Санкт-Петербургский государственный университет

Зыков Артем Юрьевич

Выпускная квалификационная работа бакалавра

Разработка программы составления оптимального плана приготовления блюд в предприятиях общественного питания

Направление 02.03.02
«Фундаментальная информатика и информационные технологии»
ООП СВ.5003.2016: «Программирование и информационные технологии»

Научный руководитель:
доктор физ.-мат. наук,
профессор кафедры математического
моделирования энергетических систем,
Крылатов Александр Юрьевич

Рецензент:
кандидат физ.-мат. наук,
доцент кафедры математического
моделирования энергетических систем,
Лежнина Елена Александровна

Санкт-Петербург
2020

Содержание

Введение

В последнее время все большую актуальность приобретает программное обеспечение, направленное на автоматизацию и оптимизацию различных рабочих процессов. Отдельный интерес представляет в этой области автоматизация предприятий общественного питания.

Среди основных пунктов этой задачи можно выделить контроль и оптимизацию деятельности предприятия, улучшение качества обслуживания посетителей, увеличение производительности труда персонала, снижение издержек предприятия и некоторые другие. На текущий момент существует ряд систем и семейств ПО (R-Keeper, Dodo IS и др.), решающих подобные задачи, но они либо не затрагивают непосредственно производственные процессы, то есть процессы приготовления пищи, либо работают с упрощенными моделями. В основном такие системы используются для работы лишь с информационной составляющей бизнеса.

В такой ситуации ПО, отвечающее за производственные процессы, целиком отсутствует на бесплатной основе, а коммерческое создается специально под нужды конкретной организации и, как правило, не использует весь доступный потенциал сферы. В этом можно увидеть перспективу для разработки программы, способной предоставить приложение, реализующее функции незанятой области, и выявить возможности для дальнейшего ее развития.

В данной работе рассматривается создание приложения, связанного непосредственно с производственными процессами на кухнях предприятий общественного питания и нацеленного на их оптимизацию.

Постановка задачи

Целью данной работы является:

1. Разработка независимого от сторонних информационных систем приложения, способного проводить операции над заказами, а также предоставляющего интерфейс, необходимый для работы поваров на кухне;
2. Создание системы организации пошагового приготовления блюд любой сложности;
3. Создание алгоритма для оптимизации очередей внутри приложения.

Задача разработки приложения состоит в формировании списка требований к будущему продукту, выборе необходимого стека технологий, проработке архитектуры и базы данных, а затем непосредственно в написании логики и интерфейса.

Задача создания системы организации пошагового приготовления блюд рассматривается отдельно в силу специфики реализованного приложения. Необходимо подобрать подходящий математический аппарат и применить его в приложении, задействуя возможности базы данных.

Создание алгоритма для оптимизации очередей предполагается реализовывать последним для возможности учета особенностей формирования очереди внутри приложения и сформированной в ходе разработки модели базы данных.

При помощи созданного приложения сотрудники кухни смогут:

1. Добавлять новые заказы;
2. Просматривать список активных и завершенных заказов;
3. Просматривать очередь приготовления блюд на кухне;
4. Взаимодействовать с очередью, отмечая выполненные шаги рецепта блюда и получая доступ к последующим.

Системные администраторы также смогут:

1. Редактировать список рабочих станций на предприятии;
2. Редактировать список категорий блюд;
3. Редактировать список блюд и добавлять к ним пошаговые рецепты приготовления, отображающиеся в рабочей очереди кухни;
4. Редактировать список пользователей (сотрудников).

Отдельно стоит отметить следующий момент: сотрудникам кухни будут отображаться не все рецепты целиком, а лишь доступные на текущей стадии приготовления шаги. Цель этого — снизить риск ошибок и уменьшить зависимость от человеческого фактора.

Обзор литературы

Для разработки приложения, ответственного за производственные процессы на предприятии, необходимо иметь по меньшей мере базовое понимание принципов организации иерархии и работы поваров, устройства кухни и ее составляющих. В ходе подготовки к работе и в процессе разработки приложения были изучены соответствующие материалы в виде книг и статей.

На профессиональной кухне актуальной является так называемая "бригадная система"(brigade de cuisine), изобретенная французским рестораторoм Огюстом Эскофье в конце XIX века. Сейчас она претерпела некоторые изменения из-за появления современного кухонного оборудования, но она продолжает использоваться в современных ресторанах. Такая система подробно рассмотрена в книге «Mastering Catering Theory» [1]. В ней описываются все профессии, задействованные в работе с кухней, включая шефов и второстепенных помощников. В несколько упрощенном виде, для большей наглядности, систему можно представить следующим образом:

1. Шеф-повар (chef de cuisine). Главный управляющий на кухне. Отвечает за приготовление блюд в ресторане, разработку меню, заказ продуктов, набор нового персонала и обеспечение бесперебойной работы кухни.

2. Помощник шеф-повара (sous chef). Второе по рангу лицо на кухне. Разделяет все обязанности шеф-повара, но проявляет больше помощи поварам на линиях. На больших кухнях может быть до четырех помощников шеф-повара.

3. Линейный повар (chef de patrie). Основная рабочая единица кухни. Отвечает за конкретную рабочую станцию (фритюр, гриль и т.д.), может иметь в подчинении более младших поваров и заместителей.

4. Повар-заготовщик (boucher). Отвечает за разделку мяса или рыбы.

5. Сменный повар (tournant). Повар, способный в нужный момент занять место на конкретной станции.

6. Ученик повара (commis). Стажер-помощник, работающий на одной из станций и отчитывающийся ответственному за эту станцию линейному повару.

Стоит учесть, что данная иерархия может меняться в зависимости от нужд конкретного предприятия и упрощаться до меньшего числа позиций. Например, в небольших кафе и ресторанах функцию заготовщика может выполнять из линейных поваров. Число работников на кухне может варьироваться от одного до более чем сотни человек, и изменения подстраивают под это. Помимо этого было рассмотрено учебное пособие по технологии продукции и организации общественного питания [2]. Данное пособие дало полное представление о цехах, имеющихся на предприятиях общественного питания:

1. Доготовочный цех и цех обработки зелени. Оба цеха используются на предприятиях, связанных с полуфабрикатами, успевшими пройти частичную механическую обработку.

2. Заготовочные цехи. Предусматриваются на предприятиях с полным производственным циклом (на сырье) и делятся на мясо-рыбный и овощной цех.

3. Горячий цех. В производственную программу цеха включают супы, горячие закуски, горячие блюда, соусы, гарниры и горячие напитки.

4. Холодный цех. Производит холодные блюда и закуски, бутерброды, сладкие блюда, холодные супы и напитки. Как правило, продукция этого цеха обладает краткосрочным сроком хранения и реализации.

5. Кондитерский цех. Производит выпечку и связанные с ней десерты.

Особое внимание было уделено частям, посвященным расчетам для горячего и холодного цехов. На них производятся все сложные блюда и они имеются в наличии практически на любом предприятии общественного питания.

В начале работы над системой организации пошагового приготовления блюд и алгоритмом оптимизации очереди также возникла необходимость обратиться к научной литературе ради расширения математической базы. Были изучены книги «Введение в исследование операций» [3] за авторством Хэмди Тахи и «Теория сетей Петри и моделирование систем» Джексона Питерсона [4]. В первой основное внимание было уделено главе «Сетевые модели» и в частности разделу, посвященному методам сетевого планирования. Вторая изучалась в связи с тем, что в сетях Петри удалось увидеть потенциал для системы приготовления блюд. Даже несмотря на то, что в будущем для системы будет использоваться более подходящий математический аппарат, изученный материал дал дополнительную теоретическую базу по сетям.

Глава 1. Обзор существующих решений

Наибольшее сходство с реализуемым приложением имеют кухонные дисплеи — ПО для визуализации заказов на кухне, используемое для налаживания взаимодействия между официантами и кухней, а также улучшения и ускорения сервиса. Данные программы можно найти как на коммерческой, так и на бесплатной основе.

Далее приведены несколько примеров таких систем. Компанией UCS разработан коммерческий кухонный модуль [5], совместимый с информационной системой R-Keeper.

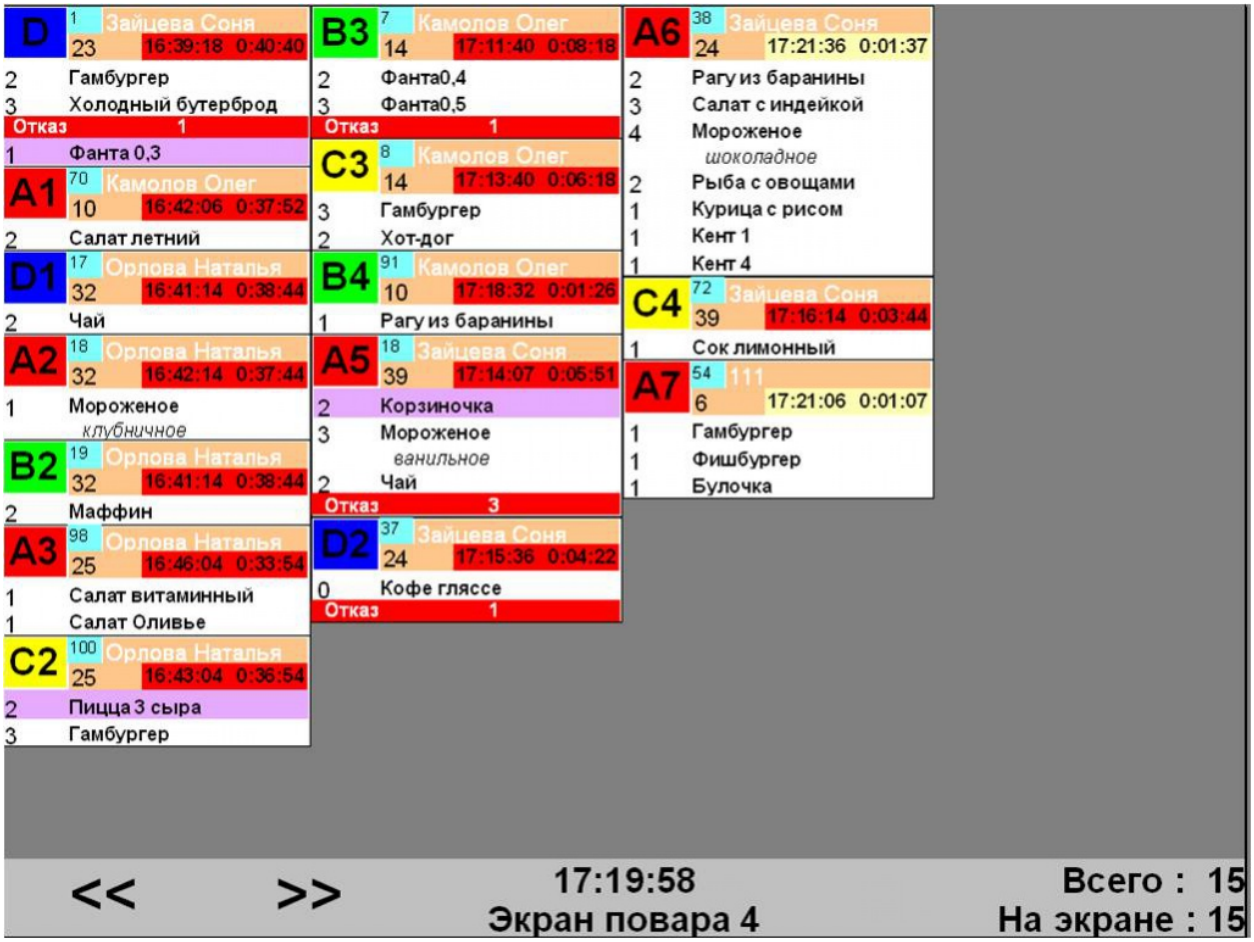

Рис. 1. Интерфейс KDS от компании UCS

В приложении существует два разных типа экранов: экран повара и экран официанта. Каждому заказу соответствует динамическое окно, а количество окон на экране варьируется в зависимости от размеров заказов. Информация о заказе может содержать различные пункты, включая порядковый номер, название блюда, номинальное время приготовления и

т.д. Отдельно стоит отметить, что помимо прочего данная система также позволяет получать отчеты об отклонениях времени приготовленя блюд. На рис. 1 можно увидеть интерфейс экрана повара.

Примером бесплатной системы визуализации заказов может послужить приложение Loyverse KDS, работающее вместе с отдельной программой для розничной торговли.

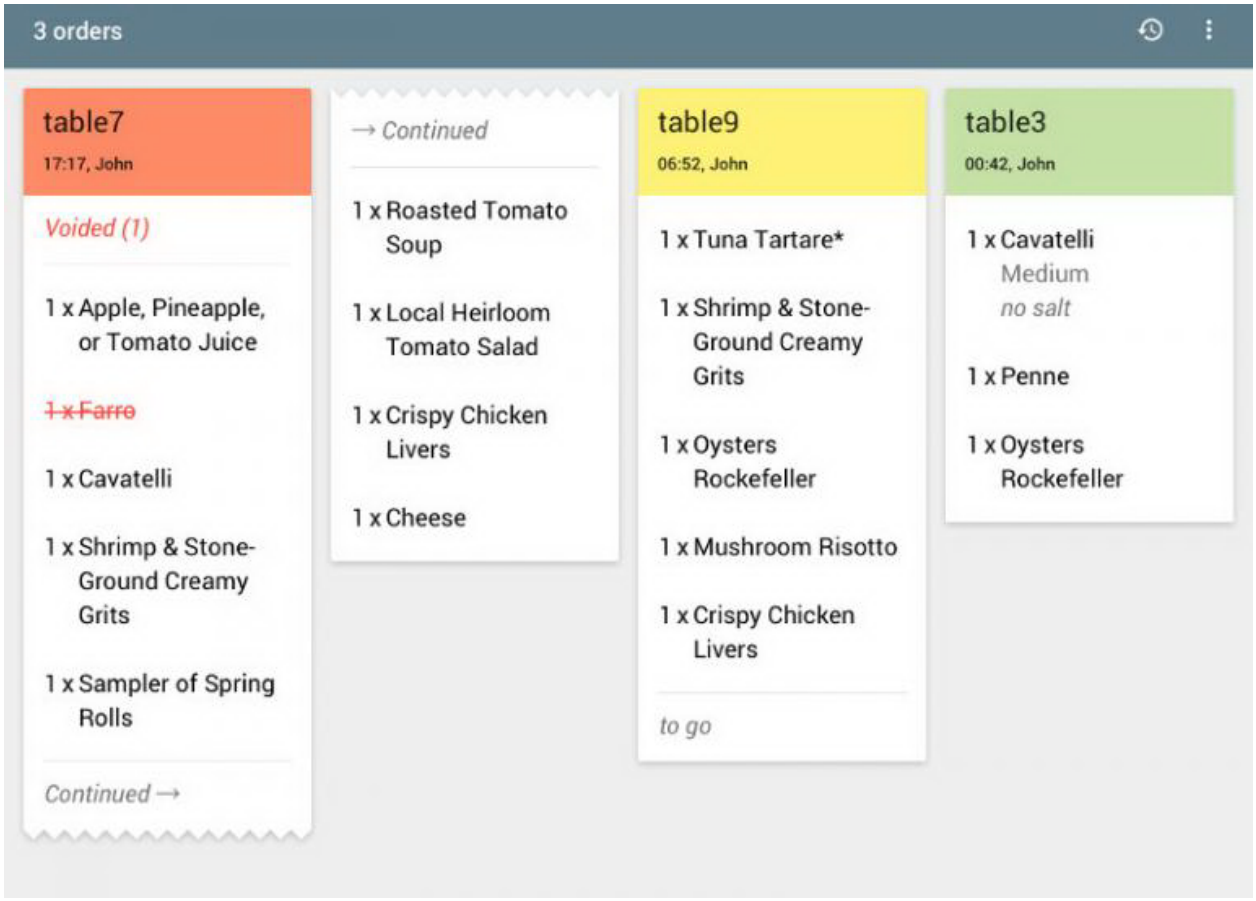

Рис. 2. Интерфейс Loyverse KDS

Программа на кассе автоматически передает заказы в кухонный модуль. Внутри приложения имеется возможность выставлять время ожидания для каждого заказа. Затем, по мере его сокращения, приложение меняет цвета заголовков, привлекая внимание поваров. На экране также выводятся модификаторы для блюд и комментарии к заказу. На рис. 2 изображен пример интерфейса приложения.

Также стоит обратить внимание на трекинговую систему [6] в Dodo IS — информационной системы, созданной для проекта «Додо Пицца». Суть системы заключается в обмене информацией между системой и исполнителями на каждом участке производства, в результате которого со-

трудник кухни получает необходимую для работы информацию и отмечает выполнение различных этапов приготовления пицц на рабочем планшете. Одной из целей системы является выстраивание и автоматизация производственного конвейера.

Такие системы также применяются в ресторанах фаст-фуда, например, в сети ресторанов McDonalds [7]. В качестве одной из возможностей программного обеспечения приводится возможность сохранять рецепты внутри системы и выводить их персоналу кухни при возникновении необходимости.

Отличия реализуемого в ходе работы приложения от представленных выше заключаются в следующем:

1. Существующие трекинговые системы работают с конвейерными производственными процессами. Все шаги выполняются последовательно, параллельность в целом отсутствует. Новое приложение позволит реализовывать рецепты для блюд любой сложности и в том числе поддерживать отображение шагов приготовления соответственно их рабочим станциям.

2. Кухонные дисплеи не отвечают за оптимизацию очереди на кухне. Они отображают заказы и могут напоминать, с какими из них необходимо торопиться в зависимости от номинального времени приготовления блюд, но не проставляют последовательность работ для поваров.

Глава 2. Математическая формализация

2.1 Проблема оптимизации очереди заказов

Задачу оптимизации очереди можно сформулировать и решить как задачу сетевого планирования. Очередь же будет представлена в виде сетевой модели. Сетевая модель — это план, по которому происходит выполнение набора взаимосвязанных работ проекта. Этот набор задается в виде специальной сети, графическое изображение которой называется сетевым графиком.

Основными элементами сетевого графика являются события и работы. Событие отражает момент завершения какого-либо процесса. Это может быть окончание как одной работы, необходимой для начала последующих, так и совокупности нескольких. Событие совершается мгновенно и не затрачивает ресурсов. Выделяют следующие разновидности событий:

1. Исходное событие — отражает факт начала выполнения работ внутри проекта. Не имеет предшествующих работ и событий;

2. Промежуточное событие — результат выполнения одной или множества работ, в связи с которым появляется возможность начать последующие работы;

3. Завершающее событие — означает факт окончания работ проекта. Не имеет последующих работ и событий.

Работы подразделяются на три категории:

1. Действительная работа — протяженный во времени процесс, требующий затрат труда, ресурсов и времени;

2. Ожидание — процесс, занимающий определенное время, но не требующий прочих затрат;

3. Зависимость или фиктивная работа — отражение связи между несколькими работами, не затрачивающее время, труд или ресурсы. Показывает, что возможность одной работы напрямую зависит от завершения другой.

На сетевом графике события изображаются в виде кружков, а работы в виде ориентированных дуг. Также существует принцип построения сети без использования событий. В этом случае вершины графа представляют собой работы, а ориентированные дуги — зависимости, определяющие порядок выполнения.

В текущем случае проектом является совокупность активных заказов на предприятии общественного питания. Очередь выполнения, как уже говорилось выше, представлена в виде сетевой модели. В таком случае заказы будут выступать в качестве внутренних процессов. Поставленная задача будет решаться с применением методов сетевого планирования.

В качестве наиболее известных методов сетевого планирования можно выделить CPM (Critical Path Method) — метод критического пути и PERT (Program Evaluation and Review Technique) — систему планирования и руководства программами разработок. В обоих указанных методах проекты анализируются для создания графиков распределения фаз проекта во времени. Выделяют следующие этапы выполнения:

1. Определяются составляющие проект процессы, связи, определяющие порядок их выполнения, а также длительность;

2. Проект представляется в виде сети, отображающей порядок выполнения процессов внутри проекта и их зависимости друг от друга;

3. Опираясь на построенную сеть, производятся вычисления, зависящие от метода, и выстраивается временной график проекта;

Отличие CPM от PERT заключается в типе длительности процессов. В CPM длительность процессов является детерминированной, в PERT — стохастической. В приложении длительность приготовления блюд будет высчитываться исходя из составляющих их шагов. В то же время каждый шаг будет иметь детерминированную длительность, выставляемую системным администратором. Следовательно, для составления алгоритма будет применяться метод критического пути.

2.2 Метод критического пути

Метод критического пути опирается на обнаружение и расчет наиболее длительной последовательности связанных друг с другом задач от момента начала проекта и до его завершения, называемой критическим путем [3]. Процессы, находящиеся на этом пути, называются критическими. Для некритических процессов возможно смещение времени начала и окончания выполнения, при этом оно не вызывает смещения сроков всего проекта. Соответственно, все критические задачи не имеют резервов времени. В случае, если происходит сдвиг сроков какого-либо критического процесса, то сдвигаются сроки выполнения всего проекта. Также в этом случае может меняться сам критический путь, т.к. изменение длительности работ способно привести к появлению новых критических процессов.

Прежде чем перейти к описанию вычислений, необходимо ввести следующие обозначения:

- e_j — наиболее раннее возможное время наступления события j;
- l_j — наиболее позднее возможное время наступления события j;
- D_{ij} — длительность процесса (i,j).

Расчет критического пути проводится в два этапа:

1. Прямой проход, во время которого происходит вычисляются ранние сроки наступления каждого события. Вычисления проводятся от исходного события к завершающему.
2. Обратный проход, в котором вычисляются поздние сроки наступления каждого события. Вычисления проводятся от завершающего события к исходному.

Во время проведения прямого прохода полагается, что $e_1 = 0$, так как проект начинается в начальный момент времени. Ранние сроки остальных событий вычисляются по формуле:

$$e_j = \max_i \{e_i + D_{ij}\}$$

для всех процессов (i,j). Иначе говоря, величина e_j соответствует наиболее длинному пути от начала проекта до события j. Проход завершается, когда вычисляется величина e_n, где n — завершающее событие.

В начале обратного прохода полагается, что $l_n = e_n$, означающее, что ранние и поздние сроки наступления последнего события равны. Остальные поздние сроки вычисляются по формуле:

$$l_i = \min_i \{l_j - D_{ij}\}$$

для всех процессов (i,j). После вычисления l_1 для исходного события проход завершается.

После проведения обоих проходов можно определить критические операции. Операция (i,j) будет критической, если выполняются три условия:

1. $e_i = l_i$
2. $e_j = l_j$
3. $e_j - e_i = l_j - l_i = D_{ij}$

Процесс, для которого данные условия не выполняются, является некритическим. Критические процессы образуют непрерывный путь, проходящий от исходного события до завершающего.

2.3 Сетевая модель рецептов блюд

Одной из основных задач работы является создание системы организации пошагового приготовления блюд любой сложности. Такая система призвана стать инструментом повышения производительности и контроля, а также помощником в обучении новых сотрудников, которым достаточно будет следовать указанным в очереди шагам приготовления блюд.

С математической точки зрения, такую систему можно представить в виде сетевой модели, реализуемой для каждого рецепта в отдельности. Работами в такой модели выступят отдельные задачи, из которых состоит

процесс приготовления блюда. Соответственно, все их можно разделить на действительные работы и процессы ожидания.

В качестве примера рассматривается следующий рецепт борща [8], представленный в виде набора работ. Стрелка указывает на работы, непосредственно следующие за текущей. Новая работа начинается только после того, как завершаются все идущие за ней:

$0 \to 1,2$. Начало приготовления.

$1 \to 3$. Наполнить кастрюлю водой.

$2 \to 3$. Помыть и порезать на порционные куски мясо.

$3 \to 4,5$. Положить мясо в кастрюлю с водой. Поставить кастрюлю на огонь.

$4 \to 6$. Дождаться кипения воды в кастрюле.

$5 \to 6$. Очистить и вымыть свеклу и морковь. Свеклу порезать на 4 части. Замочить фасоль.

$6 \to 7$. Опустить свеклу, морковь и замоченную фасоль в кипящий бульон. Уменьшить огонь.

$7 \to 8,9$. Варить до полуготовности.

$8 \to 13$. Вынуть свеклу и морковь шумовкой. Натереть на крупной терке. Мелко нарезать лук. Все вместе пассеровать в растительном (или сливочном) масле.

$9 \to 10$. Нарезать картофель. Нашинковать капусту.

$10 \to 11$. Дождаться момента, когда мясо и фасоль будут почти готовы.

$11 \to 12$. Опустить в кастрюлю нарезанный картофель и нашинкованную капусту.

$12 \to 13$. Подождать 10 минут.

13 → 14,15. Положить пассерованные овощи, томатную пасту, лавровый лист. Посолить. Добавить перца.

14 → 16. Растереть чеснок.

15 → 16. Дождаться момента, когда до готовности останется 5 минут.

16 → 17. Положить растертый чеснок в борщ.

17 → 18. Довести до готовности.

18. Разлить готовый борщ по порциям. Блюдо готово.

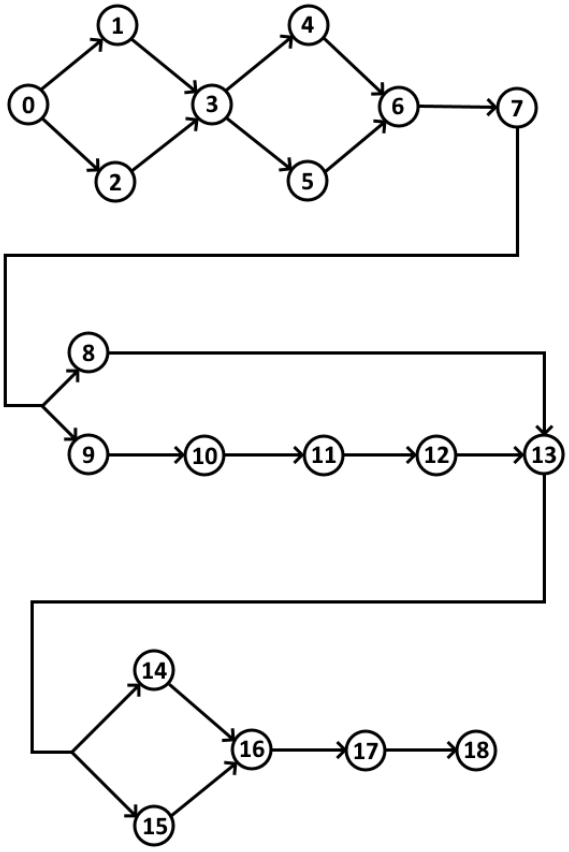

Рис. 3. Сетевой график (без событий) рецепта борща без объединенных процессов

В данном случае действительными работами являются операции под номерами 1, 2, 3, 5, 6, 8, 9, 11, 13, 14, 16, 18. Процессами ожидания являются операции под номерами 4, 7, 10, 12, 15, 17. Нулевой шаг является фиктивной работой и его выполнение инициирует приготовление. На рис. 3 изображен сетевой график рецепта без событий. Кружками обозначены шаги, а стрелками зависимости.

Данное представление имеет один важный недостаток: неудобство работы с процессами ожидания. Такие процессы растягивают схему приготовления и вынуждают сотрудников кухни чаще отвлекаться на фиксирование выполнения шагов. Предлагается внести модификацию, включающую в себя объединение шагов, связанных с добавлением заготовленных ингредиентов в блюдо, и примыкающих к ним процессов ожидания.

Модифицированный вариант рецепта выглядит следующим образом:

$0 \to 1,2$. Начало приготовления.

$1 \to 3$. Наполнить кастрюлю водой.

$2 \to 3,4$. Помыть и порезать на порционные куски мясо.

$3 \to 5$. Положить мясо в кастрюлю с водой. Поставить кастрюлю на огонь. Дождаться кипения.

$4 \to 5$. Очистить и вымыть свеклу и морковь. Свеклу порезать на 4 части. Замочить фасоль.

$5 \to 6,7$. Опустить свеклу, морковь и замоченную фасоль в кипящий бульон. Уменьшить огонь. Варить до полуготовности.

$6 \to 8$. Вынуть свеклу и морковь шумовкой. Натереть на крупной терке. Мелко нарезать лук. Все вместе пассеровать в растительном (или сливочном) масле.

$7 \to 8$. Нарезать картофель. Нашинковать капусту.

$8 \to 9$. Когда мясо и фасоль будут почти готовы, опустить в кастрюлю нарезанный картофель, нашинкованную капусту. Через 10 минут положить пассерованные овощи, томатную пасту или соус, посолить, положить лавровый лист, перец.

$9 \to 10$. Растереть чеснок.

$10 \to 11$. За 5 минут до готовности положить в борщ растертый чеснок. Довести до готовности.

11. Разлить борщ по порциям. Блюдо готово.

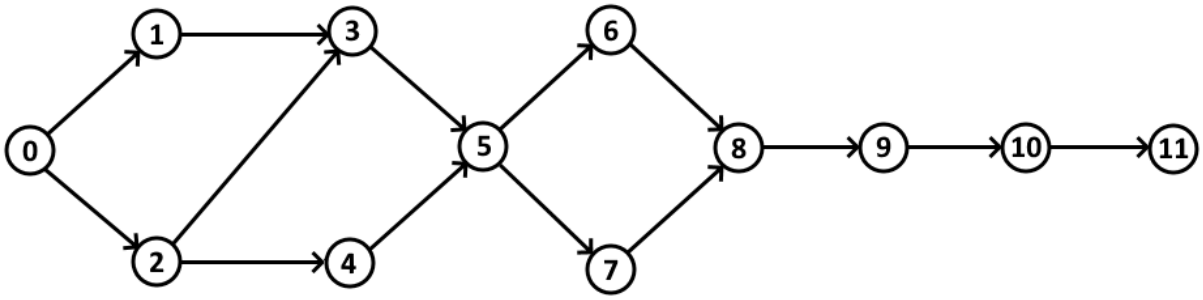

Рис. 4. Сетевой график (без событий) рецепта борща с объединенными процессами

В данном случае полноценных процессов ожидания в рецепте нет, но работать с такими шагами проще за счет возможности совместить наблюдение за временем приготовления и небольшие побочные действия. Таким образом число фиксаций прогресса выполнения, не связанных с какими-либо активными действиями повара, сведется к нулю. На рис. 4 изображен сетевой график сжатого рецепта.

Подразумевается, что для каждого шага будет выставлена длительность выполнения. С учетом человеческого фактора, она не сможет в точности соответствовать реальным процессам, однако ее определение необходимо для расчета длительности приготовления всего блюда.

Длительность приготовления блюда можно вычислить через нахождение критического пути среди шагов. Это актуально для тех случаев, когда все действительные работы, доступные на конкретном этапе приготовления блюда выполняются параллельно максимально необходимым количеством поваров. Соответственно, общая длительность реализации блюда ограничена самой продолжительной цепочкой шагов. Однако, если блюдо готовится одним поваром, то дополнительно к длительности цепочки необходимо прибавить длительность всех шагов, не связанных с ожиданием и не входящих в цепочку. Таким образом удастся учесть, что все производственные шаги будут выполняться одним человеком последовательно. Случаи, при которых блюдо готовится определенным числом поваром, в данной работе не рассматриваются в силу возможных отличий между кухонными иерархиями разных предприятий общественного питания, которые не учитываются приложением. Также в программной реализации необходимо

учесть, что длительность некоторых шагов может возрастать в зависимости от числа заказанных порций.

2.4 Алгоритм оптимизации очереди

Для реализации алгоритма оптимизации очереди необходимо учитывать следующие моменты:

1. Приложение не учитывает количество поваров на кухне. Как уже было упомянуто в разделе 2.3, различные предприятия могут иметь разную организацию сотрудников кухни, зависящую от типа предприятия, его меню и возможностей, а также различные схемы работы. В силу необходимости работать с любым типом предприятия, количество поваров будет исключено из учета;

2. Внутри приложения будут реализованы две параллельные подочереди: для горячего и холодного цехов. Указанные цехи присутствуют на большинстве предприятий общественного питания. При отсутствии определенного цеха предприятие сможет не использовать нерабочую подочередь. Таким образом, необходимо учитывать, что заказы могут выполняться как в одной подочереди, так и в двух сразу, и резервы времени для работ в обеих подочередях отличаются;

3. Более ранние заказы имеют приоритет над более поздними. Следовательно, при отсутствии резервов времени заказы должны выполняться в порядке их поступления на кухню;

4. Расчет начала и завершения конкретного заказа должен исходить из того, чтобы этапы работ на обеих очередях завершались в примерно одинаковое время для минимальных простоев блюд. Соответственно, расчетное время завершения работ по заказу в каждой из подочередей должно совпадать.

Во время составления шагов алгоритма также необходимо опираться на три этапа работы с методами сетевого планирования. Основным элементом очереди является заказ. Заказ состоит из внутренних позиций, каждая

из которых обозначает определенное блюдо и количество порций. Предполагаемая длительность выполнения позиции рассчитывается, исходя из заданной внутри системы длительности приготовлении соответствующего блюда и прибавок времени, связанных с возможным увеличением числа порций.

Для начала вводятся следующие обозначения:

- T_0 — начальный момент времени.
- CD_i — предположительная длительность выполнения заказа i в холодном цехе.
- HD_i — предположительная длительность выполнения заказа i в горячем цехе.
- CS_i — предположительное время начала работ заказа i в холодном цехе.
- HS_i — предположительное время начала работ заказа i в горячем цехе.
- F_i — предположительное время завершения работ заказа i. Высчитывается для обоих цехов одновременно в силу пункта 4 списка важных моментов выше.
- HQ_{ij} — разница во времени между предполагаемым завершением работ заказа i и предполагаемым началом работ заказа j в горячем цехе. Высчитывается только для соседних заказов в подочереди.
- CQ_{ij} — разница во времени между предполагаемым завершением работ заказа i и предполагаемым началом работ заказа j в холодном цехе. Высчитывается только для соседних заказов в подочереди.

Алгоритм, в основе которого лежит метод критического пути, представлен следующим образом:

1. Высчитывается предполагаемая длительность приготовления каждого заказа в каждом цехе. В силу отсутствия учета числа поваров на кухне берется максимальная из длительностей приготовления позиций заказа, соответствующих конкретному цеху. Если заказ готовится целиком в одном цехе, то длительность его приготовления в другом равна нулю.

2. Проводится сортировка заказов по времени поступления на кухню от самого раннего к самому позднему.

3. Рассчитывается предполагаемое время завершения каждого заказа, начиная с первого.

   Для первого заказа предполагаемое время завершения:

$$F_1 = T_0 + max\{CD_1, HD_1\}$$

   Для последующих заказов происходит разделение. Для заказов, имеющих позиции в двух цехах, завершение считается с учетом предшествующих в обеих подочередях:

$$F_i = max\{F_j + CD_i, F_k + HD_i\},$$

   где j — предыдущий заказ в подочереди холодного цеха, а k — предыдущий заказ в подочереди горячего цеха. j и k не всегда совпадают, т.к. могут присутствовать заказы, готовящиеся лишь в одном цехе.

   Для заказов, выполняющихся целиком в одном цехе, используется иная формула:

$$F_i = F_j + CD_i$$

   для заказов холодного цеха и

$$F_i = F_k + HD_i$$

для заказов горячего цеха.

4. Идет расчет предполагаемого времени начала приготовления каждого заказа:

$$CS_i = F_i - CD_i,$$

$$HS_i = F_i - HD_i$$

Для заказов, выполняющихся в одном цехе, начало в другом не высчитывается.

5. Высчитываются резервы времени между соседними заказами. Для первого заказа в каждой из подочередей расчеты проводятся по следующей формуле:

$$HQ_{0i} = HS_i - T_0,$$

$$CQ_{0i} = CS_i - T_0,$$

где i — первый заказ в подочереди. Для последующих заказов, отмеченных как i и j, из предположительного времени начала заказа j вычитается предположительное время завершения предшествующего ему заказа i:

$$CQ_{ij} = CS_j - F_i$$

$$HQ_{ij} = HS_j - F_i,$$

Для заказов, выполняющихся целиком в одном цехе, резервы времени в другом не расчитываются.

6. Создается отдельный общий для обеих подочередей список заказов,

готовящихся целиком в одном цехе. Их порядок определяется предполагаемым началом времени приготовления каждого заказа.

7. Проводится последовательный проход по заказам из созданного списка. Если в их подочереди существует резерв времени между заказами (CQ_{ij} для холодного цеха и HQ_{ij} для горячего цеха), лежащий раньше по очереди, а также превышающий или равный длительности выполнения заказа, то предполагаемое время начала и завершения заказа смещается:

$$CS_k = F_i,$$

$$F_k = CS_k + CD_k$$

для заказа в холодном цехе и

$$HS_k = F_i,$$

$$F_k = HS_k + HD_k$$

для заказа в горячем цехе, где i — заказ с левой стороны временного отрезка резерва, k — рассматриваемый заказ.

В случае сдвига одного заказа необходимо скорректировать время начала и завершения всех заказов из общей очереди, а также обновить резервы времени, исходя из скорректированной очереди.

8. Если очередной проход не дает возможности сдвинуть какой-либо заказ в свободный временной резерв, алгоритм завершает работу. Для обоих цехов имеются оптимизированные подочереди, а также общая очередь, порядок в которой определяется временем предполагаемого начала выполнения заказа.

Сдвиги не выполняются для заказов, выполняющихся в двух цехах,

потому что они лежат на критическом пути вне зависимости от доступных временных резервов. Связано это с синхронизацией времени окончания приготовления разных частей заказа для обоих цехов одновременно.

Глава 3. Программная реализация

3.1 Выбор технологий

Следуя примеру трекинговой системы Dodo IS [6], было решено реализовать продукт в виде веб-приложения. В этом случае сотрудники кухни смогут взаимодействовать с системой через Интернет и веб-браузер.

У SaaS-решения (Software as a Service, программное обеспечение как услуга), которым является реализуемое приложение, был выделен ряд преимуществ:

1. Легкость в использовании и поддержке;
2. Возможность быстро развернуть систему;
3. Единая для всех рабочих точек предприятия техническая поддержка;
4. Единое место расположения серверов, хранящих данные;
5. Независимость от мощности используемого оборудования на рабочих точках. Единственное требование — возможность запустить веб-браузер.

Также стоит отметить риски, связанные с таким приложением:

1. Зависимость от Интернет-соединения;
2. Необходимость интегрировать сторонний модуль в уже имеющуюся информационную систему.

Второй пункт связан в первую очередь с тем, что реализуемое в данной работе приложение является независимым решением и не располагает специальными инструментами для интеграции в уже созданные информационные системы, которые в том числе могут использоваться на конкретном предприятии.

В качестве фреймворка был выбран Django, входящий в список из десяти наиболее популярных веб-фреймворков [9]. Django по умолчанию

предоставляет разработчику ряд функциональностей, позволяющих сконцентрироваться на работе над бизнес-логикой вместо реализации базовых инструментов. Автоматически генерируемая панель администратора, миграции баз данных, виртуальная объектная база данных (ORM), встроенная аутентификация пользователей — данные моменты важны для реализуемого приложения. Также фреймворк включает в себя механизмы предотвращения широкоиспользуемых хакерских атак [10], например подделки межсайтовых запросов или SQL-инъекций.

Соответственно, в качестве языка программирования выбран Python 3.7 в следствие того, что именно на Python реализуется логика внутри выбранного фреймворка. Помимо этого, для веб-страниц в приложении будет использоваться язык разметки HTML и язык описания внешнего вида CSS. Для реализации некоторой интерактивности на страницах помимо возможностей Django также будет использован язык JavaScript.

По умолчанию фреймворк Django использует встраиваемую базу данных SQLite. Помимо этого рассматривались также MySQL и PostgreSQL, но итоговый выбор остановился на MySQL. Главной причиной отказа от SQLite являются трудности в ее использовании для крупных баз данных, что может быть критически важно для большого предприятия общественного питания, а также сложности в масштабировании. PostgreSQL максимально соответствует современным стандартам SQL и имеет достаточную гибкость, но он не отличается простотой администрирования, поэтому предпочтение отдается MySQL из-за его простоты, достаточной функциональности и возможности для дальнейшего масштабирования. [11] Помимо прочего, MySQL является второй по популярности базой данных. [12]

3.2 Модель базы данных

При реализации модели базы данных необходимо было учитывать как потребности непосредственно кухни, например, список блюд и заказов, так и моменты, необходимые для полноценной реализации модели приготовления блюд и алгоритма оптимизации очереди. В результате была создана модель, представленная на рис. 5

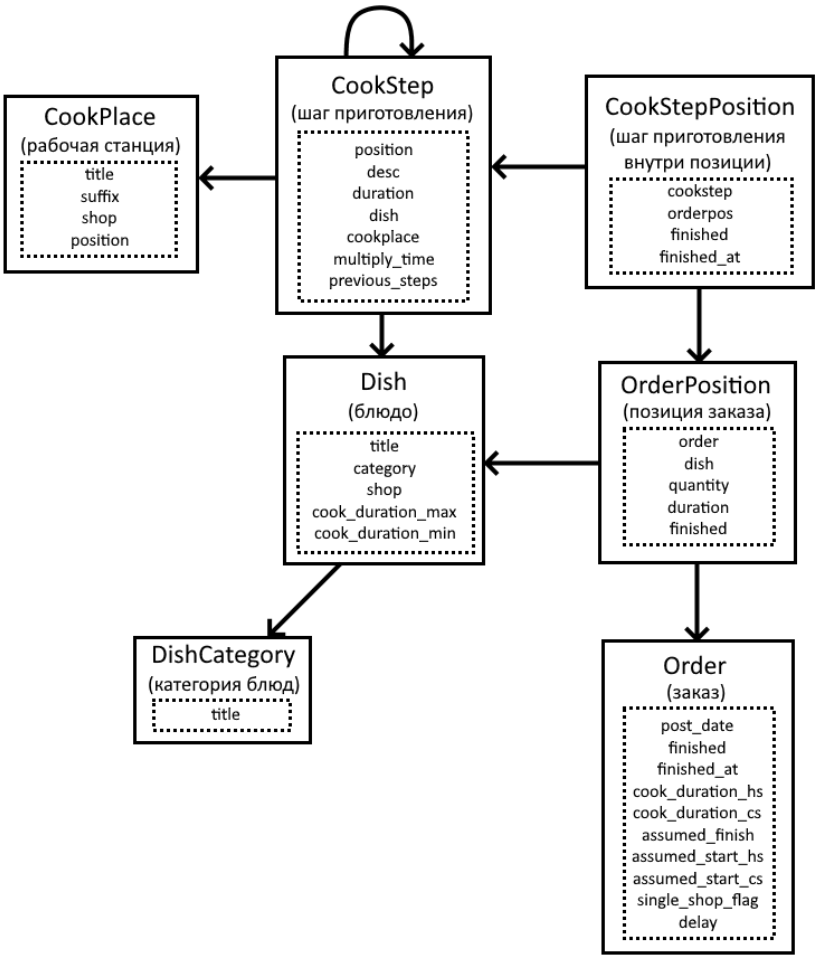

Рис. 5. Модель базы данных

Модель базы данных состоит из следующих элементов и атрибутов:

- CookStep — шаг приготовления:

    1. Position — номер шага в рецепте блюда. С выполнением последнего шага блюдо считается готовым;
    2. Desc — текстовое описание шага;
    3. Duration — длительность выполнения шага;
    4. Dish — внешний ключ. Блюдо, к которому относится шаг;
    5. Cookplace — внешний ключ. Рабочая станция, на которой необходимо выполнить шаг;
    6. Multiply time — флаг, определяющий, есть ли необходимость пропорционально увеличивать время выполнения шага при уве-

личении числа порций (выставляется, например, для шагов с нарезкой продуктов);

7. Previous steps — связь многие-ко-многим. Указывает на шаги, непосредственно предшествующие данному.

- Dish — блюдо:

1. Title — название блюда;
2. Category — внешний ключ. Категория блюда;
3. Shop — цех приготовления. Предоставляет выбор из двух вариантов: «Холодный цех» и «Горячий цех»;
4. Cook duration max — длительность приготовления блюда одним поваром. Используется внутри расчетов алгоритма оптимизации очереди;
5. Cook duration min — длительность приготовления блюда максимально возможным числом поваров при полной распараллелизации производственных процессов. В расчетах не используется.

- DishCategory — категория блюд:

1. Title — название категории.

- CookPlace — рабочая станция:

1. Title — название станции;
2. Suffix — опциональный скрытый суффикс. Отображается в панели администратора и помогает отличить станции с одинакомыми названиями, но относящиеся к разным цехам;
3. Shop — цех, к которому относится станция. Предоставляет выбор из двух вариантов: «Холодный цех» и «Горячий цех»;
4. Position — позиция станции. Используется для отображения на странице очереди цеха. Чем меньше число, тем левее будет отображаться станция и привязанные к ней шаги.

- CookStepPosition — шаг приготовления внутри конкретной позиции:
    1. Cookstep — внешний ключ. Соответствующий шаг рецепта;
    2. Orderpos — внешний ключ. Соответствующая шагу позиция заказа;
    3. Finished — флаг завершенности;
    4. Finished at — время завершения.
- OrderPosition — позиция внутри заказа:
    1. Order — внешний ключ. Заказ, к которому относится позиция;
    2. Dish — внешний ключ. Приготавливаемое блюдо;
    3. Quantity — число порций;
    4. Duration — предполагаемая длительность выполнения позиции;
    5. Finished — флаг завершенности.
- Order (заказ):
    1. Post date — дата создания заказа;
    2. Finished — флаг завершенности;
    3. Finished at — время завершения;
    4. Cook duration hs — предполагаемая длительность приготовления заказа в горячем цехе;
    5. Cook duration cs — предполагаемая длительность приготовления заказа в холодном цехе;
    6. Assumed finish — предполагаемое время завершения заказа. В отличие от «Finished at» используется для оптимизации очереди;
    7. Assumed start hs — предполагаемое время начала работ заказа в горячем цехе;
    8. Assumed start cs — предполагаемое время начала работ заказа в холодном цехе;

9. Single shop flag — флаг, отмечающий факт выполнимости заказа целиком в одном цехе;
10. Delay — предполагаемая задержка между текущим моментом времени и началом работ заказа. Используется на странице отображения очереди.

Системному администратору необходимо заполнять только список блюд, категорий блюд, шагов приготовления и рабочих станций. Заказы, позиции и шаги внутри позиций создаются и управляются самим приложением. Длительность приготовления блюд также высчитывается автоматически на основе шагов, привязанных к ним. Манипуляции с базой данных приложения проводятся через панель администратора, что также облегчает работу системному администратору.

Для реализации пошагового приготовления блюд самым важным атрибутом является *previous_steps* в таблице *CookStep*. Очередной шаг приготовления не отображается в очереди до тех пор, пока не будут отмечены завершенными предыдущие. Также благодаря этому система получает возможность рассчитать длительность приготовления блюда, проходя по списку принадлежащих ему шагов.

3.3 Интерфейс приложения

Созданный в ходе работы интерфейс удовлетворяет всем пунктам, описывающим возможности пользователей и системного администратора, определенным в разделе «Постановка задачи». Всего было создано 7 страниц, необходимых для полноценной работы с системой. Ниже представлен их список с описанием и изображениями:

1. Страница выбора панели (рис. 8). Здесь можно выбрать требуемую для дальнейшей работы панель из двух вариантов: панели работника и панели администратора. При отсутствии авторизации происходит переход на страницу входа.

2. Страница входа (рис. 9). Здесь сотрудник кухни обязан ввести кор-

ректные данные для входа, чтобы получить доступ к основным рабочим страницам.

3. Панель работника (рис. 10). Отсюда работник имеет возможность перейти в очереди горячего и холодного цехов, на страницу списка заказов и добавления заказов, а также выйти с аккаунта. Системные администраторы также имеют доступ к кнопке «Обновить блюда» отвечающей за обновление расчетов времени для блюд.

4. Очередь горячего цеха (рис. 11). Здесь повар видит активные на данный момент шаги выполнения блюд, готовящихся в горячем цехе. Все шаги размещены по соответствующим им станциям, а также для всех система выставляет приоритеты, в соответствии с которым рекомендуется их выполнять. Шаги можно переводить в категорию выполненных, а выполненные можно возвращать назад в рабочую очередь. В зависимости от списка рабочих станций интерфейс будет выглядеть иначе. Также отображается предположительная задержка до начала работы над блюдом в конкретном заказе.

5. Очередь холодного цеха (рис. 12). Все аналогично очереди горячего цеха, но блюда и станции выбираются под холодный цех.

6. Список заказов (рис. 13). Здесь отображаются активные и выполненные заказы вместе с их позициями. Выполненные заказы отмечаются зеленым цветом, активные красным.

7. Страница добавления заказов (рис. 14). Здесь сотрудники кухни могут добавлять новые заказы в очередь, выбирая в соответствующем меню (рис. 15) необходимые блюда и количество порций. Блюда и разделы в меню создаются динамически в зависимости от категорий и блюд, внесенных в базу данных.

При определении размеров элементов и шрифтов учитывалось, что основная работа будет проводиться на сенсорных экранах. Соответственно, кнопки и текст специально были сделаны крупными.

Панель администратора автоматически создана средствами Django. Через нее возможно редактировать любые объекты из базы данных, а также добавлять и редактировать пользователей. На рис.16 изображен пример блюда, отображенного в панели.

3.4 Реализация алгоритма оптимизации

Реализованный на Python алгоритм следует всем шагам, указанным в разделе 2.4. Стоит учесть, что алгоритм не высчитывает и не изменяет очередь в режиме реального времени. Его запуск проводится каждый раз, когда производится переход на страницу одной из подочередей, а также если какой-либо шаг отмечается как готовый или возвращается обратно в очередь. Учитывая, что результатом его работы становятся две оптимизированные подочереди, какие-либо различия в его исполнении для каждого цеха отсутствуют.

Файл *queryapp/views.py* [13] содержит в себе код алгоритма объемом в 298 строк внутри функций *hsqueue* и *csqueue* . Создавая первичную очередь заказов, отсортированных по времени поступления от самых ранних к самым поздним, алгоритм проводит оптимизацию и на основе улучшенной очереди формирует новый порядок заказов. Очередь горячего цеха получает заказы, связанные с горячим цехом, очередь холодного — связанные с холодным. Затем шаблон страницы, ответственный за отображение очереди, получает список позиций внутри каждого заказа и формирует список шагов, которые затем распределяются по столбцам рабочих станций. При выводе шагов учитываются следующие моменты:

1. При работе над заказом в первую очередь готовятся блюда, занимающие больше времени. Таким образом минимизируются возможные простои блюда после приготовления.

2. Для выполнения отображаются лишь шаги, не отмеченные готовыми, а также у которых отмечены готовыми все предыдущие.

Таким образом учитывается не только порядок приготовления заказов, но и длительность выполнения позиций внутри каждого заказа, а

также доступность конкретных работ для каждого блюда.

Также всем шагам выставляется приоритет, по которому можно определить, какие шаги более важны вне зависимости от загруженности очереди и количества рабочих станций. Если блюдо должно выполняться не сразу, то для него также выводится предполагаемая задержка. Задержка и приоритет имеют рекомендательный характер и при необходимости сотрудники кухни могут их игнорировать.

При выполнении определенного шага рассчитанная длительность приготовления позиции, к которой он относится, уменьшается, и система вновь запускает алгоритм для проведения корректировки очереди.

Для проверки работоспособности алгоритма в базу данных был внесен список тестовых блюд, состоящих из одного шага и равных по длительность 15 и 30 минутам. Всего было создано четыре блюда, по два на каждый цех. С их помощью был создан тестовый список заказов. Далее будут приведены длительности работ для каждого заказа в обоих цехах. Если для определенного цеха не указано время, значит заказ не имеет блюд, готовящихся в нем:

1. 15 минут в горячем цехе, 60 минут в холодном цехе;
2. 60 минут в горячем цехе, 30 минут в холодном цехе;
3. 15 минут в горячем цехе;
4. 30 минут в горячем цехе, 60 минут в холодном цехе;
5. 15 минут в горячем цехе;
6. 30 минут в горячем цехе, 30 минут в холодном цехе;
7. 30 минут в горячем цехе, 15 минут в холодном цехе;
8. 15 минут в холодном цехе.

Таким образом, здесь имеются 3 заказа, целиком готовящиеся в определенном цехе.

На рис.6 схематично изображена первичная очередь, формирующаяся на основе порядка поступления заказов на кухню. Верхний ряд обозначает порядок и длительность работ в горячем цехе, нижний — в холодном. Как видно, в горячем цехе присутствует большой свободный промежуток времени, равный 45 минутам. Заказы 3 и 5, длительность работ которых составляет по 15 минут, могут сместиться левее. После их смещения также можно сдвинуть работы заказа 6, т.к. освободившееся после сдвига заказа 5 время образует пустоту. В холодном цехе присутствует пустой промежуток в 30 минут между 1 и 2 заказами. Длительность 8 заказа составляет 15 минут и его можно сместить в этот промежуток.

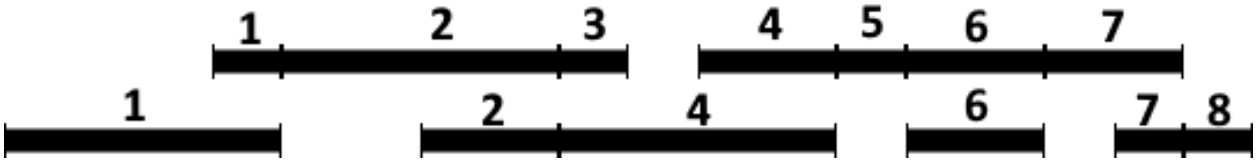

Рис. 6. Первичная очередь заказов

В результате работы алгоритма формируется очередь, схематично представленная на рис. 7.

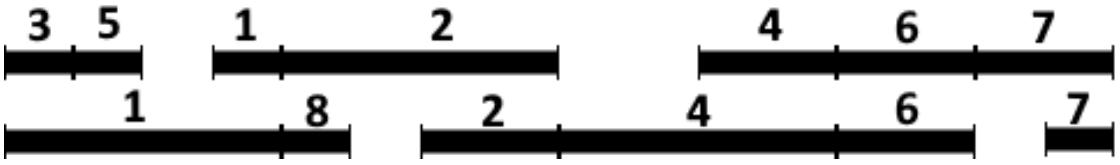

Рис. 7. Оптимизированная очередь заказов

Внутри приложения очередь горячего цеха также это отображает. На рис. 17 представлен интерфейс, отображающий все доступные шаги. Необходимо лишь сделать поправку на номера заказов, на изображении они смещены на 4. Соответственно, 1 заказ из списка стал 5, 2 заказ стал 6 и т.д. Аналогами смещенных 3 и 5 заказов являются заказы под номерами 7 и 9. Как видно, они стоят раньше первого. Более того, это подтверждается выставленной задержкой. У первого заказа она равна 45 минутам, а у пятого 15 минутам.

Положительного результата удалось достичь и в очереди холодного цеха. Делая аналогичную поправку, 8 заказ становится 12. Как видно на рис. 18, он имеет второй приоритет и задержку в 1 час. Первый приоритет имеет шаг из первого заказа, что также подтверждается приведенной выше очередью. Следовательно, алгоритм работает корректно.

Выводы

В ходе изучения предметной темы работы и существующего программного обеспечения был получен вывод, что существует слишком мало систем, непосредственно взаимодействующих с производственными процессами на кухнях предприятий общественного питания. Те же, что созданы для этого, работают с конвейерным производством и неспособны организовать работы над сложными блюдами. Следовательно, задача реализации программного обеспечения, ответственного за организацию и оптимизацию работы на любых кухнях предприятий общественного питания является актуальной.

Удалось выявить, что для оптимизации очереди на кухни можно применить один из методов сетевого планирования — метод критического пути, соответственно представив саму очередь в виде сетевой модели. Принцип построения сетевых моделей и метод также нашли свое применение в системе организации пошагового приготовления блюд. Затем, учитывая особенности реализуемого продукта, был создан алгоритм, формирующий оптимизированные подочереди для двух рабочих цехов.

После подбора стэка технологий было реализовано веб-приложение, предоставляющее сотрудникам предприятия возможность самостоятельно создавать заказы, просматривать их список, а также взаимодействовать с очередями приготовления, отмечая готовыми выполненные шаги рецептов и получая для выполнения новые, таким образом закрывая позиции и, соответственно, заказы. Системные администраторы предприятия имеют возможность заполнять категории блюд, их списки и рецепты, а также редактировать список рабочих станций.

Одной из особенностей приложения, а также его недостатком, является отсутствие учета числа поваров на кухне. Этот момент не стал учитываться в силу разницы иерархий сотрудников кухни на предприятиях, способной привести к необходимости по-разному учитывать все процессы. Тем не менее, при необходимости подстроить систему под потребности конкретного предприятия этот момент не является критическим: реализованный алгоритм рассчитывает очередь, исходя из длительности приготов-

ления позиций внутри заказов. Соответственно, все дополнения, которые необходимо будет внести, должны будут изменить лишь принцип расчета длительности приготовления заказа, а не весь алгоритм.

Также может представлять сложность отделенность приложения от существующих информационность систем. Отдельный модуль представляет мало смысла для предприятия, использующего коммерческие системы программного обеспечения, включающие в себя множество работающих совместно модулей, поэтому при возникновении необходимости присоединить реализованное приложение к уже существующей системе могут потребоваться дополнительные затраты на перенос или адаптацию. Этот момент частично сглаживается за счет наличия API в Django, через который можно взаимодействовать с базой данных приложения со стороны. Следовательно, отсутствует критическая необходимость создавать аналогичное приложение внутри коммерческой системы с нуля.

Помимо этого, есть потенциал для косметических улучшений интерфейса, таких как, например, возможность отображать на экране очередь шагов для конкретной рабочей станции, а не для всех сразу.

В целом же, приложение удовлетворяет всем поставленным изначально требованиям и реализует все требуемые функциональности. Созданный алгоритм также показывает свою работоспособность на практике.

Заключение

В ходе данной работы были получены следующие результаты:

1. Создана система пошагового приготовления блюд с использованием сетевой модели.

2. Написан и реализован алгоритм оптимизации очереди заказов и производственных процессов на кухне, использующей в работе два рабочих цеха, с применением одного из методов сетевого планирования — метода критического пути.

3. Разработано веб-приложение, предоставляющее сотрудникам кухни возможность работать над поступающими заказами в соответствии с выстроенной алгоритмом очередью, а также позволяющее добавлять новые заказы и просматривать их список. У системных администраторов есть возможность заполнять и редактировать базу данных через панель администратора, внося блюда, рецепты, категории блюд, а также рабочие станции.

Исходный код приложения размещен в публичном репозитории на GitHub [13].

Список литературы

[1] Taylor E., Taylor J. Mastering Catering Theory — London: Palgrave, 1990. — 432 p.

[2] Технология продукции и организация общественного питания. Методика выполнения выпускной квалификационной работы бакалавра: учеб. пособие / Несмелова С. В. [и др.] — СПб. : ПОЛИТЕХ-ПРЕСС, 2018. — 161 с.

[3] Таха Х. Введение в исследование операций. 6-е издание. : Пер. с англ. — М. : Издательский дом «Вильямс», 2001. — 912 с.

[4] Питерсон Дж. Теория сетей Петри и моделирование систем. — М: Мир, 1984. — 264 с.

[5] Системы KDS и VDU — мониторы для кухни [Электронный ресурс]: URL: `https://www.ucs.ru/products/r_keeper/system-kds-and-vdu-monitors-for-the-kitchen/` (дата обращения: 23.04.2020).

[6] Блог команды «Додо Пицца». Трекинг заказов [Электронный ресурс]: URL: `https://sila-uma.ru/2011/10/20/treking-zakazov/` (дата обращения: 02.04.2020).

[7] McDonald Paper Blog. Using The POS System In Restaurant Kitchen Management [Электронный ресурс]: URL: `https://mcdonaldpaper.com/blog/pos-in-restaurant-kitchen-management` (дата обращения: 07.05.2020).

[8] Борщ (ТТК4133). Технологическая карта [Электронный ресурс]: URL: `https://tekhnolog.com/2019/01/31/borshh-ttk4133/` (дата обращения: 09.05.2020)

[9] HotFrameworks: Web framework rankings [Электронный ресурс]: URL: `https://hotframeworks.com/` (дата обращения: 05.04.2020)

[10] Security in Django. Django documentation [Электронный ресурс]: URL: https://docs.djangoproject.com/en/2.2/topics/security/ (дата обращения: 26.03.2020)

[11] SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems [Электронный ресурс]: URL: https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems (дата обращения: 25.03.2020)

[12] DB-Engines Ranking [Электронный ресурс]: URL: https://db-engines.com/en/ranking (дата обращения: 28.03.2020)

[13] Репозиторий проекта в GitHub [Электронный ресурс]: URL: https://github.com/Kn1MS/CatestQuery (дата обращения: 31.05.2020).

Приложения

Рис. 8. Интерфейс страницы выбора панели

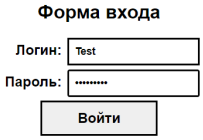

Рис. 9. Интерфейс страницы входа

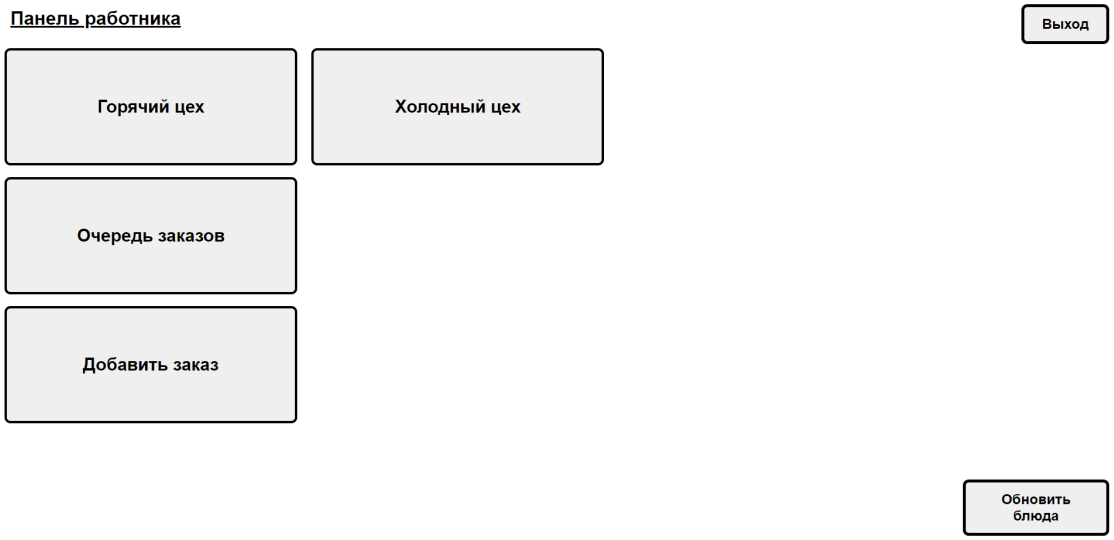

Рис. 10. Интерфейс панели работника

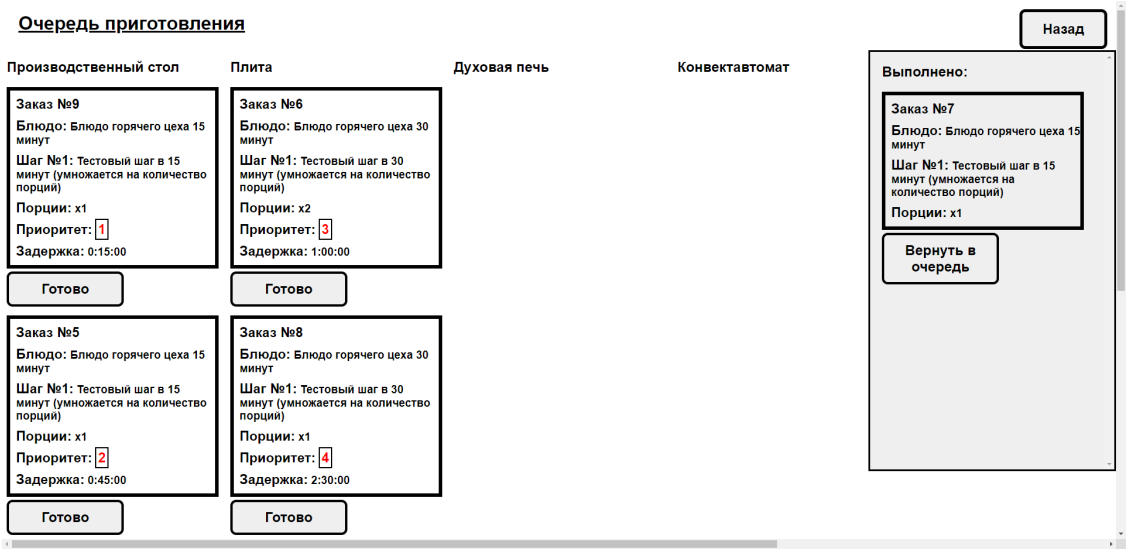

Рис. 11. Интерфейс очереди горячего цеха

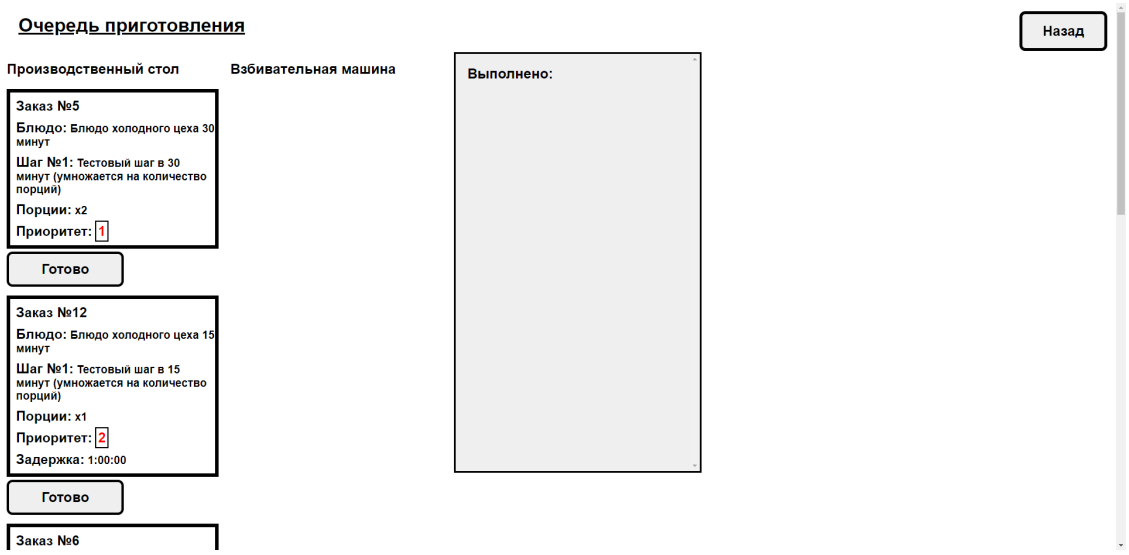

Рис. 12. Интерфейс очереди холодного цеха

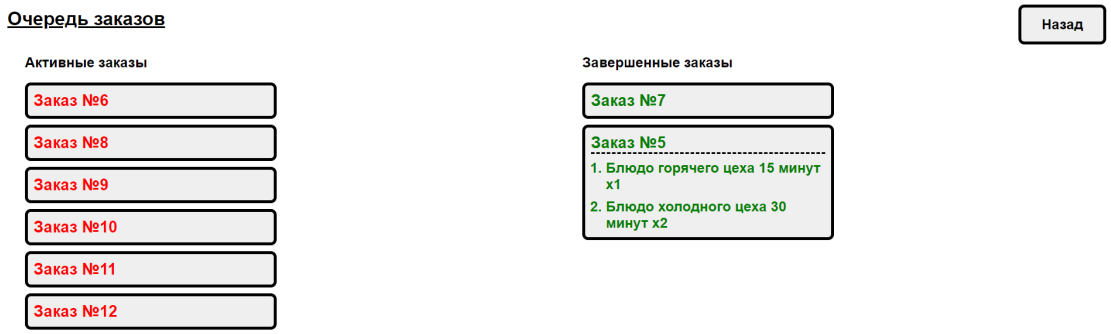

Рис. 13. Интерфейс страницы списка заказов

Рис. 14. Интерфейс страницы добавления заказов

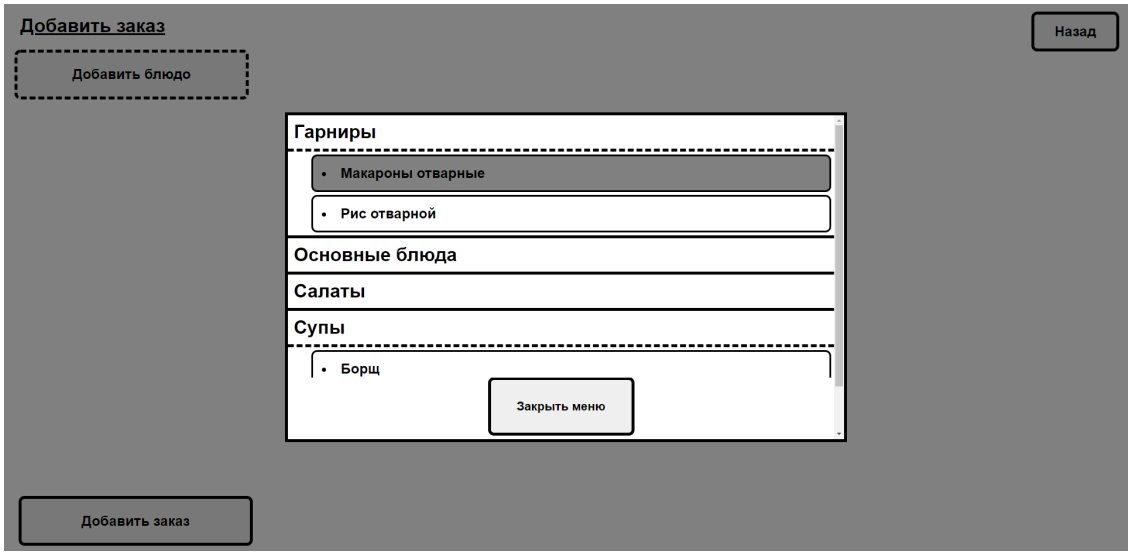

Рис. 15. Интерфейс меню выбора блюд

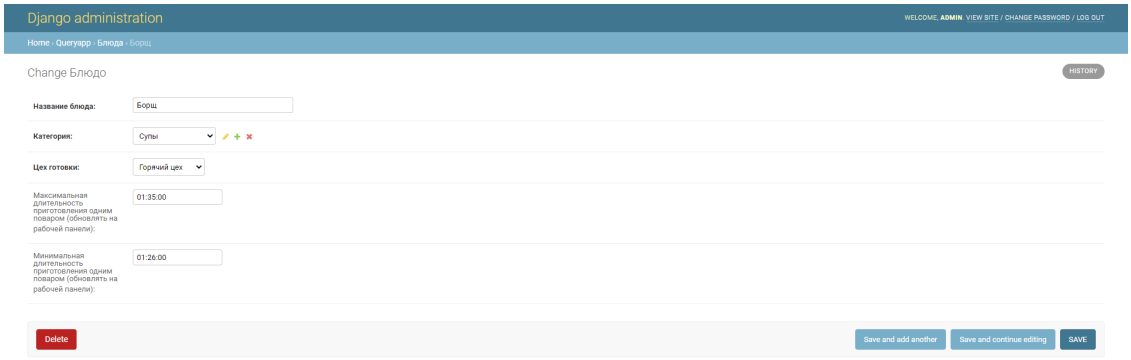

Рис. 16. Интерфейс панели администратора

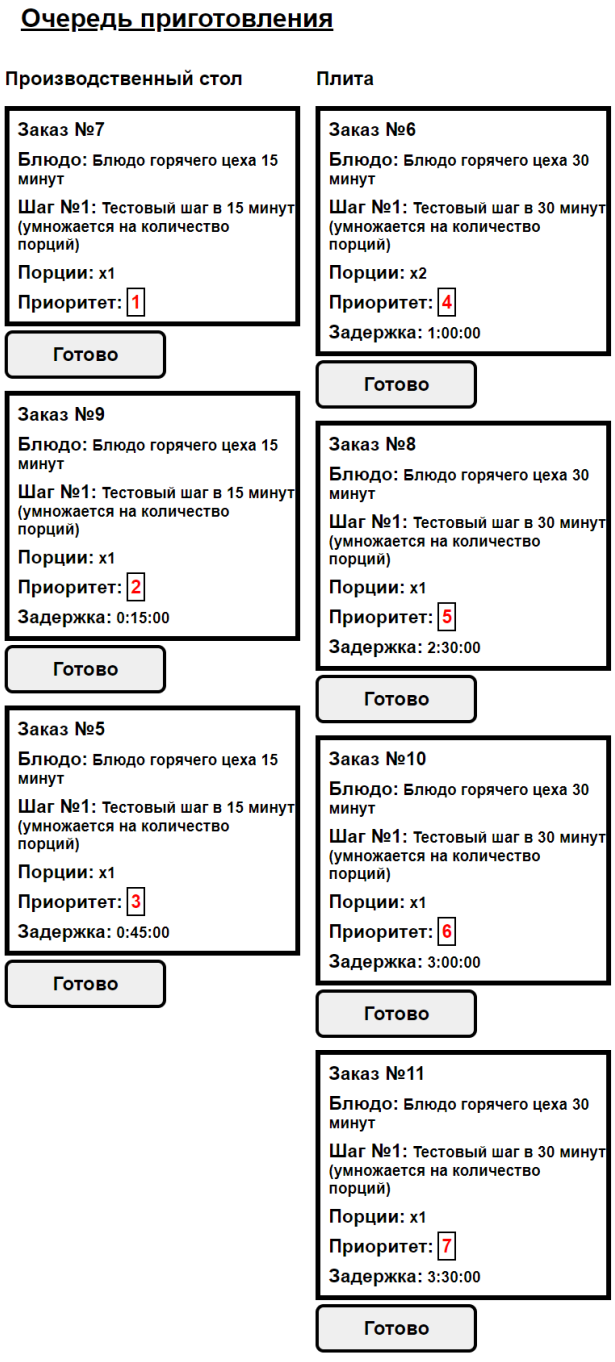

Рис. 17. Оптимизированная очередь шагов горячего цеха

Рис. 18. Оптимизированная очередь шагов холодного цеха