Міністерство освіти і науки України
Криворізький національний університет
Кафедра моделювання та програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавра

зі спеціальності 121 – Інженерія програмного забезпечення

На тему: Розробка програмного проєкту автоматизації тайм-менеджменту завдань неформальної освіти

Засвідчую, що в цій кваліфікаційній роботі немає запозичень із праць інших авторів без відповідних посилань.
Студент гр. ІПЗ–21-1
_____/ Я. А. Романенко /

Керівник кваліфікаційної роботи _____/ І. А. Котов /

Завідувач кафедри _____/ А. М. Стрюк /

Кривий Ріг
2025

Криворізький національний університет

Факультет: Інформаційних технологій

Кафедра: Моделювання та програмного забезпечення

Ступінь вищої освіти: бакалавр

Спеціальність: 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри

_____ А. М. Стрюк

«____» _____ 2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу

студенту групи ІПЗ–21-1 Романенко Ярославу Андрійовичу

1. Тема: Розробка програмного проєкту автоматизації тайм-менеджменту завдань неформальної освіти затверджена наказом по КНУ № 199с від «10» квітня 2025р.

2. Термін подання студентом закінченої роботи: «09» червня 2025р.

3. Вихідні дані по роботі: програмний комплекс для автоматизації тайм-менеджменту завдань неформальної освіти.

4. Зміст пояснювальної записки (перелік питань, що їх треба розробити): проаналізувати підходи до організації тайм-менеджменту для сфери неформальної освіти, проаналізувати існуючи програмні системи підтримки тайм-менеджменту, спроектувати додаток, розробити програмне забезпечення системи автоматизації тайм-менеджменту завдань неформальної освіти, здійснити тестування розробленого додатку.

5. Перелік ілюстративного матеріалу: функціональна схема, блок–схема алгоритму, зображення екранних форм додатку.

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Термін виконання етапів роботи
1	Пошук науково-практичних джерел та інтернет-ресурсів з тематики роботи	03.02.25 – 09.02.25
2	Аналіз існуючих підходів і методів вирішення проблеми	10.02.25 – 23.02.25
3	Формулювання актуальності і постановка завдань роботи	24.02.25 – 02.03.25
4	Оформлення матеріалів першого розділу роботи	03.03.25 – 09.03.25
5	Розробка структурної і функціональної схем та основних алгоритмів програмного проєкту	10.03.25 – 23.03.25
6	Оформлення матеріалів другого розділу роботи	24.03.25 – 06.04.25
7	Розробка візуального інтерфейсу, баз даних і програмних модулів проєкту	07.04.25 – 20.04.25
8	Оформлення матеріалів третього розділу роботи	21.04.25 – 11.05.25
9	Оформлення додатків	12.05.25 – 18.05.25
10	Тестування розробленої програмного комплексу	19.05.25 – 01.06.25
11	Оформлення пояснювальної записки та супровідних документів	02.06.25 – 08.06.25

Дата видачі завдання: «03» лютого 2025 р.

Студент: _____ / Я. А. Романенко /

Керівник роботи: _____ / І. А. Котов /

РЕФЕРАТ

АВТОМАТИЗАЦІЯ, БАЗА ДАНИХ, БЛОК-СХЕМА, ГРАФ, ІНТЕРФЕЙС, КОНТЕКСТ, МЕНЕДЖМЕНТ, МОДЕЛЬ, ОСВІТА, ОСВІТНІЙ ПРОЦЕС, ПРОЕКТ, УПРАВЛІННЯ, ТЕХНОЛОГІЯ

Пояснювальна записка: 81 с., 55 рис., 5 табл., 1 дод., 21 джерело.

Кваліфікаційна робота: «Розробка програмного проєкту автоматизації тайм-менеджменту завдань неформальної освіти».

Мета випускової кваліфікаційної роботи: проєктування і розробка багатокористувацького web-застосунку автоматизації процесів планування, управління і аналізу тайм-менеджменту занять неформальної освіти.

Об’єктом дослідження і проектування є: процеси планування, управління і аналізу тайм-менеджменту занять неформальної освіти, а також програмні комплекси автоматизації професійного тайм-менеджменту.

Теоретична частина роботи присвячена аналізу проблеми організації неформальної освіти для здобувачів у вищий школі, підходів до організації тайм-менеджменту для сфери неформальної освіти, а також існуючих програмних систем підтримки тайм-менеджменту. Сформульовано актуальність проектування web-систем для автоматизації тайм-менеджменту.

Практична частина кваліфікаційної роботи визначає розробку структурних і функціональних моделей, структур даних, алгоритмів, бази даних, моделей користувацького інтерфейсу, базових програмних модулів, програмних модулів аналізу тайм-менеджменту, програмних модулів звітів результатів неформальної освіти.

Аналіз розробленого програмного комплексу тайм-менеджментом неформальної освіти показує, що отримані програмні результати можуть бути використані під час організації навчального процесу як у середніх, так і у вищих навчальних закладах. Використання розробленого web-додатку підвищить ефективність занять неформальної освіти.

ABSTRACT

AUTOMATION, DATABASE, FLOWCHART, GRAPH, INTERFACE, CONTEXT, MANAGEMENT, MODEL, EDUCATION, EDUCATIONAL PROCESS, PROJECT, MANAGEMENT, TECHNOLOGY

Explanatory note: 81 p., 55 fig., 5 tab., 1 supplement, 21 sources.
Qualification work: «Development of a software project to automate time management of non-formal education tasks».

The purpose of the final qualifying work: design and development of multi-user web-application of automation of processes of planning, management and analysis of time-management of non-formal education tasks.

The object of research and design are: processes of planning, management and analysis of time management of non-formal education classes, as well as software complexes of automation of professional time management.

The theoretical part of the work is devoted to the analysis of the problem of organizing non-formal education for applicants in higher education, approaches to the organization of time-management for non-formal education, as well as existing software systems to support time-management. The urgency of designing web-systems for time-management automation is formulated.

The practical part of qualification work determines the development of structural and functional models, data structures, algorithms, database, user interface models, basic program modules, program modules of time-management analysis, program modules of non-formal education results reports.

The analysis of the developed program complex by time-management of non-formal education shows that the obtained program results can be used in the organization of the educational process in both secondary and higher educational institutions. The use of the developed web-application will increase the effectiveness of non-formal education classes.

ЗМІСТ

ВСТУП……. 8
1 ПРОБЛЕМА АВТОМАТИЗАЦІЇ ТАЙМ-МЕНЕДЖМЕНТУ ДЛЯ ВИРІШЕННЯ ЗАВДАНЬ НЕФОРМАЛЬНОЇ ОСВІТИ…….. 10
1.1 Аналіз проблеми організації неформальної освіти для здобувачів у вищий школі……..…….. 10
1.2 Аналіз загальних принципів тайм-менеджменту……... 15
1.3 Аналіз підходів до організації тайм-менеджменту для сфери неформальної освіти……... 19
1.4 Аналіз існуючих програмних систем підтримки тайм-менеджменту…….…….. 24
1.5 Завдання розробки системи автоматизації тайм-менеджменту завдань неформальної освіти ……...……. 30
2 РОЗРОБКА МОДЕЛЕЙ І АЛГОРИТМІВ ПРОГРАМНОГО КОМПЛЕКСУ ТАЙМ-МЕНЕДЖМЕНТУ ЗАВДАНЬ НЕФОРМАЛЬНОЇ ОСВІТИ ……..……... 33
2.1 Розробка математичних моделей тайм-менеджменту завдань неформальної освіти……...…… 33
2.2 Розробка структурної моделі програмного комплексу тайм-менеджменту……..... 38
2.3 Розробка функціональної моделі програмного комплексу……..... 41
2.4 Розробка структури бази даних системи тайм-менеджменту…….. 43
2.5 Розробка блок-схем алгоритмів програмних модулів системи тайм-менеджменту……..... 46
2.6 Розробка моделі візуального інтерфейсу користувача програмного комплексу…… 49
3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ АВТОМАТИЗАЦІЇ ТАЙМ-МЕНЕДЖМЕНТУ ЗАВДАНЬ НЕФОРМАЛЬНОЇ ОСВІТИ….. 52

3.1 Розробка програмних модулів візуального інтерфейсу системи автоматизації …….. 52
3.2 Розробка бази даних автоматизованої системи ……..... 60
3.3 Розробка основних програмних модулів автоматизованої системи. 63
3.4 Розробка програмних модулів аналізу тайм-менеджменту…….. 65
3.5 Розробка програмних модулів звітів результатів неформальної освіти……. 67
ВИСНОВКИ…… 71
ПЕРЕЛІК ПОСИЛАНЬ…….. 73
ДОДАТОК А……... 75

ВСТУП

Сьогодні проблема організації неформальної освіти для здобувачів у вищий школі стає дедалі актуальнішою в сучасному освітньому середовищі. Для розкриття цієї теми необхідно насамперед провести аналіз усієї інфраструктури неформальної освіти.

Розглянемо формальну та неформальну освіту, її можливості та атрибути. Насамперед, необхідно розглядати такі аспекти неформальної освіти, як права на неформальну освіту, принципи та атрибути формальної та неформальної освіти, загальні визначення та концепції, що стосуються освіти.

Неформальна освіта фокусується на питаннях, які стосуються повсякденного життя учасників. Основна мета цього виду освіти – підготувати громадян, здатних вирішувати повсякденні проблеми, розвивати навички, готуватися до роботи, колективно організовуватися, краще розуміти навколишній світ і критично ставитися до інформації, яку вони отримують. Це досягається шляхом максимального використання освітніх елементів, що вже існують в освітньому процесі, іноді змішаних з новими елементами, введеними викладачами, а також шляхом експериментів з колективними діями, часто організованими за тематичними напрямками.

З огляду на те, що управління часом - це навичка, яку можна розвивати й удосконалювати за допомогою практики, і що це призводить до підвищення продуктивності, зниження стресу та поліпшення якості життя, йому варто приділяти велику увагу. Це дає такі переваги - планування, організація діяльності, а також знання доступних ресурсів, чи то технологічні, чи то прості методи та інструменти для щоденного використання. Основний підхід до планування часу ґрунтується на так званій матриці Ейзенхауера. Був проведений аналіз загальних принципів тайм-менеджменту. З огляду на те, що управління часом – це навичка, яку можна розвивати й удосконалювати за допомогою практики, і що це призводить до підвищення продуктивності,

зниження стресу та поліпшення якості життя, йому варто приділяти велику увагу.

На основі проведеного ретельного аналізу сформульований перелік завдань кваліфікаційної роботи: розробка математичних моделей тайм-менеджменту завдань неформальної освіти; розробка структурної моделі програмного комплексу тайм-менеджменту; розробка функціональної моделі програмного комплексу; розробка структури бази даних системи тайм-менеджменту; розробка блок-схем алгоритмів програмних модулів системи тайм-менеджменту; розробка моделі візуального інтерфейсу користувача програмного комплексу; розробка програмних модулів візуального інтерфейсу системи автоматизації; розробка бази даних автоматизованої системи; розробка програмних модулів автоматизованої системи; розробка програмних модулів показників часової шкали тайм-менеджменту; розробка програмних модулів звітів результатів неформальної освіти**.**

Розроблена автоматизована web-орієнтована система тайм-менеджменту неформальної освіти дає змогу підвищити ефективність пошуку, обліку та аналізу занять неформальної освіти. Система забезпечує можливості статистичного аналізу кількості проведених занять, витраченого часу, тематичної специфіки конкретних дисциплін, надає багаті графічні засоби для формування та публікації аналітичних звітів.

На основі отриманих результатів можна зробити висновок, що виконана робота характеризує актуальність теми, демонструє ефективність розроблених програмних модулів.

Результати роботи можуть бути використані для організації, проведення, обліку результатів та оцінювання самостійного неформального навчання студентів за обраними спеціальностями.

1 ПРОБЛЕМА АВТОМАТИЗАЦІЇ ТАЙМ-МЕНЕДЖМЕНТУ ДЛЯ ВИРІШЕННЯ ЗАВДАНЬ НЕФОРМАЛЬНОЇ ОСВІТИ

1.1 Аналіз проблеми організації неформальної освіти для здобувачів у вищий школі

Сьогодні проблема організації неформальної освіти для здобувачів у вищий школі стає дедалі актуальнішою в сучасному освітньому середовищі. Для розкриття цієї теми необхідно насамперед провести аналіз усієї інфраструктури неформальної освіти.

Розглянемо формальну та неформальну освіту, її можливості та атрибути. Насамперед, необхідно розглядати такі аспекти неформальної освіти, як права на неформальну освіту, принципи та атрибути формальної та неформальної освіти, загальні визначення та концепції, що стосуються освіти.

Крім того, необхідно приділити увагу таким сферам, як навчальні простори та диференційовані форми навчання. Це має проводитися з метою з'ясувати, що таке навчальні простори, які питання необхідно розв'язати з погляду повсякденного освітнього процесу та як слід розуміти професійну та наукову різноманітність. Нарешті, необхідно пов'язувати розвиток системи освіти і розвиток господарського комплексу держави. Виходячи з цього погляду, можна зрозуміти, що освіта – це не тільки те, що відбувається в університетах, а й у всіх сферах життя.

Цей процес освіти є процесом "приєднання" і приналежності до певного суспільства, який відбувається через те, що ми називаємо соціалізацією. Соціалізація - це спосіб, за допомогою якого індивід стає членом групи або суспільства і починає слідувати і виконувати ті самі норми і правила, які, своєю чергою, засвоюються в процесі навчання. Заради такої професійної соціалізації реалізується вища освіта в Україні. Вища освіта дає не тільки професійні, а й соціальні компетенції. У таблиці 1.1 наведено рівні та ступені вищої освіти України [1, 2].

Таблиця 1.1 – Рівні та ступені вищої освіти України

Рівні вищої освіти	Науковий ступінь	документ про попередню освіту	ступінь вищої освіти. документ про освіту (науковий ступінь)	обсяг навчання (кредити ЄКТС)	НРК	Академічні права
Початковий рівень (короткий цикл) вищої освіти	-	атестат про повну загальну середню освіту	молодший бакалавр, диплом молодшого бакалавра	90-120 кредитів	6	доступ до програми підготовки бакалавра
Перший (бакалаврський) рівень вищої освіти	-	атестат про повну загальну середню освіту	бакалавр, диплом бакалавра	180-240 кредитів	7	доступ до програми підготовки магістра
		диплом молодшого бакалавра		обсяг може бути зменшеним за рішенням ЗВО		
Другий (магістерський) рівень вищої освіти	-	диплом бакалавра	магістр, диплом магістра	освітньо-професійна програма - 90-120 кредитів освітньо-наукова програма - 120 кредитів	8	доступ до здобуття першого наукового ступеня
		атестат про повну загальну середню освіту	магістр медичного, фармацевтичного або ветеринарного спрямування	300 - 360 кредитів		
Третій (освітньо-науковий/ освітньо-творчий) рівень вищої освіти	перший науковий	диплом магістра	доктор філософії / доктор мистецтва	30 - 60 кредитів	9	доступ до здобуття другого наукового ступеня
Науковий рівень вищої освіти	другий науковий	диплом доктора філософії	доктор наук		10	

Рівні вищої освіти докорінно впливають на результати навчання. Перший контакт із вищою професійною освітою починається з контакту з першою соціальною групою у вищій школі. Тому ця група має основне значення для процесу навчання і входження студента в професію. Через цю групу засвоюватимуться перші норми рівнів освіти та досягатимуться перші результати навчання. Отже, можна дійти висновку, що результати навчання як формальної, так і не формальної освіти залежать від формальних рівнів освіти і, також, від ступеня соціалізації у професійному середовищі. Взаємозв'язок рівнів і результатів навчання вищої освіти України наведено в таблиці 1.2 [1,2].

Таблиця 1.2 – Рівні та результати навчання вищої освіти України

Рівні вищої освіти	Результати навчання
Початковий рівень (короткий цикл) вищої освіти	здатність розв'язувати типові спеціалізовані задачі в певній галузі професійної діяльності або у процесі навчання, що передбачає застосування положень і методів відповідної науки і характеризується певною невизначеністю умов
Перший (бакалаврський) рівень вищої освіти	здатність особи вирішувати складні спеціалізовані задачі та практичні проблеми у певній галузі професійної діяльності або у процесі навчання, що передбачає застосування певних теорій та методів відповідних наук і характеризується комплексністю та невизначеністю умов
Другий (магістерський) рівень вищої освіти	здатність особи розв'язувати складні задачі і проблеми у певній галузі професійної діяльності або у процесі навчання, що передбачає проведення досліджень та/або здійснення інновацій та характеризується невизначеністю умов і вимог
Третій (освітньо-науковий/ освітньо-творчий) рівень вищої освіти	здатність особи розв'язувати комплексні проблеми в галузі професійної та/або дослідницько-інноваційної діяльності, що передбачає глибоке переосмислення наявних та створення нових цілісних знань та/або професійної практики
Науковий рівень вищої освіти	здатність особи визначати та розв'язувати соціально значущі системні проблеми у певній галузі діяльності, які є ключовими для забезпечення стійкого розвитку та вимагають створення нових системоутворювальних знань і прогресивних технологій

Неформальна освіта фокусується на питаннях, які стосуються повсякденного життя учасників. Основна мета цього виду освіти – підготувати громадян, здатних вирішувати повсякденні проблеми, розвивати навички, готуватися до роботи, колективно організовуватися, краще розуміти навколишній світ і критично ставитися до інформації, яку вони отримують. Це досягається шляхом максимального використання освітніх елементів, що вже існують в освітньому процесі, іноді змішаних з новими елементами, введеними викладачами, а також шляхом експериментів з колективними діями, часто організованими за тематичними напрямками.

Викладач неформальної освіти відіграє роль оживляючого фактора. Він повинен пробудити в учасників інтерес до контексту, в якому вони живуть, спонукаючи їх до глибшого дослідження конкретної дисципліни на індивідуальному рівні. Виходячи із загальних положень і міркувань про неформальну освіту, про її роль і про її місце в освітньому процесі, можна сформувати структуру неформальної освіти [3].

Загальна структура неформальної освіти показана на рис. 1.1.

Рисунок 1.1 – Структура неформальної освіти

Питання методології визначення та обліку результатів неформальної освіти заслуговує на окрему увагу, оскільки це одне з найслабших місць у неформальній освіті. У формальній освіті методологію зазвичай планують заздалегідь відповідно до змісту, запропонованого стандартом і планом. Видно, що неформальна освіта має свою специфіку. Тому особливу важливість має питання обліку результатів неформальної освіти. Структура організації процедури зарахування результатів неформальної освіти приведена на рис. 1.2.

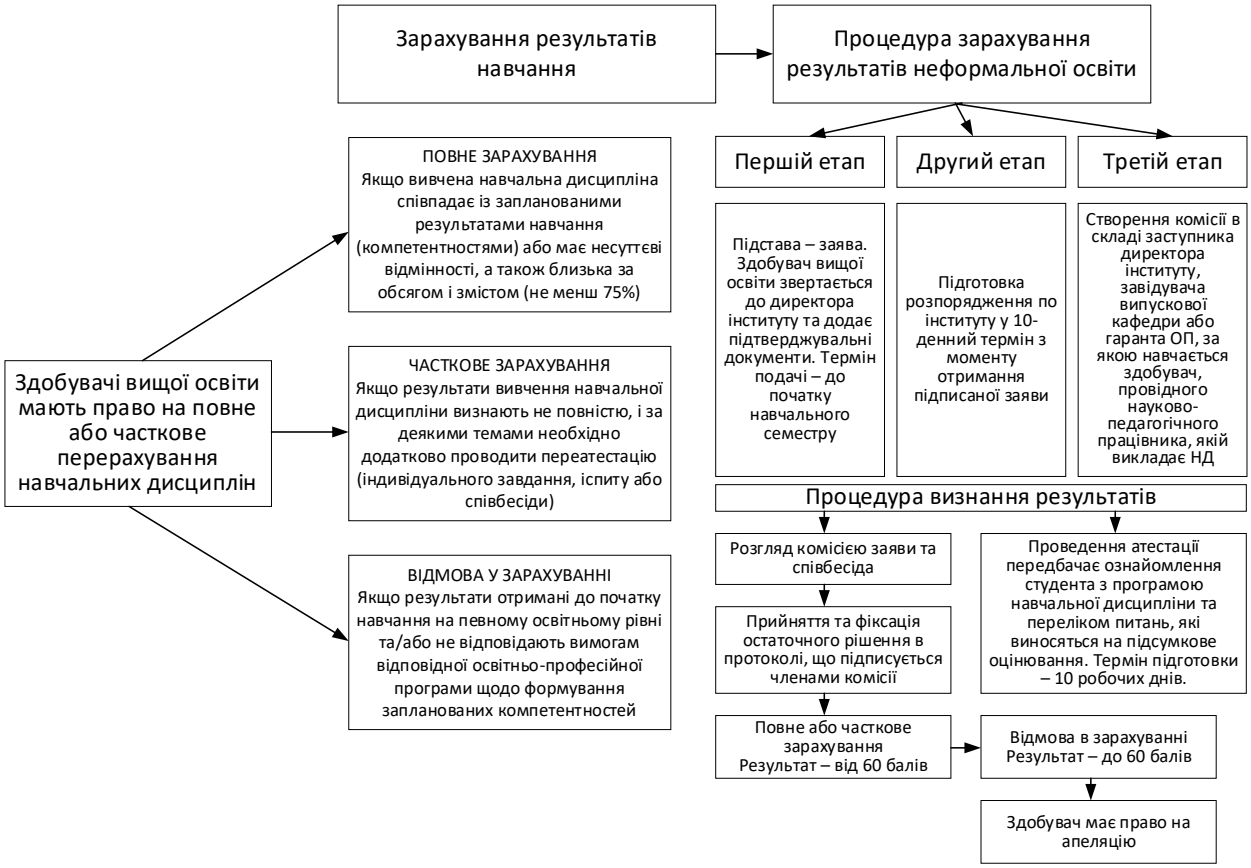


Рисунок 1.2 – Процедура зарахування результатів неформальної освіти

Неформальна освіта має як основний метод досвід і відтворення того, що відомо, відтворення досвіду відповідно до тих способів і форм, у яких він був засвоєний іншими фахівцями. У неформальній освіті методології, що використовуються в процесі навчання, засновані на компетенціях окремих людей та їхніх груп.

1.2 Аналіз загальних принципів тайм-менеджменту

Брак часу – це постійне твердження керівників різних рівнів. Як зазначено в дослідженні, проведеному 2022 року й опублікованому Revista Exame Online, 45% підприємців-учасників заявили, що їм не вистачає часу на виконання завдань. Їхня середня кількість робочих годин сягає 9,3 години на день. Однак, слід визнати, що твердження "немає часу" - це просто заява про відсутність планування. Таким чином, незважаючи на всю складність, рішення не є неможливим, все залежить від самого працівника [4, 5].

З огляду на те, що управління часом - це навичка, яку можна розвивати й удосконалювати за допомогою практики, і що це призводить до підвищення продуктивності, зниження стресу та поліпшення якості життя, йому варто приділяти велику увагу. Це дає такі переваги - планування, організація діяльності, а також знання доступних ресурсів, чи то технологічні, чи то прості методи та інструменти для щоденного використання. Основний підхід до планування часу ґрунтується на так званій матриці Ейзенхауера, яка показана на рис. 1.3 [5].

Рисунок 1.3 – Матриця Ейзенхауера для визначення пріоритетів завдань [5]

На основі матриці Ейзенхауера було запропоновано безліч моделей тайм-менеджменту. Одна з таких моделей - модель "тріада часу" [6]. У цій моделі пропонується розділити дії за трьома критеріями: важливі, термінові та другорядні. Тріада прагне спростити, модернізувати і запропонувати новий погляд на принцип Ейзенхауера і матрицю часу. В основі нової концепції лежить трійця. Три сфери разом складають увесь спосіб, яким людина використовує свій час. Головна відмінність цієї моделі від матриці полягає в тому, що тут сфери важливості та терміновості ніколи не виникають одночасно. Перетину сфер немає. Модель «тріади часу» наведено на рис. 1.4.

Рисунок 1.4 – Модель «тріада часу»

Сфера важливості охоплює всі види діяльності, які ви робите і які мають значення у вашому житті, - ті, які приносять результати в короткостроковій, середньостроковій або довгостроковій перспективі. Те, що важливо, вимагає часу, воно може чекати години, дні, тижні, місяці тощо.

Сфера терміновості охоплює всі види діяльності, на виконання яких відведено мало часу або він минув. Це дії, які виникають в останню хвилину, які неможливо передбачити, але які, як правило, викликають стрес.

Сфера обставин, своєю чергою, охоплює непотрібні завдання. Це непотрібні заняття, які витрачають час і виконуються заради зручності або тому, що вони "соціально" доречні. Це сфера дороги, яка нікуди не веде, не

приносить жодних результатів, тільки розчарування. На основі аналізу різних аспектів і моделей управління часом можна сформулювати загальну модель управління часом у тайм-менеджменті [5, 7]. Структура процесів управління часом у тайм-менеджменті показано на рис. 1.5.

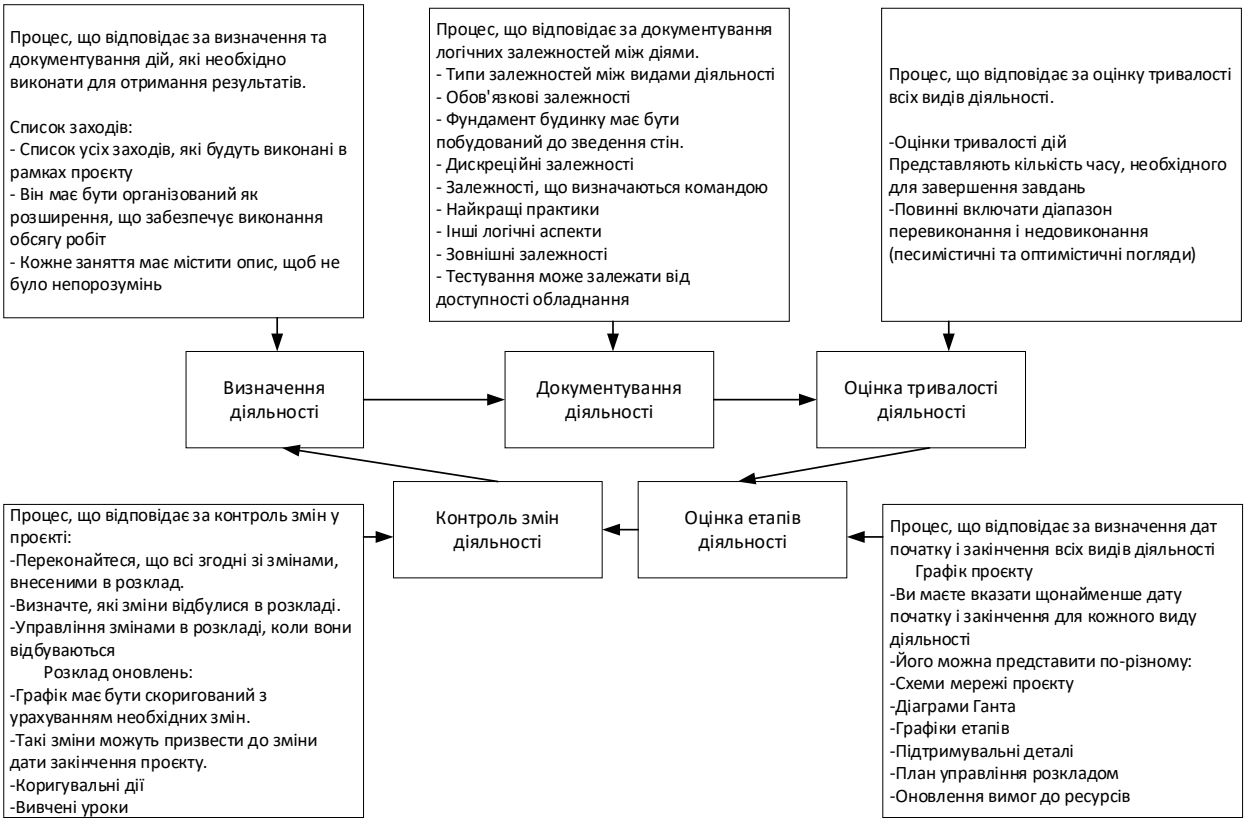


Рисунок 1.5 – Процеси управління часом у тайм-менеджменті

Процеси управління часом реалізуються на основі використання відповідних інструментів і методів. Нижче розглянемо їх.

Основні методи і інструменти тайм-менеджменту. Методи [6, 8]:

- метод мережевих діаграм;
- PDM (Precedence Diagramming Method – метод діаграми пріоритетів);
- ADM (Arrow Diagramming Method – метод стрілочної діаграми);
- методи умовної побудови діаграм: діаграми, що допускають "цикли", такі як GERT-діаграми.

Усі перелічені методи використовують графові представлення часових етапів. Загальний формат такого подання наведено на рис. 1.6.

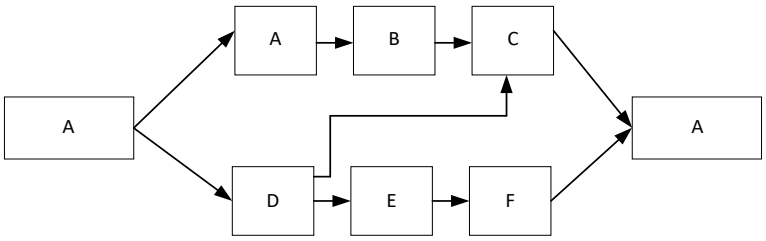


Рисунок 1.6 – Метод побудови діаграми пріоритетів (PDM)

Коротка характеристика методу PDM. Це метод побудови діаграми пріоритетів. Метод використовується для побудови мережі дій з використанням прямокутників, що представляють дії, і стрілок, що представляють пріоритет.

Функції PDM:

- відображення типів залежностей;
- ланцюжки від початку до кінця;
- початок наступного етапу залежить від закінчення попереднього етапу;
- кінець наступника події залежить від кінця прецеденту;
- початок наступника залежить від початку прецеденту;
- кінець наступника залежить від початку прецеденту;
- AON (Activity-on-Node) - відображення ознак активності вузлів.

Коротка характеристика методу ADM. Метод, який використовується для побудови мережі дій з використанням прямокутників і стрілок, що представляють дії.

Функції ADM:

- використовує тільки залежності "від завершення до початку";
- також називається AOA (активність за стрілкою);
- методи оцінки PERT і CPM можуть бути представлені тільки з використанням цього типу діаграми.

Розглянемо приклад діаграми ADM, який наочно показує реалізацію основних методів побудови діаграми. Приклад проілюстровано на рис. 1.7. Також відображено фіктивну активність.

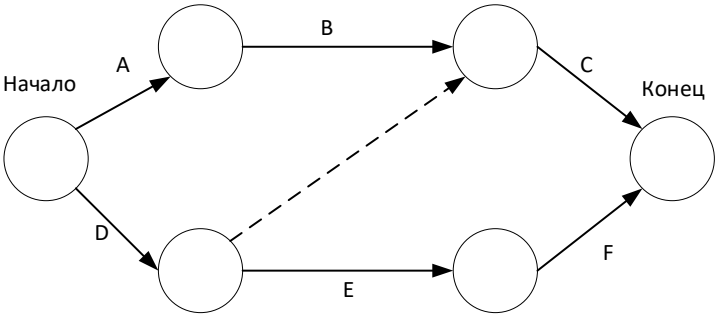


Рисунок 1.7 – Приклад діаграми ADM.

На рис. 1.7. пунктирна лінія являє собою фіктивну активність. Очевидно, що стрілочні діаграми активностей дають змогу моделювати та відстежувати події під час їхнього планування на наступні етапи часу.

1.3 Аналіз підходів до організації тайм-менеджменту для сфери неформальної освіти

Неформальна освіта – це галузь знань, що перебуває у стадії розробки. Тому питання, пов'язані з організацією процесу неформальної освіти, і, особливо, організацією доцільного розподілу часу особливо важливі. Розглянемо підходи до організації неформальної освіти для виявлення характерних механізмів тайм-менеджменту.

Практики неформальної освіти, як правило, розвиваються за межами стін вищої школи – у громадських організаціях, рухах і навчальних програмах. Ці практики лежать в основі діяльності освітніх рухів та інклюзивних програм, як у сфері точних наук, так і в області мистецтва, освіти та культури. Всесвітня декларація про освіту для всіх у період її розроблення вказувала на численні ситуації, які необхідно подолати у зв'язку із забезпеченням права на неформальну освіту для всіх людей. Серед них спостерігається високий рівень студентів, які не мають змоги відвідувати університет, а також велика кількість немобільних людей, переважну більшість з яких становлять жінки. Крім того, є велика кількість студентів, які не в змозі завершити навчання або

не в змозі ефективно засвоїти пропонований матеріал. Майже через 30 років після ухвалення декларації багато з окреслених проблем зберігаються, і знову виникають ситуації, що перешкоджають задоволенню базових потреб у навчанні.

Як формальна, так і неформальна освіта необхідні й незамінні для розуміння всеосяжної та комплексної освіти. Беручи до уваги загальну концепцію освіти та форми формальної і неформальної освіти, спостерігаються наступні випадки:

— у сучасному суспільстві, у міру того як соціальні відносини стають дедалі складнішими, освіта стає важливою до такої міри, що освітнє співтовариство не втрачає можливості відвести неформальній освіті обов'язкову роль у своїх університетах;

— велике питання неформальної освіти, якщо можна так висловитися, - це форма освіти, в якій простори для процесу викладання-навчання перебувають за межами освітнього закладу;

— неформальну освіту структуровано на основі набору принципів і правил, що є частиною правової системи та національних керівних принципів, які гарантують (або принаймні мають гарантувати) базовий набір компетенцій та їхній зміст кожному студенту;

— неформальна освіта являє собою відкритий процес. Вона може набувати різних форм і відбуватися в соціальній, культурній та психологічній сферах особистості;

— неформальна освіта - це прямий і короткий процес. Її можна визначити як «соціальну практику», мета якої – розвинути в людській особистості все, чого можна навчитися з наявних знань, формуючи суб'єктів, які поводяться і діють відповідно до потреб і запитів свого суспільства.

Візьмемо популярні підготовчі курси до вступу в університет як приклад неформальної освіти. Ми можемо підтвердити, що в цих проєктах беруть участь професійні викладачі, викладачі-неспеціалісти, активісти громадських рухів, молоді студенти університетів та інші учасники освітнього процесу.

Однак не варто забувати, що будь-яке узагальнення пов'язане з певним ризиком, тому ми не можемо вважати ці категорії фахівців єдиними, хто працює в даному типі освіти.

Сфера неформальної освіти надто широка, щоб робити узагальнення щодо однозначних підходів до організації тайм-менеджменту для сфери неформальної освіти. Відокремлення одного типу професіоналів у сфері неформальної освіти є великою помилкою, особливо тому, що ми не можемо забувати, що існують так звані проєкти неформальної освіти, практика яких потребує записів та інших педагогічних і бюрократичних документів.

Тому іноді виникає плутанина між неформальною та інформальною освітою. Ми розглядаємо неформальну освіту, як спонтанне навчання, оскільки воно відбувається в повсякденному житті, в той час як неформальне навчання планується та продумується координатором, викладачем або людиною, що працює з молодими студентами, яка також здійснює свою підтримку протягом усього процесу навчання.

Педагогічні працівники вже змогли визначити можливості побудови педагогічної освіти в неформальних освітніх просторах і сформулювати деякі проблеми, які необхідно обговорити задля їх подолання. Як потенційні можливості можна зазначити, що неформальний освітній простір дає змогу: встановлювати міжособистісні стосунки та навчатися на відмінностях; розуміти роль інших людей у суспільстві; критично осмислювати педагогічну підготовку, проводячи самооцінку; будувати діалог між науковими й традиційними знаннями; розуміти освіту як процес, який виходить за межі стін університету, відбувається в різних просторах і включає різні типи знань. Щодо труднощів, які необхідно подолати, то до них належать: труднощі в колективній роботі; створення інструменту для оцінювання виконаної роботи; визначення формальної та неформальної освіти; розуміння прогалин, які існують у цих просторах, оскільки вони вважають неформальну освіту "кращою" та не пов'язаною з простором формальної освіти. Результати порівняльного аналізу неформальної освіти приведені в табл. 1.3.

Таблиця 1.3 – Порівняльний аналіз неформальної освіти

	Формальна освіта: Академічне навчання	Формальна освіта: Професійно-технічна освіта	Неформальна освіта
Впровад-жені методи навчання	Курси, де існує вертикальний зв'язок між носієм знань і студентом.	Вертикальний зв'язок, як в академічному навчанні. Курси можуть чергуватися з практичною частиною. Під час практики можна користуватися інструкціями.	Інтерактивні стосун-ки між учнями та їхнім оточенням «Навчання через дію». Широко використовується навчання за прин-ципом "рівний-рівному" та настав-ництво.
Зміст	Здебільшого загальні. Визначається органами управління освітою.	Вони прагнуть отримати операторську кваліфікацію. Визначаються органами освіти.	На вибір студента. Немає визначення, окрім набуття конкретного досвіду.
Сертифіка ція	Надається в кінці курсу і залежить від успіху в оцінюванні знань. Встановлюється відповідно до критеріїв, визначених органами управління освітою.	Надається наприкінці навчання та залежить від успішного оцінювання знань і практики. Встановлюється відповідно до критеріїв, визначених органами управління освітою.	Поки що немає сертифікату, але його можна розглядати для навчання в уні-верситетах.
Тривалість	Зазвичай: від 6 до 18 років (початкова та середня освіта); старше 18 років (близько 10 років навчання - університет).	Зазвичай вона коротка: в деяких країнах починається з 14 років і триває 4 роки під час середньої освіти; 2-3 роки після навчання в університеті.	Безперервне навчання.
Сильні сторони	Обов'язкова для всіх (зазвичай до 16 років) з метою надання базових знань. Майже безкош-товна в державному секторі. Дає право на сертифікацію через академічні дипломи.	Короткострокові, що надають оперативну кваліфікацію, яка може бути використана безпосередньо на ринку праці. Засвідчується дипломами про професійно-технічну освіту.	Доступний кожному в будь-який час життя. "Другий шанс " для молодих людей з обмеженими можливостями.
Слабкі сторони	Вони можуть залиша-тися загальними, і необхідно поглиблювати свої знання за допомо-гою подальшого навчання або тренінгів. Підхо-дить не для всіх. Не визнаються в усій Європі (труднощі з передачею вартості диплома за кордон).	Не обирається молодими людьми, а нав'язується їм "за замовчуванням" під час навчання. Потреби ринку праці можуть змінюватися, роблячи кваліфікацію непотрібною. Вона не визнається в усій Європі (труднощі з перенесенням диплома за кордон).	Вона не має офіційного визнання.

Виходячи з результатів аналізу методології неформальної освіти, можна сформулювати загальні принципи планування або тайм-менеджменту неформальної освіти. Перелічимо деякі характеристики тайм-менеджменту, яких неформальна освіта може досягти з погляду цілей, у планованих процесах колективних групових дій:

— вивчення відмінностей. Студент вчиться взаємодіяти з іншими. Заохочується взаємна повага;

— адаптація групи до різних культур, визнання особистості та ролі інших, робота над соціалізацією;

— формування колективної ідентичності групи;

— встановлення етичних правил щодо соціально прийнятної поведінки.

Сформулюємо вимоги до неформальній освіті з точки зору організації тайм-менеджменту:

— спеціальна підготовка викладачів, заснована на визначенні їхньої ролі та видів діяльності, які необхідно виконувати;

— чіткіше визначення функцій і цілей неформальної освіти;

— систематизація методик, використовуваних у повсякденній роботі;

— розроблення методик, що дають змогу здійснювати моніторинг виконуваних робіт;

— розробка методичного інструментарію оцінки та аналізу виконаної роботи;

— побудова методологій, які можуть забезпечити моніторинг роботи випускників, які взяли участь у програмах неформальної освіти;

— створення методологій і показників для вивчення й аналізу роботи неформальної освіти в несистематизованих галузях. Навчання, здійснюване за допомогою вольових актів одержувача, наприклад, навчання через Інтернет, навчання музики, гри на музичному інструменті тощо.;

— фіксація та документування форм неформальної освіти в самостійному навчанні громадян (переважно молоді).

1.4 Аналіз існуючих програмних систем підтримки тайм-менеджменту

Експерти з управління часом рекомендують використовувати персональний інструмент планування для підвищення продуктивності. Персональні інструменти планування включають у себе планувальники, календарі, додатки для телефону, настінні діаграми, картки, кишенькові щоденники і блокноти. Записування завдань, розкладів і речей для запам'ятовування може звільнити розум, щоб зосередитися на ваших пріоритетах. Аудіали замість цього можуть віддати перевагу диктувати свої думки. Головне - знайти один інструмент планування, який підходить саме вам, і використовувати його постійно.

Рекомендації під час використання інструменту планування наступні:

— завжди записуйте інформацію на самому інструменті. Записи в іншому місці, які потрібно буде перенести пізніше, неефективні та забирають більше часу;

— щодня переглядайте свій інструмент планування;

— ведіть список своїх пріоритетів у своєму інструменті планування і часто до нього звертайтеся;

— синхронізуйте інструменти планування. Якщо у вас є кілька інструментів, переконайтеся, що ваш телефон, комп'ютер і паперові інструменти планування збігаються;

— зберігайте резервну систему.

Додатки, що використовуються для організації та оптимізації тайм-менеджменту, можуть бути чудовими інструментами планування і зазвичай потрапляють в одну з таких категорій:

— тайм-трекери - усвідомлюють, як ви витрачаєте свій час;

— заощаджувачі часу - підвищують продуктивність і позбуваються звичок, які витрачають час даремно;

— менеджери завдань - розставляють пріоритети й організовують завдання для поліпшення управління часом;

— розробники звичок - створюють корисні звички для заохочення управління часом.

Виходячи зі сформульованих категорій, наведемо характерні приклади існуючих програмних систем підтримки тайм-менеджменту.

Outlook - це популярна служба веб-пошти, що пропонує такі функції, як конфіденційність інформації, поліпшена фільтрація спаму і можливість перегляду і редагування файлів пакета Office безпосередньо в електронній пошті. Інтуїтивно зрозумілий інтерфейс робить його простим і зрозумілим, даючи змогу користувачам налаштовувати такі аспекти, як кольори та шрифти. Крім електронної пошти, в ньому є такі функції, як календар, нагадування і список справ. На рис. 1.8 наведена програма MS Outlook.

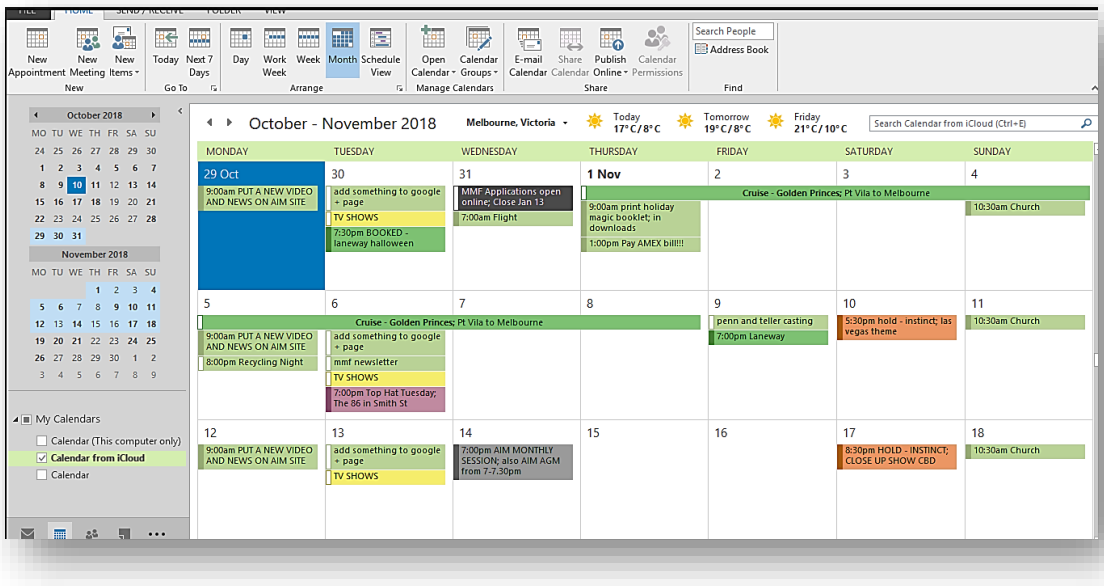


Рисунок 1.8 – Програма підтримки тайм-менеджменту MS Outlook

Розглянемо програму Any.do. Процес планування діяльності, запропонований у цій моделі, може бути адаптований до планувальника або повного щоденника (паперового або програмного), що містить усі дні року з розбивкою за місяцями та тижнями. Тижневі плани розташовуються на початку кожного тижня, а кожен день складається зі щоденного плану (списку

і щоденника). Таким чином, можна буде планувати деякі справи та зустрічі безпосередньо на відповідні тижні та дні, які міститимуть відповідні попередні списки завдань, що будуть доповнені та заплановані в момент планування. На рис. 1.9 приведена програма підтримки тайм-менеджменту Any.do

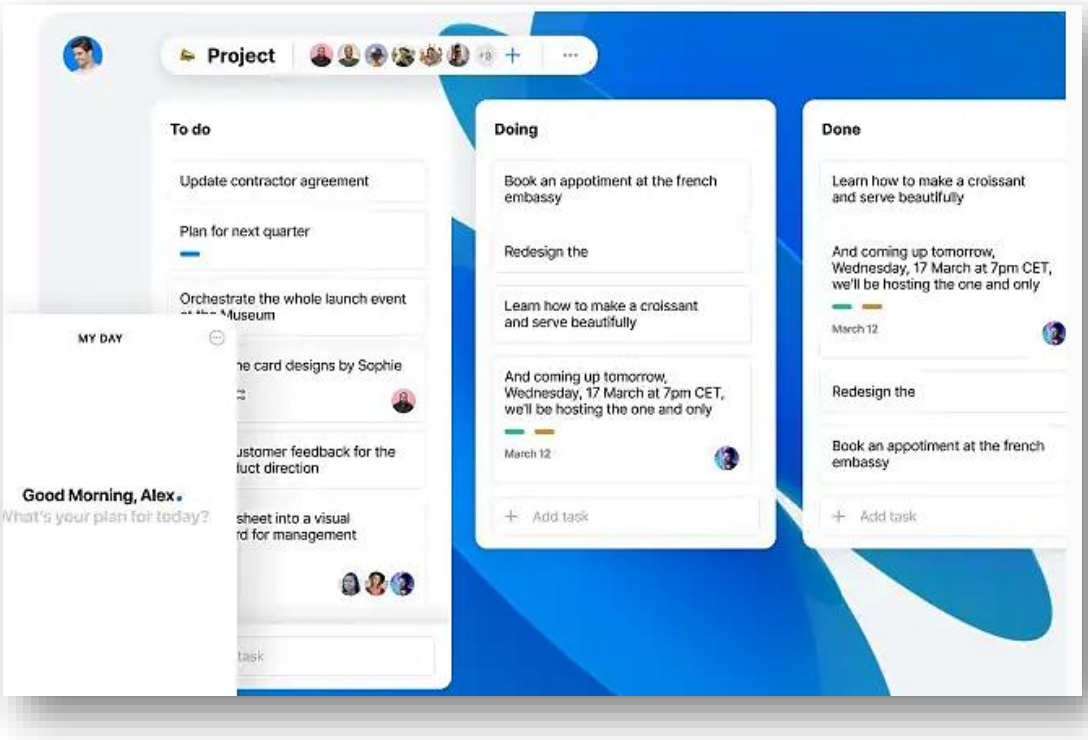


Рисунок 1.9 – Програма підтримки тайм-менеджменту Any.do

Програма Trello. Інструмент Trello використовує трикомпонентний метод управління: завершені дії, поточні дії та майбутні дії. Велика кількість людей у світі використовує цей інструмент. Після навчання використанню Trello дуже легко складається розклад, щоб узгодити терміни виконання заходів у робочих групах. Зазвичай формуються групи, і призначається керівник, який разом із викладачами керує інструментом. Щотижня в Trello оновлюється контрольний список завдань; звіти про проведені зустрічі; звіти про перебіг виконання кожного заходу, які містять фотографії, посилання та покажчики з мітками, що дає змогу всім учасникам стежити за розвитком проєктів. Ілюстрацію програми Trello наведено на рис. 1.10.

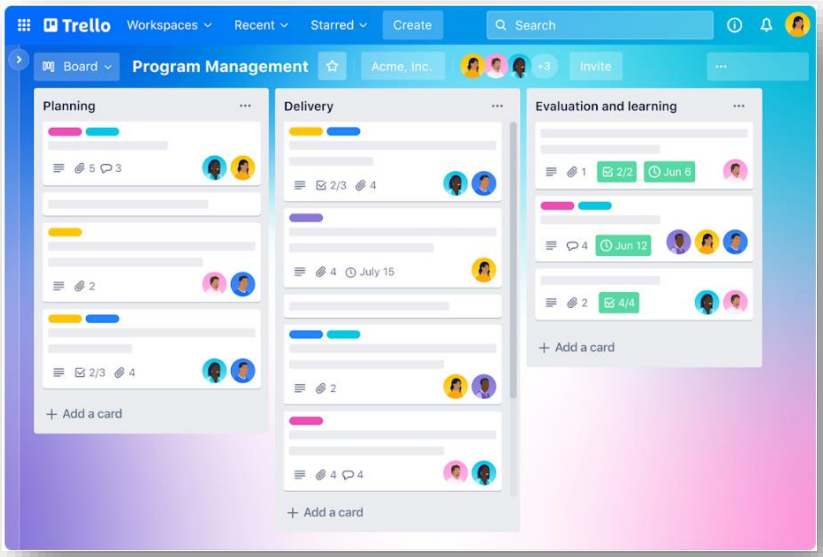


Рисунок 1.10 – Програма підтримки тайм-менеджменту Trello

Програма Google Календар. Google Календар – це онлайновий щоденник і календарний сервіс, доступний через веб-інтерфейс. У ньому ви можете додавати і контролювати події, зустрічі, ділитися своїм розкладом з іншими людьми, інтегрувати різні публічні щоденники, серед інших можливостей. Ілюстрацію програми Google Календар наведено на рис. 1.11.

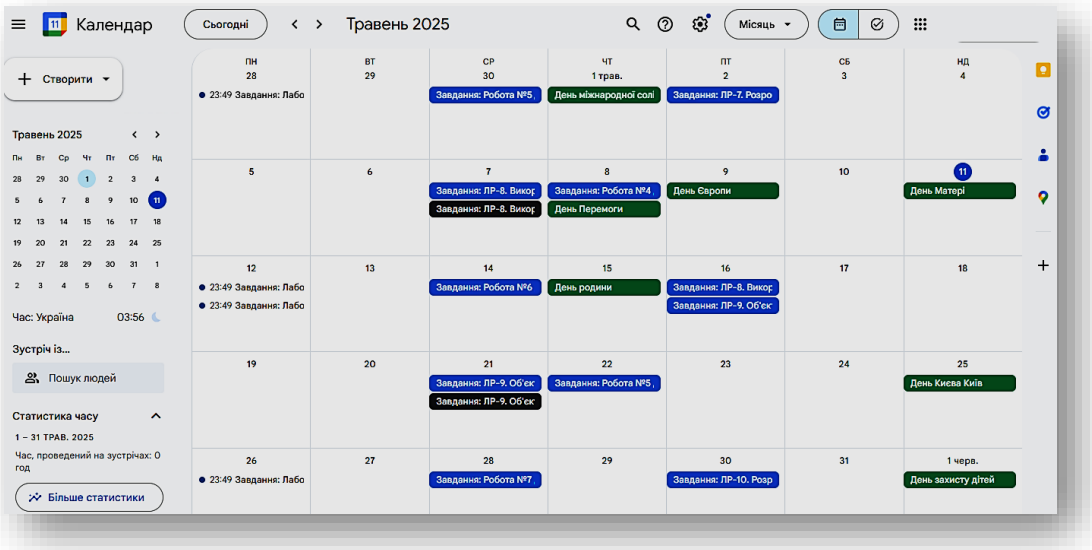


Рисунок 1.11 – Програма підтримки тайм-менеджменту Google Календарь

Програма Todoist. Todoist показує потрібні завдання в потрібний час, щоб ви завжди знали, на чому зосередитися наступного разу. Забезпечує розподіл і виконання повсякденних завдань у спільних проектах - від ділових починань до списків покупок. Підтримує створення власних уявлень завдань відповідно до вашого унікального стилю та робочого процесу. Спрощує робочий процес, пов'язавши Todoist з електронною поштою, календарем і файлами. Ілюстрацію програми Todoist наведено на рис. 1.12.

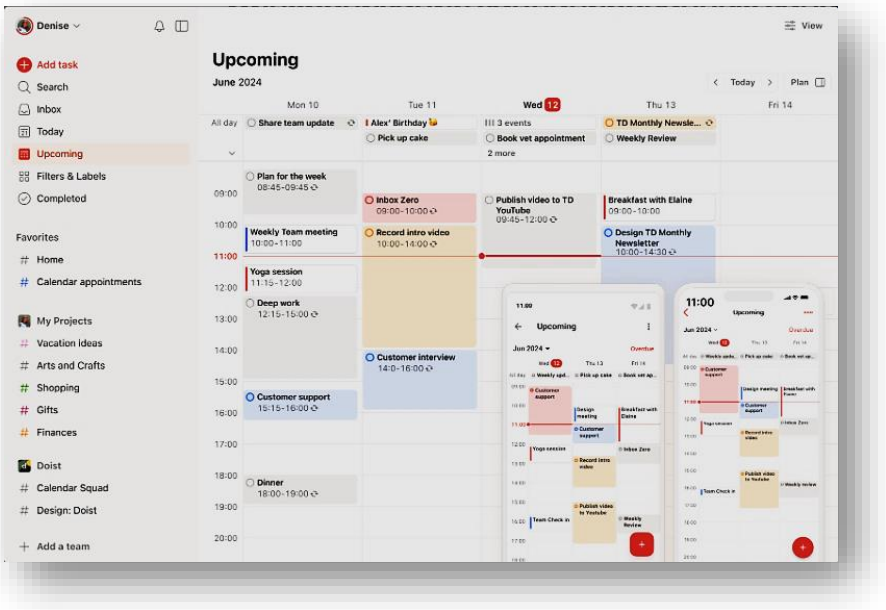


Рисунок 1.12 – Програма підтримки тайм-менеджменту Todoist

Програма TickTick. TickTick - це ефективний менеджер списку справ і поточних завдань, який може підтримувати робочий процес і допомогти встигати реалізовувати всі плани. TickTick є кросплатформним. Додатки TickTick дають змогу контролювати завдання на всіх користувацьких пристроях і також в Internet. Крім того, програма дає змогу зберігати резервні копії та синхронізувати плани на сайті TickTick.com, де користувач може організувати свій розклад і графік роботи. TickTick забезпечує доступ до більшої кількості видів календаря. Встановлює дати початку та закінчення завдань. Можна навіть підписатися на календарі сторонніх виробників. Ілюстрацію програми TickTick наведено на рис. 1.13.

Рисунок 1.13 – Програма підтримки тайм-менеджменту TickTick

Программа – визуальный тайм-менеджер Sectograph. Sectograph - это визуальный ежедневник, который меняет представление о событиях в календаре. Ілюстрацію програми Sectograph наведено на рис. 1.14.

Рисунок 1.14 – Програма підтримки тайм-менеджменту Sectograph

Завдяки своєму унікальному підходу Sectograph відображає щоденник на круговій діаграмі у вигляді годинника часу. Таке візуальне представлення дає змогу більш ефективно і раціонально розпоряджатися своїм часом. Застосунок працює як аналоговий віджет годинника, автоматично синхронізуючись із календарем і накладаючи події на часову діаграму. Кожна подія відображається у відповідному часовому інтервалі, забезпечуючи чітке та інтуїтивно зрозуміле уявлення про ваш день. Також відображається час простою між подіями, а також час від годинникової стрілки до початку кожної події. Крім того, події відображаються на кілька годин вперед, що дає змогу бути готовим до них.

1.5 Завдання розробки системи автоматизації тайм-менеджменту завдань неформальної освіти

У попередніх розділах було проведено детальний аналіз професійної галузі тайм-менеджменту. Особливу увагу було приділено автоматизації тайм-менеджменту у сфері неформальної освіти у вищій школі. Нині питанням неформальної освіти приділяється особливо багато уваги. І її роль весь час зростає.

Розглянуто формальну та неформальну освіту, її можливості та атрибути. Насамперед, необхідно розглядати такі аспекти неформальної освіти, як права на неформальну освіту, принципи та атрибути формальної та неформальної освіти, загальні визначення та концепції, що стосуються освіти.

Крім того, необхідно приділити увагу таким сферам, як навчальні простори та диференційовані форми навчання. Необхідно пов'язувати розвиток системи освіти і розвиток господарського комплексу держави. Виходячи з цього погляду, можна зрозуміти, що неформальна освіта – це не тільки те, що відбувається в університетах, а й у всіх сферах життя.

Був проведений аналіз загальних принципів тайм-менеджменту. З огляду на те, що управління часом – це навичка, яку можна розвивати й удосконалювати за допомогою практики, і що це призводить до підвищення продуктивності, зниження стресу та поліпшення якості життя, йому варто приділяти велику увагу. Це дає такі переваги – планування, організація діяльності, а також знання доступних ресурсів, чи то технологічні, чи то прості методи та інструменти для щоденного використання. Основний підхід до планування часу ґрунтується на так званій матриці Ейзенхауера.

Особливий інтерес становить питання тайм-менеджменту для сфери неформальної освіти. Тому був проведений аналіз підходів до організації тайм-менеджменту для сфери неформальної освіти. Неформальна освіта – це галузь знань, що перебуває у стадії розробки. Тому питання, пов'язані з організацією процесу неформальної освіти, і, особливо, організацією доцільного розподілу часу особливо важливі.

Сучасний тайм-менеджмент у сфері неформальної освіти має спиратися на використання автоматизованих програмних комплексів управління робочим часом. Тому було проведено аналіз існуючих програмних систем підтримки тайм-менеджменту. Експерти з управління часом рекомендують використовувати персональний інструмент планування для підвищення продуктивності. Персональні інструменти планування включають у себе планувальники, календарі, додатки для телефону, настінні діаграми, картки, кишенькові щоденники і блокноти. Записування завдань, розкладів і речей для запам'ятовування може звільнити розум, щоб зосередитися на ваших пріоритетах.

На основі проведеного ретельного аналізу можна сформулювати перелік завдань кваліфікаційної роботи:

— розробка математичних моделей тайм-менеджменту завдань неформальної освіти;

— розробка структурної моделі програмного комплексу тайм-менеджменту;

— розробка функціональної моделі програмного комплексу;

— розробка структури бази даних системи тайм-менеджменту;

— розробка блок-схем алгоритмів програмних модулів системи тайм-менеджменту;

— розробка моделі візуального інтерфейсу користувача програмного комплексу;

— розробка програмних модулів візуального інтерфейсу системи автоматизації;

— розробка бази даних автоматизованої системи;

— розробка програмних модулів автоматизованої системи;

— розробка програмних модулів показників часової шкали тайм-менеджменту;

— розробка програмних модулів звітів результатів неформальної освіти.

Розв'язання перелічених завдань дасть змогу розробити кросплатформну систему автоматизації тайм-менеджменту процесу неформальної освіти, а також аналізувати отримані результати засобами узагальнення та візуалізації.

2 РОЗРОБКА МОДЕЛЕЙ І АЛГОРИТМІВ ПРОГРАМНОГО КОМПЛЕКСУ ТАЙМ-МЕНЕДЖМЕНТУ ЗАВДАНЬ НЕФОРМАЛЬНОЇ ОСВІТИ

2.1 Розробка математичних моделей тайм-менеджменту завдань неформальної освіти

Як відомо, зусилля виправдовуються тільки кінцевими результатами, які вони дають. Тому, коли ми не встановлюємо собі крайні терміни, ми використовуємо весь наявний у нас час для виконання завдання, а не той час, який фактично потрібен для його виконання. Згідно з цим законом, існує зв'язок між часом, витраченим на виконання завдання, та отриманими результатами, що представлений на графіку, рис. 2.1.

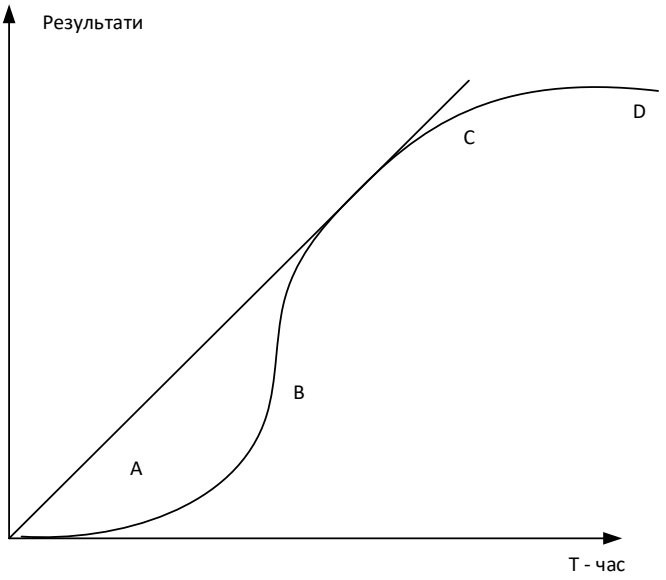


Рисунок 2.1 – Залежність результатів роботи від витраченого часу

На графіку відзначено такі зони:

— A – Зона недосконалості;

— B – Зона ефективності;

— C – Зона перфекціонізму;

— D – Зона відходів.

За час T′ виходить результат R′. Максимальна ефективність становить R′/T′. Кожна точка перегину визначає зону використання часу. Математичне моделювання для оцінки часу критичного шляху проєкту зазвичай проводять із використанням методу PERT (Program Evaluation and Review Technique - методологія аналізу та оцінки програм) [9, 10]. Такі формули PERT, крім того, що є найосновнішими, також є найбільш широко використовуваними в усьому світі:

$$\mu_e = \frac{a + 4m + b}{6}, \tag{2.1}$$

$$\sigma_e = \frac{b - a}{6}, \tag{2.2}$$

де μ_e и σ_e — це відповідно розрахункове середнє значення і розрахункове стандартне відхилення завдання за час T, а a, m и b — це відповідно експертна оцінка "оптимістичного", "найімовірнішого" і "песимістичного" висновку роботи.

Однак застосування (2.1) і (2.2) не обмежується проектами мереж PERT, оскільки їхню значущість для оцінки невизначених величин у будь-якій стохастичній моделі давно визнано. Визначимо послідовність дій для розроблення моделі на основі ідеології PERT. У табл. 2.1 наведено ідентифікацію етапів діяльності для її розгляду з точки зору PERT.

Таблиця 2.1 – Ідентифікація етапів діяльності

x – Ідентифікація діяльності d – Тривалість	
Ранній старт (ES)	Ранній фініш (EF)
Пізній старт (LS)	Пізній фініш (LF)

Виходячи з ідентифікаційної таблиці 2.1, можна зробити розрахунок показників ES і EF для тривалості процесу 18 тижнів. Приклад схеми розрахунку наведено на рис. 2.2.

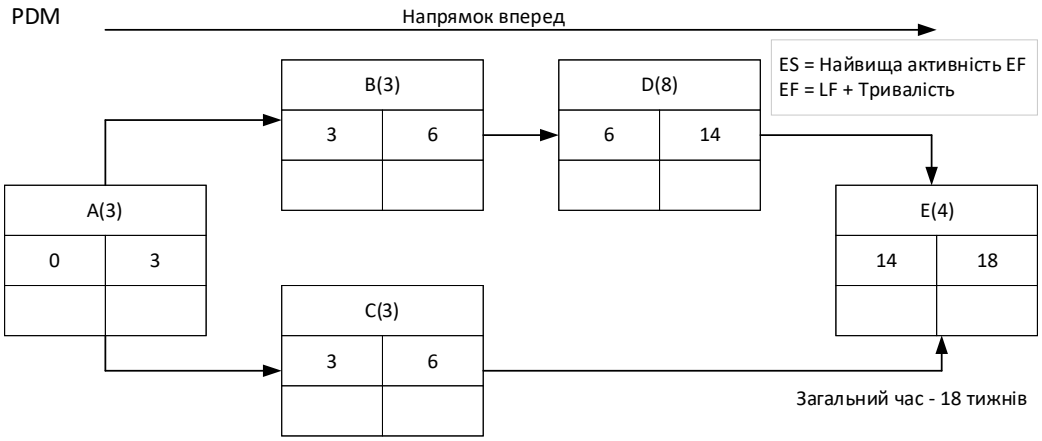


Рисунок 2.2 – Розрахунок показників ES і EF

Аналогічно можна зробити розрахунок показників LS і LF. Приклад схеми розрахунку наведено на рис. 2.3.

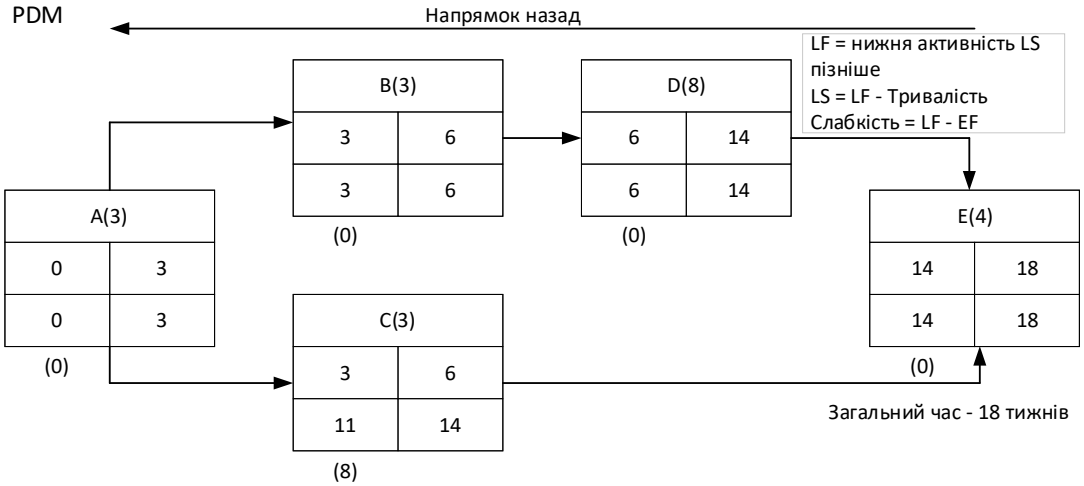


Рисунок 2.3 – Розрахунок показників LS і LF.

Крім того, можна оцінити етапи виконання робіт при так званих "слабких" ("Slack") рішеннях. Приклад схеми визначення шляху з роботами зі значенням «Slack», рівним нулю, наведено на рис. 2.4.

Для всіх перелічених видів діаграм визначено загальний підхід для обчислення параметрів: Розробка розкладу → Інструменти і методи →

Математичний аналіз → Формули методу PERT. Найбільш загальні формули розрахунків наведено в табл. 2.2.

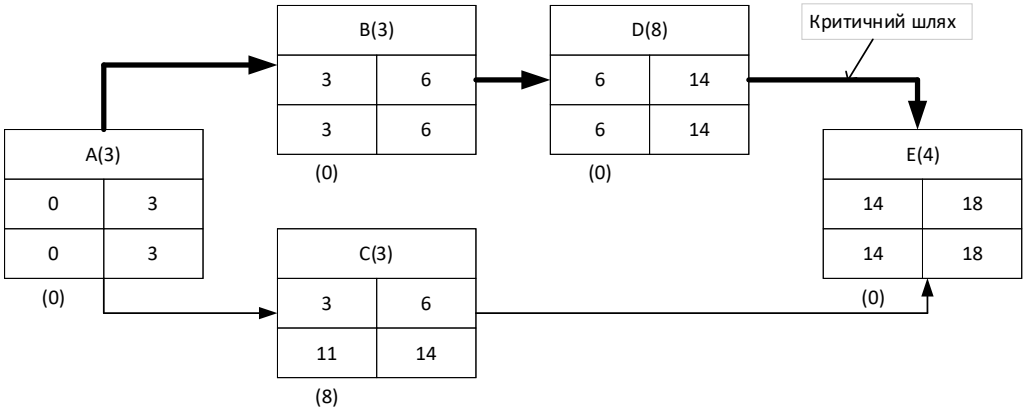


Рисунок 2.4 – Визначення шляху з роботами зі значенням «Slack», рівним нулю

Таблиця 2.2 – Формули розрахунку параметрів діаграми PERT

Очікуване значення	Стандартне відхилення (SD)	Відхилення (V)
$EV = \frac{P + 4M + O}{6}$	$SD = \pm\frac{P - 6}{6}$	$V = \left(\frac{P - O}{6}\right)^2$
Являє собою оцінку для значення, яке буде використано	Стандартне відхилення для плюс і мінус	Зміна результату

В таблиці 2.2 прийняти наступні позначення:

— P – Песимістичне значення;

— O – Оптимістичне значення;

— M – Найімовірніше значення.

На основі розрахунків за формулами, які наведені в таблиці 2.2 можна побудувати графік розподілу ймовірності тривалості діяльності проекту загалом. Такий графік наведено на рис. 2.5. Як видно з графіка, він має деякі характерні точки - оптимістичну, песимістичну, точку дистрибуції, точку найімовірнішої тривалості проєкту.

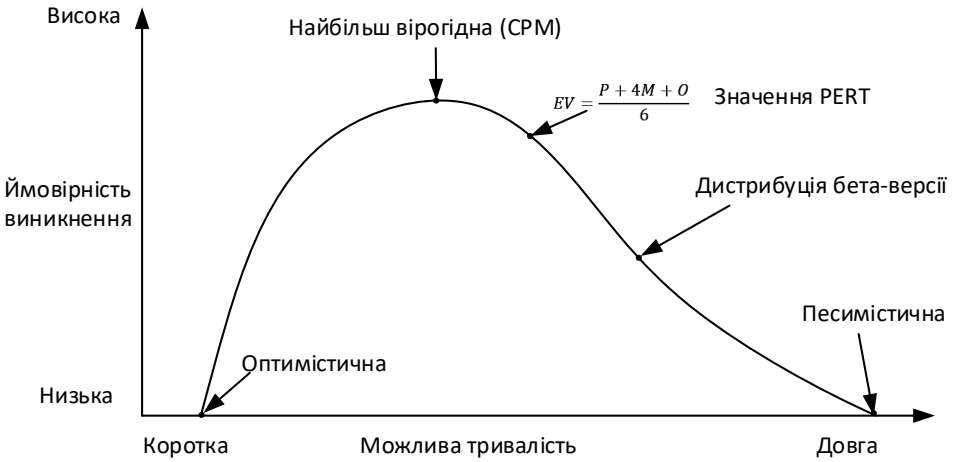


Рисунок 2.5 – Графік розподілу ймовірності тривалості діяльності проекту

Надалі застосовуються методи експериментального планування та аналізу. Після того, як фактори та їхні рівні буде встановлено, проводиться формулювання гіпотез для перевірки:

— HA: фактор року впровадження не впливає на середній час впровадження;

— HP: розмір компанії-клієнта не впливає на середні терміни впровадження;

— HI: взаємодія між факторами відсутня;

— H1A: фактор року впровадження впливає на середній час впровадження;

— H1P: розмір компанії-клієнта впливає на середній час впровадження;

— HI: між тестованими факторами існує взаємодія.

Іншими словами, перевіряється така модель:

$$Y_{ij} = \mu + \tau_i + \beta_i + \tau_i\beta_i + \xi_{ij}, \quad (2.3)$$

де Y_{ij} являє собою час впровадження, коли фактор року впровадження на рівні i, а фактор розміру компанії-клієнта перебуває на рівні j; m — загальне середнє значення, t_i — вплив i-го рівня фактора року впровадження, b_j — вплив j-го рівня фактора; t_ib_j — вплив взаємодії між i-м рівнем фактора року впровадження і j-м рівнем фактора, а x_{ij} — звичайна випадкова помилка.

2.2 Розробка структурної моделі програмного комплексу тайм-менеджменту

Один із найкращих засобів управління часом – працювати з автоматизованими системами організації часу, в організованому робочому просторі. Необхідно повністю організувати свій робочий простір для роботи. Таким чином, ключем до успіху є використання інтегрованих програмних засобів для тайм-менеджменту. У ефективних керівників завжди є засоби автоматизації обліку та управління їхнім часом. Тому люди можуть краще зосередитися і зробити більше, якісніше і за коротший термін.

У кваліфікаційній роботі було визначено взаємозв'язок завдань загального підходу до формування структурної моделі програмного проекту, який показаний на рис. 2.6.

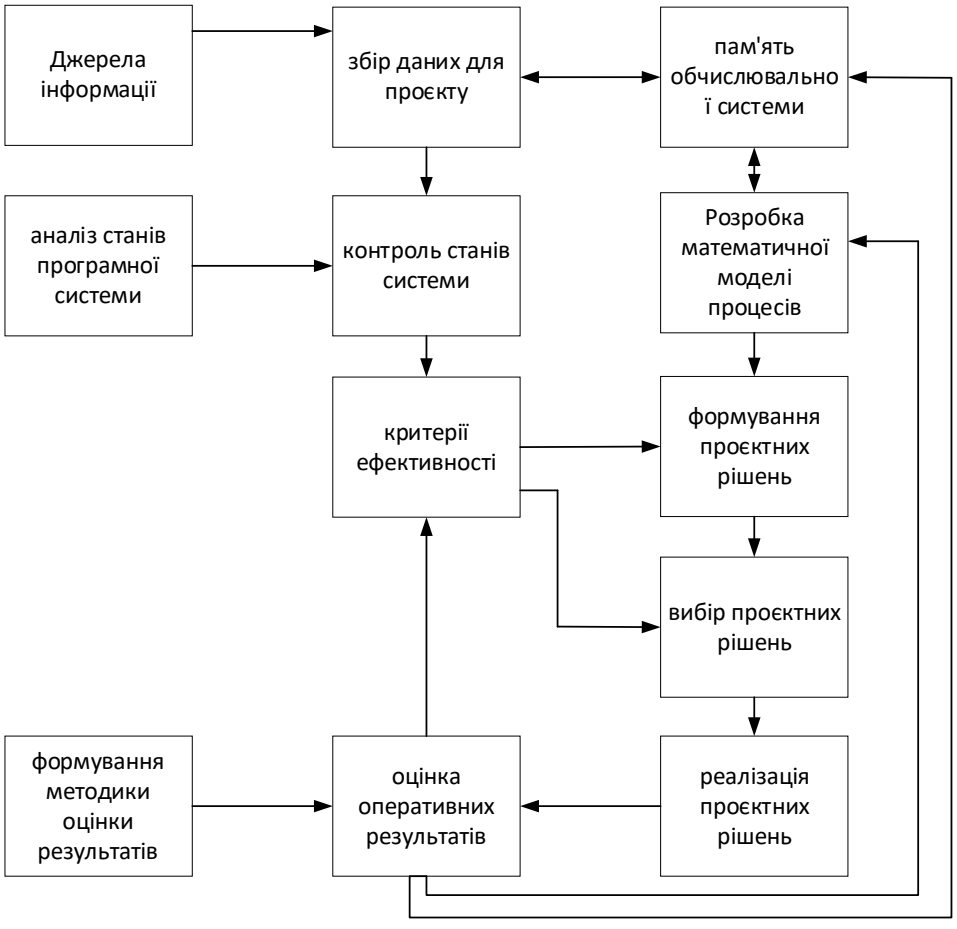


Рисунок 2.6 – Взаємозв'язок завдань загального підходу до формування структурної моделі програмного проекту

Спираючись на сформовану схему завдань загального підходу до формування структурної моделі програмного проекту, можна розробити узагальнену структуру програмного комплексу для тайм-менеджменту процесу неформальної освіти. Узагальнена структурна модель програмного комплексу показана на рис. 2.7.

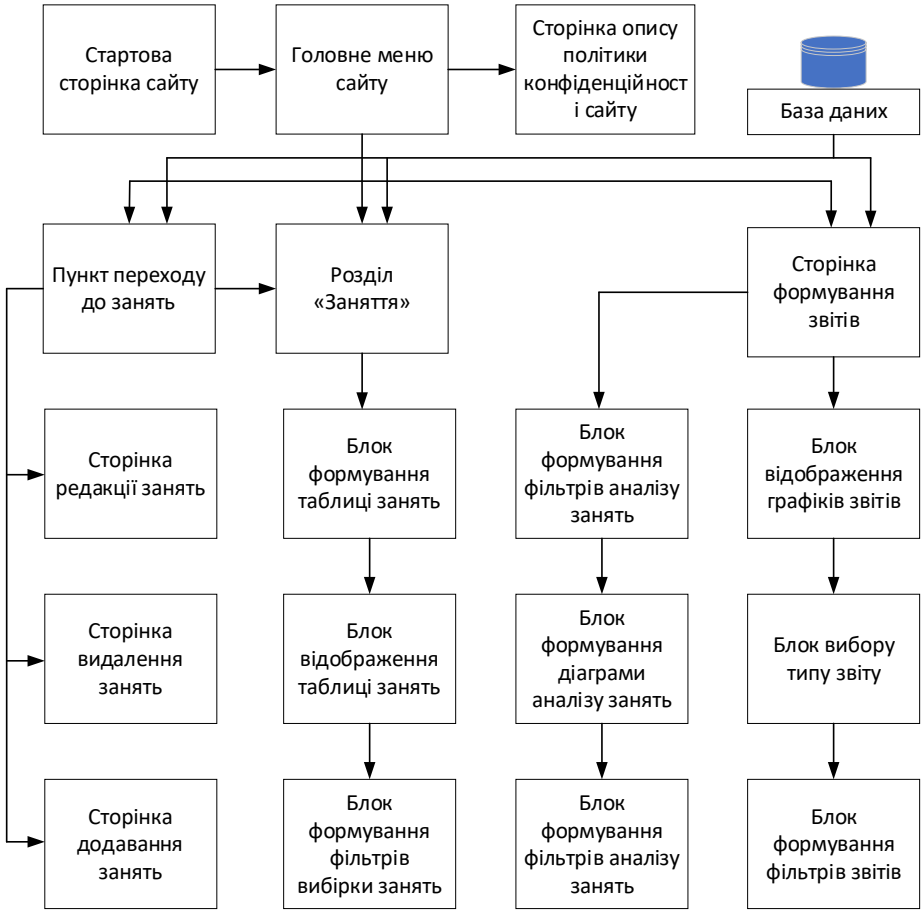


Рисунок 2.7 – Узагальнена структурна модель програмного комплексу

На основі узагальненої структурної моделі програмного комплексу можна зробити висновок про те, що розроблювана програма має являти собою багатокористувацький комплекс у web-середовищі. Тому необхідно визначити структуру технологій забезпечення програмного проекту. Вибір технологій здійснюється на основі таких міркувань. Щойно було реалізовано величезний потенціал Web, з'явився новий попит на здатність генерувати відповідний контент, який був би адаптований до його протоколів. З'явилося безліч технологій і спроб стандартизації. HTML, CSS, Javascript і мови

програмування адаптувалися до моделі Web, починаючи з найпростіших сценаріїв, коли сторінки містили тільки звичайний текст із кількома посиланнями. Веб-сайти розвивалися. Сьогодні користувачам доступні сторінки з багатим змістом і взаємодією, з функціональними можливостями, які ніколи раніше не були передбачені для середовища, яке від самого початку було настільки елементарним на інтерактивність. Web являє собою клієнт-серверне середовище, засноване на запитах і відповідях [11, 12]. Ці запити між браузером і сервером вимагали стандартизованого зв'язку, і було створено протоколи для стандартного способу обміну інформацією. У роботі було прийнято рішення використовувати такий комплекс технологій забезпечення програмного проекту, показаний на рис. 2.8.

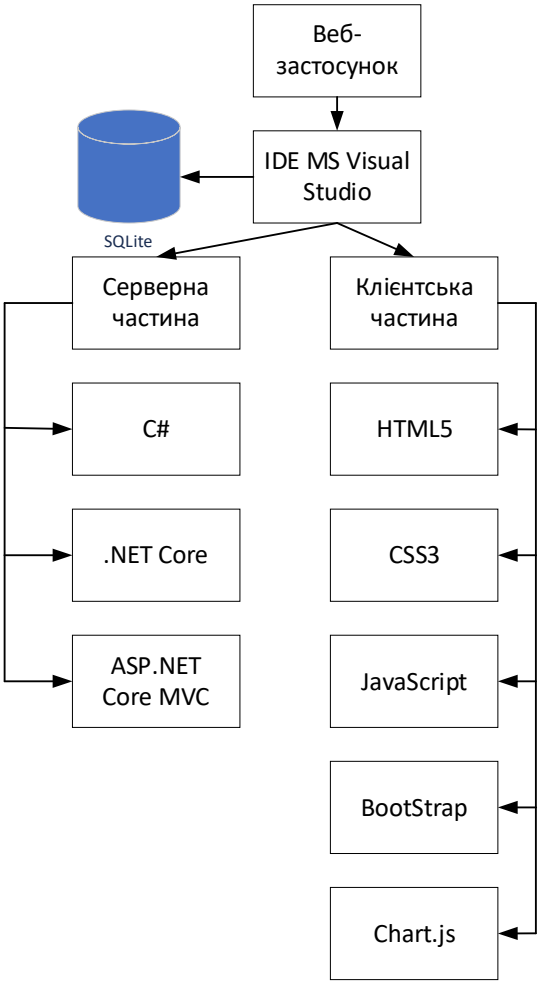


Рисунок 2.8 – Структурна схема технологій забезпечення програмного проекту

2.3 Розробка функціональної моделі програмного комплексу

Функціональна модель програмного комплексу тайм-менеджменту неформальної освіти має відображати впорядкований комплекс функцій або сценаріїв дій проєкту. Вимога функціональної моделі - це нагальна потреба. Коли ми говоримо про функціональні вимоги, ми маємо на увазі запит на функціональність, яку має виконувати програмне забезпечення. Функціональна модель програмного забезпечення може задовольняти кілька функціональних вимог. Це означає, що у функціоналі може бути виконано одну або кілька функціональних вимог, а не обов'язково тільки одну. Функціональні можливості існують тільки для виконання функціональних вимог. Тому без функціональних вимог немає функціональних можливостей, а без функціональних можливостей не існує системи. Уже одне це міркування демонструє абсолютну і незаперечну важливість незаперечну важливість функціональних вимог в обсязі системи. На рис. 2.9 показано функціональну модель програмного комплексу.

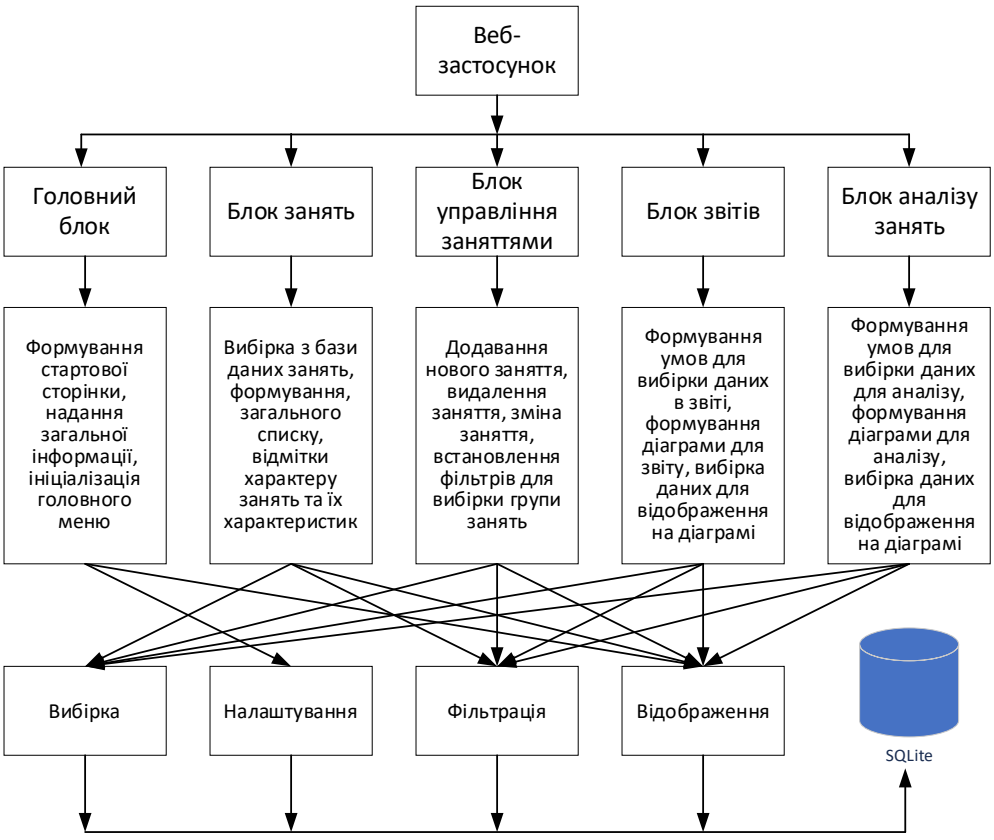


Рисунок 2.9 – Функціональна модель програмного комплексу

Іншими словами, вимога функціональності програмного забезпечення має бути реалізована. Зазвичай фахівці з розробки програмного забезпечення асоціюють ідею функціональних вимог із фактичними функціональними можливостями системи. Але необхідно розуміти, що функціональна модель не обов'язково задовольнятиме лише одну функціональну вимогу.

На основі розробленої функціональної моделі програмного комплексу тайм-менеджменту була розроблена функціональна діаграма послідовності дій застосунку, яка показана на рис. 2.10.

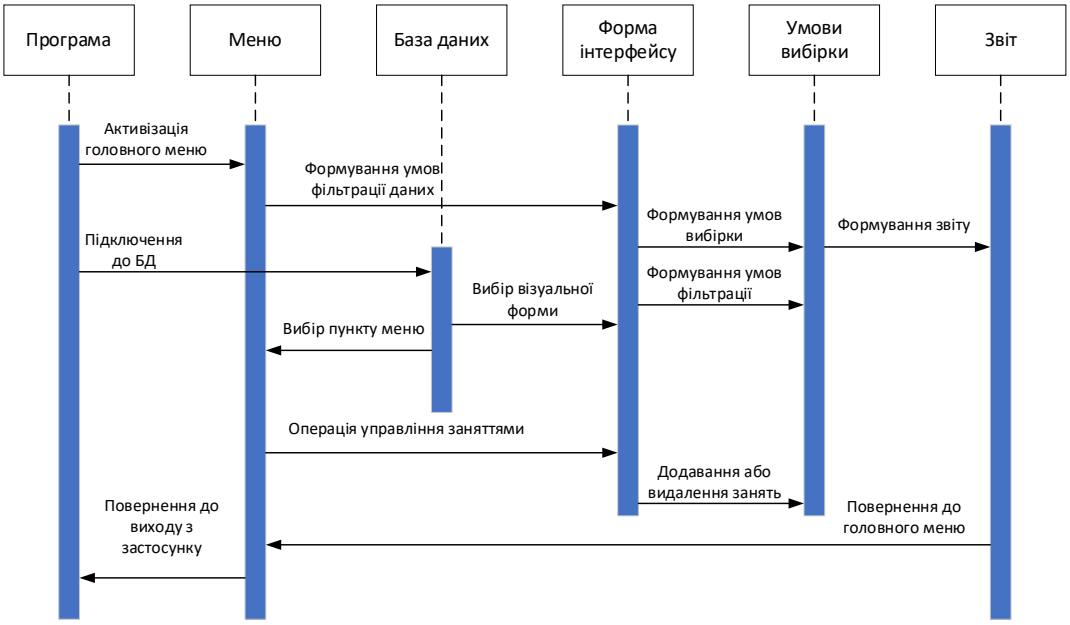


Рисунок 2.10 – Функціональна діаграма послідовності дій застосунку

Розроблена діаграма використовується для визначення потоку дій, матеріалів або людей у будь-якому процесі з метою його поліпшення. Серед поліпшень – виявлення непотрібних складнощів, проблемних ділянок і можливостей для спрощення. Вона також служить для аналізу того, де в процесі з'являється потрібна інформація. Цей інструмент часто буває дуже корисним під час розроблення аналізу ефектів відмов [13]. Діяльність має починатися від події, яка створює потребу клієнта, до моменту задоволення цієї потреби. Види діяльності мають бути розділені відповідно до того, хто їх виконує, і визначити лінії взаємодії між різними компонентами моделі.

2.4 Розробка структури бази даних системи тайм-менеджменту

Структура автоматизованої системи тайм-менеджменту неформальної освіти передбачає наявність сховища даних. Це необхідно для оперативного зберігання і опрацювання відомостей про студентів, про матеріал, що вивчається, про результати навчання і багато інших аспектів.

Для розроблення концептуальної структури бази даних насамперед необхідно провести аналіз і визначити, що є сутностями в інформаційному забезпеченні тайм-менеджменту. Структура інформаційного забезпечення тайм-менеджменту показана на рис. 2.11.

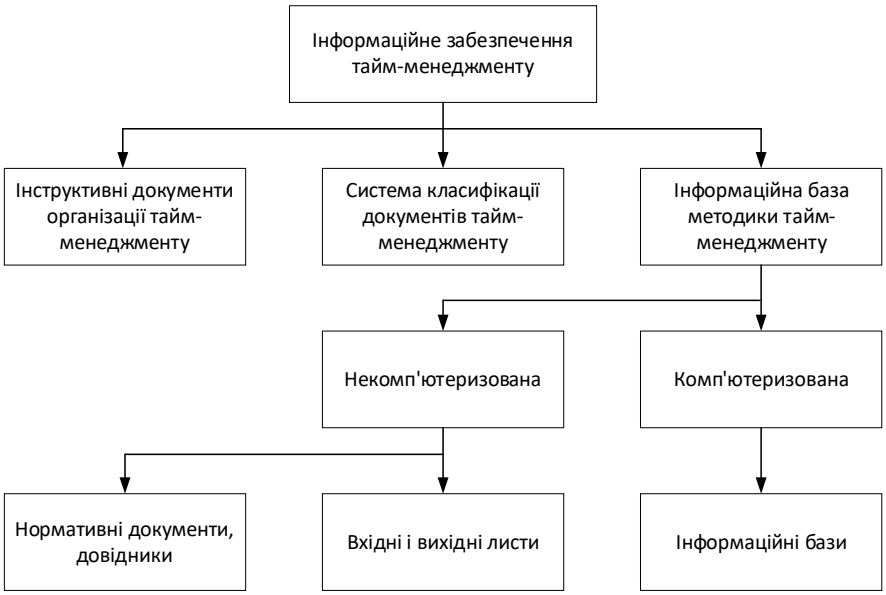


Рисунок 2.11 – Структура інформаційного забезпечення тайм-менеджменту

Проведений аналіз спрямований на підвищення ефективності використання бази даних, що забезпечує роботу всієї системи. Забезпечення швидкості та ефективності роботи бази даних є однією з основних вимог до гарантованої якості всієї автоматизованої системи.

Часто управління системою пов'язане із суперечливими проблемами, наприклад: немає часу на виконання завдання, робота занадто складна або ресурс недостатній. Для таких ситуацій рекомендується враховувати три загальні аспекти управління системою: час, завдання і ресурси. Без розуміння

того, як ці три фактори взаємопов'язані, система може легко скотитися в реактивний режим, постійно тільки реагуючи на кризові моменти. Ці три фактори постійно взаємодіють, змінюючи пріоритети і змінюючи свою важливість у міру реалізації проєкту. Розуміння того, як взаємодіють ці фактори, дає змогу об'єктивно поглянути на процес розроблення бази даних тайм-менеджменту.

Сформулюємо основні вимоги до архітектури бази даних для зберігання інформації процесу тайм-менеджмент неформальної освіти [14, 15].

Організація та структура. Дані зберігаються в таблицях, які містять рядки (записи) і стовпці (поля). Кожна таблиця призначена для зберігання інформації про конкретний тип об'єкта. Схеми визначають структуру бази даних, включно з таблицями, полями, типами даних і відношеннями між різними даними.

Доступ до даних і маніпулювання ними. Мова SQL - мова структурованих запитів (SQL) використовується для вставки, оновлення, видалення та запиту даних. Основні операції, підтримувані базами даних, - це створення, читання, оновлення та видалення даних.

Цілісність даних - Referential Integrity (посилальна цілісність). Гарантує, що відносини між таблицями залишаються послідовними.

Цілісність сутності: гарантує, що кожна таблиця має унікальний первинний ключ.

Безпека. Керування доступом: використання дозволів та аутентифікації для захисту. Конфіденційних даних: аудит і журнали для запису операцій, що виконуються з базою даних, для моніторингу та аудиту.

Надійність і відновлення. Транзакції гарантують атомарність, узгодженість, ізоляцію та довговічність, необхідні для надійності роботи бази даних.

Резервне копіювання та відновлення. Методи резервного копіювання даних та їх відновлення в разі збою.

Продуктивність і ефективність. Індекси та структури даних, що

підвищують швидкість виконання запитів до таблиць.

Нормалізація. Процес організації даних для зменшення надмірності та поліпшення цілісності.

Масштабованість. Вертикальна масштабованість - збільшення потужності серверного обладнання для підвищення продуктивності. Горизонтальна масштабованість - розподіл робочого навантаження між кількома серверами [16, 17].

На основі сформульованих вимог розроблено схему бази даних автоматизованої системи тайм-менеджменту неформальної освіти, що показана на рис. 2.12.

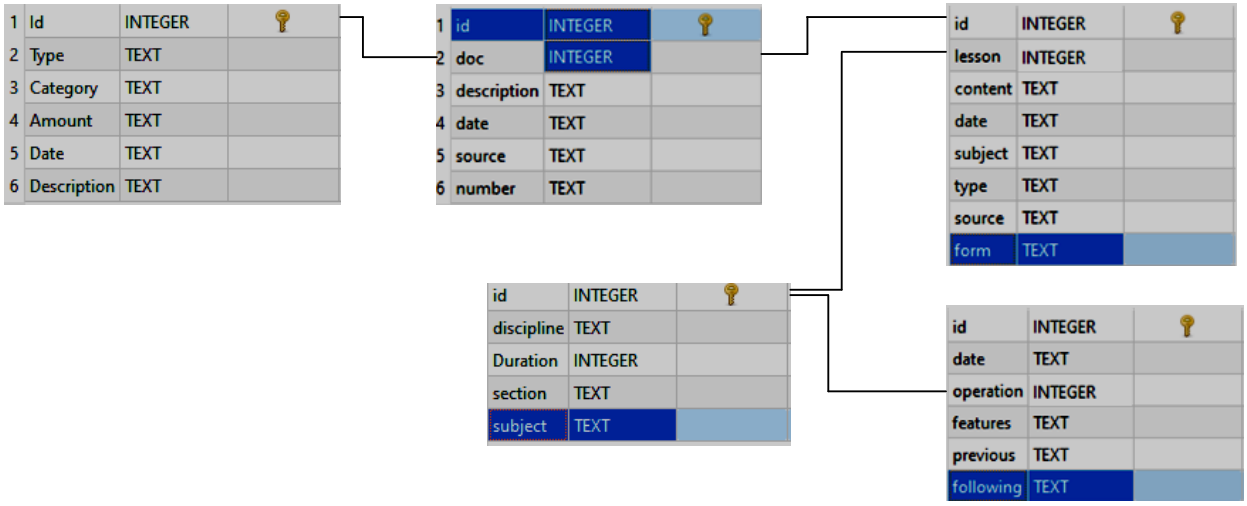


Рисунок 2.12 – Структура бази даних системи тайм-менеджменту

Наведемо деякі додаткові вимоги. Клієнтські термінали взаємодіють із додатками на Web-сервері та повинні обмінюватися даними з базою даних на сервері. Більша частина даних із бази даних сервера не повинна завантажуватися на пристрій, а тільки ті дані, які можуть бути використані користувачем автоматизованої системи. Необхідно ретельно планувати передачу даних, щоб без необхідності не збільшувати час синхронізації і навантаження на таблиці пристрою, а також скоротити час виконання запитів і використання простору для зберігання даних.

2.5 Розробка блок-схем алгоритмів програмних модулів системи тайм-менеджменту

Розробимо блок-схеми алгоритмів автоматизованої системи тайм-менеджменту. На рис. 2.13 показано алгоритм додавання заняття неформальної освіти до бази даних.

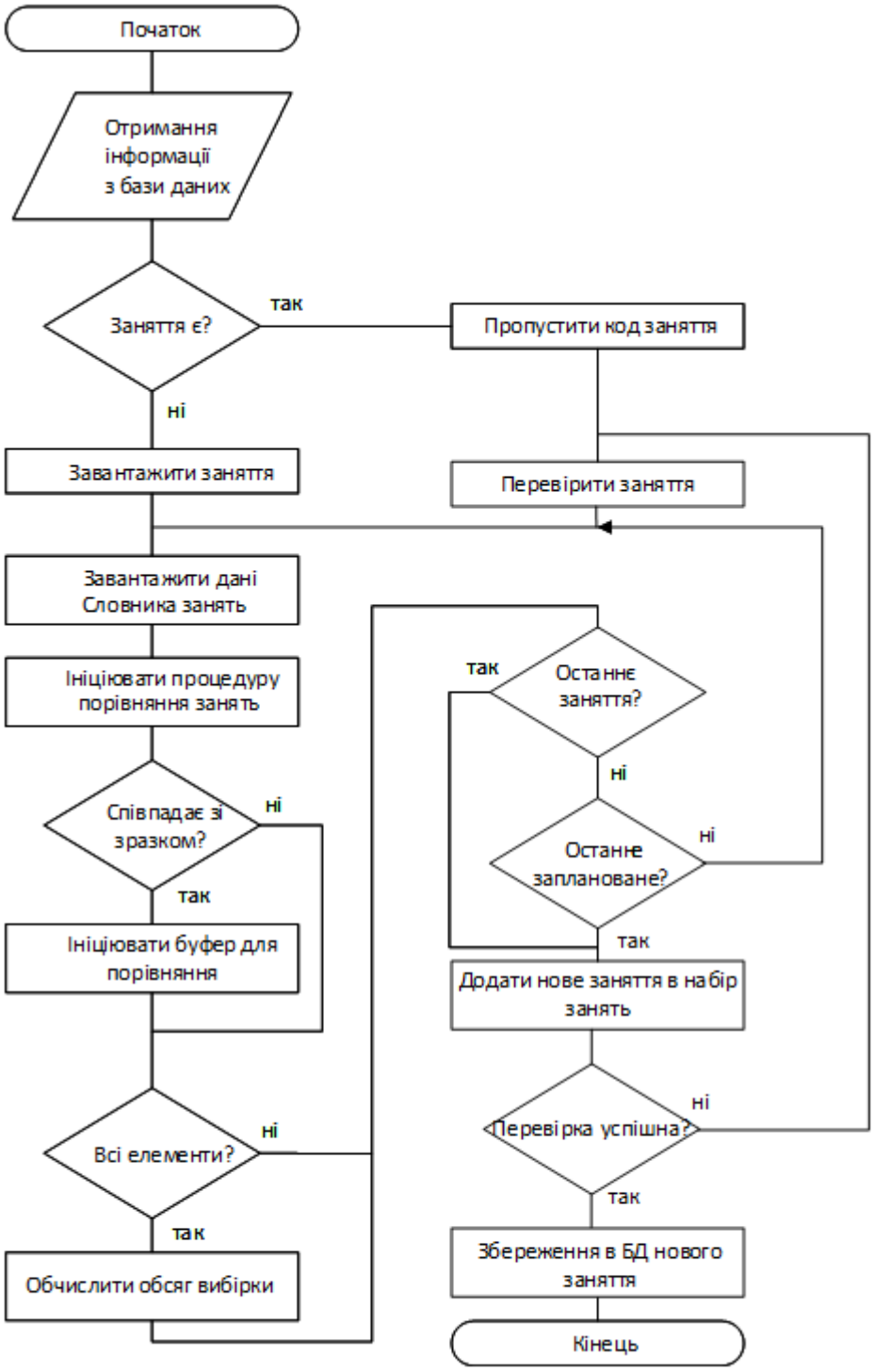


Рисунок 2.13 – Алгоритм додавання заняття в базу даних

На рис. 2.14 показано блок-схему алгоритму кодування занять в базі даних.

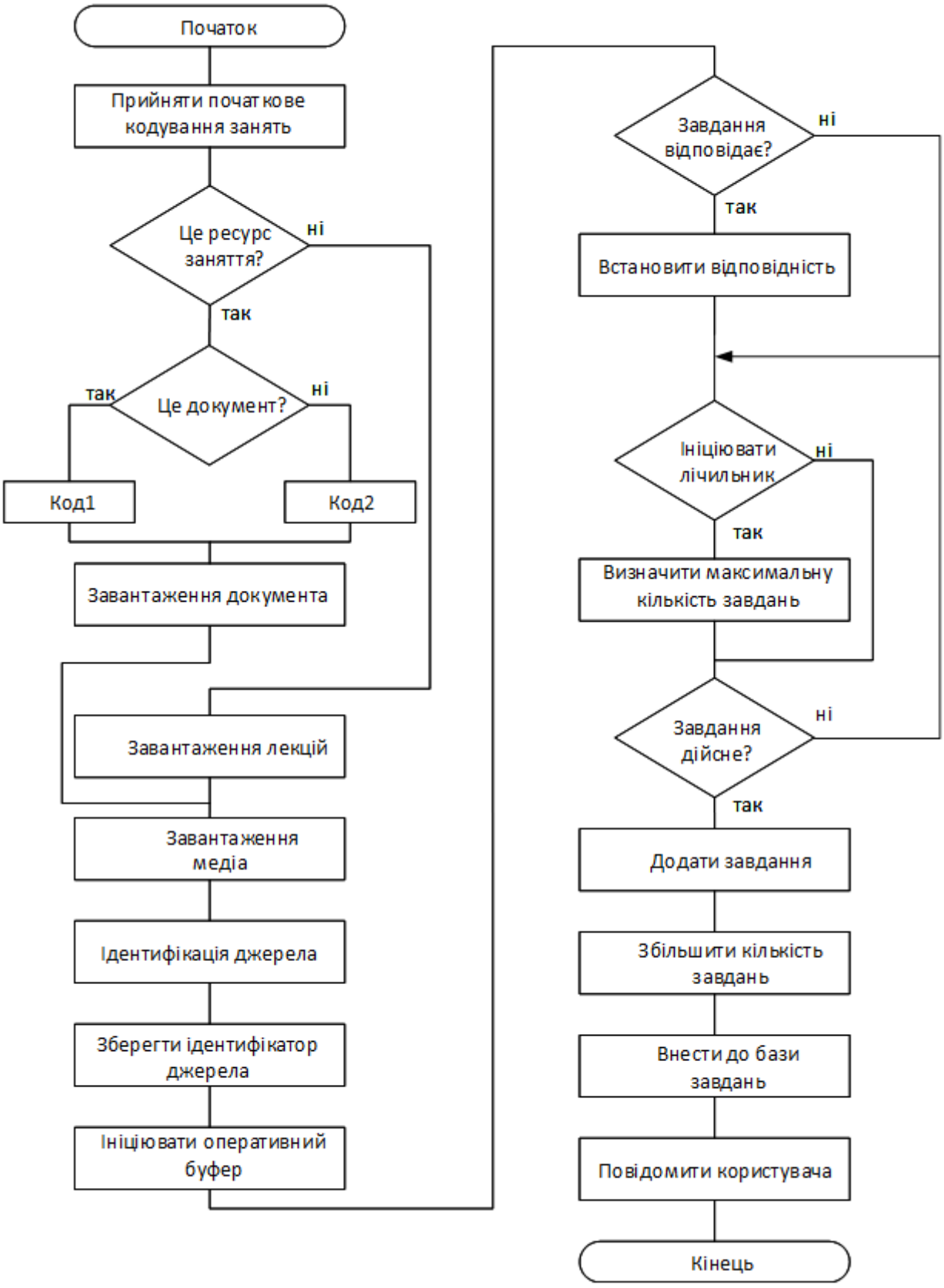


Рисунок 2.14 – Алгоритм кодування занять в базі даних

Як було зазначено раніше, основним сховищем даних у системі тайм-менеджменту є база даних. Тому вся робота можлива тільки при підключенні до бази даних. Нині технології баз даних завойовують дедалі більше місце серед різних типів користувачів для різних цілей. Тому спільним завданням є розробка програм для підключення до різних баз даних для збереження своїх даних без втрат. Блок-схема алгоритму підключення до бази даних тайм-менеджменту показана на рис. 2.15.

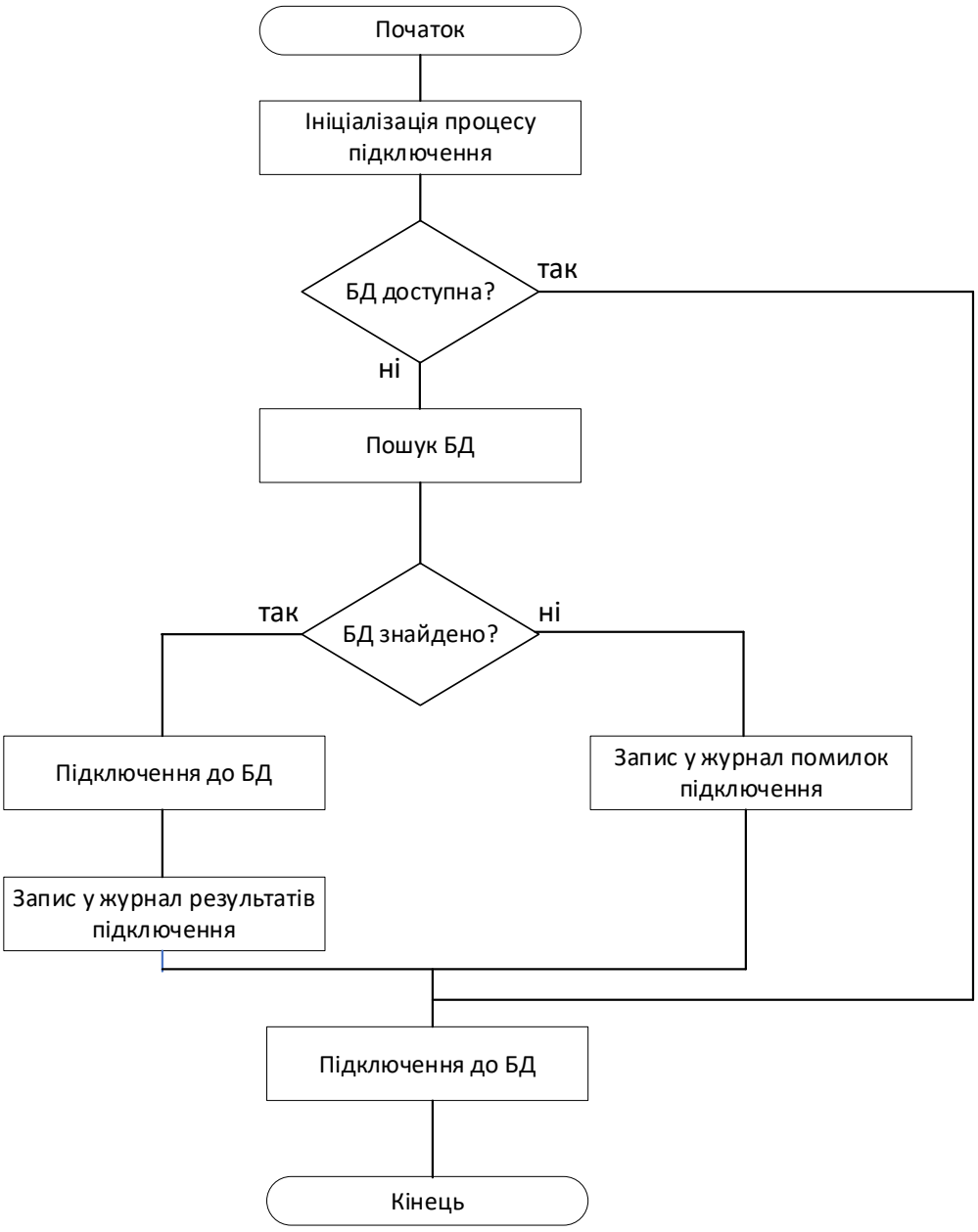


Рисунок 2.15 – Алгоритм підключення до бази даних

2.6 Розробка моделі візуального інтерфейсу користувача програмного комплексу

Розроблення моделі візуального інтерфейсу користувача автоматизованої системи тайм-менеджменту неформальної освіти потребує врахування безлічі чинників. До основних чинників належать функціональність моделі та спосіб програмної реалізації. Орієнтований на користувача дизайн є одним з основних підходів, але його застосовність ставиться під загрозу через складність розмежування користувачів. Особливості користувачів сильно впливають на модель інтерфейсу. Крім того, необхідно враховувати, що кінцева система передбачається у вигляді web-застосунку [18 - 20]. З урахуванням перерахованих факторів розробимо модель головної частини стартової сторінки сайту, яка показана на рис. 2.16.

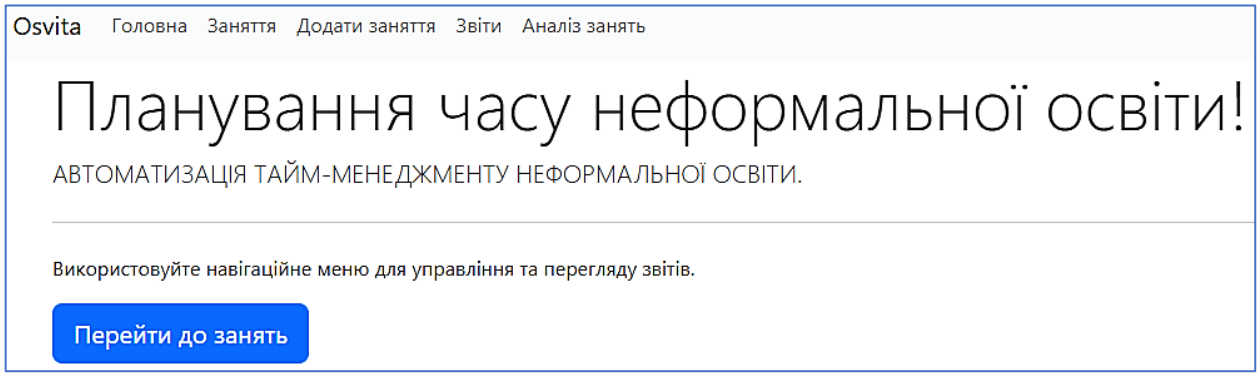


Рисунок 2.16 – Модель головної частини стартової сторінки сайту

Нижче наведено модель заголовка таблиці списку занять, показану на рис. 2.17

Рисунок 2.17 – Модель головної частини таблиці списку занять

Для реалізації функцій тайм-менеджменту система повинна мати розвинені засоби вибірки та фільтрації даних. Тому в роботі запропоновано моделі форм для функцій фільтрації. Однією з основних концепцій сучасного дизайну є дизайн, орієнтований на користувача (UCD), або дизайн, орієнтований на людину (HCD). В основі цієї концепції лежить ідея про те, що кожний процес проєктування має орієнтуватися на користувача і бути його метою під час розроблення концепції та планування. Вона передбачає активну участь користувача в процесі, а не просто обмеження його роллю кінцевого споживача або абстрактної фігури. Ця участь є основоположною в процесі проектування, починаючи з початкових етапів, оскільки дає змогу створити програму відповідно до їхніх реальних потреб. У процесі UCD вважається, що програму було розроблено відповідно до специфікацій користувача. Модель форми встановлення параметрів фільтрів показана на рис. 2.18.

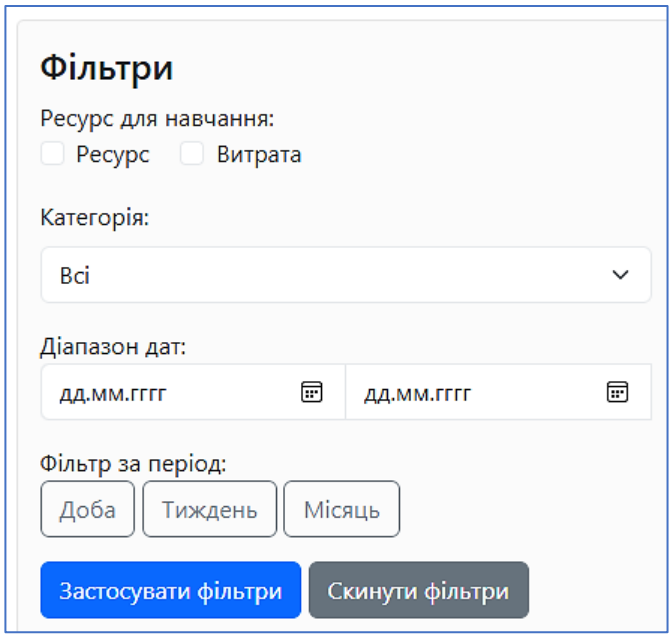


Рисунок 2.18 – Модель форми встановлення параметрів фільтрів

Окремої уваги потребує формування візуального інтерфейсу форм звітів. Звіти мають містити діаграми та графіки для здійснення повноцінного аналізу.

Концептуальні діаграми можуть бути складені для всього звіту, для частини звіту, для конкретної теми у звіті тощо. Існують різні способи побудови концептуальної форми звіту або діаграми, тобто існують різні способи представлення концептуальної ієрархії на діаграмі. Крім того, концептуальні моделі, складені різними фахівцями в одній і тій самій царині знань, імовірно, відображатимуть невеликі розбіжності в розумінні та інтерпретації відносин між ключовими поняттями у формі звіту. Важливо зазначити, що модель форми звіту завжди слід розглядати як "модель діаграми", а не як "діаграму моделі". Інакше кажучи, будь-яку модель звіту або діаграми слід розглядати тільки як одне з можливих уявлень певної структури. Модель форми відображення діаграм звітів показана на рис.2.19.

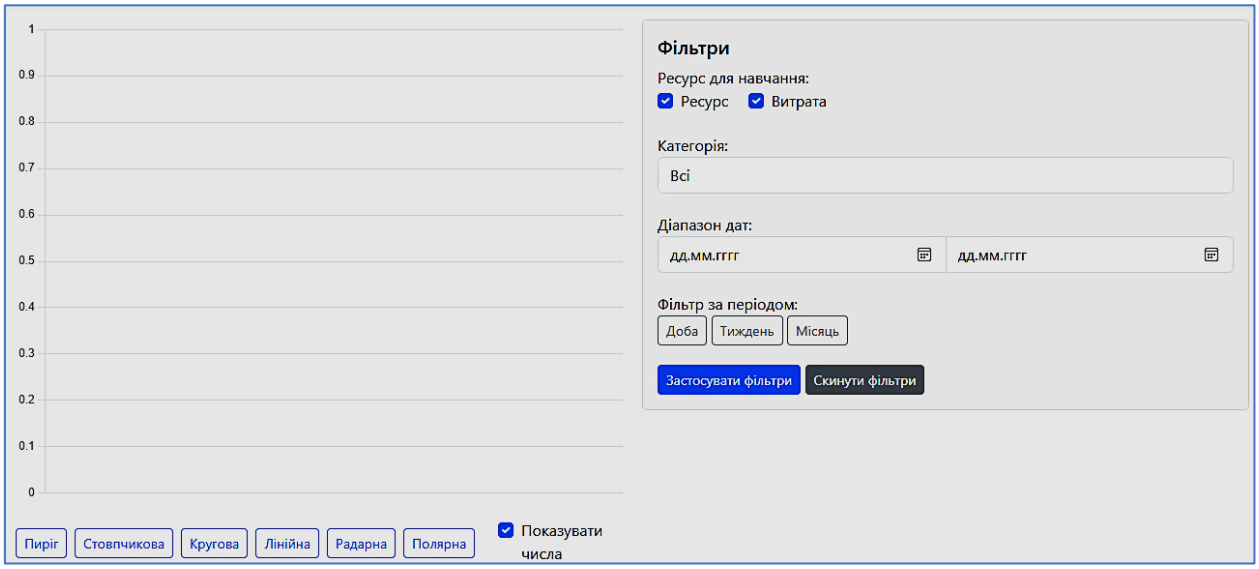


Рисунок 2.19 – Модель форми відображення діаграм звітів

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ АВТОМАТИЗАЦІЇ ТАЙМ-МЕНЕДЖМЕНТУ ЗАВДАНЬ НЕФОРМАЛЬНОЇ ОСВІТИ

3.1 Розробка програмних модулів візуального інтерфейсу системи автоматизації

Для розробки програмних модулів візуального інтерфейсу користувача системи тайм-менеджменту неформальної освіти необхідно проаналізувати та обрати основні технології проектування розроблення програмного забезпечення. Розглянемо основні технології які були використані в роботі.

Макет web-сторінки має являти собою єдине ціле з елементів. Цей принцип особливо яскраво проявляється, коли у великому проекті необхідно, щоб підсторінки були візуально уніфікованими з основним макетом проекту. Це полегшує навігацію в послідовному, інтегрованому та передбачуваному середовищі. На додаток до цих принципів, які визначають організацію макета, необхідні деякі властивості або якості, які покращують роботу з сайтом, особливо використання і навігацію. Також, необхідна розбірливість сторінки, тому що немає сенсу бути красивим, якщо це не може бути зрозуміло. І, нарешті, важлива доступність, що забезпечує загальний доступ до інформації сторінки.

Мова програмування C#. Серед основних характеристик C# можна відзначити такі. Простота: розробники C# часто кажуть, що ця мова така ж потужна, як C++, і така ж проста, як Visual Basic. Повністю об'єктно-орієнтована: у C# будь-яка змінна має бути частиною класу. Сильно типізований: це допоможе уникнути помилок, пов'язаних із неправильним маніпулюванням типами, неправильних присвоювань тощо. Генерує керований код: як середовище .NET є керованим, так і C#. Усе є об'єктом: System.Object - базовий клас усієї системи типів C#.

JavaScript - це мова програмування, яка була створена в середині 1995 року і є однією з основ для веб-розробки, тобто тих сайтів, які ми звикли переглядати в повсякденному житті [21]. Оскільки вона є однією з основ веб-розробки, ця мова присутня на переважній більшості сайтів, які ми переглядаємо. Але ця чудова мова не обмежується лише веб-сайтами. Є можливість використовувати її для багатьох інших цілей. У JavaScript є три основні характеристики.

Висока продуктивність. Крім того, що ця мова має дуже простий спосіб написання, вона не споживає занадто багато пам'яті комп'ютера.

Мультиплатформеність. Це означає, що ця мова не обмежується тільки одним типом системи або платформи, а може використовуватися на кількох різних платформах.

Мультипарадигмальний. Будучи мультипарадигмальною мовою, JavaScript схильний до різних стилів програмування, як-от імперативне програмування, функціональне програмування, подієво-орієнтоване, об'єктно-орієнтоване і засноване на прототипах.

База даних SQLite. Бібліотека, вбудована в процес, який реалізує рушій бази даних SQL - автономна, безсерверна, з нульовою конфігурацією. Код, що є суспільним надбанням і вільний для будь-якого використання - комерційне, приватне. Читання і запис безпосередньо з/до звичайних файлів на диску, повна система мови SQL з безліччю таблиць, індексів, тригерів і подань в одному файлі на диску. Файл, створений SQLite, працює на будь-якій платформі.

Середовище розробки MS Visual Studio. У візуальній мові програмування користувач розробляє програми, маніпулюючи елементами програми, замість того щоб задавати їх у текстовому вигляді. VPL дозволяє програмувати за допомогою візуальних виразів, просторового розташування тексту. Microsoft Visual Studio. Тут мови Visual Basic, Visual C# і т. д. є текстовими. Усі проаналізовані вище технології розроблення сучасного програмного забезпечення показані на рис.3.1.

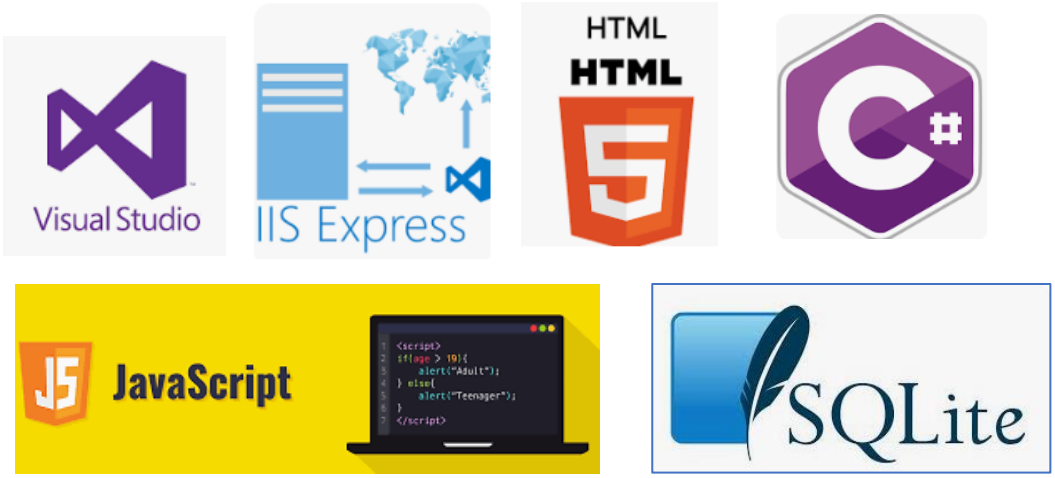


Рисунок 3.1 – Технології розробки модулів програмного проекту

На рис. 3.2 зображено головну (стартову) форму програмного проєкту.

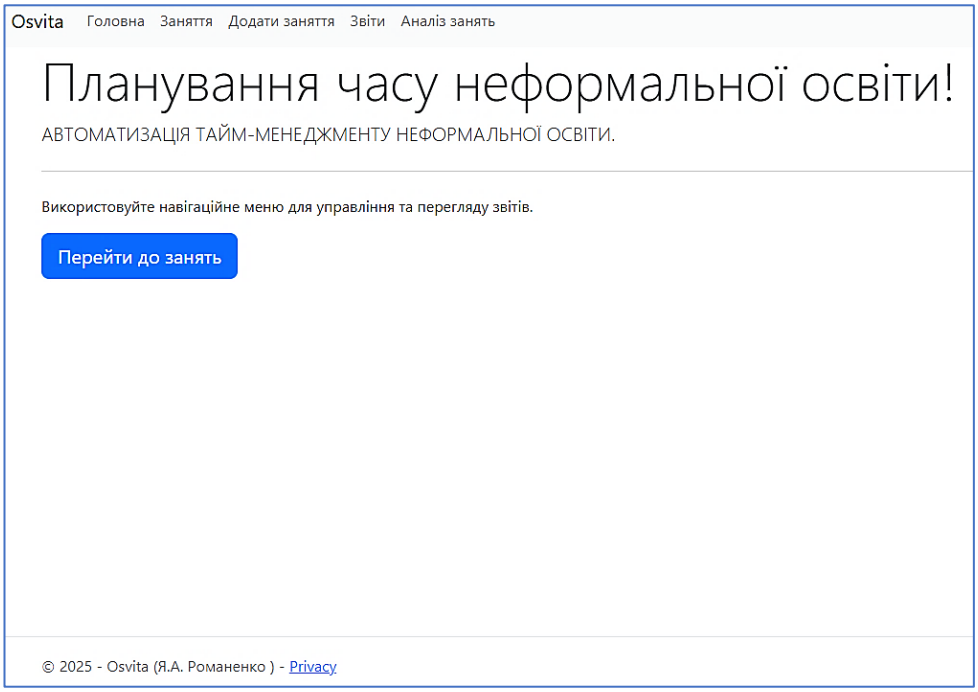


Рисунок 3.2 – Головна стартова форма сайту (застосунку)

Після завершення роботи над дизайном необхідно розпочати розробку сайту, тобто кодування сайту. Це може бути просто HTML і CSS, а може бути складнішим і включати в себе програмування, таке як PHP, MySQL. На рис. 3.3 показано фрагмент лістингу програмного коду формування стартової сторінки застосунку.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - Osvita</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css"
        crossorigin="anonymous" referrerpolicy="no-referrer" />
    <script type="importmap"></script>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/FinanceOblik.styles.css" asp-append-version="true" />
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
            <a class="navbar-brand" asp-page="/Index">     Osvita</a>
            <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
                    aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav">
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Index">Головна</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Index">Заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Create">Додати заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Reports/Index">Звіти</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/EfficiencyAnalysis/EfficiencyAnalysis">Аналіз занять</a>
                    </li>
                </ul>
            </div>
        </nav>
    </header>

    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <footer class="border-top footer text-muted">
        <div class="container">
            &copy; 2025 - Osvita (Я.А. Романенко ) - <a asp-area="" asp-page="/Privacy">Privacy</a>
        </div>
    </footer>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.2/jquery.validate.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate-unobtrusive/3.2.11/jquery.validate.unobtrusive.min.js"></script>
    @RenderSection("Scripts", required: false)

    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chartjs-plugin-datalabels"></script>


    @* @await RenderSectionAsync("Scripts", required: false) *@
</body>
</html>
```

Рисунок 3.3 – Лістинг програмного коду формування стартової сторінки застосунку

На рис. 3.4 показано розроблену форму відображення списку занять неформальної освіти. Ця форма має розвинений і виразний інтерфейс.

Рисунок 3.4 – Форма відображення списку занять неформальної освіти

На рис. 3.5 показано фрагмент лістингу програмного коду відображення списку занять.

```
@{
    ViewData["Title"] = "Список занять";
}
<p></p>
<h2>Список занять неформальної освіти</h2>

<div class="row">
    <div class="col-md-8">
        <table class="table table-striped">
            <thead>
                <tr>
                    <th>Ресурс для навчання</th>
                    <th>Категорія</th>
                    <th>Тривалість</th>
                    <th>Дата</th>
                    <th>Опис</th>
                    <th>Дії</th>
                </tr>
            </thead>
            <tbody>
                @foreach (var item in Model.Transactions)
                {
                    <tr>
                        <td>
                            @if (item.Type == "Ресурс")
                            {
                                <span class="badge bg-success">Є матеріал</span>
                            }
                            else
                            {
                                <span class="badge bg-danger">Потрібен час</span>
                            }
                        </td>
                        <td>
                            <i class="@Model.GetCategoryIcon(item.Category)"></i> @item.Category
                        </td>
```

Рисунок 3.5 – Лістинг програмного коду відображення списку занять

```
                    <td>@item.Amount.ToString("N2")</td>
                    <td>@item.Date.ToString("dd.MM.yyyy")</td>
                    <td>@item.Description</td>
                    <td>
                        <a asp-page="./Edit" asp-route-id="@item.Id" class="btn btn-sm btn-primary">Редагувати</a>
                        <a asp-page="./Delete" asp-route-id="@item.Id" class="btn btn-sm btn-danger">Видалити</a>
                    </td>
                </tr>
            }
        </tbody>
    </table>
</div>
<div class="filter-container col-md-4">
    <h4>Фільтри</h4>
    <form method="get">
        <div class="mb-3">
            <label>Ресурс для навчання:</label><br />
            @foreach (var type in Model.TypeSelectList)
            {
                <div class="form-check form-check-inline">
                    <input class="form-check-input" type="checkbox" name="FilterTypes"
                        value="@type.Value" @(Model.FilterTypes.Contains(type.Value) ? "checked" : "")>
                    <label class="form-check-label">@type.Text</label>
                </div>
            }
        </div>
        <div class="mb-3">
            <label asp-for="FilterCategory" class="form-label">Категорія:</label>
            <select asp-for="FilterCategory" asp-items="Model.CategorySelectList" class="form-select">
                <option value="">Всі</option>
            </select>
        </div>
        <div class="mb-3">
            <label>Діапазон дат:</label>
            <div class="input-group">
                <input asp-for="StartDate" class="form-control" type="date" />
                <input asp-for="EndDate" class="form-control" type="date" />
            </div>
        </div>

        <div class="mb-3">
            <label>Фільтр за період:</label><br />
            <button type="submit" name="period" value="day" class="btn btn-outline-secondary">Доба</button>
            <button type="submit" name="period" value="week" class="btn btn-outline-secondary">Тиждень</button>
            <button type="submit" name="period" value="month" class="btn btn-outline-secondary">Місяць</button>
        </div>
        <button type="submit" class="btn btn-primary">Застосувати фільтри</button>
        <a asp-page="./Index" class="btn btn-secondary">Скинути фільтри</a>
    </form>
</div>
</div>

<a asp-page="./Create" class="btn btn-success mt-3">Додати заняття</a>
```

Рисунок 3.5 – Лістинг програмного коду відображення списку занять (продовження)

Розглянемо форму фільтрації для вибору занять. Користувачі легше запам'ятовують простий і ефективний сайт. Це відбувається тому, що користувачі "ледачі", вони не читають, вони просто і швидко "сканують" сторінку, не звертаючи ані найменшої уваги на потрібну їм інформацію. Якщо вони не можуть знайти те, що їм потрібно, швидко, без будь-яких зусиль, вони просто закриють сторінку. Іншими словами, користувача легко переконати. Форма фільтрації для вибору занять розроблена, виходячи з цих міркувань. Форма показана на рис. 3.6.

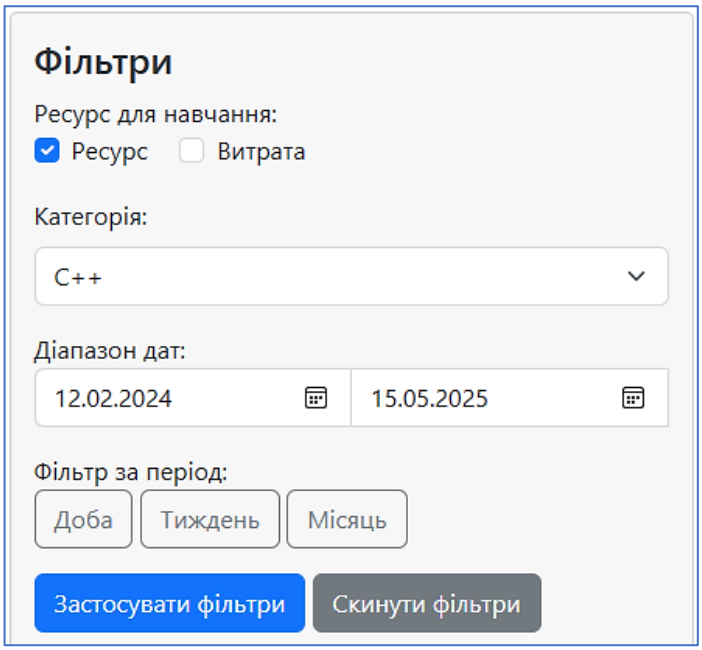


Рисунок 3.6 – Форма встановлення параметрів фільтрів для вибору занять

На рисунку 3.7 показано лістинг програмного коду відображення параметрів фільтрів для вибору занять.

```
<div class="filter-container col-md-4">
    <h4>Фільтри</h4>
    <form method="get">
        <div class="mb-3">
            <label>Ресурс для навчання:</label><br />
            @foreach (var type in Model.TypeSelectList)
            {
                <div class="form-check form-check-inline">
                    <input class="form-check-input" type="checkbox" name="FilterTypes"
                        value="@type.Value" @(Model.FilterTypes.Contains(type.Value) ? "checked" : "")>
                    <label class="form-check-label">@type.Text</label>
                </div>
            }
        </div>
        <div class="mb-3">
            <label asp-for="FilterCategory" class="form-label">Категорія:</label>
            <select asp-for="FilterCategory" asp-items="Model.CategorySelectList" class="form-select">
                <option value="">Всі</option>
            </select>
        </div>
        <div class="mb-3">
            <label>Діапазон дат:</label>
            <div class="input-group">
                <input asp-for="StartDate" class="form-control" type="date" />
                <input asp-for="EndDate" class="form-control" type="date" />
            </div>
        </div>
        <div class="mb-3">
            <label>Фільтр за період:</label><br />
            <button type="submit" name="period" value="day" class="btn btn-outline-secondary">Доба</button>
            <button type="submit" name="period" value="week" class="btn btn-outline-secondary">Тиждень</button>
            <button type="submit" name="period" value="month" class="btn btn-outline-secondary">Місяць</button>
        </div>
        <button type="submit" class="btn btn-primary">Застосувати фільтри</button>
        <a asp-page="./Index" class="btn btn-secondary">Скинути фільтри</a>
    </form>
</div>
</div>
```

Рисунок 3.7 – Лістинг програмного коду відображення параметрів фільтрів для вибору занять

На рис. 3.8 показано форму додавання нового заняття в базу даних.

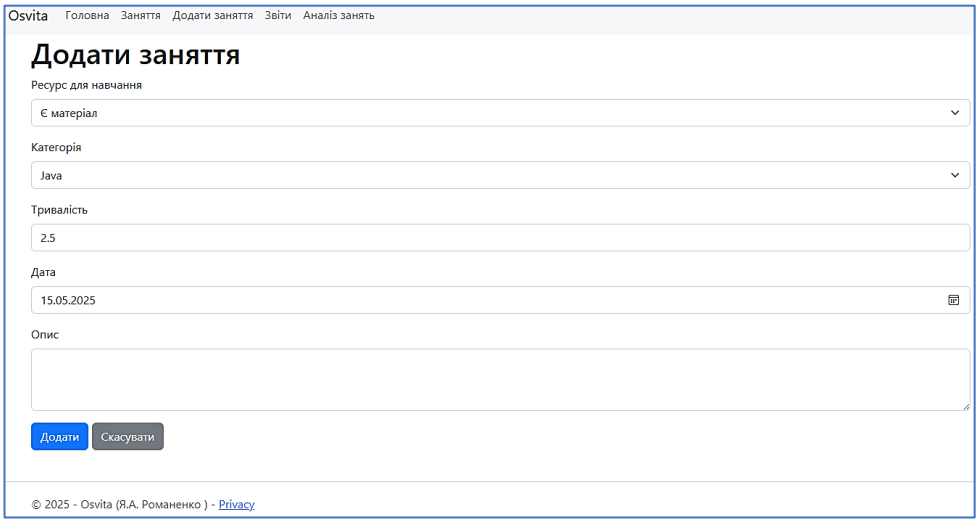


Рисунок 3.8 – Форма додавання нового заняття неформальної освіти

На рис. 3.9 показано лістинг програмного коду форми додавання нового заняття неформальної освіти.

```
<h1>Додати заняття</h1>

<form method="post">
    <div class="mb-3">
        <label asp-for="Transaction.Type" class="form-label"></label>
        <select asp-for="Transaction.Type" asp-items="Model.TypeSelectList"
        class="form-select" required id="Transaction_Type">
            <option value="">Виберіть тип</option>
        </select>
        <span asp-validation-for="Transaction.Type" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Category" class="form-label"></label>
        <select asp-for="Transaction.Category" asp-items="Model.CategorySelectList"
        class="form-select" required id="Transaction_Category">
            <option value="">Виберіть категорію</option>
        </select>
        <span asp-validation-for="Transaction.Category" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Amount" class="form-label"></label>
        <input asp-for="Transaction.Amount" class="form-control" required />
        <span asp-validation-for="Transaction.Amount" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Date" class="form-label"></label>
        <input asp-for="Transaction.Date" class="form-control" type="date" required />
        <span asp-validation-for="Transaction.Date" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Description" class="form-label"></label>
        <textarea asp-for="Transaction.Description" class="form-control" rows="3"></textarea>
        <span asp-validation-for="Transaction.Description" class="text-danger"></span>
    </div>
    <button type="submit" class="btn btn-primary">Додати</button>
    <a asp-page="./Index" class="btn btn-secondary">Скасувати</a>
</form>
```

Рисунок 3.9 – Лістинг програмного коду форми додавання нового заняття неформальної освіти

3.2 Розробка бази даних автоматизованої системи

Для розробки бази даних автоматизованої системи тайм-менеджменту необхідно проаналізувати весь комплекс інформаційних технологій для розробки програмного проекту. До таких технологій належать мова програмування C# та інтегроване середовище розробки Visual Studio.

Як відомо, C# - це об'єктно-орієнтована та компонентно-орієнтована мова програмування. Це означає, що мова підтримує конструкції та реалізації нативно, тож програмісти можуть працювати над проєктами чітко та об'єктивно, використовуючи ресурси, які реалізовані в самій мові. Наразі на ринку існує кілька IDE, що задовольняють цю потребу, але найвідомішими і найактуальнішими є Visual Studio і Visual Studio Code. Ці IDE були створені компанією Microsoft з метою полегшити нам розробку і з їхньою допомогою ми можемо створювати, аналізувати та поширювати програмне забезпечення. Visual Studio є IDE, яка добре зарекомендувала себе протягом багатьох років, а також часто вдосконалюється командою інженерів, що робить її чудовим вибором. Ви можете встановити і використовувати Visual Studio на Windows, Linux або Mac. Загальний вигляд і структура програмного проекту, який містить базу даних, наведено на рис. 3.10.

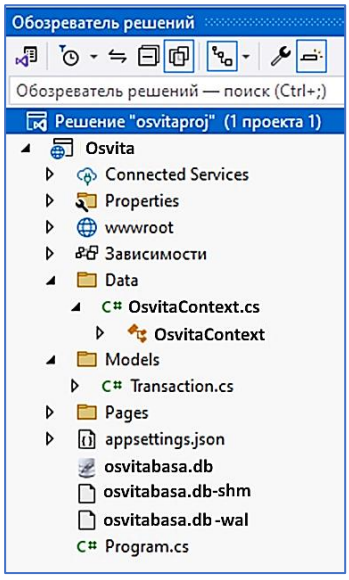
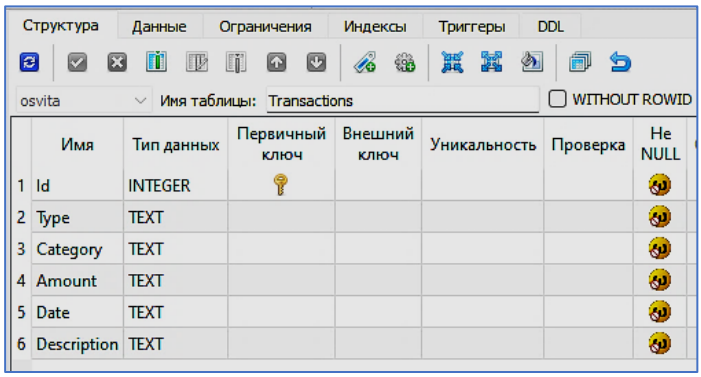


Рисунок 3.10 – Інформаційне забезпечення в менеджері проєкту Visual Studio

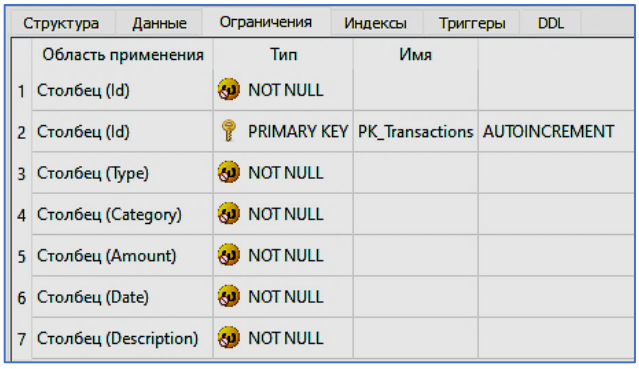
SQLite – це бібліотека, що реалізує базу даних мовою структурованих запитів SQL. Програми, що використовують бібліотеку SQLite, можуть отримувати доступ до баз даних SQL без запуску окремого процесу реляційної системи керування базами даних. Сьогодні підтримка баз даних SQLite у багатьох системах уже дає змогу виконувати високорівневі запити з використанням SQL. У випадку з проектом C# у цій кваліфікаційній роботі використовуються бази даних SQLite. Середовище Visual Studio пропонує методи доступу до цих баз даних і маніпулювання ними, а також інструменти, необхідні для адміністрування. На рис. 3.11 показано структуру таблиці бази даних SQLite управління завданнями неформальної освіти.



	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL
1	Id	INTEGER					
2	Type	TEXT					
3	Category	TEXT					
4	Amount	TEXT					
5	Date	TEXT					
6	Description	TEXT					

Рисунок 3.11 – Конструкція таблиці управління завданнями неформальної освіти бази даних SQLite

На рисунку 3.12 показано структуру таблиці характеристик завдань неформальної освіти бази даних SQLite.

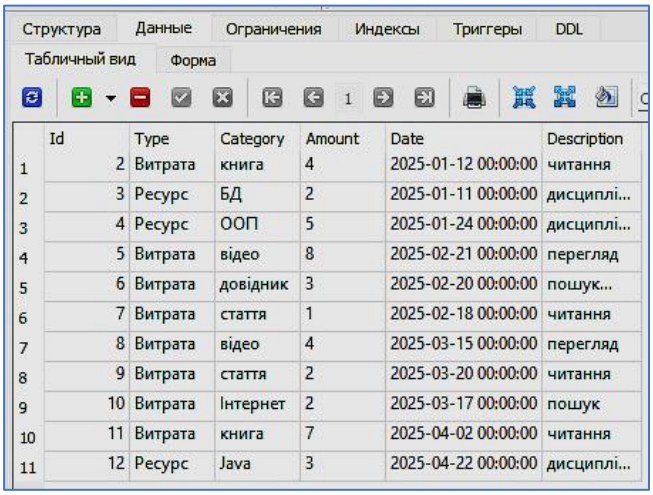


	Область применения	Тип	Имя	
1	Столбец (Id)	NOT NULL		
2	Столбец (Id)	PRIMARY KEY	PK_Transactions	AUTOINCREMENT
3	Столбец (Type)	NOT NULL		
4	Столбец (Category)	NOT NULL		
5	Столбец (Amount)	NOT NULL		
6	Столбец (Date)	NOT NULL		
7	Столбец (Description)	NOT NULL		

Рисунок 3.12 – Конструкція таблиці характеристик завдань неформальної освіти бази даних SQLite

SQLite - це вбудований механізм баз даних SQL. Він не має окремого серверного процесу. Процес. SQLite читає і записує безпосередньо у звичайні дискові файли. База даних з кількома таблицями, індексами, тригерами і поданнями міститься в одному дисковому файлі.

Усі операції, що виконуються з базою даних, можуть бути запитані в СУБД за допомогою мови SQL. На рисунку 3.13 показано таблицю даних завдань неформальної освіти бази даних SQLite.



	Id	Type	Category	Amount	Date	Description
1	2	Витрата	книга	4	2025-01-12 00:00:00	читання
2	3	Ресурс	БД	2	2025-01-11 00:00:00	дисциплі...
3	4	Ресурс	ООП	5	2025-01-24 00:00:00	дисциплі...
4	5	Витрата	відео	8	2025-02-21 00:00:00	перегляд
5	6	Витрата	довідник	3	2025-02-20 00:00:00	пошук...
6	7	Витрата	стаття	1	2025-02-18 00:00:00	читання
7	8	Витрата	відео	4	2025-03-15 00:00:00	перегляд
8	9	Витрата	стаття	2	2025-03-20 00:00:00	читання
9	10	Витрата	Інтернет	2	2025-03-17 00:00:00	пошук
10	11	Витрата	книга	7	2025-04-02 00:00:00	читання
11	12	Ресурс	Java	3	2025-04-22 00:00:00	дисциплі...

Рисунок 3.13 – Таблиця даних завдань неформальної освіти бази даних SQLite

Як видно з наведених ілюстрацій база даних SQLite цілком підходить для реалізації проєкту тайм-менеджменту неформальної освіти в web-середовищі. Мова програмування C# має всі необхідні засоби для доступу та управління базою даних.

SQLite використовує такий підхід: його не потрібно встановлювати і не потрібно налаштовувати. Це бібліотека, яка керує даними безпосередньо з файлової системи. SQLite удвічі-втричі швидший за MySQL для звичайних запитів, SQLite може бути в 60 разів швидшим за будь-якого менеджера баз даних. Щоб створити базу даних у SQLite, необхідно, щоб на вашому комп'ютері просто було встановлено SQLite.

3.3 Розробка основних програмних модулів автоматизованої системи

Розробка основних програмних модулів мовою програмування C# проводили з урахуванням методів побудови та функціонування візуального інтерфейсу в web-програмі. Запропоновані рішення стосуються елементів керування, компонентів і деяких їхніх властивостей, подій і деяких можливостей C#. На рис. 3.14 показано фрагмент основного модуля відображення стартового заголовка програми.

```
<div class="jumbotron">
    <h1 class="display-4">Планування часу неформальної освіти!</h1>
    <p class="lead">АВТОМАТИЗАЦІЯ ТАЙМ-МЕНЕДЖМЕНТУ НЕФОРМАЛЬНОЇ ОСВІТИ.</p>
    <hr class="my-4">
    <p>Використовуйте навігаційне меню для управління та перегляду звітів.</p>
    <a class="btn btn-primary btn-lg" asp-page="/Transactions/Index" role="button">Перейти до занять</a>
</div>
```

Рисунок 3.14 – Модуль відображення стартового заголовка програми

На рис. 3.15 показано лістинг підключення скриптів обробки подій форми списку занять.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - Osvita</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css" crossorigin="anonymous"
    <script type="importmap"></script>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/FinanceOblik.styles.css" asp-append-version="true" />
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
            <a class="navbar-brand" asp-page="/Index">     Osvita</a>
            <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
                    aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav">
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Index">Головна</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Index">Заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Create">Додати заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Reports/Index">Звіти</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/EfficiencyAnalysis/EfficiencyAnalysis">Аналіз занять</a>
                    </li>
                </ul>
            </div>
        </nav>
    </header>
```

Рисунок 3.15 – Модуль скриптів обробки подій форми занять

```
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <footer class="border-top footer text-muted">
        <div class="container">
            &copy; 2025 - Osvita (Я.А. Романенко ) - <a asp-area="" asp-page="/Privacy">Privacy</a>
        </div>
    </footer>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.2/jquery.validate.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate-unobtrusive/3.2.11/jquery.validate.unobtrusive.min.js"></script>
    @RenderSection("Scripts", required: false)

    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chartjs-plugin-datalabels"></script>


    @* @await RenderSectionAsync("Scripts", required: false) *@
</body>
</html>
```

Рисунок 3.15 – Модуль скриптів обробки подій форми занять (продовження)

У кожному візуальному проєкті представлено елементи управління, методи рішення, концепції, що відповідають IDE і мові C#. Програма, що розробляється, має послідовну і розгалужену структуру, в якій потік виконання є єдиним. На рис. 3.16 показано модуль web-форми обробки операцій над заняттями.

```
@page
@model FinanceOblik.Pages.Transactions.CreateModel
@{
    ViewData["Title"] = "Додати заняття";
}

<h1>Додати заняття</h1>

<form method="post">
    <div class="mb-3">
        <label asp-for="Transaction.Type" class="form-label"></label>
        <select asp-for="Transaction.Type" asp-items="Model.TypeSelectList" class="form-select" required id="Transaction_Type">
            <option value="">Виберіть тип</option>
        </select>
        <span asp-validation-for="Transaction.Type" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Category" class="form-label"></label>
        <select asp-for="Transaction.Category" asp-items="Model.CategorySelectList" class="form-select" required id="Transaction_Category">
            <option value="">Виберіть категорію</option>
        </select>
        <span asp-validation-for="Transaction.Category" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Amount" class="form-label"></label>
        <input asp-for="Transaction.Amount" class="form-control" required />
        <span asp-validation-for="Transaction.Amount" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Date" class="form-label"></label>
        <input asp-for="Transaction.Date" class="form-control" type="date" required />
        <span asp-validation-for="Transaction.Date" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Description" class="form-label"></label>
        <textarea asp-for="Transaction.Description" class="form-control" rows="3"></textarea>
        <span asp-validation-for="Transaction.Description" class="text-danger"></span>
    </div>
    <button type="submit" class="btn btn-primary">Додати</button>
    <a asp-page="./Index" class="btn btn-secondary">Скасувати</a>
</form>
```

Рисунок 3.16 – Модуль web-форми обробки операцій над заняттями

На рис. 3.17 показано модуль форми відображення категорій занять.

```
@section Scripts {
    <script>
        document.addEventListener("DOMContentLoaded", function () {
            const typeSelect = document.getElementById("Transaction_Type");
            const categorySelect = document.getElementById("Transaction_Category");

            const categories = {
                "Дохід": ["БД", "ООП", "Java", "Мережі", "ОС", "C++", "СППР"],
                "Витрата": ["книга", "відео", "довідник", "стаття", "Internet"]
            };

            typeSelect.addEventListener("change", function () {
                const selectedType = this.value;
                categorySelect.innerHTML = '<option value="">Виберіть категорію</option>';

                if (categories[selectedType]) {
                    categories[selectedType].forEach(function (category) {
                        const option = document.createElement("option");
                        option.value = category;
                        option.text = category;
                        categorySelect.appendChild(option);
                    });
                }
            });
        });
    </script>
}
```

Рисунок 3.17 – Модуль форми відображення категорій занять

3.4 Розробка програмних модулів аналізу тайм-менеджменту

Після накопичення даних щодо витрат часу на різні види неформальної освіти необхідна можливість проводити їх опрацювання та аналіз. Тому в програмному комплексі було розроблено модуль аналізу результатів планування тайм-менеджменту. На рисунку 3.18 показано форму аналізу результатів планування тайм-менеджменту.

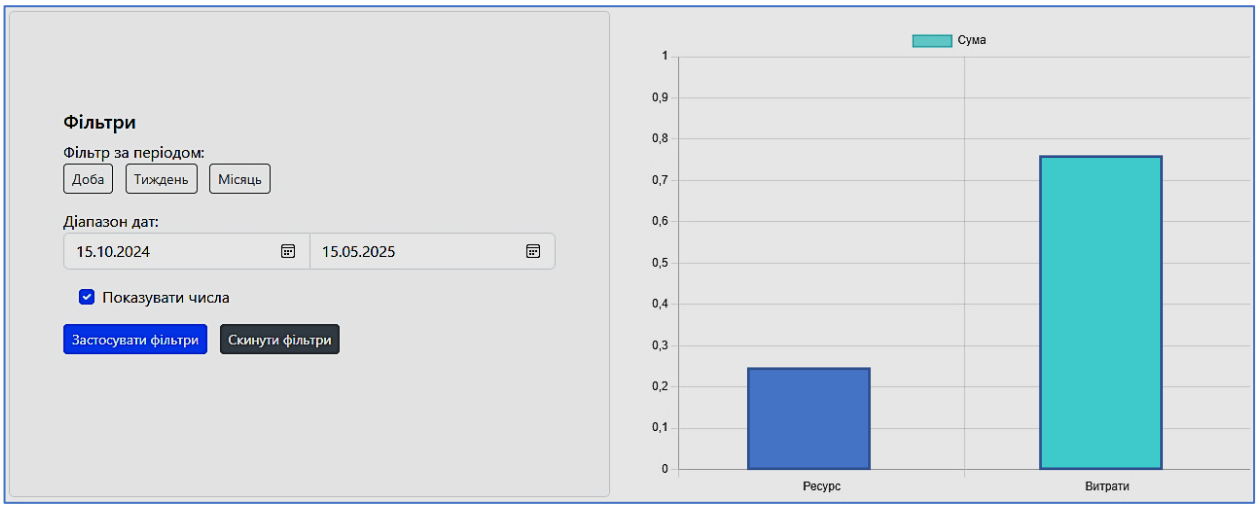


Рисунок 3.18 – Форма аналізу результатів планування тайм-менеджменту

На рис. 3.19 показано лістинг програмного коду відображення результатів ефективності тайм-менеджменту.

```
@section Scripts {
    <!-- Підключення Chart.js (версія 3.9.1) -->
    <script src="https://cdn.jsdelivr.net/npm/chart.js@3.9.1/dist/chart.min.js"></script>
    <!-- Підключення плагіна DataLabels -->
    <script src="https://cdn.jsdelivr.net/npm/chartjs-plugin-datalabels"></script>
    <script>
        document.addEventListener("DOMContentLoaded", function () {
            var ctx = document.getElementById('efficiencyChart').getContext('2d');
            var chartType = 'bar'; // Тип графіка - стовпчиковий
            var chart;

            function generateChart(totalIncome, totalExpenses, type, showLabels) {
                if (chart) {
                    chart.destroy(); // Знищити старий графік
                }

                var data = {
                    labels: ['Ресурс', 'Витрати'],
                    datasets: [{
                        label: 'Сума',
                        data: [totalIncome, totalExpenses],
                        backgroundColor: [
                            'rgba(75, 192, 192, 0.6)', // Колір для доходів
                            'rgba(255, 99, 132, 0.6)'  // Колір для витрат
                        ],
                        borderColor: [
                            'rgba(75, 192, 192, 1)',
                            'rgba(255, 99, 132, 1)'
                        ],
                        borderWidth: 1
                    }]
                };
```

```
            var options = {
                responsive: true,
                maintainAspectRatio: false,
                plugins: {
                    datalabels: {
                        color: '#000', // Чорний колір чисел
                        anchor: 'end', // Закріпити до кінця стовпчика
                        align: 'start', // Вирівняти підпис всередині стовпчика
                        offset: -10, // Підняти підпис над стовпчиком
                        clip: false, // Дозволяє підписам виходити за межі графіка
                        formatter: function (value) {
                            return value === 0 ? '' : value.toLocaleString(); // Не відображати "0"
                        },
                        display: showLabels, // Відображати або приховувати підписи
                        font: {
                            size: 12,
                            weight: 'bold',
                        }
                    },
                    legend: {
                        display: true // Відключити легенду
                    },
                    tooltip: {
                        enabled: true
                    }
                },
                scales: {
                    y: {
                        beginAtZero: true,
                        ticks: {
                            callback: function (value) {
                                return value.toLocaleString();
                            }
                        }
                    }
                }
            };
```

Рисунок 3.19 – Лістинг програмного коду відображення результатів ефективності тайм-менеджменту

```
        // Отримання даних з серверної сторони
        var totalIncome = @Html.Raw(JsonSerializer.Serialize(Model.TotalIncome));
        var totalExpenses = @Html.Raw(JsonSerializer.Serialize(Model.TotalExpenses));

        console.log('Total Income:', totalIncome);
        console.log('Total Expenses:', totalExpenses);

        // Показувати підписи за замовчуванням
        var showDataLabels = true;

        // Генерація графіка при завантаженні сторінки
        generateChart(totalIncome, totalExpenses, chartType, showDataLabels);

        // Обробка чекбоксу для відображення/приховування чисел
        var toggleDataLabelsCheckbox = document.getElementById('toggleDataLabels');
        if (toggleDataLabelsCheckbox) {
            toggleDataLabelsCheckbox.addEventListener('change', function () {
                showDataLabels = this.checked;
                generateChart(totalIncome, totalExpenses, chartType, showDataLabels);
            });
        }

        // Глобальна функція для зміни періоду та відправки форми
        window.submitPeriod = function(periodType) {
            console.log('Submitting period:', periodType); // Для перевірки
            var periodInput = document.querySelector('input[name="Period"]');
            if (periodInput) {
                periodInput.value = periodType;
                document.getElementById('filterForm').submit();
            } else {
                console.error('Hidden input for Period not found.');
            }
        };
    });
</script>
```

Рисунок 3.19 – Лістинг програмного коду відображення результатів ефективності тайм-менеджменту (продовження)

3.5 Розробка програмних модулів звітів результатів неформальної освіти

Результатом роботи програмного комплексу тайм-менеджменту вищої освіти є узагальнені дані. Ці дані мають бути відображені у вигляді спеціальних форм, які називаються звітами. У звітах особливу роль відіграє деталізація інтерфейсу. Розбірливість інтерфейсу дуже важлива і необхідна для зручності використання. Якщо символи або об'єкти, що відображаються на екрані, неможливо розрізнити, то користувач не зможе ефективно використовувати їх.

Розглянемо форму фільтрації даних для їх відображення у звіті. На рис. 3.20 показано форму встановлення параметрів фільтрів для звітів результатів неформальної освіти.

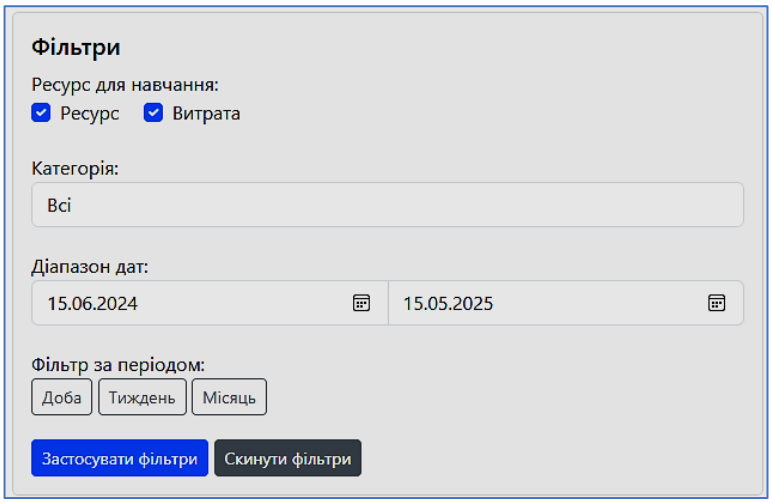


Рисунок 3.20 – Форма встановлення параметрів фільтрів для звітів результатів неформальної освіти

На рис. 3.21 показано лістинг програмного коду форми встановлення параметрів фільтрів для звітів результатів неформальної освіти.

```
<!-- Права частина: Фільтри -->
<div class="reports-right">
    <div class="filter-container">
        <h5>Фільтри</h5>
        <form method="get" id="filterForm">
            <!-- Фільтр за типом транзакції -->
            <div class="filter-section">
                <label>Ресурс для навчання:</label><br />
                <div class="form-check form-check-inline">
                    <input class="form-check-input" type="checkbox" name="FilterTypes"
                    checked value="Дохід" @(Model.FilterTypes.Contains("Дохід") ? "checked" : "")>
                    <label class="form-check-label">Ресурс</label>
                </div>
                <div class="form-check form-check-inline">
                    <input class="form-check-input" type="checkbox" name="FilterTypes"
                    checked value="Витрата" @(Model.FilterTypes.Contains("Витрата") ? "checked" : "")>
                    <label class="form-check-label">Витрата</label>
                </div>
            </div>

            <!-- Фільтр за категорією -->
            <div class="filter-section">
                <label>Категорія:</label>
                <select asp-for="FilterCategory" asp-items="Model.CategorySelectList" class="form-control select2">
                    <option value="">Всі</option>
                </select>
            </div>

            <!-- Фільтр за діапазоном дат -->
            <div class="filter-section">
                <label>Діапазон дат:</label>
                <div class="input-group">
                    <input type="date" name="StartDate" value="@(Model.StartDate?.ToString("yyyy-MM-dd"))" class="form-control">
                    <input type="date" name="EndDate" value="@(Model.EndDate?.ToString("yyyy-MM-dd"))" class="form-control">
                </div>
            </div>
```

Рисунок 3.21 – Лістинг програмного коду форми встановлення параметрів фільтрів для звітів результатів неформальної освіти

```
                <!-- Фільтри за періодом -->
                <div class="filter-section">
                    <label>Фільтр за періодом:</label><br />
                    <button type="button" onclick="changeChartType('day')" class="btn btn-outline-secondary btn-sm">Доба</button>
                    <button type="button" onclick="changeChartType('week')" class="btn btn-outline-secondary btn-sm">Тиждень</button>
                    <button type="button" onclick="changeChartType('month')" class="btn btn-outline-secondary btn-sm">Місяць</button>
                </div>

                <!-- Кнопки для застосування або скидання фільтрів -->
                <div class="filter-section">
                    <button type="submit" class="btn btn-primary btn-sm">Застосувати фільтри</button>
                    <a href="@Url.Page("/Reports/Index")" class="btn btn-secondary btn-sm">Скинути фільтри</a>
                </div>
            </form>
        </div>
    </div>
</div>
```

Рисунок 3.21 – Лістинг програмного коду форми встановлення параметрів фільтрів для звітів результатів неформальної освіти (продовження)

Мова програмування C# у поєднанні з простотою додатків Windows Forms у Visual Studio робить розробку Windows-програм швидшою і простішою. Visual C# – це універсальний і гнучкий інструмент, що дає змогу створювати складні графіки, діаграми та графічні користувацькі інтерфейси (GUI). Рівень складності та витонченості графічних і діаграмних додатків обмежується тільки потребами та уявою.

На рис. 3.22 показано аналітичні форми відображення звітів аналізу тайм-менеджменту занять неформальної освіти.

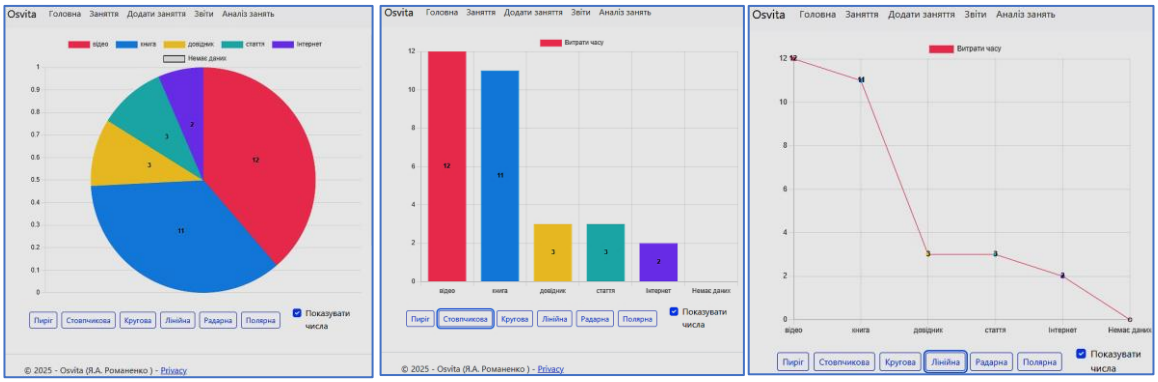


Рисунок 3.22 – Аналітичні форми відображення звітів аналізу тайм-менеджменту занять неформальної освіти

C# – це еталонна мова для .NET, оскільки її було розроблено з нуля для роботи на новій платформі. Це проста і безпечна мова програмування, тому її легко вивчити, і вона добре працює. Відповідним середовищем для реалізації

коду на C# є Visual Studio .NET, комплексний інструмент для швидкої, простої та безпечної розробки додатків. Оскільки він розроблений на платформі .NET, він поділяє її безпеку та надійність.

Графічна підсистема C# складається з двох основних підсистем: графічної моделі та конвеєра візуалізації. Графічна модель утворює абстрактний шар навколо графічної мови для забезпечення крос-платформної переносимості. Зберігаючи абстрактний шар, можна оновлювати графічну підсистему за допомогою нових технологій, не порушуючи його сумісності. Назви класів у графічній моделі було адаптовано для діаграм. Усі графічні образи - це класи, за допомогою яких програміст створює сцену для відображення.

Отже, було підтверджено, що мова програмування C# у середовищі розробки Visual Studio .NET цілком реалізує всі функції для побудови сучасних систем візуалізації даних.

ВИСНОВКИ

Була виконана випускна кваліфікаційна робота: "Розробка програмного проєкту автоматизації тайм-менеджменту завдань неформальної освіти». Робота присвячена актуальній темі - розробці автоматизованої комп'ютерної системи для тайм-менеджменту занять неформальної освіти. Актуальність теми кваліфікаційної роботи полягає у зростаючій важливості самостійної роботи студентів над навчальним матеріалом, у пошуку та систематизації тематичних блоків для дисциплін, що вивчаються. Під час здійснення кваліфікаційної роботи було спроєктовано та реалізовано програму web-застосунку для обліку та подальшого аналізу специфіки та витрат часу на заняття неформальної освіти.

Під час виконання кваліфікаційної роботи були отримані наступні результати:

— проаналізовано проблемну область кваліфікаційної роботи, сформульовано завдання роботи. Проведений аналіз рівнів та результатів навчання вищої освіти України, структура неформальної освіти, процедура зарахування результатів неформальної освіти, порівняльний аналіз формальної та неформальної освіти, аналіз існуючих програмних систем підтримки тайм-менеджменту. Зазначено, що питання методології визначення результатів неформальної освіти потребує уваги, оскільки це одне з найслабших місць у неформальній освіті;

— розроблені математичні моделі тайм-менеджменту завдань неформальної освіти. Проаналізовано графік залежності результатів роботи від витраченого часу. При розробці математичних моделей використано метод PERT. Для перелічених в роботі видів діаграм визначено загальний підхід для обчислення параметрів моделей: розробка розкладу, інструменти і методи, математичний аналіз, формули методу PERT. На основі розрахунків побудований графік розподілу ймовірності тривалості діяльності програмного проекту загалом;

— розроблено структурні та функціональні моделі програмного комплексу. Визначено взаємозв'язок завдань загального підходу до формування структурної моделі програмного проекту. Обрано програмні технології для реалізації проекту. Розроблена структура бази даних системи тайм-менеджменту неформальної освіти;

— розроблено блок-схеми алгоритмів програмних модулів системи тайм-менеджменту. Розроблено модель візуального інтерфейсу користувача програмного комплексу. Була використана концепція сучасного дизайну, орієнтована на користувача (UCD). Було розроблено візуальні форми, поля введення даних, візуальні елементи управління;

— розроблені програмні модулі візуального інтерфейсу системи автоматизації тайм-менеджменту, основні модулі, які реалізують логіку і бізнес-процеси автоматизованої системи, модулі для вибірки і фільтрації даних, які відображаються на формах, модулі для безпосереднього управління заняттями в базі даних, розроблена база даних для обліку занять неформальної освіти, програмні модулі аналізу тайм-менеджменту, програмні модулі звітів результатів неформальної освіти.

Розроблена автоматизована web-орієнтована система тайм-менеджменту неформальної освіти дає змогу підвищити ефективність пошуку, обліку та аналізу занять неформальної освіти. Система забезпечує можливості статистичного аналізу кількості проведених занять, витраченого часу, тематичної специфіки конкретних дисциплін, надає багаті графічні засоби для формування та публікації аналітичних звітів.

На основі отриманих результатів можна зробити висновок, що виконана робота характеризує актуальність теми, демонструє ефективність розроблених програмних модулів.

Результати роботи можуть бути використані для організації, проведення, обліку результатів та оцінювання самостійного неформального навчання студентів за обраними спеціальностями.

ПЕРЕЛІК ПОСИЛАНЬ

1. Закон України Про вищу освіту (Відомості Верховної Ради (ВВР), 2014. № 37- 38, ст. 2004)

2. Лукаш А.О. Університетська освіта : конспект лекцій [Текст]. / А.О.Лукаш – Суми, Сумський державний університет, 2011 – [Електронний варіант]

3. Наказ Міністерства освіти і науки України від 08 лютого 2022 року №130 Про затвердження Порядку визнання у вищій та фаховій передвищій освіті результатів навчання, здобутих шляхом неформальної та/або інформальної освіти

4. Положення про порядок визнання у Криворізькому національному університеті результатів навчання, отриманих в умовах неформальної та/або інформальної освіти, Кривий ріг, 2023 р.

5. Тайм-менеджмент: навч. посіб. для закладів вищ. освіти / МОН України, Уманський держ. пед. ун-т імені Павла Тичини; уклад. Н.М. Малярчук. – Умань : Видавець «Сочінський М. М.», 2024. - 175 с.

6. Barbosa, Christian A tríade do tempo [recurso eletrônico] / Christian Barbosa; Rio de Janeiro: Sextante, 2012 – 225p.

7. Kliem, R. L., & Anderson, M. Project management and time management: A synthesis of perspectives. International Journal of Project Management, 11(4), 2023, 211–217.

8. Barrow, C., Nejad, M. The importance of time management in project success. International Journal of Management Studies, 25(3), 2018, 456–478.

9. Що таке PERT? [Електронний ресурс] Режим доступу URL: https://experience.dropbox.com/uk- ua/resources/pert

10. Блага Н. В. Управління проєктами : навч. посібник. Львів : Львівський державний університет внутрішніх справ, 2021. 152 с.

11. Коноваленко І.В. Платформа .NET та мова програмування C# 8.0: навчальний посібник / Коноваленко І.В., Марущак П.О. – Тернопіль: ФОП Паляниця В. А., 2020 – 320 с.

12. Івохін Є.В., Махно М.Ф., Піскунов, О.Г. Розробка додатків засобами мови програмування C#: Навч.-метод. посібник для проведення лабораторних робіт для студентів вищих навчальних закладів спеціальності «Системний аналіз» /Є.В. Івохін, М.Ф. Махно, О.Г. Піскунов. – К.: Видавничо-поліграфічний центр "Київський університет", 2021. – 100 с.
13. Шарп Дж. (2023). Microsoft C# Step by Step. Microsoft Press. – 750с.
14. Вербіцький В. В. Бази даних та інформаційні системи: метод. вказівки. Одеса : Одес. нац. ун-т ім. І.І. Мечникова, 2022. 82 с.
15. Coronel C., Morris S. Database Systems: Design, Implementation, & Management. 13th Edition. Boston : Cengage Learning, 2018. – 800 p.
16. Лавренчук С. В., Коцюба А. Ю. Бази даних та мова SQL: конспект лекцій. Луцьк : Луцький НТУ, 2016. 76 с.
17. Ушенко Ю.О., Олар О.В., Галочкін О.В., Д’яченко Л.І. Сучасні технології розробки web-додатків: Фронтенд розробка: Навч. посібник / Ушенко Ю.О., Олар О.В., Галочкін О.В., Д’яченко Л.І. – Чернівці: Чернівецький нац. ун-т, 2022. – 222 с.
18. Мудрик І. Проблеми аналізу унікальності контенту веб-сайтів у роботі SEO-оптимізатора / І. Мудрик, Р. Карагодін., 2023. – 161 с. (ТНТУ).
19. Flanagan D. JavaScript: The Definitive Guide: Master the World's Most-Used Programming Language / David Flanagan. – 704 с.
20. Пасічник О. В. Веб-дизайн. / О. В. Пасічник, В. В. Пасічник. – Львів: Магнолія, 2019. – 520 с
21. Freeman, A., Robson, E., & Bates, B. (2020). Head First JavaScript Programming: A Brain-Friendly Guide (2nd ed.). O'Reilly Media.

ДОДАТОК А

Основні лістинги програмних модулів web-застосунку автоматизованої системи тайм-менеджменту неформальної освіти

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - Osvita</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css"
        crossorigin="anonymous" referrerpolicy="no-referrer" />
    <script type="importmap"></script>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/FinanceOblik.styles.css" asp-append-version="true" />
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
            <a class="navbar-brand" asp-page="/Index">     Osvita</a>
            <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
                    aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav">
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Index">Головна</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Index">Заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Create">Додати заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Reports/Index">Звіти</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/EfficiencyAnalysis/EfficiencyAnalysis">Аналіз занять</a>
                    </li>
                </ul>
            </div>
        </nav>
    </header>
```

```
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <footer class="border-top footer text-muted">
        <div class="container">
            &copy; 2025 - Osvita (Я.А. Романенко ) - <a asp-area="" asp-page="/Privacy">Privacy</a>
        </div>
    </footer>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.2/jquery.validate.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate-unobtrusive/3.2.11/jquery.validate.unobtrusive.min.js"></script>
    @RenderSection("Scripts", required: false)

    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chartjs-plugin-datalabels"></script>


    @* @await RenderSectionAsync("Scripts", required: false) *@
</body>
</html>
```

```
@{
    ViewData["Title"] = "Список занять";
}
<p></p>
<h2>Список занять неформальної освіти</h2>

<div class="row">
    <div class="col-md-8">
        <table class="table table-striped">
            <thead>
                <tr>
                    <th>Ресурс для навчання</th>
                    <th>Категорія</th>
                    <th>Тривалість</th>
                    <th>Дата</th>
                    <th>Опис</th>
                    <th>Дії</th>
                </tr>
            </thead>
            <tbody>
                @foreach (var item in Model.Transactions)
                {
                    <tr>
                        <td>
                            @if (item.Type == "Ресурс")
                            {
                                <span class="badge bg-success">Є матеріал</span>
                            }
                            else
                            {
                                <span class="badge bg-danger">Потрібен час</span>
                            }
                        </td>
                        <td>
                            <i class="@Model.GetCategoryIcon(item.Category)"></i> @item.Category
                        </td>
```

```
                        <td>@item.Amount.ToString("N2")</td>
                        <td>@item.Date.ToString("dd.MM.yyyy")</td>
                        <td>@item.Description</td>
                        <td>
                            <a asp-page="./Edit" asp-route-id="@item.Id" class="btn btn-sm btn-primary">Редагувати</a>
                            <a asp-page="./Delete" asp-route-id="@item.Id" class="btn btn-sm btn-danger">Видалити</a>
                        </td>
                    </tr>
                }
            </tbody>
        </table>
    </div>
    <div class="filter-container col-md-4">
        <h4>Фільтри</h4>
        <form method="get">
            <div class="mb-3">
                <label>Ресурс для навчання:</label><br />
                @foreach (var type in Model.TypeSelectList)
                {
                    <div class="form-check form-check-inline">
                        <input class="form-check-input" type="checkbox" name="FilterTypes"
                            value="@type.Value" @(Model.FilterTypes.Contains(type.Value) ? "checked" : "")>
                        <label class="form-check-label">@type.Text</label>
                    </div>
                }
            </div>
            <div class="mb-3">
                <label asp-for="FilterCategory" class="form-label">Категорія:</label>
                <select asp-for="FilterCategory" asp-items="Model.CategorySelectList" class="form-select">
                    <option value="">Всі</option>
                </select>
            </div>
            <div class="mb-3">
                <label>Діапазон дат:</label>
                <div class="input-group">
                    <input asp-for="StartDate" class="form-control" type="date" />
                    <input asp-for="EndDate" class="form-control" type="date" />
                </div>
            </div>
```

```
            <div class="mb-3">
                <label>Фільтр за період:</label><br />
                <button type="submit" name="period" value="day" class="btn btn-outline-secondary">Доба</button>
                <button type="submit" name="period" value="week" class="btn btn-outline-secondary">Тиждень</button>
                <button type="submit" name="period" value="month" class="btn btn-outline-secondary">Місяць</button>
            </div>
            <button type="submit" class="btn btn-primary">Застосувати фільтри</button>
            <a asp-page="./Index" class="btn btn-secondary">Скинути фільтри</a>
        </form>
    </div>
</div>

<a asp-page="./Create" class="btn btn-success mt-3">Додати заняття</a>
```

```
    <div class="filter-container col-md-4">
        <h4>Фільтри</h4>
        <form method="get">
            <div class="mb-3">
                <label>Ресурс для навчання:</label><br />
                @foreach (var type in Model.TypeSelectList)
                {
                    <div class="form-check form-check-inline">
                        <input class="form-check-input" type="checkbox" name="FilterTypes"
                            value="@type.Value" @(Model.FilterTypes.Contains(type.Value) ? "checked" : "")>
                        <label class="form-check-label">@type.Text</label>
                    </div>
                }
            </div>
            <div class="mb-3">
                <label asp-for="FilterCategory" class="form-label">Категорія:</label>
                <select asp-for="FilterCategory" asp-items="Model.CategorySelectList" class="form-select">
                    <option value="">Всі</option>
                </select>
            </div>
            <div class="mb-3">
                <label>Діапазон дат:</label>
                <div class="input-group">
                    <input asp-for="StartDate" class="form-control" type="date" />
                    <input asp-for="EndDate" class="form-control" type="date" />
                </div>
            </div>
            <div class="mb-3">
                <label>Фільтр за період:</label><br />
                <button type="submit" name="period" value="day" class="btn btn-outline-secondary">Доба</button>
                <button type="submit" name="period" value="week" class="btn btn-outline-secondary">Тиждень</button>
                <button type="submit" name="period" value="month" class="btn btn-outline-secondary">Місяць</button>
            </div>
            <button type="submit" class="btn btn-primary">Застосувати фільтри</button>
            <a asp-page="./Index" class="btn btn-secondary">Скинути фільтри</a>
        </form>
    </div>
</div>
```

```
@section Scripts {
    <script>
        document.addEventListener("DOMContentLoaded", function () {
            const typeSelect = document.getElementById("Transaction_Type");
            const categorySelect = document.getElementById("Transaction_Category");

            const categories = {
                "Дохід": ["БД", "ООП", "Java", "Мережі", "ОС", "C++", "СППР"],
                "Витрата": ["книга", "відео", "довідник", "стаття", "Internet"]
            };

            typeSelect.addEventListener("change", function () {
                const selectedType = this.value;
                categorySelect.innerHTML = '<option value="">Виберіть категорію</option>';

                if (categories[selectedType]) {
                    categories[selectedType].forEach(function (category) {
                        const option = document.createElement("option");
                        option.value = category;
                        option.text = category;
                        categorySelect.appendChild(option);
                    });
                }
            });
        });
    </script>
}
```

```
<h1>Додати заняття</h1>

<form method="post">
    <div class="mb-3">
        <label asp-for="Transaction.Type" class="form-label"></label>
        <select asp-for="Transaction.Type" asp-items="Model.TypeSelectList"
        class="form-select" required id="Transaction_Type">
            <option value="">Виберіть тип</option>
        </select>
        <span asp-validation-for="Transaction.Type" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Category" class="form-label"></label>
        <select asp-for="Transaction.Category" asp-items="Model.CategorySelectList"
        class="form-select" required id="Transaction_Category">
            <option value="">Виберіть категорію</option>
        </select>
        <span asp-validation-for="Transaction.Category" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Amount" class="form-label"></label>
        <input asp-for="Transaction.Amount" class="form-control" required />
        <span asp-validation-for="Transaction.Amount" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Date" class="form-label"></label>
        <input asp-for="Transaction.Date" class="form-control" type="date" required />
        <span asp-validation-for="Transaction.Date" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Description" class="form-label"></label>
        <textarea asp-for="Transaction.Description" class="form-control" rows="3"></textarea>
        <span asp-validation-for="Transaction.Description" class="text-danger"></span>
    </div>
    <button type="submit" class="btn btn-primary">Додати</button>
    <a asp-page="./Index" class="btn btn-secondary">Скасувати</a>
</form>
```

```
<div class="jumbotron">
    <h1 class="display-4">Планування часу неформальної освіти!</h1>
    <p class="lead">АВТОМАТИЗАЦІЯ ТАЙМ-МЕНЕДЖМЕНТУ НЕФОРМАЛЬНОЇ ОСВІТИ.</p>
    <hr class="my-4">
    <p>Використовуйте навігаційне меню для управління та перегляду звітів.</p>
    <a class="btn btn-primary btn-lg" asp-page="/Transactions/Index" role="button">Перейти до занять</a>
</div>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - Osvita</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css" crossorigin="anonymous"
    <script type="importmap"></script>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/FinanceOblik.styles.css" asp-append-version="true" />
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
            <a class="navbar-brand" asp-page="/Index">     Osvita</a>
            <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
                    aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav">
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Index">Головна</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Index">Заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Create">Додати заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Reports/Index">Звіти</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/EfficiencyAnalysis/EfficiencyAnalysis">Аналіз занять</a>
                    </li>
                </ul>
            </div>
        </nav>
    </header>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - Osvita</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css" crossorigin="anonymous"
    <script type="importmap"></script>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/FinanceOblik.styles.css" asp-append-version="true" />
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
            <a class="navbar-brand" asp-page="/Index">     Osvita</a>
            <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
                    aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav">
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Index">Головна</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Index">Заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Transactions/Create">Додати заняття</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/Reports/Index">Звіти</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" asp-page="/EfficiencyAnalysis/EfficiencyAnalysis">Аналіз занять</a>
                    </li>
                </ul>
            </div>
        </nav>
    </header>
```

```
@page
@model FinanceOblik.Pages.Transactions.CreateModel
@{
    ViewData["Title"] = "Додати заняття";
}

<h1>Додати заняття</h1>

<form method="post">
    <div class="mb-3">
        <label asp-for="Transaction.Type" class="form-label"></label>
        <select asp-for="Transaction.Type" asp-items="Model.TypeSelectList" class="form-select" required id="Transaction_Type">
            <option value="">Виберіть тип</option>
        </select>
        <span asp-validation-for="Transaction.Type" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Category" class="form-label"></label>
        <select asp-for="Transaction.Category" asp-items="Model.CategorySelectList" class="form-select" required id="Transaction_Category">
            <option value="">Виберіть категорію</option>
        </select>
        <span asp-validation-for="Transaction.Category" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Amount" class="form-label"></label>
        <input asp-for="Transaction.Amount" class="form-control" required />
        <span asp-validation-for="Transaction.Amount" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Date" class="form-label"></label>
        <input asp-for="Transaction.Date" class="form-control" type="date" required />
        <span asp-validation-for="Transaction.Date" class="text-danger"></span>
    </div>
    <div class="mb-3">
        <label asp-for="Transaction.Description" class="form-label"></label>
        <textarea asp-for="Transaction.Description" class="form-control" rows="3"></textarea>
        <span asp-validation-for="Transaction.Description" class="text-danger"></span>
    </div>
    <button type="submit" class="btn btn-primary">Додати</button>
    <a asp-page="./Index" class="btn btn-secondary">Скасувати</a>
</form>
```

```
@section Scripts {
    <!-- Підключення Chart.js (версія 3.9.1) -->
    <script src="https://cdn.jsdelivr.net/npm/chart.js@3.9.1/dist/chart.min.js"></script>
    <!-- Підключення плагіна DataLabels -->
    <script src="https://cdn.jsdelivr.net/npm/chartjs-plugin-datalabels"></script>
    <script>
        document.addEventListener("DOMContentLoaded", function () {
            var ctx = document.getElementById('efficiencyChart').getContext('2d');
            var chartType = 'bar'; // Тип графіка - стовпчиковий
            var chart;

            function generateChart(totalIncome, totalExpenses, type, showLabels) {
                if (chart) {
                    chart.destroy(); // Знищити старий графік
                }

                var data = {
                    labels: ['Ресурс', 'Витрати'],
                    datasets: [{
                        label: 'Сума',
                        data: [totalIncome, totalExpenses],
                        backgroundColor: [
                            'rgba(75, 192, 192, 0.6)', // Колір для доходів
                            'rgba(255, 99, 132, 0.6)'  // Колір для витрат
                        ],
                        borderColor: [
                            'rgba(75, 192, 192, 1)',
                            'rgba(255, 99, 132, 1)'
                        ],
                        borderWidth: 1
                    }]
                };
```

```
            var options = {
                responsive: true,
                maintainAspectRatio: false,
                plugins: {
                    datalabels: {
                        color: '#000', // Чорний колір чисел
                        anchor: 'end', // Закріпити до кінця стовпчика
                        align: 'start', // Вирівняти підпис всередині стовпчика
                        offset: -10, // Підняти підпис над стовпчиком
                        clip: false, // Дозволяє підписам виходити за межі графіка
                        formatter: function (value) {
                            return value === 0 ? '' : value.toLocaleString(); // Не відображати "0"
                        },
                        display: showLabels, // Відображати або приховувати підписи
                        font: {
                            size: 12,
                            weight: 'bold',
                        }
                    },
                    legend: {
                        display: true // Відключити легенду
                    },
                    tooltip: {
                        enabled: true
                    }
                },
                scales: {
                    y: {
                        beginAtZero: true,
                        ticks: {
                            callback: function (value) {
                                return value.toLocaleString();
                            }
                        }
                    }
                }
            };
```

```
        // Отримання даних з серверної сторони
        var totalIncome = @Html.Raw(JsonSerializer.Serialize(Model.TotalIncome));
        var totalExpenses = @Html.Raw(JsonSerializer.Serialize(Model.TotalExpenses));

        console.log('Total Income:', totalIncome);
        console.log('Total Expenses:', totalExpenses);

        // Показувати підписи за замовчуванням
        var showDataLabels = true;

        // Генерація графіка при завантаженні сторінки
        generateChart(totalIncome, totalExpenses, chartType, showDataLabels);

        // Обробка чекбоксу для відображення/приховування чисел
        var toggleDataLabelsCheckbox = document.getElementById('toggleDataLabels');
        if (toggleDataLabelsCheckbox) {
            toggleDataLabelsCheckbox.addEventListener('change', function () {
                showDataLabels = this.checked;
                generateChart(totalIncome, totalExpenses, chartType, showDataLabels);
            });
        }

        // Глобальна функція для зміни періоду та відправки форми
        window.submitPeriod = function(periodType) {
            console.log('Submitting period:', periodType); // Для перевірки
            var periodInput = document.querySelector('input[name="Period"]');
            if (periodInput) {
                periodInput.value = periodType;
                document.getElementById('filterForm').submit();
            } else {
                console.error('Hidden input for Period not found.');
            }
        };
    });
</script>
```

```
<!-- Права частина: Фільтри -->
<div class="reports-right">
    <div class="filter-container">
        <h5>Фільтри</h5>
        <form method="get" id="filterForm">
            <!-- Фільтр за типом транзакції -->
            <div class="filter-section">
                <label>Ресурс для навчання:</label><br />
                <div class="form-check form-check-inline">
                    <input class="form-check-input" type="checkbox" name="FilterTypes"
                    checked value="Дохід" @(Model.FilterTypes.Contains("Дохід") ? "checked" : "")>
                    <label class="form-check-label">Ресурс</label>
                </div>
                <div class="form-check form-check-inline">
                    <input class="form-check-input" type="checkbox" name="FilterTypes"
                    checked value="Витрата" @(Model.FilterTypes.Contains("Витрата") ? "checked" : "")>
                    <label class="form-check-label">Витрата</label>
                </div>
            </div>

            <!-- Фільтр за категорією -->
            <div class="filter-section">
                <label>Категорія:</label>
                <select asp-for="FilterCategory" asp-items="Model.CategorySelectList" class="form-control select2">
                    <option value="">Всі</option>
                </select>
            </div>

            <!-- Фільтр за діапазоном дат -->
            <div class="filter-section">
                <label>Діапазон дат:</label>
                <div class="input-group">
                    <input type="date" name="StartDate" value="@(Model.StartDate?.ToString("yyyy-MM-dd"))" class="form-control">
                    <input type="date" name="EndDate" value="@(Model.EndDate?.ToString("yyyy-MM-dd"))" class="form-control">
                </div>
            </div>

                <!-- Фільтри за періодом -->
                <div class="filter-section">
                    <label>Фільтр за періодом:</label><br />
                    <button type="button" onclick="changeChartType('day')" class="btn btn-outline-secondary btn-sm">Доба</button>
                    <button type="button" onclick="changeChartType('week')" class="btn btn-outline-secondary btn-sm">Тиждень</button>
                    <button type="button" onclick="changeChartType('month')" class="btn btn-outline-secondary btn-sm">Місяць</button>
                </div>

                <!-- Кнопки для застосування або скидання фільтрів -->
                <div class="filter-section">
                    <button type="submit" class="btn btn-primary btn-sm">Застосувати фільтри</button>
                    <a href="@Url.Page("/Reports/Index")" class="btn btn-secondary btn-sm">Скинути фільтри</a>
                </div>
            </form>
        </div>
    </div>
</div>
```