечественные коммерческие системы управления контентом (CMS), основанные на технологии PHP+MySQL: 1С-Битрикс, UMI.CMS и NetCat.

Продукт «1С-Битрикс: Управление сайтом», по мнению разработчиков, представляет собой программное ядро для всестороннего управления веб-проектами любой сложности [8].

Для поддержки данной CMS используется следующее ПО:

- язык программирования PHP (версия 5.6 и выше);
- web-сервер Apache (версия 1.3 и выше);
- СУБД (версия 1.3 MySQL 5.0 и выше).

Следует отметить, что все вышеперечисленные программные продукты относятся к бесплатному ПО.

Ядром CMS является разработанная на языке PHP платформа Bitrix Framework, которая помимо набора классов предоставляет разработчику развитый интерфейс администрирования.

Среди ключевых преимуществ 1С-Битрикс следует выделить ориентации на потребности отечественных разработчиков web-приложений и возможность интеграции с технологической платформой «1С: Предприятие».

UMI.CMS – мультисайтовая CMS, разработанная на основе технологии PHP+MySQL. Относится к категории кроссплатформенного ПО и считается одной из самых популярных систем управления контентом в России. UMI.CMS также достаточно просто интегрируется с продуктами 1С и внешними сервисами [16].

Среди недостатков модно отметить относительно высокую стоимость платформы.

NetCat представляет собой систему управления сайтом, позволяющую реализовать проекты различного уровня, от простых сайтов до интернет-магазинов и интернет-порталов [17].

Платформа NetCat состоит из двух логически связанных частей: системы ввода/вывода информации на сайте Front-Office и интерфейса управления сайтом Back-Office.

Обе части реализованы в технологии PHP +MySQL.

Для удобства сравнения все основные критерии представлены в таблице 3.1.

Таблица 3.1 – Сравнительный анализ CMS

Критерий/CMS	1С-Битрикс	UMI.CMS	NetCat
стоимость платформы	+	-	+
функциональность	+	+	-
простота интеграции с внешними сервисами	+	+	+
требовательность к ресурсам	-	-	+
распространенность	+	+	-
Итого (+)	4	3	3

По результатам анализа в качестве среды для реализации web-приложения оценки командировочных расходов выбран продукт «1С-Битрикс: Управление сайтом».

Следует также отметить, что разработчик web-приложения имеет опыт работы с этой платформой.

3.2 Разработка программы web-приложения оценки командировочных расходов

На рисунке 3.1 изображена диаграмма компонентов web-приложения.

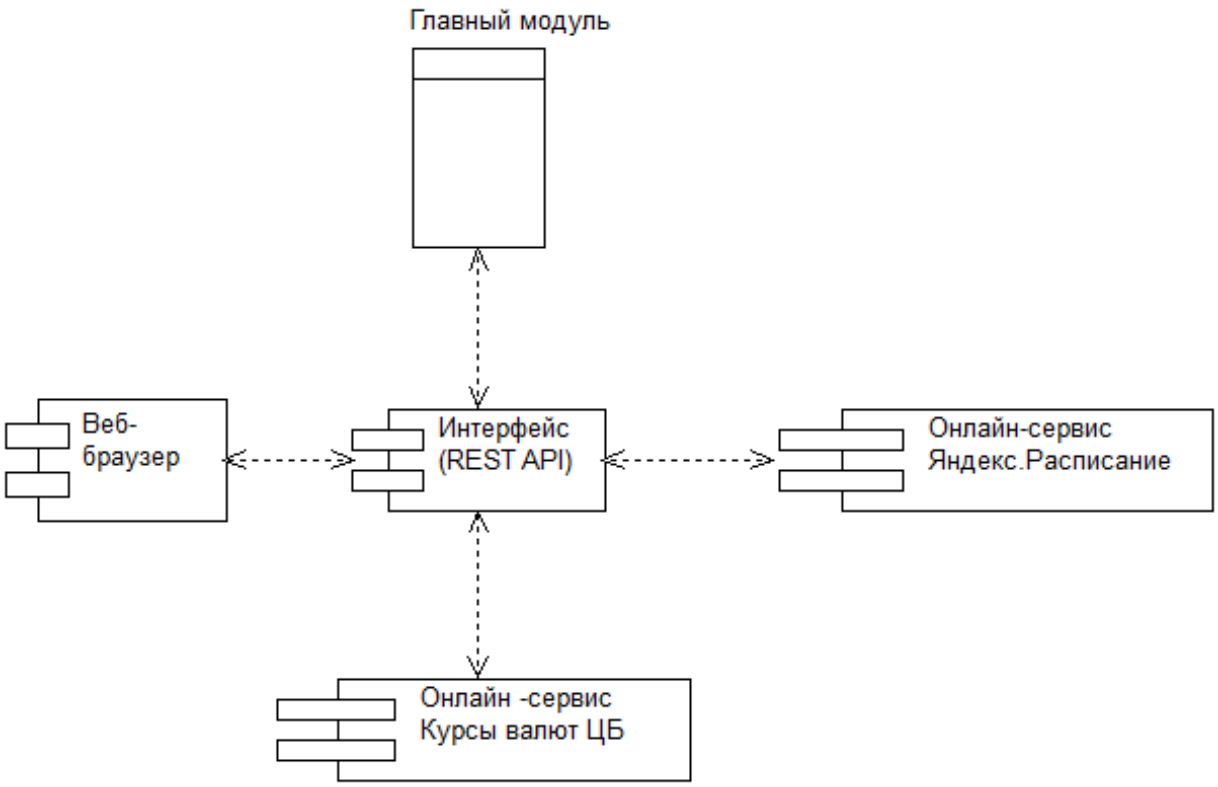

Рисунок 3.1 - Диаграмма компонентов web-приложения

Таким образом, в web-приложении используются два бесплатных онлайн-сервиса:

1) веб-сервис курсов валют ЦБ РФ [6].

С клиента посылается HTTP-запрос. Результатом является файл XML с курсом конкретной валюты на текущий день.

2) API Яндекс.Расписаний [19].

Взаимодействие с API организовано с помощью HTTPS-запроса в соответствии с принципами технологии REST.

Ответы от сервиса приходят в формате JSON.

В приложении А приведен фрагмент кода web-приложения, обеспечивающий взаимодействие с API Яндекс.Расписаний.

Как и в большинстве современных технологических платформ в 1С-Битрикс разработчик не имеет прямого доступа к БД приложения.

Создание нового объекта в БД осуществляется через интерфейсные формы web-приложения.

В таблицах 3.2-3.5 приведены описания структур основных объектов web-приложения.

Таблица 3.2 – Объект «Пользователь»

Атрибут	Тип
Код пользователя,	первичный ключ
Имя	строка
Фамилия	строка
Отчество	строка
Логин	строка
Пароль	строка
E-mail	строка
Код сотрудника	внешний ключ

Таблица 3.3 – Объект «Должность»

Атрибут	Тип
Код должности,	первичный ключ
Активность	булевое
Дата создания	дата
Дата изменения	дата
Наименование	строка

Таблица 3.4 – Объект «Сотрудник»

Атрибут	Тип
Код должности	первичный ключ
Активность	булевое

Продолжение таблицы 3.4

Дата создания	дата
Дата изменения	дата
ФИО	Строка
Код должности	внешний ключ

Таблица 3.5 – Объект «Транспортные расходы»

Атрибут	Тип
Код командировки	первичный ключ
Код вида расходов	внешний ключ
Активность	булевое
Дата создания	дата
Дата изменения	дата
Дата командировки	дата
Пункт назначения	строка
Код рейса	строка
Номер рейса	строка
Время вылета	строка
Код перевозчика	строка
Цена билета	деньги
Комментарий	текст

3.3 Описание работы web-приложения оценки командировочных расходов

Зарегистрированный и успешно прошедший авторизацию пользователь web-приложения открывает журнал командировок (рисунки 3.2, 3.3).

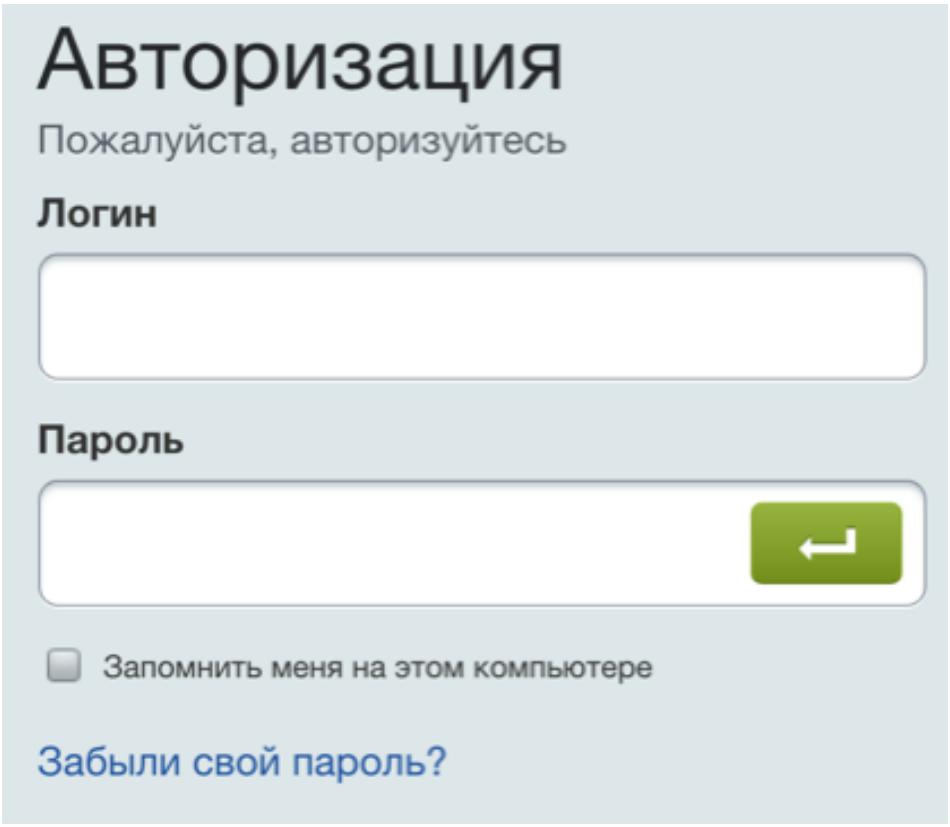

Рисунок 3.2 - Окно авторизации пользователя web-приложения

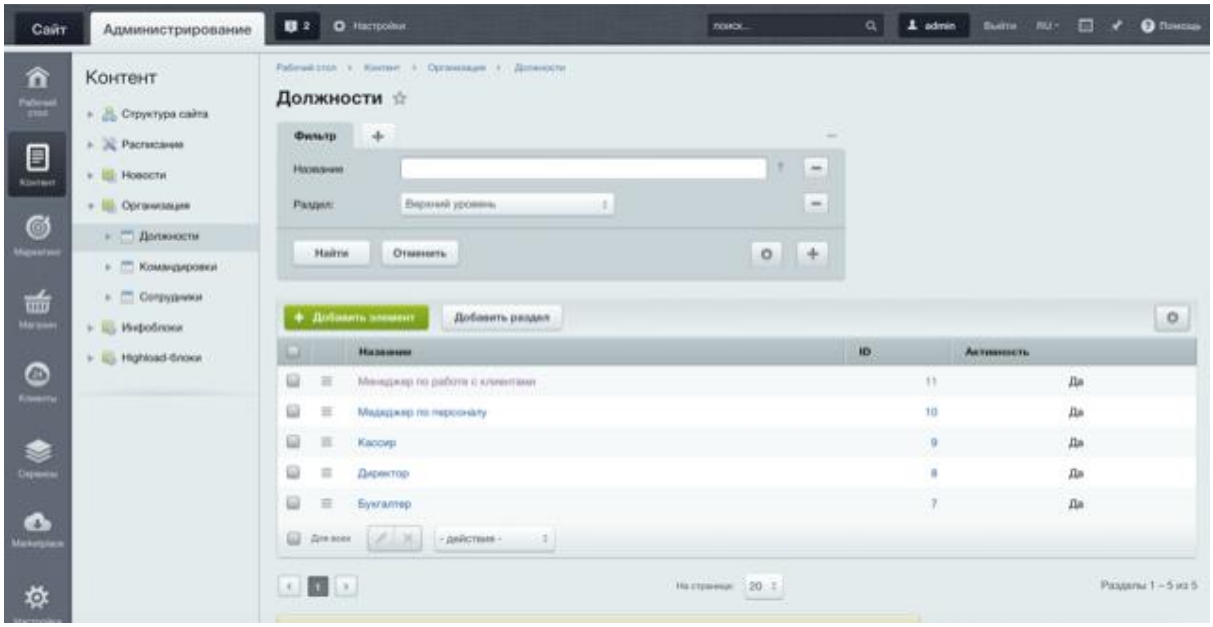

Рисунок 3.3 – Журнал командировок

Пользователь создает новый документ «Командировка» или активизирует в журнале уже существующую запись для редактирования (рисунок 3.4).

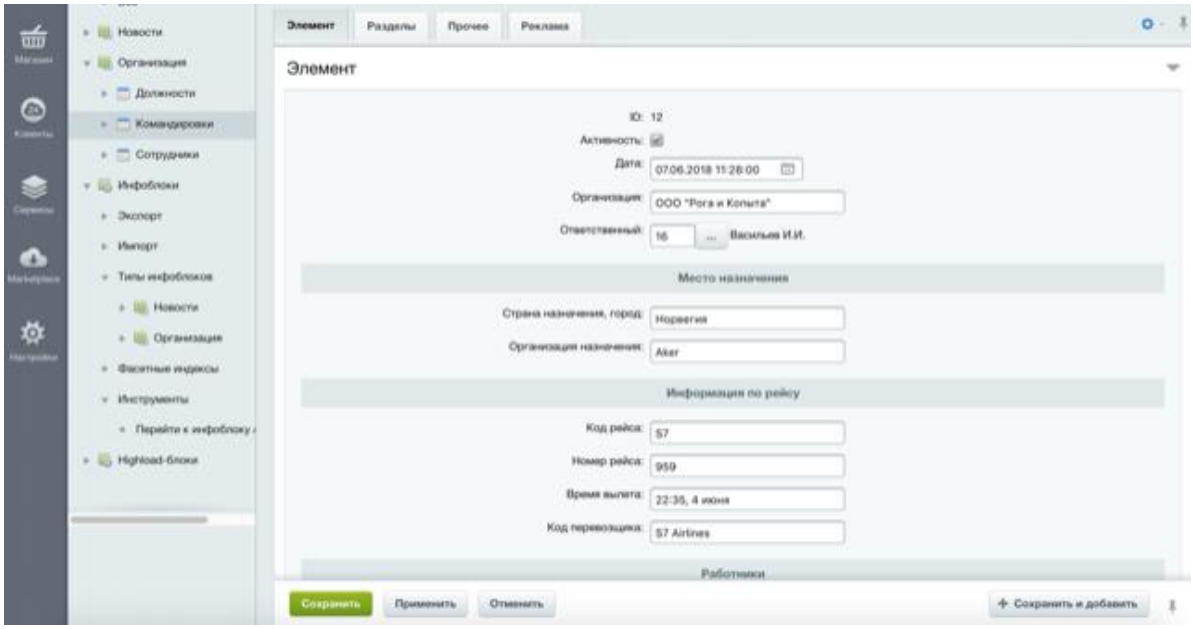

Рисунок 3.4 – Окно документа «Командировка»

Выбирается конкретный вид командировочных расходов (в рассматриваемом случае – авиабилеты), выбирается пункт назначения и активизируется онлайн-сервис Яндекс.Расписание.

Расписание авиарейсов выводится на экран в виде выпадающего списка

Пользователь выбирает аэропорт (станцию отправления).

При этом средствами ресурса выполняется автоматическая фильтрация данных, и на экран выводятся рейсы, относящиеся к выбранному аэропорту (станции отправления) (рисунок 3.5).

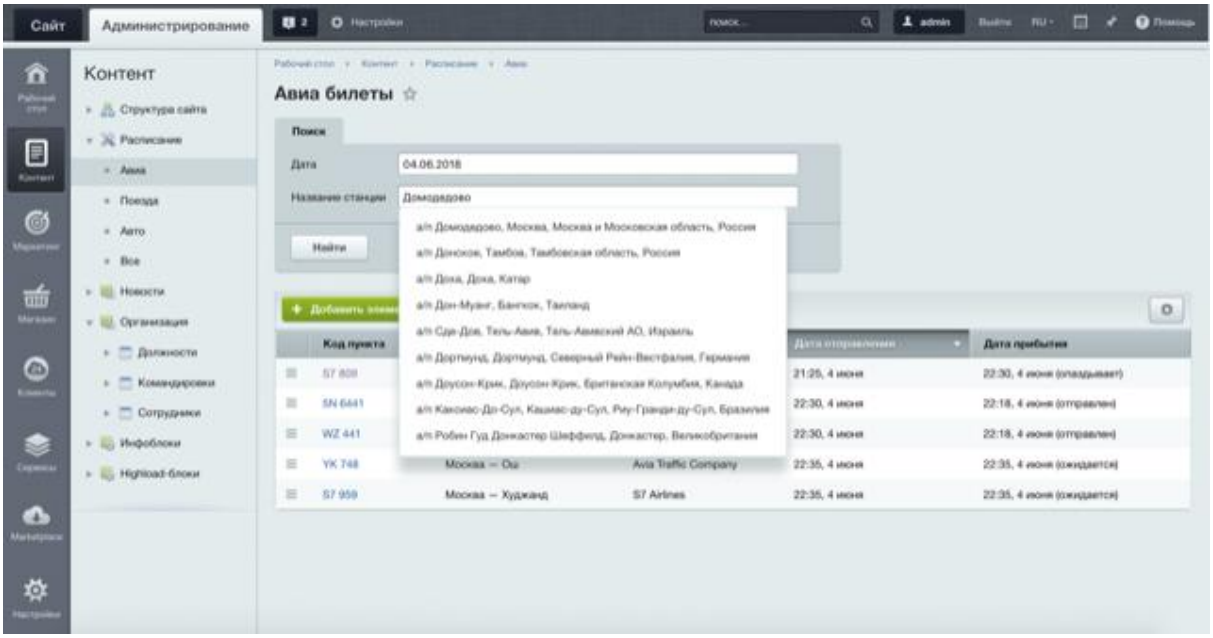

Рисунок 3.5 – Окно выбора авиарейса

Далее пользователь выбирает необходимый рейс и добавляет его в форму (рисунок 3.6).

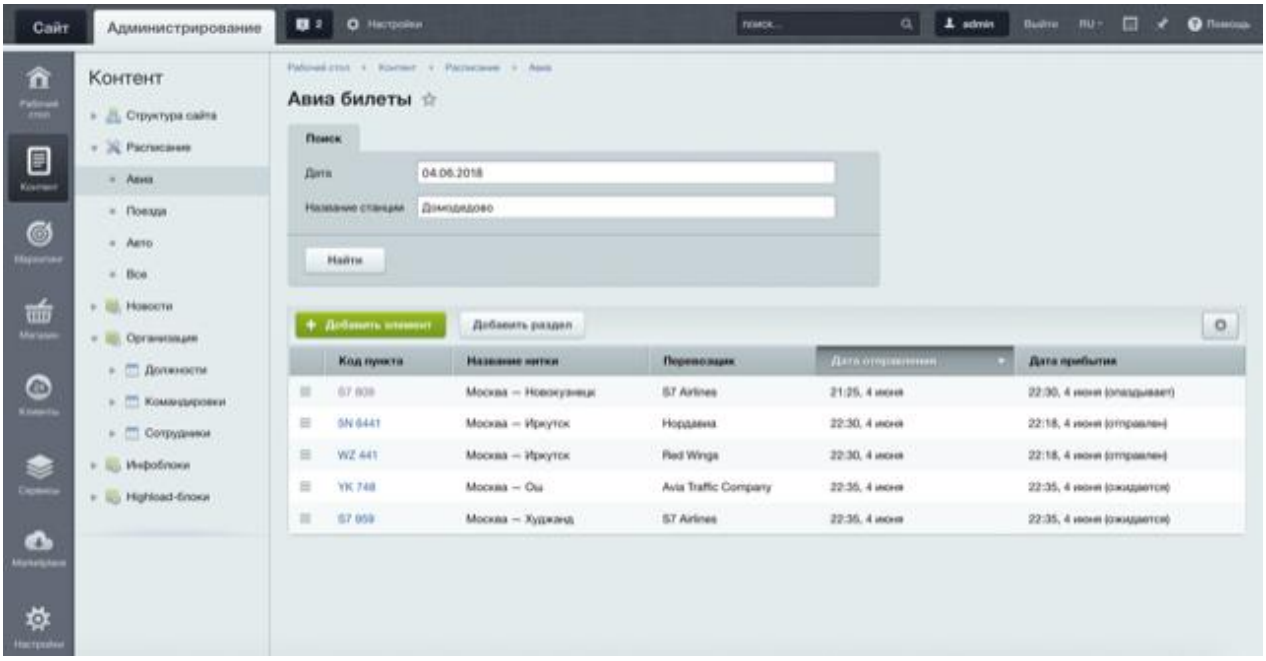

Рисунок 3.6– Окно списка выбранных рейсов

В приложении имеется возможность просмотра журнала активных пользователей (статус «Да») (рисунок 3.7).

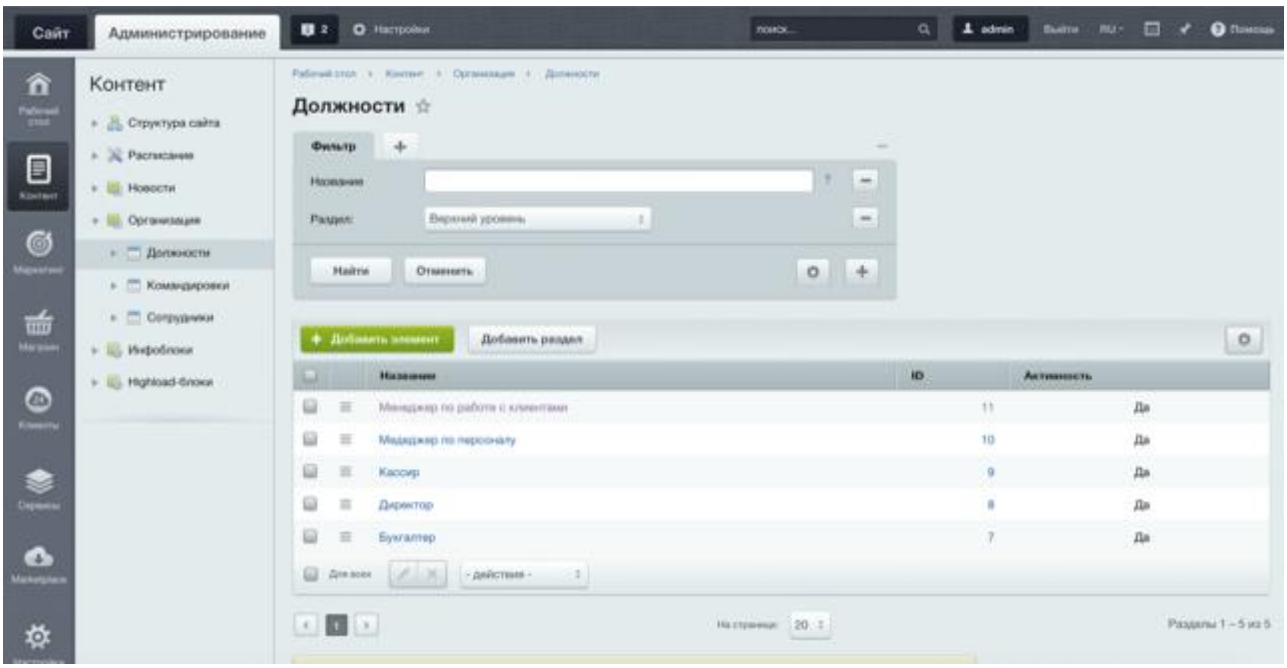

Рисунок 3.7 – Журнал активных пользователей web-приложения

В приложении имеется возможность экспорта журнала командировок в файл MS Excel для последующей обработки.

В web-приложении предусмотрен ввод данных по каждому виду расходов без использования онлайн-сервисов (режим оффлайн).

3.4. Оценка эффективности web-приложения оценки командировочных расходов

В качестве показателя оценки эффективности системы используем эффективность управления, под которой понимается степень полезности отдачи от реализации функций управления web-приложения [4].

Рассматривается несколько определений эффективности управления, такие, как целевая эффективность управления, функциональная эффективность управления и экономическая эффективность управления.

В рассматриваемом случае наиболее целесообразным представляется использование понятия функциональной эффективности управления, показатель которой может быть рассчитан с помощью следующей формулы:

$$K_{фэ} = \frac{\sum_{i=1}^{n} P_{yi}}{n},$$

где:

n - количество функций управления, реализуемых web-приложением;

P_{yi} - вероятность выработки web-приложением эффективного управляющего воздействия при реализации i-й функции управления.

В предлагаемом web-приложении реализовано 2 функции управления процессом оценки командировочных расходов:

− с использованием внешних онлайн-сервисов;

− с вводом данных в режиме оффлайн.

Единственной функцией, в которой принципиальное значение имеет человеческий фактор и, следовательно, существует вероятность ошибки при вводе данных, является работа в режиме оффлайн.

Тогда получим следующее значение показателя функциональной эффективности управления web-приложения:

$$К_{фэ} = 1/2 = 0.5$$

Если учесть, что режим работы оффлайн является не основным для разработанного web-приложения, то можно утверждать, что показатель функциональной эффективности управления web-приложения превышает значение 0.5, что соответствует требованиям, предъявляемым к системам управления в социально-экономической сфере.

Выводы к главе 3

1) Среди ключевых преимуществ 1С-Битрикс следует выделить большую популярность среди отечественных разработчиков web-приложений и возможность интеграции с технологической платформой «1С: Предприятие».

2) Как и в большинстве современных технологических платформ в 1С-Битрикс разработчик не имеет прямого доступа к БД приложения. Создание нового объекта в БД осуществляется через интерфейсные формы web-приложения.

ЗАКЛЮЧЕНИЕ

Тема бакалаврской работы посвящена актуальной проблеме разработки web-приложения оценки командировочных расходов на основе технологии СОА.

В ходе выполнения бакалаврской работы достигнуты следующие результаты:

− в технологии FURPS+ разработаны требования к web-приложению и на их основе произведен анализ известных ИТ-решений для оценки командировочных расходов. Обоснована целесообразность разработки нового web-приложения;

− для разработки web-приложения выбрана технология REST API:

− основе объектно-ориентированного подхода и языка UML разработана логическая модель web-приложения оценки командировочных расходов;

− в качестве средства реализации web-приложения web-приложения оценки командировочных расходов выбрана CMS «1С – Битрикс: Управление сайтом»;

− реализовано web-приложение оценки командировочных расходов и подтверждена его функциональная эффективность.

Разработанное web-приложение оценки командировочных расходов может быть рекомендовано для использования в различных сферах бизнеса и управления.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

Нормативно-правовые акты

1. ГОСТ 19.701-90. Единая система программной документации. Схемы алгоритмов, программ, данных и систем.

2. ГОСТ 34.602-89. Информационные технологии. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы.

3. «Трудовой кодекс Российской Федерации» от 30.12.2001 N 197-ФЗ (ред. от 05.02.2018).

Научная и методическая литература

4. Вдовин В.М. Предметно-ориентированные экономические информационные системы: Учебное пособие / В. М. Вдовин, Л. Е. Суркова, А. А. Шурупов. - М.: Издательско-торговая корпорация «Дашков и К°», 2013. — 388 с.

5. Биберштейн Н. и [др.] Компас в мире сервис-ориентированной архитектуры (SAO). - М.: КУДИЦ-Пресс, 2013. - 256 с.

Электронные ресурсы

6. Веб-сервис курсов валют ЦБ РФ [Электронный ресурс]. - Режим доступа: https://www.cbr.ru/development/DWS/ (дата обращения 05.06.2018).

7. Кириллов Ю.В. Прикладные методы оптимизации. Часть 1. Методы решения задач линейного программирования [Электронный ресурс] : учебное пособие / Ю.В. Кириллов, С.О. Веселовская. - Новосибирск: Новосибирский осударственный технический университет, 2012. - 235 с. - Режим доступа: http://www.iprbookshop.ru/45430.html (дата обращения 05.06.2018).

8. Компания «1С-Битрикс» [Электронный ресурс]. - Режим доступа: https://www.1c-bitrix.ru/ (дата обращения 05.06.2018).

9. Леоненков А.В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose. Курс лекций [Электронный ресурс]

: учебное пособие для студентов вузов, обучающихся по специальностям в области информационных технологий / А.В. Леоненков. - Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. — 318 с. - Режим доступа: http://www.iprbookshop.ru/67388.html (дата обращения 05.06.2018).

10. Методы принятия оптимальных решений. Часть 1 [Электронный ресурс] : учебное пособие / Р.М. Безбородникова [и др.]. - Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2016. - 245 с. - Режим доступа: http://www.iprbookshop.ru/69912.html (дата обращения 05.06.2018).

11. Облачное приложение «Командировочные расходы» [Электронный ресурс]. - Режим доступа: https://hamiltonapps.ru/kompensacija-rashodov-na-komandirovk (дата обращения 05.06.2018).

12. Онлайн-сервис для планирования командировок OneTwoTrip for Business [Электронный ресурс]. - Режим доступа: https://b2b.onetwotrip.com/?utm_source=startpack&utm_campaign=startpack&utm_medium=category&utm_content=booking-business-trips (дата обращения 05.06.2018).

13. Онлайн-система бронирования деловых поездок TopClient [Электронный ресурс]. - Режим доступа: https://startpack.ru/application/topclient (дата обращения 05.06.2018).

14. Ржихина Е.Г. Документальное подтверждение командировочных расходов [Электронный ресурс] / Е.Г. Ржихина. - Саратов: Ай Пи Эр Медиа, 2010. — 49 с. - Режим доступа: http://www.iprbookshop.ru/982.html (дата обращения 05.06.2018).

15. Самуйлов С.В. Объектно-ориентированное моделирование на основе UML [Электронный ресурс] : учебное пособие / С.В. Самуйлов. - Саратов: Вузовское образование, 2016. — 37 с. - Режим доступа: http://www.iprbookshop.ru/47277.html (дата обращения 05.06.2018).

16. Система управления контентом UMI.CMS [Электронный ресурс]. - Режим доступа: https://www.umi-cms.ru (дата обращения 05.06.2018).

17. Система управления сайтом Netcat [Электронный ресурс]. - Режим доступа: https://netcat.ru/ (дата обращения 05.06.2018).

18. Технология REST API [Электронный ресурс]. - Режим доступа: http://fb.ru/article/338879/rest-api---chto-eto-rest-perevod-representational-state-transfer (дата обращения 05.06.2018).

19. API Яндекс.Расписаний Электронный ресурс]. - Режим доступа: https://tech.yandex.ru/rasp/doc/concepts/about-docpage (дата обращения 05.06.2018).

20. Gruman G. SOA's Technology Underpinnings [Электронный ресурс] Режим доступа: https://www.cio.com/article/2446812/service-oriented-architecture/soa-s-technology-underpinnings.html (дата обращения 05.06.2018).

21. UML Diagram [Электронный ресурс]. - Режим доступа: https://www.smartdraw.com/uml-diagram (дата обращения 05.06.2018).

Литература на иностранном языке

22. Barker D. Web Content Management: Systems, Features, and Best Practices, O'Relly, 2016.

23. Boiko B. Content Management Bible, John Wiley & Sons, 2005, p. 1176.

24. Hailstone R., Illsley R., Jones T., Kellet A. SOA Platforms. – Butler Group, 2007.

25. Lengstorf J. PHP for Absolute Beginners, Apress, 2009, p. 387.

26. Watson M. Managing Smaller Projects: A Practical Approach. Multi-Media Publications Inc., 2006. p. 117.

ПРИЛОЖЕНИЕ А Фрагмент кода web-приложения

```
<?php
namespace Yandex\Stan\Schedule;
use GuzzleHttp\Client  as HttpClient;
use GuzzleHttp\Exception\ClientException;
use Yandex\Stan\Exceptions\YandexException;

class Client
{
    private $key;
    private $dataFormat           = self::DATA_FORMAT_JSON;
    private $lang                 = self::DATA_LANG_RU;
    private $apiUrl               = 'https://api.rasp.yandex.net/';
    const API_VERSION             = 'v1.0';
    const DATA_FORMAT_JSON        = 'json';
    const DATA_FORMAT_XML         = 'xml';
    const DATA_LANG_RU            = 'ru';
    const DATA_LANG_UK            = 'uk';
    const DATA_LANG_TR            = 'tr';
    const TRANSPORT_TYPE_PLANE    = 'plane';
    const TRANSPORT_TYPE_TRAIN    = 'train';
    const TRANSPORT_TYPE_SUBURBAN = 'suburban';
    const TRANSPORT_TYPE_BUS      = 'bus';
    const TRANSPORT_TYPE_SEA      = 'sea';
```

```
const TRANSPORT_TYPE_RIVER      = 'river';
const TRANSPORT_TYPE_HELICOPTER = 'helicopter';
const SYSTEM_YANDEX             = 'yandex';
const SYSTEM_IATA               = 'iata';
const SYSTEM_ICAO               = 'icao';
const SYSTEM_SIRENA             = 'sirena';
const SYSTEM_EXPRESS            = 'express';
const SYSTEM_ESR                = 'esr';
const ENDPOINT_SEARCH           = 'search';
const ENDPOINT_SCHEDULE         = 'schedule';
const ENDPOINT_THREAD           = 'thread';
const ENDPOINT_CARRIER          = 'carrier';
const ENDPOINT_NEAREST_STATIONS = 'nearest_stations';
const ENDPOINT_COPYRIGHT        = 'copyright';
const EVENT_ARRIVAL             = 'arrival';
const EVENT_DEPARTURE           = 'departure';
/**
 * @param string $key ключ API
 *
 * @see https://developer.tech.yandex.ru/
 */
public function __construct(string $key)
{
   $this->key = $key;
}
/**
 * Указание формата возвращаемых данных
 *
 * @param string $format json|xml
 */
```

```
    public function setDataFormat(string $format)
    {
        $this->dataFormat = $format;
    }
    /**
     * Язык возвращаемых данных
     *
     * @param string $lang Язык данных. Допустмые значения - ru|uk|tr
     */
    public function setLanguage(string  $lang)
    {
        $this->lang  = $lang;
    }

    /**
     * @param string $from Код станции отправления, например NYC (для аэропорта в Нью-Йорке)
     * @param string $to Код станции прибытия, например LED (для аэропорта Санкт-Петербурга в Пулково)
     * @param string $date Дата за которую вы хотите хотите получить список рейсов. Дата должна быть в формате "YYYY-MM-DD".  По умолчание возвращается список рейсов за все даты.
     * @param string $transportTypes Тип трансорта
     * @param string $system
     * @param int $page Страница данных (для пагинации)
     *
     * @see https://tech.yandex.ru/rasp/doc/reference/schedule-point-point-docpage/
     *
     * @throws YandexException
     * @throws ClientException
```

```
    *
    * @return string Data
    */
   public function getScheduleBetweenStations(string  $from, string $to, string
$transportTypes, string $system,
       string $date = '', int $page = 1) {
       $queryArray = [
           'from'           => $from,
           'to'             => $to,
           'date'           => $date,
           'transport_types' => $transportTypes,
           'system'         => $system,
           'page'           => $page,
       ];
       return $this->getData($this->getEndpointUrl(self::ENDPOINT_SEARCH,
$queryArray));
   }
   /**
    * @param string $station Departure station code, for example NYC (for New
York airport)
    * @param string $date Дата за которую вы хотите хотите получить список
рейсов. Дата должна быть в формате "YYYY-MM-DD".  По умолчание
возвращается список рейсов за все даты.
    * @param string $transportTypes Тип торанспортного средства
    * @param string $system
    * @param string $event  Событие графика
    * @param string $showSystems
    * @param string $direction
    * @param int $page Page of data
    *
```

```
     * @see https://tech.yandex.ru/rasp/doc/reference/schedule-on-station-docpage/
     *
     * @throws YandexException
     * @throws ClientException
     *
     * @return string Data
     */
    public function getScheduleOnStation(string $station, string $transportTypes, string $system,
        string $event = '', string $direction = '', string $showSystems = '',
        string $date = '', int $page = 1) {
        $queryArray = [
            'station'        => $station,
            'event'          => $event,
            'date'           => $date,
            'transport_types' => $transportTypes,
            'system'         => $system,
            'show_systems'   => $showSystems,
            'direction'      => $direction,
            'page'           => $page,
        ];
        return $this->getData($this->getEndpointUrl(self::ENDPOINT_SCHEDULE, $queryArray));
    }
    /**
     * @param string $uid Идентификатор нитки
     * @param string $showSystems
     * @param string $date Дата за которую вы хотите хотите получить список рейсов. Дата должна быть в формате "YYYY-MM-DD".  По умолчание возвращается список рейсов за все даты.
```

```
     *
     * @see https://tech.yandex.ru/rasp/doc/reference/list-stations-route-docpage/
     *
     * @throws YandexException
     * @throws ClientException
     *
     * @return string Data
     */
    public function getListStationsRoute(string $uid, string $showSystems = '', string $date = '')
    {
        $queryArray = [
            'uid'          => $uid,
            'date'         => $date,
            'show_systems' => $showSystems,
        ];
        return $this->getData($this->getEndpointUrl(self::ENDPOINT_THREAD, $queryArray));
    }
    /**
     * Получение перевозщика
     *
     * @param string $code Код перевозщика
     * @param string $system
     *
     * @see https://tech.yandex.ru/rasp/doc/reference/query-carrier-docpage/
     *
     * @throws YandexException
     * @throws ClientException
     *
```

```
     * @return string Data
     */
    public function getCarrier(string  $code, string $system = '')
    {
        $queryArray = [
            'code'   => $code,
            'system' => $system,
        ];
        return $this->getData($this->getEndpointUrl(self::ENDPOINT_CARRIER,
$queryArray));
    }

    /**
     * Получение списка ближайших станций
     *
     * @param string $lat Шарота согласно WGS84
     * @param string $lng Долгота согласно WGS84
     * @param integer $distance Радиус в километрах (от 0 до 50).
     * @param string $stationType
     * @param string $transportTypes Тип торанспортного средства
     * @param int $page Страница данных
     *
     * @see https://tech.yandex.ru/rasp/doc/reference/query-nearest-station-docpage/
     *
     * @throws YandexException
     * @throws ClientException
     *
     * @return string Data
     */
    public function getNearestStations(string $lat, string $lng, int $distance,
```

```
        string $stationType = '', string $transportTypes = '',
        int $page = 1) {
        $queryArray = [
            'lat'            => $lat,
            'lng'            => $lng,
            'distance'       => $distance,
            'station_type'   => $stationType,
            'transport_types' => $transportTypes,
            'page'           => $page,
        ];
        return $this->getData($this-
>getEndpointUrl(self::ENDPOINT_NEAREST_STATIONS, $queryArray));
    }
    /**
     * Получение данных по запросу
     *
     * @param string $url Возвразает полный урл адрес с параметрами
     *
     * @throws YandexException
     * @throws ClientException
     *
     * @return string Response body
     */
    protected function getData(string $url): string
    {
        $client = new HttpClient();
        try {
            $response = $client->get($url);
        } catch (ClientException $e) {
            $response = $e->getResponse();
```

```
            $responseData = $response->getBody()->getContents();
            if ($this->dataFormat == self::DATA_FORMAT_XML) {
                $xml          = simplexml_load_string($responseData);
                $responseData = json_encode($xml, JSON_UNESCAPED_UNICODE);
            }
            $dataArray = json_decode($responseData, true);
            if ($dataArray['error'] && $dataArray['error']['text']) {
                throw new YandexException($dataArray['error']['text']);
            } else {
                throw $e;
            }
        }
        return $response->getBody();
    }

    /**
     * Отправляет запрос
     * @param string $type Тип получаемых данных
     * @param array $dataArray Дополнительные данные
     * @return string Full end-point URL
     */
    protected function getEndpointUrl(string $type, array $dataArray = []): string
    {
        $settingsArray = [
            'lang'   => $this->lang,
            'format' => $this->dataFormat,
            'apikey' => $this->key,
        ];
        $url = $this->apiUrl . static::API_VERSION . DIRECTORY_SEPARATOR .
$type . DIRECTORY_SEPARATOR;
```

```
        return $url . '?' . http_build_query(array_merge($settingsArray, $dataArray));
    }
}
```