МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА СЕРВИСОВ И МОБИЛЬНЫХ ПРИЛОЖЕНИЙ УПРАВЛЕНИЯ ОЧЕРЕДЯМИ ДЛЯ ПОСТАВЩИКОВ УСЛУГ

АВТОРЕФЕРАТ МАГИСТЕРСКОЙ РАБОТЫ

студента 2 курса 273 группы
направления 02.04.03 — Математическое обеспечение и администрирование
информационных систем
факультета КНиИТ
Емельянова Михаила Алексеевича

Научный руководитель
к. ф.-м. н., доцент _____ А. С. Иванов

Заведующий кафедрой
к. ф.-м. н., доцент _____ А. С. Иванов

Саратов 2020

ВВЕДЕНИЕ

Актуальность темы. С развитием экономической отрасли у организаций появилась необходимость в обработке множественных потоков таких запросов, как клиенты или поставки продукций. Очереди довольно распространены и встречаются при телефонном сообщении, в магазине, на заправке, в компьютерных системах и т. д.

С развитием информационных технологий и глобальном покрытии сетью Интернет всей планеты, появилась возможность обрабатывать запросы, поступившие непосредственно недавно и из разных мест, с минимальной задержкой во времени. Это позволяет организациям, использующим удаленное предоставление своих услуг и контроль обработки очередей запросов, получить преимущество над конкурентом.

Известно, что очереди, как скопление запросов, требующих обработки, влекут за собой дополнительные убытки. Таким образом, оптимизация обработки запросов в очередях позволяет получить конкурентное преимущество на рынке, повысить контроль за работой и прибыльность компании.

Именно в этом и призваны помочь автоматизированные системы управления очередями. Это системы, которые позволяют поставщику услуг эффективно регулировать поток клиентов. На российском рынке распространены локальные системы управления очередями, оперирующими потоком клиентов непосредственно в месте предоставления услуги. Такие системы разрабатываются исключительно под нужды конкретного заказчика и не имеют возможности расширения на другие типы услуг, а также отсутсвует возможность удаленной записи в очереди и контроля состояния.

Цель магистерской работы — разработка системы управления очередями для поставщиков услуг.

Поставленная цель определила **следующие задачи:**

1. Проанализировать литературу, посвященную основам реализации ПО управления очередями, клиент-серверной архитектуре систем, а также созданию мобильных клиентских приложений.
2. Провести обзор инструментальных средств, использующихся при создании целевого ПО.
3. Разработать серверную часть системы.
4. Разработать фронт-клиент системы.

5. Разработать мобильное приложение.
6. Интегрировать составные части и отладить работу системы.

Методологические основы разработки клиент-серверных систем и их особенности, а также создания мобильных приложений представлены в работах Р. Т. Филдинга [1], А. Родригеса [2], Г. Краснера [3], Дж. МакВертера [4].

Практическая значимость магистерской работы. В ходе выполнения практической части магистерской работы была реализована заявленная система управления очередями для поставщиков услуг. В систему входят серверная часть, база данных, сайт, а также мобильное клиентское приложение. Система была спроектирована с учетом обобщения понятия услуги, а также предполагает особую возможность для клиентов очереди — опоздание. Для этого разработан алгоритм обработки, основанный на очереди с ожиданием и относительным приоритетом.

Разработанная система подразумевает дальнейшее развитие проекта в сторону расширения возможностей и использования других более современных и гибких технологий. Система не имеет аналогов на российском рынке, так как предоставляет возможность обобщить услуги и включить в нее разных поставщиков услуг. Также система удаленная, что позволяет клиентам заранее записываться в очереди, планировать свое время и отслеживать свои позиции в выбранных очередях на услуги. В то же время партнеры имеют возможность управлять отдельными услугами в удаленных офисах и контролировать поток клиентов.

Структура и объем работы. Магистерская работа состоит из введения, 3-х разделов, заключения, списка использованных источников и 1-го приложения. Общий объем работы — 65 страниц, из них 47 страниц — основное содержание, включая 28 изображений, список использованных источников информации — 22 наименования.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Обзор литературы» посвящен обзору существующей литературы и источников, посвященных разработке клиент-серверных систем, мобильных приложений, а также уже существующих решений.

«Основы реализации ПО управления очередями» — в этом подразделе содержится обзор литературы, посвященной разработке непосредственно систем управления очередями. [5–7]

«Основы создания мобильных приложений» — в этом подразделе содержится обзор книг, статей и руководств, посвященных созданию мобильных приложений, в которых описываются как жизненный цикл разработки, так и ее особенности. [4, 8, 9]

Второй раздел «Обзор инструментальных средств» посвящен рассмотрению инструментов, необходимых для реализации приложения. В нем рассмотрены современные инструменты разработки, обоснован выбор и описаны их особенности. Определены следующие инструменты разработки:

- PostgreSQL 10;
- Java Development Kit 8;
- Spring Boot;
- Maven;
- Hibernate;
- Angular;
- Ionic, Capacitor и Cordova;
- IntelliJ IDEA;
- Visual Studio Code;
- Android Studio(Android SDK).

Раздел состоит из нескольких подразделов.

«Архитектура клиент-сервер» — в этом подразделе содержится описание архитектуры клиент-сервер. Такая организация позволяет создавать распределенные системы и балансировать нагрузку на приложение. В подразделе представлены общие принципы создания такого ПО.

«RESTful» — в этом подразделе содержится разбор технологии REST сообщения между компонентами разрабатываемой системы. Технология REST — это способ проектирования Web-сервисов, менее зависимый от закрытого промежуточного программного обеспечения, чем модели SOAP и WSDL.

Представление системных ресурсов через RESTful API — это гибкий способ обеспечения различных приложений данными в стандартном формате. В подразделе описаны примеры этого подхода, а также лучшие практики написания RESTful API. [2, 10]

«Angular фреймворк» — в этом подразделе содержится описание технологии Angular, необходимой для написания фронт-клиентов. Angular — это платформа и фреймворк для создания клиентских приложений с помощью HTML и TypeScript. Основные элементы Angular-приложения это NgModule модули, которые предоставляют контекст для компиляции компонентов. NgModule модули объединяют в себе код в функциональный набор. Само приложение на Angular можно определить множеством его NgModule модулей. Компоненты определяются их представлением, которые являются набором элементов, отображаемых на экране, которые Angular может подхватывать и изменять согласно логике приложения или данным. Компоненты в свою очередь используют сервисы, которые предоставляют определенный функционал, напрямую не связанный с представлением данных. Сервисы могут быть внедрены в компоненты в качестве зависимостей, с помощью чего повышается переиспользование кода, а также эффективность разработки. Компоненты приложения обычно определяют множество представлений, составляющих иерархию. Для этого в Angular предусмотрены специальный Router (маршрутизатор) сервис, который помогает в определении путей для отображения конкретных представлений. Такой маршрутизатор предоставляет исчерпывающий набор возможностей для навигации в любом браузере. [11]

Организация кода в виде отдельных функциональных модулей предоставляет возможность лучше контролировать разработку сложных приложений. Также такая техника позволяет использовать ленивую инициализацию таким образом, чтобы загружать модули по требованию и минимизировать количество кода, который подгружается на старте. Перед тем как представление отрисовано Angular вычисляет директивы и связки в шаблонах для модификации HTML элементов и DOM дерева, согласно программной логике и данным. Angular поддерживат двустороннюю привязку, которая также отображает изменения в DOM дереве в данных (переменных) программы. Основным качеством Angular является предоставляемая им возможность реализации внедрения зависимостей(Dependency Injection). Она позволяет делегировать функционал

компонент сервисам.

Все эти ключевые особенности Angular позволяют использовать его в качестве гибкого фреймворка для разработки клиентских приложений.

«Нативная мобильная разработка, NativeScript и Ionic» — в этом подразделе содержится описание проблем, связанных с разработкой мобильных приложений, а также выбранное решение. Решением проблемы создания кроссплатформенного мобильного ПО стал выбор для разработки гибридного приложения. Гибридное приложение является унифицированным веб-приложением, обернутым в нативную оболочку. В подразделе были рассмотрены одни из самых развивающихся для этого проектов — NativeScript и Ionic. Были описаны их особенности и обоснован выбор инструмента Ionic. [12, 13]

Третий раздел «Описание реализованной системы» посвящен описанию спроектированной и разработанной системы, а также демонстрации примеров ее функционирования. Представлены снимки экрана браузера и разработанного мобильного приложения.

Раздел состоит из нескольких подразделов.

«Описание схемы базы данных» — этот подраздел посвящен разработанной схеме сущностей, необходимых для создания целевого программного обеспечения. С точки зрения архитектуры, должны существовать фронт клиенты для пользователей, включая устройства под управлением Android, а также серверная часть и база данных. Партнеру доступны регистрация, аутентификация, а также управление очередями и сервисами. Под этим понимается определение офисов, предоставляющих услуги, самих услуг, а также обслуживание клиентов очереди. Клиент имеет возможность зарегистрироваться, авторизоваться, а также просматривать доступные очереди на предоставляемые партнерами услуги и записываться на них с отслеживанием своей позиции.

Таким образом можно определить какие сущности необходимы для реализации данного функционала. Исходя из этого была сформирована схема базы данных(см. рис. 1).

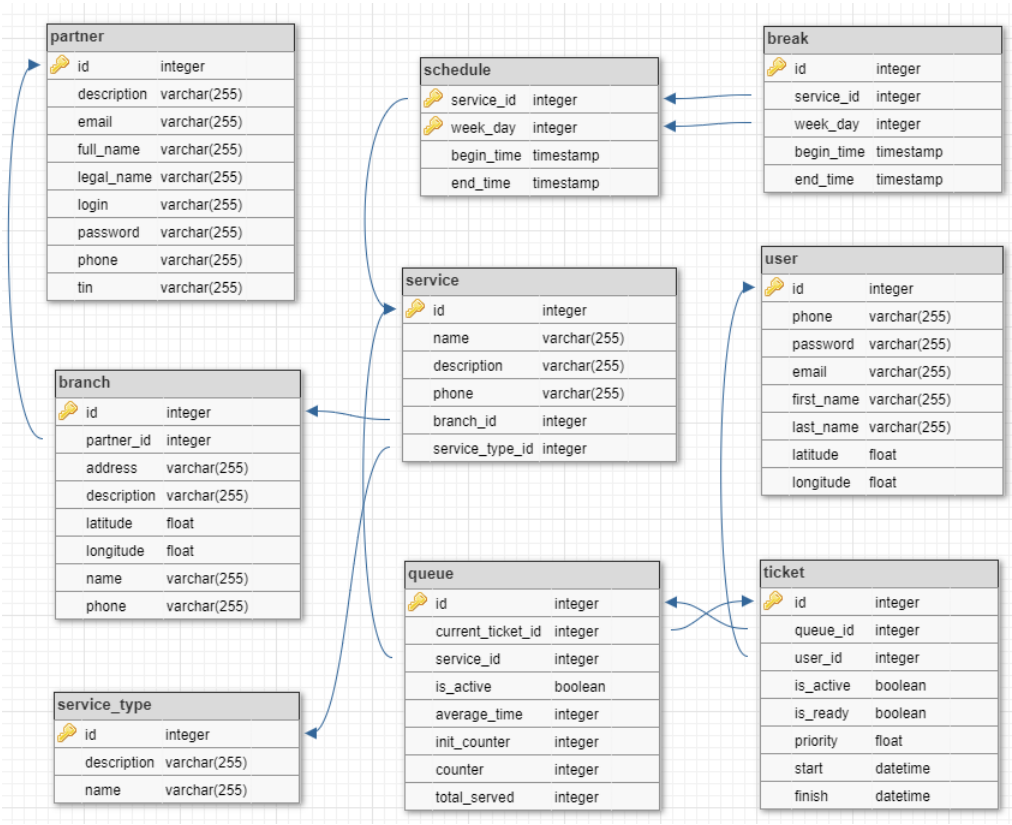

Рисунок 1 – Схема базы данных

«Описание разработанной серверной части» — подраздел, в котором описан ход разработки серверной части целевого ПО. В качестве языка разработки был выбран Java. Для написания проекта была построена абстрактная объектная модель, которая в точности повторяет схему базы данных, представленной ранее(рис. 1). На самом деле именно она реализована средствами Java и Spring с помощью POJO классов.

Для структуры проекта был выбран паттерн Model-View-Controller.

Приложение логически разделено на следующие модули:

1. `config` — отвечает за конфигурацию, в данной реализации содержит в основном настройку аутентификации;
2. `model` — отвечает за представление сущностей, сопоставляемых с данными из БД;
3. `repo` — в нем представлены интерфейсы, с описанием логики чтения и обработки данных из БД посредством запросов;
4. `service` — отвечает за реализацию бизнес-логики;
5. `controller` — отвечает за маппинг REST запросов для взаимодействия с приложением.

На данном этапе развития проекта сама логика обработки заявок в оче-

реди реализована на подобии с СМО с ожиданием и относительным приоритетом:

- Класс `Ticket` кроме прочего имеет флаг готовности, говорящий о том, сообщил ли клиент о своем опоздании, а также приоритет заявки(тикета).
- Изначально при добавлении заявки поле приоритета приравнивается значению счетчика очереди, после чего тот уменьшается на 1, но не менее 0.
- Предоставление услуги осуществляется для тикетов, поле готовности которых истинно. Заявка, обрабатываемая в данный момент, обслуживается полностью до конца.
- В случае, когда клиент оповещает об опоздании, поле готовности становится ложным, а его приоритет уменьшается на 1.5.
- При каждой успешной обработке тикета, его флаг активности обслуживания становится ложным, что свидетельствует о завершении предоставлении услуги и историчности объекта. У всех заявок, не готовых к обслуживанию (помеченных опаздывающими) приоритет уменьшается на 1.

Таким образом опаздывающие клиенты медленно перемещаются вниз в очереди согласно приоритету, уступая свое место до тех пор, пока не оповестят о своей готовности к обслуживанию.

«Описание разработанного фронт-клиента» — в этом подразделе содержится обзор созданного сайта для клиентов и поставщиков услуг. В нем определен требуемый функционал, такой как разграничение прав доступа и возможность авторизации. Исходя из этой основополагающей концепции, произведено разбиение предполагаемого набора возможностей на подмножества. Определены три роли, для каждой из которых общий набор функционала может различаться:

- посетитель,
- авторизованный пользователь,
- авторизованный поставщик услуг.

Посетитель обладает наименьшим доступом к ресурсу, в то время как пользователю или поставщику услуг должны быть доступны расширенные возможности. Например, возможности посетителя должны быть ограничены следующим образом:

— возможность авторизоваться;
— возможность зарегистрироваться в системе;
— просмотр карты партнеров и информации о предоставляемых ими услугах.

Авторизованный пользователь приобретает следующие дополнительные возможности:

— просмотр отдельных услуг, предоставляемые поставщиками услуг;
— просмотр и управление личными данными в собственном профиле;
— запись, отслеживание позиции в очередях, оказывающих услуги.

Авторизованный поставщик услуг отличается от пользователя. В сравнении с посетителем он приобретает следующие возможности:

— просмотр существующих отдельных услуг, предоставляемых другими поставщиками;
— управление личными данными в собственном профиле;
— создание, изменение и удаление оказываемых собственных услуг;
— управление существующими очередями по собственным оказываемым услугам.

Главная страница разделена на навигационное меню и блок отображения основного содержимого. Навигационное меню разделено на кнопки, позволяющие совершать переход по основным страницам системы. Переход осуществляется в рамках блока с основным содержимым, который должен подменяться актуальным контентом согласно однозначно определенным путям переходов. В процессе реализации были созданы сущности, отражающие объекты, с которыми необходимо оперировать приложению. Была реализована регистрационная форма в двух вариантах — для пользователя или для поставщика услуг. Неавторизованные посетители могут зайти в систему с помощью формы входа, как в качестве клиента, так и в качестве поставщика услуг, если они таковым являются. Посетителям и доступен простейший каталог с описанием существующих типовых услуг, предоставляемых поставщиками услуг, зарегистрированных в системе. Авторизованным пользователям доступен поиск услуг по названию или поиск партнеров по наименованию организации. При переходе на выбранную услугу отображается окно, содержащее информацию о расписании предоставления услуги и об очереди на нее. На это же окно с отображением информации по услуге авторизованные пользователи мо-

гут перейти из пункта меню «мои очереди», если они являются участниками очередей на эти услуги.

«Описание разработанного мобильного приложения» — подраздел посвящен спроектированному и реализованному мобильному приложению. Целевой аудиторией разрабатываемого мобильного приложения являются клиенты, желающие получить обслуживание в очередях. Исходя из этого для приложения была реализована авторизация клиента с возможностью регистрации аккаунта в форме. Функционал, которым должно обладать приложение:

- авторизация и регистрация;
- управление личными данными в собственном профиле;
- отслеживание очередей, в которых пользователь является участником, и управление своим состоянием в них;
- специализированный удобный поиск услуг по различным фильтрам;
- дополнительный нативный функционал оборудования, такой как навигация к месту оказания услуги через предустановленное приложение.

Была создана структура предполагаемого набора состояний приложения. Она отображена на диаграмме 2. Каждое состояние является представлением с характерным ему набором функционала. Из каждого состояния возможен возврат на предыдущий шаг при нажатии кнопки «назад» на устройстве.

Рисунок 2 – Диаграмма состояний мобильного приложения

Разработка велась с помощью выбранных фреймворков Angular и Ionic. Для отладки и компиляции нативного приложения с помощью сгенерированных gradle скриптов была использована среда разработки Android Studio. Приложение развертывалось на личном устройстве с системой Android — Samsung Galaxy A7.

Благодаря выбранным инструментов стало возможным переиспользовать часть кода, использованной для создания веб-сайта. В число переиспользованного кода входят классы сервисом, организующие REST-запросы к серверу и модели сущностей. Структура программы состоит из страниц (page), генерируемых с помощью команды `ionic g page`. Эти страницы изначально состоят из следующих файлов: тест-класса, класса-компонента, html-файла шаблона, стилевого scss-файла и класса модуля Angular этой страницы. Таким образом каждая страница это модуль с потенциально своим набором компонентов.

Разработанное приложение позволяет клиентам зарегистрировать новый аккаунт или авторизоваться в системе, после чего при подтверждении подлинности токена приложение открывает доступ к основному меню на главной странице. С главной страницы клиенту доступны переходы на его профиль,

отслеживаемые очереди, где он является участником, а также переход на поиск услуг и очередей где есть выбор поиска по партнеру или типам услуг и впоследствии услугам. При выборе партнера открывается меню со списком его офисов. Впоследствии при переходах открываются услуги и возможность записаться на очередь, после чего меню очереди будет доступно из опции «Мои очереди» на главной странице приложения.

ЗАКЛЮЧЕНИЕ

Результатом научно-исследовательской работы является спроектированная и реализованная система управления очередями для поставщиков услуг. Система построена на базе клиент-серверной архитектуры с фронт-клиентом и мобильным приложением в качестве клиентских частей. Разработка велась по гибкой методологии как с соблюдением техник на примере rest запросов, так и с использованием современных инструментов и технологий: Spring (Spring Boot), Angular, Ionic, Capacitor и Cordova, Android Studio и Android SDK.

Разработанная система имеет разграничение доступа для клиентов и поставщиков. Она предоставляет возможность клиентам искать подходящие услуги, записываться в очереди на их предоставление и отслеживать свой статус в них. Поставщики услуг имеют возможность отслеживать активность в своих офисах и управлять потоком клиентов в своих очередях. Система подразумевалась как свободно распространяемая и позволит мелким предпринимателям и компаниям наращивать своё присутствие в сети Интернет, а также удобным образом обрабатывать клиентские заявки на предоставление услуг, что также должно способствовать развитию этих компаний.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Fielding, R. T.* Architectural styles and the design of network-based software architectures. — 2000.

2 *Родригес, А.* Web-сервисы RESTful: основы / А. Родригес. — IBM Corporation, 2015.

3 *Krasner, G.* A cookbook for using the model - view controller user interface paradigm in smalltalk - 80 / G. Krasner, S. Pope // *Journal of Object-oriented Programming - JOOP*. — 01 1998. — Vol. 1.

4 *McWherter, J.* Professional Mobile Application Development / J. McWherter, S. Gowell. — Indianapolis: John Wiley & Sons, Inc., 2012.

5 The Definitive Guide to Queue Management Systems [Электронный ресурс]. — URL: `https://www.qminder.com/what-is-queue-management-system/` (Дата обращения 18.11.2018). Загл. с экр. Яз. англ.

6 *Kakengi, B.* Modern Queueing Management System: QCracker / B. Kakengi. — Helsinki: Helsinki Metropolia University of Applied Sciences, 2012.

7 *Uddin, M. N.* Automated queue management system / M. N. Uddin, M. Rashid, M. Mostafa, H. Belayet, S. Salam, N. Nithe, S. Ahmed // *Global Journal of Management and Business Research: Administration and Management*. — 2016. — Vol. 16. — Pp. 31–38.

8 Mobile Developer's Guide To The Galaxy. — Enough Software GmbH, 2017.

9 Introduction to the Mobile Software Development Lifecycle [Электронный ресурс]. — URL: `https://docs.microsoft.com/en-us/xamarin/cross-platform/get-started/introduction-to-mobile-sdlc` (Дата обращения 19.11.2018). Загл. с экр. Яз. англ.

10 *Otavio, F. F.* Semantic Web Services: A Restful Approach / F. F. Otavio, M. A. Ferreira. — University of Sao Paulo, 2009.

11 Introduction to Angular concepts [Электронный ресурс]. — URL: `https://angular.io/guide/architecture` (Дата обращения 10.10.2019). Загл. с экр. Яз. англ.

12 Developer Guides [Электронный ресурс]. — URL: https://developer.android.com/guide/ (Дата обращения 17.10.2018). Загл. с экр. Яз. англ.

13 How NativeScript Works [Электронный ресурс]. — URL: https://docs.nativescript.org/core-concepts/technical-overview (Дата обращения 12.03.2020). Загл. с экр. Яз. англ.