Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Кафедра «Информатика»
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

_____ И.В. Евдокимов
подпись инициалы, фамилия

«____» _____ 2018г.

БАКАЛАВРСКАЯ РАБОТА
09.03.04 «Программная инженерия»

Программная система анализа и проектирования показателей эффективности предприятия общественного питания

Руководитель	_____	_____	А.С. Кузнецов
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	_____		К. А. Елсуфьев
	подпись, дата		инициалы, фамилия
Выпускник	_____		В. В. Андони
	подпись, дата		инициалы, фамилия
Выпускник	_____		Т. М. Сырцова
	подпись, дата		инициалы, фамилия
Выпускник	_____		И. В. Филимонов
	подпись, дата		инициалы, фамилия
Нормоконтролер	_____	_____	О.А. Антамошкин
	подпись, дата	должность, ученая степень	инициалы, фамилия

Красноярск 2018

РЕФЕРАТ

Выпускная квалификационная работа по теме «Программная система анализа и проектирования показателей эффективности предприятия общественного питания» содержит 132 страницы текстового документа, 29 используемых источников, 82 иллюстраций, 1 таблицу и 1 формулу.

АНАЛИЗ ПОКАЗАТЕЛЕЙ ЭФФЕКТИВНОСТИ, ПРОГНОЗИРОВАНИЕ ПОКАЗАТЕЛЕЙ ЭФФЕКТИВНОСТИ, РАСЧЕТ СЕБЕСТОИМОСТИ ТОВАРОВ, ПРОГРАММА ЛОЯЛЬНОСТИ.

В нынешнее время Российский рынок общественного питания еще не так развит, как в Европе или США, но уже насчитывает более 88 тысяч предприятий. И с каждым годом эти цифры только увеличиваются. А значит, что больше и больше людей нуждаются в программах для анализа и прогнозирования работы своего предприятия.

Цель данной работы сводится к исследованию современного рынка и разработки программного продукта, способного анализировать и прогнозировать показатели эффективности предприятий.

Для достижения поставленной цели необходимо выполнить следующие задачи:

- изучение предметной области;
- анализ и сравнение конкурентных платформ в исследуемой области;
- разработка программного обеспечения для анализа и прогнозирования показателей эффективности предприятия общественного питания;
- интерпретация полученных результатов.

В результате была получена программная система для предприятий общественного питания, позволяющая проводить анализ и прогнозировать показатели эффективности.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В мире разработки программных продуктов системы анализа показателей эффективности предприятия играют большую рольв рыночной экономике.

Программные системы анализа показателей эффективности пользуются популярностью у малого и среднего бизнеса в области общественного питания, потому что на старте бизнеса очень важно уделять внимание действительно существенным показателям эффективности, которые в конечном итоге влияют на прибыльность компании.

Наш проект будет выполнен в виде веб-приложения[1] в целях облегчения распространения продукта. Как показывает практика, по состоянию на 2018 год, большинство программного обеспечения распространяется через среду Интернет.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. изучение предметной области;

2. анализ и сравнение конкурентных платформ в области предприятий общественного питания;

3. разработка программного обеспечения для анализа и прогнозирования показателей эффективности предприятия общественного питания;

4. интерпретация полученных результатов в понятные пользователю системы данные.

1 Обзор методов и средств анализа и прогнозирования показателей эффективности предприятия общественного питания

1.1 Анализ предметной области

В настоящее время общественное питание является одной из основных активно развивающихся отраслей социально-экономической деятельности. Данная отрасль имеет важную цель – удовлетворение материальных и иных потребностей пайщиков и потребителя. Главные функции общественного питания – производство, реализация и организация потребления.

Как правило,оплачиваемое питание является сферой товарного обращения, так как продукция производится потребителям в порядке обмена на их денежные средства. Следовательно, общественное питание входит в систему торговли, а его оборот является частью розничного товарооборота. Оборот общественного питания включает в себя следующие этапы: продажа товаров и производство собственной продукции.

На данный момент, большое значение в жизни человека имеют предприятия общественного питания. Основное назначение предприятий такого типа состоит в том, что они способны удовлетворять естественнуюпотребность человека и имеют возможность влиять на потребление продуктов питания. Для предпринимателей, общественное питание является перспективной и популярной отраслью, так как располагает рядом конкурентоспособныхкачеств. Основная цель общественного питания состоит также в том, что на предприятиях питания рационально расходуются продукты питания.

Основным направлением совершенствования бизнеса в сфере общественного питания является оперативная программная система с анализом и прогнозированием его бизнес процессов, способствующая выявлять недостатки организации и указывать на способы их ликвидации. Поэтому

подбор подходящего программного обеспечения сильно интересует владельцев бизнеса общественного питания.

В настоящее время в сфере общественного питания широко применяются программные системы для проектирования показателей эффективности. Инструментом для качественного анализа рейтинга эффективности является подготовленное ПО, которое соответствует не только конкретному заведению, но и любому предприятию по распространению общественного питания.

Программная реализация системы анализа и прогнозирования эффективности предприятия имеет ряд преимуществ:

1. своевременное получение результатов работы заведения и освобождение администратора от трудоемкой работы по обработке показателей цехов;

2. оценочная независимость;

3. информативные диаграммы исследований работы предприятия и автоматическое выявление пробелов в работе того или иного цеха.

Для дальнейшего анализа необходимо изучить сторонние программные системы на рынке.

Первым продуктом для сравнения является система автоматизации ресторанов «Iiko»[2]. Один из лидеров поставщиков программной системы в данном сегменте (Рисунок 1).

Рисунок 1–Система автоматизации ресторанов «Iiko»

Множественные направления в выборе программного обеспечения под определенные направления бизнеса. Предоставляется платная ежемесячная подписка от 2990 руб./месяц. Также доступны едино разовые индивидуальные покупки системы стоимостью от 55970 рублей. Платное обучение пользователей работе с ПО от 1600 рублей. Присутствуетверсия программной системы для настольных компьютеров, также версия для определенного вида устройств, представляющихся компанией своим клиентам и мобильная версия данного программного средства. Данная система предоставляет следующие возможности своим клиентам:

1. управление сетью ресторанов;
2. управление рестораном с обслуживанием у столов;
3. управление ресторанов с быстрым обслуживанием;
4. управление небольшим заведением;
5. управление службой доставки;
6. столовая в бизнес-центре или на предприятии.

Данный продукт является довольно популярным и часто применяемым у владельцев ресторанного бизнеса.

Вторым конкурентом в данном сегменте является программа для полноценного складского учета, удобного использования POS-терминалов на iPad, CRM и настраиваемой системы отчетов – «Quick Resto»[3].

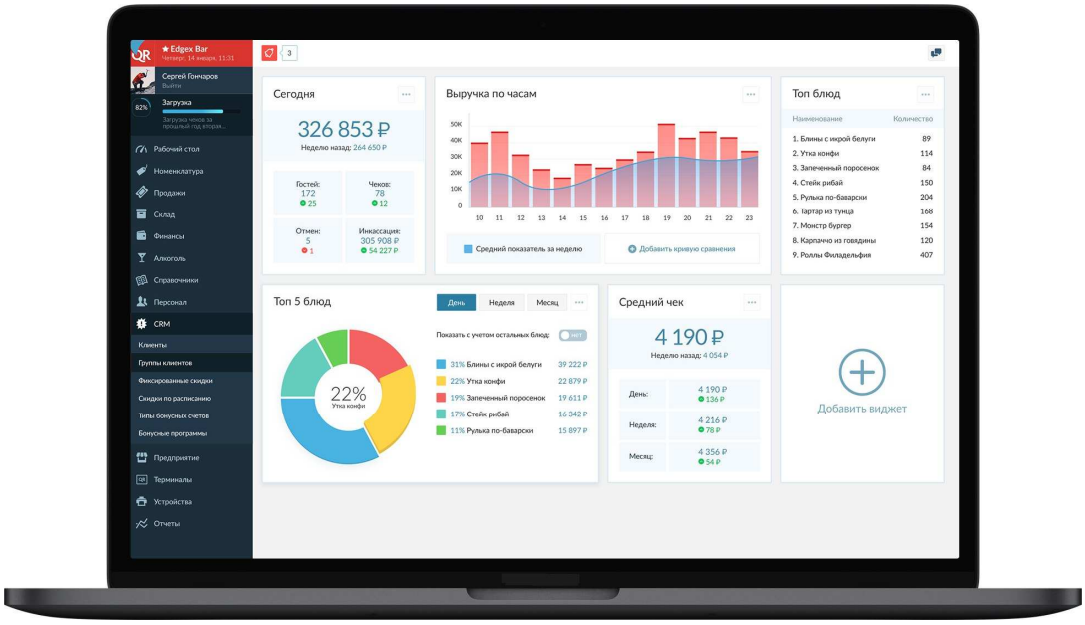

Рисунок 2– Система автоматизации «Quick Resto»

Один из ведущих представителей ПО для сферы аналитики эффективности общественного питания (Рисунок 2). Есть пробная версия продукта на 14дней. ПО предоставляется по платной подписке от 2490 руб./месяц. Есть шанс индивидуальной подписки. Стоимость предоставления оборудования для предприятия варьируется от 45900 руб. до 69900 руб. Для своих клиентов программа данная программная система предлагает следующие услуги:

1. полный контроль всего производства и конструктор отчетовдля руководителя;

2. полное описание программы лояльности и инструкции для интеграции с другими платформами маркетологу;

3. финансовый модуль и полная интеграция с 1С для бухгалтера;

4. настоящий производственный и складской учет с описанием типов номенклатур для товароведа или кладовщика;

5. кухонный экран и подробности заказа повару;

6. периферийное оборудование для работников зала.

Данный сервис «Quick Resto» предлагает сделать автоматизацию ресторанного бизнеса простой, недорогой и удобной для работников и руководства.

Третьим объектом сравнения является удобная система автоматизации для ресторанов и предприятий индустрии развлечений «Tillypad»[4].

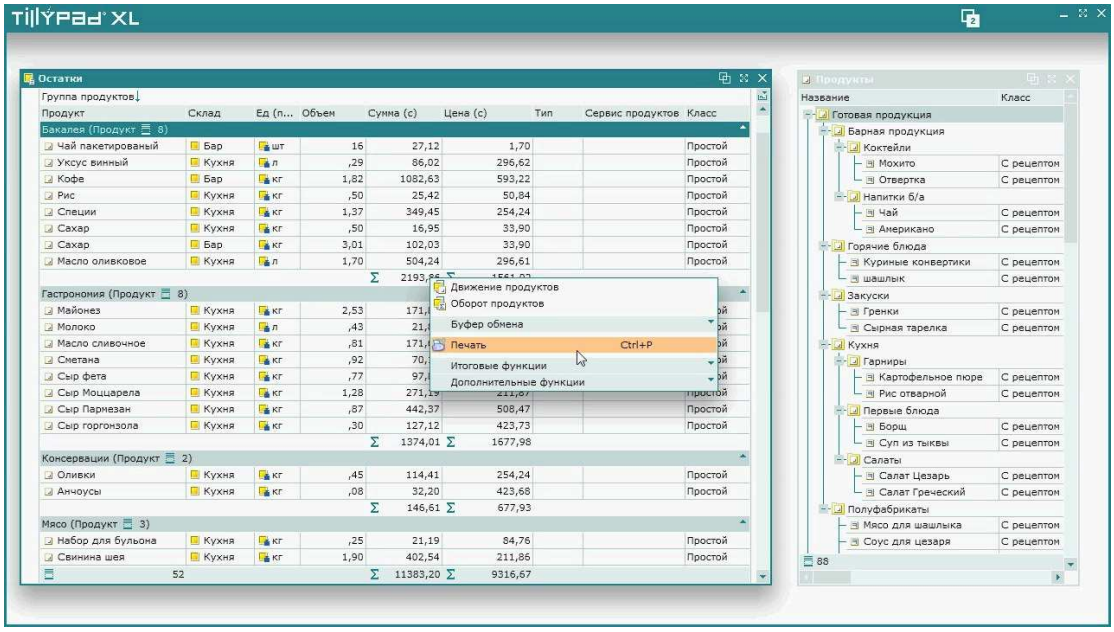

Рисунок 3–Система автоматизации для ресторанов и предприятий индустрии развлечений «Tillypad»

Один из представителей поставщиков ПО данного сегмента (Рисунок 3).Продукт предоставляется по платной подписке от 180 рублей в день.Так же, чтобы использовать одновременно и работу с поставщиками, и работу с цехами, необходимо купить несколько версий программы.

Данная система внедрила в свои устройства идентификацию по отпечатку пальца. Основные возможности модуля BIO Fingerprint Reader:

1. идентификация в TillypadPOS сотрудников и постоянных клиентов;

2. ведение фактического учета рабочего времени персонала;

3. разделение доступа к POS-терминалам/кассе;

4. просмотр информации по клиенту, а также предоставление льгот по программам лояльности;

5. возможность добавления новых сотрудников или постоянных клиентов непосредственно на POSe;

6. гибкая система настроек позволяет модуль-драйверу BIO Fingerprint Reader встраиваться в любые бизнес-процессы предприятия.

А также, эта система имеет большое количество сфер применения, таких как автоматизация ресторанов, кафе, баров, столовых и сетей фаст-фуда.

Для более детального анализа было проведено детальное сравнение конкурентно похожих систем. Результаты приведены в таблице 1.

Таблица 1– Сравнение конкурентов

	Покупка нескольких версий ПО для разных целей.	Платная ежемесячная подписка	Наличие платного оборудования	Наличие мобильного приложения
Iiko	+	+	+	+
Quick resto	+	+	+	+
tillypad	+	+	+	+

По результатам анализа видно, что все сторонние приложения не оснащают всеми функциями одно приложение, а создают отдельное для конкретных целей. У каждого из программных средств есть своя платная ежемесячная подписка. Каждая компания имеет возможность предоставить

своим клиентам платное оборудование. И каждая компания имеет мобильную версию программной системы, для специализированных целей.

Для восприятия подобных систем необходимо разобраться в таких понятиях, как анализ и прогнозирование бизнес-процессов работы предприятия, а также понять, какие показатели являются эффективными.

1.2Анализ, прогнозирование и показатели эффективности

Очень большое значение в системе управления занимает оценка финансовой деятельности, так как именно на ней концентрируется разработка финансовой политики и стратегии предприятия. Финансовый анализ в настоящее время занимает ведущее преобразование учета и переход к международным стандартам финансовой отчетности.

В последнее время коммерческие организацииработают непосредственно в условиях неопределенности и повышенного риска. Если посмотреть с одной стороны, то они имеют полное право свободно распоряжаться собственными средствами,самостоятельно заключать договора, сделки и контракты на внутреннем и внешнем рынке, что вынуждает предприятия самостоятельно заниматься проблемами поиска надежных партнеров и умения качественно оценивать их финансовую устойчивость. С другой стороны, предприятия намного серьезнее стали относиться к оценке собственных возможностей: могут ли они отвечать по своим обязательствам; эффективно ли используют имущество; рационально ли формируется капитал; приносят ли доход вложенные в активы средства; целесообразно ли расходуется чистая прибыль. Для ответа на эти вопросы, работники финансовых служб должны обладать знаниями по методике финансового анализа.

Оценка финансовой деятельности предприятия является довольно важной характеристикой его экономического достатка, характеризует результат текущего, инвестиционного и финансового развития, содержит необходимую

информацию для инвестора, а также отражает способность предприятия отвечать по своим долгам и обязательствам и наращивать свой экономический потенциал в интересах акционеров.

Если продукция соответствует всем предъявляемым требованиям, то любой бизнес-процесс показывает свою эффективность от полученных результатов.

Построение прогноза и верификация вероятностно-статистической модели берут за основу информацию двух типов:

– предопределенная информация о природе и содержательной сущности анализируемого явления. Представляется в виде теоретических закономерностей, ограничений, гипотез;

– исходные статистические данные, которые характеризуют процесс и результаты функционирования анализируемого явления или системы.

Важно выделить основные этапы прогнозирования.

Постановочный этап. Включает в себя:

– определение конечных прикладных целей прогнозирования; набора факторов и показателей (переменных);

– роли этих факторов и показателей – какие из них, в рамках поставленной конкретной задачи, можно считать входными, а какие – выходными.

Априорный или пред модельный этап. Состоит в предшествующем устройству модели анализа насыщенной сущности изучаемого процесса или явления, комплектовании и официализации имеющейся предопределенной информации об этом явлении в виде ряда допущений и исходных гипотез.

Информационно-статистический этап. Заключается в подготовке необходимой статистической информации, другими словами, регистрации значений участвующих в анализе факторов и показателей на различных временных и (или) пространственных выдержках функционирования моделируемой системы.

Этап спецификации модели. Включает в себя непосредственный вывод, вытекающий из априорного этапа, общего вида модельных соотношений, связывающих между собой интересующие нас входные и выходные переменные. Если говорить об общем виде модельных соотношений, то на данном этапе будет определена лишь структура модели, ее символическая аналитическая запись.

Этап исследований идентифицируемости и идентификация модели. Состоит в проведении статистического анализа модели с целью «настройки» значений ее неизвестных параметров на те исходные статистические данные, которыми мы располагаем. При реализации этого этапа «прогнозист» должен сначала ответить на вопрос, возможно ли в принципе однозначно восстановить значения неизвестных параметров модели по имеющимся исходным статистическим данным при, принятой на этапе спецификации модели, структуре модели. Это составляет так называемую проблему идентифицируемости модели. А затем, после положительного ответа на этот вопрос, необходимо решить уже проблему идентификации модели, т.е. предложить и реализовать математически корректную процедуру оценивания неизвестных значений параметров модели по имеющимся исходным статистическим данным. Если проблема идентифицируемости решается отрицательно, то возвращаются к этапу спецификации модели и вносят необходимые коррективы в решение задачи спецификации модели.

Этап верификации модели. Заключается в использовании различных процедур сопоставления модельных заключений, оценок, следствий и выводов с действительностью. Этот этап называют также этапом статистического анализа точности и адекватности модели. При пессимистическом характере результатов этого этапа необходимо возвратиться к этапу спецификации модели, а иногда и к постановочному этапу. Если же этап верификации модели дает положительные результаты, то модель может быть непосредственно

использована для построения прогноза в соответствии с описанной выше общей схемой.

Можно выделить макро-, мезо- (региональный и отраслевой уровни) и микропрогнозы по уровню прогнозирования. К микроуровню относится все, что связано с прогнозированием показателей, характеризующих деятельность фирм, компаний и предприятий. При описании внешней среды используются мезо- и макропрогнозы.

Следует подчеркнуть, что в реальности владелец предприятия может, конечно, успешно вести бизнес и не владеть методами построения математических моделей прогнозирования. Однако в условиях ожесточающейся конкуренции знание этих методов предоставляет бизнесмену и его бизнесу порой не менее значимые конкурентные преимущества, чем завоевание определенной доли рынка или получение выгодного кредита.

Понятие «Ключевые показатели эффективности для ресторанов, кафе» подразумевает под собой следующие пункты:

1. Уровень продаж. Если отдельно взятый товар имеет низкий уровень продаж, то есть затраты на его содержание в списке меню совпадают или превышают прибыль с этого товара, то увеличиваются операционные затраты и товарно-материальные ценности, что в конечном итоге влияет на прибыль всего предприятия.

2. Себестоимость товара. Ключевой деталью успешного бизнеса является поддержание себестоимости в одном состоянии. Увеличение себестоимости влечет за собой уменьшение прибыли.

3. Время выполнения заказа. Если время приготовления блюд не соответствует определенным предприятием нормам, то это говорит о неэффективной работе отдельно взятых пищевых цеховили кухни.

Показатели эффективности помогают найти необходимую стратегию для успешного ведения бизнеса, а именно:

1. поддерживать необходимый уровень сырья, для своевременного приготовления блюд;

2. анализировать историю продаж и вводить стратегии для их увеличения, для поддержания приемлемого уровня продаж;

3. корректировать цену товара при увеличении себестоимости.

Для понимания всей сути предприятий общественного питания необходимо понять, какие предприятия будут в дальнейшем рассматриваться в данной программной системе.

1.3 Рассматриваемые предприятия

Под понятием «предприятие» в рамках нашей системы понимается учреждение, которое ведет торговлю готовой продукцией, занимается закупкой сырья, имеет персонал, который отвечает за приготовление блюд и обслуживание гостей, а также ведет бухгалтерский учет.

Предприятие может различаться по типу обслуживания и предоставления услуг. Основные типы предприятий:

Ресторан –предприятиеобщественного питания с широким ассортиментом блюд сложного изготовления, включая заказные и фирменные блюда и изделия; спиртные, охладительные, горячие и другие виды напитков, мучные кондитерские и булочные изделия, табачные изделия, покупные товары, с высоким уровнем обслуживания и, как правило, в сочетании с организацией отдыха и развлечений.

Рестораны различают:

1. по выбору реализуемой продукции – неспециализированные и специализированные;

2. по расположению – в жилых и общественных зданиях, в том числе в отдельно стоящих зданиях, зданиях гостиниц, вокзалов, в культурно-

развлекательных и спортивных объектах, в зонах отдыха (ландшафтные), на транспорте (вагон-ресторан и пр.);

3. по увлеченности потребителей (клубный ресторан, спорт-ресторан, ресторан - ночной клуб, ресторан-салон);

4. по методам и формам обслуживания – ресторан с обслуживанием официантами, ресторан с обслуживанием по системе «шведский стол», ресторан выездного обслуживания;

5. по составу и назначению помещений – стационарные и передвижные.

Бар – предприятие общественного питания, продающие алкогольные или безалкогольные, горячие и прохладительные напитки, коктейли, в зависимости от специализации, а также оборудованное барной стойкой. Также продают холодные и горячие закуски и блюда в ограниченном ассортименте, покупные товары.

Бары различают:

1. по ассортименту реализуемой продукции и способу приготовления продукции общественного питания - бар винный, пивной (паб-бар), кофейный, десертный, молочный, коктейль-бар, гриль-бар, суши-бар и пр.;

2. по своеобразности обслуживания потребителей и (или) организации досуга (развлечений) – видео-бар, варьете-бар, диско-бар, кино-бар, танцевальный бар, лобби-бар, бар «Ночной клуб» и др.;

3. по расположению – в жилых и общественных зданиях, в том числе в отдельно стоящих зданиях, зданиях гостиниц, вокзалов; в культурно-развлекательных и спортивных объектах; в зонах отдыха;

4. по интересам потребителей (клубный бар, спорт-бар).

Рестораны и бары по уровню обслуживания и номенклатуре предоставляемых услуг подразделяют на три класса – «люкс», «высший» и «первый», которые должны соответствовать следующим требованиям:

– «люкс»– широкий выбор услуг, предоставляемых потребителям, высокий уровень комфортности и удобство размещения потребителей в зале, широкий ассортимент оригинальных, изысканных заказных и фирменных блюд, изделий, характерных для ресторанов, широкий выбор заказных и фирменных напитков, коктейлей для баров, изысканная сервировка столов, фирменный стиль, специфика подачи блюд, эксклюзивность и роскошь интерьера;

– «высший»– большой выбор услуг, предоставляемых потребителям, комфортность и удобство размещения потребителей в зале, разнообразный ассортимент оригинальных, изысканных заказных и фирменных блюд и изделий для ресторанов, широкий выбор фирменных и заказных напитков и коктейлей – для баров, фирменный стиль, изысканность и оригинальность интерьера;

– «первый»– определенный выбор услуг, предоставляемых потребителям, разнообразный ассортимент фирменных блюд и изделий и напитков сложного изготовления, характерный для ресторанов, широкий или специализированный ассортимент напитков и коктейлей, в том числе заказных и фирменных для баров, гармоничность и комфортность интерьера.

Кафе – предприятие общественного питания по организации питания и (или без) отдыха потребителей с предоставлением ограниченного по сравнению с рестораном ассортимента продукции общественного питания, реализующее фирменные, заказные блюда, изделия и алкогольные и безалкогольные напитки.

Кафе различают:

1. по ассортименту реализуемой продукции – неспециализированные и специализированные (кафе-мороженое, кафе-кондитерская, кафе-молочная, кафе-пиццерия и др.);

2. по обслуживаемому контингенту и интересам потребителей, включая оформление интерьера, – молодежное, детское, студенческое, офисное, кафе-клуб, интернет-кафе, арт-кафе, кафе-кабачок и др.;

3. по местонахождению – в жилых и общественных зданиях, в том числе, в отдельно стоящих зданиях, зданиях гостиниц, вокзалов; в культурно-развлекательных и спортивных объектах; в зонах отдыха;

4. по методам и формам обслуживания – с обслуживанием официантами и с самообслуживанием;

5. по времени функционирования – постоянно действующие и сезонные;

6. по составу и назначению помещений – стационарные и передвижные.

Чтобы детально вникнуть в работу какого-либо предприятия общественного питания, необходимо рассмотреть существующие роли компании.

1.4 Роли в работе предприятия

В работе любого предприятия общественного питания каждый работник имеет особенную роль, имеющую локальную ответственность за определенные цели.

Работники зала

Бармен, помощник бармена, бариста. Иногда, в небольших заведениях, используют одного бармена. Но если заведение имеет высокую посещаемостьи (или)широкий ассортимент коктейлей, иногда разделяют обязанности. Так, помощник может разливать сок, безалкогольные и чистые алкогольные напитки. При наличии нескольких залов в ресторане, может быть несколько барных стоек, за которыми бармены могут наливать авторские или популярные коктейли, а бариста наливать посетителям высококачественный кофе.

*Официант.*В день на работу выходит определенное количество официантов, исходя из загруженности заведения.В среднем, каждый сотрудник обслуживает не более 15 гостей. Но бывают случаи, когда официанты работают

в парах и вместе обслуживают до 8 столиков.Один занимается выносом блюд, а второй подносит меню и принимает заказ.

*Метрдотель.*Для крупных заведений привлекают метрдотеля (хостес). Он занимается приветствием клиентов, приемом первичного заказа и координациейработы официантов. Если в заведении несколько залов, то метрдотеля помещают в каждый из них.

*Кассир.*Иногда требуется кассир для выставления счета. Но в некоторых заведениях это делает бармен или сам официант.

Управляющий персонал

Количество управляющих в ресторане напрямик зависит отбюджета, уровня и популярности ресторана.В небольших заведениях с невысоким объемом работы роль управляющего может занимать владелец ресторана.

*Управляющий (менеджер) ресторана.*Несет ответственность за работу каждого работника заведения. бюджета, уровня и популярности ресторана. Может решать организационные вопросы, участвовать в составлении меню, проводить отбор персонала и определять график работы.

Можно вести рекрутинг менеджеров по:

- рекламе;
- персоналу;
- закупкам;
- логистике;
- развитию.

Если в ресторане часто проходят банкеты и прочие мероприятия, то часто назначают отдельного человека, который производит запись и договаривается об условиях торжества.

Работники кухни

Шеф-повар. Шеф-повар имеет непосредственное участие в составлении меню и разработке рецептов и технических карт для новых блюд. Вкусовой репертуар в полной мере зависит от него.Также подает заявки на закуп

необходимых продуктов и технических средств у поставщиков, координирует всю работу кухни и обучает новый персонал.Также лично занимается разработкой и приготовлением авторских блюд.

*Повар.*Человек, прошедший отбор у шеф-повара. Количество поваров зависит от посещаемости ресторана.Если заведение небольшое, то в нем обычно трудится смена из трех, четырех поваров. А если заведение рассчитано на 200 посетителей, то смена состоит из 20 поваров.

У каждого повара свои обязанности.Повара разделены по цехам. Каждый цех отвечает за свой фронт работ – супы, холодные закуски, горячие блюда и пр. В случае, если численность персонала недостаточна для такого распределения, повара могут совмещать обязанности.

В добавок, необходимо нанять низкопрофильных сотрудников, которые будутзаниматься фасовкой товаров и приготовлением заготовок.

У каждого работника предприятия общественного питания есть свои обязанности. Каждый делает свое дело. Для облегчения работы управляющему персоналу мы и хотим создать свою программную систему. Которое будет проводить анализ работу каждого из цехов и помогать принять нужное решение в работе заведения исходя из прогнозов программной системы.

Вывод по первой главе

В результате всего вышесказанного можно сделать вывод о том, что в настоящее время системы анализа и проектирования показателей эффективности часто применяются в предприятиях общественного питания. Ведь именно особенность этих систем помогает владельцам бизнеса в выяснении и устранении каких-либо сложностей и проблем путем привлечения самых различных областей практической деятельности.

Также, после сравнения с другими конкурирующими программными средствами мы делаем вывод, что все они популярны и у каждого есть свой

клиент. Но не во всех программных системах приведены решения для ресторанного бизнеса, присутствующие в одной программной системе.

Большое место в ресторанном бизнесе занимают работники заведения. У каждого из них есть свои обязанности и возможности. У каждого из служащих свой график и место в команде. Из расчёта программной системы анализа и прогнозирования мы сделали вывод о том, что нашей системой будут пользоваться лишь те, кому это необходимо. Это администратор предприятия общественного питания и рабочий персонал.

Особенное внимание стоит уделить предприятиям общественного питания. Были рассмотрены типовые рестораны, бары, кафе. Это именно те заведения, которые, в приоритете, будут задействовать нашу программную систему.

2 Проектирование программной системы анализа и прогнозирования показателей эффективности предприятия общественного питания

2.1 Варианты использования приложения предприятием общественного питания

Основная цель создания любой программной системы - создание такого программного продукта, который помогает пользователю выполнять свои повседневные задачи. Для создания таких программ первым делом определяются требования, которым должна удовлетворять система. Однако если дать пользователям написать эти требования на бумаге, то часто можно получить список функций, по которому трудно судить будет ли будущая система выполнять свое назначение и сможет ли она облегчить пользователю выполнение его работы вообще. Непонятно какие из выполняемых функций более важны и для кого. Были расписаны варианты использования для каждого компонента.

2.1.1 Время выполнения заказа

Предприятие получает обратную связь от клиентов, и со временем начинает замечать, что клиенты становятся недовольны временем доставки заказа. Чтобы разобраться в ситуации, предприятие использует компонент приложения, который анализирует время приготовления отдельно взятого заказа, определяет оптимальное время и сравнивает его с фактическим. Предприятие обнаруживает множество критически просроченных заказов. Приложение может определить какой цех вышел за рамки оптимального времени выполнения. С этой информацией предприятие может провести внутреннюю проверку и устранить недочет.

2.1.2 Анализ истории продаж и прогнозирование

Предприятие хочет увеличить прибыль своего бизнеса. Для этого оно использует компонент приложения, который выводит диаграмму товаров в порядке их количественной реализации. Таким образом можно определить какой товар продается хорошо, а какие товары продаются плохо. На основе полученной информации предприятие может предпринять меры по увеличению реализации плохо продающихся товаров.Также предприятие может получить прогноз на продажи. Таким образом можно определить, как товар будет продаваться в дальнейшем, и заранее провести компанию по увеличению популярности товара, если это необходимо.

2.1.3 Остаток на складе

Предприятие устанавливает нормы остатка сырья для бесперебойного производства товара. Когда остаток на складе становится ниже нормы, приложение сигнализирует о необходимости провести закупку сырья.

2.1.4 Учет срока годности сырья

Программная система ведет учет срока годности сырья. Если у сырья на складе заканчивается срок годности, программная система сигнализирует об этом пользователю.

2.1.5 Определение динамики потребления сырья

Используя алгоритм прогноза потребления сырья, программная система вычисляет, через какой срок полностью израсходуются запасы. Пользователю отправляется уведомление за 5 дней до полного израсходования сырья. Это позволяет пользователю вовремя скорректировать закупку необходимых продуктов.

2.1.6 Ценообразование

Предприятие вносит сумму операционных затрат в месяц. На основе этих данных и данных по себестоимости сырья, приложение производит анализ текущих цен на предмет недостаточной наценки товара и упущенной прибыли. Если себестоимость приближается к текущей цене товара, приложение сигнализирует о необходимости произвести пересмотр цен на товар.

2.1.7Программа лояльности

На основе данных, полученных через анализ и прогноз продаж, строится система акций для каждого товара, позволяющая вычислить необходимую скидку на товар. Скидка удовлетворяет два условия:

1. цена не должна быть ниже себестоимости;
2. скидка должна быть настолько высокой, чтобы поднять спрос на товар.

Для этой цели используется анализ продаж, чтобы определить насколько быстро идет спад реализации.

2.1.8Система учета смен

Предприятие хочет избавить от «бумажного» планирования смен персонала. Для этого предприятие использует компонент приложения, который строит график работ персонала, и позволяет вносить в него изменения.

Далее, для дальнейшего понимания структуры программной системы необходимо реализовать модели данных программного обеспечения.

2.2 Модели данных

Модель данных Workshop(Рисунок 4)

name - строка, название цеха

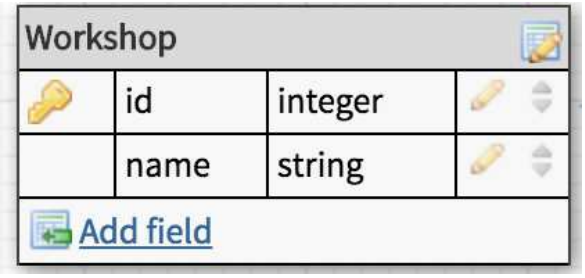

Рисунок4 – Модель данных Workshop

Модель данных Staff (Рисунок 5)

full_name - строка, полное имя работника

position - строка, должность

experience - целое число, опыт работы

rating - целое число, рейтинг

workshop внешний ключ, цех

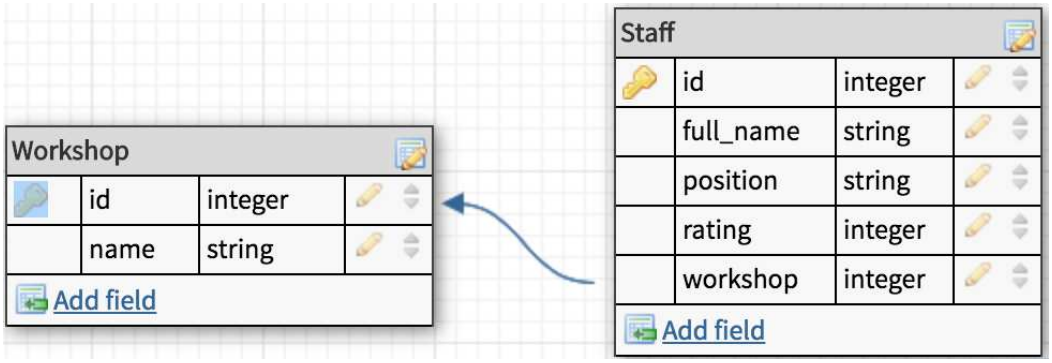

Рисунок 5 –Модель данных Staff

Модель данных Raw (Рисунок 6)

name - строка, название сырья

measure - строка, мера сырья

norm_amount - строка, норма

expiry_days целое число, срок годности в днях

cost_per_unit - число, цена за единицу

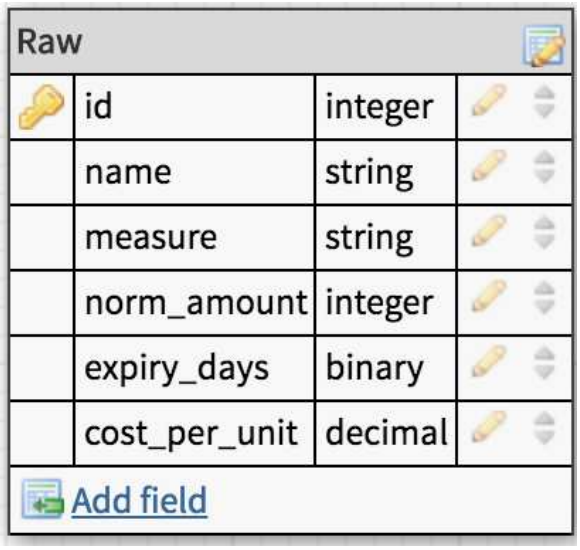

Рисунок 6 – Модель данных Raw

Модель данных RawStock (Рисунок 7)

raw - внешний ключ, сырье

receiving_date - дата, дата получения поставки

expiration_date дата, дата окончания срока годности

received - число, количество сырья в поставке

remainder - число, количество оставшегося сырья в поставке

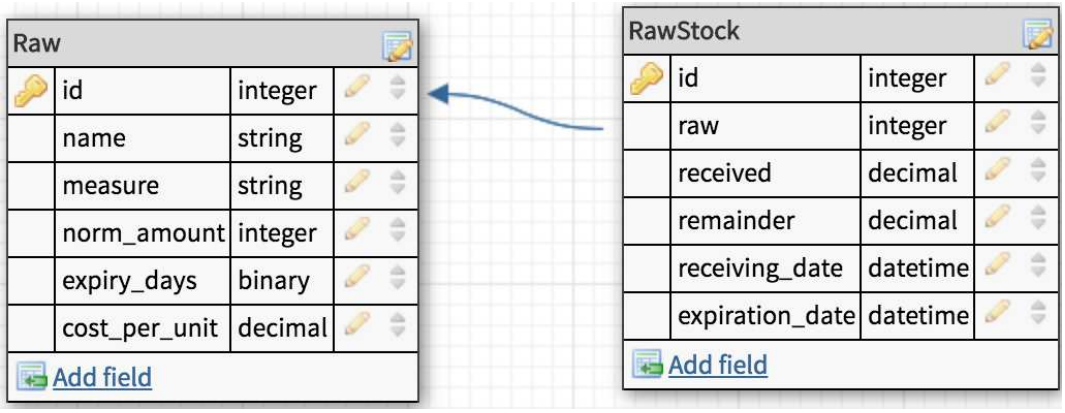

Рисунок 7 –Модель данных RawStock

Модель данных Ingredient (Рисунок 8)

amount - число, количество сырья

raw - внешний ключ, сырье

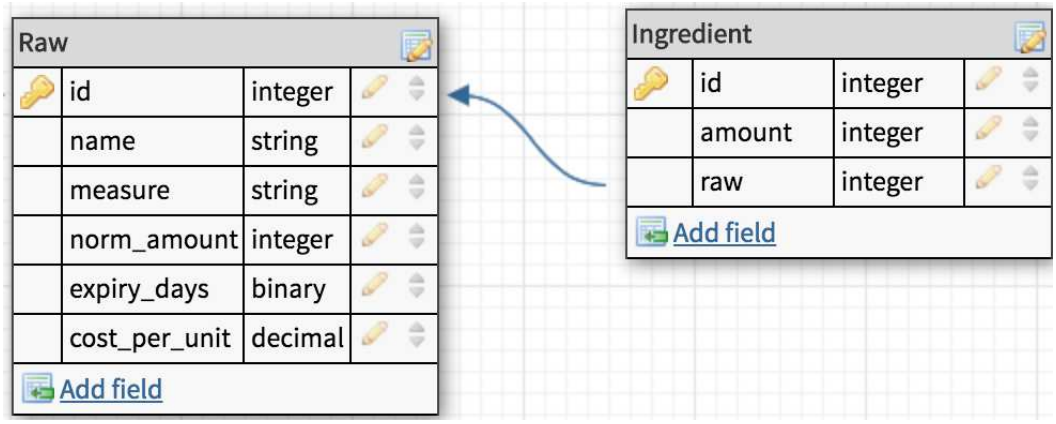

Рисунок 8 –Модель данных Ingredient

Модель данных Dish (Рисунок 9)

name - строка, название блюда

composition - связь многие ко многим, ингредиенты

workshop - внешний ключ, цех

price - число, цена

cooking_time_seconds - целое число, количество секунд во времени готовки

cooking_time_minutes - целое число, количество минут во времени готовки

cost_price - число, себестоимость блюда

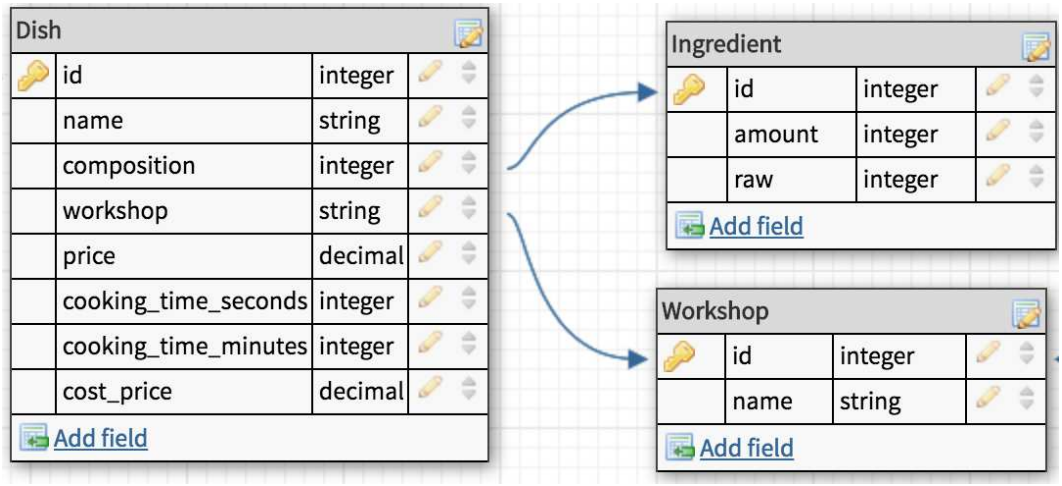

Рисунок 9 –Модель данных Dish

Модель данных PriceHistory (Рисунок 10)

dish - внешний ключ, блюдо

price - число, цена

date - дата, дата создания

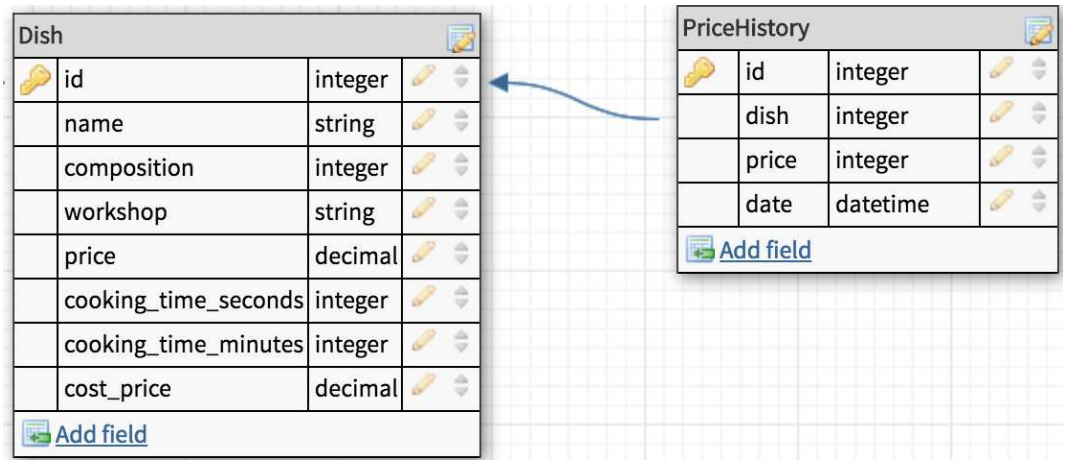

Рисунок 10 –Модель данных PriceHistory

Модель данных DishSaleStatistic (Рисунок 11)

dish - внешний ключ, блюдо

amount - целое число, количество проданных блюд

date - дата, дата создания

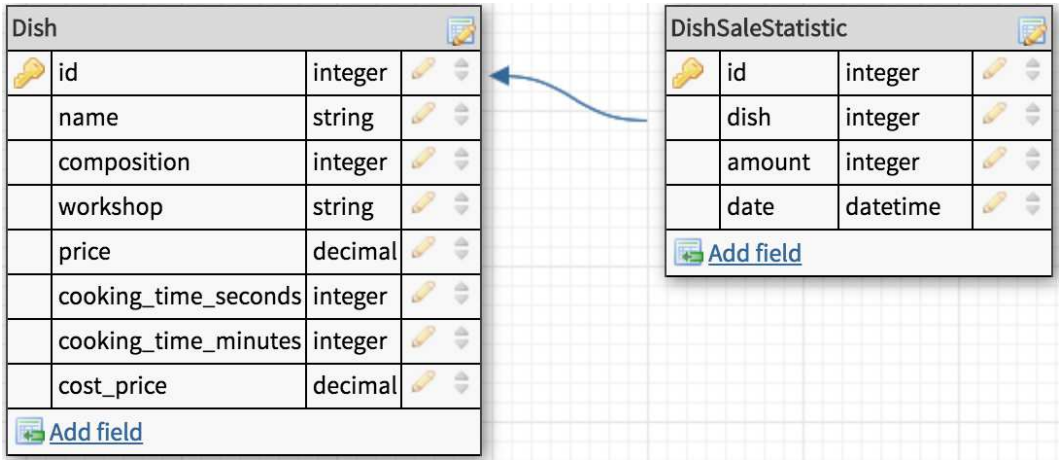

Рисунок 11 – Модель данных DishSaleStatistic

Модель данных OrderItem (Рисунок 12)

dish - внешний ключ, блюдо

amount - целое число, количество

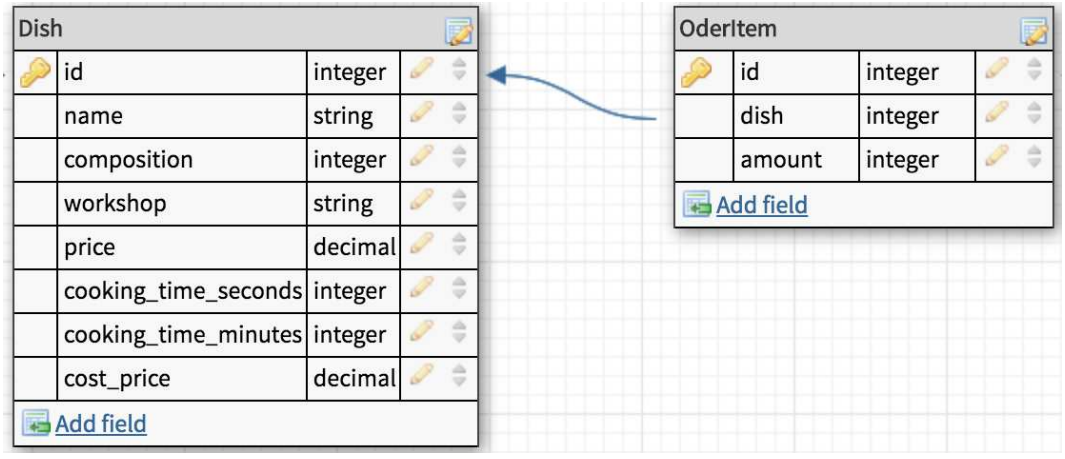

Рисунок 12 –Модель данных OrderItem

Модель данных Order (Рисунок 13)

date - дата, дата создания

started_at - дата и время, начало приготовления заказа

ended_at - дата и время, конец приготовления заказа

workers - связь многие ко многим, задействованные работники

items - связь многие ко многим, позиции заказа

is_processed - булево значение, обработан ли заказ

norm_excess - число, превышение нормы

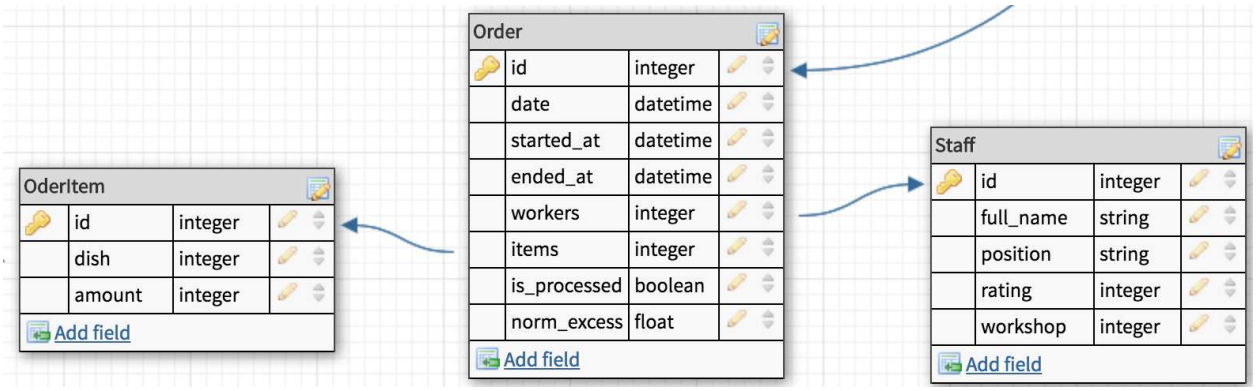

Рисунок 13 – Модель данных Order

Модель данных WastedDishes (Рисунок 14)

dish - внешний ключ, блюдо

amount целое число, количество

date - дата, дата создания

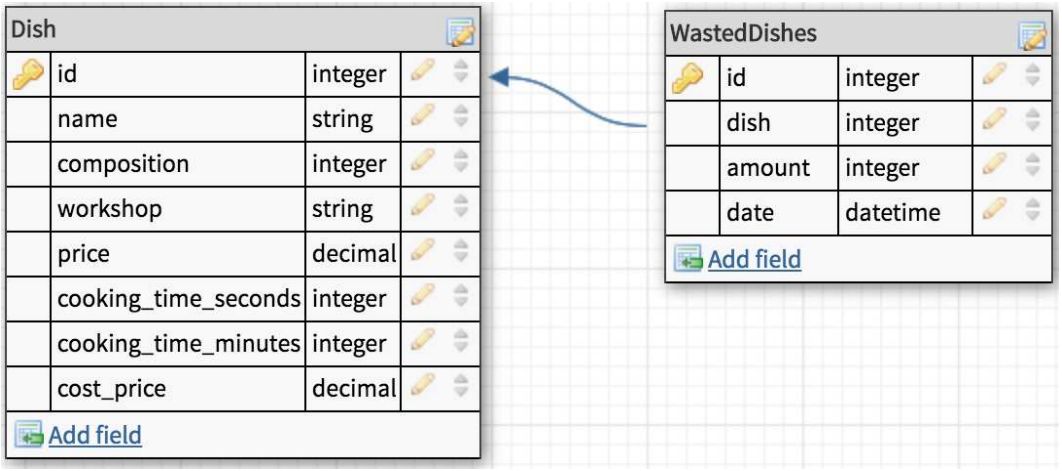

Рисунок 14 – Модель данных WastedDishes

Модель данных WastedRaws (Рисунок 15)

raw - внешний ключ, сырье

amount целое число, количество

date - дата, дата создания

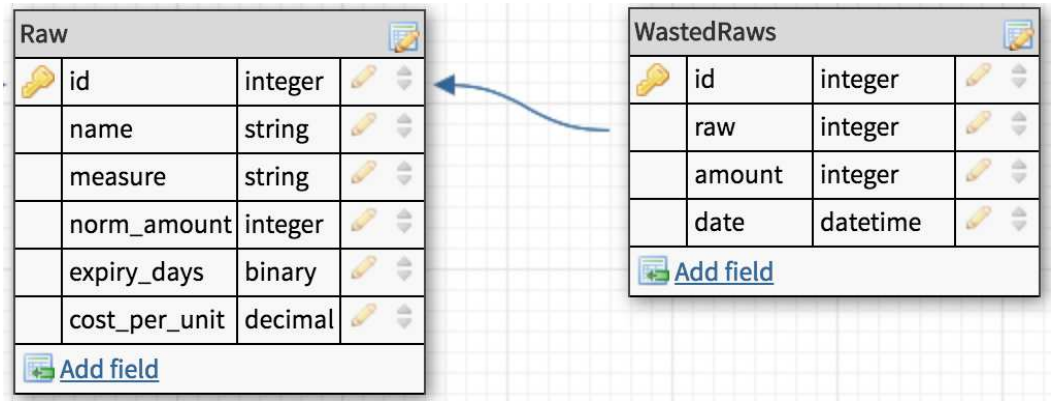

Рисунок 15 –Модель данных WastedRaws

Модель данных RemainderRaws (Рисунок 13)

raw - внешний ключ, сырье

amount целое число, количество

date - дата, дата создания

is_processed - булево значение, обработан ли

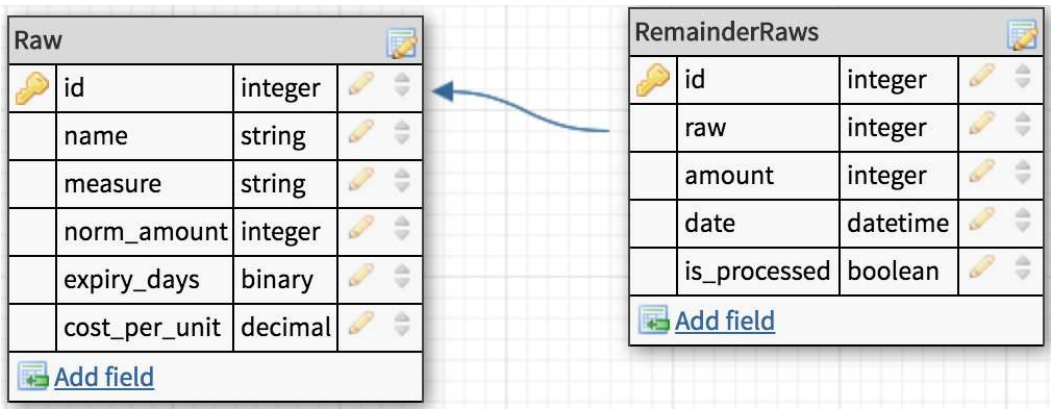

Рисунок 16 –Модель данных RemainderRaws

Модель данных Provider (Рисунок 17)

name - строка, название поставщика

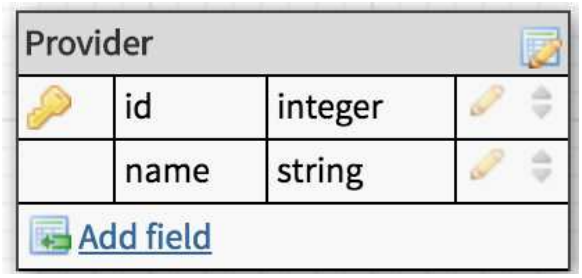

Рисунок 17 – Модель данных Provider

Модель данных Purchase (Рисунок 18)

provider - внешний ключ, поставщик

price число, цена

raw - внешний ключ, сырье

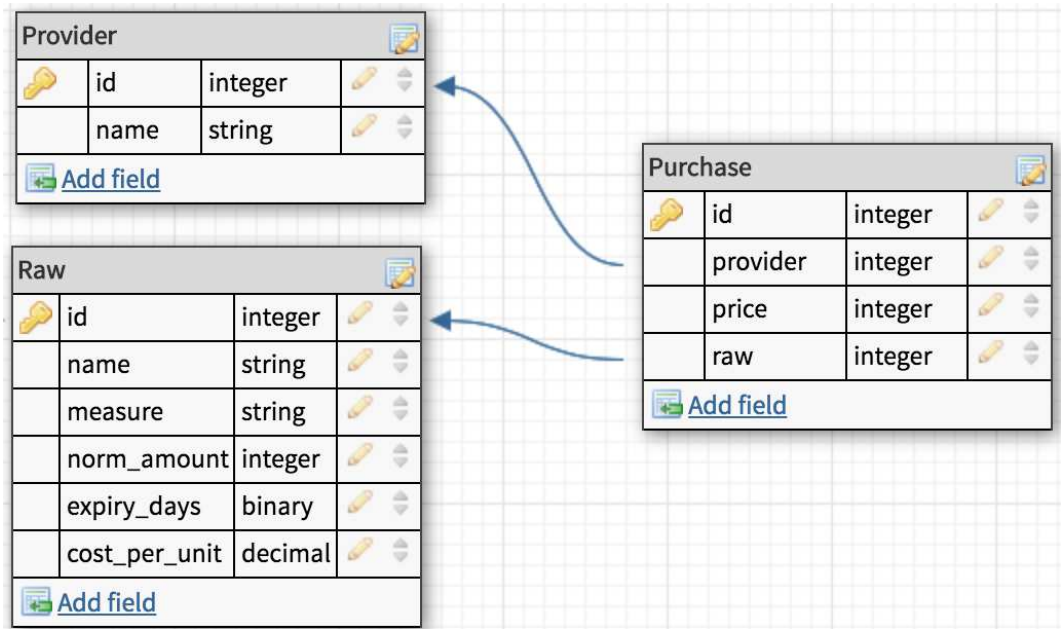

Рисунок18 – МодельданныхPurchase

Модель данных Problem(abstract)

importance - строка, важность

added_at - дата, датасоздания

is_solved - булево значение, обработана ли проблема

Модель данных OrderProblem (Рисунок 19)

order - внешний ключ, заказ

workshop - внешний ключ, цех

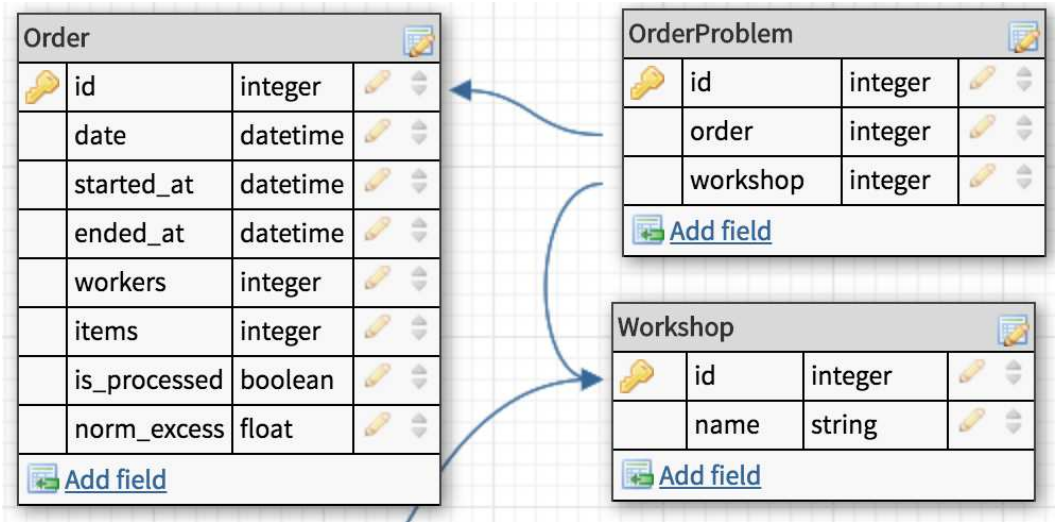

Рисунок 18 – Модель данных OrderProblem

Модель данных RemainderRawsProblem (Рисунок 20)

remainder_raws - внешнийключ, остатоксырья

advice - строка, совет по решению проблемы

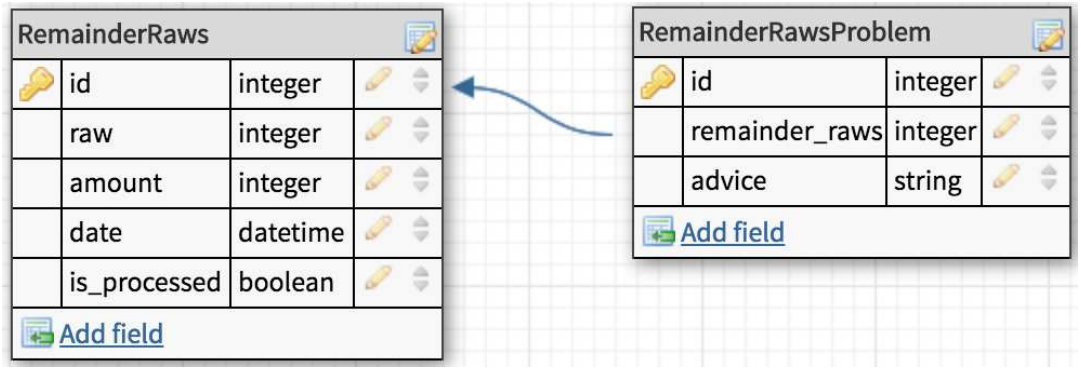

Рисунок 19 – Модель данных RemainderRawsProblem

Модель данных Shift (рисунок 21)

worker - внешний ключ, работник

date - дата, дата смены

type - строка, тип смены (дневная/ночная)

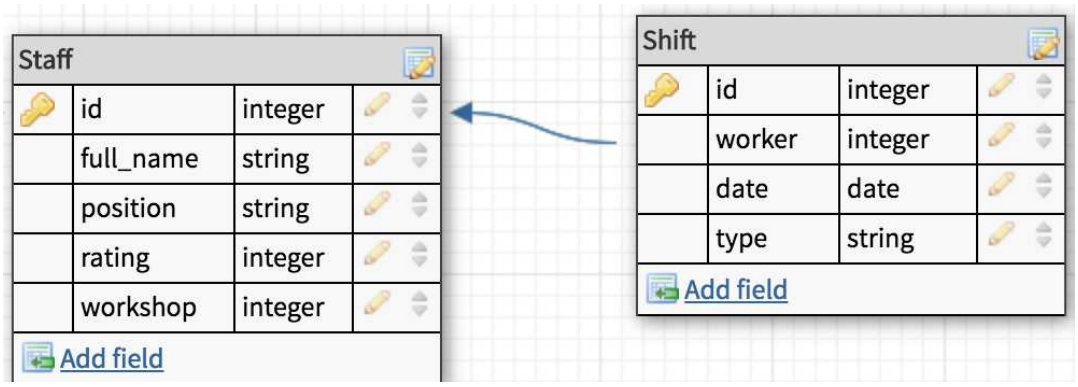

Рисунок 20 –Модель данных Shift

Модель данных Setting (Рисунок 22)

key - строка, ключ настройки

value - число, значение настройки

name - строка, название настройки

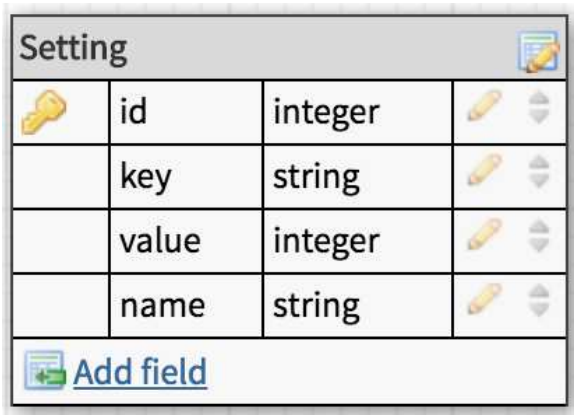

Рисунок 21 – Модель данных Setting

Модель данных Notification (Рисунок 23)

text - текст, текст уведомления

date - дата, дата создания

type - строка, тип уведомления

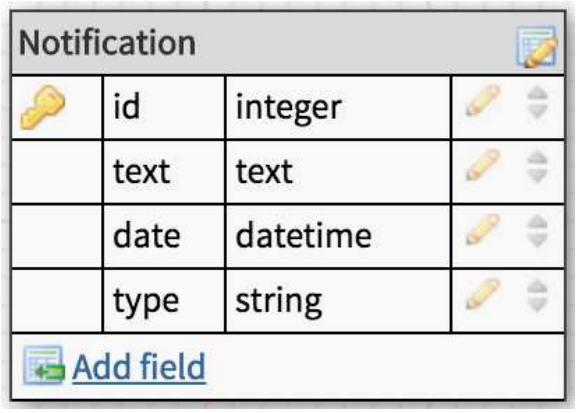

Рисунок 22– Модель данных Notification

2.3 Архитектура программного средства

Проект имеет вид клиент-серверного приложения, разделенного на отдельные компоненты (страницы). Задача архитектора состоит в детальной проработке отдельно взятого компонента. В качестве инструмента для создания схемы компонентов был выбран бесплатный сервис draw.io[5].Общая архитектура разрабатываемой программной системы показана на рисунке 24.

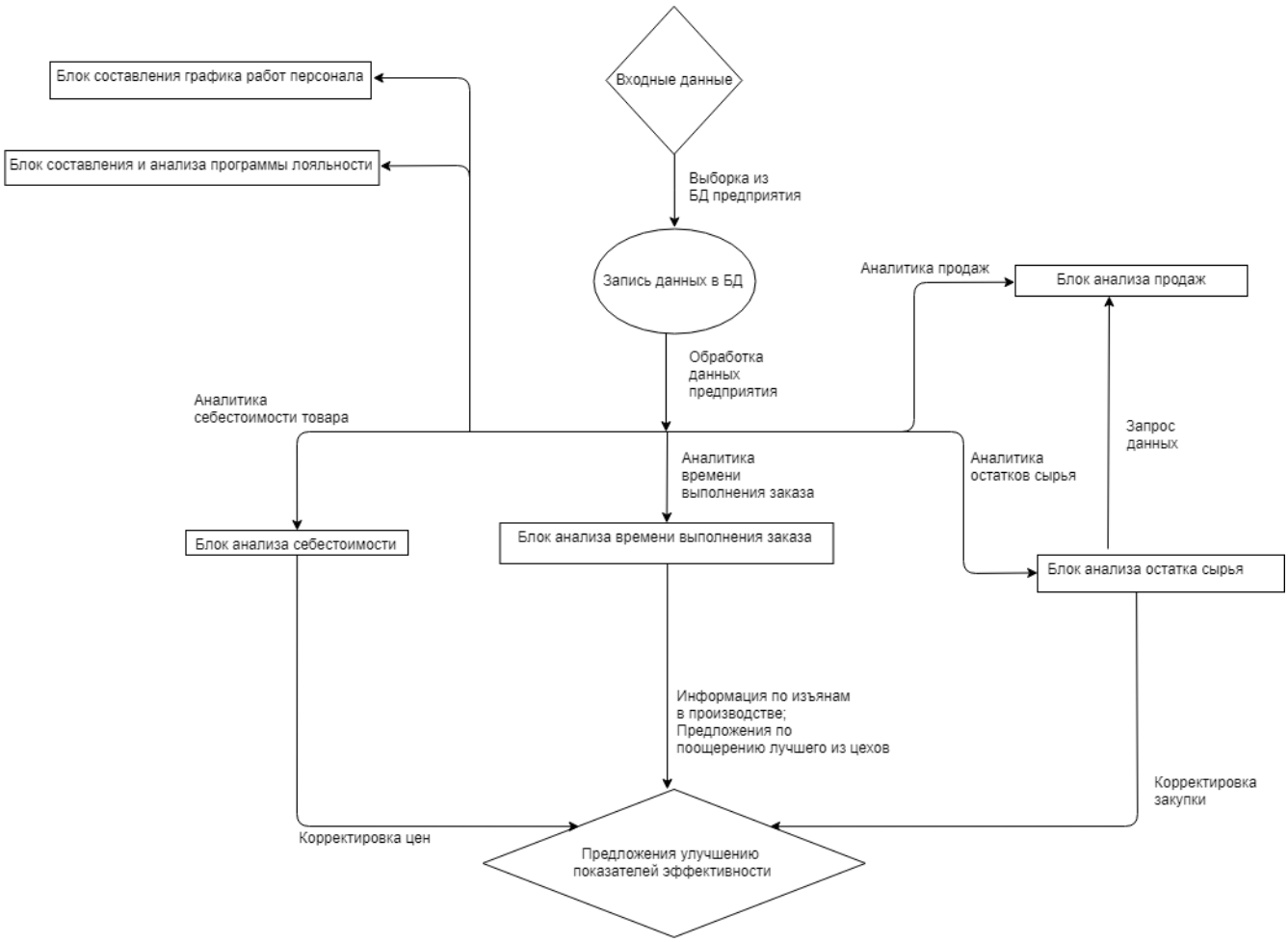

Рисунок 23 – Архитектура приложения по анализу показателей эффективности предприятия общественного питания

2.3.1 Компонент обработки входных данных

Компонент обработки входных данных показан на рисунке 25.

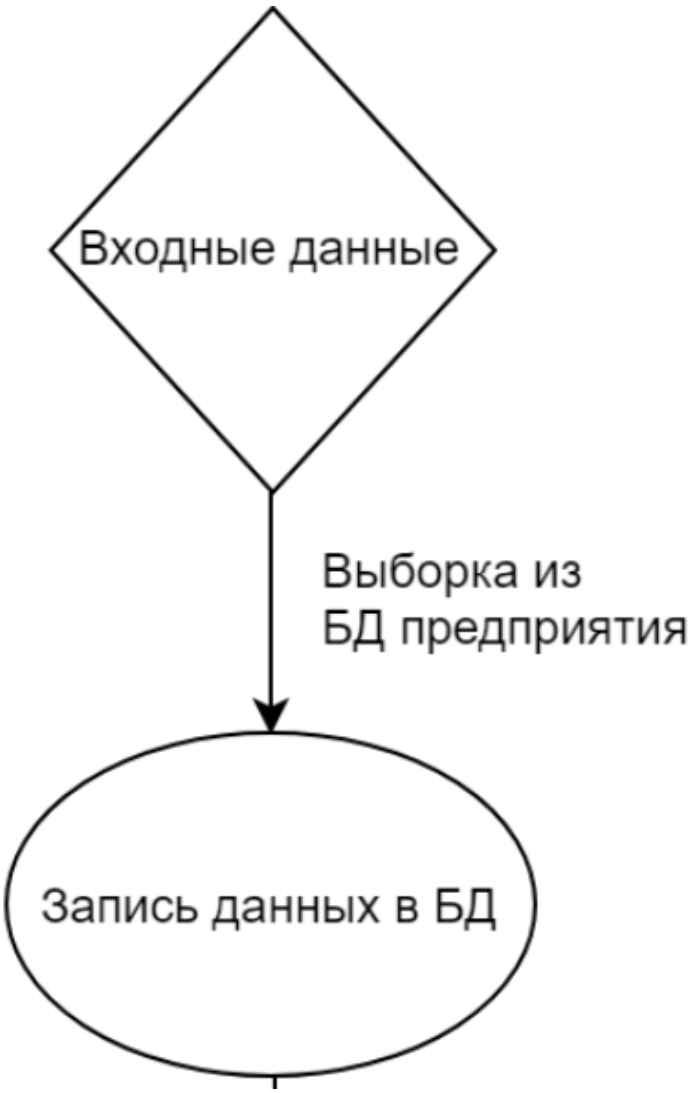

Рисунок 24– Компонент обработки входных данных приложения по анализу показателей эффективности предприятия общественного питания

Данный компонент состоит из двух частей:

1. приемник информации;
2. запись данных в базу данных.

Приемник информации обрабатывает данные, которые приходят в систему с различных учетных систем. Поддерживаются форматы XML[6]и JSON[7]. Данные из вышеперечисленных форматов преобразуются в допустимый для записи в базу данных вид. В модуле, который отвечает за запись данных в базу данных происходит запись полученной информации в соответствующие таблицы базы данных.

2.3.2 Компонент анализа времени выполнения заказа

Компонент анализа времени выполнения заказа показан на рисунке 26.

Рисунок 25– Компонент анализа времени выполнения заказа приложения по анализу показателей эффективности предприятия общественного питания

Компонент состоит из следующих частей:

- входная информация;
- определение оптимального времени выполнения;
- получение реального времени выполнения;
- выявление слабых мест в процессе приготовление;

- определение лучшего цеха на заданном временном интервале;
- выходная информация.

В качестве входной информации в данный компонент выступают данные истории заказов, которые были обработаны предприятием. Каждая запись содержит в себе список блюд в заказе, ответственных за приготовление заказа поваров и время выполнения заказа.

Определение оптимального времени выполнениявключает в себя распределение блюд между поварами и подсчет оптимального времени приготовления блюд, опираясь на нормативы.

Получение реального времени выполнения заказа происходит из входных данных.

Главная функция компонента – выявлять слабые места (цеха) в процессе приготовления. Это достигается через сравнения оптимального и реального времени приготовления заказа. Если реальное время превышает оптимальное на существенную величину, проблеме присваивается критический уровень. Если превышение не выходит за определенные системой рамки, то проблеме присваивается средний уровень.

После определения уровня проблемы, происходит выявление проблемного цеха, который вышел за рамки оптимального времени выполнения заказа. На основе полученных данных можно выделить ответственных поваров, работавших с заказом.

Определение лучшего цеха происходит через поиск в базе данных наименее причастного к проблемам цеха.

В качестве выходной информации служат данные о цехах, которые были причастны к задержке выполнения заказа.

2.3.3 Компонент анализа истории продаж и построения прогноза

Компонент анализа истории продаж и построения прогноза показан на рисунке 27.

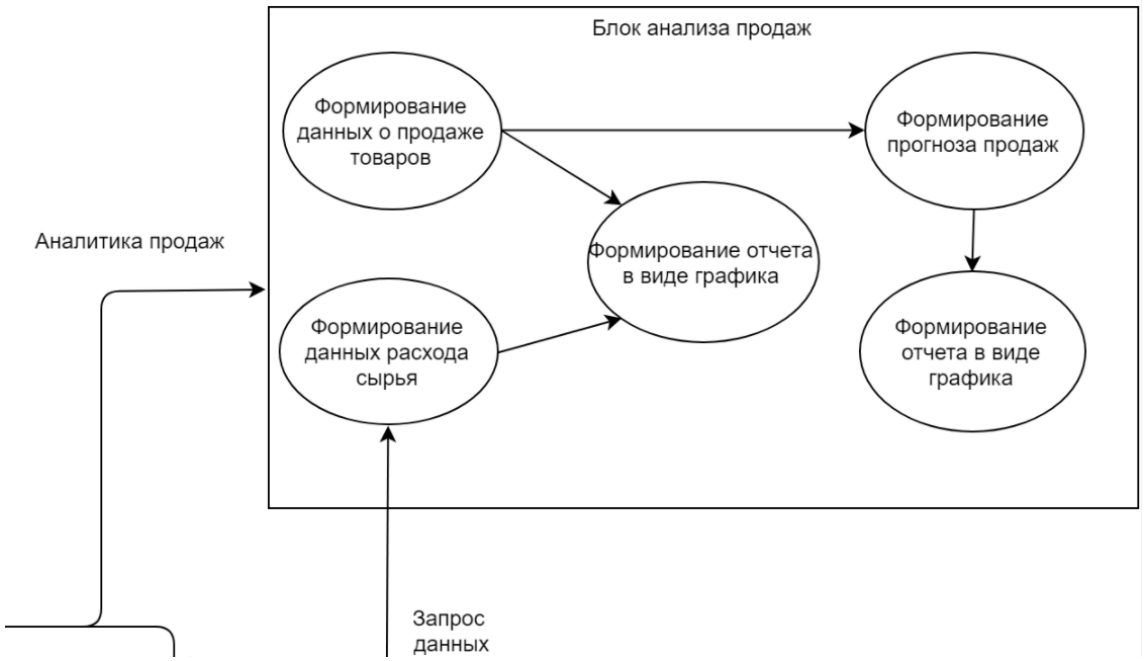

Рисунок 26– Компонент анализа истории продаж и построения прогноза приложения по анализу показателей эффективности предприятия общественного питания

Компонент состоит из следующих частей:

- входные данные;

- формирования данных по продажам каждого товара в заданном временном интервале;

- формирования данных по расходу сырья в заданном временном интервале;

- формирование отчета в виде диаграммы, где отражены продажи и расход сырья в заданном временном интервале;

- формирования прогноза продаж, на основе данных по продажам;

- формирование отчета в виде диаграммы, где отражен прогноз на выбранный временной отрезок;

- обработка входящего запроса на получение информации о расходе сырья.

В качестве входных данных в компонент поступает информация о продажах товара и расходе сырья, которая берется из соответствующих таблиц базы данных. Данная информация содержит в себе поля названия, даты, количества единиц.

На основе входных данных соответствующие методы получают на вход временной интервал, в рамках которого необходимо подсчитать продажи каждого товара и расход каждого вида сырья.

Полученная информация по продажам товаров и расходе сырья отправляется клиенту в виде столбчатой диаграммы. Элементы диаграммы можно сортировать по убыванию и по возрастанию с целью удобного восприятия информации. За формирование диаграммы отвечает метод, который получает данные о продажах товаров и расходе сырья и преобразует их в код, который может быть преобразован в столбчатую диаграмму посредством обработки браузером клиента.

Данные по продажам товаров и расходе сырья приходят в метод, который на основе входных данных строит модель функции и позволяет рассчитать какое количество товара будет продано в будущем. Для достижения этой цели используются численные методы определения функции по заданным точкам.

После обработки входных значений и построения функции, полученные данные по прогнозу продаж, передаются методу, который преобразует данные в столбчатую диаграмму, которая отражает данные по продажам и прогноз, основанный на данных по продажам.

Также в компонент может прийти запрос на анализ данных по расходу сырья. В этом случае строится модель функции по методу наименьших квадратов и определяется скорость расхода сырья. В качестве выходных данных выступает число дней, когда сырье будет полностью израсходовано.

Эти данные необходимы компоненту по анализу остатка сырья, который будет рассмотрен в следующем пункте.

2.3.5 Компонент анализа остатка сырья

Компонент анализа остатка сырья показан на рисунке 28.

Рисунок 27– Компонент анализа остатка сырья приложения по анализу показателей эффективности предприятия общественного питания

Компонент состоит из следующих частей:

- входные данные;

- определение динамики потребления сырья;

- поддержание уровня сырья не ниже нормы;

- учет срока годности сырья;

- выходные данные.

В качестве входных данных в компонент приходят данные по текущим остаткам сырья на складе предприятия, а также поступления новых партий сырья. Данные фиксируются в соответствующих таблицах базы данных для проведения дальнейшего анализа.

Метод определения динамики потребления сырья обращается к компоненту анализа истории продаж и построения прогноза (см. пункт 2.3.3). Полученные данные используются для определения динамики потребления сырья, а именно когда сырье будет полностью израсходовано. Метод берет каждое наименование сырья, и, если близится дата полного израсходования сырья, то выводит сообщение клиенту о немедленном вмешательстве в процесс закупки сырья для разрешения нависшей угрозы.

Метод поддержания уровня сырья не ниже нормы использует введенные клиентом нормы количества сырья, чтобы определять какое сырье вышло за пределы нормы, и требуется вмешательство клиента в процесс закупки. Поддержание определенного количества сырья на складе позволяет изготавливать продукцию при любом уровне спроса. В случае, когда определенное сырье опускается ниже нормы, выводится сообщение клиентку, что необходимо озаботиться дополнительной поставкой данного сырья на склад, чтобы сохранить приемлемый уровень амортизации производства.

Метод учета срока годности использует входные данные, а именно значение срока годности каждого сырья, чтобы заранее выявить сырье, которое в скором времени не сможет быть использовано в производстве из-за истекшего

срока годности. Такое сырье превращается в отходы. Задача метода – уведомить клиента о потенциальных отходах на уровне склада посредством вывода сообщения на экран с указанием названия сырья, у которого истекает срок годности. Очень важно учитывать специфику производства продукции предприятия общественного питания. Не каждая единица сырья будет использована в производстве, но необходимо минимизировать возможные отходы, дабы уменьшить операционные расходы.

2.3.6 Компонент анализа себестоимости товара

Компонент анализа себестоимости товара показан на рисунке 29.Компонент состоит из следующих частей:

- входные данные;
- получение значения операционных затрат;
- получение значений стоимости сырья;
- расчет себестоимости товара;
- предложения по корректировки цен на товары.

В качестве входных данных компонент получает значение операционных затрат предприятия общественного питания, которые задаются клиентом (предприятием) в интерфейсе приложения. В операционные затраты входят расходы, которые не участвуют в процессе производства товара: расходы на заработную плату персонала, расходы на арендную плату за предоставление помещения, расходы на коммунальные услуги и т.п.

Также на вход подаются данные стоимости сырья. Каждая поставка сырья имеет свой идентификатор и цену. По данному идентификатору происходит поиск в базе данных с целью вычленить информацию о цене сырья за единицу. Закупка сырья также является расходом, но за счет того, что сырье участвует в

процессе производства, оно принадлежит к категории расходов на товароматериальные ценности.

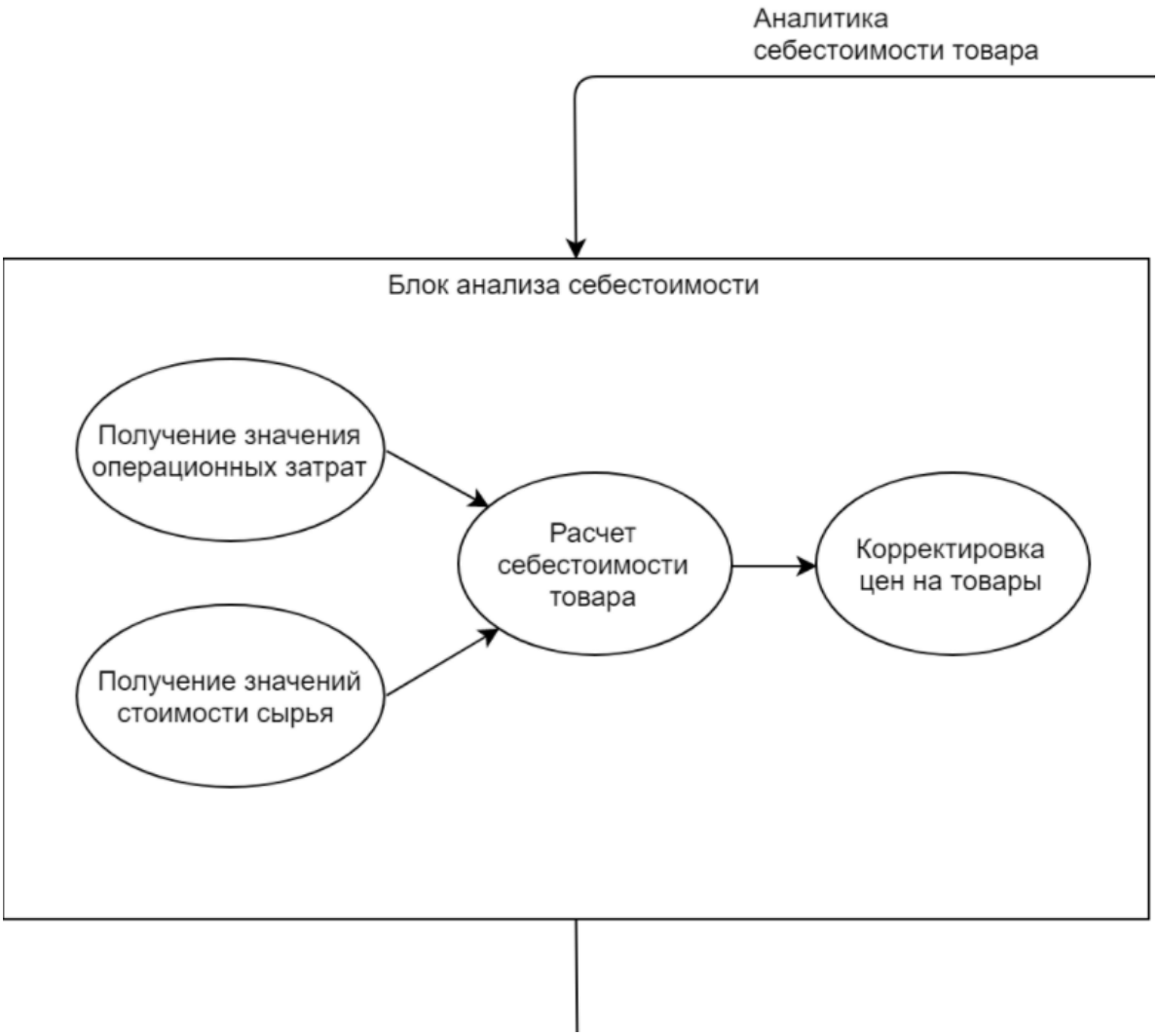

Рисунок 28– Компонент анализа себестоимости товара приложения по анализу показателей эффективности предприятия общественного питания

Модуль по расчету себестоимости товара получает на вход операционные расходы и расходы на товароматериальные ценности и производит необходимые вычисления для нахождения себестоимости товара.

Следующим этапом является анализ текущих цен для определения уровня наценки. Если какой-нибудь товар соответствует текущему уровню наценки, выводится оптимальная цена для данного товара и предлагается клиенту.

2.3.7 Компонент составления и анализа программы лояльности

Компонент составления и анализа программы лояльности показан на рисунке 30.

Рисунок 29– Компонент составления и анализа программы лояльности приложения по анализу показателей эффективности предприятия общественного питания

Компонент состоит из следующий частей:

- входные данные;
- данные о товарах с фактическим низким уровнем продаж;
- данные о товарах с потенциальным понижением уровня продаж;
- расчет размера скидки;
- корректировка цена на товары.

В качестве входных данных компонент получает данные с компонента по анализу истории продаж и построения прогноза.

Данные о товарах с фактическим низким уровнем продаж формируются из истории продаж отдельно взятых товаров. Берутся самые худшие позиции с точки зрения уровня продаж.

Данные о товарах с потенциальным понижением уровня продаж формируется из построения прогноза по продажам отдельно взятых товаров. Берется половина лучших товаров с точки зрения уровня продаж. Также добавляются данные об уровне продаж в крайних точках временного среза.

Расчет размера скидки происходит следующим образом. Для случая с худшими товарами в плане продаж:

1. Определяется из какого сырья состоит товар. Если сырье также, как и товар, плохо расходуется, то цена снижается до себестоимости с целью избавиться от неликвида.

2. Если сырье продается нормально (не 3 самых худших позиции), то снижается наценка в два раза.

3. Выводится сообщение пользователю о рекомендуемом снижении наценки.

Для случая с товарами с потенциальным снижением уровня продаж:

1. Берутся данные в первый день продаж в графике истории (- 30 день) и последний день в графике прогноза (30 день).

2. Делим первый день продаж на последний день прогноза

3. Отнимаем от единицы получившееся значение

4. Умножаем на наценку значение с предыдущего пункта.

5. Выводим сообщение пользователю о рекомендуемом снижении наценки.

Выше описанные сценарии работ для вычисления значения скидки необходимо запускать каждые две недели работы системы. Необходимо отслеживать следующие показатели:

1. Уровень продаж товаров, которые продаются по себестоимости. Если не прогресса в течение двух недель, то создается сообщение для клиента о

необходимости отказаться от товара в виду его неспособности приносить прибыль.

2. Уровень продаж товаров. Если после уменьшения цены продажи начали расти, то проводить корректировку цены на повышение.

2.3.8 Компонент составления графика работ персонала

Компонент составления графика работ персонала представлен на рисунке 31.

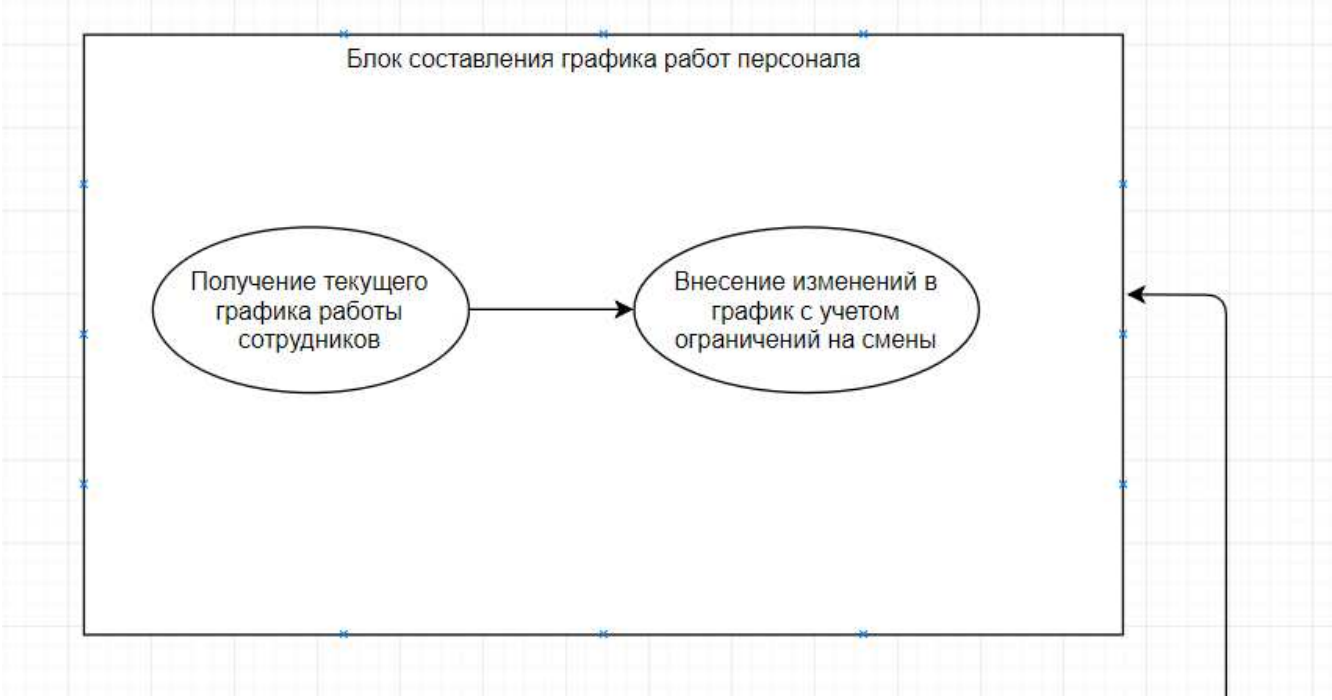

Рисунок 30– Компонент составления графика работ персонала приложения по анализу показателей эффективности предприятия общественного питания

Компонент состоит из следующий частей:

1. входные данные;
2. внесение изменений в график работ.

В качестве входных данных компонент получает набор текущего графика смен для каждого сотрудника.

Внесение изменений в график работ подразумевает следующий порядок действий:

1. выбор сотрудника, для которого требуется изменить график;
2. изменить график;
3. сохранить график.

Для корректного составления графика, требуется придерживаться следующих правил:

1. составлять график необходимо минимум на месяц вперед;
2. у работника минимум два выходных в неделю;
3. после работы в ночную смену необходимо ставить два выходных дня для отдыха;
4. при сменном графике промежуток между сменами должен быть не менее 16 часов.

2.3.9 Вывод информации

В качестве вывода информации клиенту будут использоваться веб-страницы, на которых размещаются необходимые диаграммы, графики, предложения по улучшению показателей предприятия общественного питания

Текстовые сообщения выводятся для таких компонентов как корректировка цен на товары, корректировка закупок, анализ времени выполнения заказов.

Сообщения в виде диаграммы выводятся для компонента по анализу продаж и прогнозированию.

Вывод сообщений изображен на рисунке 32.

Рисунок 31– Вывод информации приложения по анализу показателей эффективности предприятия общественного питания

Вывод по второй главе

Определение вариантов использования потенциального продукта помогло сформировать требования для последующего создания модели базы данных и архитектуры приложения.

Составив модель данных и прописав все необходимые связи между объектами, мы смогли определить масштаб приложения и разграничить функционал в отдельные компоненты.

С помощью высокоуровневого проектирования была создана архитектура приложения, которая позволяет наглядно представить, как будет выглядеть конечный для потребителя функционал. Также архитектура служит отправной точкой для низкоуровневого проектирования, которое в свою очередь используется для программирования функций.

3 Программная реализация системы анализа прогнозирования показателей эффективности предприятия общественного питания

3.1 Реализация клиентской части приложения

Ранее были объявлены необходимые требования и критерии для разработки программной системы анализа и прогнозирования показателей эффективности предприятий общественного питания.

3.1.1Сравнение и выбор средств разработки

Для определения нужных для разработки технологий, мы провели сравнительный анализ существующих на рынке технологий. Выбор правильных технологий позволит в кратчайшие сроки создать качественный и конкурентоспособный продукт.

Для реализации пользовательской части программной системы было принято решение использовать следующие технологии:

- HTML5[8];
- CSS3[9];
- SCSS[10];
- JavaScript[11];
- Flexbox[12];
- адаптивная верстка[13];
- Gulp[14].

Данные технологии были выбраны с учетом раннего опыта разработки программных средств с использованием этих средств.

Далее рассмотрим обоснование выбора каждой из вышесказанных технологий.

Для начала рассмотрим технологию HTML5. Технология поддерживается абсолютно всеми известными на данный момент браузерами и содержит в себе ряд полезных функций:

1. семантическая верстка;
2. встроенная валидация данных;
3. встроенные элементы для ввода информации.

Семантическая верстка позволяет создать понятный для чтения код страницы. Это позволяет разрабатывать структуру страницы понятным для всех членов команды языком. Также семантическая верстка играет большую роль в индексации страниц известными поисковыми системами, такими как Google[15], Яндекс[16], Mail.ru[17]и так далее. Если поисковая система находит на странице семантическую верстку, то автоматически поднимает веб-сайту рейтинг, и, следовательно, способствует распространению приложения.

Встроенная валидация данных позволяет проверять введенные пользователем данные на уровне браузера, без подключения серверных мощностей. Мы используем эту возможность для ускорения процесса разработки, так как нет необходимости писать функции на сервере для приема и проверки пользовательских данных.

Встроенные элементы для ввода информации, такие как календарь, часы, служат для повышения удобства пользования страницей. Нет необходимости вручную вводить дату или время, тем самым увеличивая вероятность ошибки из-за человеческого фактора. Достаточно нажать на поле и выбрать нужную дату в один клик, при этом валидный (правильный) формат данных подставляется автоматически, тем самым снижая нагрузку на валидацию данных. Пример элемента «Календарь» показан на рисунке 33.

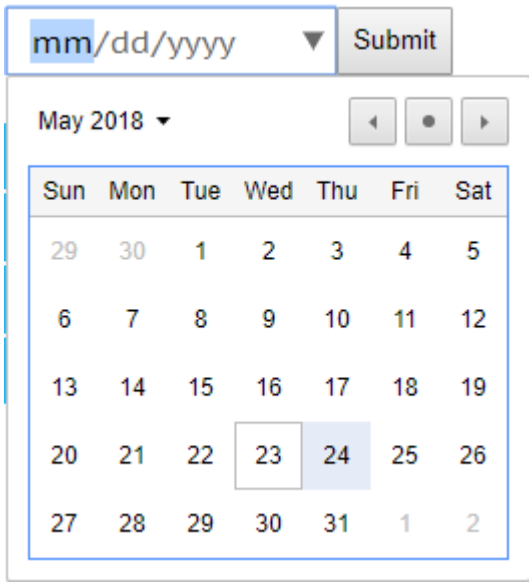

Рисунок 32– Элемент «Календарь» HTML5

HTML5 позволяет нам разрабатывать быстрее, качественнее с точки зрения поисковой оптимизации и позволяет снять нагрузку с сервера, лишь переместив валидацию на браузерный уровень.

CSS3 стал стандартом в веб-разработке несколько лет назад. В нашем проекте мы используем возможности этой технологии для создания анимации. Реализация обработчиков событий через CSS-процессор позволяет снять нагрузку с процессора компьютера, так как больше нет надобности использовать центральный процессор для обработки целого ряда информации с целью правильно перерисовки страницы. Все это делает встроенный в CSSпроцессор, который следит за состоянием объектов на странице и использует соответствующие методы для обработки различных состояний, будь то наведение мыши, клик и так далее.

Также CSS3 позволяет использовать медиа-запросы для определения различных параметров экрана пользователя:

- ширина;
- высота;
- тип устройства.

В нашем случае необходимо использовать медиа-запросы для создания адаптивности страниц, так как мы можем реагировать на изменение ширины экрана и подстраивать стили элементов под соответствующую ширину.

Еще одной важной функцией, который мы активно пользовались по ходу разработки, является наличие селекторов для логической выборки элементов, используя функциональные выражения. Таким образом у нас есть возможность выбирать, например, каждый второй или третий элемент, начать выборку с четвертого элемента и так далее.

Чтобы ускорить процесс разработки и упростить выборку элементов, используя CSS3, было принято решение подключить SCSS-фреймворк. Данный фреймворк позволяет использовать следующие функции:

- вложенность конструкций;
- переменные;
- функции.

Вложенность конструкций позволяет нам упростить способ обращения к элементам страницы. Чтобы обратиться к атомарному элементу, у которого большое количество предков, нужно поэтапно обратиться к каждому предку, что затрудняет процесс и увеличивает время, отведенное на стилизацию элемента. Используя вложенные конструкции, можно задавать стили для атомарных элементов прямо в блоке с его предками, тем самым достигаются эффекты:

- увеличивается читаемость кода;
- увеличивается скорость разработки.

Использование переменных позволяет нам хранить определенные CSS-правила, которые часто используются от элемента к элементу. Можно, например, задать переменной цвет, который используется почти на каждой странице приложения, и использовать эту переменную вместо цветовой кодировки, что, несомненно, увеличит и читаемость кода, и скорость разработки, так как больше нет необходимости подбирать цвет с нуля.

Все браузеры отличаются друг от друга рядом функций. Эти функции влияют на отображение контента от браузера к браузеру. Для того, чтобы страница отображалась везде одинаково, нужно использовать соответствующие префиксы для определенных стилей. Например, чтобы задать длительность анимации, требуется задать стандартное для CSSправило, а также ряд префиксов для каждого типа браузера. Делать это постоянно вручную неудобно, тем более тормозится процесс разработки. Для этих целей можно использовать SCSS-функции, которые можно вызывать один раз в коде, и они, в рамках нашего примера, пропишут все необходимые для работы на разных браузерах префиксы.

В целом, технология SCSSпо аналогии cHTML5 позволяет упростить процесс разработки и ускорить процесс стилизации элементов. В дальнейшем эту технологию можно интегрировать с системой сборки элементов проекта, для безостановочного процесса разработки.

Javascript-код играет большую роль в нашем приложении. Множество событий, будь то открытие модального окна или скрытие определенных элементов, не может обойтись без применения кода.

Наша цель – обеспечить пользователя необходимым функционалом, при этом сохранив приемлемый уровень производительности. Javascriptподходит под эту цель только потому, что язык разработан специально для использования браузерами.

Существует Jquery-фреймворк[18] – надстройка над чистым Javascript-кодом, который упрощаем взаимодействие с элементами страницы через сокращение количества кода. Но при этом наносится ущерб быстродействию страницы, что противоречит нашей цели. Поэтому мы отказались от использования Jqueryв угоду производительности. В данном случае быстродействию был отдан высший приоритет нежели скорости разработки, и эти жертвы оправданы, так как пользователь получает страницу, с которой он сможет работать на компьютере с любой конфигурацией.

Используя Javascriptособое внимание уделялось сохранению ресурсов процессора. Для этого велся учет количества обращений к элементам, и, если один и тот же элемент страницы использовался несколько раз в коде, то он выносился в переменную, для повторного использования без привлечения дополнительных ресурсах. В противном случае, если блок страницы используется единожды, то мы ограничивали создание переменной в угоду сокращения кодовой базы. Чем меньше кода требуется загрузить пользователю, тем быстрее страница будет доступна для использования.

Также использовалась встроенная javascript-библиотека, которая позволяет в фоновом режиме отправлять данные для сервера. Jqueryпредоставляет более удобный функционал для решения подобной задачи, но мы не могли подключить эту библиотеку из соображений быстродействия. Тем не менее, встроенная библиотека позволяет проводить более гибкую настройку запроса на сервер, чего не может сделать jquery-фреймворк. Поэтому мы выбрали чистый javascript-код.

По ходу создания страниц требуется позиционировать многие элементы страницы. Это делается через ряд доступных для использования CSS-правил:

- Float[19];
- Flexbox;
- Grid[20].

Рассмотрим каждый их этих вариантов и определимся с оптимальным.

Float – старейшее правило, которое использовалось с internetexplorerчетвертой версии. Данное правило довольно сложное в использовании, так требуется буквально вручную выставлять каждый элемент в блоке, используя внутренние правила float. Разработка бы заняла гораздо больше времени, если бы мы использовали это правило. С другой стороны, посредством Floatдостигается поддержка большого количества браузеров, и мы можем отображать страницы нашего приложения на старых компьютерах, что расширит потенциальную клиентскую базу.

Grid – самое современное правило, которое очень просто в использовании. Основная идея Gridсостоит в том, что мы в начале процесса разработки прописываем правила для страницы, и все внутренние элементы следуют им в ходе всей разработки. Из преимуществ подобной система – быстрота разработки, но ввиду относительно молодого возраста технологии, охват всевозможных браузеров не достаточен для полноценного использования.

Flexbox – золотая середина между вышеперечисленными технологиями. Эта технология позволяет в автоматическом режиме позиционировать элементы, однако сохраняется поддержка почти всего массива браузеров, что дается достаточно большой охват аудитории. Из минусов можно отнести полуавтоматическую настройку поведения элементов в строке – довольно часто можно столкнуться с проблемой переноса элементов на новую строку. Эта проблема решается ручной настройкой поведения блоков в одной строке.

Наш выбор остановился на технологии Flexbox, как на самой адаптированной к современным реалиям технологии. Этим выбором мы добились баланса между охватом аудитории через адаптацию страницы под все браузеры и скоростью разработки, так как огромная часть функционала технологии работает в автоматическом режиме.

Для адаптации страницы под различную ширину экрана мы использовали встроенные в CSS3 средства медиа-запросов. На рынке представлено множество технологий, которые позволяют быстро адаптировать страницу под множество экранов:

- Bootstrap[21];
- Foundation[22];
- Skeleton[23].

Остановимся подробнее на технологии Bootstrap. Данная технология позволяет в структуре странице прописать правила для каждого блока, которые он сможет использовать при определенной ширине экрана.

Этим достигается ускорение процесса разработки, но есть два существенных минуса:

1. падает читаемость кода;
2. увеличивается объем загружаемых пользователем данных.

Читаемость кода является важным фактором для нашего продукта, потому что проект в последствии потребует доработок. Если код страниц будет нечитаемым, то доработка продукта может стоить очень дорого или свестись на нет.

Чтобы заработала система адаптации Bootstrap, требуется загрузить ряд файлов на компьютер пользователя, что повлечет за собой потерю производительности. У нас производительность системы стоит на первом месте, поэтому от использования Bootstrapбыло решено отказаться.

Вместо подобных фреймворков решили использовать возможности CSS3-процессора и ряд соглашений для именования блоков страницы. Блоки страницы именуются таким образом, что можно взять отдельный элемент страницы и вставить его в другую страницу без потери функционала. Достигается этот эффект через вложенность дочерних компонентов страницы.

Для каждой ширины экрана меняются стили для родительского компонента, дочерние элементы лишь наследуют новое поведение.

Использование встроенных возможностей CSSпозволило сохранить быстродействие страницы, не урезая функционал адаптации под различные экраны.

Использование технологий для сборки элементов проектов воедино может значительно ускорить процесс разработки. Существует несколько решения для решения данной задачи:

1. Grunt[24];
2. Webpack[25];
3. Gulp[26].

Суть подобных программ одна – в режиме реального времени реагировать на изменение файлов и перезагружать проект, обеспечиваю бесперебойное обновление страницы. Достаточно поменять, например, цвет фона, как он автоматически изменится на страницы, без необходимости перезагружать страницу.

Grunt– самый старый из вышеперечисленных программ, используется неудобный для использования синтаксис. Также имеет ограниченный функционал. В целом, его можно использовать для сборки проектов с небольшим количеством фреймворков.

Webpack – самый большой представитель программ для сборки. В основном используется для поддержки frontend-фреймоврков, таких как React.js, Angular, Vue.js. Поддерживает огромное количество дополнений и расширений. Но он ориентирован на поддержку frontend-приложений. Для использования webpackв качестве сборщика статических страниц, необходимо проводить трудозатратную адаптацию настроек.

Наш выбор пал на Gulp. Это программа доступна для гибкой настройки под сборку статических страниц. Также поддерживает множество надстроек и дополнений. В нашем проекте Gulpследит за изменением Javascript-файлов, scss-файлов, html-файлов. Еще одну функцию, которой мы часто пользуется посредством данной программы является конечная сборка проекта. Это сборка, при которой все файлы минифицируются для уменьшения общего объема файлов, необходимых для загрузки пользователем. Gulp позволяет создавать скрипты (небольшие части кода), которые могут в автоматическом режиме производить сборку продукта.

Использование программ-сборщиков позволило сэкономить время на сокращении выполнения однотипных операций, которые могут отнимать не один час рабочего времени разработчика.

Проект разрабатывался с учетом использования статических страниц. Используется стандартная схема работы клиент-серверного приложения:

1. сервер получает запрос на выдачу страницы;
2. происходит выдача страницы.

Была возможность подключить frontend-фреймворк, такой как React.js[27], который бы позволил добиться следующих показателей:

1. перенос большинства логики с серверной части на клиентскую, тем самым уменьшая нагрузку на сервер;
2. бесшовное использование ресурса. При переходе по ссылке, страница не перезагружается, а лишь обновляется наполнение. Повышается удобство использования сайтом.

Перенос логики на клиентскую часть с одной стороны избавляет сервер от излишней нагрузки, но в то же время нагружает клиентскую часть. Слабые устройства, такие как телефоны, планшеты, могут не справляться с нагрузкой.Бесшовное использование также нагружает клиентскую часть приложения.

Однако, ввиду недостаточного для разработки полноценного frontend-приложения времени, было принято решение использовать традиционную схему клиент-серверного приложения, с постоянной выдачей страниц сервером. Также из-за соображений быстродействия, решено пользоваться встроенными средствами серверного рендеринга страниц.

В качестве статичных страниц используются Django-шаблонизатор. Шаблонизатор позволяет создаваться страницу с уже предопределенными местами, куда вставятся данные сервера.

Шаблон также позволяет использовать различные конструкции, такие как:

1. условный оператор;
2. цикл;
3. регулярные выражения;
4. функции.

Цикл позволяет создавать повторяющиеся элементы страниц, тем самым экономя размер файла.

Функции служат для преобразования серверных данных в необходимый на странице формат.

Далее рассмотрим реализацию клиентской части приложения.

3.1.2Общая разметка страниц

Визуальная часть приложения должна иметь удобный для использования интерфейс. Для решения этой задачи мы решили разделить страницу на две логические единицы:

1. меню или панель навигации;
2. рабочая область.

Структура всех страниц, следующая: в левой части страницы панель навигации (класс sidebar), остальную часть страницы занимает блок с основным контентом (класс content). Эти элементы содержатся во flex-контейнере (класс flex-container). Как эти элементы выглядят на клиентской части показано на рисунке 34.

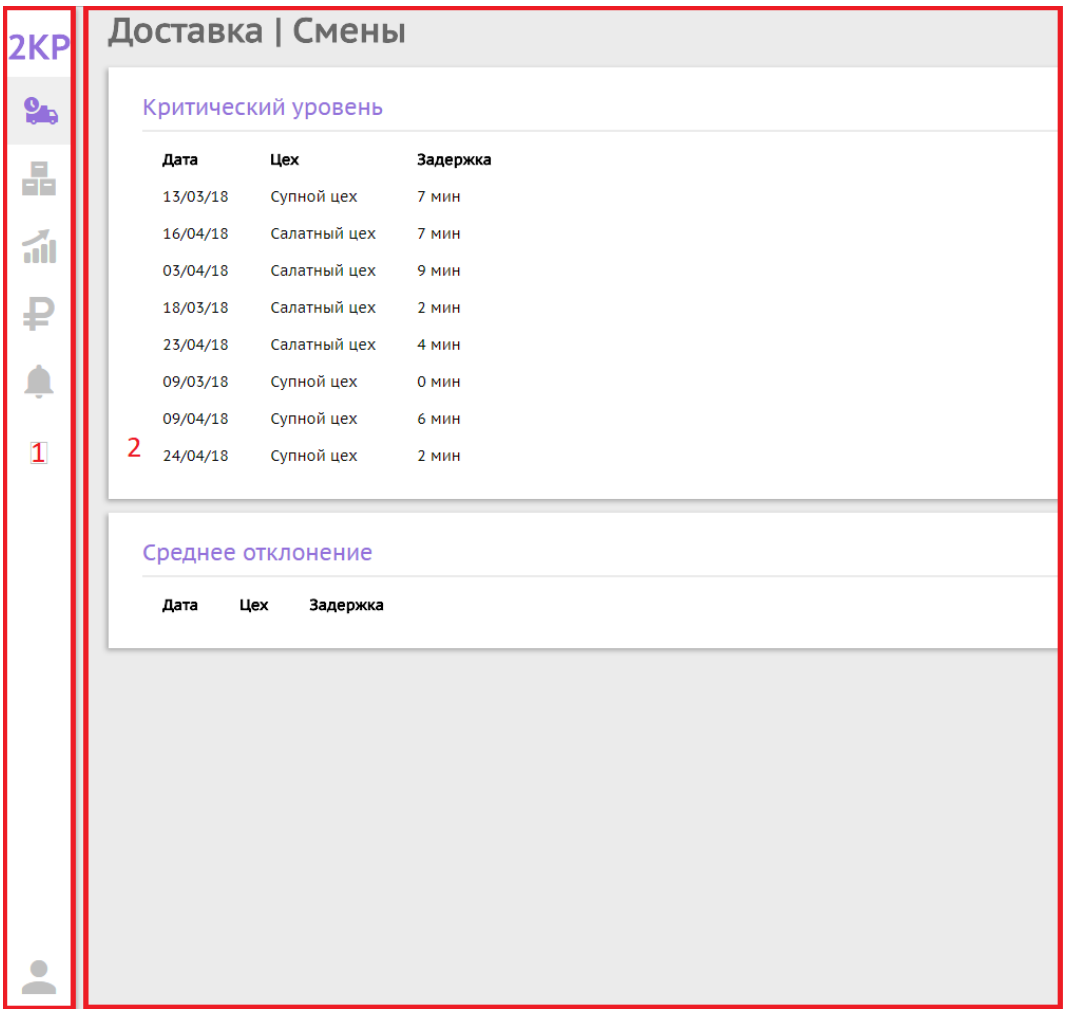

Рисунок 33– Блоки меню и рабочей области страницы приложения

Элемент меню находится строго слева от рабочей области и занимает 100% высоты экрана. Также все элементы меню расположены в столбец. Для достижения подобного отображения используется блочный подход в построении страницы – элементы страницы делятся на блоки до тех пор, пока блок не станет атомарной единицей, т.е. дальнейшее деление будет невозможно. Элемент меню разделен на следующие блоки:

1. блоки иконок;
2. блок логотипа;
3. блок основного массива иконок;
4. основной блок-обертка, который содержит в себе все остальные блоки.

Отображение элемента меню, как это представлено на рисунке 34, достигаются посредством использования следующих CSS-стилей:

1. высота (height);
2. расположение элементов (flex-direction);
3. высота слоя (z-index).

Также для улучшения пользовательского восприятия страниц, была сделана анимация блоков иконок – при наведении блок подсвечивается, давая понять пользователю на какую иконку он навелся в данный момент. При нажатии на иконку, она подсвечивается другим цветом, отличным от цвета наведения. При этом, если убрать мышь с блока иконки, подсветка не исчезнет, давая понять пользователю, что было произведено нажатие и переход к новой странице.

Элемент рабочей области находится справа от меню. Данный элемент содержит в себя общие для всех страниц блоки:

1. блок-обертка, который содержит в себе все остальные блоки;
2. блок заголовка;
3. блок информации;
4. блок подзаголовка.

Также выделены универсальные для всех страниц стили рабочей области:

1. высота (height);
2. настройки шрифтов;
3. цвет фона (background-color).

3.1.3Страница доставки

Рабочая область страницы доставки представлена на рисунке 35.

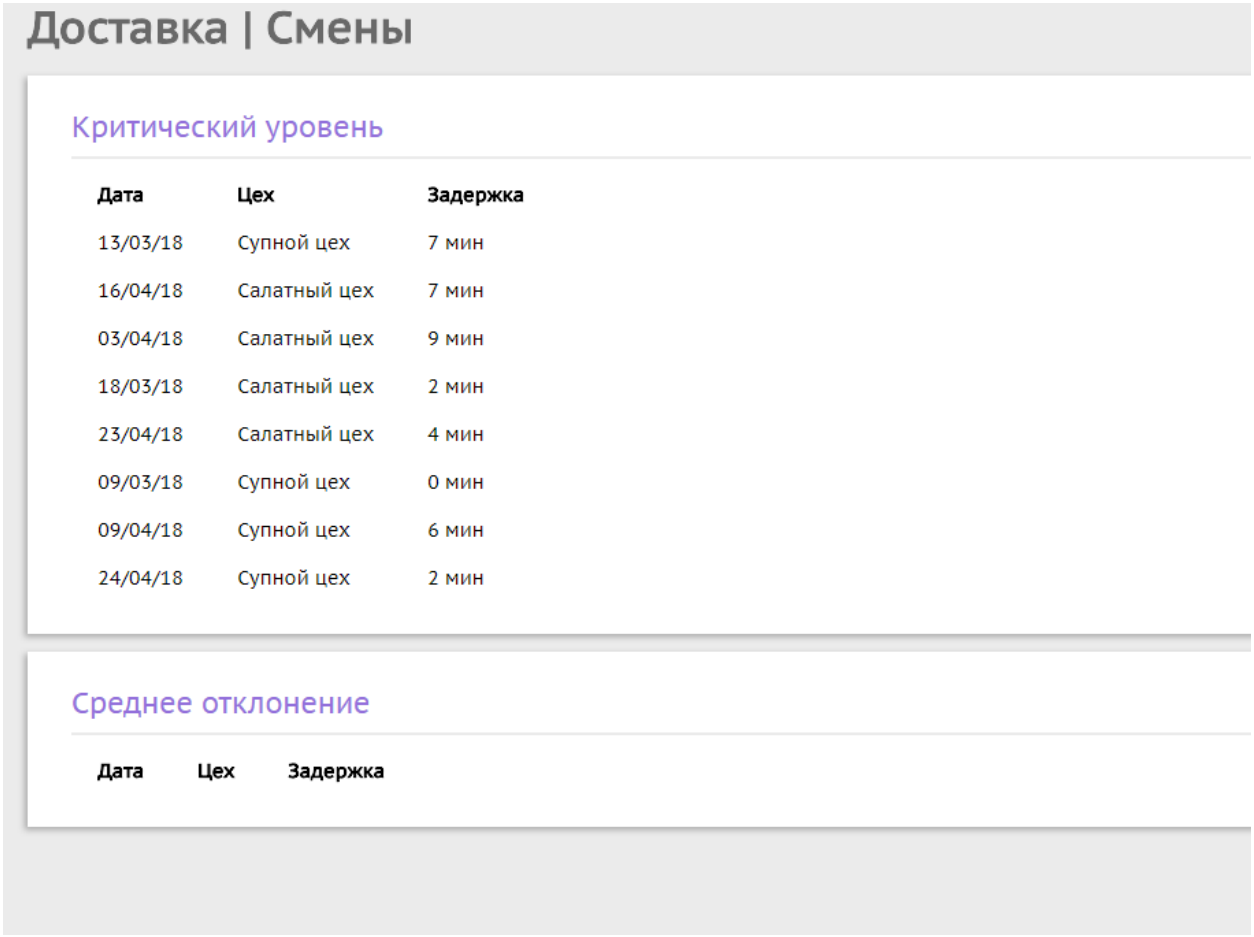

Рисунок 34– Рабочая область страницы доставки

Страница доставки состоит из двух блоков, в которых содержится информация о проблемных заказах. С точки зрения структуры и стилизации оба блока идентичны – оба имеют подзаголовок, который отражает степень критичности проблемы, оба содержат таблицу с просроченными заказами. Остановимся подробнее на реализации таблицы.

Наша задача была максимально упростить восприятие информации пользователем. Следовательно, нагружать интерфейс бесконечным количеством столбцов таблицы нельзя.

Мы выделили самую необходимую информацию в 3 столбца:

1. дата, когда произошел инцидент с заказом;
2. цех, ответственный за задержку;
3. задержка – размер задержки.

Выделив три основных столбца, мы добились сокращение потери внимания пользователем при просмотре страницы.

Но, чтобы охватить всю необходимую информацию о заказе, трех столбцов с основными показателями недостаточно. Для полного описания заказа необходимо еще 2 пункта:

1. состав заказа с количеством блюд;
2. список поваров с рейтингом.

Мы поместили эти пункты в отдельные блок, который по умолчанию скрыт от пользователя, но становится доступным при клике на строку заказа. Это показано на рисунке 36.

На этом блоке присутствуют оба вышеперечисленных пункта, а также кнопка «Решить»

Цвет блока совпадает с цветом выбранной строки для устойчивой связи «строка-дополнительны блок», возникающей у пользователя.

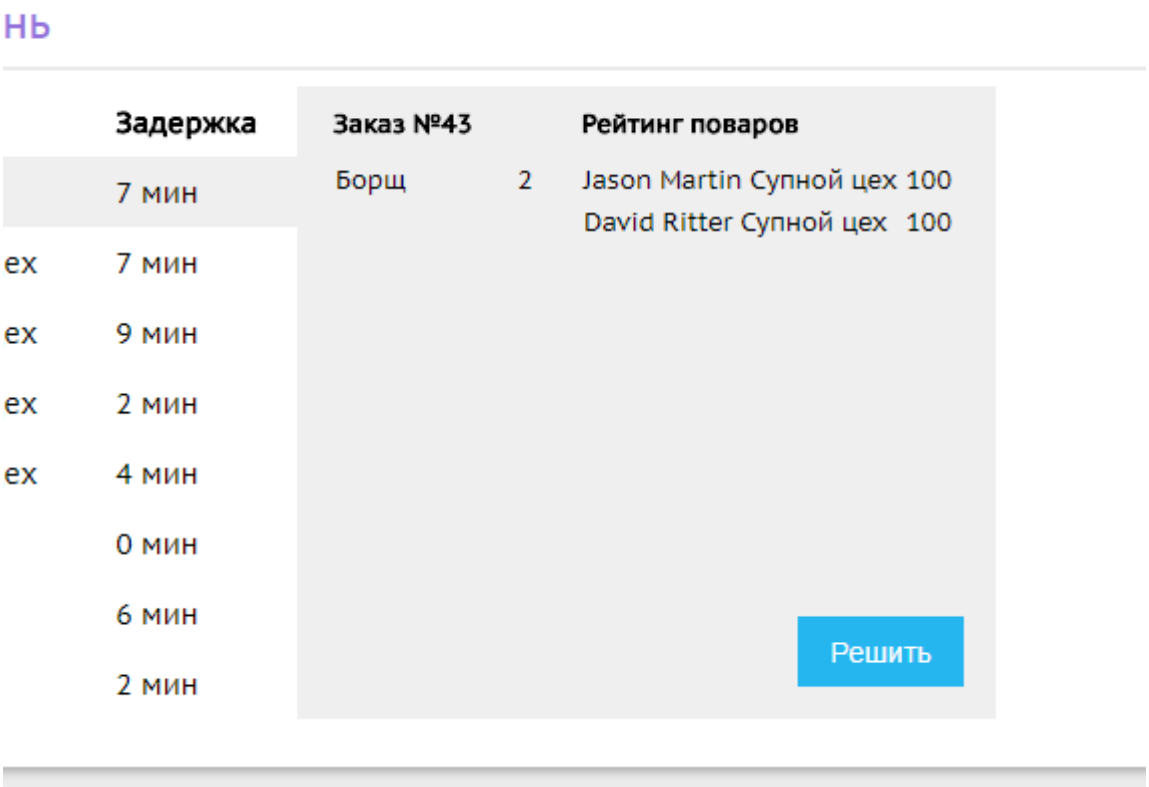

Рисунок 35 – Блок с дополнительной информацией о заказе

Для реализации таблицы была выбрана HTML-разметка с соответствующими тегами табличной части. Через CSSстили были определены шрифты, размер и цвет строк табличной части. Для того, чтобы при нажатии менялся цвет строки, был написан Javascript-код, который определяет какая именно строка была нажата, и подсвечивает ее выбранным заранее цветом. Параллельно с этим выделение других строк автоматически снимается.

Для реализации блока с дополнительной информацией было произведено разделение на блоки:

1. блок с табличной частью;
2. блок с кнопкой.

Блок с табличной частью соответствует аналогичному блоку из предыдущего абзаца. Блок с кнопкой располагается внизу блока с дополнительной информацией.У кнопки определен свой цвет. Используются следующие CSS-правила:

1. отступы (margin);
2. цвет фона (background-color).

Также написан Javascript-код для обработки появления соответствующего блока с дополнительной информацией, в зависимости от выбранной строки заказов. Алгоритм реагирует на нажатие на строку и ищет соответствующий строке блок с дополнительной информацией. При этом, если открыт другой блок с дополнительной информацией, то он скрывается, уступая место новому.

При нажатии на кнопку «Решить» открывается еще один блок, который позволяет выбрать определенных поваров и решить проблему, удалив ее из списка проблемных заказов (рисунок 37). Блок располагается поверх страницы, фон затемняется. Все это сделано для фокусировки внимания пользователя на текущем действии. При нажатии на «Отправить» или «Отмена», блок пропадает, давая продолжить пользоваться страницей.

Для отображения блока используется javascript-код, который в зависимости от блока дополнительной информации вызывает отображение блока с соответствующим содержанием.

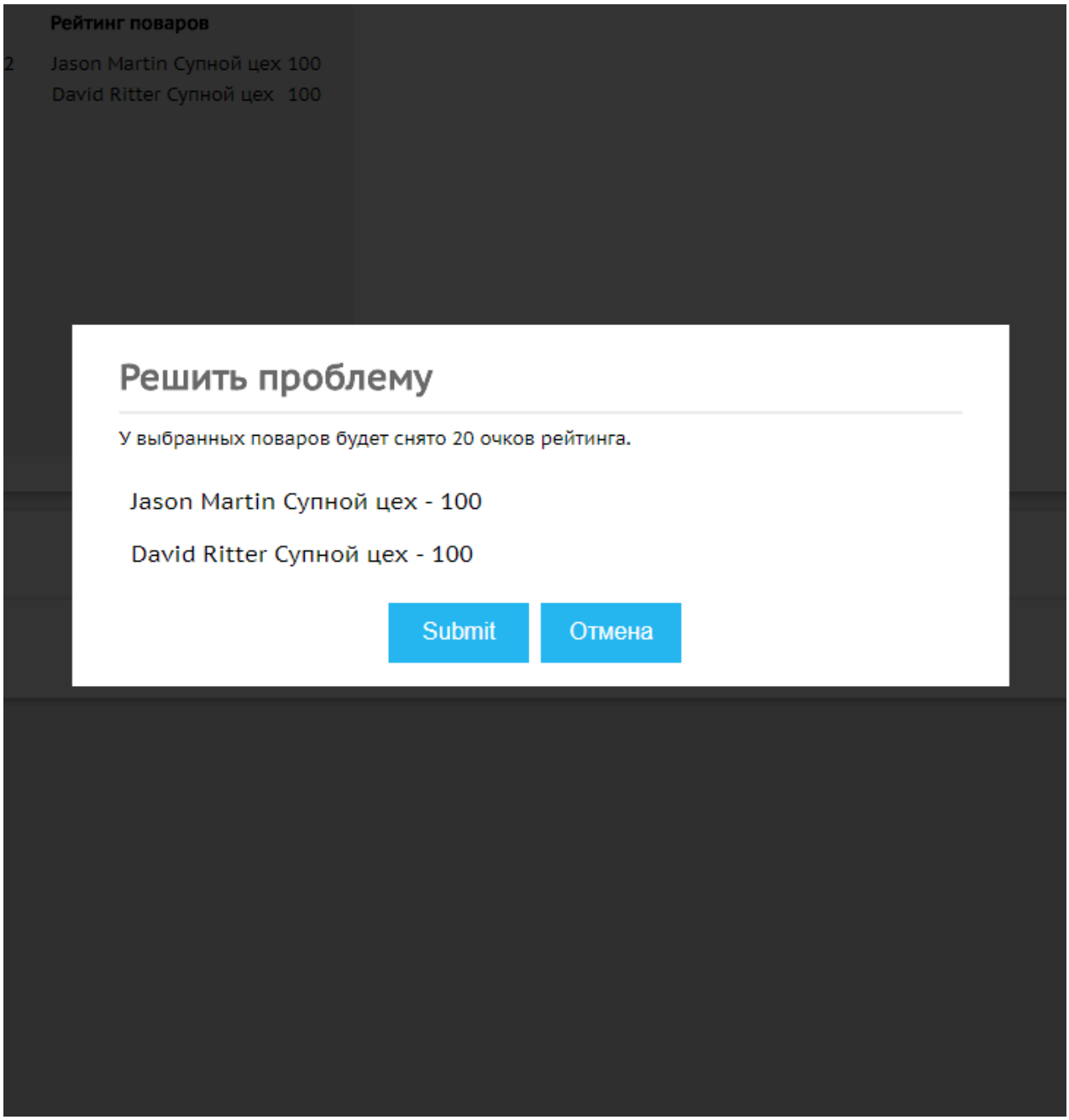

Рисунок 36– Блок решения проблемы

Через CSS-стилизацию настраивается позиционирование кнопок по центру и настраивается отображение шрифтов.

3.1.4Страница продаж

Рабочая область страницы продаж показана на рисунке 38.

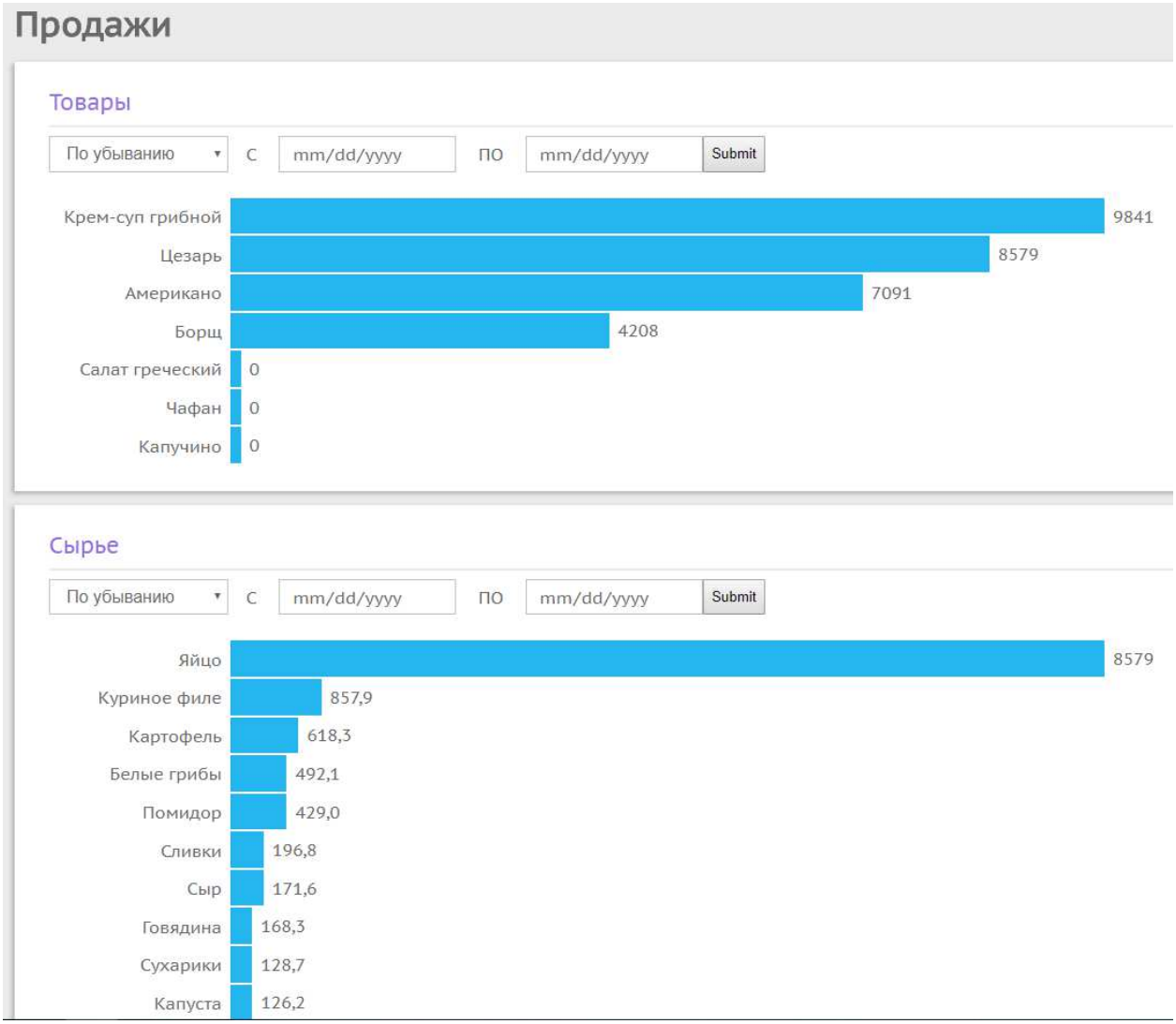

Рисунок 37– Рабочая область страницы продаж

Страница продаж, по аналогии со страницей доставки, состоит из двух, идентичных по реализации, блоков. Каждый блок состоит из следующих дочерних блоков:

1. подзаголовок;
2. панель фильтрации;
3. график.

Панель фильтрации содержит в себе элемент выпадающего списка для сортировки графика по убыванию/возрастанию и два поля, которые отражаются временной интервал, для которого отображается информация о продажах.

Чтобы не отвлекать внимание пользователя на второстепенный элемент интерфейса (панель фильтрации) через средства стилизации были применены правила, которые позволили поменять цвет полей и шрифтов на серый, менее броский цвет. Данным шагов мы избавились от информационного шума, который мог помешать пользователю сосредоточиться на анализе графика.

Панель фильтрации реализована через соответствующие HTML-теги:

1. тег формы (form);
2. тег элемента для отправки (input).

Реализация позволяет отправлять введенные пользователем данные на сервер для дальнейшей обработки.

При клике на поле введения даты, используя встроенные элементы HTML5, открывается календарь, с помощью которого можно удобно выбрать дату (рисунок 39).

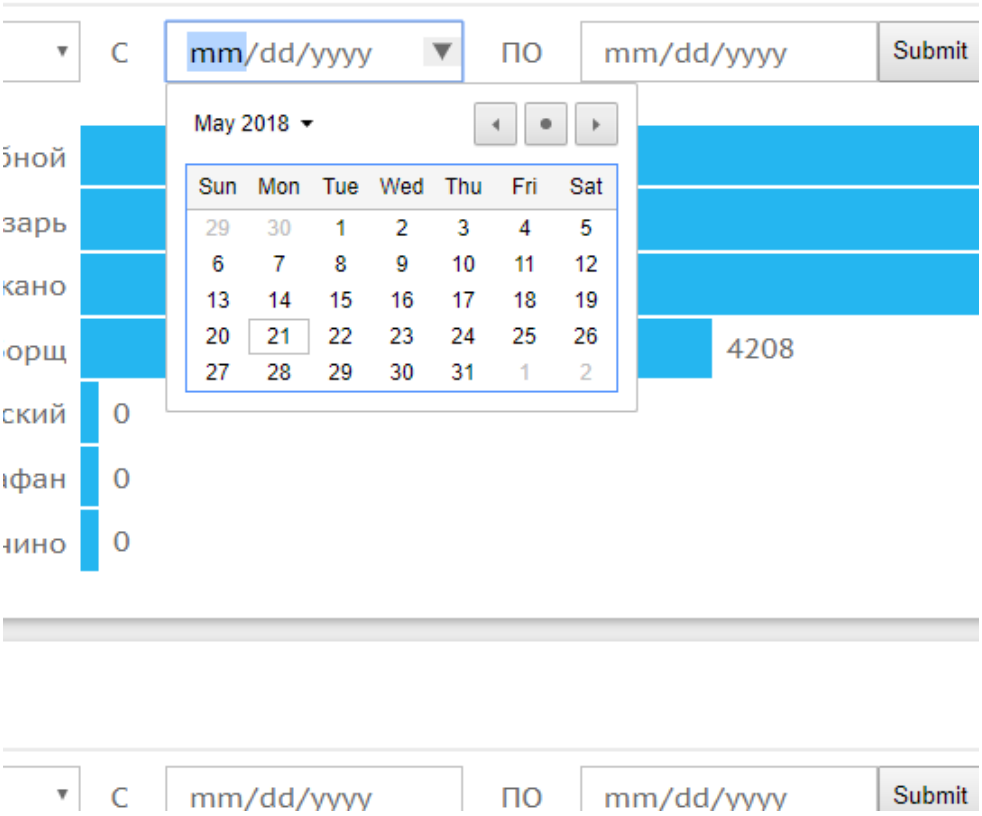

Рисунок 38– HTML5 календарь

График отображается строками, где каждая строка содержит в себе три блока:

1. название;
2. количество в виде диаграммы;
3. количество в виде числа.

В целях улучшения восприятия пользователем количественной информации, было принято решение изобразить количество продаж в виде столбчатой диаграммы. Пользователь может оценить уровень продаж, лишь взглянув на пропорции столбцов диаграммы. Также в целях удобства восприятия, длинные названия скрываются, уступая место более коротким наименованиям. При наведении мышью на название, можно прочитать полное название во всплывающей подсказке (рисунок 40).

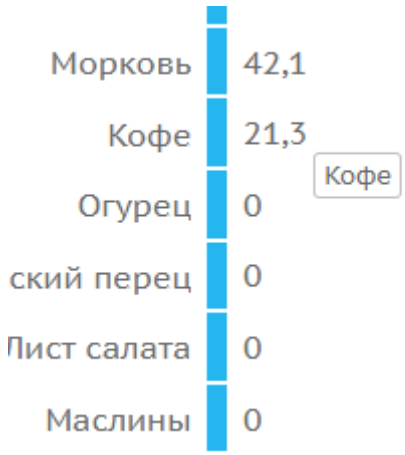

Рисунок 39– Всплывающая подсказка

В конце каждого столбца приведен количественный показатель уровня продаж определенной позиции.

Чтобы реализовать динамическое построение диаграммы через CSS-правила, использовалась inline (встроенная) стилизация. Данные с сервера подставляются в inline-стили HTML-объектов, и график заполняется данными шаг за шагом.

Для стилизации блока с графиком использовались следующие CSS-правила:

1. ширина элемента (width);
2. цвет фона (background-color);
3. настройки шрифтов.

Каждый столбец диаграммы заключен в ссылочный тег «a» для перехода на страницу прогноза.

3.1.5Страница прогноза

Страница прогноза представлена на рисунке 41.

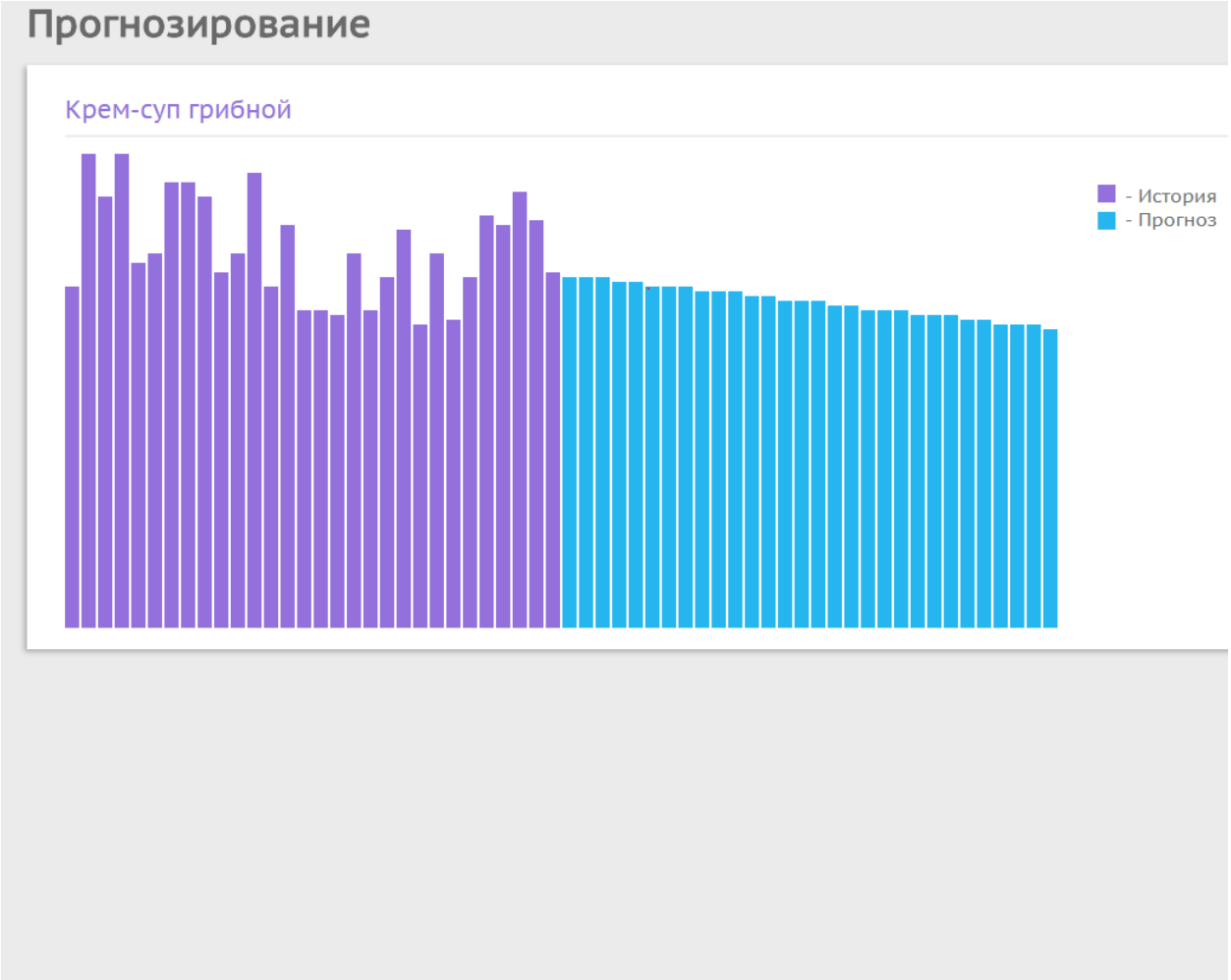

Рисунок 40– Страница прогноза

Страница прогноза состоит из одного типового блока, который, в свою очередь, состоит из следующих дочерних блоков.

1. подзаголовок;
2. график;
3. легенда.

Подзаголовок на данной странице динамический. На этапе серверного рендеринга страницы, происходит подставка значения наименования предмета, для которого строится график.

Страница создавалась с использованием принципа минимализма – самая необходимая информация представлена пользователю в первую очередь. В нашем случае, главным элементом страницы является диаграмма, которая отражает динамику уровня продаж отдельно взятого объекта системы. Стояла

задача наглядно показать пользователю системы динамику потребления. Для решения этой задачи было принято решение убрать все отвлекающие элементы:

1. количество продаж в день;
2. дату продажи.

Вышеперечисленные элементы убраны в специальную всплывающую подсказку, которая появляется при наведении мыши на определенный столбец диаграммы (рисунок 42). При этом, чтобы облегчить пользователю момент сопоставления всплывающей подсказки и выбранного столбца, столбец при наведении меняет цвет.

Половина диаграммы занимает история потребления, другую половину – прогноз. Чтобы визуально разграничить две области, мы использовали разные цвета столбцов. Для пользователя системы выведена легенда, которая позволит идентифицировать ту или иную часть графика.

Для реализации столбчатой диаграммы использовались inline-стили из предыдущего пункта – на сервере рассчитываются все значения диаграммы и подставляются в каждый столбец графика по очереди. При этом, чтобы сохранить пропорциональность всех столбцов относительно значений, которые они представляют, в формулу расчета высоты столбца (формула 1) подставляется максимальное значение графика.

$$\text{высота} = \frac{\text{значение текущего столбца}}{\text{максимальное значение графика}} \times 100\% \qquad (1)$$

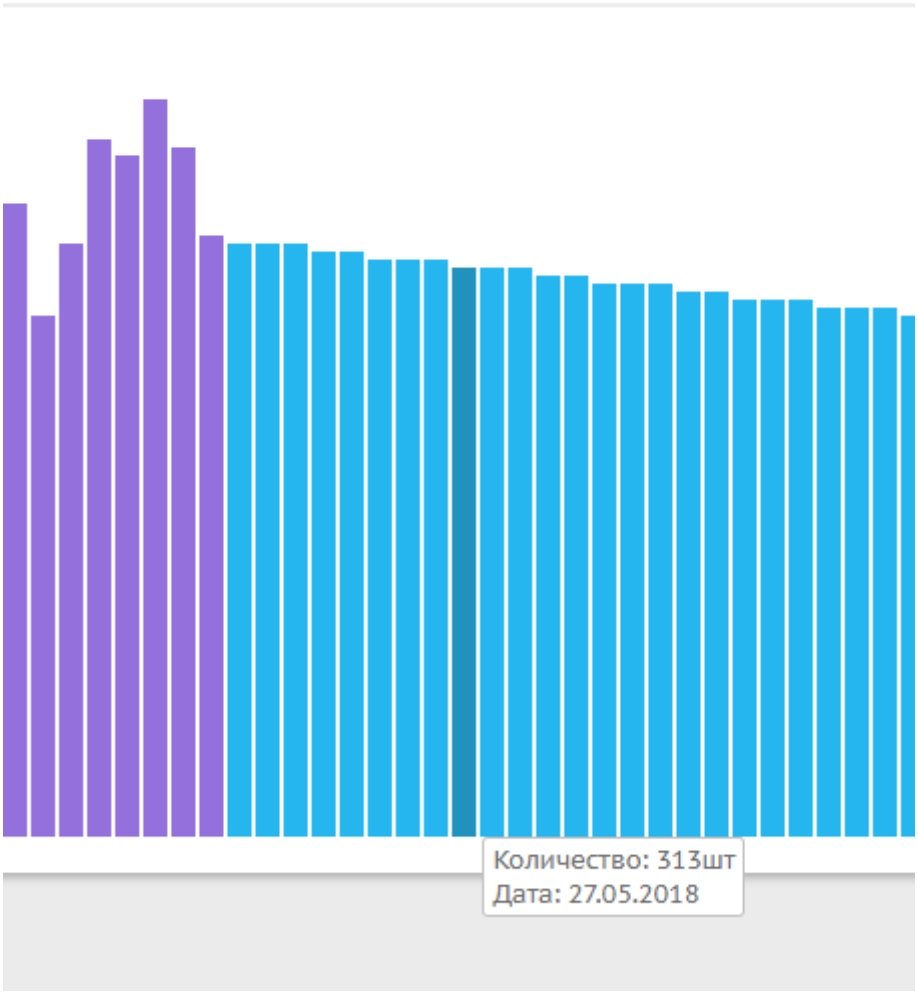

Рисунок 41– Всплывающая подсказка на странице прогнозирования

Для стилизации графика использовались стандартные CSS-правила:

1. высота (height);
2. ширина (width);
3. цвет фона (background-color).

Для реализации функционала всплывающей подсказки использовались события в CSS-процессоре, которые реагируют на наведение мыши на элемент. При наведении мыши меняются стили:

1. Отображение (display) – правило отвечает за то, как будет отображаться элемент на экране, и будет ли вообще отображаться или скрыт от глаз пользователя.

2. Цвет фона (background-color) – правило перезаписывает уже определенный для столбца цвет через постфикс !important. Это сделано с целью переопределения цвета столбца при наведении.

В качестве реализации были выбраны CSS-события только с одной целью – производительность. Использования для этой цели средства javascript-кода было бы расточительно в виду использования дополнительный ресурсов для перерисовки страницы во время исполнения кода.

Легенда для диаграммы использует стандартные стили для отображения, а именно абсолютное позиционирование для отображения в правом верхнем углу блока.

3.1.6Страница остатков

Рабочая область страницы остатков (рисунок 43) представляет из себя одиночный блок с серией графиков. Деление на блоки следующее:

1. подзаголовок;
2. график;
3. легенда.

Мы использовали ту же цель, что и в предыдущем параграфе – минимализм и наглядность. В данном случае было решено сделать серию графиков, где каждый график состоит из двух столбцов:

1. норма остатков;
2. текущее количество.

Рисунок 42– Рабочая область страницы остатков

Чтобы решить проблему с наглядностью информации, мы решили два этих столбца объединить под одним названием объекта, чтобы можно было быстро определить какой именно объект в центре внимания и какая ситуация обстоит с количеством объекта. Принцип наглядности довольно прост – если столбец текущего количества визуально длиннее столбца с нормами, то, значит, ситуация с остатками сырья нормальная.

Для сохранения пропорциональности между длиной столбца и количественным значением используется формула 1.

3.1.7 Страница уведомлений

Рабочая область страницы уведомлений (рисунок 44) делится на два идентичных блока.

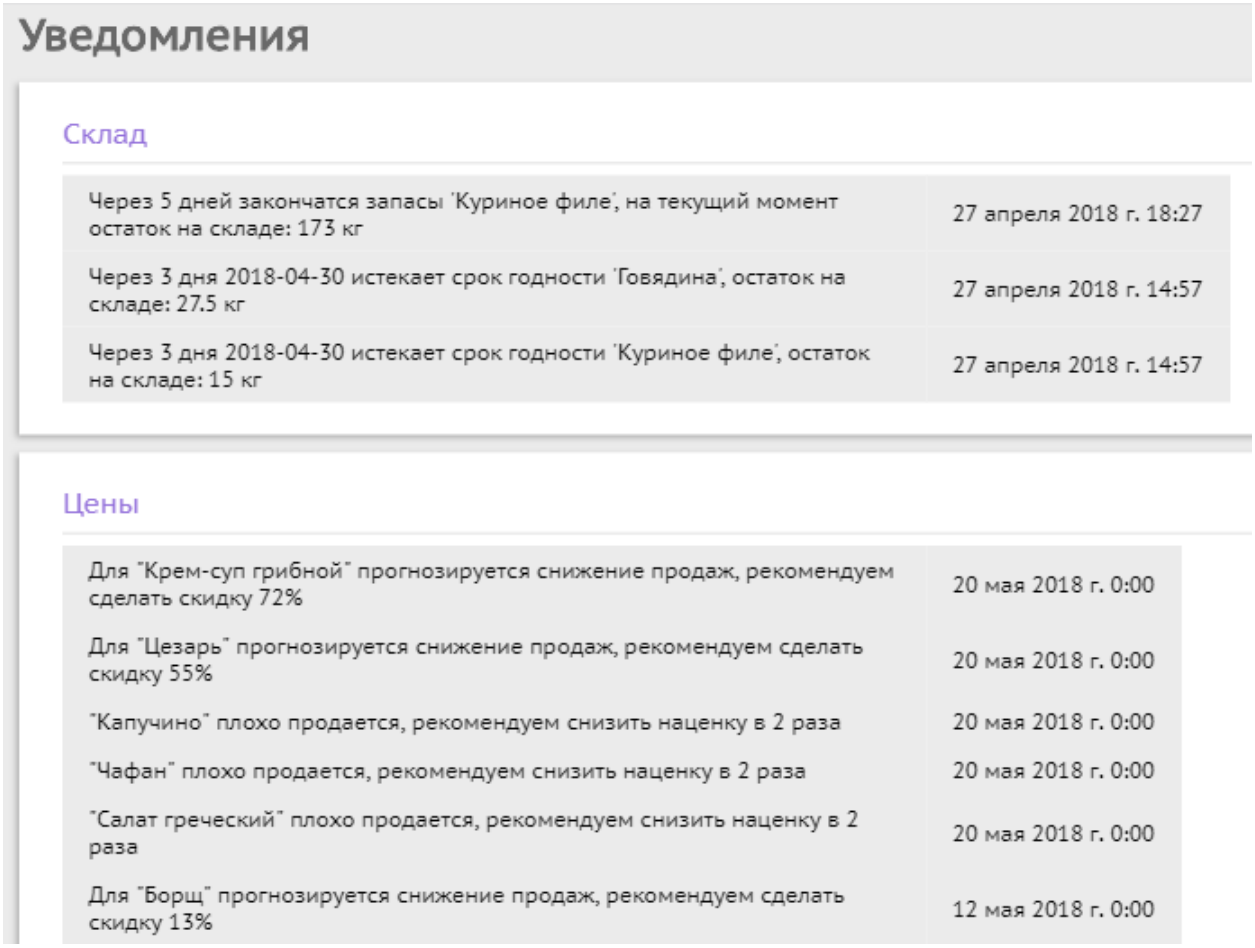

Рисунок 43 – Рабочая область страницы уведомлений

Каждый блок содержит в себе внутренние блоки:

1. подзаголовок – определяет принадлежность сообщений к одному классу;
2. сообщение.

Страница позиционируется как накопитель различного рода уведомлений, которые классифицируются по содержанию (цена, склад), так и по статусу (прочитано, не прочитано). На рисунке 45 показан пример непрочитанного и прочитанного сообщения.

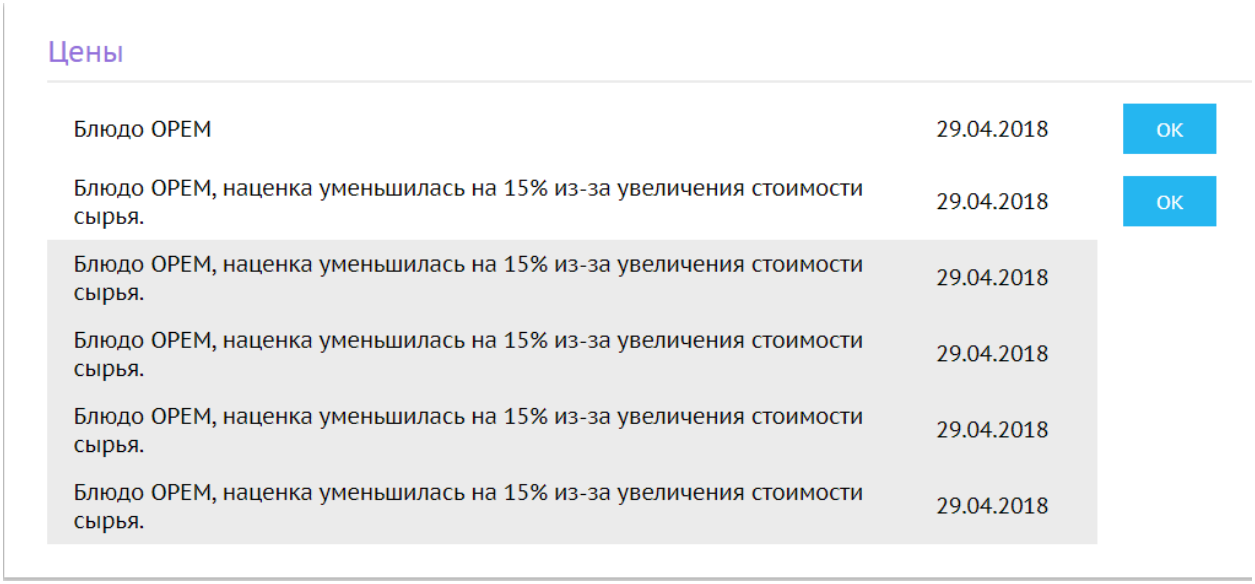

Рисунок 44– Пример непрочитанного/прочитанного сообщения

Цель для создания статусов сообщений проста – наглядно показать пользователю какие именно сообщения новые и требуют внимания, а какие лишь отражают историю работы системы.

Для разграничения статуса сообщения используются CSS-стили:

1. цвет фона (background-color);
2. отображение (display) – скрываетили отображает кнопку у сообщения.

Также к каждому новому сообщению добавляется кнопка «ОК». При нажатии на эту кнопку, идет отправка данных на сервер, и он в свою очередь помечает сообщение как прочитанное. Такая последовательность действий сделана для того, чтобы пользователь не мог упустить важные сообщения из виду. Нажимая на кнопку «ОК», пользователь подтверждает тот факт, что он ознакомился с информацией и примет соответствующие меры.

3.1.8Страница графика смен

Рабочая область страницы графика смен представлена на рисунке 46.

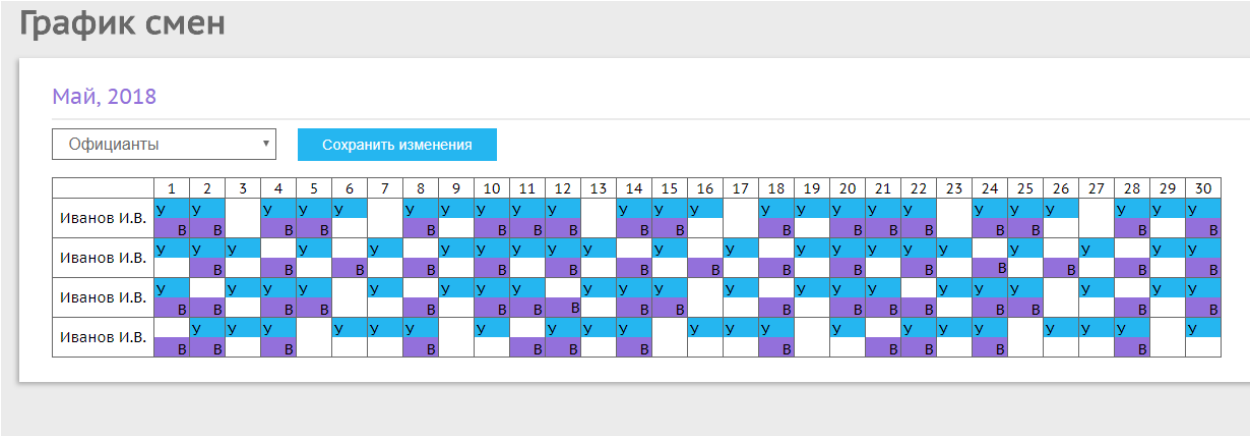

Рисунок 45– Рабочая область страницы графика смен

Страница состоит из одного блока со следующими элементами:

1. подзаголовок, который отображен информацию о месяце, для которого составляется график работы;
2. панель управления, которая содержит в себе селектор с типом персонала, для которого требуется составить расписание, и кнопку «Сохранить изменения», при нажатии на которую происходит отправка данных на сервер;
3. график работ – таблица, где каждая строка отражает расписание одного из работников.

При создании страницы, мы отталкивались от необходимости видеть пустые места в графике загрузки. Те места, где количество персонала недостаточно для функционирования предприятия. Поэтому мы решили использовать некую интерпретацию диаграммы Ганта, которая позволит с легкостью определить пустые дни и перераспределять нагрузку.

Строка таблицы – расписание отдельно взятого человека. Столбцы – дни месяца. Каждая ячейка делится пополам – на дневную и вечернюю смену.

При разработке страницы использовалась технология AJAX, которая позволяет отправлять данные на сервер в фоновом режиме, не перезагружая страницу. Сделано это для того, чтобы не отвлекать пользователя от разработки графика. Написан Javascript-код, который реагирует на любые действия

пользователя, а именно клики мыши. Если пользователь нажмет на селектор выбора типа персонала и выберет другой тип, то функция поймет, что произошло нажатие на селектор и выбран другой тип персонала. Затем создаст запрос на сервер и перерисует таблицу с пришедшими с сервера данными. Аналогично работает назначение работнику смены. Пользователь нажимает на ячейку под определенной датой, функция определяет какую именно ячейку и какую смену выбрал пользователь. Эти данные в фоновом режиме отправляются на сервер, который в свою очередь возвращает ответ – можно или нельзя ставить смену на выбранный день. Если никаких препятствий нет, ячейка закрасится в цвет, который зависит от выбранной смены.

Чтобы сохранить изменения, необходимо нажать на кнопку «Сохранить изменения» на панели управления. В случае успешного сохранения, появится соответствующее сообщение (рисунок 47).

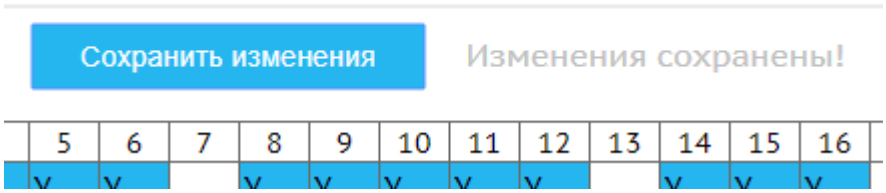

Рисунок 46– Успешное сохранение данных

В качестве CSS-правил используются стандартные стили для позиционирования и определения цвета фона.

3.1.9Адаптивность страниц

Мы считаем, что программной системой должно быть удобно пользоваться как на настольном компьютере, так и на планшете и мобильных устройствах. Поэтому для нас было важным провести адаптацию страниц, используя принципы минимализма и фокуса на основной информации.

Адаптивность представлена на странице доставки для планшета (рисунок 48) и смартфона (рисунок 49).

Для адаптации страниц под выбранные разрешения используются media-запросы CSS-процессора, который определяет ширину устройства и перезаписывает стили для каждого разрешения.

Для адаптации страницы доставки под ширину экрана планшета, мы использовали стили, которые позволяют перенести блок с дополнительной информацией под таблицу, выстраивая содержимое страницы по вертикали.

Для адаптации под мобильные устройства нам пришлось отказаться от боковой панели меню, так как она занимала и без того малую ширину экрана. Было решено перенести меню наверх экрана и скрыть, чтобы не отвлекать пользователя от основного содержимого. Меню доступно по клику на иконку в правом верхнем углу экрана (рисунок 50).

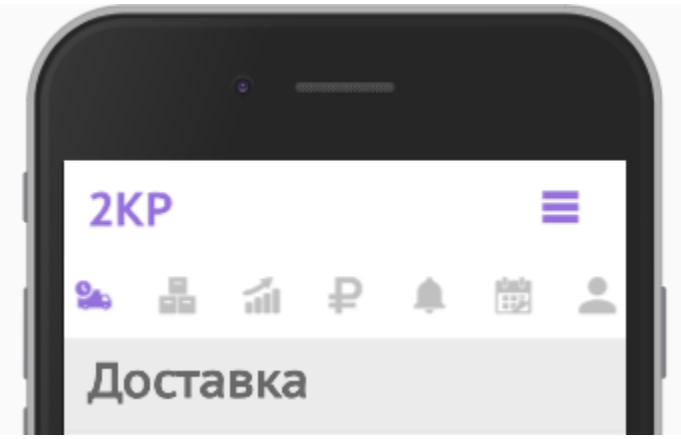

Рисунок 47– Версия меню для мобильных устройств

Для открытия меню используется javascript-функция, которая реагирует на нажатие иконки и перезаписывает CSS-правила для меню, которые отвечают за отображение элемента.

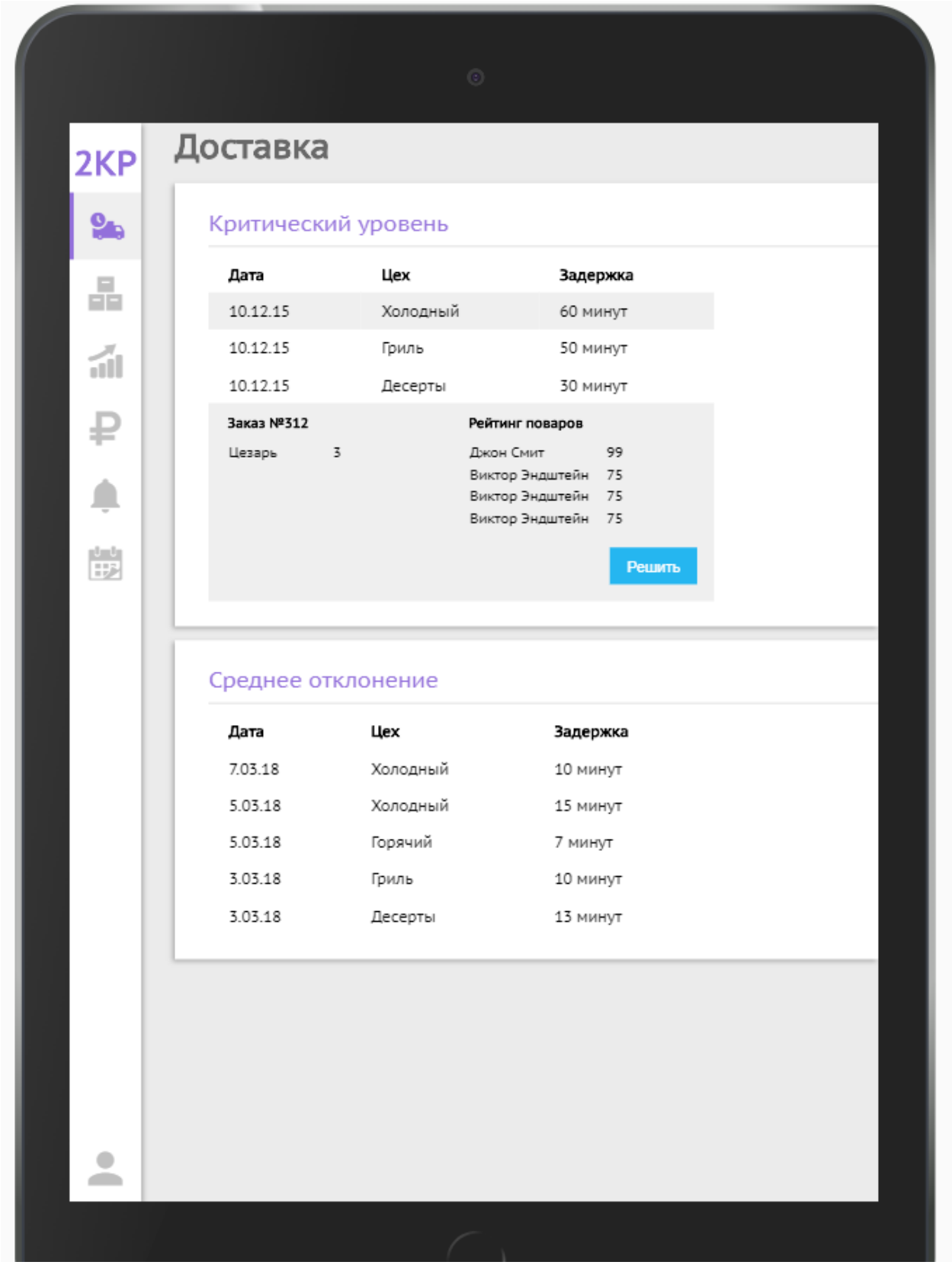

Рисунок 48– Адаптация страницы для планшета

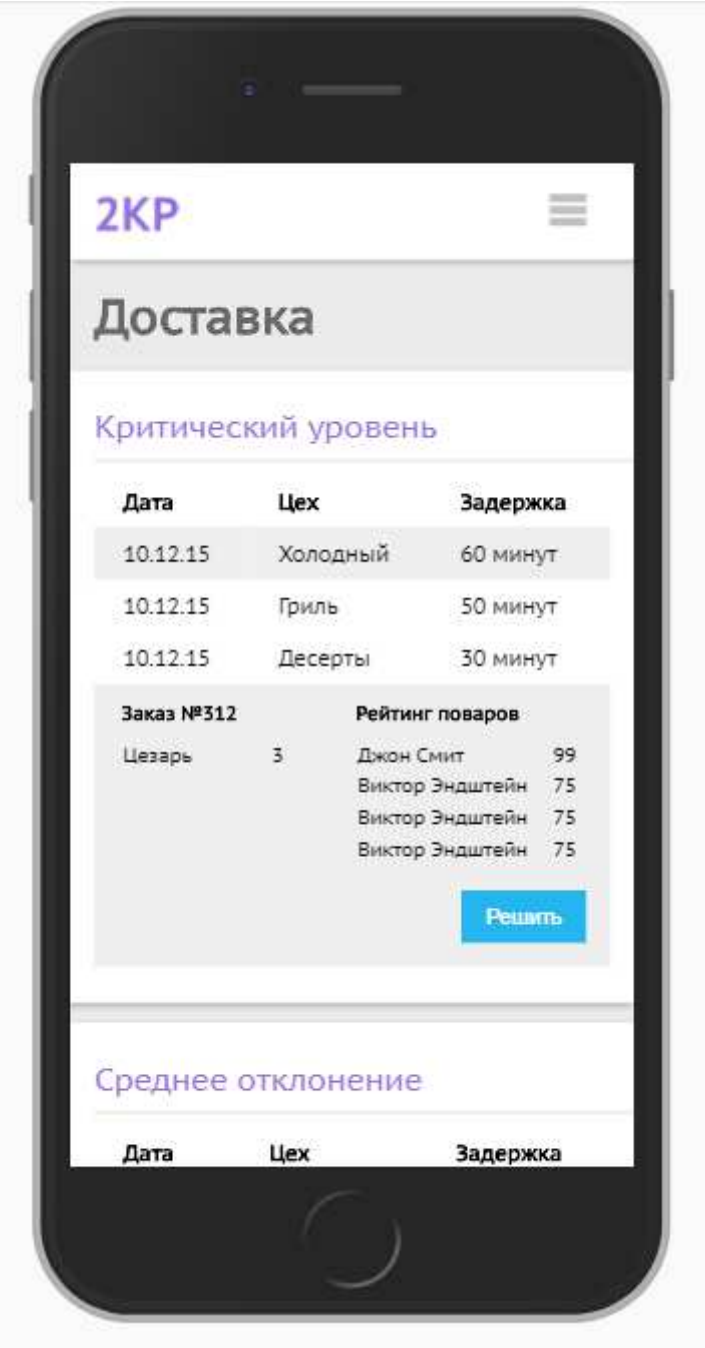

Рисунок 49– Адаптация страницы для смартфона

3.1.10 Роли в программном продукте

1. Администратор – авторизированный пользователь.

Полный доступ ко всем функциональным возможностям управления и администрирования программной системы.

- блок «Доставка/Смены» - просмотр, отправление проблем, редактирование рейтинга сотрудников;
- блок «Остатки» - просмотр;
- блок «Продажи» - просмотр, сортировка, редактирование данных, переход по товару на вкладку «Прогнозирование»;
- блок «Уведомления» - просмотр;
- блок «График смен» - просмотр, редактирование данных, сохранение измененного расписания сотрудников, выбор нужной сортировки.

2. Официант – авторизированный пользователь.

Доступ к странице учета смен.

- Блок «График смен» - просмотр, редактирование своих данных, сохранение.

3. Повар холодного цеха – авторизированный пользователь.

Доступ к странице учета смен.

- Блок «График смен» - просмотр, редактирование своих данных, сохранение.

4. Повар горячего цеха – авторизированный пользователь.

Доступ к странице учета смен.

- Блок «График смен» - просмотр, редактирование своих данных, сохранение.

5. Уборщик – авторизированный пользователь.

Доступ к странице учета смен.

- Блок «График смен» - просмотр, редактирование своих данных, сохранение.

3.2Процесс тестированияверстки

Проблема тестирования вёрстки в том, что только живой человек может сказать, хорошо свёрстан блок на странице или нет. Поэтому мы будем
тестироватьHTMLиCSSвручную:проверяем,какбудетвестисебяблок,еслив нем будет слишком много (или слишком мало) текста или дочерних элементов; смотрим, чтобы все возможные варианты отображения блока смотрелись корректно; помним о том, как блоки должны адаптироваться к разным устройствам и разрешениям экрана, к разнымбраузерам.

Сначала нужно посмотреть на требования, дизайн. На основе этого составляем список всех значимых состояний приложения, которые нужно проверить. Далее, дело за малым — проверить каждый тест-кейс.

3.2.1 Написание чек-листов

Рассмотри стандартные требования к верстке.

Проверка соответствия макету. Допускается расхождение до 5px для текста. Разрешены и даже приветствуются правки размеров и расположения криво нарисованных блоков (разница размерах в 1-2px на разных страницах). В нашейпрограммной системе нет требования на строгое соответствие макету, поэтому этот пункт тестирования мы опустим.

Кроссбраузерность. Корректное отображение в браузерах, указанных в задаче на тестирование. Для нашей программной системы: Firefox (версия 60.0.1), Chrome (версия 66.0.3359), Opera (версия 52.0.2871.37).

Проверка на всех необходимых разрешениях. Всегда следует проверять, как страница реагирует на уменьшение масштаба, таким образом, эмулируя просмотр на устройстве с большим экраном. Особенно это актуально, если на странице есть фоновая картинка, которая должна быть на

всю область экрана. Это позволяет проверить, что она не вставлена с обрубками. Даже сайты, не адаптированные под мобильное устройство, должны на нем более-менее корректно смотреться. На мобильных устройствах следует проверять хотя бы в двух браузерах. Для нашей программной системы: адаптивность для IPhone 6/7/8 и IPad.

– отсутствие js-ошибок:в консоли браузера не должно выдаваться ошибок;

– валидацияHTML:наличиепредупреждений(Warning)при проверке возможно, ошибок не должнобыть;

– валидация CSS: корректная работа при вбивании реального текста, надёжность вёрстки.

Таким образом, чек-лист для тестирования верстки в нашей программной системе выглядит следующим образом:

- Firefox (версия 60.0.1), Chrome (версия 66.0.3359), Opera (версия 52.0.2871.37);
- адаптивность: IPhone 6/7/8 и IPad;
- отсутствие JS-ошибок вконсоли;
- валидация HTML;
- валидация CSS.

3.2.2 Валидность HTML иCSS

Валидность кода подразумевает написание HTML-кода сайта в соответствии с определенными стандартами, разработанными Консорциумом ВсемирнойПаутины—WorldWideWebConsortium(W3C).Соблюдениеправил, прописанных этим стандартом, является гарантией кроссбраузерности, то есть правильного отображения созданной страницы во всех браузерах, а также отсутствию ошибок, влияющих на скорость загрузки и другиепараметры.

В современном Интернете критерии качества верстки сайта стали играть важную роль, так как теперь вебмастеру необходимо добиться корректного отображения ресурса не только на стационарных ПК и ноутбуках, но и множестве мобильных устройств с самым разным разрешением.

То, насколько чистый и структурированный код сделан разработчиками, позволяет обнаружить проверка сайта на валидность, что осуществляется через специальныйчекернаофициальномресурсеотW3C.Здесьвсвободномдоступе находитсявалидаторHTML-кода,работающийврежимеонлайн.

С его помощью возможна проверка валидности HTML кода тремя способами:

- Validate by URI — проверка поадресу;
- Validate by File Upload – анализзагружаемогофайла;
- Validate by Direct Input – проверка определенного фрагментакода.

ПроверкасайтанавалидностьHTMLкодапозволяетпонять,нужнылиему исправления и оптимизация верстки, ведь чистота и грамотная структура кода является важной составляющей внутренней оптимизации. Качественно сделанная семантическая верстка позволяет быстро разобраться в чужом коде другому исполнителю, а также повысить текстовую релевантность и ускорить загрузкустраниц.

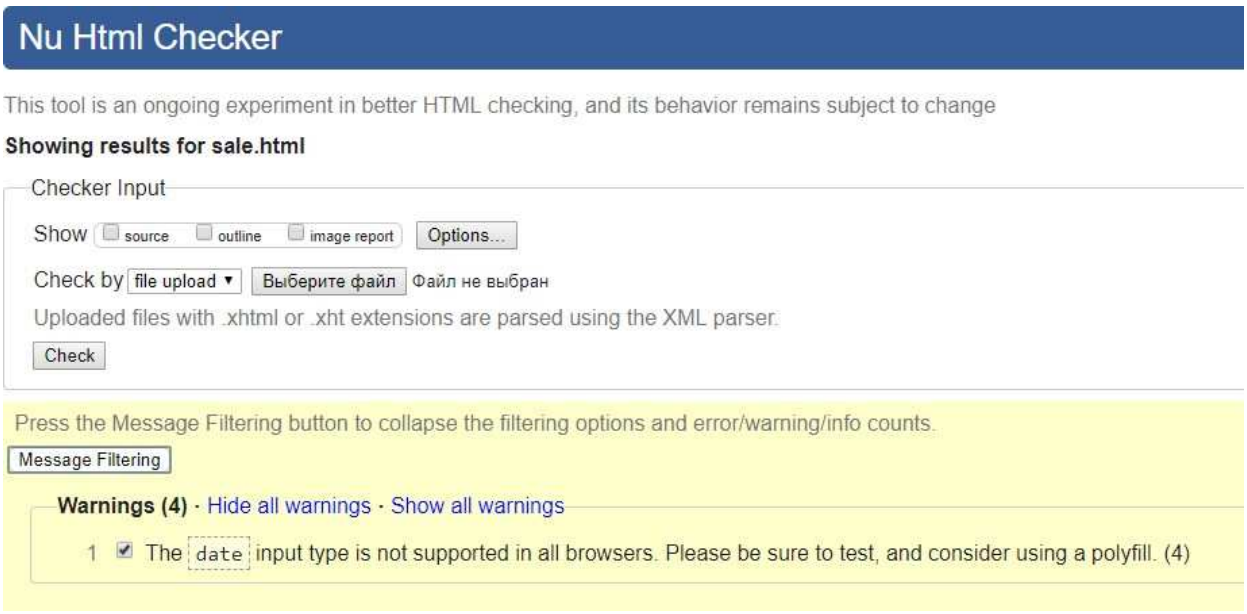

Рисунок 50– Результаты проверки на валидность файла sale.html

Результат проверки на валидность кода HTML для страницы продаж и страницыпрогнозированияпоказанынарисунке51ирисунке52 соответственно. На странице продаж имеются 4 предупреждения о том, что тип данных date в элемпенте input поддерживается не весеми браузерами. Однако, для браузеров, которыеуказанывТЗэтоттипданныхподходит.Вфайлеprediction.htmlошибок и предупреждений необнаружено.

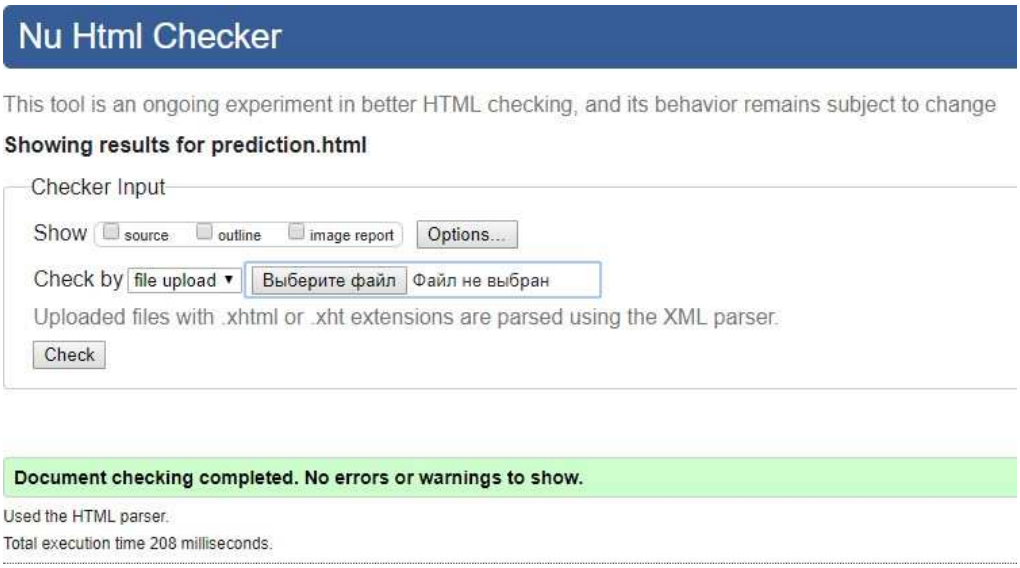

Рисунок 51– Результаты проверки на валидность файла prediction.html

Результаты проверки на валидность кода CSS показаны на рисунке 53. Каквидно, ошибок в коде не обнаружено.

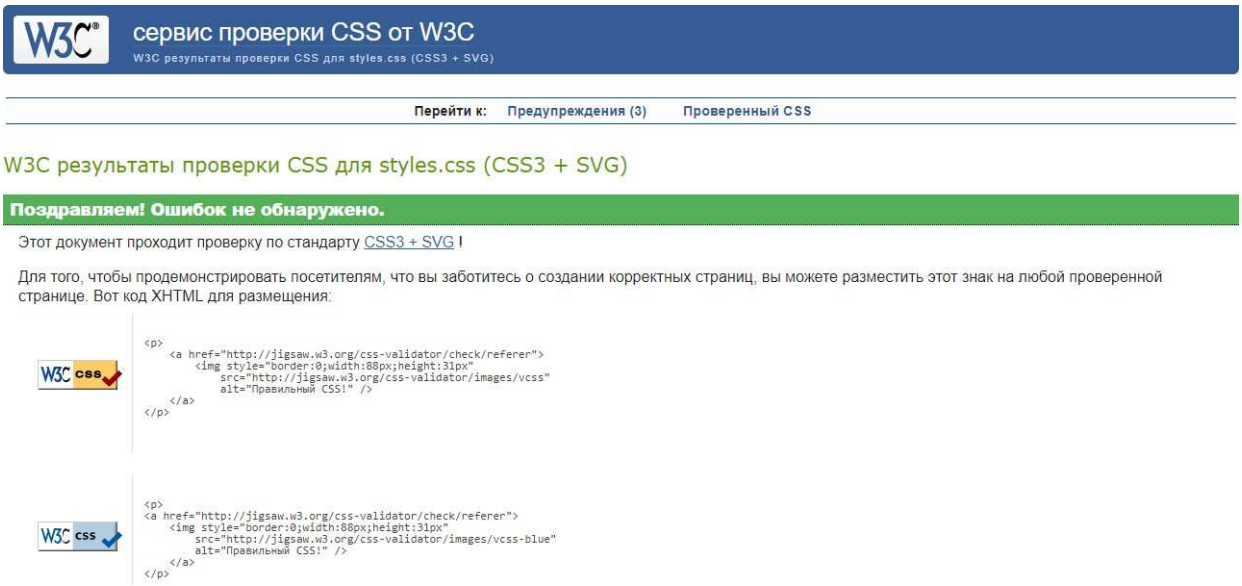

Рисунок 52– Результаты проверки на валидность файла styles.css

3.2.3 Кроссбраузерность

Кроссбраузерность — свойство веб-сайта отображаться и функционировать во всех часто используемых браузерах идентично. Под идентичностью функционирования подразумевается: отсутствие некорректной работы, ошибок в вёрстке и способность отображать материал с одинаковой степенью читабельности. Вследствие постоянного развития веб-технологий, приемлемую кроссбраузерность возможно обеспечить только для новых версий браузеров.

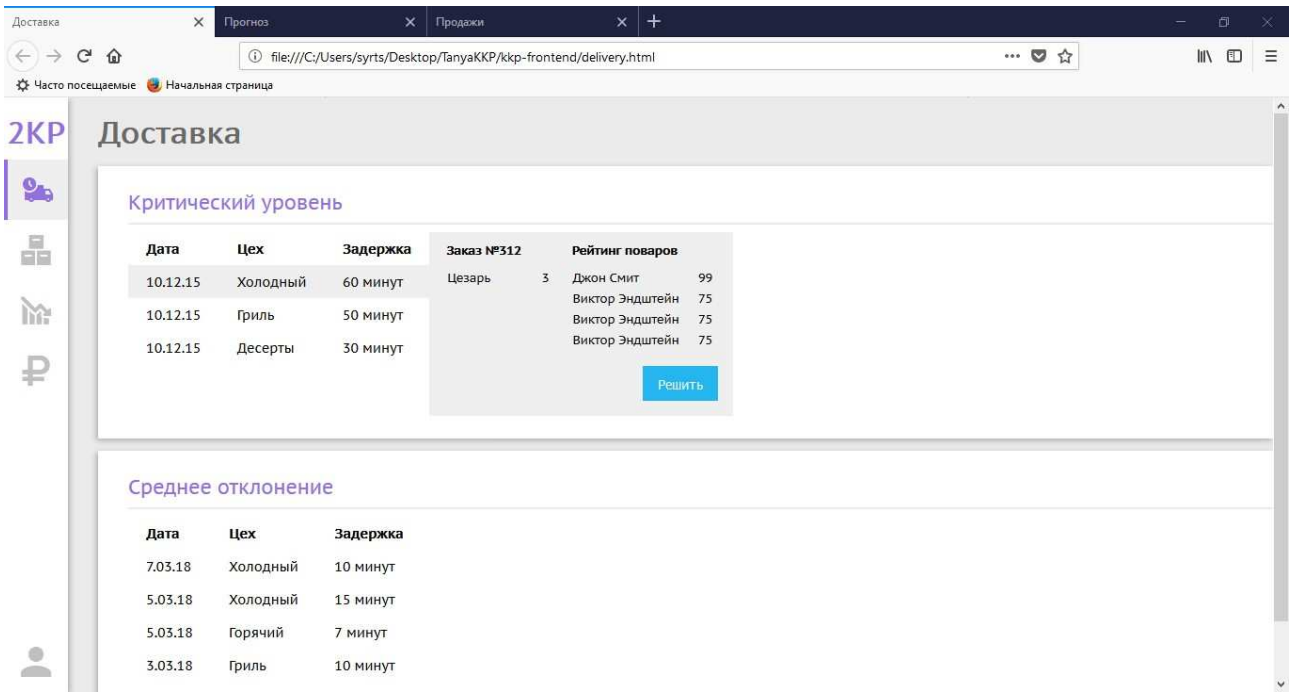

Рисунок 53– Страница доставки в браузере FireFox

Другими словами, кроссбраузерность – это правильная верстка сайта с помощью, которой страницы сайта одинаково отображаются в различных браузерах.

Проблема в том, что разные браузеры в основном соблюдают общие правила и стандарты, но в некоторых случаях бывает, что алгоритмы обработки HTML-кодов и каскадных таблиц CSS могут быть разные, это и производит к различному отображению одного и того же элемента сайта в различных браузерах.

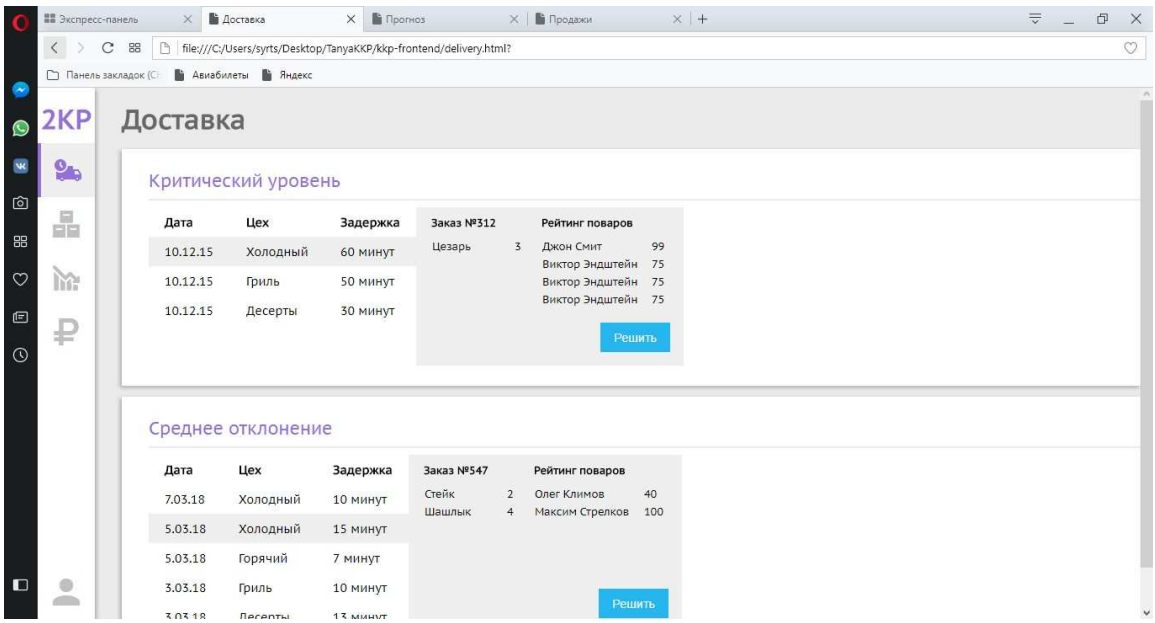

Рисунок 54– Страница доставки в браузере Opera

Для нашего приложения был определен список браузеров, в которых страницыобязательнодолжныотображатьсякорректно:всеэлементыдоступны, кликабельность ссылок и кнопок, при этом кликабельные элементы должны менять вид стрелки указателя. Далее необходимо протестировать работоспособность полей форм. Вид страницы доставки в разных браузерах показан на рисунках54-56.

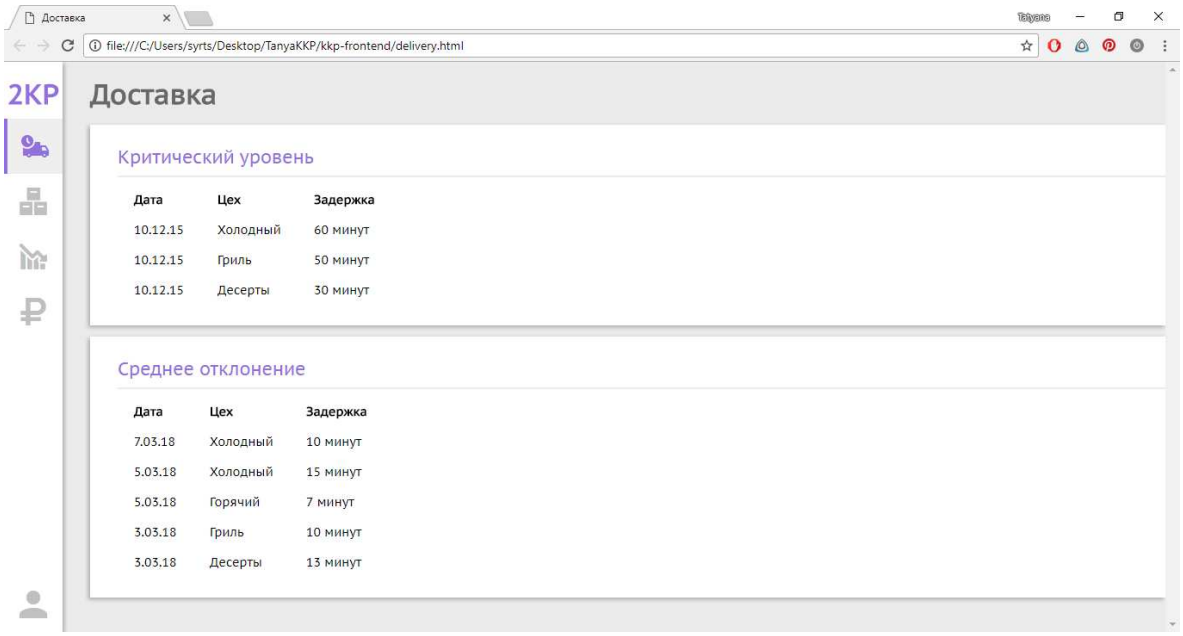

Рисунок 55– Страница доставки в браузере Google Chrome

3.4 Реализация серверной части приложения

3.4.1 Сравнительный обзор технологий

Django

Django[28] написан на Python[29]. Python - один из самых популярных, наиболее изученных языков программирования в мире. Наличие квалифицированного персонала является ключевой проблемой в текущее время. Поэтому распространенность Python является ключевым фактором для команд, надеющихся развиваться в ближайшем будущем.

Также Python как язык программирования гордится своей читабельностью. Нужно уметь понимать код, написанный 6-12 месяцев назад. Философия Python о стиле кодирования описана в PEP 20 - TheZenofPython. Использование этой философии помогает повысить читаемость кода и упростить его поддержку в будущем.

Документация, входящая в состав Django, является примером для подражания. Документация не только подробно описывает использование каждой функции в Django, но также содержит подробные примечания к релизу, включая любые обратно несовместимые изменения, вместе с каждой версией.

Документация Django важна по двум основным причинам. Во-первых, это помогает как новым, так и существующим пользователям фреймворка быстро определять, как использовать данную функцию. Во-вторых, он служит «контрактом» для обратной совместимости в Django; то есть, если функция задокументирована в Django, проект обязуется, что она будет поддерживаться как минимум для двух дополнительных релизов (если только она не была устарелым в примечаниях к релизу). Документация Django полезна как для одноразовых проектов, которые необходимо быстро построить, так и для проектов, которые необходимо развивать и улучшать спустя больше количество новых версий Django.

Django невероятно масштабируем. Фреймворк используется в таких компаниях, как EventBrite, Disqus и Instagram для обработки веб-трафика и использования API мобильных приложений для обслуживания более 500 миллионов пользователей.

Django можно считать лучшей в своем роде коллекцией решений почти всех проблем, общих для разработки веб-сайтов и restfulAPI.

Сравнение Django (Python) и Laravel (PHP)

«Django - фреймворк для перфекционистов с дедлайнами».Django написан на Python и является очень мощным и популярным фреймворком. Он был выпущен в 2005 году и с тех пор имеет множество стабильных релизов. Для начала работы с Django вам не требуется экспертное знание фреймворков, потому что у него очень простая кривая обучения, с понятной документацией и большим количеством бесплатных учебников и электронных книг. Фреймворк имеет открытый исходный код и поддерживается большим сообществом разработчиков.

«Laravel - фреймворк для веб-мастеров». Laravel известен как PHP фреймворк для веб-мастеров. Это бесплатная PHP фреймворк с открытым исходным кодом, где приложения создаются с использованием шаблона MVC. Как утверждают разработчики, фреймворк придерживается философии «с батарейками», где они получают все, что хотят «из коробки». Фреймворк может помочь вам создать любой веб-сайт от простой CMS до социальной сети.

1. Тип

Django - это фреймворк с открытым исходным кодом, который любят разработчики со всего мира. Первоначально он был разработан для создания надежных веб-приложений с использованием Python. Фреймворк оснащена библиотеками, шаблонами и API для обновлений. Любые отсутствующие плагины в Django могут быть легко добавлены через многочисленные плагины приложений. Это относится к принципу D.R.Y или «Не повторяйте себя».

Laravel также представляет собой бесплатный и открытый фреймворк для веб-мастеров с элегантным синтаксисом. Цель Laravel - сделать весь процесс веб-разработки очень простым и быстрым, позволяя разработчикам создавать лучший код. Он также решает тривиальные задачи, такие как аутентификация, маршрутизация, сеансы, кэширование. Laravel больше подходит для разработки больших, мощных приложений. Наилучшие особенности фреймворка включают выразительную систему миграций, инверсию контрольного контейнера и плотно интегрированную поддержку модульного тестирования.

2. Язык программирования

Django очень рекомендуется программистами и следует принципам MVT или ModelViewTemplate подходу, в отличие от Laravel. Согласно ранжированию, в индексе TIOBE, язык программирования Python находится на 4-й позиции.

Laravel следует методу ООП или объектно-ориентированного программирования и MVC или Model-View-Controller. Согласно ранжированию, в индексе TIOBE, язык программирования PHP находится на 9-й позиции.

3. Разработчик

Веб-программисты в газете, LawrenceJournal-World, Адриан Головатый и Саймон Виллисон создали Django в 2003 году и публично опубликовали его в 2005 году по лицензии BSD. Программисты использовали Python для создания приложений.

Тейлор Отуэлл создал фреймворк PHP, и он следует шаблону MVC. В нем есть некоторые соглашения, которые вы можете использовать в своем проекте, в комплекте с некоторыми внешними зависимостями, такими как Sentry для отслеживания ошибок.

4. Кривая обучения

Кривая обучения для Python практически не существует. Так что он прост и довольно просто использовать Django. Предлагая лучшую читабельность кода фреймворк упрощает жизнь начинающего разработчика.

Laravel очень интуитивно понятен и позволит вам освоить современную PHP-разработку с миграциями баз данных, выразительным ORM, пакетами, REST, шаблонами и т. д. Ресурсы для обучения изобилуют, но вы должны справиться с этим, учитывая крутую кривую обучения.

5. Безопасность

Вы развертываете свое приложение в совершенно враждебной среде, где злонамеренные пользователи, боты и хакеры ждут, чтобы использовать ваши дыры в безопасности. Безопасности в Django уделяется большое внимание, и это помогает разработчикам избегать распространенных ошибок, которые обычно встречаются им в разработке веб-приложений. Примерами могут быть SQL-инъекция, межсайтовый скриптинг, подделка запросов на межсайтовый запрос. Безопасность также предоставляется при управлении паролями и учетными записями пользователей.

У Laravel также есть некоторые механизмы для защиты своих пользователей в таких событиях, как атаки на межсайтовый скриптинг (XSS), уязвимость SQL-инъекций, перехват частной информации, небезопасные файлы cookie и т. д. Несмотря на все это, безопасность в Django намного превосходит Laravel.

6. Скорость

Python - очень быстрый язык, поэтому Django, естественно, очень быстрый. Несколько профессионалов в области веб-разработки провели несколько тестов на языках в 2016 году. Результаты представлены на рисунке 57.

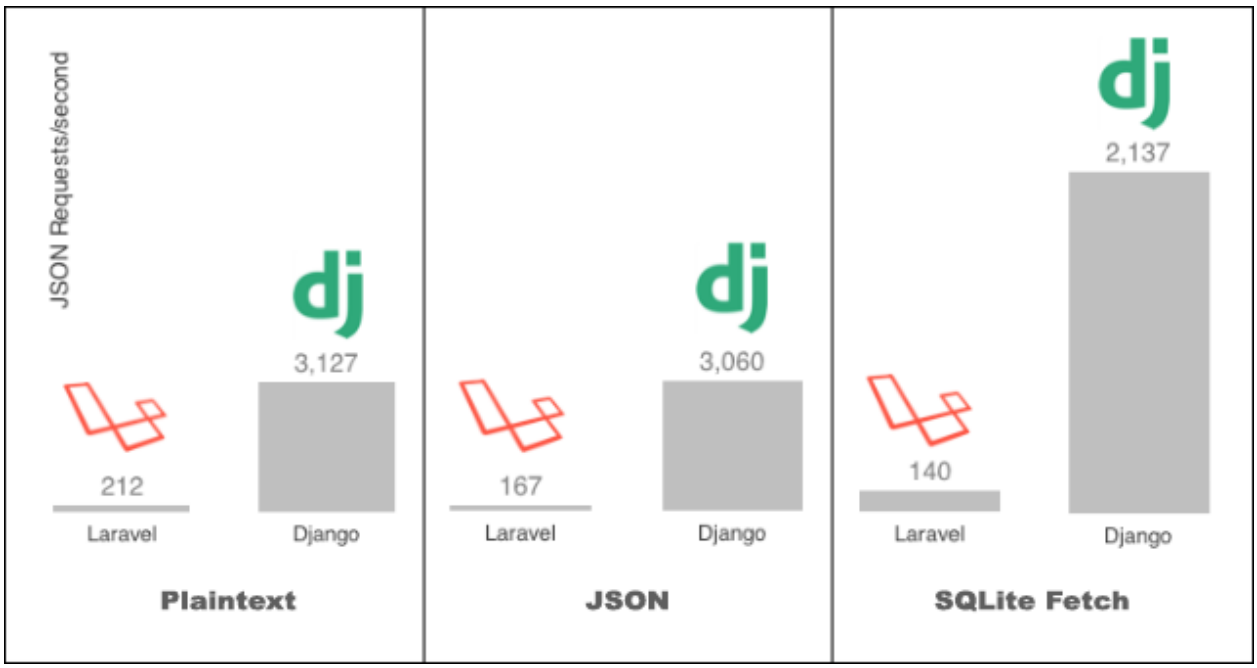

Рисунок 56 – Результаты тестов

Django: тест Plaintext - 3127 запросов в секунду, тест JSON - 3060 запросов в секунду, случайный SQLite Fetch test - 2137 запросов в секунду.

Laravel: тест Plaintext - 212 запросов в секунду, тест JSON - 167 запросов в секунду, случайный SQLiteFetchtest - 140 запросов в секунду.

7. Библиотека

Django имеет действительно сильную библиотеку. Существует множество автономных пакетов, которые предоставляют возможности повторного использования.

В Laravel библиотека также хороша, и вы можете создать с ней любой сайт. Он также обширен, с полной документацией и видео-учебной библиотекой более 1100 видеоуроков.

8. Особенности:

Маршрутизация может быть немного сложной в Django, поскольку у нее нет встроенной структуры для создания API. Вам нужно будет сделать это самостоятельно или использовать фреймворк DjangoREST.

Django поставляется с некоторыми встроенными декораторами, такими как login_required, require_POST или has_permission.

Существует приложение администратора, которое поможет вам автоматически создавать область сайта, где вы можете создавать, просматривать и изменять или удалять записи. Это экономит время при разработке.

Фреймворк предоставляет классы базового представления, подходящие для широкого спектра приложений. Он также имеет собственную систему кэширования, которая поможет вам сохранить динамические страницы, и вы можете избежать их генерации по мере необходимости. Это может быть вывод определенного view, части шаблона или всего сайта.

Фреймворк упрощает предоставление программных механизмов для объявления, рендеринга и проверки форм по отношению к набору правил. Существует поддержка промежуточного программного обеспечения для условных операций GET. Django имеет расширяемую систему аутентификации и динамический административный интерфейс.

Маршрутизация в Laravel довольно проста, и есть простой способ создания API. Laravel может похвастаться красноречивым ORM. Они обеспечивают высокоуровневую абстракцию реляционных баз данных, чтобы помочь разработчикам писать код PHP вместо SQL для создания, чтения, изменения и удаления данных и схем в своей базе данных.

Сайт Laravel может работать в несколько раз быстрее и обеспечивает внутреннюю поддержку. Laravel поддерживает несколько кеш-серверов, включая Redis и Memcached. Он также обеспечивает поддержку SASL и постоянные соединения.

9. Контрибьюторы на GitHub

Django предлагает большое сообщество из 1530 + участников, на 13 февраля 2018 года. Если вы когда-либо сталкивались с проблемой при разработке приложений, кто-нибудь сможет вам помочь.

Laravelсообщество намного меньше по сравнению с Django. С 13 февраля 2018 года было около 435 участников. Однако новые разработчики присоединяются каждый день.

Несмотря на то, что конечный выбор будет зависеть от вашего требования, Django - это основная помощь в устранении проблем веб-разработки, поскольку разработчики могут просто сосредоточиться на написании приложения, не беспокоясь о том, чтобы изобретать колесо.

Цель Django заключалась в том, чтобы быстро вывести разработчиков от концепции проекта до запуска платформы, и он действительно помогает в этом. Он также снимает ваше беспокойство по поводу проблем безопасности, таких как подделка запросов на межсайтовый запрос, SQL-инъекция, межсайтовый скриптинг и clickjacking. Разработчики могут управлять учетными записями пользователей и паролями через надежную систему аутентификации пользователей.

Сравнение PostgreSQL и MySQL

1. Исходный код и лицензия

PostgreSQL разработан PostgreSQLGlobalDevelopmentGroup, разнообразной группой из нескольких компаний и отдельных участников.

Это бесплатное программное обеспечение с открытым исходным кодом. PostgreSQL выпущен под лицензией PostgreSQL, либеральной лицензией с открытым исходным кодом, аналогичной лицензии BSD или MIT.

Проект разработки MySQL сделал исходный код доступным в соответствии с GNUGeneralPublicLicense, а также различными соглашениями о собственности.

Теперь он принадлежит корпорации Oracle и предлагает несколько платных изданий для частного использования.

2. Соответствие ACID (Атомарность, Согласованность, Изолированность, Устойчивость)

PostgreSQL совместим с ACID и обеспечивает выполнение всех требований.

MySQL работает только с ACID при использовании движков InnoDB и NDBClusterStorage.

3. Соответствие SQL

PostgreSQL в значительной степени совместим с SQL. Уровень соответствия для каждой функции четко изложен в Приложении D руководства, и любые отклонения четко описаны в разделе «Ссылка» руководства PostgreSQL.

Отрывок из документации:

PostgreSQL поддерживает большинство основных функций SQL: 2011. Из 179 обязательных функций, необходимых для полного соответствия Core, PostgreSQL соответствует не менее 160. Кроме того, существует длинный список поддерживаемых дополнительных функций.

MySQL частично совместим с некоторыми версиями (например, не поддерживает ограничения CHECK).

Отрывок из документации:

Одной из наших главных целей в этом продукте является продолжение работы над соблюдением стандарта SQL, но без ущерба для скорости или надежности. Мы не боимся добавлять расширения к SQL или поддержку функций, отличных от SQL, если это значительно увеличивает удобство использования MySQLServer для большого сегмента нашей пользовательской базы.

4. Производительность

PostgreSQL широко используется в больших системах, где скорость чтения и записи имеет решающее значение, и данные должны быть проверены. Кроме того, он поддерживает множество оптимизаций производительности, которые доступны только в коммерческих решениях, таких как поддержка

геопространственных данных, параллелизм без блокировок чтения и т. д. (Например, Oracle, SQLServer).

В целом производительность PostgreSQL лучше всего используется в системах, требующих выполнения сложных запросов.

PostgreSQL хорошо работает в OLTP / OLAP-системах, когда требуется высокая скорость чтения / записи и необходим обширный анализ данных.

PostgreSQL также хорошо работает с приложениями BusinessIntelligence, но лучше подходит для приложений хранения данных и анализа данных, требующих быстрой скорости чтения / записи.

MySQL является часто выбираемым для веб-проектов, которым нужна база данных для простых транзакций.

MySQL хорошо работает в OLAP / OLTP-системах, когда требуются только скорости чтения.

MySQL + InnoDB обеспечивает очень хорошие скорости чтения / записи для сценариев OLTP. В целом, MySQL хорошо работает с высокими сценариями параллелизма.

MySQL является надежным и хорошо работает с приложениями BusinessIntelligence, поскольку приложения бизнес-аналитики обычно имеют нагрузку на чтение.

5. Безопасность

PostgreSQL имеет роли и наследование ролей для установки и поддержки разрешений. PostgreSQL имеет встроенную поддержку SSL для соединений для шифрования сообщений клиент / сервер. Он также имеет безопасность уровня строки.

В дополнение к этому, PostgreSQL поставляется со встроенным расширением SE-PostgreSQL, который предоставляет дополнительные элементы управления доступом на основе политики безопасности SELinux.

MySQL реализует безопасность на основе списков контроля доступа (ACL) для всех подключений, запросов и других операций, которые

пользователь может попытаться выполнить. Существует также поддержка SSL-зашифрованных соединений между клиентами MySQL и серверами.

3.4.2 Реализация

Для работы алгоритмов, на которых основана работа всех компонентов системы, реализованы следующие методы:

1. find_order_problems()

   Входные данные: нет

   а. Проходим по каждому заказу среди необработанных, вычисляем норму заказа и самый долгий цех и создаем проблему.

   б. Если время выполнения превышает критический порог, то проблема переходит в массив критических.

   в. Если превышение нормы низкое и есть 2 других заказа с данным цехом и низким превышением, то помечаем найденные как решенные, а текущему заказу присваиваем высокую важность.

   Выходные данные: обработанные заказы, сохраненные в базу данных

2. find_remainder_raws_problems()

   Входныеданные: нет

   а. Проходим по каждому остатку сырья среди необработанных.

   б. Если превышение нормы высокое: если норма превышена в отрицательную сторону - советуем увеличить количество закупаемого сырья, иначе - уменьшить.

   Выходные данные: обработанные остатки сырья, сохраненные в базу данных

3. increase_rating_of_good_cooks()

   Входныеданные: нет

   а. Находим проблемы времени выполнения высокой важности за последние 2 недели.

б. Среди этих проблем находим id всех цехов и заказов, которые задействованы в проблемах.

в. С помощью этих данных находим поваров, участвовавших в проблемных заказах.

г. Проходим по всем работникам, не участвовавшим в проблемных заказах и увеличиваем им рейтинг на 20 очков.

Выходные данные: нет

4. get_raw_forecast(raw, days_count=30)
   Входные данные: сырье, количество дней.
   а. Вызываем функцию для получения списка потраченного сырья с привязкой к дате. Если списка нет, то возвращаем None.
   б. Иначе вызываем функцию прогнозирования и передаем ей список.

Выходные данные: прогноз трат на выбранное количество дней

5. collect_raw_spent_statistic(raw, days_count=30)
   Входные данные: сырье, количество дней
   а. Находим список продаж блюда с заданным сырьем и за заданное количество дней. Если продаж нет - возвращаем None.
   б. Проходим по каждой дате из заданного промежутка времени.
   в. Суммируем потраченное сырье за день и сохраняем вместе с датой в массив.

   Выходные данные: список трат сырья за заданное количество дней.

6. get_dish_forecast(dish, days_count=30)

   Входные данные: блюдо, количество дней
   а. Вызываем функцию для получения списка продаж. Если списка нет, то возвращаем None.
   б. Иначе вызываем функцию для прогноза и передаем ей список.

Выходные данные: прогноз продаж на выбранное количество дней.

7. collect_dish_sales_statistic(dish, days_count=30)

   Входные данные: блюдо, количество дней

а. Находим список продаж для заданного блюда за заданное количество дней;

б. Компонуем в массив дату и количество продаж для каждого дня.

Выходные данные: список продаж для заданного блюда на выбранное количество дней.

8. make_forecast(numbers)

Входные данные: числа для прогноза.

а. С помощью метода определения функции по заданным точкам вычисляем функцию прямой, равноудаленной от заданных чисел.

б. Находим на графике следующие точки после заданных, в количестве равному количеству переданных чисел.

Выходные данные: числа прогноза.

9. collect_predictions(statistic)

Входные данные: массив из пар дата, число

а. Вызываем функцию прогноза и получаем прогноз для заданного списка.

б. Компонуем список и прогноз в один массив.

Выходные данные: коллекция списка вместе с прогнозом

10. create_raw_expiration_notifications()

Входныеданные: нет

а. Вычисляем дату для числа, которое было 2 недели назад, вычисляем начало и конец дня.

б. Находим сырье, у которого скоро истекает срок годности.

в. Проходим по каждому найденному сырью и создаем уведомление.

Выходные данные: уведомления о сырье с истекающим сроком годности

11. create_raw_depletion_notifications()

Входныеданные: нет

а. Проходим по каждому объекту сырья. Вызываем функцию для получения количества дней до истощения запасов.

б. Если количество дней до истощения запасов равно заданной константе, то создаем уведомление.

Выходные данные: уведомления о заканчивающихся запасах сырья

12. get_raw_remainder(raw)

Входные данные: сырье

а. Суммируем остатки заданного сырья из всех поставок, чтобы получить общий остаток.

Выходные данные: общий остаток сырья на складе

13. get_days_to_raw_stocks_depletion(raw)

Входныеданные: сырье

а. Вызываем функцию для получения общего остатка сырья.

б. Вызываем функцию для получения прогноза трат сырья.

в. Проходим по каждому дню в прогнозе и суммируем прогнозируемые траты сырья, если сумма прогнозируемых трат превысила остаток, то возвращаем количество дней.

Выходные данные: прогнозируемое количество дней до истощения запасов

14. get_cost_price(dish)

Входные данные: блюдо

а. Находим список продаж заданного блюда за месяц. Вычисляем среднее количество продаж в день.

б. Получаем Операционные затраты из константы.

в. Делим операционные затраты на количество дней в месяце, затем делим на среднее число продаж за день и прибавляем стоимость сырья для блюда.

Выходные данные: себестоимость блюда

15. update_cost_prices_and_create_notification()

Входныеданные: нет

а. Проходим по каждому объекту блюда. Вызываем функцию для получения себестоимости блюда.

б. Вычисляем на сколько процентов выросла себестоимость блюда: вычитаем из текущей цены старую себестоимость блюда из базы данных и делим на новую вычисленную себестоимость.

в. Если себестоимость выросла, то создаем уведомление.

Выходные данные: уведомления о блюдах, для которых выросла себестоимость

16.handle_bad_selling_dishes()

Входныеданные: нет

а. Проходим по каждому объекту блюда. Находим статистику продаж за неделю и сохраняем в таблицу.

б. Сортируем таблицу по количеству продаж и берем самые плохо продающиеся блюда.

в. Проходим по каждому из них. Вызываем функцию для определения как расходуется сырье этого блюда.

г. Создаем уведомление. Если сырье расходуется плохо, то рекомендуем снизить наценку до себестоимости, иначе снизить наценку в 2 раза.

Выходные данные: уведомления с рекомендациями для плохо продающихся блюд

17.is_raws_bad_selling(dish)

Входныеданные: блюдо

а. Проходим по каждому объекту сырья. Находим список продаж блюда для сырья.

б. Суммируем траты сырья и сохраняем в таблицу.

в. Сортируем таблицу и находим сырье которое плохо продается.

г. Проходим по каждому объекту сырья в блюде. Если ни одно сырье не является плохо продающимся, то возвращаем False, иначе True.

Выходные данные: True или False

18. handle_long_bad_selling_dishes()

Входныеданные: нет

а. Находим в истории цены, созданные 2 недели назад и проходим по каждой из них.

б. Если цена равна себестоимости блюда, то вызываем функцию прогноза.

в. Если прогноз отрицательный, то создаем уведомление, что блюдо 2 недели плохо продается и рекомендуем отказаться от него.

г. Иначе рекомендуем повысить цену.

Выходные данные: уведомления с рекомендациями для плохо продающихся блюд долгое время

19. handle_predicted_bad_sales()

Входныеданные: нет

а. Проходим по каждому объекту блюда. Находим сумму продаж за последний месяц и сохраняем в таблицу.

б. Сортируем таблицу и проходим по половине плохо продающихся блюд.

в. Вызываем функцию прогноза продаж. Если прогнозируется снижение продаж: вычисляем коэффициент, путем вычитания из единицы меньшего числа: первого дня продаж или последнего дня прогноза.

г. Вычисляем скидку, путем вычитания из цены блюда себестоимости и умножения на коэффициент.

д. Создаем уведомление о прогнозируемом снижении продаж и рекомендуем сделать скидку.

Выходные данные: уведомления с рекомендациями сделать скидку для блюд с прогнозируемым снижением продаж

Ниже представлены методы, которые обрабатывают входящие запросы на сервер и выдают соответствующие запросу данные в виде контекстной переменной:

1. index(request)

   Входные данные: данные POST запроса

   а. Находим все нерешенные проблемы времени выполнения.

   Выходные данные: страница с проблемами времени выполнения.

2. remainder_stock(request)

   Входные данные: данные POST запроса

   а. Находим все остатки сырья.

   б. Вычисляем максимальное значение остатка, для построения графика.

   Выходные данные: страница с остатками сырья

3. sales_history(request)

   Входные данные: данные POST запроса

   в. Проходим по каждому блюду.

   г. Получаем первую дату для блюд из запроса, если дата не задана, то присваиваем ей текущее число минус 30 дней.

   д. Получаем вторую дату для блюд из запроса, если дата не задана, то присваиваем ей текущее число.

   е. Находим сумму продаж в заданном временном промежутке и сохраняем в словарь.

   ж. Получаем направление сортировки блюд из запроса, если сортировка не задана, то сортируем по убыванию.

   з. Проходим по каждому сырью

   и. Получаем первую дату для сырья из запроса, если дата не задана, то присваиваем ей текущее число минус 30 дней.

   к. Получаем вторую дату для сырья из запроса, если дата не задана, то присваиваем ей текущее число.

л. Находим статистику продаж в заданном временном промежутке где задействовано сырье.

м. Находим сумму трат сырья и сохраняем в словарь.

н. Получаем направление сортировки сырья из запроса, если сортировка не задана, то сортируем по убыванию.

Выходные данные: страница с историей продаж блюд и трат сырья

4. sales_and_forecast(request, model, pk)

Входные данные: данные POST запроса, модель, первичный ключ.

а. Получаем объект блюда или сырья по первичному ключу в зависимости от значения переменной model.

б. Вызываем функцию прогноза на 30 дней.

Выходные данные: страница с подробной историей продаж и прогнозом для блюда/сырья.

5. solve_order_problem(request, pk)

Входные данные: данные POST запроса, первичный ключ

а. Получаем проблему времени выполнения по первичному ключу.

б. Получаем список работников из запроса и проходим по каждому работнику.

в. Отнимаем у работника 10 очков рейтинга если проблема низкой важности, иначе 20 очков.

Выходные данные: страница с проблемами времени выполнения и остатками сырья.

Также для каждого объекта модели создан набор индивидуальных методов, которые участвуют в работе основных алгоритмов:

1. update_price_history(sender, instance=None, created=False, **kwargs)

Входныеданные: объектласса

а. Вызывается автоматически при изменении цены.

б. Создаем объект истории цен с новой ценой блюда.

Выходные данные: объект истории цен, сохраненный в базу данных

2. subtract_raw_for_order_item(sender, instance=None, created=False, **kwargs)

   Входныеданные: объектласса

   а. Вызывается автоматически при создании позиции заказа. Проходим по каждому ингредиенту в блюде.

   б. Вычисляем нужное количество сырья путем умножения количества сырья в ингредиенте на количество блюд.

   в. Вычитаем сырье из самой старой поставки, если не хватило, то вычитаем из следующей.

Выходные данные: при создании позиции заказа из запасов вычтено нужное количество сырья

class RawStock

Данный класс отражает модель RawStock.

1. is_expired(self)

   Входные данные: объект класса

   а. Сравниваем текущую дату с датой окончания срока годности сырья. Возвращаем True если текущая дата больше, иначе False.

   Выходные данные: True/False

2. is_soon_expired(self)

   Входные данные: объект класса

   а. Прибавляем к текущей дате константу количества дней до уведомления о скором окончании срока годности сырья и сравниваем с датой окончания срока годности сырья.

   б. Возвращаем True если даты равны, иначе False.

   Выходныеданные: True/False

classDish

Данный класс отражает модель Dish.

1. get_cooking_time_seconds(self)

   Входные данные: объект класса

   a. Умножаем количество минут во времени готовки блюда на 60 и прибавляем количество секунд.

   Выходные данные: время готовки блюда в секундах

2. get_raws_cost(self)

   Входные данные: объект класса

   а. Проходим по каждому ингредиенту в блюде.

   б. Умножаем цену за единицу сырья на количество сырья в ингредиенте и прибавляем к общей сумме.

   Выходные данные: стоимость сырья в блюде

class Order

Данный класс отражает модель Order.

1. get_lead_time_seconds(self)

   Входные данные: объект класса

   а. Вычитаем из времени конца время начала готовки.

   б. Вызываем метод для получения секунд.

   Выходные данные: время готовки заказа в секундах

2. get_workshop_with_longest_cooking(self)

   Входные данные: объект класса

   а. Получаем поваров, игнорируя работников, не привязанных к цеху.

   б. Подсчитываем количество цехов и поваров.

   в. Подсчитываем количество цехов для блюд.

   г. Проходимся по каждому цеху чтобы вычислить самый долгий.

   д. Вычисляем минимальное количество поваров для блюда.

   е. Вычисляем время готовки для блюд.

   ж. Находим максимальное время.

   з. Добавляем в словарь цех и его максимальное время.

Выходные данные: цех, в котором дольше всего готовятся блюда заказа

3. get_max_workshop_cooking_sec(self)

   Входные данные: объект класса

   а. Вызываем функцию для получения самого долгого цеха и берем время.

   Выходные данные: время готовки в самом долгом цеху в секундах

4. is_low_norm_excess(self)

   Входные данные: объект класса

   а. Сравниваем превышение нормы заказа с константами.

   б. Если превышение больше либо равно константы низкого превышения и больше, либо равно константы высокого превышения, то возвращаем True, иначе False.

   Выходные данные: True/False

5. is_high_norm_excess(self)

   Входные данные: объект класса

   а. Сравниваем превышение нормы заказа с константой.

   б. Если превышение больше либо равно константы высокого превышения, то возвращаем True, иначе False.

   Выходные данные: True/False

6. calculate_and_set_norm_excess(self)

   Входные данные: объект класса

   а. Вызываем функцию для получения самого долгого цеха и берем время.

   б. Вызываем функцию для получения время готовки заказа, делим время готовки на норму и сохраняем в переменную превышения нормы.

Выходные данные: вычисленное и сохраненное превышение нормы в заказ

class RemainderRaws

Данный класс отражает модель RemainderRaws

1. get_norm_excess(self)

   Входные данные: объект класса

   a. Делим количество сырья на константу нормы

   Выходные данные: превышение нормы

2. is_high_norm_excess(self)

   Входные данные: объект класса

   а. Сравниваем превышение нормы заказа с константой.

   б. Если превышение больше либо равно константы высокого превышения, то возвращаем True, иначе False.

   Выходныеданные: True/False

staff/views.py

shifts_table(request)

Входныеданные: данныеPOSTзапроса

а. Создаем массив из дат на месяц вперед. С его помощью создаем массив с днями недели для соответствующих дат.

б. Если получен POST запрос. Проходим по каждому работнику.

в. Получаем из запроса список со сменами для работника. Проходим по сменам и сохраняем в словарь.

г. Проходим по каждой дате из массива дат.

д. Если смена для такой даты существует, но в словаре смен нет смены с такой датой, то удаляем смену.

е. Если смена не существует и дата присутствует в словаре смен, составленномиз запроса:

ж. Если количество смен работника на текущую неделю равно 5, то добавляем в массив ошибок сообщение о том, что у работника не может быть более 5 смен в неделю.

з. Если в предыдущие 2 дня была ночная смена, то добавляем в массив ошибок сообщение о том, что после ночной смены должно быть 2 выходных.
и. Иначе создаем смену.
к. Потом проходим по всем работникам. Проходим по датам в массиве дат и добавляем в массив смен информацию о сменах работника.

Выходные данные: страница с расписанием смен

4. Демонстрация функциональных возможностей программной системы

4.1 Страница «Доставка»

Для перехода на страницу доставки пользователю нужно нажать в панели навигации на кнопку «Доставка» (рисунок 58).

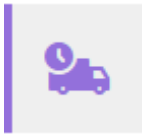

Рисунок 57 – Кнопка «Доставка»

На странице появляется два блока: «Критический уровень» и «Среднее отклонение». В первом блоке располагается таблица с информацией о заказах с критическим отклонением от нормы (рисунок 59): дата заказа, цех, в котором была выявлена задержка и время задержки в минутах.

Критический уровень

Дата	Цех	Задержка
10.12.15	Холодный	60 минут
10.12.15	Гриль	50 минут
10.12.15	Десерты	30 минут

Рисунок 58 – Таблица с критическими проблемами

Пользователь может просмотреть дополнительную информацию по проблемному заказу кликнув по строке таблицы. Блок дополнительной

информации содержит в себе номер заказа, полный состав заказа, список персонала, причастного к приготовлению и их рейтинг (Рисунок 60).

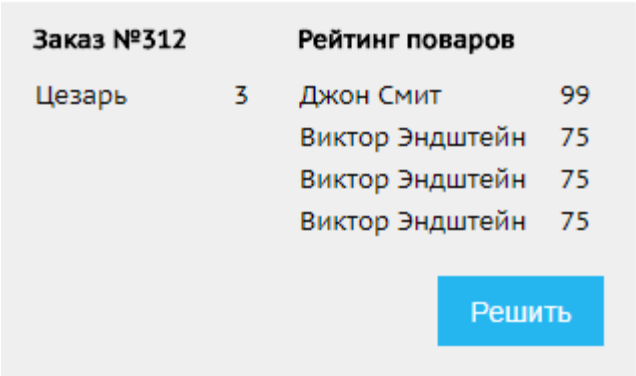

Рисунок 59– Таблица с дополнительной информацией заказа

Для того, чтобы решить проблему, пользователю нужно нажать на кнопку «Решить» в блоке с дополнительной информацией. После нажатия открывается окно (Рисунок 61), где пользователю предлагается выбрать персонал, который может быть виновен в задержке приготовления блюда. Для критических проблем у персонала отнимается 20 очков рейтинга, для проблем со средним отклонением – 10 очков. После нажатия на кнопку «отправить», проблема помечается как решенная и исчезает из таблицы с проблемами.

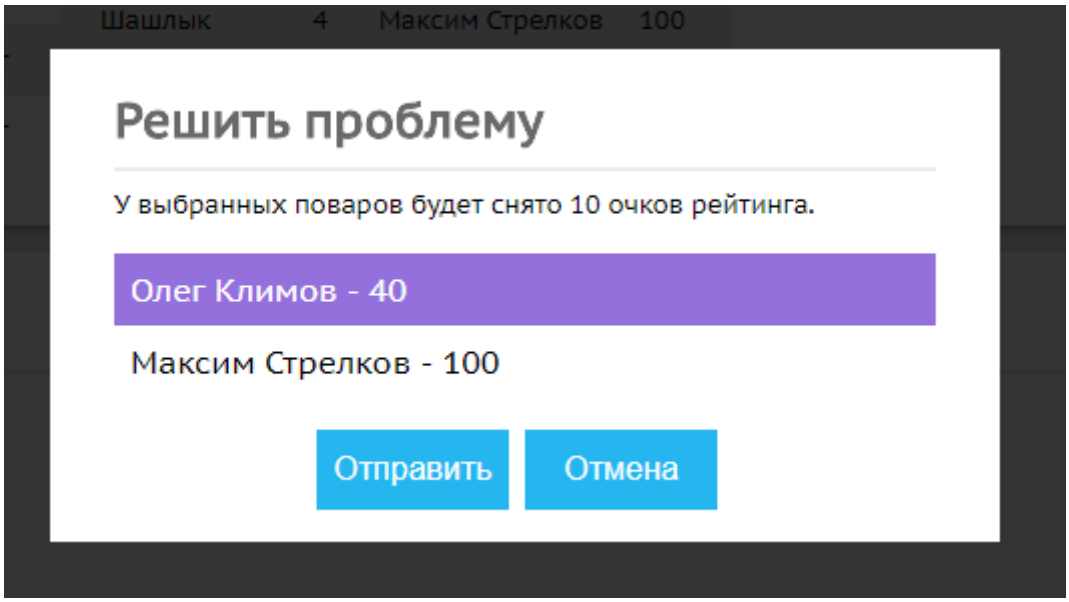

Рисунок 60 – Окно «Решить проблему»

4.2 Страница «Остатки»

Для перехода на страницу остатков сырья пользователю нужно нажать на кнопку «Остатки» на панели навигации (Рисунок 62).

Рисунок 61– Кнопка «Остатки»

Сразу после перехода на страницу отображается диаграмма, показывающая текущие остатки сырья на складе. Диаграмма «Остатки сырья на складе» показана на рисунке 63.

Рисунок 62 – Диаграмма «Остатки сырья на складе»

Один элемент диаграммы содержит в себе информацию о реальных остатках на складе и о нормах остатков, которые задает сам пользователь. Отдельный элемент диаграммы показан на рисунке 64.

Рисунок 63 – Элемент диаграммы «Остатки сырья на складе»

4.3 Страница «График смен»

Для создания и редактирования графика работы персонала пользователю нужно перейти на страницу «График смен» нажав на кнопку «График» на панели навигации (Рисунок 65).

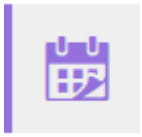

Рисунок 64 – Кнопка «График смен»

Далее пользователю нужно выбрать категорию персонала, для которого будет составляться или редактироваться график. Выпадающий список для выбора категории показан на рисунке 66.

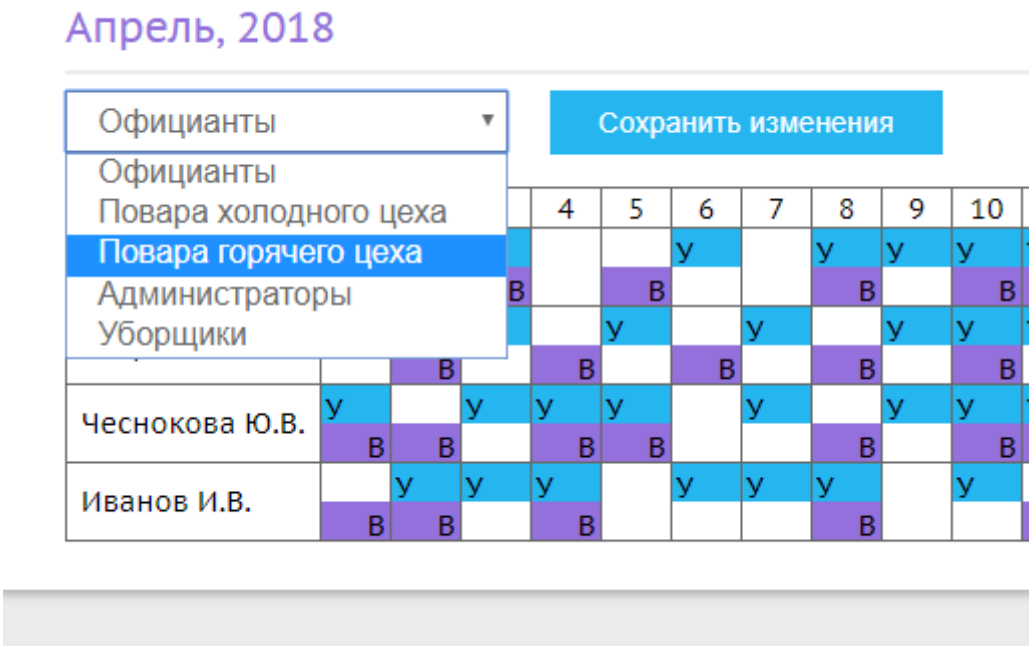

Рисунок 65 – Выбор категории персонала

Раздел создания и редактирования графика работы персонала показан на рисунке 67. Для того, чтобы поставить смену определенному работнику, пользователю нужно кликнуть левой кнопкой мыши по ячейке таблицы сопоставив ФИО нужного сотрудника и дату. Чтобы назначить утреннюю смену – кликнуть по верхней части ячейки, вечернюю – по нижней части ячейки.

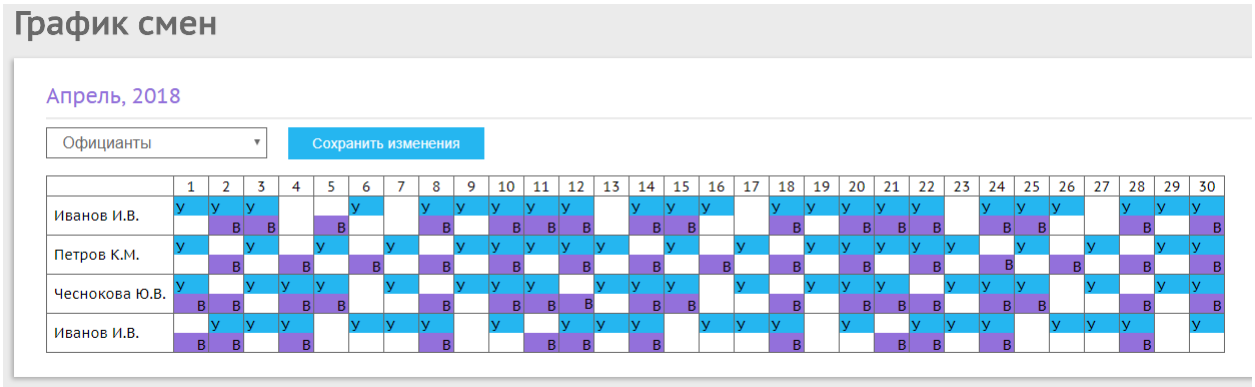

Рисунок 66– Таблица редактирования графика смен

После того, как пользователь закончит составлять(редактировать) график, нужно нажать на кнопку «Сохранить изменения» для сохранения всех изменений.

4.4 Страница «Продажи»

Пользователь нажимает на кнопку «История продаж» (Рисунок 68) и переходит на страницу «Продажи», на которой располагаются два блока: «Товары» и «Сырье» (Рисунок 69).

Рисунок 67– Кнопка «История продаж»

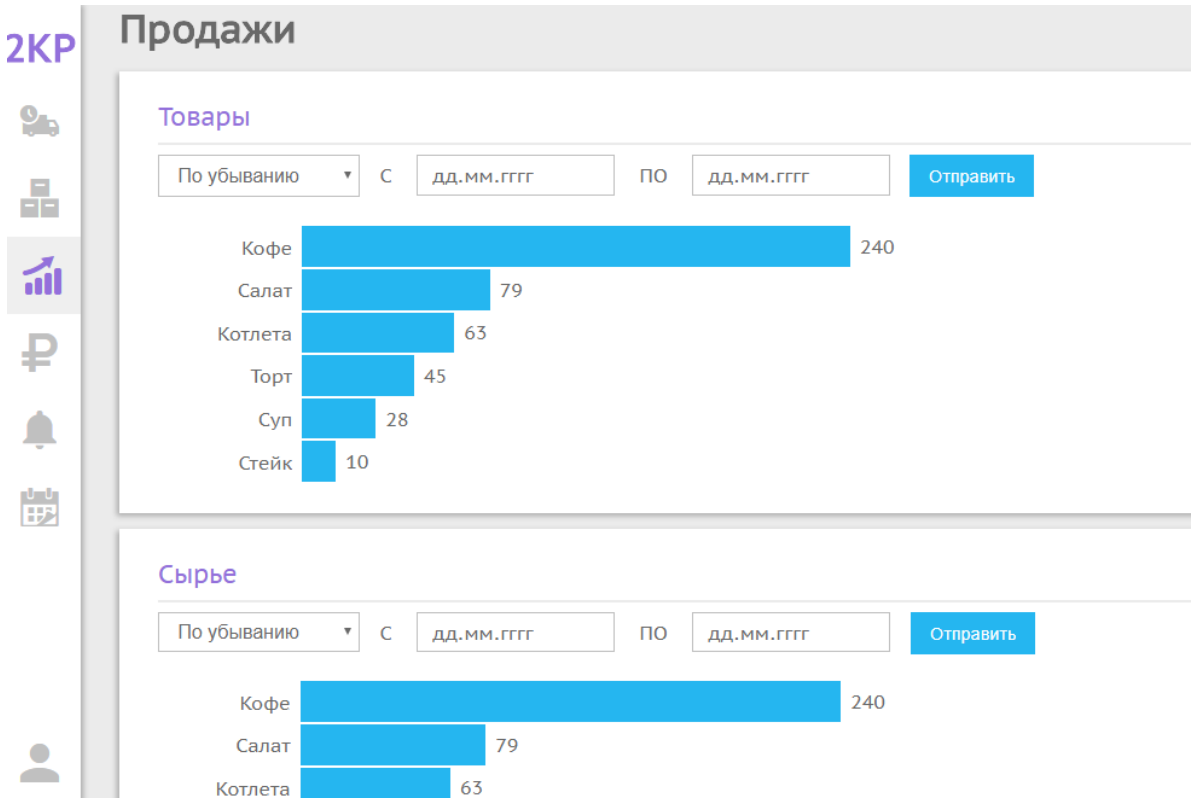

Рисунок 68– Блоки «Товары» и «Сырье»

Два этих блока несут информацию о продажах товара или расходах сырья в виде диаграммы, которая отображает сумму продаж или расход в выбранный период времени. Для более детального анализа есть специальные поля для сортировки, в которых, при нажатии на дату, можно выбрать период времени для товаров или сырья (Рисунок70).

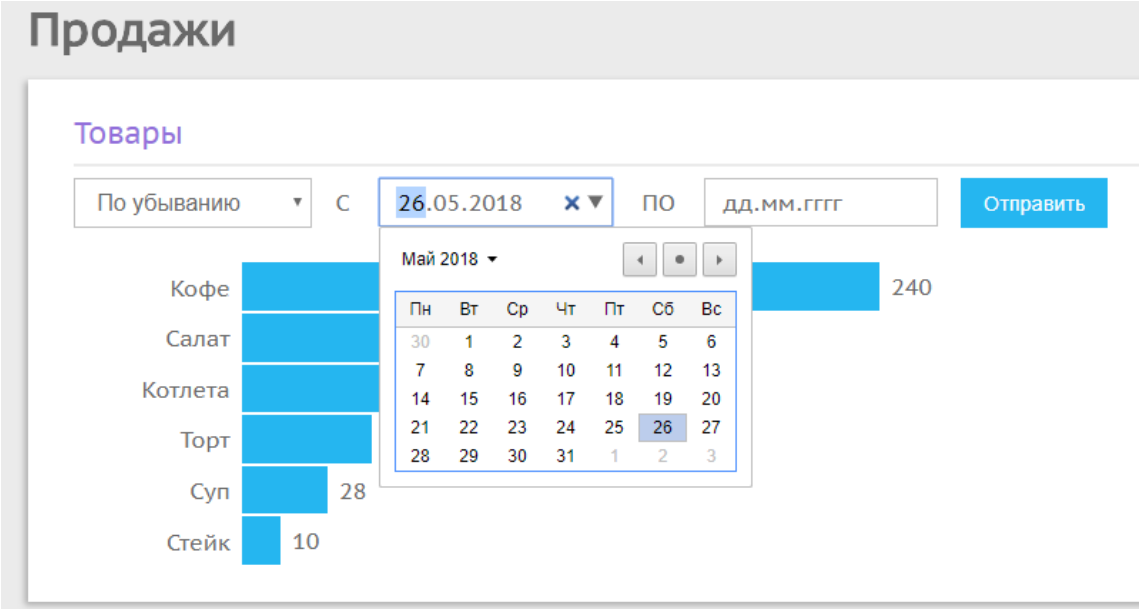

Рисунок 69 – Выбор периода в блоке «Товары»

Также, при нажатии на поле выбора сортировки (по умолчанию – «По убыванию»). На выбор пользователя будет два способа сортировки: «По убыванию» и «По возрастанию».

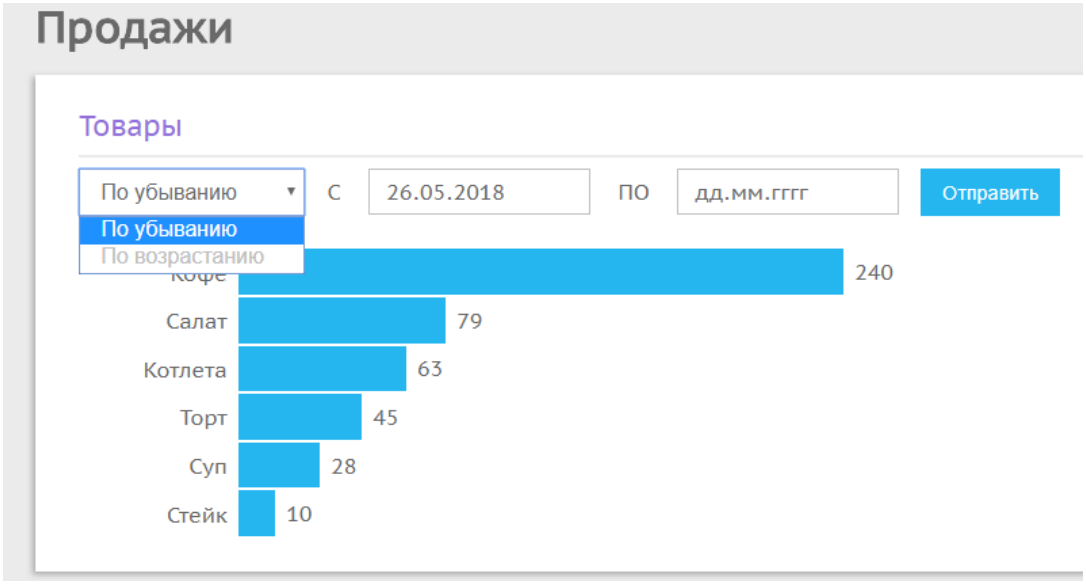

Рисунок 70 – Выбор необходимой сортировки

При выборе режима сортировки «По возрастанию» пользователю необходимо нажать на кнопку «Отправить» для получения необходимого результата (Рисунок 71).

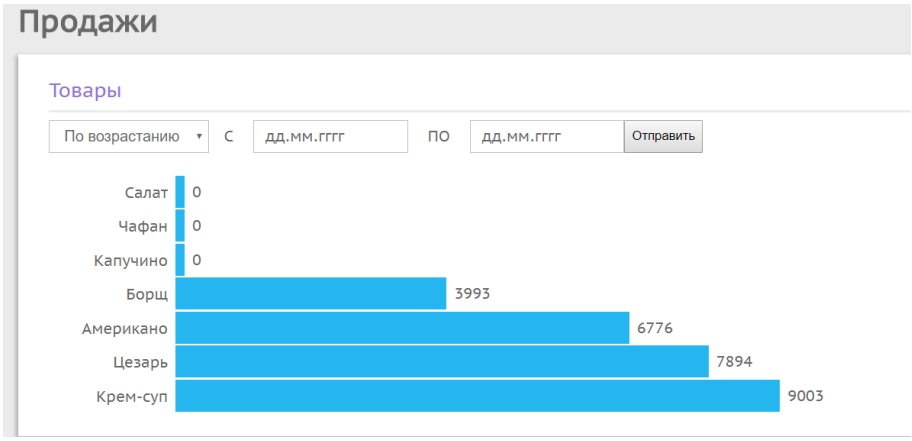

Рисунок 71. Сортировка «По возрастанию»

Далее, при выборе позиции товара на диаграмме истории продаж откроется страница прогноза продаж (Рисунок 73) на срок до 30 дней (колонки синего цвета). Для большей наглядности, на странице прогноза продаж приведены данные с истории продаж за последние 30 дней (колонки фиолетового цвета). По графику мы видим, что прогноз выстраивается таким

образом, чтобы сократить использование товара, для сокращения ненужных трат.

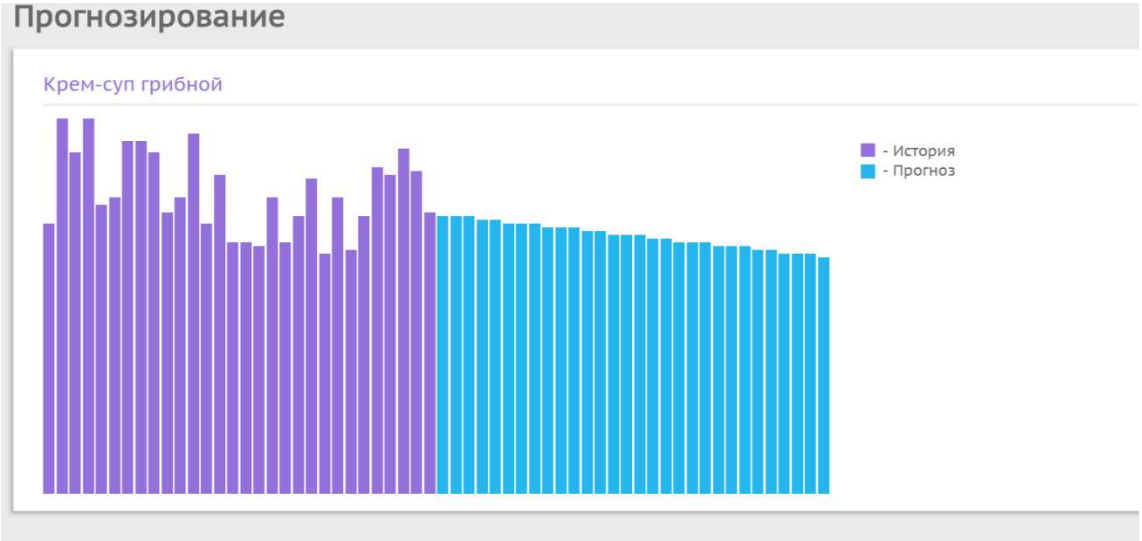

Рисунок 72 –Страница прогноза

При наведении курсором мыши на определенный столбец, выводится точная информация об истории потребления товара за определенный день, если это колонка фиолетового цвета(Рисунок 74). Если же наводиться курсор мыши на колонка синего цвета, будет выводится информация о прогнозе продаж на определнное число (Рисунок 75).

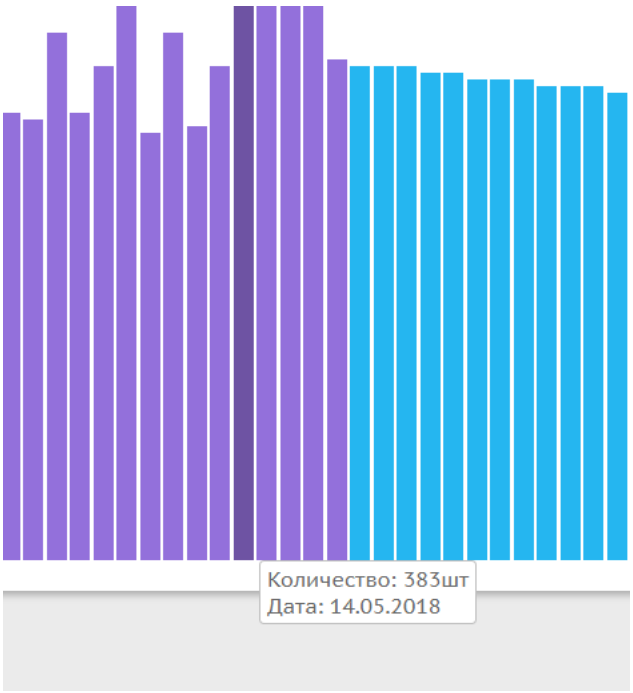

Рисунок 73 –Точная информация о продажах товара за определенный день

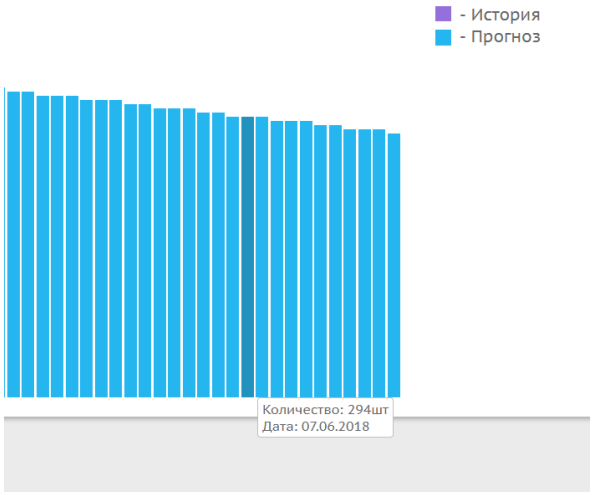

Рисунок 74 – Прогноз на определённый день

4.5 Страница «Уведомления»

При нажатии на кнопку «Уведомления» (Рисунок 76) клиент переходит на страницу со схожим названием, где расположено два блока: блоки «Склад» и «Цены» (Рисунок 77).

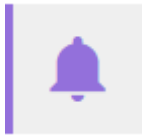

Рисунок 75 – Кнопка «Уведомления»

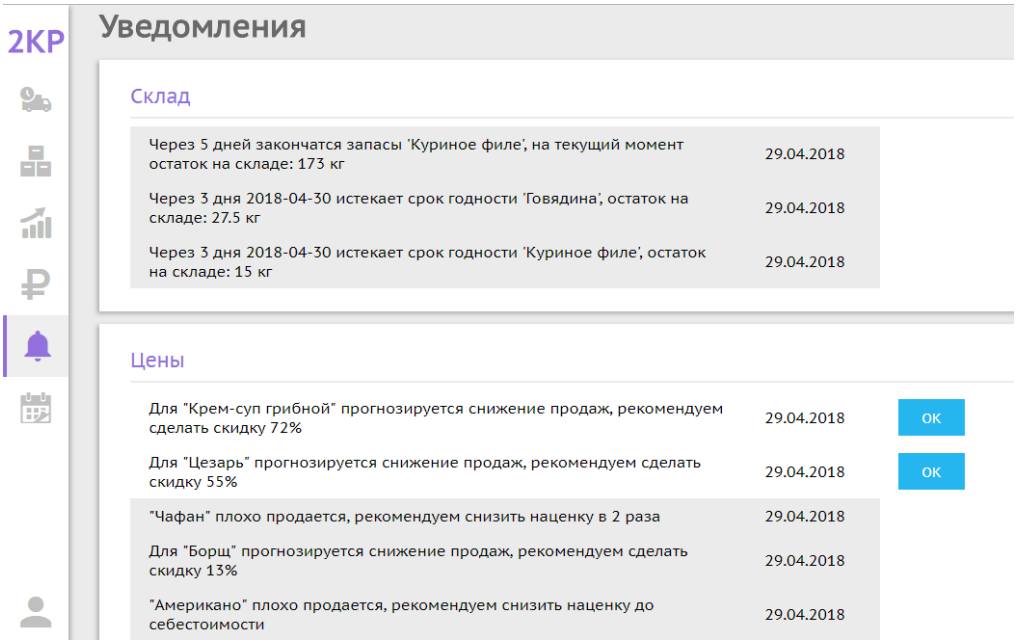

Рисунок 76 –Страница уведомлений

На блоке «Склад» хранится информация о предстоящих изменениях срока годности товаров, хранящихся на складе.

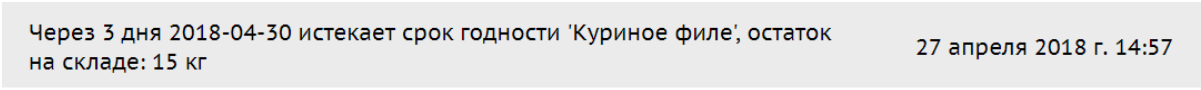

Рисунок 77 –Уведомление об истечении срока годности

Также там хранится информация о предстоящем дефиците товара.

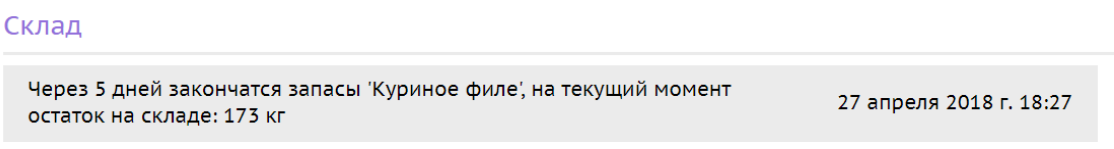

Рисунок 78–Уведомление о предстоящем дефиците товара

Блок «Цены» несет информацию обо всех прогнозируемых скидках, сокращениях цен и созданий возможных акций.

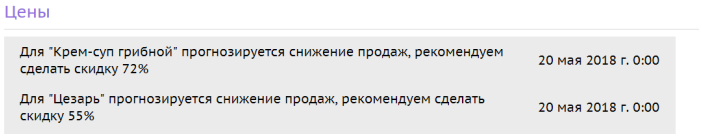

Рисунок 79 –Рекомендации о скидках на товар

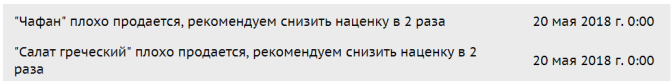

Рисунок 80 –Рекомендации о наценках

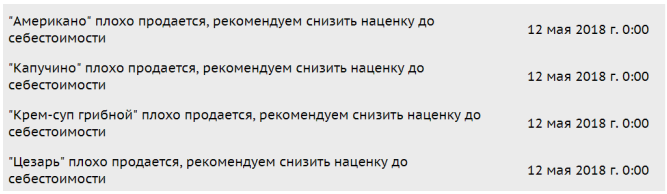

Рисунок 81–Рекомендации о снижении наценки до себестоимости

ЗАКЛЮЧЕНИЕ

В рамках бакалаврской работы была успешно реализована программная система по анализу и прогнозирования показателей эффективности предприятия общественного питания.

Система включила в себя следующие компоненты:

1. анализ времени выполнения заказа;
2. определение динамики потребления сырья;
3. анализ истории продаж и построение прогноза;
4. анализ остатка сырья;
5. учет срока годности сырья;
6. составление графика работы персонала;
7. анализ себестоимости сырья.

Дальнейшее развитие системы будет зависеть от результатов внедрения в реально существующее предприятие. От интеграции в работу реального предприятия зависят настраиваемые параметры системы. В процессе использования приложения планируется уточнять эти параметры, тем самым повышая экспертность системы. Также планируется доработать интерфейсы интеграции приложения с различными учетными системами.

СПИСОК СОКРАЩЕНИЙ

ПО – программное обеспечение

CRM – customerrelationshipmanagement (перевод – системауправлениявзаимоотношения с клиентами)

POS-терминал – pointofsaleтерминал (перевод – точка продажи)

BIO – biometric (перевод – биометрический)

XML – extensiblemarkuplanguage (перевод – расширяемыйязыкразметки)

JSON – javascriptobjectnotation (перевод – текстовый формат обмена данными, основанный на JavaScript)

HTML – hypertextmarkuplanguage, version 5 (перевод – язык гипертекстовой разметки, пятойверсии)

CSS3 – cascadingstylesheets, version 3 (перевод – каскадныетаблицыстилей, третьейверсии)

SCSS – «диалект» языка SASS

JS – javascript

W3C – world wide web consortium (перевод – консорциум всемирной паутины)

URI – uniform resource identifier(перевод – унифицированный (единообразный) идентификатор ресурса)

ТЗ – техническоезадание

API –applicationprogramminginterface (перевод – программный интерфейс приложения, интерфейс прикладного программирования)

CMS – contentmanagementsystem(перевод –система управления содержимым (контентом))

PHP – hypertextpreprocessor(перевод –препроцессор гипертекста)

D.R.Y. – don’trepeatyourself (перевод – неповторяйся)

MVT – Model View Template

MVC – Model-View-Controller

ООП – объектно-ориентированное программирование

TIOBE – индекс, оценивающий популярность языков программирования

ORM – object-relational mapping (перевод – объектно-реляционное отображение, или преобразование)

REST – representationalstatetransfer(перевод – передачасостоянияпредставления)

SQL – structured querylanguage(перевод –язык структурированных запросов)

XSS – cross-site scripting (перевод – межсайтовыйскриптинг)

SASL –simple authentication and security layer (перевод – простой уровень аутентификации и безопасности)

BSD – Berkeley Software distribution

MIT – Massachusetts Institute of Technology(перевод – массачусетский технологический институт)

ACID – Atomicity, Consistency, Isolation, Durability(перевод – атомарность, согласованность, изолированность, устойчивость)

OLAP – online analytical processing (перевод – интерактивная аналитическая обработка)

OLTP – online transaction processing (перевод – транзакционная система)

SSL – Secure Sockets Layer (перевод – уровень защищённых сокетов)

ACL – Access Control List (перевод – список управления доступом)

СПИСОКИСПОЛЬЗОВАННЫХИСТОЧНИКОВ

1. Веб-приложение [Электронная библиотека] https://ru.wikipedia.org/wiki/Веб-приложение(дата обращения – 15.05.2018);
2. Iiko [Электронный ресурс] https://iiko.ru/ (дата обращения – 15.05.2018);
3. Quick Resto [Электронный ресурс] https://quickresto.ru/ (дата обращения – 15.05.2018);
4. Tillypad [Электронный ресурс] https://tillypad.ru/ (дата обращения – 15.05.2018);
5. Draw.io [Электронный ресурс] https://www.draw.io/ (дата обращения – 15.05.2018);
6. XML [Электронная библиотека] https://ru.wikipedia.org/wiki/XML (дата обращения – 15.05.2018);
7. JSON [Электронная библиотека] https://ru.wikipedia.org/wiki/JSON (дата обращения – 15.05.2018);
8. HTML5 [Электронная библиотека] https://ru.wikipedia.org/wiki/HTML5 (дата обращения – 15.05.2018);
9. CSS3 [Электронный ресурс] http://htmlbook.ru/css3 (дата обращения – 16.05.2018);
10. SCSS [Электронный ресурс] https://sass-scss.ru/ (дата обращения – 16.05.2018)
11. JavaScript [Электронный ресурс] https://learn.javascript.ru/ (дата обращения – 16.05.2018);
12. Flexbox[Электронный ресурс] https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout/Flexbox(дата обращения – 16.05.2018);
13. Адаптивная верстка [Электронный ресурс] https://htmlacademy.ru/shorts/16(дата обращения – 17.05.2018);
14. Gulp[Электронный ресурс]https://gulpjs.com/(дата обращения – 17.05.2018);

15. Google [Электронный ресурс] https://www.google.ru/(дата обращения – 17.05.2018);
16. Яндекс [Электронный ресурс] https://yandex.ru/ (дата обращения – 18.05.2018);
17. Mail.ru[Электронный ресурс] https://mail.ru/ (дата обращения – 18.05.2018);
18. Jquery[Электронная библиотека] https://jquery-docs.ru/ (дата обращения – 19.05.2018);
19. Float[Электронная библиотека] https://www.w3schools.com/css/css_float.asp (дата обращения – 21.05.2018);
20. Grid[Электронная библиотека] https://tuhub.ru/posts/css-grid-complete-guide(дата обращения – 21.05.2018);
21. Bootstrap[Электронный ресурс] http://getbootstrap.ru/ (дата обращения – 22.05.2018);
22. Foundation[Электронный ресурс] https://foundation.zurb.com/ (дата обращения – 22.05.2018);
23. Skeleton[Электронный ресурс] http://getskeleton.com/ (дата обращения – 22.05.2018);
24. Grunt [Электронный ресурс] https://gruntjs.com/ (дата обращения – 18.05.2018);
25. Webpack [Электронный ресурс] https:// webpack.js.org/ (дата обращения – 18.05.2018);
26. Gulp [Электронный ресурс] https://gulpjs.com/ (дата обращения – 19.05.2018);
27. React.js [Электронный ресурс] https://reactjs.org/ (дата обращения – 21.05.2018);
28. Django [Электронная библиотека] https://www.djangoproject.com/ (дата обращения – 15.05.2018);
29. Python [Электронная библиотека] https://docs.python.org/3/ (дата обращения – 15.05.2018).

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Кафедра «Информатика»
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

подпись И.В. Евдокимов
инициалы, фамилия

«16» июня 2018 г.

БАКАЛАВРСКАЯ РАБОТА
09.03.04 «Программная инженерия»

Программная система анализа и прогнозирования показателей эффективности предприятия общественного питания

Руководитель 15.06.18 доцент, канд. техн. наук А.С. Кузнецов
подпись, дата; должность, ученая степень; инициалы, фамилия

Выпускник 15.06.18 К. А. Елсуфьев
подпись, дата; инициалы, фамилия

Выпускник 15.06.18 В. В. Андони
подпись, дата; инициалы, фамилия

Выпускник Сырцова 15.06.18 Т. М. Сырцова
подпись, дата; инициалы, фамилия

Выпускник 15.06.18 И. В. Филимонов
подпись, дата; инициалы, фамилия

Нормоконтролер 15.06.18 доцент, канд. техн. наук О.А. Антамошкин
подпись, дата; должность, ученая степень; инициалы, фамилия

Красноярск 2018