Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

Розробка клієнт-серверного застосунку з використанням технології NFC

Текстова частина до курсової роботи

за спеціальністю «Інженерія програмного забезпечення»- 121

Керівник курсової роботи
к.ф.-м.н., ст. викладач
Борозенний С.О.
_____
(підпис)
“____” _____ 2021 р.
Виконав студент ІПЗ-4:
Коваленко А.Ю.
“____” _____ 2021 р.

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ
Зав. кафедри мультимедійних систем,
доцент, к.ф.-м.н.
_____ Жежерун О.П.
(підпис)
„____”_____2021 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студенту Коваленко Андрію Юрійовичу факультету інформатики 4 курсу

ТЕМА Розробка клієнт-серверного застосунку з використанням технології NFC

Вихідні дані:
Зміст ТЧ до курсової роботи:
Індивідуальне завдання
РОЗДІЛ 1: Теоретична частина
РОЗДІЛ 2: Опис алгоритму, який пропонується в роботі
РОЗДІЛ 3: Реалізація алгоритму
Список використаної літератури
Програмні додатки (тексти програм)
Висновки
Список використаної літератури
Додатки

Дата видачі „___” _____ 2021 р. Керівник _____
(підпис)
Завдання отримав _____
(підпис)

Календарний план виконання роботи:

Тема: Розробка клієнт-серверного застосунку з використанням технології NFC

№	Назва етапу	Термін виконання	Примітка
1.	Отримання теми курсової	09.09.2020	
2.	Пошук тематичної наукової літератури	20.09.2020	
3.	Ознайомлення з науковою літературою	11.10.2020	
4.	Розбір предметної області	18.11.2020	
5.	Перші спроби реалізації	20.11.2020	
7.	Визначення структури програми	05.01.2021	
8.	Написання 1 частини курсової роботи	25.02.2021	
9.	Написання 2 частини курсової роботи	30.02.2021	
10.	Написання 3 частини курсової роботи	2.03.2021	
11.	Написання висновків курсової роботи	7.03.2021	
12.	Перегляд змісту роботи з керівником	31.03.2021	
13.	Внесення змін до роботи	10.04.2021	
14.	Завантаження курсової роботи	12.04.2021	

Студент Коваленко А. Ю.
Керівник Борозенний С. О.
“_____” _____

Зміст

Анотація

У курсовій роботі наведені теоретичні відомості про архітектурний шаблон програмування Модель-Вид-Контролер(MVC), та його застосування у написані коду на мові програмування ASP.NET. Проведена порівняльна характеристика та визначення переваг і недоліків написання коду використовуючи даний архітектурний шаблон. Представлено опис сфер діяльності та приклади їх застосування. Також опис технології NFC та її сфери застосування. Проведено вивчення предметної області. Наведено опис написання алгоритму та структури програми, та опис структур, використаних API та технологій для його реалізація.

В роботі використовується ASP.NET v.5, EntityFramework, SixLabors.ImageSharp, System.Linq.Dynamic.Core, Westwind.AspNetCore.Markdown.

Ключові слова : NFC, MVC, EntityFramework, ASP.NET.

Вступ

Актуальність теми. У сучасному світі людство намагається автоматизувати більшість процесів своєї життєдіяльності. Сьогодні в часи пандемії від коронавірусу, відбувається процес діджиталізації всього, що є можливим та неможливим. Також актуальним питанням повсякдення є питання екологічності виробництва. Якщо подивитися на будь-яку систему громадського харчування, вона має великі проблеми з так званим «food wasting», або інакше кажучи викидають залишки їжі просто на просто у смітник. Всім відомі Турецькі готелі з системою «Все включено» мають непристойно великі об’єми, що йдуть у смітник, а кращому випадку на корм скоту. Ця проблема є актуальною і для компаній або великих офісних центрів, що харчують своїх співробітників. Людина ідучи на обід, бере піднос та обирає все, що їй заманеться. Кількість страв, що треба приготувати, у такому випадку неможливо знати наперед та точно приготувати. А сьогодні, стіл роздачі їжі ще й важко підтримувати повністю чистим для відповідності норм безпеки щодо коронавірусу.

Мета дослідження – реалізувати систему видачі комплексів харчування, що складаються з кількох страв, крім того система повинна мати можливість вибору та замовленні комплексів. Видача повинна відбуватися за індивідуальними карткам, шляхом прикладання картки до зчитувача, запис даних на картку відбувається таким самим чином.

Досягнення мети передбачає виконання низки завдань:

- вивчення технології NFC;

- дослідження зчитувача що працює з картками;

- опис системи замовлення та вибору меню;

- опис алгоритму роботи зчитувача;

- опис роботи зчитувача та системи замовлень разом

Об’єкт дослідження – технологія ASP.NET за архітектурним шаблоном MVC та з використанням технології NFC.

Предметом дослідження є способи спрощення процесу замовлення та видачі страв, а також роботи зчитувача з віддаленим сервером, переваги його застосування.

Структура курсової роботи така: вступ, три розділи ,список використаної літератури та програмні додатки.

1.Теоретична частина

1.1 Архітектурний шаблон MVC

З самого початку розробка програмного забезпечення виконувалася таким чином, невелика команда програмістів писала сотні і тисячі рядків . Для роботи з даними інколи підключали різні бібліотеки та сервіси, тоді рішення виходило з використанням патерну Document-View. Підтримка та розширення такого коду було дуже не простою задачею, що вимагало чимало ресурсів. Розробники ставали заручниками власного ПЗ, бо додаючи нового розробника, його треба посвятити усі нюанси, який код за що в продукті відповідає, такий підхід був складний також у плані тестування адже модульне тестування стає майже неможливим.

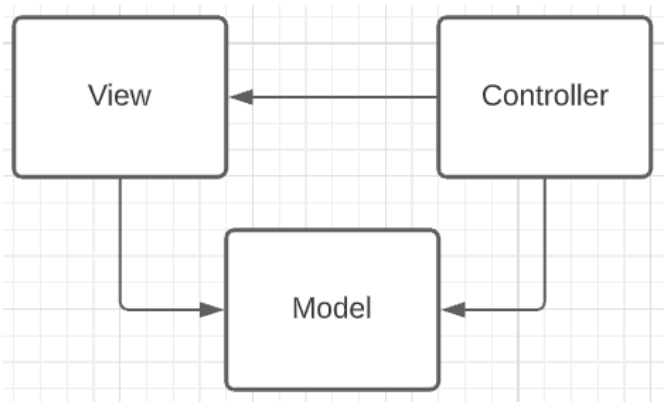
З часом розроблені ПЗ ставали більше і складніше, команду розробників стали доповнювати інші команди, що почали відповідати за певні процеси розробки ПЗ такі як: архітектура, використовуваність, дизайн, та керівництво всього процесу розробки. Таким чином кожна команда відповідає за конкретну свою область розробки: графічний інтерфейс користувача, бізнес-логіка, компоненти. Додалися такі нові відділи розробки як аналіз, тестування, архітектура. Вартість розробки ПЗ зросла в сотні і навіть тисячі разів. Саме архітектура є необхідною для функціонування всіх цих відділів, адже виконує задачу синхронізації різних функціональних областей продукту між собою.

З огляду на мету зменшення трудовитрат на розробку складного програмного забезпечення, припустимо, що необхідно використовувати готові уніфіковані рішення. Адже шаблонність дій полегшує комунікацію між розробниками, дозволяє посилатися на відомі конструкції, знижує кількість помилок.

Патерн - повторювальний архітектурний шаблон, що представляє собою рішення проблеми проектування часто повторюваного контексту.

Архітектурний шаблон Model-View-Controller (MVC) (див. рис. 1) розділяє додаток на три основні групи компонентів: Моделі, Види та Контролери. Таким чином вдається розгалузити більшість проблем. Даний шаблон надає спромогу направляти запити користувачів до контролера, який відповідає за роботу з Моделлю, а саме для виконання дій користувача такі як Get, Post, Delete та отримання результатів запитів. У контролері прописуються подання, яке відображатиметься користувачеві, де будуть відображанні будь-які потрібні користувачу дані моделі.

Рисунок 1



Таке розподілення відповідальності допомагає збільшувати програму та не робити її складною для читання, просте написання коду надає багато переваг, налагоджувати та тестувати щось одне, роблячи так звану декомпозицію (модель, вигляд або контролер), яке має одне завдання на виконання, легше ніж все в одному класі. Складніше оптимізувати, дописувати, апгрейдити, тестувати та налагоджувати код, який має посилання на інші частини, що розподілені по двох або більше із цих трьох груп. Наприклад, графічний інтерфейс користувача та його логіку має таку особливість як те що він може дуже частото зазнавати зміни, ніж логіка контролеру. Якщо код відображення та логіка обробки даних поєднані в одному об’єкті, об’єкт, що містить бізнес-логіку, повинен адаптуватися та

переписуватися кожен раз, коли змінюється інтерфейс користувача. Таке поєднання об’єктів в одному часто призводить до утворення багів і вимагає повторного тестування роботи всієї обробки даних після кожної мінімальної зміни користувальницького інтерфейсу.

Модельні обов'язки

У програмі з використанням архітектурного шаблону Model-View-Controller модель представляє всі данні та стан що зберігає в собі код під час виконання а також можливі операції над ним. Логіка роботи програми має включати в себе модель разом із будь-якою логікою реалізації для збереження стану програми. Якщо модель дуже сильно типізована і не вистачає полів для відображення зазвичай використовують типи модель представлення(ViewModel), які у додаток або замість деяких полів або ж виключають поля які не потрібні для відображення у графічному інтерфейсі користувача. Контролер відповідає за створення та наповнення екземплярів ViewModel з моделі.

Відповідальність представлення

Представлення(View) відповідають за подання вмісту через графічний інтерфейс користувача. У випадку ASP.NET View використовують механізм подання так званий шаблонізатор (Razor) для вписання коду .NET у розмітку HTML. У представленнях повинно бути мінімально написано коду, і будь-яка код в них повинен бути написаний тільки для представлення вмісту. Якщо вам потрібно виконати велику кількість коді у файлах перегляду для відображення даних із складної моделі, тоді скоріш за все вам варто скористатися таким засобами спрощення коду як, ViewModel або шаблону подання для спрощення представлення.

Обов'язки контролера

Контролери - це компоненти програми, які обробляють вхідні дані від користувачів та надають представлення на запити, працюють із моделлю та, зрештою, вибирають подання для візуалізації. У програмі MVC представлення виконує роль лише відображення інформації; контролер обробляє вхідні дані від Post запиту наприклад та взаємодіє з користувачем надаючи йому певні відповіді та інформацію. У шаблоні MVC контролер є початковою, найголовнішою точкою у програмі і відповідає за адресацію до конкретних типів моделей і яке представлення надати або правильніше сказати відрендерити (звідси його назва - він контролює, як додаток реагує на заданий запит).

1.2 Технологія NFC

Near-Field Communication (NFC) - це набір протоколів зв’язку, що допомагає двома електронними пристроями налагодити передачу даних між собою, може працювати на відстані приблизно до 20 см. NFC пропонує низько швидкісне з'єднання з простою установкою, яке можна використовувати для завантаження більш потужних бездротових з'єднань.

Судячи за назви типу передачі даних, NFC забезпечує зв'язок на короткому відстані між сумісними пристроями. Для коректної робити системи достатньо як мінімум одного пристрою для передачі, а інше - для прийому сигналів. Багато пристроїв використовує стандарт NFC і існує такі види як пасивні та активні.

Пасивні пристрої NFC це мітки, картки та інші невеликі пристрої передачі інформації, які надсилають дані на інші пристрої NFC(зчитувачі) які не мають потреби мати власне джерело живлення. Пасивність відбувається за рахунок того, що вони не обмінюються інформацією, що надходить від інших джерел та не мають можливість підключитися до інших пасивних пристроїв.

Активні пристрої мають можливість надсилати або отримувати інформацію та обмінюватися даними з іншим, у тому числі і з пасивними пристроями. Зчитувачі карт громадського транспорту та сенсорних платіжних терміналів є хорошими прикладами використання цієї технології.

Пристрої(у більшості випадків пасивні) NFC можуть виступати як електронні документи, що посвідчують особу, банківські картки та картки ключів. Такі використовуються в безконтактних платіжних системах і дозволяють замінювати або доповнювати мобільні платежі такими системами, як кредитні картки та смарт-картки електронних квитків. Це іноді називають NFC / CTLS (Contactless). NFC можна використовувати для передачі невеликих файлів, таких як контакти.

Технічні характеристики

Вартість мітки 0,10 дол. США

Сумісний з RFID

Орган стандартизації ISO / IEC

Стандарт мережі ISO 13157 тощо.

Частота 13,56 МГц

Швидкість передачі даних 424 кбіт / с

Час налаштування <0,1 с

Споживання струму <15 мА (при читанні)

NFC працює за принципом передачі інформації по радіохвилях. Комунікація ближнього поля - один із стандартів для бездротової передачі даних. NFC використовує технологію створену базі RFID (радіочастотна ідентифікація), у якій для передачі даних використовується електромагнітна індукція.

Використання даної технології суттєво відрізняє NFC та Bluetooth / WiFi. Перший варіант може бути використаний для індукції електрики в пасивні компоненти (пасивні пристрої NFC), а також просто для відправки даних. Тобто пасивні елементи не потребують джерел живлення. Вони отримують живлення від електромагнітного поля, створюваного активним NFC, коли воно входить в зону дії. Однак, технологія NFC не забезпечує достатньої індуктивності для зарядки наших смартфонів, саме бездротова зарядка QI заснована приблизно на тому ж принципі.

Для визначення типу інформації, що буде доступний для обміну між пристроями, стандарт NFC в даний час має три різних режими роботи. Сьогодні, один з найбільш поширених типів використання технології (NFC) в смартфонах - це режим однорангового зв'язку. Це дозволяє двом пристроям з підтримкою NFC обмінюватися різною інформацією один з одним. У даному режимі обидва пристрої перемикаються між активним при відправці даних і пасивним при отриманні.

Режим читання / запису є односторонньою передачею даних. Активний пристрій, можливо ваш смартфон, зв'язується з іншим пристроєм для зчитування інформації з нього. Інші теги як на Huawei Share також використовують дану технологію для початкового налаштування з’єднання пристроїв.

Один з популярніших режимів – це емуляція карти. Пристрій NFC працює у пасивному режимі як інтелектуальна або безконтактна кредитна карта для того щоб здійснювати платежі або підключатися до систем оплати громадського транспорту.

2. Опис системи, який пропонується в роботі

2.1 Загальний опис програми, що потребується

Основним завдання моєї курсової роботи є розробка системи для вирішення проблеми видачі та замовлення їжі у корпоративних компаніях, або готелях, або у будь яких інших місцях. Проблему викидання надлишкової їжі було помічено давно, а з початком карантину більшість готелів перейшло на звичайні папірці (див. рис. 2) де клієнт обирає, що хоче отримати на сніданок.

Рисунок 2

Такі папірці важко обробити вручну, тому виникла ідея автоматизувати даний процес. Також більшість готелів має картки ключі на вхід до номеру, це автоматично може ідентифікувати гостя.

2.2 Що таке CRM система

Перше що приходить в голову, це схожість системи що потребуються, зі звичайною CRM системою у ресторані, кафе або будь-якій точки харчування з доставкою.

Customer relationship management (CRM) - це процес, при якому підприємство або інша організація адмініструє свою взаємодію з клієнтами, як правило, використовуючи аналіз даних для вивчення великих обсягів інформації. [6]

Задача CRM-системи полягає у зборі даних з різних джерел контакту з клієнтами, включаючи веб-сайт компанії, телефон, електронну пошту, чат, маркетингові матеріали та соціальні медіа. Отримана інформація допомагає компаніям дізнатися більше про свою цільову аудиторію та як найкраще задовольнити їхні потреби, розвивати власні продукти, удосконалювати систему продажів та стимулювати таким чином ріст продажів. CRM працює та використовується з минулими, теперішніми або майбутніми клієнтами.

Базовий функціонал, який містить майже будь-яка CRM-система для автоматизації роботи ресторану або кафе:

- автоматичне ведення замовлення;
- можливості оплати як готівкою, так і картою;
- розрахунок і надання бонусів і знижок;
- друк замовлень і рахунків;
- формування звітів по роботі кафе і / або ресторану;
- моніторинг руху товарів;
- контроль над якістю роботи співробітників;
- система захисту від недобросовісного персоналу.

На ринку представлено дуже багато варіантів реалізації такої системи, зі своїми достатками та недоліками. Виділити якісь окремі варіанти доволі важко, але давайте розглянемо деякі приклади.

Food CRM POS - це повністю інтегроване хмарне рішення для індустрії кафе / ресторанів, яке дозволить підприємству бути розумнішим та забезпечити гостям більше індивідуального досвіду. Забезпечте клієнтам більш інтерактивний досвід, замінивши звичайне друковане меню на систему замовлення вкладки, підключену до існуючої POS-системи або кухонного принтера KOT. Автоматизуйте замовлення на доставку додому, замовлення з собою та замовити столи із спеціального мобільного додатку або веб-сайту під вашим брендом. Food CRM оснащений інформаційною панеллю в режимі реального часу з різноманітною інформацією, до якої можна отримати доступ з будь-якого місця та в будь-який час.

Tillypad - це система автоматизації ресторану, кафе, бару чи мережі закладів громадського харчування та розважальних послуг. Автоматизація закладів за допомогою Tillypad вирішує комплекс завдань, пов'язаних з адмініструванням і оптимізацією бізнес-процесів, виробничим, кадровим та управлінським обліком, скороченням витрат і збільшенням прибутковості ресторанного бізнесу будь-якого типу і масштабу.

2.3 Система, що пропонується на розробку

Для вирішення задачі розроблена система повинна перш за все отримати замовлення від клієнтів. Тому адміністратор має можливість додавати страви, які містять у собі опис. А також може додавати комплекси що складаються зі страв. Далі комплекси додаються до меню на день, це відбувається все силами адміністратора, або скоріш кухара.

Потрібна адмін-панель для керування користувачами та перегляду їх витрат. Тут адміністратор може редагувати користувачів, створювати, або приховувати. Видаляти користувачів, не треба. Клієнт може повернутися у будь-який час. І мабуть основне, що відрізняє розроблену систему від інших, це панель керування

прив’язаних карток до користувачів, до кожного користувача може бути прив’язано одну картку. При втраті картки легко записати нову та видати користувачеві. Кратка буде використана при видачі страв.

Користувач може заходити до свого облікового запису та додавати замовлення, на день. У день видачі замовлення клієнт підходить до лінії роздачі, і стає в черг, оператор роздачі бачить що саме треба видати заданому користувачеві.

Також є 2 програма яка відповідає за з’єднання з NFC reader. Ідея наступна основна система хоститься на сервері десь у хмарі, а якщо потрібно працювати з картками тоді програма ставиться локально, та за допомогою JS працює локально з зчитувачем карток.

3. Реалізація системи

3.1 Структура програми та її класів

Програма містить у своєму складі контролери до яких під’єднанні моделі, репозиторії та представлення.

AccountController – відповідає за вхід та вихід користувача до системи, також має методи, що повертають список користувачів, та методи для редагування користувачів. Працює в основному з моделлю HotelUser, UserFinance,UserFinOutcomes, а також з HotelGuests та різними моделями представлення такі як:

- UserRoleViewModel-для редагування ролей користувача
- LoginViewModel-для логіну користувача
- UpdateUserModel-для редагування користувача
- UserFinanceViewModel-для перегляду фінансів користувача

Працює з моделю та бд за допомогою UserManager<HotelUser>, SignInManager<HotelUser>, CompanyUserRepository Звідси повертає представлення:

- LoginModal-вікно логіну,
- Users-сторінка зі скриптами та куди вкладається список користувачів,
- UsersList-список користувачів,
- EditUserModal-вікно редагування користувачів,
- EditCompaniesForUser – вікно надання доступу до компаній користувачу
- RolesForUser-Вікно надання ролей користувачу,
- UserFinanceOfUser-вікно фінансів користувача

HotelUser- є унаслідуванням класу IdentityUser<string> та включає у себе основні поля, що повинен зберігати користувач.

HotelGuests – зберігає у яких компаніях/готелях користувач зареєстрован.

CompanyUserRepository- репозиторій для роботи з бд.

UserExtension -також додатковий клас для роботою з бд користувача.

CategoriesController – відповідає за створення, редагування та видалення категорій страв та комплексів успадковує GeneralController. Працює з моделлю Categories та GenericModelRepository для зв'язку з бд. Має такі представлення:

- EditModal - вікно для редагування Категорій
- Index - основний вид зі скриптом куди потім вставляються інші вікна
- ListItems – список категорій

ComplexController – створення, редагування та видалення комплексів. Працює з моделлю Complex та DishComplex. Використовує IComplexRepository для роботи з бд. Має такі представлення:

- ComplexLineDishes - вікно відображення страв, що включенні до комплексу
- CreateNewCourse - додавання нового курсу страв
- Delete - видалення комплексу
- EditDishes - редагування страв у комплексі
- EditModal - вікно редагування комплексу
- Index - основний вид зі скриптом куди потім вставляються інші вікна
- ListItems - список кмоплексів

DayDishesController – відповідає за редагування меню на тиждень. Працює з моделями DayDish, DayComplex. Працює з бд через репозиторій DayDishesRepository. Має такі представлення

- EditDay - основний вид зі скриптом куди потім вставляються інші вікна, має скрол на тиждень
- EditDayContent – у свою чергу підвантажує компонент меню комплексів DayComplexCompnent на день, у майбутньому можна додати компонент просто страв DayDishCompnent. Компоненти мають дефолтні представлення

DishController – створення, редагування та видалення страв. Працює з моделлю Dis. Використовує IDishRepository для роботи з бд. Має такі представлення:

- Delete – для видалення страви
- DishInLine – для відображення страви у складі комплексу
- EditModal – для редагування страви
- Index – основний вид зі скриптом куди потім вставляються інші вікна
- ListItems – список страв
- SearchView – випадаючий список для пошуку та додавання страви до комплексу

GeneralController – загальний контроллер у який можна передати будь-яку просту модель і тоді не треба розписувати окремо методи.

HomeController – базовий контроллер, повертає перше домашнє представлення Index, а також представлення з Правилами Privacy.

HotelController – відповідає за редагування налаштувань компанії/готелю. Працює з моделлю Hotel та напряму з бд. Повертає представлення Settings.

PicturesController – відповідає за видавання картинок.

ServiceController – це найцікавіший контролер він працює з системою карток, а саме у ньому містяться методу про прив’язання карток до користувача, а також методи що контролюють видачу страв. Працює у зв’язці з репозиторієм

ServiceRepository, з такими моделями ServiceRequest, ServiceResponse, DeliveryQueue. Повертає такі представлення:

- Cards – сторінка зі скриптом підключення до іншого сайту для отримання даних від зчитувача.
- CardsList - список користувачів з зареєстрованими картками
- Index – вікно видачі страв зі скриптом підключення до іншого сайту для отримання даних від зчитувача.
- UserCardsDetails – вікно з інформацією про картку.

UserDayDishesController – контролер що відповідає за замовлення користувачів. Працює з моделями UserDay, UserDayComplex, UserDayDish. Для роботи з бд використовує UserDayDishesRepository. Працює з такими представленнями:

- Delete – видалення замовленого комплексу
- EditComplex – редагування замовленого комплексу
- EditUserDay – збирає в собі компоненти на день замовлення UserDayOrderedComponent – кошик замовлених комплексів/страв UserOrderedDay, UserDayComplexComponent – доступні комплекси на замовлення містить варінти вибору типу замовлень 1 комплекс чи декілька в залежності від компанії UserDayComplexViewModel, UserDayDishComponent – доступні страви на замовлення UserDayDishViewModel
- Index – головна сторінка, що зберігає скрипти та сюди підвантажуються інші дані.
- WeekReport – звіт замовлень на тиждень використовує WeekReportModel

Також є окремі додатки до програми, які знаходяться в папці Core

CompanyExtension - повертає пошту всіх користувачів, готелю/компанії

ControllerExtensions - потрібен для роботи зі сторінками будь-якої моделі

CustomClaimsPrincipalFactory - створення користувача, та переключення між компаніями.

DBCache зберігає компанію в якій працює користувач.

HotelBasePage відповідає за рендеринг базової сторінки _Layout, LoginPartial, ValidationScriptsPartial, Footer, _ViewImports, _ViewStarts, Error.

LinqExtension – допомагає в написанні запитів до бази даних шляхом переформатування Linq в SQL.

MakeOrdersPaymentTask – списання грошей з рахунку та товарів після закінчення дня.

RazorExtensions – додаток для спрощення написання коду у представленні.

SharedViewLocalizer – дозволяє працювати сайту на багатьох мовах.

SortControlBuilder – контроль полів за якими можна сортувати списках об’єктів.

SQLScriptExecutor – додає всі скрипти до бд в StoredProcedure.

UIOption – вибір локалізації

UserExtension – повертає з кешу айді користувача, компанію з якою працює та задає ролі.

У папці Data зберігаються класи для роботи з EntityFramework - AppDbContext правила бд, DbInitialzer початкове створення бд.

Також є друга програма задача якої є робота з NFC зчитувачем запис/читання та передача даних. Програма містить у собі такі класи.

CardManagerEvent – клас, що реагує на підключення рідеру та клієнтів до сокету.

CardMessageHandler – реагує на повідомлення зчитувача.

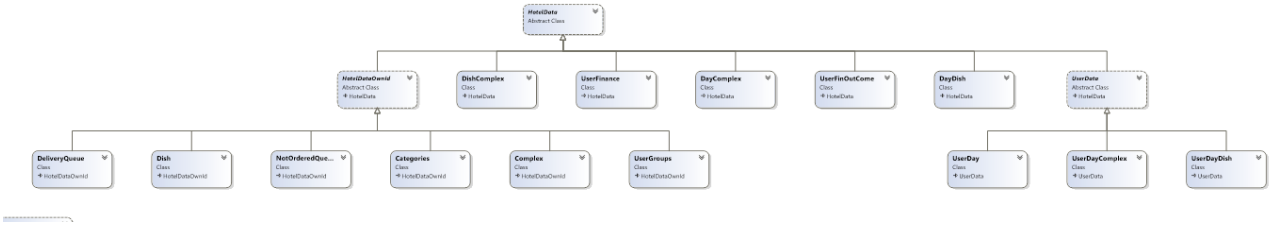
ExtensionUtils - допоміжний клас для конвертації стрічки в байти та навпаки.

GeneralAuthenticate – модель аутентифікації.

MifareCard – запис та зчитування блоків картки.

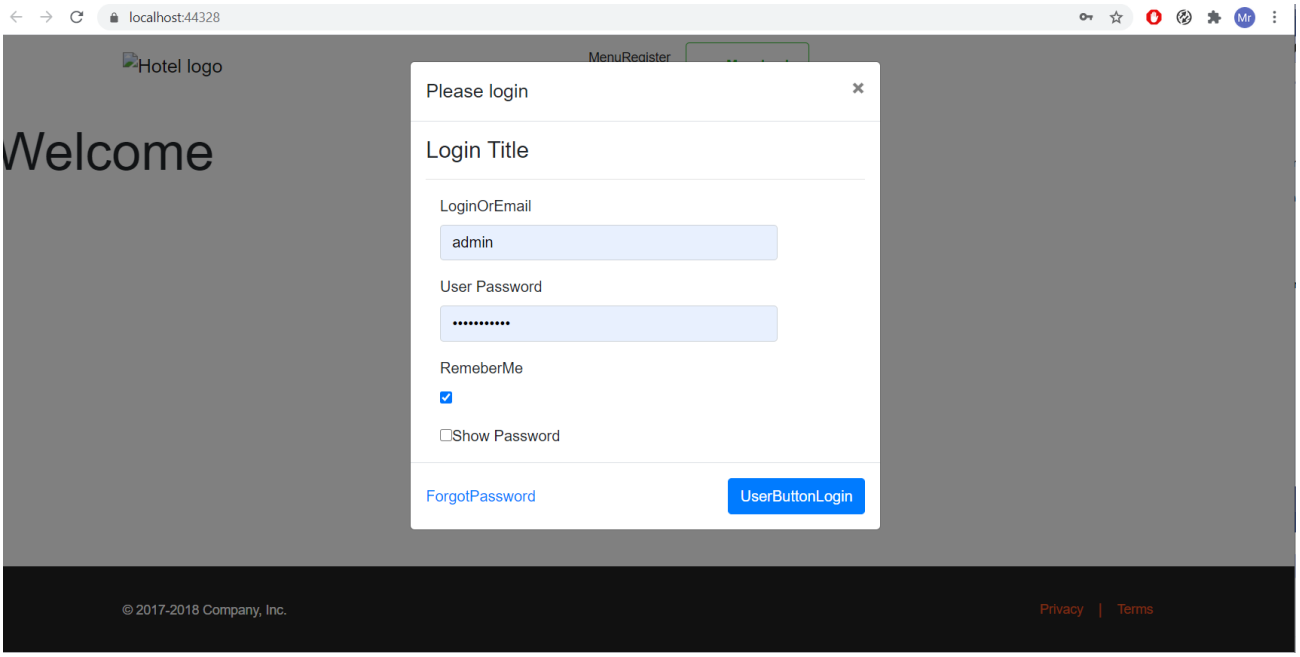
NFCSocketMessageResponse – модель відповіді по сокету.

3.2 Діаграма класів

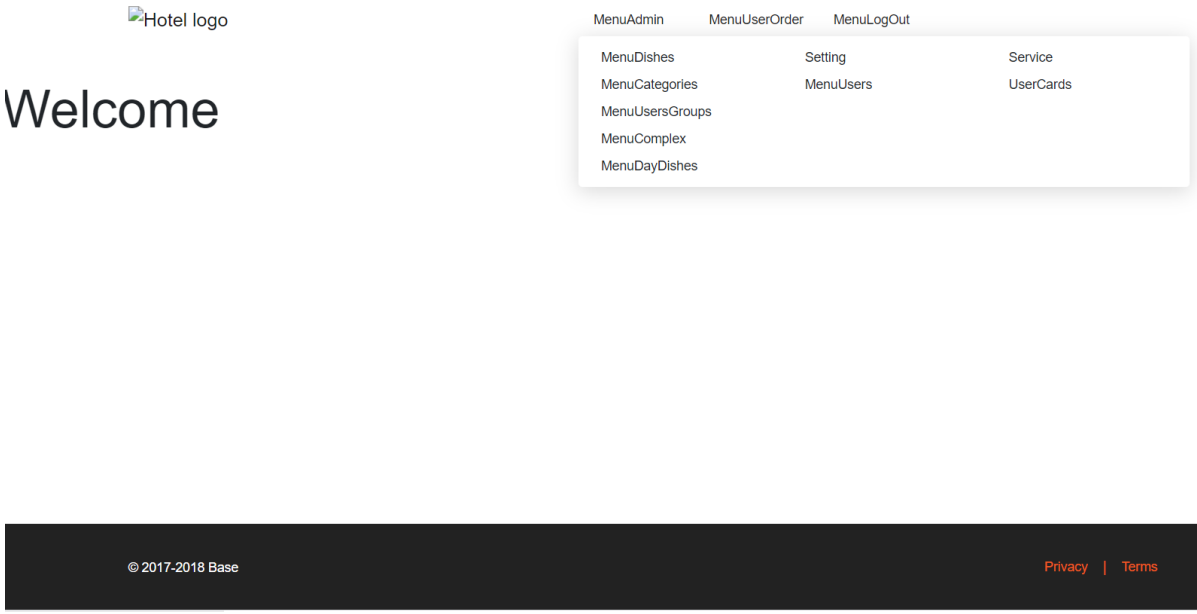


3.3 Загальний опис роботи програми та її компонентів

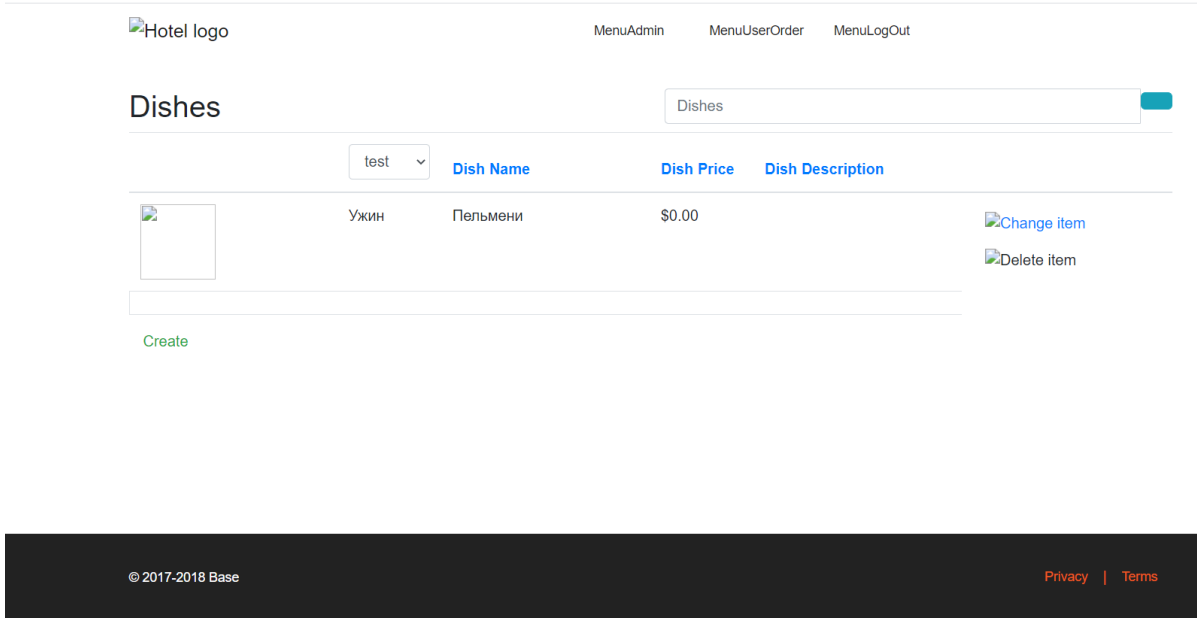
Потрапляючи на сайт, нам перш за все треба залогінитися, для доступу до повного функціоналу. Тому тиснемо кнопку Логін і нам з’являється вікно логіну. Вводимо сюди нашу пошту або логін, та пароль. Після входу в кеші зберігається токен.



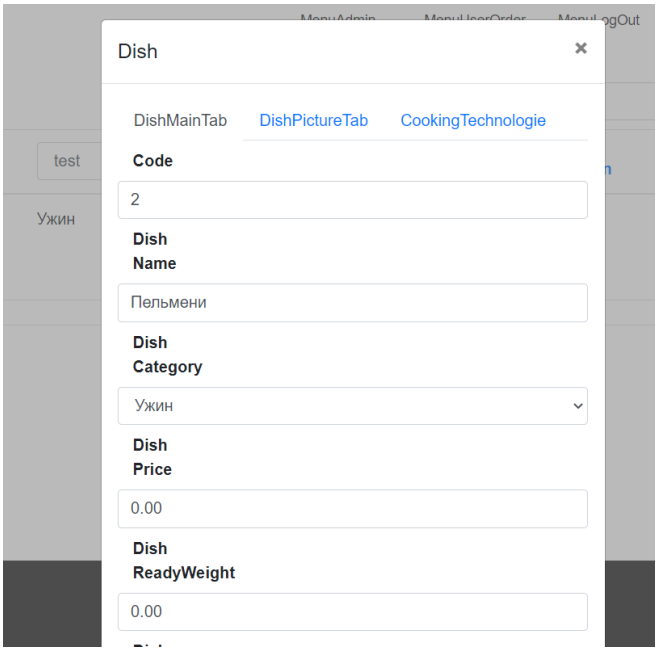
Якщо користувача є адміном, тоді йому стає доступним меню адміну, через як він повністю може керувати роботою Компанії. Також з’являється кнопка для переходу на замовлення та кнопка для виходу з системи. Після входу, знизу пишеться назва компанії в якій працює користувач.



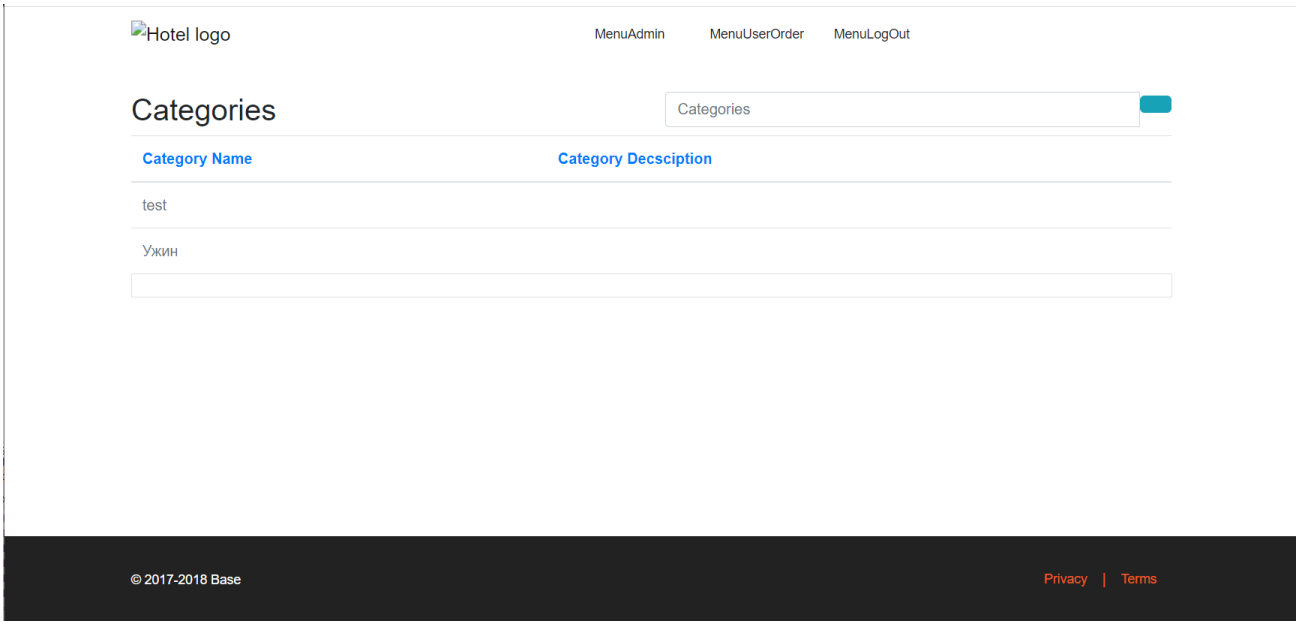
Таким чином виглядає список створених страв, можна додати нову, редагувати або видалити. Також можна шукати страви.



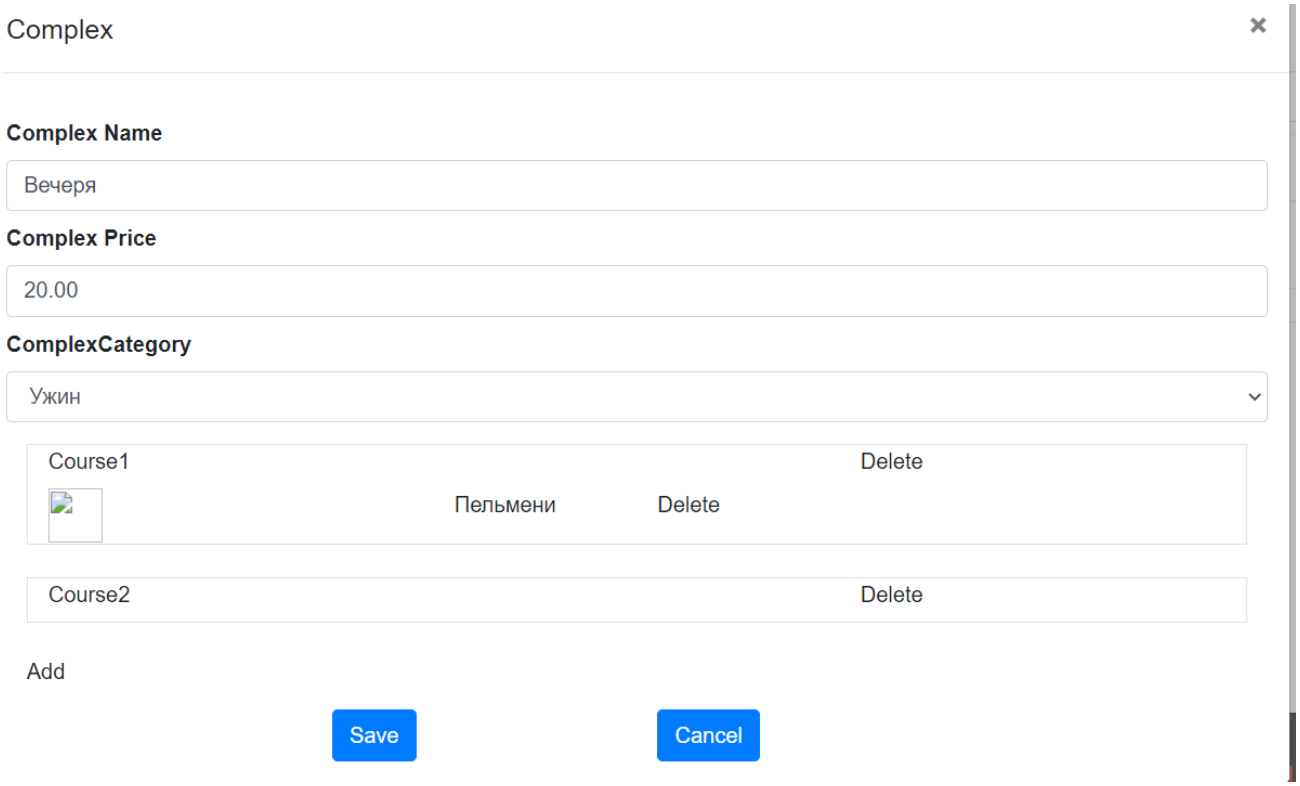
Вікно для редагування або створення страв. Для створення першої страви має бути створена категорія.



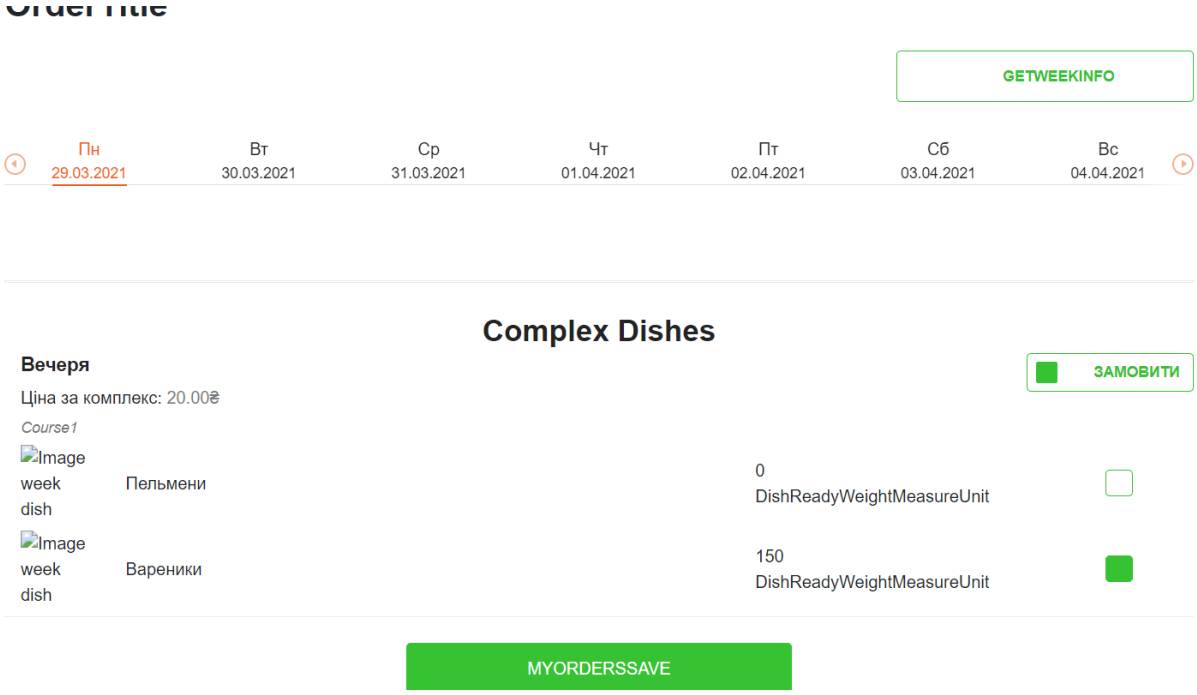
Вікно категорії виглядає аналогічно.



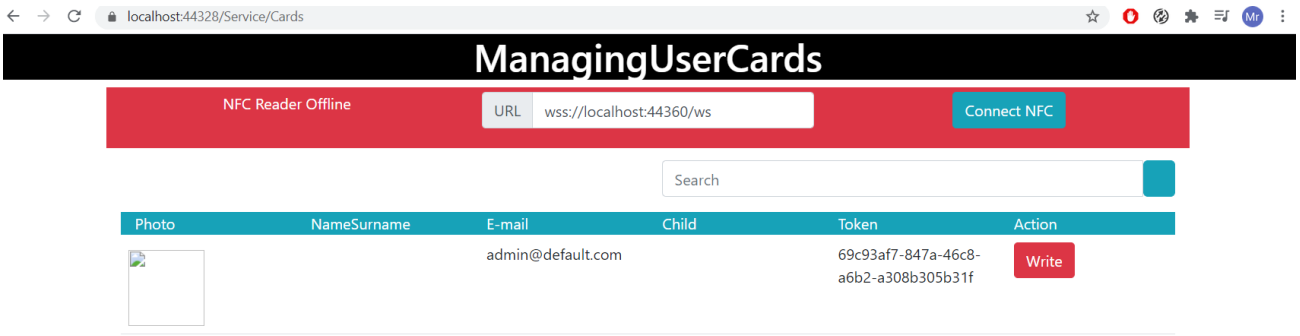
Вікно редагування комплексу. У комплексі може бути декілька страв. Також може бути декілька страв на певні курси, для того щоб клієнт міг обрати з запропонованих.



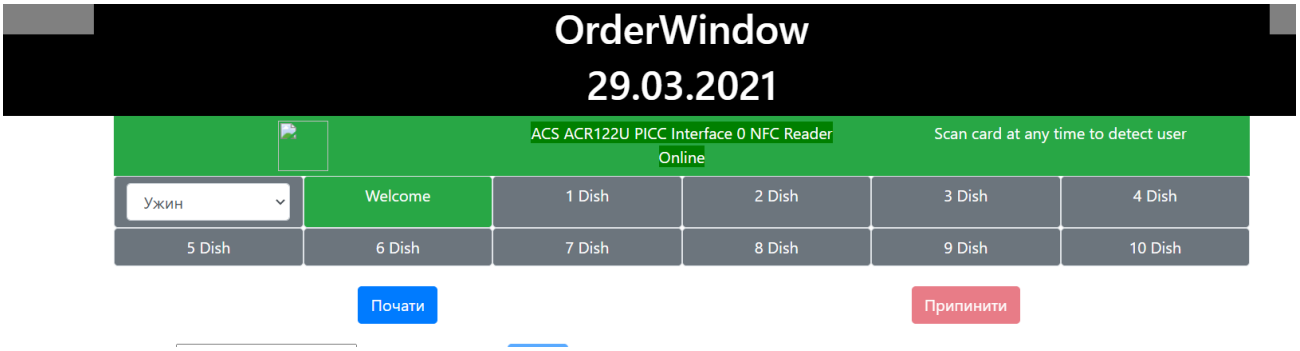
Таким чином виглядає сторінка замовлення страв. Обирається певний день тижня. Базово обрано завтрашній день. І робиться замовлення. Кошик замовлення свій на кожен день.



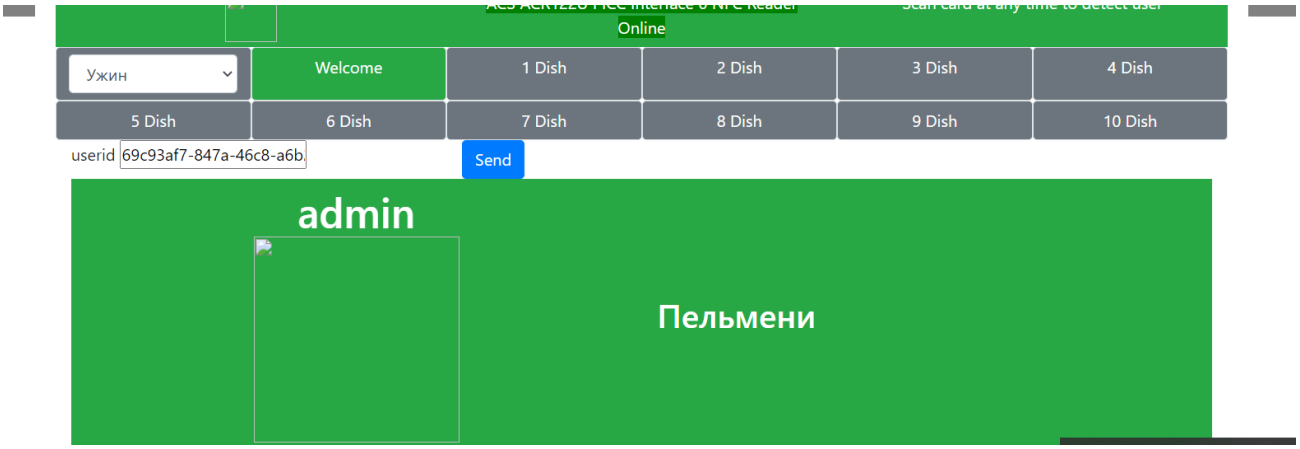
Вікно редагування прив’язаних карток до користувачів. У даному вікні сервісів адмін може прив’язати нову картку користувачеві. Крім того він може піднести будь-яку картку, і якщо на неї записано користувача, він висвітиться.



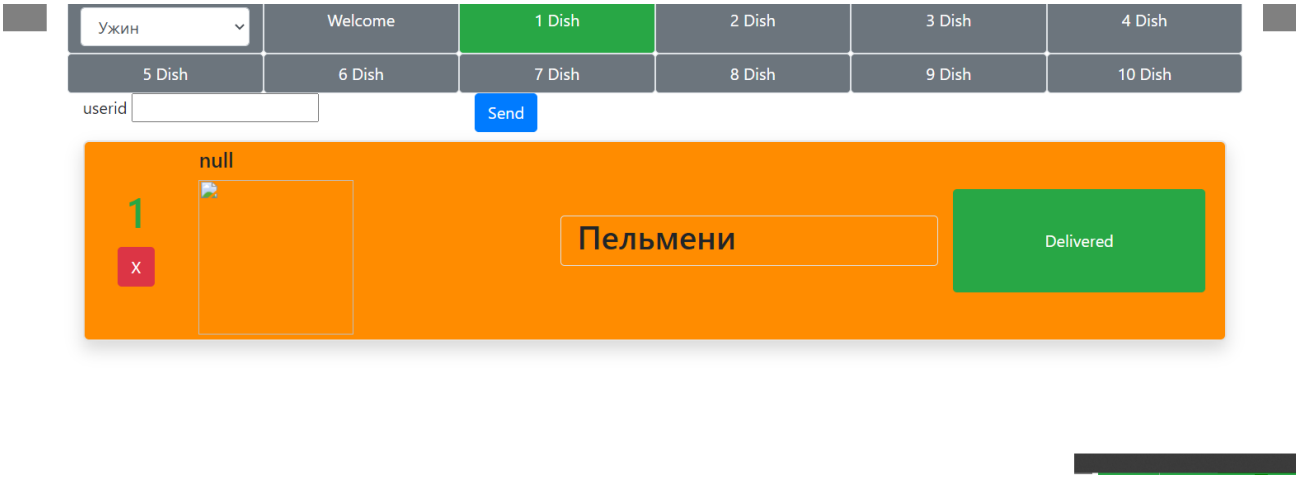
Вікно видачі замовлення. Вважається, що є один комп’ютер до якого підключено зчитувач карток. На ньому обирається що саме зараз видається (вечеря, обід і тд). Для початку видачі треба натиснути Почати.



Клієнт підносить свою картку і на екрані висвітлюється що саме замовив клієнт на видачу, або ж, що немає страв до видачі.



На робочому місці офіціанта може бути планшет, на якому він обирає, який курс видає, Одночасно зі ставня у чергу офіціанту висвітлюється, що саме повинен отримати клієнт. Після видачі, натискається кнопка Delivered.



3.4 Опис використаних технологій

Visual Studio 2019 Community - це повнофункціональне інтегроване середовище розробки (IDE) для додатків Android, iOS, Windows, веб- і хмарних додатків. Visual Studio підтримує 36 різних мов програмування та дозволяє редактору коду та виконувати наладку (різною мірою) майже на будь-якій мові програмування. Вбудовані мови включають:

- C
- C ++
- C ++ / CLI
- Visual Basic
- .NET
- C #
- F #,
- JavaScript
- TypeScript
- XML
- XSLT
- HTML та CSS.

Підтримка інших мов, таких як Python, Ruby, Node.js та M, серед інших, доступна через розширення. Раніше підтримувались Java (і J #).

ACR122U NFC Reader - це безконтактний пристрій для зчитування / запису смарт-карток, розроблений на основі безконтактної (RFID) технології 13,56 МГц. Сумісний зі стандартом ISO / IEC18092 для зв'язку наближеного поля (NFC), він підтримує не тільки карти MIFARE® та ISO 14443 A і B, але також усі чотири типи тегів NFC.
ACR122U сумісний як з CCID, так і з ПК / SC. Таким чином, це USB-пристрій plug-and-play, що забезпечує взаємодію з різними пристроями та додатками. Швидкість доступу до 424 кбіт / с та повна швидкість USB до 12 Мбіт / с. Робоча відстані ACR122U становить до 5 см, залежно від типу безконтактної мітки, що використовується.

SQL Server Management Studio (SSMS) - утиліта з Microsoft SQL Server 2005 і пізніших версій для конфігурації, управління і адміністрування всіх компонентів Microsoft SQL Server. Утиліта включає редактор скриптів і графічну програму, яка працює з об'єктами і налаштуваннями серверу.

ASP.NET - це платформа веб-додатків на стороні сервера з відкритим кодом, призначена для веб-розробки для створення динамічних веб-сторінок. Вона була розроблена Microsoft, щоб дозволити програмістам створювати динамічні веб-сайти, програми та сервіси.

Entity Framework (EF) - це структура об’єктно-реляційного відображення (ORM) із відкритим кодом для ADO.NET. Спочатку він постачався як невід’ємна частина .NET Framework. Починаючи з Entity Framework версії 6, він постачається окремо від .NET Framework.

Висновок

Отже була задача розробка клієнт-серверного застосунку з використанням технології NFC. Для її вирішення прийнято рішення розробити, щось на зразок CRM системи для керування харчування готелів або компанії, що годує своїх співробітників. Таким чином кухня може приготувати точну кількість страв на видачу та зменшити витрати на виробництво, що позитивно впливає на екологічність виробництва їжі, адже менше залишків, що викидаються.

На ринку на разі найближче, що існує це CRM системи для ресторанів, або кафе. Однак вони розраховані на замовлення у реальному часі та моментальну видачу. Клієнт також рідко прив’язуються до конкретних замовлень. Ресторани часто мають свої додатки, для накопичення балів, це максимум, що в них є.

Для вирішення проблеми видачі страв, було розроблено систему, що може приймати замовлення клієнтів та потім видавати ці страви кожному індивідуально, у порядку живої черги. Для цього кожен користувач заноситься в систему, та йому видається прив’язана до нього картка, у випадках готелів, це може біти ключем від номеру.

Для подальшого розвитку системи, вважаю доцільним доробити систему складу інгредієнтів страв, можна додати рух інгредієнтів по складу. Можливо, ще додати замовлення у реальному часі, адже страви по типу Кави навряд чи будуть замовляти заздалегідь.

Список використаної літератури

1. Паттерны для новичков: MVC vs MVP vs MVVM
   https://habr.com/ru/post/215605/
2. Патерн
   https://uk.wikipedia.org/wiki/%D0%9F%D0%B0%D1%82%D0%B5%D1%80%D0%BD
3. Overview of ASP.NET Core MVC
   https://docs.microsoft.com/en-gb/aspnet/core/mvc/overview?view=aspnetcore-5.0
4. What is NFC?
   https://nfc-forum.org/
5. Что такое NFC и как он работает. Освежим основы?
   https://habr.com/ru/post/479904/
6. What is NFC? Everything you need to know
   https://www.techradar.com/news/what-is-nfc
7. CRM
   Bardicchia, Marco (2020). Digital CRM: Strategies and Emerging Trends: Building Customer Relationship in the Digital Era. p. 12.
8. CRM система Тилипад
   https://tillypad.ru/system
9. THE FOOD CRM POS
   http://www.thefoodcrm.com/
10. ОБЗОР CRM-СИСТЕМ ДЛЯ РЕСТОРАННОГО БИЗНЕСА
    https://crm-systems.info/crm-dlya-kafe/
11. Документація Майкрософта
    https://docs.microsoft.com