

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА

Навчально-науковий інститут «Інститут інформаційних технологій в економіці»

Кафедра системного аналізу та кібербезпеки

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «СИСТЕМНИЙ АНАЛІЗ»

Галузь знань 12 «Інформаційні технології»

Спеціальність 124 «Системний аналіз»

Форма навчання: (очна) денна

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

на тему «Розробка та впровадження системи обліку автобусів компанії та продажу квитків для міжміського перевезення пасажирів»

здобувача Голика Івана Костянтиновича

(підпис)

Науковий керівник: к.е.н., доцент Лазарєва С.Ф.

(підпис)

Робота допущена до захисту перед екзаменаційною комісією з атестації здобувачів вищої освіти (ЕК)

Завідувач кафедри: д.ф.-м.н., проф. Джалладова І.А.

(підпис)

Київ 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ
ВАДИМА ГЕТЬМАНА

Навчально-науковий інститут «Інститут інформаційних технологій в економіці»

Кафедра системного аналізу та кібербезпеки

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «СИСТЕМНИЙ АНАЛІЗ»
СПЕЦІАЛЬНІСТЬ 124 «СИСТЕМНИЙ АНАЛІЗ»

ПОГОДЖЕНО

Керівник проєктної групи (гарант)
освітньо-професійної програми

_____ Галіцин В.К.
(підпис)

_____ 20__ р.

ЗАТВЕРДЖУЮ

Завідувач кафедри системного
аналізу та кібербезпеки

_____ Джалладова І.А.
(підпис)

_____ 20__ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувачу вищої освіти Голик Іван Костантинівич
денної форми навчання

на підготовку кваліфікаційної бакалаврської роботи

на тему «Розробка та впровадження системи обліку автобусів компанії та продажу квитків для міжміського перевезення пасажирів»

Тему затверджено наказом ректора Університету від «_____» 20__ р. № _____

Кваліфікаційна бакалаврська робота виконується на матеріалах
пасажирського підприємства ПП "Нагірний Бус", ресурсів Інтернет.

План кваліфікаційної бакалаврської роботи

Розділ 1	Діджиталізація сфери комерційного перевезення як один із кроків до розвитку автобусних перевезень Україною.
Розділ 2	Аналіз вимог до нової системи продажу квитків і обліку транспортних засобів підприємства.

Розділ 3	Розробка системи та її впровадження.
Об'єкт дослідження:	Сфера пасажирських перевезень.
Предмет дослідження:	Ефективність сфери автобусних перевезень.
Мета кваліфікаційної бакалаврської роботи:	Підвищення якості обслуговування пасажирів у сфері автобусних перевезень.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі 1	
1. Дослідити необхідність цифровізації процесів у сфері пасажирських перевезень.	
2. Проаналізувати існуючі системи обліку та продажу квитків для міжміських перевезень пасажирів.	
3. Проаналізувати технології, що можуть бути використані для розробки системи.	
У розділі 2	
1. Визначити вимог до системи.	
2. Проаналізувати функціональних та нефункціональних вимог.	
3. Сформулювати вимоги до архітектури системи.	
4. Сформулювати вимоги до інтерфейсу користувача.	
У розділі 3	
1. Вибрати технології та інструменти для розробки системи.	
2. Спроекувати базу даних, виділити об'єкти предметної області, створити власну структур даних.	
3. Розробити функціональність системи, протестувати та налагодити результат розробки.	

**Завдання підготував
науковий керівник**

(підпис)

Лазарєва С.Ф.

«____» _____ 20__ р.

**Завдання одержав
здобувач**

(підпис)

Голик І.К.

«____» _____ 20__ р.

Реферат

Кваліфікаційна бакалаврська робота містить 62 сторінки, 5 рисунків, список використаних джерел з 31 найменувань, 3 додатки.

«Розробка та впровадження системи обліку автобусів компанії та продажу квитків для міжміського перевезення пасажирів»

Об'єктом дослідження є сфера пасажирських перевезень, зокрема автобусні перевезення.

Предметом дослідження є ефективність та конкурентоспроможність сфери автобусних перевезень, зокрема процеси бронювання квитків, організації перевезень, управління автобусним парком, а також якість обслуговування пасажирів.

Мета кваліфікаційної бакалаврської роботи – покращення якості обслуговування пасажирів у сфері автобусних перевезень, забезпечення зручного бронювання квитків та ефективного управління автобусним парком, а також оптимізація перевезень до країн Європи для забезпечення швидкості та зручності.

Відповідно до поставленої мети були визначені такі завдання:

- визначити економічну сутність та значення цифровізації процесів у сфері пасажирських перевезень;
- проаналізувати існуючі системи обліку та продажу квитків для міжміських перевезень пасажирів;
- дослідити технології, які можуть бути використані для розробки системи;
- визначити вимоги до нової системи продажу квитків і обліку транспортних засобів підприємства;
- проаналізувати функціональні та нефункціональні вимоги до системи;

- розробити архітектуру системи та визначити вимоги до її компонентів;
- впровадити систему та протестувати її роботу;

Практичне значення отриманих результатів. На базі проведеного аналізу та розробки нової системи продажу квитків і обліку транспортних засобів надані рекомендації щодо впровадження цифрових технологій у сфері автобусних перевезень. Описана методика розробки та впровадження системи може бути використана на практиці під час оцінки діяльності та оптимізації перевезень як національних, так і міжнародних автобусних компаній.

Рік виконання кваліфікаційної бакалаврської роботи 2023. Рік захисту роботи 2024.

Ключові слова: цифровізація, автобусні перевезення, аналіз системи, розробка системи, система продажу квитків, облік транспортних засобів, пасажирські перевезення.

Відгук
про кваліфікаційну бакалаврську роботу
здобувача навчально-наукового інституту
«Інститут інформаційних технологій в економіці»
освітньо-професійної програми «Системний аналіз»
Голика Івана Костянтиновича
на тему «Розробка та впровадження системи обліку автобусів компанії та продажу
квитків для міжміського перевезення пасажирів»

Актуальність теми. Нині рівень інформатизації та диджиталізації в різних галузях досить різниться. Сфера автомобільних пасажирських перевезень належить до числа тих, де рівень цих процесів ще недостатньо високий. Отже, тема кваліфікаційної бакалаврської роботи Голика Івана Костянтиновича є актуальною

Позитивні риси кваліфікаційної бакалаврської роботи. Робота є комплексним і змістовним дослідженням проблем використання інформаційних систем і технологій у сфері автобусних перевезень, зокрема у процесах бронювання квитків, організації перевезень, управління автобусним парком, що безпосередньо впливають на якість обслуговування пасажирів.

Наявність самостійних розробок автора. При написанні КБР Голик Іван Костянтинович показав себе як сформований фахівець, здатний розв'язувати складні як теоретичні, так і практичні проблеми системного аналізу. Автор самостійно, розробив інформаційну систему для обліку автобусів і бронювання квитків.

Цінність теоретичних висновків та практичних рекомендацій. Теоретичні висновки є коректними, базуються на опрацюванні значної кількості першоджерел, аналізі досвіду передових країн світу, а також досвіду України з питань диджиталізації підприємств, що надають послуги з перевезення пасажирів. Розроблена система може бути запропонована для використання підприємствам, що надають послуги з пасажирських автоперевезень.

Наявність недоліків. Недоліків, які могли б вплинути на позитивну оцінку кваліфікаційної роботи не виявлено. Незначні редакційні та стилістичні огріхи не впливають на позитивне враження від роботи.

Загальна оцінка кваліфікаційної бакалаврської роботи та її допущення до захисту перед ЕК. Робота відповідає вимогам, що висуваються до кваліфікаційних робіт на здобуття першого (бакалаврського) освітнього ступеня, зі спеціальності 124 «Системний аналіз», за освітньо-професійною програмою «Системний аналіз».

При написанні кваліфікаційної бакалаврської роботи Голик Іван Костянтинович показав себе як сформований фахівець, здатний розв'язувати складні як теоретичні, так і практичні проблеми системного аналізу.

Кваліфікаційна бакалаврська робота Голика Івана Костянтиновича може бути рекомендована до захисту перед ЕК. Робота заслуговує високої оцінки, а її автор - присудження освітньої кваліфікації бакалавр системного аналізу.

Науковий керівник:

к.е.н., професор кафедри
системного аналізу та
кібербезпеки

Лазарева С.Ф

РЕЦЕНЗІЯ
на кваліфікаційну бакалаврську роботу
здобувача вищої освіти

Голика Івана Костянтиновича

Тема: Розробка та впровадження системи обліку автобусів компанії та продажу
квитків для міжміського перевезення пасажирів

1. Актуальність теми кваліфікаційної роботи і доцільність її розроблення

У контексті постійного зростання попиту на транспортні послуги та вимог до якості обслуговування - підвищення ефективності та конкурентоспроможності є актуальною темою в сфері пасажирських перевезень. Одним із головних шляхів задоволення цього попиту є впровадження сучасних цифрових технологій. Це надійний спосіб у підвищенні ефективності на підприємстві, тому розробка нової інформаційної системи для обліку автобусів і бронювання квитків є своєчасною та доцільною.

2. Позитивні сторони у роботі, які бажає підкреслити рецензент

Попри те, що аналіз поточної ситуації, розробка і впровадження систем є справою не однієї людини — здобувач зміг провести аналіз проблем галузі і деталізувати їх у сфері пасажирських перевезень та запропонував обґрунтовані рішення. Студентом були визначені чіткі та обґрунтовані функціональні та нефункціональні вимоги до системи. Зибір сучасних технологій був незвичайним, але це надійні рішення з поглядом на майбутнє системи.

3. Недоліки або дискусійні питання у роботі

У роботі розкрито тільки найбільш актуальні питання розробки і впровадження, з урахуванням наявних ресурсів. Проте є деякі аспекти, що були спеціально залишені студентом на потім. Як приклад описується можливість масштабування і переходу від монолітної структури до мікросервісної. Навіть якщо ресурсів на розробку і попиту на систему було небагато — все одно варто було детальніше розглянути можливі виклики при значному збільшенні користувачів.

4. Оцінка якості запропонованих студентом заходів і пропозицій, їх практична цінність

Запропоновані заходи та пропозиції мають високу якість і практичну цінність. Студент пропонує конкретні рішення для оптимізації процесів бронювання квитків та управління автобусним парком, використовуючи сучасні технології. Використання PWA дозволяє створити зручну та доступну платформу для пасажирів, що підвищує їхній рівень задоволеності. Інтеграція з платіжними системами, такими як PayPal, розширює можливості оплати, що є важливим для сучасного бізнесу. Пропозиції щодо подальшого вдосконалення системи також виглядають перспективними і мають потенціал для реалізації.

5. Загальний висновок рецензента на бакалаврську роботу студента

Студент показав високі теоретичні знання і практичні навички. Робота відзначається актуальністю, науковою новизною та практичною значущістю. Студент успішно виконав поставлені завдання, використавши сучасні технології та методи розробки інформаційних систем. Недоліки роботи є незначними і можуть бути усунені в подальшій практичній діяльності. Загалом, робота заслуговує на високу оцінку та рекомендується до захисту.

Рецензент _____

завідувач кафедри
(посада)


(підпис)

Аль-Амморі А.Н.
(ініціали, прізвище)

Кафедра Інформаційно-аналітичної діяльності та інформаційної безпеки
Національного Транспортного Університету
(місце роботи)



Печатка

« 12 » червня 2024 р.

ЗМІСТ

ВСТУП.....	3
РОЗДІЛ 1. ДІДЖИТАЛІЗАЦІЯ СФЕРИ КОМЕРЦІЙНОГО ПЕРЕВЕЗЕННЯ ЯК ОДИН ІЗ КРОКІВ ДО РОЗВИТКУ АВТОБУСНИХ ПЕРЕВЕЗЕНЬ УКРАЇНОЮ	5
1.1. Необхідність цифровізації процесів у сфері пасажирських перевезень.....	5
1.2. Аналіз існуючих систем обліку та продажу квитків для міжміських перевезень пасажирів.....	9
1.3. Аналіз технологій, що можуть бути використані для розробки системи.....	18
РОЗДІЛ 2. АНАЛІЗ ВИМОГ ДО НОВОЇ СИСТЕМИ ПРОДАЖУ КВИТКІВ І ОБЛІКУ ТРАНСПОРТНИХ ЗАСОБІВ ПІДПРИЄМСТВА.....	24
2.1 Визначення вимог до системи.....	24
2.2 Аналіз функціональних та нефункціональних вимог.....	25
2.3 Формулювання вимог до архітектури системи.....	31
2.4 Формулювання вимог до інтерфейсу користувача.....	35
РОЗДІЛ 3. РОЗРОБКА СИСТЕМИ ТА ЇЇ ВПРОВАДЖЕННЯ.....	37
3.1 Вибір технологій та інструментів для розробки системи.....	37
3.2 проектування бази даних, виділення об'єктів предметної області, створення власних структур даних.....	40
3.3 Розробка функціональності системи, тестування та налагодження системи	46
ВИСНОВКИ.....	57
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	59
ДОДАТКИ.....	63

ВСТУП

У сучасному світі, де мобільність та швидкість стали необхідністю, сфера пасажирських перевезень постійно розвивається та адаптується до змінних потреб суспільства. Особливо важливою є сфера автобусних перевезень, яка забезпечує зручні та доступні маршрути для міжміських та міжнародних поїздок. Однак, несучи на собі велике навантаження від зростаючого попиту, ця галузь стикається з рядом проблем, які обмежують її ефективність та конкурентоспроможність.

Аналіз останніх досліджень показав, що мобільні додатки стають надзвичайно популярними серед підприємств та постачальників послуг. Це дослідження використовувало гнучку методологію для розробки веб-порталу, базуючись на технології PWA(Progressive web application, прогресивний веб додаток).

Мета дослідження полягає в покращенні якості обслуговування пасажирів у сфері автобусних перевезень, забезпеченні зручного бронювання квитків та ефективного управління автобусним парком, а також в оптимізації перевезень до країн Європи для забезпечення швидкості та зручності.

Завдання дослідження включають:

4. Аналіз проблем і недоліків у сфері автобусних перевезень, які обмежують ефективність та конкурентоспроможність.
5. Визначення потреб сучасного споживача та вимог до інформаційних систем та технологій.
6. Розробка та впровадження нової інформаційної системи для зручного бронювання квитків та керування автобусним парком.
7. Встановлення системи відстеження рейсів та моніторингу роботи автобусів у реальному часі.

8. Розробка рекомендацій щодо подальшого вдосконалення та розширення системи на автобусних підприємствах, які здійснюють перевезення до країн Європи.

Об'єктом дослідження є сфера пасажирських перевезень, зокрема автобусні перевезення, які забезпечують зручні та доступні маршрути для міжміських та міжнародних поїздок.

Предметом дослідження є ефективність та конкурентоспроможність сфери автобусних перевезень, зокрема процеси бронювання квитків, організації перевезень, управління автобусним парком, а також якість обслуговування пасажирів. Також вивчаються процеси діджиталізації та впровадження нових інформаційних систем для покращення маршрутів, розкладів, моніторингу та аналізу роботи автобусів з метою оптимізації перевезень до країн Європи.

Теорією дослідження є розробка веб-додатку для покращення процесів бронювання квитків та управління автобусним парком. Ця теорія передбачає вивчення принципів веб-розробки, включаючи веб-дизайн, функціональні можливості, бази даних та безпеку. Розробка веб-додатку дозволить забезпечити зручну платформу для бронювання квитків пасажиром, а також ефективно керувати автобусним парком, включаючи моніторинг рейсів, статус автобусів та аналіз ефективності.

РОЗДІЛ 1. ДІДЖИТАЛІЗАЦІЯ СФЕРИ КОМЕРЦІЙНОГО ПЕРЕВЕЗЕННЯ ЯК ОДИН ІЗ КРОКІВ ДО РОЗВИТКУ АВТОБУСНИХ ПЕРЕВЕЗЕНЬ УКРАЇНОЮ

1.1. Необхідність цифровізації процесів у сфері пасажирських перевезень

Актуальність та значимість цифровізації в сучасному світі є невід'ємною складовою розвитку різних галузей, включаючи транспортну індустрію. Швидкі технологічні зміни, постійний розвиток інформаційних технологій та зростання впливу Інтернету мають значний вплив на спосіб, яким ми здійснюємо перевезення пасажирів.

Розглянемо еволюцію продажу автобусних квитків в Україні.

1. Перші етапи: На початку розвитку автобусного транспорту в Україні, наприклад, у 1930-1940-х роках, продаж квитків відбувався на залізничних та автовокзалах. Пасажири могли придбати квитки у касах, розташованих безпосередньо на вокзалах. Також іноді квитки можна було придбати безпосередньо у водіїв або кондукторів на місцях.

2. Систематизація: З часом, у 1950-1960-х роках, з'явилися централізовані каси продажу квитків на автовокзалах. Це спрощувало процес купівлі квитків, оскільки пасажири могли звернутися до спеціальних віконць для придбання квитків на певний маршрут. Такі каси дозволяли забезпечити більшу організацію та контроль над продажами квитків.

3. Електронні системи продажу: У 1990-2000-х роках, з появою комп'ютерів та Інтернету в Україні, автобусні компанії стали використовувати електронні системи продажу квитків. Кожна компанія може створити власний веб-сайт, де пасажери можуть переглядати розклади, обирати маршрути та придбати квитки онлайн. Наприклад, компанія "Автолюкс" має свій веб-сайт, де пасажери можуть швидко та зручно придбати квитки на автобуси по Україні. [1]

4. Онлайн-платформи: З'явилися спеціалізовані онлайн-платформи, які об'єднують багато автобусних компаній та дозволяють придбати квитки на автобуси в різних напрямках по Україні. Наприклад, платформа "Busfor" є популярним сервісом для придбання квитків на автобуси, де пасажери можуть знайти розклади, обрати зручний маршрут та забронювати квиток онлайн безпосередньо на сайті платформи.

5. Мобільні додатки: У сучасний час, багато автобусних компаній розробляють мобільні додатки для зручного придбання квитків. Наприклад, "Gunsel" та "Ecolines" мають свої мобільні додатки, де пасажери можуть швидко переглянути розклади, вибрати маршрут та купити квиток безпосередньо зі своїх смартфонів.

Однією з основних переваг цифровізації в сфері пасажирського транспорту є покращення ефективності і зручності для клієнтів. Інтернет, мобільні пристрої та хмарні технології надають можливість здійснювати онлайн-бронювання квитків, отримувати актуальну інформацію про розклади руху транспорту, використовувати електронні квитки та багато іншого. Це сприяє зручності та ефективності у плануванні подорожей для пасажирів.

Додатково, цифровізація допомагає вдосконалити управління пасажирськими перевезеннями. Системи обліку, моніторингу та аналітики надають операторам транспорту можливість отримувати реальний часовий звіт про рух транспортних засобів, моніторити завантаженість маршрутів,

прогнозувати попит на послуги перевезень. Це дозволяє покращити планування маршрутів, розподіл ресурсів та оптимізацію використання транспорту.

Вплив цифрових технологій на ефективність та конкурентоспроможність пасажирського транспорту має велике значення. Введення сучасних інформаційних систем, онлайн-сервісів та мобільних додатків у галузі пасажирських перевезень дозволяє досягти численних переваг.

По-перше, цифрові технології сприяють поліпшенню ефективності операцій. Системи моніторингу та керування дозволяють операторам перевезень в реальному часі контролювати рух транспорту, виявляти перешкоди на маршруті та прогнозувати затримки. Це дозволяє забезпечити точність розкладу, підвищити пунктуальність та знизити час очікування пасажирів.

По-друге, цифрові технології створюють нові можливості для покращення якості обслуговування пасажирів. Онлайн-бронювання квитків, електронні системи оплати та мобільні додатки дозволяють зручно та швидко придбати квиток, отримати актуальну інформацію про розклади та зміни у маршрутах. Такі сервіси сприяють зручності та задоволенню потреб пасажирів, що позитивно впливає на їхню сприйнятливність до пасажирського транспорту.

По-третє, цифрові технології дозволяють операторам транспорту отримувати інформацію про попит та пасажирські преференції, що сприяє ефективному плануванню рейсів та використанню ресурсів. Аналітичні системи допомагають виявляти тренди та патерни, що дозволяє операторам адаптувати свою пропозицію та надавати послуги, які краще відповідають потребам ринку.

Отже, впровадження цифрових технологій у пасажирському транспорті впливає на підвищення ефективності та конкурентоспроможності цієї галузі. Вони допомагають покращити якість обслуговування, забезпечують точність та надійність управління, сприяють використанню даних для управлінських рішень та стратегічного розвитку. Застосування цифрових технологій є необхідним

кроком для сучасного розвитку пасажирського транспорту та задоволення потреб сучасних мобільних пасажирів.

Діджиталізація в сфері пасажирського транспорту стикається з рядом проблем і викликів, які потребують вирішення для успішної і ефективної реалізації. Деякі з цих проблем включають:

1. Інфраструктурні обмеження: Розвиток цифрових технологій вимагає наявності відповідної інфраструктури, такої як швидкий та надійний Інтернет, мобільний зв'язок та інші технічні ресурси. Недостатня інфраструктура може ускладнити впровадження цифрових рішень у пасажирському транспорті.

2. Безпека та конфіденційність даних: Використання цифрових технологій вимагає обміну та збереження великого обсягу чутливої інформації, такої як особисті дані пасажирів, фінансові дані та інша конфіденційна інформація. Забезпечення безпеки даних та захисту приватності стає важливим аспектом діджиталізації.

3. Висока вартість впровадження: Розробка та впровадження цифрових систем може бути дорогим процесом. Необхідно враховувати витрати на розробку програмного забезпечення, апаратне забезпечення, навчання персоналу та інфраструктуру. Фінансові обмеження можуть ускладнити швидкість та масштаб впровадження цифрових технологій.

4. Відновлення та адаптація: Впровадження цифрових технологій вимагає змін у робочих процесах, структурах організацій та способах взаємодії з пасажирями. Однак, не всі автобусні підприємства готові до змін і можуть зіткнутися з опором персоналу, недостатньою підготовкою або нездатністю швидко адаптуватися до нових технологій.

5. Потреба в стандартизації: Цифрові рішення в пасажирському транспорті вимагають стандартизації, щоб забезпечити сумісність та співпрацю між різними системами та платформами. Відсутність загальноприйнятих

стандартів може призвести до фрагментації ринку та ускладнити інтеграцію технологій.

Діджиталізація сфери пасажирського транспорту вимагає вирішення цих проблем і викликів шляхом розробки та впровадження ефективних стратегій, політик та технологічних рішень. Відповідні кроки повинні бути підтримані відповідною регуляторною політикою, співпрацею між галузевими гравцями та постійним моніторингом технологічних тенденцій та інновацій.

1.2. Аналіз існуючих систем обліку та продажу квитків для міжміських перевезень пасажирів

Аналіз існуючих систем обліку та продажу квитків допомагає отримати повну картину про стан ринку та визначити напрямки для подальшої розробки та вдосконалення системи обліку та продажу квитків для міжміського перевезення пасажирів.

На ринку найпоширенішими системами, які можуть допомогти розв'язати проблеми діджиталізації, зазначені вище, є:

- Passangers S3 від Sqills [2,3]
- продукти компанії Bus Logic Development [4]

Спершу розглянемо Passangers S3 від Sqills.

Sqills - це компанія, що спеціалізується на розробці програмного забезпечення для транспортних компаній, зокрема залізничного транспорту та авіакомпаній. Sqills є відомою компанією в галузі технологій для транспорту та має партнерство з різними транспортними операторами по всьому світу. Головна мета компанії це прагнення вдосконалювати транспортні системи та забезпечувати

пасажирам найкращий досвід подорожей за допомогою своїх інноваційних рішень. Продукти цієї компанії вирішують проблеми управління бронюваннями та продажем квитків, що допомагає пасажирам бронювати та оплачувати квитки через Інтернет. Їхні програмні рішення використовуються для таких цілей, як:

- Управління запасами для залізничного та автобусного транспорту
- Системи бронювання
- Системи продажу квитків
- Sqills співпрацює з низкою провідних операторів пасажирських перевезень, включаючи SNCF, Eurostar та Irish Rail.
- Наразі Sqills є частиною Siemens Mobility [2].

Розглянемо компанію **Bus Logic**. Bus Logic розробляється як об'єднання незалежних програмних модулів (підсистем), що охоплюють різні процеси роботи автобусних операторів та реселерів. Кожна частина системи може працювати самостійно, але також може бути поєднана з іншими частинами залежно від конкретних потреб кожного клієнта.

Система Bus Logic також включає автоматизовану систему збору плати за проїзд (реалізація електронних карт), яка має всі необхідні інструменти для управління даними компанії, лініями, станціями, прайс-листами та іншими. Для програмування RF-карт, компанія має апаратно-програмне забезпечення, яке дозволяє формувати та програмувати картки з використанням спеціальних протоколів безпеки під час запису даних на карту.

Bus Logic також надає рішення для пасажироінформаційної системи (PIS), яка може використовуватися у будь-якій точці світу на різних пристроях з доступом до Інтернету.

Крім цього, компанія розробляє систему автоматичного визначення місцезнаходження транспортного засобу з інтегрованими картами Google у фінальній стадії розробки. Це рішення дозволяє відстежувати рух автобусів з використанням GPS-трекінгу на консолі водія (Android-додаток для продажу

квитків у автобусі). Користувачі системи зможуть шукати громадський транспорт та маршрути (за допомогою веб-сторінки та PWA-додатку).

Bus Logic також включає модуль диспетчеризації з усіма необхідними інструментами для планування змін, маршрутів, використання транспортних засобів та іншого.

Компанія пропонує також систему бронювання, яка включає різні програмні рішення для бронювання у автобусі (Android-додаток), на придбання квитків у касі (веб-портал для ПК і Android) та онлайн-бронювання (публічний веб-сайт і PWA-додаток).

Щодо апаратної частини, компанія фокусується на розробці програмного забезпечення і вибрала надійне апаратне забезпечення від інших виробників для інтеграції зі своєю системою.

Розглянемо програмні продукти від цих компаній.

"Passengers S3" від компанії Sqills є системою управління пасажирськими перевезеннями, що забезпечує функціональність для керування процесом бронювання та продажу квитків, управління цінами та тарифами, обробки платежів та надання зручних інтерфейсів для користувачів. Це SaaS-рішення для залізничних і автобусних операторів, яке поєднує комерційну гнучкість з операційною стабільністю. "Passengers S3" полегшує процес бронювання, забезпечує зручний досвід для пасажирів, має потужні функції післяпродажного обслуговування та широкий спектр API для інтеграції з існуючими системами. Клієнти включені в дорожню карту продукту та отримують переваги від ефективної документації та навчання користувачів.

Основний продукт **Bus Logic Development** - це веб-базова цифрова програмна система з хмарною базою даних для автобусних операторів, що включає інтегровані Android-додатки для продажу квитків у автобусі, автоматизованого збору плати за проїзд, продажу квитків на автобусних станціях та через агентства, а також контролю валідності проданих квитків і електронних

карт у автобусі. Система також містить дві кросплатформенні програми для пасажирів та клієнтів, які дозволяють шукати розклад руху, бронювати та придбавати квитки онлайн, замовляти інші послуги тощо.

Сильні та слабкі сторони існуючих систем.

Проведемо SWOT аналіз продукту “Passangers S3”

Сильні сторони (Strengths):

1. Повна функціональність:

2. Управління запасами: керує запасами, бронюванням і продажем квитків;

1. Динамічне ціноутворення та розповсюдження: пропонує гнучкі варіанти ціноутворення та керує продажами в кількох каналах;

2. Можливості самообслуговування: дозволяє пасажирам повторно бронювати, скасовувати, повертати кошти, покращувати та додавати послуги.

3. Модульна архітектура:

1. Налаштування: широкі можливості налаштування для задоволення різноманітних потреб бізнесу;

2. Масштабованість: можна збільшити або зменшити масштаб відповідно до вимог оператора, підтримується хмарним хостингом AWS.

4. Досвід у галузі:

1. За десятиліття співпраці було розроблено за участю експертів із залізничного та автобусного транспорту;

2. Успішне впровадження. Перевірений досвід успішного впровадження та задоволених клієнтів, таких як Eurostar, SNCF і SJ.

5. Постійний розвиток і регулярні оновлення. Постійне вдосконалення та автоматичні оновлення забезпечують актуальність програмного забезпечення без простоїв.

6. Безпека та відповідність:

1. **Сертифікати:** Має сертифікати безпеки ISO та якості, а також відповідає стандарту PCI DSS рівня 1;

2. **Надійний SLA:** розширені угоди про рівень обслуговування гарантують підтримку та стабільність роботи.

7. Підхід, орієнтований на клієнта:

1. **Залучення клієнтів:** клієнти включені в дорожню карту продукту, отримують переваги від спеціальної документації та навчання;

2. **Підтримка 24/7:** забезпечує глобальний моніторинг і підтримку цілодобово.

Слабкі сторони (Weaknesses):

1. **Потенційна складність. Складність впровадження:** незважаючи на 100% успіх, складність системи може викликати страх у нових користувачів на початковому етапі впровадження.

2. **Врахування вартості. Загальна вартість володіння:** Хоча продається як економічно ефективна, загальна вартість володіння може бути високою для невеликих операторів, особливо з огляду на потребу в постійних оновленнях і підтримці.

3. **Вимоги до інтеграції. Залежність від API:** вимагає широкої інтеграції з існуючими системами, що може бути складним для операторів з меншою технічною інфраструктурою чи досвідом.

4. **Покладення на постійні оновлення. Керування оновленнями:** хоч і корисні безперервні оновлення, вони можуть вимагати постійного коригування та навчання, щоб користувачі були в курсі нових функцій і змін.

Можливості (Opportunities):

1. **Розширення ринку. Глобальне охоплення:** завдяки модульній та масштабованій архітектурі S3 Passenger може націлюватися на нові ринки та розширювати свою клієнтську базу на міжнародному рівні.

2. **Технологічні досягнення. Інтеграція з новими технологіями.** Існують можливості для інтеграції з такими новими технологіями, як AI та IoT, для підвищення ефективності роботи та обслуговування пасажирів.

3. **Співпраця в галузі. Партнерство з постачальниками транспортних послуг:** співпраця з постачальниками транспортних послуг і зацікавленими сторонами галузі може створити нові можливості для інновацій і розробки продуктів.

4. **Залучення клієнтів. Покращений досвід роботи з клієнтами:** використання можливостей самообслуговування та персоналізованих пропозицій може підвищити рівень задоволеності та лояльності клієнтів.

Загрози (Threats):

1. Економічні фактори:

1. **Нестабільність ринку:** економічні спади або коливання попиту на транспортування можуть вплинути на фінансову життєздатність операторів та їхні інвестиції в програмні рішення, такі як S3 Passenger.

2. **Насичення ринку:** інтенсивна конкуренція з боку існуючих гравців і нових учасників ринку може становити загрозу частці ринку.

2. Ризики безпеки:

1. **Порушення даних:** оскільки рішення SaaS обробляє конфіденційні дані пасажирів, порушення безпеки можуть призвести до репутаційної шкоди та юридичної відповідальності.

3. **Залежність від підключення до Інтернету.** Хмарні обмеження: оскільки програмне забезпечення розміщено на AWS, будь-які проблеми з підключенням до Інтернету можуть вплинути на доступ і продуктивність.

4. **Відповідність нормативним вимогам.** Зміна нормативних актів: дотримання нормативних вимог, що розвиваються, наприклад законів про захист даних і галузевих стандартів, може створити труднощі та збільшити операційні витрати.

5. Технологічний збій. Технології, що розвиваються: швидкий розвиток технологій може зробити поточні рішення застарілими або вимагати значних інвестицій, щоб залишатися конкурентоспроможними.

6. Складність інтеграції. Інтеграція із застарілими системами або сторонніми API може спричинити технічні проблеми, що призведе до затримок або проблем із сумісністю.

Тепер проведемо SWOT-аналіз продукції компанії Bus Logic Development.

Сильні сторони (Strengths):

1. Комплексне Рішення:

1. Інтегрована Система: Пропонує повне рішення для автобусних операторів, охоплюючи продаж квитків, збір оплати, відстеження GPS, управління парку і системи інформації для пасажирів.

2. Підлаштування:

1. Модульна Архітектура: Дозволяє налаштувати систему для відповідності конкретним потребам кожного клієнта, забезпечуючи гнучкість у функціональності та масштабованості.

2. Досвід та Експертиза:

1. Досвід Галузі: Розроблена у співпраці з автобусними операторами, використовуючи роки досвіду та розуміння їх вимог.

3. Широке Поширення:

1. Клієнтська База: Використовується понад 50 компаніями по всьому світу, що свідчить про довіру та надійність продукту.

4. Постійне Вдосконалення:

1. Технологічні Інновації: Тримається в ногу з технологічними досягненнями та вимогами ринку, забезпечуючи актуальність та сучасність системи.

Слабкі сторони (Weaknesses):

1. Залежність від технологій:

- **Підпорядкованість апаратному та програмному забезпеченню:**

Ефективність системи залежить від належної роботи як апаратних (наприклад, касові пристрої, дисплеї), так і програмних компонентів. Будь-які несправності або технічні проблеми можуть порушити роботу.

2. **Виклики інтеграції:**

- **Проблеми сумісності:** Інтеграція системи з існуючою інфраструктурою або системами сторонніх постачальників може призвести до труднощів, що можуть призвести до затримок або перебоїв у впровадженні.

3. **Відповідність законодавству:**

- **Правові та регуляторні вимоги:** Дотримання різноманітних нормативних актів і стандартів, пов'язаних з конфіденційністю даних, продажем квитків та транспортом, може ускладнити роботу і збільшити витрати на відповідність.

4. **Витрати:**

- **Початкові витрати:** Початкові витрати на встановлення та впровадження, а також підтримку та обслуговування, можуть бути значними для деяких операторів, особливо для менших компаній з обмеженими бюджетами.

5. **Конкурентна боротьба на ринку:**

- **Конкурентна обстановка:** Зустрічає конкуренцію від інших постачальників систем автоматизованого збору платежів, які можуть пропонувати схожі функції або займатися тими ж ринковими сегментами.

Можливості (Opportunities):

1. **Розширення ринку:**

- **Міжнародна експансія:** Завдяки універсальності та масштабованості системи, Bus Logic може розширити свою присутність на нових ринках і залучити більше клієнтів з різних країн.

2. **Впровадження нових технологій:**

- **Інтеграція з новими технологіями:** Можливість інтеграції з новими технологіями, такими як штучний інтелект (AI) та Інтернет речей (IoT), може покращити ефективність операцій та підвищити рівень задоволення пасажирів.

3. **Партнерства та співпраця:**

- **Співпраця з транспортними операторами:** Партнерство з операторами транспортних послуг та іншими зацікавленими сторонами може відкрити нові можливості для інновацій та розвитку продуктів.

4. **Розширення функціоналу:**

- **Нові сервіси:** Запропонування додаткових сервісів, таких як програми лояльності та супутні послуги, може створити нові джерела доходу.

5. **Покращення взаємодії з клієнтами:**

- **Покращення користувацького досвіду:** Впровадження можливостей самообслуговування та персоналізованих пропозицій може підвищити задоволеність та лояльність клієнтів.

Загрози (Threats):

1. **Конкурентне середовище:**

- **Конкуренція на ринку:** Сильна конкуренція з боку інших постачальників систем автоматизованого збору платежів може знизити ринкову частку та впливати на доходи.

2. **Ризики безпеки:**

- **Кіберзагрози:** Будь-які порушення безпеки або кібератаки можуть призвести до втрати довіри клієнтів та юридичних наслідків.

3. **Регуляторні зміни:**

- **Зміни в законодавстві:** Зміни у нормативних вимогах щодо захисту даних та транспортних стандартів можуть ускладнити роботу та збільшити витрати на відповідність.

4. **Технічні збої:**

- **Збої в роботі системи:** Будь-які технічні проблеми або несправності в роботі апаратного чи програмного забезпечення можуть призвести до перебоїв в обслуговуванні та незадоволення клієнтів.

5. Економічні фактори:

- **Економічна нестабільність:** Економічні спади або коливання попиту на транспортні послуги можуть вплинути на фінансову стабільність операторів та їхні інвестиції в системи автоматизованого збору платежів.

Загалом, Bus Logic Development займається розробкою програмного забезпечення для автобусних операторів, яке охоплює широкий спектр функцій, від продажу квитків до автоматизованого збору плати за проїзд та систем пасажирського обслуговування, який можна придбати від виробника і встановити “as-is”. В той час, Passangers S3 це тільки SaaS, який ще потрібно імплементувати/інтегрувати в систему, коли все інше обладнання необхідно отримувати від інших виробників.

1.3. Аналіз технологій, що можуть бути використані для розробки системи

Аналіз технологій є важливим етапом при визначенні оптимального підходу до діджиталізації. Нижче наведено переваги та обмеження деяких сучасних технологій:

1. **Хмарні рішення:** Використання хмарних рішень дозволяє зберігати дані та ресурси на віддалених серверах і мати до них доступ з будь-якого місця та пристрою. Це забезпечує гнучкість, масштабованість та високу доступність системи. Однак, важливо враховувати питання безпеки даних та надійності інфраструктури хмарних сервісів.

2. Мобільні додатки: Розробка мобільних додатків дозволяє пасажиром зручно купувати квитки та отримувати інформацію про рейси через свої смартфони. Це надає гнучкість та зручність в користуванні системою. Однак, необхідно враховувати сумісність з різними платформами (iOS, Android) та забезпечення безпеки даних, особливо при оплаті через мобільні пристрої.

3. Інтернет-платформи: Використання інтернет-платформ дозволяє пасажиром швидко та зручно купувати квитки через веб-сайт або спеціальну онлайн-платформу. Це забезпечує доступність системи та широке охоплення аудиторії. Однак, важливо забезпечити надійну систему оплати, захист персональних даних та підтримку клієнтів.

4. Використання інтернету речей (IoT): Впровадження IoT-технологій може допомогти в моніторингу руху автобусів, зборі даних про використання місць та оптимізації розкладів. Це сприяє покращенню ефективності та точності системи. Однак, необхідно вирішувати питання безпеки та захисту даних, а також забезпечувати сумісність з різними пристроями та протоколами IoT.

Крім цього, вибір технологій повинен враховувати вимоги щодо інтеграції з існуючими системами, складність реалізації та налагодження, а також фінансові аспекти. Ретельний аналіз технологій допоможе зробити обґрунтований вибір для розробки ефективної та функціональної системи обліку та продажу квитків для міжміського пасажирського транспорту.

Вибір оптимальних технологій для реалізації системи обліку та продажу квитків є критичним етапом в процесі діджиталізації. Для забезпечення успішної реалізації системи необхідно врахувати наступні аспекти:

1. Функціональні вимоги: Перш за все, необхідно визначити основні функціональні вимоги до системи обліку та продажу квитків. Це можуть бути такі функції, як онлайн-бронювання квитків, оплата через різні платіжні системи, збереження даних про пасажирів та рейси, генерація електронних квитків тощо.

2. Масштабованість: Вибір технологій, які забезпечують гнучкість та масштабованість системи. Це особливо важливо, якщо передбачається зростання обсягів перевезень та кількості клієнтів у майбутньому. Також необхідно розглянути можливість розширення системи, інтеграції з іншими модулями та платформами.

3. Безпека: Забезпечення безпеки даних є надзвичайно важливим аспектом при розробці системи обліку та продажу квитків. Необхідно врахувати можливості технологій щодо шифрування даних, захисту від несанкціонованого доступу та забезпечення цілісності і конфіденційності інформації.

4. Інтеграція: Взяти до уваги можливості технологій щодо інтеграції з існуючими системами, такими як системи оплати, бази даних пасажирів, моніторингу руху автобусів тощо. Це допоможе забезпечити плавну взаємодію між різними компонентами системи та уникнути дублювання даних.

5. Вартість: Оцініть економічні аспекти реалізації технологій. Для оцінки треба розглянути вартість впровадження, підтримки та оновлення системи, а також порівняти вартість із потенційними користями та покращеннями, які ці технології можуть принести.

Після уважного аналізу цих аспектів, обговорення з експертами та консультацій зі стейкхолдерами можна вибрати оптимальні технології, які найкраще відповідають вимогам та цілям системи обліку та продажу квитків для міжміського пасажирського транспорту.

Врахування особливостей сфери пасажирських перевезень при виборі технологій. При виборі технологій для системи обліку та продажу квитків у сфері пасажирських перевезень важливо врахувати особливості даної галузі. Деякі з основних особливостей, які слід враховувати, включають:

1. Великий обсяг даних: Пасажирський транспорт зазвичай має велику кількість рейсів і пасажирів, що призводить до значного обсягу даних, які

потрібно обробляти. Вибрані технології повинні бути здатні ефективно обробляти, зберігати та аналізувати великі обсяги даних.

2. Висока мобільність: Пасажири потребують доступу до системи обліку та продажу квитків з різних пристроїв та місць. Тому технології повинні підтримувати мобільність і забезпечувати зручний та безпечний доступ до системи через мобільні додатки або веб-інтерфейси.

3. Інтеграція з іншими системами: У сфері пасажирських перевезень часто використовуються різні системи, такі як системи оплати, моніторингу руху автобусів, бронювання готелів тощо. Технології повинні бути здатні до ефективної інтеграції з цими системами для покращення обміну даними та забезпечення злагодженої роботи.

4. Безпека даних: У зв'язку з обробкою особистої і фінансової інформації пасажирів, безпека даних є критичним аспектом. Технології повинні мати високі стандарти безпеки, такі як шифрування даних, захист від несанкціонованого доступу та використання безпечних протоколів передачі даних.

5. Підтримка клієнтського сервісу: Сфера пасажирських перевезень вимагає ефективного клієнтського сервісу, включаючи можливість швидко реагувати на запити пасажирів, забезпечення онлайн-підтримки та надання розширених можливостей бронювання та оплати квитків. Технології повинні забезпечувати високий рівень клієнтського сервісу та зручність взаємодії з системою.

Врахування цих особливостей сприятиме вибору оптимальних технологій, які забезпечать ефективну та сучасну систему обліку та продажу квитків для міжміського пасажирського транспорту.

Методики розробки системи. Існує кілька моделей розробки ІТ-проектів, які використовуються для керування та організації процесу розробки програмного забезпечення. Ось деякі з найпопулярніших моделей:

1. Водопадна модель (Waterfall Model). Це лінійна послідовна модель, в якій кожна фаза проєкту виконується послідовно після завершення попередньої. Фази включають збір вимог, аналіз, проєктування, реалізацію, тестування, впровадження та підтримку. Ця методика підходить для проєктів з чітко визначеними вимогами та стабільними змінами. [5]

2. Модель спіралі (Spiral model): Ця модель поєднує елементи водопадку та ітеративної розробки. Розробка відбувається через повторювані ітерації, кожна з яких включає фази планування, аналізу ризиків, розробки та оцінки. [6]

3. Ітеративна та інкрементна модель (Iterative and Incremental Model): Ця методика передбачає поступове покращення продукту через повторювані цикли розробки. Кожен цикл (ітерація) включає фази вимог, проєктування, реалізації та тестування. Кожна ітерація додає новий функціонал до продукту, а команда отримує зворотний зв'язок від користувачів. Цей підхід дозволяє гнучко змінювати вимоги під час розробки. [7]

4. Методологія Scrum: Scrum є гнучким підходом до розробки, оснований на ітеративних спринтах (циклах). Кожен спринт має фіксовану тривалість (зазвичай від 1 до 4 тижнів) і завершується релізом інкременту продукту. Команда розробки працює в малих ітераціях, звітуючи про свою роботу на щоденних стендап-зустрічах. [8]

5. Методологія Kanban: Kanban є системою керування робочим потоком, яка використовує дошку з колонками для візуалізації завдань і їх стану. Команда розробки переводить завдання з колонки в колонку по мірі виконання. Цей підхід дозволяє гнучко керувати пріоритетами і змінювати завдання в режимі реального часу. [9]

6. Екстремальне програмування (Extreme Programming, XP): XP - це гнучка методологія, яка ставить акцент на комунікацію, співпрацю та вдосконалення коду. Вона включає короткі ітерації, парне програмування,

тестування на ранніх етапах, часте внесення змін та взаємодію зі спеціалістами з бізнесу. [10]

Ці методики відрізняються філософією, підходом до управління проектом та організацією роботи. Вибір конкретної методики залежить від потреб, розміру команди, характеру проекту та впливу зовнішніх факторів.

РОЗДІЛ 2. АНАЛІЗ ВИМОГ ДО НОВОЇ СИСТЕМИ ПРОДАЖУ КВИТКІВ І ОБЛІКУ ТРАНСПОРТНИХ ЗАСОБІВ ПІДПРИЄМСТВА

2.1 Визначення вимог до системи

У даному розділі проведений аналіз вимог до системи онлайн-бронювання автобусних квитків. Аналіз вимог є важливим етапом проєктування системи, оскільки визначає функціональність, характеристики та обмеження, що мають бути враховані при розробці продукту.

Розглянемо основні вимоги до системи. Визначено, що в системі повинно бути чотири основних компоненти:

1. Компонент обліку маршрутів та поїздок: Цей компонент відповідає за облік і керування маршрутами, їх розкладами та поїздками. Він забезпечує створення та збереження маршрутів у базі даних, оновлення та видалення інформації про маршрути та їх розклади, а також формування звітів і статистики щодо маршрутів та поїздок.

2. Компонент по формуванню, продажу та збереженню квитка: Цей компонент відповідає за функції, пов'язані з продажем та обробкою квитків. Він забезпечує формування квитків на основі вхідних даних, таких як маршрут, дата та час поїздки, збереження інформації про продані квитки у базі даних, а також обробку платежів та інтеграцію з платіжними системами, наприклад, PayPal або криптовалютою.

3. Компонент обліку автобусів та водіїв: Цей компонент відповідає за облік транспортних засобів та водіїв. Він включає реєстрацію та збереження інформації про автобуси та водіїв у базі даних, оновлення та видалення інформації

про автобуси та водіїв, а також моніторинг стану автобусів, включаючи їх доступність та розклад руху.

4. Компонент для роботи з користувачами. Цей компонент відповідатиме за такі функції, як вхід в систему, реєстрація, а також перегляд інформації користувача. Якщо користувач є пасажиром, то він може переглядати придбані квитки. У разі, якщо користувач є водієм, він може переглядати інформацію про поїздки, до яких він прикріплений.

Ці функціональні компоненти визначаються як важливі логічні частини системи, які виконують конкретні функції. Під час подальшої розробки системи вони будуть детальніше визначені, включаючи їх взаємодію та інтерфейси між ними.

2.2 Аналіз функціональних та нефункціональних вимог

Функціональні вимоги описують, які функції та можливості повинна мати система. Вони визначають, яким чином система повинна взаємодіяти з користувачами та іншими системами, які операції повинні бути виконані та які дані повинні бути оброблені. [11] Під час аналізу функціональних вимог проведено розбиття системи на модулі та визначено їх функціональне поведінка. Основним результатом аналізу функціональних вимог є створення детального опису функціональних можливостей системи.

Однією з основних функцій нової системи є **продаж квитків** для міжміського перевезення пасажирів. Для забезпечення зручності та ефективності процесу продажу, система повинна володіти наступними функціональними вимогами:

- Розклад рейсів та наявність квитків: Система повинна надати користувачам можливість переглядати актуальний розклад рейсів та інформацію про наявність квитків на кожен рейс.
- Онлайн-замовлення та придбання квитків: Система повинна дозволяти користувачам здійснювати онлайн-замовлення та придбання квитків. Це включає можливість вибору бажаного рейсу, типу квитка та кількості квитків.
- Вибір місця або типу квитка: Система повинна дозволяти користувачам вибирати місце та тип квитка, якщо це застосовно до конкретного рейсу. Наприклад, користувачі можуть вибрати місце біля вікна, проходу або комфорт-класу.
- Підтримка різних способів оплати: Система повинна надати можливість користувачам обирати різні способи оплати, такі як кредитні картки, електронні платіжні системи тощо. Необхідно забезпечити безпеку операцій оплати та захист особистих даних користувачів.
- Автоматичне видання квитків та електронних підтверджень: Після успішного придбання квитків, система повинна автоматично генерувати квитки та електронні підтвердження. Це забезпечить користувачам зручність та швидкість отримання необхідної інформації.

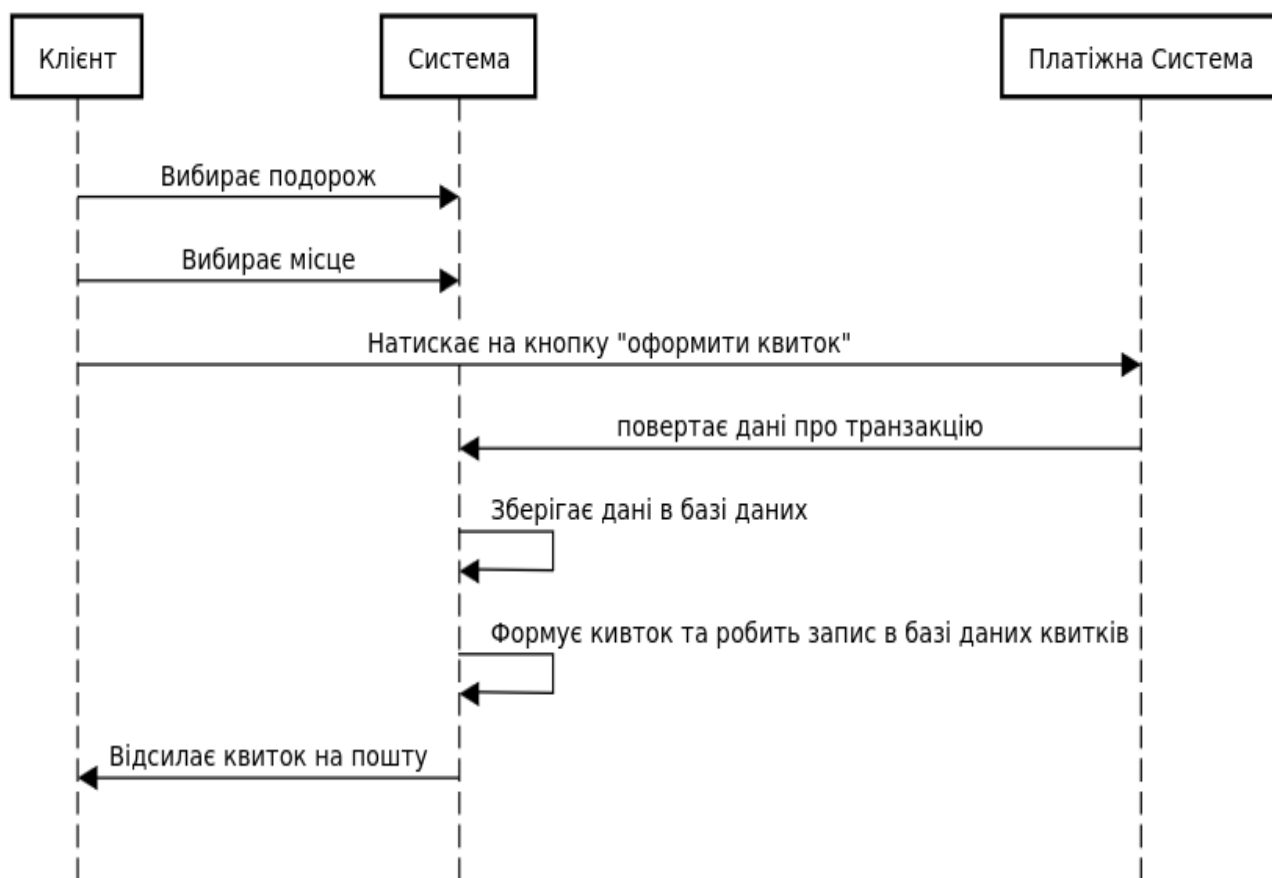


Рис 2.1 UML діаграма послідовності успішного придбання квитка

Джерело: розроблено автором

Підприємство має необхідність в ефективному обліку свого автобусного парку та пов'язаних з ними операцій. Нова **система обліку автобусів** повинна володіти наступними функціональними вимогами:

- Збір та збереження інформації про автобуси. Система повинна забезпечувати облік інформації про транспортні засоби, включаючи модель, реєстраційний номер, рік випуску, максимальну потужність тощо. Це дозволить здійснювати ефективний контроль та управління автобусами.
- Відстеження рейсів автобусів. Система повинна автоматично відстежувати рейси автобусів та реєструвати час відправлення та прибуття кожного рейсу. Це дозволить точно визначати тривалість кожного рейсу та вести облік затримок або незапланованих зупинок.
- Моніторинг роботи автобусів: Система повинна забезпечувати моніторинг роботи автобусів, включаючи кілометраж, споживання палива та інші

показники. Це дозволить вести ефективний облік ресурсів, проводити плановий технічний огляд та планувати обслуговування автобусів.

- Система повинна забезпечувати аналіз та звітність про роботу автобусів. Це включає підготовку звітів про використання ресурсів, ефективність планування маршрутів, витрати на паливо та інші показники. Звіти допоможуть управлінцям приймати обґрунтовані рішення щодо оптимізації діяльності автопарку.

Для забезпечення ефективності та оптимального планування міжміських перевезень пасажирів, нова **система** повинна володіти наступними функціональними вимогами щодо **створення маршрутів**:

- Аналіз попиту та пасажирських потоків: Система повинна здійснювати аналіз попиту та пасажирських потоків для визначення найбільш популярних маршрутів. Це дозволить ефективно розподіляти ресурси та планувати маршрути з врахуванням потреб пасажирів.

- Оптимальне планування маршрутів: Система повинна допомагати в оптимальному плануванні маршрутів, враховуючи різні фактори, такі як відстань, час, наявність автобусів та потреби пасажирів. Вона повинна забезпечувати планування маршруту слідування та створення рейсів.

- Система повинна бути гнучкою та адаптивною, щоб враховувати зміни в попиті, маршрутах або розкладах. Вона повинна дозволяти швидко внести зміни в маршрути та розклади, відповідаючи потребам пасажирів та оптимізуючи використання ресурсів.

- Інтеграція з іншими системами: Система повинна бути здатною інтегруватися з іншими системами, такими як система продажу квитків та система обліку автобусів. Це забезпечить цілісність та синхронізацію даних для ефективного планування та виконання маршрутів.

Для забезпечення зручності та доступності для пасажирів, нова система повинна мати наступні функціональні вимоги щодо **онлайн-придбання квитків**:

- Веб-інтерфейс користувача: Система повинна мати зручний та інтуїтивно зрозумілий веб-інтерфейс для користувачів, який дозволить їм швидко та легко здійснювати придбання квитків. Інтерфейс повинен бути привабливим, зручним у використанні та добре структурованим.
- Пошук та вибір маршрутів: Система повинна надавати можливість користувачам швидко знайти та вибрати потрібний маршрут. Вона повинна мати функцію пошуку за місцем відправлення та прибуття, датою, часом та іншими параметрами, що дозволять знайти оптимальний варіант.
- Онлайн-оплата: Система повинна мати можливість безпечно та зручно проводити онлайн-оплату за квитки. Вона повинна підтримувати різні методи оплати, такі як кредитні картки, електронні платіжні системи тощо, забезпечуючи безпеку та захист конфіденційної інформації користувачів.
- Електронні квитки та підтвердження: Після успішного придбання квитка, система повинна автоматично видавати електронний квиток або підтвердження. Користувачі повинні мати можливість зберігати цю інформацію на мобільних пристроях або друкувати її для використання під час поїздки.
- Керування бронюваннями та змінами: Система повинна дозволяти користувачам керувати своїми бронюваннями, вносити зміни у дату, час або маршрут поїздки, а також скасовувати бронювання, якщо потрібно. Це забезпечить гнучкість та зручність для пасажирів.

Нефункціональні вимоги (також відомі як технічні вимоги) відносяться до характеристик, якостей або обмежень системи, які не стосуються прямої функціональності програмного продукту, але мають важливе значення для його успішного функціонування. [11] Основна відмінність між функціональними та нефункціональними вимогами полягає в тому, що функціональні вимоги описують, що програма повинна робити, тоді як нефункціональні вимоги описують, як програма повинна працювати.

Зважаючи на обмежений бюджет, система повинна бути розроблена з урахуванням економічності. Необхідно забезпечити оптимальну вартість розробки, впровадження та підтримки системи.

Оскільки на початковому етапі не передбачається значний наплив користувачів, система повинна бути готова працювати з обмеженим навантаженням. Необхідно розробити архітектуру, яка забезпечить стабільну та ефективну роботу системи під час передбачуваних навантажень.

Щодо швидкості відповіді, система повинна забезпечувати швидку відповідь на запити користувачів. Час відповіді системи повинен бути менше ніж пів секунди, щоб забезпечити комфортну роботу та задоволення користувачів.

Однією з ключових вимог є надійність системи. Система повинна працювати стабільно та надійно, мінімізуючи можливість відмов та збоїв. Необхідно передбачити механізми відновлення після випадків непередбачуваних ситуацій.

Забезпечення безпеки є важливим аспектом системи. Вона повинна забезпечувати захист персональних даних пасажирів, включаючи механізми автентифікації та авторизації користувачів. Також варто враховувати заходи захисту від вторгнень та несанкціонованого доступу до системи.

А про масштабування системи варто думати на пізніших етапах розвитку проєкту. Але незважаючи на те, що на початку не передбачається великий наплив користувачів, система повинна бути готовою до розширення та масштабування в разі зростання навантаження.

2.3 Формулювання вимог до архітектури системи

Формулювання вимог до архітектури системи ґрунтується на загальних цілях та функціональних можливостях, визначених раніше і полягає в тому, щоб чітко окреслити технічні аспекти, необхідні для створення стійкої, масштабованої та безпечної системи, яка відповідає поточним та майбутнім потребам. Відповідь на наступні питання нададуть змогу сформулювати вимоги до архітектури системи:

- Які будуть основні модулі або сервіси, що складатимуть систему?
- Як будуть забезпечені вимоги до швидкості та продуктивності системи?
- Які будуть вимоги до масштабованості системи, особливо з урахуванням майбутнього росту та збільшення обсягу продажів квитків та обліку автобусів?
- Які будуть вимоги до надійності та стійкості системи, щоб уникнути втрати даних або перебоїв у роботі?
- Які будуть вимоги до безпеки системи, зокрема щодо захисту персональних даних пасажирів та забезпечення автентифікації та авторизації користувачів?
- Які будуть вимоги до інтеграції з платіжними системами, такими як PayPal або криптовалюта?

Основні модулі, що складатимуть систему — середовище збереження інформації, сервіс з бізнес-логікою, модуль відображення стану.

Оптимальний підхід до забезпечення швидкості відповіді системи полягає у використанні профайлера для виявлення та усунення факторів, що впливають на продуктивність. Аналіз результатів профайлінгу дозволяє ідентифікувати участки

коду або процесів, де спостерігаються затримки або надмірне використання ресурсів. На основі цих висновків можна прийняти відповідні заходи для оптимізації та покращення продуктивності системи.

Крім того, варто звернути увагу на кращі практики програмування та проєктування, що спрямовані на досягнення ефективності та швидкості відповіді системи. Використання оптимізованих алгоритмів, кешування даних, паралелізації та асинхронного програмування, а також оптимізація запитів до бази даних можуть суттєво покращити швидкість роботи системи.

Необхідно також врахувати вимоги до апаратного забезпечення та мережевої інфраструктури. Забезпечення достатнім рівнем обчислювальних потужностей, швидкісними з'єднаннями та оптимізація мережевої комунікації можуть допомогти досягти бажаної швидкості відповіді системи.

Ретельне тестування та профілактичне моніторингове спостереження дозволять виявити та виправити можливі проблеми з продуктивністю системи. Налаштування та оптимізація критичних частин коду, а також впровадження кешування та асинхронного виконання можуть позитивно позначитися на швидкості роботи системи.

При використанні монолітної архітектури дійсно можна досягти простоти у розробці та підтримці системи, оскільки немає потреби управляти декількома окремими модулями або сервісами.

Згідно з планом розвитку, клієнт може вибрати більш потужний та дорожчий план у провайдера хостингу, якщо з'явиться потреба у більшому обсязі ресурсів. Це може бути розумним рішенням на початкових етапах розвитку системи, коли попит на ресурси ще невеликий.

У майбутньому, коли попит на систему зростатиме, може бути розглянута можливість переходу до більш розподіленої архітектури, яка дозволить горизонтально масштабувати окремі компоненти системи та підвищити її масштабованість та надійність.

Монолітна архітектура може бути прийнятним рішенням на початкових етапах, але в подальшому слід ретельно виважити потреби у масштабованості та шукати оптимальні шляхи для забезпечення високої продуктивності та доступності системи. Тож в майбутньому відповідальність за масштабування та оптимізацію системи перейде до досвідченого архітектора. Звісно, коли компанія матиме ресурси та необхідність у збільшенні продуктивності окремих модулів або компонентів, викликати експерта з архітектури буде розумним рішенням.

Професійний архітектор зможе розібрати монолітний код на окремі мікросервіси та забезпечити їх балансування для досягнення більшої продуктивності та ефективності. Він буде враховувати потреби системи, обмеження ресурсів та вимоги до продуктивності для розробки оптимальної архітектури, що забезпечить високу продуктивність та масштабованість системи.

Таким чином, ми можемо спокійно розробляти монолітну архітектуру в даний момент, знайомлячись з реаліями своєї компанії та відповідаючи на поточні вимоги. У майбутньому, коли з'явиться потреба у масштабуванні та оптимізації, клієнт може звернутись до професіонала з архітектури, який зможе ефективно перетворити монолітний код на мікросервіси та забезпечити балансування та оптимізацію кожного з них.

Такий підхід до масштабування та оптимізації дозволить забезпечити високу продуктивність та гнучкість системи відповідно до зростаючих потреб компанії.

Для забезпечення надійності та стійкості системи рекомендується врахувати деякі важливі аспекти:

1. Резервне копіювання та відновлення даних: Необхідно буде розробити стратегію резервного копіювання, яка забезпечує регулярне створення резервних копій даних і зберігання їх в безпечному місці. Також переглянуть, чи можна відновити дані з резервних копій та як цей процес виконується.

2. **Забезпечення безпеки даних:** Розглянуть і застосувати заходи для захисту конфіденційності та цілісності даних. Також, розробити механізми автентифікації та авторизації користувачів, щоб забезпечити обмежений доступ до системи. Застосувати шифрування для захисту персональних даних пасажирів та інших конфіденційних інформаційних ресурсів.

3. **Захист від вторгнень:** Розробити систему, яка містить заходи безпеки для запобігання вторгненням, включаючи використання брандмауерів, систем виявлення вторгнень та оновлення програмного забезпечення з метою усунення вразливостей.

4. **Моніторинг та логування подій:** Забезпечити систему моніторингу та логування подій, яка дозволить відстежувати важливі операції, помилки та події, що виникають у системі. Це допоможе швидко виявляти й усувати проблеми, а також забезпечить аудит системних дій.

5. **Резервування ресурсів:** Розглянуть і застосовувати стратегію резервування ресурсів, таких як мережеве з'єднання, серверне обладнання та інші критичні компоненти. Це дозволить уникнути однотипних одночасних відмов і забезпечить продовження роботи системи при випадкових відмовах.

6. **Тестування та валідація:** Провести регулярне тестування системи, включаючи функціональні та навантажувальні тести, щоб виявити можливі проблеми та перевірити роботу системи при великому навантаженні. Розробити правила валідації вхідних даних, щоб запобігти атакам типу "ін'єкція" або "скриптінг".

7. **Відновлення системи:** Розробити план відновлення системи, який включає процедури відновлення після відмови, а також резервні ресурси та резервні розташування для неперервності роботи.

Інтеграція з платіжною системою є важливою частиною розробки системи продажу квитків. У даному випадку, планується підключення платіжних методів PayPal та криптовалюти.

Інтеграція з PayPal дозволить користувачам зручно та безпечно здійснювати оплату за придбання квитків. Для цього потрібно налаштувати відповідний інтерфейс для зв'язку з API PayPal та обробки платежів.

Щодо інтеграції з криптовалютою, необхідно визначити конкретні криптовалюти, якими користувачі зможуть здійснювати оплату. Для цього необхідно вивчити технічні аспекти взаємодії з відповідними блокчейн-платформами та забезпечити безпечний обмін даними та операцій з криптовалютою.

Важливо ретельно прорахувати інтеграцію з цими платіжними системами, забезпечити безпеку та надійність обробки платежів, а також зручний та інтуїтивно зрозумілий інтерфейс для користувачів, щоб забезпечити зручність процесу оплати квитків.

2.4 Формулювання вимог до інтерфейсу користувача

Щодо вимог до зовнішнього вигляду інтерфейсу веб-сторінки, пропонується наступне:

1. Використати кольорову палітру та принцип UI/UX “60-30-10”: Білий колір як основний фоновий колір для сторінки з метою створення світлого та чистого вигляду. Фіолетовий-в-рожевий градієнт для деяких елементів, щоб надати візуального акценту та розширити палітру кольорів. Світлозелений колір як акцентний колір для певних елементів та кнопок. [13]

2. Використати рядок пошуку: Виділений рядок пошуку поїздок у верхній частині сторінки, щоб забезпечити зручність користування. Він повинен дозволяти вводити місце походження, місце призначення, дати та інші параметри для зручного пошуку.

3. Розташувати результати пошуку у вигляді карток: Результати пошуку у вигляді горизонтального скролу карток, де кожна карта відображає окрему поїздку. Картки мають містити важливу інформацію про маршрут, час відправлення, час прибуття, тривалість та ціну. Користувачі повинні мати можливість прокручувати картки за допомогою горизонтальної стрілки або перетягування, щоб переглянути додаткову інформацію.

4. Міжнародна підтримка: Система повинна бути готовою до інтернаціоналізації та локалізації для різних мов та культур. Початково планується підтримка англійської та української мов, з можливістю легкого додавання інших мов у майбутньому.

5. Інтуїтивний інтерфейс: Система повинна мати простий та легкий у використанні інтерфейс. Користувачі повинні швидко розуміти, як взаємодіяти з системою, здійснювати необхідні дії та отримувати необхідну інформацію без зайвих зусиль.

6. Адаптивний дизайн: Дизайн системи повинен бути адаптивним до різних пристроїв та розмірів екранів. Це забезпечить зручну роботу користувачів на різних пристроях, включаючи мобільні телефони, планшети та комп'ютери.

7. Швидкість та продуктивність: Система повинна працювати швидко та ефективно. Анімації не повинні навантажувати систему, особливо мобільні пристрої.

РОЗДІЛ 3. РОЗРОБКА СИСТЕМИ ТА ЇЇ ВПРОВАДЖЕННЯ

3.1 Вибір технологій та інструментів для розробки системи

Для успішної розробки системи обліку та продажу квитків було проведено аналіз різних технологій та інструментів, що найкраще відповідають потребам проекту. На основі цього аналізу були обрані наступні технології:

1. Мова програмування Java (OpenJDK 17): Вибір Java обумовлений її багатим функціоналом, платформною незалежністю та великою спільнотою розробників. [14] Java дозволяє побудувати стабільну та надійну систему з широкими можливостями масштабування.

2. Spring Framework 5: Spring є потужним фреймворком для розробки веб-додатків на основі Java. [15] Використання Spring допоможе спростити розробку, забезпечить інтеграцію з базою даних та забезпечить ефективне керування компонентами системи.

3. PostgreSQL: В якості бази даних була обрана PostgreSQL з врахуванням її високої надійності, масштабованості та відкритого джерела. [16] PostgreSQL забезпечить збереження даних про квитки, користувачів та іншу інформацію, необхідну для системи.

Вибір цих технологій базується на їхній надійності, широкому сприйнятті у розробницькій спільноті, підтримці та розширенні можливостей системи. Ці технології відповідають вимогам проекту з точки зору швидкодії, безпеки, масштабованості та розширюваності.

Зазначені технології стануть основою для розробки системи обліку та продажу квитків, дозволяючи забезпечити потрібну функціональність, ефективність та надійність системи.

Додатково до вже обраних технологій, планується використовувати наступні інструменти та сервіси:

1. Apache Tomcat 9: Apache Tomcat є контейнером веб-додатків для Java і дозволить розгорнути систему на сервері. [17]
2. GitLab: GitLab є системою керування версіями та спільною розробкою програмного забезпечення, яка допоможе керувати та відстежувати зміни у проєкті. [18]
3. Eclipse IDE: Eclipse є популярною інтегрованою середовищем розробки Java, яке забезпечує багато функціональних можливостей та підтримку роботи з Java-проєктами. [19]
4. pgAdmin: pgAdmin є графічним інтерфейсом для керування базами даних PostgreSQL, який допоможе зручно адмініструвати базу даних. [20]
5. Visual Studio Code (VSCode): VSCode є легким та потужним редактором коду, який підтримує розширення для роботи з JSP-файлами та допоможе з розробкою фронтенд-частини системи. [21]
6. VPS сервер від HyperHost.ua: Плануємо розмістити сайт на VPS рішенні від HyperHost.ua, конкретно на тарифі "VPS Mini". [22] Це забезпечить незалежність та контроль над середовищем розгортання.

Ці додаткові інструменти та сервіси значно спростять розробку, керування версіями, адміністрування бази даних та розгортання системи на сервері.

У наступних підрозділах буде детальніше розглянуто проєктування бази даних та розробку функціональності системи на основі вибраних технологій.

Щодо методології розробки — звітність по розробці буде фіксуватися в Atlassian JIRA, система відстеження помилок, призначена для організації спілкування з користувачами, і для управління проєктами. [23]

Перелік питань, які можуть бути вирішені за допомогою Jira:

1. Управління задачами: Jira дозволяє створювати, призначати, відстежувати та оновлювати задачі. Ви можете створювати задачі для конкретних розробників, присвоювати їм пріоритети, встановлювати дедлайни та контролювати стан виконання.

2. Стеження за прогресом розробки: Jira надає зручні засоби для стеження за прогресом проєкту. Ви можете створювати дошки (дошки Kanban або дошки Scrum), переглядати завдання в різних статусах, візуалізувати потік роботи та отримувати звіти про продуктивність команди.

3. Управління вимогами: Jira дозволяє відстежувати вимоги до прогварамного забезпечення, зберігати їх, описувати та пов'язувати зі створеними задачами. Ви можете створювати епіки, історії користувачів та інші форми вимог, а також простежувати стан їх виконання.

4. Управління багами: Jira дозволяє створювати та відстежувати баги або проблеми, що виникають під час розробки. Ви можете описувати проблеми, призначати їх відповідним розробникам, присвоювати пріоритети та відстежувати їх виправлення.

5. Спілкування та співпраця: Jira надає засоби для спілкування та співпраці між членами команди. Ви можете коментувати задачі, обговорювати питання, обмінюватися ідеями та документацією, використовуючи функції коментарів та прикріплення файлів.

6. Звітність та аналітика: Jira надає можливості для створення звітів та аналітики про прогрес проєкту, продуктивність команди, терміни виконання,

якість та інші показники. Ви можете використовувати ці дані для оцінки ефективності та прийняття управлінських рішень.

Команда розробників складатиметься з однієї людини — доповідача. Активність зі сторони замовника мінімальна і кардинальних правок не передбачається, тому можна використати найпростішу методологію яку можна запропонувати це Waterfall, але для підтримки контакту і періодичної звітності з замовником краще використати спринти в 2 тижні. Цей підхід надає методологія Scrum. Але для зручності управління потоком завдань краще використати дошку з методології Kanban. Тому данна методологія є гібридом між Scrum і Kanban. Нащастя, Jira надає можливість працювати в цьому гібриді. Також, якщо в майбутньому команда буде розширюватись, Jira дозволить організувати роботу на всіх учасників розробки.

3.2 проєктування бази даних, виділення об'єктів предметної області, створення власних структур даних

З розділу 2.2 данної роботи аналіз функціональних вимог показав що ми маємо 4 системи:

- Система продажу квитків
- Система обліку автобусів
- Система створення маршрутів
- Онлайн-придбання квитків

В об'єктно-орієнтованому підході до розробки системи продажу квитків для міжміського перевезення пасажирів з кожної системи можна виділити основні предмети об'єктної області.

З системи продажу квитків можна виділити такі об'єкти:

- Квиток
- Пасажир
- Подорож
- Результат про оплату.

З системи обліку автобусів можна виділити наступні об'єкти:

- Автобус
- Рейс

З системи створення маршрутів можна виділити такі об'єкти:

- Маршрут
- Подорож
- Водій

Якщо згрупуємо і об'єднаємо всі об'єкти які описані вище, то матимемо наступне:

- Квиток
- Звіт про оплату
- Подорож
- Транспорт
- Користувач
- Маршрут
- Місто
- Країна

Отже ми має 8 об'єктів предметної області. Після детального аналізу цих об'єктів, маємо опис кожних з них:

- **Квиток** повинен містити інформацію про пасажера, покупця й придбання, подорож, місце, ідентифікатор в системі.
- **Звіт про оплату** міститиме в собі все що прийшло у відповідь після оплати від сервісу Пейпал, очікується що у відповіді міститься інформація про покупця, ціна і валюта операції а також інформація про самий квиток.
- Об'єкт **подорожі** матиме інформацію про маршрут, транспорт і водія, дату та час відправлення, кількість місць і їх ціну а також стан подорожі.
- Об'єкт **транспорту** характеризуватиметься характерними будь-якому транспорту описом, а саме — марка та модель, колір, номерний знак, схема сидінь і їх кількість, а також ідентифікатор в системі.
- Об'єкт **користувача** повинен містити ідентифікатор в системі, роль користувача, прізвище та ім'я, пошта, логін та пароль.
- Завершуючи об'єктами **маршруту**, **містом** та **країною**, то маршрут містить власне назву маршруту, а також місто відправлення і прибуття; місто має ідентифікатор в системі, власне назву, і країну; країна має ідентифікатор та назву.

З даними об'єктами предметної області і їх детальним описом можна побудувати UML Class Diagram (Діаграму Класів). [24] Це робиться для кращого розуміння розташування об'єктів і зв'язків між ними.

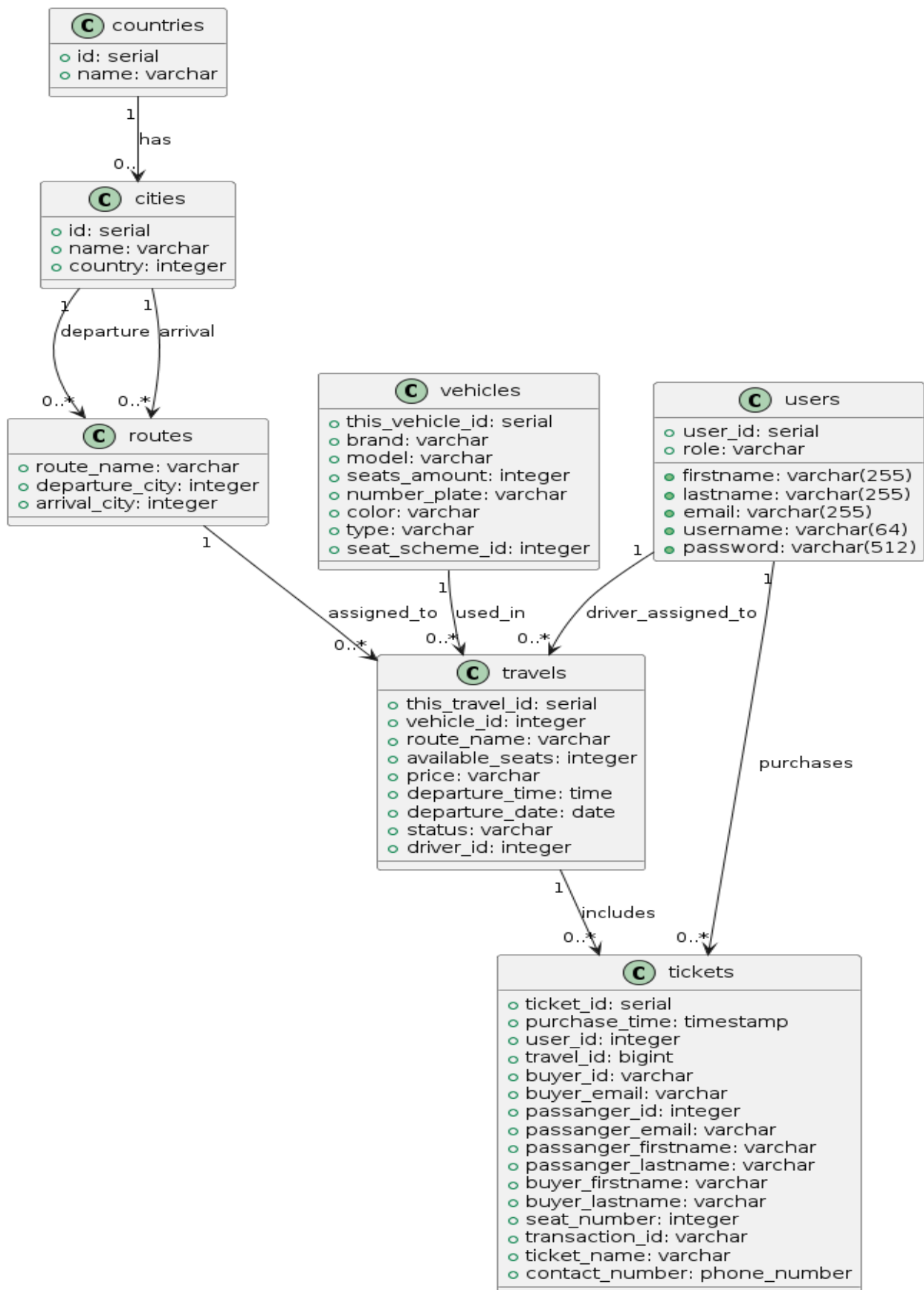


Рисунок 3.1 - UML Class Diagram предметної області

Джерело: розроблено автором на основі[25]

При детальному розгляді рисунку 3.1 можна задатися питанням — чому об'єкт маршрут не має поля ідентифікації? В процесі початкового планування з'ясувалося, що маршрут між містами є завжди один і унікальний. Тож за ідентифікацію маршруту відповідатиме назва самого маршруту, яка, в свою чергу, являє собою текстовим полем, що включає в себе назву міста відправлення, пробіл, тире, пробіл і міста прибуття. У майбутньому це може призвести до певних ускладнень, але наразі це є прийнятним рішенням.

Також в об'єкті транспорту наявне поле `seat_scheme_id` з типом даних `integer`, яке являє собою ідентифікатор схеми сидінь, тому можна логічно припустити що повинно десь бути й сама схема сидінь проте ніде на рисунку 3.1 немає пояснення де вона. Проблема в тому що немає `primary` типу даних для схеми сидінь, і воно потребує детального аналізу.

До прикладу, розглянемо Mercedes-Benz Sprinter, а конкретніше варіант 2023 L.A. West Sprinter Mercedes 13 Passenger Luxury Bus. [26] На ресурсі є схема підлоги(Floorplan) даного автобуса.

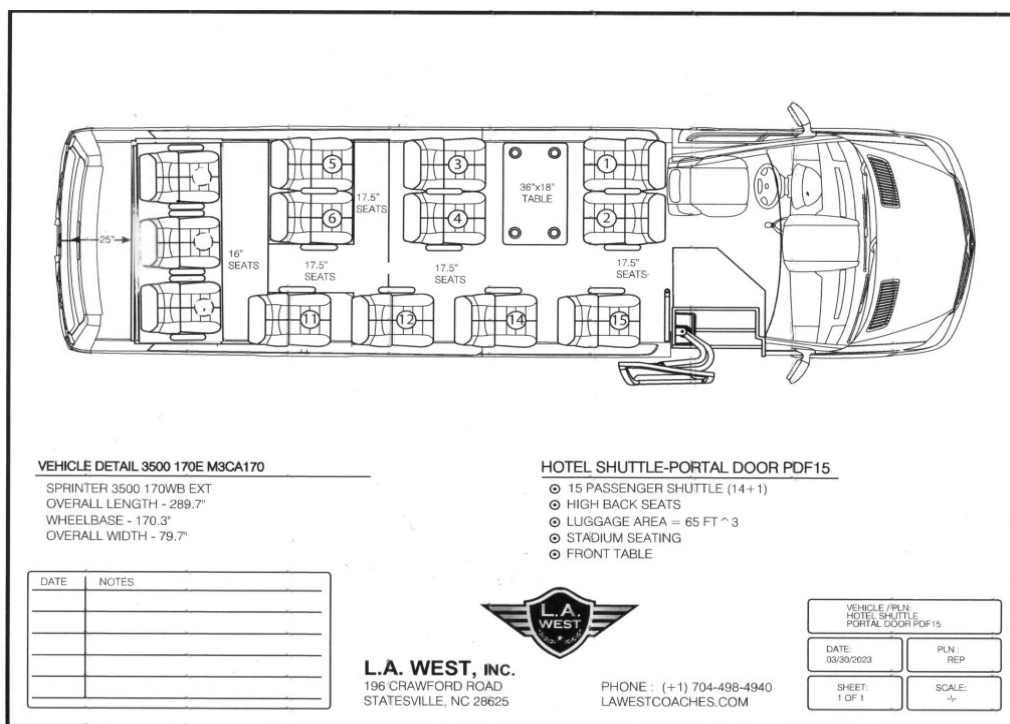


Рисунок 3.2 Схема підлоги автобуса 2023 L.A. West Sprinter Mercedes 13 Passenger Luxury Bus

Джерело: [26]

Якщо уявити квадратну сітку там де сидіння, прохід і двері автобуса то вийде сітка 4х6.

M7	M5	M3	С	M1	В
M8	M6	M4	С	M2	П
M9	П	П	П	П	
	M10	M11	M12	M13	Д

Рисунок 3.3 План підлоги автобуса у вигляді таблиці

Джерело: розроблено автором на основі [рисунок 3.2]

На рисунку 3.3 В позначає місце водія, Д - двері, M1-M13 місця для сидіння, С - столик, П - прохід

Запишемо цю схему у форматі JSON, по ряду сидінь починаючи з водія, і додамо мета-інформацію про схему.

- `passenger_seats_amount` (ціле число): загальна кількість доступних пасажирських місць.
- `total_elements` (ціле число): загальна кількість елементів у схемі, включаючи місця, вхід, водія та столи.
- `max_in_row` (ціле число): максимальна кількість елементів в будь-якому окремому рядку.
- `scheme` (масив): масив, що представляє розташування місць.
 - `row` (масив рядків): масив, що представляє кожен ряд у розкладі місць.

Кожен елемент може бути:

- `driver`: місце водія.
- `way`: прохід.
- `entrance`: Вхід до автобуса.
- `seatX` (де X — число): пасажирське сидіння.
- `table`: стіл.

Так як автопарк на початках буде не великий то ця схема буде зберігатися як .json файл в структурі проєкту і матиме назву busSeatScheme{id}.json, де {id} це ідентифікатор схеми. [Додаток А]

Перейдемо до створення власне бази даних, в якій ми створимо вищевказані об'єкти. За допомогою pgAdmin-а створюємо базу даних [27], потім в станадртній публічній схемі створюємо таблиці [28] як на рисунку 3.1. Також створимо первинні і зовнішні ключі для полів ідентифікації. Коли таблиці створені можна подивитися на SQL-скрипт їхнього створення. Отримавши скрипти створення всіх таблиць і скомпонувавши їх в порядку від простої таблиці країн до складних таблиць квитків, ми отримаємо повний скрипт створення таблиць. [Додаток Б]

3.3 Розробка функціональності системи, тестування та налагодження системи

Використання Spring MVC відповідає за відображення даних та отримання їх від користувача. Тримаючися трьохшарової архітектури, було розподілено функціональність додатка на різні шари.

1. Шар відображення (Presentation Layer або Controllers): Цей шар включає контролери, які приймають запити від користувача, взаємодіють з моделями та передають результати або як дані у форматі JSON, XML та інші, або як відображення на вигляд. Контролери забезпечують обробку запитів, валідацію даних та виконання відповідних операцій.

2. Шар бізнес логіки (Business Layer): У цьому шарі знаходяться ваші сервіси, та об'єкти предметної області, які являють собою бізнес-логіку додатка. Вони виконують обчислення, обробку даних та виконання бізнес-процесів.

Сервіси отримують дані з контролерів або репозиторіїв, виконують потрібні операції та повертають результати назад.

3. Шар доступу до бази даних (Data Access Layer або Repositories): Цей шар включає репозиторії, які забезпечують доступ до бази даних та виконання операцій збереження, оновлення, видалення та отримання даних. Репозиторії використовуються сервісами для отримання необхідної інформації з бази даних.

Ця архітектура дозволяє забезпечити розділення відповідальностей між різними компонентами додатка, спрощує тестування, розширення та обслуговування системи. Крім того, використання Simple Domain Model сприяє збереженню об'єктів без поведінки, а функціональність логіки розміщується в сервісних компонентах.

Розглянемо UML діаграму послідовностей яка була зазначена в другому розділі.

Першими двома пунктами є дві дії від користувача системи — дія “Вибрати подорож” та дія “Вибрати місце”.

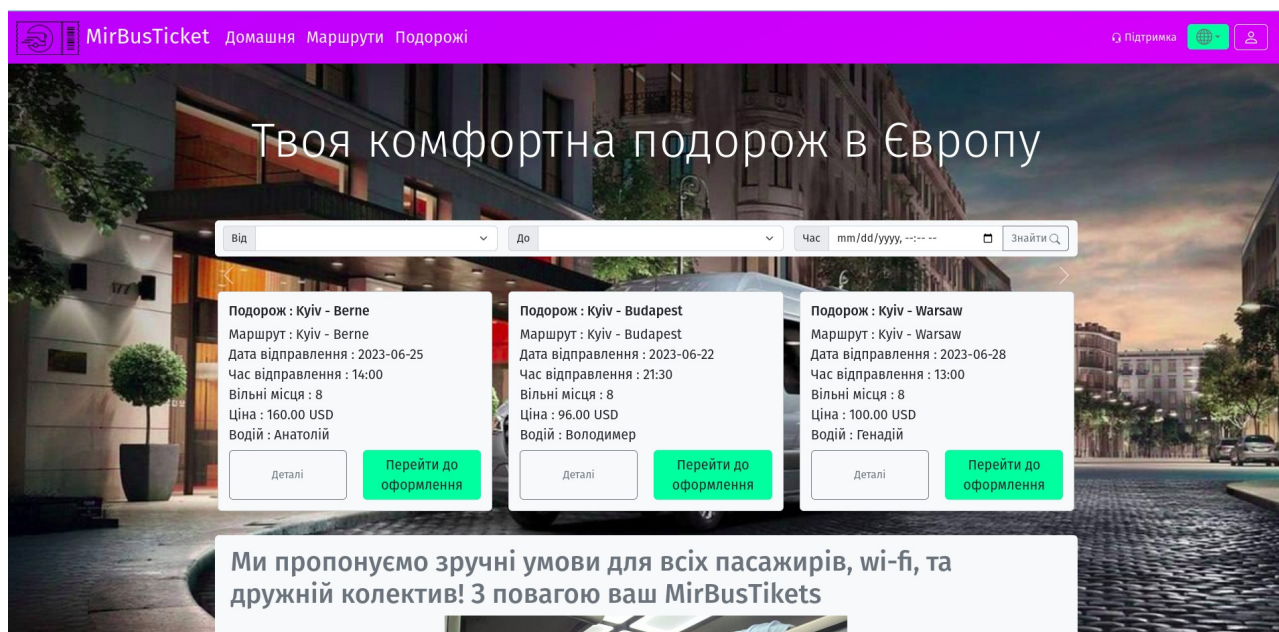


Рисунок 3.4 - Головна сторінка сайту

Джерело: розроблено автором

Користувача зустрічає головна сторінка, побудована за допомогою CSS фреймворку Bootstrap. Кольори були підібрані ще в розділі 2.4. Потім він переходить від головної сторінки до сторінки вибору місця, вибравши певну подорож. Якщо користувач не зайшов в свій кабінет то система відведе користувача на сторінку логіну. Після авторизації, користувач має змогу оформити квиток, коли проведе оплату через PayPal Checkout. Потім, після успішного завершення транзакції, користувач має змогу переглянути свій квиток на сторінці “Придбані квитки”. Відобразимо це кодом програми.

Спершу данні щодо вибраної поїздки відправляються на сервер по REST API які потім вже зберігаються в сесії. За цю операцію відповідає REST-контролер BookingRestController, який надає API для бронювання квитків.

Основні методи контролера:

1. setTicketsToSession - POST-метод, який отримує список TicketDTO у форматі JSON і зберігає його в сесії.
 - Отримується список TicketDTO з тілом HTTP-запиту у форматі JSON.
 - Список TicketDTO зберігається в сесії під іменем "listOfBookedTickets".

- Повертається код статусу HTTP 200 (OK).
- 2. `getTicketsFromSession` - GET-метод, який повертає список `TicketDTO` з сесії у форматі JSON.
 - Отримує список `TicketDTO` з сесії за допомогою імені `"listOfBookedTickets"`.
 - Список `TicketDTO` перетворюється в рядок JSON за допомогою бібліотеки `Gson`.
 - Повертається рядок JSON зі списком `TicketDTO`.

Цей контролер має шлях `/api/v1/booking` і використовує анотації `@RestController` та `@RequestMapping` для визначення маппінгу шляхів запитів.

Обидва методи підтримують JSON як формат вхідних та вихідних даних, вказано за допомогою анотацій `produces` та `consumes` з використанням `MediaType.APPLICATION_JSON_VALUE`.

У контролері використовується також клас `Gson` для серіалізації об'єктів у формат JSON.

Далі після збереження в сесії і повторної видачі вибраних квитків, користувача перекидає на сторінку оплати. За цю дію відповідає контролер `BookingController`, який обробляє HTTP-запити, пов'язані з бронюванням квитків.

Основні методи контролера:

1. `showBookingPageWithTravel` - цей метод обробляє запит на відображення сторінки бронювання для певного `travelId`.
 - Виконується перевірка, чи користувач не увійшов у систему (не має сесії). Якщо це так, користувач перенаправляється на сторінку входу в систему.
 - Якщо в сесії не має об'єкта `"listOfBookedTickets"`, користувач перенаправляється на сторінку з переліком поїздок.
 - Отримується об'єкт `Travel` за допомогою `travelService` на основі `travelId`.

- Об'єкт `Travel` додається до моделі `model` для передачі даних на сторінку.

- Повертається назва представлення `"booking"`.

Цей контролер залежить від декількох служб та класу `AccessControl`, які автоматично впроваджуються за допомогою анотації `@Autowired`.

Додаткові рядки, які виводять повідомлення у консоль для логування, допомагають відстежувати послідовність виконання методу під час відладки.

Користувач проводить оплату через платіжну систему `PayPal`. Результат успішної операції відпраіляється на URL `/api/v1/payment/successPayment/$travelId/$transactionId`. Цей результат обслуговує REST-контролер `PaymentController`, який надає API для обробки платежів.

Основний метод контролера:

`setTransaction` - POST-метод, який отримує дані платежу, інформацію про транзакцію та список заброньованих квитків і зберігає їх у системі.

Отримує шлях до транзакції (`transactionId`) та маршруту (`travelId`), а також JSON-рядок з даними платежу.

- Отримує список `TicketDTO` з сесії за допомогою імені `"listOfBookedTickets"`.

- Отримує об'єкт користувача з сесії за допомогою імені `"user"`.

- Створює об'єкт `Transaction` з використанням `transactionService.getTransaction(json)`.

- Викликає `ticketService.saveTickets(travelId, transaction, list, buyer)` для збереження квитків.

- Якщо при збереженні квитків виникає помилка `SeatOccupiedExeption`, змінна `success` встановлюється в `false`.

- Встановлює значення атрибуту `"listOfBookedTickets"` у сесії на `null`.

- Серіалізує змінну `success` у формат JSON за допомогою класу `Gson` і повертає рядок JSON.

Цей контролер має шлях `/api/v1/payment` і використовує анотації `@RestController` та `@RequestMapping` для визначення маппінгу шляхів запитів.

Метод `setTransaction` використовує анотації `@PostMapping`, `@ResponseBody`, `@ResponseStatus` для вказання методу HTTP (POST), типу вихідних даних (JSON), статусу HTTP (201 - CREATED) та повернення результату у відповідь HTTP.

Після успішного придбання квитків, користувач може їх переглянути у вкладці "Придбані квитки". По натисканню на цю кнопку користувач переходить до сторінки, url якої `/tickets/`. Цей URL обслуговує контролер `TicketController`, який відповідає за обробку запитів, пов'язаних з квитками.

Основні методи контролера:

1. `showTicketsByUserId` - GET-метод, який показує всі квитки, пов'язані з користувачем за його ідентифікатором.

- Отримує об'єкт користувача з сесії за допомогою імені "user".
- Якщо користувача немає, перенаправляє на сторінку `/home`.
- Отримує список квитків `List<Ticket>` за допомогою `ticketService.getAllTicketsByUser(user)`.

- Додає список квитків до моделі за ключем "tickets".
- Повертає назву представлення "tickets".

2. `showPassangers` - GET-метод, який показує список пасажирів для певної поїздки за її ідентифікатором.

- Перевіряє, чи має поточний користувач права доступу "admin" або "driver" за допомогою `accessControl.isAcceptable(session, "admin", "driver")`.

- Якщо у користувача немає цих прав доступу, перенаправляє на сторінку `/travels/`.

- Отримує список квитків `List<Ticket>` за допомогою `ticketService.getTicketsByTravelId(travelId)`, де `travelId` - це ідентифікатор поїздки.

- Додає список квитків до моделі за ключем "tickets".

- Додає атрибут "driverLook" до моделі зі значенням true.
 - Якщо роль користувача є "admin", додає атрибут "adminLook" зі значенням true.
 - Повертає назву представлення "tickets".
3. showTicket - GET-метод, який показує деталі певного квитка за його ідентифікатором.
- Отримує об'єкт користувача з сесії за допомогою імені "user".
 - Якщо користувача немає, перенаправляє на сторінку "/home".
 - Отримує об'єкт квитка Ticket за допомогою `ticketService.getTicket(Integer.parseInt(ticketId))`, де `ticketId` - це ідентифікатор квитка.

- Додає об'єкт квитка до моделі.
- Повертає назву представлення "ticket".

Цей контролер має шлях /tickets і використовує анотації @Controller та @RequestMapping для визначення мапінгу шляхів запитів.

Якщо прослідкувати то є численна кількість звернень до об'єкту ticketService. Це клас-сервіс об'єкта моделі логіки "Квиток"(англ.- Ticket), що TicketService містить логіку для обробки операцій з квитками..

Основні методи сервісу:

1. saveTickets - метод, який зберігає квитки та пов'язану з ними транзакцію.
 - Приймає ідентифікатор поїздки (travelId), об'єкт транзакції (transaction), список об'єктів TicketDTO (list) та об'єкт користувача-покупця (buyer).
 - Конвертує список TicketDTO в список об'єктів Ticket.
 - Встановлює час покупки кожного квитка з транзакції (transaction) у поле purchaseTime квитка.

- Генерує унікальне ім'я для кожного квитка, яке складається з ідентифікатора транзакції та номера місця.
 - Викликає метод `ticketTransactional.saveTickets` для збереження квитків, транзакції та їх пов'язаності з поїздкою та користувачем.
 - Повертає результат збереження квитків (`true` у разі успіху, інакше - `false`).
2. `sendTicket` - метод, який надсилає квиток на вказану електронну пошту.
- Приймає адресу електронної пошти (`email`) та об'єкт квитка (`ticket`).
 - Реалізація методу відсутня у даному фрагменті коду.
3. `getAllTicketsByUser` - метод, який повертає всі квитки, пов'язані з певним користувачем.
- Приймає об'єкт користувача (`user`).
 - Викликає метод `ticketTransactional.getAllTicketsByUser` для отримання списку квитків, пов'язаних з користувачем.
4. `getTicketsByTravelId` - метод, який повертає всі квитки, пов'язані з певною поїздкою за її ідентифікатором.
- Приймає ідентифікатор поїздки (`travelId`).
 - Викликає метод `ticketDao.getTicketsByTravelId` для отримання списку квитків, пов'язаних з поїздкою.
5. `getTicket` - метод, який повертає об'єкт квитка за його ідентифікатором.
- Приймає ідентифікатор квитка (`ticketId`).
 - Викликає метод `ticketDao.getTicket` для отримання об'єкта квитка за його ідентифікатором.

Цей сервіс використовує репозиторій `TicketDao` та транзакційний репозиторій `TicketTransactional` для доступу до даних про квитки.

Загалом цей клас-сервіс зберігає і запитає дані від/до бази даних. Це робиться через виклик методів DAO (Data Access Object, об'єкт доступу до даних)

шару, або класу, який в ієрархії шарів лежить між моделлю і доступом до даних – transactional.

У випадку з сервісом квитків спочатку розглянемо клас TicketTransactional. Транзакційний DAO-клас TicketTransactional, виконує операції з квитками у межах транзакцій.

Основні методи класу:

1. saveTickets - метод, який зберігає квитки, пов'язану з ними транзакцію та змінює доступні місця в поїзді.

- Приймає список об'єктів Ticket (tickets), об'єкт транзакції (transaction), ідентифікатор поїздки (travelId) та об'єкт користувача-покупця (buyer).

- Перевіряє, чи зайняте місце на поїзді для кожного квитка зі списку.

- Встановлює автоматичне керування транзакцією.

- Викликає метод ticketDao.saveTickets для збереження квитків, транзакції та їх пов'язаності з поїздом та користувачем.

- Змінює доступні місця на поїзді відповідно до кількості збережених квитків.

- Застосовує зміни до бази даних та фіксує транзакцію.

- Повертає значення true.

- У випадку виняткової ситуації (exception) транзакція не проходить, і всі зміни які мали були збережені – не зберігаються. А обробка помилки передається "на верх" по ієрархії шарів.

2. getAllTicketsByUser - метод, який повертає всі квитки, пов'язані з певним користувачем. Головна відміна цього методу від попереднього так це те що дані не зберігаються, але великий об'єм інформації вимагає його повноти – тому цей метод перейшов до проміжного шару між моделлю і шаром доступу до даних

- Приймає об'єкт користувача (user).

- Створює множину (ticketsSet) для збереження унікальних квитків.

- Встановлює автоматичне керування транзакцією.
- Отримує всі квитки за адресою електронної пошти пасажирів та адресою електронної пошти покупця з використанням методів `ticketDao.getAllByPassangerEmail` та `ticketDao.getAllByBuyerEmail`.
- Додає квитки до множини `ticketsSet`.
- Застосовує зміни до бази даних та фіксує транзакцію.
- Повертає список унікальних квитків.

Цей клас використовує DAO-класи `TicketDaoConnectable` та `TravelDaoConnectable` для доступу до даних про квитки та поїздки в межах транзакцій.

А тепер розглянемо найнижчий шар ієрархії – шар доступу до даних. У випадку це "перехрестя" об'єкту квитків і DAO layer є клас `TicketDaoImpl`.

Цей клас є реалізацією інтерфесу `TicketDao` та `TicketDaoConnectable` з використанням JDBC для доступу до даних про квитки. Основні методи включають збереження квитків у базі даних, отримання квитків за певними умовами, перевірку наявності зайнятого місця та отримання окремого квитка за його ідентифікатором.

Нижче перераховано деякі з основних методів, що використовуються в цьому класі:

- `save(Ticket ticket)`: Зберігає квиток у базі даних.
- `saveTickets(List<Ticket> tickets, Transaction transaction, int travelId, User buyer)`: Зберігає список квитків у базі даних як частину транзакції.
- `getAllByUserId(long userId)`: Отримує всі квитки за ідентифікатором користувача.
- `getAllByPassangerEmail(String email)`: Отримує всі квитки за email пасажирів.
- `getAllByBuyerEmail(String email)`: Отримує всі квитки за email покупця.

- `getTicketsByTravelId(String travelId)`: Отримує всі квитки за ідентифікатором поїздки.
- `checkIfSeatIsOccupied(List<Ticket> tickets, int travelId)`: Перевіряє, чи є зайняте місце на поїздку.
- `getTicket(int ticketId)`: Отримує окремий квиток за його ідентифікатором.

Цей клас також використовує `DataSource` та `TicketMapper` для підключення до бази даних та відображення результатів запитів, відповідно.

ВИСНОВКИ

У даній роботі було проведено аналіз вимог до нової системи обліку автобусів та розроблено концепцію її інтеграції з платіжними системами. Головною метою дослідження була допомога сфері пасажирських перевезень, як у теперішньому, так і в післявоєнному майбутньому, а також підтримка організації та діджиталізації процесів на автобусному підприємстві для перевезень до країн Європи.

Аналіз вимог до системи обліку автобусів дозволив виявити ключові функціональні вимоги, такі як збір та збереження інформації про автобуси, відстеження рейсів, моніторинг роботи автобусів та аналіз роботи автобусного парку. Інтеграція з платіжними системами розширила можливості системи, дозволяючи забезпечити зручну оплату послуг та збільшити рівень обслуговування пасажирів.

Результати дослідження вказують на необхідність впровадження нової системи обліку автобусів, що дозволить підприємствам пасажирських перевезень підвищити ефективність своєї роботи, забезпечити кращий контроль та управління автобусним парком, покращити якість обслуговування пасажирів та забезпечити конкурентні переваги на ринку.

У майбутньому рекомендується дослідження можливостей використання розширених аналітичних інструментів та штучного інтелекту для покращення аналізу та прийняття рішень у сфері пасажирських перевезень. Крім того, важливим є подальший розвиток інтеграції з платіжними системами та використання новітніх технологій для забезпечення безпеки та зручності пасажирів.

Ця робота вносить свій внесок у розвиток сфери пасажирських перевезень та використання інформаційних технологій для покращення процесів у даній галузі. Результати дослідження можуть бути використані підприємствами

пасажирських перевезень для впровадження нових систем обліку та покращення їх роботи, що сприятиме підвищенню рівня обслуговування пасажирів та розвитку даного сектору.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

4. Сайт автоперевізника Autolux [Електронний ресурс] // <https://autolux.ua/> – Режим доступу до ресурсу: <https://autolux.ua/ua/uk>. (дата звернення: 19.05.2024).
5. Офіційний сайт Sqills S3 Passenger [Електронний ресурс] – Режим доступу до ресурсу: <https://www.sqills.com/s3-passenger>. (дата звернення: 10.05.2024).
6. Сторінка продукту S3 Passenger на Siemens Xcelerator Marketplace [Електронний ресурс] // Siemens Xcelerator Marketplace – Режим доступу до ресурсу: <https://xcelerator.siemens.com/global/en/all-offerings/products/s/s3-passenger.html>. (дата звернення: 10.05.2024).
7. Головна сторінка компанії Bus Logic [Електронний ресурс] – Режим доступу до ресурсу: <https://buslogic.net/en/index.html>. (дата звернення: 10.05.2024).
8. Бору́та Я. Agile чи Waterfall – який варіант відповідає вашому бізнесу?. worksection.com. URL: <https://worksection.com/ua/blog/waterfall-vs-agile.html>. (дата звернення: 20.05.2024).
9. Зосим М. Спіральна модель Боема (Boehm's Spiral Model) [Електронний ресурс] / Максим Зосим // maxzosim.com. – 2023. – Режим доступу до ресурсу: <https://www.maxzosim.com/boehms-spiral-model/>. (дата звернення: 21.05.2024).
10. Лікаренко Я. Ітеративна модель розробки. Dou. URL: <https://dou.ua/lenta/articles/iterative-development-model/> (дата звернення: 21.05.2024).
11. Гранько О. Scrum чи не-Scrum - який підхід обрати?. worksection.com. URL: <https://worksection.com/ua/blog/scrum.html> (дата звернення: 20.05.2024).
12. Андерсон Д. Дж. Канбан : монографія / пер. з англ. Г. Якубовська. Фабула, 2021. 29-31 с.
13. Гранько О. Екстримальне програмування (XP) не для людей зі слабкими нервами. worksection.com. URL: <https://worksection.com/ua/blog/extreme-programming.html>. (дата звернення: 21.05.2024).

14. Гобов Д. Стан бізнес-аналізу в Україні (2020). Частина друга: Виявлення та аналіз вимог [Електронний ресурс] / Денис Гобов. – 2021. – Режим доступу до ресурсу: https://www.artofba.com/uk/post/state_of_business_analysis_ukraine_2020_part2. (дата звернення: 21.05.2024).
15. Документація до PayPal Checkout API [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.paypal.com/docs/checkout/>. (дата звернення: 06.05.2024).
16. Sakimoto F. The Psychology of Colors in UI/UX [Електронний ресурс] / Felipe Sakimoto – Режим доступу до ресурсу: <https://buzzvel.com/blog/the-psychology-of-colors-in-uiux>. (дата звернення: 08.05.2024).
17. Офіційний сайт проєкту OpenJDK [Електронний ресурс] – Режим доступу до ресурсу: <https://openjdk.org/>. (дата звернення: 10.03.2024).
18. Офіційна сторінка Spring Framework [Електронний ресурс] – Режим доступу до ресурсу: <https://spring.io/projects/spring-framework>. (дата звернення: 10.03.2024).
19. Офіційна сторінка бази даних PostgreSQL [Електронний ресурс] – Режим доступу до ресурсу: <https://www.postgresql.org/>. (дата звернення: 10.03.2024).
20. Офіційна сторінка проєкту Apache Tomcat [Електронний ресурс] – Режим доступу до ресурсу: <https://tomcat.apache.org/>. (дата звернення: 10.03.2024).
21. Сайт Git репозиторію [Електронний ресурс] – Режим доступу до ресурсу: <https://about.gitlab.com/>. (дата звернення: 10.03.2024).
22. Сторінка завантаження Eclipse IDE [Електронний ресурс] – Режим доступу до ресурсу: <https://www.eclipse.org/downloads/packages/release/2023-03/r/eclipse-ide-enterprise-java-and-web-developers>. (дата звернення: 10.03.2024).
23. Офіційна документація до pgAdmin 4 [Електронний ресурс] // [pgadmin.org](https://www.pgadmin.org/) – Режим доступу до ресурсу: <https://www.pgadmin.org/docs/pgadmin4/latest/index.html>. (дата звернення: 10.03.2024).

24. Сторінка завантаження Visual Studio Code [Електронний ресурс] – Режим доступу до ресурсу: <https://code.visualstudio.com/>. (дата звернення: 10.03.2024).
25. Сторінка вибору VPS тарифу [Електронний ресурс] – Режим доступу до ресурсу: <https://hyperhost.ua/en/vps-vds>. (дата звернення: 10.03.2024).
26. Сторінка проєкту Jira від Atlassian [Електронний ресурс] – Режим доступу до ресурсу: <https://www.atlassian.com/software/jira>. (дата звернення: 10.03.2024).
27. UML Class Diagram Tutorial [Електронний ресурс] // <https://www.visual-paradigm.com/> – Режим доступу до ресурсу: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>. (дата звернення: 12.04.2024).
28. UML Class Diagram предметної області автоперевезень [Електронний ресурс] // plantuml.com – Режим доступу до ресурсу: https://www.plantuml.com/plantuml/uml/ZLNDRjim3BxxAOISzeCMQx7kuCFGDp4OcOk9oR8XKXw2OUy-E_IYHBAHzM1K_FZ_VLBUMI18x5rLMGVCnWxHIo1ayxioyzzdQcl360YSdZtqM9i9WkqWL7-kSIJ_6wGcBNik3Nd14zwa2aEKbEfoQZOIj3WkhGTiB4dAeo24G1CuXSeI4tPaNieY7N6p6fky-MC0tsR5-w55bzaOGRY1VXaiQyV7_eYX6Hr8kGStXCmecn4taZTiE-ormMAao1Ykyoo7OeuV55Zk5ldXzUtjeu8EzZ7iWLM9Q9dxe6zVLsW4vbz3Q1FqzIAwOc5mMnAI01EwUqxKLeoIgCeMIQiMc80S7EUO2qrPs1Z8xkjBQ4ctV7EalV2vV1LX0OcS9meqOTXcJSZ-HBdEhCTYt34kwVYs4nRenyH2kOcRpHtfD4EhcEDPILEFoNnFT28I_AaeaQ27WONm2e-algAh8Zn3oz1qN_JQbOln35PeyDaU4W6RxvKNi2dds0qUzQHyNX_Emyl1F3rzDuSlpy-V3klBM9iEkDB_Smzzy6fprTK3l_Lng_GMb7vA_UmO6Tk6VAM1ksx0J2S_kyfGBMBQzrob_3X0PLsRVyhbldFiUvAtBhQpO_MEIhty8lq5. (дата звернення: 12.04.2024).
29. 2023 L.A. West Sprinter Mercedes 13 Passenger Luxury Bus [Електронний ресурс] // tecobus.com – Режим доступу до ресурсу: https://www.tecobus.com/buses-for-sale/2023-L.A.-West-Sprinter-Mercedes-13-Passenger-Luxury-Bus_LA4504/. (дата звернення: 12.04.2024).

30. Документація по створенню баз даних за допомогою pgAdmin
[Електронний ресурс] – Режим доступу до ресурсу:
https://www.pgadmin.org/docs/pgadmin4/8.8/database_dialog.html. (дата звернення:
10.03.2024).

31. Документація по створенню таблиць за допомогою pgAdmin
[Електронний ресурс] – Режим доступу до ресурсу:
https://www.pgadmin.org/docs/pgadmin4/latest/table_dialog.html. (дата звернення:
10.03.2024).

ДОДАТКИ

Додаток А

Схема підлоги автобуса в форматі json

```
{  
  "passanger_seats_amount": 13,  
  "total_elements": 15,  
  "max_in_row": 4,  
  "scheme": [  
    {  
      "row": [  
        "driver",  
        "way",  
        "way",  
        "entrance"  
      ]  
    },  
    {  
      "row": [  
        "seat1",  
        "seat2",  
        "way",  
        "seat13",  
      ]  
    },  
    {  
      "row": [  
        "table",  
        "table",  
        "way",  
      ]  
    }  
  ]  
}
```

```
        "seat12"
    ]
},
{
    "row": [
        "seat3",
        "seat4",
        "way",
        "seat11"
    ]
},
{
    "row": [
        "seat5",
        "seat6",
        "way",
        "seat10"
    ]
},
{
    "row": [
        "seat7",
        "seat8",
        "seat9"
    ]
},
]
}
```

```
-- Table: public.countries
-- DROP TABLE IF EXISTS public.countries;
CREATE TABLE IF NOT EXISTS public."countries"
(
    id serial,
    name character varying COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT countries_pkey PRIMARY KEY (id)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.countries
    OWNER to postgres;
-- Table: public.cities
-- DROP TABLE IF EXISTS public.cities;
CREATE TABLE IF NOT EXISTS public."cities"
(
    id serial,
    name character varying COLLATE pg_catalog."default" NOT NULL,
    country integer NOT NULL,
    CONSTRAINT cities_pkey PRIMARY KEY (id),
    CONSTRAINT city_country FOREIGN KEY (country)
        REFERENCES public.countries (id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.cities
    OWNER to postgres;
-- Table: public.routes
-- DROP TABLE IF EXISTS public.routes;
CREATE TABLE IF NOT EXISTS public."routes"
(
    route_name character varying COLLATE pg_catalog."default" NOT NULL,
    departure_city integer NOT NULL,
    arrival_city integer NOT NULL,
    CONSTRAINT route_pkey PRIMARY KEY (route_name),
    CONSTRAINT route_arrival_city FOREIGN KEY (arrival_city)
        REFERENCES public.cities (id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
```

```

    NOT VALID,
    CONSTRAINT route_departure_city FOREIGN KEY (departure_city)
    REFERENCES public.cities (id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.routes
    OWNER to postgres;
-- Table: public.vehicles
-- DROP TABLE IF EXISTS public.vehicles;
CREATE TABLE IF NOT EXISTS public."vehicles"
(
    brand character varying COLLATE pg_catalog."default" NOT NULL,
    model character varying COLLATE pg_catalog."default" NOT NULL,
    seats_amount integer NOT NULL,
    number_plate character varying COLLATE pg_catalog."default" NOT NULL,
    color character varying COLLATE pg_catalog."default" NOT NULL,
    this_vehicle_id serial,
    type character varying COLLATE pg_catalog."default" NOT NULL,
    seat_scheme_id integer NOT NULL,
    CONSTRAINT vehicles_pkey PRIMARY KEY (this_vehicle_id)
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.vehicles
    OWNER to postgres;
-- Table: public.users
-- DROP TABLE IF EXISTS public.users;
CREATE TABLE IF NOT EXISTS public."users"
(
    user_id serial,
    firstname character varying(255) COLLATE pg_catalog."default" NOT NULL,
    lastname character varying(255) COLLATE pg_catalog."default" NOT NULL,
    email character varying(255) COLLATE pg_catalog."default" NOT NULL,
    username character varying(64) COLLATE pg_catalog."default",
    password character varying(512) COLLATE pg_catalog."default",
    role character varying COLLATE pg_catalog."default" NOT NULL DEFAULT
'user'::character varying,
    CONSTRAINT users_pkey PRIMARY KEY (user_id),
    CONSTRAINT email UNIQUE (email)
)
TABLESPACE pg_default;

```

```

ALTER TABLE IF EXISTS public.users
    OWNER to postgres;
-- Table: public.travels
-- DROP TABLE IF EXISTS public.travels;
CREATE TABLE IF NOT EXISTS public."travels"
(
    vehicle_id integer NOT NULL,
    route_name character varying COLLATE pg_catalog."default" NOT NULL,
    available_seats integer NOT NULL,
    price character varying COLLATE pg_catalog."default" NOT NULL,
    departure_time time without time zone NOT NULL,
    departure_date date NOT NULL,
    this_travel_id serial,
    status character varying COLLATE pg_catalog."default" NOT NULL DEFAULT
'created'::character varying,
    driver_id integer NOT NULL ,
    CONSTRAINT travels_pkey PRIMARY KEY (this_travel_id),
    CONSTRAINT travel_route FOREIGN KEY (route_name)
        REFERENCES public.routes (route_name) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT travel_vehicle FOREIGN KEY (vehicle_id)
        REFERENCES public.vehicles (this_vehicle_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.travels
    OWNER to postgres;
-- CHECK: public.phone_number.phone_number_check
-- ALTER DOMAIN public.phone_number DROP CONSTRAINT
phone_number_check;
ALTER DOMAIN public.phone_number
    ADD CONSTRAINT phone_number_check CHECK (VALUE ~ '^(?:((?:\d{2})?[ -]?(?:((?:\d{2})\d{2})|(?:0\d{2}))?)?[ -]?(?:\d{3}[ -]?\d{2}[ -]?\d{2}|\d{2}[ -]?\d{2}[ -]?\d{3}|\d{7})|(?:((?:\d{2})\d{2})?[ -]?\d{3}[ -]?\d{3}[ -]?\d{3}))$':text);
-- Table: public.tickets
-- DROP TABLE IF EXISTS public.tickets;
CREATE TABLE IF NOT EXISTS public.tickets
(
    purchase_time timestamp without time zone NOT NULL,

```

```

user_id integer,
ticket_id serial NOT NULL,
travel_id bigint NOT NULL,
buyer_id character varying COLLATE pg_catalog."default" NOT NULL,
buyer_email character varying COLLATE pg_catalog."default",
passanger_id integer,
passanger_email character varying COLLATE pg_catalog."default",
passanger_firstname character varying COLLATE pg_catalog."default",
passanger_lastname character varying COLLATE pg_catalog."default",
buyer_firstname character varying COLLATE pg_catalog."default",
buyer_lastname character varying COLLATE pg_catalog."default",
seat_number integer NOT NULL,
transaction_id character varying COLLATE pg_catalog."default" NOT NULL,
ticket_name character varying COLLATE pg_catalog."default" NOT NULL,
contact_number phone_number COLLATE pg_catalog."default",
CONSTRAINT tickets_pkey PRIMARY KEY (ticket_id),
CONSTRAINT ticket_travel FOREIGN KEY (travel_id)
    REFERENCES public.travels (this_travel_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION,
CONSTRAINT ticket_user FOREIGN KEY (user_id)
    REFERENCES public.users (user_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
)
TABLESPACE pg_default;
ALTER TABLE IF EXISTS public.tickets
    OWNER to postgres;

```

Додаток В

Фрагмент коду

```
package com.niki.TicketSellService.model.service;
import java.sql.SQLException;
import java.time.LocalDateTime;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.niki.TicketSellService.dao.TravelDao;
import com.niki.TicketSellService.dao.VehicleDao;
import com.niki.TicketSellService.model.dto.TravelDTO;
import com.niki.TicketSellService.model.entity.Travel;
@Service
public class TravelService {
    @Autowired
    private TravelDao travelDao;
    @Autowired
    private VehicleDao vehicleDao;
    public TravelService() {
    }
    public TravelService(TravelDao travelDao) {
        this.travelDao = travelDao;
    }
    public List<Travel> getAllCreatedTravels() throws SQLException {
        return travelDao.getAllCreatedTravels();
    }
    public String getPrice(String travelId) throws SQLException {
        return travelDao.getTravelPriceById(travelId);
    }
    public Travel getTravelById(String travelId) throws SQLException {
```

```

        return travelDao.getTravelById(travelId);
    }

    public int getAmountCreatedTravels() throws SQLException {
        return travelDao.getAmountCreatedTravels();
    }

    public List<Travel> getTravelsBySearch(String departCityId, String
arrivalCityId, String time) throws SQLException {
        if (arrivalCityId != null && time != null) {
            return travelDao.getTravelsBy(Integer.valueOf(departCityId),
Integer.valueOf(arrivalCityId), LocalDateTime.parse(time));
        } else if (arrivalCityId == null && time == null) {
            return travelDao.getTravelsBy(Integer.valueOf(departCityId));
        } else if (arrivalCityId != null) {
            return travelDao.getTravelsBy(Integer.valueOf(departCityId),
Integer.valueOf(arrivalCityId));
        } else {
            return travelDao.getTravelsBy(Integer.valueOf(departCityId),
LocalDateTime.parse(time));
        }
    }

    public List<Integer> getAvailableSeats(String travelId) throws
NumberFormatException, SQLException {
        return travelDao
            .getListAvailableSeatsByTravelId(Integer.parseInt(travelId));
    }

    public void changeTravelStatus(String status, int travelId) throws SQLException
    {
        travelDao.changeTravelStatus(status, travelId);
    }

    public void createTravel(TravelDTO travelDTO) throws SQLException {

```

```
travelDTO.setSeatAmount(vehicleDao.getSeatsAmountByVehicleId(travelDTO.getVehicleId()));
```

```
    travelDao.createTravel(travelDTO);
```

```
}
```

```
public List<Travel> getTravels() throws SQLException {
```

```
    return travelDao.getAllCreatedTravels();
```

```
}
```

```
}
```

Ім'я користувача:
Комп'ютерної математики та інформаційної безпеки...

ID перевірки:
1016345484

Дата перевірки:
11.06.2024 02:02:49 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
14.06.2024 13:58:07 EEST

ID користувача:
100005746

Назва документа: Диплом_Голик_І_К

Кількість сторінок: 54 Кількість слів: 9753 Кількість символів: 76796 Розмір файлу: 1.53 MB ID файлу: 1016147131

2.79% Схожість

Найбільша схожість: 0.21% з джерелом з Бібліотеки (ID файлу: 1015251001)

0.98% Джерела з Інтернету

24

Сторінка 56

2.62% Джерела з Бібліотеки

153

Сторінка 56

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0% Вилучень

Немає вилучених джерел