

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра методов оптимального управления**

НЕДЕЛЬКО
Дмитрий Валерьевич

**РАЗРАБОТКА СИСТЕМЫ АВТОМАТИЗАЦИИ ПРОЦЕССА
НАПОЛНЕНИЯ БАНКОМАТА ДЕНЕЖНЫМИ СРЕДСТВАМИ**

Дипломная работа

Научный руководитель:
кандидат физико-математических
наук, доцент Л.И. Лавринович

Допущен к защите

«___»_____ 2017 г.

Зав. кафедрой методов оптимального управления,
кандидат физико-математических наук, доцент Н.М. Дмитрук

Минск, 2017

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра методов оптимального управления

Утверждаю

Заведующий кафедрой _____ Дмитрук Н.М.
(подпись)

Дата _____

ЗАДАНИЕ НА ДИПЛОМНУЮ РАБОТУ

Студенту Неделько Д.В.

1. Тема дипломной работы:

Разработка системы автоматизации процесса наполнения банкомата денежными средствами.

Утверждена приказом ректора БГУ от 14.12.2016 № 1745-ПС

2. Исходные данные к дипломной работе:

- Angular Documentation [Electronic resource]. – Mode of access: <https://angular.io/docs/ts/latest/>. – Date of access: 29.11.2016.
- Дэвид Флэнаган, JavaScript. Подробное руководство / Дэвид Флэнаган. – 6-е издание, серия O'Reilly. – Символ-Плюс, 2012. – 20 с.
- Webpack Documentation [Electronic resource]. – Mode of access: <https://webpack.js.org/configuration/>. – Date of access: 29.11.2016.
- TypeScript Documentation [Electronic resource]. – Mode of access: <https://www.typescriptlang.org/docs/tutorial.html>. – Date of access: 29.11.2016.
- Steve Fenton, Pro TypeScript. Application-Scale JavaScript Development / Steve Fenton. – APPRESS, 2016. – 17 p.
- Material Components [Electronic resource]. – Mode of access: <https://material.angular.io/components>. – Date of access: 29.11.2016.
- Node.js Documentation [Electronic resource]. – Mode of access: <https://nodejs.org/en/docs/>. – Date of access: 29.11.2016.
- Mario Casciaro, Node.js Design Patterns / Mario Casciaro, Luciano Mammino. – Second Edition. – Packt Publishing Ltd., 2016. – 23 p.
- Express Documentation [Electronic resource]. – Mode of access: <https://expressjs.com/en/api.html>. – Date of access: 29.11.2016.
- Mongo DB Documentation [Electronic resource]. – Mode of access:

<https://docs.mongodb.com/manual/>. – Date of access: 29.11.2016.

3. Перечень подлежащих разработке вопросов или краткое содержание расчетно-пояснительной записки:

- исследовать проблемы, возникающие в процессе наполнения банкоматов денежными средствами;
- освоить современные технологии веб-разработки;
- разработать веб-приложение.

4. Примерный календарный график выполнения дипломной работы:

- декабрь 2016 – исследование проблем, возникающих в процессе наполнения банкоматов денежными средствами;
- январь 2017 – проработка бизнес-логики веб-приложения;
- февраль 2017 – выбор стека технологий и изучение документации;
- март 2017 – разработка веб-приложения;
- апрель 2017 – разработка веб-приложения, тестирование, исправление ошибок в программном коде;
- май 2017 – оформление дипломной работы, подготовка презентации.

5. Дата выдачи задания: 29.11.2016

6. Срок сдачи законченной дипломной работы: 10.05.2017

Руководитель _____ Лавринович Л.И.
(подпись)

Подпись студента _____

Дата _____

РЕФЕРАТ

Дипломная работа, 28 страниц, 13 рисунков, 10 источников.

Ключевые слова: БАНКОМАТ, ВЕБ-ПРИЛОЖЕНИЕ, БАЗА ДАННЫХ, ANGULAR, TYPESCRIPT, WEBPACK, MATERIAL, NODE.JS, EXPRESS, MONGO DB.

Объект исследования: процесс пополнения банкоматов денежными средствами.

Цель работы: исследовать проблемы, возникающие в процессе наполнения банкоматов денежными средствами, освоить современные технологии веб-разработки, разработать веб-приложение, позволяющее частично автоматизировать процесс пополнения банкоматов денежными средствами.

Методы исследования: изучение документации, разработка веб-приложения.

Результат работы: веб-приложение, которое позволяет формировать список банкоматов, в которых в скором времени закончатся денежные средства.

Область применения: подразделения банковской системы, занимающиеся пополнением банкоматов денежными средствами.

РЭФЕРАТ

Дыпломная праца, 28 старонкі, 13 малюнкаў, 10 крыніц.

Ключавыя словы: БАНКАМАТ, ВЭБ-ДАДАТАК, БАЗА ДАНЫХ, ANGULAR, TYPESCRIPT, WEBPACK, MATERIAL, NODE.JS, EXPRESS, MONGO DB.

Аб'ект даследавання: працэс папаўнення банкаматаў грашовымі сродкамі.

Мэта працы: даследаваць праблемы, якія ўзнікаюць у працэсе напаўнення банкаматаў грашовымі сродкамі, асвойць сучасныя тэхналогіі вэб-распрацоўкі, распрацаваць вэб-прыкладанне, якое дазваляе часткова аўтаматызаваць працэс папаўнення банкаматаў грашовымі сродкамі.

Метады даследавання: вывучэнне дакументацыі, распрацоўка вэб-прыкладанні.

Вынік работы: вэб-прыкладанне, якое дазваляе фарміраваць спіс банкаматаў, у якіх у хуткім часе скончацца грашовыя сродкі.

Вобласць ўжывання: падраздзялення банкаўскай сістэмы, якія займаюцца папаўненнем банкаматаў грашовымі сродкамі.

ABSTRACT

Diploma work, 28 pages, 13 drawings, 10 sources.

Keywords: ATM, WEB-APPLICATION, DATABASE, ANGULAR, TYPESCRIPT, WEBPACK, MATERIAL, NODE.JS, EXPRESS, MONGO DB.

Object of study: the process of replenishing ATMs in cash.

Purpose of work: investigate the problems arising in the process of filling ATMs in cash, learn modern technologies of web development, implement web application that allows to partially automate the process of replenishing ATMs in cash.

Research methods: documentation study, web application development.

The result of work: a web application that allows you to create a list of ATMs in which money will soon run out.

Area of application: banking system units engaged in replenishment of ATMs in cash.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	6
1 Процесс наполнения банкоматов денежными средствами	7
1.1 Устройство банкомата	7
1.2 Описание процесса наполнения банкоматов денежными средствами	8
1.3 Пиковые дни снятия денежных средств	9
2 Разработка веб-приложения	11
2.1 Постановка задачи	11
2.2 Оценка наполненности банкомата	12
2.3 Алгоритм попадания банкомата в очередь	12
2.4 Используемые технологии и инструменты	13
2.4.1 Angular.	14
2.4.2 Webpack.	14
2.4.3 TypeScript.....	14
2.4.4 Google Maps APIs.	15
2.4.5 Material	15
2.4.6 Node.js.....	15
2.4.7 Express.	15
2.4.8 Mongo DB.....	16
2.5 Структура базы данных.....	16
3 Демонстрация веб-приложения на тестовых данных	18
3.1 Вход в приложение	18
3.2 CRUD операции с банкоматами на карте города.....	18
3.3 Просмотр таблицы банкоматов	22
3.4 Статистика выбранного банкомата	23
3.5 Изменения правила попадания банкомата очередь	24
3.6 Печать отчетов	26
ЗАКЛЮЧЕНИЕ	27
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ	28

ВВЕДЕНИЕ

Ключевое понятие, которое выражает сущность рыночных отношений, является понятие конкуренции. Конкуренция в банковской сфере представляет собой более сложную систему, чем в любом другом секторе экономики. Ее специфика определяется многими факторами, главным из которых является «лояльность клиента к банку». На сегодняшний день в Республике Беларусь работает несколько десятков банковских компаний, прибыль которых напрямую зависит от выбора клиентом обслуживания у того или иного банка.

В современной сфере банковского обслуживания широко популярны безналичные операции посредством использования банковских карт. Однако часто, в силу определенных жизненных ситуаций бывают случаи, когда человеку необходимы именно наличные денежные средства.

У каждого банка есть собственная сеть банкоматов, в которой клиент может снять наличные денежные средства без комиссии банка. Но случается, что в банкомате иногда заканчиваются наличные денежные средства. Такие моменты негативно сказываются на лояльности клиента к банку.

В данной дипломной работе предлагается решение, минимизирующее временные интервалы, когда в банкомате отсутствуют наличные денежные средства. Будет спроектировано и разработано веб-приложение, которое позволяет формировать список банкоматов, в которых в скором времени закончатся денежные средства.

1 Процесс наполнения банкоматов денежными средствами

В настоящей главе описывается само устройство банкомата, весь процесс наполнения банкоматов денежными средствами, а так же особенности, связанные с этим процессом.

1.1 Устройство банкомата

Банкомат (АТМ от англ. Automated Teller Machine) – программно-технический комплекс, предназначенный для автоматизированных выдачи и/или приёма наличных денежных средств как с использованием платежных карт, так и без, а также выполнения других операций, в том числе оплаты товаров и услуг, составления документов, подтверждающих соответствующие операции. Банкомат изготовлен из прочных материалов которые защищают эго от взлома и других воздействий. Банкомат подключен к сети, по которой передается информация об осуществленных операциях в процессинговый центр банка.

Каждый банкомат включает в себя:

- клиентскую часть;
- инженерную часть;
- сейф.

Клиентская часть банкомата видна и доступна всем окружающим. На ней размещены экран, клавиатура, устройство приема банковских карт (картридер), принтер, устройство для выдачи и приема денежных средств.

На экране банкомата представлены операции, которые можно выбрать путем нажатия на соответствующие кнопки, которые находятся слева и справа от экрана. Стоит отметить, что уже существуют банкоматы с сенсорными экранами, операции в которых осуществляется путем нажатия на соответствующую область экрана.

Картридер – это механическое устройство, которое принимает и, по завершению операций, возвращает карту, а также считывает информацию с чипа или с магнитной полосы карты.

После завершения ряда операций банкомат выдает денежные средства через щель для выдачи денег. Также существуют банкоматы, которые позволяют совершать платежи с использованием наличных денежных.

В конце всех операций банкомат предлагает напечатать чек, на котором отображается название проводимой операции, сумма денежных средств, участвующих при проведении данной операции.

В сейфе банкомата находятся кассеты с денежными средствами в

количестве от 4 до 6 штук, при чем каждая кассета наполнена купюрами одного номинала (Рисунок 1.1). Одна кассета банкомата изначально пуста, так как служит хранилищем для забытых и (или) отбракованных купюр.



Рисунок 1.1 – Кассеты с денежными средствами

Инженерная часть банкомата представляет собой компьютер, на который записана специальная операционная система. Через инженерную часть работники банка проводят операции с настройкой и управлением работы банкомата.

1.2 Описание процесса наполнения банкоматов денежными средствами

Предположим, что клиент захотел получить наличные денежные средства в одном из банкоматов. Но введя желаемую сумму, банкомат вывел на экран

сообщение, что в банкомате закончились денежные средства. Как следствие, пользователь остался недоволен. Стоит отметить, что на всех банкоматах установлен лимит выдачи денежных средств за одну операцию, таким образом клиент не может запросить сумму, которой попросту не сможет оказаться в полностью загруженном банкомате. Так как каждый банкомат оборудован компьютером, то он в режиме реального времени отправляет все операции, которые выполнялись с ним в отдел мониторинга. В данном случае было послано сообщение о том, что в кассетах недостаточно купюр, чтобы выдать клиенту требуемую сумму. Обработав данные, специалисты отдела мониторинга банковской сети отправляют заявку в отдел формирования кассет с денежными средствами для их замены в банкоматах. После завершения наполнения кассет, данные передаются отделу безопасности для утверждения маршрута, по которому будет осуществляться доставка кассет. После чего, все кассеты погружаются в инкассаторский автомобиль и по установленному маршруту доставляются к соответствующим банкоматам, где производится замена неполных кассет на новые. После проведения мероприятий по замене кассет, банкомат переходит в штатный режим работы.

Таким образом, необходимо автоматизировать мониторинг сети банкоматов, и заблаговременно формировать заявки на замену кассет с наличными средствами, чтобы минимизировать интервал времени, когда в банкомате недостаточно денежных средств.

1.3 Пиковые дни снятия денежных средств

Зачастую люди сталкиваются с проблемой невозможности снятия денежных средств в каком-либо банкомате. Дело в том, что людям перечисляют зарплаты в конкретные даты и, как правило всем сразу. По данным Национального банка Республики Беларусь, доля безналичных расчетов по состоянию на 1 января 2017 года насчитывает около 30%, таким образом в нашей стране не особо развита культура пользования пластиковыми картами. Следовательно, основная часть этих людей сразу выстраивается в очереди ко всем ближайшим банкоматам и за несколько часов сметает всю наличность из них. Надо понимать, что банкоматы обладают ограниченным количеством денежных средств. Часто в дни выплат зарплат люди будут сталкиваться с тем, что в банкомате закончились наличные денежные средства. Как уже понятно, такая нагрузка происходит в дни массовых выплат. Разумеется, что этот момент негативно сказывается на постоянстве и имидже банковской организации, которая обслуживает банкомат. Для решения данной проблемы, необходимо, чтобы в дни пиковой нагрузки банкоматы были достаточно загружены, чтобы удовлетворить потребность клиентов.

2 Разработка веб-приложения

В данной главе отражены основные моменты, связанные с разработкой веб-приложения, которые включают в себя постановку задачи, используемые технологии, бизнес-логика, алгоритм вычисления пикового дня, структура базы данных.

2.1 Постановка задачи

Разработать веб-приложение, используя современные технологии, которое позволяет формировать список банкоматов (очередь), в которых в скором времени закончатся денежные средства для города Минска. Данный список формируется по определенным правилам.

Основная бизнес-логика приложения:

- Приложение могут использовать только аутентифицированные пользователи.
- Пользователь должен иметь возможность выполнять CRUD (создание, чтение, редактирование, удаление) операции с банкоматами, при этом иметь наглядное представление локаций банкоматов на карте.
- Пользователь должен иметь возможность представления списка банкоматов в табличном виде, с возможностью фильтрации по заданным параметрам.
- Пользователь должен наглядно видеть статистику по наличным денежным операциям конкретного банкомата за определенный период.
- Пользователь должен иметь возможность изменять правила в системе, при которых банкомат попадает список банкоматов (в очередь), в которых в скором времени закончатся денежные средства.
- Пользователь должен иметь возможность помечать банкоматы в системе, для которых формируются новые кассеты с денежными средствами.
- Пользователь должен иметь возможность помечать банкоматы в системе, когда бригаде инкассаторов переданы новые сформированные кассеты.
- Пользователь должен иметь возможность отправить на печать список банкоматов, отфильтрованных произвольным образом.

2.2 Оценка наполненности банкомата

Для удобного анализа текущей наполненности банкомата, оценка будет вычисляться в процентах. При этом стоит учесть тот факт, что банкомат может выдавать несколько видов валют. Будем считать, что банкомат имеет одинаковые по вместимости кассеты. Пусть в банкомате C кассет, V – количество валют, которые загружены в банкомат, $0 < V \leq C$, c_j – количество кассет j -ой валюты, N_{max} – максимальное количество купюр, которое вмещает одна кассета, x_i^j – текущее количество купюр в i -ой кассете j -ой валюты, n_i^j – номинал купюры в i -ой кассете j -ой валюты, $i = \overline{0, c_j - 1}$, $j = \overline{0, V - 1}$. Таким образом, формула для вычисления процента наполненности кассет j -ой валюты будет выглядеть следующим образом:

$$p_j(x) = \frac{\sum_{i=0}^{c_j-1} x_i^j n_i^j}{N_{max} \sum_{i=0}^{c_j-1} n_i^j}. \quad (2.1)$$

Оценка текущей наполненности банкомата (учитываются кассеты всех валют) имеет вид:

$$P(x) = \sum_{j=0}^{V-1} \frac{c_j}{C} \frac{\sum_{i=0}^{c_j-1} x_i^j n_i^j}{N_{max} \sum_{i=0}^{c_j-1} n_i^j}. \quad (2.2)$$

2.3 Алгоритм попадания банкомата в очередь

Зададим следующие параметры:

- *statistic_period* – период, по которому будет искать дни массовых снятий денежных средств (последние 3 месяца, последние 6 месяцев и так деле);
- *inner_days* – внутренний интервал пиковых дней;
- *outer_days* – внешний интервал пиковых дней (своего рода погрешность);
- *delta_percent* – процент денежных средств, выданных за внутренний промежуток пиковых дней;
- *critical_percent* – отметка денежных средств в процентах, перешагнув которую, банкомат попадает в очередь;
- *secondary_critical_percent* – вторичная отметка денежных средств в процентах (используется при расчете пиковых дней), перешагнув которую, банкомат попадает в очередь;

- *granularity* – периодичность наступления пиковых дней (каждый месяц);
- *matched* – количество совпадений внутренних интервалов пиковых дней.

Существует два случая, при которых банкомат попадает в очередь на пополнение денежными средствами:

1. в случаи, когда процент денежных средств в кассетах одной из валют (2.1) меньше значения *critical_percent*;
2. в случаи, когда в скором времени предположительно будет массовое снятие денежных средств.

Далее опишем подробно алгоритм второго случая.

Шаг 1. Находим все интервалы *inner_days*, в течении которых процент денежных средств банкомата (2.2) уменьшился больше, чем на значение *delta_percent* на промежутке *statistic_period*.

Шаг 2. Для всех найденных интервалов, отличных от первого, добавляем слева и справа *outer_days* (Рисунок 2.1).

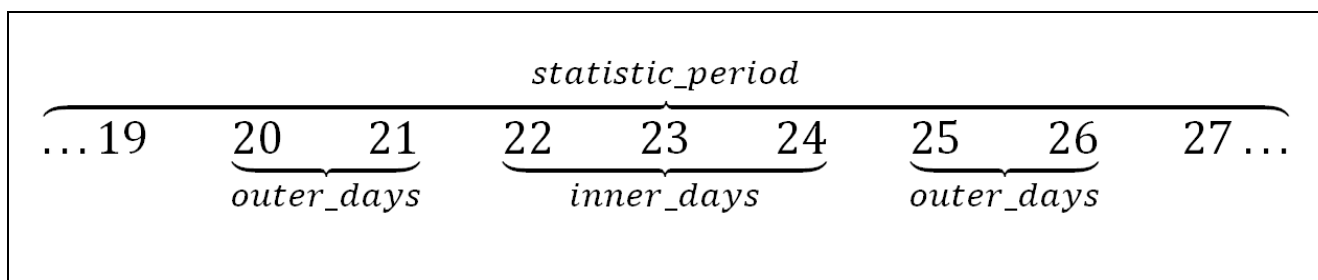


Рисунок 2.1 – Пример выбора интервалов

Шаг 3. Помечаем промежутки, увеличенные на *outer_days*, в которые полностью попадает первый интервал с периодичностью в *granularity*. Если количество совпадений внутренних интервалов пиковых дней для первого интервала, больше заданного числа *matched* – заносим *inner_days* первого интервала в список пиковых дней с периодом *granularity*, при этом, удаляя *inner_days*, соответствующих помеченным из списка, полученного после шага 1.

Повторяя шаг 2 и шаг 3 для всех оставшихся интервалов, получим полный список пиковых дней с периодом *granularity*.

Таким образом, банкомат будет находиться в очереди на пополнение денежными средствами в течении *outer_days*, которые находятся слева от первого дня *inner_days*, если текущий процент денежных средств в кассетах одной из валют (2.1) меньше значения *secondary_critical_percent*.

2.4 Используемые технологии и инструменты

Для реализации был выбран стек современных технологий веб-разработки. Опишем основные особенности каждой из них.

2.4.1 Angular.

В качестве фреймворка, который используется для разработки клиентской части веб-приложения был выбран Angular [1]. Angular – это общедоступный JavaScript фреймворк, который позволяет реализовывать достаточно большие, высоконагруженные приложения [Ошибка! Источник ссылки не найден.]. В данном веб-приложении используется Angular версии 4, который был выпущен 23 марта 2017 года. Стоит отметить, что изначально был выпущен фреймворк версии 1, который назывался AngularJS. Начиная с версии 2, было принято решение переименовать на Angular.

Основные преимущества Angular:

- лучшая производительность по сравнению с AngularJS;
- использует компонентно-ориентированный подход;
- кроссплатформенный (может использоваться как для мобильных так и для стационарных устройств);
- позволяет легко поддерживать код в больших приложениях;
- позволяет создавать высокопроизводительную анимацию в компонентах;
- адаптивность к тестированию.

Стоит заметить, что Angular обладает более высоким порогом вхождения, чем AngularJS.

2.4.2 Webpack.

Хорошей практикой в современной веб-разработке является применение инструментов, которые помогают упростить и ускорить процесс разработки, а также подготовить файлы проекта непосредственно для сервера. Для этих целей уже существуют популярные исполнители задач, такие как Gulp или Grunt, с помощью которых можно минимизировать файлы с кодом, соединить их в единый файл и так далее. Webpack также отлично справляется с этой задачей [1].

Webpack – это сборщик модулей, который берет JavaScript-модули с необходимыми зависимостями, и затем соединяет их вместе как можно более эффективным способом, на выходе создавая единый JavaScript-файл. На первый взгляд – ничего особого, так как такие инструменты, как RequireJS позволяют делать подобные вещи. Преимущество Webpack заключается в том, что он не ограничен в использовании только JavaScript-модулей. Применяя специальные загрузчики, Webpack понимает, что JavaScript-модулям могут потребоваться для их работы, например, файлы со стилями, а им, в свою очередь, изображения. При этом, результат работы Webpack будет содержать только те ресурсы, которые действительно нужны для работы приложения.

2.4.3 TypeScript.

TypeScript – это надстройка над языком JavaScript, которая позволяет определять новые типы данных [4]. Разработчики Angular рекомендуют использовать TypeScript при разработке приложений, который позже

транспирируется в JavaScript [**Ошибка! Источник ссылки не найден.**]. Декларирование переменных, используя строгую типизацию, открывает широкую возможность в удобстве разработки и поддержки написании кода за счет использования статического анализатора кода. Таким образом, во время написания кода в среде разработки (WebStorm/IntelliJ Idea, Visual Studio Code, Sublime Text и так далее), имеется возможность руководствоваться контекстно-зависимыми рекомендациями, которые предлагают доступные методы объекта или аргументов функции. В среде разработки также помечается части кода, где обнаружены ошибки, с подробным их описанием.

Даже если приложение TypeScript использует стороннюю библиотеку, написанную на JavaScript, вы можете установить файл (имеющий расширение `.d.ts`), который будет отвечать за соответствие типов для этой библиотеки. Объявления типов для сотен популярных библиотек JavaScript доступны в свободном доступе, и вы можете легко установить их с Typings – диспетчером определения TypeScript. Представьте, что вы хотите использовать jQuery, написанный на JavaScript, из вашего кода TypeScript. Файлы соответствия типов для jQuery будут содержать объявления с типами всех API jQuery, чтобы ваша среда разработки могла предложить вам, какие типы использовать или выделить какой-либо ошибочный код.

2.4.4 Google Maps APIs.

С помощью Google Maps APIs можно создавать интерактивные веб-приложения, используя мощную картографическую платформу Google Maps, которая включает в себя маршруты проезда, снимки просмотра улиц и многое другое.

2.4.5 Material

В данном веб-приложении используются Material компоненты, которые соответствуют Google Material спецификации и отвечают за стилевую составляющую графического интерфейса [4].

2.4.6 Node.js.

Node.js – программная платформа, основанная на движке V8 (транслирующем JavaScript в машинный код), превращающая JavaScript из узкоспециализированного языка в язык общего назначения [7]. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода/вывода через свой API (написанный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на Node.js и десктопные оконные приложения и даже программировать микроконтроллеры [**Ошибка! источник ссылки не найден.**]. В основе Node.js лежит событийно-ориентированное и асинхронное программирование с неблокирующим вводом/выводом.

2.4.7 Express.

Express – это минималистичный и гибкий веб-фреймворк для приложений Node.js, реализованный как свободное открытое программное обеспечение,

предоставляющий обширный набор функций для мобильных и веб-приложений [7]. Имеет в своем распоряжении множество служебных методов HTTP и промежуточных обработчиков. Позволяет создавать надежный API быстро и легко.

2.4.8 MongoDB.

MongoDB – это документоориентированная система управления базами данных (далее СУБД) с открытым исходным кодом, не требующая описания схемы таблиц. СУБД [10]. Данные в MongoDB хранятся в документах, которые объединяются в коллекции. Каждый документ представляет собой JSON-подобную структуру. Проведя аналогию с реляционными СУБД, можно сказать, что коллекциям соответствуют таблицы, а документам – строки в таблицах. Классифицирована как NoSQL.

Особенности MongoDB:

- не требует какого-либо описания схемы базы данных – она может постепенно меняться по мере развития приложения;
- поддерживаются индексы, в том числе по массивам и вложенным документам;
- наличие атомарных операций;
- размер коллекции в MongoDB может быть ограничен числом документов или мегабайтами;
- в документах можно хранить бинарные данные (картинки, аудио файлы и так далее).

2.5 Структура базы данных

Структура базы данных представлена на следующей схеме (Рисунок 2.2).

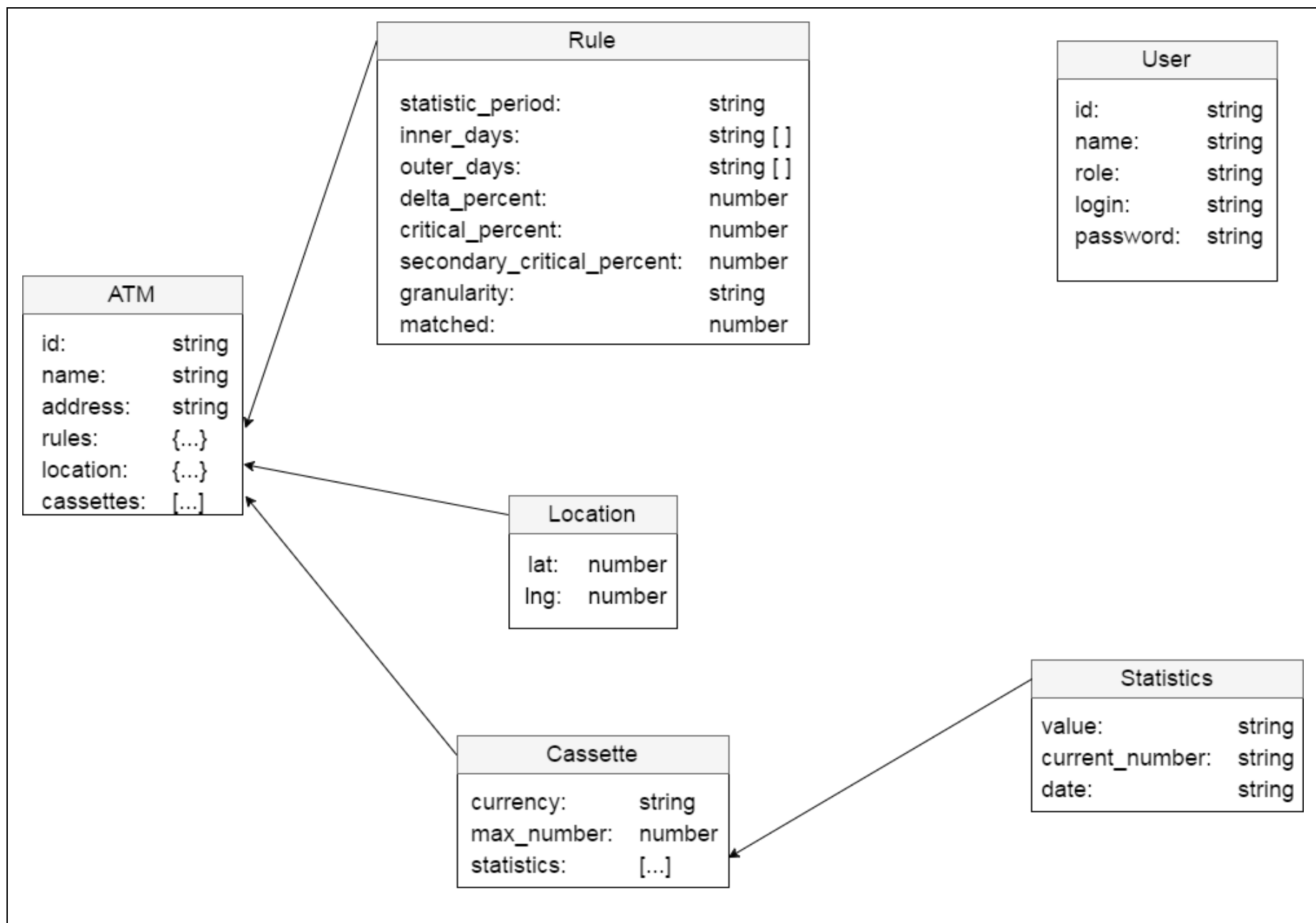


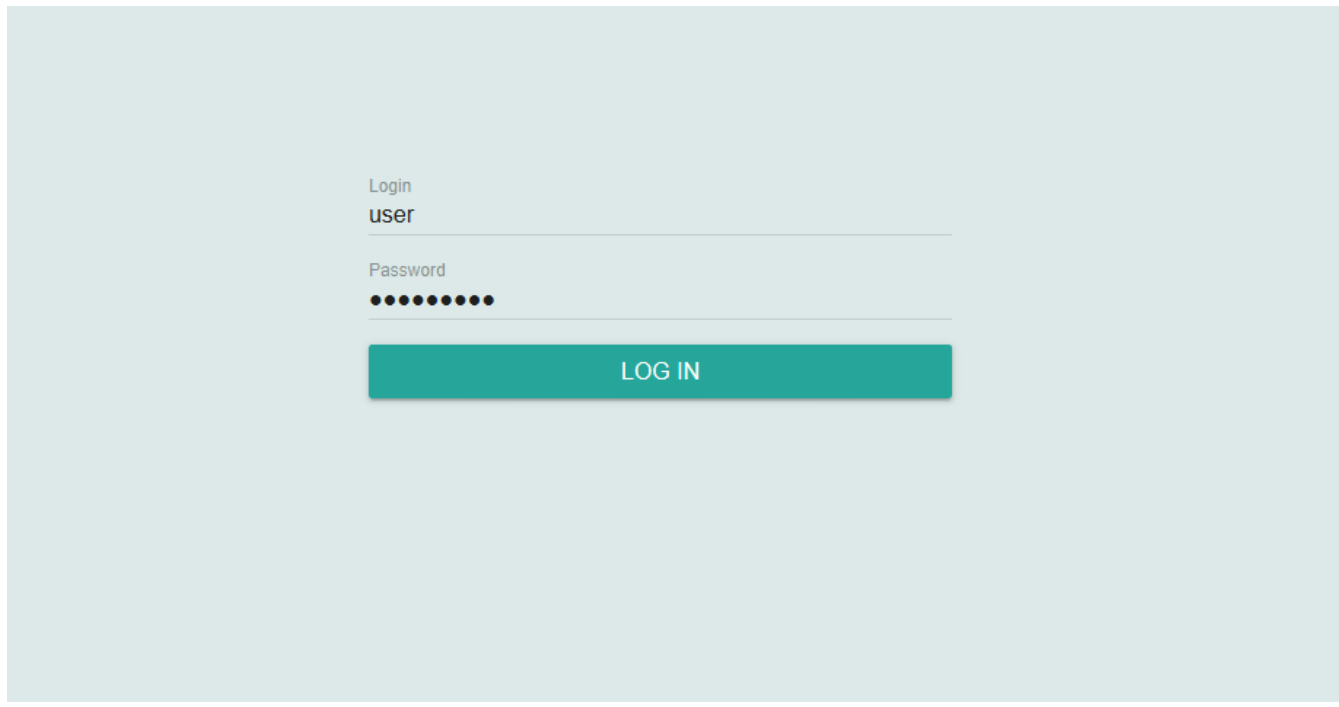
Рисунок 2.2 – Схема базы данных

3 Демонстрация веб-приложения на тестовых данных

В данной главе демонстрируются основные возможности разработанного веб-приложения. Весь исходный код данного приложения находится в свободном доступе, ознакомиться с которым можно, перейдя по ссылке <https://github.com/dniadzelka/routeBuilder>.

3.1 Вход в приложение

Введя в адресную строку браузера адрес веб-приложения, пользователь должен пройти процесс аутентификации, введя свои логин и пароль (Рисунок 3.1).



The image shows a login interface with a light blue background. In the center, there is a white rectangular box containing the login form. The form has two input fields: the first is labeled 'Login' and contains the text 'user'; the second is labeled 'Password' and contains a series of dots to mask the password. Below these fields is a teal button with the text 'LOG IN' in white capital letters.

Рисунок 3.1 – Форма ввода логина и пароля

После того, как пользователь успешно ввел логиин и пароль, он перенаправляется на страницу с картой, динамическая часть адреса которой является «/map». В правом верхнем углу браузера можно найти текущего пользователя, под которым была выполнена аутентификация в приложение.

3.2 CRUD операции с банкоматами на карте города

CRUD (сокращенно от англ. *create, read, update, delete* – «создать, прочесть, обновить, удалить») – акроним, обозначающий четыре базовые функции, используемые при работе с данными.

На странице изображена карта города с кликабельными отметками, которые отвечают за месторасположения банкоматов (Рисунок 3.2).

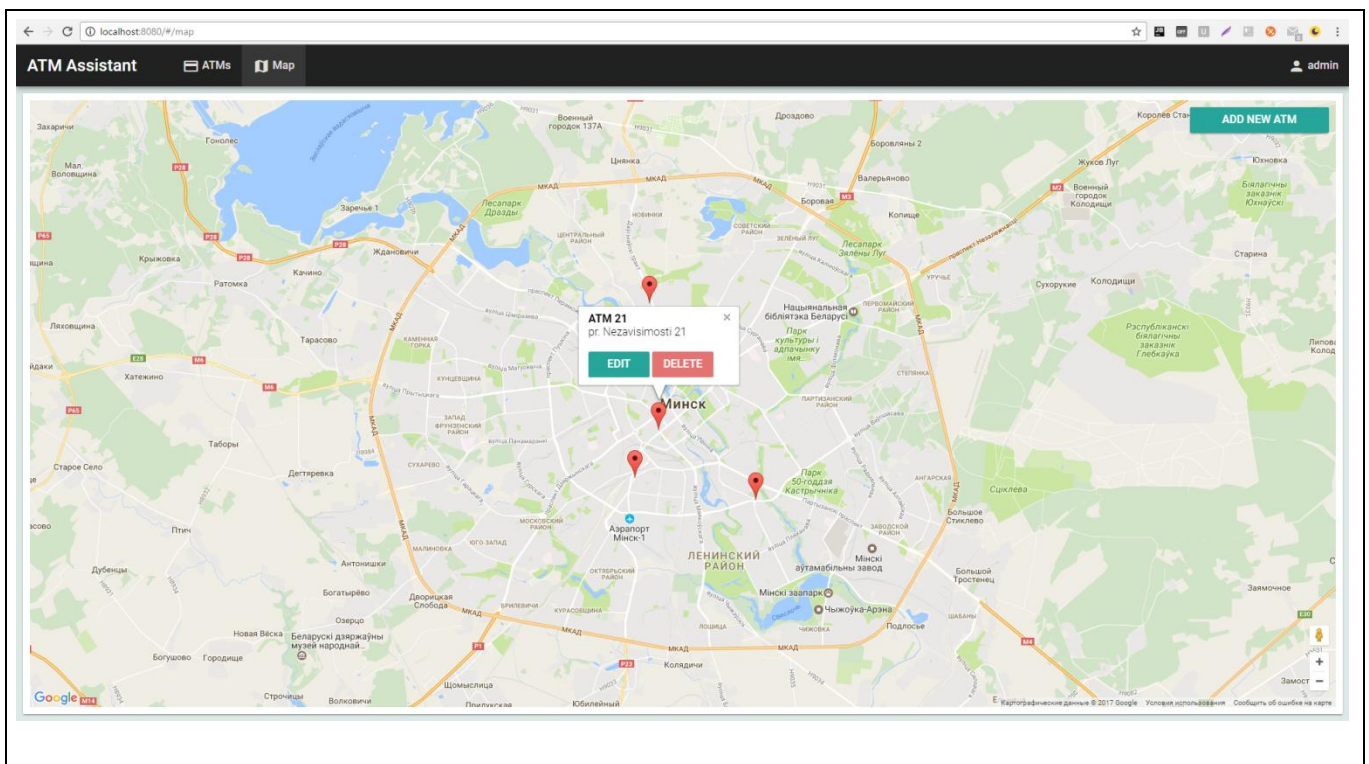


Рисунок 3.2 – Карта с месторасположениями банкоматов

В правом верхнем углу расположена кнопка добавления нового банкомата на карту – «ADD NEW ATM», кликнув на которую, появляется модальное окно (Рисунок 3.3), которое информирует о том, что для начала нужно задать месторасположение нового банкомата, выбрав соответствующую локацию на карте.

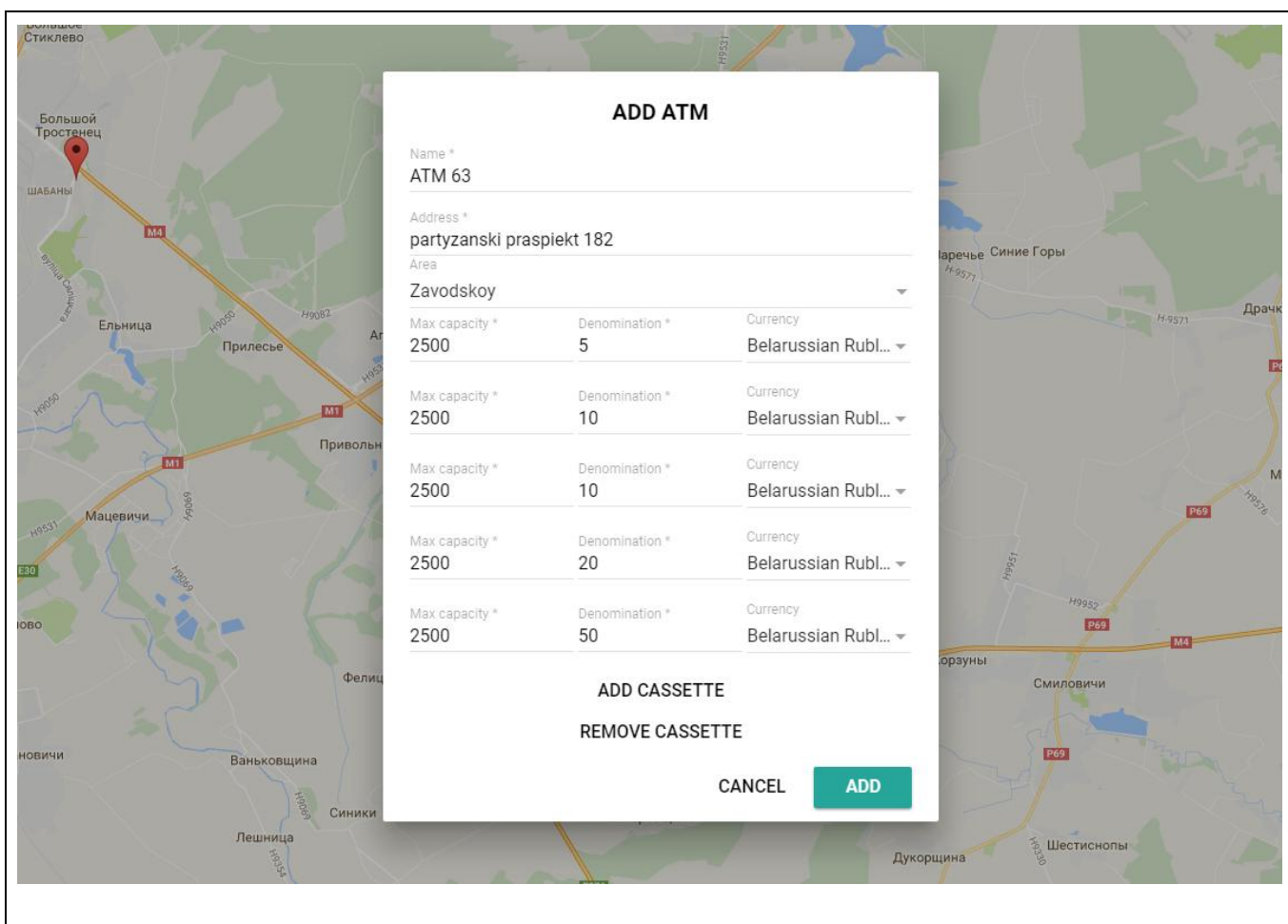


Рисунок 3.4 – Модальное окно для добавления банкомата

Данное модальное окно предлагает ввести следующие поля:

- «Name» – название банкомата;
- «Address» – адрес банкомата;
- «Area» – район, в котором находится банкомат;
- «Max Capacity» – максимальное число купюр, вмещающееся в кассету;
- «Denomination» – номинал купюры, находящееся в кассете;
- «Currency» – валюта, находящееся в кассете;

Так как в каждом банкомате находится от 4 до 6 кассет с банкнотами, пользователь может добавить кассеты, используя кнопку «ADD CASSETTE» или убрать, используя кнопку «REMOVE CASSETTE».

После ввода обязательных полей, помеченных знаком (*), пользователь нажимает кнопку «ADD».

Также пользователь имеет возможность отредактировать или удалить информацию о существующем банкомате. Для этого необходимо кликнуть на интересующий банкомат на карте, после чего появляется информация о банкомате,

а также возможные действия на нем – кнопки «EDIT» и «DELETE». При нажатии на кнопку «EDIT» появляется модальное окно, аналогичное модальному окну, используемого при добавлении банкомата (Рисунок 3.4), но уже с текущими данными.

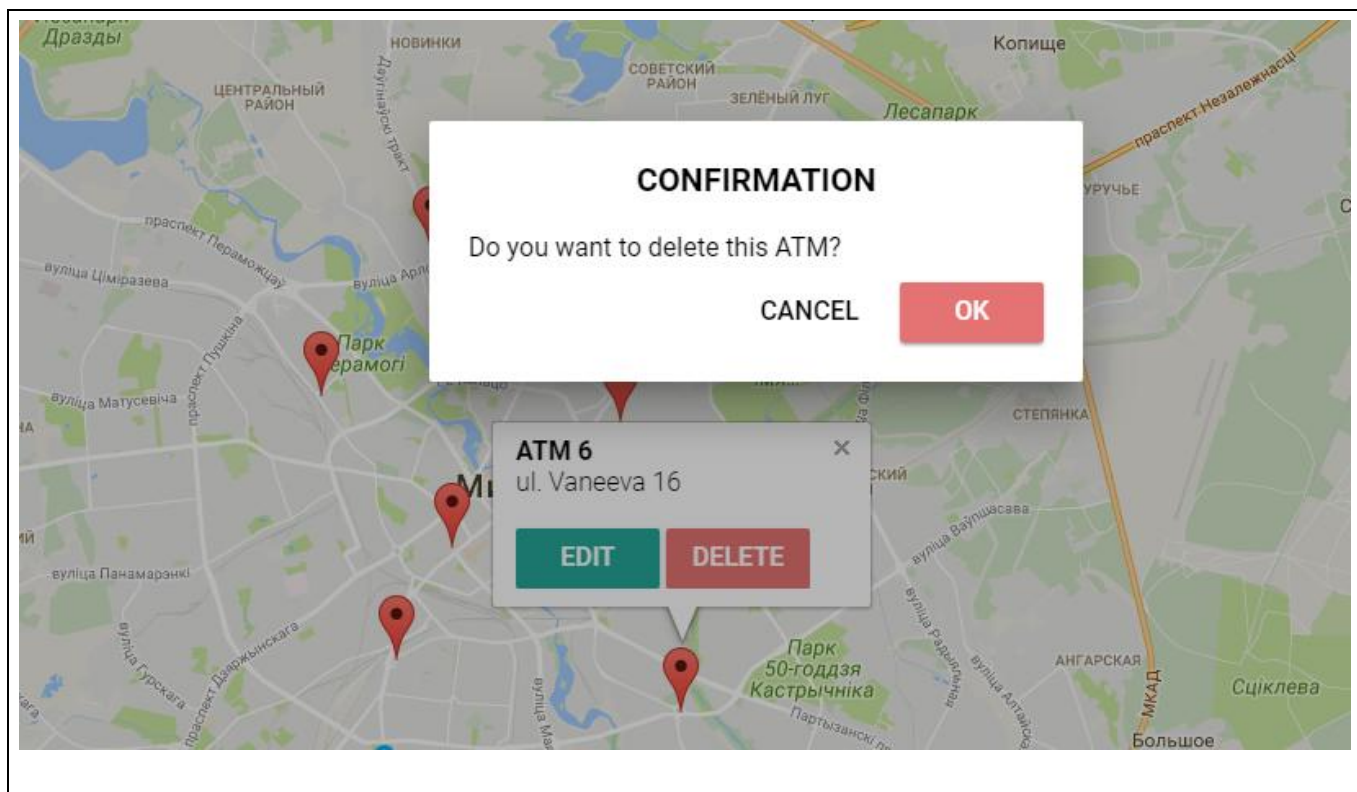


Рисунок 3.5 – Модальное окно для подтверждения удаления банкомата

При клике на кнопку «DELETE», пользователю требуется подтвердить свое действие, кликнув «ОК» на всплывшем модальном окне. После успешной обработки запроса сервером, банкомат считается удаленным из системы и больше не отображается на карте.

3.3 Просмотр таблицы банкоматов

Кликнув на вкладку «ATMs», пользователь попадает на страницу с таблицей банкоматов (Рисунок 3.6), динамическая часть адреса которой является «/atms».

ATM	Address	Area	Status	Amount	Currencies
ATM 44	Timiraziva 44	Zavodskoy	Regular	63	
ATM 6	ul. Vaneeva 16	Zavodskoy	In Queue	4	
ATM 21	pr. Nezavisimosti 21	Zavodskoy	Regular	88	
ATM 9	ul. Samsonava 13/4	Zavodskoy	Regular	24	
ATM 10	pr. Nezavisimosti 57	Zavodskoy	Regular	40	
test33	test33	Zavodskoy	Regular	15	
ATM 63	partyzanski praspiekt 182	Zavodskoy	Regular	54	

Рисунок 3.6 – Таблица банкоматов

На данной странице слева расположен список фильтров:

- «Amount From, %» – процент денежных средств, начиная с которого будут отображаться банкоматы в списке;
- «Amount To, %» – процент денежных средств, до которого будут отображаться банкоматы в списке;
- «Status» – статус, который отображает текущее состояние банкомата;
- «Area» – район, в котором находится банкомат.

Возможные значения фильтра для статусов:

- «All Statuses» – все статусы;
- «Regular» – банкомат работает в штатном режиме;
- «In Queue» – банкомат нуждается в пополнении денежными средствами;
- «Processing» – для банкомата формируются кассеты;
- «Ready For Delivery» – кассеты сформированы, готовы к отправке;
- «Delivery» – кассеты переданы инкассаторской бригаде.

3.4 Статистика выбранного банкомата

Кликнув на любую строку, соответствующую какому-либо банкомату, пользователь попадает на страницу, на которой отображена статистика снятия денежных средств по дням за период, заданный в фильтре слева (Рисунок 3.7).



Рисунок 3.7 – Статистика снятия денежных средств банкомата

Наведя курсор мыши на интересующую координату на графике, показываются данные, соответствующие дню, ближайшему к курсору. Таким образом работники банка могут наглядно видеть статистику снятия денежных средств и корректировать входные данные правил попадания банкомата в очередь.

3.5 Изменения правила попадания банкомата очередь

У каждого банкомата есть правила, по которым он попадает в очередь на пополнение денежными средствами. Их можно изменить, выбрав интересующие банкоматы в таблице и кликнув на кнопку «RULES» в правом верхнем углу. После чего откроется модальное окно, в котором можно изменить правила для выбранных банкоматов (Рисунок 3.8).

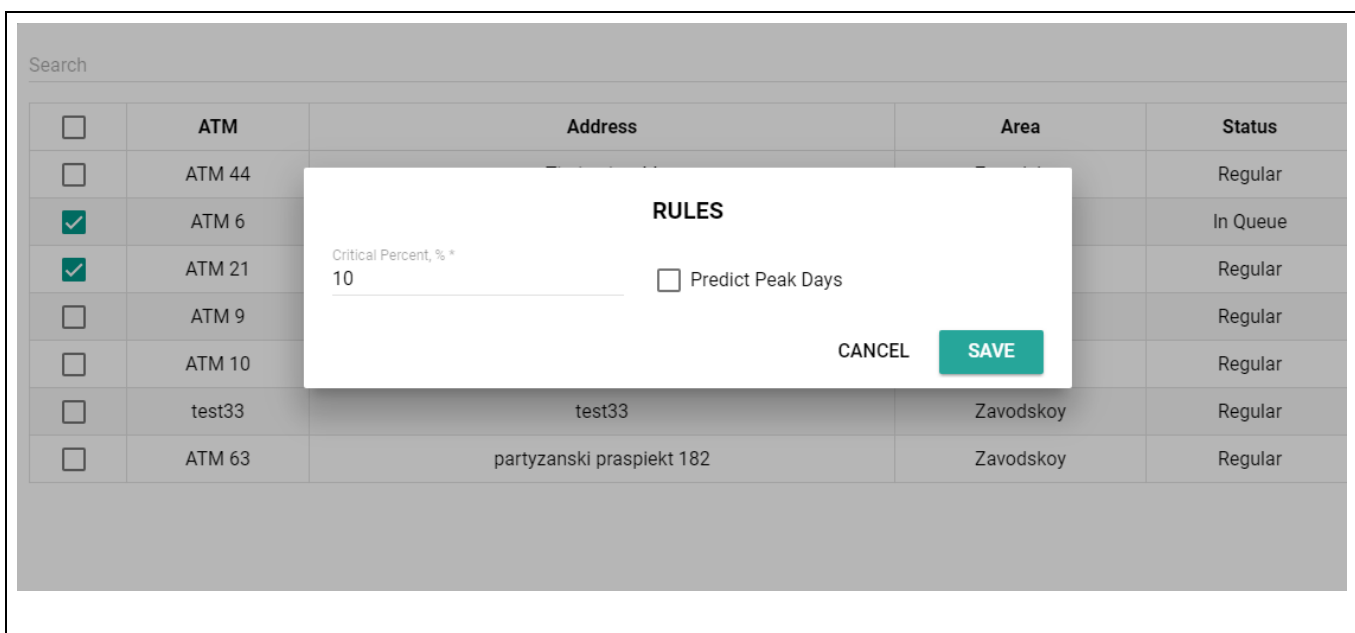


Рисунок 3.8 – Модальное окно для правил

Поставив галочку в поле «Predict Peak Days» модально окно перейдет в режим задания параметров для расчета пиковых дней (Рисунок 3.9).

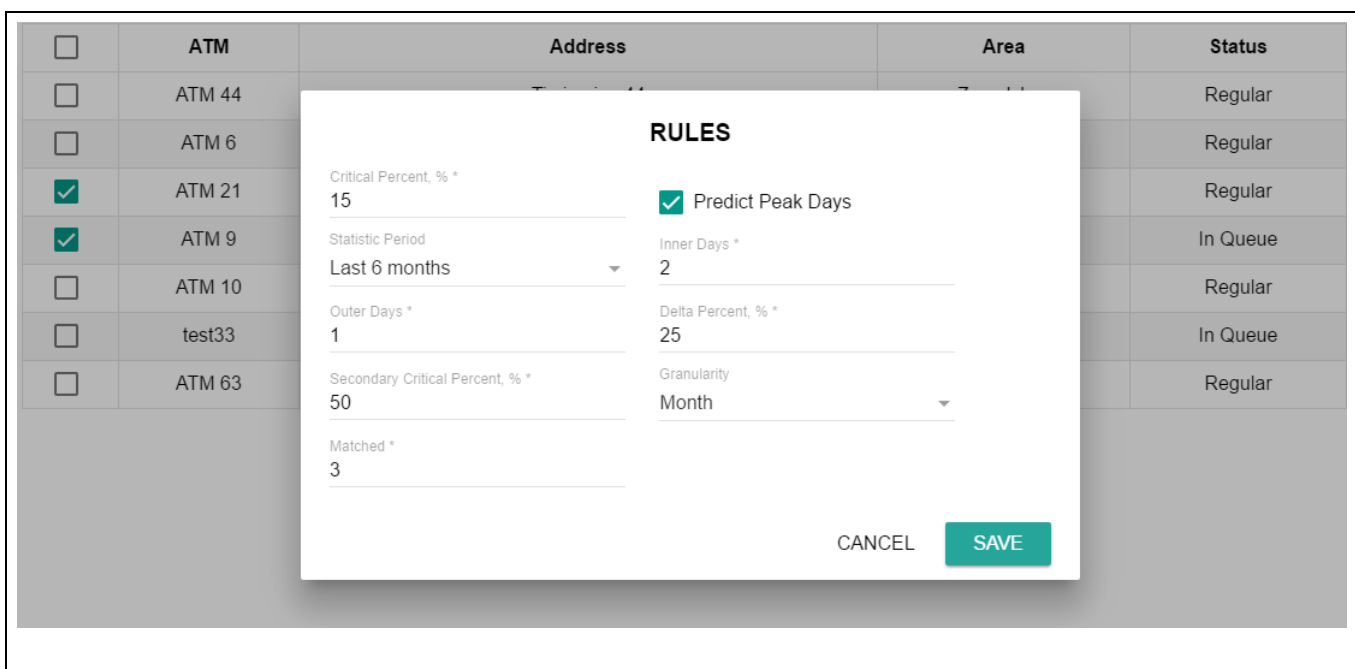


Рисунок 3.9 – Модальное окно для правил прогнозирования пиковых дней

Подробную информацию о значении каждого поля данного модального окна можно получить в описании алгоритма попадания банкомата в очередь на обслуживание (Алгоритм попадания банкомата в очередь).

3.6 Печать отчетов

Часто некоторым работникам банка требуются отчеты в печатном виде с информацией о банкоматах. Для печати необходимо перейти на вкладку «ATMs», отфильтровать список банкоматов, выбрав нужные позиции фильтров на левой панели и нажать кнопку «PRINT» в правом верхнем углу. После чего в браузере откроется окно с изображением, которое будет напечатано (Рисунок 3.10).

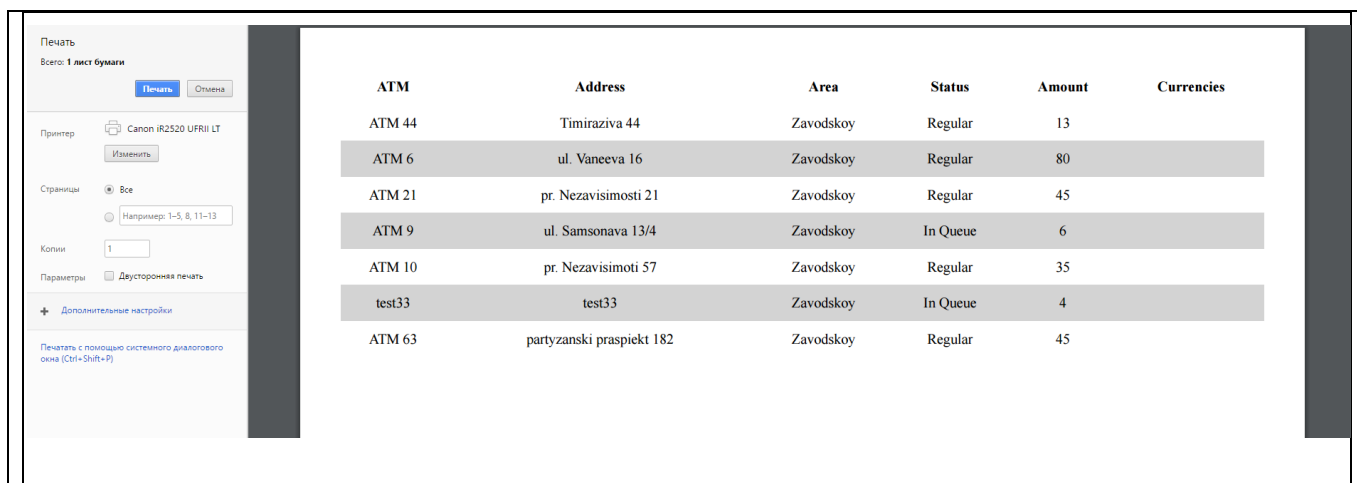


Рисунок 3.10 – Всплывающее окно печати в браузере

ЗАКЛЮЧЕНИЕ

В данной дипломной работе были изучены проблемы, возникающие в процессе наполнения банкоматов денежными средствами. Заблаговременное формирование списка банкоматов, в которых закончатся денежные средства, прогнозирование массового снятия наличных плодотворно влияет на лояльность клиента. Было разработано веб-приложение, помогающее решить перечисленные выше проблемы путем заблаговременного и наглядного оповещения о необходимости пополнить банкоматы денежными средствами. Освоены современные технологии веб-разработки, фреймворки. Были применены лучшие практики при реализации проекта, его структуризация. Все это было сделано для того, чтобы в дальнейшем было легко поддерживать продукт, развивать его, решать задачи, которые будут продиктованы бизнесом в будущем.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Angular Documentation [Electronic resource]. – Mode of access: <https://angular.io/docs/ts/latest/>. – Date of access: 05.04.2017.
2. Дэвид Флэнаган, JavaScript. Подробное руководство / Дэвид Флэнаган. – 6-е издание, серия O'Reilly. – Символ-Плюс, 2012. – 20 с.
3. Webpack Documentation [Electronic resource]. – Mode of access: <https://webpack.js.org/configuration/>. – Date of access: 05.04.2017.
4. TypeScript Documentation [Electronic resource]. – Mode of access: <https://www.typescriptlang.org/docs/tutorial.html>. – Date of access: 05.04.2017.
5. Steve Fenton, Pro TypeScript. Application-Scale JavaScript Development / Steve Fenton. – APPRESS, 2016. – 17 p.
6. Material Components [Electronic resource]. – Mode of access: <https://material.angular.io/components>. – Date of access: 05.04.2017.
7. Node.js Documentation [Electronic resource]. – Mode of access: <https://nodejs.org/en/docs/>. – Date of access: 05.04.2017.
8. Mario Casciaro, Node.js Design Patterns / Mario Casciaro, Luciano Mammino. – Second Edition. – Packt Publishing Ltd., 2016. – 23 p.
9. Express Documentation [Electronic resource]. – Mode of access: <https://expressjs.com/en/api.html>. – Date of access: 05.04.2017.
10. Mongo DB Documentation [Electronic resource]. – Mode of access: <https://docs.mongodb.com/manual/>. – Date of access: 05.04.2017.