МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**

Кафедра теории функций и стохастического анализа

РАЗРАБОТКА WEB-ПРИЛОЖЕНИЯ «БЮРО НАХОДОК»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 451 группы
направления 38.03.05 — Бизнес-информатика

механико-математического факультета
Корневой Дарьи Геннадиевны

Научный руководитель
д. ф.-м. н., доцент _____ С. П. Сидоров

Заведующий кафедрой
д. ф.-м. н., доцент _____ С. П. Сидоров

Саратов 2022

ВВЕДЕНИЕ

Актуальность темы исследования. С каждым годом web-приложения приобретают все большую популярность из-за их универсальности, удобства использования и гибкости. Web-системы имеют преимущество перед обычными desktop-системами, которые работают по технологии клиент-сервер. Главное преимущество web-приложений — это удобство в поддержке и администрировании: отсутствие необходимости установки приложения на каждое рабочее место, удобство при обновлении версий web-приложения, возможность настройки интерфейса для каждого пользователя.

Основные плюсы построения веб-приложений для поддержки стандартных функций браузера состоят в том, что работа программы должна поддерживаться независимо от операционной системы любого клиента). Таким образом, вместо того, чтобы писать различные версии одного и того же приложения для Windows, MacOS, Linux и других операционных систем, приложение создаётся один раз для любой платформы и на ней использую функции браузера поддерживается работоспособность. Однако при создании веб-приложений и их последующей поддержке различные варианты реализации языков разметки и других спецификаций в браузерах может вызвать проблемы. Помимо этого, возможность юзера настраивать многие параметры браузера, как то, например: размер шрифта, цвета, переход на мобильную версию - может препятствовать корректной работе приложения.

Объект исследования: сетевые технологии, результатом развития которых является появление web-приложений.

Цель бакалаврской работы: спроектировать и разработать Web-приложение для автоматизированного поиска потерянных и найденых вещей в различных категориях с помощью языка программирования Python3 на базе фреймоворка Django.

Объект исследования: создание web-приложения, которое помогает найти потерянные вещи и сообщить о найденных, реализованное с помощью базы данных SQLite, фреймворка Django, языка web-программирования JavaScript и сред разработки HTML5, CSS3.

Предмет исследования: разработка web–приложения обработка объ-

явлений, созданных пользователями по различным параметрам.

Для достижения поставленных целей сформулированы следующие **задачи**:

— Изучить теоретические основы разработки WEB-приложений;
— Разработка и создание веб-интерфейса с основной информацией для запуска приложения;
— Обеспечить логичный поиск объявлений с заданными условиями
— Занесение в базу данных информации введенной пользователями : объект поиска/пропажи, город, контактные данные, фотография;
— Предоставить пользователю возможность изменять объявления, делиться ими
— Предоставить пользователю доступ к базе данных объявлений, для осуществляения прямого назначения приложения, а именно поиска вещей;

Практическая значимость проводимой работы заключается в разработке и создании web-приложения "Бюро находок".

Основное содержание работы

Бакалаврская работа состоит из введения, одного теоретического и четырех практических глав, заключения, списка использованных источников, приложения.

Введение содержит основные положения: актуальность темы исследования; цель, объект, предмет, задачи исследования; практическую значимость исследования.

Первый раздел "Основные этапы разработки web-приложений" описывает комплекс мер и действий по планированию и созданию сайта.

Определение целей и задач проекта. На этапе проектирования формулируются бизнес-цели создаваемого проекта, определяются требования, которым он должен удовлетворять, разрабатывается общая концепция.

На данном этапе уточняются пожелания, а также производится исследование целевой аудитории. Для более глубокого анализа запрашиваются соответствующие материалы: сопутствующие данные — всё, что поможет составить представление о том, кто и с какой целью будет посещать сайт, какие задачи будут выполняться на сайте. Немаловажно выяснить технические

возможности будущей основной пользовательской аудитории — пропускную способность каналов связи, используемые Internet-браузеры и пр.

Для нахождения целевой аудитории целесообразно войти в роль квалифицированного пользователя на аналогичных создаваемому Internet-ресурсах. Помимо прочего это поможет выявить новые креативные концепции для того, чтобы сайт был более конкурентоспособным и «не затерялся» среди множества других.

Когда цели определены, необходимо приступить к составлению расширенного плана проекта, отражающего сколько времени, и иных средств понадобится для выполнения работ на каждом из последующих этапов. После детального рассмотрения и утверждения плана можно приступить к выполнению работы.

Разработка структуры сайта. Разработка структуры сайта включает всё, что касается его содержания и информационной стратегии, определяющей, как должна быть организована подача информации, чтобы будущие посетители сайта могли быстро и легко её найти. Первоочередной задачей на данном этапе является создание карты сайта, отражающей взаимосвязи типовых страниц и их наиболее значимые функциональные возможности.

Карту сайта представляют в виде чертёжа (UML-диаграммы), на котором каждая страница отображается отдельным прямоугольником. Связи между ними показывают схему переходов по страницам.

Также создают каркасы главной и основных типовых страниц, показывающие расположение текста и графики на странице, а также то, как пользователи будут работать с этими элементами. Каркас страницы должен предполагать возможности последующего расширения.

Более эффективную работу сайта обеспечивает соблюдение принципа: «пользователю удобнее добираться до нужной ему страницы максимум за два щелчка мыши». Поэтому обычно рекомендуется использовать не более двух уровней вложенности в пределах каждой группы элементов.

Завершив формирование облика сайта, компоновку страниц и определив размещение содержимого, переходят к следующему этапу web-разработки — к визуальному оформлению.

Разработка дизайн-макетов. Дизайн-макет — это графическое, на-

глядное изображение основных элементов сайта. Дизайн-макет полностью воплощает визуальную концепцию сайта. Его разработка будет выполняться в графической программе Figma. В процессе разработки необходимо руководствоваться письменным соглашением (брифом) на создание дизайн-макета, который содержит пожелания к дизайну: тип, предпочтительные цвета, наличие тех или иных графических элементов и пр.

На этой стадии создаются все элементы web-дизайна в соответствии со стилем подачи информации и общей концепцией. Главным при дизайне сайта является умение разработать графические объекты, которые бы быстро загружались и хорошо смотрелись, независимо от используемого Internet-браузера.

Важным элементом web-дизайна является графика, которую условно можно разделить на три категории:

— иллюстрированная графика — пояснительные изображения, схемы и чертежи, фотографии;
— функциональная графика — кнопки навигации, счётчики и другие элементы управления сайтом;
— декоративная графика — эстетические элементы дизайна страниц — фоновый рисунок, заголовки, выполненные в виде графических файлов и пр.

Такая классификация предполагает использование различных мультимедийных форматов. Например, для чётких, контрастных изображений с мелкими деталями и тонкими линями используются jpeg-файлы, а для красочных изображений с мягкими переходами цветов — gif-файлы.

Html-вёрстка. Html-вёрстка макета является следующим шагом после разработки дизайна сайта. Вёрстка — это преобразование созданных графических макетов страниц в html-код, который бы отображался в Internet-браузере в точном соответствии с исходным макетом. Сложность вёрстки зависит от сложности дизайна. Основными задачами при вёрстке являются:

— корректность отображения страниц сайта при разных разрешениях экрана;
— кросбраузерность — единообразность отображения страниц сайта в наиболее популярных браузерах — Yandex, Internet Explorer, Mozilla Firefox,

Opera, Chrome.

Программирование и контроль качества. Программирование — это практическая реализация проекта, интеграция наработок по отдельным направлениям. Другими словами, это процесс построения функциональных инструментов для наполнения и обработки данных. Программирование определяет насколько стабильным и защищённым будет функционирование сайта. Выбор платформы, технологий и грамотного подхода к программированию играет существенную роль. На данном этапе важно определиться с подходом к созданию Internet-реусурса: будет ли он статическим или динамическим.

Статический Internet-сайт представляет собой совокупность html-файлов, каждый из которых представляет отдельную страницу (или её часть). Такой подход используется, в основном, для размещения файловых архивов и медиа-коллекций. Статические сайты программирования, как такового, не требуют.

Страницы динамического Internet-сайт формируются сервером в ответ на запрос пользователя (передаваемый в виде URL-адреса страницы).

Инструментальные средства разработки и базы данных. В настоящее время актуальным программным функционалом среди множества сред разработки обладает высокоуровневый фреймворк Django. Его возможности позволяют существенно повысить продуктивность web-программирования по сравнению с такими технологиями, как PHP или Perl.

Язык Python, лежащий в основе фреймворка Django, имеет быстрый цикл разработки (редактирование – запуск – редактирование), реализован в виде интерпретатора, поддерживает нетипизированные переменные, не требующие объявления. Django позволяет добиться высокой надёжности и гибкости разработки проектов любого масштаба. Фреймворк включает подсистему тестирования, которая помогает существенно сэкономить время разработки и повысить качество web-проекта.

В основе большинства динамических web-проектов лежит база данных. Существует множество различных систем управления базами данных (СУБД), но в современных хостинг-центрах — организациях по размещению web-проектов на серверах — как правило, применяются СУБД MySQL и PostgreSQL. Причинами тому являются фактическая ориентация этих СУБД

на хостинговые задачи, доступность на всех популярных серверных операционных системах, а также относительная простота настройки и администрирования. СУБД упрощает управление данными и сокращает время и издержки на разработку и развёртывание приложений. MySQL и PostgreSQL обеспечивают приемлемый уровень безопасности, надёжности и масштабируемости.

Запуск и сопровождение. После исправления ошибок и презентации сайта в сети Internet, начинается новый этап работ, связанный с его сопровождением. Основная цель сопровождения — поддержание стабильности работы web-ресурса и актуальности информации. Обязательным условием квалифицированного сопровождения web-сайта является защита информации, включающая в себя антивирусную защиту и защиту баз данных на сервере от действий злоумышленников, в частности, от SQL injection.

Кроме того, необходимо своевременное обновление содержимого сайта, исправление ошибок, не выявленных или не исправленных на стадии проверки качества. Ещё одним важным моментом сопровождения является постоянный мониторинг эффективной работоспособности сайта, контроль посещаемости и учёт данных статистики.

Создание UML-диаграммы С помощью блок-схемы в соответсвии с рисунком 1 создадим структуру сайта, имея 5 сущностей: пользователи, администратора, главная страница сайта, содержащая объявления, страница добавления новой публикации и база данных, в которую будут записываться выложенные объяления.

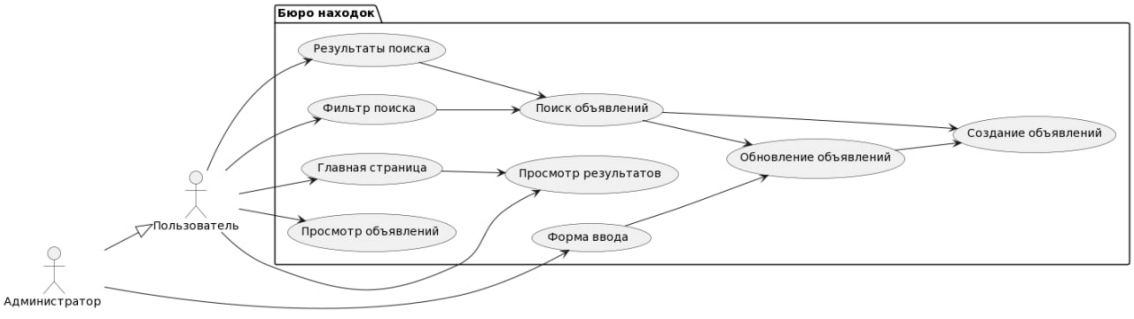

Рисунок 1 – UML-диаграмма сайта

Реализация дизайн-макета сайта. Для создания прототипа web-приложения необходимо обратиться к графическому онлайн-редактору Figma.

Используя Rectangle из Shape tools сформируем будущие полотна страниц размерностью 1440 на 900 пикселей.

Далее, можно приступить к разметке страниц разделив каждую на 3 блока: header, main, footer.

Header будет в себе содержать логотип и основные кнопки навигации сайта.

В блоке main главной страницы создадим следующие элементы: фильтры поиска (запрос, раздел, категория, город) и кнопка поиска, а так же блок, в котором будут отображаться объявления; на странице добавления нового объявления релизуем соответсвующую форму, включающая в себя: заголовок, раздел, категория, город, текст объявления, контактные данные и добавление файла фотографии с расширением jpg или png.

Footer будет в себе содержать знак copyright и наименование организации, которой принадлежат авторские права.

Верстка сайта по дизайн-макету, с помощью фрейморка Django. Прежде чем приступить к верстке сайта, необходимо создать проект, используя в терминале команду django-admin startproject django_example. После чего будет создана следующая директория:

— django_example/__init__.py — пустой файл, который говорит Python, что данная директория должна восприниматься в качестве пакета.
— django_example/settings.py содержит конфигурацию нашего проекта.
— django_example/urls.py — здесь объявляются URL.
— django_example/wsgi.py — с помощью него приложение может работать с веб-сервером по протоколу WSGI.
— manage.py позволяет взаимодействовать с проектом.

Далее, можно приступить к созданию приложения. Это делается следующим образом: python manage.py startapp riddles. После создания веб-приложения, необходимо следовать следующим правилам Django:

— views.py - файл, содержащий все виды;
— urls.py - файл, где выполняется привязка вида к URL;
— python manage.py runserver - запуск приложения и локального сервера.

В заключительном этапе подготовки осталось создать скелет нашей страницы - создание папки templates, где будут храниться все html-файлы.

Подготовив проект, выполняется верстка страниц web-приложения, используя подготовленный ранее дизайн-макет.

Проектирование базы данных сайта. Выполнив все этапы, описанные ранее, можно перейти к созданию базы данных, для полного функционирования сайта.

Для функционирования приложенний, работающих с базами данных выполним: python manage.py migrate. После чего, создам модель, которая будет содержать две функции Riddle и Option. Данная модель обеспечивает Django информацией, необходимой для создания схемы базы данных и database-access API для доступа к объектам. Затем выполняется привязка приложения к проекту и последующая миграция: python manage.py makemigrations riddles.

Последующим шагом, будет создание атрибутов базы данных и её сохране в проекте.

Запуск и тестирование web-приложения. При запуске главной страницы выводится следующее сообщение: "Объявлений по вашему запросу не найдено". Это значит, что в базе данных отсутвуют публикации. Поэтому необходимо нажать на кнопку «Добавить объявление». Далее, пользователя перенаправит на страницу формы добавления нового объявления.

Заполнив все поля, нажимаем на «Сохранить и добавить», после чего выполниться запись объявления в базу данных и автоматическое перенаправление на главную страницу сайта.

Для того, чтобы узнать подробную информацию, о только что выложенной публикации, требуется нажать на её фотографию.

Теперь для тестирования фильтров запроса необходимо создать несколько новых объявлений по алгоритму, описанному ранее. Последующим шагом, введём интересующие параметры и нажмем на «Поиск». В случае если объявления существуют, то информация на сайте отсортируется, иначе будет выведено сообщение, описанное в начале данного пункта.

Основные результаты

1. Изучена теория по основам разработки web - приложений.
2. Рассмотрена структура веб-интрфейса.

3. Разработан дизайн - макет.
4. Верстка сайта с последующим созданием базы даннх.
5. Разработано, протестировано и запущено web - приложение «Бюро находок».

   Программный код приводится в **приложении А** и **приложении Б**.