Звіт до проектної роботи

на тему: "Розробка системи замовлення для бургерної"

# Реферат

Пояснювальна записка до розрахунково-графічної роботи містить: 45 сторінок, 8 рисунків, 3 додатоки.

Метою даної розрахунково-графічної роботи є розробка веб сайту для замовлення бургерів.

Об’єктом дослідження є засоби проектування та розробки веб сайтів, які реалізують функціонал замовлення товару, а предметом дослідження проектування та розробка веб сайту за концепцією MVC за допомогою фреймворку CodeIgniter.

Методи дослідження – теоретичний (ознайомлення з документацією специфікаціями та предметною областю) та практичний (проектування сайту та його розробка).

Результатами роботи є веб сайт замовлення бургерів.

ФРЕЙМВОРКИ, PHP, CODEINGER, MVC, ПАТЕРН, ПРОЕКТУВАННЯ, ВЕБ САЙТ, SEMANTIC UI, РОЗРОБКА, АРХІТЕКТУРА, КОНТРОЛЕР, МОДЕЛЬ, УЯВЛЕННЯ, ШАБЛОН

# 

# Зміст

[Вступ 4](#_Toc309236)

[1 Обзор засобів розробки 5](#_Toc309237)

[1.1 Вибір мови розробки бекенду 5](#_Toc309238)

[1.2 Вибір мов та засобів створення фронтенду 7](#_Toc309239)

[1.3 Фреймворк CodeIgniter 8](#_Toc309240)

[1.4 Інші технології розробки 9](#_Toc309241)

[2 Проектування програмної системи 10](#_Toc309242)

[2.1 Проектування архітектури системи 10](#_Toc309243)

[2.1.1 Проектування архітектури бекенду 10](#_Toc309244)

[2.1.2 Проектування структури фронтенду 11](#_Toc309245)

[2.2 Проектування загальної структури програми 11](#_Toc309246)

[2.3 Проектування бази даних 14](#_Toc309247)

[2.4 Визначення необхідного функціоналу сайту 15](#_Toc309248)

[3 Розробка системи 19](#_Toc309249)

[3.1 Встановлення CodeIgniter 19](#_Toc309250)

[3.2 Реалізація моделей 19](#_Toc309251)

[3.3 Реалізація контролерів 20](#_Toc309252)

[3.4 Реалізація фронтенду адміністраторської панелі 20](#_Toc309253)

[Висновки 21](#_Toc309254)

[Список використаної літератури 22](#_Toc309255)

[Додаток А Код розгортання бази даних 23](#_Toc309256)

[Додаток Б Вихідний код 25](#_Toc309257)

[Б.1 Вихідний код контролеру Admins 25](#_Toc309258)

[Б.2 Вихідний код контролеру Welcome 29](#_Toc309259)

[Додаток В Код шаблонів сторінок 32](#_Toc309260)

[В.1 Вихідний код шаблону activeorders 32](#_Toc309261)

[В.2 Вихідний код шаблону closeorders 33](#_Toc309262)

[В.3 Вихідний код шаблону createproduct 33](#_Toc309263)

[В.4 Вихідний код шаблону createslider 34](#_Toc309264)

[В.5 Вихідний код шаблону editproduct 34](#_Toc309265)

[В.6 Вихідний код шаблону editslider 35](#_Toc309266)

[В.7 Вихідний код шаблону footer 35](#_Toc309267)

[В.8 Вихідний код шаблону header 36](#_Toc309268)

[В.9 Вихідний код шаблону index 36](#_Toc309269)

[В.10 Вихідний код шаблону login 36](#_Toc309270)

[В.11 Вихідний код шаблону loginedit 37](#_Toc309271)

[В.12 Вихідний код шаблону orderinfo 37](#_Toc309272)

[В.13 Вихідний код шаблону products 38](#_Toc309273)

[Г.14 Вихідний код шаблону sliders 39](#_Toc309274)

[Додаток Г Екрані форми розробленої адміністраторської панелі 40](#_Toc309275)

# Вступ

З моменту появи першої сторінки в інтернеті веб сайти еволюціонували з статичних сторінок до багатофункціональних навантажених порталів. Сьогодні в інтернеті можливо знайти тисячі веб сайтів, які пропонують користувачу динамічний функціонал.

Нещодавно веб сайт був невід’ємним атрибутом будь−якого крупного відомого бренду та магазину. Але часи змінились. Тепер і маленькі фірми та заклади мають повноцінні представництва в мережі інтернет. Так представництва торгових точок дозволяють тим чи іншим способом зробити придбання товару. Щоб забрати його потім або отримати його з доставкою. Такі системи називають електроні магазини чи системи замовлень.

Хто б міг подумати десять років тому назад, що для того, щоб замовити чи придбати якийсь товар не буде необхідності йти до магазину або шукати телефон магазина у довідниках, а необхідно буде вбити запит у пошукову мережу, знайти сайт, обрати товар та придбати його.

Між системами замовлення і інтернет магазинами є суттєва різниця. Цю різницю можна побачити на прикладі з реальним світом, так відповідником інтернет магазинам слугують реальні магазини, а системам замовлення − гарячі лінії магазинів, де можливо замовити той чи інший товар з доставкою.

Яскраві приклади систем замовлення можна побачити на сайтах ресторанів, піцерій, кафе, бургерних та інших закладів харчування. Це зв’язано з тим, що специфіка товару вимагає швидкої оперативності системи замовлення та доставки. Також із цього можна підсумувати, що системи замовлення на відмінно від інтернет магазинів продають якусь окрему групу товарів, а інтернет магазини можуть торгувати великим різноманіттям товарів.

Із вище наведеного систему замовлення в деякому роді можна назвати сучасним MacDrive, на який не має необхідності їхати, щоб оформити та забрати замовлення.

# Обзор засобів розробки

## Вибір мови розробки бекенду

Мабуть самий складний вибір мови програмування зараз серед мов програмування бекенду. Оскільки немає такої сталості, як серед мов програмування звичайних настільних програм. Так в топ 10 мов програмування (рис. Рисунок 1.1) перебуває 8 мов за допомогою, яких можлива розробка бекенду, а саме Java, PHP, Python, C#(ASP.NET), JavaScript(Node.JS), Perl, Ruby.

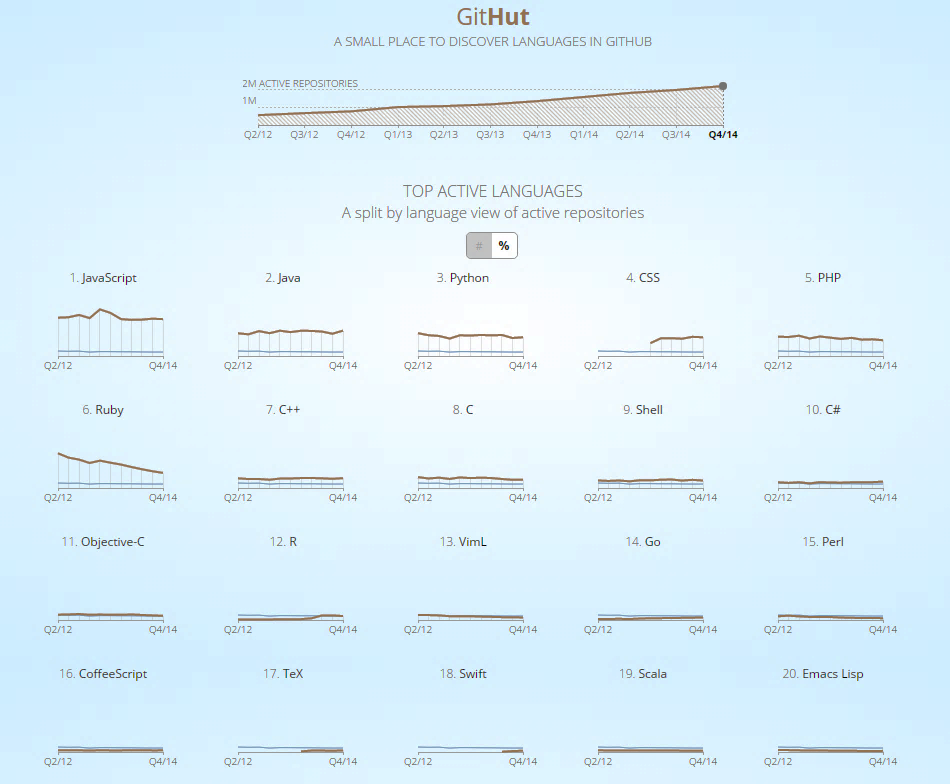


Рисунок 1.1 – Статистика використання мов програмування на GitHub

Серед цих мов програмування було обрано саме мову PHP, оскільки вона є гарно документованою, під неї існує велика кількість фреймворків. А також досить просто знайти хостинг та вони стоять достатньо дешево.

Історія PHP починається восени 1994 року. Коли Расмус Лердорф (Rasmus Lerdorf) почав працювати над тим, що згодом стало PHP, єдиною метою, яка була у нього в думках, з'ясувати, хто читає його резюме. У той час, будучи незалежним підрядником, Лердорф розсилав по потенційним роботодавцям своє міні-резюме з URL посиланням на його повну версію. Щоб стежити за відвідувачами, він створив CGI скрипт на Perl, який вставлявся як спеціальний тег в HTML код його сторінки та збирав інформацію про відвідувачів. Щоб справити враження на потенційних роботодавців, він дозволив будь кому переглядати статистку відвідувань.

Він назвав цей код для збору статистики «PHP-Tools for Personal Home Page», оскільки сам використовував його на своїй персональній домашній сторінці (personal home page). Кілька людей поцікавилися тим, як вони могли б отримати цей інструмент, і Лердорф прийняв рішення надати можливість ставити його іншим особам. «Це диво програмного забезпечення. Ви можете дати це і тим не менш залишити це собі», - дотепно зауважив Лердорф. У той час руху Open Source не існувало. Тоді воно назвалося free-ware. Ближче до кінця 1995 р. Лердорф відкрив для людей перший список розсилки по PHP, щоб можна було обмінюватися ідеями, виправленнями помилок і кодом.

Так зародилася мова PHP. Зараз це прогресивна мова програмування, яка еволюціонувала від функціональної до об’єктно−орієнтованої. В останній версії PHP, а саме PHP 7. Було введено багато нового: серед цього додаткові оператори, контроль типу аргументу функції та було оптимізовано швидкість виконання скриптів. Тому насправді можна сказати, що мова розвивається. Також мова є 5 за популярністю за версією GitHub.

## Вибір мов та засобів створення фронтенду

Серед мов створення фронтенду немає такої великої конкуренції, як серед мов бекенду. Хоча є декілька мов розмітки альтернативної до HTML, але сучасні масові браузери її не підтримують, або ця розмітка повністю формується за допомогою JavaScript, чи іншого допоміжного скрипта, який має досить великий розмір для завантаження користувачем. Тому золотою серединою є вибір мови програмування HTML, яка на гарному рівні підтримується усіма сучасними браузерами, а також мови CSS, яка призначена для описання HTML розмітки документа та стилів його об’єктів. На даний час останніми версіями цих мов програмування HTML5 та CSS3.

Також для написання скриптів для обробки дії користувача на сторінці веб сайту було обрано мову JavaScript. З тих самих міркувань про сумісність з максимальною кількістю веб сайтів.

Верстка веб сторінок дуже тривалий і важкий процес. Оскільки потрібно реалізувати дизайн, що вписується в заздалегідь відому концепцію дизайну. Та такий набір стилів, щоб він не розпадався у сучасному браузері. Є багато концепцій підходу до процесу розробки дизайну сайту. Але провідними командами розробників зазвичай використовується наростаюча схема у якій з початку строюється первинний набір UI Kit головних елементів веб сторінок. UI Kit в загальному розумінні набір елементів, котрі потім поєднується в веб сторінку. Розвитком UI Kit є фронтенд фреймворки Такі фреймворки призвані скоротити та полегшити один з ключових етапів розробки сайту.

Для створення сторінок сайту було прийнято рішення використовувати фронтенд фреймворк Semantic UI. З коробки фреймворк надає колекцію елементів оформлену в стилістиці flat дизайну.

Flat дизайн це підхід до побудови інтерфейсів програм і операційних систем, представлений, як протилежність скевоморфізму. За задумом «плоский дизайн» повинен підкреслювати ефект «чарівною простоти» і витонченості. Став набирати популярність з 2010 року, а з 2014 - стає новим стандартом в дизайнерському комп'ютерному напрямку.

Плаский дизайн бере своє натхнення в основному в трьох стилях мистецтва: Міжнародними типографічним стилем (також відомий як Swiss Style), мінімалізмі та стилях, що походять із Bauhaus. Міжнародний типографічні стиль має найбільш істотний вплив на плаский дизайн, його виникнення та популяризацію в 1950-х і 1960-х роках і розглядається як відправна точка плаского дизайну, хоча до поширення стилю в цифровому світі залишались десятки років.

Спочатку концепція плоского дизайну полягає у відмові від скевоморфізму – ілюзії тривимірного зображення або об'єкта, яка досягається шляхом створення відображення, текстур, тіней, і створенні так званого плоского дизайну з використанням найпростіших одноколірних елементів. Згідно із задумом, візуально плоский дизайн виглядає більш привабливо і витончено. Також простий дизайн знімає додаткове навантаження на сайти або програми, завдяки якому вони працюють і завантажуються швидше. Плоский дизайн протиставляється так званому «багатому дизайну» (англ. Rich design), в якому значний акцент робиться на різноманітні візуальні і звукові ефекти.

Плоский дизайн був широко поширений в 80-і роки, у зв'язку з тим, що рівень технологій не дозволяв створювати складні текстури, однак і тоді вперше стали з'являтися прості елементи скевоморфізм щоб створити візуальну зв'язок з цифровим об'єктом.

## Фреймворк CodeIgniter

Це не той випадок, коли мови програмування обмежені технологіями, та різноманітними засобами. Вибраний перелік мов програмування можна використовувати в різних версіях побудови сайтів. Але існують загальні прийняті концепції проектування сайтів, яких слід притримуватися, щоб на виході отримувати гарний зрозумілий не тільки розробнику програмний код, який буде легким у підтримці, а також його можна буде легко розширити для отримання нових функціональних можливостей.

Тому для бекенду було вирішено обрати MVC (Model-View-Controller) фреймворк, оскільки він має базовий необхідний набір функції та дозволяє абстрагуватися від прямої роботи з базою даних.

Таким вибором став фреймворк CodeIgniter, який має масу відео уроків, які можна знайти як на офіційному сайті, так і на інших ресурсах. Оскільки CodeIgniter має гарну документацію, продуману архітектуру та легкий для вивчення починаючими програмістами PHP.

## Інші технології розробки

Фреймворк CodeIgniter прямо з коробки реалізує велику кількість патернів програмування. Першим з них без якого не було б можливим реалізувати більшість інших − це Single Entry Point або Єдина точка входу до скрипту. За замовчанням PHP веб сервери, які обслуговують запити користувачів перенаправляють запит до файлу, адреса якого передається за протоколом HTTP. Патерн єдиної точки входу полягає в переадресації усіх запитів до певного скрипта і побудові власної системи адресації.

Другим ключовим патером є MVC, який призваний розділити роботу з базою даних, користувацьке уявлення та бізнес логіку, а саме моделі працюють з базами даних, контролери відповідають за бізнес логіку, а уявлення за відображення отриманих результатів. CodeIgniter реалізує класичну версію даного патерну.

Також фреймворк реалізує зручну ORM (Object Relative Model) для роботи з об’єктами баз даних та дозволяє отримувати данні з бази даних без довжелезних SQL запитів у коді.

Окрім цього для розширення можливостей JavaScript було обрано бібліотеку jQuery, яка у дозволяє посилати до серверу асинхронні AJAX запити до серверу.

# Проектування програмної системи

## Проектування архітектури системи

### Проектування архітектури бекенду

Архітектура бекенду зумовлена обраними засобами розробки, а саме фреймворком CodeIgniter, що реалізує патерн MVC (рис. Рисунок 2.1). Патерн заснований на трьох базових елементах: модель, уявлення, контролер.

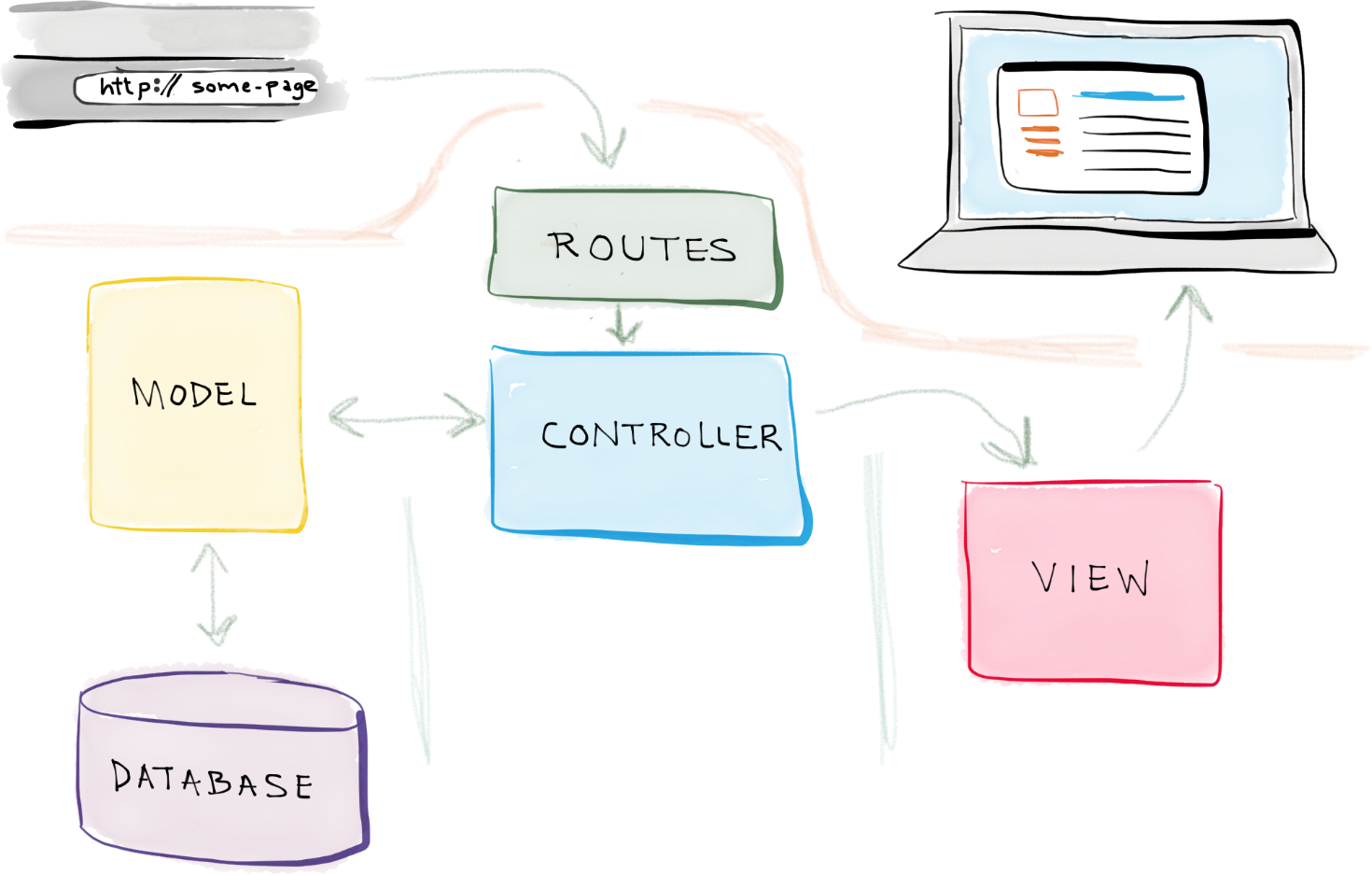


Рисунок 2.1 – Загальна концепція MVC

Модель являє структури даних. Зазвичай модельні класи містять набор функцій, які допоможуть вам витягувати, вставляти і оновлювати інформацію в базі даних. Уявлення є інформація, яка видається користувачеві. Уявлення, як правило, може бути веб сторінкою, але в CodeIgniter, також може бути фрагментом сторінки, як верхній або нижній колонтитул. Він також може бути RSS-сторінкою, або будь-яким іншим типом "сторінок". Контролер виступає в якості посередника між моделлю, уявленням, а також будь-якими іншими ресурсами, необхідними для обробки запиту HTTP і створення веб сторінки.

CodeIgniter має досить вільний підхід до MVC, так як моделі не вимагається. Якщо вам не потрібно додаткове відділення, або вважають, що підтримка моделей вимагає більше складності, ніж потребується, ви можете ігнорувати їх і побудувати мінімальний додаток з контролерів і уявлень. CodeIgniter дозволяє також включати свої власні існуючі сценарії, або навіть розробити основні бібліотеки для системи, що дозволяє працювати таким чином, що робить самий сенс для вас.

### Проектування структури фронтенду

Структура фронтенду також має свої особливості, внесені структурою бекенду. Фронтенд поділений на шаблони та збирається у "сторінку" відповідно до необхідної структури, а також деяка логіка на фронтенді організована за допомогою AJAX запитів.

## Проектування загальної структури програми

Файлова структура програми повністю зумовлена обраним фреймворком та наведена на рисунку Рисунок 2.2.

Файлова структура фреймворку CodeIgniter не є чимось не звичайним. Така файлова структура характерна для багатьох сучасних MVC фреймворків.

За замовчанням в кожній під директорії додатку знаходиться файл "index.html", який призваний захистити файлову структуру сайту у випадку проблем з сервером, коли сервер починає віддавати файлову структуру замість потрібного результату запиту. Це може виникнути в випадку не вірних налаштувань, які знаходяться у файлі конфігурації ".htaccess" або при недокументованій роботі веб серверу.

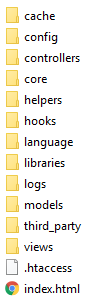


Рисунок 2.2 – Файлова структура програми

Як правило, у папці "cache" зберігаються дані, які повинні бути використані в майбутньому. Таким чином, можливо сказати, що якщо ваш додаток використовує дані, які використовуються в 100 разів на хвилину, було б краще, щоб зберігалися обчислювальні дані. В такому випадку час і обчислювальна потужність зберігається за рахунок скорочення процесу "завантаження".

У CodeIgniter, ви можете виконати кешування також. Після того, як ваша сторінка завантажується, наступні навантаження будуть викликати додаток для завантаження з кеша. Це дозволить підвищити продуктивність вашого веб додатка.

Папка "config" призначена для збереження конфігураційних файлів, які відповідають за налаштування CodeIgniter. Наприклад, файл конфігурації "database.php" дозволяє встановити одну чи декілька баз даних, які будуть використовуватися у вашому проекті. Конфігураційний файл "Autoload.php" відповідає за завантаження бібліотек, хелперів чи інших конфігураційних файлів.

Папка "controllers" містить у собі контролери на дії користувача. Наприклад, якщо користувач, ініціює якусь дію контролера, то контролер виконує свої функціональні можливості на стороні сервера. Грубо кажучи, ми можемо сказати, що контролер є мозком веб додатку CodeIgniter. Якщо контролер не працює, нічого не буде працювати.

Контролер може передавати параметри, такі як ім'я користувача для зареєстрованого користувача. Перший контролер буде використовувати модель, щоб отримати якісь дані з бази даних. Після цього контролер буде передавати цей параметр до уявлення і воно матиме можливість використовувати його, щоб показати його в браузері.

Папка "core" містить у собі класи, які є заміною системних класів. Тобто, якщо у папці є певний клас CodeIgniter, буде використовувати його замість рідного класу системи.

Папка "Helpers" містить файли допоміжних файлів, які містять функції з тої чи іншої категорії, які допомагають створювати URL, форматувати текст чи створювати форми.

Папка "Languages" дозволяє зберігати різноманітні локалізації сайту. Є можливість створювати файли і папки для потрібної мови і використовувати їх у заданому проекті.

Папка "Libraries" зберігає бібліотеки, реалізуючи деякий повторюваний з проекту в проект функціонал. У випадку наявності такого функціоналу потрібно просто створити бібліотеку і носити її з проекту в проект, де використовується заданий функціонал. Також не слід забувати, що бібліотеки можуть визивати моделі. Грубо кажучи, бібліотеки в деякому роді схожі на контролери.

Папка "Logs" призначена для зберігання повідомлень про зміну стану, таких як помилки, обробка виключень повідомлень або повідомлень журналу призначених для користувача. Вони відкладаються в цій папці. Це дуже корисна функція.

Папка "Models" містить у собі моделі для роботи з базою даних. Контролер запитує модель для виконання запитів до бази даних. Після завершення модель (якщо вона запрограмована) буде посилати відповідні дані до контролера і контролер може використовувати їх.

Папка "third\_party" призначена для встановлення розширень сторонніх розробників.

Папка "views" зберігає у собі всі ваші елементи веб дизайну. Ви можете тримати як повністю готові шаблони сторінок веб сайту, так і часткові шаблони, з яких ви можете формувати сторінку веб сайту. Сторінки шаблонів зберігаються з розширенням ".php".

## Проектування бази даних

База даних (рис. Рисунок 2.3) включає до себе 7 таблиць в яких зберігаються дані за предметною областю. Першою з таких таблиць є таблиця "admins", яка зберігає у собі дані про персонал. Таблиця "products" призначена для збереження інформації стосовно пропонованих продуктів, а у таблиці "slider" зберігається інформація про товари, які виводять у галереї на сторінці сайту. Таблиця "users" зберігає у собі інформацію про зареєстрованих користувачів сайту, які мають змогу замовляти товари. Таблиці "orders", "cooks" та "delivers" призначені для збереження інформації про замовлення та їх виконання. Перша таблиця зберігає у собі інформацію про замовлення, друга таблиця зберігає інформацію про товари, які наявні у певному замовленні. Таблиця "delivers" зберігає інформацію про замовлення товару.

Також необхідно зауважити, що для розробки були використані таблиці типу MyIsam, які не підтримують зовнішніх зав’язків. Використання таблиць типу MyIsam у даному конкретному випадку обґрунтовано тим, що швидкість їх роботи на маленьких об’ємах даних є більшою. Крім цього були виставлені необхідними індексні поля для пришвидшення роботи бази даних.

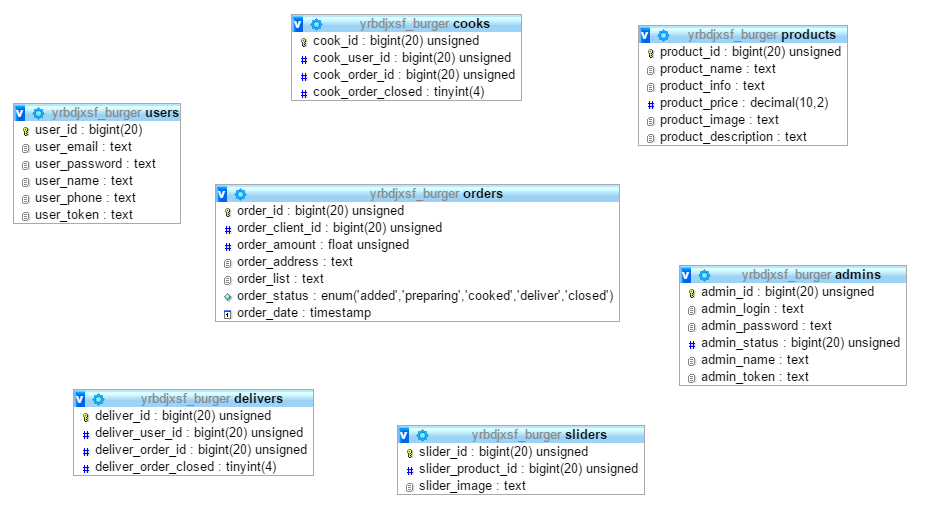


Рисунок 2.3 – Структура бази даних

## Визначення необхідного функціоналу сайту

Оскільки даний сайт призначений для замовлення бургерів, то він повинен організовувати функції вибору та придбання бургерів.

Із за специфіки сайту він поділений на дві частини – основана та адміністраторська. Основна частина сайту призначена для користувачів, а адміністраторська для входу персоналу.

На основній частині сайту користувач може ознайомитися з меню бургерної, обрати бургери та зробити замовлення. Діаграма варіантів використання зображена на рис. Рисунок 2.4.

Адміністраторська частина дозволяє обробляти замовлення, які були зроблені користувачами. Є три загальні ролі персоналу: адміністратор, повар, доставщик. В залежності від ролі, набір доступних функцій буде відрізнятися.

Адміністратор може переглядати усі активні замовлення, а також історію завершених замовлень. Також до функції адміністратора входить призначення замовлення окремому повару, а коли замовлення буде приготоване адміністратор може назначити доставщика, який забере замовлення та доставить його до вказаної адреси. Є можливість змінення даних для авторизації користувача.

Повари та доставщики можуть переглянути список активних заказів, що були відправлені до них, а також позначити заказ, як приготований або доставлений відповідно. Якщо заказ був позначений, як приготований то у адміністратора він позначиться, як приготований та він зможе обрати доставщика, який доставить заказ замовнику.

Для користувача, який має роль адміністратора, діаграма використання наведена на рисунку Рисунок 2.5, а для користувачів з ролями повара та доставщика діаграма використання наведена на рисунку Рисунок 2.6 та на рисунку Рисунок 2.7 відповідно.



Рисунок 2.4 – Діаграма використання основної частини сайту



Рисунок 2.5 – Діаграма використання адміністраторської частини сайту



Рисунок 2.6 – Діаграма використання сайту поваром



Рисунок 2.7 – Діаграма використання сайту доставщиком

# Розробка системи

## Встановлення CodeIgniter

Встановлення фреймворку здійснювалось за допомого системи Composer. Ця система є інструментом для управління залежностями в PHP. Вона дозволяє оголошувати бібліотеки вашого проекту що залежать одна від одної, і він буде керувати встановленням оновлень.

Композитор не менеджер пакетів, в тому ж сенсі, як Yum або APT. Так, мова йде про "пакети" або бібліотеки, але він керує ними на основі кожного проекту, встановлюючи їх в директорії всередині проекту. За замовчуванням він нічого не встановлює на глобальному рівні. Таким чином це менеджер залежностей.

Для встановлення CodeIgniter потрібно виконати наступну команду в терміналі Composer:

$ composer create-project kenjis/codeigniter-composer-installer codeigniter

Під час встановлення фреймворку Composer створить директорії структури проекту, частина яких була описана раніше.

## Реалізація моделей

Із-за невеликої складності та маленького розміру проекту було прийнято рішення не виносити логіку роботи з базою даних до модельного уявлення. Тому що створення уявлень тільки ускладнить структуру проекту та вимагатиме додаткового часу. З цієї причини така логіка була розміщена прямо у контролерах сторінок сайту.

## Реалізація контролерів

Для реалізації функціоналу сайту було створено два контролери. Це обумовлено тим, що на сайті є два глобальні розділи. Перший − користувацька зона, що реалізується за допомогою контролера Welcome, а друга − це адміністраторська панель сайту, за яку відповідає контролер Admins.

Контролери повинні наслідуватися від базового класу CI\_Controller, що відповідає за контролери.

Кожен контролер має методи, які відповідають певним діям користувача та оброблюють їх, передавши оброблений результат користувачу.

## Реалізація фронтенду адміністраторської панелі

Сторінки адміністраторської панелі сайту створювались, як звичайні шаблони сторінок CodeIgniter, та знаходяться у папці "admins", котра знаходиться за стандартною адрескою шаблонів. Було розроблено 14 сторінок шаблону, що представляють собою 6 окремих сторінок адміністраторської панелі. Також, щоб заощадити час на написанні візуальних стилів, використовувався візуальний фреймворк Semantic UI.

# Висновки

В даній роботі був розроблений веб сайт для замовлення бургерів. Розроблена система замовлення являється актуальною, тому що надає можливість оформляти та забирати замовлення через Інтернет.

В роботі досліджені засоби проектування та розробки веб сайтів, які реалізують функціонал замовлення товару.

Під час написання даної роботи було ознайомлено з основними принципами побудови веб сайтів, також було детально досліджено питання побудови бекенду сайтів, а також концепцію MVC у побудові сучасних веб сайтів. Знання, отримані в ході даної роботи, можуть бути використані при проектуванні та створенні різноманітних інтернет порталів.

Програму було реалізовано в повному планованому обсязі.

# Список використаної літератури

1. CodeIgniter по-русски [Електроний ресурс] – <http://code-igniter.ru/> (Дата звернення: 20/12/2016)
2. jQuery API Documentation [Електроний ресурс] – <http://api.jquery.com/> (Дата звернення: 20/12/2016)
3. Open Server руководство пользователя по эксплуатации [Електроний ресурс] – <https://www.ospanel.io/docs/> (Дата звернення: 20/12/2016)
4. Офіційний сайт HTML UI фреймворку Semantic [Електроний ресурс] – <http://semantic-ui.com/> (Дата зверененн: 20/12/2016)
5. CodeIgniter Documentation [Електроний ресурс] – [https://codeigniter.com/  
   docs](https://codeigniter.com/docs) (Дата звернення: 20/12/2016)

# Додаток А Код розгортання бази даних

--

-- База данных: `burgers`

--

-- --------------------------------------------------------

--

-- Структура таблицы `admins`

--

CREATE TABLE IF NOT EXISTS `admins` (

`admin\_id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT,

`admin\_login` text NOT NULL,

`admin\_password` text NOT NULL,

`admin\_status` bigint(20) UNSIGNED NOT NULL,

`admin\_name` text NOT NULL,

`admin\_token` text NOT NULL,

PRIMARY KEY (`admin\_id`),

KEY `admin\_status` (`admin\_status`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--

-- Структура таблицы `cooks`

--

CREATE TABLE IF NOT EXISTS `cooks` (

`cook\_id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT,

`cook\_user\_id` bigint(20) UNSIGNED NOT NULL,

`cook\_order\_id` bigint(20) UNSIGNED NOT NULL,

`cook\_order\_closed` tinyint(4) NOT NULL DEFAULT '0',

PRIMARY KEY (`cook\_id`),

KEY `cook\_user\_id` (`cook\_user\_id`),

KEY `cook\_order\_id` (`cook\_order\_id`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--

-- Структура таблицы `delivers`

--

CREATE TABLE IF NOT EXISTS `delivers` (

`deliver\_id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT,

`deliver\_user\_id` bigint(20) UNSIGNED NOT NULL,

`deliver\_order\_id` bigint(20) UNSIGNED NOT NULL,

`deliver\_order\_closed` tinyint(4) NOT NULL DEFAULT '0',

PRIMARY KEY (`deliver\_id`),

KEY `deliver\_user\_id` (`deliver\_user\_id`),

KEY `deliver\_order\_id` (`deliver\_order\_id`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--

-- Структура таблицы `orders`

--

CREATE TABLE IF NOT EXISTS `orders` (

`order\_id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT,

`order\_client\_id` bigint(20) UNSIGNED NOT NULL,

`order\_amount` float UNSIGNED NOT NULL,

`order\_address` text,

`order\_list` text NOT NULL,

`order\_status` enum('added','preparing','cooked','deliver','closed') NOT NULL,

`order\_date` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP,

PRIMARY KEY (`order\_id`),

KEY `order\_client\_id` (`order\_client\_id`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--

-- Структура таблицы `products`

--

CREATE TABLE IF NOT EXISTS `products` (

`product\_id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT,

`product\_name` text NOT NULL,

`product\_info` text NOT NULL,

`product\_price` decimal(10,2) NOT NULL,

`product\_image` text NOT NULL,

`product\_description` text NOT NULL,

`product\_created` datetime NOT NULL,

`product\_available` tinyint(4) NOT NULL,

PRIMARY KEY (`product\_id`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--

-- Структура таблицы `sliders`

--

CREATE TABLE IF NOT EXISTS `sliders` (

`slider\_id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT,

`slider\_product\_id` bigint(20) UNSIGNED NOT NULL,

`slider\_image` text NOT NULL,

PRIMARY KEY (`slider\_id`),

KEY `slider\_product\_id` (`slider\_product\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--

-- Структура таблицы `users`

--

CREATE TABLE IF NOT EXISTS `users` (

`user\_id` bigint(20) NOT NULL AUTO\_INCREMENT,

`user\_email` text NOT NULL,

`user\_password` text NOT NULL,

`user\_name` text NOT NULL,

`user\_phone` text NOT NULL,

`user\_token` text NOT NULL,

`user\_address` text NOT NULL,

PRIMARY KEY (`user\_id`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

Додаток Б  
Вихідний код

Б.1 Вихідний код контролеру Admins

<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Admins extends CI\_Controller {

/\*base\*/

function \_show($page, $data = null){

$this->load->view('admin/header.php');

if($data == null) {

$this->load->view('admin/' . $page . '.php');

} else {

$this->load->view('admin/' . $page . '.php', $data);

}

$this->load->view('admin/footer.php');

}

function \_check(){

if (!empty($\_COOKIE['admintoken'])) {

$this->db->where('admin\_token', $\_COOKIE['admintoken']);

$query = $this->db->get('admins');

$cnt = $query->num\_rows();

if($cnt != 1){

redirect("admins/login");

} else{

$res = $query->row\_array();

return $res['admin\_id'];

}

} else {

redirect("admins/login");

}

}

function checkdir(){

echo 'Document root: '.$\_SERVER['DOCUMENT\_ROOT'].'<br>';

echo 'Полный путь к скрипту и его имя: '.$\_SERVER['SCRIPT\_FILENAME'].'<br>';

echo 'Имя скрипта: '.$\_SERVER['SCRIPT\_NAME'];

}

function \_getstatus(){

$this->db->where('admin\_token', $\_COOKIE['admintoken']);

$query = $this->db->get('admins');

$row = $query->row\_array();

return $row['admin\_status'];

}

function login(){

$showData = array('loginError' => false);

if (!empty($\_POST)) {

if (!empty($\_POST['login']) && !empty($\_POST['password'])) {

$pass = md5($\_POST['password']);

$this->db->where('admin\_login', $\_POST['login']);

$this->db->where('admin\_password', $pass);

$query = $this->db->get('admins');

$cnt = $query->num\_rows();

if ($cnt == 1) {

$token = md5(rand(0, 1000) \* time());

$d = array('admin\_token' => $token);

$this->db->where('admin\_login', $\_POST['login']);

$this->db->update('admins', $d);

setcookie("admintoken", $token);

redirect("admins/");

} else {

$showData['loginError'] = true;

}

}

}

$this->\_show('login', $showData);

}

function loginedit(){

$this->\_check();

$showData = array('message' => false);

if(isset($\_POST['submit'])){

$ud = array(

'admin\_login' => $\_POST['login'],

'admin\_password' => md5($\_POST['password'])

);

$this->db->where('admin\_token',$\_COOKIE['admintoken']);

$this->db->update('admins', $ud);

$showData['message'] = true;

}

$this->db->where('admin\_token', $\_COOKIE['admintoken']);

$query = $this->db->get('admins');

$showData['info'] = $query->row\_array();

$this->\_show('loginedit', $showData);

}

function logout(){

setcookie("admintoken", "");

redirect("admins/");

}

function index(){

$this->\_check();

$status = $this->\_getstatus();

$data = array('status' => $status);

$this->\_show('index', $data);

}

/\*main\*/

function closedorders($page = 0){

$this->\_check();

$showData = array();

$offset = $page \* 20;

$this->db->where('order\_status', 'closed');

$this->db->order\_by('order\_date', 'desc');

$this->db->limit(20, $offset);

$query = $this->db->get('orders');

$cnt = $this->db->count\_all\_results('orders');

$showData['list'] = $query->result\_array();

$showData['maxPages'] = ceil($cnt / 20.0);

$showData['curPage'] = $page;

$this->\_show('closedorders', $showData);

}

function activeorders(){

$id = $this->\_check();

$showData = array();

$showData['status'] = $this->\_getstatus();

switch ($showData['status']){

case '1':

$this->db->select('deliver\_order\_id');

$this->db->where('deliver\_user\_id', $id);

$q = $this->db->get('delivers');

$r = $q->result\_array();

$ids = array();

foreach ($r as $item){

$ids[] = $item['deliver\_order\_id'];

}

if(count($ids) == 0)

$ids[] = 0;

$this->db->where('order\_status', 'deliver');

$this->db->where\_in('order\_id', $ids);

$q = $this->db->get('orders');

$r = $q->result\_array();

$showData['list'] = $r;

break;

case '2':

$this->db->select('cook\_order\_id');

$this->db->where('cook\_user\_id', $id);

$q = $this->db->get('cooks');

$r = $q->result\_array();

$ids = array();

foreach ($r as $item){

$ids[] = $item['cook\_order\_id'];

}

if(count($ids) == 0)

$ids[] = 0;

$this->db->where('order\_status', 'preparing');

$this->db->where\_in('order\_id', $ids);

$q = $this->db->get('orders');

$r = $q->result\_array();

$showData['list'] = $r;

break;

case '4':

$this->db->where('order\_status !=', 'closed');

$q = $this->db->get('orders');

$r = $q->result\_array();

$showData['list'] = $r;

$this->db->where('admin\_status', 2);

$q = $this->db->get('admins');

$r = $q->result\_array();

$showData['cooks'] = $r;

$this->db->where('admin\_status', 1);

$q = $this->db->get('admins');

$r = $q->result\_array();

$showData['delivers'] = $r;

break;

}

$clients = array();

foreach ($showData['list'] as $item){

$clients[] = $item['order\_client\_id'];

}

if(empty($clients)) {

$showData['list'] = array();

$this->\_show('activeorders', $showData);

return;

}

$this->db->where\_in('user\_id', $clients);

$client\_list = $this->db->get('users')->result\_array();

foreach ($showData['list'] as $key => $item){

foreach ($client\_list as $value){

if($item['order\_client\_id'] == $value['user\_id']){

$showData['list'][$key]['order\_client\_name'] = $value['user\_name'];

break;

}

}

}

$this->\_show('activeorders', $showData);

}

function orderinfo($order\_id){

$id = $this->\_check();

$showData = array();

$showData['status'] = $this->\_getstatus();

$this->db->where('order\_id', $order\_id);

$info = $this->db->get('orders')->row\_array();

$showData['item'] = $info;

$showData['item']['order\_list'] = json\_decode($showData['item']['order\_list'], true);

$showData['item']['order\_address'] = json\_decode($showData['item']['order\_address'], true);

if($showData['status'] == '4'){

$this->db->where('admin\_status', 2);

$q = $this->db->get('admins')->result\_array();

$showData['cooks'] = $q;

$this->db->where('admin\_status', 1);

$q = $this->db->get('admins')->result\_array();

$showData['delivers'] = $q;

}

$this->db->where('user\_id', $showData['item']['order\_client\_id']);

$client = $this->db->get('users')->row\_array();

$showData['client'] = $client;

$this->\_show('orderinfo', $showData);

}

/\*set personal\*/

function setCook($order\_id,$cook\_id, $r = 0){

$this->\_check();

$ud = array('order\_status' => 'preparing');

$this->db->where('order\_id', $order\_id);

$this->db->update('orders', $ud);

$id = array('cook\_user\_id' => $cook\_id, 'cook\_order\_id' => $order\_id);

$this->db->insert('cooks', $id);

if($r == 0)

redirect('admins/activeorders');

else if($r == 1)

redirect('admins/orderinfo/'.$order\_id);

}

function setDeliver($order\_id,$cook\_id, $r = 0){

$this->\_check();

$ud = array('order\_status' => 'deliver');

$this->db->where('order\_id', $order\_id);

$this->db->update('orders', $ud);

$id = array('deliver\_user\_id' => $cook\_id, 'deliver\_order\_id' => $order\_id);

$this->db->insert('delivers', $id);

if($r == 0)

redirect('admins/activeorders');

else if($r == 1)

redirect('admins/orderinfo/'.$order\_id);

}

function orderCooked($order\_id, $r = 0){

$this->\_check();

$ud = array('cook\_order\_closed' => '1');

$this->db->where('cook\_order\_id', $order\_id);

$this->db->update('cooks', $ud);

$ud = array('order\_status' => 'cooked');

$this->db->where('order\_id', $order\_id);

$this->db->update('orders', $ud);

if($r == 0)

redirect('admins/activeorders');

else if($r == 1)

redirect('admins/orderinfo/'.$order\_id);

}

function orderClose($order\_id, $r = 0){

$this->\_check();

$ud = array('deliver\_order\_closed' => 1);

$this->db->where('deliver\_order\_id', $order\_id);

$this->db->update('delivers', $ud);

$ud = array('order\_status' => 'closed');

$this->db->where('order\_id', $order\_id);

$this->db->update('orders', $ud);

if($r == 0)

redirect('admins/activeorders');

else if($r == 1)

redirect('admins/orderinfo/'.$order\_id);

}

function products(){

$this->\_check();

$showData['products'] = $this->db->query("SELECT \* FROM products")->result\_array();

$this->\_show("products", $showData);

}

function createproduct(){

$this->\_check();

if(!empty($\_POST)){

$uploaddir = 'D:/WAMP/burger1/img/';

$cnt = count($\_FILES['productFile']['name']);

$str = array();

for ($i = 0; $i < $cnt; $i++){

$uploadfile = $uploaddir . basename($\_FILES['productFile']['name'][$i]);

move\_uploaded\_file($\_FILES['productFile']['tmp\_name'][$i], $uploadfile);

$str[$i] = basename($\_FILES['productFile']['name'][$i]);

}

$this->db->query("INSERT INTO products (product\_name, product\_info, product\_description, product\_price, product\_available, product\_image) VALUES ('" . $\_POST['name'] . "', '" . $\_POST['info'] . "', '" . $\_POST['description'] . "', '".$\_POST['price']."', '".$\_POST['available']."','".implode(",", $str)."')");

header("Location: /admins/products");

}

$this->\_show("createproduct");

}

function deleteproduct($id){

$this->\_check();

$this->db->query("DELETE FROM products WHERE product\_id=?", $id);

header("Location: /admins/products");

}

function editproduct($id){

$this->\_check();

if (!empty($\_POST)) {

if (!in\_array(4, $\_FILES['productFile']['error'])) {

$replaced = true;

$uploaddir = 'D:/WAMP/burger1/img';

$cnt = count($\_FILES['productFile']['name']);

$str = array();

for ($i = 0; $i < $cnt; $i++){

$uploadfile = $uploaddir . basename($\_FILES['productFile']['name'][$i]);

move\_uploaded\_file($\_FILES['productFile']['tmp\_name'][$i], $uploadfile);

$str[$i] = basename($\_FILES['productFile']['name'][$i]);

}

} else {

$replaced = false;

}

if ($replaced) {

$this->db->query("UPDATE products SET product\_info='".$\_POST['info']."', product\_name='".$\_POST['name']."', product\_description='".$\_POST['description']."', product\_price='".$\_POST['price']."', product\_available='".$\_POST['available']."', product\_image='".implode(",", $str)."' WHERE product\_id='$id'");

} else {

$this->db->query("UPDATE products SET product\_info='".$\_POST['info']."', product\_name='".$\_POST['name']."', product\_description='".$\_POST['description']."', product\_price='".$\_POST['price']."', product\_available='".$\_POST['available']."' WHERE product\_id='$id'");

}

header("Location: /admins/products");

} else {

$showData['editEntry'] = $this->db->query("SELECT \* FROM products WHERE product\_id=?", $id)->row\_array();

$this->\_show('editproduct', $showData);

}

}

function sliders(){

$sliders = $this->db->get('sliders')->result\_array();

foreach ($sliders as $key => $slider) {

$info = $this->db->get\_where('products', array('product\_id' => $slider['slider\_product\_id']))->row\_array();

$sliders[$key]['product\_name'] = $info['product\_name'];

}

$data['sliders'] = $sliders;

$this->\_show('sliders', $data);

}

function createslider(){

$data['products'] = $this->db->get('products')->result\_array();

if(!empty($\_POST)){

if (!in\_array(4, $\_FILES['productFile']['error'])) {

$uploaddir = 'D:/WAMP/burger1/img/';

$cnt = count($\_FILES['productFile']['name']);

$str = array();

for ($i = 0; $i < $cnt; $i++){

$uploadfile = $uploaddir . basename($\_FILES['productFile']['name'][$i]);

move\_uploaded\_file($\_FILES['productFile']['tmp\_name'][$i], $uploadfile);

$str[$i] = basename($\_FILES['productFile']['name'][$i]);

}

$this->db->insert('sliders', array(

'slider\_product\_id' => $\_POST['product\_id'],

'slider\_image' => implode(",", $str)

)

);

}

header("Location: /admins/products");

}

$this->\_show('createslider', $data);

}

function editslider($id){

$data['products'] = $this->db->get('products')->result\_array();

$this->db->where('slider\_id', $id);

$data['slider'] = $this->db->get('sliders')->row\_array();

$info = $this->db->get\_where('products', array('product\_id' => $data['slider']['slider\_product\_id']))->row\_array();

$data['slider']['product\_name'] = $info['product\_name'];

if(!empty($\_POST)){

if (!in\_array(4, $\_FILES['productFile']['error'])) {

$uploaddir = 'D:/WAMP/burger1/img/';

$cnt = count($\_FILES['productFile']['name']);

$str = array();

for ($i = 0; $i < $cnt; $i++){

$uploadfile = $uploaddir . basename($\_FILES['productFile']['name'][$i]);

move\_uploaded\_file($\_FILES['productFile']['tmp\_name'][$i], $uploadfile);

$str[$i] = basename($\_FILES['productFile']['name'][$i]);

}

$this->db->insert('sliders', array(

'slider\_product\_id' => $\_POST['product\_id'],

'slider\_image' => implode(",", $str)

)

);

}

header("Location: /admins/products");

}

$this->\_show('editslider', $data);

}

function deleteslider($id){

$this->db->delete('sliders', array('slider\_id' => $id));

header("Location: /admins/sliders");

}

}

Б.2 Вихідний код контролеру Welcome

<?php

defined('BASEPATH') OR exit('No direct script access allowed');

class Welcome extends CI\_Controller

{

/\*\*

\* Index Page for this controller.

\*

\* Maps to the following URL

\* http://example.com/index.php/welcome

\* - or -

\* http://example.com/index.php/welcome/index

\* - or -

\* Since this controller is set as the default controller in

\* config/routes.php, it's displayed at http://example.com/

\*

\* So any other public methods not prefixed with an underscore will

\* map to /index.php/welcome/<method\_name>

\* @see https://codeigniter.com/user\_guide/general/urls.html

\*/

public function index()

{

$showData['products'] = $this->db->get('products')->result\_array();

$sliders = $this->db->get('sliders')->result\_array();

foreach ($sliders as $key => $slider) {

$sliders[$key]['product'] = $this->db->get\_where('products', array('product\_id' => $slider['slider\_product\_id']))->row\_array();

}

$showData['sliders'] = $sliders;

$this->load->view('welcome\_message', $showData);

}

function addcart($id, $amount = 1)

{

if (isset($\_COOKIE['cart']))

$cart = json\_decode($\_COOKIE['cart'], true);

else

$cart = array();

$updated = false;

foreach ($cart as $key => $value) {

if ($value['id'] == $id) {

$cart[$key]['amount'] = $value['amount'] + $amount;

$updated = true;

}

}

if ($updated == false) {

$item = array(

'id' => $id,

'amount' => $amount

);

array\_push($cart, $item);

}

$cart = json\_encode($cart);

setcookie('cart', $cart, 3600000000, '/');

setcookie('orderid', rand(0, 10000), 3600000000, '/');

setcookie('customerid', rand(0, 10000), 3600000000, '/');

//print\_r($\_COOKIE);

echo '1';

}

function deletecart($id)

{

if (isset($\_COOKIE['cart'])) {

$cart = json\_decode($\_COOKIE['cart'], true);

foreach ($cart as $key => $value) {

if ($value['id'] == $id)

unset($cart[$key]);

}

setcookie('cart', json\_encode($cart), 3600000000, '/');

header("Location: " . base\_url());

echo 1;

} else echo 0;

}

function updatecart()

{

$count = 0;

$price = 0;

if (isset($\_COOKIE['cart'])) {

$cart = json\_decode($\_COOKIE['cart'], true);

foreach ($cart as $key => $item) {

$info = $this->db->query("SELECT \* FROM products WHERE product\_id = ?", $item['id'])->row\_array();

$count = $count + $item['amount'];

$price = $price + $item['amount'] \* $info['product\_price'];

}

}

echo '

<p class="name">ИТОГО</p>

<p class="count">' . $count . '</p>

<p class="sum">' . $price . ' Р</p>

<a href="#" class="submit-form">готово</a>';

}

function countcart()

{

if (isset($\_COOKIE['cart'])) {

$cart = json\_decode($\_COOKIE['cart'], true);

echo count($cart);

} else {

echo 0;

}

}

function updatecartitems()

{

$text = '';

if (isset($\_COOKIE['cart'])) {

$cart = json\_decode($\_COOKIE['cart'], true);

$text = "";

foreach ($cart as $key => $item) {

$info = $this->db->query("SELECT \* FROM products WHERE product\_id = ?", $item['id'])->row\_array();

$text .= '

<div class="block-total">

<div class="wrapper">

<div class="item-total first">

<p class="name">' . $info['product\_name'] . '</p>

<p class="count">

< ' . $item['amount'] . ' >

</p>

<p class="sum">' . $info['product\_price'] \* $item['amount'] . ' Р </p>

<a href="' . base\_url() . 'welcome/deletecart/' . $item['id'] . '"> x</a>

</div>

</div>

</div>

';

}

} else {

$text .= '

<div class="block-total">

<div class="wrapper">

<div class="item-total first">

<p class="name"></p>

<p class="count">

</p>

<p class="sum"></p>

<a href="' . base\_url() . 'welcome/deletecart/"></a>

</div>

</div>

</div>

';

}

echo $text;

}

function buy1($name, $email, $phone, $address, $delivery, $comment = "")

{

if (isset($\_COOKIE['cart'])) {

$cart = json\_decode($\_COOKIE['cart'], true);

$price = 0;

foreach ($cart as $key => $item) {

$info = $this->db->query("SELECT \* FROM products WHERE product\_id = ?", $item['id'])->row\_array();

$price = $price + $item['amount'] \* $info['product\_price'];

}

$this->db->query("INSERT INTO orders(order\_id, order\_amount, order\_description, order\_customer\_name, order\_customer\_email, order\_customer\_address, order\_customer\_delivery, order\_customer\_phone, order\_comment, customer\_id) VALUES(?,?,?,?,?,?,?,?,?,?)", $\_COOKIE['orderid'], urldecode($price), $\_COOKIE['cart'], urldecode($name), urldecode($email), urldecode($address), urldecode($delivery), urldecode($phone), urldecode($comment), $\_COOKIE['customerid']);

$to = $email;

/\* тема/subject \*/

$subject = "Спасибо за покупку!";

/\* сообщение \*/

$message = 'Вы оформили покупку на сайте woodshield.ru, суммарной стоимостью ' . $price . '.руб.<br> Номер заказа ' . $\_COOKIE['orderid'] . '<br> Спасибо за покупку наш менеджер скоро свяжется с Вами для уточнения подробней заказа.';

/\* Для отправки HTML-почты вы можете установить шапку Content-type. \*/

$headers = "MIME-Version: 1.0\r\n";

$headers .= "Content-Type: text/html; charset=\"utf-8\"\r\n";

/\* дополнительные шапки \*/

$headers .= "From:<info@woodshield.ru>\r\n";

/\* и теперь отправим из \*/

mail($to, $subject, $message, $headers);

setcookie('cart', '');

setcookie('orderid', rand(0, 10000), 3600000000, '/');

setcookie('customerid', rand(0, 10000), 3600000000, '/');

echo 1;

}

}

function thank()

{

$this->load->view('thank');

}

function checkemail($email)

{

$check = $this->db->get\_where('users', array('user\_email' => $email))->row\_array();

if (count($check) > 0) {

echo 0;

} else echo 1;

}

function checkemailpass($email, $pass)

{

$check = $this->db->get\_where('users', array('user\_email' => $email, 'user\_password' => md5($pass)))->row\_array();

if (count($check) > 0) {

echo 1;

} else echo 0;

}

function buy()

{

if (isset($\_COOKIE['cart'])) {

$cart = json\_decode($\_COOKIE['cart'], true);

$price = 0;

if (isset($\_POST['retype'])) {

$address = array(

'city' => $\_POST['city'],

'street' => $\_POST['street'],

'house' => $\_POST['house'],

'entrance' => $\_POST['entrance'],

'floor' => $\_POST['floor'],

'flat' => $\_POST['flat'],

'comment' => $\_POST['comment']

);

$address\_json = json\_encode($address);

$sql\_array = array(

'user\_name' => $\_POST['name'],

'user\_email' => $\_POST['email'],

'user\_phone' => $\_POST['phone'],

'user\_password' => md5($\_POST['pass']),

'user\_address' => $address\_json

);

$this->db->insert('users', $sql\_array);

}

$client = $this->db->get\_where('users', array('user\_email' => $\_POST['email'], 'user\_password' => md5($\_POST['pass'])))->row\_array();

$order = array();

foreach ($cart as $key => $item) {

$info = $this->db->query("SELECT \* FROM products WHERE product\_id = ?", $item['id'])->row\_array();

$buffer['name'] = $info['product\_name'];

$buffer['count'] = $item['amount'];

array\_push($order, $buffer);

$price += $item['amount'] \* $info['product\_price'];

}

$this->db->insert('orders', array(

'order\_client\_id' => $client['user\_id'],

'order\_amount' => $price,

'order\_address' => $client['user\_address'],

'order\_list' => json\_encode($order)

)

);

}

$this->load->view('thank');

//print\_r($order);

}

}

# Додаток В Код шаблонів сторінок

## В.1 Вихідний код шаблону activeorders

<script>

function setCook(sel) {

if(sel.value == ''){

return;

}

$('#setCookLink').attr('href','<?=base\_url('admins/setCook/')?>' + sel.value);

}

function setDeliver(sel) {

if(sel.value == ''){

return;

}

$('#setDeliverLink').attr('href','<?=base\_url('admins/setDeliver/')?>' + sel.value);

}

</script>

<div class="field eight wide column">

<a href="<?=site\_url('admins/')?>" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

</div>

<table class="ui celled table" style="margin-bottom: 50px;">

<thead>

<tr>

<th>ID</th>

<th>Дата</th>

<th>Цена</th>

<th>Имя заказчика</th>

<th>Статус</th>

<th>Действие</th>

<th>Инфо</th>

</tr>

</thead>

<tbody>

<?foreach ($list as $item){?>

<tr>

<td><?=$item['order\_id']?></td>

<td><?=$item['order\_date']?></td>

<td><?=$item['order\_amount']?></td>

<td><?=$item['order\_client\_name']?></td>

<td>

<?switch ($item['order\_status']){

case 'added':

echo 'Добавлен';

break;

case 'preparing':

echo 'Готовится';

break;

case 'cooked':

echo 'Приготовлен';

break;

case 'deliver':

echo 'Доставляется';

break;

case 'closed':

echo 'Закрыт';

break;

}?>

</td>

<td>

<?switch ($status){

case 4:

switch ($item['order\_status']){

case 'added':

echo "<select class=\"ui dropdown\" id=\"cookSelect\" onchange='setCook(this);'>";

echo "<option value=\"\">Выбрать</option>";

foreach ($cooks as $cook){

echo "<option value=\"".$item['order\_id'].'/'.$cook['admin\_id']."\">".$cook['admin\_name']."</option>";

}

echo "</select>";

echo "<a href=\"#\" id='setCookLink' class=\"ui primary button\" style='margin: 10px;'><i class=\"plus icon\"></i>Назначить</a>";

break;

case 'cooked':

echo "<select class=\"ui dropdown\" id=\"deliverSelect\" onchange='setDeliver(this);'>";

echo "<option value=\"\">Выбрать</option>";

foreach ($delivers as $deliver){

echo "<option value=\"".$item['order\_id'].'/'.$deliver['admin\_id']."\">".$deliver['admin\_name']."</option>";

}

echo "</select>";

echo "<a href=\"#\" id='setDeliverLink' class=\"ui primary button\" style='margin: 10px;'><i class=\"plus icon\"></i>Назначить</a>";

break;

}

break;

case 2:

echo "<a href=\"".base\_url('admins/orderCooked/').$item['order\_id']."\" class=\"ui primary button\"><i class=\"reply icon\"></i>Готово</a>";

break;

case 1:

echo "<a href=\"".base\_url('admins/orderClose/').$item['order\_id']."\" class=\"ui primary button\"><i class=\"checkmark box icon\"></i>Готово</a>";

break;

}?>

</td>

<td><a href="<?=base\_url('admins/orderinfo/').$item['order\_id']?>" class="ui orange button"><i class="info icon"></i>Подробнее</a></td>

</tr>

<?}?>

</tbody>

</table>

## В.2 Вихідний код шаблону closeorders

<div class="field eight wide column">

<a href="<?=site\_url('admins/')?>" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

</div>

<div class="field eight wide column" style="margin-top: 15px;">

<?if($curPage != 0){?>

<a href="<?= site\_url('admins/closedorders/'.($curPage - 1)) ?>" class="ui orange button"><i class="arrow left icon"></i>Пред. страница</a>

<?}?>

<?if($curPage < $maxPages - 1){?>

<a href="<?=site\_url('admins/closedorders/'.($curPage + 1))?>" class="ui orange button">След. страница<i class="arrow right icon"></i></a>

<?}?>

</div>

<table class="ui celled table" style="margin-bottom: 50px;">

<thead>

<tr>

<th>ID</th>

<th>Дата</th>

<th>Цена</th>

<th>Статус</th>

</tr>

</thead>

<tbody>

<?foreach ($list as $item){?>

<tr>

<td><?=$item['order\_id']?></td>

<td><?=$item['order\_date']?></td>

<td><?=$item['order\_amount']?></td>

<td><?=$item['order\_status'] == 'closed' ? 'Закрыт' : '' ?></td>

</tr>

<?}?>

</tbody>

</table>

## В.3 Вихідний код шаблону createproduct

<script type="text/javascript">

function changed() {

$('#\_attachmentName').val($('#attachmentName').val());

}

</script>

<a href="/products" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

<h2>Добавить товар</h2>

<div id="add-admin-form">

<form method="post" class="ui large form" enctype="multipart/form-data">

<div class="field eight wide column">

<label>Введите название</label>

<input placeholder="Товар" type="text" name="name" required="required"/>

</div>

<div class="field eight wide column">

<label>Введите краткое описание</label>

<input placeholder="Товар" type="text" name="info" required="required"/>

</div>

<div class="field eight wide column">

<label>Введите состав</label>

<input placeholder="Описание" type="text" name="description" required="required"/>

</div>

<div class="field eight wide column">

<label>Введите цену товара</label>

<input placeholder="Цена" type="text" name="price" required="required"/>

</div>

<div class="field eight wide column">

<label>Наличие</label>

<select name="available" required="required" />

<option value="1">Есть</option>

<option value="0">Нет</option>

</select>

</div>

<!--<div class="field eight wide column">

<div class="ui action input">

<input type="text" id="\_attachmentName">

<label for="attachmentName" class="ui icon large button btn-file">

<i class="file icon"></i>

<input type="file" id="attachmentName" name="productFile[]" onchange="changed();" style="display: none" accept="image/jpeg,image/png" required="required" multiple="multiple" />

</label>

</div>

</div>-->

<input type="file" id="attachmentName" name="productFile[]" accept="image/jpeg,image/png" required="required" multiple="multiple" />

<div class="field eight wide column">

<button class="ui green button" type="submit" name="submit">Готово</button>

</div>

</form>

</div>

## В.4 Вихідний код шаблону createslider

<script type="text/javascript">

function changed() {

$('#\_attachmentName').val($('#attachmentName').val());

}

</script>

<a href="http://moulin-rouge.biz.ua/admin/products" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

<h2>Добавить слайдер</h2>

<div id="add-admin-form">

<form method="post" class="ui large form" enctype="multipart/form-data">

<div class="field eight wide column">

<label>Выберите товар</label>

<select name="product\_id" required="required" />

<?foreach($products as $key => $product){?>

<option value="<?=$product['product\_id']?>"><?=$product['product\_name']?></option>

<?}?>

</select>

</div>

<div class="field eight wide column">

<label>Выберите изображение слайдера</label>

<input type="file" id="attachmentName" name="productFile[]" accept="image/jpeg,image/png" required="required" multiple="multiple" />

</div>

<div class="field eight wide column">

<button class="ui green button" type="submit" name="submit">Готово</button>

</div>

</form>

</div>

## В.5 Вихідний код шаблону editproduct

<!--<script type="text/javascript">

function changed() {

$('#\_attachmentName').val($('#attachmentName').val());

}

</script>-->

<a href="/admin/products" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

<h2>Редактировать товар "<?=$editEntry['product\_name']?>"</h2>

<div id="add-admin-form">

<form method="post" class="ui large form" enctype="multipart/form-data">

<div class="field eight wide column">

<label>Введите имя </label>

<input placeholder="Товар" type="text" name="name" required="required" value="<?=$editEntry['product\_name']?>"/>

</div>

<div class="field eight wide column">

<label>Введите краткое описание </label>

<input placeholder="Товар" type="text" name="info" required="required" value="<?=$editEntry['product\_info']?>"/>

</div>

<div class="field eight wide column">

<label>Введите состав</label>

<input placeholder="Описание" type="text" name="description" required="required" value="<?=$editEntry['product\_description']?>"/>

</div>

<div class="field eight wide column">

<label>Введите цену товара</label>

<input placeholder="Цена" type="text" name="price" required="required" value="<?=$editEntry['product\_price']?>"/>

</div>

<div class="field eight wide column">

<label>Наличие</label>

<select name="available" required="required" />

<option value="1" <?=$editEntry['product\_available'] == 1 ? "selected" : ""?>>Есть</option>

<option value="0" <?=$editEntry['product\_available'] == 0 ? "selected" : ""?>>Нет</option>

</select>

</div>

<!--<div class="field eight wide column">

<div class="ui action input">

<input type="text" id="\_attachmentName">

<label for="attachmentName" class="ui icon large button btn-file">

<i class="file icon"></i>

<input type="file" id="attachmentName" name="productFile" onchange="changed();" style="display: none" accept="image/jpeg,image/png" />

</label>

</div>

</div>-->

<input type="file" id="attachmentName" name="productFile[]" accept="image/jpeg,image/png" multiple="multiple" />

<div class="field eight wide column">

<button class="ui green button" type="submit" name="submit">Готово</button>

</div>

</form>

</div>

## В.6 Вихідний код шаблону editslider

<!--<script type="text/javascript">

function changed() {

$('#\_attachmentName').val($('#attachmentName').val());

}

</script>-->

<a href="/admin/products" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

<h2>Редактировать слайдер "<?=$slider['product\_name']?>"</h2>

<div id="add-admin-form">

<form method="post" class="ui large form" enctype="multipart/form-data">

<div class="field eight wide column">

<label>Выберите товар</label>

<select name="product\_id" required="required" />

<?foreach($products as $key => $product){?>

<option value="<?=$product['product\_id']?>"><?=$product['product\_name']?></option>

<?}?>

</select>

</div>

<div class="field eight wide column">

<label>Выберите изображение слайдера</label>

<input type="file" id="attachmentName" name="productFile[]" accept="image/jpeg,image/png" required="required" multiple="multiple" />

</div>

<div class="field eight wide column">

<button class="ui green button" type="submit" name="submit">Готово</button>

</div>

</form>

</div>

## В.7 Вихідний код шаблону footer

</body>

</html>

## В.8 Вихідний код шаблону header

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta http-equiv="X-UA-Compatible" content="IE=edge"/>

<meta http-equiv="Cache-Control" content="no-cache"/>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

<meta name="viewport" content="width=device-width, initial-scale=1"/>

<meta name="description" content=""/>

<meta name="author" content=""/>

<title>burgers</title>

<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.11.2.min.js"></script>

<script src="<?=base\_url('semantic/semantic.min.js')?>"></script>

<link rel="stylesheet" href="<?=base\_url('semantic/semantic.min.css')?>"/>

<link href="<?=base\_url('style.css')?>" rel="stylesheet"/>

<script>

function bodyLoad() {

$('.ui.dropdown').dropdown();

$('.ui.accordion').accordion();

}

</script>

</head>

<body onload="bodyLoad();">

## В.9 Вихідний код шаблону index

<a href="<?= site\_url('admins/activeorders') ?>" class="ui button">Активные заказы</a>

<? if ($status == 4) { ?>

<a href="<?= site\_url('admins/closedorders') ?>" class="ui button">Закрытые заказы</a>

<? } ?>

<a href="<?= site\_url('admins/loginedit') ?>" class="ui button">Изменить данные входа</a>

<a href="<?= site\_url('admins/logout') ?>" class="ui button">Выйти</a>

## В.10 Вихідний код шаблону login

<div id="login">

<h1>Привет, админ!</h1>

<h2>Вход</h2>

<? if ($loginError) { ?>

<div class="ui negative message">

<i class="close icon"></i>

<div class="header">

Неверный email или пароль.

</div>

<p id="errMess">

Проверьте правильность ввода и попробуйте ещё раз.

</p>

</div>

<? } ?>

<form method="post" class="ui large form">

<div class="field">

<label>Login</label>

<input placeholder="Login" type="text" name="login" id="email" required="required"/>

</div>

<div class="field">

<label>Password</label>

<input placeholder="Password" type="password" name="password" id="pass" required="required"/>

</div>

<button class="ui fluid primary button" id="submit">OK</button>

</form>

</div>

## В.11 Вихідний код шаблону loginedit

<a href="<?=site\_url('admins/')?>" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

<div id="add-admin-form">

<form method="post" class="ui large form" enctype="application/x-www-form-urlencoded">

<div class="field eight wide column">

<label>Введите новый логин</label>

<input placeholder="Логин" type="text" name="login" required="required" value="<?=$info['admin\_login']?>"/>

</div>

<div class="field eight wide column">

<label>Введите новый пароль</label>

<input placeholder="Пароль" type="password" name="password" required="required"/>

</div>

<div class="field eight wide column">

<button class="ui green button" type="submit" name="submit">Готово</button>

</div>

<?if($message){?>

<div class="ui green message">Данные успешно обновлены!</div>

<?}?>

</form>

</div>

## В.12 Вихідний код шаблону orderinfo

<script>

function setCook(sel) {

if(sel.value == ''){

return;

}

$('#setCookLink').attr('href','<?=base\_url('admins/setCook/')?>' + sel.value);

}

function setDeliver(sel) {

if(sel.value == ''){

return;

}

$('#setDeliverLink').attr('href','<?=base\_url('admins/setDeliver/')?>' + sel.value);

}

</script>

<div class="field eight wide column">

<a href="<?=site\_url('admins/activeorders')?>" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

</div>

<table class="ui celled table" style="margin-bottom: 50px;">

<thead>

<tr>

<th width="25%">Поле</th>

<th>Значение</th>

</tr>

</thead>

<tbody>

<tr>

<td>ID</td>

<td><?=$item['order\_id']?></td>

</tr>

<tr>

<td>Дата</td>

<td><?=$item['order\_date']?></td>

</tr>

<tr>

<td>Цена</td>

<td><?=$item['order\_amount']?></td>

</tr>

<tr>

<td>Имя заказчика</td>

<td><?=$client['user\_name']?></td>

</tr>

<tr>

<td>Список</td>

<td>

<?foreach ($item['order\_list'] as $value){?>

<?=$value['name'].'x'.$value['count'].'<br>'?>

<?}?>

</td>

</tr>

<tr>

<td>Адрес</td>

<td>

<?='Город: '.$item['order\_address']['city'].', улица: '.$item['order\_address']['street'].', дом: '.$item['order\_address']['house']?>

<?=', подъезд: '.$item['order\_address']['entrance'].', этаж: '.$item['order\_address']['floor'].', кБ.: '.$item['order\_address']['flat']?>

<?=', домофон: '.$item['order\_address']['intercom'].'<br>Комментарий: '.$item['order\_address']['comment']?>

</td>

</tr>

<tr>

<td>Статус</td>

<td>

<?switch ($item['order\_status']){

case 'added':

echo 'Добавлен';

break;

case 'preparing':

echo 'Готовится';

break;

case 'cooked':

echo 'Приготовлен';

break;

case 'deliver':

echo 'Доставляется';

break;

case 'closed':

echo 'Закрыт';

break;

}?>

</td>

</tr>

<tr>

<td>Действие</td>

<td>

<?switch ($status){

case 4:

switch ($item['order\_status']){

case 'added':

echo "<select class='ui dropdown' id='cookSelect' onchange='setCook(this);'>";

echo "<option value=''>Выбрать</option>";

foreach ($cooks as $cook){

echo "<option value='".$item['order\_id'].'/'.$cook['admin\_id']."/1"."'>".$cook['admin\_name']."</option>";

}

echo "</select>";

echo "<a href='#' id='setCookLink' class='ui primary button' style='margin: 10px;'><i class='plus icon'></i>Назначить</a>";

break;

case 'cooked':

echo "<select class='ui dropdown' id='deliverSelect' onchange='setDeliver(this);'>";

echo "<option value=''>Выбрать</option>";

foreach ($delivers as $deliver){

echo "<option value='".$item['order\_id'].'/'.$deliver['admin\_id']."/1"."'>".$deliver['admin\_name']."</option>";

}

echo "</select>";

echo "<a href='#' id='setDeliverLink' class='ui primary button' style='margin: 10px;'><i class='plus icon'></i>Назначить</a>";

break;

}

break;

case 2:

if($item['order\_status'] == 'preparing')

echo "<a href='".base\_url('admins/orderCooked/').$item['order\_id']."/1"."' class='ui primary button'><i class='reply icon'></i>Готово</a>";

break;

case 1:

if($item['order\_status'] == 'deliver')

echo "<a href='".base\_url('admins/orderClose/').$item['order\_id']."/1"."' class='ui primary button'><i class='checkmark box icon'></i>Готово</a>";

break;

}?>

</td>

</tr>

</tbody>

</table>

## В.13 Вихідний код шаблону products

<table class="ui celled table">

<a href="/admin" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

<a href="categories" class="ui orange button"><i class="pencil icon"></i>Редактировать категории</a>

<a href="parameters" class="ui orange button"><i class="pencil icon"></i>Редактировать параметры</a>

<a href="createproduct" class="ui green button"><i class="plus icon"></i>Добавить товар</a>

<thead>

<tr>

<th>ID</th>

<th>Название</th>

<th>Цена</th>

<th>Дата добавления</th>

<th>Статус наличия</th>

<th>Редактировать</th>

<th>Удалить</th>

<th>Изменить статус</th>

</tr>

</thead>

<tbody>

<?foreach ($products as $product){?>

<tr>

<td><?=$product['product\_id']?></td>

<td><?=$product['product\_name']?></td>

<td><?=$product['product\_price']?></td>

<td><?=$product['product\_created']?></td>

<td><?=$product['product\_available'] == 1 ? "Есть" : "Нет"?></td>

<td><a href="editproduct/<?=$product['product\_id']?>" class="ui orange button"><i class="pencil icon"></i>Редактировать</a></td>

<td><a href="deleteproduct/<?=$product['product\_id']?>" class="ui red button"><i class="trash icon"></i>Удалить</a></td>

<td><a href="changeproductstatus/<?=$product['product\_id']?>/<?=$product['product\_available']?>" class="ui <?=$product['product\_available'] == 1 ? "red" : "green"?> button"><i class="<?=$product['product\_available'] == 1 ? "off" : "on"?> icon"></i>Изменить статус</a></td>

</tr>

<?}?>

</tbody>

</table>

## Г.14 Вихідний код шаблону sliders

<table class="ui celled table">

<a href="/admins" class="ui primary button"><i class="arrow left icon"></i>Назад</a>

<a href="createslider" class="ui green button"><i class="plus icon"></i>Добавить слайдер</a>

<thead>

<tr>

<th>ID</th>

<th>Товар</th>

<th>Редактировать</th>

<th>Удалить</th>

</tr>

</thead>

<tbody>

<?foreach ($sliders as $slider){?>

<tr>

<td><?=$slider['slider\_id']?></td>

<td><?=$slider['product\_name']?></td>

<td><a href="editslider/<?=$slider['slider\_id']?>" class="ui orange button"><i class="pencil icon"></i>Редактировать</a></td>

<td><a href="deleteslider/<?=$slider['slider\_id']?>" class="ui red button"><i class="trash icon"></i>Удалить</a></td>

</tr>

<?}?>

</tbody>

</table>

# Додаток Г Екрані форми розробленої адміністраторської панелі

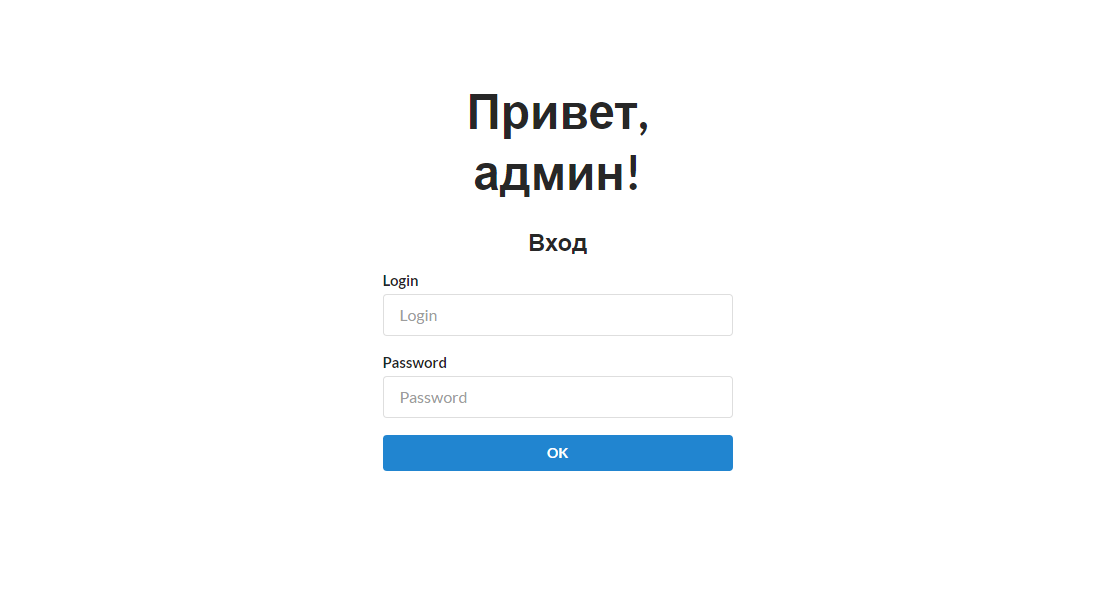


Рисунок Г.1 – Сторінка авторизації до адміністраторської панелі

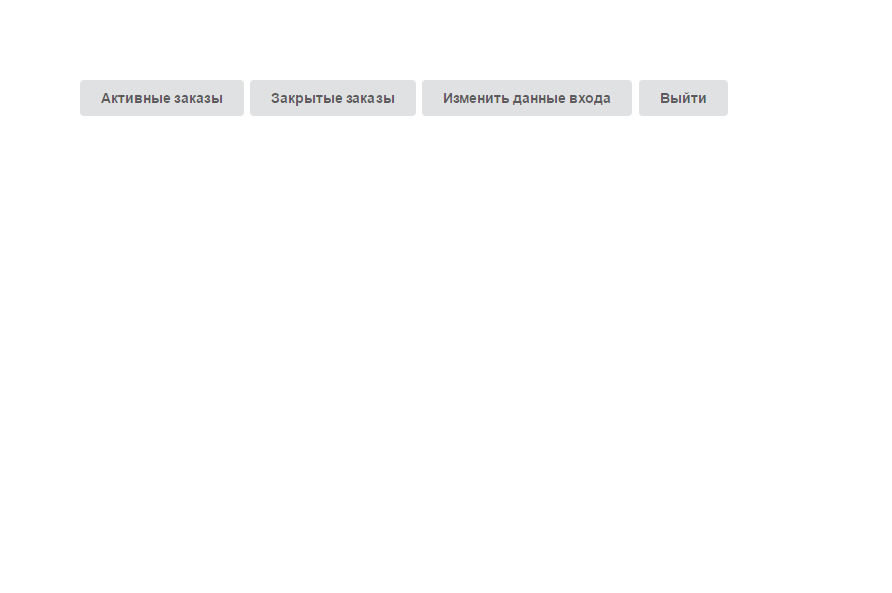


Рисунок Г.2 – Головна сторінка адміністраторської панелі для адміністратора

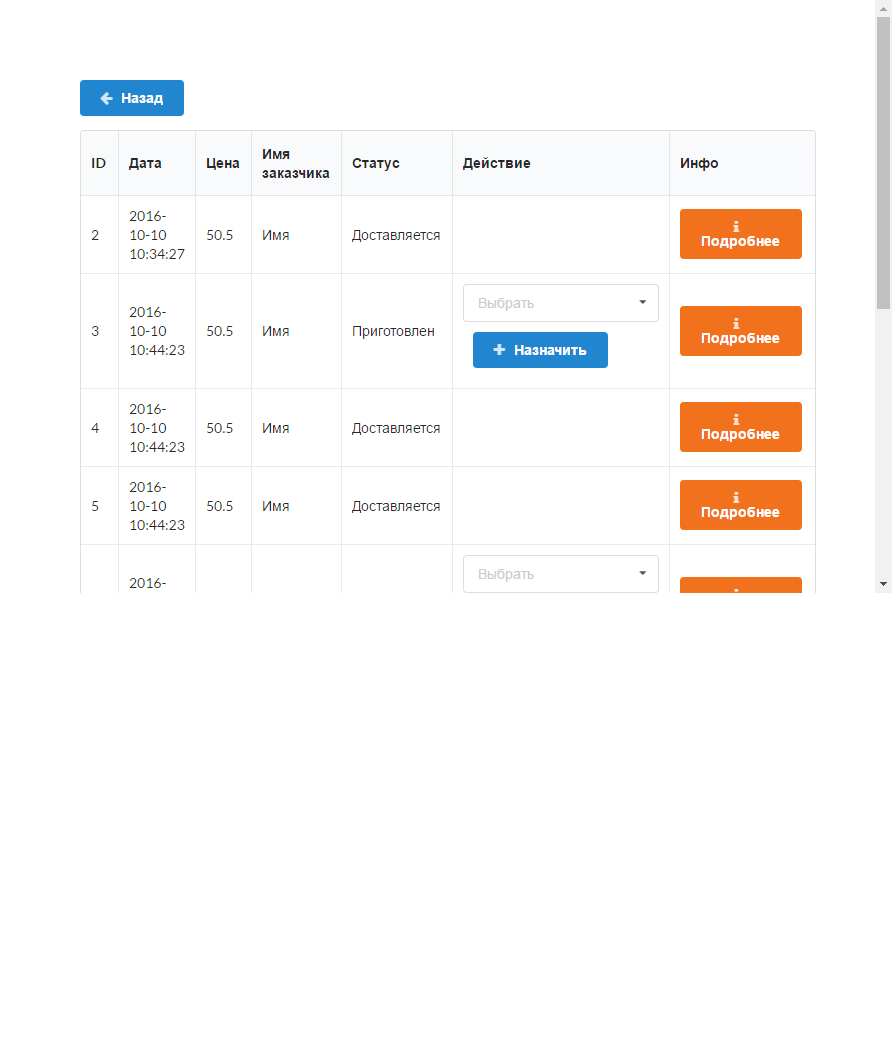


Рисунок Г.3 – Сторінка обробки замовлень

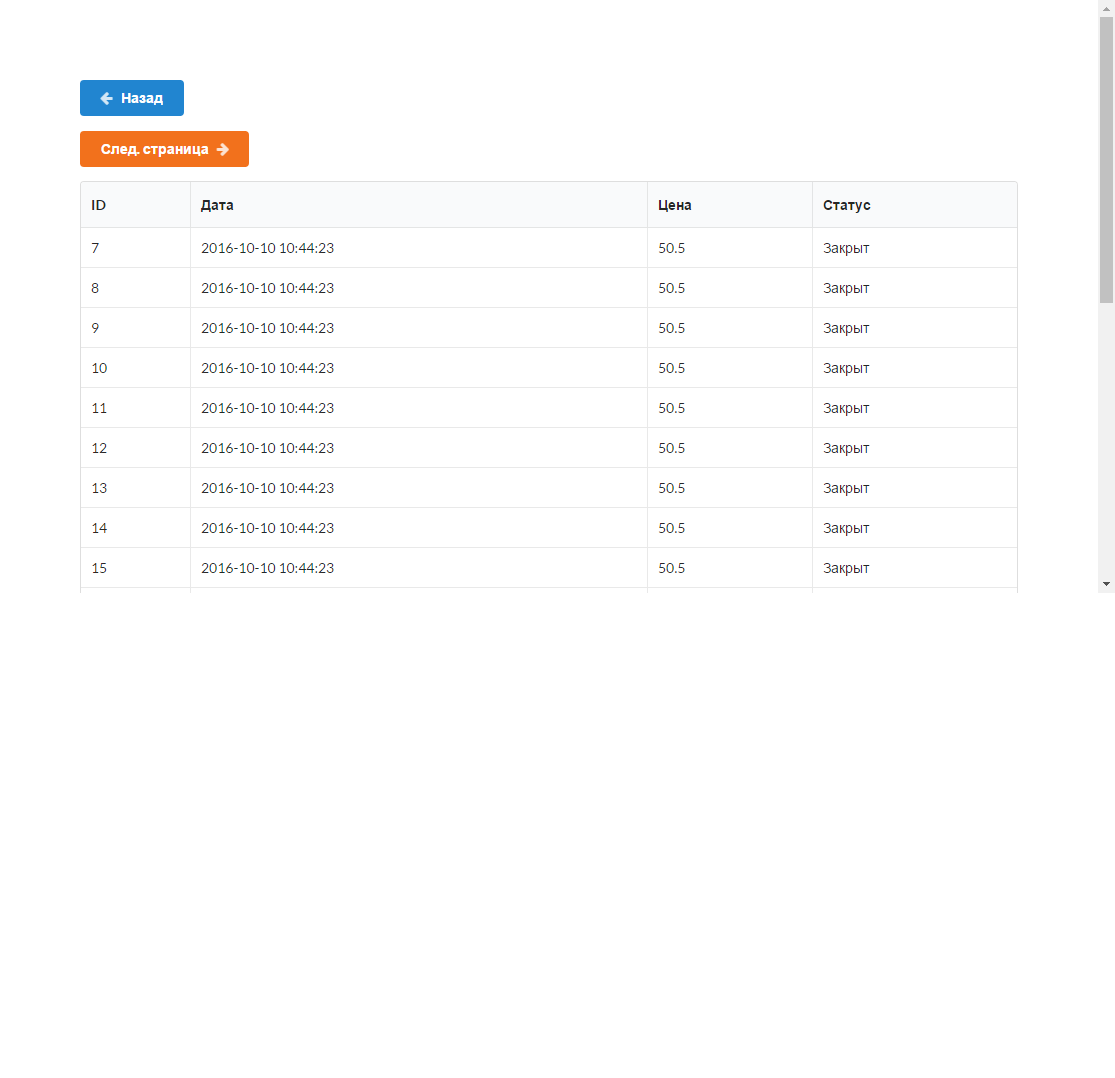


Рисунок Г.4 – Сторінка завершених замовлень

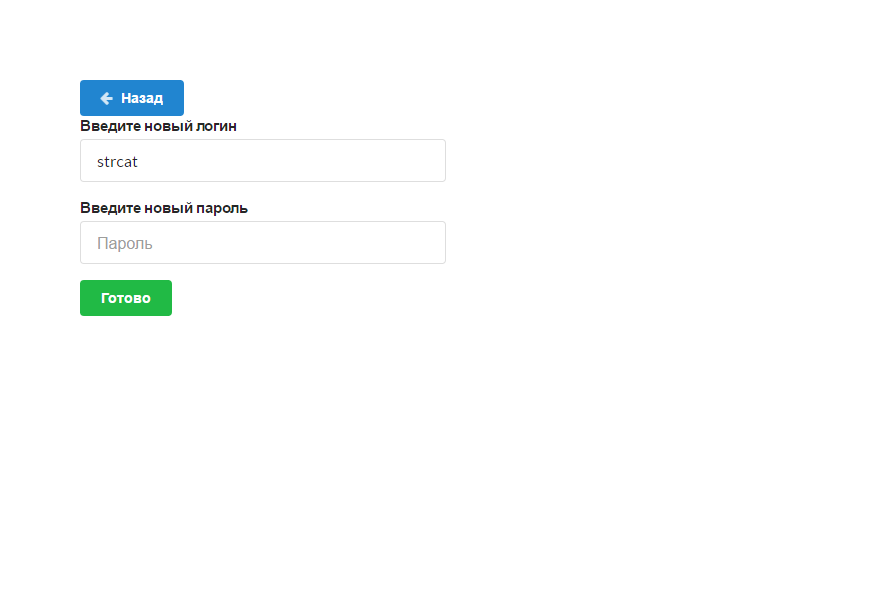


Рисунок Г.5 – Сторінка зміни даних для авторизації

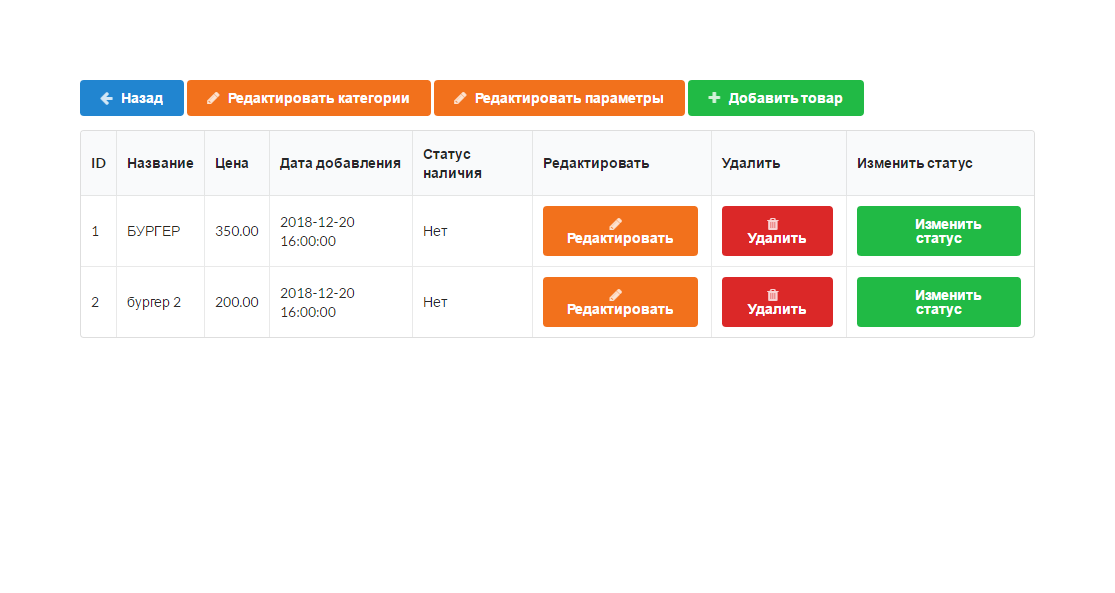


Рисунок Г.6 – Сторінка меню редагування товарів

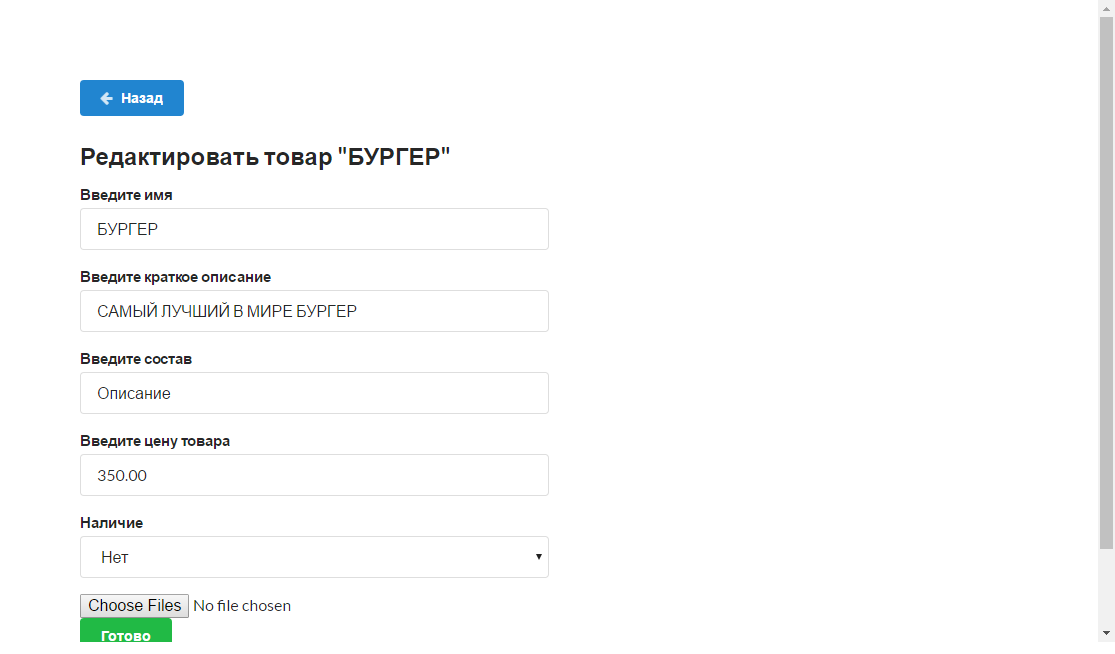


Рисунок Г.7 – Сторінка редагування товару

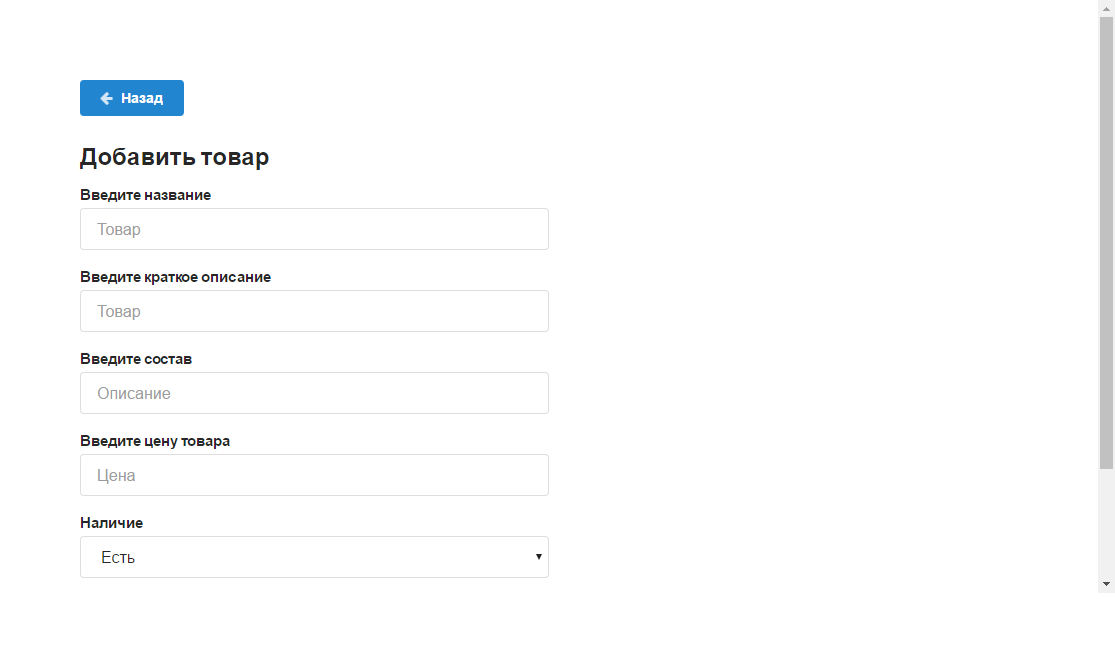


Рисунок Г.8 – Сторінка додання товару