МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Бакалаврська кваліфікаційна робота

на тему: Розробка WEB-додатку бізнес-аналізу підприємства

Виконав студент 4 курсу групи К-42
Напрямок 6.050101 комп'ютерні науки,
Воробйов Руслан Олександрович

Керівник к.т.н., доцент
Онищенко Сергій Михайлович

Консультант

Рецензент к.геогр.н., доцент
Лужбін Анатолій Михайлович

Одеса 2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИУ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук, управління та адміністрування
Кафедра інформаційних технологій
Рівень вищої освіти бакалавр
напрямок 6.050101 комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
_____
"____" _____ 2019 р.

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Воробйову Руслану Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Розробка WEB-додатку бізнес-аналізу підприємства»

керівник роботи к.т.н., доцент Онищенко Сергій Михайлович
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "____"_____ 2019 р. №____
2. Строк подання студентом проекту _____
3. Вихідні дані до роботи методології моделювання IDEF0, IDEF3, DFD, UML, веб-додаток phpMyAdmin

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Способи моделювання інформаційних систем
2. Моделювання інформаційної системи підприємства
2.1 Аналіз предметної області
2.2 Побудова моделей IDEF0, IDEF3, DFD
2.3 Побудова діаграми варіантів використання та діаграми класів
3 Розробка ВЕБ-додатку підприємства

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
1.ER структура бази даних
2.Діаграми IDEF0, IDEF3, DFD
3.Діаграма варіантів використання
4.Діаграма класів

6. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання "____"_____ 201 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту	Термін виконання етапів проекту	Оцінка виконання етапу	
			у %	за 4-х бальною шкалою
1	Дослідження літературних джерел за темою кваліфікаційної роботи			
2	Аналіз підходів до моделювання інформаційних систем			
3	Формулювання завдання			
4	Аналіз предметної області та побудова концептуальної моделі			
5	Побудова моделей IDEF0, IDEF3, DFD			
6	Рубіжна атестація			
7	Побудова діаграми класів			
8	Розробка ВЕБ-додатку підприємства			
9	Тестування та перевірка роботи додатку			
	Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)			

Студент _____ Воробйов Р.О.
(підпис) (прізвище та ініціали)

Керівник проекту _____ Онищенко С.М.
(підпис) (прізвище та ініціали)

ЗМІСТ

СКОРОЧЕННЯ І УМОВНІ ПОЗНАКИ

АТ – акціонерне товариство.

БД – база даних.

ДП – дипломний проект.

ІС – інформаційна система.

ПЗ – програмне забезпечення.

СКБД – система керування базами даних.

Apache HTTP-сервер (названий ім'ям групи племен північноамериканських індіанців апачів, крім того, є перекрученим скороченням від англ. A patchy server; серед українських користувачів загальноприйнято перекладне апач) – вільний веб-сервер.

bootstrap – вільний набір інструментів для створення веб-додатків.

CASE (англ. Computer-aided software engineering) – набір інструментів і методів програмної інженерії для проектування програмного забезпечення, який допомагає забезпечити високу якість програм, відсутність помилок і простоту в обслуговуванні програмних продуктів.

CSS (англ. Cascading Style Sheet) – каскадні таблиці стилів.

DFD (англ. Data Flow Diagrams) – діаграми потоків даних у нотаціях Гейна – Сарсона, Йордана та інших, що забезпечують аналіз і функціональне проектування інформаційних систем.

ERD (англ. Entity-Relationship Diagrams) – діаграми "сутність – зв’язок" у нотаціях Чена и Баркера.

ERwin – засіб для проектування ІС та розробки структури бази даних.

Java-script – мова сценарієв.

jQuery – бібліотека, що фокусується на взаємодії JavaScript і HTML.

HTML (англ. HyperText Markup Language – «мова гіпертекстової розмітки») – стандартизована мова розмітки документів у Всесвітній павутині.

ICAM (англ. Integrated Computer Aided Manufacturing) – інтеграція комп'ютер-них і промислових технологій.

IDEF0 – методологія функціонального моделювання, що є складовою частиною SADT і що дозволяє описати бізнес-процес у вигляді ієрархічної системи взаємопов’язаних функцій.

IDEF1 – методологія аналізу і вивчення взаємозв’язків між інформаційними потоками в рамках комерційної діяльності підприємства.

IDEF1X – методологія інформаційного моделювання, заснована на концепції "сутність – зв’язок". Застосовується для розробки реляційних баз даних і використовує умовний синтаксис, що спеціально розроблений для зручної побудови концептуальної схеми і забезпечує універсальне подання структури даних в рамках підприємства, незалежне від кінцевої реалізації бази даних і апаратної платформи;

IDEF3 – методологія моделювання потоків робіт підприємства, що дозволяє подати їх сценарії за допомогою опису послідовності змін властивостей об’єкта в рамках даного процесу.

PHP (англ. Hypertext Preprocessor – «PHP: препроцесор гіпертексту»; спочатку Personal Home Page Tools – «Інструменти для створення персональних веб-сторінок») – скриптова мова загального призначення, інтенсивно застосовується для розробки веб-додатків.

phpMyAdmin – веб-додаток з відкритим кодом, написаний на мові PHP, який являє собою веб-інтерфейс для адміністрування СКБД MySQL.

SADT (англ. Structured Analysis and Design Technique) – технологія структурного аналізу й моделювання.

SQL (англ. Structured query language – «мова структурованих запитів») - декларативна мова програмування, застосовувана для створення, модифікації та управління даними в реляційній базі даних, керованої відповідною системою управління базами даних.

MySQL – вільна система керування реляційними базами даних.

UML (англ. Unified Modeling Language) – уніфікована мова моделювання.

ВСТУП

У сучасному світі управління підприємством є досить складним і громіздким завданням, що вимагає постійного контролю над бізнес-процесами. Бізнес-аналіз робить можливим проводити зміни в організації, шляхом виявлення потреб і обґрунтування рішень, які можуть бути застосовані для реалізації змін. Бізнес-аналіз дозволяє [1][1)]:

- зменшити витрати й закінчити проекти вчасно;
- задокументувати вірні вимоги;
- поліпшити ефективність проектів.

Метою бакалаврської кваліфікаційної роботи є розробка WEB-додатку бізнес-аналізу підприємства.

У першому розділі роботи будуть розглянуті деякі інструменти бізнес-аналізу. Другий розділ роботи присвячений практичному застосуванню цих інструментів на прикладі підприємства «Залізничний вокзал». У кінцевому, третьому розділі буде реалізована веб-сторінка даного підприємства.

Отже, основною метою ДП є бизнес-аналіз підприємства «Залізничний вокзал» та створення веб-додатку цього підприємства.

Для досягнення поставленої мети треба виконати наступні завдання:

- провести аналіз предметної області ІС, та побудувати концептуальну модель (IDEF1X);
- розробити функціональну модель IDEF0;
- побудувати модель послідовного виконання робіт IDEF3 та модель діаграм потоків даних DFD;
- виконати бізнес-аналіз за допомогою уніфікованої мови моделювання UML;
- розробити веб-сторінку підприємства.

[1)] [1] Бизнес-анализ. URL : https://ru.wikipedia.org/wiki/Бизнес-анализ (дата звернення 25.04.2019)

Кваліфікаційна робота складається з 83 сторінок, має три розділи та сім додатків.

1 СПОСОБИ МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

1.1 Вступ

Історично склалося так, що бізнес-аналіз був найбільш потрібний і довгий час розвивався саме в області інформаційних технологій. У даній області найбільш поширеним рішенням є автоматизація бізнес-процесів організації, тобто розробка інформаційної системи [1][1)].

Бізнес-аналіз можна проводити, використовуючи різні інструменти, програми, мови моделювання тощо. В даному розділі буде викладена основна інформація про способи моделювання даних, а також буде нада основна інформація як проводити графічний опис інформаційної системи з метою визначення основних властивостей цієї системи і функцій, які вона повинна виконувати.

В якості ПЗ (програмне забезпечення) будуть використовуватися такі програми як:

- AllFusion ERwin Data Modeler (раніше ERwin) – CASE-засіб для проектування та документування баз даних, який дозволяє створювати, документувати і супроводжувати бази даних, сховища і вітрини даних.
- CA ERwin process modeler – програма в якій проектуються моделі Business Process (IDEF0), Process Flow (IDEF3), Data Flow (DFD).
- Rational Rose – CASE засіб проектування та розробки інформаційних систем і програмного забезпечення для управління підприємствами. Принципова відмінність Rational Rose від інших засобів полягає в об'єктно-орієнтованому підході. Графічні моделі,

[1)] [1] Бизнес-анализ. URL : https://ru.wikipedia.org/wiki/Бизнес-анализ (дата звернення 25.04.2019)

що створюються за допомогою цього засобу, засновані на об'єктно-орієнтованих принципах і мові UML (Unified Modeling Language)[2][1)].

1.2 Опис метода моделювання даних IDEF1X

Найбільш поширеним засобом моделювання даних є діаграма "сутність – зв’язок". ERD-діаграма (Entity-Relationship Diagrams) дозволяє визначити основні об'єкти системи, атрибути цих об'єктів, а також відносини між об'єктами (зв'язки).

Методологія IDEF1X (Integration DEFinition for information modeling) є логічним продовженням методології IDEF1, яка у свою чергу була створена для побудови моделей даних, еквівалентних реляційним моделям в третій нормальній формі. У моделі IDEF1X були враховані такі вимоги як простота вивчення і можливість автоматизації. Дана методологія використовується деякими CASE-засобами (ERWin, Design / IDEF).

Моделі даних за методологією IDEF1X мають два рівні представлення моделі – логічний і фізичний [3][2)].

Логічний рівень представлення моделі – це абстрактний погляд на дані, на ньому дані представляються так, як вони виглядають в реальному світі. Об'єкти моделі на цьому рівні – сутності й атрибути. Логічна модель даних є універсальною і ніяк не пов'язана з конкретною СКБД (система керування базами даних). Метою побудови цієї структури є виявлення й об'єднання інформаційних вимог користувача, зв'язків між елементами даних безвідносно до їх змісту і середовища їх зберігання.

[1)] [2] Rational Rose. URL : http://kpms.ru/Automatization/Rational_Rose.htm. (дата звернення 26.04.2019)

[2)] [3] Пасічник В.В., Литвин В.В., Шаховська Н.Б. Проектування інформаційних систем. Навчальний посібник. Львів, 2013. 380 с.

Для того щоб інформаційна система адекватно відображала предметну область, необхідно добре уявляти всі нюанси, властиві даній предметній області. Предметна область повинна бути попередньо описана з використанням спеціальних мовних засобів, що не залежать від використовуваних надалі програмних засобів. Для виконання цього завдання використовують інфологічну модель.

Фізичний рівень представлення моделі описує логічну модель даних засобами конкретної СКБД. У фізичної моделі атрибути представляються як стовпці таблиць, домени перетворюються в типи даних (прийняті в обраній СКБД). Відносини й зв'язки, розроблені в логічної моделі даних, перетворюються в таблиці та у в зв'язки між ними. Також в обраній СКБД реалізуються обмеження, які мали місце в логічної моделі даних. Для цього використовуються індекси, обмеження цілісності, тригери та процедури.

Процес побудови інформаційної моделі складається з наступних кроків:

а) визначення сутностей;

б) визначення залежностей між сутностями;

в) задачі первинних і альтернативних ключів;

г) визначення атрибутів сутностей;

д) приведення моделі до необхідного рівня нормальної форми;

е) перехід до фізичного опису моделі;

1) призначення відповідностей;
   - ім'я сутності – ім'я таблиці;
   - атрибут сутності – атрибут таблиці;
2) завдання тригерів, процедур і обмежень;

ж) генерація бази даних.

Також слід зазначити, що у методології IDEF1X сутність візуально представляє три основних види інформації (див. табл. 1.1):

- атрибути, складові первинного ключа;

– не ключові атрибути;

– тип сутності (незалежна / залежна).

Таблиця 1.1 – Представлення сутностей

Ім’я сутності

Атрибути, які складають первинний ключ
Атрибути

Erwin [4][1] – засіб для проектування ІС та розробки структури бази даних (БД). ERwin поєднує графічний інтерфейс Windows, інструменти для побудови ER-діаграм (entity-relationship diagram), редактори для створення логічного та фізичного опису моделі даних і прозору підтримку провідних реляційних СКБД і настільних баз даних. За допомогою ERwin можна створювати або проводити зворотне проектування (реінжиніринг) баз даних. Цей програмний продукт – лідер серед засобів моделювання баз даних і сховищ даних. Дозволяє проектувати, документувати і супроводжувати бази даних різних типів. Підтримуючи пряме і зворотне проектування для 20 типів СКБД, ERwin підвищує якість розроблюваної БД, продуктивність праці і швидкість розробки.

1.3 Методологія функціонального моделювання SADT

Методологія SADT [5][2] (Structured Analysis and Design Technique – технологія структурного аналізу й моделювання) розроблена Дугласом Т. Россом в 1969-1973 р. SADT – одна з найвідоміших і широко використовуваних методик моделювання. Нова назва методики, прийнята як стандарт – IDEF0

[1] [4] Маклаков С.С. BPwin и Erwin. CASE- средства разработки информационных систем. М.:ДИАЛОГ-МИФИ, 2000. 256с.

[2] [5] Ременяк Л.В. Управління ІТ проектами. Конспект лекцій. Одеса.: ОДЕКУ, 2015, 168 с.

(Icam DEFinition) – частина програми ICAM (Integrated Computer Aided Manufacturing – інтеграція комп'ютерних і промислових технологій).

Методологія SADT являє собою сукупність методів, правил і процедур, призначених для побудови функціональної моделі об'єкта якої-небудь предметної області. Процес моделювання починається з аналізу предметної області й включає:

а) збір інформації про досліджувану область;

б) документування отриманої інформації;

в) подання її у вигляді моделі.

Аналіз предметної області безпосередньо пов'язаний з поняттями, що визначають суб'єкт моделювання, мету й точку зору на модель, що дозволяють найбільше точно розглянути досліджувану область [6][1)].

Модель в SADT може містити наступні типи діаграм:

а) контекстну діаграму (у кожній моделі може бути тільки одна контекстна діаграма);

б) діаграми декомпозиції;

в) діаграми дерева вузлів.

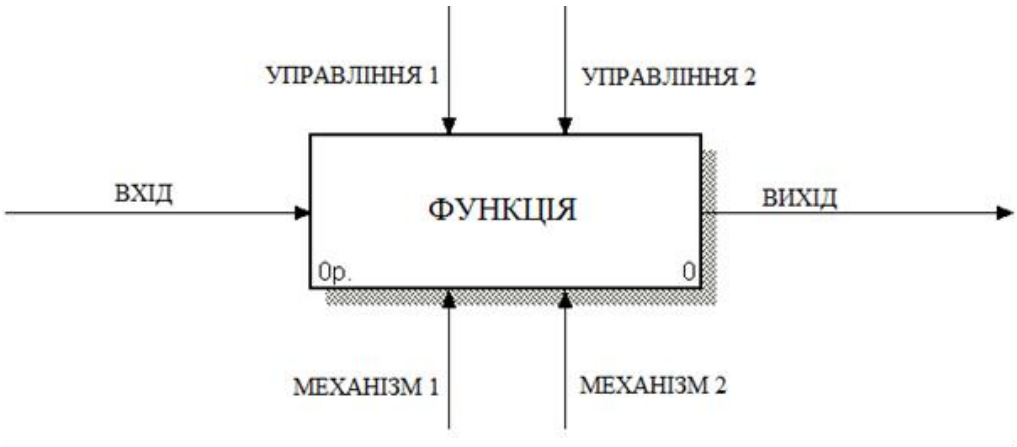


Рисунок 1.1 – Функціональний блок із граничними стрілками

[1)] [6] Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. М: «Финансы и статистика», 1998. 123 с.

Контекстна діаграма є вершиною деревоподібної структури діаграм й являє собою найзагальніший опис системи і її взаємодію із зовнішнім середовищем. Моделювання починається з подання системи як єдиного цілого – одного функціонального блоку із граничними стрілками, що простираються за межі розглянутої області.

Після опису системи в цілому проводиться розбивка її на великі фрагменти. Цей процес називається функціональною декомпозицією, а діаграми, які описують кожен фрагмент і взаємодію фрагментів, називаються діаграмами декомпозиції. Після декомпозиції контекстної діаграми проводиться декомпозиція кожного великого фрагмента системи на більш дрібні й так далі, до досягнення потрібного рівня подробиці опису.

Діаграма дерева вузлів показує ієрархію робіт у моделі й дозволяє розглянути всю модель цілком, але не показує взаємозв'язку між роботами. Діаграм дерев вузлів може бути в моделі як завгодно багато, оскільки дерево може бути побудоване на довільну глибину й не обов'язково з кореня [7][1)].

1.4 Методологія послідовного виконання процесів (стандарт IDEF3)

Для опису логіки взаємодії інформаційних потоків підходить методологія, яка називається Workflow diagramming – методологія моделювання, що використовує графічний опис інформаційних потоків послідовного виконання дій у часі, взаємин між процесами обробки інформації та об'єктів, що є частиною цих процесів. Діаграми Workflow можуть бути використані в моделюванні бізнес-процесів для аналізу завершеності процедур обробки інформації [5][2)].

[1)] [7] Вендров А.М. Проектирование программного обеспечения экономических информационных систем. М.: Финансы и статистика, 2006. 544 с.

[2)] [5] Ременяк Л.В. Управління ІТ проектами. Конспект лекцій. Одеса.: ОДЕКУ, 2015, 168 с.

Кожна робота в IDEF3 описує сценарій будь-якого процесу і може бути складовою іншої роботи. Оскільки сценарій описує мету і рамки моделі, важливо, щоб роботи іменувалися віддієслівним іменником, що позначає процес дії, або фразою, що містить такий іменник.

Графічні елементи, використані в цій методології опису процесів, включають одиниці роботи UOW (Unit Of Work), зв'язки старшинства, вузли або перехрестя, модулі посилань і приміток.

Одиниці роботи, також звані роботами, є центральними компонентами моделі. В IDEF3 роботи зображуються прямокутниками із прямими кутами й мають ім'я, виражене віддієслівним іменником, що позначає процес дії, одиничний або в складі фрази. Ім'я іменник у складі фрази звичайно відображає основний вихід роботи (наприклад, "Виготовлення виробу"). Крім імені кожна робота має свій порядковий номер, що визначає його місце в діаграмах і складається з номера батьківської роботи, номера версії декомпозиції й порядкового номера на поточній діаграмі (див. табл.1.2).

Таблиця 1.2 – Одиниця роботи

Ім'я роботи	
Номер	

Зв'язки показують взаємини робіт. Всі зв'язки в IDEF3 односпрямовані й можуть бути спрямовані куди завгодно, але зазвичай діаграми IDEF3 намагаються побудувати так, щоб зв'язки були спрямовані зліва направо.

В IDEF3 розрізняють три типи стрілок, що зображують зв'язки [7][1)]:

а) Старший зв'язок – суцільна лінія зі стрілкою, що зв'язує одиниці робіт. Малюється зліва направо або зверху вниз. Показує, що робота-

[1)] [7] Вендров А.М. Проектирование программного обеспечения экономических информационных систем. М.: Финансы и статистика, 2006. 544 с.

джерело повинна закінчитися перш, ніж почнеться робота-мета (рис.1.2).

б) Потік об'єктів – стрілка із двома наконечниками, підсилює старший зв'язок і застосовується для опису того факту, що результатом виконання роботи-джерела стає об'єкт, необхідний для виконання роботи-мети (рис.1.3).

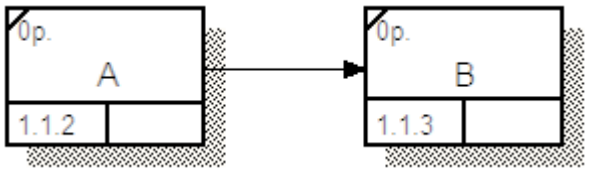


Рисунок 1.2 – Старший зв'язок

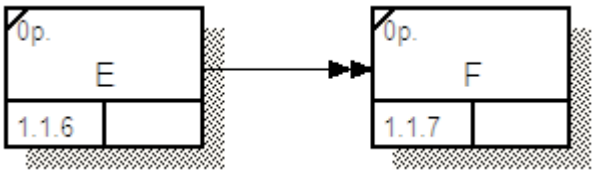


Рисунок 1.3 – Потік об'єктів

в) Відношення – пунктирна лінія зі стрілкою, яка використовується для зображення зв'язків між одиницями робіт (рис.1.4).

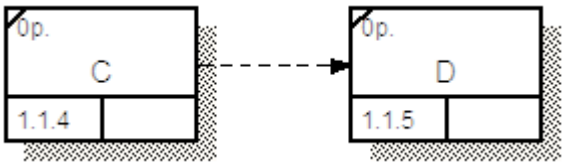


Рисунок 1.4 – Зв'язок відносини

Відношення показує, що стрілка є альтернативою старшій стрілці або потоку об'єктів у сенсі завдання послідовності виконання робіт – робота-

джерело не обов'язково повинна закінчитися, перш ніж робота-ціль почнеться. Більше того, робота-ціль може закінчитися перш, ніж закінчиться робота-джерело.

Закінчення однієї роботи може служити сигналом до початку декількох робіт, або ж одна робота для свого запуску може очікувати закінчення декількох робіт. Перехрестя використовують для відображення логіки взаємодії стрілок при злитті й розгалуженні або для відображення безлічі подій, які можуть або повинні бути завершені до початку наступної роботи.

Розрізняють перехрестя розгалуження стрілок і перехрестя для злиття стрілок. Перехрестя не може використовуватися одночасно для злиття й для розгалуження (рис.1.5).

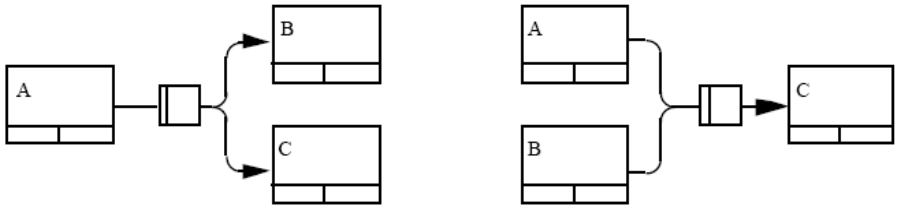


Рисунок 1.5 – Приклади перехресть розгалуження й злиття

Розрізняють кілька типів перехресть, що визначають логіку поводження паралельно розташованих робіт і залежність від моменту початку або закінчення тієї або іншої роботи. Побудову діаграм варто починати з контекстної діаграми, що зображує основну функцію системи. В IDEF3, також як й в IDEF0, для більш детального подання дій в описуваному процесі використовується декомпозиція функціональних модулів. Декомпозиції модулів представляються на окремих діаграмах. На рис.1.6 представлений приклад декомпозиції модулів і принцип формування їхніх номерів. Для наочності всі модулі представлені на одному малюнку, але в IDEF3 описі вони будуть представлені на трьох різних діаграмах.

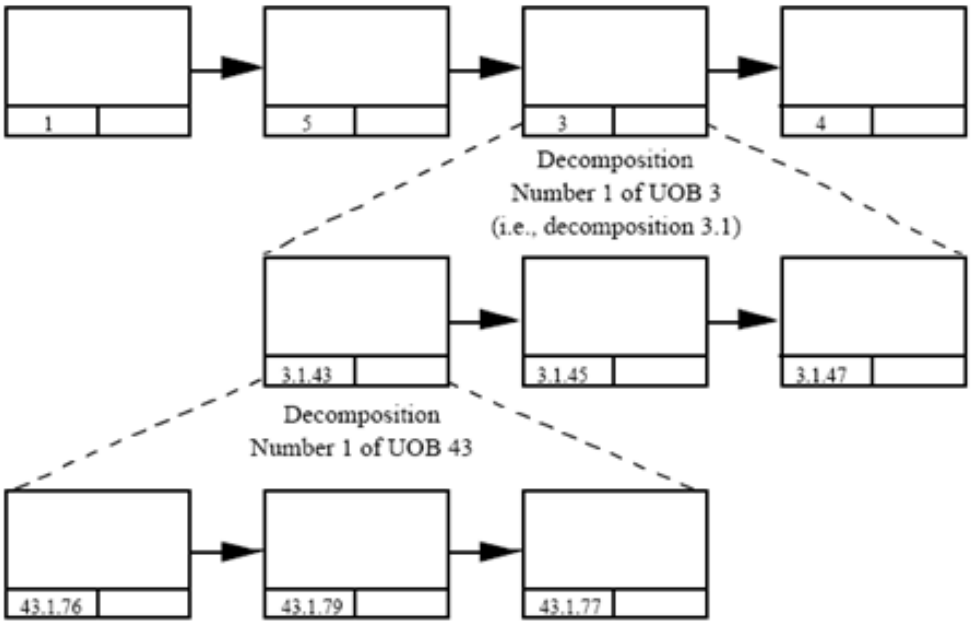


Рисунок 1.6 – Декомпозиція модулів в IDEF3 діаграмах

Для опису відносин між роботами в IDEF3 використовується термін активізація. Під активізаціями розуміємо можливий набір станів частини або всіх модулів, що задовольняють тимчасовим або логічним умовам, заданим схемою, при яких активізується один або кілька зображених на схемі модулів.

1.5 Методологія моделювання діаграм потоків даних DFD

Діаграми потоків даних використовуються для опису руху документів й обробки інформації як доповнення до методології функціонального моделювання IDEF0. На відміну від методології IDEF0, стрілки на діаграмах DFD показують лише те, як об'єкти (включаючи дані) рухаються від однієї роботи до іншої. Діаграма потоків даних DFD – це граф, на якому показаний рух

значень даних від їхніх джерел через процеси, що їх перетворюють до споживачів в інших об'єктах [7][1].

Основними елементами моделі, що поєднує діаграми потоків даних, є:

а) процеси;

б) зовнішні сутності;

в) сховища даних;

г) потоки даних.

Процеси являють собою перетворення вхідних потоків даних у вихідні відповідно до певного алгоритму. У реальному житті процес може виконуватися деяким підрозділом організації, що виконує обробку вхідних документів і випуск звітів. Ці дії можуть виконуватися окремим співробітником, програмою, установленої на комп'ютері, спеціальним логічним пристроєм тощо [6][2].

Процеси на діаграмі потоків даних зображуються прямокутниками з округленими кутами (рис.1.7).

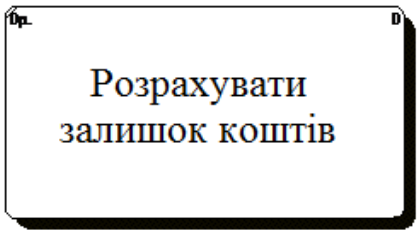


Рисунок 1.7 – Приклад процесу

Зовнішня сутність являє собою матеріальний об'єкт, що є джерелом або приймачем інформації або даних. Визначення деякого об'єкта як зовнішня

[1] [7] Вендров А.М. Проектирование программного обеспечения экономических информационных систем. М.: Финансы и статистика, 2006. 544 с.

[2] [6] Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. М: «Финансы и статистика», 1998. 123 с

сутність вказує на те, що він перебуває за межами границь аналізованої предметної області.

Зовнішні сутності зображуються у вигляді прямокутників з тінню (рис.1.8) і зазвичай розташовуються по краях діаграми.

Зовнішня сутність ідентифікується буквою "E" і відповідним номером. Усередині символу вказується його ім'я, наприклад, замовник, персонал, постачальник, клієнт.

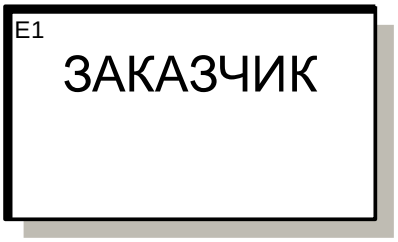


Рисунок 1.8 – Приклад зовнішньої сутності

У процесі аналізу деякі зовнішні сутності можуть бути використані багаторазово на одній або декількох діаграмах. Крім того, частина процесів системи може бути винесена за межі діаграми й представлена як зовнішні сутності.

Сховище даних являє собою абстрактний пристрій для зберігання інформації, яку можна в будь-який момент помістити в сховище й через якийсь час витягти, причому способи переміщення та вилучення можуть бути будь-якими.

Сховище даних на діаграмі потоків даних зображується, як показано на рис.1.9.

Сховище даних ідентифікується буквою "D" і відповідним номером. Усередині вказується його унікальне в рамках даної моделі ім'я, що найбільше точно, з погляду аналітика, відображає інформаційну сутність умісту, наприклад, "Відомості про постачальників", "Накладні".

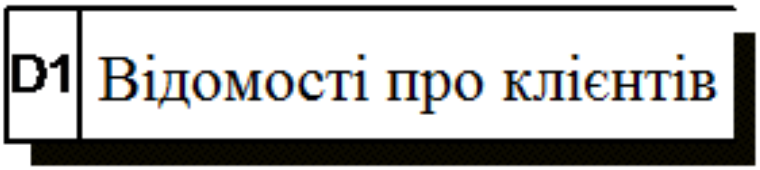


Рисунок 1.9 – Приклад сховища даних

Сховища даних у загальному випадку є прообразом таблиць майбутньої бази даних й опис даних, що зберігаються в них, повинні бути вв'язані з інформаційною моделлю.

Потоки даних описують рух інформації або об'єктів з однієї частини системи в іншу.

Потоки даних зображуються лініями зі стрілками, що показують їхній напрямок. Оскільки кожна сторона прямокутника, що зображує процес, не має певного призначення, потоки даних можуть підходити до будь-якої сторони й виходити з будь-якої сторони. Кожному потоку даних привласнюється ім'я, що відображає його зміст (рис.1.10).

Рисунок 1.10 – Потік даних від процесу до зовнішньої сутності

Діаграми потоків даних будуються за ієрархічним принципом. Першим кроком при побудові ієрархії діаграм є побудова контекстної діаграми (рис.1.11).

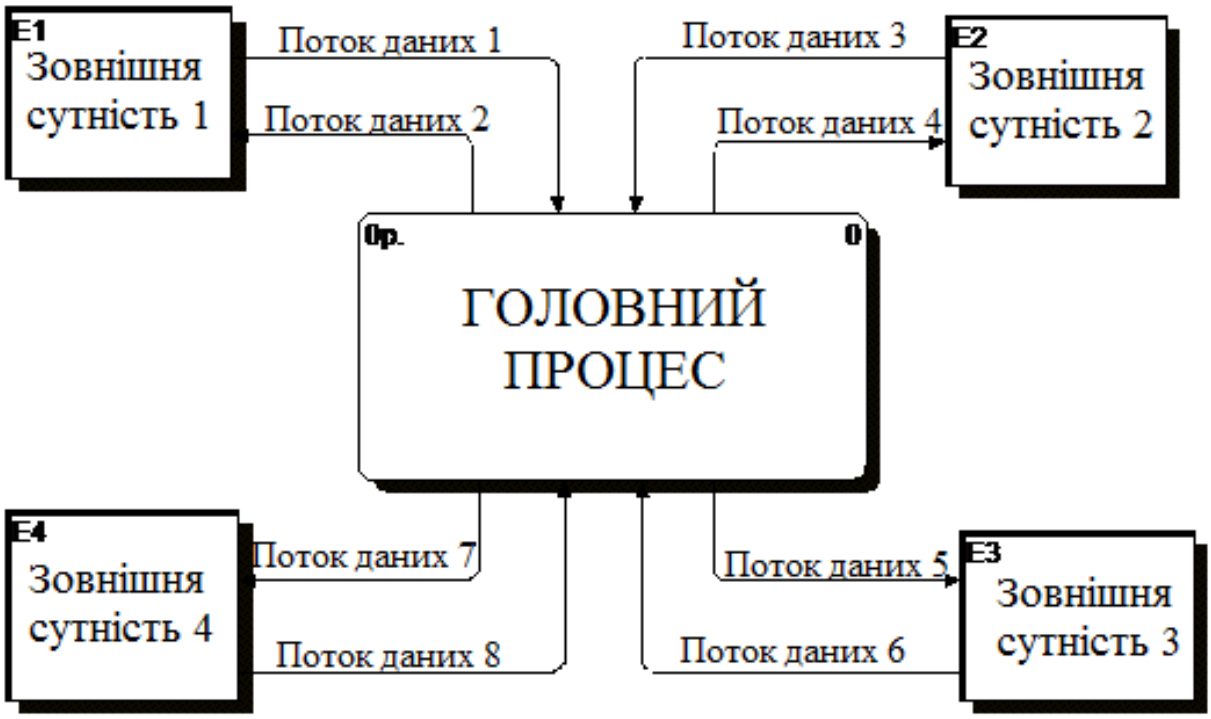


Рисунок 1.11 – Контекстна діаграма потоків даних

Контекстна діаграма визначає границі моделі. Як правило, вона має зіркоподібну топологію, у центрі якої перебуває головний процес, з'єднаний із приймачами й джерелами інформації, що є зовнішніми сутностями інформаційної системи, що моделюється.

Для головного процесу, який присутній на контекстній діаграмі, проводиться декомпозиція. На першому рівні ієрархії показуються основні внутрішні процеси системи й відповідні їм зовнішні сутності, сховища й потоки даних (рис.1.12).

Для кожного процесу діаграми першого рівня може бути зроблена декомпозиція, що, у свою чергу, також може бути розкрита більш докладно. Декомпозиція процесів закінчується, коли досягнутий необхідний ступінь деталізації або відображені на черговому рівні діаграм процеси є елементарними й не можуть бути розбиті на більш дрібні.

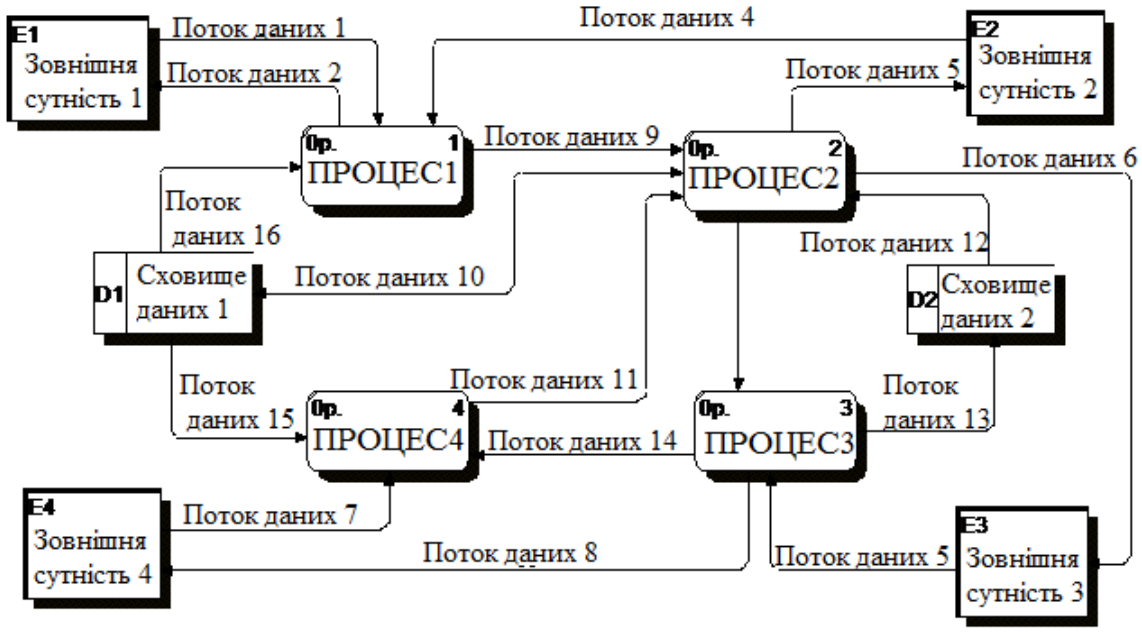


Рисунок 1.12 – Діаграма декомпозиції в моделі DFD

При проведенні декомпозиції повинне виконуватися правило балансування – при декомпозиції процесу дочірня діаграма як зовнішні сутності може мати тільки ті об'єкти (процеси, зовнішні сутності, сховища даних), з якими має інформаційний зв'язок процес, що деталізується на батьківській діаграмі [7][1)].

1.6 Побудування ІС за допомогою UML

На сьогодні для об’єктно-орієнтованого моделювання проблемної сфери широко використовується уніфікована мова моделювання UML (Unified Modeling Language), яка розроблена групою провідних комп’ютерних фірм світу OMG (Object Management Group) і фактично є стандартом з об’єктно-орієнтованих технологій. Мова UML реалізована багатьма фірмами-виробниками програмного забезпечення в рамках CASE-технологій, напри-

[1)] [7] Вендров А.М. Проектирование программного обеспечения экономических информационных систем. М.: Финансы и статистика, 2006. 544 с.

клад Rational Rose (Rational), Natural Engineering Workbench (Software AG), ARIS Toolset (IDS prof. Scheer) та ін.

Уніфікована мова моделювання є графічною мовою для візуалізації, специфікації, конструювання і документування систем, у яких велика роль належить програмному забезпеченню. За допомогою мови UML можна розробити детальний проект створюваної системи, що відображає [6][1]:

- концептуальні елементи: системні функції і бізнес-процеси;
- конкретні особливості реалізації: класи, написані спеціальними мовами програмування, схеми баз даних, програмні компоненти багаторазового використання.

На рис. 1.13 показані відносини між різними видами діаграм UML [8][2].

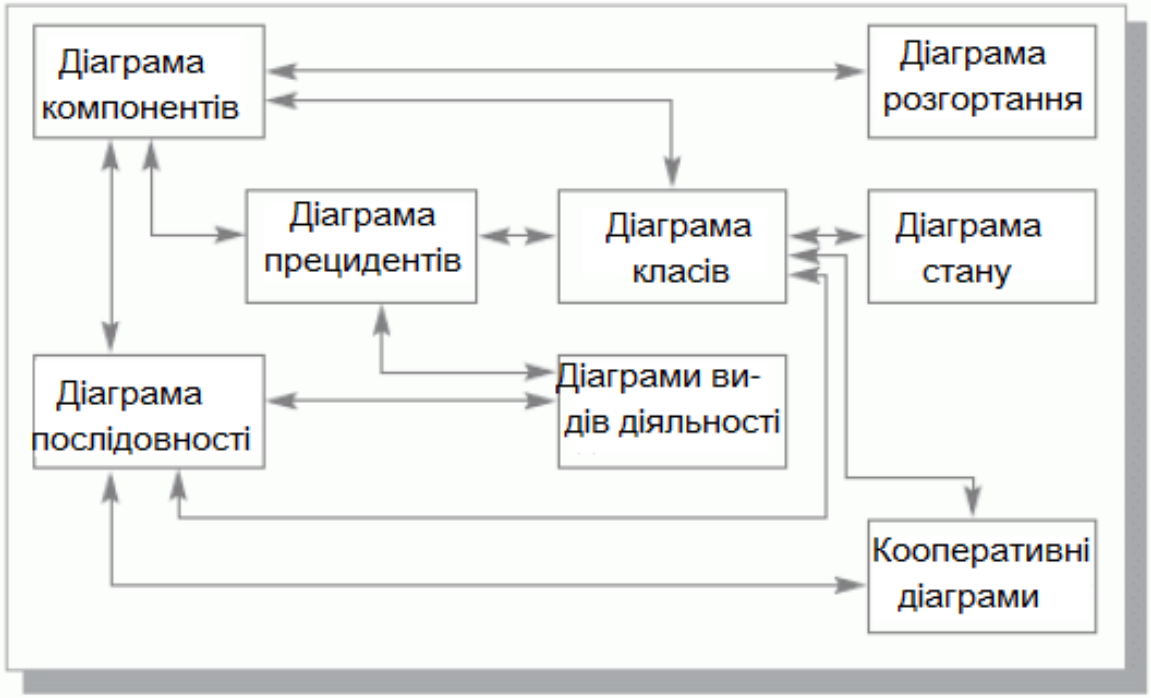


Рисунок 1.13 Взаємозв'язки між діаграмами UML

[1] [6] Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. М: «Финансы и статистика», 1998. 123 с.

[2] [8] Грекул В. И., Денищенко Г. Н., Коровкина Н. Л. Проектирование информационных систем. М. : Интернет-Ун-т Информ технологий, 2005. 304 с.

В даному дипломному проекті було розглянуто два види UML-діаграм:

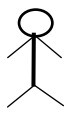
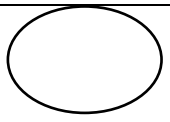
а) Діаграма варіантів використання (Use-case diagram), що відображає функціональність ІС у вигляді сукупності послідовностей, що виконуються та транзакцій;

б) Діаграма класів об’єктів (Class diagram), що відображає структуру сукупності взаємозалежних класів об’єктів аналогічно до ER-діаграми функціонально-орієнтованого підходу;

1.6.1 Діаграма варіантів використання

Діаграма варіантів використання виявляє основні бізнес-процеси як послідовності транзакцій, які повинні виконуватися повністю, коли виконання відособленої підмножини дій не має значення без виконання всієї послідовності. Варіанти використання ініціюються із зовнішнього середовища користувачами ІС, так званими акторами. На цьому рівні моделювання не розкривається механізм реалізації процесів. Наведені сутності мають графічні позначення, наведені в табл.1.3 [9][1)].

Таблиця 1.3 – Об’єкти, використовувані в діаграмі варіантів використання

Найменування об’єкта	Опис	Графічне позначення
Актор	Зовнішній користувач процесу	
Варіант використання	Бізнес-процес	
Пойменована стрілка	Позначення події	

[1)] [9] Ременяк Л.В. Проектування інформаційних систем. Конспект лекцій. Одеса.: ОДЕКУ, 2016, 152 с.

Актор ініціює виконання варіанта використання і отримує від нього результати. Взаємодія (асоціація) актора з варіантом використання здійснюється внаслідок події, яка позначається пойменованою стрілкою. Один актор може брати участь у декількох варіантах використання, а в одному варіанті використання може бути зайнято декілька акторів.

У реалізації варіанта використання можливе виділення декількох потоків подій:

- основний потік подій, який приводить до необхідного результату найбільш коротким шляхом;
- альтернативні потоки подій.

Основний і альтернативний потоки подій у моделі варіантів використання описуються у вигляді неформальних текстових коментарів.

Декілька варіантів використання можуть мати спільну частину, що виділяється в самостійний варіант використання, з яким установлюються відносини використання (uses). З іншого боку, певні варіанти використання можуть бути розширені деталями. У такому разі створюється додатковий варіант використання, з яким встановлюються відносини розширення (extends).

1.6.2 Діаграма класів об'єктів

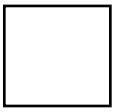
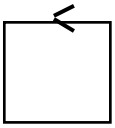
Діаграми класів об'єктів відображають статичну структуру класів об'єктів. Ця діаграма розглядає внутрішню структуру предметної сфери, ієрархію класів об'єктів, статичні зв’язки об'єктів [9][1)].

Класи об'єктів можуть мати різні стереотипи поведінки: об'єкти- сутності, керівні об'єкти, інтерфейсні об'єкти (табл.1.4). Об'єкти, відображені в діаграмі класів об'єктів, пов’язуються статичними відносинами, які відбивають постійні зв'язки між об'єктами незалежно від виконання конкретного бізнес-

[1)] [9] Ременяк Л.В. Проектування інформаційних систем. Конспект лекцій. Одеса.: ОДЕКУ, 2016, 152 с.

процесу. До статичних відносин відносяться узагальнення, агрегація, асоціація об'єктів.

Таблиця 1.4 – Об'єкти, використовувані в діаграмі класів об'єктів

Найменування об'єкта	Опис	Графічне позначення
Об'єкт-сутність	Пасивний об'єкт, над яким виконуються операції обробки процесу	
Керівний об'єкт	Активний об'єкт, що координує виконання функцій	
Відносини асоціації	Відносини типу 0.. 1:1;0..1:M; M:N. Відносини можуть бути пойменованими. 0..1 – необов'язковість зв'язку; * – множинність зв'язку	
Відносини узагальнення	Відносини спадкоємства	
Відносини агрегації	Відносини «ціле – частина»	

Діаграма класів є ключовим елементом в об'єктно-орієнтованому моделюванні. На діаграмі класи представлені в рамках, що містять три компоненти:

У верхній частині написано ім'я класу. Ім'я класу вирівнюється по центру і пишеться напівжирним шрифтом. Імена класів починаються з великої літери. Якщо клас абстрактний – то його ім'я пишеться напівжирним курсивом. Посередині розташовуються поля (атрибути) класу. Вони вирівняні по лівому краю і починаються з маленької літери. Нижня частина містить методи класу. Вони також вирівняні по лівому краю і пишуться з малої літери.

Отже, у цьому розділі були розглянуті різні інструменти бізнес-аналізу підприємства, які дозволяють:

а) розробити інформаційну систему підприємства;
    1) визначити основні атрибути і зв'язки цієї системи;
    2) створити базу даних системи;
б) побудувати функціональну модель об'єкта цієї системи;
в) описати логіку взаємодії інформаційних потоків;
г) описати рух документів й обробку інформації, виявити основні бізнес-процеси як послідовності транзакцій, які повинні виконуватися повністю, коли виконання відособленої підмножини дій не має значення без виконання всієї послідовності.
д) відобразити статичну структуру класів об’єктів.
е) розглянути:
    1) внутрішню структуру предметної сфери;
    2) ієрархію класів об’єктів;
    3) статичні зв’язки об’єктів.

2 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ПІДПРИЄМСТВА

2.1 Формулювання завдання

В даному ДП під АТ (акціонерне товариство) «Залізничний вокзал» мається на увазі деяке гіпотетичне підприємство, основною діяльністю якого є перевезення пасажирів. У перевезенні бере участь персонал, поїзд і сам пасажир. Основне завдання даного розділу – це розробити інформаційну систему використовуючи методи описані в першому розділі цього ДП.

Ця система допоможе краще зрозуміти струтуру підприємства, а також його основні потреби. Так само вона полегшить створення програмного забезпечення необхідного для успішного функціонування даного підприємства.

2.2 Аналіз предметної області ІС, та побудова концептуальної моделі (IDEF1X)

Назва системи: пасажирський залізничний вокзал. Згідно з першим розділом процес побудови ІС був розбитий на окремі пункти [10][1)]:

а) Основні функції системи, а також інформаційні об'єкти, пов'язані з кожною функцією наведені в таб. 2.1.

Таблиця 2.1 – Основні функції та об’єкти системи

Функції системи	Інформаційні об'єкти
1	2
Список поїздів	Поїзд
Список маршрутів	Маршрут
Кількість персоналу	Персонал
Фінансові дані	Квиток
Дані про пересування поїздів	Графік
Реєстрація пасажирів	Пасажир
Список вокзалів	Вокзал
Надати послугу	Послуга
Надати вагони	Вагон
Надати послуги	Каса

б) Опис сутностей з відповідними атрибутами:

1) поїзд (id поїзда, кількість вагонів, тип, швидкість, рік випуску);
2) маршрут (id маршруту, назва, дистанція, кількість зупинок);
3) персонал (id персоналу, ПІБ, посада, адреса, телефон);
4) квиток (№ квитка, вартість, дата відправлення);

[1)] [10] Ременяк Л.В. Методичні вказівки для виконання лабораторних робіт по дисципліні «Проектування інформаційних систем». Одеса: ОДЕКУ, 2016. 24 с.

5) графік (id графіка, час прибуття, пункт прибуття, кількість часу зупинки, час зупинки, пункт відбуття);
6) пасажир (id паспорта, ПІБ, телефон);
7) вокзал (id вокзалу, назва, адреса, телефон, кількість гілок);
8) послуга (id послуги, назва, вартість, дата надання);
9) вагон (№ вагона, кількість місць, клас);
10) каса (id каси, список квитків, вартість квитків);

в) Визначення відносин між сутностями (зв'язок):

1) поїзд <складається з / входять до складу> вагонів;
2) поїзд <їздить по> маршрутам;
3) персонал <обслуговує> потяги;
4) вокзал <приймає> потяги;
5) вагони <обслуговуються> персоналом;
6) на один вагон можна купити кілька квитків;
7) вокзал обслуговується персоналом;
8) персонал працює за графіком;
9) вокзал надає послуги;
10) один квиток можна купити в декількох касах;
11) один пасажир може купити кілька квитків;

Після опису предметної області, слідує етап інфологічного проектування, у якому була складена інфологічна схема предметної області (рис. А.1).

Далі на основі логічної моделі була спрограмована фізична модель даних (рис. А.2).

На останньому етапі, після створення фізичної моделі даних було виконано генерацію самої бази даних (рис. 2.1). Це можна зробити в середовищі ERwin Data Modeler указавши необхідний тип СКБД.

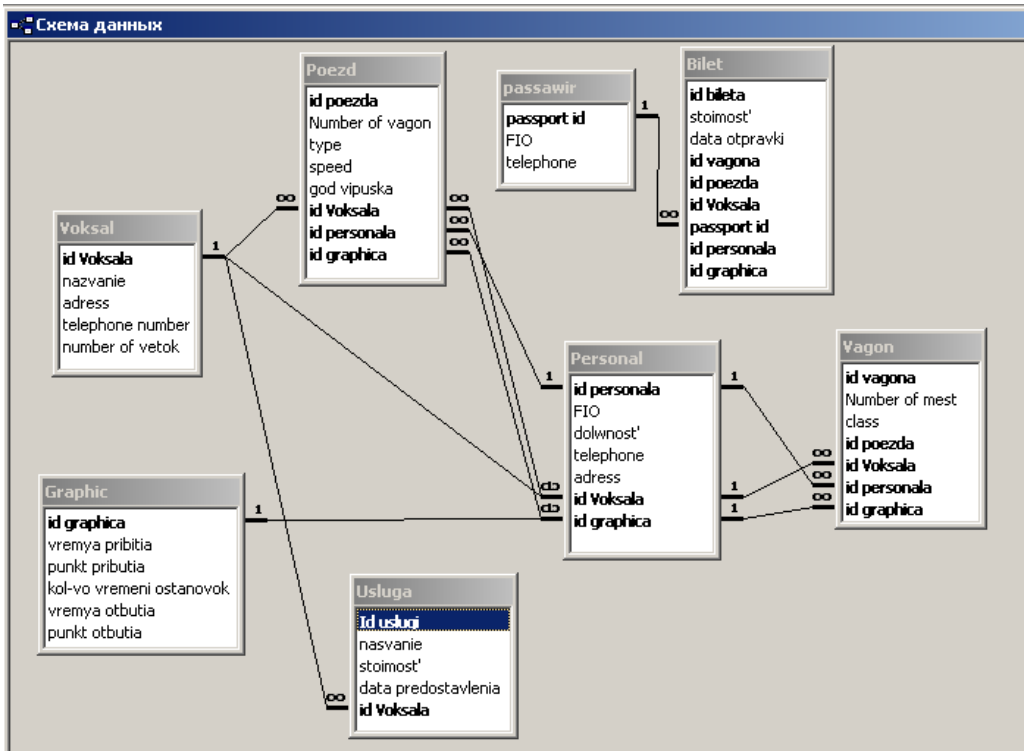


Рисунок 2.1 – Схема БД яка була згенерована середовищем ERWin

2.3 Побудова функціональної моделі IDEF0

Спочатку була побудована контекстна діаграма, в якості мети було вказано, що необхідно організувати перевезення пасажирів [11][1)]. Точка зору – персонал, під суб'єктом мається на увазі сама система в якій пасажир повинен бути перевезений. Контекстна діаграма зображена на рис. 2.2.

Наступним етапом є декомпозиція першого рівня: з якої випливають п'ять робіт: продати квитки, прийняти пасажира в поїзд, перевезти пасажира, відремонтувати поїзд і нарешті виплатити зарплату.

Далі в залежності від потреби кожну з робіт діаграми декомпозиції першого рівня можна розбити на діаграми декомпозиції другого рівня і так далі. В даному дипломному проекті обмежимося першим рівнем (рис. Б.1).

[1)] [11] Ременяк Л.В. Методичні вказівки для виконання лабораторних робіт по дисципліні «Управління ІТ-проектами». Одеса: ОДЕКУ, 2016. 53 с.

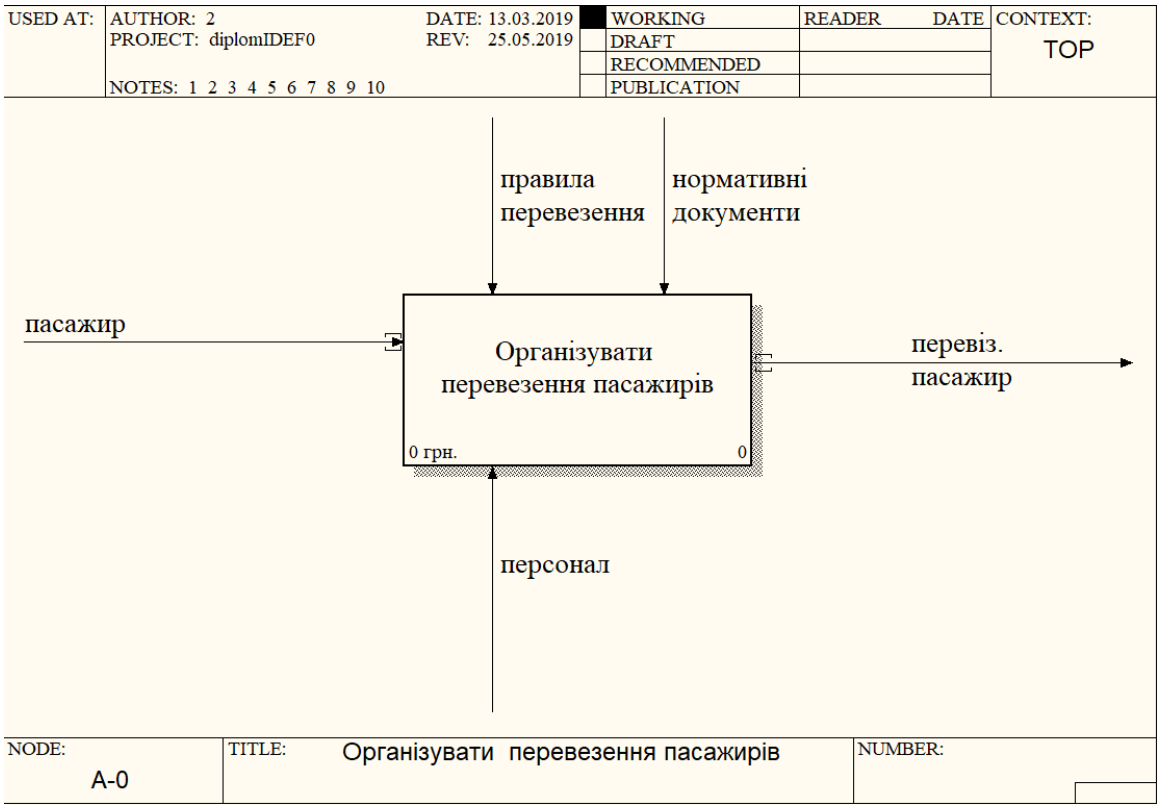


Рисунок 2.1 – Контекстна діаграма моделі IDEF0

Результатом роботи блоку 1 є список проданих квитків, який направляється як елемент керування до наступного блоку. Результатом роботи блоку 2 є прийнятий пасажир який потрапляє до роботи «перевезти пасажира».

Перевезений пасажир є метою моделювання тому стрілка з блоку 3 відразу проходить в тунель, а також передається як елемент управління до блоку відремонтувати потяг (перед тим як ремонтувати потяг потрібно провести висадку пасажирів). Останній блок описує сплату зарплатні персоналу, а тому сам пасажир не є частиною цієї роботи. До роботи лише надходять гроші пасажира у якості елемента управління.

Далі була побудована діаграма дерева вузлів, яка показує ієрархічну залежність робіт, але не взаємозв'язок між роботами (рис. Б.2). В ній можна побачити блок контекстної діаграми, перший рівень декомпозиції у формі

прямокутників, а також наступний рівень декомпозиції, який представлений списком робіт.

2.4 Побудова моделі послідовного виконання робіт IDEF3

За аналогією з моделлю IDEF0 спочатку була побудована контекстна діаграма. Вона складається з однієї роботи і має аналогічну назву [11][1)].

Далі була побудована діаграма декомпозиції першого рівня. По ній можна визначити порядок виконання робіт: спочатку повинні бути запущені роботи (асинхронне «І») 1.1.2 і 1.1.3. Далі для того щоб виконалася основна робота 1.1.4 (Перевезти пасажира) повинні виконуватися всі інші роботи.

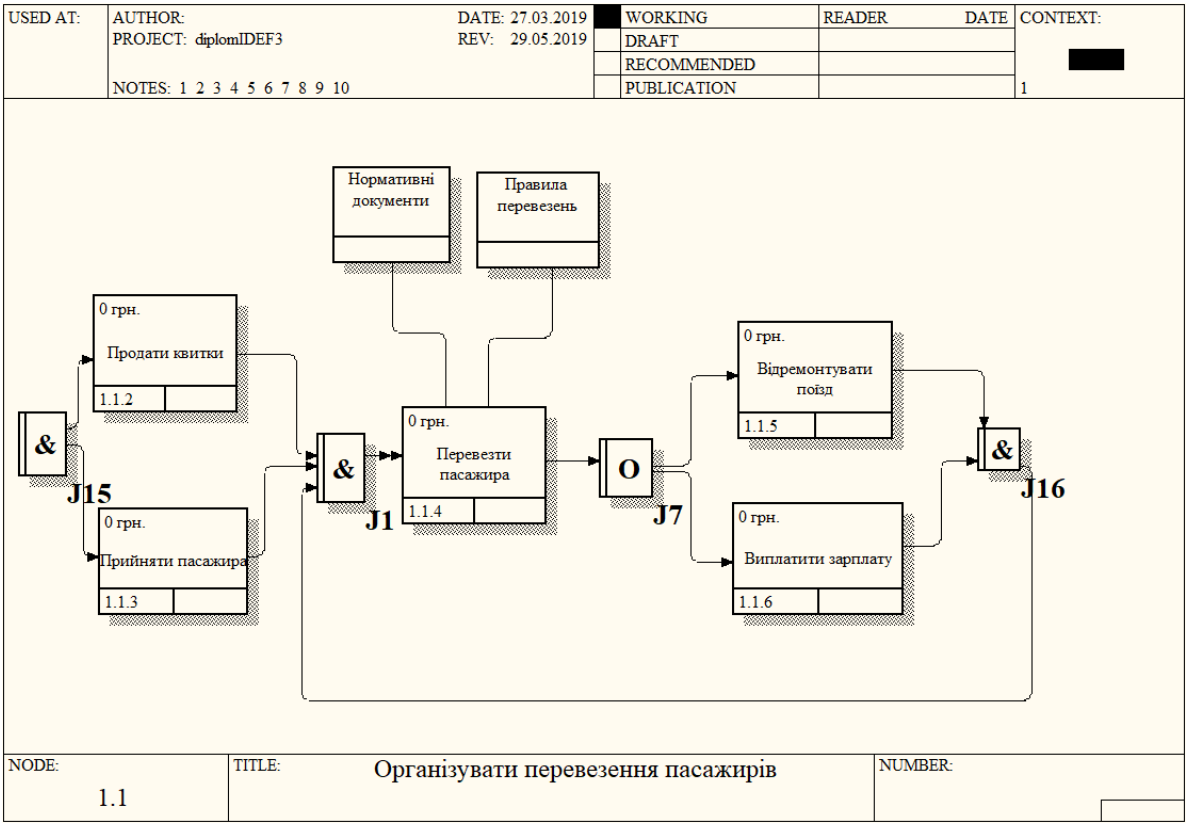


Рис. 2.3 – Діаграма декомпозиції першого рівня (IDEF3)

[1)] [11] Ременяк Л.В. Методичні вказівки для виконання лабораторних робіт по дисципліні «Управління ІТ-проектами». Одеса: ОДЕКУ, 2016. 53 с.

Варто також звернути увагу на роботи 1.1.5 і 1.1.6 які можуть бути запущені тільки після того як буде проведене перевезення пасажирів, тому що щоб виплатити зарплату персоналу, а так же відремонтувати поїзд необхідно для початку заробити гроші перевізши пасажира. Перехрестя J7 (асинхронне «або») так само говорить нам про те, що один або кілька наступних процесів повинні бути запущені.

Далі, орієнтуючись на діаграму декомпозиції першого рівня, була побудована схема активізації для опису відносин між роботами. Схема представлена на рис. 2.3.

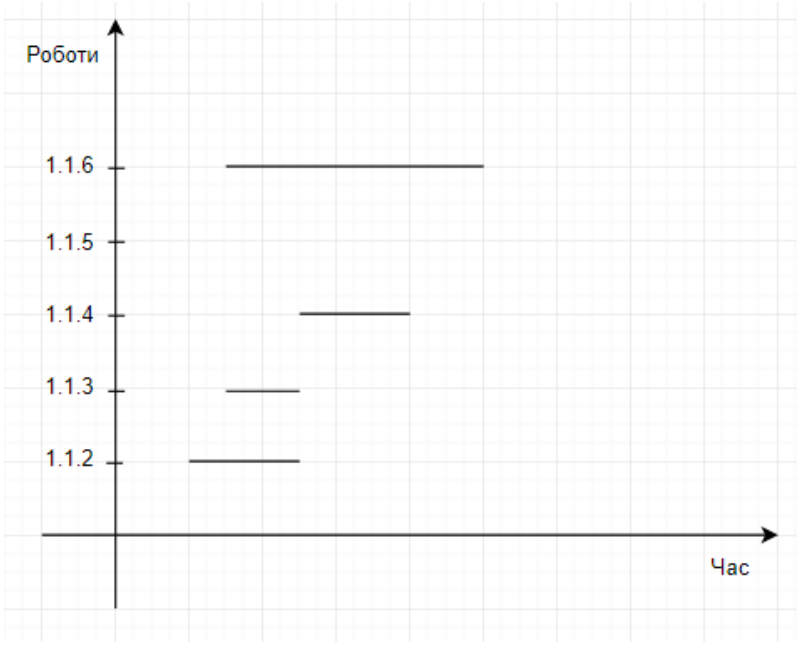


Рисунок 2.3 – Схема активізацій діаграми декомпозиції першого рівня

2.5 Побудова моделі діаграм потоків даних DFD

В першу чергу була побудована контекстна діаграма (рис. В.1).

Головний процес в системі відображений прямокутником із закругленими краями, і має назву «Організувати перевезення пасажирів» [11][1)].

В якості зовнішніх сутностей які взаємодіють з головним процесом на нашій контекстної діаграмі виступають суті E1 («Пасажир») і E3 («Персонал»). Об'єкт «Пасажир» звертається до головного процесу поставляючи в нього гроші і бажання подорожувати, а на виході з головного процесу отримує задоволене бажання. Виконується це бажання за допомогою обслуговування персоналом, який керується нормативними документами та правилами перевезень, відображені на цій діаграмі сховищами даних, які постачають в головний процес відповідно документи і правила.

Далі слідує діаграма декомпозиції першого рівня (рис В.2). У ній показуються основні внутрішні процеси системи й відповідні їм зовнішні сутності, сховища й потоки даних. Головний процес ділиться на п'ять дочірніх процесів які взаємодіють між собою інформацією і кожен з яких так само може бути розкладений більш детально.

2.6 Побудова діаграми варіантів використання

Діаграма варіантів використання зображена у додатку Г (рис. Г.1). Вона дозволяє описати основні вимоги до системи «Залізничий вокзал». Дійовими особами або «акторами», які зображені на діаграмі в формі чоловічка відповідають чотири сутності: вокзал, поїзд, персонал і пасажир. Вони взаємодіють один з одним в системі за допомогою передачі та прийняття повідомлень від варіантів використання [10][2)].

Пасажир має такі функції в системі: він може купити послугу, в яку може входити і покупка квитка. Так само він має можливість вибрати спосіб оплати.

[1)] [11] Ременяк Л.В. Методичні вказівки для виконання лабораторних робіт по дисципліні «Управління ІТ-проектами». Одеса: ОДЕКУ, 2016. 53 с.

[2)] [10] Ременяк Л.В. Методичні вказівки для виконання лабораторних робіт по дисципліні «Проектування інформаційних систем». Одеса: ОДЕКУ, 2016. 24 с.

Персонал займається продажем квитків, складанням графіка, ремонтом поїздів, а також бере участь у перевезенні пасажирів.

Поїзд може прибути або на вокзал, або на ремонт, а основним варіантом використання для поїзда в цій системі є перевезення пасажирів.

Вокзал займається прийманням на роботу, розкладом потягів, формуванням цін на послуги і маршрутизацією поїздів. Фінансується вокзал коштом грошей отриманих з продажу квитків і послуг.

Для дійових осіб так само можна створити діаграму варіантів використання. Наприклад, створимо діаграму варіантів використання конкретно для дійової особи «вокзал» (рис. Г.2).

Вокзал становить список маршрутів який використовується для формування списку квитків. Так само список квитків міститьв собі такі варіанти використання як пошук квитків в списку, включення і видалення квитка зі списку, а також список поїздів. Список поїздів вже включає в себе такі варіанти використання як пошук поїздів, включення і виключення поїзда зі списку. Крім квитків вокзал може надавати і інші послуги, такі як наприклад транспортування автомобіля або якого-небудь багажу. Для цього Списку послуг необхідні такі параметри як назва послуги, тип послуги і термін надання. Іноді послуги можуть надаватися безкоштовно тому вартість пов'язана зі списком послуг відношенням розширення.

2.7 Побудова діаграми класів

Будуючи діаграму класів для нашої системи обмежимося одним актором. Інші діаграми будуються схожим чином. Діаграма класів об'єктів для актора «Вокзал» приведена в додатку Д (рис. Д.1). На цій діаграмі знаходиться інтерфейс Terminal, що відповідає актору, він взаємодіє з абстрактними класами TrainList, RouteList, TicketList, а також ListOfServices.

Абстрактний клас TrainList має атрибут який визначає кількість потягів та три методи за допомогою яких можна проводити пошук, додавання чи ви-

далення потягу зі списку. Його наслідує клас Train, який має такі атрибути як номер, тип, швидкість та рік випуску. Клас Train володіє методами які дозволяють: заправити потяг, перемістити до депо чи терміналу, і два методи які дозволяють формувати склад потягу.

Абстрактний клас TicketList містить атрибут який визначає кількість квитків, а також методи їх перегляду, додавання, чи видалення. Цей абстрактний клас наслідує клас Ticket який має такі атрибути: номер квитка, маршрут, час відправки, ціну. Методи цього класу дозволяють додавати інформацію до квитка. Також з цим класом асоційован інтерфейс RoutList, у якому є два методи котрі відповідають за місце посадки та висадки пасажиру.

Абстрактний клас ListOfServices містить атрибут який визначає кількість послуг, а також методи їх перегляду, додавання, чи видалення. Цей абстрактний клас наслідує клас Service який має такі атрибути: назва послуги, тип послуги, строк надання, ціна та знижка. Методи цього класу дозволяють перевіряти ціну чи знижку.

Отже, у цьому розділі були на практиці використані інструменти бізнес-аналізу, описані в попередньому розділі, для проведення бізнес-аналізу підприємства «Залізничий вокзал».

3 РОЗРОБКА WEB-ДОДАТКУ ПІДПРИЄМСТВА

3.1 Вступ

Основою розроблювальної інформаційної системи підприємства «Залізничний вокзал» являється база даних, оскільки її архітектура надає можливість зберігати інформацію в такому вигляді, в якому вона буде найбільш сприйнятлива та зручна для виконання основних функцій бізнес-аналізу, а саме формування квитка пасажира на основі таких таблиць як список паса-

жирів, список маршрутів якими подорожують ці пасажири, список вокзалів та інша інформація яка використовується у формуванні квитка [12][1)].

Веб-додаток був створений мовою препроцесора гіпертексту PHP, за допомогою Apache HTTP-серверу [13][2)]. Генерація та взаємодія з базою даних була виконана за допомогою веб-додатку phpMyAdmin [14][3)].

Інтерфейс веб-додатка був налаштований за допомогою CSS (Cascading Style Sheets — каскадні таблиці стилів) та мови сценаріїв java-script. У якості набору інструментів використовувався фреймворк Bootstrap.

3.2 Реалізація БД за допомогою phpMyAdmin

База даних – це набір логічно зв'язаних даних (і опис цих даних), який спільно використовується, і призначений для задоволення інформаційних потреб організації[15].

Реляційні СКБД використовують реляційну модель даних, за якою структура даних представляється у вигляді прямокутних таблиць, що складаються зі стовпців і рядків. Прямокутні таблиці реляційної бази даних називаються відношеннями. Відношення (таблиця) описує деякий об'єкт реального миру або взаємозв'язок між об'єктами [15][4)].

MySQL – вільна система керування реляційними базами даних. Вона використовується, в першу чергу, для створення динамічних веб-сторінок, оскільки має чудову підтримку з боку різноманітних мов програмування.

phpMyAdmin – веб-додаток з відкритим кодом, написаний на мові PHP, який являє собою веб-інтерфейс для адміністрування СКБД MySQL. PHPMyAdmin дозволяє через браузер і не тільки здійснювати

---

1) [12] Томашевський О. М., Цегелик Г. Г., Вітер М. Б., Дудук В. І. Інформаційні технології та моделювання бізнес-процесів. Навч. посіб. К.: «Видавництво «Центр учбової літератури», 2012. 296 с.

2) [13] Котеров, Д. В. PHP 7. СПб.: БХВ-Петербург, 2016. 1088 с.

3) [14] Дюбуа Поль. MySQL. М.: Издательский дом «Вильямс», 2004г. 1056с.

4) [15] Козловская В.П. Организация баз данных и знаний. Конспект лекцій. Одеса, ОДЕкУ, 2014, 125 с.

адміністрування сервера MySQL, запускати команди мовою SQL (structured query language) і переглядати вміст таблиць і баз даних. Додаток користується великою популярністю у веб-розробників, бо дозволяє управляти СКБД MySQL без безпосереднього введення SQL команд, надаючи дружній інтерфейс.

Орієнтуючись на базу даних яка була згенерована у другому розділі (рис. 2.1) створюється база даних у середовищі phpMyAdmin. Для цього потрібно у спеціальному меню (рис 3.1) вписати назву бази даних та кодування інформації. Назва бази даних цього проекту tryRailWay, а кодування utf8-general-ci.

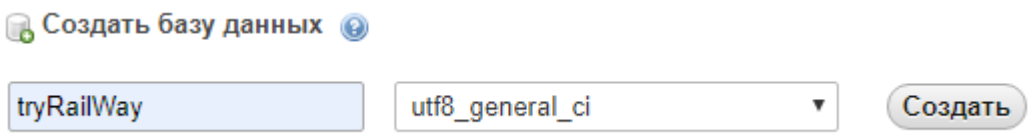


Рисунок 3.1 – Створення бази даних tryRailWay

Після цього створюється таблиці цієї бази даних. Зробити це можливо двома способами: використовуючи інтерфейс веб-додатку phpMyAdmin який автоматично формує SQL-запит в залежності від обраних нами опцій, або вручну зробивши SQL-запит який був написаний мовою SQL.

Таблиця бази даних була створена за допомогою мови SQL вручну. Спочатку створимо таблицю «pasawir», яка є найбільш простою й складається лише з трьох атрибутів: первинного ключа, ПІБ і телефона (рис. 3.2). Мовою SQL цей запит, а також всі інші, буде виглядати так, як приведено у додатку Е.

Наступна таблиця має назву «voksal» та відповідає за вокзали. Вона складається з п’яти атрибутів: первинний ключ, назва вокзалу, назва міста, телефон та кількість платформ. Зображена таблиця «voksal» на рис. 3.3.

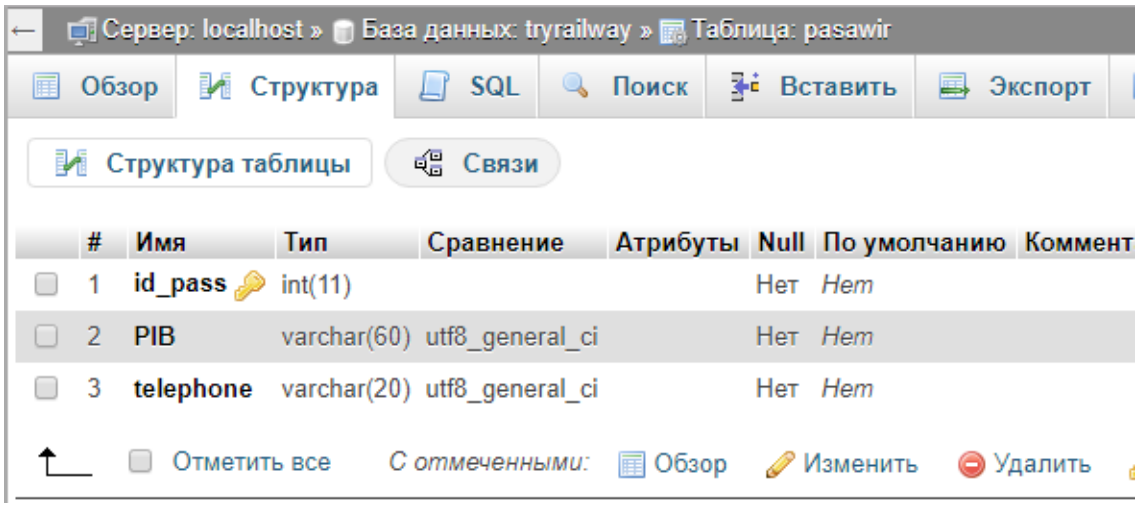


Рисунок 3.2 – Інтерфейс таблиці pasawir

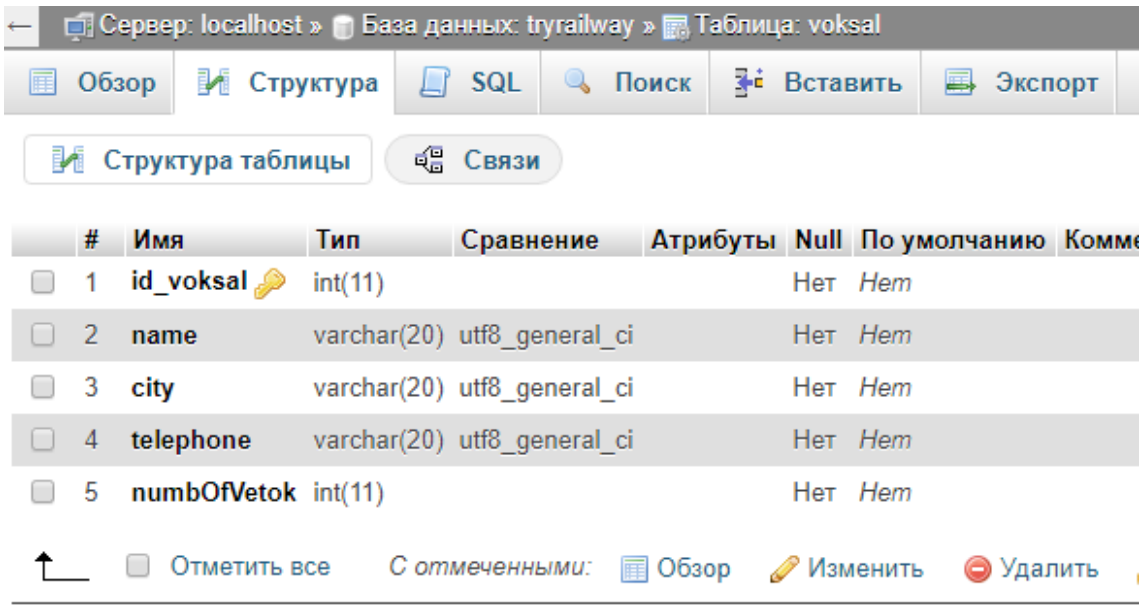


Рисунок 3.3 – Інтерфейс таблиці voksal

Таблиця графіку руху поїздів «graphic» крім первинного ключа складається також з таких атрибутів: місто початку подорожі, місто закінчення подорожі, а також дата початку подорожі і дата закінчення подорожі. У phpMyAdmin ця таблиця виглядає як на рис. 3.4.

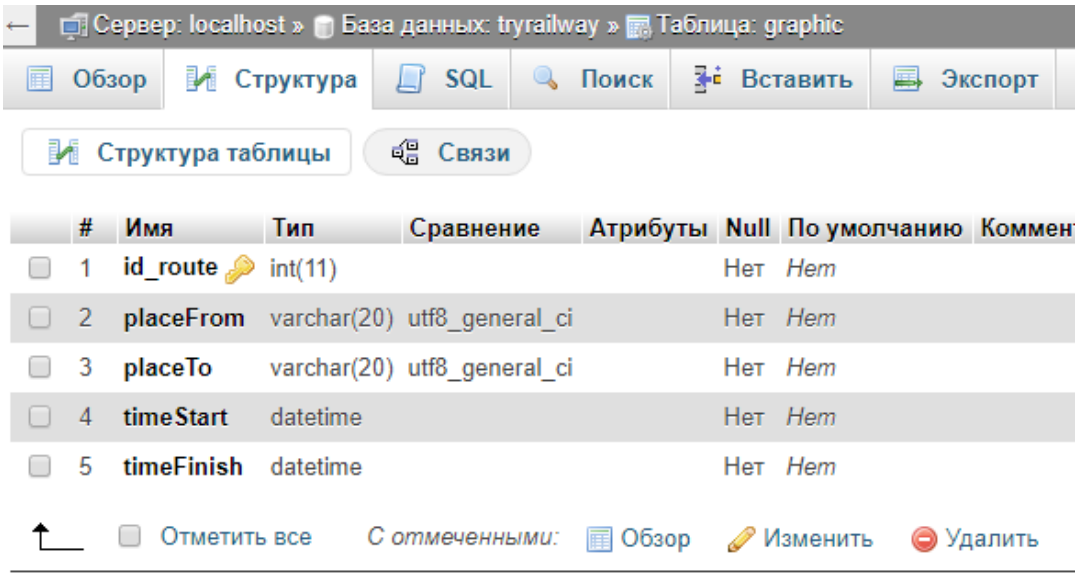


Рисунок 3.4 – Інтерфейс таблиці graphic

Потяги представленні у базі даних таблицею «train» і окрім первинного ключа мають такі атрибути як кількість вагонів, назва потягу, максимальна швидкість, рік випуску (рис. 3.5).

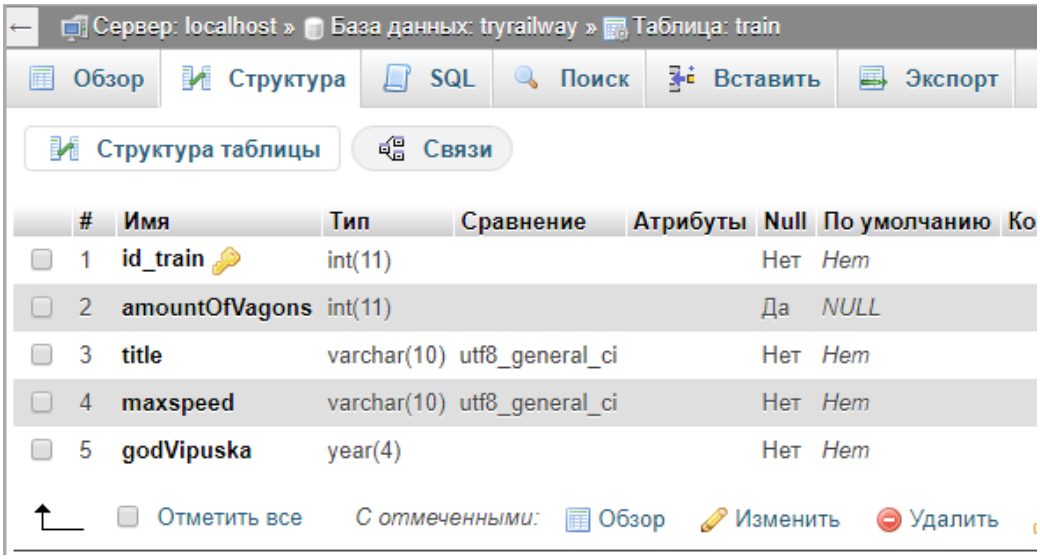


Рисунок 3.5 – Інтерфейс таблиці train

Таблиця «vagon», що описує вагони, зображена на рис. 3.6. Вона складається з первинного ключа, кількості місць у вагоні та класом вагона.

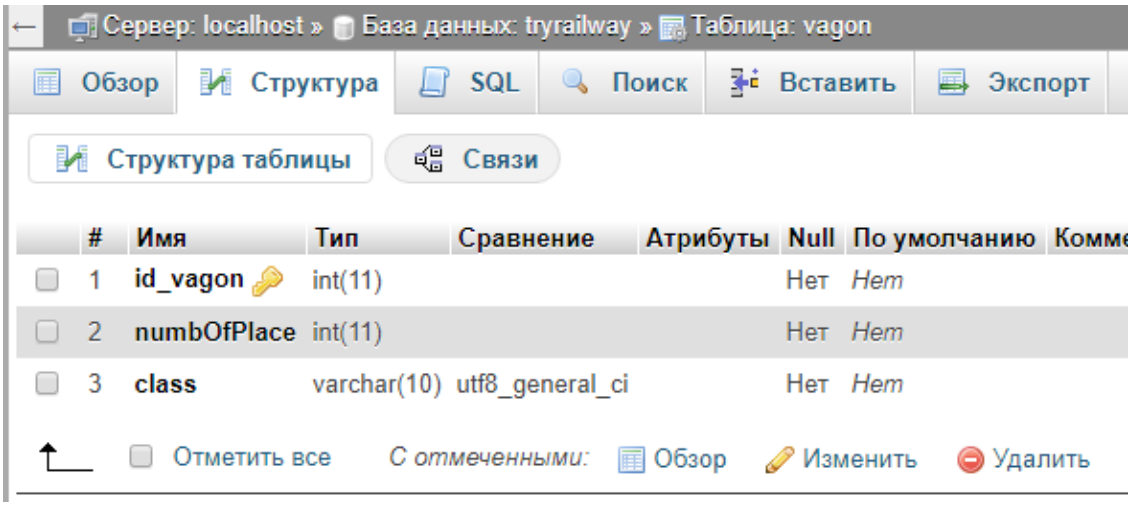


Рисунок 3.6 – Інтерфейс таблиці vagon

За персонал відповідає таблиця «personal». Ця таблиця складається з ПІБ працівника, його посади, телефону, місця праці та заробітної платні. Зображена ця таблиця на рис. 3.7.

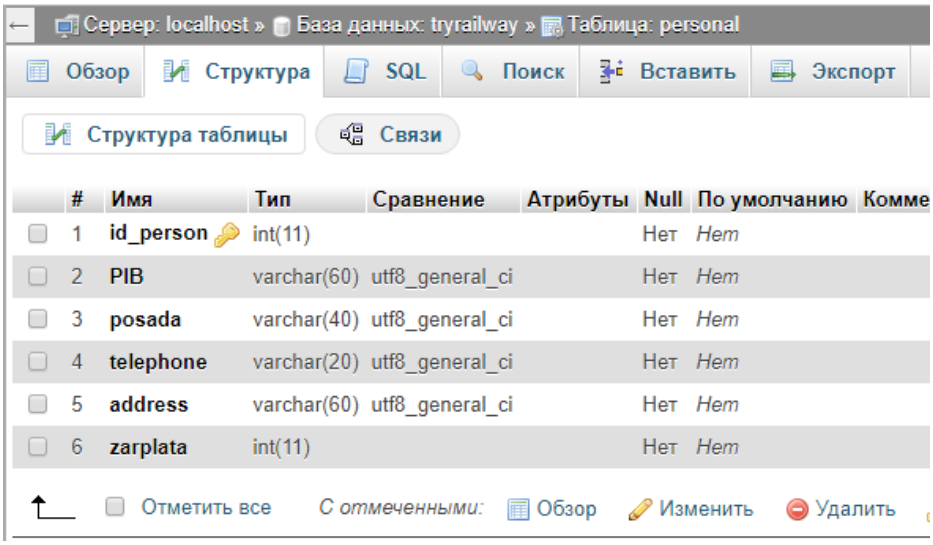


Рисунок 3.7 – Інтерфейс таблиці personal

Таблиця «usluga» в яку вносяться послуги надавані підприємством складається з назви послуги та її ціни в гривнях. Зображена ця таблиця на рис. 3.8.

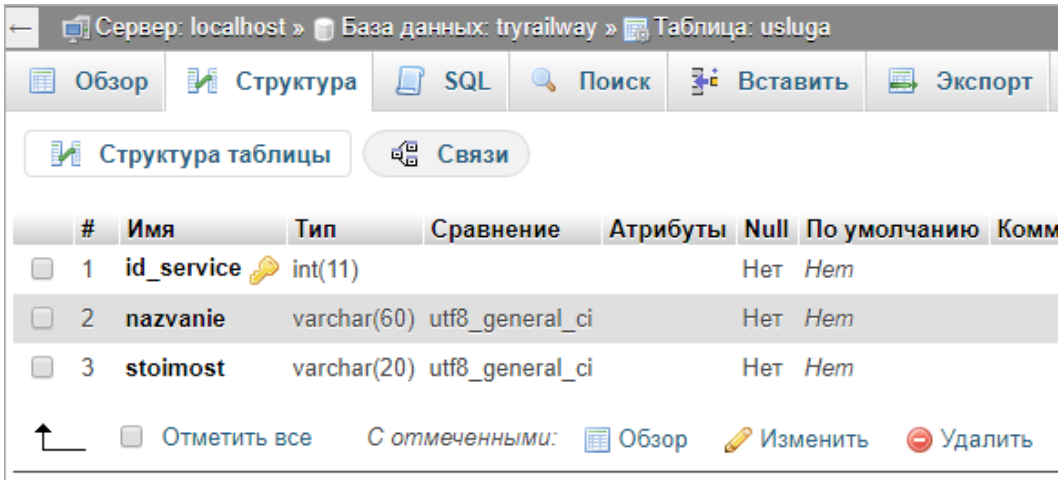


Рисунок 3.8 – Інтерфейс таблиці usluga

І нарешті остання таблиця «bilet» поєднує в собі всі інші таблиці і слугує для формування саме квитка. Окрім цього вона має такі параметри як порядковий номер вагону у поїзді та порядковий номер місця в вагоні, а також ціну квитка. Зображена ця таблиця на рис. 3.9.

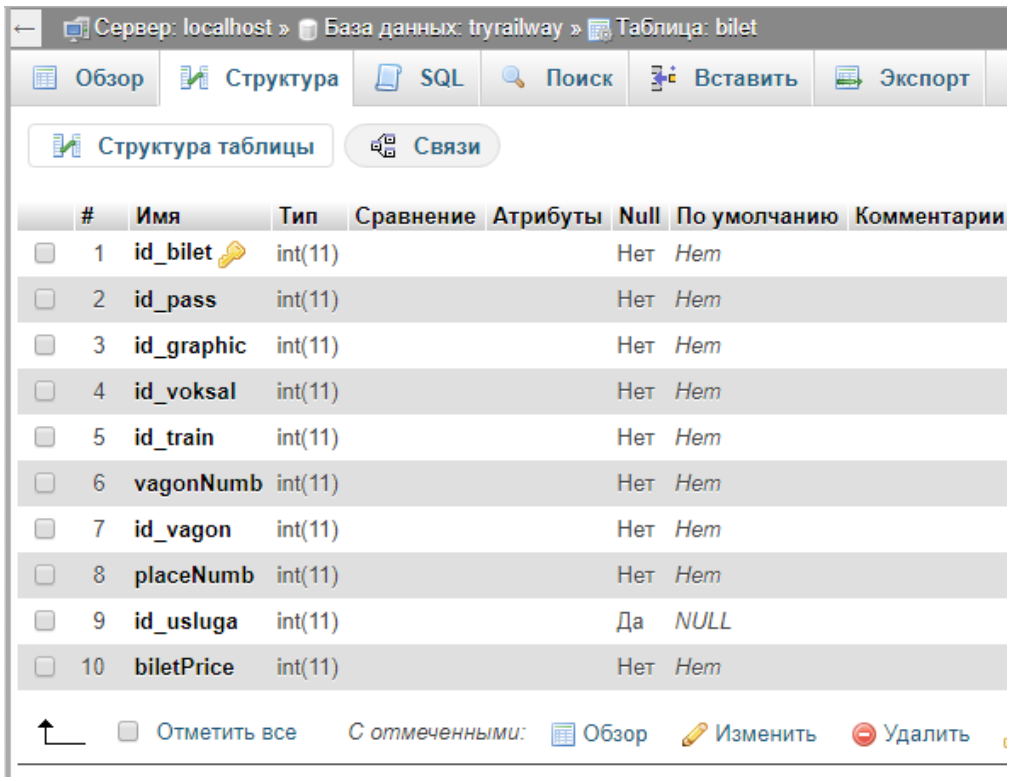


Рисунок 3.9 – Інтерфейс таблиці bilet

3.3 Розробка WEB-додатку підприємства

Для входу в систему достатньо відкрити браузер і ввести в адресний рядок адресу веб-додатку – на екрані буде відображена головна сторінка (рис. 3.10).

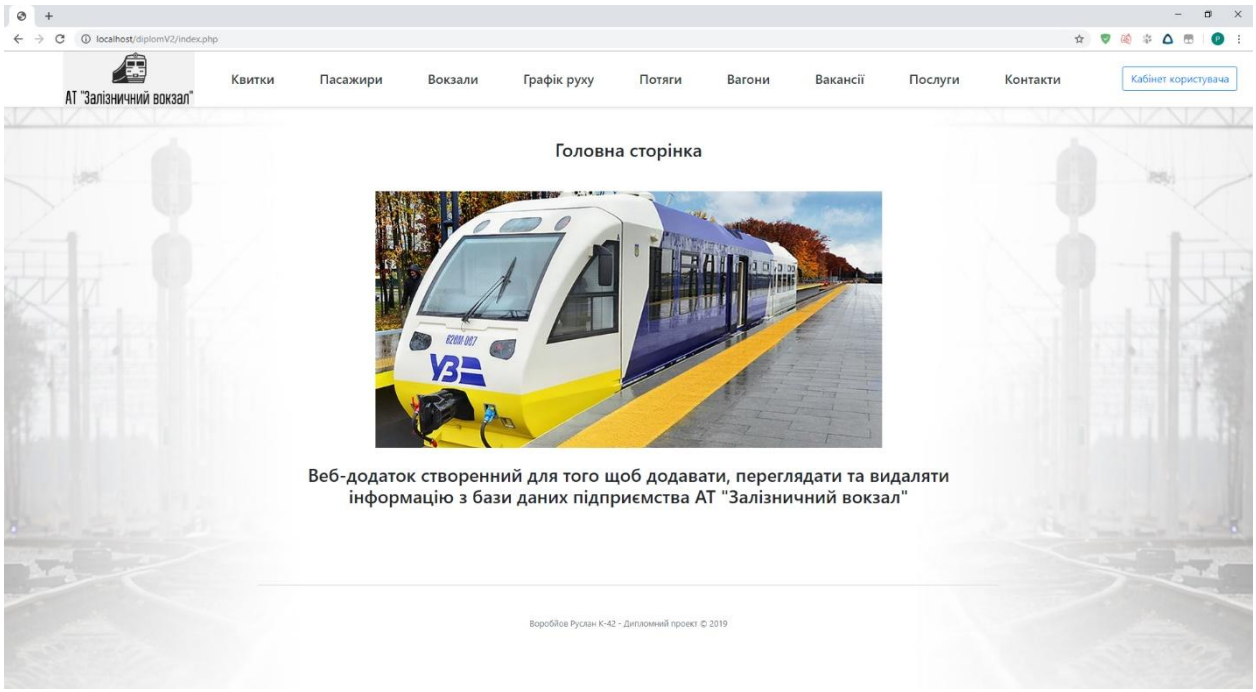


Рисунок 3.10 – Головна сторінка

Сам веб-додаток складається з трьох основних блоків: header, content і footer.

У шапці додатку (header) зліва розташована емблема підприємства, кликнувши на яку користувач потрапить на головну сторінку сайту де б він не знаходився. По середині розташовані посилання на перегляд таблиць, а також знаходиться посилання на форму у якій користувач може надіслати повідомлення адміністратору цієї сторінки вказавши свій email. Справа знаходиться емблема користувача, де він може ідентифікуватися: після натискання кнопки «Увійти» встановлюються cookie – невеликий фрагмент даних, відправлений веб-сервером і які зберігаються на комп'ютері користувача з

метою аутентифікації користувача – терміном на 2 години, що відкривають доступ до вмісту додатка.

Підвал (footer) розташований завжди знизу і в ньому знаходиться основна інформація про виконавця додатку. Обидва блоки для зручності винесені в окремий файл, і додаються в інші вкладки додатку за допомогою команди:

```
<?php    require "blocks/header(footer).php" ?>
```

Основна інформація розташовується у блоці content (рис. 3.11).

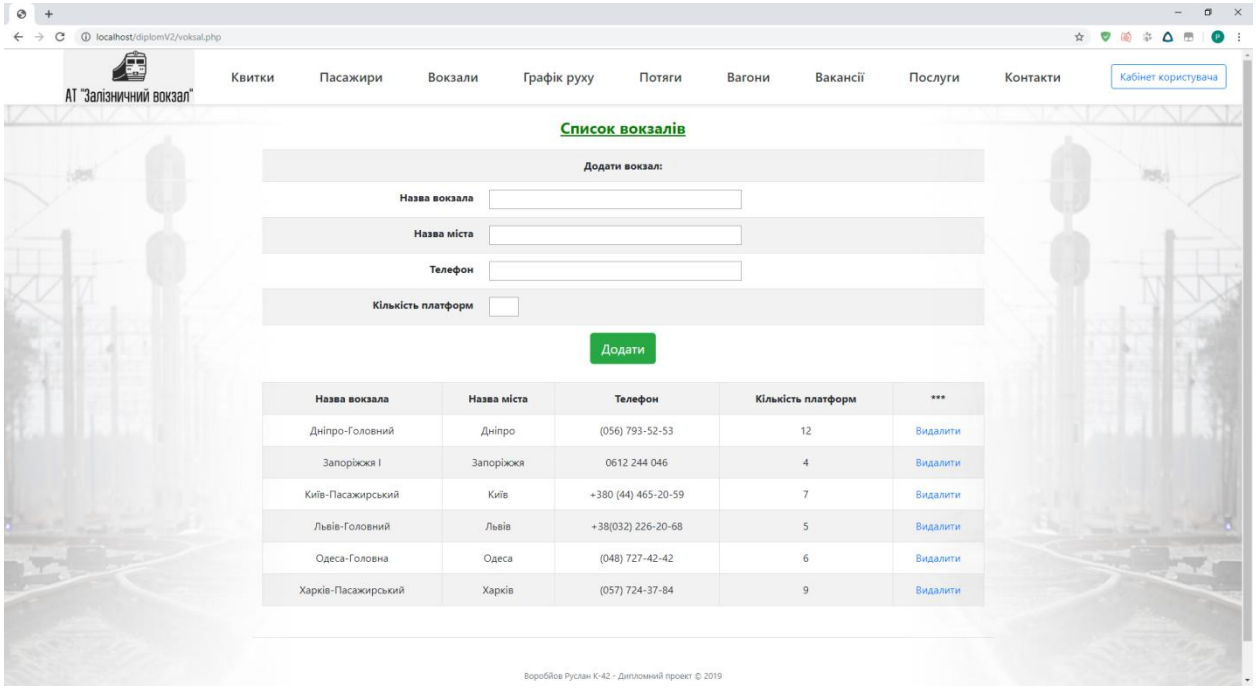


Рисунок 3.11 – Інтерфейс сторінки «Вокзали»

Перш за все кожна сторінка містить інформацію потрібну браузеру чи пошуковій системі. Це такі теги як <!DOCTYPE html> – який призначений для визначення типу поточного документа. Далі слідують <meta> теги які визначають способ кодування зовнішнього CSS-файлу, адаптацію додатка до моніторів різної довжини та справну працю в деяких браузерах, наприклад у браузеру Microsoft Edge.

Далі за допомогою тега <link> вказується шлях до файлів, що відповідають за роботу скриптів і зовнішній вигляд сторінки. Він може бути як до

віддаленого сервера так і до локального файлу. У цьому ДП використовується framework bootstrap версії 4.3.1, скачаний з офіційного сайту https://getbootstrap.com у якості набору інструментів для завдання зовнішнього виду елементів frame-вікна, відображення таблиць, а також для роботи деяких скриптів. Також слід зазначити що у деяких випадках використовувалась бібліотека jQuery, яка фокусується на взаємодії javascript та html.

Далі відбувається з'єднання з базою даних котра була згенерована у попередньому пункту. Щоб з'єднатися з базою даних потрібно написати наступний код:

```
<?php
$db = new mysqli( 'localhost', 'try1', 'try1',
'tryrailway');
if(mysqli_connect_error ()){
die ('Помилка підключення('.mysqli_connect_errno(). ')'
        . mysqli_connect_error ()); }  ?>
```

У цьому коді створюється об'єкт класу mysqli (MySQL Improved), у першому параметрі якого вказується хост. Через те, що запуск відбувається за допомогою локального сервера Apache, то вказується localhost. Далі вказується ім'я користувача MySQL, а також його пароль. Після чого вказується ім'я бази даних з якою потрібно працювати. Також в деяких випадках вказується номер порту з’єднання. Через те, що з’єднання відбувається майже в усіх сторінках, цей код було винесено в окремий файл з назвою connect.php.

Після з'єднання з базою даних створюється форма введення інформації для додавання записів в базу даних (рис. 3.12). В якості програми або документа, який обробляє дані форми виступає сама сторінка. Для відсилання даних був обраний метод запиту post. Метод post посилає на сервер дані в запиті браузера. Це дозволяє відправляти більшу кількість даних, ніж є методом get, оскільки у нього встановлено обмеження в 4 Кб. Великі обсяги даних використовуються у форумах, поштових службах, заповненні бази даних, при пересиланні файлів і ін.

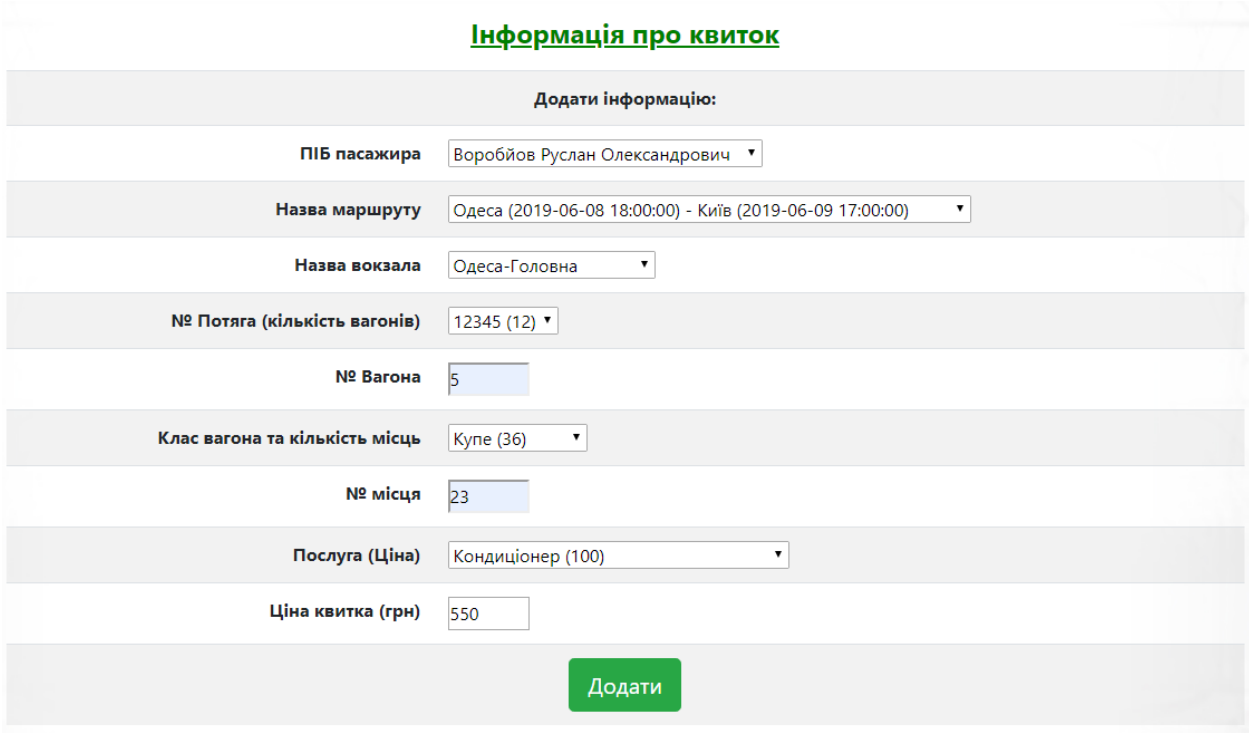


Рисунок 3.12 – Додавання інформації до БД

Після введення даних у форму і натискання кнопки "Додати" відбувається наступне. Спочатку виходять дані з форми й присвоюються змінним. Потім кілька змінних перевіряється на наявність у цих змінних даних. Далі відбувається з'єднання з базою даних і перевіряється відповідність введеної інформації з тієї що є в базі даних. Якщо така інформація вже присутня то блок додавання завершується і виводиться повідомлення про те що такий запис вже присутній в таблиці. Якщо ж кількість знайдених записів дорівнює нулю, то відбувається додавання нового запису з параметрами, які були введені у форму до таблиці. Після завершення операції сторінка перезавантажується і виводиться напис про те що запис було успішно додано.

Після блоку додавання інформації йде блок видалення інформації, він реалізований досить просто. Отримується змінна $del, що містить первинний ключ обраного для видалення запису, і відправляється запит до таблиці з метою видалити запис, який містить цей первинний ключ.

І останній блок тіла сторінки відповідає за відображення інформації з таблиці на екран сторінки веб-додатка (рис. 3.13). Спочатку йде звернення до певної таблиці БД за допомогою функції query (). У змінну $contactnum записується кількість рядків таблиці, після чого витягується асоціативний масив з даними кожного запису. Ці дані присвоюються локальним змінним, після чого створюється таблиця html в якій спочатку створюється шапка, потім додаються дані з локальних змінних по рядках.

Окремо варто згадати про таблицю "bilet", яка об'єднує в собі інші таблиці БД. Для відображення інформації тут формуються запити SELECT для всіх інших таблиць, з яких необхідно отримати будь-які дані вказавши id запису. Для цього треба, щоб id запису обраної таблиці у таблиці «bilet» сходився з id записом у необхідній таблиці.

ПІБ пасажира	Маршрут	Назва вокзала	№ потяга (кількість вагонів)	№ вагону	Клас вагону (місць)	№ місця	Послуга (ціна)	Ціна квитка (грн)	***
Васильчук Євгенія Йосиповна	Дніпро (2019-06-08 17:59:00) - Харків (2019-06-09 03:45:00)	Дніпро-Головний	12345 (12)	12	Плацкарт (54)	34	Кондиціонер (100)	350	Видалити
Шевчук Алла Петровна	Харків (2019-06-09 09:00:00) - Львів (2019-06-10 17:00:00)	Харків-Пасажирський	12345 (12)	11	Люкс (18)	11	Кондиціонер (100)	750	Видалити
Таращук Євгенія Івановна	Київ (2019-06-11 09:45:00) - Запоріжжя (2019-06-11 22:55:00)	Київ-Пасажирський	122 (22)	1	Люкс (18)	15	кондиціонер та спальна билизна (200)	680	Видалити
Воробйов Руслан Олександрович	Одеса (2019-06-08 18:00:00) - Київ (2019-06-09 17:00:00)	Одеса-Головна	12345 (12)	5	Купе (36)	23	Кондиціонер (100)	550	Видалити

Рисунок 3.13 – Відображення інформації вилучиної з БД

У цьому розділі було розглянуто побудову БД MySQL за допомогою веб-додатка phpMyAdmin, а також був розроблений веб-додаток, що дозволяє додавати, видаляти й переглядати інформацію БД tryRailWay умовного підприємства АТ "Залізничний вокзал". Код веб-додатку наведений у додатку Є.

ВИСНОВКИ

В даній дипломній роботі були проаналізовані різні інструменти бізнес-аналізу, створена концептуальна модель інформаційної системи підприємства, названі основні атрибути системи та їх зв’язки. На базі цього була згенерована база даних підприємства мовою MySQL.

Була створена функціональна модель підприємства, що показує, які бізнес-процеси розташовані в системі, і як вони взаємодіють друг з другом.

Так само було проаналізовано поведінка цих атрибутів у часі й порядок виконання. Побудована графічна модель з використанням мови UML, а також класова модель, що відображає структуру сукупності взаємозалежних класів об’єктів.

На базі класової моделі було розроблено веб-додаток підприємства АТ«Залізничий вокзал», який містить в собі всі ті дані в системі які були створені у другому розділі і реалізує логіку їх роботи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бизнес-анализ. URL : https://ru.wikipedia.org/wiki/Бизнес-анализ (дата звернення 25.04.2019)
2. Rational Rose. URL : http://kpms.ru/Automatization/Rational_Rose.htm. (дата звернення 26.04.2019)
3. Пасічник В.В., Литвин В.В., Шаховська Н.Б. Проектування інформаційних систем. Навчальний посібник. Львів, 2013. 380 с.
4. Маклаков С.С. BPwin и Erwin. CASE- средства разработки информационных систем. М.:ДИАЛОГ-МИФИ, 2000. 256с.
5. Ременяк Л.В. Управління ІТ проектами. Конспект лекцій. Одеса.: ОДЕКУ, 2015, 168 с.
6. Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. М: «Финансы и статистика», 1998. 123 с.
7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. М.: Финансы и статистика, 2006. 544 с.
8. Грекул В. И., Денищенко Г. Н., Коровкина Н. Л. Проектирование информационных систем. М. : Интернет-Ун-т Информ технологий, 2005. 304 с.
9. Ременяк Л.В. Проектування інформаційних систем. Конспект лекцій. Одеса.: ОДЕКУ, 2016, 152 с.
10. Ременяк Л.В. Методичні вказівки для виконання лабораторних робіт по дисципліні «Проектування інформаційних систем». Одеса: ОДЕКУ, 2016. 24 с.
11. Ременяк Л.В. Методичні вказівки для виконання лабораторних робіт по дисципліні «Управління ІТ-проектами». Одеса: ОДЕКУ, 2016. 53 с.

12. Томашевський О. М., Цегелик Г. Г., Вітер М. Б., Дудук В. І. Інформаційні технології та моделювання бізнес-процесів. Навч. посіб. К.: «Видавництво «Центр учбової літератури», 2012. 296 с.
13. Котеров, Д. В. PHP 7. СПб.: БХВ-Петербург, 2016. 1088 с.
14. Дюбуа Поль. MySQL. М.: Издательский дом «Вильямс», 2004г. 1056с.
15. Козловская В.П. Организация баз данных и знаний. Конспект лекцій. Одеса, ОДЕкУ, 2014, 125 с.

ДОДАТОК А
Логічна та фізична схема ІС «Залізничний вокзал»

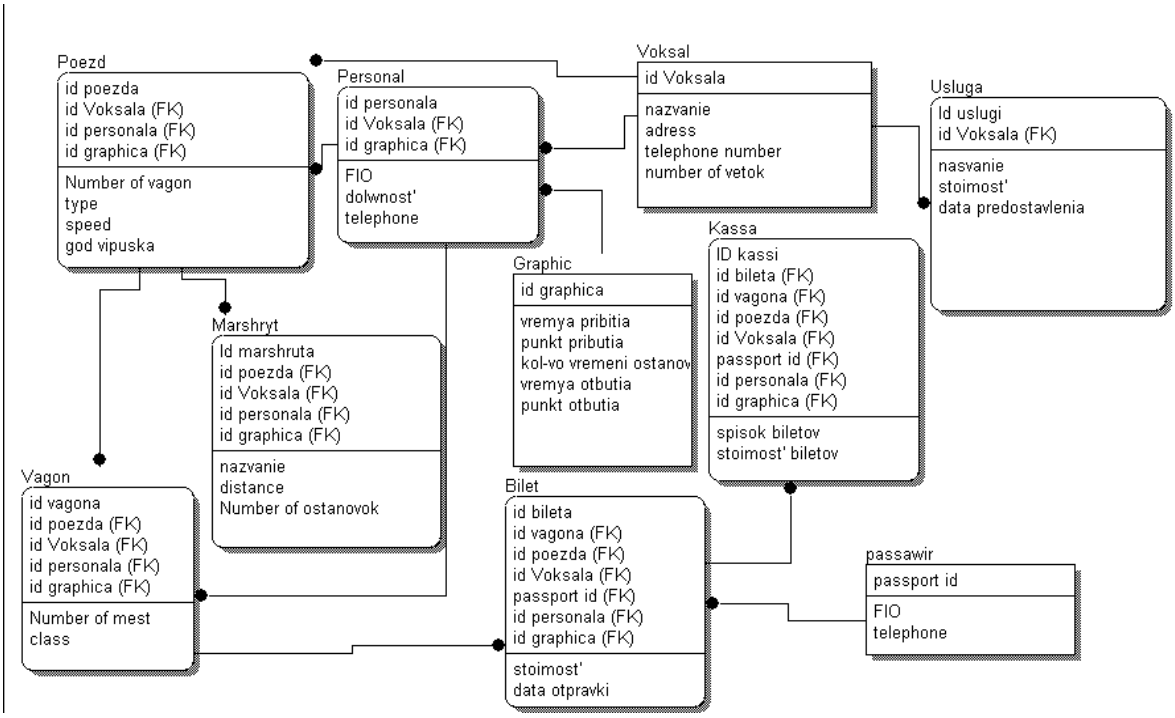


Рисунок А.1 – Логічна схема ІС «Залізничний вокзал»

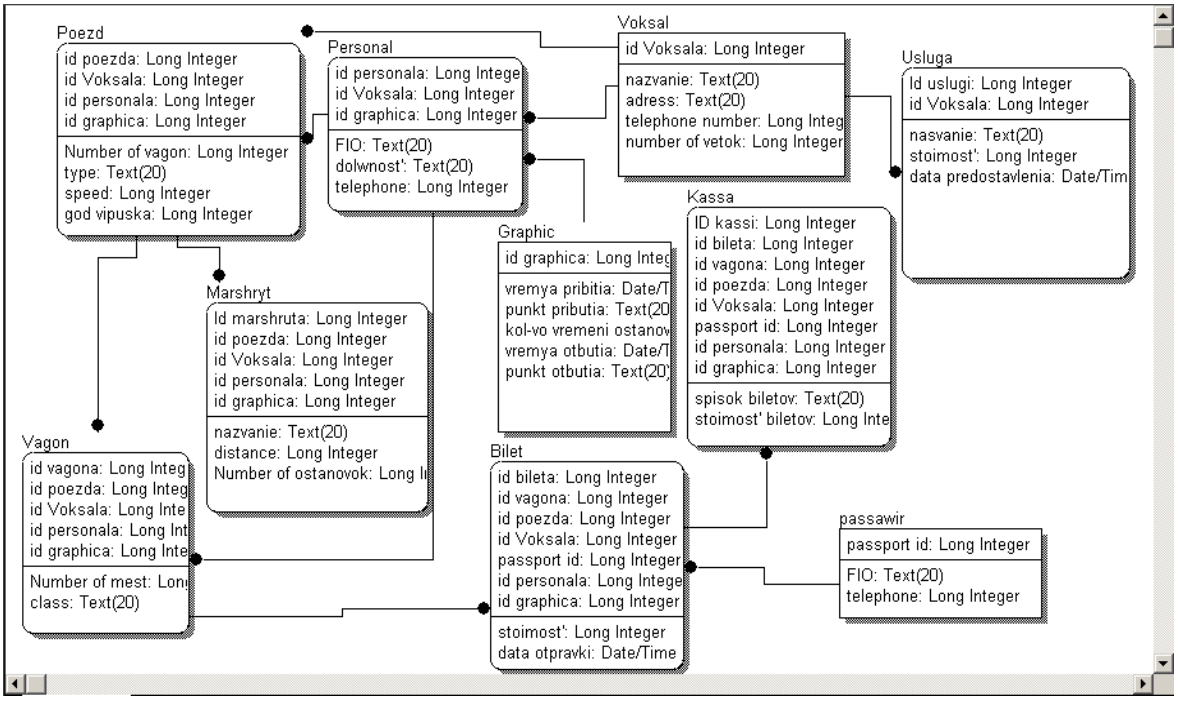


Рисунок А.2 – Фізична схема ІС «Залізничний вокзал»

ДОДАТОК Б

Діаграми функціональної моделі IDEF0

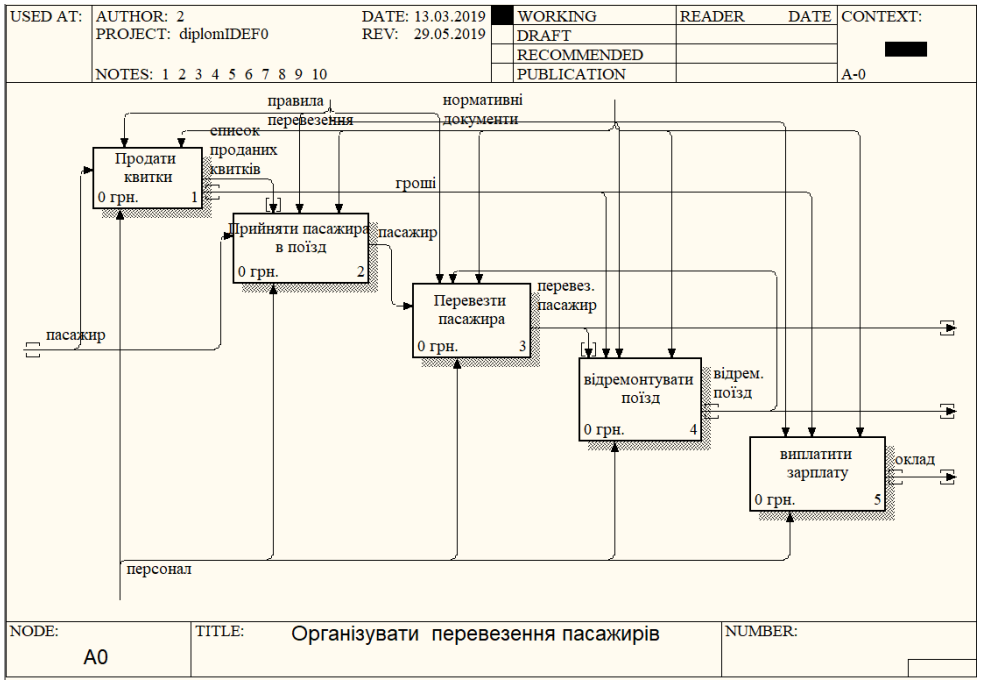


Рисунок Б.1 – Діаграма декомпозиції першого рівня

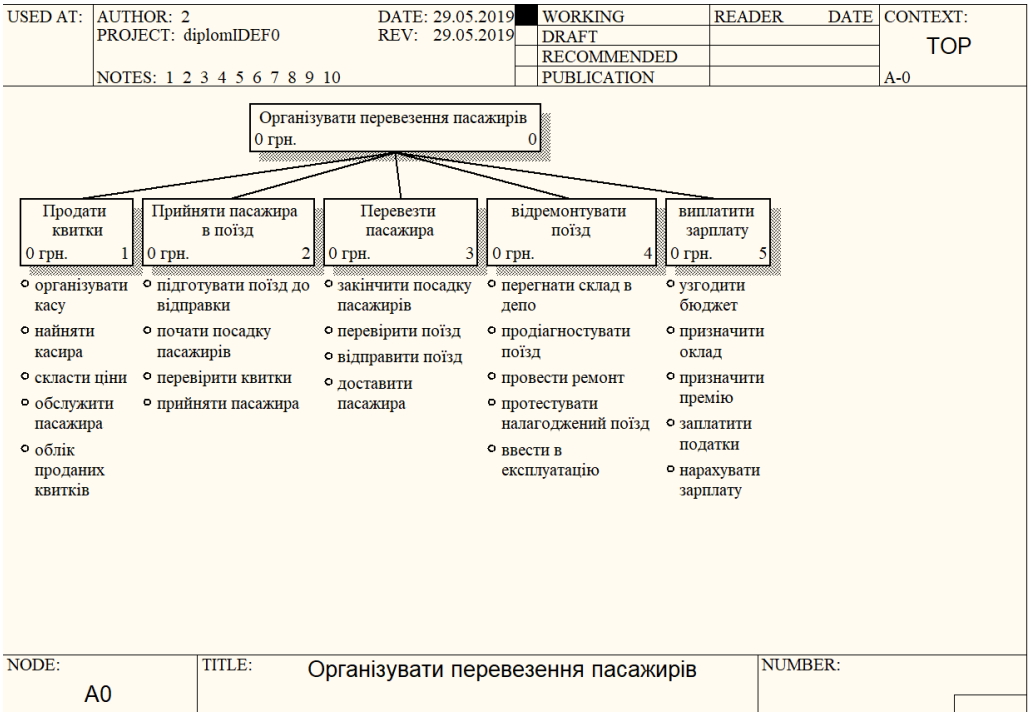


Рисунок Б.2 – Диаграма дерева вузлів

ДОДАТОК В

Діаграми потоків даних (DFD)

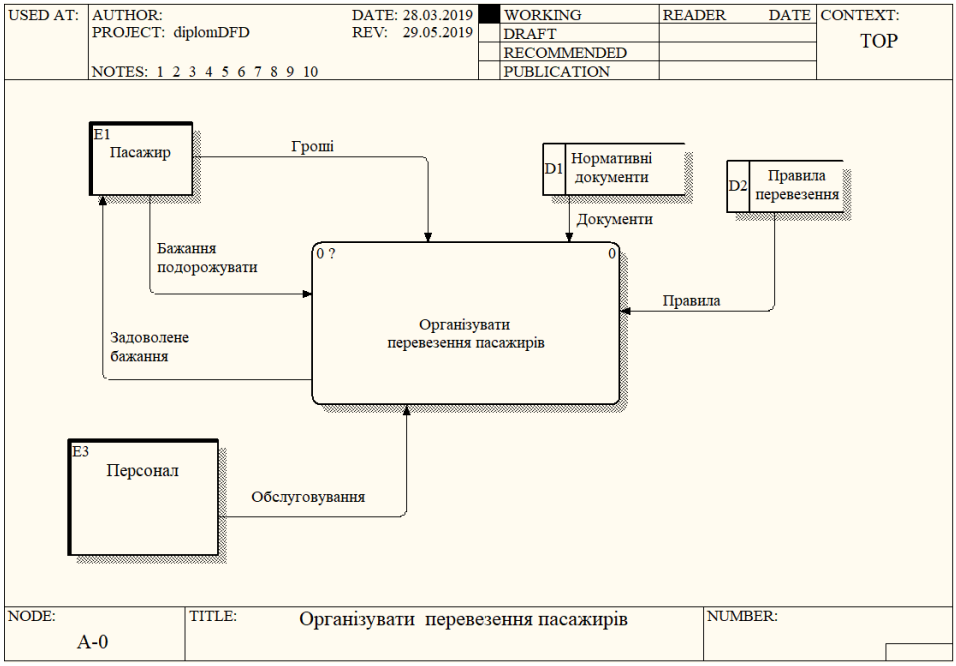


Рисунок В.1 – Контекстна діаграма DFD

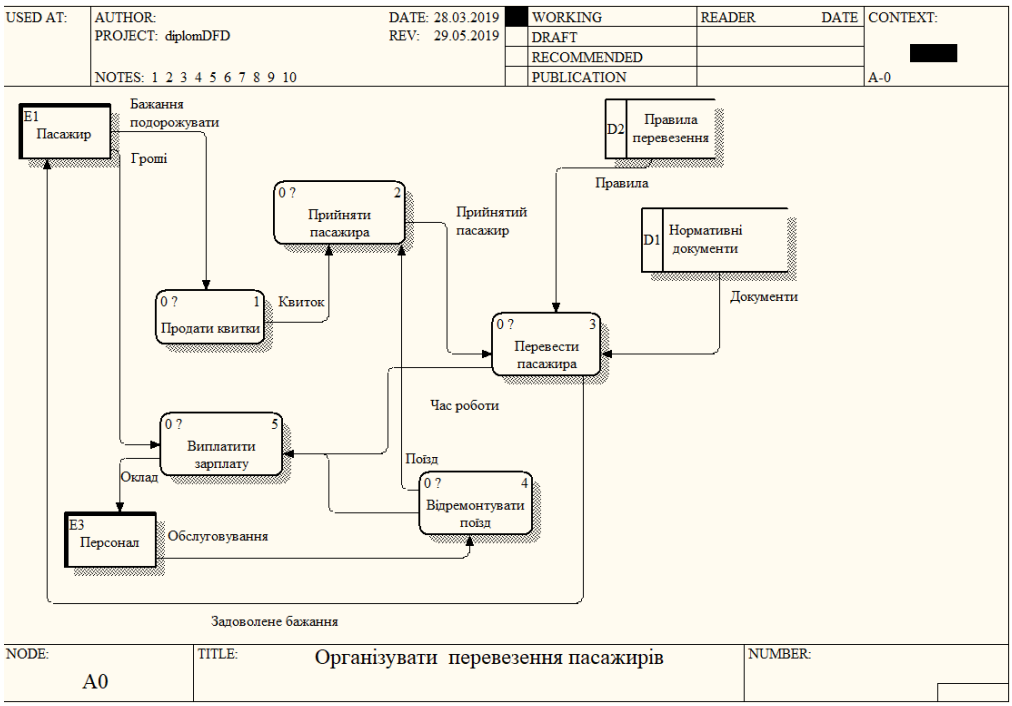


Рисунок В.2 – Діаграма декомпозиції першого рівня (DFD)

ДОДАТОК Г

Діаграми варіантів використання

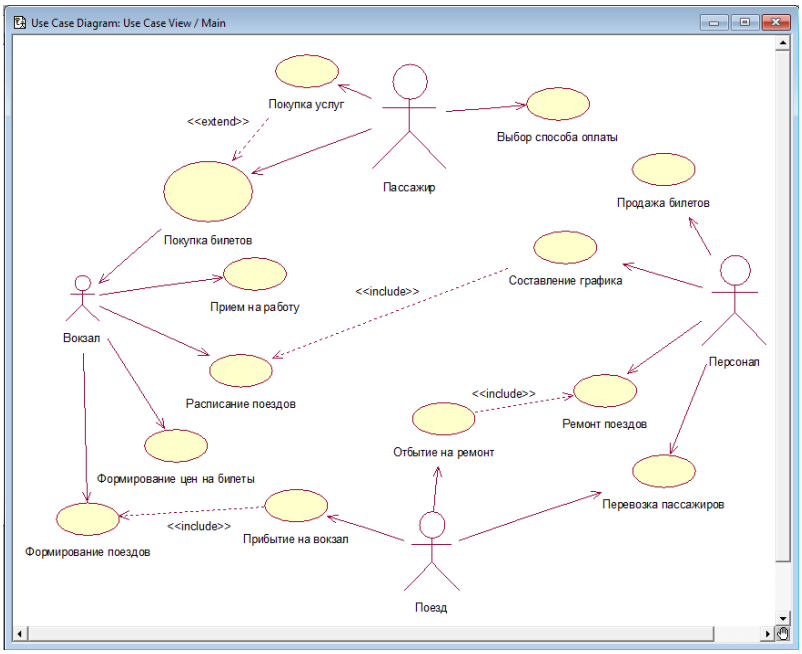


Рисунок Г.1 – Діаграма варіантів використання «Залізничний вокзал»(main)

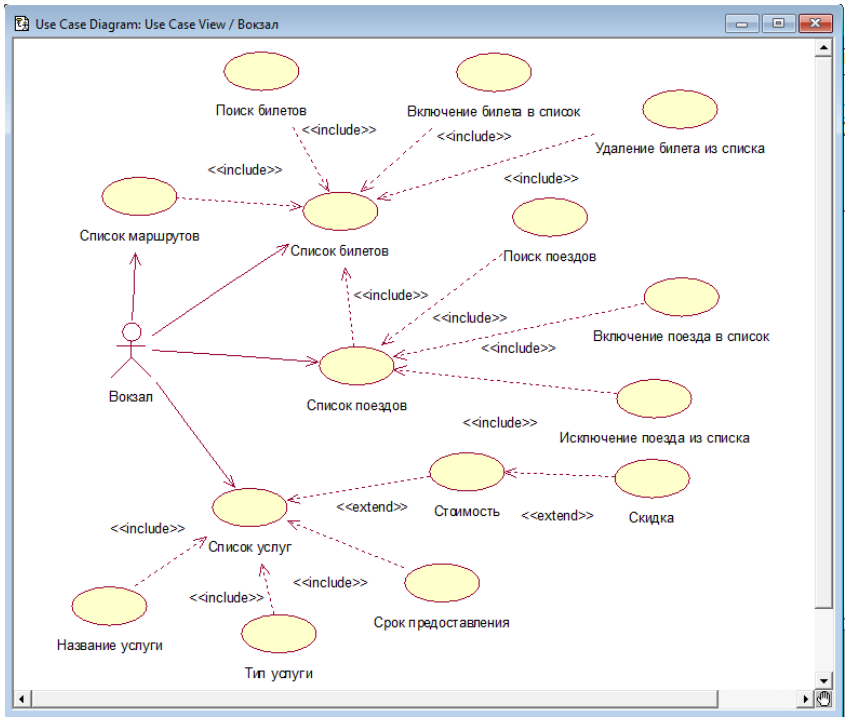


Рисунок Г.2 – Діаграма варіантів використання «Вокзал»

ДОДАТОК Д

Діаграма класів

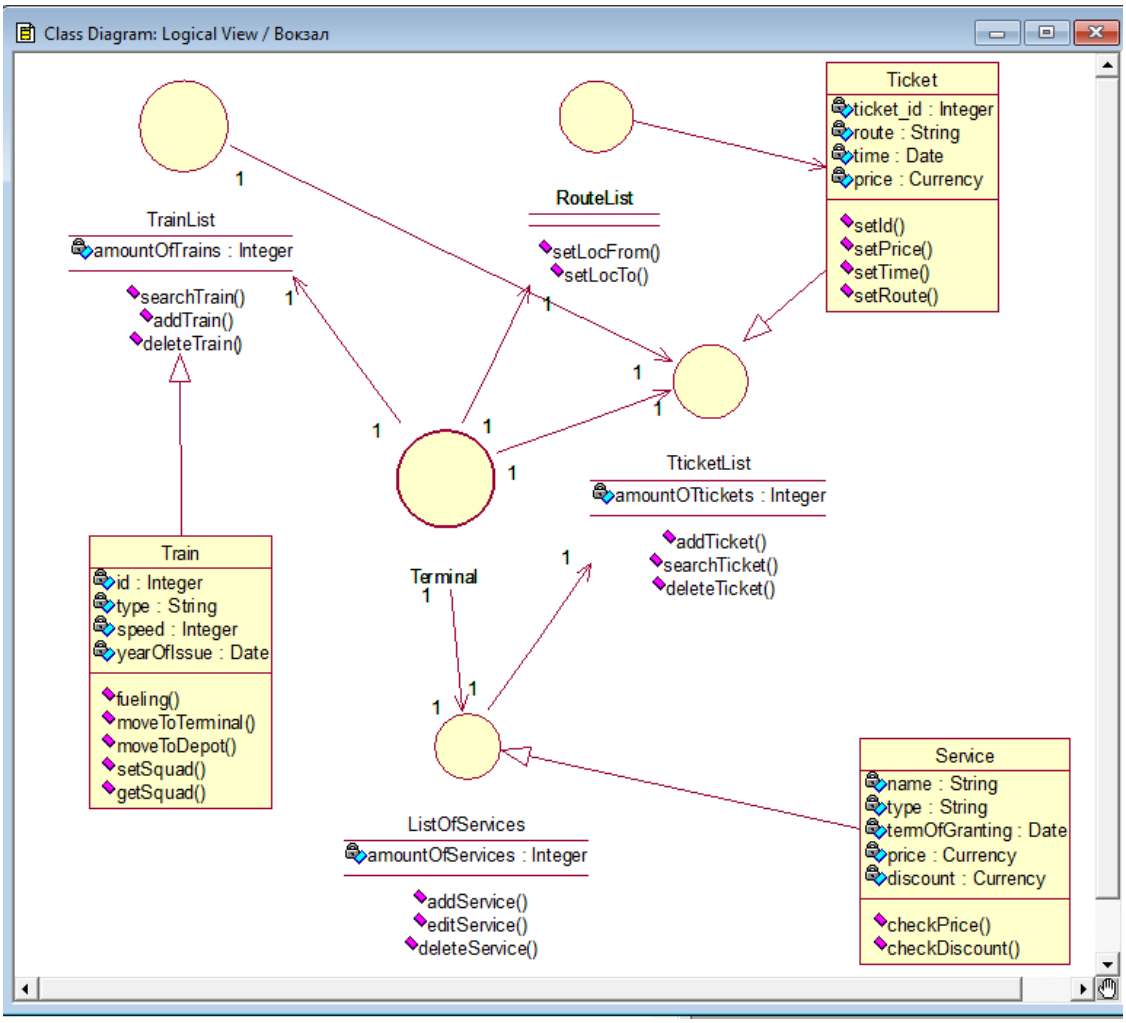


Рисунок Д.1 – Діаграма класів «Вокзал»

ДОДАТОК Е

SQL-запит для створювання таблиць бази даних «tryRailWay»

```
use tryRailWay;
create table pasawir(
id_pass integer not null auto_increment,
PIB varchar(60) not null,
telephone varchar(20) not null,
primary key(id_pass));

create table voksal (
id_voksal integer not null auto_increment,
name varchar(20) not null,
city varchar(20) not null,
telephone varchar(20) not null,
numbOfVetok integer not null,
primary key(id_voksal));

create table graphic (
id_route integer not null auto_increment,
placeFrom varchar(20) not null,
placeTo varchar(20) not null,
timeStart datetime not null,
timeFinish datetime not null,
primary key(id_route));

create table train (
id_train integer not null auto_increment,
amountOfVagons integer,
title varchar (10) not null,
maxspeed varchar(10) not null,
godVipuska year not null,
primary key(id_train));

create table Vagon (
id_vagon integer not null auto_increment,
numbOfPlace integer not null,
class varchar (10) not null,
primary key(id_vagon));

create table personal (
id_person integer not null auto_increment,
PIB varchar(60) not null,
posada varchar (40) not null,
telephone varchar(20) not null,
address varchar(60) not null,
```

```
primary key(id_person));

create table usluga (
id_service integer not null auto_increment,
nazvanie varchar(60) not null,
stoimost varchar (20) not null,
primary key(id_service));

create table bilet (
id_bilet integer not null auto_increment,
id_pass integer not null,
id_graphic integer not null,
id_voksal integer not null,
id_train integer not null,
vagonNumb integer not null,
id_vagon integer not null,
placeNumb integer not null,
id_usluga integer,
biletPrice integer not null,
primary key(id_bilet));
```

ДОДАТОК Ж

Лістинг коду WEB-додатку підприємства

Ж.1 Лістинг коду index.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width, initial-cale=1.0">
	<meta http-equiv="X-UA-Compatible" content="ie=edge">
	<link rel="stylesheet" href="css/style.css">
	<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.cs
s">
	<title>АТ "Залізничний Вокзал"</title>
</head>
<body class ="bg">
	<?php   require "blocks/header.php" ?>
	<div class="container mt-5 " >
		<h3 class="mb-5 text-center">Головна сторінка</h3>
		<img src="pic/train.jpg" class="img-rounded mx-auto d-block">
	<BR><h3 class="mb-5 text-center">Веб-додаток створенний для того щоб
додавати, переглядати та видаляти інформацію з бази даних підприємства АТ
"Залізничний вокзал"</h3><BR>
    </div>
<?php   require "blocks/footer.php" ?>
</body>
</html>
```

Ж.2 Лістинг коду connect.php

```
<?php
	$db = new mysqli( 'localhost', 'try1', 'try1', 'tryrailway');
	if(mysqli_connect_error ()){
	die   (   'Помилка   підключення   ('.   mysqli_connect_errno   ().   ')'
. mysqli_connect_error ()); } ?>
```

Ж.3 Лістинг коду header.php

```
<div class=" d-flex flex-column flex-md-row align-items-center p-0  px-md-4
mb-3 bg-white border-bottom shadow-sm">
<a href="index.php"><img src="pic/logo3.jpg" width="75%"
```

```
   height="75%" alt="Пример" class="rounded float-right"></a>
  <h4 class="my-0 mr-md-auto  headerFont">
  <nav class="my-2 my-md-0 mr-md-3 ml-5">
    <a class="p-2 ml-0 text-dark" href="bilet.php">Квитки</a>
    <a class="p-2 ml-5 text-dark" href="pasawir.php">Пасажири</a>
    <a class="p-2 ml-5 text-dark" href="voksal.php">Вокзали</a>
    <a class="p-2 ml-5 text-dark" href="route.php">Графік руху</a>
    <a class="p-2 ml-5 text-dark" href="train.php">Потяги</a>
    <a class="p-2 ml-5 text-dark" href="vagon.php">Вагони</a>
    <a class="p-2 ml-5 text-dark" href="personal.php">Вакансії</a>
    <a class="p-2 ml-5 text-dark" href="usluga.php">Послуги</a>
    <a class="p-2 ml-5 text-dark" href="about.php">Контакти</a>
  </h4>
</nav>
 <?php
    if(isset($_COOKIE['user']) == 'True'): {
      ?>
  <a class="btn btn-outline-primary" href="login.php">Кабінет користувача</a>
  <a class="btn btn-outline-primary ml-1" href="auth.php">Вийти</a>
  <?php } else: ?>
  <a class="btn btn-outline-primary" href="login.php">Увійти</a>

  <?php endif; ?></div>
```

Ж.4 Лістинг коду footer.php

```
<footer class="container pt-4 my-md-5 pt-md-5 border-top ">
  <div class="row">
      <div class="col-12 col-md d-flex justify-content-center">
        <small class="d-block mb-3 text-muted">Воробйов Руслан К-42 -
Дипломний проект © 2019</small>
      </div>
  </div>
</footer>
```

Ж.5 Лістинг коду auth.php

```
<?php
      if($_COOKIE['user'] == 'True')
            setcookie('user', 'True', time() - 7200, '/');
      else
            setcookie('user', 'True', time() + 7200, '/');
      header('Location: /diplomV2/');?>
```

Ж.6 Лістинг коду login.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <title>АТ "Залізничний Вокзал"</title>
</head>
<body class="bg">
<?php   require "blocks/header.php" ?>
<?php   require "connect.php" ?>
<div class="container">

 <p align="center"><u><b><font size="5" color="#008000">Sign
UP</font></b></u></p>
    <form action=login.php method=post>
        <table border="0" cellpadding="2" class="table table-striped text-
left">
            <tr><th scope="row" class ="text-right">Логін</td><td><input
type="text" name="user" size="50"></th> </tr>
            <tr><th scope="row" class ="text-right">Пароль</td><td><input
type="password" name="pass" size="50"></th></tr>
            <tr><th scope="row" colspan="2" class ="text-center"><input
type="submit" value="Увійти" name="B1" class="btn btn-success btn-
lg"></th></tr>
        </table>
    </form>
<?php

if (isset($_POST['user'])) $userName = $_POST['user'];
if (isset($_POST['pass'])) $password = $_POST['pass'];

if (isset($userName) && isset($password)) {
    $result=$db->query("SELECT * from login where userName='$userName' and
password = '$password'");
    $count=$result->num_rows;
    if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        echo "<b><font color=red>Неправильний логін або
пароль!</font></b><br><br>";}

 else {
```

```
if($_COOKIE['user'] == 'True')
      setcookie('user', 'True', time() - 7200, '/');
   else
      setcookie('user', 'True', time() + 7200, '/');
      $_SESSION['user_id']=$row['id_login'];
      header('Location: /TETS/');} }  ?>
<?php   require "blocks/footer.php" ?>
</body>
</html>
```

Ж.7 Лістинг коду about.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-width, initial-cale=1.0">
      <meta http-equiv="X-UA-Compatible" content="ie=edge">
      <link rel="stylesheet" href="css/style.css">
      <link rel="stylesheet" href="css/bootstrap.min.css">
      <title>АТ "Залізничний Вокзал"</title>
</head>
      <body class="bg">
      <?php   require "blocks/header.php" ?>
            <div class="container">
            <h3>Контактна форма</h3>
      <form action="check.php" method="post">
            <input type="email" name="email" placeholder="Введіть Email"
                  class="form-control"><br>
            <textarea name="message" class="form-control"
                  placeholder="Введите ваше сообщение"></textarea><br>
            <button type="submit" name="send" class="btn btn-success btn-
lg">Відправити</button>
      </form></div>
      <?php   require "blocks/footer.php" ?>
      </body>
</html>
```

Ж.8 Лістинг коду check.php

```
<?php
      $email = $_POST['email'];
      $message = $_POST['message'];
      $error = '';
      if(trim($email) == '')
```

```
            $error = "Введіть вашу електронну адресу";
      else if(trim($message) == '')
            $error = 'Введіть ваше повідомлення';
      else if(strlen($message)<10)
            $error = 'Повідомлення повинно містити не меньш ніж 10 символів';
      if($error != '') {
            echo $error;
            exit; }
      $to = 'ruslanvorobievk42@gmail.com';
      $subject = "=?utf-8?B?".base64_encode("TEST")."?=";
      $headers = "From: $email\r\nReply-to: $email\r\nContent=type:
text/html; charset=utf-8\r\n;"
      mail($to, $subject, $message, $headers);
      header('Location: about.php');?>
```

Ж.9 Лістинг коду pasawir.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <title>АТ "Залізничний Вокзал"</title>
</head>
    <body class="bg">
    <?php   require "blocks/header.php" ?>
    <?php   require "connect.php" ?>

    <div class="container">
    <p align="center"><u><b><font size="5" color="#008000">Список пасажи-
рів</font></b></u></p>
    <form action=pasawir.php method=post>
        <table border="0" cellpadding="2" class="table table-striped text-
left">
            <tr><th scope="row" colspan="2" class ="text-center">Додати інфо-
рмацію:</th></tr>
            <tr><th scope="row" class ="text-right">Прізвище ім'я по батько-
ві</td><td><input type="text" name="add_PIB" size="50"></th> </tr>
            <tr><th scope="row" class ="text-right">Телефон</td><td><input
type="text" name="add_telephone" size="50"></th></tr>
            <tr><th scope="row" colspan="2" class ="text-center"><input
type="submit" value="Додати" name="B1" class="btn btn-success btn-
lg"></th></tr>
        </table>
```

```
    </form>
<?php
    if(isset($_POST['add_PIB'])) $add_PIB = $_POST['add_PIB'];
    if(isset($_POST['add_telephone'])) $add_telephone =
$_POST['add_telephone'];
    if(isset($_GET['del'])) $del = $_GET['del'];
//обработчик формы добавления нового тренера
    if (isset($add_PIB)) {
    $result=$db->query("SELECT * from pasawir where PIB='$add_PIB'");
    $count=$result->num_rows;
    if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        $query=("INSERT into pasawir values (0, '$add_PIB',
'$add_telephone')");
        $result=$db->query($query);
        echo "<b><font color=green>Інформація про пасажира успішно дода-
на!</font></b><br><br>"; }
        else { echo "<b><font color=red>Такий пасажир вже іс-
нує!</font></b><br><br>";}}
//конец обработчика формы

//удаление элемента
    if (isset($del)){
        $query=("DELETE from pasawir where id_pass='$del'");
        $result=$db->query($query);}
//конец блока удаления

//вывод информации базы данных
$result=$db->query("SELECT  *  from pasawir order by PIB");
$contactnum=$result->num_rows;
$i=0;
if ($res = $result) {
    /* извлечение ассоциативного массива */
    while ($row = $res->fetch_assoc()) {
        $id[$i] = $row["id_pass"]; $PIB[$i] = $row["PIB"];
        $telephone[$i] = $row["telephone"];
        $i++;  }
    /* удаление выборки */
    $res->free(); }
if ($contactnum<>0) { ?>
<table width=100%  border="1" cellspacing="0" cellpadding="0"
bordercolor="#E2E2E2" class="table table-striped text-center"><tr>
    <th>Прізвище ім'я по батькові</th>
    <th>Номер телефону</th>
    <th>***</th></tr>

<?php } $i=0;
```

```
while ($i<$contactnum) {
    echo "<tr><td>".$PIB[$i]."</td>";
    echo "<td>".$telephone[$i]."</td>";
    echo "<td><a href=pasawir.php?del=$id[$i]>Видалити</a></td></tr>";
$i++; }
//конец блока вывода информации
?>
</table>
</div>
<?php   require "blocks/footer.php" ?>
</body>
</html>
```

Ж.10 Лістинг коду voksal.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <title>АТ "Залізничний Вокзал"</title>
</head>
<body class="bg">
<?php   require "blocks/header.php" ?>
<?php   require "connect.php" ?>

<div class="container">
    <p align="center"><u><b><font size="5" color="#008000">Список вокза-
лів</font></b></u></p>
<form action=voksal.php method=post>
    <table border="0" cellpadding="2" class="table table-striped text-left">
    <tr><th scope="row" colspan="2" class ="text-center">Додати вок-
зал:</th></tr>
    <tr> <th scope="row" class ="text-right">Назва вокзала</th> <td> <input
type="text" name="add_name" size="50"></td> </tr>
    <tr> <th scope="row" class ="text-right">Назва міста</th> <td> <input
type="text" name="add_city" size="50"></td> </tr>
    <tr> <th scope="row" class ="text-right">Телефон</th> <td> <input
type="text" name="add_telephone" size="50"></td> </tr>
    <tr> <th scope="row" class ="text-right">Кількість платформ</th> <td>
<input type="text" name="add_vetki" size="1"></td> </tr>
    <tr><th scope="row" colspan="2" class ="text-center"><input type="submit"
value="Додати" name="B1" class="btn btn-success btn-lg"></th></tr>
    </table>
```

```
</form>
<?php
      if(isset($_POST['add_name'])) $add_name = $_POST['add_name'];
    if(isset($_POST['add_city'])) $add_city = $_POST['add_city'];
    if(isset($_POST['add_telephone'])) $add_telephone =
$_POST['add_telephone'];
    if(isset($_POST['add_vetki'])) $add_vetki = $_POST['add_vetki'];
    if(isset($_GET['del'])) $del = $_GET['del'];
//обработчик формы добавления нового вокзала
    if (isset($add_name) && isset($add_city)) {
    $result=$db->query("SELECT * from voksal where name='$add_name' and city
='$add_city'");
    $count=$result->num_rows;
    if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        $query=("INSERT into voksal values (0, '$add_name', '$add_city',
'$add_telephone', '$add_vetki')");
        $result=$db->query($query);
        echo "<b><font color=green>Інформація про вокзал успішно дода-
на!</font></b><br><br>"; }
        else { echo "<b><font color=red>Такий вокзал вже іс-
нує!</font></b><br><br>";}}
//конец обработчика формы

//удаление элемента
    if (isset($del)){
        $query=("DELETE from voksal where id_voksal='$del'");
        $result=$db->query($query);}
//конец блока удаления

//вывод информации о вокзалах, хранящихся в базе данных
$result=$db->query("SELECT  *  from voksal order by name");
$contactnum=$result->num_rows;
$i=0;
if ($res = $result) {
    /* извлечение ассоциативного массива */
    while ($row = $res->fetch_assoc()) {
        $id[$i] = $row["id_voksal"];
        $name[$i] = $row["name"];
        $city[$i] = $row["city"];
        $telephone[$i] = $row["telephone"];
        $vetki[$i] = $row["numbOfVetok"];
        $i++; }
    /* удаление выборки */
    $res->free(); }

if ($contactnum<>0) { ?>
```

```
<table width=100% border=1 cellspacing="0" cellpadding="0"
bordercolor="#E2E2E2" class="table table-striped text-center" ><tr>
      <th>Назва вокзала</td>
    <th>Назва міста</td>
    <th>Телефон</td>
    <th>Кількість платформ</td>
    <th>***</td></tr>

<?php } $i=0;
while ($i<$contactnum) {
    echo "<tr><td>".$name[$i]."</td>";
    echo "<td>".$city[$i]."</td>";
    echo "<td>".$telephone[$i]."</td>";
    echo "<td>".$vetki[$i]."</td>";
    echo "<td><a href=voksal.php?del=$id[$i]>Видалити</a></td></tr>";
$i++; }
//конец блока вывода информации
?>
</table>
</div>
<?php   require "blocks/footer.php" ?>
</body>
</html>
```

Ж.11 Лістинг коду route.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
<script src="js/jquery.min.js"></script>
<!-- Bootstrap library -->
<link href="css/bootstrap.min.css" rel="stylesheet">
<script src="js/bootstrap.min.js"></script>
<link href="css/bootstrap-datetimepicker.css" rel="stylesheet">
<script src="js/bootstrap-datetimepicker.min.js"></script>
    <title>АТ "Залізничний Вокзал"</title>
</head>
<body class="bg">
<?php   require "blocks/header.php" ?>
<?php   require "connect.php" ?>
<div class="container">
    <p align="center"><u><b><font size="5" color="#008000">Графік руху поїз-
дів</font></b></u></p>
```

```
<form action=route.php method=post>
    <table border="0" cellpadding="2" class="table table-striped text-left">
    <tr><th scope="row" colspan="2" class ="text-center">Додати марш-
рут:</th></tr>
    <tr><th scope="row" class ="text-right">Звідки</th><td><input type="text"
name="add_From" size="50"></td></tr>
    <tr><th scope="row" class ="text-right">Куди</th><td><input type="text"
name="add_To" size="50"></td> </tr>
    <tr><th scope="row" class ="text-right">Відбуття</th><td><input
type="text" name="add_start"  id="datetime1" readonly></td></tr>
    <tr><th scope="row" class ="text-right">Прибуття</th><td><input
type="text" name="add_finish" id="datetime2" readonly></td></tr>
    <tr><th scope="row" colspan="2" class ="text-center"><input type="submit"
value="Додати" name="B1" class="btn btn-success btn-lg"></th></tr>
    </table>
</form>

<script>
    var today = new Date();
            $("#datetime1,#datetime2").datetimepicker({
                format: 'yyyy-mm-dd hh:ii',
                autoclose: true,
                todayBtn: true,
                startDate : today
            });
</script>
<?php
    if(isset($_POST['add_From'])) $add_From = $_POST['add_From'];
    if(isset($_POST['add_To'])) $add_To = $_POST['add_To'];
    if(isset($_POST['add_start'])) $add_start = $_POST['add_start'];
    if(isset($_POST['add_finish'])) $add_finish = $_POST['add_finish'];
    if(isset($_GET['del'])) $del = $_GET['del'];
//обработчик формы добавления маршрута
    if (isset($add_From) && isset($add_To) && isset($add_start)) {
    $result=$db->query("SELECT * from graphic where placeFrom='$add_From' and
placeTo = '$add_To'
         and timeStart = '$add_start'");
    $count=$result->num_rows;
    if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        $query=("INSERT into graphic values (0, '$add_From', '$add_To',
'$add_start', '$add_finish')");
        $result=$db->query($query);
        echo "<b><font color=green>Інформація про маршрут успішна дода-
на!</font></b><br><br>"; }
        else { echo "<b><font color=red>Такий маршрут вже іс-
нує!</font></b><br><br>";}}
```

```
//конец обработчика формы
//удаление
    if (isset($del)){
        $query=("DELETE from graphic where id_route='$del'");
        $result=$db->query($query);}
//конец блока удаления

//вывод информации о маршрутах, хранящихся в базе данных
$result=$db->query("SELECT  *  from graphic order by timeStart");
$contactnum=$result->num_rows;
$i=0;
if ($res = $result) {
    /* извлечение ассоциативного массива */
    while ($row = $res->fetch_assoc()) {
        $id[$i] = $row["id_route"]; $placeFrom[$i] = $row["placeFrom"];
        $placeTo[$i] = $row["placeTo"]; $timeStart[$i] = $row["timeStart"];
        $timeFinish[$i] = $row["timeFinish"];
        $i++;  }
    /* удаление выборки */
    $res->free(); }

if ($contactnum<>0) { ?>
<table border="1" width="100%" cellspacing="0" cellpadding="0"
bordercolor="#E2E2E2" class="table table-striped text-center"><tr>
    <th>Місце відправлення</th>
    <th>Місце прибуття</th>
    <th>Час відправлення</th>
    <th>Час прибуття</th>
    <th>***</th></tr>

<?php } $i=0;
while ($i<$contactnum) {
    echo "<tr><td>".$placeFrom[$i]."</td>";
    echo "<td>".$placeTo[$i]."</td>";
    echo "<td>".$timeStart[$i]."</td>";
    echo "<td>".$timeFinish[$i]."</td>";
    echo "<td><a href=route.php?del=$id[$i]>Видалити</a></td></tr>";
$i++; }
//конец блока вывода информации
?>
</table>
</div>
<?php   require "blocks/footer.php"?>
</body>
</html>
```

Ж.12 Лістинг коду train.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <title>АТ "Залізничний Вокзал"</title>
</head>
<body class="bg">
<?php   require "blocks/header.php" ?>
<?php   require "connect.php" ?>

<div class="container">
    <p align="center"><u><b><font size="5" color="#008000">Список потя-
гів</font></b></u></p>
<form action=train.php method=post>
    <table border="0" cellpadding="2" class="table table-striped text-left">
    <tr><th scope="row" colspan="2" class ="text-center">Додати по-
тяг:</th></tr>
    <tr><th scope="row" class ="text-right">Кількість вагонів</th><td><input
type="text" name="add_vagons" size="50"></td></tr>
    <tr><th scope="row" class ="text-right">№ Потяга</th><td><input
type="text" name="add_title" size="50"></th></tr>
    <tr><th scope="row" class ="text-right">Максимальна швидкість
(км/год)</th><td><input type="text" name="add_speed" size="50"></td></tr>
    <tr><th scope="row" class ="text-right">Рік випуску</th><td><input
type="text" name="add_year" size="50"></td></tr>

    <tr><th scope="row" colspan="2" class ="text-center"><input type="submit"
value="Додати" name="B1" class="btn btn-success btn-lg"></th></tr>
    </table>
</form>

<?php
    if(isset($_POST['add_vagons'])) $add_vagons = $_POST['add_vagons'];
    if(isset($_POST['add_title'])) $add_title = $_POST['add_title'];
    if(isset($_POST['add_speed'])) $add_speed = $_POST['add_speed'];
    if(isset($_POST['add_year'])) $add_year = $_POST['add_year'];
    if(isset($_GET['del'])) $del = $_GET['del'];
//обработчик формы добавления нового поезда
    if (isset($add_title)) {
    $result=$db->query("SELECT * from train where title='$add_title'");
    $count=$result->num_rows;
```

```
    if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        $query=("INSERT into train values (0, '$add_vagons', '$add_title',
'$add_speed', '$add_year')");
        $result=$db->query($query);
        echo "<b><font color=green>Інформація про потяг успішно дода-
на!</font></b><br><br>"; }
        else { echo "<b><font color=red>Такий потяг вже іс-
нує!</font></b><br><br>";}}
//конец обработчика формы

//удаление элемента
    if (isset($del)){
        $query=("DELETE from train where id_train='$del'");
        $result=$db->query($query);}
//конец блока удаления

//вывод информации о поездах, хранящихся в базе данных
$result=$db->query("SELECT  *  from train order by title");
$contactnum=$result->num_rows;
$i=0;
if ($res = $result) {
    /* извлечение ассоциативного массива */
    while ($row = $res->fetch_assoc()) {
        $id[$i] = $row["id_train"]; $amountOfVagons[$i] =
$row["amountOfVagons"];
        $title[$i] = $row["title"];
        $speed[$i] = $row["maxspeed"]; $year[$i] = $row["godVipuska"];
        $i++;  }
    /* удаление выборки */
    $res->free(); }

if ($contactnum<>0) { ?>
<table width=100% border=1 cellspacing="0" cellpadding="0"
bordercolor="#E2E2E2" class="table table-striped text-center">
    <th>Кількість вагонів</th>
    <th>№ Потяга</th>
    <th>Макс. швидкість</th>
    <th>Рік випуска</th>
    <th>***</th></tr>

<?php } $i=0;
while ($i<$contactnum) {
    echo "<tr><td>".$amountOfVagons[$i]."</td>";
    echo "<td>".$title[$i]."</td>";
    echo "<td>".$speed[$i]."</td>";
    echo "<td>".$year[$i]."</td>";
```

```
    echo "<td><a href=train.php?del=$id[$i]>Видалити</a></td></tr>";
$i++; }
//конец блока вывода информации
?>
</table>
</div>
<?php   require "blocks/footer.php" ?>
</body>
</html>
```

Ж.13 Лістинг коду vagon.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <title>АТ "Залізничний Вокзал"</title>
</head>
<body class="bg">
<?php   require "blocks/header.php" ?>
<?php   require "connect.php" ?>

<div class="container">
    <p align="center"><u><b><font size="5"
color="#008000"></font></b></u></p>
<form action=vagon.php method=post>
    <table border="0" cellpadding="2" class="table table-striped text-left">
    <tr><th scope="row" colspan="2" class ="text-center">Додати ва-
гон:</th></tr>
    <tr><th scope="row" class ="text-right">Кількість місць</th><td><input
type="text" name="add_mesta" size="50"></td></tr>
    <tr><th scope="row" class ="text-right">Клас</th><td><input type="text"
name="add_class" size="50"></td></tr>


    <tr><th scope="row" colspan="2" class ="text-center"><input type="submit"
value="Додати" name="B1" class="btn btn-success btn-lg"></th></tr>
    </table>
</form>

<?php
    if(isset($_POST['add_mesta'])) $add_mesta = $_POST['add_mesta'];
    if(isset($_POST['add_class'])) $add_class = $_POST['add_class'];
```

```
    if(isset($_GET['del'])) $del = $_GET['del'];
//обработчик формы добавления
    if (isset($add_mesta) && isset($add_class)) {
    $result=$db->query("SELECT * from vagon where class='$add_class' and
numbOfPlace ='$add_mesta'");
    $count=$result->num_rows;
    if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        $query=("INSERT into vagon values (0, '$add_mesta', '$add_class')");
        $result=$db->query($query);
        echo "<b><font color=green>Інформація про вагон успішно дода-
на!</font></b><br><br>"; }
        else { echo "<b><font color=red>Такий вагон вже іс-
нує!</font></b><br><br>";}}
//конец обработчика формы

//удаление элемента
    if (isset($del)){
        $query=("DELETE from vagon where id_vagon='$del'");
        $result=$db->query($query);}
//конец блока удаления

//вывод информации о вагонах, хранящихся в базе данных
$result=$db->query("SELECT  *  from vagon order by class");
$contactnum=$result->num_rows;
$i=0;
if ($res = $result) {
    /* извлечение ассоциативного массива */
    while ($row = $res->fetch_assoc()) {
        $id[$i] = $row["id_vagon"]; $mesta[$i] = $row["numbOfPlace"];
        $class[$i] = $row["class"];
        $i++;  }
    /* удаление выборки */
    $res->free(); }
if ($contactnum<>0) { ?>
<table width=100% border=1 cellspacing="0" cellpadding="0"
bordercolor="#E2E2E2" class="table table-striped text-center"><tr>
    <th>Кількість місць</th>
    <th>Клас</th>
    <th>***</th></tr>

<?php } $i=0;
while ($i<$contactnum) {
    echo "<tr><td>".$mesta[$i]."</td>";
    echo "<td>".$class[$i]."</td>";
    echo "<td><a href=vagon.php?del=$id[$i]>Видалити</a></td></tr>";
$i++; }
```

```
//конец блока вывода информации
?>
</table>
</div>
<?php   require "blocks/footer.php" ?>
</body>
</html>
```

Ж.14 Лістинг коду personal.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <title>АТ "Залізничний Вокзал"</title>
</head>
<body class="bg">
<?php   require "blocks/header.php" ?>
<?php   require "connect.php" ?>
<div class="container">
    <p align="center"><u><b><font size="5"
color="#008000">Персонал</font></b></u></p>
<form action=personal.php method=post>
    <table border="0" cellpadding="2" class="table table-striped text-left">
    <tr><th scope="row" colspan="2" class ="text-center">Додати працівни-
ка:</th></tr>
    <tr><th scope="row" class ="text-right">ПІБ</th> <td> <input type="text"
name="add_PIB" size="50"></td> </tr>
    <tr><th scope="row" class ="text-right">Посада</th> <td> <input
type="text" name="add_posada" size="50"></td> </tr>
    <tr><th scope="row" class ="text-right">Телефон</th> <td> <input
type="text" name="add_telephone" size="50"></td> </tr>
    <tr><th scope="row" class ="text-right">Місце роботи</th> <td> <input
type="text" name="add_address" size="50"></td> </tr>
    <tr><th scope="row" class ="text-right">Заробітна плата (грн)</th> <td>
<input type="text" name="add_zarplata" size="50"></td> </tr>
    <tr><th scope="row" colspan="2" class ="text-center"><input type="submit"
value="Додати" name="B1" class="btn btn-success btn-lg"></th></tr>
    </table></form>
<?php
    if(isset($_POST['add_PIB'])) $add_PIB = $_POST['add_PIB'];
    if(isset($_POST['add_posada'])) $add_posada = $_POST['add_posada'];
```

```
    if(isset($_POST['add_telephone'])) $add_telephone =
$_POST['add_telephone'];
    if(isset($_POST['add_address'])) $add_address = $_POST['add_address'];
    if(isset($_POST['add_zarplata'])) $add_zarplata = $_POST['add_zarplata'];
    if(isset($_GET['del'])) $del = $_GET['del'];
//обработчик формы добавления нового тренера
    if (isset($add_PIB) && isset($add_posada)) {
    $result=$db->query("SELECT * from personal where PIB='$add_PIB' and
posada = '$add_posada'");
    $count=$result->num_rows;
    if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        $query=("INSERT into personal values (0, '$add_PIB', '$add_posada',
'$add_telephone', '$add_address', '$add_zarplata')");
        $result=$db->query($query);
        echo "<b><font color=green>Інформація про персонал успішно дода-
на!</font></b><br><br>"; }
        else { echo "<b><font color=red>Такий працівник вже іс-
нує!</font></b><br><br>";}}

//конец обработчика формы
//удаление элемента
    if (isset($del)){
        $query=("DELETE from personal where id_person='$del'");
        $result=$db->query($query);}
//конец блока удаления

//вывод информации о персонале, хранящимся в базе данных
$result=$db->query("SELECT  *  from personal order by PIB");
$contactnum=$result->num_rows;
$i=0;
if ($res = $result) {
    /* извлечение ассоциативного массива */
    while ($row = $res->fetch_assoc()) {
        $id[$i] = $row["id_person"]; $FIO[$i] = $row["PIB"];
        $posada[$i] = $row["posada"]; $telephone[$i] = $row["telephone"];
        $address[$i] = $row["address"]; $zarplata[$i] = $row["zarplata"];
        $i++;  }
    /* удаление выборки */
    $res->free(); }
if ($contactnum<>0) { ?>
<table width=100% border=1 cellspacing="0" cellpadding="0"
bordercolor="#E2E2E2" class="table table-striped text-center"><tr>
    <th>ПІБ</th>
    <th>Посада</th>
    <th>Номер телефону</th>
    <th>Місце працювання</th>
```

```
    <th>Заробітна платня (грн)</th>
    <th>***</th></tr>

<?php } $i=0;
while ($i<$contactnum) {
    echo "<tr><td>".$FIO[$i]."</td>";
    echo "<td>".$posada[$i]."</td>";
    echo "<td>".$telephone[$i]."</td>";
    echo "<td>".$address[$i]."</td>";
    echo "<td>".$zarplata[$i]."</td>";
    echo "<td><a href=personal.php?del=$id[$i]>Удалить</a></td></tr>";
$i++; }
//конец блока вывода информации
?>
</table>
</div>
<?php   require "blocks/footer.php" ?>
</body>
</html>
```

Ж.15 Лістинг коду usluga.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <title>АТ "Залізничний Вокзал"</title>
</head>
<body class="bg">
<?php   require "blocks/header.php" ?>
<?php   require "connect.php" ?>
<div class="container">
    <p align="center"><u><b><font size="5" color="#008000">Список пос-
луг</font></b></u></p>
<form action=usluga.php method=post>

    <p><b>Додати послугу:</b> </p>
    <table border="0" cellpadding="2" class="table table-striped text-left">
    <tr><th scope="row" colspan="2" class ="text-center">Додати послу-
гу:</th></tr>

    <tr><th scope="row" class ="text-right">Назва послуги</th><td><input
type="text" name="add_nazva" size="50"></td></tr>
```

```
    <tr><th scope="row" class ="text-right">Вартість (грн)</th><td><input
type="text" name="add_cost" size="50"></td></tr>

    <tr><th scope="row" colspan="2" class ="text-center"><input type="submit"
value="Додати" name="B1" class="btn btn-success btn-lg"></th></tr>
    </table>
</form>

<?php
    if(isset($_POST['add_nazva'])) $add_nazva = $_POST['add_nazva'];
    if(isset($_POST['add_cost'])) $add_cost = $_POST['add_cost'];
    if(isset($_GET['del'])) $del = $_GET['del'];
//обработчик формы добавления новой услуги
    if (isset($add_nazva)) {
    $result=$db->query("SELECT * from usluga where nazvanie='$add_nazva'");
    $count=$result->num_rows;
    if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        $query=("INSERT into usluga values (0, '$add_nazva', '$add_cost')");
        $result=$db->query($query);
        echo "<b><font color=green>Інформація про послугу успішно дода-
на!</font></b><br><br>"; }
        else { echo "<b><font color=red>Така послуга вже іс-
нує!</font></b><br><br>";}}

//конец обработчика формы
//удаление элемента
    if (isset($del)){
        $query=("DELETE from usluga where id_service='$del'");
        $result=$db->query($query);}
//конец блока удаления

//вывод информации о услугах, хранящихся в базе данных
$result=$db->query("SELECT  *  from usluga order by nazvanie");
$contactnum=$result->num_rows;
$i=0;
if ($res = $result) {
    /* извлечение ассоциативного массива */
    while ($row = $res->fetch_assoc()) {
        $id[$i] = $row["id_service"]; $nazvanie[$i] = $row["nazvanie"];
        $stoimost[$i] = $row["stoimost"];
        $i++;  }
    /* удаление выборки */
    $res->free(); }
if ($contactnum<>0) { ?>
<table width=100% border=1 cellspacing="0" cellpadding="0"
bordercolor="#E2E2E2" class="table table-striped text-center"><tr>
```

```
    <th>Назва послуги</th>
    <th>Вартість (грн)</th>
    <th>***</th></tr>

<?php } $i=0;
while ($i<$contactnum) {
    echo "<tr><td>".$nazvanie[$i]."</td>";
    echo "<td>".$stoimost[$i]."</td>";
    echo "<td><a href=usluga.php?del=$id[$i]>Видалити</a></td></tr>";
$i++; }
//конец блока вывода информации
?>
</table>
</div>
<?php   require "blocks/footer.php" ?>
</body>
</html>
```

Ж.16 Лістинг коду bilet.php

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-cale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <title>АТ "Залізничний Вокзал"</title>
</head>
<body class="bg">
<?php   require "blocks/header.php" ?>
<?php   require "connect.php" ?>
<div class="container">
<p align="center"><u><b><font size="5" color="#008000">Інформація про кви-
ток</font></b></u></p>
<form action=bilet.php method=post>
<table border="0" id="table2" class="table table-striped text-left">
    <tr><th scope="row" colspan="2" class ="text-center">Додати інформа-
цію:</th></tr>

      <tr><th scope="row" class ="text-right">ПІБ пасажира</th><td>
        <select name="add_pass" size="1">
        <option></option> <?php $result=$db->query("SELECT * from pasawir
order by PIB");
        while ($row = $result->fetch_assoc()) {
        print "<option value=\"$row[id_pass]\">$row[PIB]</option>";  } ?>
```

```
        </select></td></tr>

    <tr><th scope="row" class ="text-right">Назва маршруту</th><td >
        <select name="add_route" size="1">
        <option></option> <?php $result=$db->query("SELECT * from graphic
order by timeStart");
        while ($row = $result->fetch_assoc()) {
        print "<option value=\"$row[id_route]\">$row[placeFrom]
($row[timeStart]) - $row[placeTo] ($row[timeFinish]) </option>";  } ?>
        </select></td></tr>

    <tr><th scope="row" class ="text-right">Назва вокзала</th><td >
        <select name="add_voksalName" size="1">
        <option></option> <?php $result=$db->query("SELECT * from voksal
order by name");
        while ($row = $result->fetch_assoc()) {
        print "<option value=\"$row[id_voksal]\">$row[name]</option>";  } ?>
        </select></td></tr>

    <tr><th scope="row" class ="text-right">№ Потяга (кількість ваго-
нів)</th><td >
        <select name="add_train" size="1">
        <option></option> <?php $result=$db->query("SELECT * from train order
by title");
        while ($row = $result->fetch_assoc()) {
        print "<option value=\"$row[id_train]\">$row[title]
($row[amountOfVagons])</option>";  } ?>
        </select></td></tr>

    <tr><th scope="row" class ="text-right">№ Вагона </th><td>
            <input type="text" name="vagonNumber" id="vagonNumber" size ="5">
        </td></tr>

      <tr><th scope="row" class ="text-right">Клас вагона та кількість
місць</th><td>
        <select name="add_class" size="1">
        <option></option> <?php $result=$db->query("SELECT * from vagon order
by class");
        while ($row = $result->fetch_assoc()) {
        print "<option value=\"$row[id_vagon]\">$row[class]
($row[numbOfPlace])</option>";  } ?>
        </select></td></tr>

         <tr><th scope="row" class ="text-right">№ місця </th><td>
            <input type="text" name="PlaceNumber" id="PlaceNumber" size ="5">
        </td></tr>

        <tr><th scope="row" class ="text-right">Послуга (Ціна)</th><td>
```

```
        <select name="add_usluga" size="1">
        <option></option> <?php $result=$db->query("SELECT * from usluga
order by nazvanie");
        while ($row = $result->fetch_assoc()) {
        print "<option value=\"$row[id_service]\">$row[nazvanie]
($row[stoimost])</option>";  } ?>
        </select></td></tr>

         <tr><th scope="row" class ="text-right">Ціна квитка (грн) </th><td>
            <input type="text" name="add_cost" id="add_cost" size ="5">
        </td></tr>
        <tr><th scope="row" colspan="2" class ="text-center"><input
type="submit" value="Додати" name="B1" class="btn btn-success btn-
lg"></th></tr>
    </table>
    </form>
<?php
    if(isset($_POST['add_pass'])) $add_pass = $_POST['add_pass'];
    if(isset($_POST['add_route'])) $add_route = $_POST['add_route'];
    if(isset($_POST['add_voksalName'])) $add_voksalName =
$_POST['add_voksalName'];
    if(isset($_POST['add_train'])) $add_train = $_POST['add_train'];
    if(isset($_POST['vagonNumber'])) $vagonNumber = $_POST['vagonNumber'];
    if(isset($_POST['add_class'])) $add_class = $_POST['add_class'];
    if(isset($_POST['PlaceNumber'])) $PlaceNumber = $_POST['PlaceNumber'];
    if(isset($_POST['add_usluga'])) $add_usluga = $_POST['add_usluga'];
    if(isset($_POST['add_cost'])) $add_cost = $_POST['add_cost'];
    if(isset($_GET['del'])) $del = $_GET['del'];

//обработчик формы добавления информации
if (isset($add_route) && isset($add_train) && isset($vagonNumber) &&
isset($PlaceNumber) ) {
$res=$db->query("SELECT  amountOfVagons  from train where id_train =
'$add_train'");
    while ($row = $res->fetch_assoc()) {
    $maxVagon=$row["amountOfVagons"];} $res->free();
    if ($maxVagon<$vagonNumber) {echo "<b><font color=red>Перевищено номер
вагона</font></b><br><br>"; return;}
    else {
$res=$db->query("SELECT  numbOfPlace  from vagon where id_vagon =
'$add_class'");
    while ($row = $res->fetch_assoc()) {
    $maxMest=$row["numbOfPlace"];} $res->free();
    if ($maxMest<$PlaceNumber) {echo "<b><font color=red>Перевищено максима-
льний ліміт місць у вагоні</font></b><br><br>"; return;
        }}
```

```
      $result=$db->query("SELECT  *  from bilet where id_graphic =
'$add_route' and id_train='$add_train' and vagonNumb ='$vagonNumber' and
placeNumb  = '$PlaceNumber'");
    $count=$result->num_rows;
if (!$result) {
    trigger_error('Invalid query: ' . $db->error); }
    if ($count==0) {
        $query=("INSERT into bilet values (0, '$add_pass', '$add_route',
'$add_voksalName', '$add_train',
            '$vagonNumber', '$add_class', '$PlaceNumber',
'$add_usluga','$add_cost')");
         $result=$db->query($query);
         echo "<b><font color=green>Інформація успішно дода-
на!</font></b><br><br>"; }
        else { echo "<b><font color=red>Інформація вже присут-
ня!</font></b><br><br>"; }}
//конец обработчика формы добавления

//удаление
if (isset($del)){
$query=("DELETE from bilet where id_bilet='$del'");
$result=$db->query($query); }
//конец блока удаления

//вывод информации, хранящейся в базе данных
$result=$db->query("SELECT  *  from bilet order by id_bilet");
$contactnum=$result->num_rows; $i=0;
if ($res = $result) {
    /* извлечение ассоциативного массива */
    while ($row = $res->fetch_assoc()) {

        $id[$i]=$row["id_bilet"]; $id_pass[$i]=$row["id_pass"];
        $id_route[$i]=$row["id_graphic"];
         $id_voksal[$i]=$row["id_voksal"];
        $id_train[$i]=$row["id_train"];
        $id_vagon[$i]=$row["id_vagon"];
        $id_usluga[$i]=$row["id_usluga"];
         $i++;  }
    /* удаление выборки */
    $res->free(); }

if ($contactnum<>0) { ?>
<table border="1" width="100%" id="table3" cellspacing="0" cellpadding="0"
bordercolor="#E2E2E2" class="table table-striped text-center">
      <tr><th scope="col">ПІБ пасажира</th>
            <th>Маршрут</th>
            <th>Назва вокзала</th>
        <th>№ потяга (кількість вагонів)</th>
```

```
        <th>№ вагону</th>
        <th>Клас вагону (місць)</th>
        <th>№ місця</th>
        <th>Послуга (ціна)</th>
        <th>Ціна квитка (грн)</th>
        <th>***</th> </tr>
<?php }

$i=0;
while ($i<$contactnum) {
$result=$db->query("SELECT  PIB  from pasawir where id_pass='$id_pass[$i]'");
while ($row = $result->fetch_assoc()) {
    $pass=$row["PIB"];} $result->free();

$result=$db->query("SELECT  placeFrom, placeTo, timeStart, timeFinish
      from graphic where id_route='$id_route[$i]'");
while ($row = $result->fetch_assoc()) {
    $pFrom=$row["placeFrom"]; $pTo=$row["placeTo"];
    $tStart=$row["timeStart"]; $tFinish=$row["timeFinish"];
} $result->free();

$result=$db->query("SELECT  name  from voksal where
id_voksal='$id_voksal[$i]'");
while ($row = $result->fetch_assoc()) {
    $voksal=$row["name"];} $result->free();

$result=$db->query("SELECT  title, amountOfVagons  from train where
id_train='$id_train[$i]'");
while ($row = $result->fetch_assoc()) {
    $trainName=$row["title"]; $vagons=$row["amountOfVagons"]; } $result-
>free();

$result=$db->query("SELECT  class, numbOfPlace  from vagon where
id_vagon='$id_vagon[$i]'");
while ($row = $result->fetch_assoc()) {
    $class=$row["class"]; $numbOfPlace=$row["numbOfPlace"]; } $result-
>free();
$result=$db->query("SELECT  nazvanie, stoimost  from usluga where
id_service='$id_usluga[$i]'");
while ($row = $result->fetch_assoc()) {
    $uslagaName=$row["nazvanie"]; $uslugaPrice=$row["stoimost"]; } $result-
>free();
$result=$db->query("SELECT  vagonNumb, placeNumb, biletPrice  from bilet
where id_bilet='$id[$i]'");
while ($row = $result->fetch_assoc()) {
    $vagonNumb=$row["vagonNumb"];
    $placeNumb=$row["placeNumb"]; $biletPrice=$row["biletPrice"]; } $result-
>free();
```

```
echo "<tr><td align='left'>".$pass."</td>";
echo "<td align='left'>".$pFrom.' ('.$tStart.')'.' - '.$pTo.' ('.$tFinish.')
'."</td>";
echo "<td>".$voksal."</td>";
echo "<td>".$trainName.' ('.$vagons.') '."</td>";
echo "<td>".$vagonNumb."</td>";
echo "<td>".$class.' ('.$numbOfPlace.') '."</td>";
echo "<td>".$placeNumb."</td>";
echo "<td>".$uslagaName.' ('.$uslugaPrice.') '."</td>";
echo "<td>".$biletPrice."</td>";
echo "<td><a href=bilet.php?del=$id[$i]>Видалити</a></td></tr>";
$i++;} ?>
</table>
<?php   require "blocks/footer.php" ?>
</div>
</body>
</html>
```

Ж.17 Лістинг коду style.css

```
.headerFont{
  height: 100%;
      font-size: 1.25rem;
      color: black;      }
.bg {
    background-image: url(../pic/back.jpg);
    background-position-x: center;
    background-position-y: top;
    background-size: initial;
    background-repeat-x: no-repeat;
    background-repeat-y: no-repeat;
    background-attachment: initial;
    background-origin: initial;
    background-clip: initial;
    background-color: rgb(255, 255, 255); }
   .table {
   margin: auto;
   width: 100% !important;  }
```