УДК 004.42

И. Р. Кузнецова
Санкт-Петербургский государственный архитектурно-строительный университет

А. С. Букунов
Санкт-Петербургский политехнический университет Петра Великого

С. В. Букунов
Санкт-Петербургский государственный архитектурно-строительный университет

Автоматизированная система организации деятельности клуба активных путешествий

Описывается разработанная на кафедре информационных технологий Санкт-Петербургского архитектурно-строительного университета автоматизированная система, позволяющая существенно упростить организацию деятельности клуба активных путешествий INSLED. Работа выполнялась на основании технического задания, выданного руководством клуба, и передана в клуб для опытной эксплуатации. Система ориентирована на пользователей двух категорий – внешних (неавторизованных) и внутренних (авторизованных). Внутренние пользователи, в свою очередь, подразделяются на три типа в зависимости от их статуса: участники, организаторы и администраторы. Каждая группа пользователей обладает своим набором возможностей при работе с приложением. Несмотря на то что система разработана для конкретного клуба, реализованные в ней подходы и решения могут быть успешно использованы при создании подобных систем для других организаций, занимающихся схожими видами деятельности. Система реализована в виде веб-приложения, написанного на языке Python с помощью библиотеки Django. Для хранения информации о проводимых клубом мероприятиях, участниках мероприятий, клубных картах и т. п. была спроектирована и реализована реляционная база данных (БД), работа с которой осуществляется с помощью системы управления базами данных (СУБД) MySQL. Для решения ряда задач использовались языки HTML, CSS, JavaScript, технология AJAX. Работа с системой не требует установки дополнительного программного обеспечения (ПО) – для этого достаточно наличия доступа к сети Интернет.

Автоматизация бизнес-процессов, автоматизированная информационная система, веб-приложение, база данных, объектно-ориентированное программирование

В настоящее время в условиях бурного развития информационных технологий многие организации из различных сфер деятельности ищут новые возможности как для продвижения производимых ими товаров и услуг, так и для автоматизации своих бизнес-процессов. К наиболее популярным способам решения задач такого рода можно отнести создание собственного веб-приложения. Наличие такого приложения позволяет компании привлечь новых клиентов, оптимизировать работу сотрудников, увеличить эффективность управления, повысить оперативность и снизить расходы. Неоспоримое преимущество веб-приложения состоит в возможности работы с ним из любой точки мира и с любого устройства, будь то настольный компьютер, ноутбук, планшет или телефон, необходимы лишь браузер и доступ к сети Интернет.

Данное приложение разработано по просьбе клуба активных путешествий INSLED, организующего для всех желающих различные активные мероприятия (велопоездки, пешие походы, лыжные походы, интеллектуальные игры и проч.). Весь

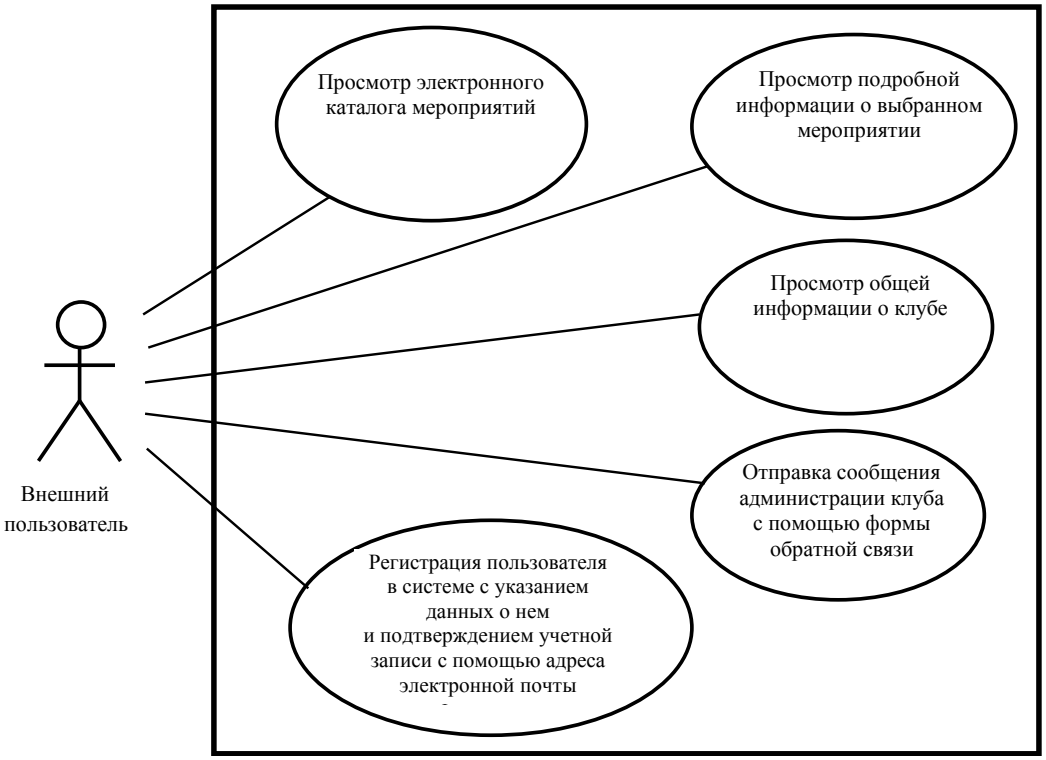

Рис. 1

управленческий учет деятельности клуба осуществлялся с помощью электронных таблиц Excel. По мере накопления информации такой способ ее обработки с каждым годом требовал все больших затрат, что и привело к необходимости разработки новой автоматизированной системы.

Основная функциональность приложения. С приложением работают пользователи двух типов:

1) внешние (неавторизованные);

2) внутренние (авторизованные).

Взаимодействие пользователей с информационной или иной системой принято отображать с помощью диаграмм вариантов использования на основе унифицированного языка моделирования UML (Unified Modeling Language) [1], таблиц с описанием сценариев взаимодействия пользователей с системой или текстового описания. UML-диаграмма для внешних пользователей приведена на рис. 1.

Внутренние пользователи могут быть трех видов:

1) участник;

2) организатор;

3) администратор.

Пользователь, имеющий статус *участника*, может:

– просматривать электронный каталог всех мероприятий;

– регистрироваться в качестве участника планируемого мероприятия;

– отменять свое участие в планируемом мероприятии;

– просматривать информацию о мероприятиях, в которых пользователь участвовал или записан в качестве потенциального участника;

– добавлять и редактировать личную информацию;

– при наличии клубной карты – просматривать информацию о ней, при отсутствии клубной карты – просматривать информацию о местонахождении спрятанных клубных карт (клубные карты могут быть спрятаны организаторами клуба в любой точке мира по системе геокэшинга).

Пользователь, имеющий статус *организатора*, может:

– просматривать электронный каталог всех мероприятий;

– регистрироваться в качестве участника планируемого мероприятия, созданного другим организатором;

– отменять свое участие в планируемом мероприятии, созданным другим организатором;

– создавать новое мероприятие;

– редактировать информацию о мероприятии, организатором которого он является;

– просматривать список участников, зарегистрированных на мероприятие, организатором которого он является;

– регистрировать участников на мероприятие, организатором которого он является;

– удалять участников из списка участников мероприятия, организатором которого он является;

– просматривать список зарегистрированных клубных карт;

– регистрировать новые клубные карты;

– редактировать информацию о клубных картах;

– просматривать информацию о том, какие карты находятся у организатора на хранении;

– просматривать информацию о мероприятиях, в которых участник участвовал или записан в качестве потенциального участника;

– просматривать информацию обо всех мероприятиях, организатором которых он являлся;

– добавлять и редактировать личную информацию.

Пользователь, имеющий статус *администратора*, может:

– добавлять новых пользователей;

– редактировать сведения обо всех существующих пользователях;

– удалять существующих пользователей;

– создавать новое мероприятие;

– редактировать информацию о созданных мероприятиях;

– удалять существующие мероприятия;

– регистрировать организатора мероприятия;

– удалять пользователя из списка организаторов мероприятия;

– регистрировать участников мероприятия;

– удалять пользователя из списка участников мероприятия;

– регистрировать новые клубные карты;

– редактировать информацию о клубных картах;

– удалять клубные карты;

– добавлять типы мероприятий;

– удалять типы мероприятий.

База данных. Для хранения всех данных, необходимых для работы приложения, была спроектирована и реализована реляционная база данных [2]. Одним из наиболее удобных инструментов унифицированного представления данных является ER-модель (Entity-relationship model), которая представляет собой схему «сущность – связь» [3].

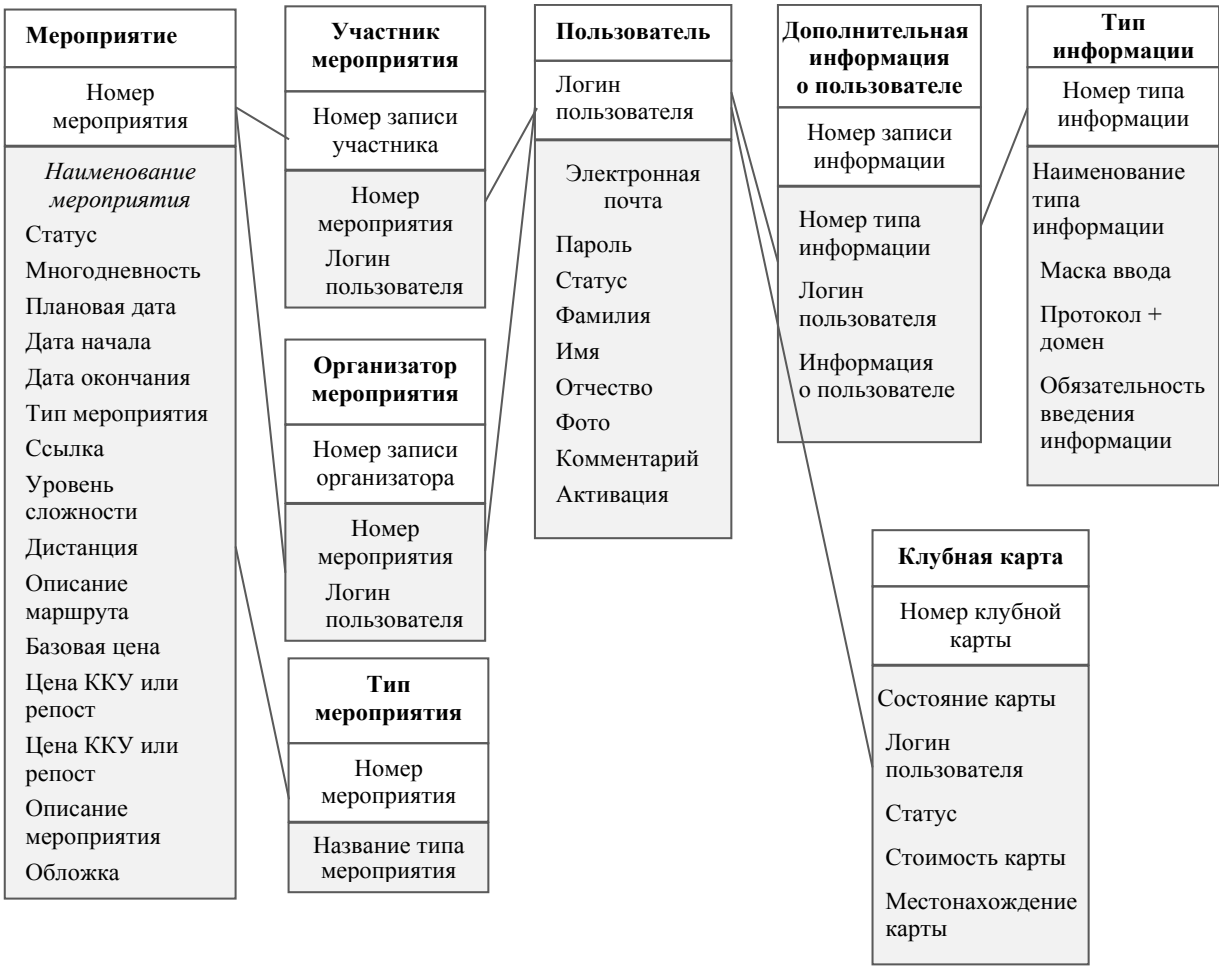

Рис. 2

При создании базы данных было выделено восемь сущностей (или таблиц БД):

1) пользователь;
2) тип информации;
3) дополнительная информация о пользователе;
4) клубная карта;
5) мероприятие;
6) тип мероприятия;
7) участник мероприятия;
8) организатор мероприятия.

ER-модель разработанной базы данных представлена на рис. 2.

Используемые технологии. Веб-приложение разработано на языке Python [4] с помощью библиотеки Django [5], [6] и состоит из клиентской и серверной частей, реализуя технологию «клиент–сервер».

Клиентская часть веб-приложения – это пользовательский интерфейс, который формирует запросы к серверу и обрабатывает ответы от него. В качестве основного средства для разработки клиентской части веб-приложения использовался язык разметки документов HTML [7]. В качестве вспомогательных инструментов для упрощения оформления и интерактивности использованы формальный язык описания внешнего вида документа CSS [7] и язык программирования JavaScript (JQuery) [8]. Наряду с ними для обмена информацией с сервером асинхронными запросами применялась технология AJAX [9].

При написании клиентской части приложения был использован фреймворк Bootstrap [10], который содержит в себе CSS- и HTML-шаблоны оформления компонентов веб-интерфейса, включая расширения JavaScript. Bootstrap основан на сеточной системе, удобной для адаптивной разработки.

Серверная часть веб-приложения – это программа или скрипт на сервере, обрабатывающая запросы браузера. Серверная часть получает запрос от клиента, производит необходимые вычисления, после чего формирует веб-страницу и отправляет ее клиенту по сети, используя протокол HTTP (HyperText Transfer Protocol). Серверная часть приложения реализована на базе встроенного в Django административного сайта.

Для работы с базой данных использовалась система управления базами данных MySQL [11].

Основные алгоритмы. При проектировании приложения был использован паттерн проектирования MVT (Model-View-Template), который реализован в библиотеке Django и представляет собой модификацию паттерна проектирования MVC (Model-View-Controler). Схема архитектуры MVT представлена на рис. 3.

Паттерн проектирования MVT позволяет разделять данные приложения, пользовательского интерфейса и логику на отдельные компоненты:

– URLS – при получении запроса определяет, какое представление должно его обрабатывать;

– VIEW – функция обработчика запросов; если для обработки запроса необходимо обращение к модели и базе данных, то VIEW взаимодействует с ними; для создания ответа применяются шаблоны;

– MODEL – описывает данные, используемые в приложении, и представляет собой совокупность классов, написанных на языке Python, каждый из которых соответствует определенной таблице базы данных;

– TEMPLATE – текстовый файл, определяющий структуру или разметку страницы.

Представления-классы, хранящиеся в файле views.py, унаследованы от базовых представлений-

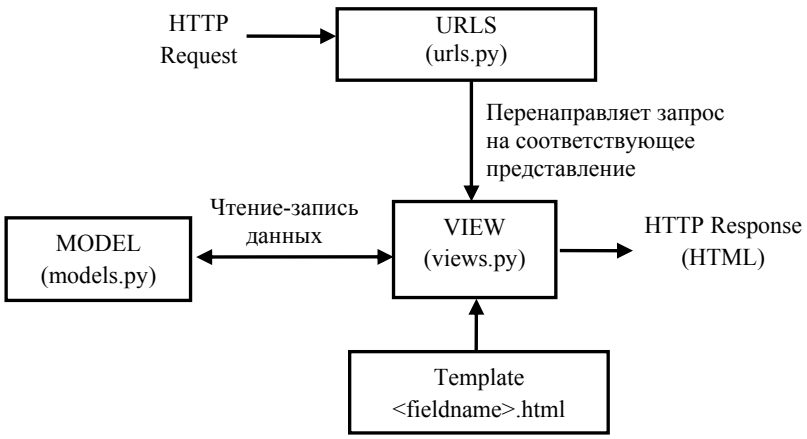

Рис. 3

классов (Classed-based views), объявленных в модуле django.views.generic.base. Следует отметить, что для взаимодействия с клиентами можно также использовать функции вместо классов, однако в данной работе был реализован именно подход с классами, позволяющий пользоваться различными возможностями объектно-ориентированного программирования – наследование, примеси и др.

В Django реализовано большое количество классов, которые заметно облегчают разработку классов представлений для различных целей.

Например, в личном кабинете любой пользователь должен иметь возможность изменять свои личные данные. Для совершения данной процедуры необходимо проверять пользователя на авторизацию, извлекать данные из базы данных о нем, заполнять ими форму для редактирования, а затем после редактирования пользователем полей формы извлекать из нее данные, изменять данные о пользователе в базе данных и выводить сообщение об успешном изменении данных. При использовании функций вместо классов для решения этой задачи пришлось бы написать функцию, состоящую из огромного количества строк кода. Объектно-ориентированный подход позволяет существенно сократить размер кода.

В данной работе для обновления данных пользователя был разработан класс ChangeUserInfoView. При его разработке было использовано множественное наследование, а именно класс был унаследован от базовых представлений классов Django – таких, как LoginRequiredMixin, SuccessMessageMixin и UpdateView. В разработанном классе были переопределены два метода базового класса: dispatch() для получения уникального идентификатора пользователя и get_object() для получения данных о пользователе.

Пример работы данного класса-представления показан на рис. 4.

Библиотека Django предоставляет встроенную систему аутентификации пользователя, однако в данной работе ее потребовалось расширить, поскольку возникла необходимость хранения данных о пользователях, не предусмотренных пользовательской моделью User системы аутентификации.

Класс User наследует абстрактному классу AbstractUser, объявленному в модуле django.contrib.auth.models. Класс AbstractUser обладает той же функциональностью и теми же атрибутами, что и класс User.

Поскольку атрибутов модели User оказалось недостаточно для хранения всех необходимых данных о пользователях, она была заменена созданной моделью avduser, также унаследованной от класса AbstractUser.

При создании модели avduser были добавлены новые атрибуты в соответствии с разработанной схемой базы данных, а также присвоены дополнительные параметры модели уже существующим атрибутам, унаследованным от модели AbstractUser. Так, например, с помощью дополнительного параметра модели была присвоена уникальность атрибуту email. Это позволило реализовать возможность корректного восстановления

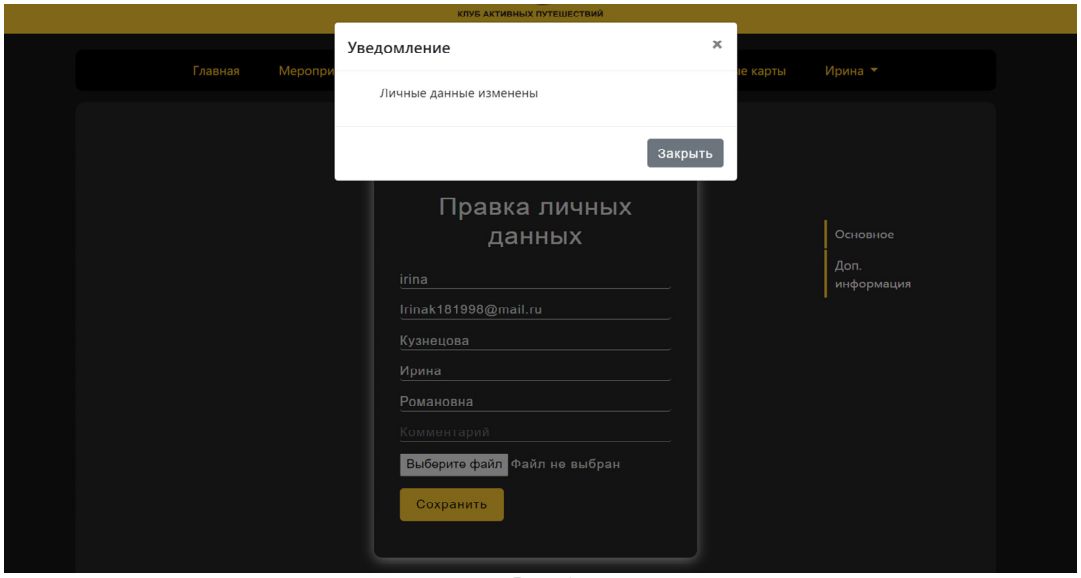

Рис. 4

пароля при наличии в базе данных нескольких пользователей с одинаковыми адресами электронной почты. Использование в таких ситуациях атрибутов стандартной модели User могло привести к ситуации, когда система не знала бы, какой из профилей требуется восстановить, что могло привести к ошибкам.

Для реализации ряда функций был использован подход к построению интерактивных пользовательских интерфейсов веб-приложений, называемый AJAX (Asynchronic JavaScript and XML). Главное отличие AJAX от стандартного подхода заключается в обращении к серверу без перезагрузки страницы, за счет чего значительно уменьшается время отклика.

В частности, с помощью этой технологии в данном приложении был реализован алгоритм регистрации на мероприятие нескольких участников без перезагрузки страницы. Для реализации алгоритма был написан соответствующий класс на языке Python, который добавляет записи о новом участнике мероприятия в базу данных, и с помощью набора функций JQuery языка JavaScript написана функция-обработчик события (нажатие на кнопку «Добавить участника»).

При создании пользовательского интерфейса использовалась библиотека bootstrap4, содержа-

Рис. 5

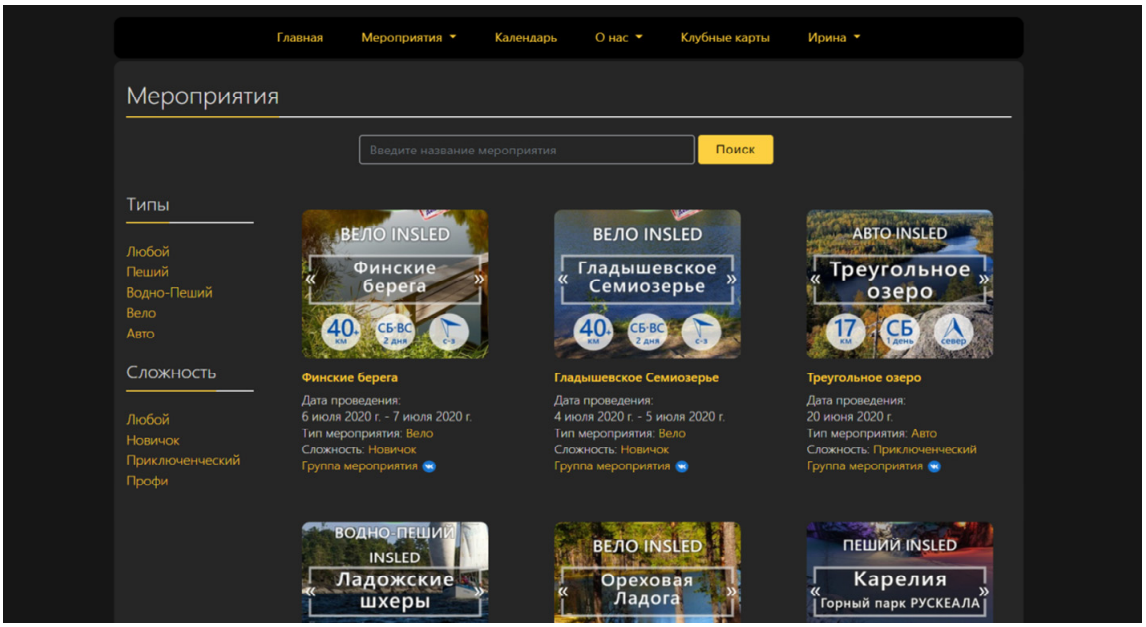

Рис. 6

Рис. 7

щая различные CSS- и HTML-шаблоны. Во всех шаблонах проекта использовался механизм наследования от базового шаблона, отображающего верхнее меню, шапку и подвал сайта.

Результаты. Далее приводятся некоторые примеры работы приложения для различных групп пользователей.

Функциональные возможности внешнего пользователя. Неавторизованный пользователь, попадая на главную страницу, имеет возможность перейти на различные статические страницы, которые несут информацию о клубе активных путешествий, а также на динамическую страницу с каталогом мероприятий.

На рис. 5 изображена главная страница веб-приложения, а на рис. 6 – каталог мероприятий.

Каталог мероприятий отсортирован по датам в порядке убывания и имеет постраничную навигацию. Пользователь имеет возможность фильтровать каталог мероприятий по типам и уровню сложности, а также может перейти на страницу с подробной информацией о выбранном мероприятии. Кроме того, поля «Типы» и «Сложность» реализованы в виде динамических ссылок, т. е. при нажатии на это поле осуществляется переход на страницу со всеми мероприятиями этого типа или уровня сложности.

На странице мероприятия присутствует кнопка «Стать участником», которая перенаправляет неавторизованных пользователей на страницу с формой авторизации. Кроме того, пользователь может отправить сообщение администрации клуба с помощью формы обратной связи, указав свои имя и электронную почту.

Функциональные возможности пользователя, имеющего статус участниика. Авторизованный пользователь, имеющий статус участника, помимо функциональных возможностей неавторизованного пользователя имеет ряд дополнительных возможностей.

Участник обладает личным кабинетом, в котором отображена основная и дополнительная информация о пользователе, может увидеть номер своей клубной карты (при ее наличии) и последние два мероприятия, в которых он участвовал или зарегистрирован в качестве участника. При отсутствии у участника клубной карты он может увидеть в своем личном кабинете в разделе «клубные карты» номер и местонахождение скрытой клубной карты. При каждом обновлении страницы выводятся данные о новой скрытой клубной карте, отличной от предыдущей.

Личный кабинет участника представлен на рис. 7.

С помощью специальной формы участник может изменять основную и дополнительную информацию о себе, а также пароль.

Кроме того пользователь имеет возможность зарегистрироваться на мероприятие и отменить свое участие в мероприятии в качестве участника, если оно имеет статус «В разработке».

На отдельной странице «Мои мероприятия» участник может увидеть все мероприятия, в которых он участвовал или зарегистрирован в качестве участника.

Рис. 8

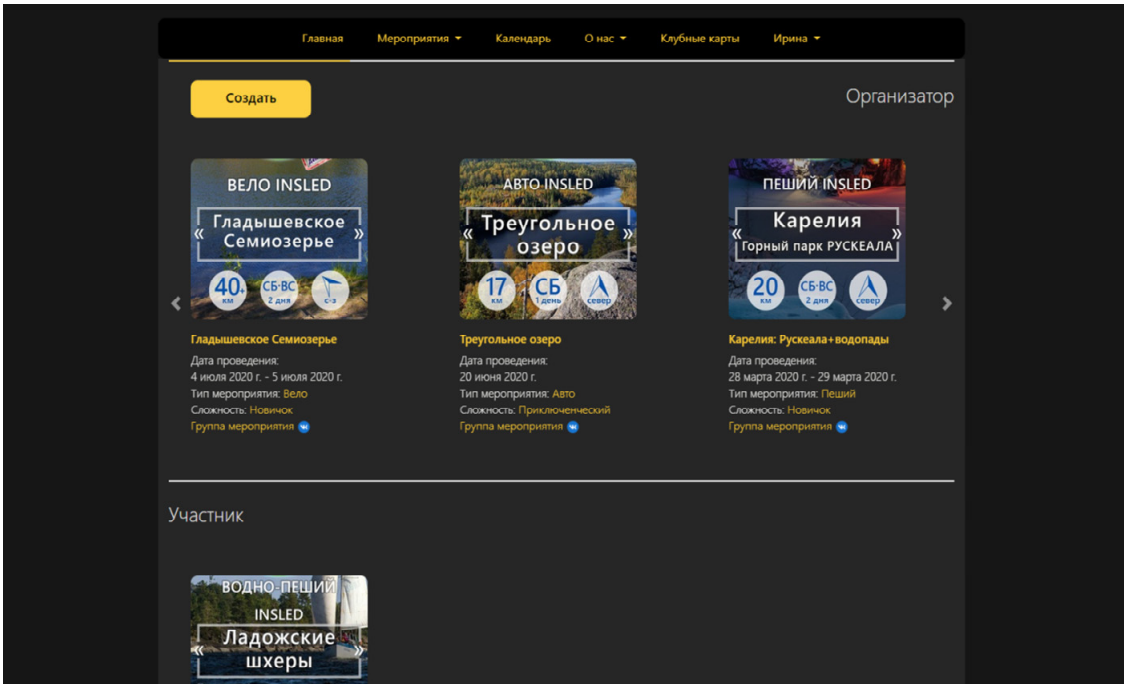

Рис. 9

Функциональные возможности пользователя, имеющего статус организатора. Организатор также имеет личный кабинет. В нем отображается основная и дополнительная личная информация организатора, два последних его мероприятия (планируемых или проведенных), а также информация о том, какие клубные карты находятся у него на хранении. Организатор может редактировать клубные карты и регистрировать новые. Для нахождения клубной карты по номеру существует панель поиска.

На рис. 8 представлена страница «Клубные карты», доступная организатору.

Организатор имеет возможность с помощью специальной формы создавать новое мероприятие и редактировать мероприятие, созданное им же и имеющее статус «В разработке».

Организатор может регистрировать пользователей в качестве участников своего мероприятия и удалять пользователей из списка участников. Список участников мероприятия может меняться до нажатия организатором кнопки «Закрыть мероприятие». С ее помощью организатор меняет

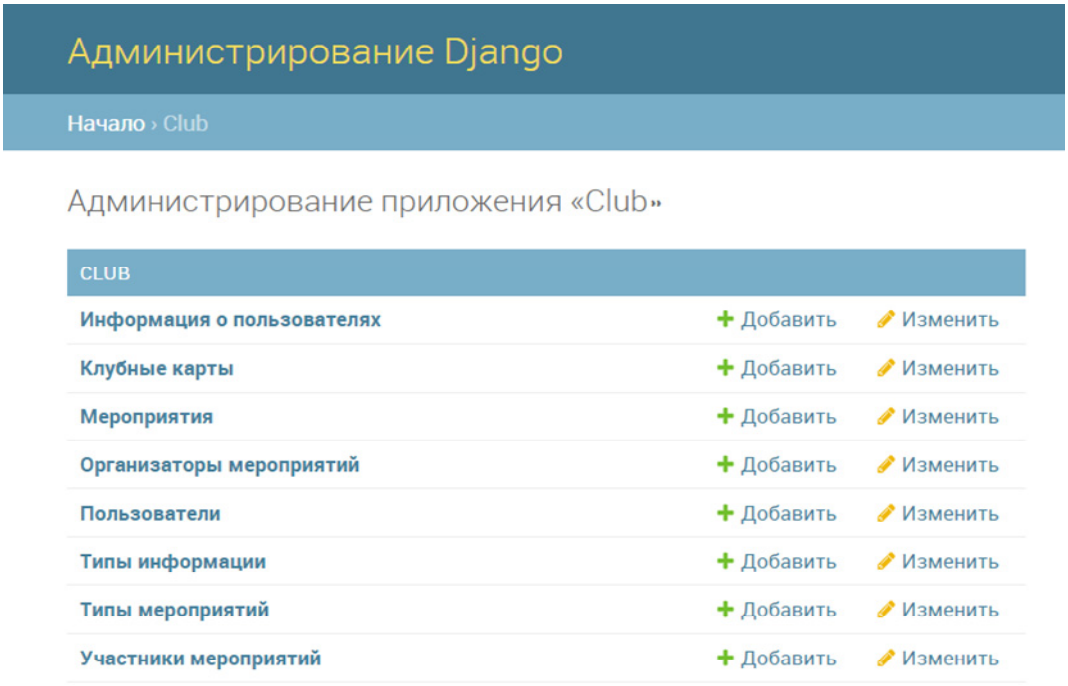

Рис. 10

статус мероприятия с «В разработке» на «Утверждено».

Организатор может стать участником мероприятия, организуемого другим организатором.

На рис. 9 представлена страница организатора «Мои мероприятия», на которой отображены как организуемые им мероприятия, так и те, в которых он принимает участие.

Функциональные возможности пользователя, имеющего статус администратора. Администратор с помощью специальной панели администрирования (рис. 10) имеет доступ к любой информации.

Как видно на рисунке, каждый пункт панели администрирования соответствует определенной сущности (таблице) базы данных (см. рис. 2). При выборе любого пункта администратор попадает на специальную вкладку, где у него имеется возможность создавать, просматривать, редактировать и удалять соответствующую информацию (о мероприятиях, об участниках, о клубных картах и т. д.). Дополнительно на всех вкладках реализованы режимы фильтрации и поиска нужной информации.

С помощью библиотеки Django и языков программирования Python и JavaScript разработано веб-приложение для автоматизации организации деятельности клуба активных путешествий. Несмотря на то что приложение разрабатывалось для конкретного клуба, используемые при его создании подходы и технологии могут быть применены для аналогичных продуктов в схожих сферах деятельности.

СПИСОК ЛИТЕРАТУРЫ

1. Флегонтов А. В., Матюшичев И. Ю. Моделирование информационных систем. Unified Modeling Language: учеб. пособие. 2-е изд. СПб.: Лань, 2019.

2. Фомичева Т. Г. Базы данных. Проектирование приложений реляционных БД: конспект лекций. Ч. 1. СПб.: Изд-во СПбЭТУ «ЛЭТИ», 2008.

3. Тарасов С. В. СУБД для программиста. Базы данных изнутри. Омск: Соломон, 2015.

4. Златопольский Д. М. Основы программирования на языке Python. М.: ДМК-Пресс, 2018.

5. Дронов В. А. Django 2.1. Практика создания веб-сайтов на Python. СПб.: БХВ-Петербург, 2019.

6. Меле А. Django 2 в примерах. М.: ДМК-Пресс, 2019.

7. Дакетт Дж. HTML и CSS. Разработка и дизайн веб-сайтов. М.: Эксмо, 2020.

8. Никольский А. П. JavaScript на примерах. Практика, практика и только практика. СПб.: Наука и техника, 2018.

9. Браун Э. Изучаем JavaScript. Руководство по созданию современных веб-сайтов. М.: Альфа-книга, 2017.

10. Моретто С. Bootstrap в примерах. М.: ДМК-Пресс, 2017.

11. Шварц Б., Ткаченко В., Зайцев П. MySQL по максимуму. Оптимизация, репликация, резервное копирование. СПб.: Питер, 2018.

I. R. Kuznetsova
Saint Petersburg State University of Architecture and Civil Engineering

A. S. Bukunov
Peter the Great St. Petersburg Polytechnic University

S. V. Bukunov
Saint Petersburg State University of Architecture and Civil Engineering

AUTOMATED SYSTEM OF ACTIVE TRAVEL CLUB PROCESS ORGANIZATION

An automated system developed at the Department of Information Technologies of the Saint Petersburg State University of Architecture and Civil Engineering, which makes it possible to significantly simplify the work of the active travel club INSLED, is described. The work was carried out according to the technical specification given by the club management and transferred to the club for trial operation. The system is aimed at users of two categories: external users (not authorized) and internal users (authorized). External users in turn, are divided into three categories depending on their status: participants, organizers and administrators. Each user group has its own set of features when working with the application. Despite the fact that the system is being developed for the particular club INSLED, the approaches and technological solutions implemented in it can be successfully used to create similar systems for other organizations and companies involved in similar activities. The system is implemented as a web-application developed by Python programming language and Django framework. Relational data base was designed and implemented for data storage. MySQL data base management system (DBMS) is used to organize interaction with the database. To solve a number of problems, the such languages as HTML, CSS, JavaScript (JQuery) and AJAX technology were used. The system does not require the installation of additional software. Access to the Internet needs to use the system only.

Business process automation, computer-aided information system, web-application, data base, object-oriented programming

УДК 519.764

А. Н. Субботин, Н. А. Жукова
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

Возможности обработки видеоинформации в системе интеллектуального видеонаблюдения в туманных средах с применением концепции интернета вещей

Рассматриваются методы и средства обработки видеоинформации в туманных средах с применением нейронных сетей и концепции интернета вещей. Рассмотрены основные аспекты, определяющие выбор средств обработки для оператора видеонаблюдения с учетом особенностей стоящих перед ним задач. Предложены решения по повышению скорости обработки видеоинформации в условиях ограничений их надежности и стоимости. Представлен обзор возможных решений для разработчика системы интеллектуального видеонаблюдения. Предложены нетривиальные методы решения поставленных задач. Предлагается двухуровневая обработка видеоинформации высокого качества для систем интеллектуального видеонаблюдения. Рассмотрены способы организации взаимодействия с графическими станциями и серверами высокопроизводительных вычислений. Предложены метрики эффективности. Замерены показатели эффективности предложенных методов для обработки видеоинформации в туманных средах и доказано их превосходство по сравнению с другими методами обработки видеоинформации. Приведены конкретные примеры.

Обработка видеоинформации в туманных средах, нейронные сети, туманные вычисления, обработка видеоизображений, концепция интернета вещей

Текущее состояние диагностики видеоинформации и повсеместное распространение цифровых технологий, разработка новых видеостандартов и развитие техники привели к необходи-