МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп’ютерних наук та управління проєктами

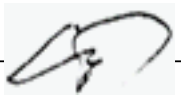
Кафедра програмного забезпечення автоматизованих систем

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма «Інженерія програмного забезпечення»

«Допущений до захисту»

Завідувач кафедри

_  _ проф. Приходько С.Б.

«___» _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти «бакалавр»

на тему: Розробка програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Виконав: студент групи 4151ст3

_____ Хлистов А.О.

(підпис, ПІБ)

Керівник роботи:

доцент кафедри ПЗАС, к.т.н., доцент

(посада, науковий ступень вчене звання)

_____Макарова Л. М

(підпис, ПІБ)

Миколаїв – 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____ доц. Макарова Л.М.
«_08_» ___04___ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «бакалавр»

Студенту_____Хлистову Андрію Олександровичу_____
(Прізвище, ім'я, по батькові)

1. Тема роботи: Розробка програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією_____

Керівник роботи_____доцент кафедри ПЗАС, к.т.н., доцент Макарова Л. М_____

Затверджені наказом ректора №153 від 08.04.2025 р.

2. Термін подання роботи: 02.06.2025 р. _____

3. Вихідні дані по роботі:_____

4. Перелік питань, що належать до розробки (найменування розділів)_____

Титульний аркуш; Завдання на кваліфікаційну роботу; Анотація (українською та англійською мовами); Зміст; Перелік умовних позначень, символів, одиниць та термінів (при необхідності); Вступ; Аналіз практичної задачі інженерії програмного забезпечення; Проєкт програмного забезпечення; Результати розробки програмного забезпечення; Розділ з охорони праці; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма та методика випробувань програмного забезпечення).

5. Перелік презентаційних матеріалів _____

Титульний слайд, Мета та завдання кваліфікаційної роботи, Аналіз вимог до ПЗ, Архітектура програмного забезпечення, Концептуальна модель даних, Сценарії варіантів використання ПЗ, Діаграма класів ПЗ, Діаграми послідовності ПЗ, Логічна модель бази даних, Стек технологій для розробки ПЗ, Результати розробки ПЗ, Висновки, Заключний слайд._____

_____

_____

_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Гурець Н.В., ст. викладач		

7. Дата видачі завдання 08.04.2025 р.

КАЛЕНДАРНИЙ ПЛАН

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу ВСТУП	31.03.2025	ПП*
2.	Аналіз практичної задачі інженерії ПЗ	07.04.2025	ПП*
3.	Аналіз вимог до ПЗ	14.04.2025	ПП*
4.	Розробка архітектури ПЗ	21.04.2025	
5.	Розробка проєкту ПЗ	05.05.2025	
6.	Реалізація та тестування ПЗ	12.05.2025	
7.	Підготовка розділу Результати розробки ПЗ	16.05.2025	
8.	Підготовка розділу з охорони праці	19.05.2025	ПП*
9.	Підготовка розділу ВИСНОВКИ	23.05.2025	
10.	Оформлення списку використаних джерел та додатків	26.05.2025	
11.	Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версій роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.)	02.06.2025	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку
12.	Підготовка презентації та доповіді	06.06.2025	
13.	Попередній захист роботи на засіданні кафедри ПЗАС	09.06.2025	
14.	Подання на кафедру ПЗАС електронних версій наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді, а також Авторського договору з додатком	16.06.2025	

* - за результатами переддипломної практики (ПП), яка була з 03.02.2025 до 23.03.2025 р.

Студент _____ Хлистов А.О.
(підпис) (ПІБ)

Керівник роботи _____ Макарова Л.М.
(підпис) (ПІБ)

АНОТАЦІЯ

Кваліфікаційна робота присвячена вирішенню практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов, а саме розробці програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією.

У кваліфікаційній роботі проведено аналіз практичної задачі інженерії програмного забезпечення з розробки вказаного програмного забезпечення, розглянуто існуючі аналогічні програмні продукти, сформульовано постановку задачі на розробку програмного шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією. Представлено проєкт програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією, результати розробки програмного забезпечення та розділ з охорони праці.

Кваліфікаційна робота викладена на 84 сторінках друкованого тексту, містить 15 рисунків, 15 таблиць, список використаних джерел з 18 найменувань та 5 додатків.

ABSTRACT

The qualification work is dedicated to solving the practical task of software engineering, characterized by complexity and uncertainty of conditions, namely Software development for the school library of the Kiryakovsky Lyceum with an elementary school and gymnasium.

In the qualification work, the analysis of the practical task of software engineering for the development of the specified software is carried out, the existing similar software products are considered, the statement of the task for the development of software for the school library of the Kiryakovsky Lyceum with an elementary school and gymnasium were formulated. The software project for the school library of the Kiryakovsky Lyceum with an elementary school and gymnasium, the results of software development and the section on labor protection were presented.

The qualification work is presented on 84 pages of typewritten text and contains: 15 figures, 15 tables, list of references from 18 names, 5 appendices.

ЗМІСТ

ВСТУП

У сучасному інформаційному суспільстві роль бібліотек суттєво змінюється: від традиційних книгозбірень вони трансформуються у центри зберігання, доступу та обробки інформації. Особливо це актуально для шкільних бібліотек, які відіграють важливу роль у супроводженні навчального процесу, формуванні читацької культури учнів, забезпеченні вчителів необхідною навчальною та методичною літературою для роботи зі школярами. В той же час шкільні бібліотеки часто стикаються з проблемами ведення документації, обліку літератури та ефективного обслуговування читачів у зв'язку з відсутністю сучасних автоматизованих бібліотечних систем, орієнтованих саме на шкільні бібліотеки.

Автоматизація роботи шкільної бібліотеки сприятиме оптимізації щоденних робочих процесів: створення електронного каталогу, обліку наявного книжкового фонду, видачі та повернення літератури, ведення читацьких формулярів. Це дозволить не лише зменшити навантаження на бібліотекаря, а й забезпечити більш якісне та швидке обслуговування читачів. Тому питання розробки доступного, адаптованого до потреб шкільної бібліотеки, програмного забезпечення є особливо актуальним [1, 2].

Взагалі, існує доволі багато програмного забезпечення для автоматизації роботи бібліотеки, зокрема: IRBIS64-UA, УФД/Бібліотека [3], Koha (локалізована українська версія) [4], Автоматизована Бібліотечно-Інформаційна Система (АБІС) «МАРК-SQL» та інші. Детальний огляд деяких з цих систем буде наведено нижче, у підрозділі 1.2.

Деякі із них мають високу ціну, деякі - потребують добре обізнаного фахівця для встановлення та налаштування. Ці та інші аналогічні програмні продукти частіше використовують у великих бібліотеках. Але коли йде мова

про автоматизацію невеликої шкільної бібліотеки, можливо розглянути також і варіант власної розробки.

Метою даної кваліфікаційної роботи є розв’язання практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов - розробка програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією, що дозволить автоматизувати, спростити та пришвидшити роботу бібліотекаря та полегшити ведення звітності.

Для досягнення цієї мети необхідно:

- виконати аналіз практичної задачі інженерії програмного забезпечення з теми, що розглядається;

- проаналізувати існуючі аналогічні програмні рішення;

- здійснити постановку задачі на розробку програмного забезпечення;

- виконати аналіз вимог до програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією;

- розробити архітектуру програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією;

- розробити ескізний, технічний та робочий проекти програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією;

- здійснити кодування, тестування та випробування розробленого програмного забезпечення;

- розробити усю необхідну програмну документацію.

Практичне значення роботи полягає у створенні придатного до впровадження програмного продукту, який буде використаний у бібліотеці Кир'яківського ліцею з початковою школою та гімназією для підвищення ефективності функціонування бібліотеки та поліпшення якості обслуговування

читачів, ефективного управління фондом бібліотеки, обліку дій користувачів та формування звітності.

Об’єктом кваліфікаційної роботи є процес розробки програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією.

1 АНАЛІЗ ПРАКТИЧНОЇ ЗАДАЧІ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ШКІЛЬНОЇ БІБЛІОТЕКИ КИР'ЯКІВСЬКОГО ЛІЦЕЮ З ПОЧАТКОВОЮ ШКОЛОЮ ТА ГІМНАЗІЄЮ

1.1 Аналіз практичної задачі інженерії програмного забезпечення

У Кир'яківському ліцеї з початковою школою та гімназією функціонує шкільна бібліотека, яка обслуговує учнів та вчителів. Облік літератури, читачів та видачі книг досі ведеться вручну за допомогою паперових формулярів та журналів. Це призводить до:

- низької продуктивності обліку;
- ймовірності помилок у записах;
- ускладнення пошуку інформації;
- відсутності аналітики та звітів про фонд і користувачів;
- втрати часу на повторні дії, наприклад, при поверненні книги без точного запису дати видачі.

До того ж, неможливо швидко та точно дати відповіді на такі питання:

- наявна або відсутня книга в бібліотеці;
- місце знаходження книги;
- автор конкретної книги;
- які книги конкретного автора наявні в бібліотеці;
- кількість примірників конкретної книги в бібліотеці.

Для ведення бібліотечних каталогів, організації пошуку необхідних видань і бібліотечної статистики повинні зберігатися відомості, велика частина яких розміщуються в анотованих каталожних картках. Аналіз запитів на

літературу показує, що для пошуку видань, наприклад, за тематикою, автором або видавництвом слід виділити наступні атрибути:

- автор видання;
- назва видання;
- вид видання (підручник, довідник, атлас і т.п.);
- укладач видання;
- місце видання (місто);
- видавництво;
- рік випуску видання;
- авторський знак.

Бібліотечний шифр і авторський знак можуть використовуватися для складання каталогів та організації розстановки видань на полицях.

До об'єктів і атрибутів, що дозволяють охарактеризувати окремі екземпляри видань, можна віднести [5]:

- дата придбання конкретної книги;
- ціна конкретної книги;
- номер читацького квитка (формуляра);
- дата видачі читачу конкретного екземпляра книги;
- дата повернення книги.

До об'єктів і атрибутів, що дозволяють охарактеризувати читача, можна віднести:

- прізвище читача, ім'я, по батькові;
- адреса читача;
- телефон читача;
- статус читача (учень чи учитель);
- клас для учнів.

1.2 Аналіз існуючих аналогічних програмних рішень

У ході аналізу існуючих аналогічних програмних рішень було розглянуто декілька програм: УФД/Бібліотека [3], Koha (локалізована українська версія) [4], Evergreen [6], OpenBiblio [7]. Розглянемо більш детально кожну із них, їх переваги та недоліки для з точки зору використання у шкільній бібліотеці Кир'яківського ліцею з початковою школою та гімназією.

УФД/Бібліотека

Розробник програми ТОВ "УкрФінДані". Призначена вона для автоматизації бібліотечних процесів у навчальних закладах, технікумах, школах. Встановлюється на Windows-платформу.

Основні можливості: має повну підтримку української мови, ведення електронного каталогу, обліку літератури, читацьких формулярів, пошуку книг. Дає можливість отримати розширені звіти та статистику.

Недоліки: існує тільки платна версія (по ліцензії), відсутня можливість розширення та підтримки стандартів MARC.

Koha (локалізована українська версія)

Koha – це міжнародна Open Source система, яку останнім часом активно адаптують та локалізують в Україні. Призначена для всіх типів бібліотечних закладів, зокрема і шкільних. Відомі приклади впровадженняцієї програми: бібліотеки ВНЗ, публічні бібліотеки в ОТГ. Встановлюється на вебплатформу (сервер).

Основні можливості: підтримує веб-інтерфейс, електронний каталог, облік літератури, читацьких формулярів, розширений пошук книг. Має гнучкі шаблони звітів та статистики.

Недоліки: незважаючи на те, що вона є безкоштовною, потребує серйозного налаштування.

Evergreen

Веб-орієнтована ILS (Integrated Library System), яка працює за ліцензією GNU GPL.

Основні можливості: підтримує веб-інтерфейс, розширений електронний каталог, облік літератури з автоматизованим нагадуванням про необхідність повернення, розширене управління читацькими формулярами. Широкі можливості налаштування звітів.

Більш потужне рішення, яке краще впроваджувати у великих навчальних закладах або на рівні шкільної мережі.

Недоліки: мова інтерфейсу англійська з частковою локалізацією, впровадження складне для непрофесіоналів, потребує окремого адміністратора, предявляє високі вимоги до ресурсів, оскільки має серверне розгортання на платформі Linux-серверу.

OpenBiblio

Веб-орієнтована ILS, яка працює за ліцензією GNU GPL.

Основні можливості: підтримує веб-інтерфейс, базовий електронний каталог (автор, назва, ISBN, категорії), облік літератури стандартний, управління читацькими формулярами також стандартне. Має прості фіксовані звіти.

Недоліки: мова інтерфейсу англійська з можливістю ручної локалізації, впровадження просте, але працює на платформі Apache/PHP/MySQL, що, скоріш за все, потребує окремого адміністратора, предявляє мінімальні вимоги до ресурсів.

Підводячи підсумки проведеного аналізу, можна зробити висновки про наступне.

УФД/Бібліотека: автоматизована інформаційно-бібліотечна система більше призначена для комплексної автоматизації діяльності великої бібліотеки, забезпечує автоматизацію основних виробничих процесів.

Koha: призначена для підтримки традиційних бібліотечних технологічних процесів, добре працює з періодичними виданнями, що не характерно для шкільної бібліотеки.

Evergreen: більш потужне рішення, яке краще впроваджувати у великих навчальних закладах або на рівні мережі бібліотек.

OpenBiblio: більш проста система, яку можна швидко встановити та використовувати для базового обліку книг і читачів.

1.3 Постановка задачі на розробку програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Виходячи з аналізу практичної задачі інженерії програмного забезпечення та дослідження існуючих програмних рішень, було прийнято рішення розробити власне програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією. Користувачем даного програмного забезпечення є бібліотекар.

В БД програмного забезпечення повинні використовуються наступні вхідні дані.

Читацький квиток: ПІБ, адреса, телефон, статус ,клас.

Інформація по книзі: ПІБ автора, назва книги, вид видання, назва видавництва, дата отримання, ціна.

Інформація про зарезервовані книги: номер читацького квитка, назва книги, дата резерву.

Інформація про видані книги: номер читацького квитка, назва книги, дата видачі, дата повернення.

В БД програмного забезпечення повинні використовуються наступні вихідні дані у форматі звітів.

Звіт «Видані книги»: номер читацького квитка, назва книги, дата видачі книги.

Звіт «Інвентаризація: ПІБ автора, назва книги, вид видання, назва видавництва, дата отримання, ціна.

Звіт «Планове повернення»: номер читацького квитка, назва книги, дата повернення книги.

Звіт «Заборгованість»: номер читацького квитка, назва книги, дата повернення книги.

Програмне забезпечення повинно надавати наступні можливості для користувача:

- авторизація;
- занесення даних про книгу;
- редагування даних про книгу;
- перегляд інформації про книгу;
- видалення інформації про книгу;
- оформлення читацького квитка;
- редагування читацького квитка;
- перегляд інформації про читацький квиток;
- видалення читацького квитка;
- оформлення бронювання книги;
- оформлення видачі книги;
- пошук необхідної інформації;
- формування необхідних звітів.

Вимоги до організації вхідних та вихідних даних

Введення даних здійснюється за допомогою пристроїв введення даних (клавіатури та миші). Дані або вводяться вручну, або обираються із представлених списків.

Обмін інформацією між програмним забезпеченням і базою даних відбувається за допомогою файлів з довільним доступом з розширенням .myd.

У програмному забезпеченні доступ до даних має тільки бібліотекар. Для роботи з програмним забезпеченням необхідно пройти авторизацію.

Було виділено вимоги до складу та параметрів технічних засобів.

Система повинна задовольняти вище вказаним вимогам на комп’ютері наступній мінімальній комплекції:

– CPU: Intel(R) i5 2,16 МГц або аналогічний за параметрами AMD;

– RAM: 4 GB;

– HDD: 10 GB вільного місця.

Для повноцінного функціонування системи необхідно використовувати ОС Windows версії не нижче 7. Необхідна наявність встановленого JDK 17.

Детальна постановка задачі на розробку програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлена у технічному завданні, яке наведено у додатку А.

2 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ШКІЛЬНОЇ БІБЛІОТЕКИ КИР'ЯКІВСЬКОГО ЛІЦЕЮ З ПОЧАТКОВОЮ ШКОЛОЮ ТА ГІМНАЗІЄЮ

2.1 Аналіз вимог до програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

2.1.1 Побудова моделі варіантів використання

Діаграма варіантів використання (Use Case Diagram) - це один із типів UML-діаграм, який дозволяє графічно представити вимоги до створюваної системи. Вона є концептуальною моделлю, що відображає очікувану взаємодію користувачів із системою, не занурюючись у її внутрішню реалізацію.

Основними елементами діаграми є:

- варіанти використання (use case) - дії або функції, які система має виконувати;

- актори (actors) - зовнішні користувачі чи системи, що взаємодіють із системою;

- відносини (relationships) - зв’язки між акторами та варіантами використання.

В якості актора може виступати будь-який об’єкт, суб’єкт або система, що взаємодіє із розроблюваною системою ззовні з метою досягнення певної мети або виконання завдань. Актором може бути людина, інша програма, технічний пристрій чи будь-яка інша система, яка ініціює взаємодію з моделлю [8].

Діаграма варіантів використання програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлена на рисунку 2.1.

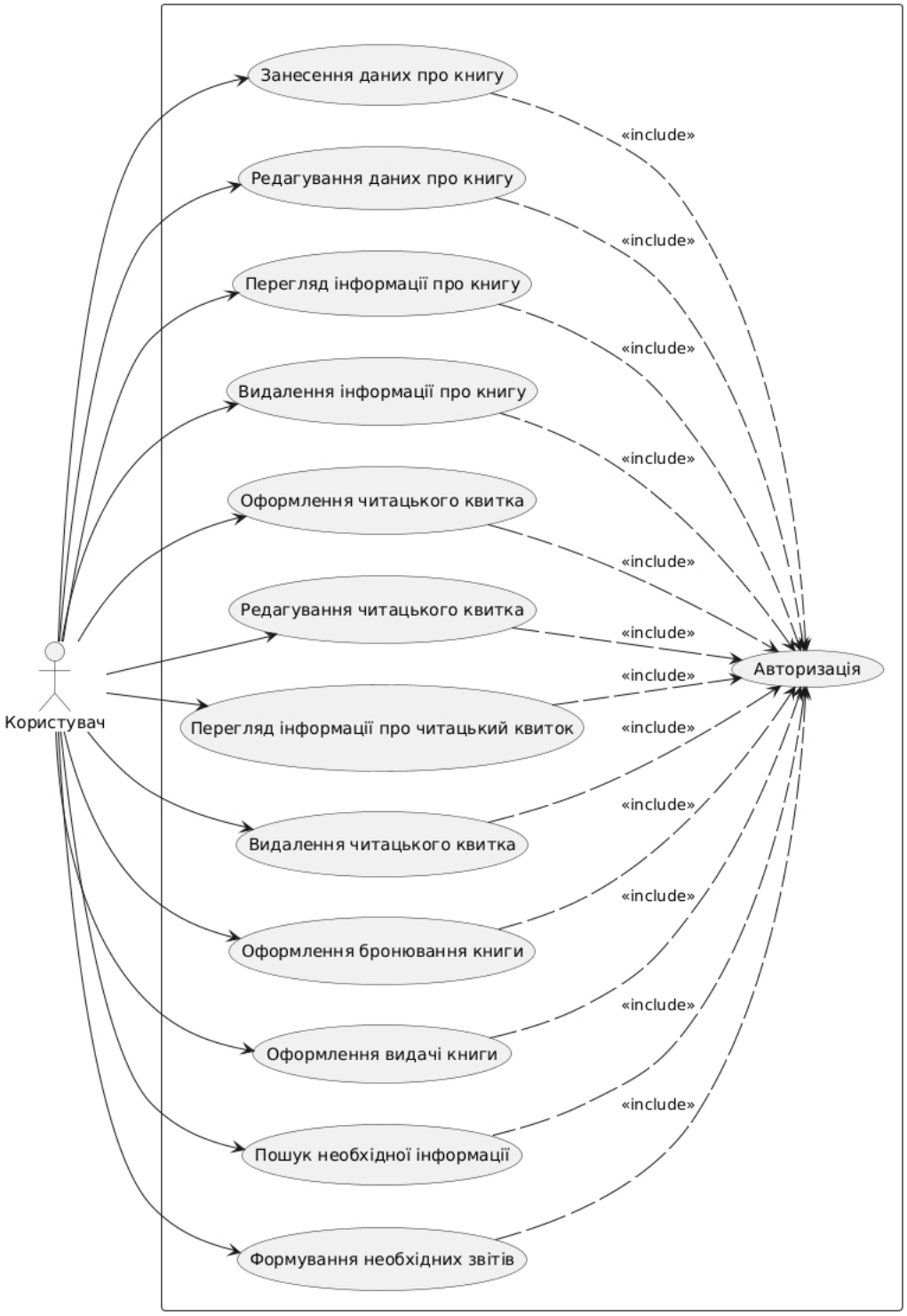


Рисунок 2.1 – Діаграма варіантів використання програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Програмне забезпечення має одного користувача (біліотекаря). Перш ніж почати роботу з програмним забезпеченням, користувач має виконати

авторизацію. Лише після успішної авторизації він може виконувати всі інші варіанти використання. Така структура добре підходить для систем з єдиним обов’язковим входом перед будь-якими діями.

Варіанти групуються логічно: робота з книгами, читацькими квитками, видачею/бронюванням, звітністю та пошуком.

2.1.2 Розробка специфікації варіантів використання

Розглянемо більш детально кожний варіант використання. Потік подій варіанта використання складається з:

- короткого опису;
- передумов;
- основного потоку подій;
- альтернативного потоку подій;
- постумов [8].

Ці складові частини для основних варіантів використання розроблюваного програмного забезпечення приведені у таблицях 2.1 – 2.7.

Таблиця 2.1 – Опис варіанту використання "Авторизація"

Опис	Складова
1	2
Короткий опис	Дана функція дозволяє особі пройти перевірку для доступу до системи.
Передумови	Система запущена та виводить форму авторизації. Наявність ідентифікаційного запису у базі.
Основний потік подій	1. Система виводить форму для ведення логіну та паролю 2. Користувач водить логін та пароль та підтверджує введення 3. Система перевіряє вірність ведення логіну та паролю 4. Якщо дані достовірні, то завантажується головне вікно програми

Продовження таблиці 2.1

1	2
Альтернативний потік подій	1 Система виводить форму для ведення логіну та паролю 2 Користувач водить логін та пароль та підтверджує введення 3 Система перевіряє вірність ведення логіну та паролю 4 Повідомлення про те, що логін або пароль не вірні 5 Надання повторного ведення даних
Постумови	Користувач авторизований в системі.

Таблиця 2.2 – Опис варіанту використання "Занесення даних про книгу"

Опис	Складова
Короткий опис	Призначений для заповнення довідкової інформації.
Передумови	Користувач авторизований у системі. Користувач повинен обрати режим для роботи з довідниками.
Основний потік подій	1. Авторизація; 2. Система виводить головне вікно роботи з програмою 3. Користувач вибирає режим для роботи з довідниками 4. Система виводить список довідників 5. Користувач вибирає довідник для роботи з книгою 6. Система відкриває список записів у довіднику 7. Користувач вибирає «Додати» 8. Система виводить пустий запис 9. Користувач заповнює всі потрібні реквізити нового елементу та натискає зберегти 10. Система обробляє інформацію, якщо нема помилок зберігає запис
Альтернативний потік подій	Відсутній.
Постумови	Створено новий запис в довіднику.

Таблиця 2.3 – Опис варіанту використання "Оформлення читацького квитка"

Опис	Складова
Короткий опис	Призначення для створення нового читацького квитка.
Передумови	Користувач авторизований в системі. Користувач повинен обрати режим для роботи з довідниками.
Основний потік подій	1. Авторизація 2. Система виводить головне вікно роботи з програмною 3. Користувач обирає режим для роботи з довідниками 4. Система виводить список довідників 5. Користувач обирає довідник «Читацький квиток» та натискає кнопку «Додати» 6. В новій формі заповнює всі необхідні реквізити нового елементу та натискає кнопку «Зберегти» 7. Система обробляє інформацію, якщо немає помилок, зберігає запис
Альтернативний потік подій	Відсутній.
Постумови	Створено новий читацький квиток.

Таблиця 2.4 – Опис варіанту використання "Оформлення бронювання книги"

Опис	Складова
1	2
Короткий опис	Призначений для оформлення бронювання книги
Передумови	Користувач авторизований в системі. Користувач повинен обрати режим для роботи з документами.

Продовження таблиці 2.4

Основний потік подій	1. Авторизація 2. Система виводить головне вікно роботи з програмною 3. Користувач обирає режим для роботи з документами 4. Обирає документ «Бронювання книги» та натискає кнопку «Добавить» 5. В новій формі заповнює всі необхідні реквізити нового елементу та натискає кнопку «Зберегти» 6. Система обробляє інформацію, якщо немає помилок, зберігає запис
Альтернативний потік подій	Відсутній.
Постумови	Сформовано нове бронювання на книгу.

Таблиця 2.5 – Опис варіанту використання "Оформлення видачі книги"

Опис	Складова
Короткий опис	Призначений для оформлення видачі книги
Передумови	Користувач авторизований в системі. Користувач повинен обрати режим для роботи з документами.
Основний потік подій	1. Авторизація 2. Система виводить головне вікно роботи з програмною 3. Користувач обирає режим для роботи з документами 4. Обирає документ «Видача» та натискає кнопку «Додати» 5. В новій формі заповнює всі необхідні реквізити нового елементу та натискає кнопку «Зберегти» 6. Система обробляє інформацію, якщо немає помилок, зберігає запис
Альтернативний потік подій	Відсутній.
Постумови	Сформовано видачу книги.

Таблиця 2.6 – Опис варіанту використання "Пошук необхідної інформації"

Опис	Складова
Короткий опис	Дана функція дозволяє користувачу здійснювати пошук необхідної інформації.
Передумови	Користувач авторизований у системі.
Основний потік подій	1. Авторизація 2. Система виводить головне вікно роботи з програмною 3. Користувач обирає таблицю по якій буде відбуватись пошук необхідних даних 4. Користувач водить необхідні дані для пошуку 5. Підтверджує пошук даних.
Альтернативний потік подій	Відсутній.
Постумови	Вивід знайденої інформації.

Таблиця 2.7 – Опис варіанту використання "Формування необхідних звітів"

Опис	Складова
Короткий опис	Призначений для формування звітів.
Передумови	Користувач авторизований в системі. Користувач повинен обрати режим для роботи зі звітами.
Основний потік подій	1. Авторизація 2. Система виводить головне вікно роботи з програмою 3. Користувач обирає режим для роботи зі звітами 4. Система виводить список звітів 5. Користувач обирає потрібний звіт 6. Система виводить форму для роботи зі звітом 7. Користувач заповнює всі потрібні для формування звіту реквізити та натискає «Сформувати» 8. Система обробляє інформацію, якщо нема помилок виводить сформований звіт
Альтернативний потік подій	Відсутній.
Постумови	Сформовано видача книги.

Таким чином, варіанти використання, які наведені у таблицях 2.1 – 2.7, описують основну необхідну функціональність розроблюваного програмного забезпечення.

2.2 Архітектура програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Архітектурна схема програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією побудована за шаблоном архітектури MVC (Model-View-Controller) - найзручнішої для подібного типу програм.

Архітектурна схема програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією наведена на рисунку 2.2.

Призначення такої архітектури - розділити логіку програми на частини, які легко підтримувати й розширювати:

- Model відповідає лише за збереження та обробку даних;
- View можна змінювати, не торкаючись логіки;
- Controller обробляє взаємодії з інтерфейсом.

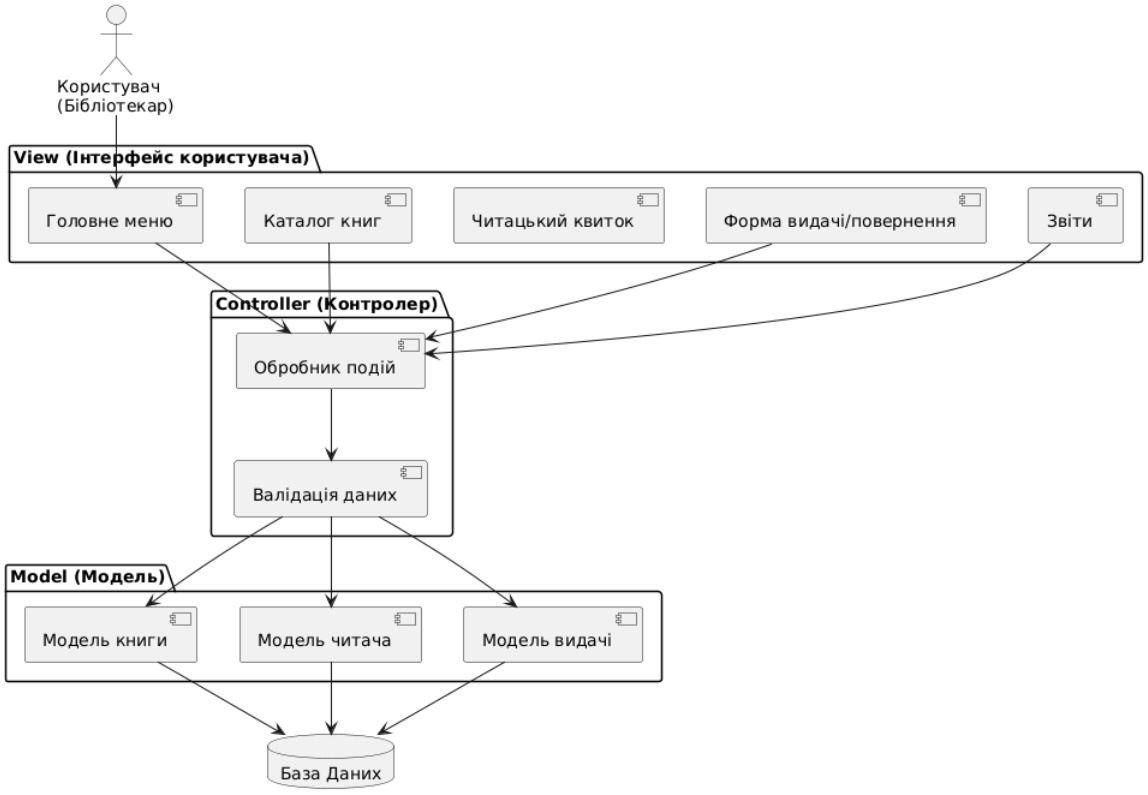


Рисунок 2.2 – Архітектурна схема програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Можна виділити наступні зв’язки між елементами, представленими на рисунку 2.2:

Користувач (Бібліотекар) → Головне меню

Головне меню → Обробник подій

Каталог книг → Обробник подій

Читацький квиток → Обробник подій

Форма видачі/повернення → Обробник подій

Звітність → Обробник подій

Обробник подій → Валідація даних

Валідація даних → Модель книги

Валідація даних → Модель читача

Валідація даних → Модель видачі

Модель книги → База даних

Модель читача → База даних

Модель видачі → База даних

2.3 Проєкт програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

2.3.1 Ескізний проєкт програмного забезпечення

Побудова концептуальної моделі даних

Моделі типу «сутність–зв’язок» (ER-моделі) є зручними тим, що їх розробка відбувається ітераційно. Після створення початкового варіанту моделі, її можна поступово уточнювати, залучаючи експертів предметної області. При цьому сама ER-модель виступає своєрідною документацією, яка фіксує результати таких обговорень [9].

Сутність - це будь-який об’єкт (реальний або абстрактний), що належить до певної предметної області. Сутності є базовими одиницями інформації, які зберігаються в базі даних. У реляційних базах даних кожній сутності, як правило, відповідає окрема таблиця.

Атрибут - це характеристика або властивість сутності в межах предметної області. Назва атрибута повинна бути унікальною в межах відповідного типу сутності.

Зв’язок - це відношення між сутностями, яке відображає їх взаємозалежність. У базі даних зв’язки реалізуються як логічні з’єднання між сутностями, у реляційній моделі - це відношення між записами таблиць.

Концептуальна модель даних програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлена на рисунку 2.3.

Рисунок 2.3 – Концептуальна модель даних програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Атрибути кожної з сутностей представлені нижче.

Книга: ID книги, назва, вид, автор, укладач, місце видання, видавництво, рік випуску, авторський знак.

Примірник книги: ID примірника, ID книги, дата придбання, ціна.

Читацький квиток: номер формуляра, прізвище, ім’я, по батькові, адреса, телефон, статус, клас.

Бронювання книги ID бронювання, ID примірника, номер формуляра, дата бронювання, статус бронювання .

Видача книги: ID видачі, ID примірника, номер формуляра, дата видачі, дата повернення.

В результаті виділено ключові сутності з їх атрибутами, що присутні в концептуальній моделі даних і відносини, які можуть встановлюватися між цими сутностями.

Розробка сценаріїв варіантів використання

Кожен об'єкт системи, що володіє певною поведінкою, може знаходиться в певних станах, переходити зі стану в стан, здійснюючи певні дії в процесі реалізації сценарію поведінки об'єкта. Поведінка більшості об'єктів реальних систем можна представити з точки зору теорії кінцевих автоматів, тобто поведінка об'єкта відбивається в його станах, і даний тип діаграм дозволяє відобразити це графічно [10].

Нижче представлено кілька сценаріїв програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією для таких варіантів використання: «Авторизація», «Оформлення бронювання книги», «Оформлення видачі книги». Відповідні сценарії варіантів використання представлено на рисунках 2.4 – 2.6.

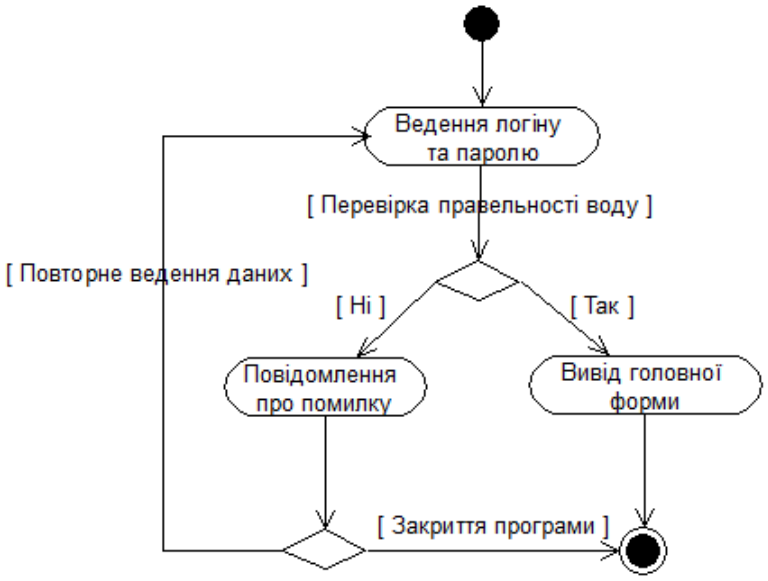


Рисунок 2.4 – Сценарій варіанту використання «Авторизація»

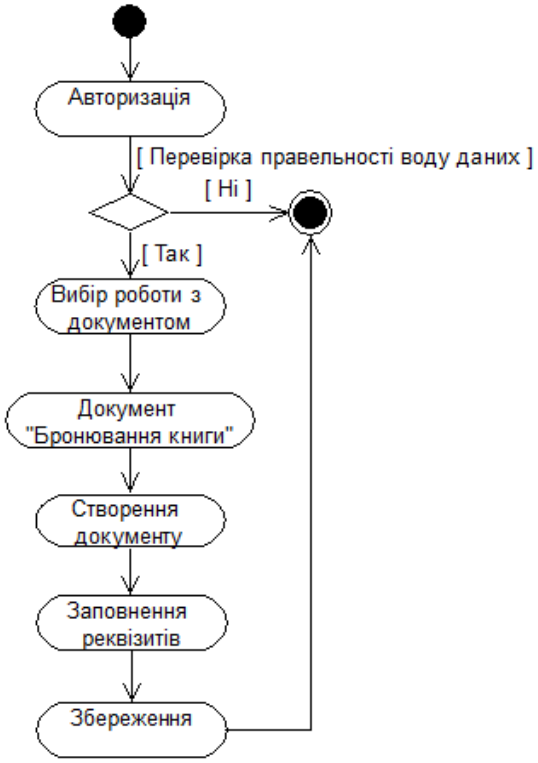


Рисунок 2.5 – Сценарій варіанту використання «Оформлення бронювання книги»

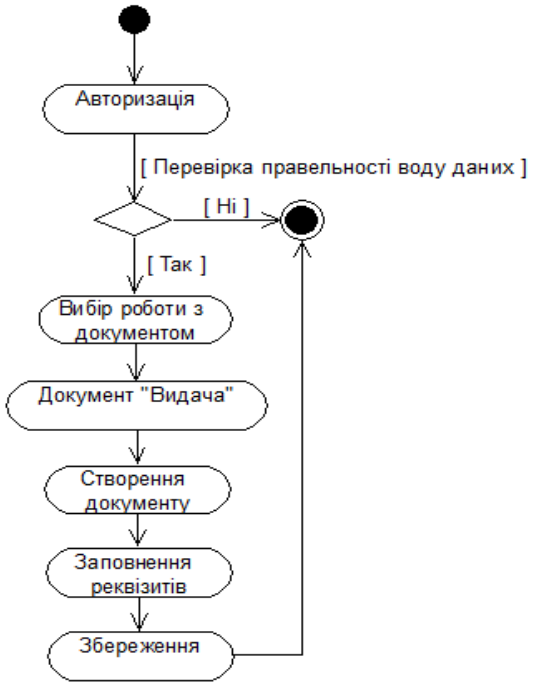


Рисунок 2.6 – Сценарій варіанту використання «Оформлення видачі книги»

Розробка прототипу користувацького інтерфейсу

Користувацький інтерфейс являє собою сукупність програмного та апаратного забезпечення, яке забезпечує ефективну взаємодію між користувачем і обчислювальною системою. Основним способом такої взаємодії є діалог - регламентований процес обміну інформацією між людиною та комп’ютером, що здійснюється у реальному часі та має на меті спільне вирішення певного завдання шляхом передачі даних і координації дій.

Діалогова взаємодія реалізується через серію операцій введення та виведення інформації, які фактично забезпечують зв’язок між користувачем і комп’ютерною системою [11].

На рисунках 2.7 – 2.9 представлено прототипи користувацького інтерфейсу кількох екранних форм програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією.

Рисунок 2.7 – Прототип користувацького інтерфейсу екранної форми «Авторизація»

Рисунок 2.8 – Прототип користувацького інтерфейсу екранної форми «Бронювання книг»

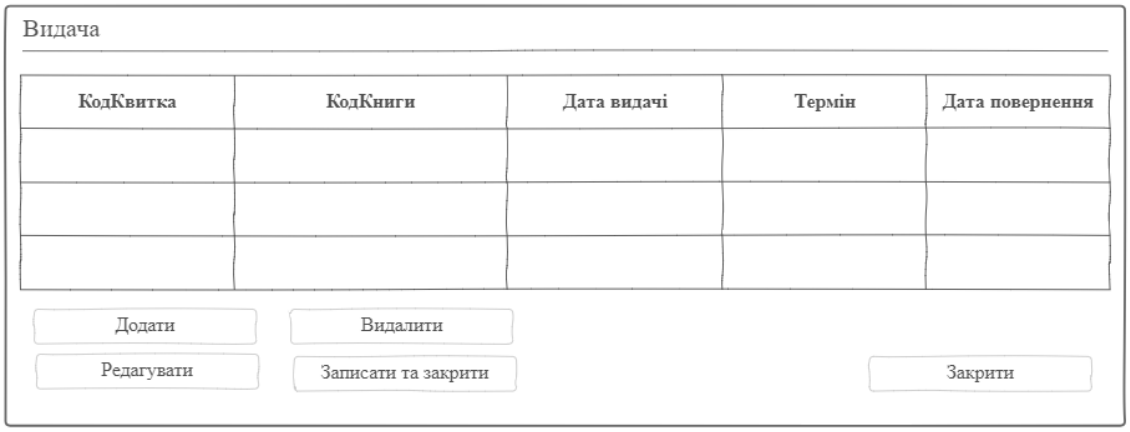


Рисунок 2.9 – Прототип користувацького інтерфейсу екранної форми «Видача»

Таким чином, розроблено прототип користувацького інтерфейсу програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією.

2.3.2 Технічний проєкт програмного забезпечення

Побудова статичної моделі програмного забезпечення

Діаграма класів – статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення [12].

Діаграма класів програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлена на рисунку 2.10.

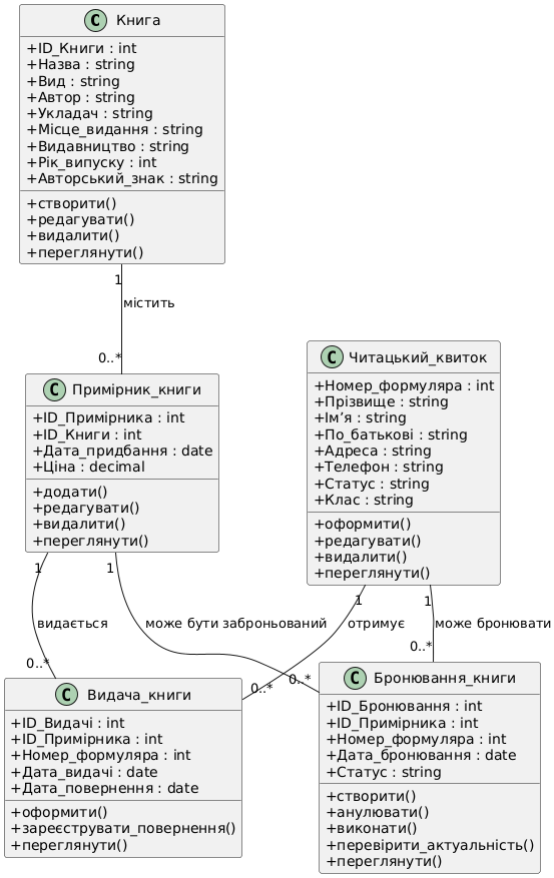


Рисунок 2.10 – Діаграма класів програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Розробка специфікації класів програмного забезпечення

Специфікація класу для об'єкта – це визначення його атрибутів, які характеризують стан об'єкта, та методів, які є способом реалізації його поведінки [12].

Специфікація класів програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією наведена нижче.

Клас "Книга"

Атрибути: ID_Книги, Назва, Вид, Автор, Укладач, Місце_видання, Видавництво, Рік_випуску, Авторський_знак.

Методи: створити(), редагувати(), видалити(), переглянути().

Клас "Примірник книги"

Атрибути: ID_Примірника, ID_Книги, Дата_придбання, Ціна.

Методи: додати(), редагувати(), видалити(), переглянути().

Клас"Читацький квиток"

Атрибути: Номер_формуляра, Прізвище, Ім’я, По_батькові, Адреса, Телефон, Статус, Клас.

Методи: оформити(), редагувати(), видалити(), переглянути().

Клас "Бронювання книги"

Атрибути: ID_бронювання, ID_примірника, Номер_формуляра, Дата_бронювання, Статус_бронювання .

Методи: створити(), анулювати(), виконати(), перевірити_актуальність(), переглянути().

Клас "Видача книги"

Методи: ID_видачі, ID_примірника, Номер_формуляра, Дата_видачі, Дата_повернення.

Методи: оформити(), зареєструвати_повернення(), переглянути().

Побудова динамічної моделі програмного забезпечення

Динамічна модель програмного забезпечення може бути представлена у вигляді діаграми діяльності, діаграми послідовності, діаграми взаємодії або діаграми станів. Динамічну модель програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представимо у вигляді діаграми послідовності - діаграми, яка показує взаємодії об'єктів, упорядковані за часом їхнього прояву [13].

Діаграми послідовності для деяких варіантів використання програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлені на рисунках 2.11 – 2.13.

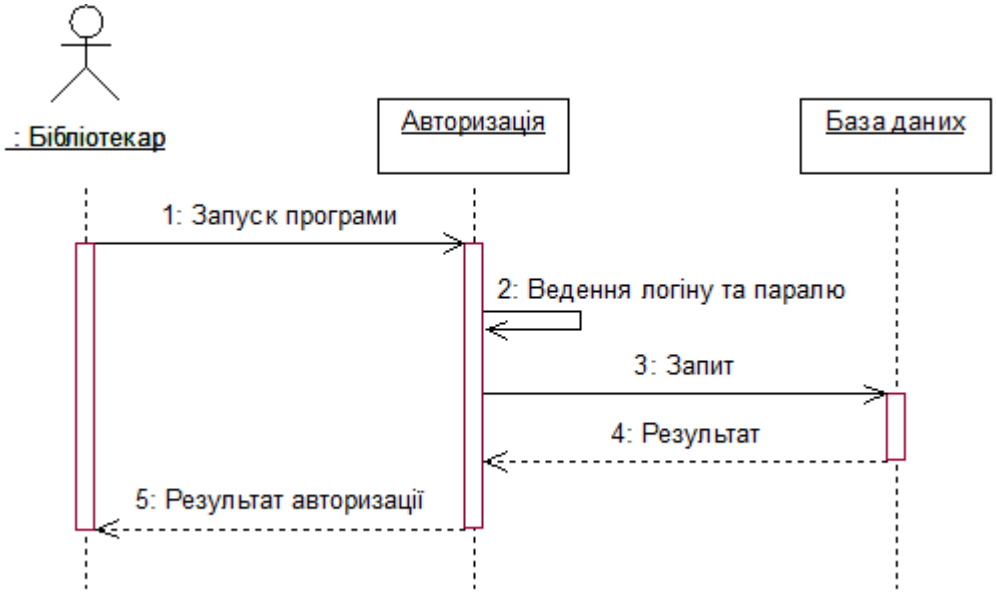


Рисунок 2.11 – Діаграма послідовності для варіанту використання «Авторизація»

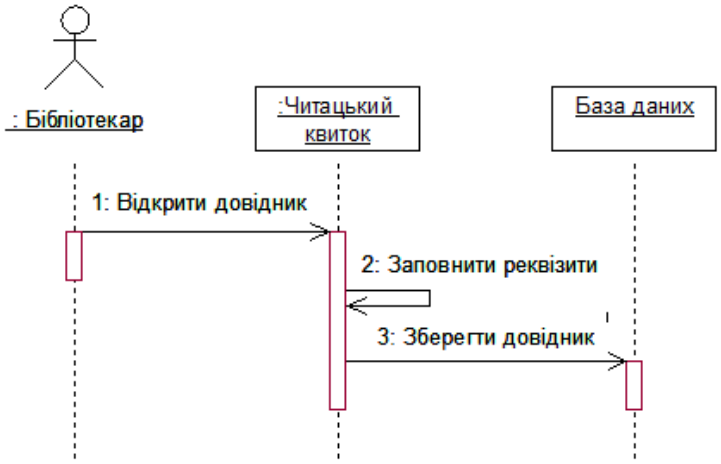


Рисунок 2.12 – Діаграма послідовності для варіанту використання заповнення довідника «Оформлення читацького квитка»

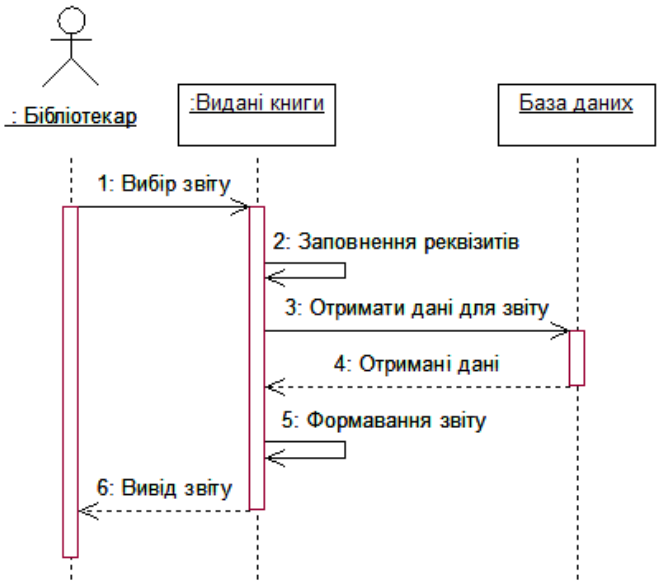


Рисунок 2.13 – Діаграма послідовності для варіанту використання «Формування необхідних звітів»

Побудова логічної моделі бази даних

Перед тим як створювати таблиці бази даних, форми та інші об'єкти, потрібно задати структуру бази даних. Логічна модель є основою майбутньої

побудови фізичної бази даних, вона повинна відображати взаємозв’язки між реляційними таблицями.

На основі попередньо розроблених концептуальної моделі даних, діаграми класів та сценаріїв варіантів використання програмного забезпечення спроєктуємо логічну модель бази даних у вигляді чітко структурованих таблиць з назвами, атрибутами, типами даних та зв’язками.

Логічна модель бази даних програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлена на рисунку 2.14.

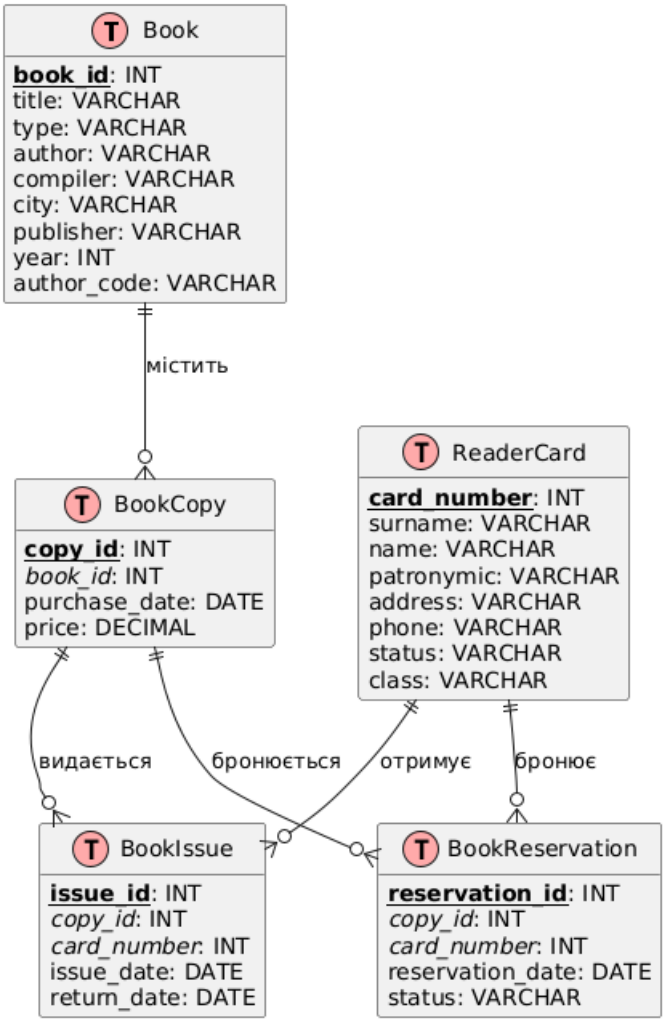


Рисунок 2.14 – Логічна модель бази даних програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Детальний опис таблиць розробленої бази даних програмного забезпечення представлено у пункті 2.3. Робочий проєкт програмного забезпечення.

2.3.3 Робочий проєкт програмного забезпечення

Обґрунтування вибору стеку технологій для розробки

В якості мови програмування для розробки програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією обрана мова Java [14].

В якості СКБД для розробки програмного забезпечення була обрана СКБД MySQL - надійна та безкоштовна СКБД з підтримкою багатьох інструментів [15].

Використання поєднання Java та MySQL - це відмінна комбінація для десктопної програми для операційної системи Windows, яка дозволяє:

- мати універсальність розробки;
- використовувати методологію ООП та багаторівневу архітектуру;
- реалізувати потужну та масштабовану систему з графічним інтерфейсом.

До того ж, така комбінація - це гарна основа для розширення та масштабування програмного забезпечення у майбутньому.

В якості графічного інтерфейсу було обрано JavaFX, в якості драйверу бази даних - mysql-connector-java.

В якості інтегрованого середовища для розробки та налагодженя програмного забезпечення було обрано IntelliJ IDEA [16] та MySQL Workbench - для адміністрування бази даних.

Обрана раніше в якості архітектури модель MVC виглядає наступним чином:

View - JavaFX UI: форми, таблиці, кнопки;

Controller - обробники подій, логіка UI;

Model - класи сутностей (Book, Reader, Issue і т.д.);

DB - MySQL база з таблицями з ER-моделі;

DAO - окремий шар для доступу до MySQL через JDBC.

Побудова фізичної моделі бази даних

Детальний опис таблиць розробленої фізичної моделі бази даних програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлено у таблицях 2.8 - 2.12.

Таблиця 2.8 – Опис структури таблиці БД «Книга» (Book)

Ім’я поля	Тип даних	Ключ
book_id	int	PK
title	varchar(255)	
type	varchar(100)	
author	varchar(255)	
compiler	varchar(255)	
city	varchar(100)	
publisher	varchar(255)	
year	int	
author_code	varchar(20)	

Таблиця 2.9 – Опис структури таблиці БД «Примірник книги» (BookCopy)

Ім’я поля	Тип даних	Ключ
copy_id	int	PK
book_id	int	FK
purchase_date	date	
price	decimal(8,2)	

Таблиця 2.10 – Опис структури таблиці БД «Читацький квиток» (ReaderCard)

Ім’я поля	Тип даних	Ключ
card_number	int	PK
surname	varchar(100)	
name	varchar(100)	
patronymic	varchar(100)	
address	varchar(255)	
phone	varchar(20)	
status	varchar(50)	
class	varchar(10)	

Таблиця 2.11 – Опис структури таблиці БД «Видача книги» (BookIssue)

Ім’я поля	Тип даних	Ключ
issue_id	int	PK
copy_id	int	FK
card_number	int	FK
issue_date	date	
return_date	date	

Таблиця 2.12 – Опис структури таблиці БД «Бронювання книги» (BookReservation)

Ім’я поля	Тип даних	Ключ
reservation_id	int	PK
copy_id	int	FK
card_number	int	FK
reservation_date	date	
status	varchar(20)	

Тестування основних класів програмного забезпечення

Для тестування програмного забезпечення зазвичай використовуються такі основні групи методів:

- методи чорного ящика: методи тестування по формальним специфікаціям;

- методи білого ящика: методи структурного тестування.

Методи структурного тестування вимагають значних затрат ресурсів, тому в кваліфікаційній роботі виконувалось тестування по формальним специфікаціям (методи чорного ящика), а саме розбиття на класи еквівалентності.

При цьому програма розглядається як чорний ящик, а тестування зводиться до послідовного вводу тестових наборів даних та аналізу отриманих результатів.

В таблиці 2.13 представлені класи еквівалентності та граничні значення вхідних даних класу «Читацький квиток» (ReaderCard).

Таблиця 2.13 – Класи еквівалентності вхідних даних для класу «Читацький квиток» (ReaderCard)

Вхідні умови	Правильні класи еквівалентності	Неправильні класи еквівалентності
Прізвище	Рядок	Графічні зображення (в тому числі і смайлики)
Ім’я	Рядок	Графічні зображення (в тому числі і смайлики)
По-батькові	Рядок	Графічні зображення (в тому числі і смайлики)
Адреса	Рядок	Графічні зображення (в тому числі і смайлики)
Телефон	Рядок	Графічні зображення (в тому числі і смайлики)
Статус	Рядок	Графічні зображення (в тому числі і смайлики)
Клас	Рядок	Графічні зображення (в тому числі і смайлики)

Почнемо з тестування для правильних класів еквівалентності класу «Читацький квиток» (ReaderCard), наведених в табл. 2.14.

Таблиця 2.14 – Тести з правильних класів еквівалентності для класу «Читацький квиток» (ReaderCard)

Номер тесту	Вхідні умови	Правильні класи еквівалентності	Вихідні дані
1	Прізвище	В поле «Прізвище» введено прізвище читача – Коваленко.	Вивід на екран прізвище читача. Результат вірний.
2	Ім’я	В поле «Ім’я» введено ім’я читача – Дмитро.	Вивід на екран ім’я читача. Результат вірний.
3	По-батькові	В поле «По-батькові» введено по-батькові читача – Петрович.	Вивід на екран по-батькові читача. Результат вірний.
4	Адреса	В поле «Адреса» введено адресу читача – пр. Центральний, 19, кв. 19.	Вивід на екран адреси читача. Результат вірний.
5	Телефон	В поле «Телефон» введено номер телефону читача – 0501234567.	Вивід на екран номеру телефону читача. Результат вірний.
6	Статус	В поле «Статус» введено статус читача – учень	Вивід на екран статуса читача. Результат вірний.
7	Клас	В поле «Клас» введено клас читача – 5А.	Вивід на екран класа читача. Результат вірний.

Отже, за результатами тестування правильних класів еквівалентності класу «Читацький квиток» (ReaderCard) програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією працює правильно і без збоїв.

Наступним кроком буде перевірка неправильних класів еквівалентності класу «Читацький квиток» (ReaderCard), наведених в табл. 2.15.

Таблиця 2.15 – Тести з неправильних класів еквівалентності класу «Читацький квиток» (ReaderCard)

Номер тесту	Вхідні умови	Неправильні класи еквівалентності	Вихідні дані
1	Прізвище	В поле «Прізвище» введемо будь-який псевдографічний символ ASCII	Програма виводить повідомлення про помилку. Результат підтверджено.
2	Ім’я	В поле «Ім’я» ведемо графічний елемент – смайлик	Після ведення смайлика виводиться повідомлення про помилку. Програма далі працює правильно. Результат підтверджено
3	По-батькові	В поле «По-батькові» ведемо будь-які цифри.	Після ведення цифр виводиться повідомлення про помилку. Програма далі працює правильно. Результат підтверджено
4	Адреса	В поле «Адреса» ведемо будь-який псевдографічний символ ASCII.	Програма виводить повідомлення про помилку. Результат підтверджено
5	Телефон	В поле «Телефон» ведемо символ ASCII, відмінний від цифри	Програма виводить повідомлення про помилку. Результат підтверджено
6	Статус	В поле «Статус» ведемо графічний елемент – смайлик	Після ведення смайлика виводиться повідомлення про помилку. Програма далі працює правильно. Результат підтверджено
7	Клас	В поле «Клас» ведемо графічний елемент – смайлик	Після ведення смайлика виводиться повідомлення про помилку. Програма далі працює правильно. Результат підтверджено

Отже, за результатами тестування неправильних класів еквівалентності класу «Читацький квиток» (ReaderCard) програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією працює

правильно, не допускаючи до введення будь-яких інших елементів, окрім стандартних символів вводу. Тестування інших класів виконується аналогічно.

Випробування програмного забезпечення

Під час випробування програмного забезпечення було проведено повне функціональне тестування, а також навантажувальне тестування і тестування на відмову всього програмно-апаратного комплексу.

Випробування програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією проводилося на цільовому обладнанні в тій конфігурації, яка заявлена у Технічному завданні, яке наведено у Додатку А.

Всі виявлені недоліки програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією були зафіксовані в протоколах і усунуті розробником до моменту впровадження програмного забезпечення в дію.

Випробування було проведено за стратегією тестування по формальним специфікаціям (чорного ящика). За рахунок великої кількості функцій, які потрібно випробовувати, представлено декілька результатів випробувань функцій програмного забезпечення.

Функція «Оформлення читацького квитка»

На головному вікні програми обрали довідник «Читацький квиток» та натиснули кнопку «Додати». З’явилось відповідне вікно, у якому обрали читача, внесли усі необхідні реквізити та натисли кнопку «Закрити та зберегти». У результаті отримали сформований новий читацьких квиток.

Функція «Оформлення бронювання книги»

На головному вікні програми обрали документ «Бронювання книги» та натиснули кнопку «Додати». З’явилось відповідне вікно, у якому обрали читача,

книгу та натисли кнопку «Закрити та зберегти». У результаті отримали сформоване нове бронювання книги.

Функція «Оформлення видачі книги»

На головному вікні обрали документ «Видача книги» та натиснули кнопку «Додати». З’явилось відповідне вікно, у якому обрали читача, книгу та натиснули кнопку «Закрити та зберегти». У результаті отримали сформовану нову видачу книги.

Функція «Формування звіту (Видані книги)»

В головному меню натиснули на кнопку «Звіти». З’явилося вікно з можливими звітами. Вибрали звіт «Видані книги» та виконали встановлені початкові дії (вибрали відповідні параметри звіту). В результаті був сформований новий звіт «Видані книги».

Повністю програма та методика випробувань програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією наведено у Додатку Д.

3 РЕЗУЛЬТАТИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ШКІЛЬНОЇ БІБЛІОТЕКИ КИР'ЯКІВСЬКОГО ЛІЦЕЮ З ПОЧАТКОВОЮ ШКОЛОЮ ТА ГІМНАЗІЄЮ

В кваліфікаційній роботі була розв’язана практична задача інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов, а саме розроблено програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією.

Розроблене програмне забезпечення забезпечує автоматизоване ведення обліку бібліотечного фонду та читачів, а також дозволяє формувати заявки на бронювання та видачу книг, створювати необхідні звіти.

У процесі роботи здійснено аналіз літературних джерел та матеріалів з мережі Інтернет з метою дослідження діяльності бібліотеки та можливостей її автоматизації.

Також проведено огляд і аналіз наявних програмних рішень у цій сфері, обґрунтовано доцільність розробки власного програмного забезпечення та сформульовано задачу автоматизації діяльності шкільної бібліотеки.

Були розроблені архітектура та проєкт програмного забезпечення. Розроблений ескізний проєкт містить концептуальну модель даних, сценарії варіантів використання та прототип користувацького інтерфейсу. Розроблений технічний проєкт містить: статичну модель програмного забезпечення, специфікації класів програмного забезпечення та динамічну модель програмного забезпечення. Розроблений робочий проєкт містить: обґрунтування вибору стеку технологій для розробки, фізичну модель бази даних, кодування, тестування та випробування програмного забезпечення.

Програмне забезпечення було розроблено з використанням мови програмування Java, СУБД MySQL, бібліотеки JavaFX та драйверу бази даних mysql-connector-java.

Зовнішній вигляд головного вікна програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією наведено на рисунку 3.1.

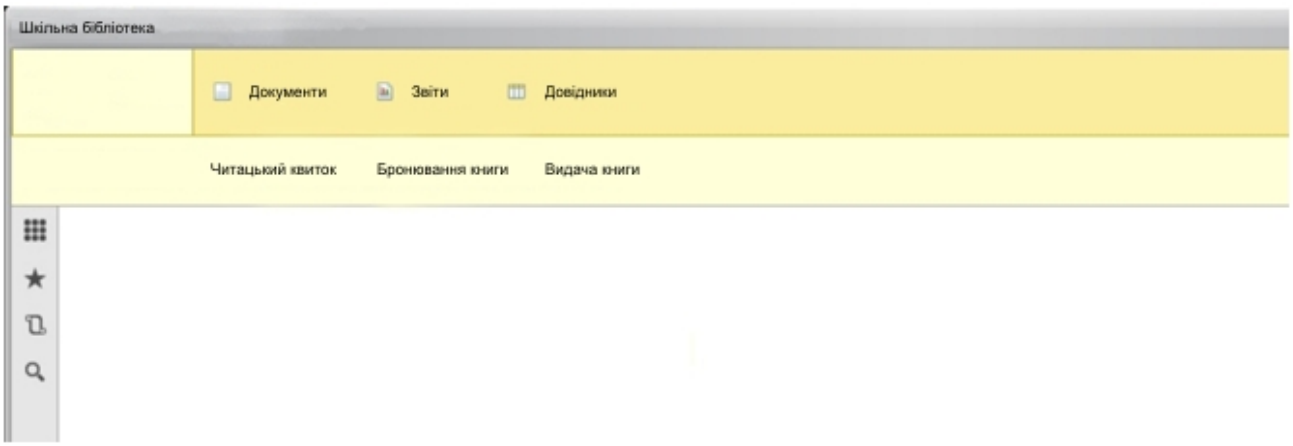


Рисунок 3.1 – Зовнішній вигляд головного вікна програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Результатом роботи являється програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією з можливістю ведення обліку книг та читачів, що дозволило не лише зменшити навантаження на бібліотекаря, а й забезпечити більш якісне та швидке обслуговування читачів.

Технічне завдання на розробку програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлено у Додатку А.

Код програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлений у Додатку Б.

Опис програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлено у Додатку В.

Інструкція користувача програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлена у Додатку Г.

Програма та методика випробування програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією представлена у Додатку Д.

4 ОХОРОНА ПРАЦІ

4.1 Вступ

Охорона праці - це цілісна система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і профілактичних заходів, спрямованих на збереження життя, здоров’я та працездатності працівників у процесі виконання ними трудових обов’язків [17].

Актуальність теми зумовлена складною ситуацією у сфері охорони праці в Україні. Незважаючи на наявність великої кількості законодавчих актів, більшість із них не виконується на практиці. Значна частина трудящих працює без офіційного оформлення, що унеможливлює захист їхніх трудових прав у законний спосіб.

Система охорони праці включає три ключові компоненти:

- норми трудового законодавства,
- техніку безпеки, виробничу санітарію та гігієну,
- заходи з протипожежного захисту й електробезпеки [18].

Основна мета охорони праці - створення безпечних, нешкідливих і комфортних умов праці шляхом вирішення комплексу важливих завдань, зокрема:

- проєктування підприємств, обладнання та технологічних процесів із дотриманням вимог охорони праці;
- досягнення оптимального балансу між факторами виробничого середовища з метою зниження їх негативного впливу на працівників;
- нормативне закріплення гранично допустимих рівнів шкідливих і небезпечних виробничих факторів, а також забезпечення контролю за їх дотриманням;

- впровадження інноваційних заходів для покращення умов праці з урахуванням досягнень науки й техніки;

- застосування ефективних засобів індивідуального та колективного захисту, а також організаційних заходів, що зменшують дію шкідливих чинників;

- розробка й упровадження методик для оцінки ефективності реалізованих заходів з охорони праці.

4.2 Аналіз шкідливих та небезпечних факторів при роботі на персональному комп'ютері

У процесі роботи за персональним комп'ютером працівники піддаються впливу ряду фізичних та психологічних факторів, які можуть негативно позначатися на їхньому здоров'ї. Серед основних загроз виділяють: підвищену температуру в робочому приміщенні, недостатню або неприродну освітленість, дію електричного струму та статичної електрики, візуальне й розумове перенапруження, монотонність діяльності, а також емоційне виснаження.

Вплив випромінювання моніторів

При роботі дисплейних пристроїв виникають електростатичне та електромагнітне випромінювання. Електростатичне поле сприяє накопиченню пилу навколо монітора, що може викликати подразнення шкіри. Електромагнітне випромінювання, яке генерується магнітними котушками в основі електронно-променевих трубок (ЕПТ), при тривалому впливі знижує загальну продуктивність праці оператора.

Відповідно до сучасних вимог, частота вертикальної розгортки ЕПТ-моніторів повинна становити не менше 75 Гц, що зменшує навантаження на зір.

Мікрокліматичні умови

Температурно-вологісні параметри в приміщенні впливають не лише на самопочуття працівників, але й на надійність роботи комп’ютерної техніки. Комфортною вважається температура близько 23 °C влітку та 18 °C взимку при відносній вологості повітря 55%. Порушення мікроклімату може спричинити перегрів обладнання або погіршення самопочуття персоналу.

Ураження електричним струмом

Обладнання, яке використовується в комп’ютерних системах, є електроустановками і становить потенційну загрозу у разі порушення правил безпеки. Особливо небезпечно торкатися струмопровідних частин під час технічного обслуговування або ремонту. Дослідження показують, що змінний струм силою 0,6–1,5 мА і постійний струм 5–7 мА викликають відчутне подразнення, але не становлять загрози життю. Водночас змінний струм понад 6 мА вже ускладнює самостійне вивільнення від контакту, а постійний струм понад 15 мА викликає сильний біль і може бути небезпечним.

Освітлення робочого місця

Освітлення в робочому приміщенні суттєво впливає на працездатність і комфорт користувача. Недостатнє або неправильно організоване освітлення призводить до перевтоми зору та швидкої втоми. Оптимальний рівень освітленості для роботи за дисплеєм становить 400–750 люксів, а співвідношення яскравості між екраном і навколишнім простором має бути в межах 1:5 – 1:10.

Пожежна безпека

Приміщення з обчислювальною технікою належать до категорії «В» за пожежною небезпекою. Основні потенційні джерела займання: кабельні лінії з напругою 220 В, елементи живлення, електронно-променеві трубки моніторів, легкозаймисті меблі та паперові носії. Наявність великої кількості обладнання

потребує систематичного контролю за станом електромережі та забезпечення приміщення засобами пожежогасіння.

4.3 Розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів при роботі на персональному комп’ютері

Під час розробки заходів з охорони праці враховують усі потенційно небезпечні та шкідливі фактори, прагнучи зменшити їхній вплив до безпечного рівня. Однак тільки суворе дотримання правил техніки безпеки та охорони праці створює належні умови праці й зменшує ризик нещасних випадків.

Вибір обладнання

При закупівлі моніторів рекомендується обирати моделі із сертифікацією щодо зниженої електрополя, електромагнітного випромінювання та частотою вертикальної розгортки не менш ніж 75 Гц.

Мікроклімат

Для підтримки комфортних умов в приміщенні рекомендується:

- встановлення кондиціювання або вентиляції;
- регулярні вологі прибирання, що зменшують кількість пилу.

Електробезпека

Безпека роботи з електрообладнанням забезпечується наступними процедурами:

- допуск до виконання робіт лише після інструктажу;
- нагляд під час обслуговування;
- відключення обладнання перед ремонтними роботами;
- розміщення попереджувальних плакатів;
- перевірка відсутності напруги й встановлення заземлення.

Для безпеки працівників:

- усі металеві частини мають бути заземлені;

- в машинних приміщеннях рекомендується підлога з технічним підпіллям.

Для запобігання накопиченню статичного заряду використовують антистатичний лінолеум (марки ASN), а також прилади для загального чи місцевого зволоження або іонізації повітря.

Освітлення

При організації дисплейного залу слід:

- забезпечити одностороннє бічне освітлення, розташовуючи вікна у північному або північно-східному напрямку;

- використовувати спеціальні штори чи жалюзі для запобігання прямих сонячних променів;

- встановлювати монітори так, щоб джерела світла падали сбоку;

- у разі необхідності застосовувати штучне освітлення люмінесцентними лампами потужністю 40–80 Вт (типу ЛБ або ЛТБ), з рівнем освітленості 400–750 лк та співвідношенням яскравості екрану до навколишнього простору 1:5–1:10;

- розміщувати світильники уздовж лінії зору оператора та паралельно вікнам, щоб уникнути світлових смуг і тіней.

Пожежна безпека

Приміщення з комп'ютерною технікою належить до категорії пожежної безпеки «В». Для запобігання пожежі необхідно:

- дотримуватися протипожежних норм при проектуванні вентиляції;

- використовувати електрообладнання згідно із правилами експлуатації;

- встановити пожежну сигналізацію та системи пожежогасіння (датчики ЛІП-1, ІП-105; АУП);

- мати інструкції з евакуації, план дій у складі ППБ;

- облаштувати підлогу з негорючих матеріалів та технологічним покриттям;

- зберігати папір у металевих шафах;

- встановити щонайменше два вогнегасники типу ОУ-5 і два димові датчики.

Загальні правила безпеки

Користувачі мають:

- дотримуватися внутрішнього трудового розпорядку;

- не приносити й не споживати алкогольні напої;

- інформуватися про техніку безпеки перед початком роботи.

На території підприємства необхідно:

- пересуватися тротуарами і йти назустріч руху транспорту;

- не перебігати проїжджу частину;

- уникати небезпечних зон, де ведуться ремонтні роботи.

В машинному залі мають бути рівна, неслизька підлога, захищені кабелі та закриті люки. У разі виявлення загоряння слід терміново відключити електроживлення та повідомити про інцидент.

Безпека на початку, під час та після роботи

До початку роботи оператор повинен:

- перевірити чистоту робочого місця;

- ознайомитися з дефектами, що зазначені попереднім персоналом;

- отримати завдання від керівника;

- пройти інструктаж з техніки безпеки;

- підготувати інструменти та перевірити їх справність;

- оглянути кабелі, мережі на відсутність замикань;

- впевнитися в наявності справних запобіжників.

Під час роботи категорично забороняється:

- залишати включений персональний комп’ютер без нагляду;

- монтаж або демонтаж деталей при ввімкненому обладнанні;
- користування несправними інструментами;
- підключати або від’єднувати контакти під напругою;
- знімати захисні кришки, міняти запобіжники або паяти без заземлення;
- виконувати роботи з електрикою без присутності другого працівника.

Після завершення роботи необхідно:

- відключити електроживлення;
- прибрати робоче місце;
- звітувати керівнику про виконану роботу;
- зробити запис у журналі щодо роботи з персональним комп’ютером.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи було вирішено практичну задачу інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов - розроблено програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією. Розроблене програмне забезпечення дозволило автоматизувати, спростити та пришвидшити роботу бібліотекаря та полегшити ведення звітності.

У процесі роботи було здійснено аналіз літературних та Інтернет джерел, проведено огляд і аналіз наявних програмних рішень у сфері автоматизації роботи бібліотек, обґрунтовано доцільність розробки власного програмного забезпечення та сформульовано постановку задачу на розробку програмного забезпечення.

Були розроблені архітектура програмного забезпечення, проєкт програмного забезпечення, який містить ескізний проєкт з концептуальною моделлю даних, сценаріями варіантів використання, прототипом користувацького інтерфейсу; технічний проєкт зі статичною моделлю програмного забезпечення, специфікаціями класів програмного забезпечення, динамічною моделлю програмного забезпечення; робочий проєкт з обґрунтуванням вибору стеку технологій для розробки програмного забезпечення, фізичною моделлю бази даних, кодуванням, тестуванням та випробуванням програмного забезпечення.

Мета роботи досягнута, усі поставлені задачі виконано. Подальша робота може бути пов’язана з удосконаленням роботи програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією та збільшенням його функціональності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Програмне забезпечення бібліотеки. URL: https://library.gov.ua/prohramne-zabezpechennia-bibliotek/ (дата звернення: 20.03.2025).

2. Безкоштовні АБІС – вільне відкрите програмне забезпечення. URL: http://nbuv.gov.ua/node/1336 (дата звернення: 20.03.2025).

3. УФД/Бібліотека. URL: https://www.ush.com.ua/# (дата звернення: 20.03.2025).

4. Koha - автоматизована інтегрована бібліотечна система. URL: https://koha.org.ua/ (дата звернення: 20.03.2025).

5. Опис роботи бібліотеки. URL: http://conference.nbuv.gov.ua/report/view/id/177 (дата звернення: 20.03.2025).

6. Evergreen. Open Source Library Software. URL: http://evergreen-ils.org/ (дата звернення: 20.03.2025).

7. OpenBiblio. URL: https://sourceforge.net/projects/obiblio/ (дата звернення: 20.03.2025).

8. Діаграма варіантів використання. URL: http://moodle.ipo.kpi.ua/moodle/mod/resource/view.php?id=28086 (дата звернення: 25.04.2025).

9. Елементи моделі «сутність-зв'язок». URL: http://easy-code.com.ua/2012/09/elementi-modeli-sutnist-zvyazok-integraciya-dodatkiv-i-danix-bazi-danix-statti/ (дата звернення: 25.04.2025).

10. Діаграма станів. URL: http://moodle.ipo.kpi.ua/moodle/mod/resource/view.php?id=2808710 (дата звернення: 27.04.2025).

11. Інтерфейс користувача. URL: http://sven.ua/ua/service/glossary/ukr/11/ (дата звернення: 27.04.2025).

12. Діаграма класів. URL: http://moodle.ipo.kpi.ua/moodle/mod/resource/view.php?id=28088 (дата звернення: 29.04.2025).

13. Діаграма послідовності. URL: http://moodle.ipo.kpi.ua/moodle/mod/resource/view.php?r=16538 (дата звернення: 29.04.2025).

14. Java. URL: https://www.java.com/ (дата звернення: 05.05.2025).

15. MySQL. The world's most popular open source database. URL: https://www.mysql.com/ (дата звернення: 05.05.2025).

16. IntelliJ IDEA IDE для профійної розробки на Java та Kotlin. URL: https://www.jetbrains.com//idea/# (дата звернення: 05.05.2025).

17. Про охорону праці : Закон України від 21.11.2002 р. № 229-IV. Дата оновлення: 04.02.2021. URL: https://zakon.rada.gov.ua/laws/main/2694-12#Text. (дата звернення: 12.05.2025).

18. Поняття, мета і завдання охорони праці. URL: http://ncpn.net.ua/oxrana_truda.html (дата звернення: 12.05.2025).

Додаток А Технічне завдання на розробку програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Вступ

Назва розроблюваного проекту: «Програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією».

Галузь застосування – Кир'яківський ліцей з початковою школою та гімназією.

1 Підстава для розробки

Розробка програмного забезпечення виконується на підставі завдання на кваліфікаційну роботу, виданого кафедрою ПЗАС НУК імені адмірала Макарова.

2 Призначення розробки

2.1 Експлуатаційне призначення

Експлуатаційним призначенням програмного забезпечення є підвищення ефективності функціонування шкільної бібліотеки та поліпшення якості обслуговування читачів, ефективне управління фондом бібліотеки, облік дій користувачів та формування звітності.

2.2 Функціональне призначення

Функціональним призначенням програмного забезпечення є автоматизація роботи працівників шкільної бібліотеки, забезпечення оперативного отримання інформації про наявність книг; формування читацького квитку; оформлення бронювання та видачі книг.

3 Вимоги до програмного забезпечення

3.1 Вимоги до функціональних характеристик

3.1.1 Вимоги до складу виконуваних функцій

Програмне забезпечення повинно виконувати наступні функції:

- авторизація;
- занесення даних про книгу;
- редагування даних про книгу;
- перегляд інформації про книгу;
- видалення інформації про книгу;
- оформлення читацького квитка;
- редагування читацького квитка;
- перегляд інформації про читацький квиток;
- видалення читацького квитка;
- оформлення бронювання книги;
- оформлення видачі книги;
- пошук необхідної інформації;
- формування необхідних звітів.

3.1.2 Вимоги до організації вхідних та вихідних даних

Дані зберігаються в файлі БД з довільним доступом типу .myd.

Введення даних здійснюється за допомогою пристроїв введення даних (клавіатури та миші). Дані або вводяться вручну, або обираються із представлених списків.

Виведення даних здійснюється за допомогою пристроїв виведення даних (монітор та принтер).

3.2 Вимоги до надійності

Програмний продукт повинен функціонувати при безперебійній роботі комп’ютера. При виникненні збоїв в роботі комп’ютера, відновлення нормальної роботи повинне проводитися після перезавантаження програмного продукту.

Для забезпечення надійності інформації повинна використовуватися СКБД, що забезпечує цілісність транзакцій і несе відповідальність за цілісність інформації.

У разі введення користувачем некоректної інформації програмний продукт повинен повідомити про помилку і надати можливість виправити її.

3.3 Вимоги до умов експлуатації

Необхідний рівень підготовки користувачів: мінімальні навики в користуванні комп’ютером. Комп’ютер призначений для роботи в закритому опалювальному приміщені.

3.4 Вимоги до складу та параметрів технічних засобів

Система повинна задовольняти вказаним вимогам на комп’ютері наступної мінімальної комплекції:

– CPU: Intel(R) i5 2,16 МГц або аналогічний за параметрами AMD;

– RAM: 4 GB;

– HDD: 10 GB вільного місця.

3.5 Вимоги до інформаційної та програмної сумістності

Для повноцінного функціонування системи необхідно використовувати ОС Windows версії не нижче 7.

Необхідна наявність встановленого Необхідна наявність встановленого JDK 17.

3.6 Вимоги до маркування та пакування

Програмне забезпечення буде упаковано в електронний архів і мати найменування school_library.zip та містити:

– виконуваний файл school_library.jar ;
– файл, який містить структуру БД school_library.sql;
– файл з описом ReadMe.txt.

4 Вимоги до програмної документації

Програмна документація повинна містити:

– технічне завдання;
– код програми;
– опис програми;
– інструкцію користувача;
– програму і методику випробування програми.

5 Стадії та етапи розробки

Стадії та етапи розробки програмного забезпечення представлені в таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки програмного забезпечення

№ п/п	Назва етапів роботи	Кінцевий термін виконання
1.	Аналіз практичної задачі інженерії ПЗ	15.02.2025
2.	Аналіз вимог до ПЗ	25.03.2025
3.	Розробка архітектури ПЗ	05.04.2025
4.	Розробка проєкту ПЗ	20.04.2025
5.	Реалізація та тестування ПЗ	20.05.2025

6 Порядок контролю та прийому

Контроль за аналізом та проектуванням кожної окремої частини програмного забезпечення відбувається на кожному етапі з урахуванням вимог, визначених у технічному завдані. Кожна стадія розробки повинна бути представлена в зазначені строки та узгоджена із замовником.

Прийом проводять відповідно до програми і методики випробувань і документують за допомогою протоколу проведення випробувань. У разі знаходження помилок підчас прийому програмного виробу складається акт про знайдені помилки, який підписуються представниками замовника і розробника і затверджуються керівником організації – замовника та організації – розробника. Розробник повинен на протязі не більше ніж 2 тижнів виправити зазначені помилки і оповістити замовника про повторне проведення перевірки.

Додаток Б Код програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

```
CREATE TABLE BookReservation (
    reservation_id SERIAL PRIMARY KEY,
    copy_id INT REFERENCES Примірник_книги(copy_id),
    card_number INT REFERENCES Читацький_квиток(card_number),
    reservation_date DATE NOT NULL,
    status VARCHAR(20) DEFAULT 'активне' -- активне, анульоване,
виконане
);

CREATE OR REPLACE FUNCTION Cancel_Reservation ()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE BookReservation
    SET status = 'анульоване'
    WHERE copy_id = NEW. copy_id
      AND status = 'активне';

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER Trigger_ Cancel_Reservation
AFTER INSERT ON BookIssue
FOR EACH ROW
EXECUTE FUNCTION Cancel_Reservation ();

UPDATE BookReservation
SET status = CASE
    WHEN card_number = NEW. card_number THEN 'виконане'
    ELSE 'анульоване'
END
WHERE copy_id = NEW. copy_id
  AND status = 'активне';

CREATE VIEW Current_Reservation AS
SELECT * FROM BookReservation
WHERE status = 'активне'
  AND reservation_date >= CURRENT_DATE - INTERVAL '3 days';

CREATE OR REPLACE FUNCTION Cancel_Expired_Reservation ()
RETURNS void AS $$
BEGIN
```

```
  UPDATE BookReservation
  SET status = 'анульоване'
  WHERE status = 'активне'
    AND reservation_date < CURRENT_DATE - INTERVAL '3 days'
    AND copy_id NOT IN (
      SELECT copy_id FROM BookIssue
      WHERE issue_date >= reservation_date
    );
END;
$$ LANGUAGE plpgsql;

SET GLOBAL event_scheduler = ON;

CREATE EVENT IF NOT EXISTS daily_Expired_Reservation
ON SCHEDULE EVERY 1 DAY
STARTS CURRENT_TIMESTAMP + INTERVAL 1 HOUR
DO
  CALL Cancel_Expired_Reservation ();
```

```
package sample;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.image.Image;
import javafx.stage.Stage;
public class Main extends Application {@Override
public void start(Stage primaryStage) throws Exception{
Parent root =
FXMLLoader.load(getClass().getResource("sample.fxml"));
primaryStage.setTitle("Головне");
primaryStage.getIcons().add(new Image("img/book.png"));
primaryStage.setScene(new Scene(root));
primaryStage.show();
}
public static void main(String[] args) {
launch(args);
}
}
```

```
package sample;

import java.net.URL;
import java.time.LocalDate;
import java.time.LocalTime;
import java.time.format.DateTimeParseException;
```

```
import java.util.List;
import java.util.Optional;
import java.util.ResourceBundle;

import dao.ReaderDAO;
import dao.BookDAO;
import dao.BookingDAO;
import dao.TransferDAO;
import javafx.collections.FXCollections;
import javafx.fxml.FXML;
import javafx.geometry.Insets;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.layout.GridPane;
import javafx.scene.paint.Color;
import tables.Reader;
import tables.Book;
import tables.Booking;
import tables.Transfer;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

public class Controller {

private List<Reader> Readers;
private List<Book> Books;
private List<Booking> Bookings;
private List<Transfer> Transfers;
private ReaderDAO ReaderDAO;
private BookDAO BookDAO;
private BookingDAO BookingDAO;
private TransferDAO TransferDAO;
private EntityManagerFactory entityManagerFactory;
private EntityManager em;
private AlertWindow alertWindow = new AlertWindow();

@FXML
void initialize() {

setCellValueFactories();
entityManagerFactory =
Persistence.createEntityManagerFactory("MyPU");
em = entityManagerFactory.createEntityManager();

ReaderDAO = new ReaderDAO(em);
BookDAO = new BookDAO(em);
BookingDAO = new BookingDAO(em);
TransferDAO = new TransferDAO(em);
```

```
addBookRecordButton.setOnAction(event -> addBook());
addTransferRecordButton.setOnAction(event -> addTransfer());
addBookingRecordButton.setOnAction(event -> addBooking());
addReaderRecordButton.setOnAction(event -> addReader());
showAllBooksButton.setOnAction(event -> showAllBooks());
editSelectedBookRecordButton.setOnAction(event -> editBook());
deleteSelectedBookRecordButton.setOnAction(event ->deleteBook());
showAllReadersButton.setOnAction(event -> showAllReaders());
editReaderSelectedRecordButton.setOnAction(event -> editReader());
deleteReaderSelectedRecordButton.setOnAction(event -
>deleteReader());
showAllBookingButton.setOnAction(event -> showAllBookings());
editSelectedBookingRecordButton.setOnAction(event ->editBooking());
deleteSelectedBookingRecordButton.setOnAction(event ->
deleteBooking());
showAllTransfersButton.setOnAction(event -> showAllTransfers());
editSelectedTransfersRecordButton.setOnAction(event ->
editTransfer());
deleteSelectedTransfersRecordButton.setOnAction(event -
>deleteTransfer());
}

private void deleteTransfer() {
Transfer Transfer =
TransferTable.getSelectionModel().getSelectedItem();if (Transfer !=
null &&
alertWindow.confirmationAlert("Записати",Transfer.toString())) {
TransferDAO.delete(Transfer);

TransferTable.setItems(FXCollections.observableList(TransferDAO.fin
dAll()));
}
}

private void editTransfer() {
Transfer Transfer =
TransferTable.getSelectionModel().getSelectedItem();if (Transfer ==
null)
return;

Dialog<R> dialog = new Dialog<Object>();
var window = addReaderRecordButton.getScene().getWindow();
dialog.initOwner(window);
dialog.setTitle("Інформація по книзі");

ButtonType okButtonType = new ButtonType("Редагувати",
ButtonType.OK.getButtonData());
```

```
ButtonType cancelButtonType = new ButtonType("Відмінити",
ButtonType.CANCEL.getButtonData());
dialog.getDialogPane().getButtonTypes().addAll(okButtonType,
cancelButtonType);
GridPane grid = new GridPane();grid.setHgap(10); grid.setVgap(10);

grid.setPadding(new Insets(20, 20, 10, 10));

TextField BookDate = new
TextField(Transfer.getBookDate().toString());
BookDate.setPromptText("Дата");
TextField seat = new TextField(Transfer.getSeat());
seat.setPromptText("Читацький квиток");
TextField TransferStatus = new
TextField(Transfer.getTransferStatus());
TransferStatus.setPromptText("Статус");

grid.add(new Label("Дата"), 0, 0);
grid.add(BookDate, 1, 0);
grid.add(new Label("Читацький квиток"), 0, 5);
grid.add(TransferStatus, 1, 5);
grid.add(new Label("Статус"), 0, 7);
grid.add(seat, 1, 7);
dialog.getDialogPane().setContent(grid);
dialog.setResultConverter(dialogButton -> {
if (dialogButton == okButtonType) {
try {
int indexSelected = Transfers.indexOf(Transfer);
Transfer.setBookDate(LocalDate.parse(BookDate.getText().trim()));
Transfer.setBookTime(LocalTime.parse(BookTime.getText().trim()));
Transfer.setBookAddress(BookAddress.getText().trim());
Transfer.setBookCity(BookCity.getText().trim());
Transfer.setBookPlace(BookPlace.getText().trim());
Transfer.setTransferStatus(TransferStatus.getText().trim());

Transfer.setPrice(Double.parseDouble(price.getText().trim()));
Transfer.setSeat(seat.getText().trim());

Transfers.set(indexSelected, Transfer);
TransferDAO.editRecord(Transfer);

TransferTable.setItems(FXCollections.observableArrayList(Transfers)
);
TransferTable.refresh();

} catch (NullPointerException e) {
alertWindow.errorAlert("Помилка! Введені невірні дані!");
```

```
} catch (NumberFormatException e) {
alertWindow.errorAlert("Помилка! Введений текст замість числа!");


} catch (DateTimeParseException e) {
alertWindow.errorAlert("Помилка! Введена некоректна дата!");
}
}
alertWindow.errorAlert("Помилка! Одне або декілька полів
порожні!");}

return null;
});
Optional<Transfer> result = dialog.showAndWait();
}

private void showAllTransfers() {
try {
Transfers = TransferDAO.findAll();
if (Transfers.isEmpty()) {
alertWindow.errorAlert("Записи не знайдені!");return;
}
TransferTable.getItems().clear();
TransferTable.setItems(FXCollections.observableList(Transfers));
}
catch (Exception ex) {
alertWindow.errorAlert(ex.getMessage());
}
}

private void deleteBooking() {Booking Booking =
BookingTable.getSelectionModel().getSelectedItem();if (Booking !=
null &&
alertWindow.confirmationAlert("Записати",Booking.briefInfo())) {
BookingDAO.delete(Booking);

BookingTable.setItems(FXCollections.observableList(BookingDAO.findA
ll()));
}
}

private void editBooking() {Booking Booking =
BookingTable.getSelectionModel().getSelectedItem();
if (Booking == null)
return;

Dialog<Booking> dialog = new Dialog<>();
var window = addReaderRecordButton.getScene().getWindow();
dialog.initOwner(window);
```

```
dialog.setTitle("Бронювання книги");

ButtonType okButtonType = new ButtonType("Редагувати",
ButtonType.OK.getButtonData());
ButtonType cancelButtonType = new ButtonType("Вдмінити",
ButtonType.CANCEL.getButtonData());
dialog.getDialogPane().getButtonTypes().addAll(okButtonType,
cancelButtonType);
GridPane grid = new GridPane();grid.setHgap(10); grid.setVgap(10);

grid.setPadding(new Insets(20, 20, 10, 10));

TextField secondName = new TextField(Booking.getSecondName());
secondName.setPromptText("Прізвище");
TextField firstName = new TextField(Booking.getFirstName());
firstName.setPromptText("Им`я");
TextField middleName = new TextField(Booking.getMiddleName());
middleName.setPromptText("По батькові");
TextField phone = new TextField(Booking.getPhone());
phone.setPromptText("Контактний телефон");
TextField hiringDate = new
TextField(Booking.getHiringDate().toString());
hiringDate.setPromptText("Дата");

grid.add(new Label("Прізвище"), 0, 0);
grid.add(secondName, 1, 0);
grid.add(new Label("Им`я"), 0, 1);
grid.add(firstName, 1, 1);
grid.add(new Label("По батькові"), 0, 2);
grid.add(middleName, 1, 3);
grid.add(new Label("Контактний телефон"), 0, 6);
grid.add(phone, 1, 6);
grid.add(new Label("Дата"), 0, 7);
grid.add(occupation, 1, 7);

dialog.getDialogPane().setContent(grid);
dialog.setResultConverter(dialogButton -> {
if (dialogButton == okButtonType) {
try {
int indexSelected = Bookings.indexOf(Booking);
Booking.setSecondName(secondName.getText().trim());
Booking.setFirstName(firstName.getText().trim());

Booking.setHiringDate(LocalDate.parse(hiringDate.getText().trim()))
; Booking.setMiddleName(middleName.getText().trim());
Booking.setCity(city.getText().trim());
Booking.setOccupation(occupation.getText().trim());
Booking.setPhone(phone.getText().trim());
Booking.setOccupation(occupation.getText().trim());
```

```
Bookings.set(indexSelected, Booking);
BookingDAO.editRecord(Booking);

BookingTable.setItems(FXCollections.observableArrayList(Bookings));
BookingTable.refresh();
} catch (NullPointerException e) {
BookingTable.getItems().clear();
BookingTable.setItems(FXCollections.observableList(Bookings));
}
catch (Exception ex) {
alertWindow.errorAlert(ex.getMessage());
}
}

private void deleteReader() {
Reader Reader = ReaderTable.getSelectionModel().getSelectedItem();
if (Reader != null &&
alertWindow.confirmationAlert("Читацький квиток",
Reader.briefInfo())) {
ReaderDAO.delete(Reader);

ReaderTable.setItems(FXCollections.observableList(ReaderDAO.findAll
()));
}
}

private void editReader() {
Reader Reader = ReaderTable.getSelectionModel().getSelectedItem();
if (Reader == null)
return;

Dialog<Reader> dialog = new Dialog<>();
var window = addReaderRecordButton.getScene().getWindow();
dialog.initOwner(window);
dialog.setTitle("Видача книги");

ButtonType okButtonType = new ButtonType("Редагувати",
ButtonType.OK.getButtonData());
ButtonType cancelButtonType = new ButtonType("Відмінити",
ButtonType.CANCEL.getButtonData());
dialog.getDialogPane().getButtonTypes().addAll(okButtonType,
cancelButtonType);
GridPane grid = new GridPane();grid.setHgap(10); grid.setVgap(10);
grid.setPadding(new Insets(20, 20, 10, 10));

dialog.getDialogPane().setContent(grid);
dialog.setResultConverter(dialogButton -> {
if (dialogButton == okButtonType) {
```

```
try {
int indexSelected = Readers.indexOf(Reader);
Reader.setSecondName(secondName.getText().trim());
Reader.setFirstName(firstName.getText().trim());

Reader.setBirthday(LocalDate.parse(birthday.getText().trim()));
Reader.setMiddleName(middleName.getText().trim());
Reader.setCity(city.getText().trim());

Reader.setTransferNum(Integer.parseInt(TransferNum.getText().trim()
));
Reader.setBookTitle(BookTitle.getText().trim());

Readers.set(indexSelected, Reader);
ReaderDAO.editRecord(Reader);

ReaderTable.setItems(FXCollections.observableArrayList(Readers));
ReaderTable.refresh();
} catch (NullPointerException e) {
ReaderTable.getItems().clear();
ReaderTable.setItems(FXCollections.observableList(Readers));
}
catch (Exception ex) {
alertWindow.errorAlert(ex.getMessage());
}
}

private void deleteBook() {
Book Book = BookTable.getSelectionModel().getSelectedItem();if
(Book != null &&
alertWindow.confirmationAlert("Записати",Book.briefInfo())) {
BookDAO.delete(Book);

BookTable.setItems(FXCollections.observableList(BookDAO.findAll()))
;
}
}

private void editBook() {
Book Book = BookTable.getSelectionModel().getSelectedItem();if
(Book == null)
return;

Dialog<Book> dialog = new Dialog<>();
var window = addReaderRecordButton.getScene().getWindow();
dialog.initOwner(window);
dialog.setTitle("Читацький квиток");
```

```
ButtonType okButtonType = new ButtonType("Редагувати",
ButtonType.OK.getButtonData());
ButtonType cancelButtonType = new ButtonType("Відмінити",
ButtonType.CANCEL.getButtonData());
dialog.getDialogPane().getButtonTypes().addAll(okButtonType,
cancelButtonType);
GridPane grid = new GridPane();grid.setHgap(10);
grid.setVgap(10);
grid.setPadding(new Insets(20, 20, 10, 10));

TextField title = new TextField(Book.getTitle());
title.setPromptText("Назва книги");
TextField performer = new TextField(Book.getPerfomer());
performer.setPromptText("Автор");
TextField genre = new TextField(Book.getGenre());
genre.setPromptText("Вид");
TextField ageRestriction = new TextField(Book.getAgeRestriction());
BookCity.setPromptText("Місто");
TextField BookTime = new TextField(Book.getBookTime().toString());
BookTime.setPromptText("Рік");

grid.add(new Label("Назва книги"), 0, 0);
grid.add(title, 1, 0);
grid.add(new Label("Автор"), 0, 1);
grid.add(performer, 1, 1);
grid.add(new Label("Вид"), 0, 2);
grid.add(genre, 1, 2);
grid.add(new Label("Місто"), 0, 4);
grid.add(BookCity, 1, 4);
grid.add(new Label("Рік"), 0, 5);
grid.add(BookTime, 1, 5);

dialog.getDialogPane().setContent(grid);
dialog.setResultConverter(dialogButton -> {
if (dialogButton == okButtonType) {
try {
int indexSelected = Books.indexOf(Book);
Book.setTitle(title.getText().trim());
Book.setPerfomer(performer.getText().trim());

Book.setAgeRestriction(ageRestriction.getText().trim());
Book.setGenre(genre.getText().trim());
Book.setBookCity(BookCity.getText().trim());

Book.setBookTime(LocalTime.parse(BookTime.getText().trim()));
Book.setTransferPrice(Double.parseDouble(TransferPrice.getText().tr
im()));
Books.set(indexSelected, Book);
BookDAO.editRecord(Book);
```

```
BookTable.setItems(FXCollections.observableArrayList(Books));
 BookTable.refresh();
} catch (NullPointerException e) {
BookTable.getItems().clear();
BookTable.setItems(FXCollections.observableList(Books));
}
catch (Exception ex) {
alertWindow.errorAlert(ex.getMessage());
}
}

private void addReader() {
if(secondNameReaderTextField.getText().isEmpty() ||
firstNameReaderTextField.getText().isEmpty() ||
middleNameReaderTextField.getText().isEmpty()
|| ddReaderTextField.getText().isEmpty() ||
mmReaderTextField.getText().isEmpty() ||
yyyyReaderTextField.getText().isEmpty()
|| cityReaderTextField.getText().isEmpty() ||
TransferNumReaderTextField.getText().isEmpty() ||
BookTitleReaderTextField.getText().isEmpty()) {
try { addReaderResultLabel.setText("");Reader Reader = new
Reader();

Reader.setFirstName(firstNameReaderTextField.getText().trim());
Reader.setSecondName(secondNameReaderTextField.getText().trim());
Reader.setMiddleName(middleNameReaderTextField.getText().trim());
Reader.setCity(cityReaderTextField.getText().trim());

Reader.setTransferNum(Integer.parseInt(TransferNumReaderTextField.g
etText().trim(
)));
Reader.setBookTitle(BookTitleReaderTextField.getText().trim());
Reader.setBirthday(LocalDate.parse(yyyyReaderTextField.getText().tr
im() + "-"
+ mmReaderTextField.getText().trim() + "-" +
ddReaderTextField.getText().trim()));

ReaderDAO.add(Reader);
addReaderResultLabel.setTextFill(Color.web("#00FF00"));
addReaderResultLabel.setText("Запис успішно додано!");

if (!ReaderTable.getItems().isEmpty())

ReaderTable.setItems(FXCollections.observableList(ReaderDAO.findAll
()));

firstNameReaderTextField.clear();
```

```
secondNameReaderTextField.clear();
middleNameReaderTextField.clear();
cityReaderTextField.clear();
TransferNumReaderTextField.clea();
BookTitleReaderTextField.clear();
yyyyReaderTextField.clear(); mmReaderTextField.clear();
ddReaderTextField.clear();
} catch (NullPointerException e) {
}
}

private void addBooking() {
if(secondNameBookingTextField.getText().isEmpty() ||
firstNameBookingTextField.getText().isEmpty() ||
middleNameBookingTextField.getText().isEmpty()
|| ddBirthdayBookingTextField.getText().isEmpty() ||
mmBirthdayBookingTextField.getText().isEmpty()
|| yyyyBirthdayBookingTextField.getText().isEmpty() ||
phoneBookingTextField.getText().isEmpty()
|| ddHiringBookingTextField.getText().isEmpty() ||
mmHiringBookingTextField.getText().isEmpty()
|| yyyyHiringBookingTextField.getText().isEmpty() ||
occupationBookingTextField.getText().isEmpty()
|| cityBookingTextField.getText().isEmpty()) {
try { addBookingResultLabel.setText("");Booking Booking = new
Booking();


Booking.setFirstName(secondNameBookingTextField.getText().trim());
Booking.setSecondName(firstNameBookingTextField.getText().trim());
Booking.setMiddleName(middleNameBookingTextField.getText().trim());
Booking.setPhone(phoneBookingTextField.getText().trim());
Booking.setCity(cityBookingTextField.getText().trim());

Booking.setOccupation(occupationBookingTextField.getText().trim());

Booking.setBirthday(LocalDate.parse(yyyyBirthdayBookingTextField.ge
tText().trim() + "-" + mmBirthdayBookingTextField.getText().trim()
+ "-" +
ddBirthdayBookingTextField.getText().trim()));

Booking.setHiringDate(LocalDate.parse(yyyyHiringBookingTextField.ge
tText().trim() + "-" + mmHiringBookingTextField.getText().trim() +
"-" +
ddHiringBookingTextField.getText().trim()));

BookingDAO.add(Booking);
addBookingResultLabel.setTextFill(Color.web("#00FF00"));
addBookingResultLabel.setText("Запис успішно додано!");
```

```
if (!ReaderTable.getItems().isEmpty())

ReaderTable.setItems(FXCollections.observableList(ReaderDAO.findAll
()));secondNameBookingTextField.clear();
firstNameBookingTextField.clear();
middleNameBookingTextField.clear();
phoneBookingTextField.clear();
cityBookingTextField.clear();
occupationBookingTextField.clear();
yyyyBirthdayBookingTextField.clear();
mmBirthdayBookingTextField.clear();
ddBirthdayBookingTextField.clear();
yyyyHiringBookingTextField.clear();
mmHiringBookingTextField.clear();
ddHiringBookingTextField.clear();
} catch (NullPointerException e) {
}
}

private void addTransfer() {
if(ddTransferTextField.getText().isEmpty() ||
mmTransferTextField.getText().isEmpty() ||
yyyyTransferTextField.getText().isEmpty()
|| hoursTransferTextField.getText().isEmpty() ||
minutesTransferTextField.getText().isEmpty()
|| cityTransferTextField.getText().isEmpty() ||
addressTransferTextField.getText().isEmpty()
|| placeTransferTextField.getText().isEmpty() ||
seatTransferTextField.getText().isEmpty()
|| priceTransferTextField.getText().isEmpty() ||
TransferStatusTextField.getText().isEmpty()) {
try { addTransferResultLabel.setText("");
Transfer Transfer = new Transfer();
Transfer.setTransferStatus(TransferStatusTextField.getText().trim()
);
Transfer.setPrice(Double.parseDouble(priceTransferTextField.getText
().trim()));
Transfer.setSeat(seatTransferTextField.getText().trim());
Transfer.setBookPlace(placeTransferTextField.getText().trim());

Transfer.setBookAddress(addressTransferTextField.getText().trim());
Transfer.setBookCity(cityTransferTextField.getText().trim());

Transfer.setBookDate(LocalDate.parse(yyyyTransferTextField.getText(
).trim() +
"-" + mmTransferTextField.getText().trim() + "-" +
ddTransferTextField.getText().trim()));
```

```
Transfer.setBookTime(LocalTime.parse(hoursTransferTextField.getText
().trim() +
":" + minutesTransferTextField.getText().trim()));

TransferDAO.add(Transfer);
addTransferResultLabel.setTextFill(Color.web("#00FF00"));
addTransferResultLabel.setText("Запис про читацький квиток успішно
додано!");

if (!TransferTable.getItems().isEmpty())

TransferTable.setItems(FXCollections.observableList(TransferDAO.fin
dAll()));

TransferStatusTextField.clear();
priceTransferTextField.clear();
seatTransferTextField.clear();
placeTransferTextField.clear();
addressTransferTextField.clear();
cityTransferTextField.clear();
yyyyTransferTextField.clear();
mmTransferTextField.clear();
ddTransferTextField.clear();
hoursTransferTextField.clear();
minutesTransferTextField.clear();
} catch (NullPointerException e) {
}
}
```

Додаток В Опис програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

1 Загальні відомості

Розроблене програмне забезпечення призначене для автоматизації роботи шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією та представляє собою програму для бібліотекаря, який займається веденням обліку книг та читачів бібліотеки та формуванням і зберіганням звітної інформації.

2 Функціональне призначення

Функціональним призначенням програмного забезпечення є автоматизація роботи працівників шкільної бібліотеки, забезпечення оперативного отримання інформації про наявність книг; формування читацького квитку; оформлення бронювання та видачі книг.

3 Опис логічної структури

Програма розроблена на модульній основі, тобто кожний компонент (модуль) програми відповідає за конкретну дію.

Програмні дані зберігаються у нормалізованих таблицях бази даних.

4 Технічні та програмні засоби

Система повинна задовольняти вказаним вимогам на комп’ютері наступної мінімальної комплекції:

– CPU: Intel(R) i5 2,16 МГц або аналогічний за параметрами AMD;

– RAM: 4 GB;

– HDD: 10 GB вільного місця.

Для повноцінного функціонування системи необхідно використовувати ОС Windows версії не нижче 7.

Необхідна наявність встановленого Необхідна наявність встановленого JDK 17.

5 Виклик та завантаження

Для того, щоб почати працювати з програмою, потрібно запустити виконуваний файл school_library.jar, який знаходиться на робочому столі або в меню «Пуск». Після чого з'явиться вікно авторизації. Після проходження авторизації з’явиться головне вікно програми.

6 Вхідні та вихідні дані

Дані зберігаються в файлі БД з довільним доступом типу .myd.

Введення даних здійснюється за допомогою пристроїв введення даних (клавіатури та миші). Дані або вводяться вручну, або обираються із представлених списків.

Виведення даних здійснюється за допомогою пристроїв виведення даних (монітор та принтер).

Додаток Г Інструкція користувача програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

Програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією можна запустити з виконуваного файлу school_library.jar, який знаходиться на робочому столі або в меню «Пуск»

Після цього з’являється вікно авторизації користувача, яке показано на рисунку Г.1. Якщо логін або пароль було ведено не вірно, з’являється попередження. Після вдалої авторизації з’являється головне вікно програми, яке представлено на рисунку Г.2.

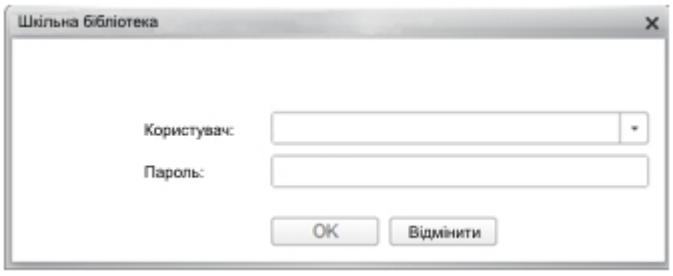


Рисунок Г.1 – Вікно авторизації користувача

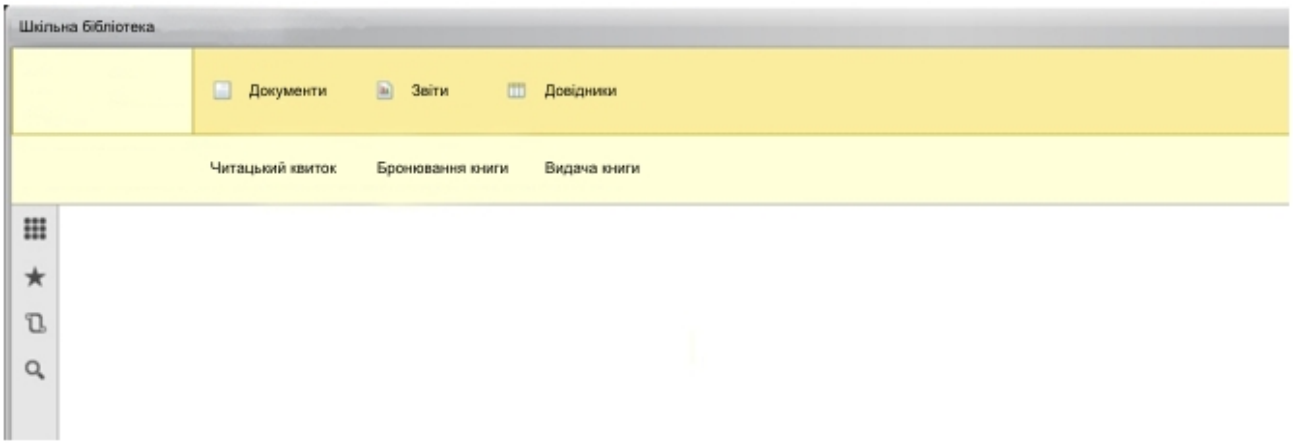


Рисунок Г.2 – Головне вікно програми

На головне вікно програми винесено основні довідники та докумети, з якими йде робота в програмі: довідник "Книга", довідник "Примірник книги", довідник "Читацький квиток", документ "Бронювання книги", документ "Видача книги".

Крім того, також є кнопка "Звіти", при виборі якої є можливість сформувати відповідні звіти: "Видані книги", "Інвентаризація", "Планове повернення", "Заборгованість".

Для заповнення довідника "Читацький квиток", необхідно виконати наступні дії:

1. Перейти на головне вікно програми, обрати довідник "Читацький квиток".

2. Для створення нового запису потрібно натиснути кнопку «Створити».

З’явилась нова форма для створення нового запису, у якій потрібно заповнити усі необхідні реквізити та натиснути кнопку «Записати та закрити» (див. рис. Г.3).

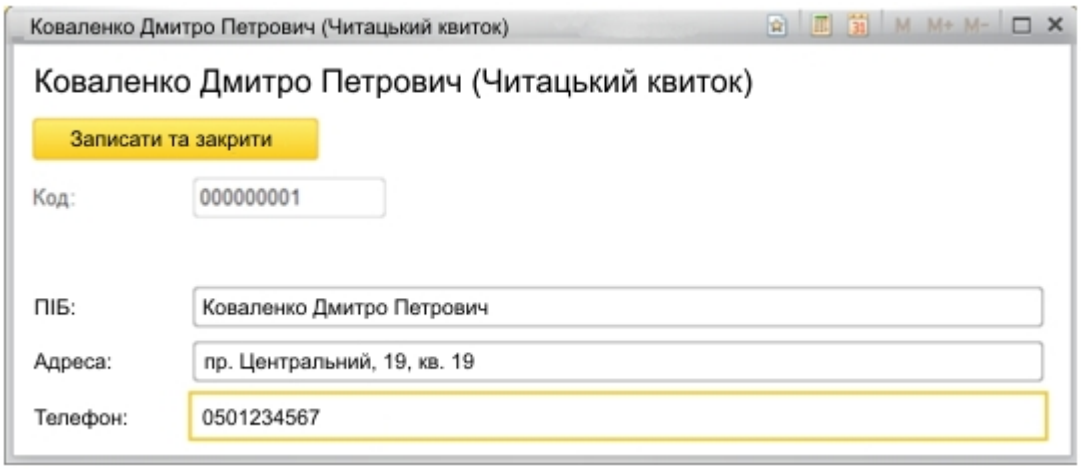


Рисунок Г.3 – Форма для заповнення реквізитів «Читацький квиток»

Інші довідники та документи заповнюються аналогічно.

Для формування звіту "Видані книги" необхідно виконати наступні дії:

1. Перейти на головне вікно програми, обрати кнопку "Звіти".
2. Обрати потрібний звіт, у даному випадку "Видані книги".
3. Обрати потрібний період.
4. Натиснути кнопку "Сформувати" (див. рис. Г.4).

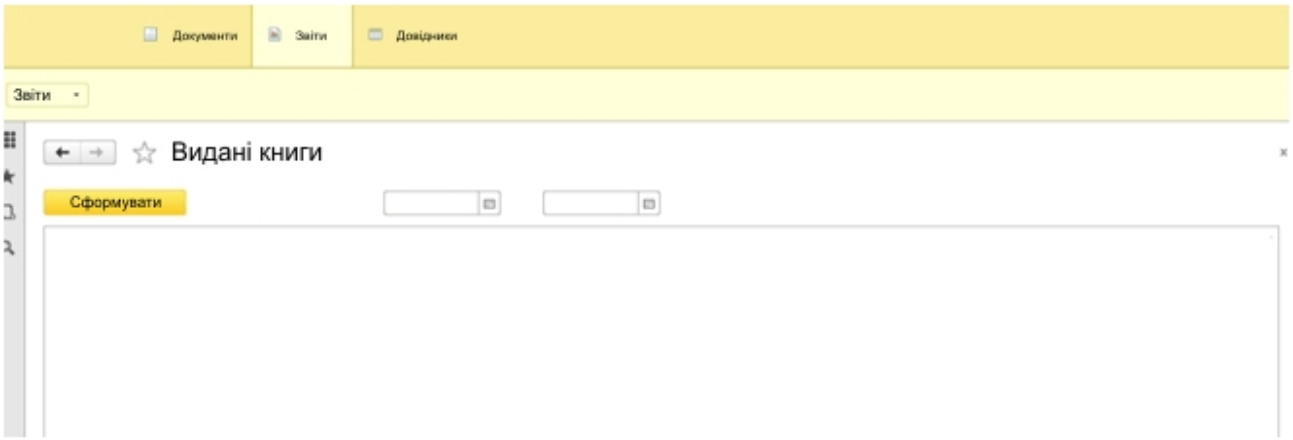


Рисунок Г.4 – Звіт "Видані книги"

Інші звіти формуються аналогічно.

Додаток Д Програма та методика випробування програмного забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією

1 Об’єкт випробування

1.1 Назва об’єкту

Повна назва об’єкта випробувань: «Програмне забезпечення шкільної бібліотеки Кир'яківського ліцею з початковою школою та гімназією».

Коротка назва: програма.

1.2 Область застосування

Програма призначена для забезпечення оперативного отримання інформації про наявність книг; формування читацького квитку; оформлення бронювання та видачі книг шкільної бібліотеки.

Программа забезпечує виконання таких функцій:

- авторизація;
- занесення даних про книгу;
- редагування даних про книгу;
- перегляд інформації про книгу;
- видалення інформації про книгу;
- оформлення читацького квитка;
- редагування читацького квитка;
- перегляд інформації про читацький квиток;
- видалення читацького квитка;
- оформлення бронювання книги;
- оформлення видачі книги;
- пошук необхідної інформації;
- формування необхідних звітів.

Мета випробування

Мета проведення випробування: оцінка експлуатаційних характеристик програми, перевірка і підтвердження працездатності програми в умовах, максимально наближених до умов реальної експлуатації.

2 Вимоги до програмного забезпечення

Вимоги до програми, що підлягали перевірці, представлені у технічному завданні, яке наведено у Додатку А.

3 Вимоги до програмної документації

Для проведення випробування програми повинна бути надана наступна документація:

- технічне завдання;
- текст програми;
- програма і методика випробувань ПЗ;
- керівництво користувача.
- опис програми

4 Засоби та порядок випробувань

Випробування програми проводиться на цільовому обладнанні в тій конфігурації, яка заявлена в Технічному завданні.

Під час випробування проводиться повне функціональне тестування, а також навантажувальне тестування і тестування на відмову всього програмно-апаратного комплексу. Функції тестуються у наступному порядку: авторизація, оформлення читацького квитка, оформлення бронювання книги, оформлення видачі книги, формування звіту (Видані книги).

Всі виявлені недоліки програми повинні бути зафіксовані в протоколах і усуваються розробником до моменту впровадження програми в дію.

5 Методи випробування

Випробування проводиться за стратегією «чорного ящика». За рахунок значної кількості функцій, які потрібно випробувати, представлено методи випробування декількох функцій програми.

Функція «Оформлення читацького квитка»

На головному вікні програми обираємо довідник «Читацький квиток» та натискаємо кнопку «Додати». Повинно з’явитися відповідне вікно, у якому обираємо читача, вносимо усі необхідні реквізити та натискаємо кнопку «Закрити та зберегти». У результаті повинен сформуватися новий читацький квиток.

Функція «Оформлення бронювання книги»

На головному вікні програми обираємо документ «Бронювання книги» та натискаємо кнопку «Додати». Повинно з’явитися відповідне вікно, у якому обираємо читача, книгу та натискаємо кнопку «Закрити та зберегти». У результаті повинно сформуватися нове бронювання книги.

Функція «Оформлення видачі книги»

На головному вікні програми обираємо документ «Видача книги» та натискаємо кнопку «Додати». Повинно з’явитися відповідне вікно, у якому обираємо читача, книгу та натискаємо кнопку «Закрити та зберегти». У результаті повинна сформуватися нова видача книги.

Функція «Формування звіту (Видані книги)»

В головному меню натискаємо на кнопку «Звіти». Повинно з’явитися вікно з можливими звітами. Обираємо звіт «Видані книги» та виконуємо встановлені початкові дії (вибираємо відповідні параметри звіту). В результаті повинен сформуватися новий звіт «Видані книги».