

**Відокремлений структурний підрозділ
«Фаховий коледж інженерії та управління
Національного авіаційного університету»**

КУРСОВИЙ ПРОЕКТ

Київ 2021

Відокремлений структурний підрозділ
«Фаховий коледж інженерії та управління
Національного авіаційного університету»

КУРСОВИЙ ПРОЕКТ
Пояснювальна записка

Тема: «Інформаційно довідкова система обліку бібліотечного фонду»

Студента Бичек Даниїл Максимович
групи 407-ІІЗ

Напрямок підготовки: **121 Інженерія програмного забезпечення**

Студент

(підпис)

Д. Б. Бичек

Керівник курсового проекту

(підпис)

П. Ю. Родіонов

Зміст

Вступ.....	4
1 Проектування інформаційної системи.....	5
1.1 Обґрунтування необхідності проектування інформаційної системи	5
1.2 Аналіз предметної області	5
1.3 Розробка функціональної моделі інформаційної системи.....	7
1.4 Моделювання потоків даних	14
2 Реалізація інформаційної системи	15
2.1 Об’єктно-орієнтований аналіз інформаційної системи	15
2.2 Розроблення технічних вимог до інформаційної системи	20
2.3 Програмування інформаційної системи	21
2.4 Тестування та налагодження інформаційної системи	25
2.5 Економічна оцінка проекту розроблення інформаційної системи ..	25
Висновки	32
Список літератури.....	33
Додатки	34

Вступ

На сьогоднішня кількість літератури в сучасному світі має воістину неосяжні розміри. В реальності ознайомлення з літературними шедеврами різних жанрів є майже традицією. Починаючи зі школи кожна дитина повинна читати велику кількість різноманітної літератури для розвитку свого кругозору. Таким чином можна точно стверджувати що велика кількість людей тримає постійний «зв'язок» з різноманітними книжками, починаючи від авторів фантастів і закінчуючи научною літературою. Саме приріст літератури і робить тему даного курсового проекту актуальною та корисною для бібліотек, як приватних так і державних.

Метою курсового проектування є розробка та реалізація проекту інформаційної системи управління книжками. Це ставить необхідним розв'язок наступних задач:

- 1 Провести аналіз предметної області.
- 2 Сформулювати технічне завдання.
- 3 Побудувати функціональну модель.
- 4 Створити об'єктно-орієнтовану модель роботи інформаційної системи, використовуючи мову UML.
- 5 Реалізувати спроектовану інформаційну систему на мові програмування високого рівня у відповідному програмному середовищі.
- 6 Провести тестування та налагоджування системи.
- 7 Розробити трудомісткість розроблювання інформаційної системи.

Для розв'язання поставлених задач використовувалося відповідне програмне забезпечення, зокрема Visual Studio з пакетом для роботи на мові програмування C#, All Fusion Process Modeler для візуалізації діаграм логіки роботи системи, та Rational Rose[4] для проектування класів системи. Також було задіяно MAMP для навігації по створеній базі даних системи через PhpMyAdmin.

1 Проектування інформаційної системи

1.1 Обґрунтування необхідності проектування інформаційної системи

Інформаційною системою яка буде розроблятися є «Інформаційно-довідкова система обліку бібліотечного фонду». На сьогоднішня бібліотека в будь-якому місті є історичним надбанням та зберігає за собою безліч безцінної інформації, яку інколи відвідувач бібліотеки навіть не зможе знайти через непрактичну систему впорядкування книг на полицях за алфавітом. Тому саме актуальною проблемою для вирішення є саме оптимізація пошуку необхідної інформації в бібліотеці і позбавлення від будь якої паперової документації що до моніторингу книжок в бібліотеці та їх позичення відвідувачами.

Найпростішим способом для структурування даних в бібліотеках використовують таблиці Microsoft Excel або Microsoft Access, тому було вирішено розробити спеціалізоване програмне забезпечення виключно для обліку книг в бібліотеках.

Призначенням системи можна вважати оптимізацію процесів моніторингу книжок та їх пошуку. Автоматизація контролю позичення та повернення книжок читачами з бібліотеки за принципом фіксації дат за яких книжки відсутні в бібліотеці.

Основні функції:

- збереження всіх даних бібліотеки в базі даних;
- швидкий пошук книжок за різними характеристиками;
- контроль і фіксація заборгованостей окремих читачів
- швидка та легка реєстрація і видалення книжок

Користувачами системи будуть виступати читачі (клієнти) та бібліотекарі (адміністратори) що будуть мати окремі можливості.

За сценарієм роботи програма буде мати два можливих шляхи, в залежності від типу користувача і його мети. Таким чином читач зможе здійснювати пошук книжок та переглядати їх характеристики. Читач буде мати можливість позичити певну кількість книжок з бібліотеки, здійснювати пошук

по бібліотеці за певними. Бібліотекар чи адміністратор буде також мати змогу переглядати книжки та здійснювати пошук по бібліотеці але на відміну від користувача також зможе додавати або видаляти книжки з бібліотеки, переглядати користувачів бібліотеки і їх заборгованість по книжкам та здійснювати реєстрацію нових користувачів програми.

В якості сховища інформації буде використовуватись база даних із спеціалізованими таблицями під вимоги програми. Орієнтовно буде 3 великі таблиці в яких буде зберігатись інформація про книжки, заборгованості та користувачів. Також таблиці будуть пов'язані між собою ключовими полями.

Вихідними даними програми буде вибірка певної інформації з бази даних та змінення цих даних завдяки чому програма буде виступати у якості довідника.

1.2 Аналіз предметної області

В процесі аналізу предметної було проаналізовано 2 інформаційні системи, що мають схожі характеристики з розроблюваною.

Першим етапом аналізу предметної області було визначено архітектуру та мову на котрій створені дані інформаційні системи. Також на цьому етапі було виявлено основні функції даної інформаційної системи. Результати даного етапу аналізу предметної області є на таблиці 1.

Таблиця 1 – Аналіз інформаційних систем

Назва ІС	Розробник	Мова програмування	Джерело інформації
«Calibre»	«Kovid Goval»	Phyton, C(qt), CoffeeScript, Javascript – desktop application	https://calibre-ebook.com/
«MyRuLib»	Денис Кандрашин (на правах ліцензії NTU GPL)	C++ – desktop application	http://myrulib.lintest.ru/

1.3 Розробка функціональної моделі інформаційної системи

Для розробки функціональної було обрано Allfusion Process Modeler (AfPM) - це стандартний і простий у використанні програмний продукт для моделювання процесів, який допомагає користувачам розробляти та документувати свої бізнес-процеси. За допомогою цього потужного інструменту можливо використовувати та аналізувати документи більш організовано. Це дозволяє користувачам точно та конкретно впорядковувати всю свою документацію. Це також забезпечує гнучке робоче середовище для бізнесу та потік інформації для унікальної ідентифікації організації.

На першому етапі створення функціональної моделі інформаційної системи було створено розроблюємо першу основну контекстну діаграму в якій були описані всі зовнішні зв'язки інформаційної системи (Рис. 1).

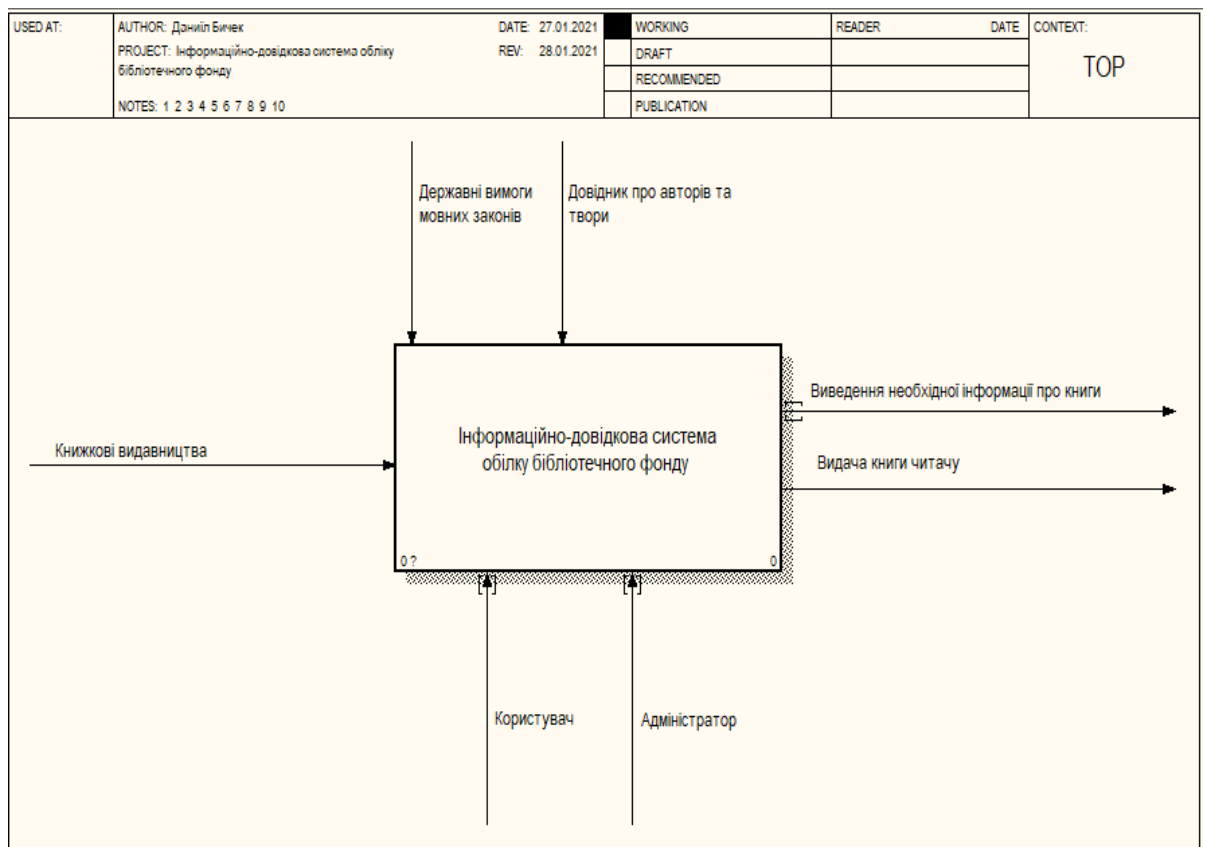


Рисунок 1 – контекстна діаграма IDEF0

Таблиця 2 – Стрілки контекстної діаграми IDEF0

Ім'я стрілки (Arrow Name)	Визначення стрілки (Arrow Definition)	Тип стрілки (Arrow Type)
Книжкові видавництва	Постачання нових книжок до бібліотеки	Input
Державні вимоги мовних законів	Закони, яких повинна дотримуватися система реєстрації книжок	Control
Довідник про авторів та твори	Джерело інформації про існуючих авторів світу та їх твори	Control
Виведення необхідної інформації про книги	Структуровані дані, які необхідні користувачу для вирішення необхідних потреб	Output
Видача книги читачу	Книга, яка необхідна читачу за його потреб	Output
Користувач	Читач бібліотеки	Mechanism
Адміністратор	Людина яка безпосередньо взаємодіє з даними які зберігаються в системі	Mechanism

Наступним етапом є декомпозиція контекстної діаграми A0 з метою створення діаграми A0 з конкретно розписаними модулями інформаційної системи (Рис. 2).

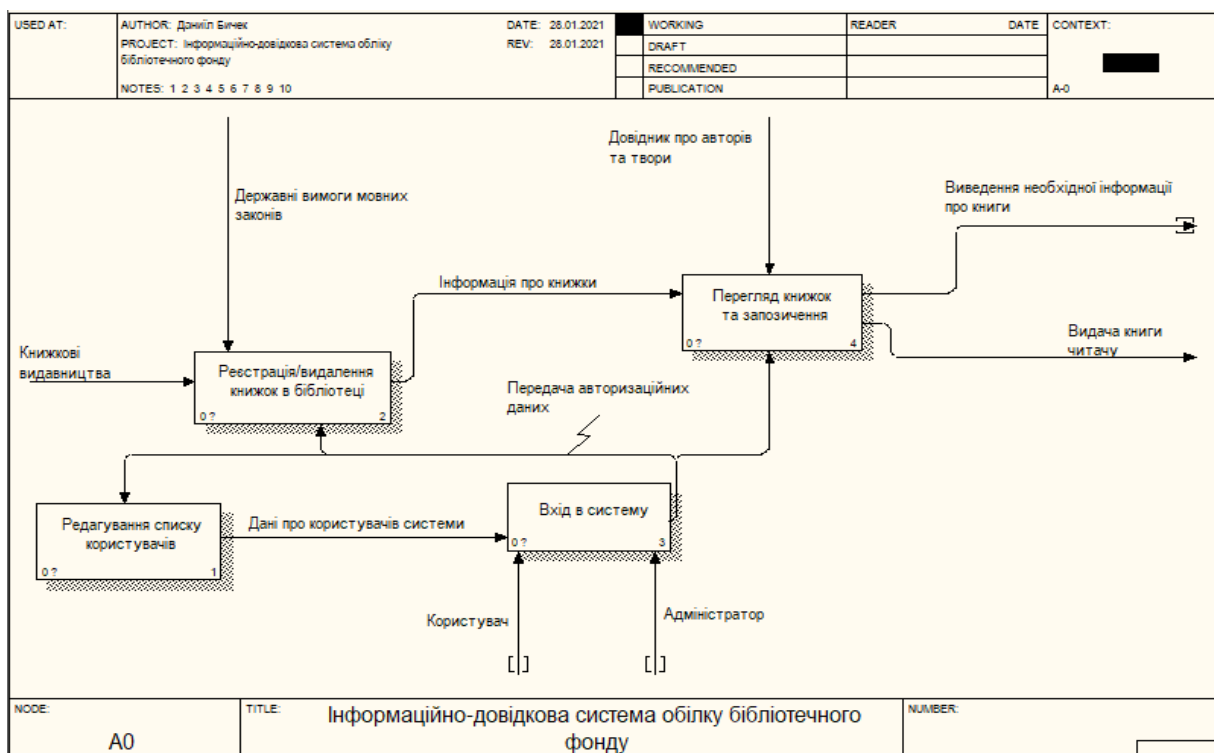


Рисунок 2 – декомпозиція контекстної діаграми

Таблиця 3 – Роботи діаграми декомпозиції A0

Ім'я роботи (Activity name)	Визначення (Definition)
Реєстрація/Видалення книжок в бібліотеці	Змінити інформацію про наявні книжки в бібліотеці
Перегляд книжок та запозичення	Переглянути наявні книжки в бібліотеці за вказаними параметрами; запозичити книжки з бібліотеки
Редагування списку користувачів	Змінити інформацію про користувачів бібліотеки
Вхід в систему	Авторизуватися в системі для подальшого користування програмою

Після визначення з модулями програми проводимо декомпозицію кожного з наявних модулів і отримуємо діаграми A1, A2, A3, A4 (Рис. 3-6).

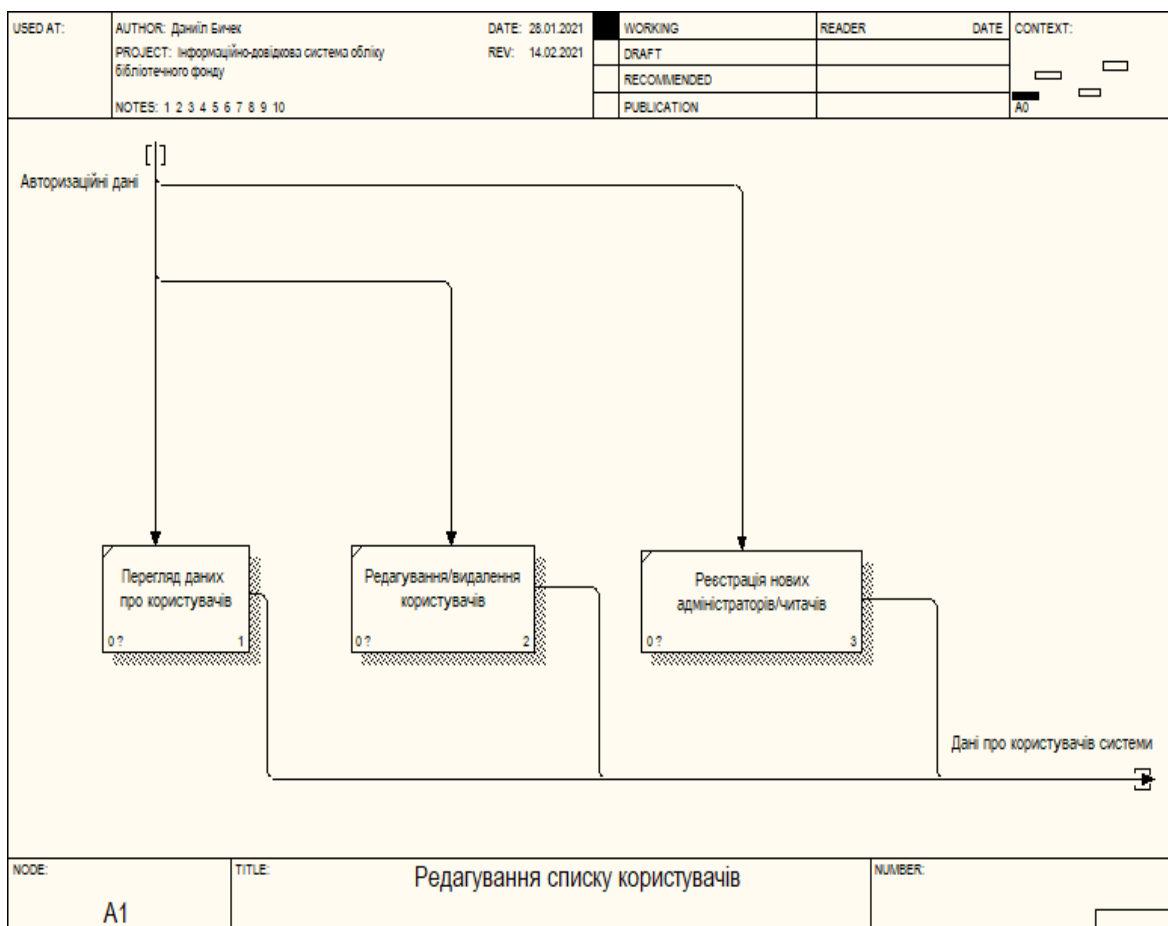


Рисунок 3 – декомпозиція блоку A1

Таблиця 4 – Роботи діаграми декомпозиції A1

Ім'я роботи (Activity name)	Визначення (Definition)
Перегляд даних про користувачів	Переглянути наявних читачів та адміністраторів бібліотеки
Редагування/Видалення користувачів	Видалити/Змінити наявних читачів та адміністраторів бібліотеки
Реєстрація нових адміністраторів/читачів	Додати читача або адміністратора до бібліотеки

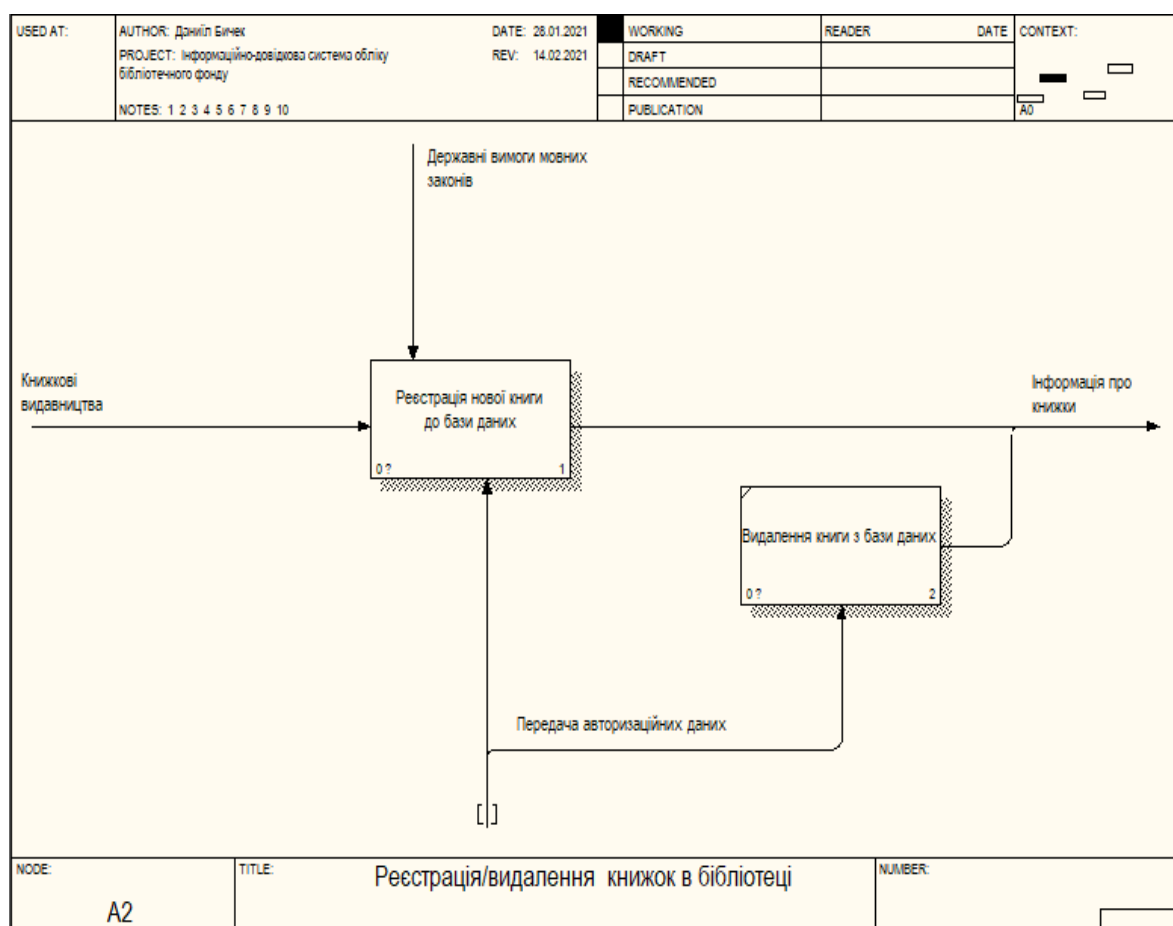


Рисунок 4 – декомпозиція блоку A2

Таблиця 5 – Роботи діаграми декомпозиції A2

Ім'я роботи (Activity name)	Визначення (Definition)
Реєстрація нової книги до БД	Зареєструвати новий екземпляр книги до бібліотеки
Видалення книги з БД	Видалити наявний екземпляр книги з бібліотеки

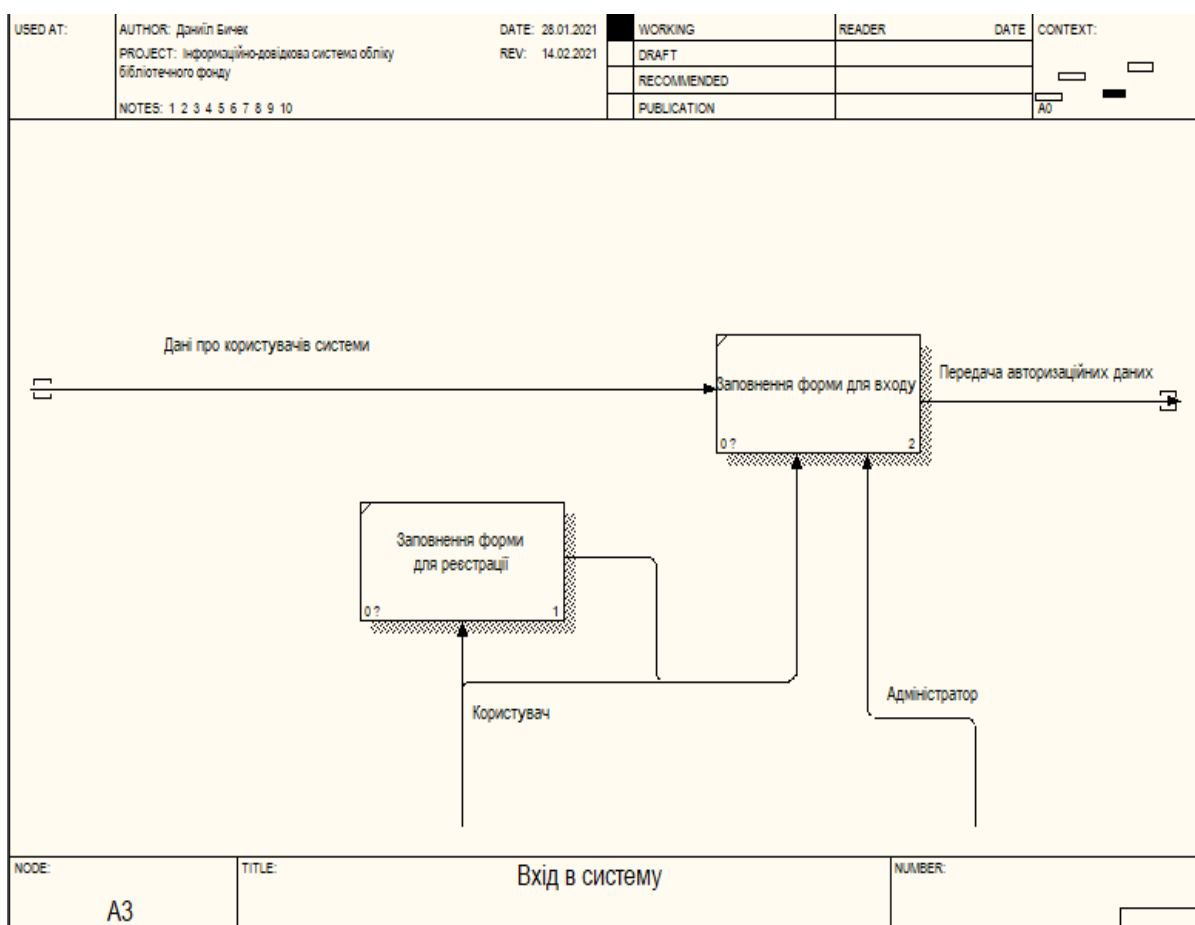


Рисунок 5 – декомпозиція блоку А3

Таблиця 6 – Роботи діаграми декомпозиції А3

Ім'я роботи (Activity name)	Визначення (Definition)
Заповнення форми для реєстрації	Заповнення форми для реєстрації в системі
Заповнення форми для входу	Заповнення форми для входу в систему

Останнім етапом в візуалізації проектованої інформаційної системи є побудова дерева вузлів. В результаті розроблених вище діаграм, маємо наступне дерево (Рис. 7):

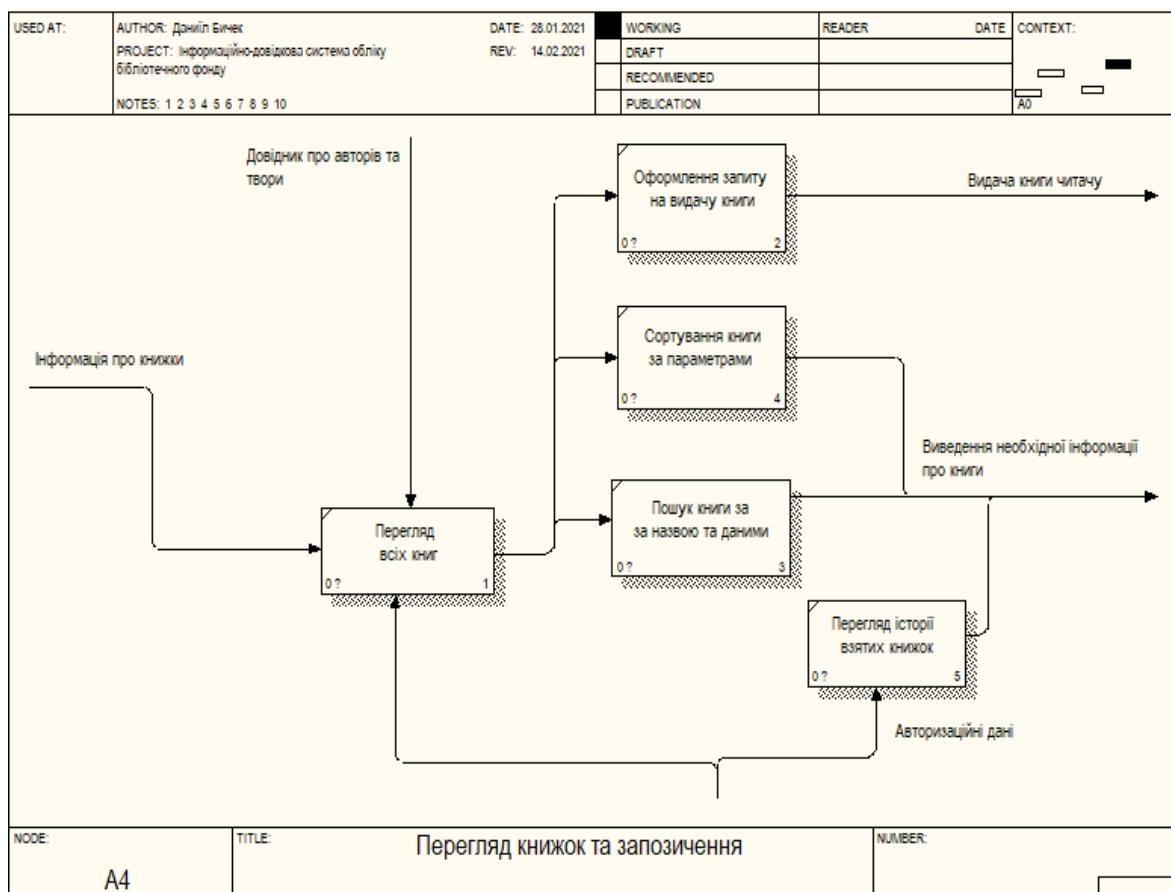


Рисунок 6 – декомпозиція блоку A4

Таблиця 7 – Роботи діаграми декомпозиції A4

Ім'я роботи (Activity name)	Визначення (Definition)
Оформлення запиту на видачу книги	Створити запит на запозичення книги з бібліотеки
Сортування книги за параметрами	Знайти перелік книжок в бібліотеці за заданими параметрами
Перегляд всіх книг	Переглянути всі наявні книги в бібліотеці
Пошук книги за назвою та даними	Знайти книгу в бібліотеці за назвою
Перегляд історії взятих книжок	Переглянути всі запозичені раніше книжки в бібліотеці

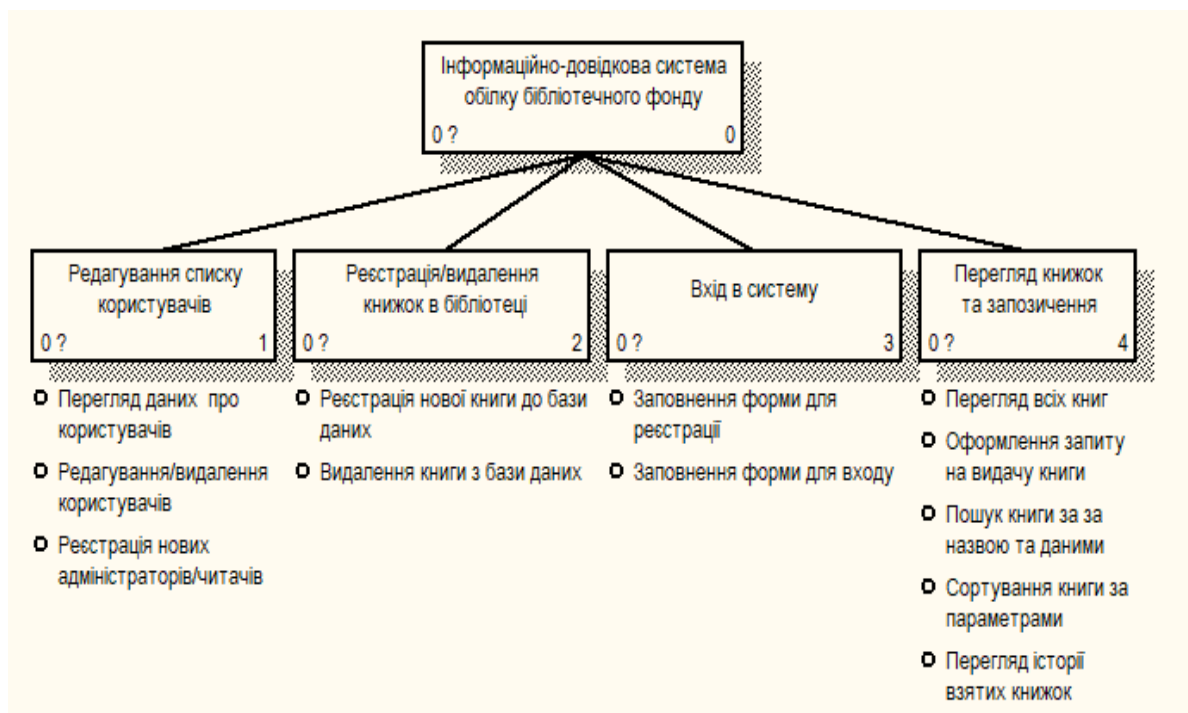


Рисунок 7 – дерево вузлів

Для завершення створення функціональної моделі було створено декомпозицію модуля «Реєстрація нової книги до бази даних» в нотації IDEF3 до якої додаємо стрілки у вигляді елементів діаграми «refferent» та основні блоків дій «activity tool box» (Рис. 8).

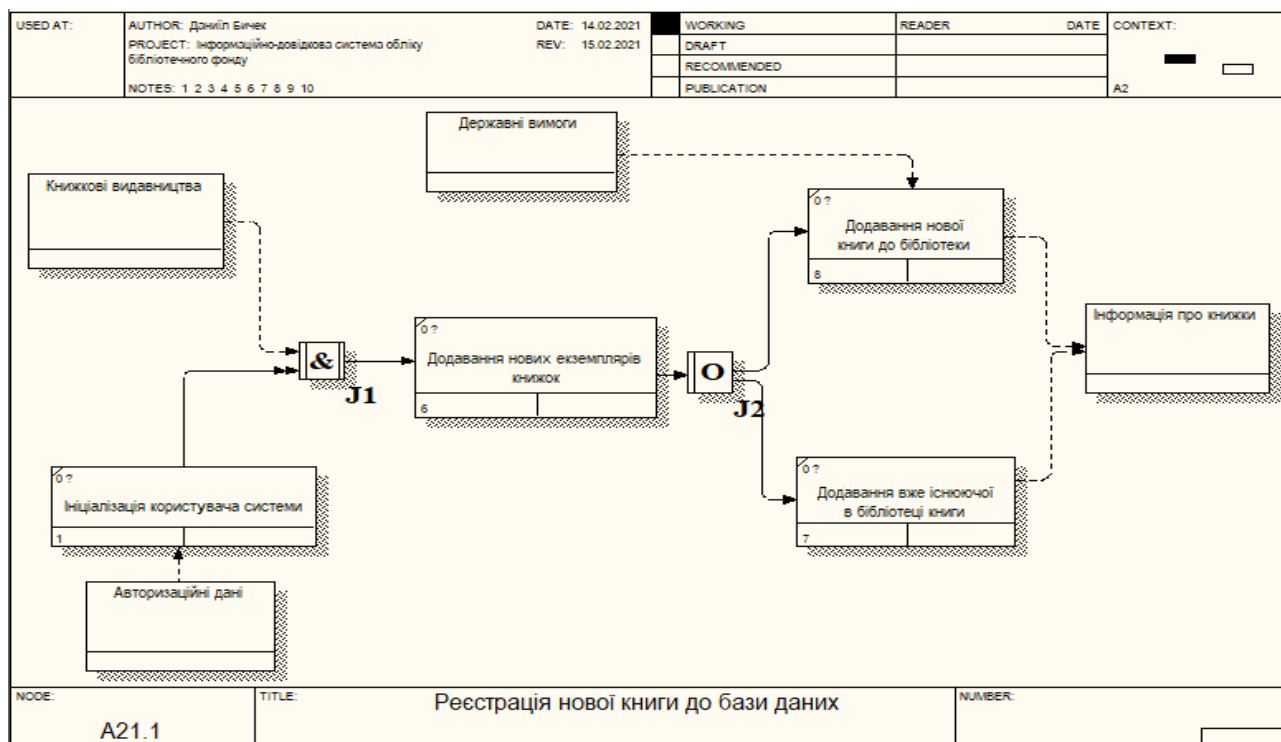


Рисунок 8 – Декомпозиція модуля «Реєстрація нової книги до бази даних» в IDEF3.

Також в моїй схемі присутні 2 типи перехресть:

- 1 Синхронне «І» - всі попередні процеси завершуються одночасно.
- 2 Асинхронне «Або» OR – один або декілька процесів мають бути завершені.

Даний модуль виходячи зі схеми буде працювати наступним чином:

Маємо авторизаційні дані користувача які ініціалізуються і отримуючи синхронно всю необхідну інформацію про нові книжки від книжкових видавництв, користувач починає додавати новий екземпляр книги в бібліотеку. У даному випадку може статися так, що реєстрована книга вже існує в бібліотеці, тому заново прописувати всю інформацію про книгу є зайвим. Тому користувач має два різних сценарії дій. Можна додати вже існуючу книгу в бібліотеку, або зареєструвати книгу в бібліотеку використовуючи правила які регламентуються державними вимогами. В результаті повертається інформація про нові додані книжки для подальшого використання даних..

1.4 Моделювання потоків даних

Діаграма DFD (Data Flow Diagramming) – це методологія графічного структурного аналізу, що описує зовнішні по відношенню до системи джерела і адресати даних, логічні функції, потоки даних і сховища даних, до яких здійснюється доступ. Діаграма потоків даних (DFD) - один з основних інструментів структурного аналізу і проектування інформаційних систем, що існували до широкого поширення Unified Modeling Language (UML)[5]. У випадку проектованої системи дана діаграма зможе надати інформацію про обмін даних між різними модулями системи.

Для моделювання потоків даних було використано програму All Fusion Process Medeler. Готова діаграма DFD зображена на рисунку 9.

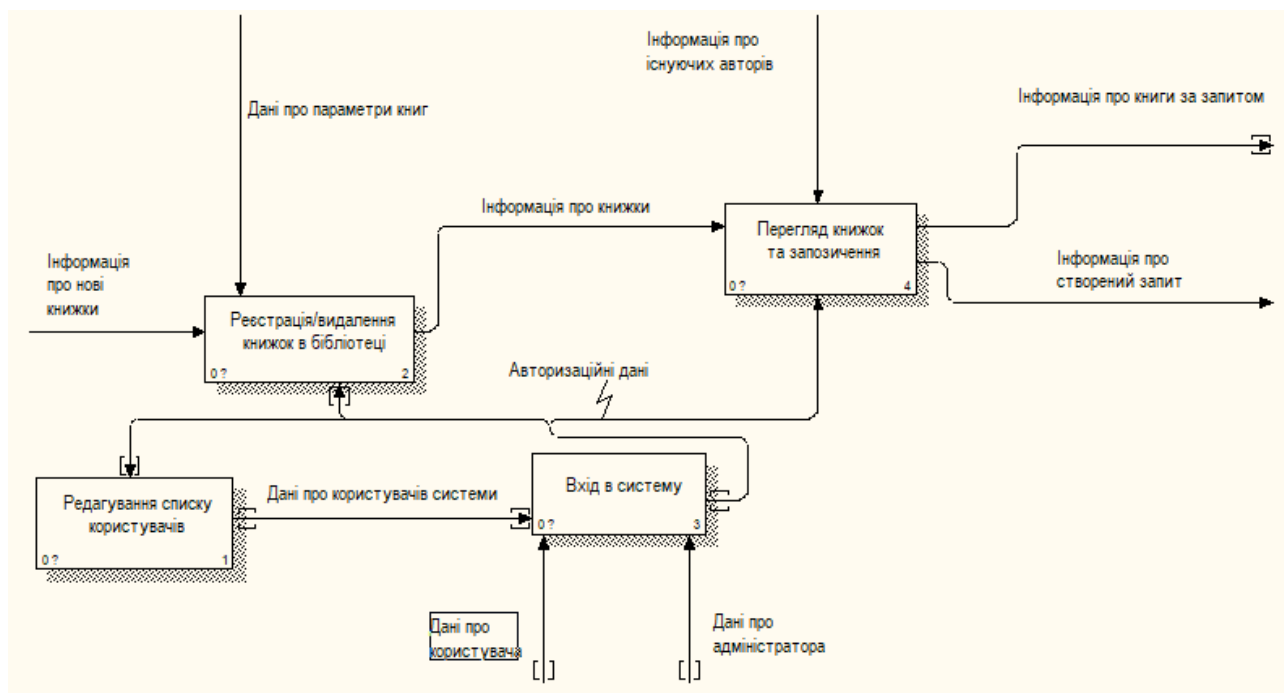


Рисунок 9 – Діаграма DFD.

2 Реалізація інформаційної системи

2.1 Об'єктно-орієнтований аналіз інформаційної системи

Для об'єктно-орієнтованого аналізу системи було розроблено наступні діаграми: діаграму варіантів використання, діаграму класів, кооперативну діаграму, діаграму станів та діаграму діяльності.

Під час першого етапу об'єктно-орієнтованого аналізу було розроблено діаграму використання котра зображена на рисунку 10.

Дана діаграма відображає роботу програми з такими акторами як адміністратор, користувач та база даних.

Варіанти використання:

- 1 Реєстрація в системі – користувач вносить дані про свою особистість;
- 2 Введення авторизаційних даних – користувач підтверджує себе вводячи логін та пароль ;
- 3 Відказ в доступі – користувач підтверджує не вірність введених даних при авторизації;
- 4 Вхід в систему в якості бібліотекаря – в залежності від авторизаційних даних користувач входить в якості бібліотекаря в систему;

- 5 Вхід в систему в якості читача – в залежності від авторизаційних даних користувач входить в якості читача в систему;
- 6 Додати/Видалити/Переглянути користувачів в системі – робота з користувачами в системі;
- 7 Додати/Видалити/Переглянути книжки в системі – робота з книжками в системі;
- 8 Здійснити запит на отримання книги – надсилання запиту на отримання певної книги;
- 9 Переглянути історію запозичених книжок – структурований перегляд всіх минулих запозичень;
- 10 Відкрити список книжок наявних в бібліотеці – структурований перегляд всіх книжок в бібліотеці;
- 11 Сортувати книжки за параметрами – перегляд книжок відсортованих у певній послідовності;
- 12 Знайти вибірку книжок – перегляд книжок за певними параметрами.

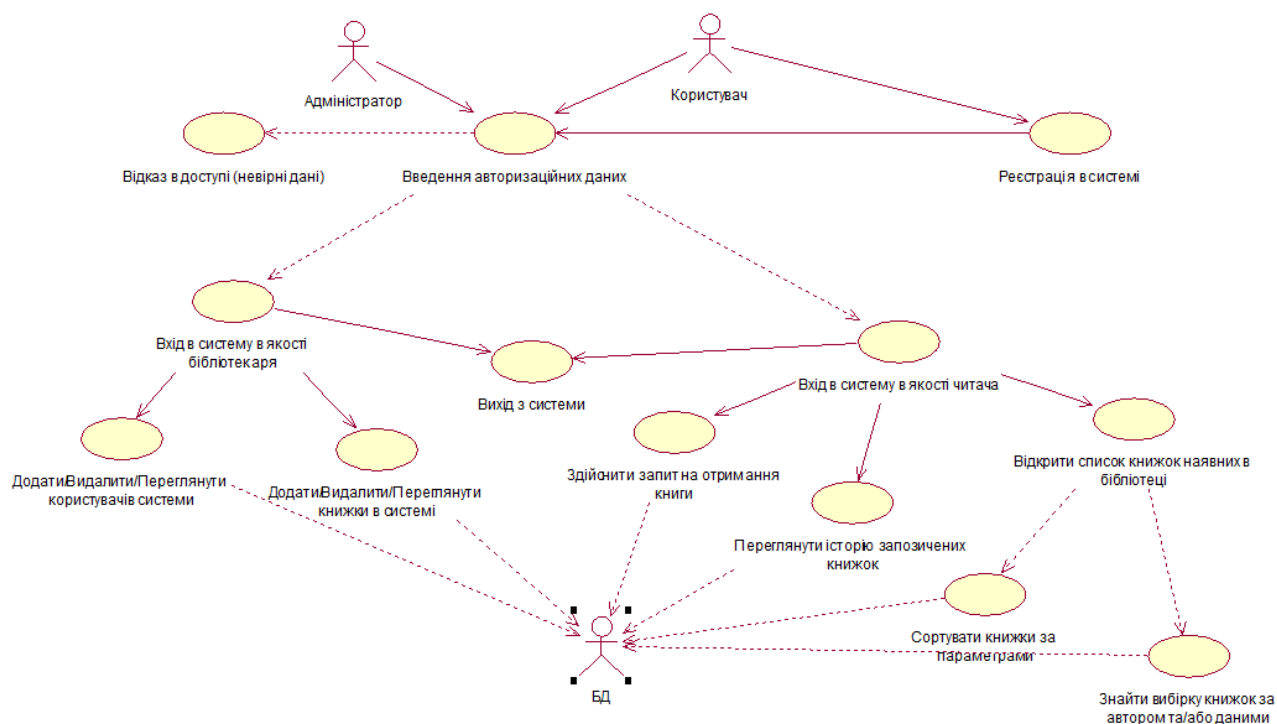


Рисунок 10 – Діаграма варіантів використання.

Під час другого етапу об'єктно-орієнтованого аналізу системи було розроблено діаграму класів котра зображена на рисунку 11.

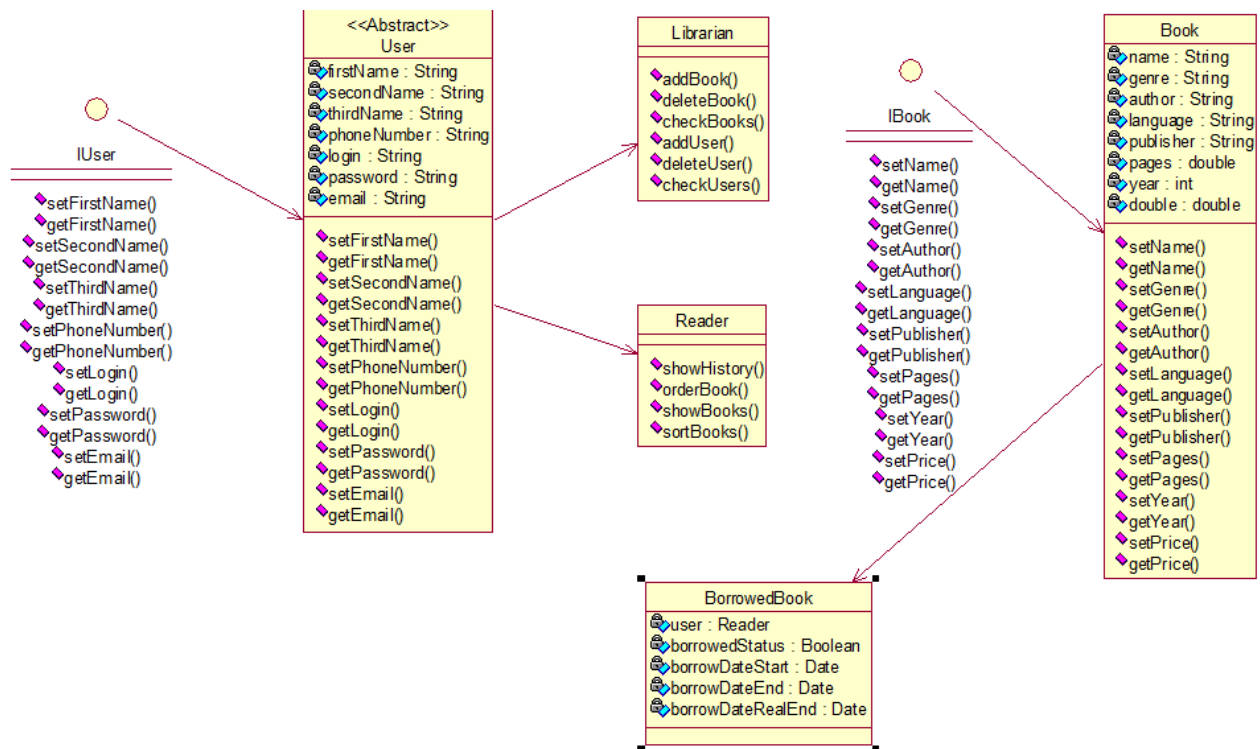


Рисунок 11 – Діаграма класів.

В даній діаграмі можна визначити два незалежних один від одного модулі: користувач та книга. Таким чином до користувача можна віднести класи:

- IUser: інтерфейс для класів User, Librarian та Reader, який визначає основні спільні методи для даних класів.
- User: абстрактний клас який задає необхідні поля і визначає тіла методів для класів Librarian та Reader.
- Librarian: клас який визначає основні властивості бібліотекаря.
- Reader: клас який визначає основні властивості читача.

Аналогічно до книги можна віднести класи:

- IBook: інтерфейс для класів Book та BorrowedBook, який визначає основні спільні методи для даних класів
- Book: клас який визначає основні властивості книги
- BorrowedBook: клас який визначає основні властивості запозиченої книги

Під час третього етапу об'єктно-орієнтованого аналізу системи було розроблено діаграму кооперацій системи котра зображена на рисунку 12.



Рисунок 12 – Кооперативна діаграма.

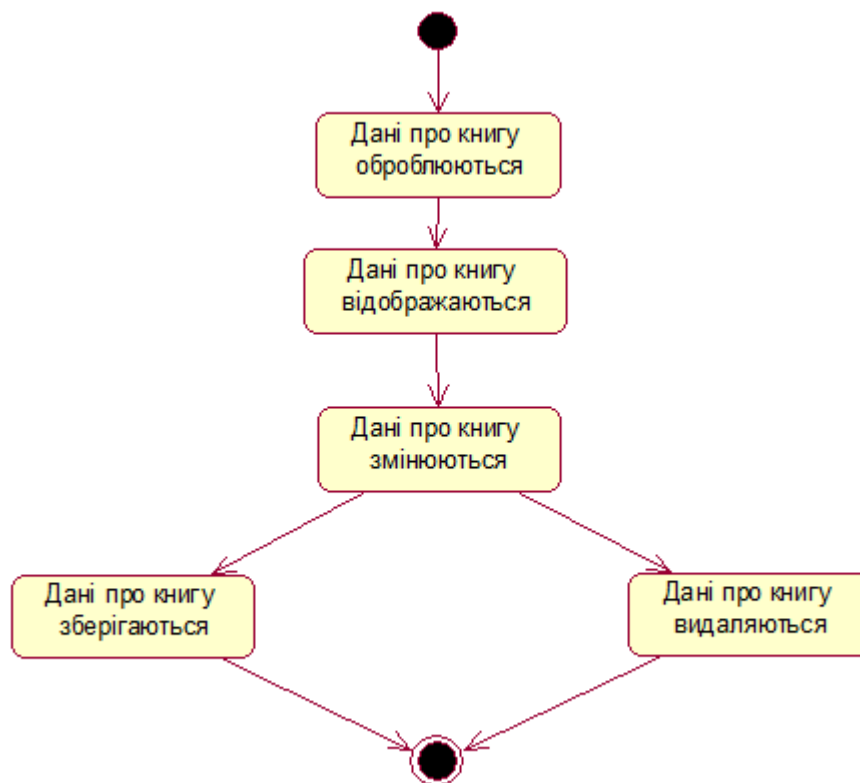
В даній діаграмі було визначено поведінку взаємодіючих груп об'єктів завдяки чому було зображено всі взаємозв'язки в програмі між різними елементами.

Під час четвертого етапу об'єктно-орієнтованого аналізу системи було розроблено діаграму станів котра зображена на рисунку 13.

В даній діаграмі було відображено всі стани в яких може перебувати основний об'єкт інформаційної системи – книга.

Книга може мати наступні стани:

- Дані про книгу обробляються – внесені дані про книгу зберігаються в базі даних;
- Дані про книгу відображаються – дані виносяться на екран;
- Дані про книгу змінюються – вносяться певні зміни що до інформації про книгу;
- Дані про книгу зберігаються – внесені дані зберігаються в системі.
- Дані про книгу видаляються – книга повністю видаляється з системи



Рисуно 13 – Діаграма станів.

Під час завершального етапу об’єктно-орієнтованого аналізу системи було розроблено діаграму діяльності котра зображена на рисунку 14.

В даній діаграмі система має два різні сценарії роботи. Системою може керувати як бібліотекар так і читач, тому їх можливості розмежовані. Таким чином бібліотекар має змогу змінювати певні дані в програмі в залежності від потреб. Читач в свою чергу може переглядати книжки та історію своїх запитів на запозичення. Також є розгалуження у випадку коли читач може запросити собі знайдену книгу, або її пропустити.

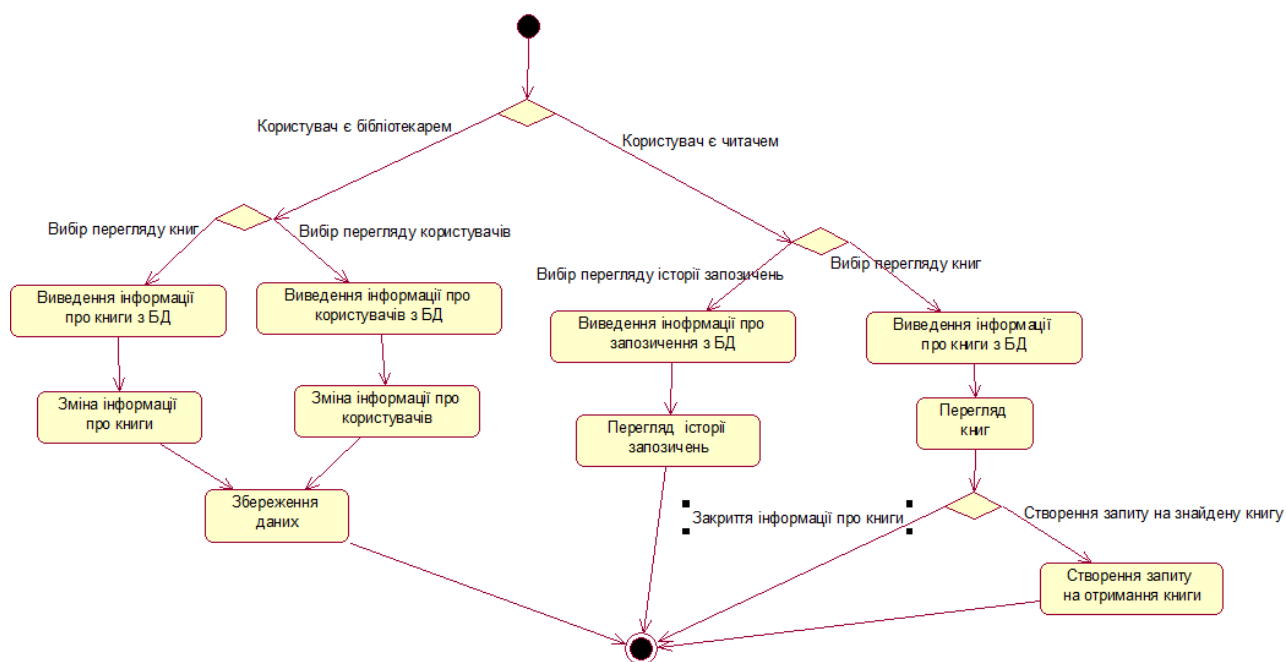


Рисунок 14 – Діаграма діяльності.

2.2 Розроблення технічних вимог до інформаційної системи

В процесі розроблення технічних вимог до інформаційної системи було потрібно вирішити такі питання як:

- Призначення та цілі інформаційної системи;
- Вимоги до надійності системи;
- Вимоги до загальної архітектури.

2.2.1 Призначення та цілі

Відповідаючи поставленому призначенню системи а саме оптимізації процесів моніторингу книжок та їх пошуку було поставлено певні цілі які повинен виконувати проект, а саме:

- Робота з великою кількістю користувачів
- Навігація по книжкам в бібліотеці
- Можливість внесення змін про книжки та користувачів
- Розподіленість системи для читачів та бібліотекарів
- Можливість запозичення книжок з контролем дат
- Реєстрація нових користувачів до системи

2.2.2 Вимоги до надійності системи

Система повинна працювати на локальному сервері бібліотеки таким чином не залежачи від мережевого підключення до інтернету.

Сам процес користування програмою повинен проходити лише через авторизацію користувача, з метою контролю виконаних дій і збереження інформації від третіх осіб.

2.2.3 Вимоги до загальної архітектури

Проект повинен мати раціонально побудовану систему класів з метою їх подальшої модифікації. Таким чином об'єкти системи повинні обов'язково мати інтерфейси для можливості подальшого легкого оновлення програми і додавання нових об'єктів. Використовувати стандартизовані назви елементів і класів в системі.

2.3 Програмування інформаційної системи

Для створення проекту було обрано мову програмування C# через певні переваги над іншими мовами:

- Значний досвід роботи
- Об'єктно орієнтований підхід
- Можливість швидко створювати якісні інтерфейсні вікна
- Сумісність і оптимізованість з операційною системою Windows

Завдяки обраній мові програмування C# можливо розробити додаток для Windows з більш точним використанням всіх переваг операційної системи.

Для збереження даних в даній інформаційній системі було використано базу даних Microsoft SQL на основі інтерфейсу PhpMyAdmin.

В якості коду наведено саме клас для роботи програми з створеною базою даних:

```
class DB
{
    //Строка підключення до бази даних з адресою
    MySqlConnection connection = new
    MySqlConnection("server=localhost;port=3306;username=root;password=root;datab
ase=librarydata");
```

```

        //Метод під'єднання до бази даних
public void openConnection()
{
    if (connection.State == System.Data.ConnectionState.Closed)
        connection.Open();
}
//Метод від'єднання від бази даних
public void closeConnection()
{
    if (connection.State == System.Data.ConnectionState.Open)
        connection.Close();
}
//Метод видачі адреси бази даних
public MySqlConnection getConnection()
{
    return connection;
}
//Метод оновлення певної таблиці
public void updateDB(DataGridView GridView, String TableName)
{
    DataTable table = new DataTable();

    MySqlDataAdapter adapter = new MySqlDataAdapter();

    this.openConnection();

    MySqlCommand command = new MySqlCommand("SELECT *
FROM `" + TableName + "`", this.getConnection());

    DataSet data = new DataSet();

    adapter.SelectCommand = command;
    adapter.Fill(data);
    adapter.Fill(table);

    GridView.DataSource = data.Tables[0];

    this.closeConnection();
}

```

В наведеному програмному коді можна виділити поле класу в якому зберігається адресний рядок підключення до бази даних та методи:

- OpenConnection – під’єднання до бази даних з метою подальшого обміну інформацією;
- CloseConnection – відключення від бази даних;
- GetConnection – отримання адресного рядка під’єднання до бази даних;
- UpdateDB – оновлення таблиці новою інформацією в базі даних;

При запуску програми користувач бачить форму входу до системи (Рис. 15). При вході в систему в якості бібліотека користувач бачить меню з всім необхідним функціоналом який можливо використовувати завдяки перемикачам в меню.

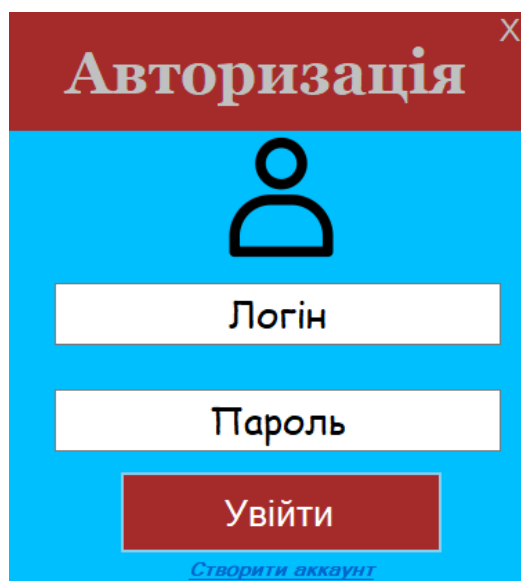


Рисунок 15 – Меню входу в програму.

Таким чином бібліотекар може переглядати інформацію про користувачів, книжки та запити. Змінювати, видаляти та додавати нову інформацію як про користувачів так і про книги. Також на цьому меню бібліотекар має можливість підтвердження запитів на запозичення книжок (Рис. 16).

Вітаю Даниїл

	id	firstName	econdName	thirdName	phoneNumber	login	pas
▶	22	Андрій	Набаран...	Олександро...	+380673756647	andrej	
	23	Іван	Джух	Вікторович	+380967445833	mylog	my
	21	Даниїл	Бичек	Максимович	+380989334408'	Dan	r
*							

Пошук
Видалення
Зміна
Додавання

Параметри пошуку

id:
 Ім'я:
 Прізвище:
 По-батькові:
 Пошта:
 Телефон:

Оновити базу даних

Користувачі
Книги
Запити

Рисунок 16 – Інтерфейс бібліотекаря програми.

При необхідності бібліотекар може сортувати на шукати необхідну інформацію, а також змінювати дані в системі (Рис. 17).

Вітаю Даниїл

	id	name	genre	author	language	publisher	pages	year
▶	1	Хоббіт	Фентезі	Толкєн	Українс...	Абабага...	436	1982
	7	Мобі-Дік	Роман	Мелвіл	Українс...	Абабага...	238	1851
	8	Великий...	Наратив	Фійджер...	Українс...	Абабага...	1200	1982
*								

Пошук
Видалення
Зміна
Додавання

Параметри пошуку

id:
 Назва:
 Жанр:
 Автор:
 Мова:
 Видавництво:

Оновити базу даних

Користувачі
Книги
Запити

Рисунок 17 – Пошук книжок за вказаними параметрами.

2.4 Тестування та налагодження інформаційної системи

В процесі тестування системи не було виявлено критичних помилок в програмі. При перемиканні типу користувача який входить в систему було виявлено передчасний вихід з програми. Тому було налагоджено на оптимізовано логіку появи та зникнення форма в програмі, так їх коректного закриття. При спробі введення пустої інформації під час реєстрації програма не дозволяє здійснити збереження користувача. Також не можливо створити користувачів з однаковими логінами.

2.5 Економічна оцінка проекту розроблення інформаційної системи

Для економічної оцінки проекту розроблення інформаційної системи було використано дві моделі економічної оцінки вартості розробки програмного забезпечення. Ці моделі мають наступні назви: Use Case Points та COCOMO II.

Спочатку було оцінено розроблювану інформаційну систему за методом Use Case Points.

Use-Case Points (UCP)[3] – це методика оцінки яка використовується для прогнозування обсягу програмного забезпечення в проектах розробки програмного забезпечення. Метод застосовується при використанні Unified Modeling Language (UML) та методології розробки Rational Unified Process (RUP) для розробки програмного забезпечення. Ідея методу заснована на тому, що вимоги до системи записані у вигляді варіантів використання, що є частиною методів моделювання UML. Обсяг програмного забезпечення розраховується на основі елементів варіантів використання системи шляхом розкладання з урахуванням технічних припущень і припущень про оточення. Метод може бути використаний для підрахунку передбачуваних витрат за проектом.

На першому етапі економічної оцінки проекту за моделлю Use Case Points було оцінено складність варіантів використання (таблиця 8).

Таблиця 8 – Вага варіантів використання

Use Case	Кількість транзакцій	Класифікація	Вага
Вхід	2	простий	5

Реєстрація в системі	2	простий	5
Відказ в доступі (невірні дані)	1	простий	5
Вихід	1	простий	5
Додати/Видалити/Переглянути користувачів системи	4	середній	10
Додати/Видалити/Переглянути книжки системи	4	середній	10
Здійснити запит на отримання книги	2	простий	5
Переглянути історію запозичених книжок	1	простий	5
Відкрити список книжок наявних в бібліотеці	1	простий	5
Сортувати книжки за параметрами	3	простий	5
Знайти вибірку книжок за автором та/або даними	3	простий	5

В результаті було пораховано Unadjusted Use-Case Weight (UUCW) за формулою (1)

$$UUCW = 5 * NSUC + 10 * NAUC + 15 * NCUC,$$

де NSUC – це кількість простих випадків використання;
 NAUC – це кількість середніх випадків використання;
 NCUC – це кількість складних випадків використання.

Для розроблюваної системи $UUCW = 5 * 9 + 10 * 2 + 15 * 0 = 65$.

$UUCW = 65$.

На другому етапі економічної оцінки проекту за моделлю Use Case Points було класифіковано акторів, а також призначено вагу (таблиця 9).

Таблиця 9 – актори

Актор	Приклад	Вага
Клієнт	Читач в бібліотеці	3
Адміністратор	Бібліотекар	3

База даних	Система	2
------------	---------	---

Потім було пораховано Unadjusted Actor Weight (UAW) за формулою (2)

$$UAW = 1 * NSA + 2 * NAA + 3 * NCA,$$

де NSA – це значення простих акторів;

NAA – це значення середніх акторів;

NCA – це значення складних акторів.

Для розроблюваної системи $UAW = 1 * 0 + 2 * 1 + 3 * 2 = 8$

$UAW = 8$.

Далі було визначено TF(коефіцієнт технічної складності) проекту (таблиця 10).

Таблиця 10 – Коефіцієнт технічної складності

Фактор	Опис	Вага(W)	Номінальне значення (RV)	Вплив фактору (I = W * RV)
T1	Розподілена система	2.0	3.0	6.0
T2	Час відповіді/продуктивність	1.0	2.0	2.0
T3	Ефективність для кінцевого користувача	1.0	4.0	4.0
T4	Складність внутрішньої обробки	1.0	2.0	2.0
T5	Можливість повторного використання коду	1.0	1.0	1.0
T6	Легко встановити	0.5	2.5	1.25
T7	Легко використовувати	0.5	5.0	2.5
T8	Фактор крос-платформеності	2.0	0	0
T9	Обслуговування системи	1.0	3.5	3.5
T10	Багатопоточна обробка	1.0	3.0	3.0
T11	Особливості безпеки	1.0	3.0	3.0
T12	Доступ третіх осіб	1.0	1.0	1.0
T13	Навчання кінцевих користувачів	1.0	1.0	1.0
TF:				30.25

Потім було визначено TCF(коефіцієнт технічної складності) за формулою (3)

$$TCF = 0,6 + (0,01 * TF)$$

Для розроблюваної системи $TCF = 0,6 + (0,01 * 30,25) = 0,9025$

$$TCF = 0,97$$

Наступним етапом потрібно визначити EF(фактор навколишнього середовища) проекту(таблиця 11).

Таблиця 11 – Фактор навколишнього середовища

Фактор	Опис	Вага(W)	Номінальне значення (RV)	Вплив фактору (I = W * RV)
E1	Схожість на подібні системи	1.5	2.0	3.0
E2	Досвід використання	0.5	2.0	1.0
E3	Об'єктно-орієнтований досвід	1.0	4.0	4.0
E4	Здібність головного аналітика	0.5	2.0	1.0
E5	Мотивація	1.0	2.0	2.0
E6	Стійкість вимог	2.0	5.0	10
E7	Не повних графік роботи персоналу	-1.0	2.0	-2.0
E8	Складність мови програмування	-1.0	1.5	-1.5
EF:				17.5

Далі було обчислено ECF(фактор навколишнього середовища) за формулою (4)

$$ECF = 1,4 + (-0,03 * EF)$$

Для розроблюваної системи $ECF = 1,4 + (-0,03 * 17,5) = 0,875$

$$ECF = 1,085$$

Отримавши всі необхідні значення вартості системи було обчислено UCP(коригована точка використання) за формулою (5)

$$UCP = (UUCW + UAW) * TCF * ECF$$

Для розроблюваної системи $UCP = (70 + 8) * 0,97 * 1,085 = 82,0911$

$$UCP = 82,0911$$

Далі було оцінено розроблювану інформаційну систему за методом COCOMO II.

Розглянемо спершу модель COCOMO (Constructive Cost Model)[2], яка являється конструктивною моделлю вартості, що була розроблена на початку 80-х років Баррі Боемом для оцінки трудомісткості розробки програмних продуктів. Модель складається з ієрархії трьох послідовно деталізованих та уточнюючих рівнів (режимів). На кожному рівні всі проекти розбиваються на три групи за рівнем складності:

- 1) розповсюджений або органічний тип (organic projects);
- 2) напівнезалежний або напіврозподілений тип (semidetached projects);
- 3) вбудований тип (embedded projects).

У 1997 методика COCOMO була вдосконалена і отримала назву COCOMO II.

Розрізняють дві стадії оцінки проекту:

- 1) попередня оцінка на початковій фазі (Early Design);
- 2) детальна оцінка після опрацювання архітектури (Post Architecture).

Для розрахунку трудомісткості необхідно спочатку оцінити фактори (чинники) масштабу (Scale Drivers) та множники трудомісткості (Cost Drivers або Effort Multipliers). Фактори масштабу застосовуються на двох стадіях оцінки проекту. Множники трудомісткості відрізняються для різних стадій оцінки проекту. На різних стадіях відрізняється їх кількість і значення. Для стадії Early Design необхідно оцінити сім множників трудомісткості, а для стадії Post Architecture – сімнадцять.

Формула оцінки трудомісткості проекту в люд. × міс. має вигляд

$$PM = EAF * A * (SIZE)^E$$

$$E = B + 0.01 * \sum_{j=1}^5 SF_j,$$

де $B=0.91$;

$A=2.94$ для попередньої оцінки;

$A=2.45$ для детальної оцінки;

SF_j - фактори (чинники) масштабу (Scale Factors)(таб.11);

SIZE - обсяг програмного продукту в тисячах рядків вихідного тексту (KSLOC – Kilo of Source Line of Code).

EAF (Effort Adjustment Factor) – добуток обраних множників трудоемкості:

$$EAF = \prod_{k=1}^n EM_k,$$

де EM - множники трудомісткості (Effort Multipliers);

n=7 – для попередньої оцінки (таб.12);

n=17 – для детальної оцінки(таб.13).

EAF (Effort Adjustment Factor) – добуток обраних множників трудоемкості:

$$EAF = \prod_{k=1}^n EM_k .$$

Отже по методології COSOMO II[1] було розраховано наступні значення (Рис. 18):

Software Size Sizing Method **Source Lines of Code** ▼

[SLOC](#) % Design Modified % Code Modified % Integration Required Assessment and Assimilation (0% - 8%) Software Understanding (0% - 50%) Unfamiliarity (0-1)

New

Reused

Modified

Software Scale Drivers

Precedentedness ▼ Architecture / Risk Resolution ▼ Process Maturity ▼

Development Flexibility ▼ Team Cohesion ▼

Software Cost Drivers

Product

Required Software Reliability ▼

Data Base Size ▼

Product Complexity ▼

Developed for Reusability ▼

Documentation Match to Lifecycle Needs ▼

Personnel

Analyst Capability ▼

Programmer Capability ▼

Personnel Continuity ▼

Application Experience ▼

Platform Experience ▼

Language and Toolset Experience ▼

Platform

Time Constraint ▼

Storage Constraint ▼

Platform Volatility ▼

Project

Use of Software Tools ▼

Multisite Development ▼

Required Development Schedule ▼

Maintenance ▼

Software Labor Rates

Cost per Person-Month (Dollars)

Results**Software Development (Elaboration and Construction)****Staffing Profile**

Effort = 0.0 Person-months
 Schedule = 0.5 Months
 Cost = \$0

Your project is too small to display a staffing profile due to truncation.

Total Equivalent Size = 1 SLOC
 Effort Adjustment Factor (EAF) = 1.59

Acquisition Phase Distribution

Phase	Effort (Person- months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	0.0	0.1	0.0	\$0
Elaboration	0.0	0.2	0.0	\$0
Construction	0.0	0.3	0.0	\$0
Transition	0.0	0.1	0.0	\$0

Рисунок 18 – результат розрахунку економічної вартості проекту за допомогою COSOMO II

В результаті розрахована система має строк виконання пів місяця (15 днів) і має добуток обраних множників трудомісткості (Effort Adjustment Factor) рівний 1,59.

Висновки

В процесі курсового проектування було створено інформаційну систему “Інформаційно довідкова система обліку бібліотечного фонду”. Після виконання всіх завдань, поставлених в технічному завданні та перевірки створеної системи на працездатність, можна з точністю сказати що в результаті було створено вправно функціонуючу інформаційну систему яка може бути вільно застосована в відповідних сферах діяльності.

В процесі роботи над курсовим проектом було проаналізовано предметну область. Зокрема, визначено функції схожих, існуючих систем; майбутній функціонал проектованої інформаційної системи. Для вирішення задачі опису схеми функціонування системи було створено контекстну діаграму, діаграми декомпозиції окремих модулів та діаграму потоків даних.

Для детального розгляду системи було створено об’єктно-орієнтовану модель з використанням мови UML. В результаті було створено такі діаграми як: діаграма варіантів використання, діаграма класів, станів, діяльності, кооперативна діаграма.

Таким чином, під час курсового проектування було розв’язано поставлені задачі та створено інформаційну систему організації. Створена інформаційна система дозволяє підвищити ефективність за рахунок автоматизації ряду задач що робить розроблення і впровадження доцільним.

На основі розроблених діаграм було з нуля створено інформаційну систему на мові програмування C#. Завдяки створеній системі стало можливим легко відслідковувати книжки в бібліотеці та розподіляти користувачів.

Для оцінювання ресурсів, необхідних для реалізації проекту було розраховано важкість розробки за допомогою двох методик: Use Case Point та COSOMO II. По першій методиці було отримано значення UCP = 82,0911 , а по другій методології було визначено що розробка проекту буде займати 15 днів, що і було дуже близько до реальності.

Список літератури

- 1 Онлайн калькулятор COCOMO II

<http://softwarecost.org/tools/COCOMO/>.

- 2 <https://ru.wikipedia.org/wiki/COCOMO>.

- 3 https://ru.wikipedia.org/wiki/Метод_балльной_оценки_вариантов_использования.

- 4 https://uk.wikipedia.org/wiki/Rational_rose.

- 5 <https://habr.com/ru/post/458680/>.

- 6 <https://calibre-ebook.com/>.

- 7 <http://myrulib.lintest.ru/>.

Додатки

Програмна збірка системи прикріплена в архіві разом із пояснювальною запискою.