Міністерство освіти и науки України
Харківський національний університет радіоелектронікі

Факультет комп'ютерно наук_____
(Повна назва)
Кафедра Системотехніки_____
(Повна назва)

КУРСОВИЙ ПРОЕКТ
ПОЯСНЮВАЛЬНА ЗАПИСКА

«Магазин кондитерських виробів»
(Тема роботи)
з дисципліни Технології комп'ютерного проектування_____
(Назва дисципліни)

Керівник Міщеряков Ю.В._____
(Посада, прізвище, ініціали)

Студент Савченко С.С._____
(Група, прізвище, ініціали)

Робота захищено з оцінкою «_____»

«_____» _____ 20__ р.

комісія:

_____
(Підпис, посада, прізвище, ініціали)

_____
(Підпис, посада, прізвище, ініціали)

_____
(Підпис, посада, прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

факультет *коп'ютерних наук*

Кафедра *системотехніки*

Напрям *122 - Комп'ютерні науки*

курс *1 група* ІТКНу-18-1 семестр 2

ЗАВДАННЯ
НА КУРСОВЕ ПРОЕКТУВАННЯ

студенту Савченко Сергію Сергійовичу

(Прізвище, ім'я, по батькові)

1. Тема роботи: «Магазин кондитерських виробів»

2. Термін здачі студентом закінченої роботи: 18.06.19

3. Вихідні дані до проекту: Розробити серверну и клієнтську частини магазину кондитерських товарів. Серверна частина являє собою реалізацію бази даних, розроблення для платформи СУБД MySQL. Клієнтська частина має забезпечувати виконання таких бізнес- функцій як оформлення замовлення, реєстрація та авторизація, перегляд виконаних заказів та їх фільтрація по статусу. Бізнес-функції системи для незареєстрованих користувачів: переглядання каталогу, фільтрація каталогу за потрібним параметром, реєстрація та авторизація на сайті, можливість виконати замовлення. Бізнес-функції системи для зареєстрованих користувачів: можливість виконати замовлення за скороченою формою; зміна профілю (ім'я, пароль, контактна інформація та лист інтересів тощо); переглядання оформлених замовлень та їх станів; перегляд отриманих знижок на товари. Бізнес-функції системи для менеджерів: вхід в систему керування замовленнями та каталогом; Додавання нових виробів; додавання нових категорій; зміна статусу обраного замовлення; формування звіту замовлень за визначенням період. Операційна система - Windows XP або вище, програмне забезпечення: утиліта командного рядка MySQL Command Line Client; програмний пакет Workbench; середа розробки Visual Studio 2012 та вище, CASE-засіб All Fusion Data Modeler (ERWin), засіб об’єктно-орієнтованого моделювання UML StarUML.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці): Провести аналіз бізнес-процесів (бізнес-функцій)

предметної області та виділити ті з них, які вимагають автоматизації; сформулювати та оформити вимоги до інформаційної системи; створити функціональну модель інформаційної системи («TO-BE»), використовуючи стандарт IDEF0; провести функціональне моделювання, визначити й уточнити вимоги до інформаційної системи; провести логічне моделювання БД з Використання стандарту IDEF1X; обґрунтувати вибір платформи СУБД для реалізації БД; провести фізичне моделювання БД з використання стандарту IDEF1X; провести UML-моделювання проектованої клієнтської частини інформаційної системи, створити діаграму прецедентів (Use Case Diagram), діаграму послідовності дій (Sequence Diagram), діаграму станів (Statechart Diagram), діаграму активності (Activity Diagram) и діаграму класів (Class Diagram); провести аналіз и виділити основні бізнес-процеси з поділом бізнес-функцій, що виконуються на стороні клієнтської и серверної частин інформаційної системи; реалізувати фізичну модель БД для обраної платформи СУБД, створити серверну частину інформаційної системи; реалізувати посилальну цілісність даних, а також одну з функцій бізнес-процесу на стороні серверної частини інформаційної системи; реалізувати один або кілька бізнес-процесів на стороні клієнтської частини інформаційної системи, розробити інтерфейс доступу до БД; Розробити відповідно до ГОСТ 34.69890 експлуатаційний документ «Керівництво користувача»; підготувати відповідно до ГОСТ 19.401-78 програмний документ «Текст програми». реалізувати фізичну модель БД для обраної платформи СУБД, створити серверну частину інформаційної системи; реалізувати посилальну цілісність даних, а також одну з функцій бізнес-процесу на стороні серверної частини інформаційної системи; реалізувати один або кілька бізнес-процесів на стороні клієнтської частини інформаційної системи, розробити інтерфейс доступу до БД; підготувати відповідно до ГОСТ 19.401-78 програмний документ «Текст програми».

5. Перелік графічного матеріалу (з точним визначенням обов'язкових креслень): _____

6. Дата видачі завдання: _____

Керівник роботи _____

(Підпис) (прізвище, ім'я, по батькові)

студент _____

(Підпис) (прізвище, ім'я, по батькові)

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін Виконання етапів роботи	Примітка
1	Аналіз предметної області		
2	Визначення основних бізнес-функцій інформаційної системи		
3	Визначення функцій інтерфейсу клієнтської частини інформаційної системи		
4	Розробка серверної частини інформаційної системи		
5	Логічне та фізичне моделювання даних		
6	Створення і заповнення баз даних		
7	Розробка підтримки цілісності даних		
8	Реалізація бізнес-функцій інформаційної системи на стороні сервера MySQL (процедур, функцій, тригерів)		
9	Розробка програмного забезпечення клієнтської частини інформаційної системи		
10	Тестування розроблення програмного забезпечення		
11	Підготовка пояснювальної записки та її додатків		

Керівник роботи (проекту)_____

(Підпис) (посада, прізвище, ініціали)

студент_____

(Підпис) (посада, прізвище, ініціали)

«____» _____ 20___ р

РЕФЕРАТ

Пояснювальна записка до курсового проекту: 48 с., 15 рис., 2 додатка, 10 джерел інформації.

БАЗА ДАНИХ, СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ, MYSQL, ІНФОРМАЦІЙНА СИСТЕМА, ІНТЕРФЕЙС ДОСТУПУ, СИСТЕМА ОБЛІКУ, ЕЛЕКТРОННА КОМЕРЦІЯ

Об'єктом досліджень курсового проекту є процес електронного бізнесу інтернет магазину, оформлення виконання замовлень, документування облікових даних.

Предметом досліджень курсового проекту є інформаційні технології і програмні методи створення клієнтської і серверної частин інформаційної системи, що дозволяє автоматизувати облік продажів інтернет магазину.

Мета досліджень: розробка клієнтської і серверної частин інформаційної системи інтернет магазину кондитерських виробів.

Методи дослідження – системний підхід, методи структурного аналізу і моделювання реляційних баз даних, методи реляційної алгебри і реляційного числення, методи проектування клієнт-серверних додатків для Інтранет- та Інтернет-мереж, подієвого об'єктно-орієнтованого програмування.

У роботі проведено аналіз предметної області, що належить до діяльності інтернет магазину кондитерських виробів. Для визначення та уточнення вимог до розроблюваної ІС проведено функціональне моделювання (відповідно до стандарту IDEF0), логічне і фізичне моделювання даних (відповідно до стандарту IDEF1X). Розроблено діаграми UML-моделі. Проведено проектування клієнтської і серверної частин ІС інтернет магазину кондитерських виробів. За результатами тестування проведено аналіз відповідності розробленого програмного забезпечення ІС висунутим вимогам.

Галузь застосування – підтримка електронного бізнесу інтернет магазину кондитерських виробів.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ІС – інформаційна система.

СУБД – система управління базами даних.

Замовлення – бронювання купівлі в інтернет-магазині.

Інтернет-магазин – сайт в мережі інтернет, в якому виконується продаж різних товарів.

Товар – річь, в інтернет-магазині, яку користувач може купити.

Програмна платформа – середа виконання коду, в якій повинен виконуватися фрагмент програмного забезпечення, що розробляється.

ВСТУП

Відомо, що в останнє десятиріччя інтернет відчуває значні темпи зростання популярності серед людей, що мають різне соціальне становище, будь-якого віку і національностей. Інтернет є вельми зручним і швидким засобом передачі інформації і при цьому досить дешевим. З кожним роком в Україні кількість користувачів мережі інтернет збільшується в порівнянні з попереднім.

Ще з початку третього тисячоріччя в інтернеті пішов великий ажіотаж на інтернет-магазини. Вони стали дуже комфортним способом шукати, покупати та продавати найрізноманітніші товари. А найбільшою перевагою інтернет-магазину можна вважати те, що покупки можна роботи не виходячи з дому.

Стан ринку кондитерських виробів в Україні в останні роки визначалося двома обставинами. По-перше, українці люблять солодке, і тому на нашому ринку велика кількість виробників різноманітної кондитерської продукції, які конкурують між собою за увагу покупців. По-друге, кондитерські вироби все-таки не входять в перелік першочергових продуктів харчування, що викликає зменшення їх споживання під час економічних потрясінь.

При проведенні аналізу було виявлено, що в інтернет-магазини кондитерських виробів є дуже популярними. Серед найбільших інтернет-магазинів в Україні можна виділити наступні: «Синдикат вкуса», «BlackBerry Cake», «Нота вкуса», та інші. Але в наявності також є багато невеликих інтернет-магазинів, які теж користуються великим попитом. Це показує, що інтернет-магазин кондитерських виробів є досить актуальним проектом.

Через актуальність даної проблеми було вирішено зробити метою курсового проектування створення інтернет магазину кондитерських виробів, який може використовуватися в галузі роздрібної торгівлі кондитерських виробів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної області, яка визначає діяльність підприємства

Інтернет-магазин - сайт, який торгує товарами в інтернеті. Дозволяє користувачам сформувати замовлення на покупку, вибрати спосіб оплати і доставки замовлення в мережі Інтернет. Предметною областю даної курсової роботи є «Інтернет магазин кондитерських виробів».

Увійшовши на сайт користувач на головній сторінці сайту бачить каталог товарів. Зазвичай, на сторінці каталогу товарів можна побачити фільтри, завдяки яким користувач може більш точно підібрати кондитерський виріб.

Вибравши необхідні товари, користувач зазвичай має можливість тут же на сайті перейти на сторінку оформлення замовлення. Сукупність відібраних товарів, спосіб оплати і доставки являють собою закінчене замовлення, яке оформлюється на сайті шляхом повідомлення мінімально необхідної інформації про покупця. Інформація про покупця може зберігатися в базі даних магазину якщо бізнес-модель магазину розрахована на повторні покупки, або ж відправлятися разово. В інтернет-магазинах, розрахованих на повторні покупки, також ведеться відстеження повернень відвідувача і історія покупок. Часто при оформленні замовлення передбачається можливість повідомити деякі додаткові побажання від покупця продавцю.

За доставку відповідає адміністратор служби доставки. В його обов’язки входить призначення відповідальної за доставку особи (кур’єра). Кожне замовлення містить у собі дані про коштовність товару, адресу доставки, та дату, коли було зроблено замовлення. Завдяки спеціальному пристрою та програмному забезпеченню кур’єр зможе відстежувати замовлення, які були назначені на нього, та змінювати статус замовлення на «Доставлено» якщо замовлення прибуло до пункту призначення і клієнт здійснив оплату. Уся інформація про статуси замовлень та самі замовлення відображаються в програмному забезпеченні менеджера служби доставки. Менеджер також повинен назначати на замовлення певного кур’єра виходячи з того де кур’єр базується.

1.2 Визначення основних бізнес процесів, що вимагають автоматизації

У цьому проекті виконується автоматизація процесу оформлення замовлення користувачем інтернет-магазину кондитерських виробів.

Для бізнес процесу оформлення замовлення можна автоматизувати наступні функції:

a) Незареєстрований користувач:
   a. Перегляд каталогу кондитерських виробів.
   b. Пошук кондитерського виробу за категоріями а також пошук по найменуванню
   c. Перегляд інформації про терміни придатності і склад обраного вироби
   d. Додавання в корзину (швидке замовлення для незареєстрованих користувачів)
   e. Оформлення замовлення (для цього достатньо заповнити контактну інформацію: телефон та / або e-mail, адреса)
   f. Реєстрація на сайті
   g. Вхід на сайт.
b) Зареєстрований користувач (роль: user):
   a. Зміна профілю (e-mail, ім'я, пароль ...)
   b. Перегляд оформлених замовлень і їх станів
   c. Перегляд постійних знижок по, вже зібраної, статистикою користувача
   d. Вихід
c) Менеджер (роль: admin):
   a. Додавання нових позицій
   b. Додавання категорій
   c. Розподіл товарів за категоріями
   d. Перегляд замовлень з фільтрацією по статусу (новий, в виконанні, виконаний ...)
   e. Зміна статусу обраного замовлення
   f. Формування звіту
   g. Вихід.

1.3 Постановка завдання

Основним завданням цього курсового проекту є створення клієнт-серверного проекту інтернет-магазину кондитерських виробів. Основним бізнес процесом цього проекту є оформлення замовлення користувачем, та підтвердження цього замовлення менеджером, що буде нести в собі взаємодію ролей.

2 РОЗРОБКА ВИМОГ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Визначення функціональних вимог до інформаційної системи

Для визначення функціональних вимог до інформаційної системи інтернет-магазин кондитерських виробів була створена функціональна модель з використанням стандарту IDEF0. Модель SADT являє собою серію діаграм, що розбивають складний об’єкт на складові частини та надаються у вигляді блоків.

На концептуальній діаграмі (рис 2.1), яка створювалася з точки зору менеджера, можна побачити основний бізнес-процес та інтерфейси. Інтерфейси поділяються на:

a) Керуюча інформація:
   a. Порядок робіт;
   b. Каталог товарів.
b) Інформація для обробки:
   a. Переваги клієнта.
c) Результати:
   a. Сповіщення користувачу;
   b. Готове замовлення.
d) Механізми (Людина чи автоматизована система):
   a. Менеджер/Адміністратор;
   b. Інформаційна система;
   c. Користувач;

Після створення концептуальної діаграми треба зробити її декомпозицію на інші діаграми, які будуть ілюструвати структуру «батьківської» діаграми.

У наступній схемі (рис 2.2), яку ми отримали після декомпозиції можна побачити наступні блоки:

а) Враховувати замовлення;

а) Сформувати корзину;

а) Сформувати замовлення;

а) Обробити замовлення.

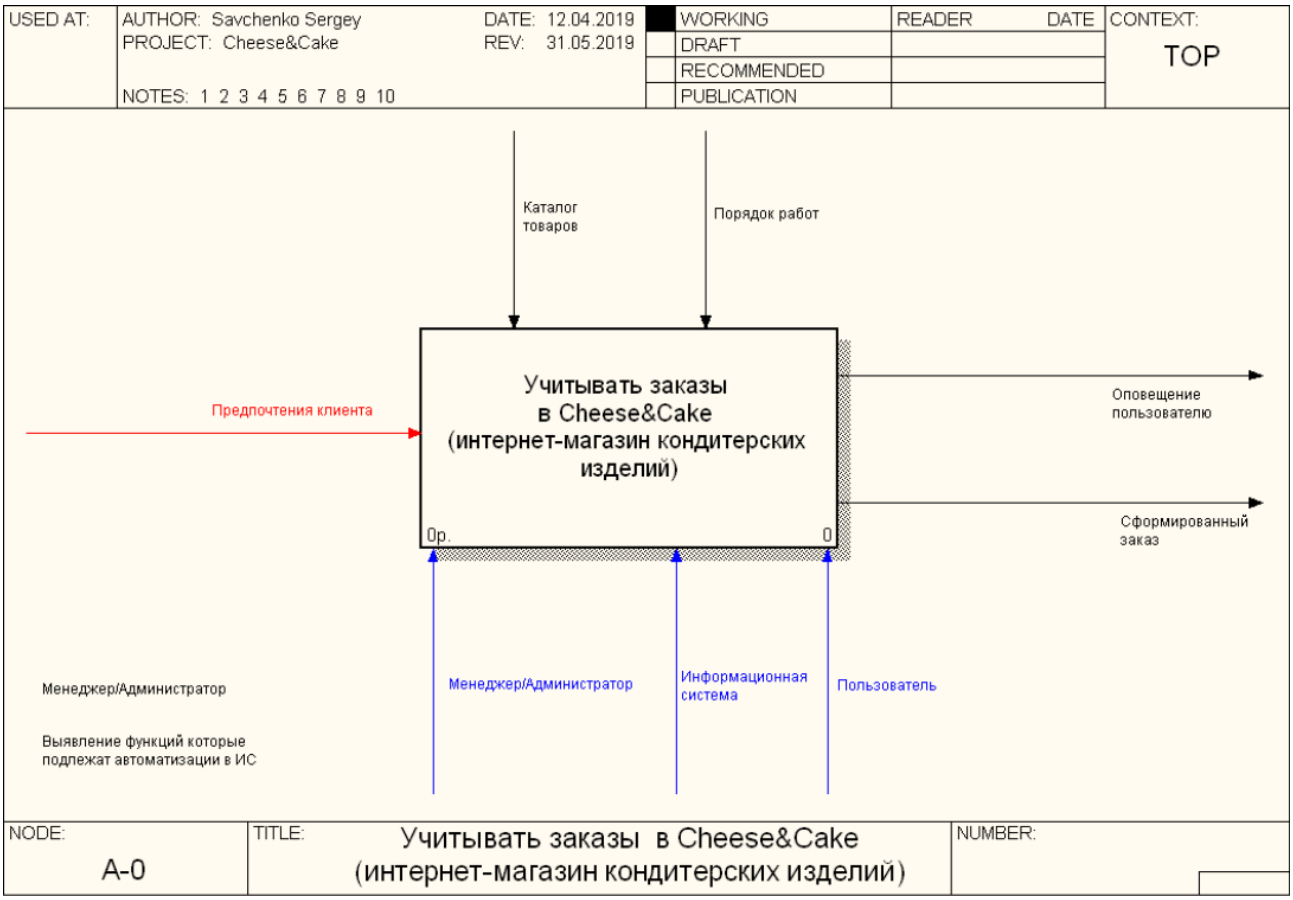

Рисунок 2.1 – Концептуальна діаграма

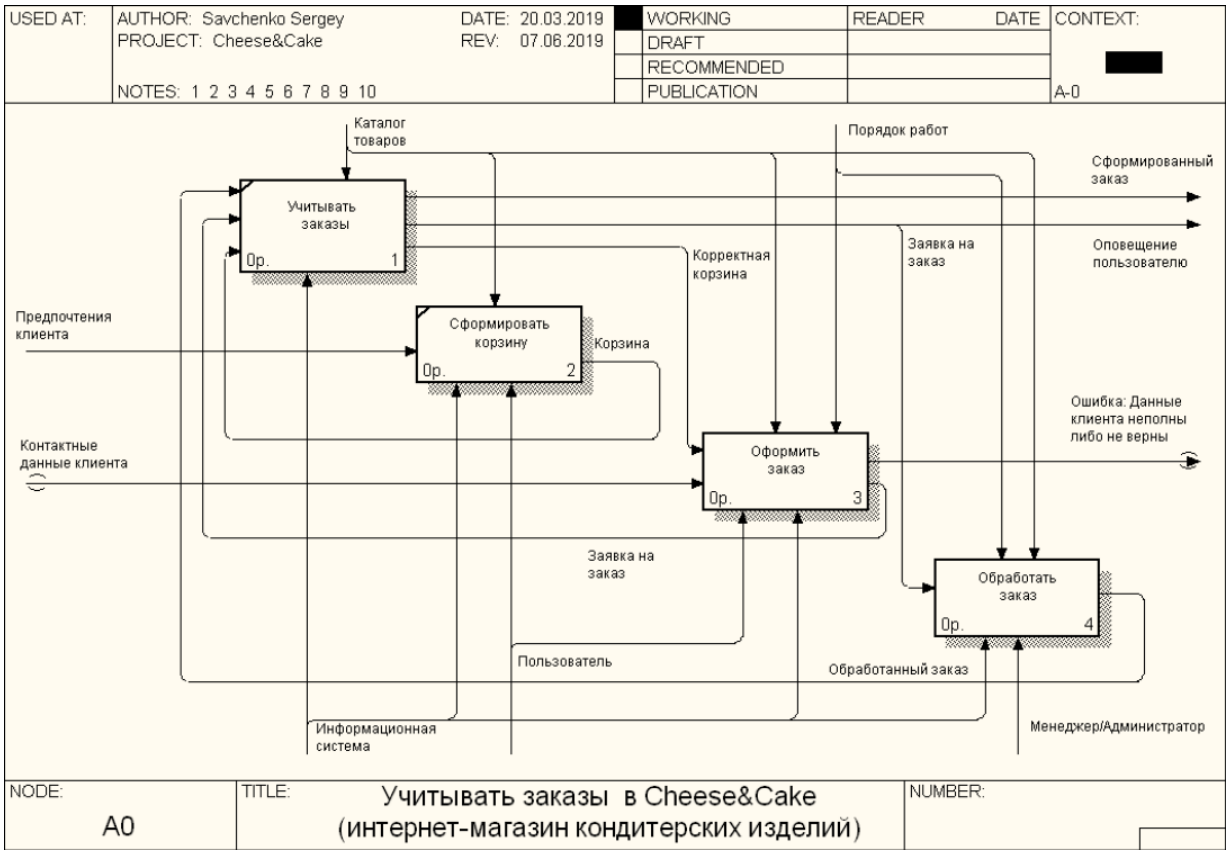

Рисунок 2.2 – Діаграма декомпозиції бізнес-процесу «Враховувати замовлення»

Після цього виконується декомпозиція блоку «Оформити замовлення» та «Обробити замовлення». Після декомпозиції ми отримаємо діаграми (рис 2.3), (рис 2.4) які відображають складову частину цих бізнес-процесів. До діаграми декомпозиції блоку «Оформити замовлення» входять наступні блоки:

a) Ввести контактні данні;

b) Перевірити коректність контакту;

c) Сформувати заявку на замовлення.

До діаграми декомпозиції блоку «Обробити замовлення» входять наступні блоки:

a) Зв’язатися з клієнтом;

b) Змінити адресу доставки;

c) Змінити дані з кількості позицій в замовленні;

d) Змінити статус заказу

В діаграмі наводиться послідовність дій ІС при оформлені замовлення. Після початку оформлення замовлення ІС повинна перевірити наявність товару, та попередити користувача якщо його немає у наявності. Після вводу даних користувачем потрібно ІС перевіряє їх і якщо вони виявляються некоректними, то користувачу видається повідомлення про наявність помилки. При вдало сформованому замовленні, ІС на виході з діаграми формується сутність «Нове замовлення».

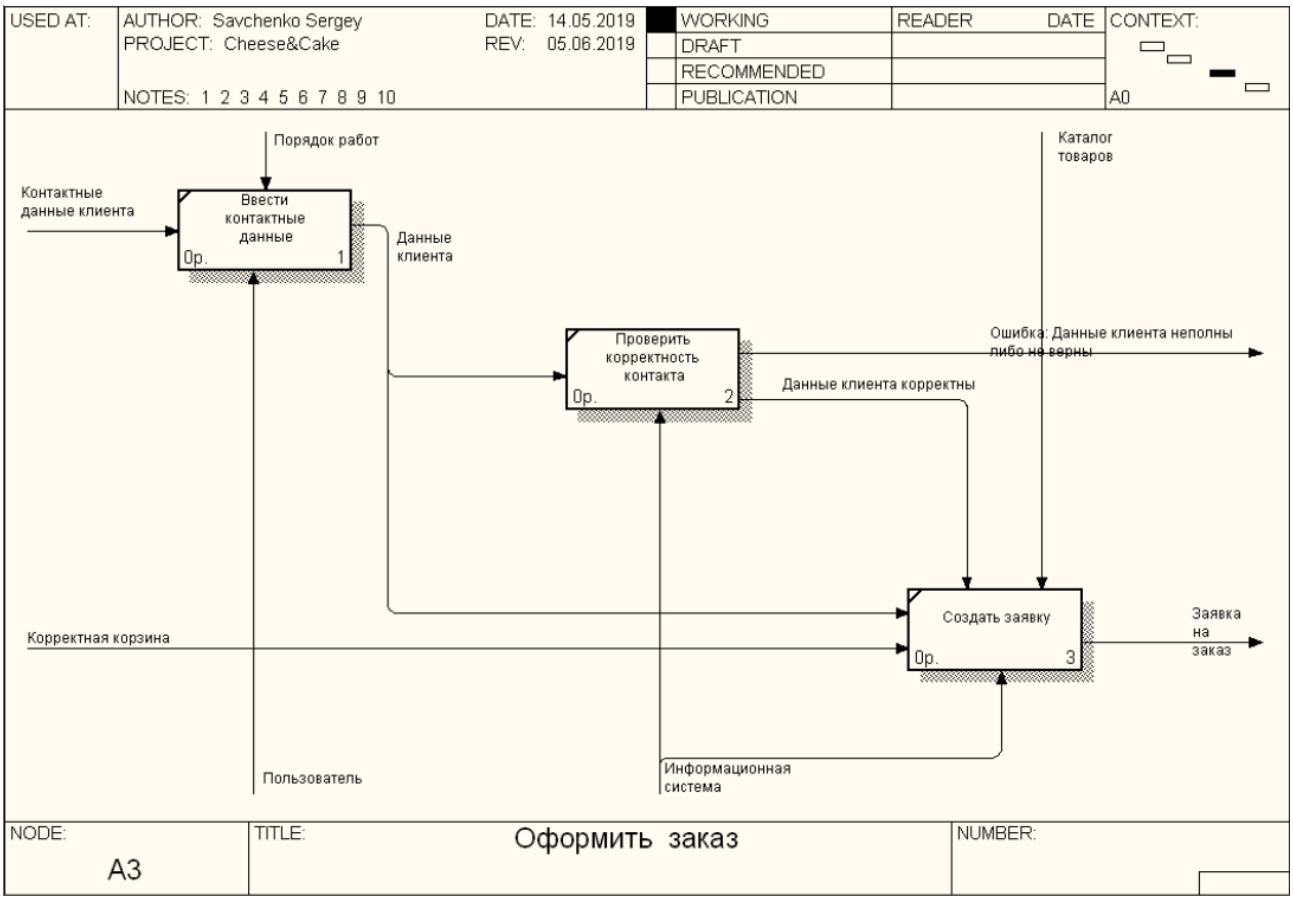

Рисунок 2.3 – Діаграма декомпозиції блоку «Оформлення замовлення»

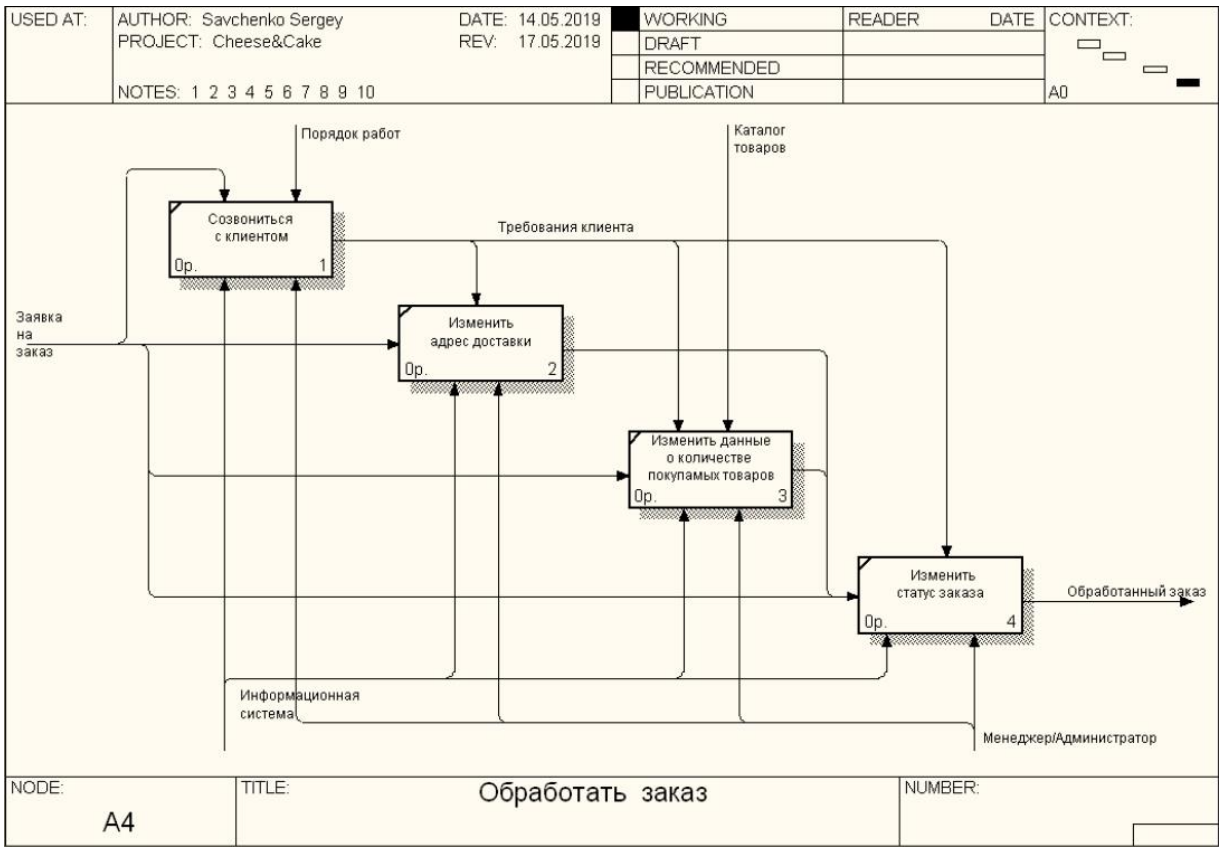

Рисунок 2.4 – Діаграма декомпозиції блоку «Обробка замовлення»

Після створення діаграм, визначення та уточнення функціональних вимог до ІС, аби визначити точну структуру моделі, було сформовано діаграму дерева вузлів (рис. 2.5).

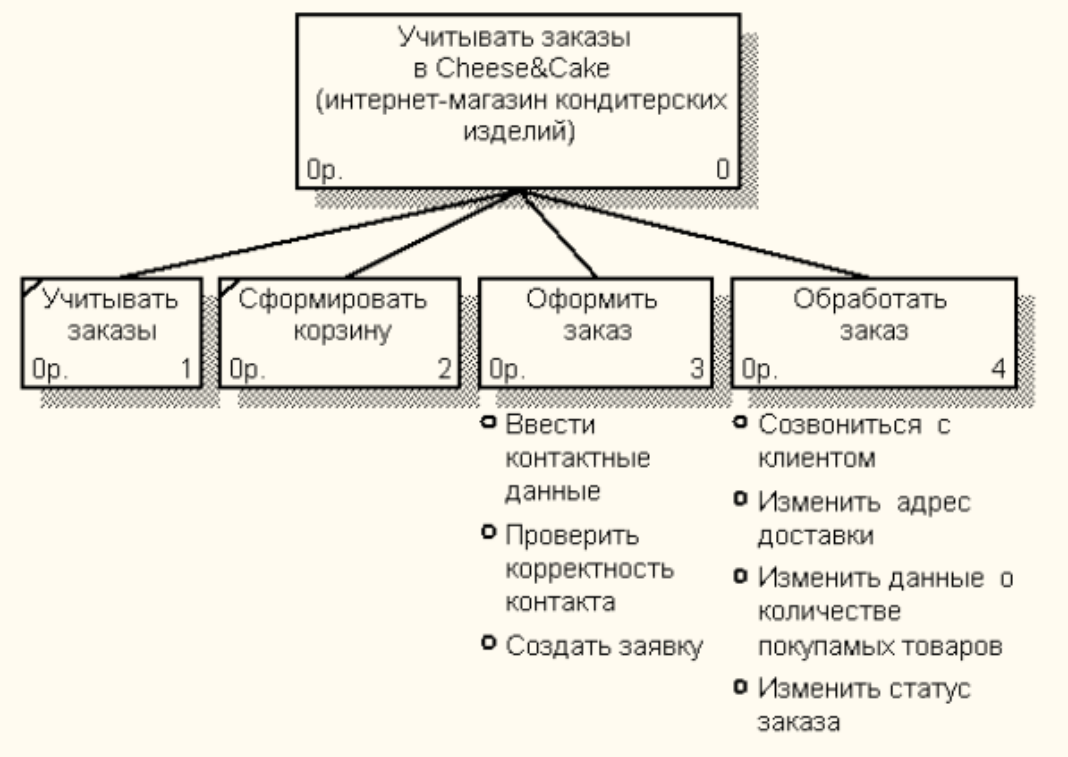

Рисунок 2.5 – Діаграма дерева вузлів

2.2 Логічне та фізичне моделювання бази даних інформаційної системи

При складанні моделі бази даних (рис. 2.6) було враховано, що в системі передбачена наявність декількох ролей (user, admin). А також, що схема бази даних буде складатися з 2-х схем, які відносяться до різних бізнес-процесів, пов’язаних між собою.

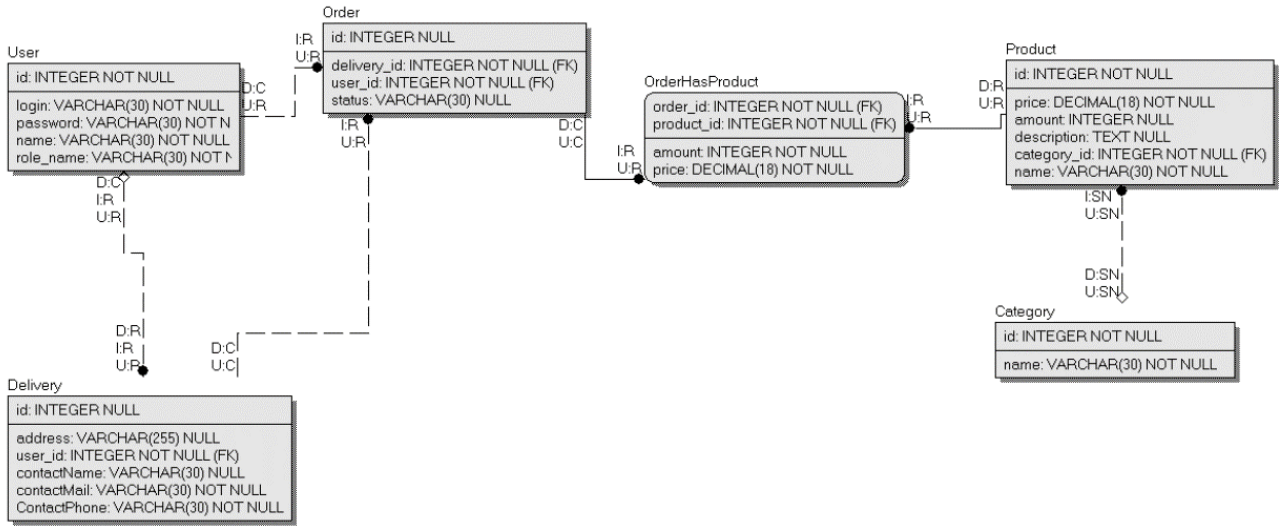

Рисунок 2.6 – Схема бази даних інформаційної системи

Схема створена за використанням стандарту IDEF1X. База даних була зроблена, за допомогою СУБД MySQL. При створенні фізичної моделі бази даних, було враховано, що СУБД MySQL має спільну мову з СУБД MySQL. Тому за відсутності в програмі ERWin можливості створення моделі для СУБД MySQL, модель була створена схема для СУБД MySQL, яку можна використати для створення бази в СУБД MySQL.

2.3 Діаграма прецедентів (Use Case Diagram) інформаційної системи

Враховуючи наявність в інформаційній системі 3 ролей, було створено діаграму прецедентів (рис 2.7).

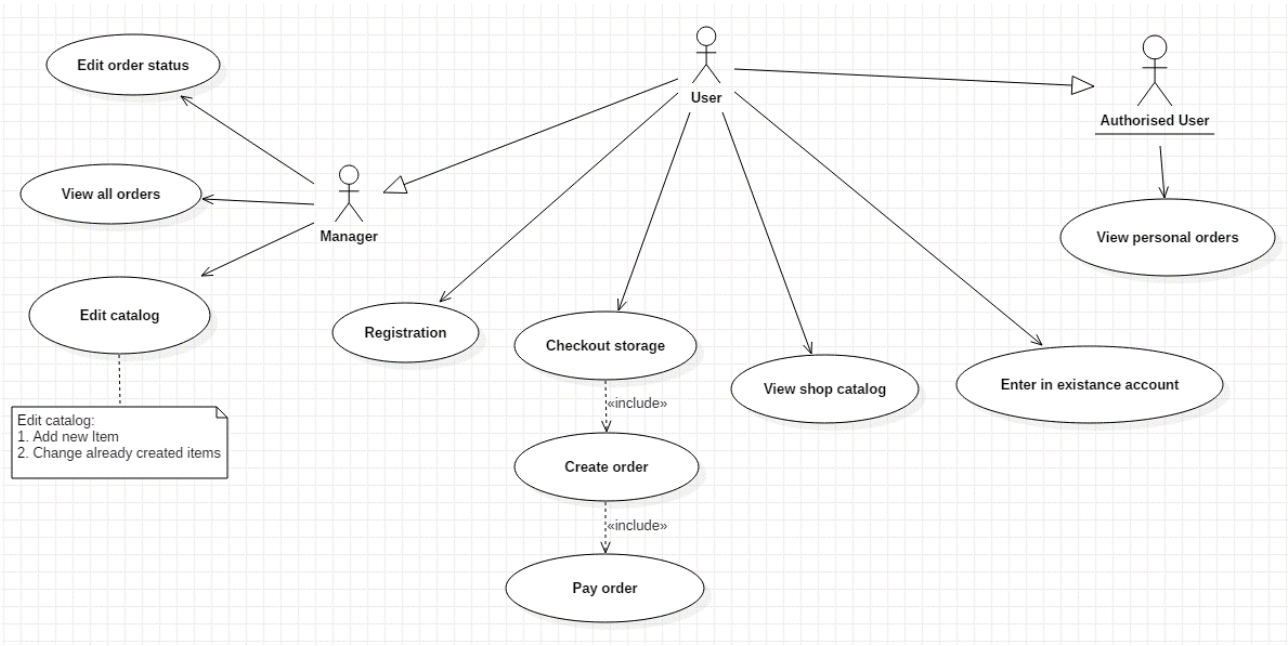

Рисунок 2.7 – Діаграма прецедентів інформаційної системи «Магазин кондитерських виробів»

В створеній діаграмі передбачена наявність 3 ролей з наступними функціями:

a) Неавторизований користувач:

a. Можливість зареєструватися;

b. Можливість увійти у обліковий запис;

c. Переглядати каталог товарів з можливістю пошуку та сортування;

d. Додавання товарів до кошика, але при спробі оформити замовлення, попросити його авторизуватися.

b) Авторизований користувач (user):

a. Можливість робити покупки та оформляти замовлення;

b. Переглядати усі замовлення зроблені раніше та переглядати їх статус;

c. Переглядати в профілі введену інформацію

c) Менеджер (admin);

a. Перегляд усіх замовлень зроблених користувачами;

a. Зміна статусу замовлень (на «В доставці», «Підтверджено», «Відхилено»);

b. Додання нового товару до каталогу;

c. Редагування вже існуючого товару.

2.4 Діаграма класів (Class Diagram) інформаційної системи.

Для інформаційної системи було створено діаграму класів (рис 2.8) в якій знаходяться основні класи програми, а також їх властивості та методи.

Діаграма класів — статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення.

2.5 Розробка вимог до серверної частини системи

Для створення серверної частини інформаційної системи використовується програмна платформа ASP.NET Web Forms 4.5. А збереження даних відбувається у СУБД MySQL. Основні функції серверної частини:

a) Зберігати інформацію про замовлення;
b) Зберігати інформацію про користувачів;
c) Обробляти замовлення;
d) Обробляти дані при авторизації користувача;
e) Приймати запроси від клієнтської частини;
f) Передавати інформацію до клієнтської частини.

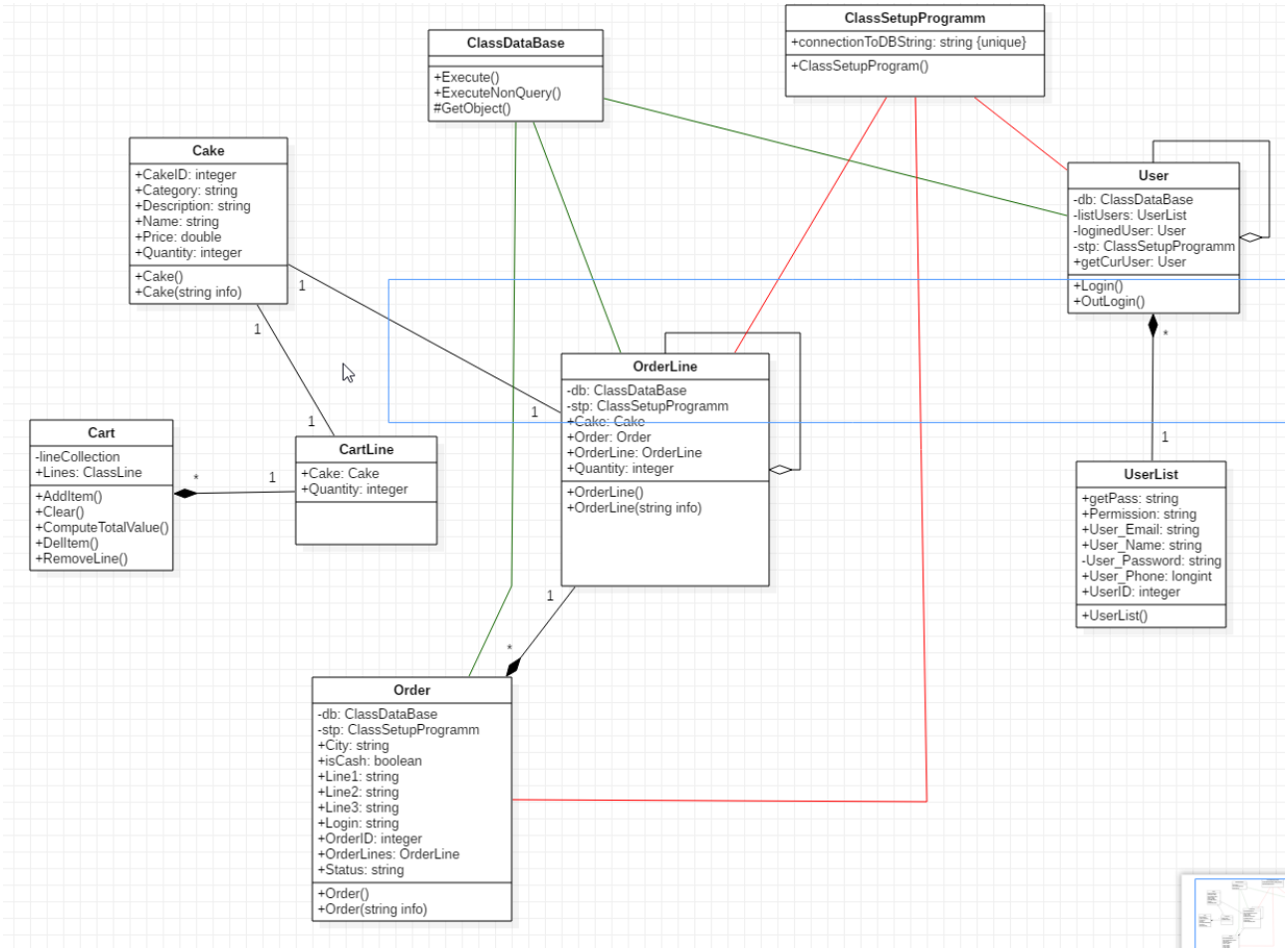

Рисунок 2.8 – Діаграма класів

2.6 Розробка вимог до функцій інтерфейсу клієнтської частини системи

Клієнтська частина інформаційної системи інтернет-магазину кондитерських виробів несе в собі функції як і для звичайних користувачів, так і для менеджерів.

Функції користувачів, реалізовані в клієнтській частині:

b) Неавторизований користувач:

a. Можливість зареєструватися;

b. Можливість увійти у обліковий запис;

c. Переглядати каталог товарів з можливістю пошуку та сортування;

d. Додавання товарів до кошика, але при спробі оформити замовлення, попросити його авторизуватися.

c) Авторизований користувач (user):

a. Можливість робити покупки та оформляти замовлення;

b. Переглядати усі замовлення зроблені раніше та переглядати їх статус;

c. Переглядати в профілі введену інформацію

d) Менеджер (admin);

a. Перегляд усіх замовлень зроблених користувачами;

b. Зміна статусу замовлень (на «В доставці», «Підтверджено», «Відхилено»);

c. Додання нового товару до каталогу;

d. Редагування вже існуючого товару.

Для опису процесу оформлення замовлення була створена діаграма діяльності (рис 2.9), яка описує дії користувача при вході на сайт.

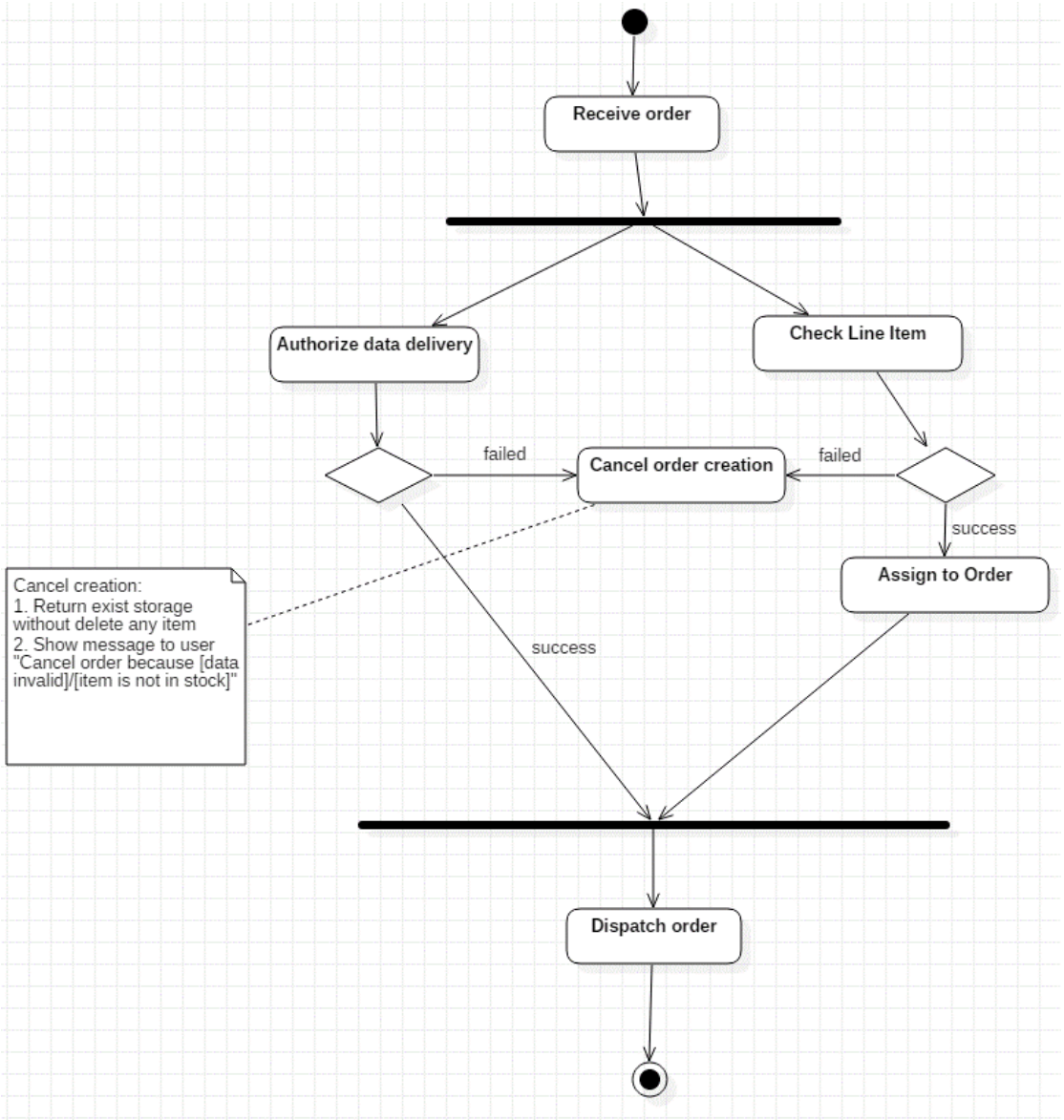

Рисунок 2.9 – Діаграма діяльності

Для кращого розуміння послідовності дій під час оформлення замовлення, було розроблено діаграму послідовності (рис 2.10).

Діаграма послідовності для прецеденту - це послідовність подій, необхідних для забезпечення необхідного поведінки. Потік подій описується в термінах того, «що» система повинна робити, а не «як» вона повинна це робити. Тобто він описується на мові предметної області, а не термінами реалізації.

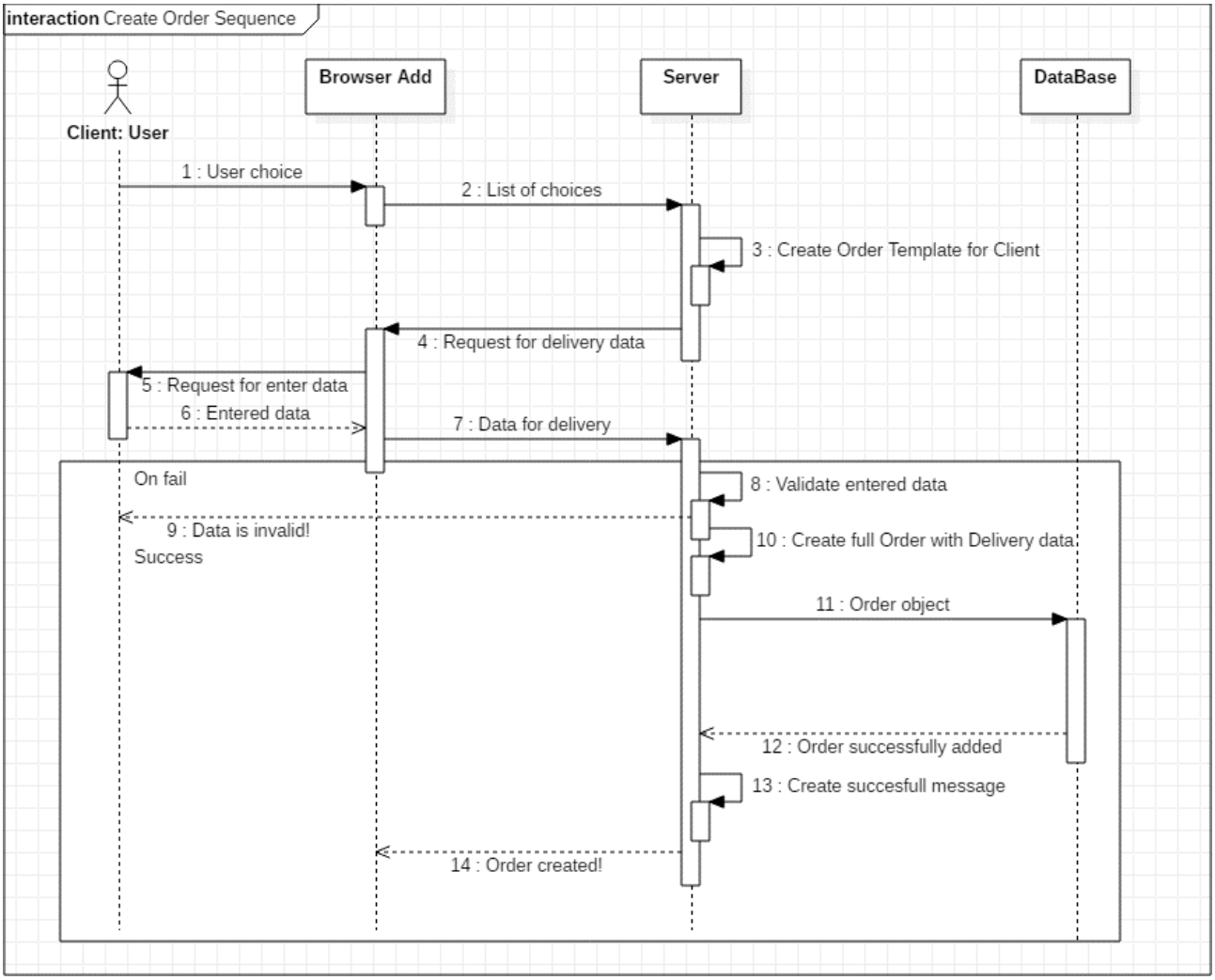

Рисунок 2.10 – Діаграма послідовності для події «Оформлення замовлення»

Діаграма станів (рис 2.11) — діаграма, що визначає зміну станів об'єкта у часі, одна з діаграм моделювання поведінки в UML[1]. Представляє об'єкт як автомат з теорії автоматів зі стандартизованими умовними позначеннями.

Елементами діаграми є:

a) Коло, що позначає початковий стан.

b) Коло з маленьким колом усередині, що позначає кінцевий стан (якщо є).

c) Округлений прямокутник, що позначає окремий стан. Верхівка прямокутника містить назву стану, в середині може бути горизонтальна лінія, під якою записуються активності, що відбуваються в даному стані.

d) Стрілка, що позначає перехід. Назва події (якщо є), що викликає перехід, відзначається над/під стрілкою. Вартовий вираз може бути доданий перед «/» і укладений у квадратні дужки (назва_події), він означає, що перехід відбувається лише за умови істинності виразу. Якщо при переході відбувається якась активність, то воно додається після «/» (назва події).

e) Товста горизонтальна лінія, яка є точкою об'єднання або розгалуження переходів.

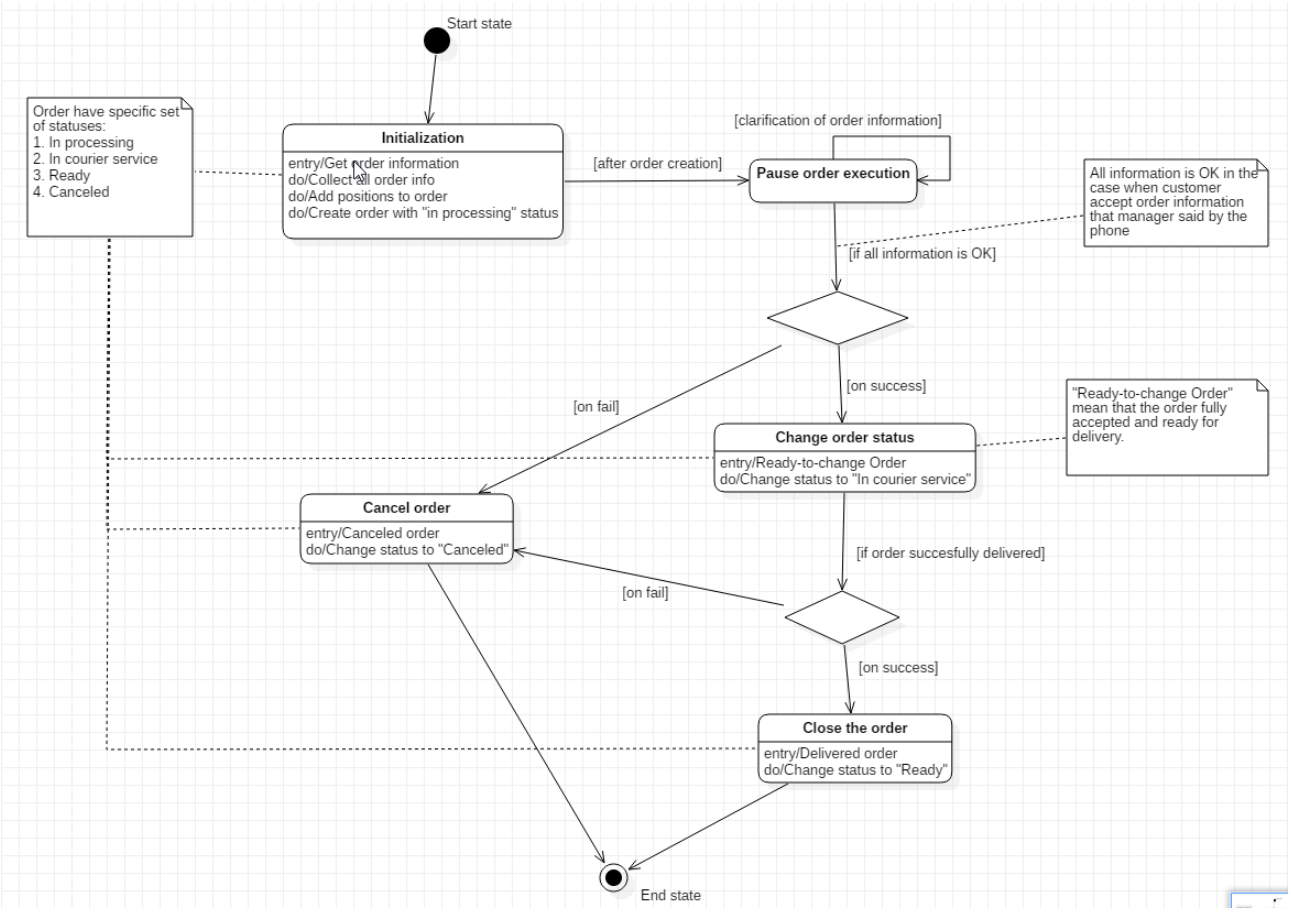

Рисунок 2.11 – Діаграма станів для об’єкту «Замовлення»

3 ОПИС ПРИЙНЯТТЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Обґрунтування вибору мови програмування

Для вирішення проблем веб-додатків може знадобитися чимало часу і зусиль. Веб-форм ASP.NET і платформи ASP.NET вирішують ці завдання одним із таких способів:

a) **Інтуїтивно зрозуміла і узгоджена об'єктна модель сторінок** - ASP.NET надає об'єктну модель, що дає можливість працювати з формами як єдине ціле, а не як окремі клієнтські і серверні частини. У цій моделі можна програмувати сторінки більш інтуїтивно зрозумілим способом, ніж в традиційних веб-додатків, включаючи можливість задати властивості для елементів сторінки і реагувати на події.

b) **Модель програмування на основі подій** - Веб-форм ASP.NET надають веб-додатків звичну модель написання обробників подій для подій, що відбуваються на стороні клієнта або сервера.

c) **Управління станом інтуїтивно зрозуміле** - сторона ASP.NET автоматично вирішує завдання підтримки стану сторінки і її елементів управління і надає явні способи збереження стану відомостей про програму. Це досягається без використання значних ресурсів сервера і може бути реалізована з або без відправки файлів cookie в браузері.

d) **Додатки, незалежні від браузера-сторінок** ASP.NET дозволяє вам створювати вся логіка додатки на сервері, що усуває необхідність явно створювати код для різних оглядачів.

e) **Підтримка .NET framework common language runtime** - сторона ASP.NET заснована на .NET Framework, тому всі можливості платформи доступні для будь-якої програми ASP.NET.

3.2 Обґрунтування вибору СУБД

MySQL - це реляційна СУБД. MySQL підтримує SQL (структурована мова запитів) і може застосовуватися в якості SQL-сервера. Це означає, що спілкуватися з сервером можна на мові SQL: клієнт посилає серверу запит, той його обробляє і віддає клієнту тільки ті дані, які були отримані в результаті цього

запиту. Тим самим клієнтові не потрібно викачувати дані і робити обчислення, як, наприклад, в Microsoft Access.

Крім того, MySQL - це ПЗ з відкритим кодом, тобто його можна вільно вивчати та змінювати. Пакет розповсюджується на умовах GPL (General Public License), його можна безкоштовно завантажити з Інтернету (http://www.mysql.com) для некомерційного використання.

MySQL має низку значних переваг:

a) **Швидкодія**. СУБД MySQL є однією з найшвидших баз даних з наявних на сучасному ринку.

b) **Простота використання.** СУБД MySQL є високопродуктивної і відносно простий у використанні СУБД, яку значно простіше інсталювати і адмініструвати, ніж багато великі системи.

c) **Підтримка мови запитів.** MySQL "розуміє" команди мови SQL, мови запитів, який знайшов застосування в усіх сучасних СУБД.

d) **Можливості обробки.** Кількість рядків в таблицях може досягати 50 млн. Компанія-розробник стверджує, що використовує MySQL з 1996 року на сервері з більш ніж 40 банками даних, які містять 10.000 таблиць, з яких більш ніж 500 мають більше 7 мільйонів рядків.

e) **Можливості доступу.** Сервер дозволяє одночасно підключатися необмеженій кількості користувачів, що одночасно працюють з базою даних. Доступ до сервера СУБД MySQL можна здійснити в інтерактивному режимі.

3.3 Створення бази даних для платформи «MySQL»

Створення бази даних для платформи MySQL було здійснено за допомогою програми MySQL Workbench. За допомогою цього додатку можна створити базу даних, за допомогою коду, згенерованого при побудуванні фізичної схеми бази даних (рис 2.6).

Додаток MySQL Workbench несе багатий функціонал для створення і модифікації бази даних. Також він має дуже приємній інтерфейс для створення таблиць (рис 3.1), що дозволяє створювати бази даних за лічені хвилини.

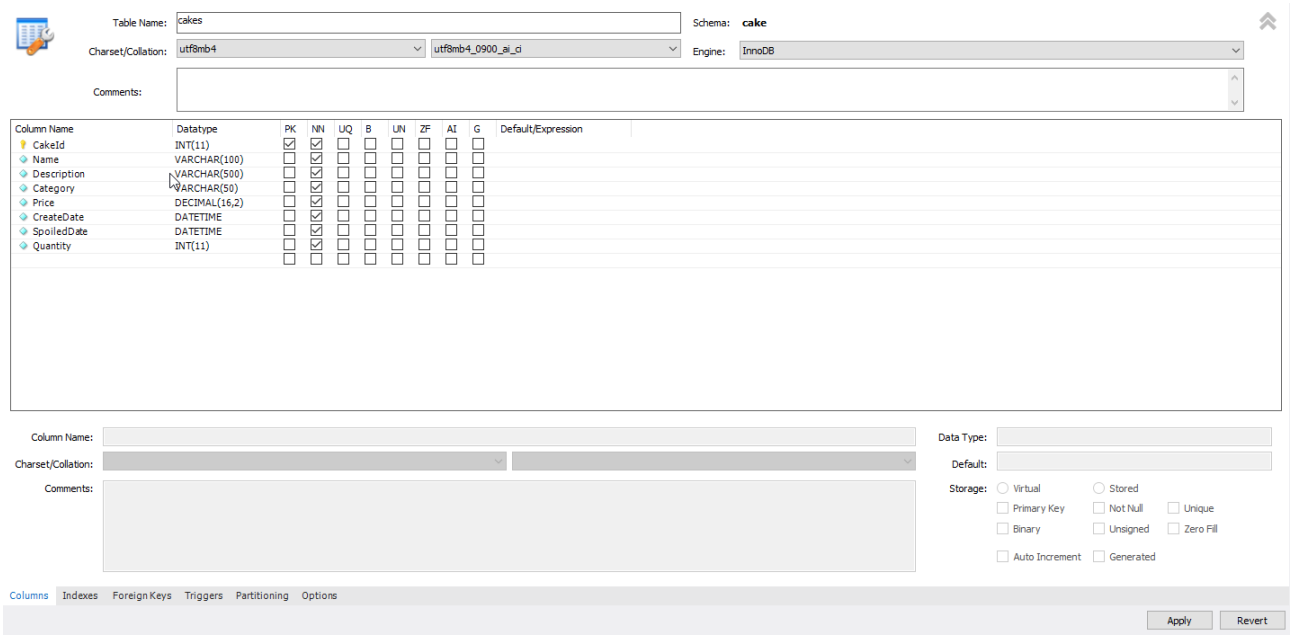

Рисунок 3.1 – Інтерфейс створення таблиць у MySQL Workbench

3.4 Розробка інтерфейсу клієнтської частини системи

Клієнтська частина для інтернет-магазину грає найважливішу роль, бо багато людей одразу ж оцінює магазин за зовнішнім виглядом та зрозумілістю його інтерфейсу.

Клієнтська частина поділяється на 2: сторінки доступні для усіх користувачів та сторінки доступні тільки для менеджерів.

Сторінки для всіх користувачів:

a) Кошик. Частина системи куди зберігаються товари, які користувач вибрав для подальшої покупки, а також є можливості вибрати кількість товару;

b) Сторінка товарів. Сторінка, де показуються усі вироби доступні в магазині. На цій сторінці користувач може побачити загальний асортимент товарів, та для більш зручного перегляду товарів реалізовано сортування по категоріям товарів;

c) Профіль. На сторінці профіля користувач має можливість переглянути усі замовлення, які він зробив раніше та побачити їх статус;

d) Сторінка входу. Користувач має можливість увійти у свій профіль, за допомогою email та паролю, який він задав при реєстрації;

e) Сторінка реєстрації. Неавторизований користувач може створити свій обліковий запис в магазині;

Сторінки для менеджерів:

a) Сторінка замовлень. менеджер мають змогу переглядати усі замовлення, що надійшли до системи, бачити інформацію про замовлення, а також змінювати їх статус.

b) Сторінка товарів. На цій сторінці для менеджерів відображаються усі товари у системі, які вони можуть змінити.

c) Сторінка редагування та додання нового товару. Менеджер може змінити інформацію про товар, а також додати новий товар у систему.

3.5 Тестування розробленого програмного забезпечення

Для тестування програмного додатку було обрано мануальне тестування.

Ручне тестування (manual testing) - частина процесу тестування на етапі контролю якості в процесі розробки програмного забезпечення. Воно проводиться тестувальником без використання програмних засобів, для перевірки програми або сайту шляхом моделювання дій користувача. У ролі тестувальників можуть виступати і звичайні користувачі, повідомляючи розробникам про знайдені помилки.

Були протестовані основні функції користувача, та менеджерів шляхом моделювання їх дій. При фінальному тестуванні не було знайдено помилок, які загрожували роботі системи.

3.6 Аналіз дослідницької експлуатації і варіантів використання інформаційної системи

Інформаційна система, що розробляється у рамках цього курсового проекту призначена для невеликих інтернет магазинів кондитерських виробів. Переваги інформаційної системи полягають у її компактності, зручному та зрозумілому інтерфейсу. У можливостях менеджерів є функції яких достатньо для правильної та гнучкої роботи системи.

ВИСНОВКИ

Під час курсового проектування було розроблено інформаційну систему інтернет-магазин кондитерських виробів. Вона являє собою клієнт-серверний додаток з базою даних. Інформаційна система володіє усіма функціями, достатніми для коректної та правильної роботи системи.

При виконанні курсового проектування було проведено функціональне моделювання та побудовано функціональну модель бізнес-процесу і модель декомпозиції інформаційної системи. Було розроблено фізичну модель бази даних для збереження необхідної інформації про користувачів, товарів та замовлень. За допомогою візуального моделювання UML діаграм були побудовані діаграми, які описують основні бізнес-процеси, події, стани та структуру інформаційної системи.

Розроблена інформаційна система призначена для невеликих магазинів кондитерських виробів, та може використовуватися у реальному бізнесі для реалізації продажу товарів.

Удосконалення створеної інформаційної системи може проводитися у наступних напрямках:

a) Покращення безпеки. Для більш безпечного використання інформаційної системи в бізнесі, бажано провести роботу по пошуку вразливостей у системі.

b) Додання нових функцій. Для більш комфортного користування системою можна додати нову функціональність, яка покращить взаємодію користувача з системою.

c) Багатоплатформність. У майбутньому є можливість розробки додатків для мобільних пристроїв.

ПЕРЕЛІК ПОСИЛАНЬ

1. Дейт К. Дж. Введение в системы баз данных, 6-е издание. – К.; М.; СПб.: Издательский дом "Вильямс", 2008. – 848 с.

2. Маклаков С.В. BPWin, ERWin. CASE – средства разработки информационных систем. – М.: Диалог-МИФИ, 2007.

3. Вигерс, К. Разработка требований к программному обеспечению: пер. англ. / К. Вигерс – М.: Издательско-торговый дом «Русская Редакция», 2004. – 576с.

4. Мацяшек, Л. Анализ требований и проектирование систем. Разработка информационных систем с использованием UML: пер. с англ. / Л. Мацяшек. – М.: Издательский дом «Вильямс», 2002. – 432 с.

5. Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя. / Г. Буч, Д. Рамбо. – М.: ДМК Пресс, 2001. – 432 с.

6. Розенберг, Д. Применение объектного моделирования с использованием UML и анализ прецедентов / Д. Розенберг, К. Скотт. – М.: ДМК-издательство, 2002. – 169 с.

7. Калянов Г.Н. CASE - технологии: Консалтинг в автоматизации бизнес-процессов. - 3-е изддание. - М.: Горячая линия-Телеком, 2008. - 320 с

8. Коннолли, Т., Бегг, К., Страчан, А. Базы данных. Проектирование, реализация и сопровождение: Теория и практика : Пер. с англ. / Т. Коннолли, К. Бегг, А. Страчан. – М.: Издательский дом «Вильямс», 2003. – 1436 с.

9. Маклаков, С.В. BPwin и Erwin : CASE-средства разработки информационных систем / С.В. Маклаков. – М.: Диалог-МИФИ, 1999. – 256 с.

10. Microsoft ASP.NET 3.5 с примерами на C# 2008 для профессионалов (+ CD-ROM)Мэтью Мак-Дональд, Марио Шпушта Издательство: Вильямс 2008 г. ISBN 978-5-8459-1403-3, 978-1-59-059893-1, 1424 стр.

Додаток А
Керівництво користувача

Міністерство освіти и науки України

Затверджую

Керівник
курсового проекту,
доц. кафедри системотехніки
_____ Ю.В. Міщеряков
(Підпис, дата)

ІНФОРМАЦІЙНА СИСТЕМА «ІНТЕРНЕТ-МАГАЗИН КОНДИТЕРСЬКИХ ВИРОБІВ»

Керівництво користувача

Аркуш затвердження
ГЮИК.508100.020 – 34 01 ІЗ – ЛУ

Студент групи _____
(Назва групи)

_____
(Підпис, дата, прізвище, ім'я, по батькові)

2019

Міністерство освіти и науки України

Затверджено
ГЮИК.508100.111 ІЗ – ЛУ

ІНФОРМАЦІЙНА СИСТЕМА «ІНТЕРНЕТ-МАГАЗИН КОНДИТЕРСЬКИХ ВИРОБІВ»

Керівництво користувача

ГЮИК.508100.020 - 34 01 ІЗ

аркушів _____

2019

А.1 ВСТУП

Інформаційна система, що розробляється у рамках цього курсового проекту призначена для невеликих інтернет магазинів кондитерських виробів. У можливостях менеджерів є функції яких достатньо для правильної та гнучкої роботи системи. Користувач, у свою чергу, може здійснювати покупки товарів.

Для користування системою у ролі користувача, потрібно лише мінімальне вміння користування мережею інтернет. Для користування системою у ролі менеджера, потрібне мінімальне знання мережі інтернет та самої інформаційної системи в цілому. Також потрібна наявність знань у сфері баз даних, для можливої відладки бази в екстрених ситуаціях.

Для користування системою користувачу не потрібно ніякої документації.

А.2 ПРИЗНАЧЕННЯ ТА УМОВИ ЗАСТОСУВАННЯ

Інформаційна система інтернет магазин кондитерських виробів призначена для роздрібного продажу кондитерських виробів. Адмін-панель системи призначена для обліку товару та перевірки потоку реалізації замовлень в системі.

Інформаційна система розроблювалась для всесвітньої мережі інтернет, що означає, що звичайному користувачу для її користування потрібен лише браузер починаючи від Microsoft Edge та новіше. Для встановлення системи потрібний будь-який сервер, який буде підтримувати технологію ASP.NET Web Forms 4.5.

А.3 ПІДГОТОВКА ДО РОБОТИ

Для роботи з системою звичайному користувачу потрібен лише звичайний браузер починаючи від Microsoft Edge та новіше. Для входу в систему користувачу потрібно ввести в адресний рядок браузера – адрес інформаційної системи у мережі інтернет. Та для можливості використання усіх користувацьких можливостей, користувачу рекомендується зареєструватися на сайті, якщо він не був зареєстрований до цього. Або увійти у власний обліковий запис, якщо він вже раніше зареєструвався.

А.4 ОПИС ОПЕРАЦІЙ

При вході до сайту можна бачити список кондитерських виробів, які зараз доступні для продажу. Після цього можна виконувати наступні дії:

a) Відфільтрувати вироби за категоріями доступними у боковому меню (рис А.4.1). На цій сторінці відображаються усі товари кожної з категорій.

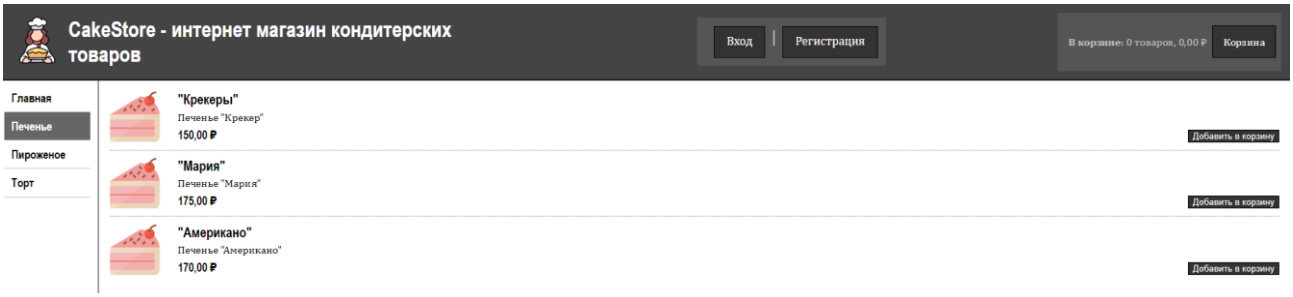

Рисунок А.4.1 – Відфільтрована сторінка за категорією «Печиво»

b) Перейти до сторінки профіля (рис А.4.2). На цій сторінці можна переглянути усі замовлення які були зроблені користувачем.

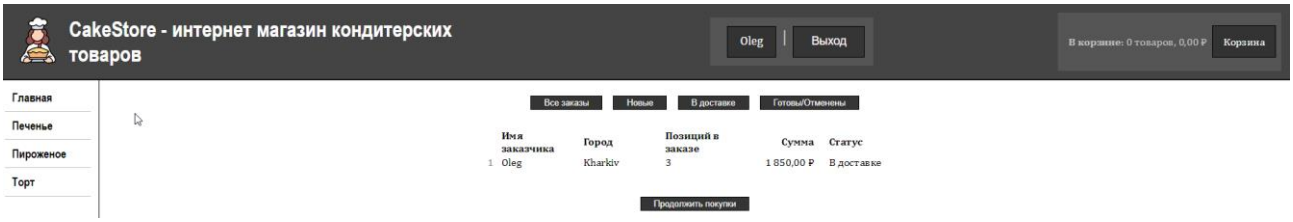

Рисунок А.4.2 – Сторінка профіля «Oleg»

c) Перейти до сторінки кошика (рис А.4.3). На цій сторінці можна побачити усі товари, які були «зарезервовані» для купівлі та змінити їх кількість. Основна функція цієї сторінки – оформлення замовлення. При оформленні замовлення, потрібно ввести вказані дані.

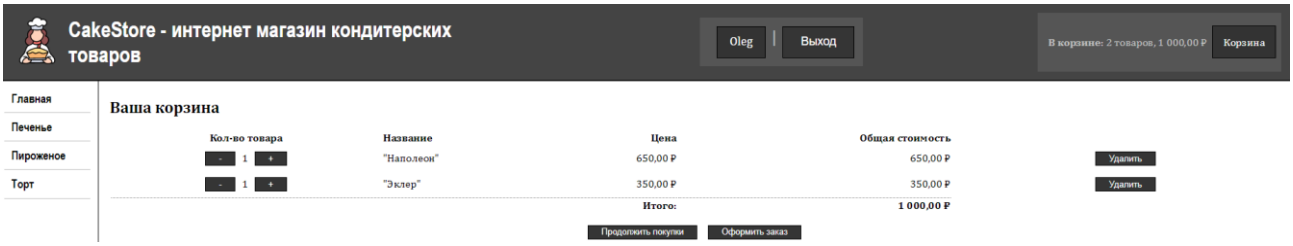

Рисунок А.4.3 – Сторінка кошика користувача

d) Вийти з системи. При цій дії користувач виходить з свого облікового запису.

А.5 АВАРІЙНІ СИТУАЦІЇ

У випадку виявлення помилок в даних, потрібно подивитися у консоль, чи є там вивід явних помилок. При повній зупинці серверу, потрібно перезавантажити програму серверної частини повним перезавантаженням системи.

А.6 ПІДГОТОВКА ДО РОБОТИ

Для встановлення інформаційної системи на сервер, на сервері повинна бути встановлена програмна платформа з підтримкою технології ASP.NET Web Forms. Після встановлення всіх бібліотек, можна запускати програму сервера.

Додаток Б
Текст програми

Міністерство освіти и науки України

Затверджую

Керівник
курсового проекту,
доц. кафедри системотехніки
_____ Ю.В. Міщеряков
(Підпис, дата)

ІНФОРМАЦІЙНА СИСТЕМА «ІНТЕРНЕТ-МАГАЗИН КОНДИТЕРСЬКИХ ВИРОБІВ»

Текст програми

Аркуш затвердження
ГЮИК.508100.020 – 01 12 01 – ЛУ

Студент групи _____
(Назва групи)

_____
(Підпис, дата, прізвище, ім'я, по батькові)

2019

Міністерство освіти и науки України

Затверджено
ГЮИК.508100.111 ІЗ – ЛУ

ІНФОРМАЦІЙНА СИСТЕМА «ІНТЕРНЕТ-МАГАЗИН КОНДИТЕРСЬКИХ ВИРОБІВ»

Текст програми

ГЮИК.508100.020 - 01 12 01

аркушів _____

2019

Б.1 Скрипт створення бази даних

У ході моделювання фізичної моделі бази даних інтернет магазину «Магазин кондитерських товарів» було автоматично сгенеровано необхідний скрипт створення бази даних для необхідної СУБД MySQL:

```
CREATE TABLE Category
(
	id		INTEGER NOT NULL,
	name		VARCHAR(30) NOT NULL
);

ALTER TABLE Category
	ADD  PRIMARY KEY (id);

CREATE TABLE Delivery
(
	id		INTEGER NULL,
	address		VARCHAR(255) NULL,
	user_id		INTEGER NOT NULL,
	contactName		VARCHAR(30) NULL,
	ContactPhone		VARCHAR(30) NOT NULL,
	contactMail		VARCHAR(30) NOT NULL
);

ALTER TABLE Delivery
	ADD  PRIMARY KEY (id);

CREATE TABLE Order
(
	id		INTEGER NULL,
	delivery_id		INTEGER NOT NULL,
	user_id		INTEGER NOT NULL,
	status		VARCHAR(30) NULL
);
```

```
ALTER TABLE Order
        ADD  PRIMARY KEY (id);

CREATE TABLE OrderHasProduct
(
        order_id            INTEGER NOT NULL,
        product_id          INTEGER NOT NULL,
        amount              INTEGER NOT NULL
                CONSTRAINT  amount CHECK (amount >= 0),
        price               DECIMAL(18) NOT NULL
                CONSTRAINT  price CHECK (price >= 0)
);

ALTER TABLE OrderHasProduct
        ADD  PRIMARY KEY (order_id,product_id);

CREATE TABLE Product
(
        id                  INTEGER NOT NULL,
        price               DECIMAL(18) NOT NULL
                CONSTRAINT  price CHECK (price >= 0),
        amount              INTEGER NULL
                CONSTRAINT  amount CHECK (amount >= 0),
        description         TEXT NULL,
        category_id         INTEGER NOT NULL,
        name                VARCHAR(30) NOT NULL
);

ALTER TABLE Product
        ADD  PRIMARY KEY (id);

CREATE TABLE User
(
```

```
    id              INTEGER NOT NULL,
    login           VARCHAR(30) NOT NULL
        CONSTRAINT  login CHECK (%Attr unique),
    password        VARCHAR(30) NOT NULL,
    name            VARCHAR(30) NOT NULL,
    role_name       VARCHAR(30) NOT NULL
);

ALTER TABLE User
    ADD  PRIMARY KEY (id);

ALTER TABLE Delivery
    ADD FOREIGN KEY R_9 (user_id) REFERENCES User(id)
        ON DELETE CASCADE;

ALTER TABLE Order
    ADD FOREIGN KEY R_2 (user_id) REFERENCES User(id)
        ON DELETE CASCADE;

ALTER TABLE Order
    ADD FOREIGN KEY R_8 (delivery_id) REFERENCES Delivery(id)
        ON DELETE CASCADE;

ALTER TABLE OrderHasProduct
    ADD FOREIGN KEY R_4 (order_id) REFERENCES Order(id)
        ON DELETE CASCADE;

ALTER TABLE OrderHasProduct
    ADD FOREIGN KEY R_13 (product_id) REFERENCES Product(id);

ALTER TABLE Product
    ADD FOREIGN KEY R_6 (category_id) REFERENCES Category(id);
```

Б.2 Текст програми з середовища розробки

В процесі програмування було створено 9 класів (рис 2.8), 2 класи обробки заявок на додавання в базу даних та реалізовано 7 веб-форм користувача.

Б.2.1 Код основного класу Order.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace ShopASP.Models
{

    public class Order
    {
        ClassDataBase db = new ClassDataBase();
        ClassSetupProgram stp = new ClassSetupProgram();

        public int OrderId { get; set; }
        public string Name { get; set; }
        public string Line1 { get; set; }
        public string Line2 { get; set; }
        public string Line3 { get; set; }
        public string City { get; set; }
        public bool isCash { get; set; }
        public string Status { get; set; }
        public virtual List<OrderLine> OrderLines { get; set; }
        public Order()
        {
            OrderId = 0;
            Name = "";
            Line1 = "";
            Line2 = "";
            Line3 = "";
            City = "";
            isCash = false;
            Status = "";
            OrderLines = new List<OrderLine>();
        }
        public Order(string info)
        {
            if (info != null && info != "")
            {
                string[] val = info.Split('!');
                OrderId = Convert.ToInt32(val[0]);
                Name = val[1];
                Line1 = val[2];
                Line2 = val[3];
                Line3 = val[4];
                City = val[5];
                if (Convert.ToInt32(val[6]) == 1)
                    isCash = true;
                else
                    isCash = false;
```

```
                Status = val[7];
                List<OrderLine> tempOrderLines = new List<OrderLine>();

                db.Execute<OrderLine>(ref stp, "SELECT * FROM cake.orderlines where
Order_OrderId = " + val[0] + ";", ref tempOrderLines);

                OrderLines = tempOrderLines;
            }
        }



        public class OrderLine
        {

            ClassDataBase db = new ClassDataBase();
            ClassSetupProgram stp = new ClassSetupProgram();

            public int OrderLineId { get; set; }
            public int Quantity { get; set; }
            public Cake Cake { get; set; }
            public int Order { get; set; }

            public OrderLine(int quantity, Cake cake, int orderID)
            {
                OrderLineId = 0;
                Quantity = quantity;
                Cake = cake;
                Order = orderID;
            }

            public OrderLine(string info)
            {
                if (info != null && info != "")
                {
                    string[] val = info.Split('!');
                    OrderLineId = Convert.ToInt32(val[0]);
                    Quantity = Convert.ToInt32(val[1]);

                    List<Cake> tempCakes = new List<Cake>();
                    tempCakes.Clear();

                    db.Execute<Cake>(ref stp, "SELECT * FROM cake.cakes where CakeId = " +
val[2] + ";", ref tempCakes);


                    if (tempCakes.Count > 0)
                    {
                        Cake = tempCakes[0];
                    }
                    else
                    {
                        Cake = new Cake();
                    }

                    Order = Convert.ToInt32(val[3]);
                }
            }

        }
    }
}
```

Б.2.2 Код класу-обробника заявок на додавання в базу даних EfDBContext.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;


namespace ShopASP.Models.Repository
{
    public class EFDbContext
    {
        public ClassDataBase db = new ClassDataBase();
        public ClassSetupProgram stp = new ClassSetupProgram();
        public List<Cake> Cakes = new List<Cake>();
        public List<Order> Orders = new List<Order>();
        public List<User.UserList> UserList = new List<User.UserList>();
        public List<Flag> FilterFlag = new List<Flag>();

        public string getFlag()
        {
            FilterFlag.Clear();
            db.Execute<Flag>(ref stp, "SELECT * FROM cake.testblob;", ref FilterFlag);
            return FilterFlag[0].FlagStatus;
        }

        public void SetFlag(string value)
        {
            db.ExecuteNonQuery(ref stp, "UPDATE `cake`.`testblob` SET `Content` = '"+ value
+"' WHERE (`ID_Image` = '1');");
        }

        public List<Cake> getCakes() {

            Cakes.Clear();

            db.Execute<Cake>(ref stp, "SELECT * FROM `cake`.`cakes` where Quantity > 0;",
ref Cakes);

            return Cakes;
        }

        public List<User.UserList> getUsers() {

            UserList.Clear();

            db.Execute<User.UserList>(ref stp, "SELECT * FROM cake.users;", ref UserList);

            return UserList;
        }

        public void insertUser(string User_Name, string User_Password, string User_Phone,
string User_EMail)
        {
            UserList.Clear();

            db.Execute<User.UserList>(ref stp, "SELECT * FROM cake.users;", ref UserList);

            int userCount = (UserList.Count() > 0) ? UserList[UserList.Count() - 1].UserID +
1 : 1;
```

```
            User.UserList temp = UserList.Where(u => u.User_EMail ==
User_EMail).FirstOrDefault();


            if (temp == null)
            {
                db.ExecuteNonQuery(ref stp, "INSERT INTO `cake`.`users` (`User_ID`,
`User_Name`, `User_Password`, `Permission`, `User_Phone`, `User_EMail`) VALUES ('" +
userCount.ToString() + "', '" + User_Name + "', '" + User_Password + "', 'User', '" +
User_Phone + "', '" + User_EMail + "');");
            }
        }


        public List<Order> getOrders()
        {
            Orders.Clear();

            db.Execute<Order>(ref stp, "SELECT * FROM Orders;", ref Orders);

            return Orders;
        }


        public void insertOrder(Order order)
        {
            Orders = getOrders();

            if (Orders.Exists(x => x.OrderId == order.OrderId))
            {

                int temp = (order.isCash == true) ? 1 : 0, counter = 0;
                Order tempOrder = Orders.Find(x => x.OrderId == order.OrderId);
                List<Order.OrderLine> tempLines = tempOrder.OrderLines;

                foreach (Order.OrderLine lines in order.OrderLines) {
                    if (lines != tempLines[counter])
                    {
                        db.ExecuteNonQuery(ref stp, "UPDATE `cake`.`orderlines` SET
`Quantity` = '"+ lines.Quantity.ToString() +"', `Cake_CakeId` = '"+
lines.Cake.CakeId.ToString() +"', `Order_OrderId` = '" + lines.Order.ToString() + "' WHERE
(`OrderLineId` = '" + lines.OrderLineId.ToString() +"');");

                        counter++;
                    }
                    else counter++;
                }


                db.ExecuteNonQuery(ref stp, "UPDATE `cake`.`Orders` SET `Name` = '"+
order.Name.ToString() +"', `Line1` = '"+ order.Line1.ToString() + "', `Line2` = '"+
order.Line2.ToString() + "', `Line3` = '"+ order.Line3.ToString() + "', `City` = '"+
order.City.ToString() +"', `isCash` = '"+ temp.ToString() + "', `Status` = '"+
order.Status.ToString() +"' WHERE (`OrderID` = '"+ order.OrderId.ToString() + "')");
            }
            else
            {
                List<Order.OrderLine> orderLines = new List<Order.OrderLine>();


                db.Execute<Order.OrderLine>(ref stp, "SELECT * FROM cake.orderlines;", ref
orderLines);
```

```
                int linesCount = (orderLines.Count() > 0) ? orderLines[orderLines.Count() -
1].OrderLineId + 1 : 1;
                int orderCounter = (Orders.Count() > 0) ? Orders[Orders.Count() - 1].OrderId
+ 1 : 1;
                int temp = (order.isCash == true) ? 1 : 0;

                db.ExecuteNonQuery(ref stp, "INSERT INTO `cake`.`orders` (`OrderID`, `Name`,
`Line1`, `Line2`, `Line3`, `City`, `isCash`, `Status`) VALUES ('" + orderCounter.ToString()
+ "', '" + order.Name.ToString() + "', '" + order.Line1.ToString() + "', '" +
order.Line2.ToString() + "', '" + order.Line3.ToString() + "', '" + order.City.ToString() +
"', '" + temp.ToString() + "', 'В обработке');");

                foreach (Order.OrderLine line in order.OrderLines)
                {
                    db.ExecuteNonQuery(ref stp, "INSERT INTO `cake`.`orderlines`
(`OrderLineId`, `Quantity`, `Cake_CakeId`, `Order_OrderId`) VALUES ('" +
linesCount.ToString() + "', '" + line.Quantity.ToString() + "', '" +
line.Cake.CakeId.ToString() + "', '" + orderCounter.ToString() + "');");
                    db.ExecuteNonQuery(ref stp, "UPDATE `cake`.`cakes` SET `Quantity` = '"+
(line.Cake.Quantity - line.Quantity).ToString() +"' WHERE (`CakeId` = '"+
line.Cake.CakeId.ToString() +"');");
                    linesCount++;
                }
            }
        }
    }
}
```

Б.2.3 Код веб-форми користувача з каталогом товарів Listing.aspx.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web.Routing;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using ShopASP.Pages.Helpers;
using ShopASP.Models.Repository;

namespace ShopASP.Pages
{
    public partial class Listing : System.Web.UI.Page
    {

        private Repository repository = new Repository();
        private int pageSize = 5;

        protected int CurrentPage
        {
            get
            {
                int page;
                page = GetPageFromRequest();
                return page > MaxPage ? MaxPage : page;
            }
        }

        protected int MaxPage
        {
            get
```

```
            {
                int prodCount = FilterCakes().Count();
                return (int)Math.Ceiling((decimal)prodCount / pageSize);
            }
        }



        private int GetPageFromRequest()
        {
            int page;
            string reqValue = (string)RouteData.Values["page"] ??
                Request.QueryString["page"];
            return reqValue != null && int.TryParse(reqValue, out page) ? page : 1;
        }

        public IEnumerable<Models.Cake> GetCakes()
        {
            return FilterCakes()
                .OrderBy(c => c.CakeId)
                .Skip((CurrentPage - 1) * pageSize)
                .Take(pageSize);
        }


        private IEnumerable<Models.Cake> FilterCakes()
        {
            string currentCategory = (string)RouteData.Values["category"] ??
                Request.QueryString["category"];

            return currentCategory == null ? repository.Cakes :
                repository.Cakes
                    .Where(p => p.Category == currentCategory);
        }

        protected void Page_Load(object sender, EventArgs e)
        {
            if (IsPostBack)
            {
                int selectedCakeId;
                if (int.TryParse(Request.Form["add"], out selectedCakeId))
                {
                    Models.Cake selectedCake = repository.Cakes
                        .Where(g => g.CakeId == selectedCakeId).FirstOrDefault();

                    if (selectedCake != null)
                    {
                        SessionHelper.GetCart(Session).AddItem(selectedCake, 1);
                        SessionHelper.Set(Session, SessionKey.RETURN_URL,
                            Request.RawUrl);
                    }
                }
            }
        }
    }
}
```