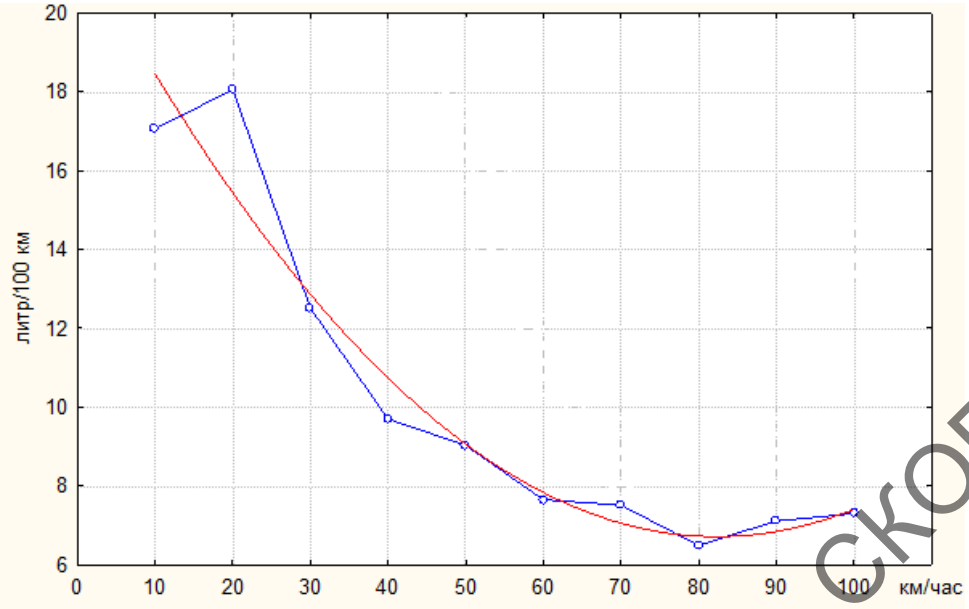

Рисунок 2 – Зависимость расхода топлива от скорости автомобиля

Вся полученная и зафиксированная в базе данных на сервере информация, обработанная с использованием схем реализованного приложения, во-первых, позволит составить описание автомобиля и особенностей его эксплуатации для пользователя; во-вторых, будет служить основой для выявления неисправностей при диагностике автомобиля; в-третьих, обеспечит формирование общей картины состояния дорог для ограниченного региона за продолжительный период.

Литература

1 Рэндалл, М. Электрическое и электронное оборудование автомобилей / М. Рэндалл. – Санкт-Петербург : Алфамер Паблишинг, 2008. – 284 с.

2 Электронные системы автомобиля [Электронный ресурс]. – Режим доступа : http://awtoel.narod.ru/index.html. – Дата доступа : 08.05.2020.

3 Ютт, В. Е. Электрооборудование автомобилей: учеб. для студентов вузов / В. Е. Ютт. – 2-е изд., перераб. и доп. – Москва : Транспорт, 2005. – 304 с.

УДК 004.42:004.738.1:004.773

С. И. Коровкин

СОЗДАНИЕ СЕРВИСА ДЛЯ УПРАВЛЕНИЯ ДАННЫМИ В ЛИЧНОМ КАБИНЕТЕ МОБИЛЬНОГО ОПЕРАТОРА

Статья посвящена разработке проекта по созданию автоматизированной системы, отвечающей за перенос клиентских номеров из других мобильных операторов. Сделано описание функциональных возможностей проекта, определение ролей и выполняемых сценариев, отображение информационно-логической модели данных и описание архитектуры проекта. Созданы структуры подсистемы и всех необходимых объектов, а также проведено тестирование полученного продукта и оформление сопроводительной документации.

Провайдеры мобильной связи предоставляют своим клиентам возможность перемещения мобильного номера из личного кабинета. При реализации этих действий возникают трудности. Разработанная функциональность расширяет возможности личного кабинета пользователя. Она значительно облегчает переход клиентов от одного мобильного оператора к другому и способствует увеличению потока клиентов. А так как у клиентов появляется возможность совершения операций по перемещению номера в личном кабинете без согласования с мобильным оператором, значительно экономится их личное время.

Работа состоит из следующих этапов. *Позиционирование проекта.* Здесь описывается объект проектирования, целевая аудитория проекта и актуальность разработки. *Разработка проекта.* Здесь определяются списки ролей приложения, приводятся основные и альтернативные сценарии, иллюстрируемые диаграммой последовательности. Кроме того, приводится описание информационно-логической модели и предоставляется информация об архитектуре проекта с обоснованием выбора инструментария. *Реализация приложения.* Здесь описывается поэтапный процесс реализации приложения, а также различные компоненты подсистемы «Keeping Your Number» [1, с. 149].

Для подсистемы «Keeping Your Number» были определены следующие роли:

– Клиент. Инициирует процесс переноса своего номера из другой сети;

– Администратор. Получает заявку от клиента через электронное сообщение и начинает процесс перемещения номера.

Прецеденты подсистемы: ввод PAC кода; ввод номера, который будет перемещен; ввод временного номера EE; ввод пользовательских данных; отправка формы на обработку; получение сообщения на электронную почту об успешной активации процесса.

Описание сценариев. Прецедент «Ввод PAC кода». Главный поток событий: пользователь вводит PAC или STAC код, в зависимости от того, чего хочет: сохранить текущий номер из другого провайдера или переместиться без сохранения номера. STAC код должен начинаться из 6 цифр, далее идут 3 буквы. PAC код состоит из 3 букв и 6 цифр. Если формат кода прошел проверку, то пользователь переходит на этап ввода номеров. Альтернативный поток событий осуществляется в том случае, если был введен PAC код неверного формата. В такой ситуации пользователь видит сообщение об ошибке.

Прецедент «Ввод номеров». Главный поток событий: пользователь вводит временный номер мобильного оператора EE и номер из другой сети. Система производит статическую проверку номеров на формат, номер может начинаться с префиксом 07 или 44, после этого должно идти 11 цифр. В случае, если введенные данные корректны, система с помощью «Ajax» запроса выполняет проверку номера с использованием «APIgee API». Здесь проверяется PAC код и номер, который должен быть перемещен.

Альтернативный поток событий. Если PAC код оказался не действительным, пользователь возвращается на первый этап ввода PAC кода и видит сообщение об ошибке. Если же номера оказались не существующими, то пользователь видит сообщение соответствующего содержания об ошибке на странице ввода номеров.

Прецедент «Ввод пользовательских данных». Главный поток событий. Этот этап является последним. Пользователю необходимо ввести почтовый адрес, имя и фамилию, а также свою дату рождения. После этого нажимается кнопка подтверждения правильности введенных данных.

Далее система проверяет введенные данные и в случае успеха пользователь переходит на страницу подтверждения успешности операции. Здесь же он может прочесть дополнительную информацию о процессе перемещения своего номера. Кроме того, система отправляет и почтовое сообщение пользователю с необходимой информацией о перемещении. Альтернативный поток событий осуществляется, если имя, фамилия или почтовый адрес не соответствует формату. В этом случае пользователь видит сообщение об ошибке.

Проект имеет Model-View-Controller архитектуру и использует следующие технологии:

– язык программирования Java 8 и Java Script;

– сервлет контейнер Jetty;

– фреймворк Spring MVC с использованием шаблонов jsp;

– средство сборки проекта Maven;

Model-View-Controller (MVC, «Модель-Представление-Контроллер») – это схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер. Происходит это таким образом, что модификация каждого компонента может осуществляться независимо.

Модель (Model) предоставляет данные и реагирует на команды контроллера, изменяя своё состояние.

Представление (View) отвечает за отображение данных модели пользователю, реагируя на изменения модели.

Контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений.

Основная цель применения этой концепции состоит в отделении бизнес-логики (модели) от её визуализации (представления, вида). За счёт такого разделения повышается возможность повторного использования кода. Наиболее полезно применение данной концепции в тех случаях, когда пользователь должен видеть те же самые данные одновременно в различных контекстах и/или с различных точек зрения.

К одной модели можно присоединить несколько видов, при этом не затрагивая реализацию модели. Например, некоторые данные могут быть одновременно представлены в виде электронной таблицы, гистограммы и круговой диаграммы.

Не затрагивая реализацию видов, можно изменить реакции на действия пользователя (нажатие мышью на кнопке, ввод данных) – для этого достаточно использовать другой контроллер.

Проект основан на архитектуре Фреймворка Spring MVC. На рисунке 1 можно увидеть диаграмму обработки запроса и ответов Фреймворком.

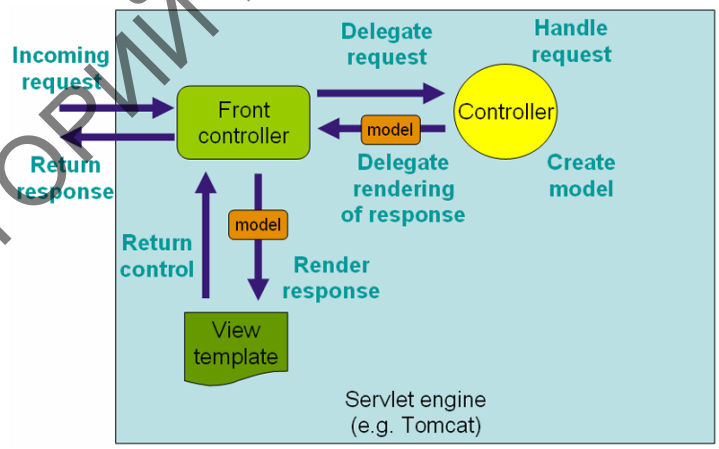

Рисунок 1 – Диаграмма обработки запроса и ответа на основе Фреймворка Spring MVC

Итак, во время проектирования были созданы следующие контроллеры (Controllers):

– KeepingYourNumberHandler.java;

– PhoneNumberAjaxHandler.java;

– MoveYourNumberHandler.java.

Кроме того, были созданы некоторые модели данных (models):

– KeepYourNumberViewData.java;

– MoveYourNumberConfirmationViewData.java;

– MoveYourNumberValidationResult.java.

А что касается шаблонов отображения, то в ходе проектирования были созданы следующие из них (views): keepyournumber.jsp; confirmation.jsp.

Сборка проекта включает в себя компиляцию исходного кода и создание war файла, который будет использоваться сервлет контейнером Jetty для запуска приложения. Для автоматизации процесса сборки используется Фреймворк Apache Maven.

Проведено модульное и функциональное тестирование. Целью модульного тестирования является проверка работы прикладной логики всего приложения или отдельных его частей при разных исходных данных. Также проводится анализ правильности получаемых результатов.

Ручное тестирование (manual testing) – часть процесса тестирования на этапе контроля качества в процессе разработки программного обеспечения. Целью мануального тестирования является проверка работы приложения без использования программных средств. При этом производится моделирование действий пользователя.

С помощью созданной в результате проектирования функциональности «Keeping Your Number» пользователь получает возможность пройти все этапы перенесения номера из одного мобильного провайдера к другому не выходя из дома, из своего онлайн-аккаунта. Таким образом, экономится личное время пользователей и значительно ускоряется процесс перемещения номера. Благодаря расширению функциональных возможностей личного кабинета пользователя и облегчению перехода клиентов в мобильную сеть ЕЕ увеличивается поток клиентов.

Литература

1 Коровкин, С. И. Управления данными в личном кабинете мобильного оператора / С. И. Коровкин, Г. Л. Карасёва // Новые математические методы и компьютерные технологии в проектировании, производстве и научных исследованиях: материалы XXIII Республиканской научной конференции студентов и аспирантов, Гомель, 23–25 марта 2020 г. / ГГУ им. Ф. Скорины; редкол.: С. П. Жогаль (гл. ред.) [и др.]. – Гомель, 2020. – С. 149–150.

УДК 338.2

О. Ф. Кузьмич

СТАТИСТИЧЕСКОЕ ИССЛЕДОВАНИЕ ИНДИКАТОРОВ ЗЕЛЕНОЙ ЭКОНОМИКИ В БЕЛАРУСИ, ПОЛЬШЕ, ЛИТВЕ И ЛАТВИИ

Статья посвящена исследованиям развития зеленой экономики в Беларуси, Латвии, Литве и Польше. Дана оценка состояния зеленой экономики в рассматриваемых странах. Определена статистическая взаимосвязь между индикаторами зеленой экономики в каждой стране. Методами дисперсионного анализа показана статистическая неоднородность динамики энергоэффективности в рассматриваемых странах.

Зеленая экономика – это путь устойчивого развития, основанный на решении проблемы взаимозависимости между экономическим ростом, социальным развитием и экологическим благополучием. Зеленую экономику нужно рассматривать как важный