**Содержание**

[Введение 3](#_Toc36909349)

[1 Характеристика предприятия 4](#_Toc36909350)

[1.1 О предприятии 4](#_Toc36909351)

[1.2 Организационная структура 5](#_Toc36909352)

[2 Индивидуальное задание 7](#_Toc36909353)

[3 Обзор инструментальных средств 8](#_Toc36909354)

[3.1 ASP.NET Core 8](#_Toc36909355)

[3.2 Angular 9](#_Toc36909356)

[3.3 Bootstrap 9](#_Toc36909357)

[3.4 MS SQL 10](#_Toc36909358)

[3.5 Visual Studio 2019 11](#_Toc36909359)

[3.6 Visual Studio Code 11](#_Toc36909360)

[3.7 Postman 12](#_Toc36909361)

[4 Программная реализация 13](#_Toc36909362)

[Заключение 22](#_Toc36909363)

[Список литературы 23](#_Toc36909364)

[Приложение А 24](#_Toc36909365)

[Приложение Б 27](#_Toc36909366)

# **Введение**

Производственная практика является одним из важнейших этапов подготовки специалистов.

Целью производственной практики является ознакомление с организационной структурой, производственными процессами и автоматизированными системами предприятия, приобретение профессиональных навыков, изучение информационных технологий, разрабатываемых и используемых на предприятии, приобретение навыков эксплуатации ИТ-систем и их компонентов.

В последние годы люди всех возрастов большую часть своего времени проводят в интернете. Всемирная сеть дает возможность развлечься, научиться чему-нибудь или познакомиться с кем-нибудь. Для большинства из этих целей можно найти специализированное Web-приложение, которое поможет Вам достигнуть поставленной цели с наименьшими потерями времени и средств, не загружая при этом устройство пользователя, ведь все вычисления выполняются на стороне сервера, а страница в браузере не перезагружается полностью.

За годы существования Всемирной паутины состав Web-приложений, выполняемые ими функции, принципы и архитектура их построения претерпели значительные изменения — от простейших средств хранения HTML-страниц до решений, ориентированных на поддержку работы корпоративными информационных систем и их партнеров.

1. **Характеристика предприятия**

ЗАО «Оксаджайл» — динамично растущая компания-поставщик услуг в области IT-аутсорсинга и разработки программного обеспечения на заказ. Предприятие было основано в 2011 году и специализируется на работе с европейскими и американскими клиентами. Пакет услуг компании охватывает все этапы жизненного цикла разработки ПО, включая формирование требований и бизнес-анализ, проектирование архитектуры, создание прототипа, собственно разработку, контроль качества и поддержку конечных пользователей.

* 1. **О предприятии**

Специалисты «Оксаджайл» имеют большой опыт построения программных систем любой сложности: интерактивных веб-приложений, корпоративных SaaS-решений для автоматизации бизнес-процессов, систем управления информационными ресурсами предприятия, а также мобильных приложений для всех основных платформ. Квалификация сотрудников подтверждена международными сертификатами Microsoft Certified Professional Developer (MCPD), Oracle Certified Professional (OCPJP), Mongo M101P, Certified ScrumMaster (CSM), Project Management Institute Certificate (PMI) и International Software Testing Qualifications Board Certificate (ISTQB).

За время работы на рынке «Оксаджайл» сумела блестяще выполнить ряд проектов для компаний с мировым именем. Среди них — Discovery Communications (научно-популярная телесеть, насчитывающая более трех миллиардов абонентов в 220 странах мира) и DeVry University (учреждение высшего образования с 84-летней историей и 55 филиалами в США). Кроме того, компания одной из первых в мире применила революционную технологию WebRTC. Технология реализует коммуникацию в реальном времени непосредственно в окне браузера и не требует специального ПО или плагинов, что значительно упрощает организацию аудио- и видеоконференций.

В компетенции компании находятся:

* Backend (PHP, J2EE, .NET, Ruby, Node.js) и frontend (Bootstrap, AngularJS, Backbone.js, KnockoutJS)
* iOS, Android и кроссплатформенная разработка (PhoneGap, Xamarin)
* WebRTC и online video
* Комплексы управления процессом дистанционного обучения
* Системы для оптимизации бизнес-процессов
* Программное обеспечение для трансляции потокового видео
* Реклама и маркетинг
* Цифровое телевидение и интернет-вещание

Компания разрабатывает программные решения будущего, многие из которых не имеют аналогов в мире, а также работает по следующим ключевым направлениям:

* Online video: создаем высоконагруженные интернет-платформы для видеотрансляций, пишем видеоприложения под все типы устройств, работаем с потоковым видео.
* Real-time Communications: реализуем видеоконференции в Web и поднимаем масштабируемые WebRTC-платформы.
* Computer Vision и Machine Learning: применяем современные достижения в области искусственного интеллекта, экспериментируем с подходами, исследуем неизведанное для нестандартного решения бизнес-задач.

Проекты реализованы в Media & Entertainment, AdTech, EdTech, IoT доменах для таких брендов, как Beeline, Discovery, Vodafone и etc.

* 1. **Организационная структура**

На данный момент в компании работают более 350 сотрудников. В это число входят как руководящие роли, так и позиции разработчиков (рисунок 1.1). На данный момент существует множество отделов разработки, что и обеспечивает широкий спектр оказываемых услуг (PHP, J2EE, .NET, Ruby, Node.js, Bootstrap, AngularJS, Backbone.js, KnockoutJS, iOS, Android, PhoneGap, Xamarin, WebRTC и др.).

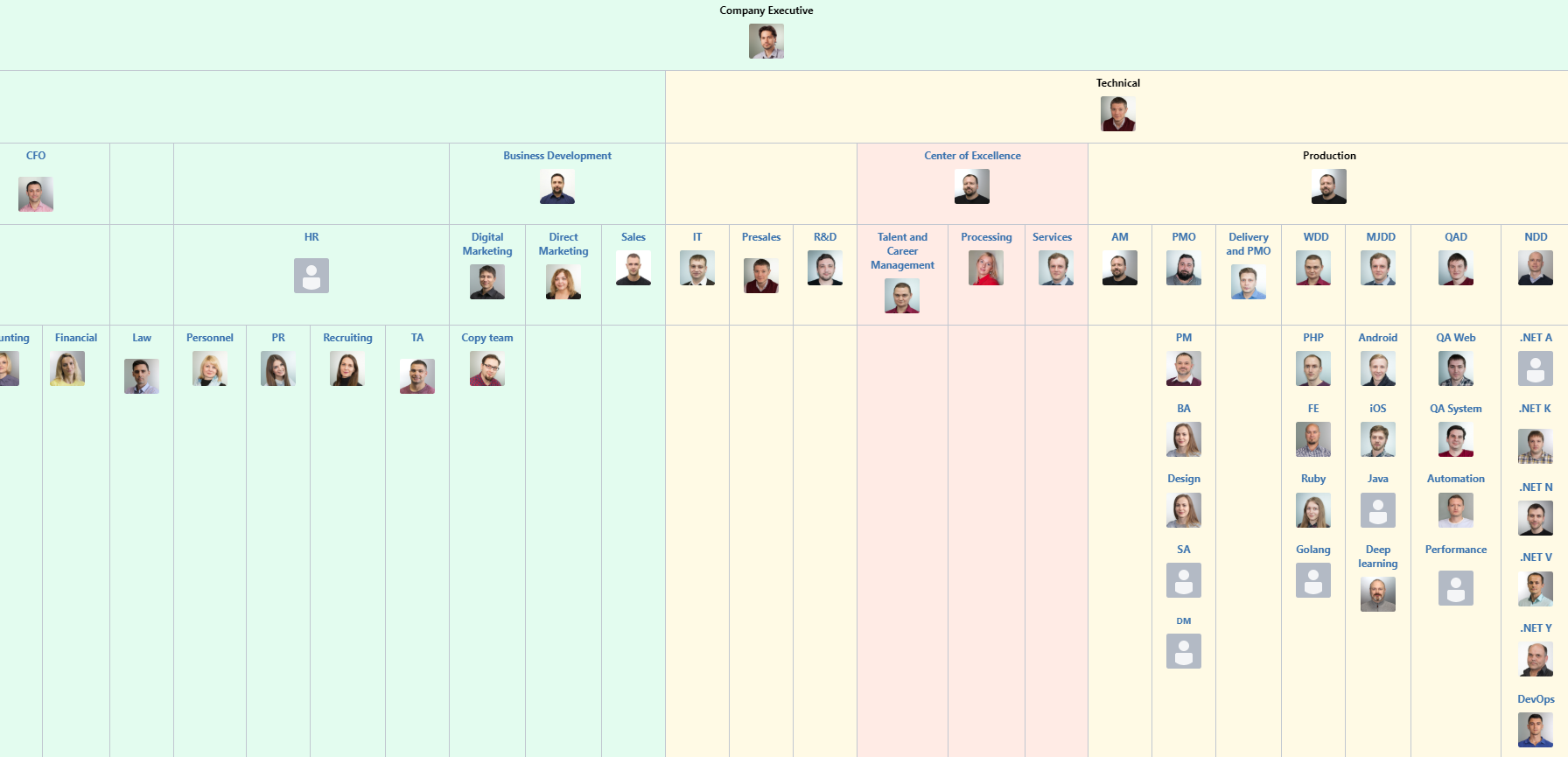


Рисунок 1.1 – Структура ЗАО «Оксаджайл»

Как видно, организационная структура по виду передаваемых полномочий является линейной - во главе каждого подразделения находится руководитель, наделенный всеми полномочиями и осуществляющий единоличное руководство подчиненными ему работниками, сосредотачивающий в своих руках все функции управления. Сам, руководитель находится в непосредственном подчинении руководителя высшего уровня.

Организационная структура управления предприятием характеризуется большим диапазоном управления и небольшим количеством уровней управления. Полномочия и вся информация сосредотачивается в руках руководителя каждого уровня. Это гарантирует быстроту принятия управленческих решений и возможность легкого контроля за работой исполнителей. К преимуществам данной линейной структуры можно также отнести согласованность действий исполнителей, четкая система взаимных связей между руководителем и подчиненным. Главный недостаток такой структуры – высокий уровень требований к компетентности руководителей при выполнении любых управленческих функций, перегрузка менеджеров высшего уровня, огромное количество информации, множественность контактов с подчиненными и вышестоящими.

1. **Индивидуальное задание**

Необходимо разработать Web-приложение с применением стека технологий:

* ASP.NET Core (MVC, Web API);
* ASP.NET Core SignalR;
* Java Script;
* Angular;
* JQuery, Bootstrap, CSS3, HTML5;
* GIT;
* MS SQL;
* ORM Entity Framework Core.

Основные требования к проекту:

* возможность регистрации новых пользователей в роли посетителей и работников заведения;
* возможность регистрации новых родительских организаций и их заведений;
* возможность заполнения информации о заведении, которая доступна к просмотру посетителям;
* возможность создания каталогов меню и наполнения их позициями, как скрытых от посетителей, так и доступных для заказа;
* возможность создания каталогов столиков, доступных для просмотра посетителям;
* возможность забронировать столик на определенный период;
* возможность обновления информации о загруженности зала заказами и бронированием в режиме реального времени;
* возможность создания и обслуживания заказа в режиме реального времени;
* возможность создания коллективного заказа с ведением персонализированного отчета (история заказа, обновление статуса каждого пункта заказа в режиме реального времени, итоговый чек на текущий момент времени);
* возможность вызвать закрепленного за заказом официанта в режиме реального времени с указанием причины (комментария);

1. **Обзор инструментальных средств**

При реализации проекта использовалась платформа ASP.NET Core для разработки back-end.

Фреймворк Angular — свободный набор инструментов для создания сайтов и веб-приложений, который включает в себя HTML и CSS, были выбраны для реализации front-end.

MS SQL Server взят в качестве базы данных.

* 1. **ASP.NET Core**

ASP.NET Core – фреймворк от Microsoft для разработки Веб приложений, который появился вследствие редизайна ранее существующего фреймворка ASP.NET MVC. Нужно понимать, что ASP.NET Core не обязательно должен базироваться на .NET Core. Можно создать ASP.NET Core приложение на основе старого доброго .NET. Ruby также абсолютно открыт. Открыт для бесплатного использования, изменений, копирования и распространения.

ASP.NET Core предоставляет следующие преимущества:

* Единое решение для создания пользовательского веб-интерфейса и веб-API;
* Разработано для тестируемости;
* [Razor Pages](https://docs.microsoft.com/ru-ru/aspnet/core/razor-pages/index?view=aspnetcore-2.2) делает создание кодов сценариев для страниц проще и эффективнее;
* [Blazor](https://docs.microsoft.com/ru-ru/aspnet/core/blazor/index?view=aspnetcore-2.2) позволяет использовать в браузере язык C# вместе с JavaScript, совместное использование серверной и клиентской логик приложений, написанных с помощью .NET;
* Возможность разработки и запуска в ОС Windows, macOS и Linux;
* Открытый исходный код и [ориентация на сообщество](https://live.asp.net/);
* Интеграция [современных клиентских платформ](https://docs.microsoft.com/ru-ru/aspnet/core/blazor/index?view=aspnetcore-2.2) и рабочих процессов разработки;
* Облачная [система конфигурации](https://docs.microsoft.com/ru-ru/aspnet/core/fundamentals/configuration/index?view=aspnetcore-2.2) на основе среды.
* Встроенное [введение зависимостей](https://docs.microsoft.com/ru-ru/aspnet/core/fundamentals/dependency-injection?view=aspnetcore-2.2);
* Упрощенный [высокопроизводительный](https://github.com/aspnet/benchmarks) модульный конвейер HTTP-запросов;
* Возможность размещения в [IIS](https://docs.microsoft.com/ru-ru/aspnet/core/host-and-deploy/iis/index?view=aspnetcore-2.2), [Nginx](https://docs.microsoft.com/ru-ru/aspnet/core/host-and-deploy/linux-nginx?view=aspnetcore-2.2), [Apache](https://docs.microsoft.com/ru-ru/aspnet/core/host-and-deploy/linux-apache?view=aspnetcore-2.2), [Docker](https://docs.microsoft.com/ru-ru/aspnet/core/host-and-deploy/docker/index?view=aspnetcore-2.2) или в собственном процессе;
* Параллельное управление версиями приложения, ориентированное на [.NET Core](https://docs.microsoft.com/ru-ru/dotnet/articles/standard/choosing-core-framework-server);
* Инструментарий, упрощающий процесс современной веб-разработки.
  1. **Angular**

Angular представляет фреймворк от компании Google для создания клиентских приложений. Прежде всего он нацелен на разработку SPA-решений (Single Page Application), то есть одностраничных приложений. В этом плане Angular является наследником другого фреймворка AngularJS. В то же время Angular это не новая версия AngularJS, а принципиально новый фреймворк.

Angular предоставляет такую функциональность, как двустороннее связывание, позволяющее динамически изменять данные в одном месте интерфейса при изменении данных модели в другом, шаблоны, маршрутизация и так далее.

Одной из ключевых особенностей Angular является то, что он использует в качестве языка программирования TypeScript. Но мы не ограничены языком TypeScript. При желании можем писать приложения на Angular с помощью таких языков как Dart или JavaScript. Однако TypeScript все-таки является основным языком для Angular.

Последняя версия Angular - Angular 8 вышла в мае 2019 года.

* 1. **Bootstrap**

Для поддержки интерактивности Web-страниц и улучшения их внешнего вида был выбран фреймворк Bootstrap.

Bootstrap (также известен как Twitter Bootstrap) — свободный набор инструментов для создания сайтов и Web-приложений. Включает в себя HTML- и CSS-шаблоны оформления для типографики, Web-форм, кнопок, меток, блоков навигации и прочих компонентов Web-интерфейса, включая JavaScript-расширения. Его цель — расширение браузерных приложений на основе MVC шаблона, а также упрощение тестирования и разработки.

Фреймворк работает с HTML, содержащим дополнительные пользовательские атрибуты, которые описываются директивами, и связывает ввод или вывод области страницы с моделью.

Основные инструменты Bootstrap:

* Сетки — заранее заданные размеры колонок, которые можно сразу же использовать, например ширина колонки 140 px относится к классу, span2 (.col-md-2 в третьей версии фреймворка), который можно использовать в CSS-описании документа.
* Шаблоны — фиксированный или резиновый шаблон документа;
* Типографика — описания шрифтов, определение некоторых классов для шрифтов, таких как код, цитаты и т. п.;
* Медиа — представляет некоторое управление изображениями и видео;
* Таблицы — средства оформления таблиц, вплоть до добавления функциональности сортировки;
* Формы — классы для оформления форм и некоторых событий, происходящих с ними;
* Навигация — классы оформления для табов, вкладок, страничности, меню и панели инструментов;
* Алерты — оформление диалоговых окон, подсказок и всплывающих окон.
  1. **MS SQL**

В качестве базы данных для приложения был выбран MS SQL благодаря его производительности, удобству написания запросов и взаимодействия с ним стандартной ORM Entity Framework Core.

SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире. Данная СУБД подходит для самых различных проектов: от небольших приложений до больших высоконагруженных проектов.

SQL Server был создан компанией Microsoft. Первая версия вышла в 1987 году. А текущей версией является версия 16, которая вышла в 2016 году, и которая будет использоваться в текущем руководстве.

SQL Server долгое время был исключительно системой управления базами данных для Windows, однако начиная с версии 16 эта система доступна и на Linux.

SQL Server характеризуется такими особенностями как:

* Производительность. SQL Server работает очень быстро.
* Надежность и безопасность. SQL Server предоставляет шифрование данных.
* Простота. С данной СУБД относительно легко работать и вести администрирование.

Для работы с MS SQL базой данных был выбран SQL Server Management Studio (SSMS) – интегрированная среда для управления любой инфраструктурой SQL, от SQL Server до базы данных SQL Azure. SSMS предоставляет инструменты для настройки, мониторинга и администрирования экземпляров SQL Server и баз данных. Используйте SSMS для развертывания, мониторинга и обновления компонентов уровня данных, используемых вашими приложениями, а также для создания запросов и сценариев.

Используйте SSMS для запросов, проектирования и управления базами данных и хранилищами данных, где бы они ни находились - на локальном компьютере или в облаке.

* 1. **Visual Studio 2019**

Для написания кода серверной части приложения использована Visual Studio 2019 — IDE для разработки программного обеспечения, полный набор инструментов для всех этапов разработки. Данные продукты позволяют разрабатывать как [консольные](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) [приложения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), так и приложения с [графическим интерфейсом](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F), в том числе с поддержкой технологии [Windows Forms](https://ru.wikipedia.org/wiki/Windows_Forms" \o "Windows Forms), а также [веб-сайты](https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%B9%D1%82), [веб-приложения](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5), [веб-службы](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) как в [родном](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4), так и в [управляемом](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) кодах для всех платформ, поддерживаемых [Windows](https://ru.wikipedia.org/wiki/Windows" \o "Windows), [Windows Mobile](https://ru.wikipedia.org/wiki/Windows_Mobile" \o "Windows Mobile), [Windows CE](https://ru.wikipedia.org/wiki/Windows_CE" \o "Windows CE), [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework), [Xbox](https://ru.wikipedia.org/wiki/Xbox" \o "Xbox), [Windows Phone](https://ru.wikipedia.org/wiki/Windows_Phone" \o "Windows Phone)[.NET Compact Framework](https://ru.wikipedia.org/wiki/.NET_Compact_Framework) и [Silverlight](https://ru.wikipedia.org/wiki/Silverlight" \o "Silverlight).

Visual Studio включает в себя [редактор исходного кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D0%B0) с поддержкой технологии [IntelliSense](https://ru.wikipedia.org/wiki/IntelliSense" \o "IntelliSense) и возможностью простейшего [рефакторинга кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3" \o "Рефакторинг). Встроенный [отладчик](https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio_Debugger) может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) и дизайнер [схемы базы данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%85%D0%B5%D0%BC%D0%B0_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Visual Studio позволяет создавать и подключать сторонние дополнения ([плагины](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%B3%D0%B8%D0%BD)) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем [контроля версий исходного кода](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D1%8F%D0%BC%D0%B8) (как, например, [Subversion](https://ru.wikipedia.org/wiki/Subversion" \o "Subversion) и [Visual SourceSafe](https://ru.wikipedia.org/wiki/Microsoft_Visual_SourceSafe" \o "Microsoft Visual SourceSafe)), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на [предметно-ориентированных языках программирования](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%B4%D0%BC%D0%B5%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA)) или инструментов для прочих аспектов [процесса разработки программного обеспечения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F) (например, клиент Team Explorer для работы с [Team Foundation Server](https://ru.wikipedia.org/wiki/Team_Foundation_Server" \o "Team Foundation Server)).

* 1. **Visual Studio Code**

Для написания кода клиентской части приложения использована Visual Studio Code — [редактор исходного кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D0%B0), разработанный [Microsoft](https://ru.wikipedia.org/wiki/Microsoft) для [Windows](https://ru.wikipedia.org/wiki/Windows), [Linux](https://ru.wikipedia.org/wiki/Linux) и [macOS](https://ru.wikipedia.org/wiki/MacOS). Позиционируется как «лёгкий» редактор кода для [кроссплатформенной](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D1%8C) разработки [веб](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5)- и [облачных](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BB%D0%B0%D1%87%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F) приложений. Включает в себя [отладчик](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BB%D0%B0%D0%B4%D1%87%D0%B8%D0%BA), инструменты для работы с [Git](https://ru.wikipedia.org/wiki/Git), [подсветку синтаксиса](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D1%81%D0%B2%D0%B5%D1%82%D0%BA%D0%B0_%D1%81%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81%D0%B0), [IntelliSense](https://ru.wikipedia.org/wiki/IntelliSense) и средства для [рефакторинга](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3). Имеет широкие возможности для [кастомизации](https://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D1%81%D1%82%D0%BE%D0%BC%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F" \o "Кастомизация): пользовательские темы, [сочетания клавиш](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%87%D0%B5%D1%82%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BA%D0%BB%D0%B0%D0%B2%D0%B8%D1%88) и [файлы конфигурации](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB_%D0%BA%D0%BE%D0%BD%D1%84%D0%B8%D0%B3%D1%83%D1%80%D0%B0%D1%86%D0%B8%D0%B8). Распространяется бесплатно, разрабатывается как программное обеспечение с [открытым исходным кодом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), но готовые сборки распространяются под [проприетарной лицензией](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BF%D1%80%D0%B8%D0%B5%D1%82%D0%B0%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5).

Visual Studio Code основан на [Electron](https://ru.wikipedia.org/wiki/Electron) — [фреймворк](https://ru.wikipedia.org/wiki/%D0%A4%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA), позволяющий с использованием [Node.js](https://ru.wikipedia.org/wiki/Node.js) разрабатывать настольные приложения, которые работают на [движке Blink](https://ru.wikipedia.org/wiki/Blink_(%D0%B4%D0%B2%D0%B8%D0%B6%D0%BE%D0%BA)). Несмотря на то, что редактор основан на [Electron](https://ru.wikipedia.org/wiki/Electron), он не использует редактор [Atom](https://ru.wikipedia.org/wiki/Atom_(%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80)). Вместо него реализуется веб-редактор Monaco, разработанный для Visual Studio Online.

* 1. **Postman**

Для работы с разработанным серверным приложением и тестированием его возможностей без использования клиентской части приложения использован Postman — очень популярный инструмент, используемый командами разработчиков программного обеспечения во многих компаниях. Удобный при создании веб-сервисов RESTful он также используется многими инженерами QA для тестирования и автоматизации. RESTful API повсюду, но в то же время они начинают усложняться: различные методы HTTP, заголовки, файлы cookie, работа с загрузкой файлов или аутентификацией с помощью ключей api, токенов, OAuth и тд. Postman! позволяет очень быстро создать запрос с требуемым методом и параметрами HTTP, отправить запрос и легко проверить результаты.

Основное предназначение приложения — создание коллекций с запросами к вашему API. Любой разработчик или тестировщик, открыв коллекцию, сможет с лёгкостью разобраться в работе вашего сервиса. Ко всему прочему, Postman позволяет проектировать дизайн API и создавать на его основе Mock-сервер. Вашим разработчикам больше нет необходимости тратить время на создание "заглушек". Реализацию сервера и клиента можно запустить одновременно. Тестировщики могут писать тесты и производить автоматизированное тестирование прямо из Postman. А инструменты для автоматического документирования по описаниям из ваших коллекций сэкономят время. Есть кое-что и для администраторов — авторы предусмотрели возможность создания коллекций для мониторинга сервисов.

1. **Программная реализация**

При разработке системы индивидуальных пользовательских аккаунтов использована технология Identity ASP.NET Core. Пользователь при входе на главную страницу может выполнить вход на сайт или зарегистрироваться (рисунок 4.1.1-4.1.2).

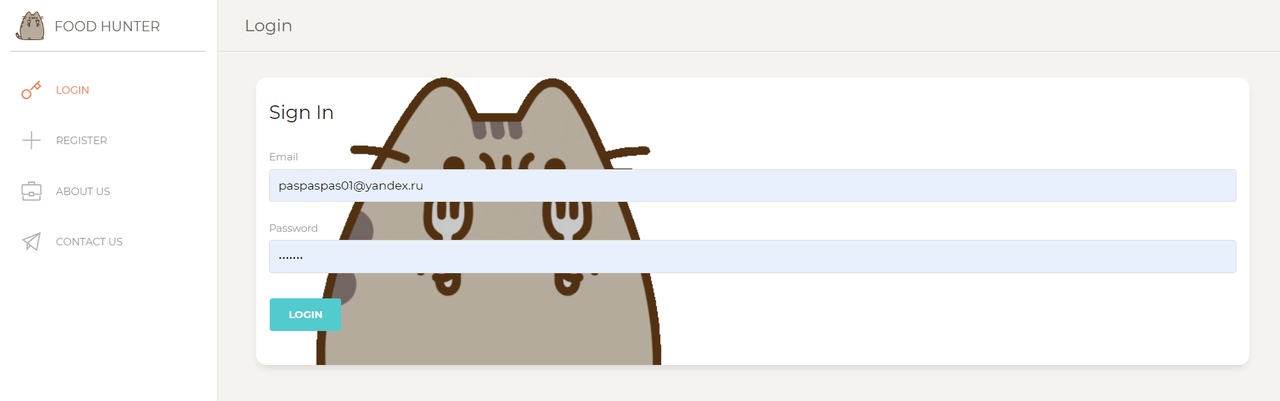


Рисунок 4.1.1 – Страница входа

Для регистрации пользователь должен заполнить поля (фото, email, имя, дата рождения, пол, цель регистрации, номер телефона и пароль).

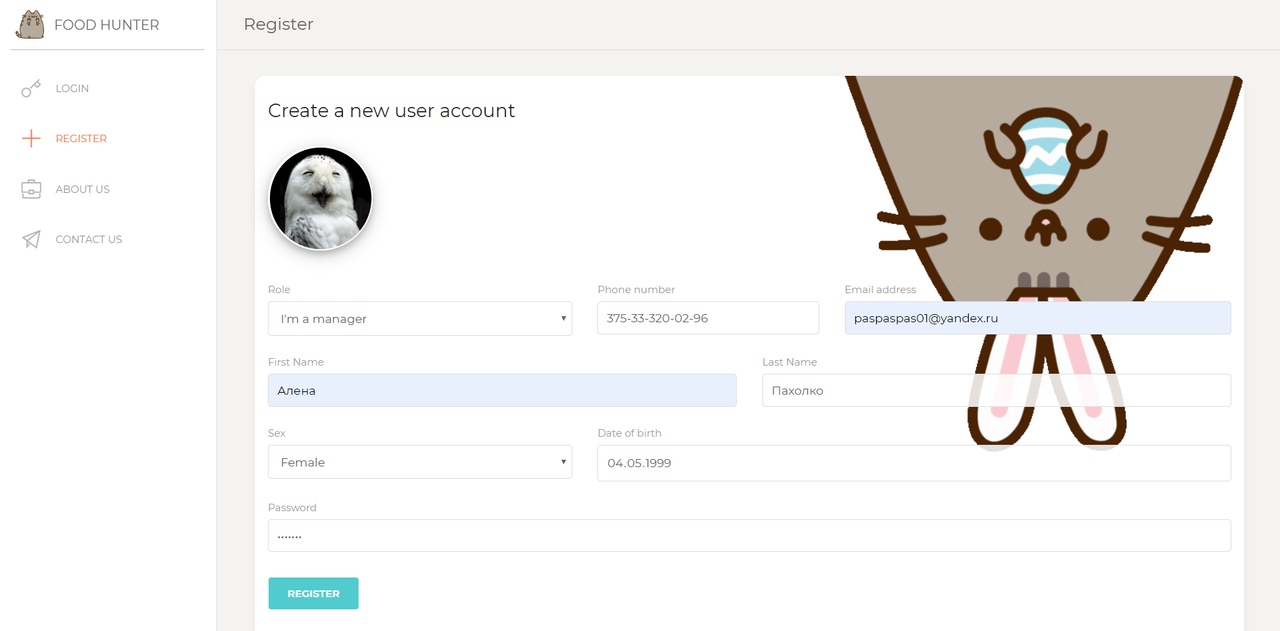


Рисунок 4.1.2 – Страница регистрации

После входа в приложение пользователь уже обладает одной из ролей, которую он выбрал при регистрации: посетитель или работник заведения. Для аутентифицированного пользователя открывается меню управления, либо для посетителя, либо для работника заведения (рисунок 4.2).

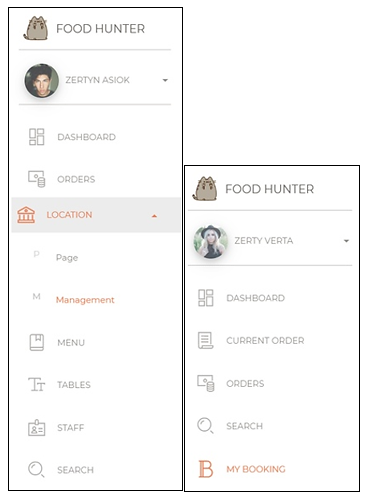


Рисунок 4.2 – Навигационное меню

После входа происходит перенаправление на страницу показателей – отчёты (рисунок 4.3), составленные по данным клиента либо заведения (в зависимости от роли). Так, например, можно увидеть следующие показатели:

* Количество заказов в этом месяце;
* Оплата в текущем месяце;
* Количество клиентов за этот день;
* Средний рейтинг заведения;
* Загруженность столиков по сравнению со вчерашним днем;
* И т.д.

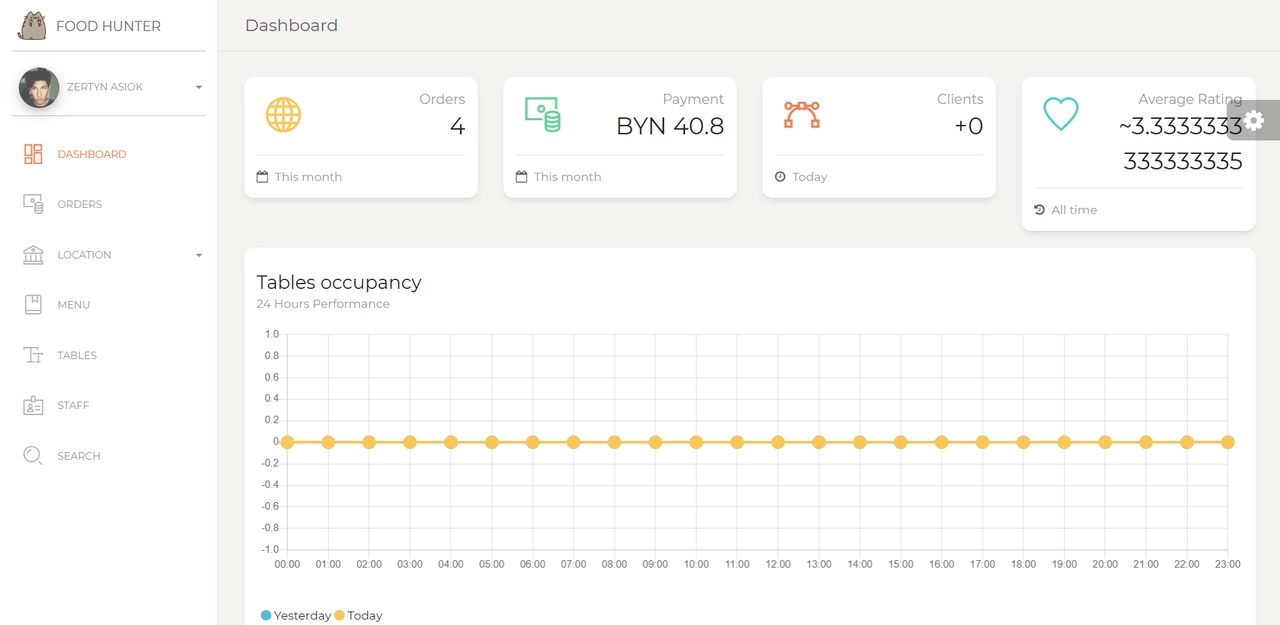
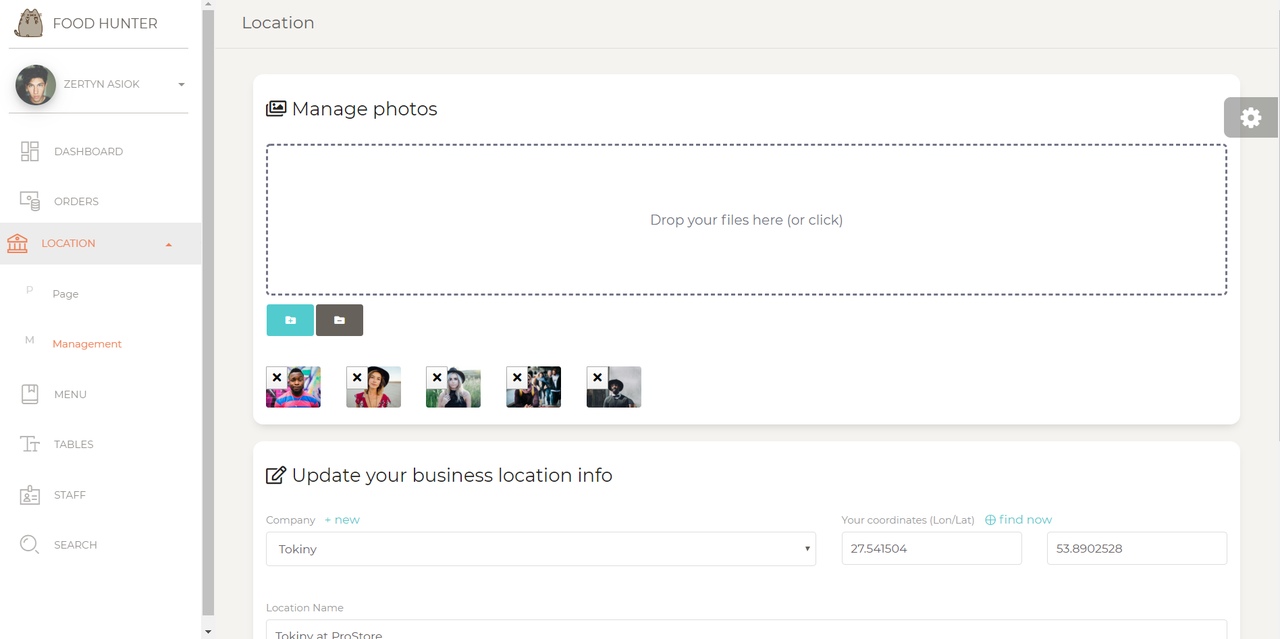


Рисунок 4.3 – Страница показателей

Для управления профилем заведения можно воспользоваться страницей управления заведением (рисунок 4.4). Кроме обновления информации о самом заведении, можно управлять фотоальбомом, доступным для просмотра клиентам. Так, например, можно загрузить план расположения столиков в помещении, фото главного входа, счастливых и довольных посетителей и т.д.).



Страница 4.4 – Страница управления заведением

Далее необходимо добавить книги-меню с помощью специальной формы (рисунок 4.5). Далее, каждую отдельную книгу можно наполнять пунктами меню (рисунок 4.6-4.9). При добавлении нового пункта можно указать наличие цены со скидкой, а также можно скрыть позицию для посетителей, т.е. сам пункт будет недоступен для заказа (что удобно, например, для скрытия сезонных или праздничных предложений). В полях описания и примечания следует указывать информацию о блюде, которая будет полезна посетителям для ознакомления (например, размер порции или особенности вкуса).

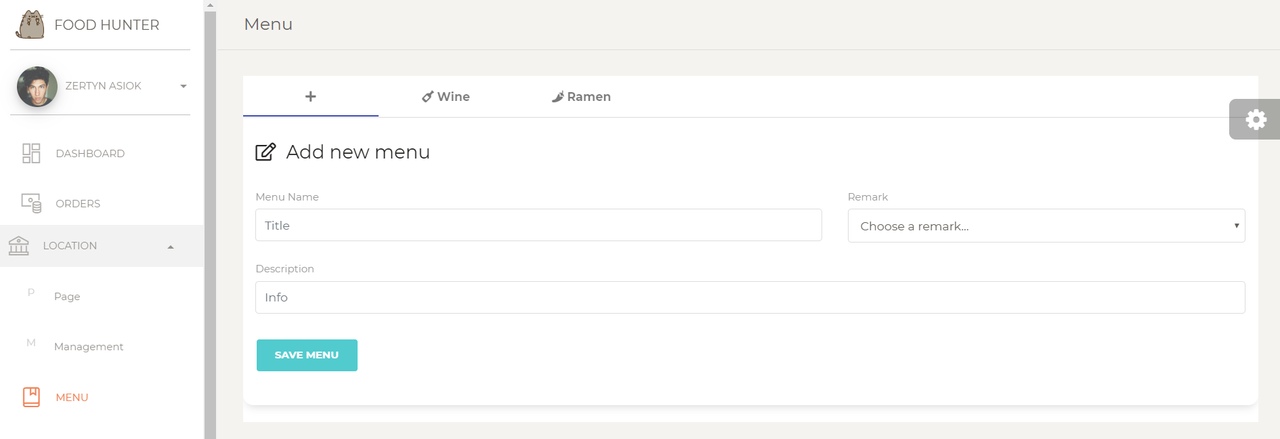


Рисунок 4.6 – Форма добавления книги-меню

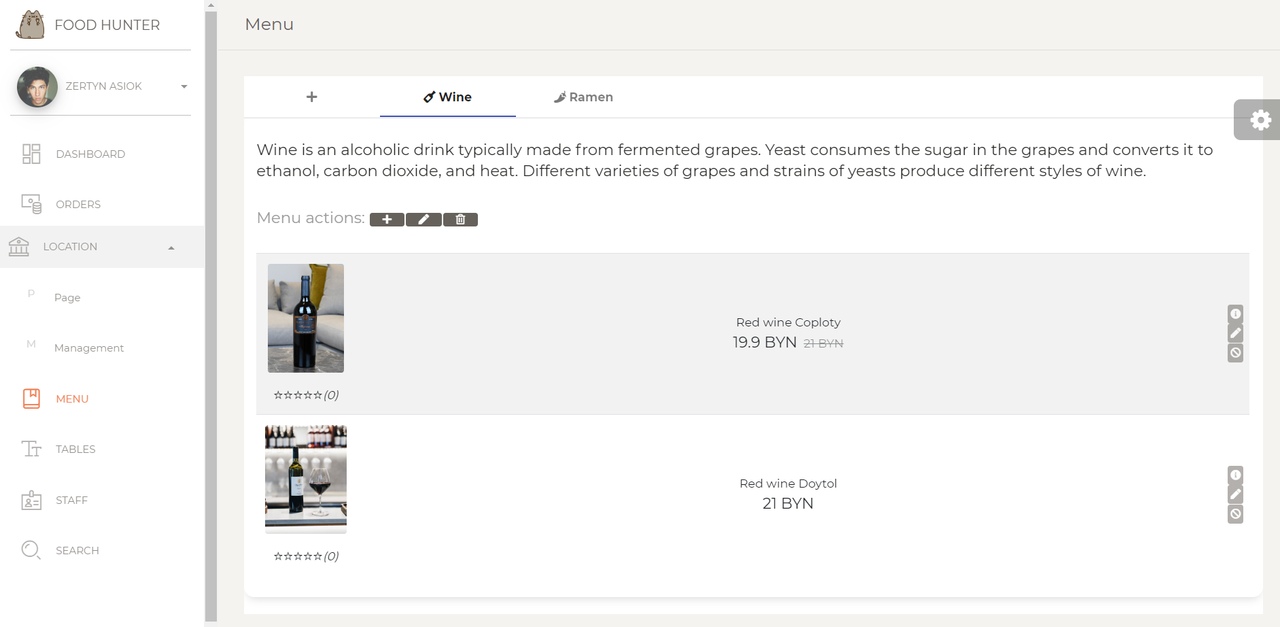


Рисунок 4.7 – Страница управления книгой-меню

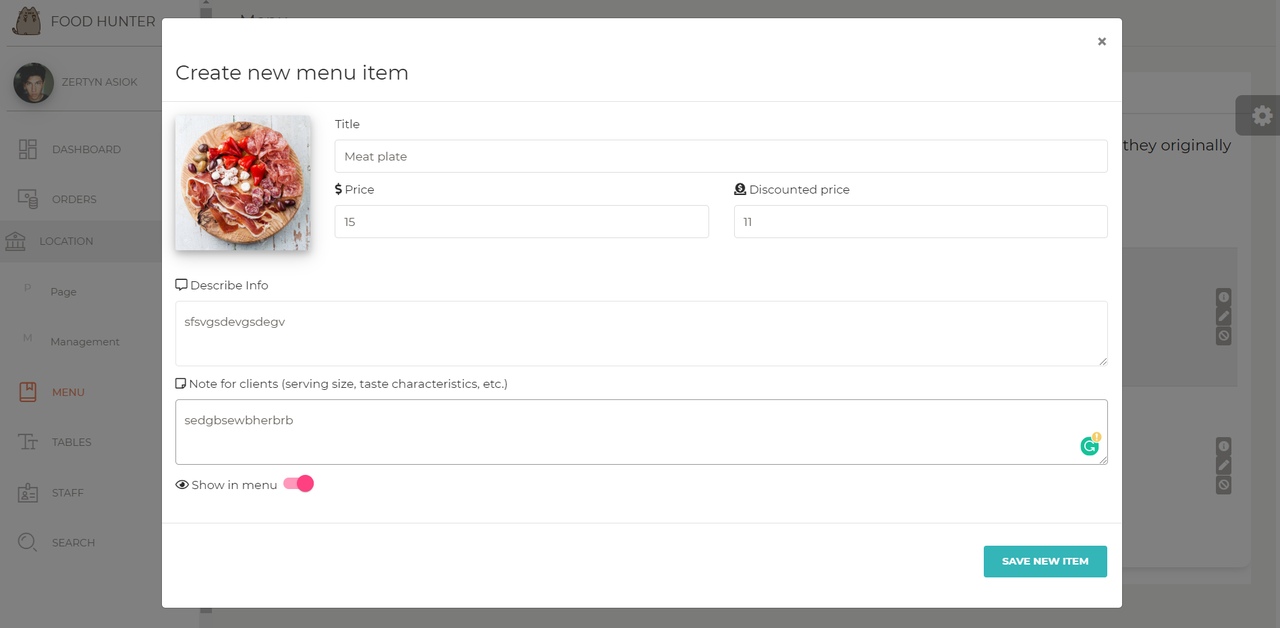


Рисунок 4.8 – Форма добавления нового пункта меню

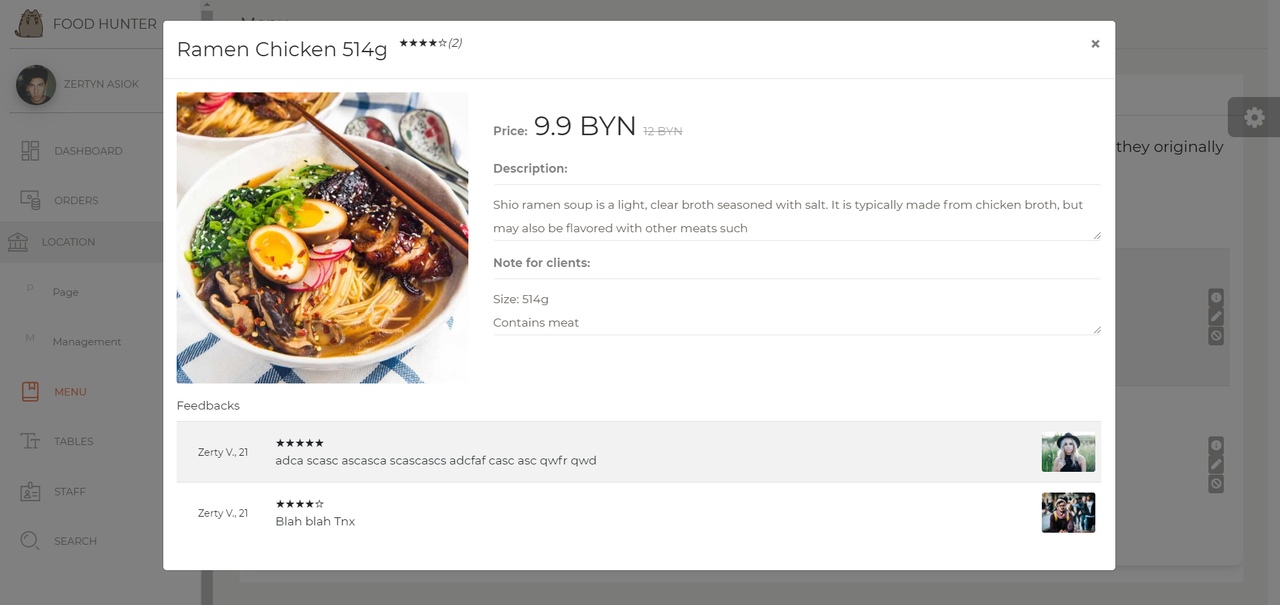


Рисунок 4.9 – Страница просмотра детальной информации о пункте меню

После заполнения всей информации о меню заведения необходимо приступить к добавлению информации о посадочных местах зала обслуживания. На странице управления столиками (рисунок 4.10) можно как добавить новый столик с его описанием (кол-во мест, особенность расположения), так и просмотреть историю бронирования.

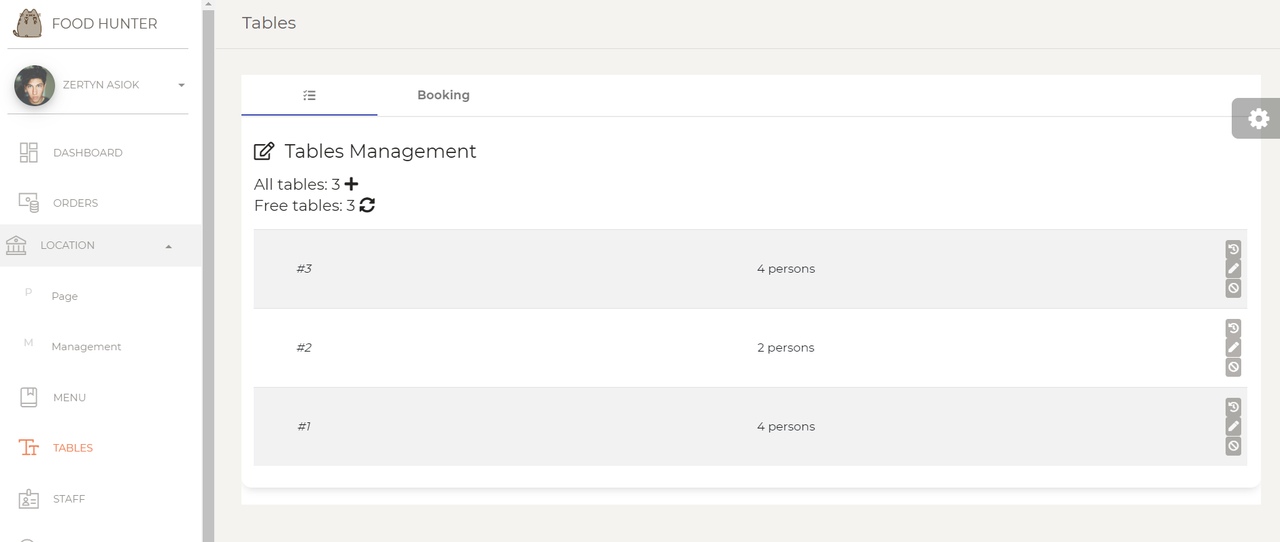


Рисунок 4.10 – Страница управления посадочными местами

Во вкладке «Booking» можно создать новую запись о бронировании столика (рисунок 4.11). Там же можно управлять входящими заявками на бронь, т.е. подтвердить, оставить на рассмотрении или отклонить заявку.

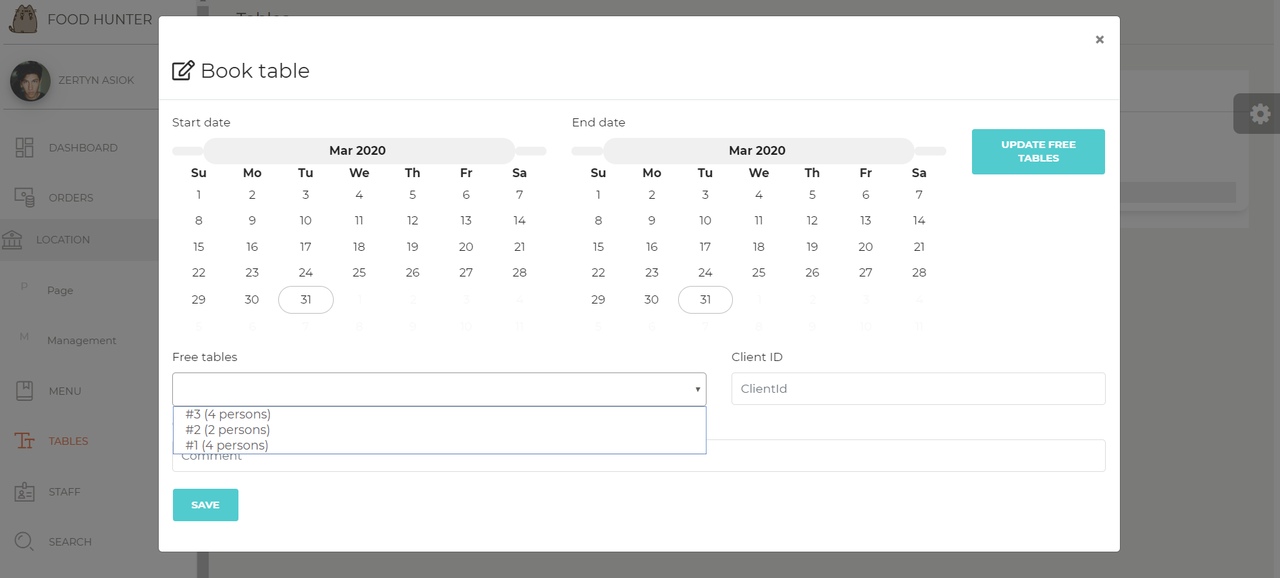


Рисунок 4.11 – Добавление новое заявки на бронирование столика

На странице управления персоналом (рисунок 4.12) можно как добавить новых работников заведения, так и просмотреть список уже зарегистрированных.

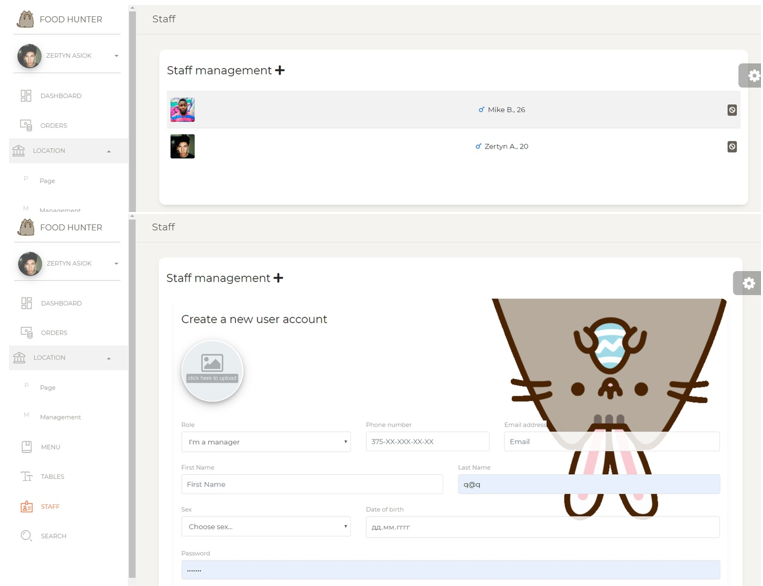


Рисунок 4.12 – Управление персоналом заведения

На странице поиска (рисунок 4.13) можно осуществить поиск по ключевому слову (либо его частью), среди компаний и их заведений.

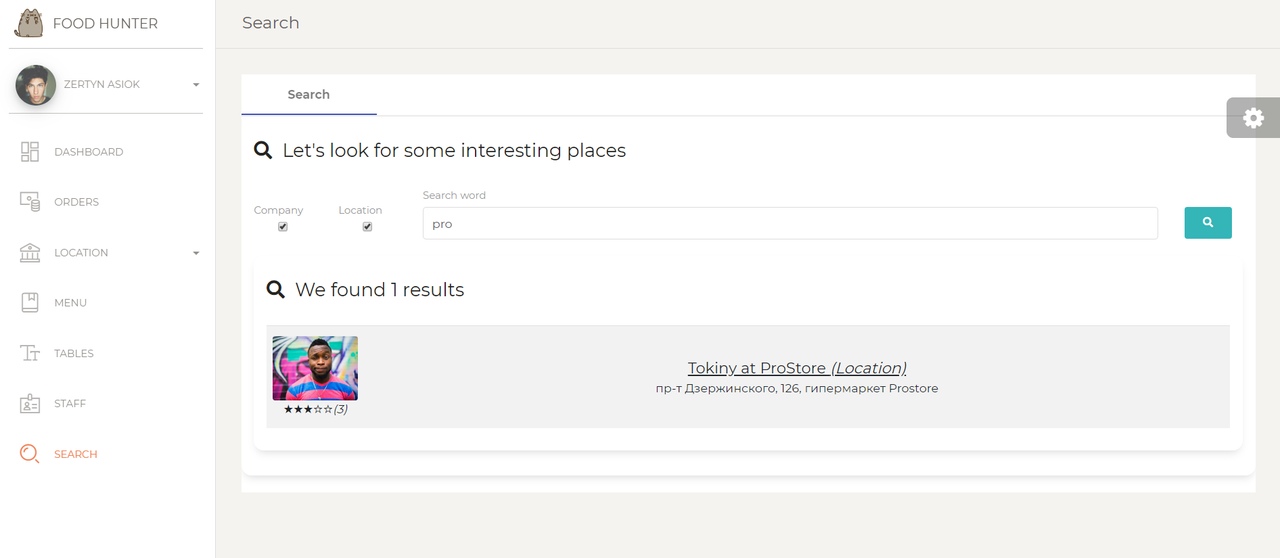


Рисунок 4.13 – Страница поиска

После успешного заполнения всех необходимых форм, создания объектов и регистрации персонала, можно наблюдать полностью готовую персональную страницу заведения, с помощью которой и будет происходит основное взаимодействие с посетителями. Так, на странице можно найти некоторые полезные вкладки, такие как:

* Фотоальбом (рисунок 4.14);
* Книги-меню, где можно заказывать понравившееся блюда в текущий заказ (рисунок 4.15);
* Столики, где можно забронировать столик или начать заказ, выбрав свободный столик (рисунок 4.16);
* Отзывы посетителей (рисунок 4.17);

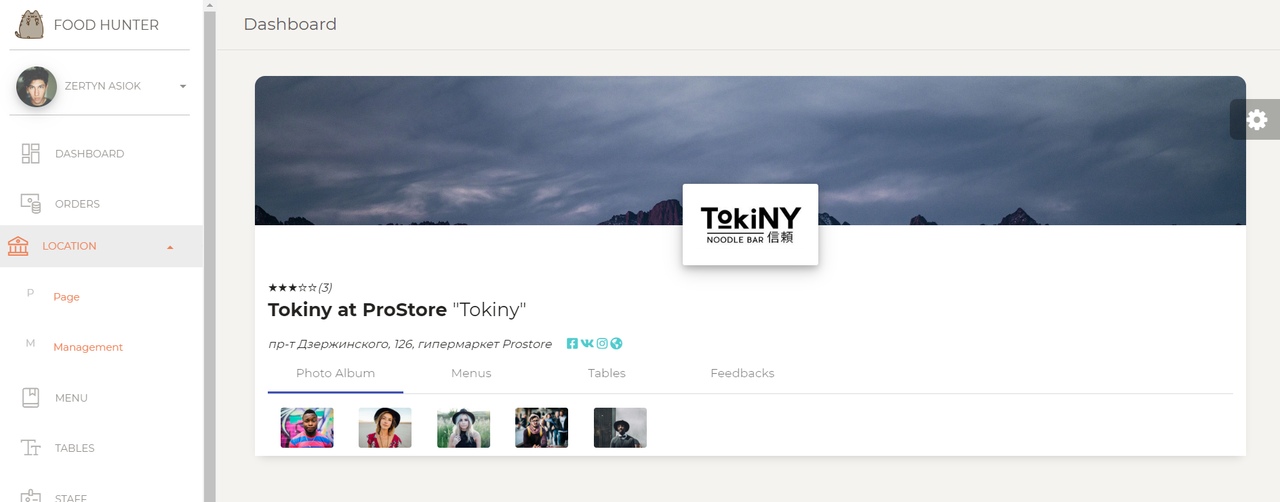


Рисунок 4.14 – Панель фотоальбома

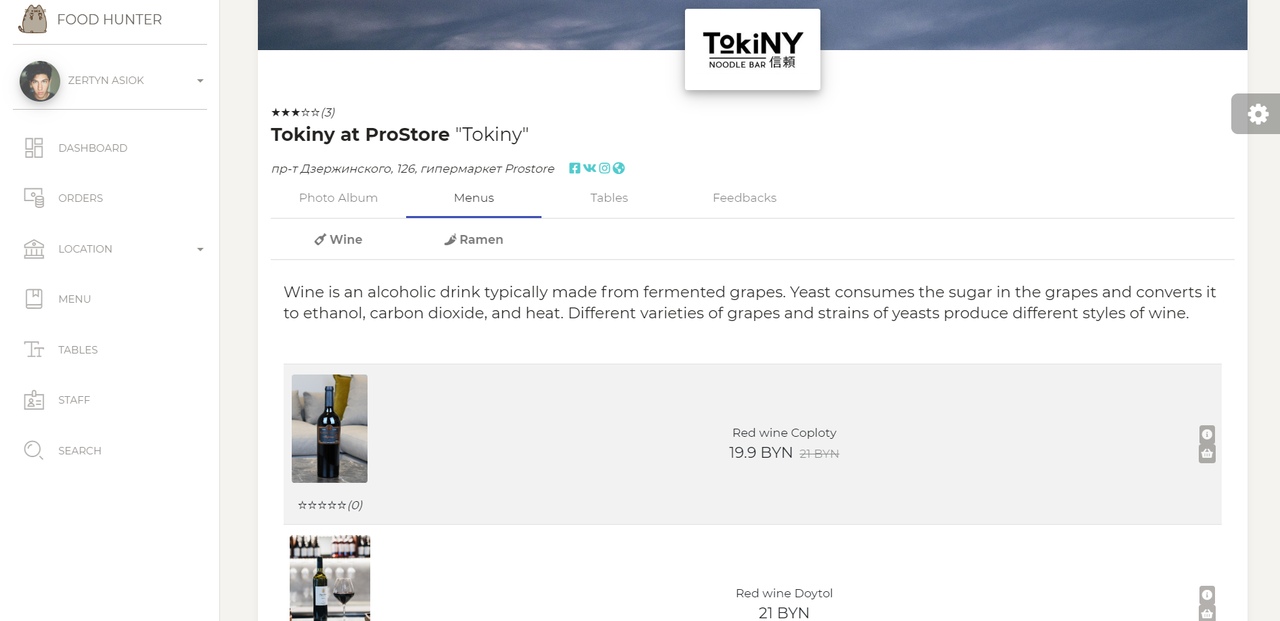


Рисунок 4.15 – Панель меню

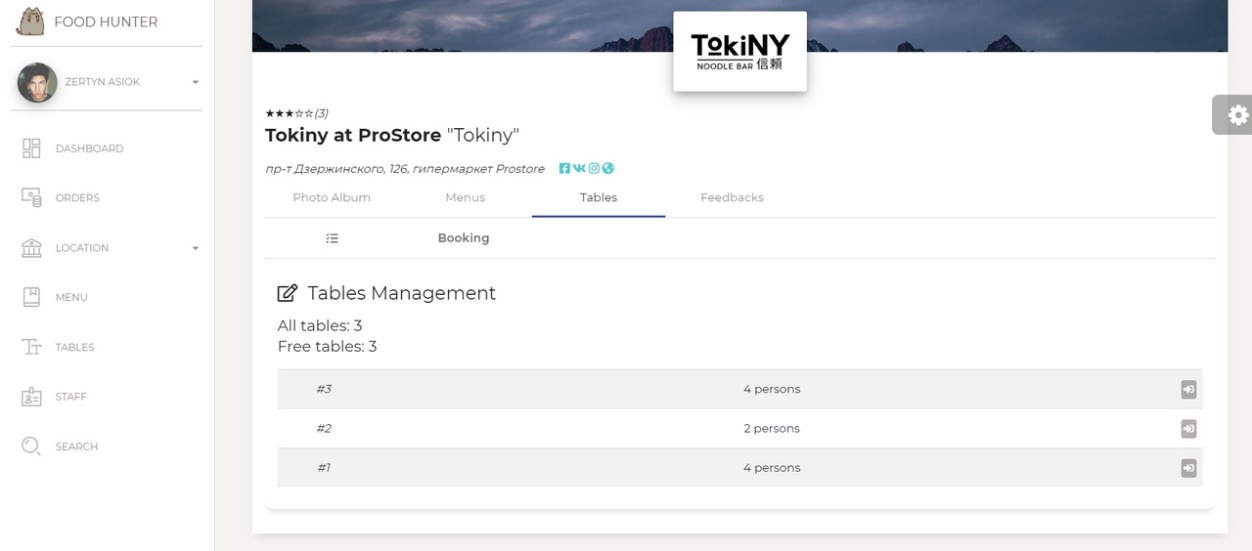


Рисунок 4.16 – Панель столиков

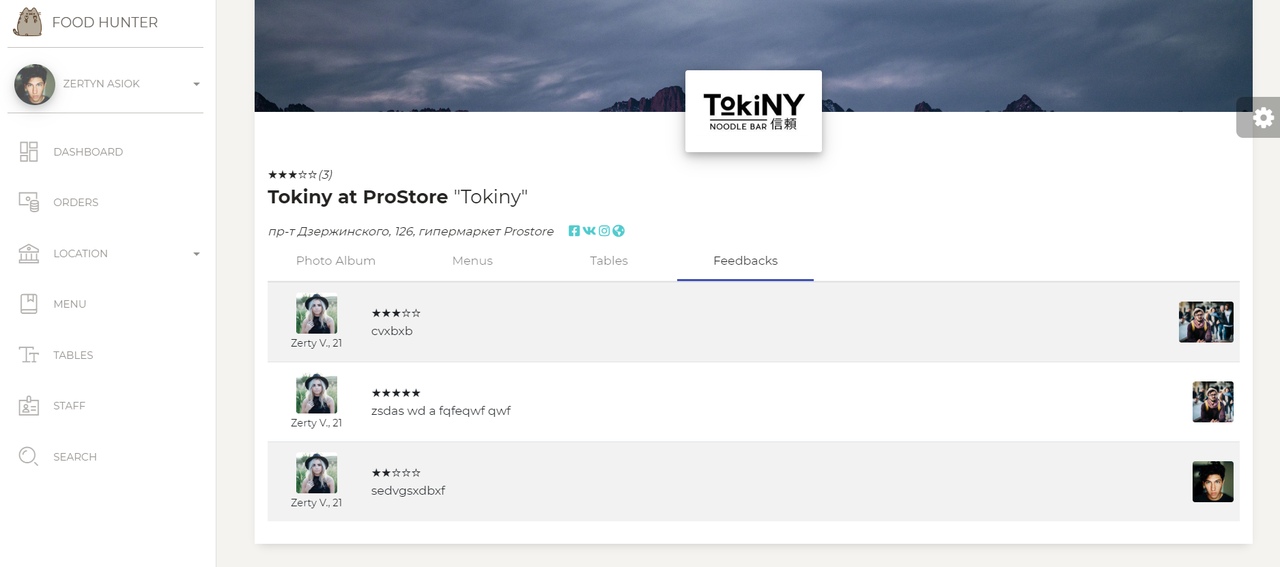


Рисунок 4.17 – Панель отзывов

Для работы с заказами, можно воспользоваться страницей истории заказов, где активные заказы выделены зеленым цветом (рисунок 4.18). Для того, чтобы официант смог управлять заказом, ему необходимо присоединиться к нему, нажав соответствующую кнопку. После этого откроется страница управления самим заказом (рисунок 4.19). На этой странице клиент может следить за обновлением статусов своих заказов (рисунок 4.20), позвать официанта с указанием причины (комментария) (рисунок 4.21), посмотреть свой текущий счет (рисунок 4.22) и оставить отзывы на заказанные блюда и на само заведение (рисунок 4.23).

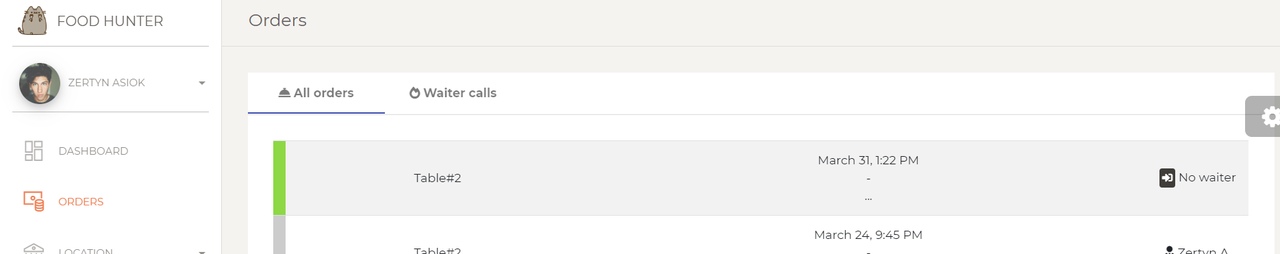


Рисунок 4.18 – История заказов

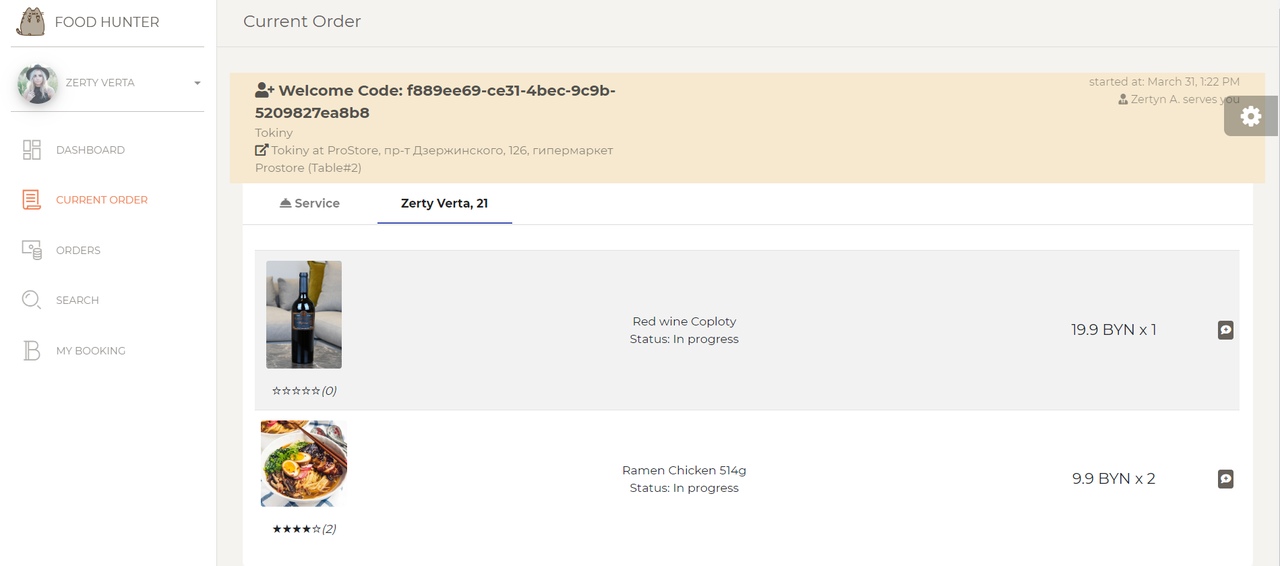


Рисунок 4.19 – Страница управления заказом

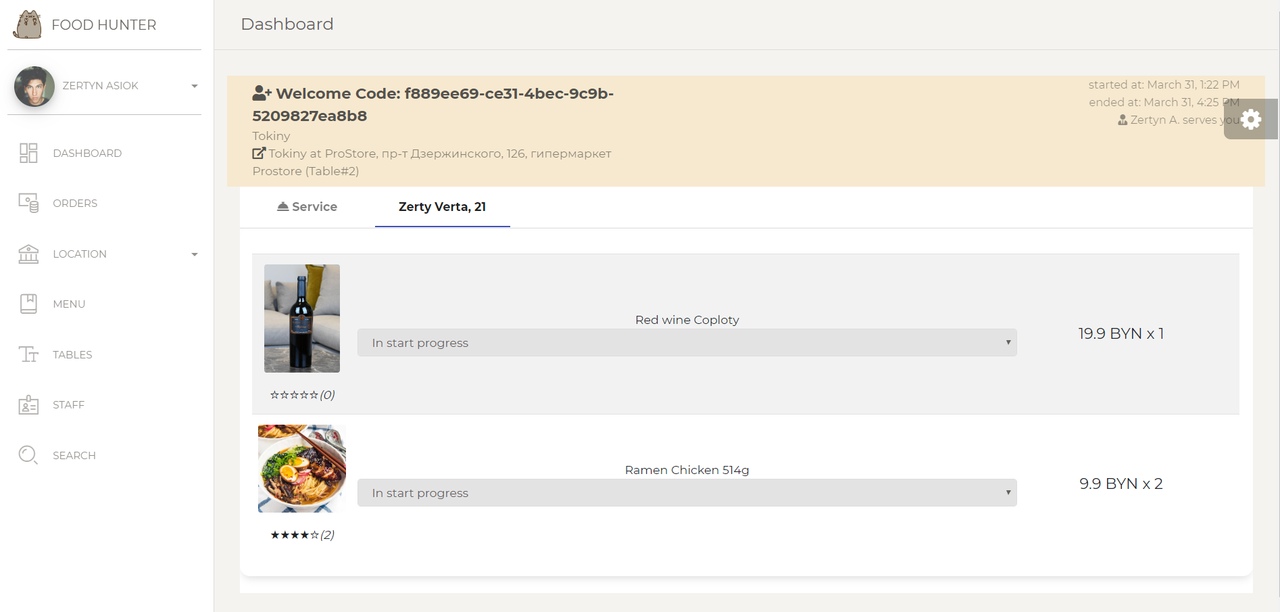


Рисунок 4.20 – Статусы заказанных блюд

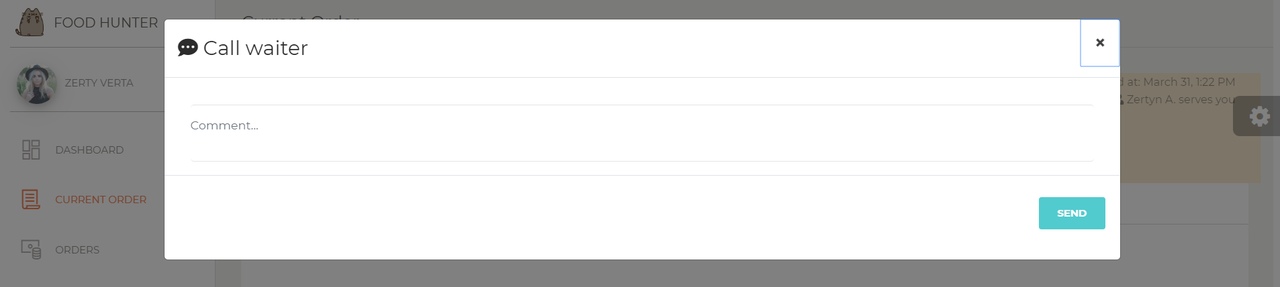


Рисунок 4.21 – Форма вызова официанта

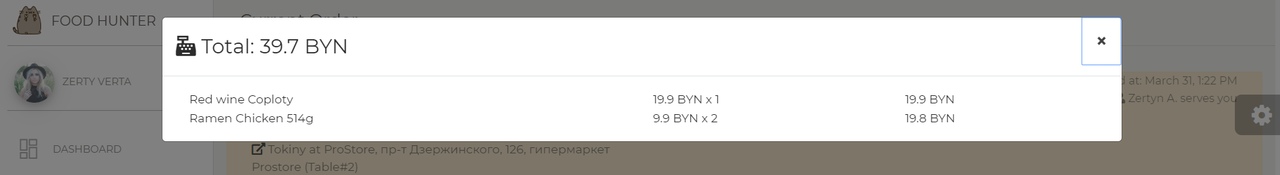


Рисунок 4.22 – Текущий счет

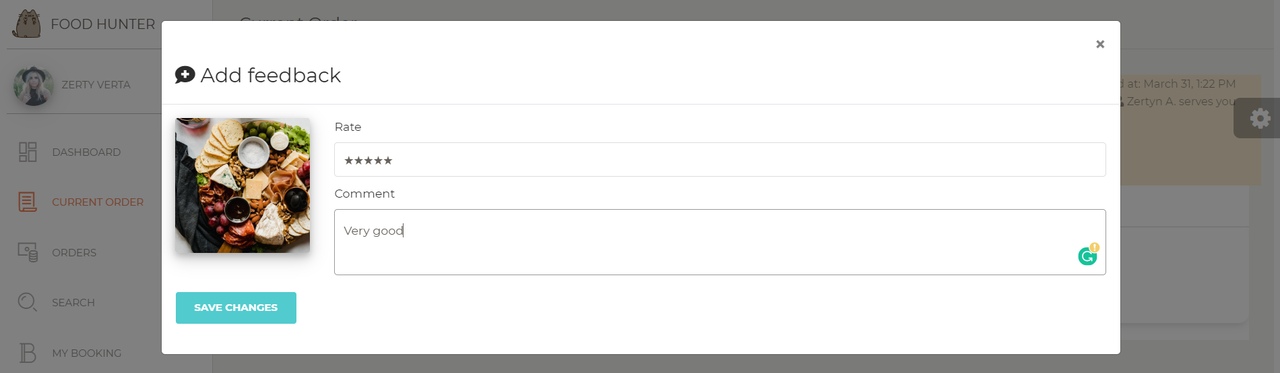


Рисунок 4.23 – Форма отзыва

Для сохранения всех данных использовалась база данных MS SQL. Работа с ней происходила при помощи ORM Entity Framework Core. Архитектурные аспекты реализации серверной части приложения находятся в приложении А. Архитектурные аспекты реализации клиентской части приложения находятся в приложении Б.

# **Заключение**

В результате производственной практики было сконструировано и разработано программное средство в полном соответствии с требованиями.

Для создания приложения были освоены навыки работы с рядом программных продуктов:

* 1. ASP.NET Core и Angular
  2. Bootstrap
  3. База данных MySQL (MySQL Workbench)
  4. IDE Visual Studio 2019
  5. IDE Visual Studio Code
  6. Postman

В процессе прохождения практики в отделе разработки ПО компании ЗАО «Оксаджайл» были поставлены цели изучить организационную структуру предприятия и разработать собственное Web-приложение.

Отдел разработки существует, находясь в базовом, для подобных отделов, состоянии. Несмотря на все имеющиеся недостатки, процесс модернизации и развития предприятия идёт быстро, в связи с большими объёмами задач и вариаций. Рабочих мест достаточно и ими обеспечен каждый работник предприятия, однако есть множество задач, которые, в итоге, решаются вручную. В целом можно сделать выводы, что ЗАО «Оксаджайл» функционирует успешно и является высокотехнологичным, динамично развивающимся предприятием.

За время прохождения практики были закреплены и углублены теоретические знания за время обучения, приобретены навыки самостоятельной практической работы.

Поставленные цели прохождения производственной практики были достигнуты.

# **Список литературы**

1. Сайт компании ЗАО «Оксаджайл» [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://www.oxagile.com/> – Дата доступа : 01.04.2020.
2. Информационный портал [Электронный ресурс]. – Электронные данные. – Режим доступа: <http://ru.wikipedia.org> – Дата доступа : 01.04.2020.
3. Информационный портал [Электронный ресурс]. – Электронные данные. – Режим доступа: <http://it-job.by> – Дата доступа : 01.04.2020.
4. Информационный портал [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://msdn.microsoft.com> – Дата доступа : 01.04.2020.
5. Википедия [Электронный ресурс]. – Электронные данные. – Режим доступа: [http://ru.wikipedia.org/wiki/Web-приложение](http://ru.wikipedia.org/wiki/Веб-приложение) – Дата доступа : 01.04.2020.
6. Википедия [Электронный ресурс]. – Электронные данные. – Режим доступа: <http://ru.wikipedia.org/wiki/Википедия> – Дата доступа : 01.04.2020.
7. MySQL — Википедия [Электронный ресурс]. – Электронные данные. – Режим доступа : <https://ru.wikipedia.org/wiki/MySQL> – Дата доступа : 01.04.2020.

# **Приложение А**

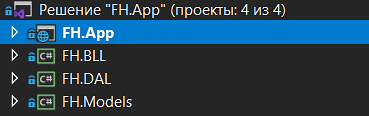


Рисунок А.1 – Общее строение серверной части проекта

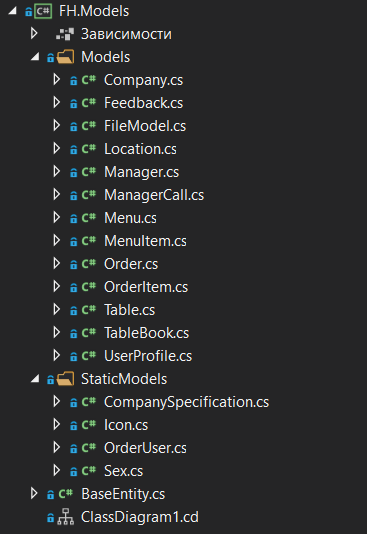


Рисунок А.2 – Строение уровня сущностей

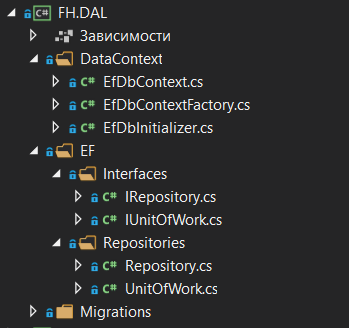


Рисунок А.3 – Строение уровня доступа к данным

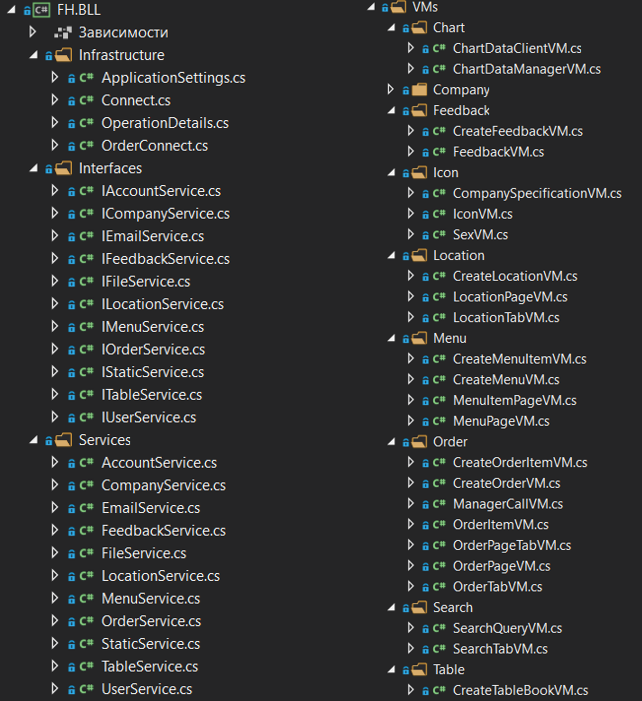


Рисунок А.4 – Строение уровня бизнес-логики

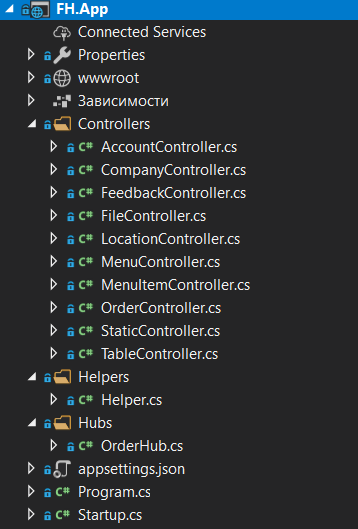


Рисунок А.5 – Строение веб-уровня

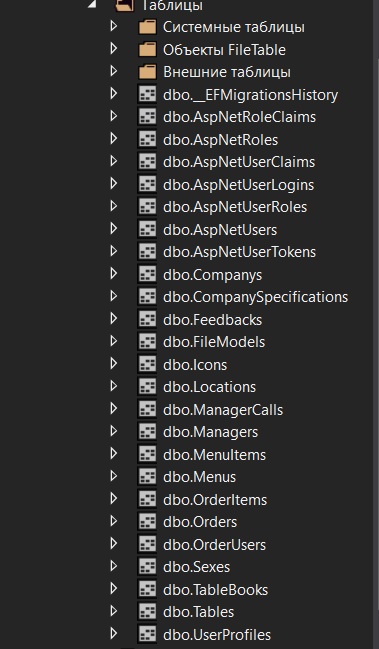


Рисунок А.6 – Таблицы созданной базы данных

# **Приложение Б**

Листинг 1. Таблица маршрутов

|  |
| --- |
| export const AppRoutes: Routes = [    {      path: 'welcome', component: LayoutComponent,      children: [        { path: 'forbidden', component: ForbiddenComponent },        { path: 'oops', component: OopsComponent },        { path: 'login', component: LoginComponent },        { path: 'register', component: RegisterComponent },        { path: 'about-us', component: AboutUsComponent },        { path: 'contact-us', component: ContactUsComponent },      ]    }, {      path: 'dashboard-manager', component: LayoutComponent, canActivate: [AuthGuard],      children: [        { path: 'dashboard', component: DashboardComponent, canActivate: [AuthGuard] },        { path: 'location', component: LocationManagerComponent, canActivate: [AuthGuard] },        { path: 'location/:id', component: LocationPageComponent, canActivate: [AuthGuard] },        { path: 'menu/:id', component: MenuManagerComponent, canActivate: [AuthGuard] },        { path: 'table/:id', component: TablesManagerComponent, canActivate: [AuthGuard] },        { path: 'orders', component: OrderListComponent, canActivate: [AuthGuard] },        { path: 'order/:id', component: OrderManagerComponent, canActivate: [AuthGuard] },        { path: 'feedbacks', component: FeedbackListComponent, canActivate: [AuthGuard] },        { path: 'staff', component: StaffComponent, canActivate: [AuthGuard] },      ]    }, {      path: 'dashboard-user', component: LayoutComponent, canActivate: [AuthGuard],      children: [        { path: 'dashboard', component: DashboardComponent, canActivate: [AuthGuard] },        { path: 'location/:id', component: LocationPageComponent, canActivate: [AuthGuard] },        { path: 'menu/:id', component: MenuManagerComponent, canActivate: [AuthGuard] },        { path: 'table/:id', component: TablesManagerComponent, canActivate: [AuthGuard] },        { path: 'orders', component: OrderListComponent, canActivate: [AuthGuard] },        { path: 'order/:id', component: OrderManagerComponent, canActivate: [AuthGuard] },        { path: 'feedbacks', component: FeedbackListComponent, canActivate: [AuthGuard] },        { path: 'search', component: LocationSearchComponent, canActivate: [AuthGuard] },      ]    },    {      path: '\*\*',      redirectTo: 'welcome/login'    }  ] |

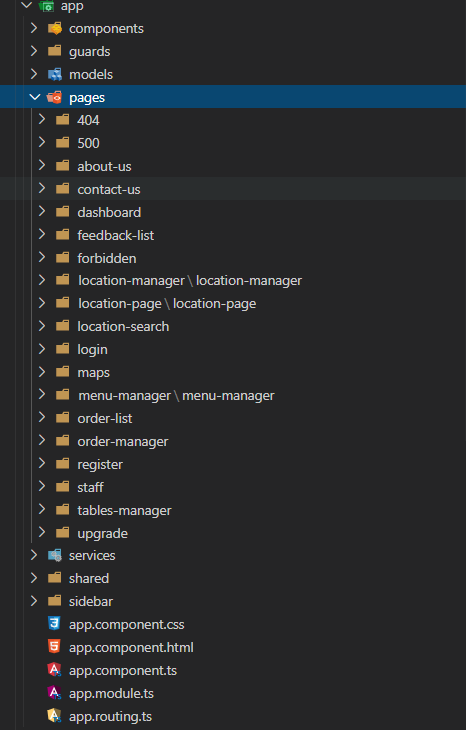


Рисунок Б.1 – Общее строение клиентской части проекта