

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ

к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "магістр"

на тему:

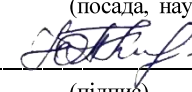
Виконав. студент групи 6141 м

 **Кравченко Р.К.**
(підпис)

Керівник роботи:

доцент кафедри ІУСТ, к.т.н.,

(посада, науковий ступінь вчене звання)

 **Казимиренко Ю.О.**
(підпис)

м. Миколаїв – 2024 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій
Спеціальність 122 "Комп'ютерні науки"
Освітня програма "Комп'ютерні науки"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми

В. Ажищев

к.т.н, доц. Ажищев В.Ф.

"_____" "_____" 2024 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ на здобуття ступеня вищої освіти "магістр"

Студенту Краченку Роману Костянтиновичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Дослідження оптимальних методів візуалізації технології
скловареного виробництва

Керівник роботи Казимиренко Юлія Олексіївна

Затверджені наказом ректора № 1055-уч. від "08" жовтня 2024 року.

2. Термін подання роботи: 15 грудня 2024 р.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні
джерела, технічна документація на існуючі аналоги систем, матеріали наукового
стажування

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

5. Перелік презентаційних матеріалів

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Казимиренко Ю.О.		
2	Казимиренко Ю.О.		
3	Казимиренко Ю.О.		
4	Казимиренко Ю.О.		
5	Казимиренко Ю.О.		
6	Казимиренко Ю.О.		

7. Дата видачі завдання 05 вересня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Дослідження та аналіз предметної галузі	22.02.2024	
2	Розробка концепції	15.04.2024	
3	Розділ з розробки проектних рішень по системі (та її частинах)	15.06.2024	
4	Реалізація проекту та розробка робочої документації	25.09.2024	
5	Обчислення економічної ефективності	14.11.2024	
6	Вирішення питань охорони праці та навколишнього середовища	14.11.2024	
7	Оформлення пояснювальної записки	14.11.2024	
8	Подання магістерської роботи на перевірку на плагіат	22.12.2024	
9	Подання магістерської роботи рецензенту	22.12.2024	
10	Подання магістерської роботи на державний захист	12.12.2024	

Студент

Керівник роботи


(підпис)


(підпис)

Кравченко Р.К.

(ПІБ)

Казимиренко Ю.О.

(ПІБ)

АНОТАЦІЯ

Кваліфікаційна робота «**Дослідження оптимальних методів візуалізації технології скловареного виробництва**» на здобуття вищої освіти «магістр» за спеціальністю 122 Комп'ютерні науки, освітня програма Комп'ютерні науки.

Кваліфікаційна робота присвячена вирішенню **важливого науково-практичного завдання** з підвищення ефективності візуалізації технологічних характеристик спеціальних стекол за допомогою нової інформаційної системи.

Мета роботи полягає у розробці нової інформаційної системи з вдосконаленим графічним відображенням систематизації експериментальних даних отримання стекол спеціального призначення.

Об'єкт дослідження – процеси впорядкування і аналізу експериментальних даних технологічного циклу отримання стекол спеціального призначення.

Предмет дослідження – математичні моделі кінетики спікання скляних мікросфер.

Наукова новизна роботи полягає у тому що вперше для технологічного циклу спікання скляних мікросфер запропоновано нові закономірності систематизації та візуалізації оброблених експериментальних даних, що дає змогу прогнозувати властивості отримуваних спецстекло.

Практична цінність роботи полягає в зручності інтерфейсу, швидкості фіксації експериментальних даних, наявності мобільного застосунок.

Методи дослідження: статистична обробка даних, графічна візуалізація даних, технології реляційних баз даних, мови програмування C# та Python.

Ключові слова: інформаційна система, візуалізація даних, графічний редактор, мобільний застосунок, стекла спеціального призначення.

ABSTRACT

Qualification work "**Research of optimal methods of visualization of the technology of glass production**" for obtaining a higher education "master" in the specialty 122 Computer sciences, educational program Computer sciences.

The qualification work is devoted to the solution of **an important scientific practical task** of increasing the efficiency of visualization of technological characteristics of special glasses using a new information system.

The purpose of the work is to develop a new information system with an improved graphical representation of the systematization of experimental data for the production of special-purpose glasses.

The object of the research is the process of arrangement and analysis of experimental data of the technological cycle of obtaining special-purpose glasses.

The subject of research is mathematical models of the kinetics of sintering of glass microspheres.

The scientific novelty of the work is that, for the first time, for the technological cycle of sintering of glass microspheres, new laws of systematization and visualization of the processed experimental data are proposed, which makes it possible to predict the properties of the obtained special glasses.

The practical value of the work lies in the convenience of the interface, the speed of recording experimental data, the presence of a mobile application.

Research methods: statistical data processing, graphical data visualization, relational database technologies, C# and Python programming languages.

Keywords: information system, data visualization, graphic editor, mobile application, special purpose glasses.

ЗМІСТ

ВСТУП.....	4
РОЗДІЛ 1. Сучасний стан і проблемні питання ІТ-забезпечення скловареної галузі	6
1.1 Загальна характеристика технологічного процесу виробництва скла і критерії його якості.....	6
1.2 Технологічний процес виробництва синтактичного піни та особливості виробництва синтактичного піноскла.....	9
1.3 Організація дослідницької лабораторії композитних матеріалів	12
1.4 Методологія постановки досліджень процесів отримання синтактичних піноскла	15
1.5 Методи візуалізації даних процесів отримання синтактичного піноскла.	16
1.6 Аналіз невирішених питань і постановка задачі	18
Висновки до розділу	22
РОЗДІЛ 2. Моделі і методи візуалізації експериментальних і розрахункових даних в технологічних процесах отримання Синтактичних ПІНОскла	24
2.1 Математична модель кінетики спікання скляних мікросфер	24
2.2 Вимоги до інформаційної системи наукового співробітника.....	26
2.3 Розробка концепції ІС наукового співробітника.....	28
Висновки до розділу	31
РОЗДІЛ 3. Розробка і практична реалізація інформаційної системи у дослідницькій лабораторії.....	32
3.1 Розробка моделі ІС	32
3.1.1 Розробка діаграми варіантів використання	32
3.1.2 Визначення потоків даних ІС	35
3.1.2 Визначення діаграми станів ІС.....	41
3.1.3 Концептуальна модель даних.....	42
3.2 Технічний проект	43
3.2.1 Логічна модель даних.....	43
3.2.2 Архітектура ІС.....	45

3.2.3	Розробка діаграми розгортання	47
3.2.4	Основні прикладні алгоритми	48
3.3	Робочий проект	50
3.3.1	Вибір засобів розробки.....	50
3.3.2	Фізична модель даних	51
3.3.3	Кодування ІС.....	53
3.3.4	Тестування ІС.....	58
	Висновки до розділу	60
РОЗДІЛ 4. ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ДОЦІЛЬНОСТІ РОЗРОБКИ.....		61
4.1	Розрахунок вартості матеріально-енергетичних витрат.....	61
4.2	Розрахунок фонду оплати праці за виконану роботу	63
4.3	Розрахунок ціни розробки програми.....	66
4.4	Розрахунок економічної ефективності та строку окупності програмного забезпечення.....	68
	Висновки до розділу	69
РОЗДІЛ 5. ОХОРОНА ПРАЦІ		71
5.1	Умови праці та організація робочого місця	71
5.2	Техніка безпеки під час роботи з обладнанням та програмним забезпеченням	71
	Висновки до розділу	72
РОЗДІЛ 6. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА		73
6.1	Мінімізація екологічного сліду під час розробки ІС.....	73
6.2	Управління відходами та вторинне використання ресурсів.....	74
	Висновки до розділу	75
ВИСНОВКИ.....		76
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ		78
ДОДАТКИ.....		80
Додаток А – Програмний код.....		80
Додаток В – Презентаційні матеріали		101

ВСТУП

Актуальність дослідження зумовлена необхідністю підвищення ефективності обробки експериментальних даних технологічного циклу створення стекол спеціального призначення.

Мета роботи полягає у розробці нової інформаційної системи з вдосконаленим графічним відображенням систематизації експериментальних даних отримання стекол спеціального призначення.

Для досягнення поставленої мети в роботі необхідно вирішити наступні завдання:

1. Проаналізувати сучасний стан і проблемні питання ІТ-забезпечення скловареної галузі.
2. Розробити моделі та методи візуалізації експериментальних і розрахункових даних у технологічних процесах отримання синтактичних піностекол.
3. Розробити і реалізувати інформаційну систему у дослідницькій лабораторії.
4. Обґрунтувати економічну доцільність розробки інформаційної системи.
5. Забезпечити охорону праці під час виконання робіт з обладнанням та програмним забезпеченням. Визначити заходи з охорони навколишнього середовища під час розробки інформаційної системи.

Об'єкт дослідження – процеси впорядкування і аналізу експериментальних даних технологічного циклу отримання стекол спеціального призначення.

Предмет дослідження – математичні моделі кінетики спікання скляних мікросфер.

Наукова новизна роботи полягає у тому що вперше для технологічного циклу спікання скляних мікросфер запропоновано нові закономірності

систематизації та візуалізації оброблених експериментальних даних, що дає змогу прогнозувати властивості отримуваних спецстекло.

Практична цінність роботи полягає в зручності інтерфейсу, швидкості фіксації експериментальних даних, наявності мобільного застосунку.

Методи дослідження: статистична обробка даних, графічна візуалізація даних, технології реляційних баз даних, мови програмування C# та Python.

РОЗДІЛ 1. СУЧАСНИЙ СТАН І ПРОБЛЕМНІ ПИТАННЯ ІТ-ЗАБЕЗПЕЧЕННЯ СКЛОВАРЕНОЇ ГАЛУЗІ

1.1 Загальна характеристика технологічного процесу виробництва скла і критерії його якості

Завдяки своїм унікальним фізико-хімічним властивостям, у поєднанні із відносною дешевизною та поширеністю складових, скло є чи не найбільш поширеним матеріалом у техніці та господарстві, що не відноситься до металів. Воно знаходить застосування в безлічі галузях – у будівництві, приладобудуванні, оптиці, світлотехніці, електротехніці, радіоелектроніці, атомній енергетиці, хімічній промисловості і багатьох інших.

Скло – це аморфна тверда речовина, яка утворюється в результаті охолодження розплаву його складових компонентів, що мають склотвірні властивості[1].

Речовини в твердому стані за звичайних температури та тиску можуть мати кристалічну або аморфну структуру. Найпоширенішими у природі є кристалічні тверді речовини, для яких характерний чіткий геометричний порядок розташування атомів/іонів у просторі. Такі речовини мають чіткі геометричні форми, фіксовані температури плавлення і, як правило, проявляють анізотропію властивостей. Кристалічний стан є стабільним за звичайних умов і відзначається мінімальною внутрішньою енергією.

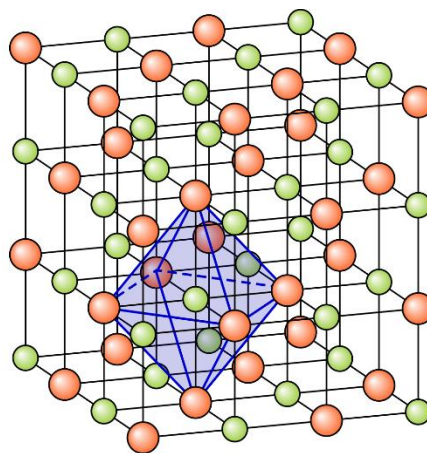


Рис. 1.1. Кристалічна структура NaCl[1]

На противагу кристалічним речовинам, речовини аморфної структури (речовини у склоподібному стані) мають невпорядковану внутрішню організацію. Частинки у таких речовинах розташовуються хаотично, через що речовини склоподібного стану мають ізотропні властивості. Склоподібний стан менш поширений у природі, є енергетично не вигідним і метастабільним, тобто схильним до повільної рекристалізації (розсклуванню)[1, 2].

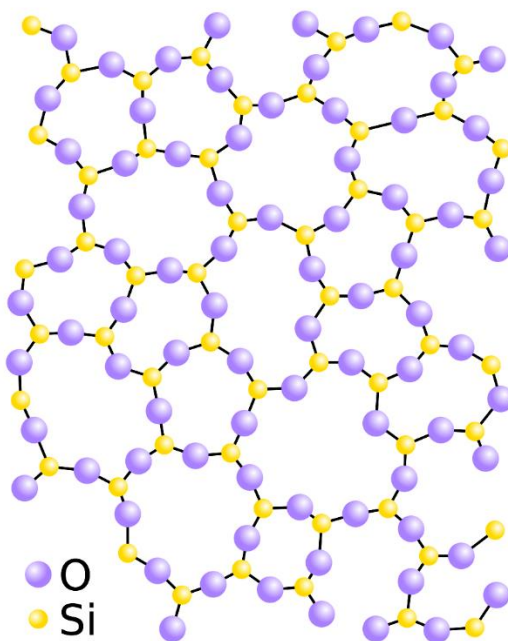


Рис. 1.2. Речовина аморфної структури (SiO_2)[1]

За довгу історію виробництва скла було розроблено і вдосконалено багато технологічних процесів отримання скла з різними властивостями та у різних макроструктурних конфігураціях.

Найбільш поширеним шляхом отримання скла є переохолодження розплавів склотвірних речовин із метою запобігання кристалізації. У загальному вигляді процес виглядає наступним чином:

1. Підготовка сировини. Найпоширенішими складовими є кварцовий пісок (SiO_2), технічний оксид алюмінію (Al_2O_3), крейда або вапняк (CaO), сода або селітра (Na_2O) (рисунок 1.3). Окрім того використовуються допоміжні матеріали – прискорювачі варіння, освітлювачі, барвники тощо. Сировина піддається первинній очистці за необхідністю, проходить просушку,

подрібнення та просівання. Із підготовленої сировини за допомогою дозатора формують шихту і ретельно її перемішують.

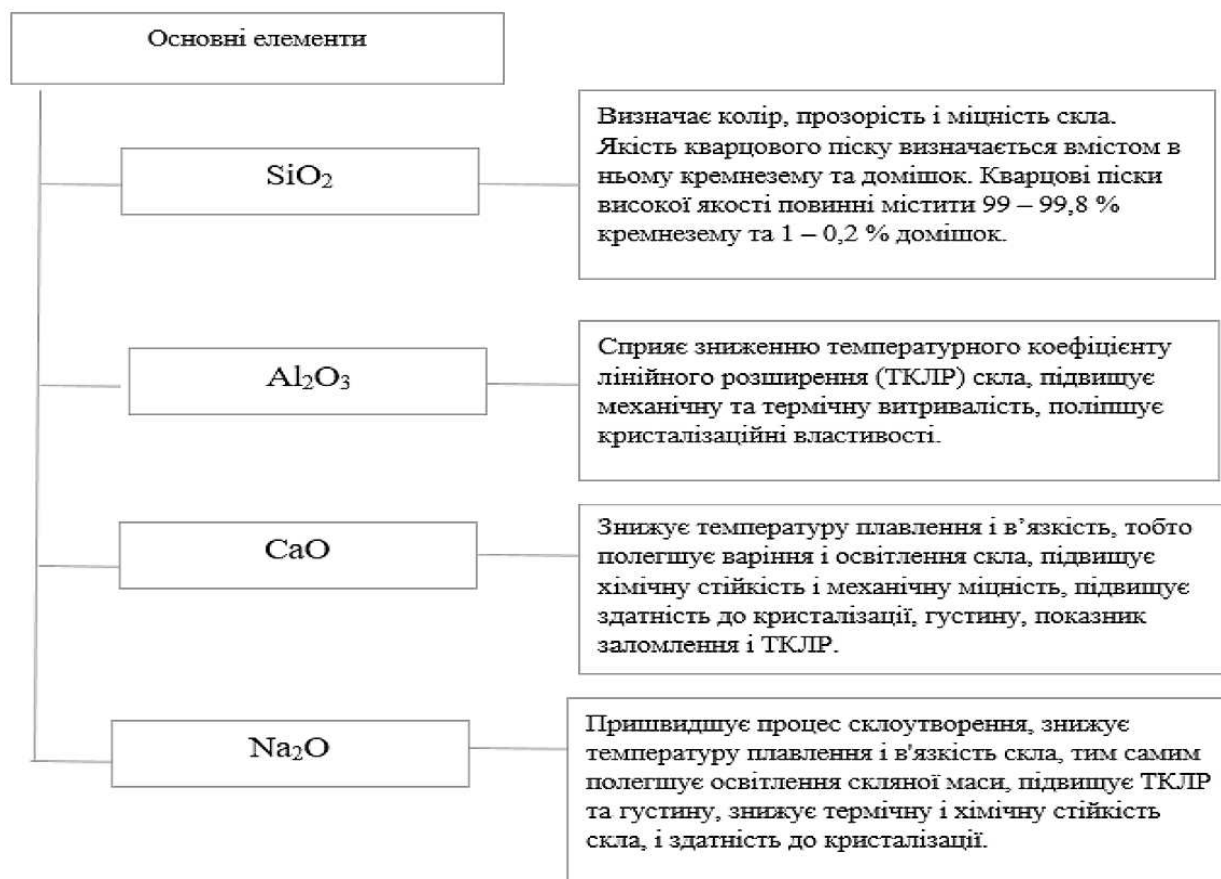


Рис. 1.3. Функції основних сировинних елементів скловареного виробництва[2].

2. Плавлення. Суміш нагрівають у спеціальній печі до необхідної температури. У цьому процесі відбувається видалення бульбашок повітря і забезпечується однорідність маси.

3. Формування. Рідке скло формують у виробі потрібної форми. Це може бути лиття, пресування, видування чи прокатка, залежно від призначення скла.

4. Охолодження та відпал. Заготовки повільно охолоджують у печах для зняття внутрішніх напружень і запобігання розтріскуванню.

Розплавлене скло при охолодженні не переходить у твердий стан, а плавно нарощує в'язкість, доки не набуває властивостей однорідної твердої речовини. Через відсутність кристалічної структури скло не має чітко визначеної точки плавлення. При повторному нагріванні скло поступово змінює свої властивості

в протилежному напрямку: від твердого і крихкого стану воно поступово стає пластичним, а згодом переходить у рідку форму, утворюючи скломасу.

Серед основних якісних характеристик скла можна виділити:

- Макроструктура. Чи є скло пористим, волокнистим, суцільним.
- Прозорість та інші оптичні властивості. Характерний для скла коефіцієнт світлового заломлення, середня дисперсія, наявність технологічних дефектів що впливають на оптичні властивості (камені, бульбашки, внутрішні напруги);
- Міцність. Здатність витримувати механічні навантаження різного характеру;
- Термічна стійкість. Здатність скла до термічної ізоляції, коефіцієнт теплового розширення;
- Хімічна стійкість. Стійкість до дії кислот та лугів/основ, дового знаходження у воді.

В залежності від мети застосування, модифікуючи склад скла та перебудовуючи технологічні процеси виробництва, можна отримувати скло із суттєво відмінними комбінаціями властивостей: оптичне скл, світлотехнічне, вибірно-прозоре, радіаційно стійке, радіаціодозиметричне, зміцнене, тугоплавке, термостійке, хімічно стійке, апаратне, термометричне, електроізоляційне, електродне, електровакуумне, люмінісцуюче, волокнисте, розчинне, пінне тощо.

1.2 Технологічний процес виробництва синтактичного піни та особливості виробництва синтактичного піноскла

На початку 1960-х років було розроблено цілий клас скловмісних матеріалів, що отримав загальну назву «синтактична піна». Синтактичні піни являють собою багатокомпонентний матеріал, у якому матриця з металу, полімеру, цементу або кераміки наповнюється порожнистими або непорожнистими скляними сферами, що виконують роль наповнювачів. Завдяки

включенню скляних часток цей матеріал отримує нові властивості: знижена щільність, як наслідок, підвищена питома міцність, збільшена гідростатична міцність, демпферна здатність, зменшений коефіцієнт теплового розширення, прозорість у радіо діапазоні та для ехолокації.

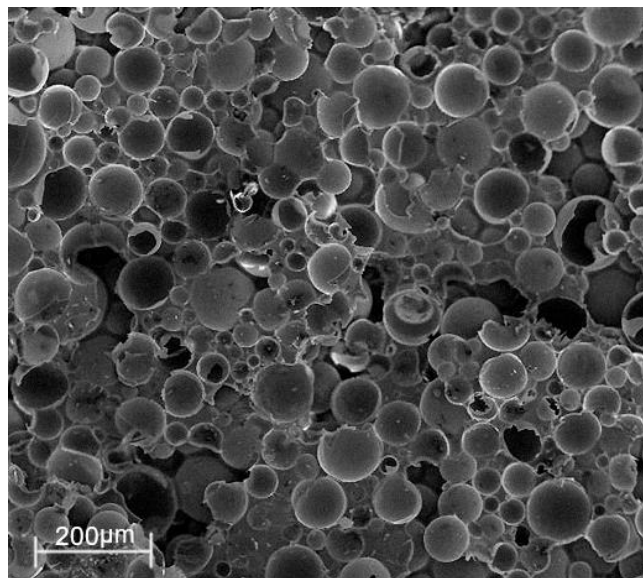


Рис. 1.4. Синтактична піна(епоксидна смола + скляні мікросфери) під електронним мікроскопом[3]

Завдяки новим властивостям синтактичні піни отримали широке застосування у суднобудуванні, виробництві підводної технік, у ракетобудуванні, авіації та космічній техніці [3].

Більшість технологій виготовлення синтактичної піни базується на фізико-хімічних процесах, що включають низькотемпературне приготування колоїдних розчинів. У цих розчинах неорганічні мікросфери рівномірно змішуються з такими сполучниками, як амінові смоли, синтетичний латекс або акрилові суспензії. Це забезпечує однорідність матеріалу, задану щільність і підвищену міцність зчеплення покриття з основною поверхнею. У загальному вигляді процес виглядає наступним чином:

1. Вибір і підготовка сировини. Пористі скляні мікросфери є наповнювачем. Їх виготовляють із силікатного скла за допомогою роздування розплаву або термічного розширення. Вони мають низьку щільність і високі теплоізоляційні властивості. Матриця може бути термореактивною (епоксидна

смола, поліефірна смола) або термопластичною (наприклад, поліамід, поліуретан). Металева матриця застосовується для виробів, що потребують високої термічної стабільності та міцності.

2. Формування виробу. Для полімерних матриць застосовують лиття під тиском. Суміш заливають у форму та полімеризують за допомогою нагрівання або добавок-ініціаторів. У випадку металевих матриць застосовують пресування. Суміш формують у прес-формах під високим тиском і температурою.

3. Механічна обробка. Синтактичні піни легко піддаються шліфуванню, різанню та іншим способам механічної постобробки, завдяки чому початковій заготовці легко надати необхідну форму.

Не дивлячись на відносну легкість, високі показники теплоізоляції та високу хімічну стійкість синтактичної піни, деструкційні процеси та плинність у міжфазній області між полімером і наповнювачем обмежують їх використання в умовах високих температур та теплових навантажень. Застосування ж високотемпературних технологій, таких як спікання, дозволяє об'єднувати скляні мікросфери без використання полімерних сполучників. Матеріали, створені таким чином, отримали назву «синтактичне піноскло». Вони характеризуються високими акустичними властивостями та гідростатичною міцністю, що дозволяє вирішувати низку складних інженерних завдань [4].

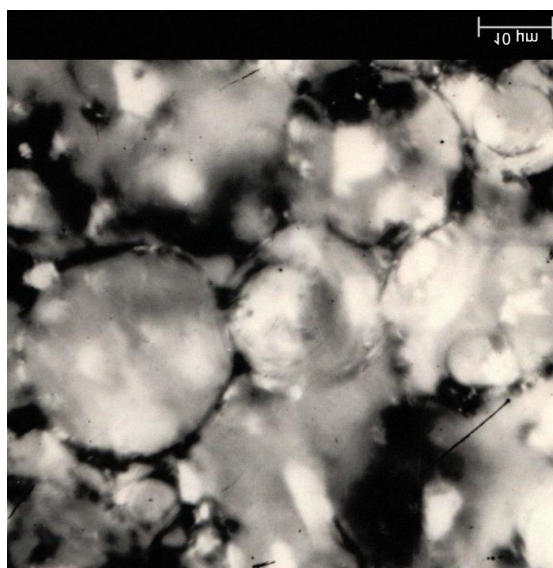


Рис. 1.5. Синтактичне піноскло під оптичним мікроскопом[4]

Ключовим фактором формування їх властивостей є особливості структури, яка залежить від збереження форми скляних частинок, максимально наближеної до сферичної. Отже важливим питанням для майбутнього вдосконалення технології є вивчення ролі температурних та тискових чинників у процесі досліджень структуроутворення під час спікання мікросфер, що безпосередньо визначає експлуатаційні характеристики результуючих матеріалів.

1.3 Організація дослідницької лабораторії композитних матеріалів

Науково-дослідницький відділ — це структурний підрозділ інституту, університету або підприємства, основним завданням якого є проведення фундаментальних і прикладних досліджень у певній науковій галузі.

Керівництво науково-дослідницького відділу композитних матеріалів складається з директора та його заступника. Директор лабораторії відповідає за загальне управління, стратегічне планування та координацію роботи всіх відділів. Він приймає рішення щодо напрямів досліджень і фінансування. Заступник директора з наукової роботи курує наукову діяльність, забезпечує зв'язок між дослідниками та адміністрацією, формує плани досліджень.

Адміністративний відділ відповідає за організацію фінансування, закупівлю обладнання, ведення документації, а також кадрове забезпечення. До його складу входять фінансовий відділ, відділ закупівель, та HR-відділ.

Наукові підрозділи розподіляються за напрямками досліджень та за класами матеріалів — відділи металевих матеріалів (вивчення структури, властивостей, термічної обробки металів та сплавів), полімерних матеріалів (дослідження синтезу, властивостей та модифікації полімерів), скла, наноматеріалів (розробка й дослідження наноструктурованих матеріалів).

Експериментальні лабораторії відіграють центральну роль у безпосередньому виконанні дослідів. До них належать аналітична лабораторія (хімічний, фізичний та спектральний аналізи матеріалів), лабораторія механічних випробувань (тестування міцності, жорсткості, в'язкості та інших

фізико-механічних властивостей), лабораторія термічних випробувань (дослідження термічної стійкості та теплових властивостей), а також лабораторія корозійної стійкості (вивчення поведінки матеріалів у агресивних середовищах).

Технічна підтримка та обслуговування забезпечують безперебійну роботу обладнання. Інженери та техніки відповідають за його належний стан, а ІТ-відділ підтримує комп'ютерну інфраструктуру, наукове моделювання та обробку даних.

Науково-освітній відділ організовує стажування, практику для студентів, а також курси підвищення кваліфікації для співробітників. Відділ підтримує співпрацю з університетами та освітніми центрами, що сприяє обміну знаннями та розвитку кадрів.

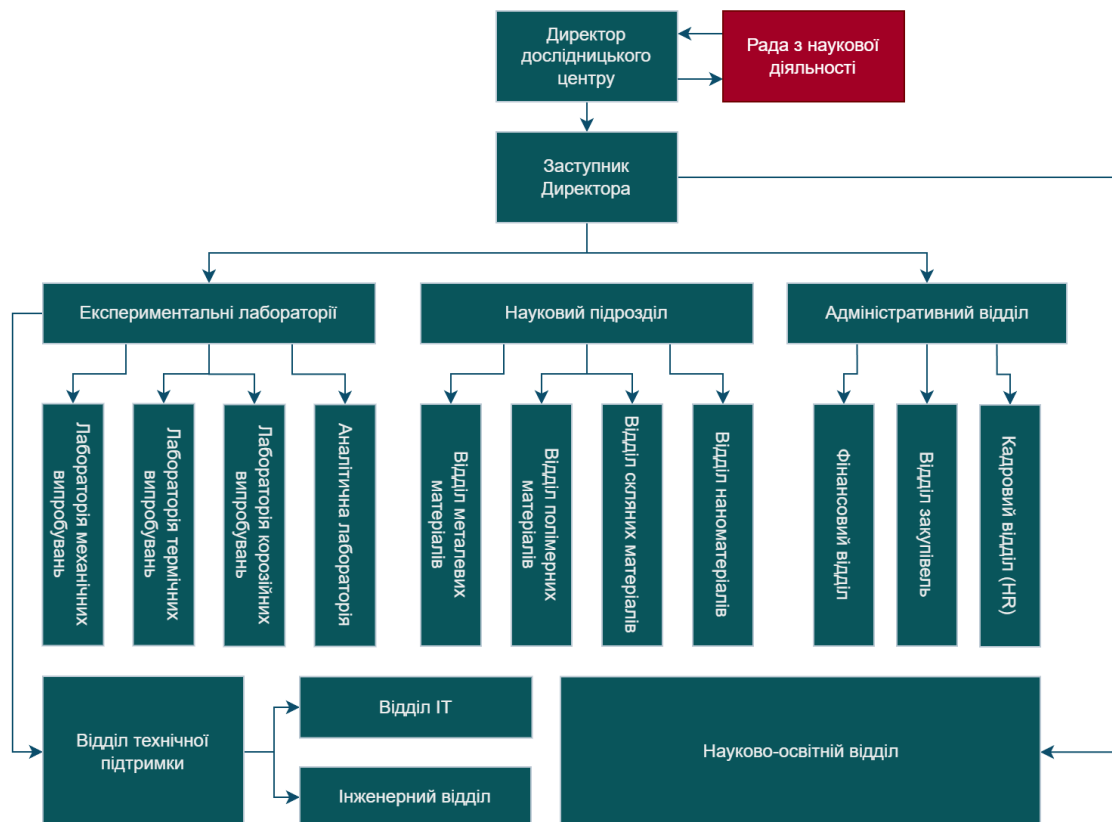


Рис. 1.6. Організаційна структура науково-дослідницького відділу композиційних матеріалів [створено автором]

Подана структура дозволяє ефективно організувати наукову діяльність, забезпечувати її матеріальну базу та швидку інтеграцію результатів у промисловість [5].

Для ефективної роботи окремої експериментальної лабораторії необхідна участь як мінімум трьох працівників: наукового керівника, наукового співробітника(аспіранта) та лаборанта.

Науковий керівник відповідає за склад та порядок виконуваних експериментів, виконує якісний контроль технологічного процесу, надає експертну допомогу в аналізі отриманих даних. Науковий співробітник відповідає за безпосереднє виконання експериментів, фіксування результатів експериментів (проміжних та підсумкових), виконує первинний аналіз отриманих даних. Лаборант виконує підготовчі операції, такі як підготовка шихт для запланованих експериментів, здійснює контроль за справністю обладнання та калібровку вимірювальних приладів.



Рис. 1.7. Організаційна структура експериментальної лабораторії [створено автором]

Об'єкт дослідження – процеси впорядкування і аналізу експериментальних даних технологічного циклу отримання стекол спеціального призначення. Предмет дослідження – математичні моделі кінетики спікання скляних мікросфер.

1.4 Методологія постановки досліджень процесів отримання синтактичних піноскло

Для проведення експериментальних досліджень використовуються експериментальну установку, що складається з шахтної печі з вертикальним завантаженням. Піч оснащена механічним пристроєм для пресування, який має індикатор часового типу для вимірювання процесів усадки.

Температурні параметри керуються блоком управління, який забезпечує підйом і підтримання необхідного рівня температури. Для вимірювання температури використовується термопара, яка розміщується безпосередньо в засипці скляних мікросфер і підключається до мілівольтметра. Додаткова контрольна термопара може бути вставлена у технологічну кришку через спеціальний отвір для перевірки рівномірності температурного поля.

Для кореляції даних, отриманих при нагріванні до цільової температури, проводиться порівняння показників контрольної термопари з даними скляного ртутного термометра, розташованого в графітовому контейнері разом із мікросферами. Результати обчислень демонструють середньоквадратичні відхилення, та використовуються у подальших розрахунках. Спостереження за процесами усадки дозволяють зупинити спікання у потрібний момент для досягнення необхідної структури матеріалу і запобігання дефектів.

Сировиною для проведення експериментів є скляні мікросфери, що в стані вільної засипки мають щільність $\sim 190 \text{ кг/м}^3$. Перед проведенням експерименту виконується процедура вібраційного ущільнення з частотою 50 Гц, завдяки чому щільність зростає до 200 кг/м^3 . Частки продовжують перегруповуватися також у процесі рівномірного прогрівання.

Цільові температури зазвичай знаходяться у діапазоні 550–650 °C, саме за цієї температури відбувається основне збільшення контактних зон між мікросферами.

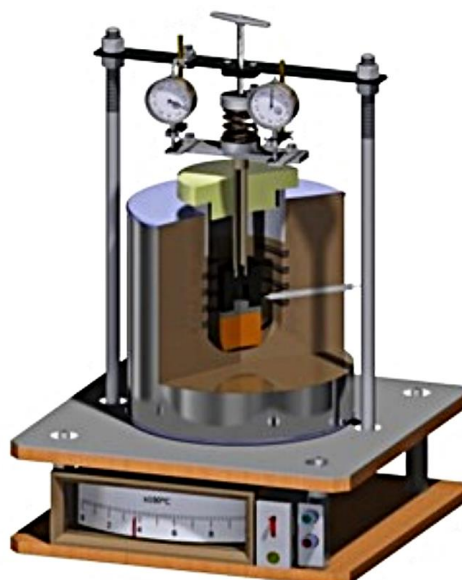


Рис. 1.8. Експериментальна установка для дослідження процесу спікання скляних мікросфер[4]

Метою досліджень є з'ясування оптимальних значень температурних показників та показників тиску, таких що будуть достатньо високими для досягнення високої щільності та малої пористості зразків (що позитивно впливає на міцність у статичних і динамічних умовах) і достатньо низькими для збереження загальної сферичної форми мікросфер (що забезпечує збереження інших корисних властивостей синтактичних піноматеріалів).

1.5 Методи візуалізації даних процесів отримання синтактичного піноскла

У дослідженні та вдосконаленні технологій отримання синтактичного піноскла важливу роль відіграють методи візуалізації даних. Вони дозволяють оцінити параметри якості матеріалу, вивчити структуру матеріалу, аналізувати фізико-механічні властивості та оптимізувати виробничі процеси. Одним із ключових методів є мікроскопічна візуалізація. Оптична мікроскопія дозволяє аналізувати морфологію мікросфер, оцінювати розподіл їхніх розмірів, виявляти дефекти та контролювати цілісність структури. Скануюча електронна мікроскопія (SEM) використовується для детального вивчення поверхневих

структур і розподілу пористості, тоді як трансмісійна електронна мікроскопія (ТЕМ) дозволяє аналізувати внутрішню структуру матеріалу на нанорівні.

Томографічні методи, такі як комп'ютерна томографія (СТ), забезпечують тривимірну візуалізацію внутрішньої структури синтактичного піноскла, включаючи форму, розмір і розподіл пор. Мікротомографія дозволяє детально вивчати мікроструктуру матеріалу з високою роздільною здатністю, що робить її незамінним інструментом для оцінки однорідності розподілу мікросфер та виявлення дефектів.

Раманівська спектроскопія дозволяє визначати хімічний склад компонентів, візуалізуючи спектри, що відображають взаємодію мікросфер. ІЧ-спектроскопія (FTIR) використовується для аналізу хімічних зв'язків у матеріалі, забезпечуючи інформацію про структурні зміни, що відбуваються під час полімеризації.

Для вивчення механічних властивостей матеріалу застосовуються такі методи, як динамічний механічний аналіз (DMA) та акустична емісія. DMA дозволяє візуалізувати механічну поведінку матеріалу за різних температур і навантажень, будуючи графіки модуля зсуву та тангенса втрат. Акустична емісія, у свою чергу, дає змогу в реальному часі відстежувати процеси пошкодження та руйнування матеріалу, візуалізуючи ділянки концентрації напружень за допомогою графіків енергетичних сигналів.

Процеси виробництва синтактичного піноскла також візуалізуються за допомогою тепловізійного аналізу та цифрової фото- і відеозйомки. Тепловізійний аналіз дозволяє контролювати рівномірність нагрівання під час термообробки, допомагаючи уникнути локальних перегрівів чи недостатньо прогрітих зон. Фото- і відеозйомка використовується для моніторингу фазових переходів, змішування компонентів та контролю рівномірності їхнього розподілу.

Статистичні методи візуалізації, такі як гістограми розподілу розмірів мікросфер, забезпечують графічне представлення варіативності розмірів частинок, оцінюючи ефективність сировинних матеріалів. 3D-графіки та

контурні карти демонструють кореляції між параметрами виробництва, наприклад температурою, тиском і характеристиками готового матеріалу.

Для числового моделювання і візуалізації використовуються методи кінцевих елементів (FEM) і комп'ютерна динаміка рідин (CFD). FEM дозволяє прогнозувати поведінку синтактичного піноскла під дією зовнішніх сил, візуалізуючи розподіл напружень, тоді як CFD допомагає моделювати потоки рідин під час змішування компонентів[3].

Під час спікання синтактичне піноскло являє собою пластичну, в'язку систему, у якій рівномірно розподіляються як відкриті, так і закриті пори. Завдяки широкому діапазону розмірів мікросфер (10–100 мкм) об'єм насипу можна вважати компактно упакованим: більші сфери утворюють основну структуру, а проміжки між ними заповнюються середніми та дрібними сферами. У даному випадку зміна в'язкості та зміна геометричних розмірів зразка характеризує динаміку щільності та пористості засипки, а також міру збереження сферичності скляних мікросфер, що у свою чергу, як було зазначено, багато у чому визначає якості кінцевого результату. Тому чи не найважливішу роль відіграють відповідні графіки зміни геометричних розмірів насипу та в'язкості розплаву скляних мікросфер[4].

1.6 Аналіз невирішених питань і постановка задачі

Процес виконання експерименту в експериментальній лабораторії як правило виглядає наступним чином – науковий керівник виходячи із поточних завдань лабораторії визначає мету та план роботи. Отримавши задачу від керівника, науковий співробітник організовує підготовку до експерименту, запрошує в лаборанта необхідне обладнання та матеріали, за потреби надає додаткові інструкції. Лаборант проводить підготовчі роботи, калібрує прилади, заготовляє шихту. На початку експерименту науковий співробітник фіксує початкові умови експерименту (цільову температуру, заданий тиск, діаметр, висота та маса засипки), в паперовому журналі, після чого експеримент

розпочинається. У ході експерименту науковий співробітник виконує основні вимірювання кожну хвилину (поточна температура, усадка засипки), які фіксуються у згаданому журналі у вигляді таблиць і текстових приміток.

Після завершення експерименту науковим співробітником фіксуються в журналі вихідні результати (фінальна висота засипки). Паперові записи систематизуються і переносяться до електронного табличного процесора MS Excel, де дані упорядковуються для подальшого аналізу і побудови графіків усадки засипки та зміни її в'язкості.

Науковий керівник наглядає за правильністю виконання експерименту, перевіряє отримані результати, надає експертну допомогу при аналізі результатів та формулюванні висновків та планує подальші дії.

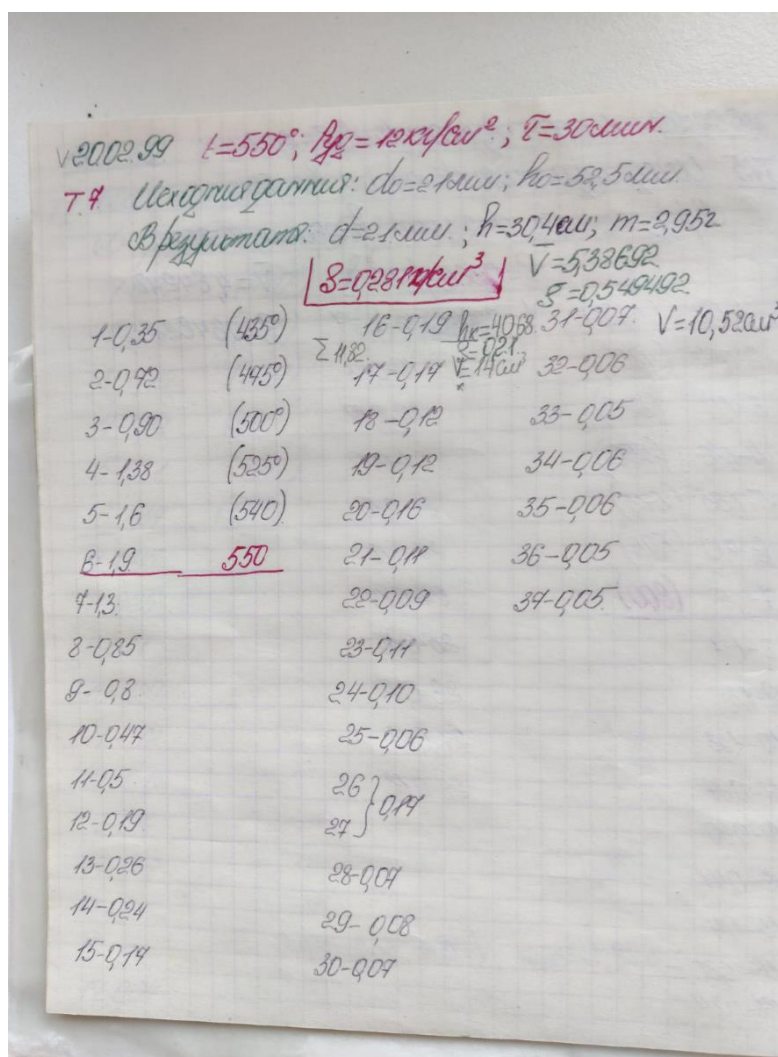


Рис. 1.9. Приклад сторінки з паперового журналу експериментів
[створено автором]

Фіксування проміжних і підсумкових результатів експериментів шляхом паперових записів і подальшої візуалізації через MS Excel має низку недоліків.

Паперові записи можуть легко пошкодитися, загубитися або бути зруйнованими під час зберігання. Процес перенесення результатів з паперу в електронний вигляд є трудомістким. Виникає висока ймовірність помилок, які можуть суттєво спотворити дійсні результати дослідів. Відсутність автоматизованої обробки даних уповільнює роботу та ускладнює аналіз.

Не існує способу вносити дані одразу в MS Excel, адже науковий співробітник не може одночасно працювати з персональним комп'ютером та фіксувати експериментальні дані

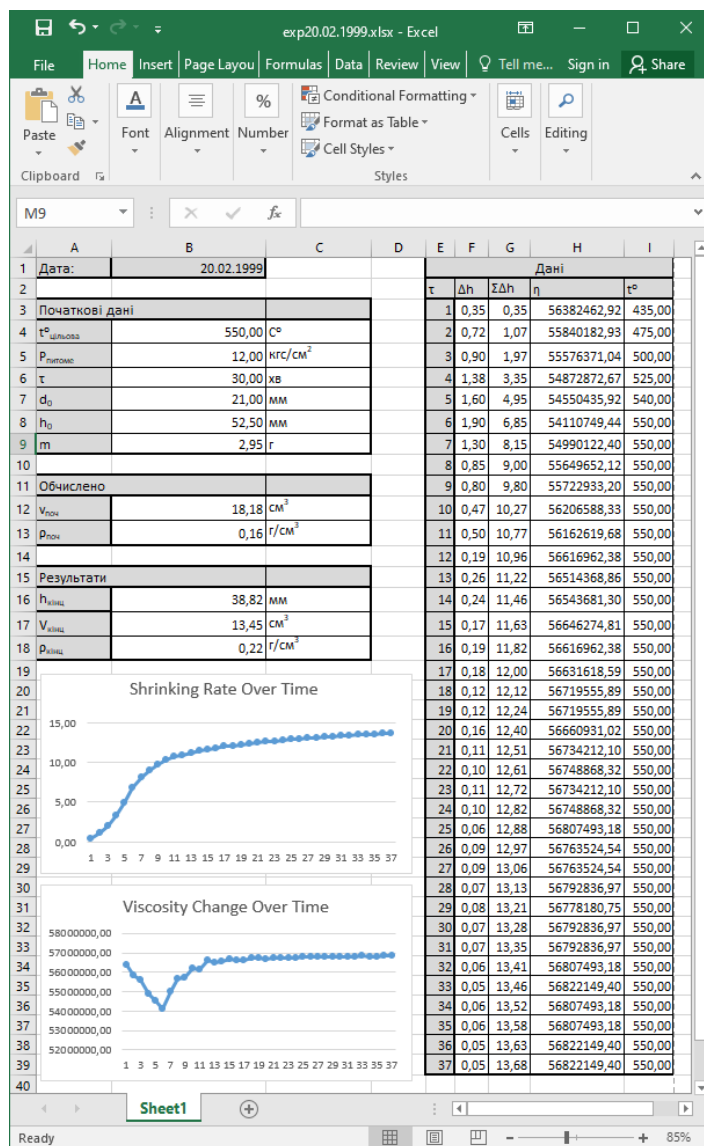


Рис. 1.10. Сторінка з журналу експериментів перенесена в MS Excel
[створено автором]

Візуалізація даних у MS Excel забезпечує лише базові можливості й не завжди відповідає потребам конкретних наукових досліджень. Наприклад не існує простого способу порівнювати отримувані графіки зміни висоти засипки та її в'язкості. Окрім того такий підхід обмежує майбутні потенційні можливості з інтеграції з іншими системами обробки даних.

Крім того, використання паперових носіїв не є екологічним і є ресурсомістким.

Зазначені проблеми є завадою для своєчасного виконання дослідів та компрометують точність їх результатів. Вони можуть бути вирішені за допомогою впровадження ІС наукового співробітника для дослідження геометричних розмірів та в'язкості синтактичного піноскла.

Основним завданням впровадження ІС наукового співробітника є спрощення процесу фіксування, аналізу та візуалізації експериментальних даних під час дослідження в'язкості синтактичного піноскла в процесі спікання під заданим тиском. Встановлення такої системи на мобільні пристрої дозволить забезпечити зручний доступ до даних і оперативне внесення результатів безпосередньо під час експериментів, що мінімізує ризики людських помилок і сприяє підвищенню точності наукових досліджень.

Після аналізу робочих процесів наукового співробітника визначено доцільність створення ІС, яка автоматизує такі завдання:

- Відстеження досліджуваних складів синтактичного піноскла. ІС надає можливість зберігати інформацію про хімічні склади, властивості та параметри досліджуваних матеріалів.
- Відстеження проведення експериментів за датами. Система дозволяє організувати журнал експериментів із зазначенням дати, умов проведення та основних параметрів.
- Фіксація даних під час експериментів. Науковий співробітник може вносити значення в режимі реального часу, автоматично додаючи коментарі та примітки до кожного етапу дослідження.

- Побудова графіків зміни відносної висоти засипок у часі. ІС автоматично будує графіки на основі внесених даних, що дозволяє оперативно аналізувати динаміку змін.

- Побудова графіків зміни в'язкості синтактичного піноскла у часі. Система обробляє експериментальні результати та створює точні графіки, які можна порівнювати між собою.

- Порівняння створюваних графіків. ІС забезпечує можливість накладення графіків один на одного для оцінки впливу різних параметрів на в'язкість матеріалу.

- Створення звітів та інтеграція з іншими системами. Система генерує звіти, готові до друку чи передачі у вигляді файлів, сумісних із сучасними офісними пакетами.

Критеріями якості впроваджуваної ІС є:

- Надійність функціонування системи. Система має стабільно працювати за будь-яких умов.

- Функціональна повнота та швидкодія. ІС повинна швидко обробляти великі обсяги даних і відповідати всім потребам наукового співробітника.

- Можливість інтеграції з іншими системами. Забезпечення сумісності з існуючим програмним забезпеченням для аналізу даних.

- Мінімізація витрат. Оптимальна вартість апаратного забезпечення, розробки, супроводу та розвитку системи.

Таким чином, впровадження інформаційної системи дозволить значно підвищити ефективність виконання досліджень, зменшити навантаження наукового співробітника та покращити точність отриманих результатів.

Висновки до розділу

У розділі представлено огляд сучасного стану і проблемних питань ІТ-забезпечення скловареної галузі. Дано загальну характеристику технологічного процесу виробництва скла і критерії його якості. Описано технологічний процес

виробництва синтактичної піни та особливості виробництва синтактичного піноскла. Організовано дослідницьку лабораторію композитних матеріалів. Визначено методологію постановки досліджень процесів отримання синтактичних піноскел. Представлено методи візуалізації даних процесів отримання синтактичного піноскла. Проведено аналіз невирішених питань і сформульовано постановку задачі.

Об'єктом дослідження визначено процеси впорядкування і аналізу експериментальних даних технологічного циклу отримання скел спеціального призначення. Предметом дослідження визначення математичні моделі кінетики спікання скляних мікросфер.

РОЗДІЛ 2. МОДЕЛІ І МЕТОДИ ВІЗУАЛІЗАЦІЇ ЕКСПЕРИМЕНТАЛЬНИХ І РОЗРАХУНКОВИХ ДАНИХ В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ ОТРИМАННЯ СИНТАКТИЧНИХ ПІНОСТЕКОЛ

2.1 Математична модель кінетики спікання скляних мікросфер

Пористість – це характеристика матеріалу, що визначає частку його об'єму, зайняту порами (порожнинами). Відкрита пористість – це частка пор у матеріалі, які з'єднані між собою та доступні для проникнення рідин або газів. Закрита пористість – це частка ізольованих пор, недоступних для зовнішнього середовища та не пов'язаних між собою.

В'язкість – це фізична властивість рідин і газів, що характеризує їх здатність чинити опір відносному переміщенню частинок або шарів відносно один одного. Тверді тіла, такі як скло, також можуть характеризуватися в'язкістю[4, 6].

Для розрахунку величини в'язкості синтактичного піноскла взято формулу 2.1.

$$\theta = \theta_0 - \frac{P}{\sqrt{2}\eta} \tau, \quad (2.1)$$

де θ – відкрита пористість піноскла отриманого у результаті спікання мікросфер;

θ_0 – початкова відкрита пористість засипки мікросфер;

P – прикладений тиск пресування;

η – в'язкість синтактичного піноскла;

τ – час спікання.

Фізичний зміст даного виразу полягає у врахуванні впливу часового параметру τ тиску P та в'язкості η на процес зменшення початкової пористості θ_0 .

Очевидно, що загальний об'єм засипки синтактичного піноскла складається з об'єму самого скляного матеріалу, а також з об'єму закритої та відкритої пористості. У такому випадку загальний об'єм зразка можна дати у вигляді формули 2.2.

$$V = V_B + V_{\text{скл}} + V_3,$$

де V_B – об'єм відкритої пористості;

$V_{\text{скл}}$ – об'єм займаємий безпосередньо склом;

V_3 – об'єм закритої пористості замкненої всередині мікросфер.

Враховуючи циліндричну форму засипки величина відкритої пористості визначається формулою 2.3.

$$P_0 = 1 - \frac{4m}{\pi d^2 h \rho_M}, \quad (2.3)$$

де m – маса засипки;

d, h – діаметер та висота зразка відповідно;

ρ_M – гадана щільність мікросфер.

Таким чином, виходячи з формули 2.2, об'єм закритої пористості визначається за формулою 2.4.

$$V_3 = \frac{m}{\rho_M} - \frac{m}{\rho_{\text{скл}}}, \quad (2.4)$$

де $\rho_{\text{скл}}$ – щільність скла що складає мікросфери.

Об'єм займаний склом вираховується за формулою 2.5.

$$V_{\text{скл}} = \frac{m}{\rho_{\text{скл}}}, \quad (2.5)$$

Тоді об'єм займаний відкритою пористістю для зразка циліндричної форми буде визначатися за формулою 2.6.

$$V_0 = \frac{\pi d^2 h}{4} - \frac{m}{\rho_M}, \quad (2.6)$$

Значення початкової відкритої пористості зразка визначається за формулою 2.7.

$$\theta_0 = 1 - \frac{4m}{\pi d_0^2 h_0 \rho_M}, \quad (2.7)$$

де d_0, h_0 – початкові діаметер та висота зразка відповідно.

Величина кінцевої пористості визначається відповідним чином за формулою 2.8.

$$\theta = 1 - \frac{4m}{\pi d_K^2 h_K \rho_M}, \quad (2.8)$$

де d_K, h_K – початкові діаметер та висота зразка відповідно.

Зробивши підстановку для початкової і кінцевої пористості можна отримати вираз 2.11 для обчислення в'язкості синтактичного піноскла.

$$\theta_0 - \theta = \frac{P}{\sqrt{2}\eta} \tau \quad (2.9)$$

$$\eta = \frac{P\tau}{\sqrt{2}(\theta_0 - \theta)} \quad (2.10)$$

$$\eta = \frac{\pi P \rho_M d^2 \Delta h}{4\sqrt{2}m} \tau \quad (2.11)$$

2.2 Вимоги до інформаційної системи наукового співробітника

ІС для наукового співробітника, яка автоматизує процес дослідження процесу спікання синтактичного піноскла, має виконувати ряд ключових функцій, спрямованих на полегшення фіксації та аналізу експериментальних даних, а також на підвищення ефективності наукових досліджень.

ІС повинна надавати науковому співробітникові можливість:

- Відстежувати досліджувані склади синтактичного піноскла. ІС повинна дозволяти зберігати і редагувати інформацію про хімічні склади матеріалів, які підлягають дослідженню.
- Фіксувати дані, отримані під час експериментів. Науковий співробітник повинен мати можливість вносити дані в режимі реального часу, забезпечуючи точність і швидкість введення.
- Автоматично будувати графіки. ІС має забезпечувати автоматичну побудову графіків зміни відносної висоти засипок та в'язкості синтактичного піноскла в часі, що дозволяє науковому співробітнику швидко оцінювати результати експериментів.

- Порівнювати графіки. ІС повинна дозволяти накладати та порівнювати різні графіки, що дозволить аналізувати вплив різних параметрів на в'язкість матеріалу.

- Зберігати та архівувати результати. Усі дані повинні зберігатися у базі даних або на локальному жорсткому диску, що дозволяє науковому співробітнику зберігати результати для подальшого аналізу.

- Підготовка звітів. ІС повинна мати функціонал для автоматичного генерування звітів на основі отриманих результатів, які можуть бути збережені у вигляді файлів (наприклад, PDF, Excel) для подальшої обробки чи друку.

Технічні вимоги до ІС:

- ІС повинна бути сумісною з операційною системою Windows, що встановлена на робочому комп'ютері співробітника.

- ІС повинна працювати з системою керування бази даних MySQL версії не нижче 5.6 для зберігання даних про склади матеріалів, результати експериментів та побудовані графіки на виділеному сервері.

- ІС повинна підтримувати програмну платформу .NET версії не нижче 8.0 для забезпечення ефективного виконання додатків.

- ІС повинна бути адаптована для встановлення на мобільні пристрої під керуванням Android та iOS для забезпечення мобільності та зручності фіксації даних безпосередньо під час проведення експериментів.

Ключові вимоги до функціональності ІС:

- Безперебійна робота. ІС повинна працювати без збоїв під час проведення експериментів, оскільки будь-яка неполадка може призвести до втрати важливих даних.

- Швидкодія. ІС має бути досить швидкою, щоб забезпечити оперативну обробку та відображення графіків і результатів без затримок, що дозволить зекономити час наукового співробітника.

- Захист даних. ІС повинна забезпечувати надійний захист даних, включаючи шифрування результатів досліджень, щоб запобігти несанкціонованому доступу до чутливої інформації.

- Зручний інтерфейс. ІС повинна мати простий і інтуїтивно зрозумілий інтерфейс, щоб науковий співробітник міг швидко освоїти систему і не витрачати час на навчання роботі з нею.

2.3 Розробка концепції ІС наукового співробітника

На основі визначених вимог була розроблена концепція[7] інформаційної системи наукового співробітника, призначеного для фіксування, зберігання та візуалізації результатів експериментів. Модель концепції АРМ зображена на рисунку 2.1.

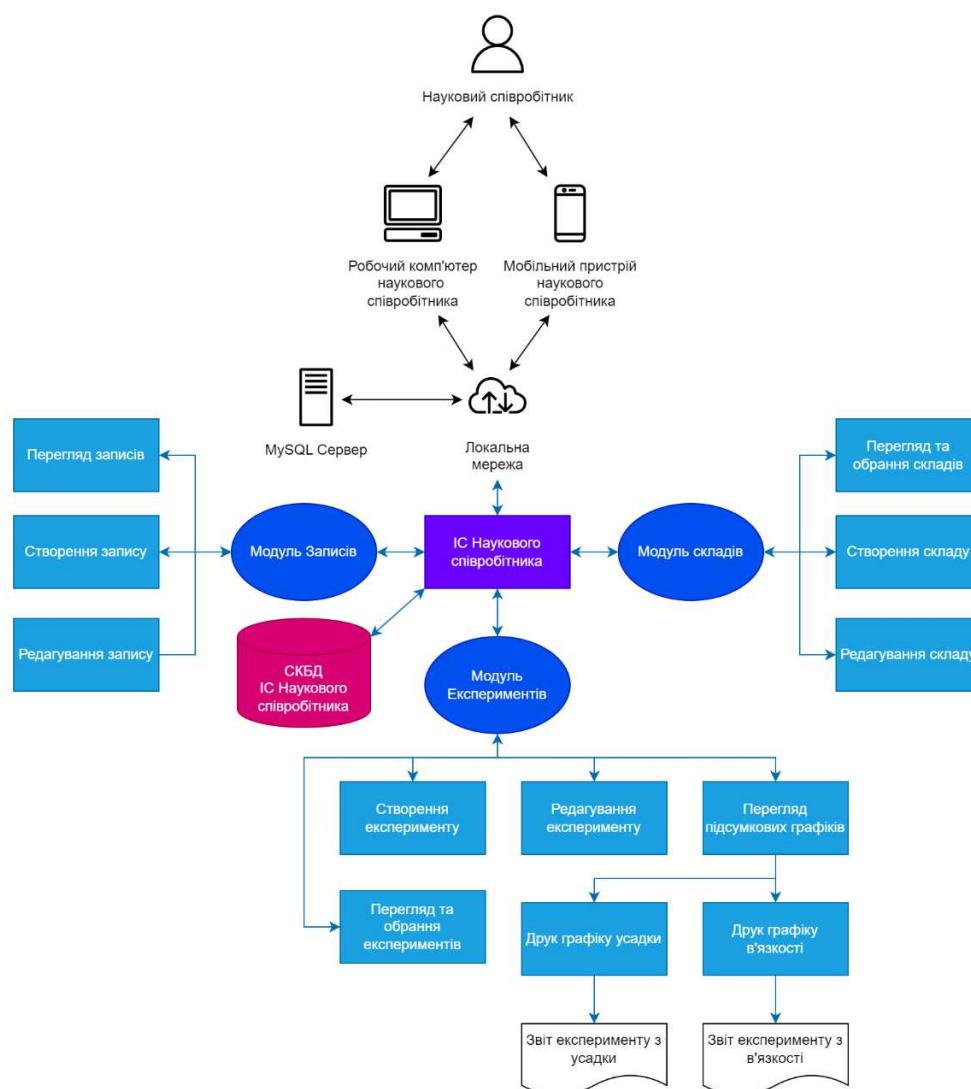


Рис. 2.1. Модель концепції ІС наукового співробітника для проведення дослідів процесів формування синтактичного піноскла [створено автором]

Склад функціональних задач ІС наступний:

1. Фіксація складів мікросфер, що розглядаються у дослідях.
2. Фіксація часу проведення, обраного складу та початкових даних експерименту. ІС повинна дозволяти науковому вводити та зберігати початкові умови дослідів, зокрема, цільову температуру, тиск, масу засипки, діаметр засипки, висоту засипки, час спікання.
3. Збір похвилинних даних. ІС має надавати можливість науковому співробітнику здійснювати похвилинну фіксацію даних під час експерименту, зокрема температури та величини усадки.
4. Аналіз та візуалізація даних. Науковий співробітник повинен мати можливість підключитися до сервера для аналізу зібраних даних і побудови графіку відносної усадки засипки та графіку зміни в'язкості засипки.
5. Зберігання результатів. ІС повинна надавати можливість зберігати кінцеві результати експериментів, зокрема кінцеву висоту засипки, кінцевий об'єм засипки, кінцеву щільність засипки, а також збереження побудованих графіків для подальшого використання або публікацій. Всі дані повинні зберігатися в базі MySQL[11, 14], доступ до якої здійснюється віддаленно через локальну мережу.
6. Друк результатів. ІС повинна надавати можливість друкувати отримувані у результаті досліджень графіки.

Вхідні дані: Вхідними даними є назви та хімічний склад скляних мікросфер, дату та час проведення дослідів, що дозволяє точно зафіксувати час початку експерименту, початкові умови проведення дослідів(цільова температура, тиск, маса засипки, діаметр засипки, висоту засипки та час спікання).

Проміжні дані: об'єм засипки та початкова щільність засипки

Вихідні дані: кінцева висота засипки, кінцевий об'єм засипки та кінцева щільність засипки, а також графіки, які відображають відносну усадку засипки в часі та зміни в'язкості засипки під час експерименту.

ІС наукового співробітника розгорнута на його робочому комп'ютері і складається із трьох модулів – модуль складів, модуль експериментів та модуль похвилинних записів[9]. ІС наукового співробітника також може для зручності бути розгорнута на його мобільних пристроях, де матиме аналогічну функціональність.

Модуль складів дозволяє зберігати та переглядати назви та детальний опис хімічного складу мікросфер, з якими виконуються досліді.

Модуль експериментів дозволяє фіксувати та переглядати дати проведених експериментів, їх початкові та підсумкові дані. Також цей модуль дозволяє будувати та друкувати графік відносної усадки засипки та графік зміни в'язкості мікросфер на базі прикріплених до експерименту похвилинних записів. Кожен експеримент прив'язаний до складу з яким він виконується.

Модуль похвилинних записів дозволяє фіксувати і переглядати похвилинні вимірювання зміни усадки засипки та зміни температури під час виконання експериментів. Кожен запис прив'язується до експерименту до якого він належить.

Окремо в лабораторії розгортається віддалений сервер СКБД MySQL підключений до її локальної мережі. ІС наукового співробітника, не залежно від того де саме вона встановлена, на мобільному пристрої чи на робочому комп'ютері, зберігає та отримує дані із зазначеного серверу. Це забезпечує можливість працювати з даними у будь-якій точці лабораторії. Окрім того це забезпечує певний рівень безпеки даних, адже СКБД MySQL[14] надає надійні механізми авторизації, а підключитися до неї можна буде лише безпосередньо з локальної мережі лабораторії[8].

В процесі проектування використано сервіс малювання діаграм app.diagrams.net, текстовий процесор Google Documents, табличні процесори MS Excel та Google Sheets та система управління проектам Trello.

Для розробки було обрано мову програмування C#[13]. Для роботи з візуалізацією даних застосовано мову програмування Python в поєднанні з бібліотекою matplotlib. У якості системи керування базами даних використано

MySQL. Основними інструментами для розробки є Microsoft Visual Studio 2022, Microsoft Visual Studio Code для роботи з Python, програмна платформа .NET 8.0 та засіб візуальної розробки баз даних MySQL Workbench[23].

Для розробки буде потрібно щонайменше два комп'ютери з 64-розрядним процесором, 8-ма ГБ оперативної пам'яті, 240 ГБ вільного місця на жорсткому диску та графічним прискорювачем. Окрім того необхідний один мобільний пристрій під керуванням ОС Android і один мобільний пристрій під керуванням ОС iOS та принтер. Всі пристрої повинні бути підключені до однієї локальної мережі Ethernet[8]. Крім того, необхідні монітори з роздільною здатністю не менше ніж 1920x1080 пікселів, клавіатури, миші.

Цільовими операційними системами обрані Windows 10, iOS та Android адже саме цим ОС користується науковий співробітник експериментальної лабораторії що розглядається.

На основі цієї концепції розроблено проект, який дозволить покращення швидкості роботи наукового співробітника під час виконання дослідів, а також зменшити ризики виникнення помилок у процесі їх проведення.

Висновки до розділу

У розділі представлено моделі та методи візуалізації експериментальних і розрахункових даних у технологічних процесах отримання синтактичних піностекол. Зокрема, розроблено математичну модель кінетики спікання скляних мікросфер. Визначено вимоги до інформаційної системи наукового співробітника. Розроблено концепцію інформаційної системи наукового співробітника.

РОЗДІЛ 3. РОЗРОБКА І ПРАКТИЧНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ У ДОСЛІДНИЦЬКІЙ ЛАБОРАТОРІЇ

3.1 Розробка моделі ІС

3.1.1 Розробка діаграми варіантів використання

З метою з'ясування основних функціональних можливостей системи створено діаграму варіантів використання (use-case діаграму).

Use Case – це графічний інструмент, що використовується для опису функціональності системи з точки зору її взаємодії з користувачами (акторами). Є одним із ключових елементів методології Unified Modeling Language (UML)[10]. Графічний інструмент дає змогу описати функціональність системи з позиції її взаємодії із користувачами (акторами) і є ключовим елементом аналізу та проєктування в межах методології Unified Modeling Language (UML). Діаграма відображає, які функції може виконувати система, як саме взаємодіють із нею зовнішні учасники, а також межі системи.

Рисунок 3.1 демонструє діаграму варіантів використання ІС наукового співробітника.



Рис. 3.1. Діаграма варіантів використання ІС наукового співробітника [створено автором]

На діаграмі відображаються три основні тип сутностей:

- Актори – зовнішні учасники, які взаємодіють із системою (користувачі, інші системи або зовнішні пристрої).
- Варіанти використання (use cases) – конкретні дії або функції, які система виконує у відповідь на запити акторів.
- Зв'язки – взаємодії між акторами та варіантами використання (асоціації, включення, розширення).

З метою точного визначання сценаріїв використання системи, необхідно врахувати унікальні способи взаємодії кожного користувача з нею. Це вимагає детального вивчення їхніх потреб, аналізу зібраних даних та систематизації їх у чіткі варіанти використання. При цьому слід відокремити конкретні дії акторів від їх індивідуальних особливостей, зосереджуючись на ключових завданнях, які має вирішувати система.

На основі зібраних вимог формується структура і дизайн інтерфейсу, необхідні для реалізації кожного сценарію взаємодії. Для цього використовується моделювання варіантів застосування за допомогою Use-Case діаграм, які дозволяють наочно зобразити взаємодію користувачів із системою та розкрити її функціональні можливості.

Створення таких діаграм здійснюється за допомогою уніфікованої мови моделювання (UML) у середовищі створення схем-діаграм app.diagrams.net.

Таблиця 3.1. Опис use-case діаграми [створено автором]

Актор:	Науковий співробітник
Прецеденти	Опис
Створити склад	Науковий співробітник може створити новий запис про склад, зазначивши його назву та опис. Назва складу є обов'язковим параметром нового складу. Опис складу можна додавати за бажанням
Редагувати склад	Науковий співробітник має можливість вносити зміни до раніше створеного запису про склад.

Актор:	Науковий співробітник
Видалити склад	Науковий співробітник може видалити непотрібний запис про склад зі списку.
Створити експеримент	Науковий співробітник може створити новий запис про експеримент, вказуючи дату, початкові дані експерименту. Кожен експеримент повинен бути прив'язаний до раніше створеного складу.
Редагувати експеримент	Науковий співробітник має змогу редагувати існуючий запис про експеримент, уточнюючи дані.
Видалити експеримент	Науковий співробітник може видалити запис про експеримент, якщо дані вже не є актуальними.
Створити похвилинний запис	Науковий співробітник може створити новий похвилинний запис, вказавши хронологічну інформацію про хід експерименту. Похвилинний запис обов'язково прікріплений до певного експерименту. Похвилинний запис обов'язково містить хвилину експерименту, на якій здійснена фіксація даних, поточну температуру та зміну висоти засипки.
Редагувати похвилинний запис	Науковий співробітник має можливість редагувати існуючі похвилинні записи, щоб уточнити або виправити дані.
Видалити похвилинний запис	Науковий співробітник може видалити непотрібний похвилинний запис, якщо він був створений помилково, або якщо хвилина фіксації вказана невірно.
Візуалізувати графіки експерименту	Науковий співробітник може створювати графіки для аналізу експерименту. Це графіки зміни відносної висоти засипки та зміни в'язкості засипки у часі.
Друкувати графіки експерименту	Науковий співробітник може зберігати на жорсткий диск у вигляді зображень, а згодом і друкувати отримані графіки для подальшого включення їх до звітів.

3.1.2 Визначення потоків даних ІС

Діаграма потоків даних (ДПД, Data Flow Diagram, DFD) – графічний документ, що використовується для моделювання потоків даних у системі[10]. Відображає, як дані входять у систему, обробляються, зберігаються та виходять із неї. ДПД допомагає зрозуміти взаємодію між різними компонентами системи та джерелами даних.

Основними елементами ДПД є процеси, які виконують певні операції над даними; потоки даних, що позначають передачу інформації між елементами; зовнішні сутності, які представляють джерела або споживачів даних поза межами системи; а також сховища даних. Діаграми потоків даних можуть бути представлені на різних рівнях деталізації. Основою для створення різних рівнів ДПД є принцип декомпозиції.

Декомпозиція – це процес розподілу складного об’єкта, системи, задачі на більш прості та зрозумілі компоненти або підсистеми, які легше аналізувати, проектувати, реалізовувати чи управляти ними.

Основна мета декомпозиції – спрощення складності шляхом розгляду окремих частин системи чи процесу, а також полегшення виявлення залежностей між ними.

На рисунку 3.2 зображена діаграма ДПД 0-го рівня, продемонстровано взаємодію наукового співробітника з ІС.



Рис. 3.2. Діаграма ДПД 0-го рівня [створено автором]

З метою переходу до діаграми ДПД 1-го рівня необхідно провести декомпозицію діаграми ДПД 0-го рівня.

Діаграма ДПД 0-го рівня демонструє, що на найвищому рівні абстракції існує два основні потоки – вхідні та вихідні дані наукового співробітника. Розглянувши вхідні дані більш детально можна виокремити окремі потоки даних з двох визначених попередньо (таблиця 3.2).

Таблиця 3.2. Декомпозиція ДПД 0-го рівня [створено автором].

ДПД 0-го рівня	ДПД 1-го рівня
Вхідні дані	Дані про склади
	Дані про експерименти
	Дані про записи
Вихідні дані	Дані про існуючі склади
	Дані про проведені експерименти
	Дані про існуючі записи

На базі створеної таблиці будуватиметься більш детальна діаграма ДПД 1-го рівня (рисунок 3.3).



Рис. 3.3. Діаграма ДПД 1-го рівня [створено автором]

Побудувавши ДПД 1-го рівня, аналогічним чином виконується розбір вже визначених потоків даних на більш дрібні. Процес декомпозиції носить ітеративний характер, супроводжується все більшою і більшою деталізацією окремих областей діаграми і відбувається до моменту визначення елементарних (атомарних) атрибутів.

Таблиця 3.3. Декомпозиція ДПД 1-го рівня (процес 1) [створено автором].

ДПД 1-го рівня	ДПД 2-го рівня	Атрибути
Дані про склади	Дані про склад	Назва Опис
	Коригувальні дані	Опис
	Склад до видалення	Назва
Дані про існуючі склади	Створений склад	Назва Опис
	Внесені редагування	Назва Опис
	Видалення складу	Назва Опис



Рис. 3.4. Діаграма ДПД 2-го рівня. Управління складами [створено автором]

Таблиця 3.4. Декомпозиція ДПД 1-го рівня (процес 2) [створено автором].

ДПД 1-го рівня	ДПД 2-го рівня	Атрибути
Дані про експерименти	Дані про експеримент	Дата і Час Ціл. температура Діаметер Висота Тиск Маса
	Коригувальні дані	Ціл. температура Діаметер Висота Тиск Маса
	Експеримент до видалення	Дата і Час
	Обрані експерименти	Дата і Час
Дані про проведені експерименти	Створений експеримент	Дата і Час Ціл. температура Діаметер Висота Тиск Маса
	Внесені редагування	Ціл. температура Діаметер Висота Тиск Маса
	Видалення експеримент	Дата і Час
	Отримані графіки	-

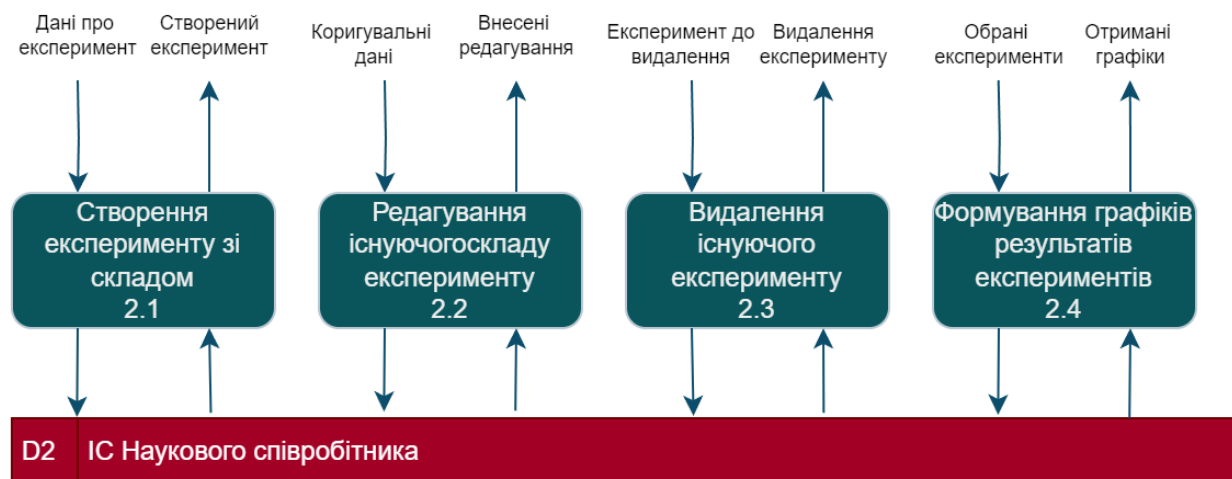


Рис. 3.5. Діаграма ДПД 2-го рівня. Управління експериментами [створено автором]

Таблиця 3.5. Декомпозиція ДПД 1-го рівня (процес 2) [створено автором].

ДПД 1-го рівня	ДПД 2-го рівня	Атрибути
Дані про записи	Дані про запис	Хвилина Усадка Температура
	Коригувальні дані	Усадка Температура
	Запис до видалення	Хвилига
Дані про існуючі записи	Створений запис	Хвилина Усадка Температура
	Внесені редагування	Усадка Температура
	Видалення запису	Хвилина



Рис. 3.6. Діаграма ДПД 2-го рівня. Управління похвилинними записами
[створено автором]

Таблиця 3.6. Декомпозиція ДПД 1-го рівня (процес 2) [створено автором].

ДПД 2-го рівня	ДПД 3-го рівня	Атрибути
Отримані графіки	Графік зміни відносної висоти засипки	Зображення
	Графік зміни в'язкості засипки	Зображення

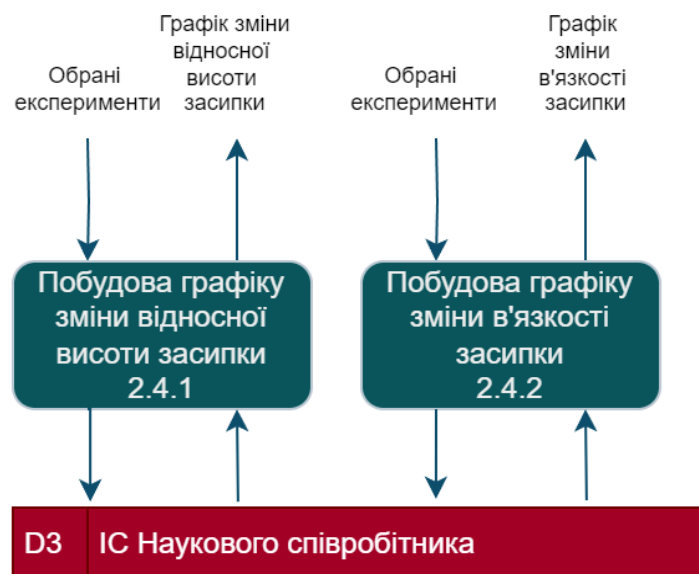


Рис. 3.7. Діаграма ДПД 3-го рівня. Аналіз та побудова графіків [створено автором]

3.1.2 Визначення діаграми станів ІС

Діаграма станів (State Diagram) — це метод графічного моделювання, один із методів моделювання методології UML[7, 9, 10], який відображає всі можливі стани системи та можливі переходи між ними. Використовується для аналізу й опису поведінки система в залежності від подій, що відбуваються в середовищі. Допомагає зрозуміти, як змінюється системи протягом її життєвого циклу.

Діаграма станів включає:

- Стани – різні положення у просторі станів, яких може набувати система.
- Події – умови що виникають в середовищі або дії зовнішніх акторів, що викликають зміну стану системи.
- Переходи – зв'язки між станами, які відображають процес трансляції з одного положення у просторі стані до іншого.

Окремо, виділяють початковий і кінцевий стани – стан системи на початку її існування та стан, у якому вона існування завершує відповідно.

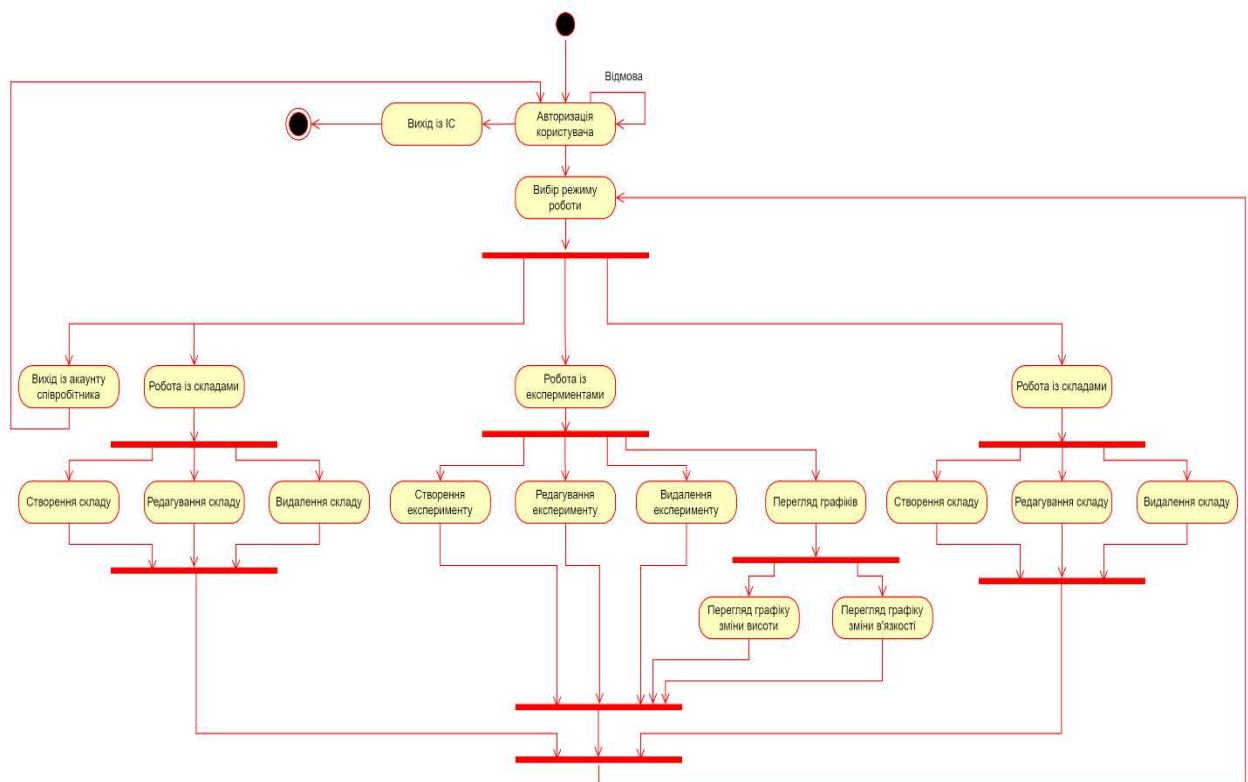


Рис. 3.4. Діаграма станів [створено автором]

3.1.3 Концептуальна модель даних

Концептуальна модель даних представляє собою формалізоване уявлення інформаційної системи на концептуальному рівні, що відображає загальну логічну структуру БД. Головною метою концептуального інфологічного проектування є створення логічної моделі бази даних, яка описує об'єкти інформаційної системи та зв'язки між ними. Модель є незалежною від конкретної СКБД і відображає інформаційні потреби користувачів системи[11].

Концептуальна модель даних ІС наукового співробітника подана на рисунку 3.8.

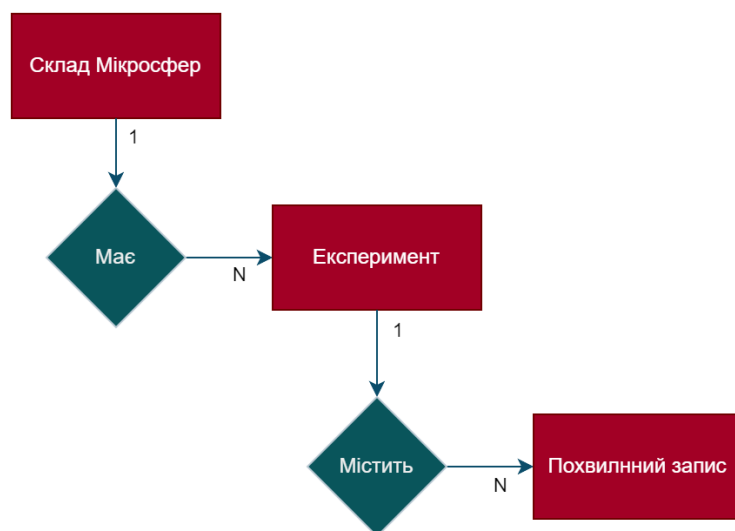


Рис. 3.8. Концептуальна модель даних ІС наукового співробітника
[створено автором]

Опис сутностей концептуальної моделі даних:

- Склад Мікросфер. Містить інформацію про назву складу, хімічний склад та фізичні характеристики мікросфер, які використовуються для проведення експериментів.
- Експеримент. Містить дані про конкретний експеримент, включаючи умови проведення (цільова температура, тиск, маса тощо).
- Похвилинний запис. Містить детальні дані, зафіксовані під час проведення експерименту, такі як поточна температура, усадка засипки та інші вимірювання, виконані у різні моменти часу.

Опис зв'язків між сутностями концептуальної моделі даних:

- «Склад Мікросфер» - «Експеримент». Тип зв'язку: 1:Б (один до багатьох), обов'язковий-обов'язковий. Один склад мікросфер може бути використаний у кількох експериментах. Кожен експеримент пов'язаний з одним конкретним складом мікросфер. Оновлення: Каскадне. Видалення: Каскадне.
- «Експеримент» - «Запис» Тип зв'язку: 1:Б (один до багатьох), обов'язковий-обов'язковий. Один експеримент може містити кілька записів даних, зафіксованих під час його проведення. Кожен запис належить до одного конкретного експерименту. Оновлення: Каскадне. Видалення: Каскадне.

3.2 Технічний проект

3.2.1 Логічна модель даних

Логічна модель даних, або логічна схема, представляє модель даних для певної предметної області, яка виражається незалежно від конкретного програмного забезпечення для управління базами даних або технології зберігання, відомої як фізична модель даних. Логічна модель оперує структурами даних, такими як реляційні таблиці і колонки, об'єктно-орієнтовані класи або теги XML[11].

Вона відрізняється від концептуальної моделі даних, яка фокусується на семантиці організації інформації, не враховуючи конкретних технологічних рішень.

Логічна модель даних ІС наукового співробітника розроблена і подана у таблицях 3.7-9.

Таблиця 3.7. Логічна модель відношення Склад [створено автором].

Назва поля	Тип даних	Модифікатори
Назва	Текстовий	Первинний ключ
Опис	Текстовий	Обов'язкове

Таблиця 3.8. Логічна модель відношення Експеримент [створено автором].

Назва поля	Тип даних	Модифікатори
Дата і Час	Дата Час	Первинний ключ
Назва Складу	Текстовий	Зовнішній ключ, Обов'язкове
Діаметер	Число	Обов'язкове
Початкова Висота	Число	Обов'язкове
Питомий Тиск	Число	Обов'язкове
Маса	Число	Обов'язкове
Цільова Температура	Число	Обов'язкове

Таблиця 3.9. Логічна модель відношення Запис [створено автором].

Назва поля	Тип даних	Модифікатори
Дата і Час Експерименту	Дата Час	Первинний ключ, Зовнішній ключ, Обов'язкове
Хвилина	Ціле	Первинний ключ
Зміна Висоти	Число	Обов'язкове
Температура	Число	Обов'язкове

ER-діаграма (Entity-Relationship Diagram) — це графічна модель, яка використовується для відображення логічної структури бази даних. Вона є інструментом концептуального проектування баз даних і слугує для моделювання зв'язків між сутностями в певній предметній області[11].

Основними елементами ER-діаграми є:

- Сутності (Entities) — об'єкти реального світу, які зберігають інформацію. Наприклад, сутностями можуть бути "Студент", "Курс" або "Викладач".

- Зв'язки (Relationships) — відношення між сутностями. Вони показують, як одна сутність взаємодіє з іншою. Наприклад, "Студент" може бути пов'язаний з "Курсом" через зв'язок "Записаний на".

В даному випадку сутностями є «Склад», «Експеримент», «Запис». ER-діаграма подана на рисунку 3.9.



Рис. 3.9. Логічна модель даних ІС наукового співробітника [створено автором]

3.2.2 Архітектура ІС

Архітектура програмного забезпечення є важливою частиною проектування системи, оскільки визначає, як її компоненти взаємодіють між собою для виконання заданих функцій. Вона передбачає структурний підхід, що включає різні аспекти системи, такі як функціональність, продуктивність, безпека та масштабованість. Створення архітектури є основою для подальшої розробки, впровадження та підтримки програми, оскільки вона визначає загальний план, якого необхідно дотримуватися під час роботи[7].

Структура архітектури програмного забезпечення зазвичай поділяється на окремі модулі, що відповідають за різні функціональні частини системи. Розробка цих модулів ґрунтується на діаграмі варіантів використання, де кожен

модуль відображає окремі функції, які виконують користувачі системи (менеджери, логісти, оператори складів тощо). Керування цими модулями здійснюється через головний модуль, який займається ідентифікацією користувачів і надає їм доступ до необхідних функцій системи, відповідно до їх прав і ролей[9].

Модуль управління складами мікросфер. Призначення: Зберігання та управління інформацією про хімічні склади та властивості досліджуваних скляних мікросфер для синтактичних піноскел. Вхідні дані: Назва складу, хімічний склад, властивості матеріалів (щільність, в'язкість, температура плавлення тощо). Дія: Додавання нових складів, редагування існуючих, видалення та перегляд складів параметрами. Вихідні дані: Інформація про склади матеріалів для подальшого використання в експериментах.

Модуль управління експериментами. Призначення: Ведення реєстру експериментів, включаючи дату проведення, початкові умови та параметри. Вхідні дані: Дата експерименту, цільова температура, тиск, діаметр засипки, маса, цільові параметри тощо. Дія: Фіксація нових експериментів, редагування даних про експеримент, видалення та перегляд. Вихідні дані: Перелік експериментів з усіма параметрами, що дозволяє науковому співробітнику мати доступ до історії проведених досліджень.

Модуль управління похвилинними записами. Призначення: Фіксація даних експерименту похвилинно, на основі вимірювань температури, усадки та інших параметрів кожну хвилину. Вхідні дані: Поточні вимірювання (температура, усадка). Дія: Запис даних кожну хвилину, можливість коригування або видалення записів. Вихідні дані: Часові ряди даних, що можна використовувати для подальшого аналізу та побудови графіків.

Модуль побудови графіків. Призначення: Автоматичне побудова графіків на основі експериментальних даних для аналізу процесів усадки засипки та зміни в'язкості. Вхідні дані: Дані про усадку засипки, в'язкість, температуру, та час. Дія: Побудова графіків за різними параметрами (висота засипки, в'язкість)

для порівняння та оцінки результатів експериментів. Вихідні дані: Графіки, що дозволяють візуалізувати зміни параметрів у часі, для подальшого аналізу.

3.2.3 Розробка діаграми розгортання

Діаграма розгортання (Deployment Diagram) – це тип діаграми методології моделювання UML (Unified Modeling Language), яка показує фізичну структуру програмної системи, її компоненти та розміщення в апаратному середовищі[10]. Основні елементи діаграми розгортання:

- Ноди – це фізичні пристрої, на яких виконуються програми, або ресурси, що використовуються системою (сервери, комп'ютери, пристрої).
- Компоненти – програмні модулі або частини програми, що розміщуються на нодах. Це можуть бути веб-сервери, бази даних, додатки тощо.
- Зв'язки – лінії, що з'єднують ноди та компоненти, відображаючи фізичні з'єднання, такі як мережеві лінії, канали передачі даних, порти.
- Інтерфейси – можуть бути присутніми для вказівки, через які канали відбувається взаємодія між компонентами та нодами.

Діаграма розгортання ІС наукового співробітника подана на рисунку 3.10.

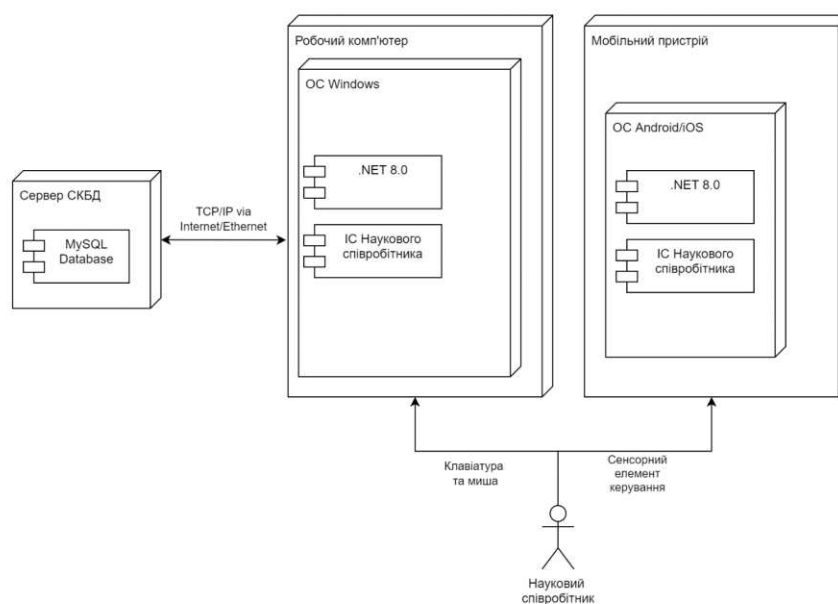


Рис. 3.10. Діаграма розгортання ІС наукового співробітника [створено автором]

3.2.4 Основні прикладні алгоритми

Алгоритм — це чітка та послідовна сукупність інструкцій або кроків, які визначають порядок виконання операцій для досягнення певної мети або розв'язання задачі. Алгоритм повинен бути чітким (кожен крок повинен бути однозначним та виконуваним) та кінцевим (кількість кроків алгоритму кінцева).

Алгоритми використовуються в розробці інформаційних систем для розв'язання задач сортування, пошуку оптимальних шляхів на графах та інших задач з обробки даних. Алгоритм може бути представлений у різних формах, зокрема у вигляді текстових інструкцій, псевдокоду, проте найнаочнішим видом представлення є блок-схема (графічний спосіб).

Блок-схема алгоритму — це графічне представлення алгоритму, яке використовує стандартизовані символи для відображення окремих етапів обробки даних або операцій. Кожен символ в блок-схемі має певне значення, і їх взаємне розташування вказує на послідовність виконання кроків алгоритму.

ІС наукового співробітника застосовує алгоритм зображений на рисунку 3.11 для отримання сумарної усадки зразка на задану хвилину експерименту.

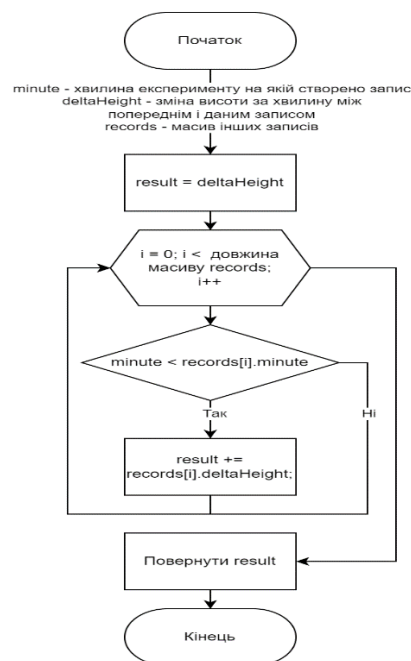


Рис. 3.11. Блок схема алгоритму підрахунку сумарної усадки зразка на дану хвилину [створено автором]

Важливою для проведення аналізу проведених експериментів є можливість сортування похвилинних записів за часом запису, усадкою, температурою тощо з метою подальшого аналізу. ІС наукового співробітника використовує алгоритм quicksort (рисунок 3.12) для виконання цієї задачі.

Quicksort — це алгоритм сортування, який ґрунтується на принципі "розділай і володарюй". Алгоритм обирає опорний елемент з масиву, розділяє масив на дві частини так, щоб елементи, менші за опорний, були зліва, а більші — справа, і рекурсивно сортує кожну частину. Алгоритм працює швидко завдяки тому, що кожне розділення зменшує проблему сортування вдвічі, що дає йому часову складність $O(n \log n)$ в середньому випадку [12].

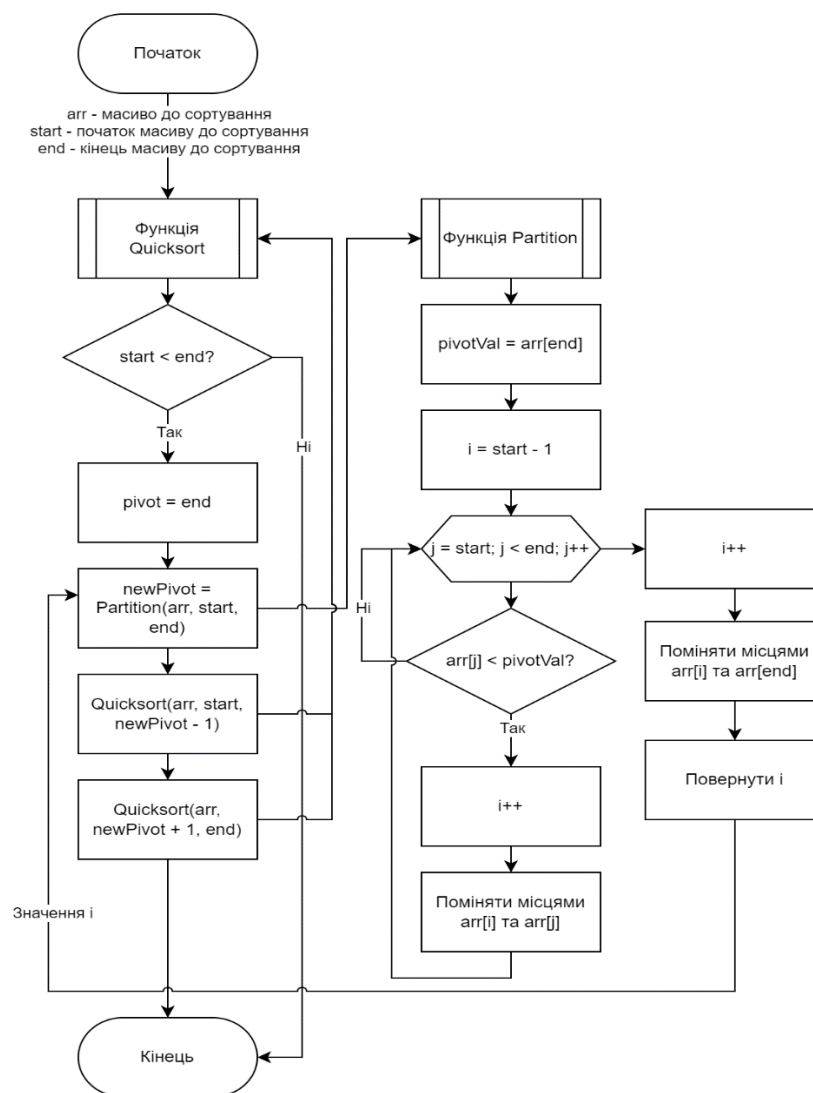


Рис. 3.12. Блок схема алгоритму сортування похвилинних записів методом quicksort [створено автором]

3.3 Робочий проект

3.3.1 Вибір засобів розробки

У якості засобу розробки графічного інтерфейсу обрано програмну мову C#. Програмна платформа - .Net 8.0. Основною бібліотекою для розробки графічного інтерфейсу виступає AvaloniaUI. Для побудови і обробки графіків відносної усадки та в'язкості використано мову програмування Python у поєднанні із модулем matplotlib. У якості СКБД обрано СКБД MySQL. Для взаємодії додатку з базою даних використано бібліотеку Entity Framework Core.

C# — це об'єктно-орієнтована мова програмування від Microsoft, яка є частиною платформи .NET. Вона широко використовується для створення різноманітних додатків, від настільних до веб та мобільних, і підтримує парадигми ООП, функціональне програмування та асинхронні операції. Перевагами C# є сучасний синтаксис та статична сувора типізація[13].

.NET 8.0 — це остання версія платформи для розробки кросплатформених додатків. Основні переваги .NET 8.0: великий набір стандартних та користувацьких бібліотек, кросплатформенність, висока продуктивність, вбудовані інструменти для тестування та дебагінгу.

AvaloniaUI — це кросплатформена бібліотека для розробки графічних інтерфейсів, що підтримує Windows, Linux, macOS, а в останніх версіях Android та iOS. Вона використовує MVVM-патерн, що сприяє чистоті коду та ефективному управлінню станами програми. Переваги AvaloniaUI: кросплатформенність, схожість синтаксису та процесу розробки до WPF, повна інтеграція з платформою .NET та її набором базових бібліотек.

Python — це високоуровнева, інтерпретована мова програмування, яка розроблена з акцентом на простоту читання коду та зручність у використанні. Python підтримує об'єктно-орієнтоване та функціональне програмування. Переваги Python: зрозумілий синтаксис, велика кількість бібліотек та підтримка різноманітних фреймворків для наукових обчислень та аналізу даних[23].

matplotlib — бібліотека Python для побудови графіків і візуалізації даних. Вона дозволяє створювати статичні та інтерактивні графіки, що зручно для аналізу даних у наукових дослідженнях. Основні переваги: підтримка різноманітних типів графіків, висока налаштовуваність.

MySQL — реляційна СКБД, що дозволяє зберігати і обробляти великі обсяги даних за допомогою SQL. Вона підтримує швидке виконання запитів і масштабованість. Переваги: висока швидкість роботи, надійність і підтримка складних запитів[11, 14].

Entity Framework Core — це ORM для .NET, що дозволяє працювати з базами даних за допомогою об'єктно-орієнтованого підходу. Автоматично генерує SQL-запити з моделей. Переваги: зручність у написанні коду без SQL-запитів, підтримка різних СКБД, підвищена продуктивність розробки[15].

3.3.2 Фізична модель даних

Фізична модель даних — це конкретне представлення даних, яке описує зберігання інформації безпосередньо в контексті обраної СКБД. Фокусується на технічних аспектах організації даних і включає визначення того, як таблиці, індекси, зв'язки між даними та інші елементи будуть реалізовані в обраній СКБД [11].

Основні аспекти фізичної моделі включають визначення структури зберігання даних, типів даних для кожного поля, індексів для прискорення доступу до даних, механізмів доступу, нормалізації для уникнення дублювання даних, а також методів резервного копіювання та відновлення даних. Фізична модель також враховує, як дані будуть розподілені по дисках та іншому обладнанні, оптимізуючи процеси зберігання та обробки запитів. Вона сприяє забезпеченню надійності, безпеки та продуктивності бази даних, адаптуючи її до специфічних вимог системи.

Фізична модель даних розроблена та зафіксована у таблицях 3.10-12.

Таблиця 3.10. Фізична модель відношення Запис [створено автором].

Логічна назва поля	Фізична назва поля	Тип та довжина даних	Модифікатори
Назва	name	VARCHAR(30)	Primary Key, Not NULL
Опис	description	VARCHAR(225)	Not NULL

Таблиця 3.11. Фізична модель відношення Експеримент [створено автором].

Логічна назва поля	Фізична назва поля	Тип та довжина даних	Модифікатори
Дата і Час	name	DATETIME	Primary Key, Not NULL
Назва Складу	compositionName	VARCHAR(30)	Foreign Key, Not NULL
Діаметер	diameter	DOUBLE	Not NULL
Початкова висота	startHeight	DOUBLE	Not NULL
Маса	mass	DOUBLE	Not NULL
Цільова Температура	targetTemperature	DOUBLE	Not NULL

Таблиця 3.12. Фізична модель відношення Записів [створено автором].

Логічна назва поля	Фізична назва поля	Тип та довжина даних	Модифікатори
Дата і Час Експерименту	experimentDate Time	DATETIME	Primary Key, Foreign Key, Not NULL
Хвилина	compositionName	INT	Primary Key, Not NULL
Зміна Висоти	diameter	DOUBLE	Not NULL

Логічна назва поля	Фізична назва поля	Тип та довжина даних	Модифікатори
Температура	temperature	DOUBLE	Not NULL

Візуалізація фізичної моделі даних створено в програмному засосунку MySQL Workbench та портована на сервер MySQL командою Forward Engineering.

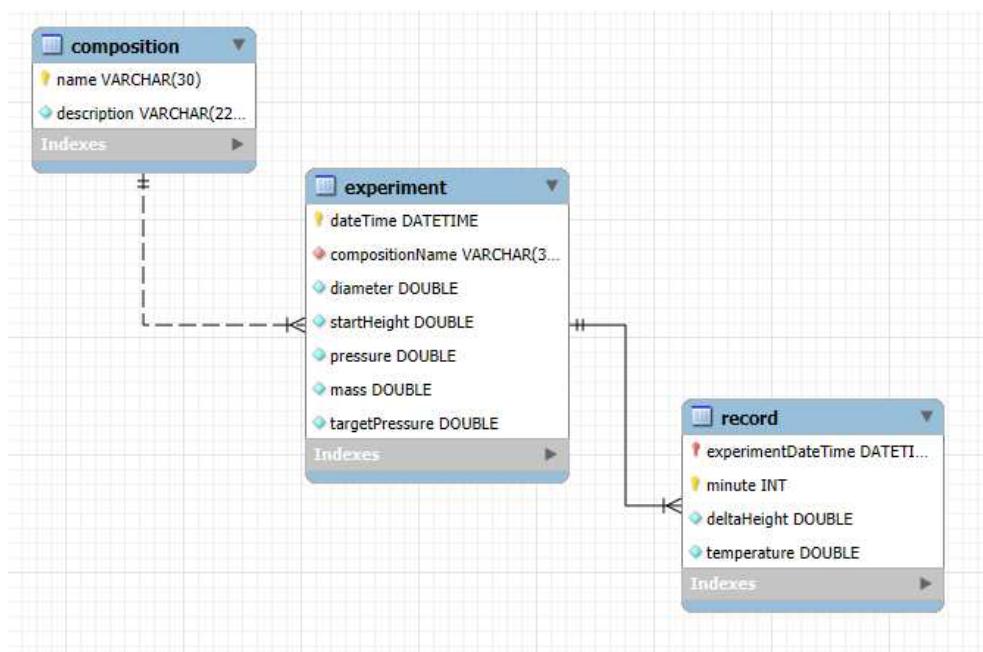


Рис. 3.13. Фізична модель даних [створено автором]

3.3.3 Кодування ІС

Процес кодування — це етап розробки інформаційної системи, під час якого розробник перетворює логіку програми та алгоритми у доступний для виконання для комп'ютера формат, використовуючи конкретну мову програмування та компілятор або інтерпритатор[7].

Процес кодування є невід'ємною частиною розробк, який вимагає від розробника високих навичок і знань у вибраному інструменті розробки, а також здатності працювати з іншими етапами розробки, такими як тестування та розгортання.

Кодування окремих модулів може відбуватися паралельно, за умови що модулі досить незалежні, а інтерфейси взаємодії між модулями попередньо узгоджені.

Першим реалізованим модулем став модуль даних. На основі розгорнутої на сервері MySQL схеми даних створено відповідні класи та логіку взаємодії із СКБД.

На наступному кроці було розроблено інтерфейс програми – основне вікно, вікна для роботи із складами, експериментами, записами (рисунки 3.14-17). Вікна також інкапсують логіку роботи із модулем даних.

LMOULD SF Viscosity Calculator

Composition

Composition: Composition #2

Description: Default composition #2.

Experiments

Experiments:

- 20.02.1999 19:30
- 01.03.1999 17:00
- 03.03.1999 19:00

Preconditions:

Target t°: 550.0000 C°
Diameter: 21.0000 mm
Start height: 52.5000 mm
Pressure: 12.0000 kg/cm2
Mass: 2.9500 g
Start p: 0.1622 g/cm3

Results:

Total time: 37 min
Final height: 39.7810 mm
Final p: 0.2141 g/cm3

PLOT

Records

Minute	Δh (mm)	Total Δh (mm)	t° (C°)
1	0.3500	0.3500	435.0000
2	0.7200	1.0700	475.0000
3	0.9000	1.9700	500.0000
4	1.3800	3.3500	525.0000
6	1.6000	4.9500	540.0000
7	1.9000	6.8500	550.0000
8	1.3000	8.1500	550.0000
9	0.8000	8.9500	550.0000
10	0.4700	9.4200	550.0000
11	0.5000	9.9200	550.0000
12	0.1900	10.1100	550.0000
13	0.2600	10.3700	550.0000
14	0.2400	10.6100	550.0000
15	0.1700	10.7800	550.0000
16	0.1900	10.9700	550.0000
17	0.1700	11.1400	550.0000
18	0.1200	11.2600	550.0000
19	0.1200	11.3800	550.0000
20	0.1600	11.5400	550.0000
21	0.1100	11.6500	550.0000
22	0.0090	11.6590	550.0000
23	0.1100	11.7690	550.0000
24	0.1000	11.8690	550.0000
25	0.0600	11.9290	550.0000
27	0.1700	12.0990	550.0000
28	0.0700	12.1690	550.0000
29	0.0800	12.2490	550.0000
30	0.0700	12.3190	550.0000
31	0.0700	12.3890	550.0000
32	0.0600	12.4490	550.0000

Рис. 3.14. Основне вікно програми [створено автором]

Рис. 3.15. Створення складу мікросфер [створено автором]

Рис. 3.16. Редагування експерименту [створено автором]

Рис. 3.17. Створення похвилинного запису [створено автором]

Для побудови графіків відносної усадки та зміни в'язкості, як було зазначено, використано мову програмування Python у поєднанні із бібліотекою matplotlib. Для цього було створено скрипт plot.py (лістинг 3.1). Результати роботи скрипту продемонстровано на рисунку 3.18.

Лістинг 3.1. Скрипт побудови графіків plot.py [створено автором].

```
from cProfile import label
```

```

from operator import imod
from matplotlib import markers
import matplotlib.pyplot as plt
from scipy.interpolate import interp1d
import numpy as np
import json
import sys
DATA_FILE = sys.argv[1]
XVALS_NAME = "XVals"
YVALS_NAME= "YVals"
TITLE_NAME= "Title"
SCALE_NAME= "Scale"
with open(DATA_FILE) as f:
    data = json.load(f)
plt.xlabel(sys.argv[3]);
plt.ylabel(sys.argv[4]);
for v in data:
    x = np.array(v[XVALS_NAME])
    y = np.array(v[YVALS_NAME])
    interpolated_model = interp1d(x, y, kind = "cubic")
    X_ = np.linspace(x.min(), x.max(), 500)
    Y_ = interpolated_model(X_)
    t = plt.plot(X_, Y_, label=v[TITLE_NAME], marker
="o",markevery=10)
    plt.yscale(v[SCALE_NAME])
    plt.legend(loc="lower right")
plt.title(sys.argv[2])
plt.grid()
plt.show()

```

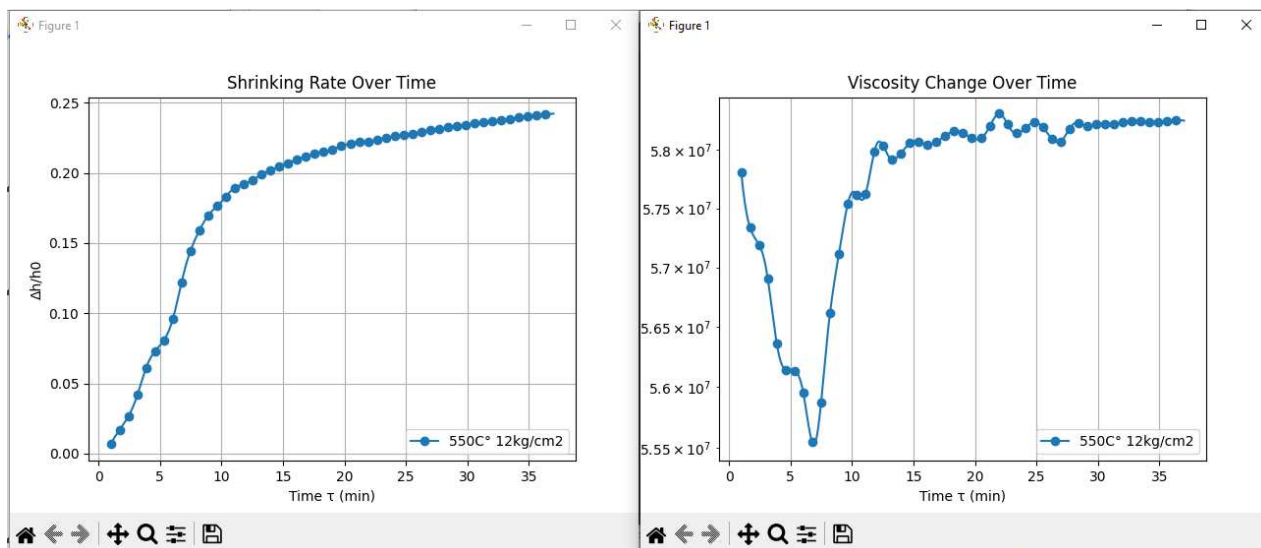


Рис. 3.18. Побудова графіків усадки та зміни в'язкості [створено автором]

Для взаємодії із скрипт plot.py із середовища C# застосовано підхід, що полягає у виконанні скрипту через командний рядок (лістинг 3.2). Для обміну даних використовується окремий файл формату JSON, назва якого передається скрипту як аргумент командного рядка. Скрипт підтримує роботу із багатьма наборами даних одночасно для виконання порівняння графіків(рисунок 3.19).

Лістинг 3.2. Виконання скрипту шляхом його запуску через командний рядок [створено автором].

```
Process.Start("python", @"py\plot.py py\viscosityPlotData.json " +
"\"Viscosity Change Over Time\" \"Time τ (min)\" \"log η (Ba/s)\"");
```

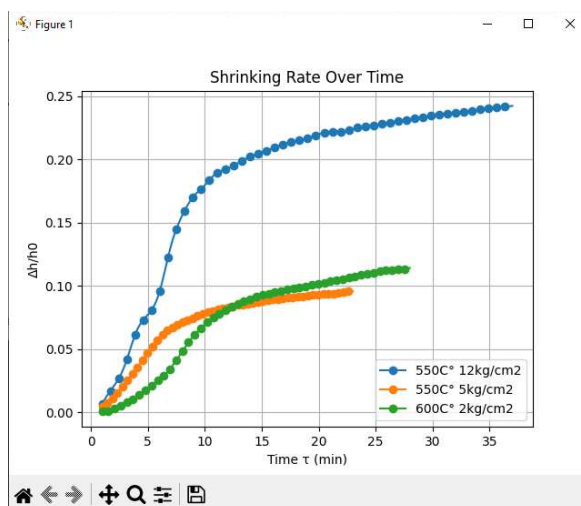


Рис. 3.19. Режим порівняння графіків [створено автором]

Для підтримки роботи в мобільному додатку інтерпретатор python включений в основний виконуваний пакет програми (рисунок 3.20).



Рис. 3.20. Робота мобільного додатку [створено автором]

3.3.4 Тестування ІС

Тестування — це процес встановлення відповідності між реальною та очікуваною поведінкою програми. В більш широкому сенсі, тестування є важливою частиною контролю якості, що включає в себе планування робіт, розробку тестів, їх виконання та аналіз результатів[9, 16].

Існує кілька основних методів тестування:

- Статичне тестування — це перевірка програмного забезпечення без його виконання.
- Динамічне тестування — передбачає тестування програмного продукту в реальних умовах експлуатації.
- Сліпе тестування — зосереджується на перевірці функціональності програми, не враховуючи її внутрішню структуру чи реалізацію.
- Відкрите тестування — орієнтоване на перевірку внутрішньої логіки програми, включаючи коректність роботи окремих елементів програми та їх взаємодію.
- Unit тестування — це метод тестування, при якому перевіряються окремі компоненти програмного забезпечення (наприклад, функція чи метод) на відповідність вимогам.

Основними техніками, які були застосовані, стали поділ на класи еквівалентності та аналіз граничних значень.

Клас еквівалентності – це набір вхідних даних, обробка яких призводить до одного і того ж результату. Техніка поділу на класи еквівалентності дозволяє зменшити кількість тестів, видаляючи дублюючі перевірки для однакових груп значень, що мають однаковий вплив на систему.

Аналіз граничних значень – це метод тестування, який фокусується на перевірці поведінки системи при введенні значень на межах класів еквівалентності[16].

Приклад визначення класу еквівалентності поданий у таблиці 3.13. Лістинг 3.4 демонструє перевірку введеної користувачем назви складу. Застосунок запобігає ведення невірних вхідних даних користувачем і робить кнопку збереження неактивною у разі, якщо дані не проходять перевірку.

Таблиця 3.13. Класи еквівалентності параметру Назва Складу [створено автором].

№	Параметр, що перевіряється	Вірно	Невірна довжина рядку (<3)	Назва «Composition #1» вже в базі
1	Назва складу	Composition #3	-	-
2		-	Co	-
3		-	-	Composition #1

Лістинг 3.3. Перевірка правильності введеної назви складу [створено автором].

```
saveButton.IsEnabled = nameTextBox.Text.Length >= 3 &&
db.Compositions.Find(nameTextBox.Text) == null;
```

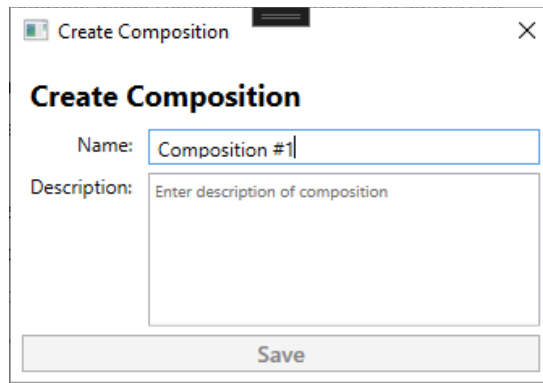


Рис. 3.21. Запобігання збереження помилкових даних («Composition #1» вже в базі) [створено автором]

Висновки до розділу

У розділі представлено розробку і практичну реалізацію інформаційної системи у дослідницькій лабораторії. Розроблено модель інформаційної системи, включаючи діаграму варіантів використання, визначення потоків даних, діаграму станів та концептуальну модель даних. Створено технічний проект, що містить логічну модель даних, архітектуру інформаційної системи, діаграму розгортання та основні прикладні алгоритми. Виконано робочий проект, де обрано засоби розробки, створено фізичну модель даних, проведено кодування і тестування інформаційної системи.

РОЗДІЛ 4. ОБГРУНТУВАННЯ ЕКОНОМІЧНОЇ ДОЦІЛЬНОСТІ РОЗРОБКИ

4.1 Розрахунок вартості матеріально-енергетичних витрат

Для забезпечення безперебійного процесу розробки ІС замовник повинен надавати працівникам усі необхідні матеріальні ресурси. Будь-який виробничий процес, супроводжується витратами як трудових, так і матеріально-енергетичних ресурсів. Ефективне управління цими витратами вимагає завчасного планування, яке включає визначення ключових етапів роботи, розподіл обов'язків серед співробітників, встановлення термінів виконання, а також розрахунок потреби в людських, матеріальних і технічних ресурсах.

Витрати можна визначити як зменшення економічних вигід підприємства, що виникає внаслідок вибуття активів або збільшення зобов'язань, що, у свою чергу, зменшує власний капітал.

Матеріальні витрати – витрати на придбання та використання матеріалів, комплектуючих, енергії, палива, запасних частин та інших ресурсів, залучених в операційній діяльності підприємства.

Проектування і розробка ІС, подібно до інших видів виробничої діяльності, також передбачає певні матеріальні витрати. До таких витрат можна віднести використання паперу, флеш-накопичувачів, чорнильних картриджів для принтерів, канцелярських приладь, витрати на електроенергію, доступ до Інтернету та інших ресурсів, необхідних для створення програмної системи [17].

Таблиця 5.1. Розрахунок вартості матеріальних витрат.

Найменування матеріальних затрат	Кількість	Одиниця виміру	Ціна за од. (грн)	Загальна сума в грн.
Степлер канцелярський для паперу на 50 аркушів	1	шт.	274,00	274,00

Найменування матеріальних затрат	Кількість	Одиниця виміру	Ціна за од. (грн)	Загальна сума в грн.
Упаковка папок для паперів картонних 50 шт	3	пачка	473,00	1419,00
Ящик паперу офісного Zoom 80г/м ² 500 аркушів	2	ящик	859,00	1712,00
Набір ручок кулькових BIC	1	пачка	135,00	135,00
Чорнило для струменевого принтера Epson	4	бутылка	613,00	2452,00
Кабель Apple Lightning to USB	1	шт.	999,00	1998,00
Кабель Samsung USB Type-C	1	шт.	459,00	459,00
Флеш пам'ять USB Transcend	2	шт.	474,00	948,00
Всього				9397,00

Транспортно-заготівельні витрати становлять 21% від загальної вартості матеріалів. Таким чином, обраховане значення ТЗВ дорівнює 1973,37 грн.

Сукупні матеріальні ($S_{\text{мат}}$) витрати необхідні для розробки програмного забезпечення, становлять 11370,37 грн.

Як і будь-яке виробництво, створення програмного продукту супроводжується енергетичними витратами. До них належать витрати на різні види палива та енергії, включаючи електроенергію, пару, гарячу воду тощо. У випадку розробки програмного забезпечення основну частину енергетичних витрат становить споживання електроенергії.

$$S_{\text{ел/ен}} = C_{\text{комп}} \cdot W_{\text{ден}} \cdot W_{\text{год}} \cdot P \cdot A, \quad (4.1)$$

де $S_{\text{ел/ен}}$ – вартість електроенергії;

$C_{\text{комп}}$ – споживана потужність одного комп'ютера;

$W_{\text{ден}}$ – кількість робочих днів;

$W_{\text{год}}$ – середня тривалість роботи в день в годинах;

P – вартість 1кВт-год електроенергії;

A – кількість комп'ютерів.

Низьке споживання електроенергії принтером та мобільними пристроями, у поєднанні із невеликою тривалістю їх роботи порівнювано із комп'ютерами дозволяє знехтувати енергетичними витратами на їх роботу та зосередитися лише на комп'ютерах.

Споживання персонального комп'ютера за паспортом становить 650 Вт/год, або 0,65 кВт/год. Середня тривалість роботи ПК за день сягає 7,3 годин. Таким чином для двох таких пристроїв витрати на оплату споживання електроенергію складатимуть.

$$S_{\text{ел/ен}} = 0,65 \cdot 30 \cdot 7,3 \cdot 5,29 \cdot 2 = 1506,06.$$

Матеріально-енергетичні витрати складається із суми витрат відповідно матеріальних та енергетичних.

$$S = S_{\text{мат}} + S_{\text{ел/ен}}, \quad (4.2)$$

де $S_{\text{мат}}$ – матреальні витрати.

$$S = 11370,37 + 1506,06 = 12876,43.$$

Таким чином матеріально-енергетичні витрати на розробку складатимуть 12876,43 грн.

4.2 Розрахунок фонду оплати праці за виконану роботу

Оплата праці, або заробітна плата, є грошовим вираженням вартості та ціни робочої сили, яка виплачується працівнику за виконану ним роботу. Вона є основним показником, що відображає рівень трудового внеску працівника,

залежить від результатів діяльності підприємства та регулюється податковими нормами.

Заробітна плата не обмежується максимальними розмірами і може варіюватися в залежності від умов праці, кваліфікації працівника та інших факторів.

Оплата праці включає основну заробітну плату та додаткову оплату праці, які складають приблизно 70% і 30% від загальної суми відповідно. Основна заробітна плата визначається за виконану роботу відповідно до встановлених ставок і норм, тоді як додаткова оплата включає різні доплати, премії та бонуси, що залежать від результатів праці, кваліфікації працівника та інших факторів.

Фонд основної заробітної плати включає суму, нараховану за виконану роботу відповідно до встановлених розцінок, тарифних ставок або посадових окладів. Це також включає премії за перевиконання норм виробітку (за умови, що премії не є одноразовими), доплати за роботу в нічний час, понаднормові години та оплати за простої, які виникають не з вини робітників.

Фонд додаткової заробітної плати охоплює надбавки, доплати, що регулюються законодавством, а також виплати за щорічні та додаткові відпустки, а також оплати за час, коли працівники виконують державні чи громадські обов'язки[17, 18].

Загальний фонд оплати праці працівника формується як сума основної та додаткової заробітної плати.

$$F = \rho_{\text{осн}} + \rho_{\text{дод}}, \quad (4.3)$$

де $\rho_{\text{осн}}$ – основа заробітня плата;

$\rho_{\text{дод}}$ – додаткова заробітня плата.

$$\rho_{\text{осн}} = L * \bar{w}, \quad (4.5)$$

де L – трудомісткість виконання робіт;

$\bar{w}$ – середньогодинна тарифна ставка.

$$\bar{w} = \frac{\bar{w}_1 + \bar{w}_2 + \dots + \bar{w}_n}{n}, \quad (4.4)$$

де n – кількість робітників;

$\bar{w}_1, \bar{w}_2, \dots, \bar{w}_n$ – серед. год. тарифна ставка n -го виконавця.

Середня годинна тарифна ставка інженера-програміста становить 330,00 грн/год, а програміста-тестувальника – 290,00 грн/год.

$$\bar{w} = \frac{330,00 + 290,00}{2} = 310,00,$$

$$\rho_{\text{осн}} = 674 \cdot 310,00 = 208940,00.$$

Додаткова заробітна плата включає премії та бонуси та складає, за даними установи 25%.

$$\rho_{\text{дод}} = \rho_{\text{осн}} \cdot 0,25, \quad (5.6)$$

$$\rho_{\text{дод}} = 208940,00 \cdot 0,25 = 52235,00,$$

$$F = 208940,00 + 52235,00 = 261175,00.$$

Середня заробітна плата визначається на основі розрахованого фонду оплати праці. Середньоденна $\rho_{\text{ден}}$ одного працівника визначається за формулою:

$$\bar{\rho}_{\text{ден}} = \frac{F}{4W_{\text{ден}}}, \quad (4.7)$$

$$\bar{\rho}_{\text{ден}} = \frac{261175,00}{4 \cdot 30} = 2176,45.$$

Середня заробітна плата одного працівника за день складає 2176,45 грн. Розрахуємо середню заробітну плату одного працівника за годину за формулою:

$$\bar{\rho}_{\text{год}} = \frac{F}{W_{\text{год}}}, \quad (4.8)$$

$$\bar{\rho}_{\text{год}} = \frac{261175,00}{674} = 387,50.$$

Середня заробітна плата одного працівника за годину складає 387,50 грн.

Єдиний соціальний внесок (ЄСВ) є обов'язковим платежем, що сплачується до системи загальнообов'язкового державного соціального страхування в Україні. Мета — забезпечення страхових виплат за різними видами соціального страхування, такими як пенсійне, медичне, страхування від безробіття, а також для покриття витрат на інші соціальні послуги, передбачені законодавством.

$$T_{\text{ЄСВ}} = 0,22 \cdot F, \quad (4.9)$$

$$T_{\text{ЄСВ}} = 0,22 \cdot F = 57458,50$$

Таким чином ЄСВ сягатиме 57458,50 грн.

4.3 Розрахунок ціни розробки програми

Собівартість продукції — це грошовий вираз витрат, пов'язаних з підготовкою виробництва, виготовленням і збутом продукції. Вона комплексно відображає ефективність використання всіх ресурсів підприємства, що, в свою чергу, характеризує рівень технічного забезпечення, технології та організації виробництва.

Розраховується за такимим статтями:

- «Сировина і матеріали» включає витрати на матеріали для комп'ютерних робіт, інтернет-послуги, а також транспортно-заготівельні витрати (транспортування, вантажно-розвантажувальні роботи).
- «Енергія технологічна» включає витрати на електроенергію для роботи необхідних пристроїв відповідно до їх характеристик і тарифів.
- «Основна заробітна плата» включає витрати на зарплату працівників, що займаються розробкою інформаційної системи.
- «Додаткова заробітна плата» включає нарахування на основну зарплату у вигляді додаткових виплат для працівників.
- «Відрахування на соціальні заходи» включають ЄСВ на рівні 22% від фонду заробітної плати.
- «Амортизаційні відрахування» включають відрахування на відшкодування не ремонт і компенсацію зносу пристроїв.
- «Загальновиробничі витрати» включають витрати на утримання технічних засобів програмування.

Для розрахунку амортизації використано кумулятивний метод. Мінімально допустимий термін використання електронних пристроїв сягає лише 2 років(у зв'язку зі швидкими темпами застарівання). Річна сума амортизації визначається як добуток вартості яка амортизується на кумулятивний коефіцієнт[18].

$$R_{\text{рік}} = v \cdot k_{\text{кум}}, \quad (4.10)$$

де v – різниця первісної та ліквідаційних вартостей;

$k_{\text{кум}}$ – кумулятивний коефіцієнт.

$$v = v_0 - v_1, \quad (4.11)$$

де v_0 – первісна вартість основних засобів;

v_1 – ліквідаційна вартість основних засобів.

$$k_{\text{кум}} = \frac{\sum t_{\text{кум}}}{\sum t_1 + t_2 + \dots + t_m}, \quad (4.12)$$

Кумулятивний коефіцієнт першого року складає:

$$k_{\text{кум}1} = \frac{2}{1+2} = 0,67.$$

Кумулятивний коефіцієнт другого року:

$$k_{\text{кум}2} = \frac{1}{1+2} = 0,33.$$

Вартість принтера дорівнює 5000,00 грн., а його ліквідаційна вартість передбачається в сумі 1100,00 грн.

$$v = 5000,00 - 1100,00 = 4900,00,$$

$$R_{\text{рік.принт}} = 4900,00 \cdot 0,67 = 3283,00 \text{ (1 рік експлуатації)}.$$

Розмір амортизаційних відрахувань за місяць визначаємо за формулою:

$$R_{\text{міс.принт}} = \frac{R_{\text{рік}}}{12}, \quad (4.13)$$

$$R_{\text{міс.принт}} = \frac{3283,00}{12} = 273,58.$$

Програмний продукт розробляється 4 місяці.

$$R_{\text{принт}} = 273,58 \cdot 4 = 1094,32.$$

Обидва комп'ютери коштують 23500,00 грн, були введені в експлуатацію одночасно і наразі працюють вже другий рік. Їх очікувана ліквідаційна вартість становить 11500,00 грн.

$$v = 47000,00 - 23000,00 = 24000,00,$$

$$R_{\text{рік.комп}} = 24000,00 \cdot 0,33 = 7920,00,$$

$$R_{\text{міс.комп}} = \frac{7920,00}{12} = 660,00,$$

$$R_{\text{комп}} = 660,00 \cdot 4 = 2640,00.$$

Амортизація для мобільних пристроїв не обраховується, адже працівники погодилися надати власні для проведення тестування створюваної ІС. Тоді сумарна амортизація становитиме:

$$R = R_{\text{комп}} + R_{\text{принт}}, \quad (4.14)$$

$$R = 2640,00 + 1094,32 = 3734,32.$$

Таблиця 4.2. Розрахунок ціни розробки програми.

Статті витрат	Методика розрахунку	Сума грн.
1. Матеріальні та ТЗВ	Табл. 4.1	9397,00
2. Вартість енергетичних витрат	Розрахунок за формулою 4.1-2	1506,06
3. Основна заробітна плата	За формулою 4.2-2	208940,00
4. Додаткова заробітна плата	За формулою 4.2-4	52235,00
5. Єдиний соціальний внесок	За формулою 4.2-8	57458,50
6. Амортизаційні відрахування	За формулою 4.3-6	3734,32
7. Загальновиробничі витрати	91% від $Z_{\text{осн}}$	190135,40
8. Виробнича собівартість	Сума всіх попередніх статей	523406,28
9. Позавиробничі витрати	12% від Ст.8	62808,75
10. Адміністративні витрати	70% від Ст.3	146258,00
11. Повна собівартість	Сума 3-х попередніх статей	732473,03
12. Рівень рентабельності	15 % від Ст.11	109870,95
13. Ціна розробки	Сума 2-х попередніх статей	842344,00

Отже ціна розробки складає 842344,00 грн.

4.4 Розрахунок економічної ефективності та строку окупності програмного забезпечення

Економічна ефективність – це отримання максимальних результатів при мінімальних витратах людських зусиль та матеріальних ресурсів.

Строк окупності – це період, необхідний для відшкодування витрат на реалізацію проекту або повернення вкладених підприємством коштів за рахунок доходів, отриманих від основної діяльності за цим проектом[18].

$$E = \Delta M - k_{\text{еф}} \cdot U, \quad (4.15)$$

де ΔM – вивільнені кошти після впровадження програмного забезпечення;

$k_{\text{еф}}$ – коефіцієнт економічної ефективності;

U – розрахована виробнича собівартість.

$$\Delta M = F_p - Q, \quad (4.16)$$

де Q – експлуатаційні витрати.

$$Q = R + S, \quad (4.17)$$

$$Q = 3734,32 + 12876,43 = 16610,75.$$

З урахуванням того, що після створення програмного продукту звільняється один працівник, це дозволяє заощадити річний фонд заробітної плати цього співробітника. Якщо середня місячна заробітна плата працівника за даними базової установи становить 33175,00 грн, то річна економія фонду заробітної плати складе:

$$F_{p12} = 33175,00 \cdot 12 = 398100,00,$$

$$\Delta M = 398100,00 - 16610,75 = 381489,25.$$

Обчислимо економічну ефективність (вивільнені кошти).

$$E = 381489,25 - 0,67 \cdot 523406,28 = 30807,096.$$

Обчислимо строк окупності.

$$H = \frac{U}{\Delta M}, \quad (4.18)$$

$$H = \frac{523406,28}{381489,25} = 1,3.$$

Отже строк окупності складе 1,37 років, або 16,44 місяців.

Висновки до розділу

У розділі обгрунтовано економічну доцільність розробки інформаційної системи. Виконано розрахунок вартості матеріально-енергетичних витрат.

Проведено розрахунок фонду оплати праці за виконану роботу. Розраховано ціну розробки програми. Виконано розрахунок економічної ефективності та строку окупності програмного забезпечення.

РОЗДІЛ 5. Охорона праці

5.1 Умови праці та організація робочого місця

Організація робочого місця розробника інформаційної системи (ІС) повинна відповідати ергономічним та санітарно-гігієнічним нормам, передбаченим українським законодавством. Зокрема, Закон України «Про охорону праці» (ст. 13) зобов'язує роботодавця створювати безпечні умови праці, враховуючи специфіку використання комп'ютерної техніки. Робоче місце має бути обладнане стільцем з регульованою висотою сидіння та спинки, а також столом, що відповідає зросту працівника та дозволяє вільно розташувати ноги.

Відповідно до Державних санітарних норм і правил роботи з комп'ютерами (ДСанПіН 3.3.2.007-98), монітор повинен розташовуватися на відстані 50–70 см від очей, під кутом, що не викликає відблисків. Робоче місце необхідно забезпечити відповідним освітленням: рівень освітленості має становити не менше 300 люкс. Для зменшення втоми очей законодавство рекомендує обладнати монітори антивідблисковими фільтрами та забезпечити регулярні перерви: 10 хвилин відпочинку після кожної години роботи за комп'ютером[19].

Санітарно-гігієнічні вимоги також зобов'язують роботодавця дотримуватись температурного режиму у приміщенні (18–25 °С) та підтримувати відносну вологість повітря на рівні 40–60%. Забезпечення якісної вентиляції чи встановлення кондиціонерів є обов'язковими для створення комфортного мікроклімату, що сприяє збереженню працездатності працівників.

5.2 Техніка безпеки під час роботи з обладнанням та програмним забезпеченням

Дотримання техніки безпеки регламентується Законом України «Про охорону праці» та низкою інших нормативно-правових актів. Відповідно до цих

документів, роботодавець зобов'язаний проводити інструктаж із безпеки праці, а працівник – дотримуватись встановлених правил. Зокрема, всі електронні пристрої повинні бути заземленими, а їх використання має відповідати нормам електробезпеки (ПУЕ – Правила улаштування електроустановок). Важливо уникати перевантаження електромережі, яке може призвести до короткого замикання чи займання[20].

Під час роботи з конфіденційними даними розробники повинні забезпечувати їх захист відповідно до Закону України «Про захист інформації в інформаційно-комунікаційних системах». Це включає створення резервних копій даних, використання ліцензійного програмного забезпечення та регулярне оновлення систем безпеки. Законодавство також вимагає проводити періодичний аудит кібербезпеки для виявлення можливих загроз і їх своєчасного усунення.

Згідно з ДСанПіН 3.3.2.007-98, комп'ютерне обладнання повинно проходити технічне обслуговування для підтримання його працездатності. Роботодавець зобов'язаний забезпечити регулярний огляд і ремонт техніки, а також провести навчання працівників щодо безпечного поводження з апаратним забезпеченням. У разі виникнення аварійних ситуацій має бути розроблений алгоритм дій, включаючи евакуацію з приміщення.

Дотримання вимог законодавства щодо охорони праці дозволяє забезпечити безпечні та комфортні умови для роботи розробника, зменшити ризик нещасних випадків і зберегти високу продуктивність працівника.

Висновки до розділу

У розділі розглянуто питання охорони праці. Описано умови праці та організацію робочого місця. Розглянуто техніку безпеки під час роботи з обладнанням та програмним забезпеченням.

РОЗДІЛ 6. Охорона навколишнього середовища

6.1 Мінімізація екологічного сліду під час розробки ІС

Розробка інформаційних систем (ІС) потребує значних ресурсів, включаючи електроенергію та матеріали для обладнання. Для мінімізації екологічного сліду важливо враховувати принципи енергоефективності та ресурсозбереження на всіх етапах проєкту. Згідно з Законом України "Про охорону навколишнього природного середовища", суб'єкти господарської діяльності зобов'язані здійснювати заходи для зменшення негативного впливу на довкілля (ст. 4). При розробці ІС необхідно використовувати енергоефективне обладнання, застосовувати оптимальні методи програмування для зменшення споживання електроенергії, а також впроваджувати технології, які дозволяють скоротити викиди парникових газів. Також доцільно передбачити можливість рециркуляції матеріалів та компонентів, що використовуються під час розробки та експлуатації ІС, відповідно до законодавства України про поводження з відходами.

Необхідною складовою є впровадження принципів зеленої ІТ, яка передбачає використання відновлюваних джерел енергії, таких як сонячна або вітрова енергія, для живлення ІС. Це не лише сприяє зменшенню викидів вуглекислого газу, але й знижує залежність від викопних видів палива. Закон України "Про альтернативні джерела енергії" (ст. 3) підтримує розвиток та використання відновлюваних джерел енергії, що робить цей підхід легітимним і важливим для збереження довкілля.

Слід враховувати екологічні аспекти при виборі постачальників обладнання та програмного забезпечення. Вибір партнерів, які дотримуються принципів сталого розвитку та екологічної відповідальності, дозволяє мінімізувати вплив на довкілля протягом всього життєвого циклу ІС. Це включає зменшення відходів, ефективне використання ресурсів та впровадження технологій, що сприяють скороченню екологічного сліду. Закон України "Про

державну політику у сфері управління відходами" (ст. 4) акцентує на необхідності інтеграції екологічних стандартів в управління відходами, що робить вибір екологічно відповідальних постачальників ще більш важливим[21].

6.2 Управління відходами та вторинне використання ресурсів

У процесі розробки та експлуатації ІС утворюються різні види відходів, включаючи електронне обладнання та матеріали. Важливо забезпечити належне управління цими відходами, дотримуючись вимог Закону України "Про відходи", який зобов'язує організації до роздільного збору, утилізації та переробки відходів (ст. 17). Необхідно впроваджувати практики вторинного використання ресурсів, зокрема повторне використання компонентів та переробку електронного обладнання. Цей підхід допомагає знизити навантаження на довкілля та сприяє сталому розвитку. Використання технологій, які мінімізують генерування відходів, а також підвищення обізнаності працівників про екологічні аспекти їхньої діяльності, сприятимуть досягненню екологічних цілей підприємства. Окрім цього, забезпечення прозорості та відповідальності у поводженні з відходами підвищить довіру суспільства до діяльності підприємства та зміцнить його репутацію[21, 22].

Важливим аспектом є впровадження політики зменшення використання шкідливих матеріалів у процесі розробки та експлуатації ІС. Відповідно до Закону України "Про охорону навколишнього природного середовища" (ст. 16), підприємства зобов'язані мінімізувати використання шкідливих речовин і матеріалів. Це включає вибір екологічно безпечних матеріалів для виробництва обладнання, а також пошук альтернативних рішень для заміни небезпечних компонентів. Впровадження таких заходів сприяє покращенню екологічних показників підприємства та зниженню ризиків для здоров'я працівників і навколишнього середовища.

Висновки до розділу

У розділі розглянуто питання охорони навколишнього середовища. Визначено заходи з мінімізації екологічного сліду під час розробки інформаційної системи. Розглянуто управління відходами та вторинне використання ресурсів.

ВИСНОВКИ

1. Виконано огляд сучасного стану і проблемних питань ІТ-забезпечення скловареної галузі. Дано загальну характеристику технологічного процесу виробництва скла і критерії його якості. Описано технологічний процес виробництва синтактичної піни та особливості виробництва синтактичного піноскла. Організовано дослідницьку лабораторію композитних матеріалів. Визначено методологію постановки досліджень процесів отримання синтактичних піностекол. Представлено методи візуалізації даних процесів отримання синтактичного піноскла. Проведено аналіз невирішених питань і сформульовано постановку задачі.

2. Представлено моделі та методи візуалізації експериментальних і розрахункових даних у технологічних процесах отримання синтактичних піностекол. Розроблено математичну модель кінетики спікання скляних мікросфер. Визначено вимоги до інформаційної системи наукового співробітника. Розроблено концепцію інформаційної системи наукового співробітника.

3. Проведено розробку і практичну реалізацію інформаційної системи у дослідницькій лабораторії. Розроблено модель інформаційної системи, включаючи діаграму варіантів використання, визначення потоків даних, діаграму станів та концептуальну модель даних. Створено технічний проект, що містить логічну модель даних, архітектуру інформаційної системи, діаграму розгортання та основні прикладні алгоритми. Виконано робочий проект, де обрано засоби розробки, створено фізичну модель даних, проведено кодування і тестування інформаційної системи.

4. Обґрунтовано економічну доцільність розробки інформаційної системи. Виконано розрахунок вартості матеріально-енергетичних витрат. Проведено розрахунок фонду оплати праці за виконану роботу. Розраховано ціну розробки програми. Виконано розрахунок економічної ефективності та строку окупності програмного забезпечення.

5. Розглянуто питання охорони праці. Описано умови праці та організацію робочого місця. Розглянуто техніку безпеки під час роботи з обладнанням та програмним забезпеченням. Розглянуто питання охорони навколишнього середовища. Визначено заходи з мінімізації екологічного сліду під час розробки інформаційної системи. Розглянуто управління відходами та вторинне використання ресурсів.

Наукова новизна роботи полягає у тому що вперше для технологічного циклу спікання скляних мікросфер запропоновано нові закономірності систематизації та візуалізації оброблених експериментальних даних, що дає змогу прогнозувати властивості отримуваних спецстекло.

Практична цінність роботи полягає в зручності інтерфейсу, швидкості фіксації експериментальних даних, наявності мобільного застосунку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Вахула Я.І., Магорівська Г.Я. Фізико-хімічні властивості скла. Методи визначення та розрахунку: навчальний посібник. – Львів: Видавництво «Левада», 2022. – 158 с
2. Ящишин Й. М., Жеплинський Т. Б., Дяківський С. І. Технологія скла у трьох частинах: Ч.ІІ. Технологія скляної маси: Навчальний підручник. — Львів: Видавництво «Бескид Біт», 2004. — 250 с.
3. Shutov, F.A. Syntactic polymer foams // *Advances in Polymer Science*. — 1986. — Т. 73—74. — С. 63—123.
4. Казимиренко Ю.О., Гайдаєнко О.В. Комп'ютерне моделювання розподілу температур спікання скляних мікросфер. – *Voprosy khimii i khimicheskoi tekhnologii*, 2024, No. 3, pp. 67-74
5. Шейко В.М., Кушнарєнко Н.М. Організація та методика науково-дослідницької діяльності: Підручник. — 4-те вид., випр. і доп. — К.: Знання, 2004. — 307 с.
6. Константінов Ю. М., Гіжа О. О. Технічна механіка рідини і газу: Підручник.- К.: Вища школа, 2002.-277с.
7. Грекул В. І. Проектування інформаційних систем: підручник та практикум для вузів // В. І. Грекул, Н. Л. Коровкіна, Г. А. Львовчкіна. - 2023. - 385 с.
8. Комп'ютерні мережі [навчальний посібник] – Львів, «Магнолія 2006», 2013. – 256 с.
9. Маслов А. В. Проектування інформаційних систем економіки: навчальний посібник / Маслов А.В. - Томськ: Вид-во Томського політехнічного університету, 2008. - 216 с.Fowler, Martin (2004).
10. UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd ed.). Addison-Wesley.
11. Халпін Т., Морган Т. (2008) Інформаційне моделювання та реляційні бази даних, 2-е видання, Сан-Франциско, Каліфорнія: Морган Кауфманн.

12. Т. Кормен; Ч. Лейзерсон; Р. Рівест; К. Стайн (2009) [1990]. 7 Швидке сортування. Вступ до алгоритмів (вид. 3rd). MIT Press і McGraw-Hill.
13. International standard ISO/IEC 23270:2006. Information technology – Programming languages – C#. Second edition. – ISO/IEC. – 2006.
14. Бази даних MySQL: Навчальний посібник. — Тернопіль: Навчальна книга – Богдан, 2010.— 160 с.
15. Entity Framework. Сучасний засіб відображення об'єктних зав'язків .NET (C#). URL: <https://learn.microsoft.com/en-us/ef/core/providers/?tabs=dotnet-core-cli>.
16. Канер Кем, Фолк Джек, Нгуен Енг Кек. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений. — Киев : ДиаСофт, 2001. — 544 с.
17. В. І. Мацибора, В. К. Збарський, Т. В. Мацибора Економіка підприємства: Навч. посібник.- К. : Каравела, 2012. – 320с.
18. Мацелюх Н. П. [та ін.] Основи економічної науки : Навч. посібник для самост. підгот. до курсов. екзамену та комплекс. держ. екзамену з економ. теорії Нац. ун-т держ. податкової служби України. - К. : Центр учб. літ., 2015. - 324 с.
19. Закон України про охорону праці [Електронний ресурс] – Режим доступу: <https://zakon.rada.gov.ua/laws/show/2694-12#Text>
20. Правила пожежної безпеки в Україні [Електронний ресурс] – Режим доступу: <https://zakon.rada.gov.ua/laws/show/z0252-15#n14>
21. Туниця Ю. Екологічна конституція Землі. Ідея. Концепція. Проблеми. Львів: Видавничий центр Львівського національного університету імені Івана Франка, 2002. 298 с.
22. Охорона природи // Юридична енциклопедія : [у 6 т.] / ред. кол.: Ю. С. Шемшученко (відп. ред.) [та ін.]. — К.
23. Downey, Allen B. (May 2012). Think Python: How to Think Like a Computer Scientist (version 1.6.6 ed.). Cambridge University Press

ДОДАТКИ

Додаток А – Програмний код

Програмний код додатку. Клас Склад, Composition.cs

```
namespace GraphBuilder.Data;
public partial class Composition
{
    public string Name { get; set; } = null!;
    public string Description { get; set; } = null!;
    public virtual ICollection<Experiment> Experiments { get; set; }
} = new List<Experiment>();
public Composition(string name, string description)
{
    Name = name;
    Description = description;
}
}
```

Клас Експеримент, Experiment.cs

```
namespace GraphBuilder.Data;
public partial class Experiment
{
    public string DateTime { get; set; } = null!;
    public string CompositionName { get; set; } = null!;
    public double Diameter { get; set; }
    public double StartHeight { get; set; }
    public double Pressure { get; set; }
    public double Mass { get; set; }
    public double TargetTemperature { get; set; }
    public double StartDensity => Mass / (Math.Pow((Diameter / 20),
2) * Math.PI * StartHeight / 10);
    public int TotalTime
    {
        get
```

```

        {
            try { return Records.Max(r => r.Minute); }
            catch { return 0; }
        }
    }

    public double FinalHeight
    {
        get
        {
            var result = StartHeight;
            foreach (var r in Records)
                result -= r.DeltaHeight;
            return result;
        }
    }

    public double FinalDensity => Mass / (Math.Pow((Diameter / 20),
2) * Math.PI * FinalHeight / 10);

    public virtual Composition CompositionNameNavigation { get; set;
} = null!;

    public virtual ICollection<Record> Records { get; set; } = new
List<Record>();

    public Experiment(string dateTime, double targetTemperature,
double diameter, double startHeight, double pressure, double mass)
    {
        DateTime = dateTime;
        TargetTemperature = targetTemperature;
        Diameter = diameter;
        StartHeight = startHeight;
        Pressure = pressure;
        Mass = mass;
    }
}

```

Контекст даних, DbContext.cs

```
using Microsoft.EntityFrameworkCore;
```

```

namespace GraphBuilder.Data;
public partial class DbDataContext : DbContext
{
    public DbDataContext()
    {
    }
    public DbDataContext(DbContextOptions<DbDataContext> options)
        : base(options)
    {
    }
    public virtual DbSet<Composition> Compositions { get; set; }
    public virtual DbSet<Experiment> Experiments { get; set; }
    public virtual DbSet<Record> Records { get; set; }
    protected override void OnConfiguring(DbContextOptionsBuilder
optionsBuilder)
        => optionsBuilder.UseSqlite("DataSource=data.db");
    protected override void OnModelCreating(ModelBuilder
modelBuilder)
    {
        modelBuilder.Entity<Composition>(entity =>
        {
            entity.HasKey(e => e.Name);
            entity.ToTable("composition");
            entity.Property(e => e.Name)
                .HasColumnType("TEXT (30)")
                .HasColumnName("name");
            entity.Property(e => e.Description)
                .HasColumnType("TEXT (255)")
                .HasColumnName("description");
        });
        modelBuilder.Entity<Experiment>(entity =>
        {
            entity.HasKey(e => e.DateTime);
            entity.ToTable("experiment");
            entity.Property(e => e.DateTime)

```

```

        .HasColumnType("TEXT (20)")
        .HasColumnName("dateTime");
entity.Property(e =>
e.CompositionName).HasColumnName("compositionName");
entity.Property(e
e.Diameter).HasColumnName("diameter");
entity.Property(e => e.Mass).HasColumnName("mass");
entity.Property(e
e.Pressure).HasColumnName("pressure");
entity.Property(e
e.StartHeight).HasColumnName("startHeight");
entity.Property(e
e.TargetTemperature).HasColumnName("targetTemperature");
entity.HasOne(d
d.CompositionNameNavigation).WithMany(p
p.Experiments).HasForeignKey(d => d.CompositionName);
});
modelBuilder.Entity<Record>(entity =>
{
    entity.HasKey(e => new { e.ExperimentDateTime, e.Minute
});
entity.ToTable("record");
entity.Property(e => e.ExperimentDateTime)
    .HasColumnType("TEXT (20)")
    .HasColumnName("experimentDateTime");
entity.Property(e => e.Minute).HasColumnName("minute");
entity.Property(e
e.DeltaHeight).HasColumnName("deltaHeight");
entity.Property(e
e.Temperature).HasColumnName("temperature");

entity.HasOne(d
d.ExperimentDateTimeNavigation).WithMany(p
p.Records).HasForeignKey(d => d.ExperimentDateTime);
});

```

```

        OnModelCreatingPartial(modelBuilder);
    }
    partial void OnModelCreatingPartial(ModelBuilder modelBuilder);
}

```

Клас Запис, Record.cs

```

namespace GraphBuilder.Data;
public partial class Record
{
    public string ExperimentDateTime { get; set; } = null!;

    public int Minute { get; set; }
    public double DeltaHeight { get; set; }
    public double InterDeltaHeight
    {
        get
        {
            var result = DeltaHeight;
            foreach (var r in ExperimentDateTimeNavigation.Records)
                if (r.Minute < Minute)
                    result += r.DeltaHeight;
            return result;
        }
    }
    public double Temperature { get; set; }
    public virtual Experiment ExperimentDateTimeNavigation { get;
set; } = null!;
    public Record(int minute, double deltaHeight, double
temperature)
    {
        Minute = minute;
        DeltaHeight = deltaHeight;
        Temperature = temperature;
    }
}

```

Користувачський елемент керування, CUControl.xaml.cs

```
using System.Windows;
using System.Windows.Controls;
namespace GraphBuilder
{
    public partial class CUControl : UserControl
    {
        private static readonly RoutedEventHandler emptyDelegate =
delegate { };

        public event RoutedEventHandler Create = emptyDelegate;
        public event RoutedEventHandler Update = emptyDelegate;
        public event RoutedEventHandler Delete = emptyDelegate;
        public bool IsCreatePossible { get =>
createButton.IsEnabled; set => createButton.IsEnabled = value; }
        public bool IsUpdatePossible { get =>
updateButton.IsEnabled; set => updateButton.IsEnabled = value; }
        public bool IsDeletePossible { get =>
deleteButton.IsEnabled; set => deleteButton.IsEnabled = value; }
        public CUControl()
        {
            InitializeComponent();
            createButton.Click += (sender, args) => Create(sender,
args);
            updateButton.Click += (sender, args) => Update(sender,
args);
            deleteButton.Click += (sender, args) => Delete(sender,
args);
        }
    }
}
```

Основне вікно програми, MainWindow.cs

```
using GraphBuilder.Data;
using GraphBuilder.Windows;
using System.Diagnostics;
using System.IO;
```

```

using System.Text.Json;
using System.Windows;
using System.Windows.Controls;
namespace GraphBuilder
{
    public partial class MainWindow : Window
    {
        private const double kgCm2ToBa = 980.665 * 1000;
        private static readonly DbDataContext db = new();
        private Composition? SelectedComposition =>
compositionsComboBox.SelectedItem is null
        ? null
        :
db.Compositions.Find(compositionsComboBox.SelectedItem.ToString())
;
        private Experiment[] SelectedExperiments
        {
            get
            {
                var result = new List<Experiment>();
                var expDates =
experimentsListBox.SelectedItems.Cast<string>();
                foreach (var exp in expDates)
                {
                    result.Add(db.Experiments.Find(exp)!);
                }
                return result.ToArray();
            }
        }
        private Record? SelectedRecord =>
recordsDataGrid.SelectedItem is null
        ? null
        : recordsDataGrid.SelectedItem as Record;
        public MainWindow()
        {

```



```

        InitializeComponent();
        ResizeMode = ResizeMode.NoResize;
        using var context = new DbDataContext();
        UpdateCompositionsComboBox();
    }

    #region Compositions
    private void CompositionsComboBoxSelectionChanged(object
sender, SelectionChangedEventArgs e) =>
CompositionSelectionChanged();
    private void CompositionCUDCreate(object sender,
RoutedEventArgs e)
    {
        var window = new CUCompositionWindow(db);
        window.ShowDialog();
        UpdateCompositionsComboBox();
    }

    private void CompositionCUDUpdate(object sender,
RoutedEventArgs e)
    {
        var window = new CUCompositionWindow(db,
SelectedComposition!);
        window.ShowDialog();
        UpdateCompositionsComboBox();
    }
    private void CompositionCUDDelete(object sender,
RoutedEventArgs e)
    {
        db.Compositions.Remove(SelectedComposition!);
        db.SaveChanges();
        UpdateCompositionsComboBox();
    }
    private void CompositionSelectionChanged()
    {

```

```

        var comp = SelectedComposition;
        if (comp != null)
        {
            db.Entry(comp).Collection(c =>
c.Experiments).Load();

            experimentCUD.IsCreatePossible =
compositionCUD.IsUpdatePossible = compositionCUD.IsDeletePossible =
true;

            compositionDescriptionTextBlock.Text =
comp.Description;
        }
        else
        {
            experimentCUD.IsCreatePossible =
compositionCUD.IsUpdatePossible = compositionCUD.IsDeletePossible =
false;

            compositionDescriptionTextBlock.Text = "No
Composition selected.";
        }
        UpdateExperimentsListBox();
    }
    private void UpdateCompositionsComboBox()
    {
        var selectedItem = compositionsComboBox.SelectedItem;
        compositionsComboBox.Items.Clear();
        foreach (var c in db.Compositions)
            compositionsComboBox.Items.Add(c.Name);
        if (compositionsComboBox.Items.Contains(selectedItem))
            compositionsComboBox.SelectedItem = selectedItem;
        else compositionsComboBox.SelectedIndex = 0;
        CompositionSelectionChanged();
    }
}
#endregion

#region Experiments

```

```

        private void ExperimentsListBoxSelectionChanged(object
sender, SelectionChangedEventArgs e) =>
ExperimentSelectionChanged();

        private void ExperimentCUDCreate(object sender,
RoutedEventArgs e)
        {
            var window = new CUExperimentWindow(db,
SelectedComposition!);
            window.ShowDialog();
            UpdateExperimentsListBox();
        }

        private void ExperimentCUDUpdate(object sender,
RoutedEventArgs e)
        {
            var window = new CUExperimentWindow(db,
SelectedExperiments[0]);
            window.ShowDialog();
            UpdateExperimentsListBox();
        }

        private void ExperimentCUDDelete(object sender,
RoutedEventArgs e)
        {
            db.Experiments.Remove(SelectedExperiments[0]);
            db.SaveChanges();
            UpdateExperimentsListBox();
        }

        private void ExperimentSelectionChanged()
        {
            var exps = SelectedExperiments;
            if (exps.Length == 1)
            {
                db.Entry(exps[0]).Collection(c
=>
c.Records).Load();
            }
        }
    }
}

```

```

        recordCUD.IsCreatePossible =
experimentCUD.IsUpdatePossible = experimentCUD.IsDeletePossible =
true;

        experimentPrecoditionsTextBlock.Text = $"Target t°:
{exps[0].TargetTemperature:f4} C°\n" +
        $"Diameter: {exps[0].Diameter:f4} mm\n" +
        $"Start height: {exps[0].StartHeight:f4} mm\n"
+
        $"Pressure: {exps[0].Pressure:f4} kg/cm2\n" +
        $"Mass: {exps[0].Mass:f4} g\n" +
        $"Start ρ: {exps[0].StartDensity:f4} g/cm3";
        experimentResultsTextBlock.Text = $"Total time:
{exps[0].TotalTime} min" +
        $"\nFinal height: {exps[0].FinalHeight:f4}
mm\n" +
        $"Final ρ: {exps[0].FinalDensity:f4} g/cm3";
    }
    else
    {
        recordCUD.IsCreatePossible =
experimentCUD.IsUpdatePossible = experimentCUD.IsDeletePossible =
true;

        experimentPrecoditionsTextBlock.Text = "No/Many
Experiments Selected.";
        experimentResultsTextBlock.Text = "No/Many
Experiments Selected.";
    }
    plotButton.IsEnabled = exps.Length > 0;
    UpdateRecordsDataGrid();
}
private void UpdateExperimentsListBox()
{
    var comp = SelectedComposition;
    var selectedItem = experimentsListBox.SelectedItem;
    experimentsListBox.Items.Clear();

```

```

        if (comp != null)
        {
            foreach (var e in comp.Experiments)
                experimentsListBox.Items.Add(e.DateTime);
        }
        if (experimentsListBox.Items.Contains(selectedItem))
            experimentsListBox.SelectedItem = selectedItem;
        else experimentsListBox.SelectedIndex = 0;
        ExperimentSelectionChanged();
    }
#endregion

#region Records
private void RecordsDataGridSelectionChanged(object sender,
SelectionChangedEventArgs e) => RecordSelectionChanged();
private void RecordCUDCreate(object sender, RoutedEventArgs
e)
{
    var window = new CURecordWindow(db,
SelectedExperiments[0]);
    window.ShowDialog();
    UpdateRecordsDataGrid();
}
private void RecordCUDUpdate(object sender, RoutedEventArgs
e)
{
    var window = new CURecordWindow(db, SelectedRecord!);
    window.ShowDialog();
    UpdateRecordsDataGrid();
}
private void RecordCUDDelete(object sender, RoutedEventArgs
e)
{
    db.Records.Remove(SelectedRecord!);
    db.SaveChanges();
}

```

```

        UpdateRecordsDataGrid();
    }
    private void RecordSelectionChanged()
    {
        var rec = SelectedRecord;
        recordCUD.IsUpdatePossible = recordCUD.IsDeletePossible
= rec != null;
    }
    private void UpdateRecordsDataGrid()
    {
        var exps = SelectedExperiments;
        var selectedItem = recordsDataGrid.SelectedItem;
        recordsDataGrid.Items.Clear();
        if (exps.Length == 1)
        {
            foreach (var r in exps[0].Records)
                recordsDataGrid.Items.Add(r);
        }
        if (recordsDataGrid.Items.Contains(selectedItem))
            recordsDataGrid.SelectedItem = selectedItem;
        else recordsDataGrid.SelectedIndex = 0;
        RecordSelectionChanged();
    }
#endregion

    private void PlotButtonClick(object sender, RoutedEventArgs
e)
    {
        var options = new JsonSerializerOptions() {
IncludeFields = true };
        var exps = SelectedExperiments;
        #region HeightPlot
        var heightSeries = new List<Series>();
        foreach (var exp in exps)
        {

```

```

        db.Entry(exp).Collection(c => c.Records).Load();
        var xVals = new List<double>();
        var yVals = new List<double>();
        foreach (var r in exp.Records)
        {
            xVals.Add(r.Minute);
            yVals.Add(r.InterDeltaHeight /
exp.StartHeight);
        }

        var newSeries = new
Series($"{exp.TargetTemperature}C° {exp.Pressure}kg/cm2",
xVals.ToArray(), yVals.ToArray(), "linear");
        heightSeries.Add(newSeries);
    }

    File.WriteAllText(@"py\heightPlotData.json",
JsonSerializer.Serialize(heightSeries.ToArray(), options));

    Process.Start("python", @"py\plot.py
py\heightPlotData.json " + "\"Shrinking Rate Over Time\" \"Time τ
(min)\" \"Δh/h0\"");
    #endregion
    #region ViscosityPlot
    var viscositySeries = new List<Series>();
    foreach (var exp in exps)
    {
        var xVals = new List<double>();
        var yVals = new List<double>();
        foreach (var r in exp.Records)
        {
            xVals.Add(r.Minute);
            yVals.Add(Math.PI * exp.Pressure * kgCm2ToBa *
exp.StartDensity * Math.Pow(exp.Diameter / 20, 2) * (exp.FinalHeight
/ 10 - r.DeltaHeight / 10) * exp.TotalTime / (4 * Math.Sqrt(2) *
exp.Mass));
        }
    }

```

```

        var newSeries = new
Series($"{exp.TargetTemperature}C° {exp.Pressure}kg/cm2",
xVals.ToArray(), yVals.ToArray(), "log");
        viscositySeries.Add(newSeries);
    }
    File.WriteAllText(@"py\viscosityPlotData.json",
JsonSerializer.Serialize(viscositySeries.ToArray(), options));
    Process.Start("python", @"py\plot.py
py\viscosityPlotData.json " + "\"Viscosity Change Over Time\" \"Time
τ (min)\\" \|log η (Ba/s)\");
    #endregion
}

private record Series(string Title, double[] XVals, double[]
YVals, string Scale);
}
}

```

Вікно роботи зі Складами, CUCompositionWindow.xaml.cs

```

using GraphBuilder.Data;
using System.Windows;
namespace GraphBuilder.Windows
{
    public partial class CUCompositionWindow : Window
    {
        private readonly Composition? composition = null;
        private readonly DbDataContext db;
        public CUCompositionWindow(DbDataContext db, Composition
composition) : this(db)
        {
            this.composition = composition;
            nameTextBox.Text = composition.Name;
            descriptionTextBox.Text = composition.Description;
            nameTextBox.IsEnabled = false;
            saveButton.IsEnabled = true;
            titleLabel.Content = Title = "Update Composition";
        }
    }
}

```



```

public CUCompositionWindow(DbDataContext db)
{
    InitializeComponent();
    this.db = db;
}

private void NameTextBoxTextChanged(object sender,
System.Windows.Controls.TextChangedEventArgs e)
{
    saveButton.IsEnabled = nameTextBox.Text.Length >= 3 &&
db.Compositions.Find(nameTextBox.Text) == null;
}

private void SaveButtonClick(object sender, RoutedEventArgs
e)
{
    if (composition is null)
        db.Compositions.Add(new
Composition(nameTextBox.Text, descriptionTextBox.Text));
    else
    {
        composition.Name = nameTextBox.Text;
        composition.Description = descriptionTextBox.Text;
    }
    db.SaveChanges();
    Close();
}
}
}

```

Вікно роботи із Експериментами, CUExperimentWindow.xaml.cs

```

using GraphBuilder.Data;
using System.Globalization;
using System.Windows;
namespace GraphBuilder.Windows
{
    public partial class CUExperimentWindow : Window

```

```

{
    private readonly Composition? composition;
    private readonly Experiment? experiment = null;
    private readonly DbDataContext db;
    public CUExperimentWindow(DbDataContext db, Experiment
experiment) : this(db, composition: null!)
    {
        this.experiment = experiment;
        datePicker.Value =
DateTime.ParseExact(experiment.DateTime, Globals.TimeFormat,
CultureInfo.InvariantCulture);
        temperatureUpDown.Value = experiment.TargetTemperature;
        diameterUpDown.Value = experiment.Diameter;
        heightUpDown.Value = experiment.StartHeight;
        pressureUpDown.Value = experiment.Pressure;
        massUpDown.Value = experiment.Mass;
        datePicker.IsEnabled = false;
        saveButton.IsEnabled = true;
        titleLabel.Content = Title = "Update Experiment";
    }
    public CUExperimentWindow(DbDataContext db, Composition
composition)
    {
        InitializeComponent();
        this.db = db;
        this.composition = composition;
    }
    private void SaveButtonClick(object sender, RoutedEventArgs
e)
    {
        if (experiment is null)
        {
            var newExperiment = new
Experiment(datePicker.Value!.Value.ToString(Globals.TimeFormat),

```

```

temperatureUpDown.Value!.Value,          diameterUpDown.Value!.Value,
heightUpDown.Value!.Value,

pressureUpDown.Value!.Value,

massUpDown.Value!.Value);
        composition!.Experiments.Add(newExperiment);
    }
    else
    {
        experiment.TargetTemperature =
temperatureUpDown.Value!.Value;
        experiment.Diameter = diameterUpDown.Value!.Value;
        experiment.StartHeight = heightUpDown.Value!.Value;
        experiment.Pressure = pressureUpDown.Value!.Value;
        experiment.Mass = massUpDown.Value!.Value;
    }
    db.SaveChanges();
    Close();
}

private void ValueChanged(object sender,
RoutedEventArgs<object> e)
{
    saveButton.IsEnabled = diameterUpDown.Value.HasValue
        && heightUpDown.Value.HasValue
        && pressureUpDown.Value.HasValue
        && massUpDown.Value.HasValue
        && datePicker.Value.HasValue
        && (experiment is not null ||
db.Experiments.Find(datePicker.Value.Value.ToString(Globals.TimeFo
rmat)) == null);
}
}
}

```

```

Вікно роботи із Записами, CUExperimentWindow.xaml.cs
using GraphBuilder.Data;
using System.Windows;

namespace GraphBuilder.Windows
{
    public partial class CURecordWindow : Window
    {
        private readonly Experiment? experiment;
        private readonly Record? record;
        private readonly DbDataContext db;

        public CURecordWindow(DbDataContext db, Record record) :
this(db, experiment: null!)
        {
            this.record = record;
            minuteUpDown.Value = record.Minute;
            heightUpDown.Value = record.DeltaHeight;
            temperatureUpDown.Value = record.Temperature;
            minuteUpDown.IsEnabled = false;
            saveButton.IsEnabled = true;
            titleLabel.Content = Title = "Update Record";
        }

        public CURecordWindow(DbDataContext db, Experiment
experiment)
        {
            InitializeComponent();
            this.db = db;
            this.experiment = experiment;
            if (experiment != null)
            {
                temperatureUpDown.Value =
experiment.TargetTemperature;
                minuteUpDown.Value = experiment.TotalTime + 1;
            }
        }
    }
}

```

```

    }
    private void SaveButtonClick(object sender, RoutedEventArgs
e)
    {
        if (record is null)
        {
            var newRecord = new
Record(minuteUpDown.Value!.Value, heightUpDown.Value!.Value,
temperatureUpDown.Value!.Value);
            experiment!.Records.Add(newRecord);
        }
        else
        {
            record.DeltaHeight = heightUpDown.Value!.Value;
            record.Temperature =
temperatureUpDown.Value!.Value;
        }
        db.SaveChanges();
        Close();
    }

    private void ValueChanged(object sender,
RoutedPropertyChangedEventArgs<object> e)
    {
        saveButton.IsEnabled = minuteUpDown.Value.HasValue
            && heightUpDown.Value.HasValue
            && temperatureUpDown.Value.HasValue
            && (record is not null ||
db.Records.Find(experiment!.DateTime, minuteUpDown.Value.Value) ==
null);
    }
}
}

```

Модуль відображення графіків, plot.py

```

from cProfile import label

```

```

from operator import imod
from matplotlib import markers
import matplotlib.pyplot as plt
from scipy.interpolate import interp1d
import numpy as np
import json
import sys
DATA_FILE = sys.argv[1]
XVALS_NAME = "XVals"
YVALS_NAME= "YVals"
TITLE_NAME= "Title"
SCALE_NAME= "Scale"
with open(DATA_FILE) as f:
    data = json.load(f)
plt.xlabel(sys.argv[3]);
plt.ylabel(sys.argv[4]);
for v in data:
    x = np.array(v[XVALS_NAME])
    y = np.array(v[YVALS_NAME])
    interpolated_model = interp1d(x, y, kind = "cubic")
    X_ = np.linspace(x.min(), x.max(), 500)
    Y_ = interpolated_model(X_)
    t = plt.plot(X_, Y_, label=v[TITLE_NAME], marker
="o",markevery=10)
    plt.yscale(v[SCALE_NAME])
    plt.legend(loc="lower right")
plt.title(sys.argv[2])
plt.grid()
plt.show()

```

DDL Бази Даних

```

CREATE TABLE composition (
    name          TEXT (30)  PRIMARY KEY,
    description TEXT (255) NOT NULL
);
CREATE TABLE experiment (

```

```

        dateTime          TEXT (20) PRIMARY KEY,
        compositionName    TEXT          REFERENCES composition (name) ON
DELETE CASCADE
                                                    ON
UPDATE CASCADE

                                                    NOT NULL,
        diameter          REAL          NOT NULL,
        startHeight       REAL          NOT NULL,
        pressure          REAL          NOT NULL,
        mass              REAL          NOT NULL,
        targetTemperature REAL          NOT NULL
CREATE TABLE record (
experimentDateTime TEXT (20) REFERENCES experiment (dateTime) ON
DELETE CASCADE
ON UPDATE CASCADE
NOT NULL,
minute INTEGER NOT NULL,
deltaHeight REAL NOT NULL,
temperature REAL NOT NULL,
PRIMARY KEY (
experimentDateTime,
minute
)
);

```

Додаток В – Презентаційні матеріали