

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

с

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ ВІРТУАЛЬНОГО СЕРЕДОВИЩА ДЛЯ
ВІЗУАЛІЗАЦІЇ ЕКСПЕРИМЕНТІВ З ДОСЛІДЖЕННЯ МАТЕРІАЛІВ
МЕДИЧНОГО ПРИЗНАЧЕННЯ. SOFTWARE IMPLEMENTATION OF A
VIRTUAL ENVIRONMENT FOR VISUALIZING THE EXPERIMENTS FOR
THE MEDICAL MATERIALS RESEARCH

Виконав: студент(ка) 2 курсу, групи КНТ-110м
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)
Інженерія програмного забезпечення

Васькін Б.В.

(прізвище та ініціали)

Керівник Табунщик Г.В.

(прізвище та ініціали)

Рецензент Шитікова О.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти магістр
 Спеціальність 121 Інженерія програмного забезпечення
 (код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТА(КИ)

Васькіна Богдана Вікторівна

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Програмна реалізація віртуального середовища для візуалізації експериментів з дослідження матеріалів медичного призначення.
Software Implementation of a Virtual Environment for Visualizing the Experiments for the Medical Materials Research.

керівник проекту (роботи) Табунщик Галина Володимирівна, професор,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “5” листопада 2021 року № 435

2. Строк подання студентом проекту (роботи) 2 грудня 2021 року

3. Вихідні дані до проекту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми. 3. Основні рішення щодо реалізації компонентів системи. 4. Експлуатація, тестування та експериментальне дослідження програми. 5. Технічне завдання. 6. Текст програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Табунщик Г.В., професор		
6 Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання 1 жовтня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Постановка завдання роботи.	1-2 тижні	Завдання, ТЗ
2	Аналіз предметної області.	3-5 тижні	Розділ 1
3	Розробка архітектури програми.	6-7 тижні	Розділ 2
4	Розробка програми.	8-11 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми.	12-13 тижні	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	14-15 тижні	Додатки
7	Захист роботи.	16 тиждень	

Студент (ка) _____
(підпис) (прізвище та ініціали)

Керівник проекту (роботи) _____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 86 с., 14 табл., 36 рис., 2 дод., 46 джерел.

МАТЕРІАЛИ МЕДИЧНОГО ПРИЗНАЧЕННЯ, ВІРТУАЛЬНЕ СЕРЕДОВИЩЕ, ВІЗУАЛІЗАЦІЇ ЕКСПЕРИМЕНТІВ, WEB, CANVAS.

Об'єкт дослідження – процес дослідження матеріалів медичного призначення.

Предмет дослідження – віртуальне середовище для дослідження матеріалів медичного призначення.

Мета роботи – покращення процесів цифрової трансформації в дослідницькому середовищі, за рахунок візуалізації процесів корозії магнієвих сплавів за допомогою віртуального середовища.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, методи та засоби оцінювання корозії магнієвих сплавів, Canvas.

Результати. Створено програмний Web-застосунок, який моделює перебіг експерименту за заданими параметрами.

Висновки. В роботі проведено дослідження методів та засобів візуалізації результатів досліджень, розроблено віртуальне середовище для дослідження матеріалів медичного призначення.

Галузь використання – дослідницькі центри, лабораторії фізичного матеріалознавства та біоінженерії.

ABSTRACT

Explanatory note to the final qualifying work of the bachelor: 86 pages, 36 figures, 14 tables, 2 appendixes, 46 sources.

MATERIALS FOR MEDICAL PURPOSE, VIRTUAL ENVIRONMENT, VISUALIZATION OF EXPERIMENTS, WEB, CANVAS.

The object of research is the process of research of medical materials.

The subject of research - a virtual environment for the study of medical materials.

The aim of the work is to improve the processes of digital transformation in the research environment by visualizing the corrosion processes of magnesium alloys with the help of a virtual seat.

Materials, methods and technical means: structural and object-oriented programming, methods and means of corrosion assessment of magnesium alloys, Canvas.

Results. A software Web-application has been created, which simulates the course of the experiment according to the set parameters.

Conclusions. The research of methods and means of visualization of research results is carried out in the work, the virtual environment for research of materials of medical appointment is developed.

Field of application - research centers, laboratories of physical materials science and bioengineering.

ЗМІСТ

	С.
Перелік скорочень та умовних познач	7
Вступ	8
1 Аналіз предметної області	9
1.1 Огляд середовищ віртуалізації	9
1.2 Технології обробки медичних даних	15
1.3 Процеси корозії металів медичних сплавів	19
2 Матеріали та методи	25
2.1 Web технології віртуалізації	25
2.2 Огляд мов програмування	28
2.3 Огляд обраного середовища розробки	30
2.4 Методи проведення випробувань матеріалів медичного призначення	34
3 Основні рішення щодо реалізації компонентів системи	36
3.1 Проектування інтерфейсу	36
3.2 Опис модулів розроблюваної системи	43
3.3 Проектування бази даних	43
4 Експлуатація, тестування та експериментальне дослідження програми	46
4.1 Призначення й умови застосування програми	46
4.2 Характеристики програми	46
4.3 Звертання до програми	46
4.4 Початкові та вихідні дані	47
4.5 Повідомлення програмісту	47
4.6 Опис методики проведення тестування	47
4.7 Результати використання розробленого програмного забезпечення	50
4.7 Тестування розробленого програмного забезпечення	52
Висновки	55
Перелік джерел посилань	56
Додаток А Текст програми	61
Додаток Б Слайди презентації	80

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
IDE	– Integrated Development Environment,
PGA	– Programmable-Gain Amplifier;
SAR	– Successive Approximation Register;
SoC	– System-on-Chip;
TRM	– Technical Reference Model;
UDB	– Universal Digital Block;
ПЗ	– програмне забезпечення.

ВСТУП

Проблема наукового аналізу медичних даних з кожним днем стає все більш актуальною, оскільки питання, що виникають у лікарів при прийнятті правильного рішення, стають все складнішими. В основі обробки й аналізу даних лежать математичні методи, що здебільшого є незмінними вже протягом багатьох десятиліть. Незмінними залишаються й загальні принципи та послідовність дій при обробці даних, проте технологія обробки даних міняється, і досить істотно.

В останні роки обробка й аналіз будь-якої інформації стають не просто неможливими, а й недопустимими без використання комп'ютерів і відповідного сучасного програмного забезпечення. Комп'ютерний аналіз медичних даних припускає деяке математичне перетворення даних за допомогою певних програмних засобів, а отже користувачу необхідно мати уявлення не лише про математичні методи обробки даних, а й про відповідні програмні засоби [1].

Актуальність розробки програмного застосунку пов'язана з запровадженням інформаційно-комунікаційних технологій та розвитком комп'ютерного моделювання. Комп'ютерні моделі передбачають створення програмного середовища, яке поєднує в собі на основі математичної моделі явища або процесу засоби інтерактивної взаємодії з об'єктом дослідження і розвинуті засоби відображення інформації [2].

Метою роботи є покращення процесів цифрової трансформації в дослідницькому середовищі, за рахунок візуалізації процесів корозії магнієвих сплавів за допомогою віртуального середовища.

Для досягнення поставленої мети у роботі вирішувалися такі завдання:

- проаналізувати середовища віртуалізації;
- проаналізувати медичну інформаційну інфраструктуру;
- розробити віртуальне середовище для візуалізації результатів випробувань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд середовищ віртуалізації

Віртуальна лабораторія (ВЛ) – це віртуальна навчальна середа, яка дозволяє моделювати поведінку об'єктів реального світу в комп'ютерному середовищі і допомагає в оволодінні новими знаннями та вміннями. Така лабораторія може виступати апаратом досліджень різних природних явищ з можливістю побудови їх математичних моделей [3]. Використання ВЛ дає змогу не лише спостерігати за певним експериментом, а й безпосередньо брати в ньому участь, а це в свою чергу сприяє засвоєнню знань на більш свідомому та глибокому рівні.

Наразі розроблено велику кількість віртуальних лабораторій, які візуалізують та спрощують сприйняття научного матеріалу, допомагають більш наочно та безпечніше провести різноманітні досліди та експерименти [4].

За визначенням В.В. Трухіна [5], ВЛ являє собою програмно-апаратний комплекс, що дозволяє проводити досліди без безпосереднього контакту з реальною установкою або при повній її відсутності. За Е.О. Козловським і Г.М. Кравцовим, ВЛ – це віртуальне програмне середовище, в якому організована можливість поводження досліджень моделей об'єктів, їх сукупностей і похідних, заданих з певною часткою деталізації щодо реальних об'єктів, в рамках певної галузі знань [6].

Слід зазначити, що переваги ВЛ над реальними проявляється у наступному: відсутність необхідності придбання дорогого устаткування і матеріалів; можливість моделювання процесів, протікання яких принципово неможливо в лабораторних умовах; можливість проникнення в тонкощі процесів і спостереження відбувається в іншому масштабі часу; безпека при роботі з небезпечними речовинами чи приладами; можливість швидкого проведення серії дослідів з різними значеннями вхідних параметрів; економія часу і ресурсів; можливість використання віртуальної лабораторії в дистанційному навчанні [7].

При проведенні роботи необхідно пам'ятати, що віртуальна модель відображає реальні процеси і явища в більш-менш спрощеному, схематичному вигляді, тому з'ясування питання, що насправді підкреслено в процесі, а що залишилося за кадром, може бути однією з форм завдання. Роботи такого виду можна виконувати цілком в комп'ютерному варіанті або зробити одним з етапів в ширшій роботі, яка включає також роботу з натуральними об'єктами та лабораторним обладнанням [8].

Віртуальні симулятори – це реально виконувані лабораторні роботи, під час яких визначені дані можуть бути занесені до пам'яті персонального комп'ютера та дистанційно опрацьовані на віртуально представленій комп'ютером засобі. Це так званий тренажер – інструмент, що імітує експерименти, демонстрації чи процеси. Однією з головних особливостей віртуальних симуляторів є їх інтерактивні можливості. Інтерактивне моделювання набуває все більшого значення як засіб для вивчення і розуміння складних ідей. Інтерактивні тренажери можуть бути реалізовані при поєднанні комп'ютерів, графіки з високою чіткістю та програмних засобів.

При виконанні віртуальної лабораторної роботи у дослідника створюється ілюзія роботи на реальному обладнанні. Причому «експериментальні» значення вимірюваних величин визначаються самостійно на реальних установках певної роботи. Ці дані заносяться та відображаються у вигляді графіків або таблиць. [9].

Традиційно навчальні лабораторії оснащуються певним набором технічних засобів, що дозволяють проводити найпростіші дослідження. Тому для більшої наочності процесу використовуються наступні засоби: відеофільми, комп'ютерні моделі процесів, експериментальні моделі технологічних апаратів, математичні моделі процесів, дослідження на реальному технологічному об'єкті. Відзняті на виробництві відеофільми демонструють роботу устаткування, технологічних ліній. Для демонстрації процесів, важких для сприйняття й теоретичного аналізу, використовується моделювання фундаментальних завдань за допомогою комп'ютерних моделей, що потребує значних вкладень в

закордонні ліцензійні програми. Під час розробки віртуальних лабораторій віддаленого доступу на базі імітаційних моделей і реального устаткування виникає проблема дистанційного керування лабораторним стендом і відображення пульта управління на екрані.

Використання віртуальних лабораторій дозволяє забезпечити:

- індивідуальне навчання без перерв;
- можливість модульного поділу лабораторної роботи;
- можливість паралельного використання на лекційних, практичних і лабораторних заняттях;
- поетапне вивчення технологічних процесів;
- можливість аналізу експериментальних даних одночасно з експериментом;
- можливість модифікації і вдосконалення, внесення коректив до існуючої моделі.

Використання віртуальних лабораторій дає змогу, з одного боку, отримати практичні навички проведення експериментів, ознайомитися детально з комп'ютерною моделлю сучасного обладнання, досліджувати пожежо- і вибухонебезпечні процеси і явища, не побоюючись за можливі наслідки. З іншого боку, є можливість організувати взаємодію віртуального лабораторного комплексу із реально діючим обладнанням підприємств для збору експериментальних даних, що забезпечить відповідний рівень розвитку наукових розробок і технологічного керування процесом [10].

Порівняння лабораторії з віддаленим доступом та традиційної локальної лабораторій представлено у таблиці 1.1, де показано їх основні переваги та недоліки.

Таблиця 1.1 – Порівняльна характеристика лабораторій

	Лабораторії з віддаленим доступом	Традиційна локальна лабораторія
Переваги	– виконання дослідів в онлайн режимі на коштовному, часто унікальному устаткуванні; – можливість використання у будь-якій формі навчального процесу; – немає потреби в установці спеціального програмного забезпечення; – об'єднання дослідників із різних організацій, університетів, обговорення результатів роботи.	– висока наочність процесів; – засвоєння техніки безпеки і вміння працювати із обладнанням; – сприймаються більшістю видів почуттів.
Недоліки	– відсутність біля технологічного об'єкту під час експериментів; – узгодження роботи із технологічних обладнанням з іншими обслуговуючим персоналом.	– значні фінансові витрати на створення стенду; – забезпечення функціонування стенду спеціальним персоналом; – проведення занять за узгодженим графіком; – заняття колективні, тому обмежений доступ студентів до устаткування; – неможливість дослідження екстремальних і аварійних режимів.

Найефективнішим є використання комп'ютерної моделі для демонстрації під час пояснення нового матеріалу і розв'язування задач. Комп'ютерна модель є основою навчального віртуального експерименту. Важливою функцією

віртуального середовища є його інтерактивність, що забезпечує динамічність і керуваність користувачем зображення досліджуваної реальності.

Для максимальної реалізації функцій віртуальних моделей потрібно забезпечити використання їх різноманітних видів. Тому питання класифікації віртуальних моделей є важливим для повного й об'єктивного оцінювання різноманітності функцій віртуального середовища, дає можливість уточнити перспективні напрями подальшого їх розвитку, а також вказати на найефективніші способи їх використання [11].

Віртуальний експеримент має задовольняти такі основні вимоги:

- очевидна схожість з реальними фізичними приладами і їх реальною поведінкою в часі і просторі,
- візуально не повинен суттєво відрізнятися від реального;
- хід роботи й обробка результатів не мають відрізнятися від відповідних для реального експерименту;
- як і в проведенні експерименту на реальній установці, у віртуальній роботі учасники мають стикатися з перехідними процесами, необхідністю тимчасової витримки перед зняттям показів;
- у моделях має бути врахована випадкова помилка, що вносить похибку в результат, завдяки чому результати можуть бути відмінні один від одного, як і в проведенні роботи на реальних установках [12].

Віртуальні демонстрації – навчальні віртуальні моделі, для яких характерні ознаки реальних об'єктів і процесів призначені для засвоєння елементів «готового знання». Поширені два варіанти віртуальної демонстрації фізичного досліду: віртуальна демонстрація явища (подається модель явища в природних умовах його протікання); та віртуальна демонстрація фізичного експерименту (подається модель роботи експериментальної установки і спостережуваного на ній ефекту). Також можливі віртуальні демонстрації технічних об'єктів.

Віртуальна фізична лабораторія – програмний засіб, призначений для імітації роботи в фізичній лабораторії під час дослідження фізичних процесів

або явищ. На сьогоднішній день представлено цілий ряд програмних пакетів, призначених для здійснення віртуальних навчальних експериментів. Наприклад можна розглянути такі рішення, як віртуальні онлайн-лабораторії, що дозволяють проводити комп'ютерні досліди, не купуючи додаткових програм.

Наприклад, PhET (Physics Education Technology) – програмний засіб під ліцензією GNU/GPL, що вільно поширюється. Цей пакет створено для інтерактивного моделювання фізичних явищ та демонстрації їх у процесі навчання. На цьому ресурсі, розробленому Університетом Колорадо, розміщені віртуальні лабораторії, які демонструють різні явища в галузі фізики, хімії, біології, геології, а також інтерактивні математичні інструменти, як зображено на рисунку 1.1.

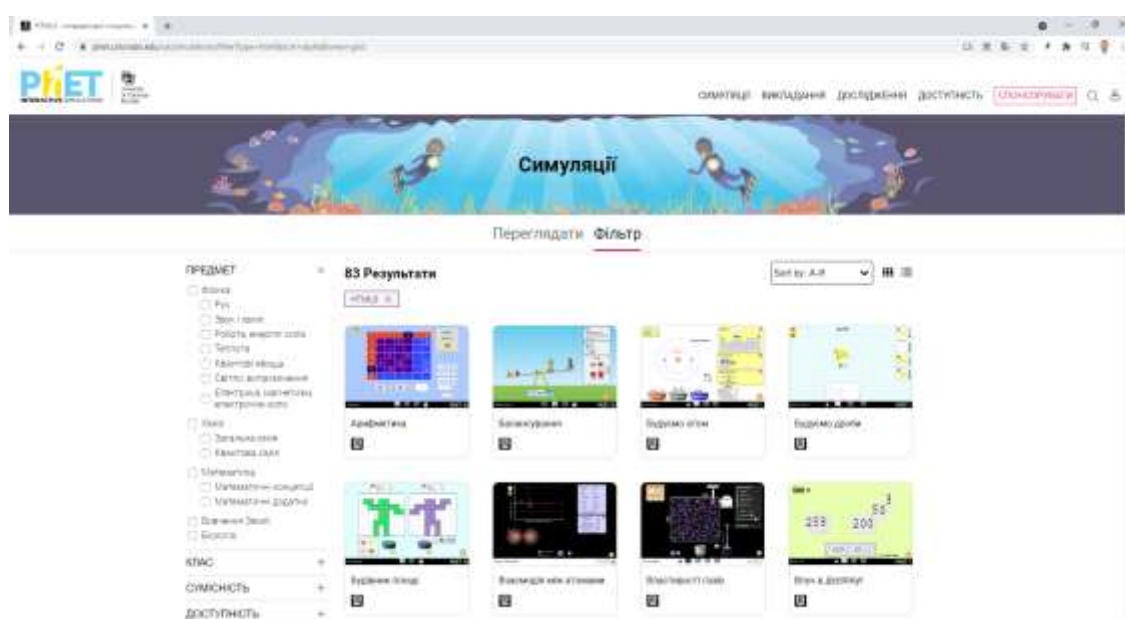


Рисунок 1.1 – головна сторінка проекту PhET [13]

Демонстрації PhET розроблені з використанням технології Java та HTML5, що дозволяє запускати онлайн експерименти, завантажувати аплети на локальний комп'ютер, та впроваджувати їх на веб-сторінки свого дистанційного курсу як віджети. Усі ці опції передбачені на сторінці кожної

демонстрації PhET. Також є можливість завантажити демонстрацію й адаптувати її до власних потреб, наприклад, перекласти українською мовою.

Усі експерименти PhET є інтерактивними. Інтерфейс віртуальних лабораторних робіт є інтуїтивно зрозумілим та не вимагає спеціальних знань і навичок у користувачів. Усі дії з віртуальними об'єктами нагадують прийоми використання реальних об'єктів. Віртуальні лабораторії містять одне або декілька завдань, а також набір усіх елементів, що необхідні для їх розв'язування. Головною метою демонстрацій є візуалізація і пояснення явищ, а не тестування знань і навичок користувача.

Окрім ресурсу PhET доцільно звернути увагу на проект VirtuLab. Це найбільший у сучасному російському сегменті інтернету збірник віртуальних дослідів з різних навчальних дисциплін. Основна одиниця колекції – віртуальний експеримент. Кожен ролик дозволяє проводити певний експеримент, що має початкову мету та чітке завдання. Користувачеві пропонуються всі інструменти й об'єкти, що необхідні для отримання результату[14].

1.2 Технології обробки медичних даних

Інформатизація є частиною процесу технологічного оновлення медицини, що стимулює процеси її стандартизації. Цей процес відбувається через необхідність використовувати однакову систему кодування та класифікації, упорядковувати термінологію, а також регламентувати основні процеси в медичних інформаційних системах (МІС).

Стандартизація є діяльністю, спрямованою на розробку та встановлення правил, вимог, характеристик та норм як обов'язкових для виконання, так і рекомендованих. Вона призначена забезпечувати право кожного споживача на придбання товарів і послуг належної якості, та право на безпечну і комфортну працю. Метою стандартизації є знаходження оптимального рішення для впорядкування в тій чи іншій галузі за допомогою багаторазового

використання встановлених вимог, положень та норм для вирішення дійсно існуючих, потенційних або планованих завдань [15].

Основні математичні методи аналізу й обробки даних, що найчастіше використовують при типових медичних дослідженнях, подаються у табл. 1.2

Таблиця 1.2 – Основні математичні методи обробки й аналізу даних, які використовуються при типових медичних дослідженнях[1]

Джерело інформації, завдання дослідження	Методи обробки й аналізу, які використовуються для реалізації поставлених завдань
Опитувальники, анкети, тести Обстеження стану здоров'я; думка того, кого обстежують; історії хвороби Виявлення прихованих взаємозв'язків	Кореляційний аналіз, метод автоматичної класифікації тощо Факторний аналіз
Скринінгові обстеження	Дискримінантний аналіз, кластерний аналіз, методи розпізнавання образів
Історії хвороби Клінічні обстеження, лікування та реабілітація хворих; ускладнення при лікуванні. Дослідження ефективності різних процедур, вивчення зв'язків між процедурами та їхніми наслідками	Кореляційний аналіз, дисперсійний аналіз, регресійний аналіз Методи оцінювання гіпотез, регресійний аналіз
Медико-статистичні дані Дослідження захворюваності, динаміка захворюваності, виявлення періодичності захворюваності	Методи аналізу випадкових процесів, спектральний аналіз, математичне моделювання

Продовження таблиці 1.2

Основні математичні методи обробки й аналізу даних

Експерименти Лабораторні експерименти й дослідження на тваринах при заданих умовах	Методи планування експериментів, регресійний аналіз, дисперсійний аналіз, багатомірний статистичний аналіз, методи математичного моделювання
Клінічні дослідження Порівняльні лікувальні дослідження, аналіз виживаності й спадковості з урахуванням належності пацієнта до певної групи, вивчення дозування препаратів Розробка методів діагностики	Дисперсійний аналіз, регресійний аналіз, дискримінантний аналіз, методи оцінювання гіпотез Дискримінантний аналіз, кластерний аналіз, методи розпізнавання образів
Дослідження механізмів дії патогенних факторів	Дисперсійний аналіз, регресійний аналіз, методи математичного моделювання
Клінічні лабораторні дані Зберігання, збір і передача клінічної інформації, аналіз якості й надійності лабораторних досліджень, догляду за пацієнтами	Дисперсійний аналіз, регресійний аналіз

Однак, досить часто заздалегідь важко визначити, який метод дасть найкращий результат. Тому варто передбачити можливість застосування різних способів обробки даних, що при використанні комп'ютерного аналізу стає набагато простішим[1].

Сучасні світові медичні інформаційні стандарти можна розділити на дві групи: термінологічні стандарти і стандарти обміну інформацією. Найбільш поширеними термінологічними стандартами є LOINC, MeSH, SNOMED та ін. Медичні предметні рубрики (Medical Subject Headings, MeSH) – всеосяжний

контрольований словник, що індексує журнальні статті та книги з природничих наук.

SNOMED (Systematized Nomenclature of Human and Veterinary Medicine – Міжнародна систематизована номенклатура людської і ветеринарної медицини) дозволяє кодування декількох аспектів хвороби та є багатоосьовою системою кодування. Вона складається з 11 пов'язаних взаємними посиланнями класифікацій, званих модулями (таблиця 1.3).

Таблиця 1.3 – Розділи міжнародної класифікації SNOMED

Вісь	Визначення	Опис
T	Топографія	Анатомічні умови
M	Життя організмів	Зміни в клітинах, тканинах і органах
L	Життя організмів	Бактерії і віруси
C	Хімічні	Ліки
F	Функція	Знаки і симптоми
J	Заняття	Умови, які описують заняття
D	Діагноз	Діагностичні умови
P	Процедура	Адміністративні, діагностичні та терапевтичні процедури
A	Фізичні представники	Пристрої, дії, пов'язані з хворобою
S	Громадський контекст	Суспільні умови і взаємодії в медицині
G	Загальний	Синтаксичні об'єднання і визначення

Розглядаючи можливості застосування номенклатури SNOMED, слід мати на увазі, що вона насамперед призначена для формалізації опису результатів клінічних спостережень. За її логічну модель автори пропонують наступну формулу: Для пацієнта з соціального середовища S з професією J [процедура P, застосована до місця T, показала/виявила морфологію M, живий

організм L, 10 хімічний, лікарський і біологічний продукт C, функцію F, фізичний вплив A, захворювання або діагноз D] [з модифікатором G] [16].

1.3 Процеси корозії металів медичних сплавів

У всьому світі та Україні щорічно виконуються мільйони хірургічних операцій пов'язаних з переломами кісткової тканини, більша частина з них передбачає фіксацію хірургічними імплантами [17]. Традиційні методи лікування використовують імпланти, зроблені з нержавіючих сталей, кобальтових, титанових сплавів та інші. Такі довговічні імпланти є інородними тілами, що може несути високий ризик запалення. Окрім того, вони назавжди заблоковують кістку від механічної експозиції, що ускладнює стабілізацію кісткової тканини, яка необхідна при механічних навантаженнях. З метою уникнення негативних наслідків проводять операції з видаленням імплантів. Такі операції дорого коштують та не виключають ризиків повторних переламів, а також потребують додаткового лікування [18].

При корозії імплантів з нержавіючих сталей в тканинах підвищується концентрація хрому, титану, заліза та нікелю. Тривала присутність таких імплантів в організмі людини може призводити до накопичення токсичних елементів, таких як алюміній, нікель і хром. Металоз тканин та корозія металевих імплантів із нержавіючої сталі найчастіше спостерігається у випадках застосування великих фіксаторів, ендопротезів, при тривалому перебуванні їх у організмі [19].

Титан має високу біологічну інертність, корозійну стійкість, більшу втомну міцність, легше піддається механічній обробці, а також він легше нержавіючої сталі приблизно в 3 рази [20]. Металоз тканин на титановій конструкції незначний. Недоліком титану і його сплавів вважається недостатня стійкість до механічного тертя та висока вартість, яка перевищує вартість сталі приблизно в 5 разів. Титанові сплави мають більш високі механічні характеристики і корозійну стійкість порівняно з чистим титаном. Однак їх

біосумісність набагато нижча. При їх використанні відзначалися прояви алергічних і дратівливих реакцій організму[20]–[21].

Властивості металів, що застосовуються для виготовлення імплантів, наведено в таблиці 1.4.

Таблиця 1.4 – Властивості металів і сплавів [23]

Склад матеріалу (% мас.)	Модуль пружності, ГПа	Границя міцності при розтягу, МПа	Відносне видовження при розриві, %	Склад поверхневого шару зразку
Ti(99)	97	240-550	15	TiO ₂
Ti(90) + Al(6)+V(4)	110	930	10 – 15	TiO ₂
Ti(90) + Al(6)+Nb(7)	105	860	10	TiO ₂
Нержавіюча сталь Fe (70)+Cr(18)+Ni(12)	193	485-860	12-40	Cr ₂ O ₃
Co(66)+Cr(27)+Mo(7)	235	655	8	Cr ₂ O ₃
Co(55)+Cr(20)+W(15)+Ni(10)	235	860	30	Cr ₂ O ₃
Co(45)+Ni(35)+Cr(20)+Mo(10)	235	793-1793	50-8	Cr ₂ O ₃
Co(52)+Ni(20)+Cr(20)+Mo(4)+W(4)	235	600-1310	50-12	Cr ₂ O ₃
Zr (99)	97	552	20	ZrO ₂
Ta (99)	188-190	205-517	1-30	Ta

Загалом металеві імпланти, завдяки великій границі міцності та модуля пружності, добре справляються з навантаженнями, що сприймають імпланти в процесі використання. Механічні властивості металевих імплантів значно

перевищують властивості кісток, що призводить до явища «стрес-екранування». Воно відбувається коли імплант приймає все механічне навантаження на себе, а кістка при цьому не зміцнюється[23].

Металеві імпланти не є біорозчинними, їх міцність та пружність вищі за характеристики кісток, що викликає «стрес-екранування», в той час як біорозчинні полімери та композити, навпаки, мають занадто низькі механічні властивості. Усунути недоліки можна використанням магнію та сплавів.

Перспективним матеріалом для виготовлення таких імплантів вважаються біорозчинні матеріали, що здатні розчинятися в організмі людини через певний проміжок часу.

Магній та продукти його корозії мають відмінну біосумісність. Дослідження показують позитивний вплив на остеогенез, магній бере участь в обмінних процесах, тасно взаємодіє з калієм, натрієм і кальцієм [22]. При цьому механічні властивості магнію, зокрема, модуль пружності, максимально близькі до властивостей кісток, що відображається в таблиці 1.5.

Таблиця 1.5 – Порівняння механічних властивостей магнію та кісток людини [23]

Матеріал	Модуль пружності, ГПа	Границя міцності, МПа	Відносне видовження, %
Стегно	17,2	121	1,4 – 3,1
Велика гомілкорова кістка	18,1	140	
Мала гомілкорова кістка	18,6	146	
Плечорова кістка	17,2	30	
Променеорова кіста	18,6	149	
Ліктьорова кістка	18,0	148	2-3
Магній	45,0	113	

Вирішенням даної проблеми може стати застосування магнієвих імплантів, що здатні розчинятися в організмі людини. Магній представляє собою природній елемент організму, який входить у склад м'язової та кісткової тканин, бере участь у обмінних процесах організму. Його фізико-хімічні властивості наближені до властивостей кісткової тканини. Також доведено, що магній та його продукти корозії мають хорошу біосумісність. Проте, чистий магній дуже крихкий та має низький рівень міцності для застосування його в якості імпланту. Покращити його механічні властивості можливо за рахунок легування [24].

Порівняння властивостей магнію, сплаву МЛ10 і кісткової тканини людини наведено в таблиці 1.6.

Таблиця 1.6 – Порівняльні фізико-механічні властивості магнію, сплаву МЛ10 і кісткової тканини людини [25]

Матеріал	Модуль пружності, ГПа	Границя міцності, МПа	Відносне видовження, %	Щільність, г/см
Кісткова тканина	3-20	30-150	1,4 – 3,1	1,8 – 2,1
Чистий магній	41-45	113	2 – 3	1,74
МЛ10	44	>235	>3	1,78

З урахуванням цього, магнієві сплави Mg-Nd-Zr перспективні для створення біорозчинних імплантів, а сплав МЛ10 найкраще підходить за своїми механічними та технологічними властивостями. Проте тривала корозія знижує його механічні властивості. Витримка зразків в гелофузині (штучний замінник крові) протягом 3 місяців, що дорівнює середньому часу консолідації перелому, показала значну втрату його міцності [26].

Виходячи з цього, актуальним стає дослідження механічних властивостей магнієвого сплаву для оптимізації його хімічного складу.

Механічні властивості сплаву МЛ10 після тривалої витримки в гелофузині наведені в таблиці 1.7.

Таблиця 1.7 – Механічні властивості сплаву МЛ10 після тривалої витримки в гелофузині [27]

Вихідний		Термін витримки					
		1 місяць		2 місяці		3 місяці	
σ_B , МПа	δ , %	σ_B , МПа	δ , %	σ_B , МПа	δ , %	σ_B , МПа	δ , %
235	3	178	2,6	146	2,3	115	1,2

На кафедрі фізичного матеріалознавства проф. Шаломеевим було розроблено та випробувано біодеградуєчий ливарний сплав на основі магнію для остеосинтезу з оптимальним хімічним складом [28], властивості якого наведені в таблиці 1.8.

Таблиця 1.8 – Механічні властивості сплаву з оптимальним хімічним складом при витримці в штучних замінниках крові [28]

Початковий			1 місяць			2 місяці			3 місяці		
σ_B , МПа	δ , %	E, ГПа	σ_B , МПа	δ , %	E, ГПа	σ_B , МПа	δ , %	E, ГПа	σ_B , МПа	δ , %	E, ГПа
Гелофузин											
271	4,4	45,3	246	4,1	45,3	220	3,7	45,2	188	3,3	45,3
Венофундин											
266	4,8	44,2	238	4,3	44,4	212	3,7	44,3	171	3,2	44,1
Фізрозчин											
270	4,6	45,1	259	4,4	45,1	241	4,2	44,9	218	3,8	45,0

Сплав, узятий за прототип, має низькі показники міцності, як при кімнатних, так і при підвищених температурах і механічних властивостях. Введення скандію в сплав у зазначених межах збільшує кількість і розміри

інтерметалідної фази й зменшує кількість евтектоїда, що позитивно позначається на механічних властивостях, як при звичайних, так і підвищених температурах [28].

Були проведені порівняльні випробування механічних властивостей і тривалої міцності зразків, виготовлених з відомого й пропонованого сплавів. Отримані порівняльні результати досліджень наведені в таблиці 1.9.

Таблиця 1.9 – Порівняльні таблиці результатів досліджень [29]

Сплав	Вміст скандію, %	Механічні властивості			Тривала міцність		
		$T_{\text{вип.}}, ^\circ\text{C}$	$\sigma_{\text{в}}, \text{МПа}$	$\delta, \%$	$T_{\text{вип.}}, ^\circ\text{C}$	$s, \text{МПа}$	$t_{\text{р}}, \text{час.}$
Аналог	-	20	226,0	3,5	150	80	260 ⁴⁰
Пропонований сплав	0,38	20	243,0	4,5	150	80	338 ³⁰
	0,65	20	254,0	5,6	150	80	375 ⁴⁰
	0,9	20	250,0	7,3	150	80	416 ²⁰

Аналіз результатів механічних властивостей досліджуваних сплавів демонструє, що запропонований сплав має більш високий комплекс механічних властивостей та тривалої міцності в порівнянні з аналогом за рахунок утворення дрібнодисперсних інтерметалідних сполук [29].

Таким чином, використання запропонованого сплаву дозволяє одержувати магнієві сплави з високим комплексом механічних властивостей та тривалої міцності.

2 МАТЕРІАЛИ ТА МЕТОДИ

2.1 Web технології віртуалізації

У процесі розвитку інтернет технологій з'явилася потреба створювати web додатки зі складною графікою. До появи HTML5 основним інструментом для цього була технологія Flash. Основними причинами, з яких Flash поступився першістю іншим технологіям можна назвати наявність серйозних уразливостей та проблеми у роботі Flash-плагінів. У 2015 році компанія Adobe оголосила Flash застарілою технологією і попросила розробників та користувачів відмовитися від технології, остаточно визнавши першість HTML5, WebGL і WebAssembly в області інтерактивного та мультимедійного веб-контенту. Підтримка Flash здійснювалась до кінця 2020 року, після чого оновлення та розповсюдження плагіна було припинено[30]-[31].

Adobe Flash — це середовище для створення застосунків під Flash платформу (Flash Platform), разом з нею існують і інші інструменти (середовища): Adobe Flex Builder, Flash Development Tool (FDT), та інші.

Flash-Файли мають розширення .swf і для перегляду вимагають наявності Adobe Flash Player, що може бути встановлений як плагін у браузер. Flash Player поширюється безплатно через сайт Adobe. Вихідні файли з розширенням .fla створюються в середовищі розробки Macromedia Flash, а потім компілюються в зрозумілий для Flash Player формат — .swf.

В основі Flash лежить векторний морфінг, тобто плавне «перетікання» одного ключового кадру в інший. Це дозволяє робити досить складні мультиплікаційні сцени, задаючи лише кілька ключових кадрів для кожного персонажа.

Другий «кит» Flash'a — повна програмованість. Flash використовує мову програмування ActionScript, яка по синтаксисі є схожою із JavaScript. Остання версія мови (ActionScript 3.0) є повноцінною об'єктно-орієнтованою мовою [31].

Недоліки Flash: необхідність встановлення Adobe Flash Player; великоваговий контент Flash-елементів, який довго кешується; високі навантаження процесора та споживання оперативної пам'яті.

На зміну технології Flash приходить HTML5 Canvas і WebGL - два елементи, які об'єднують HTML5 та JavaScript. У них є свої переваги та недоліки, особливості та тонкощі у використанні. Але, обидва інструменти спрямовані на створення та демонстрацію графічних складових HTML5. Це можуть бути растрові зображення, фони сайтів, анімована графіка, динамічні заставки, а також браузерні 2D та 3D ігри.

Canvas – це HTML5-елемент, за допомогою якого можна працювати з растровою графікою через JavaScript. До Canvas звертаються за необхідності розробки графіків, додавання анімації та застосування інших графічних елементів у веб-додатках.

Ідея елемента canvas (полотно) була вперше висунута та реалізована компанією Apple у продукті Mac OS X WebKit для створення міні додатків - так званих віджетів. До появи цього елемента використання функцій малювання в браузерах було можливо лише за умови залучення модулів, що підключаються, таких як Flash плагіни Adobe, мови векторної розмітки (VML; тільки в Internet Explorer) або ж нестандартних програмних рішень на основі JavaScript.

При використанні елемента canvas на своїй сторінці створюється прямокутна область. Включеним у сторінку елементом canvas можна маніпулювати, використовуючи JavaScript. До нього можна додавати графіку, лінії та текст, малювати на ньому і навіть додавати до нього повноцінну анімацію.

Програмний інтерфейс HTML5 Canvas підтримує ті самі операції 2D малювання, що і більшість сучасних операційних систем та кадрів ворків. Специфікація HTML5 Canvas спеціально спроектована так, щоб нагадувати існуючі системи для програмування 2D графіки.

Щоб програмним шляхом використовувати полотно, насамперед необхідно одержати його контекст (context). Після цього можна виконати

необхідні дії по відношенню до контексту, а потім остаточно застосувати його результат. Виконання змін на полотні аналогічні до виконання транзакцій у базах даних: спочатку підготовлюються деякі дії, а потім фіналізується транзакція.

Як і у випадку більшості елементів HTML, для додавання рамок, пробілів, полів тощо. елемент canvas можна використовувати CSS стилі. Крім того, деякі атрибути CSS успадковуються вмістом canvas.

Модифікації, або, в термінології графічних систем, перетворення можуть застосовуватися послідовно, об'єднуватися або видозмінюватися за необхідністю. Кожна графічна операція пропускається через модифікуючий шар, де графічні об'єкти зазнають усіх необхідних змін, перш ніж виводяться на полотно.

Ключова рекомендація щодо створення повторно використовуваного коду полягає в тому, що малювання, як правило, слід виконувати з прив'язкою до початку координат, а для надання малюнку остаточного вигляду та поміщення його в потрібне місце на сторінці необхідно застосовувати перетворення - масштабування, трансляцію (перенесення), повороти тощо [34].

До переваг Canvas можна віднести: наявність апаратного прискорення, можливість точкового попиксельного редагування за рахунок растрового формату, можливість використання фільтрів для ретуші зображень, наявність багатого асортименту бібліотек, що підтримуються, зручність роботи з великою кількістю елементів.

До недоліків: відсутність можливості очищення пам'яті, через що сильно споживаються ресурси процесора та ОЗУ; низька продуктивність при використанні растрових форматів високих дозволів; необхідність вручну промальовувати кожен елемент окремо; можливість впровадження спеціальних елементів-маячків для стеження за онлайн-активністю користувачів; немає офіційної підтримки 3D тому для створення 3D доведеться використовувати спеціальні бібліотеки, такі як WebGL [30],[33].

2.2 Огляд мов програмування

Мова розмітки гіпертекстових сторінок (HTML - Hypertext Markup Language) є мовою, розробленою спеціально для створення Web-документів. Він визначає синтаксис та розміщення спеціальних інструкцій (тегів), які не відображаються на екрані, але вказують браузеру, як відображати вміст документа. Він також використовується для створення посилань на інші документи, локальні чи мережеві.

Документ HTML містить текст (вміст сторінки) та вбудовані теги - інструкціями про структуру, зовнішній вигляд та функції вмісту. Документ HTML поділяється на дві основні частини: заголовок – head та тіло – body. Заголовок містить такі відомості про документ, як його назва та методична інформація, що описує вміст. У тілі знаходиться самий вміст документа. Кожен тег складається з імені, за яким може йти список необов'язкових атрибутів, всі вони знаходяться всередині кутових дужок `< >`.

Більшість тегів є контейнерами. Це означає, що вони мають початковий (відкриваючий чи стартовий) і кінцевий (закриває) теги. Текст, що знаходиться між тегами, буде виконувати інструкції, що містяться в них.

Деякі теги не мають завершальних тегів, тому що вони використовуються для розміщення окремих (автономних) елементів на сторінці. Одним з них є тег `<img>`, він просто поміщає графіку в потік сторінки. Інші автономні теги - це розрив рядка (`<br>`), горизонтальна лінія (`<hr>`) та теги, що містять інформацію про документ і не впливають на вміст, що виводиться на екран, такі як `<meta>` та `<base>`.

Атрибути додаються до тега для розширення або модифікації його дій. До одного тега можна додати кілька атрибутів. Якщо атрибути тега слідує після імені тега, вони розділяються одним або декількома пробілами. Порядок прямування не важливий. Більшість атрибутів мають значення, які йдуть за знаком рівності (=), що знаходиться після імені атрибута. Довжина значень обмежена 1024 символами. Значення можуть бути чутливими до регістру [34].

JavaScript – це мова керування сценаріями перегляду гіпертекстових сторінок Web на стороні клієнта. Основна ідея JavaScript полягає у можливості зміни значень атрибутів HTML-контейнерів та властивостей середовища відображення у процесі перегляду HTML-сторінки користувачем. При цьому перезавантаження сторінки не відбувається.

На практиці це виявляється у тому, що можна, наприклад, змінити колір фону сторінки чи інтегровану у документ картинку, відкрити нове вікно чи попередження.

Для створення механізму керування сторінками на стороні клієнта було запропоновано використовувати об'єктну модель документа. Суть моделі в тому, що кожен HTML-контейнер – це об'єкт, який характеризується трійкою:

- властивості;
- методи;
- події.

Об'єктну модель можна як спосіб зв'язку між сторінками і браузером. Об'єктна модель - це представлення об'єктів, методів, властивостей та подій, які присутні та відбуваються у програмному забезпеченні браузера, у вигляді зручному для роботи з ними коду HTML та вихідного тексту сценарію на сторінці. Ми можемо з її допомогою повідомляти наші побажання браузеру і далі відвідувачу сторінки. Браузер виконає наші команди та відповідно змінить сторінку на екрані [35].

Багато HTML-контейнерів мають атрибути. Наприклад, контейнер якоря `<A ...>...</A>` має атрибут HREF, який перетворює його на гіпертекстове посилання.

Не всі атрибути можуть змінювати значення. Наприклад, висота і ширина графічної картини визначаються за першою завантаженою в момент відображення сторінки картинці. Усі наступні картини, які замінюють початкову, масштабуються до неї.

У термінології JavaScript методи об'єкта визначають функції зміни його властивостей. Наприклад, з об'єктом "документ" пов'язані методи open(), write(), close(). Ці методи дозволяють згенерувати чи змінити зміст документа.

Крім методів та властивостей об'єкти характеризуються подіями. Власне, суть програмування JavaScript полягає в написанні обробників цих подій. Наприклад, з об'єктом типу button (контейнер INPUT типу button - "кнопка") може відбуватися подія click, тобто. користувач може натиснути кнопку. Для цього атрибуту контейнера INPUT розширені атрибутом обробки події click – onClick.

Браузер виконує JavaScript-код. У нього вбудований інтерпретатор JavaScript. Виконання програми залежить від того, коли і як цей інтерпретатор отримує управління. Це, своєю чергою, залежить від функціонального застосування коду. У загальному випадку можна виділити чотири способи функціонального застосування JavaScript:

- гіпертекстове посилання (схема URL);
- обробник події (handler);
- підстановка (entity) (у Microsoft Internet Explorer реалізована у версіях від 5.X і вище);
- вставлення (контейнер SCRIPT) [37].

2.3 Огляд середовищ розробки

Pycharm – інтегроване середовища розробки для мови програмування Python, розроблене компанією JetBrains.

Редактор "отримує" код та розуміє його структуру, підтримуючи всі можливості мови для проектів. Він підтримує автозавершення коду, рефакторинг, запобігання помилок та інше.

Також, підтримується крос-технологічна розробка. Крім основної мови програмування, підтримуються:

- JavaScript;

- TypeScript;
- CoffeeScript;
- Cython;
- HTML / CSS;
- SQL;
- Node.js;
- AngularJS;
- Emmet.

Всі перелічені вище технології також мають підтримку рефакторинга, налагодження, юніт-тестування та інше [38].

PyCharm дозволяє підключатися до різних машин та віддалено створювати програмні додатки. Для цього можна використовувати вбудовану консоль SSH для підключення до машин та віддаленого виконання різних завдань. Можливо запускати, налагоджувати та профілювати програми Python у віддаленому середовищі, замінюючи локальний інтерпретатор на віддалений інтерпретатор. Крім того, PyCharm дозволяє програмістам створювати відтворювані середовища розробки за допомогою Vagrant, та спрощувати розробку розподілених програм через Docker [39].

Також середовища розробки також мають вбудовані інструменти розробника. Вони допомагають виконувати багато рутинних завдань безпосередньо з середовища розробки, завдяки інтеграції з системами керування версіями, підтримці віддаленого розгортання, баз даних/SQL, інструментів командного рядка, Docker, Composer, REST Client та багатьох інших інструментів [39].

PyCharm забезпечує підтримку популярних веб-фреймворків, таких як Django, Flask, Google App Engine, Pyramid та web2py. Він надає можливість створювати та налагоджувати Django-шаблони, працювати з утилітами manage.py та appcfg.py, а також використовувати специфічні для фреймворків автодоповнення та навігацію.

Функція Live Editing Preview дозволяє відкрити редактор і браузер одночасно і слідкувати за результатами внесених до коду змін на веб-сторінці. PyCharm зберігає зміни автоматично, і вони миттєво відображаються у браузері без перезавантаження сторінки.

З PyCharm можливо працювати з ноутбуками Jupyter, запускати команди в інтерактивній консолі Python, підключати бібліотеки Anaconda, а також працювати з іншими бібліотеками для наукових обчислень та аналізу даних, включаючи Matplotlib та NumPy.

За роки розробки платформи IntelliJ було створено більше 50 плагінів для PyCharm, які забезпечують підтримку додаткових систем контролю версій, інтеграцію з інструментами та фреймворками, а також розширюють можливості редактора, наприклад, за рахунок емуляції Vim [40].

Аналогом PyCharm для створення web-застосунків може вважатися Visual Studio Code.

Visual Studio Code – це сервіс для кроссплатформенної розробки веб- та хмарних додатків. VS Code дозволяє розробляти як консольні програми, так і програми з графічним інтерфейсом, у тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-програми, веб-служби як у рідному, так і в керованому кодах для всіх платформ.

У редакторі присутні вбудований налагоджувач, інструменти для роботи з Git та засоби рефакторингу, навігації за кодом, автодоповнення типових конструкцій та контекстної підказки.

Продукт підтримує розробку для платформ ASP.NET і Node.js, і вважається легковажним рішенням, яке дозволяє обійтися без повного інтегрованого середовища розробки. Великим плюсом редактора є підтримка великої кількості мов, таких як C++, C#, Python, PHP, JavaScript та інші.

Особливості VSCode:

- вбудовані інструменти інтеграції з GitHub, GIT, а також Visual Studio Team Services для швидкого тестування, складання, пакування та розгортання різних типів додатків;

- зручність роботи з Unity-проектами;
- робота з Mono та Node.js за допомогою вбудованого налагодчика;
- підтримка TypeScript та JavaScript;
- публікація створених програм у Microsoft Azure через сервіс Visual Studio Team Services;
- написання коду для конкретного завдання з його подальшою інтеграцією в проект (з надбудовою або безпосередньо);
- велика бібліотека шаблонів, готових фрагментів коду та сніпетів з можливістю додавання своїх елементів;
- одночасна робота з кількома проектами (у кількох вікнах);
- інтерфейс можна розділити на дві панелі для порівняння коду;
- функція налагодження [41].

Порівняльта таблиця середовищ розробки наведено в таблиці 2.1.

Таблиця 2.1 – Порівняльта таблиця середовищ розробки

	PyCharm	VSCode	CodeLite
Мультиплатформність	Windows, Linux, Mac OS X, Android	Windows, Linux, Mac OS X	Windows
Вартість	Умовно безкоштовний (версія Community)	Умовно безкоштовний	Платний (70\$)
Наявність візуального конструктора	+	+	+
Ліцензія	ASL	ASL (Пропрієтарна)	GPL
Автодоповнення	+	-	+
Статичний аналізатор коду	+	-	-
Дизайн на основі GUI	+	-	-

В якості середовища для розробки було вирішено використовувати Pycharm.

2.4 Методи проведення випробувань матеріалів медичного призначення

Для проведення випробувань матеріалів медичного призначення на кафедрі радіотехніки та телекомунікацій Національного університету «Запорізька Політехніка» було створено фізичну модель що дозволяє імітувати процес корозії біодеградуємих матеріалів. Ця модель дозволяє імітувати процес тиску крові в артеріях та отримувати дані. Тиск регулюється насосом, температура термopарою. Для управління насосом було створено програмне забезпечення, що працює на платі Raspberry Pi.

Тривалість випробувань для магнію і його сплавів – 60-90 діб.

Оцінку результатів випробувань проводять відповідно до виду корозії і за обраним критерієм:

- зміна зовнішнього вигляду поверхні зразків;
- зміна по глибині в кількості корозійних поразок;
- зміна механічних властивостей;
- зміна маси;
- кількість виділеного водню;
- зміна глибини і характеру корозійних поразок.

Устаткування для проведення випробувань зображено на рисунку 2.1.

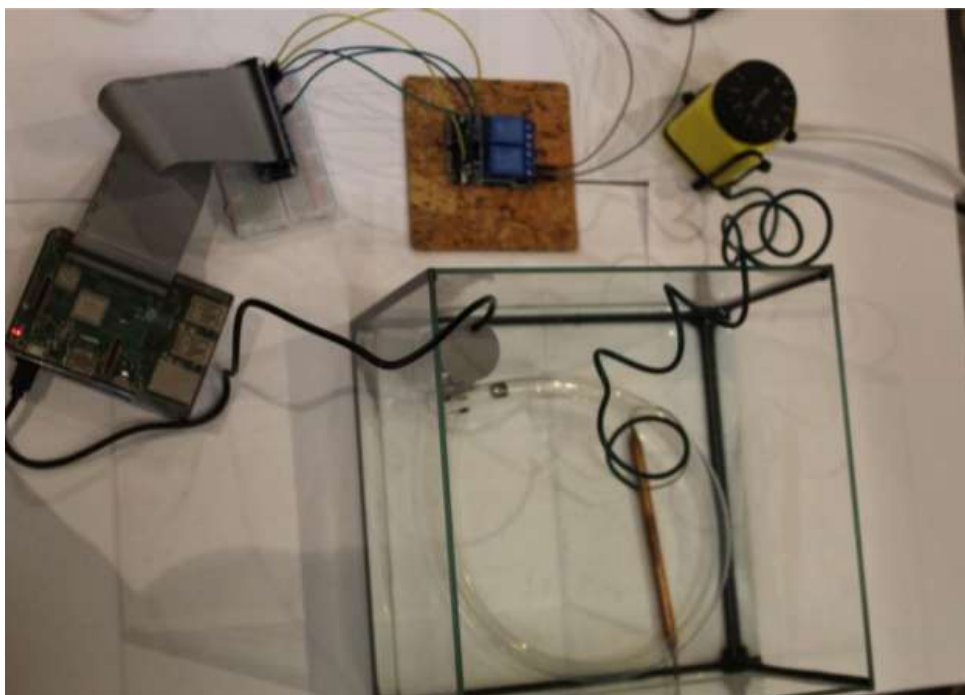


Рисунок 2.1 – Експериментальне устаткування [41]

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Проектування інтерфейсу

Інтерфейс віртуального середовища умовно поділений на інтерфейс керування, основну сцену візуалізації експерименту та відображення результатів.

Для підвищення якості зображення та уникнення використання повторного коду було вирішено створити зображення в середовищі Illustrator.

Adobe Illustrator – програма для роботи з векторною графікою. За допомогою Adobe Illustrator створюють ілюстрації, іконки, патерни, логотипи, різноманітні макети для друку та багато іншого.

Основними функціями Adobe Illustrator є:

- робота з геометричними фігурами;
- створення ілюстрацій;
- трасування растрової графіки;
- редагування контурів;
- застосування заливок;
- підготовка до друку;
- робота з монтажними областями;
- перетворення тексту на криві [42].

Кожен елемент, що використовується у віртуальному середовищі, створено окремо та згруповано, що надає можливості динамічно завантажувати зображення, спрощує анімацію, та у разі необхідності робить елементи «клікабельними».

Основними елементами проведення експерименту є:

- прозорий бокс (акваріум);
- рідина (гелофузин, венофузин або фізрозчин);
- насос, що створює тиск;
- термопара, для підтримки сталої температури;

- плата управління;
- термометр;
- напівпрозора трубка;
- стенд, над яким проводиться експеримент.

На рисунках 3.1 – 3.7 зображено елементи віртуального середовища.



Рисунок 3.1 – Прозорий бокс (акваріум)



Рисунок 3.2 – Насос

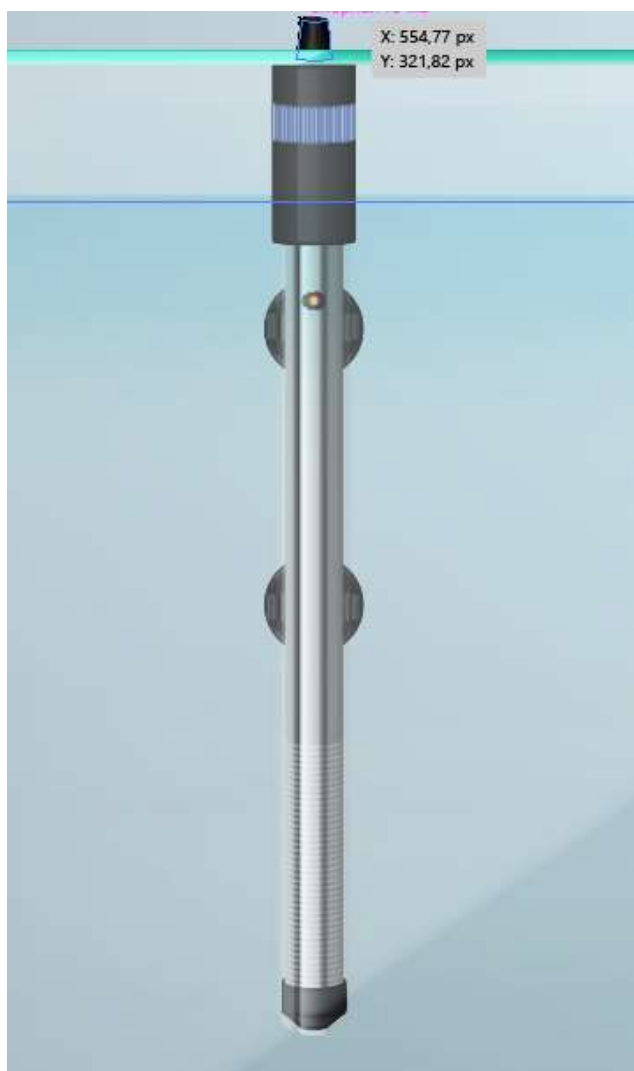


Рисунок 3.3 – Термостат

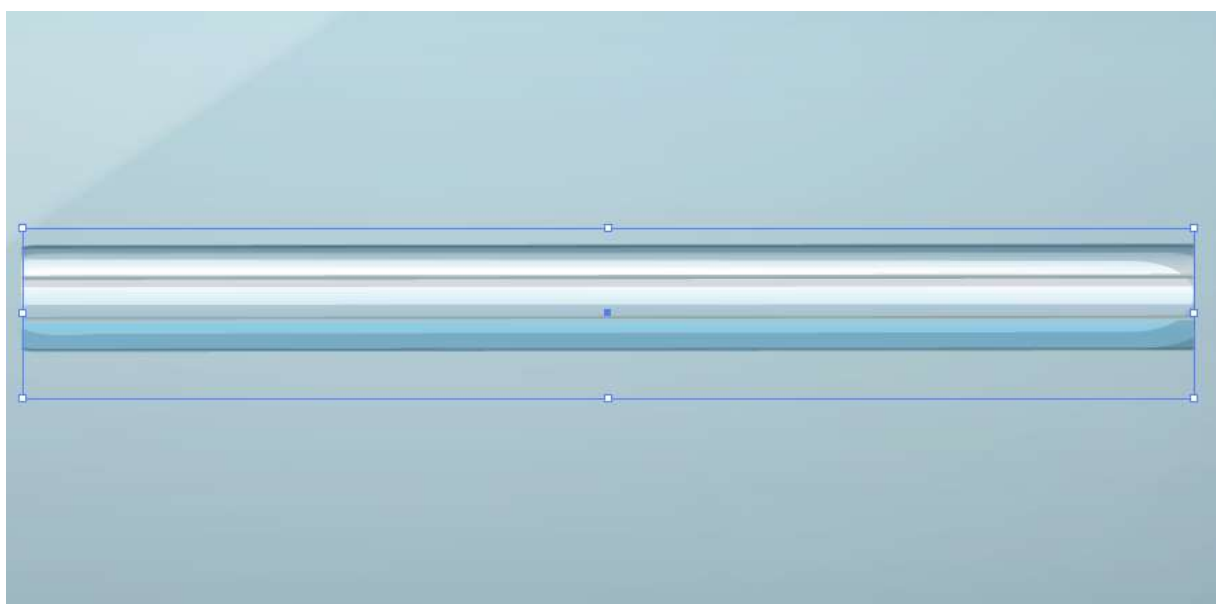


Рисунок 3.4 – Напівпрозора трубка



Рисунок 3.5 – Стент на початку експерименту

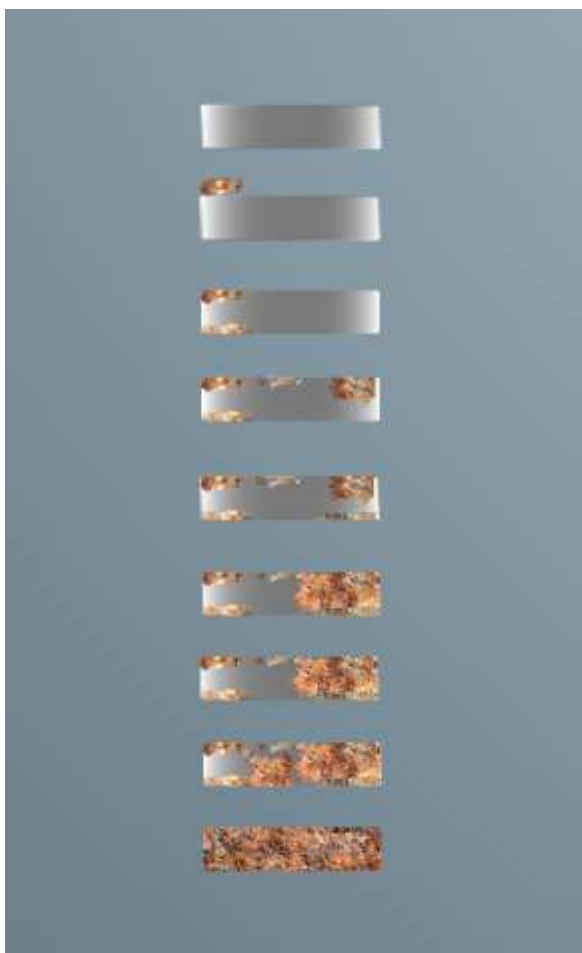


Рисунок 3.6 – Стенти з різною ступінню корозії



Рисунок 3.7 – Термометр

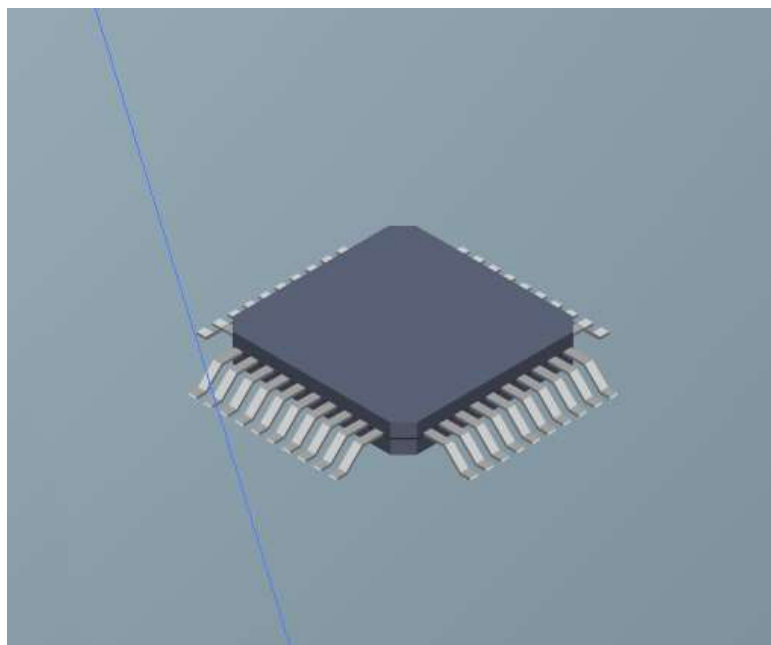


Рисунок 3.8 – Плата управління експериментом

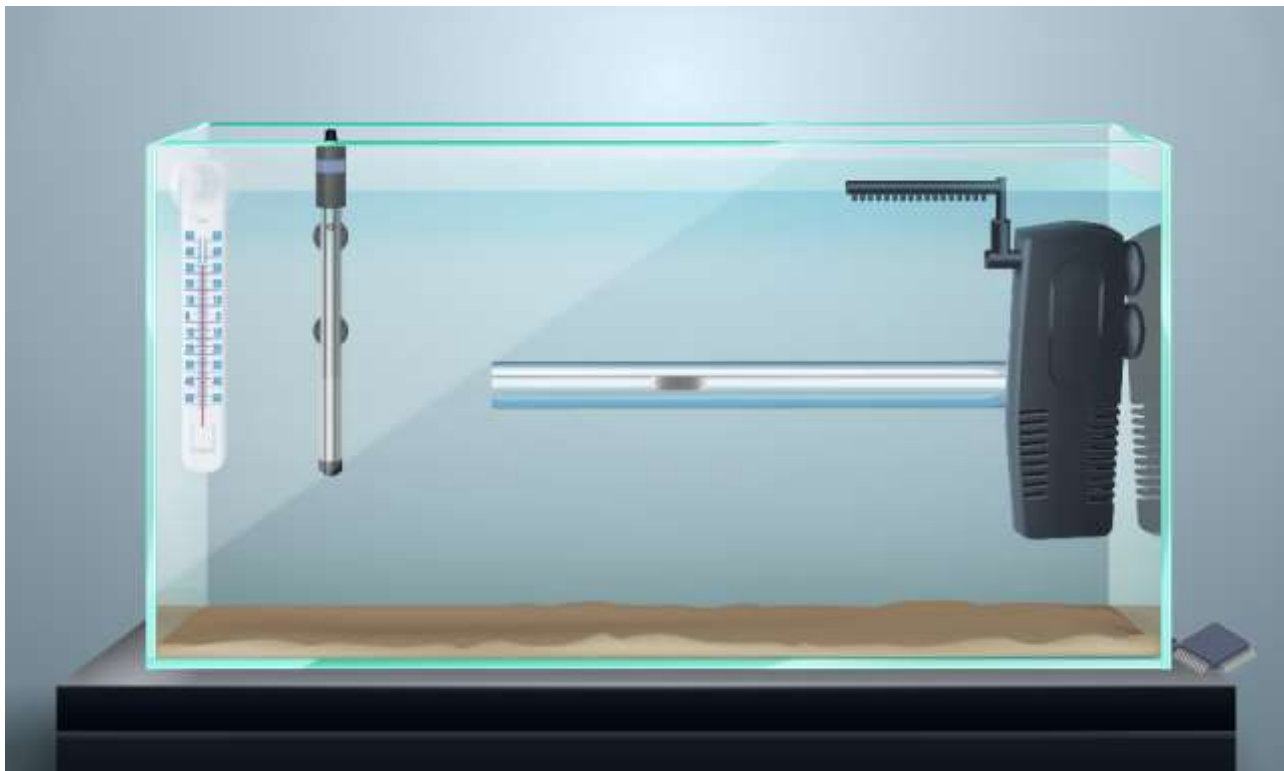


Рисунок 3.9 – Основна сцена візуалізації експерименту

Вхідними даними для запуску експерименту є тиск, температура, рідина, час та склад стенту.

Для створення інтерфейсу керування використовувались елементи управління html5 checkbox, button, input, text. Зовнішній вигляд інтерфейсу керування експериментом наведено на рисунку 3.10.

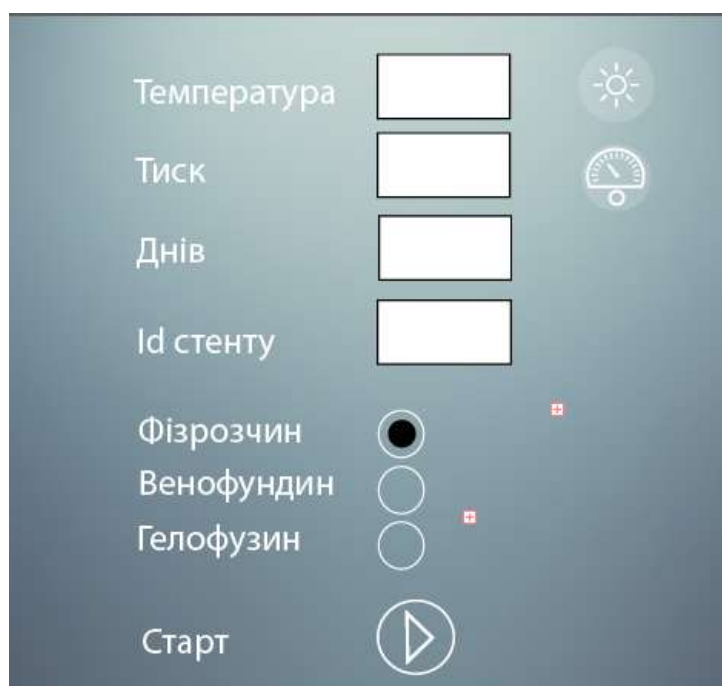


Рисунок 3.10 – Інтерфейс керування експериментом

Після візуалізації експерименту відображаються результати досліджень для обраного стенту. Надається інформація про механічні властивості та тривалу міцність. Інтерфейс для відображення результатів досліджень зображено на рисунку 3.11.



Рисунок 3.11 – Інтерфейс керування експериментом

3.2 Опис модулів розроблюваної системи

Розробка програмного забезпечення виконувалась в середовищі розробки PyCharm. Розробка програмного продукту відбувалася, використовуючи Rest api.

Модуль `index.html` відповідає за інтерфейс користувача, зовнішнє відображення моделювання експерименту. Модуль `style.css` відповідає за доповнення основного модуля стилями. Модуль `canvas.js` відповідає за обробку даних та містить у собі класи для анімації середовища. Модуль `database.json` містить у собі дані результатів експериментів.

В таблиці 3.1 наведено опис модулів реалізованого ПЗ з визначенням їх основних функцій.

Таблиця 3.1 – Опис модулів реалізованого ПЗ з визначенням основних функцій

№	Модуль	Опис
1	<code>index.html</code>	Основний модуль середовища
2	<code>style.css</code>	Відповідає за стилі та оформлення
3	<code>database.json</code>	База даних
4	<code>canvas.js</code>	Відповідає за логіку та анімацію
5	<code>img</code>	Містить ресурси середовища

3.3 Проектування бази даних

Використовувана база даних проектувалася з використанням концепції CRUD (Create, read, update, delete) – створення, зчитування, оновлення та видалення. Данна концепція використовується для розділення програмного коду на частини відповідно до виконуваної дії.

Концептуальну модель бази даних зображено на рисунку 3.12.

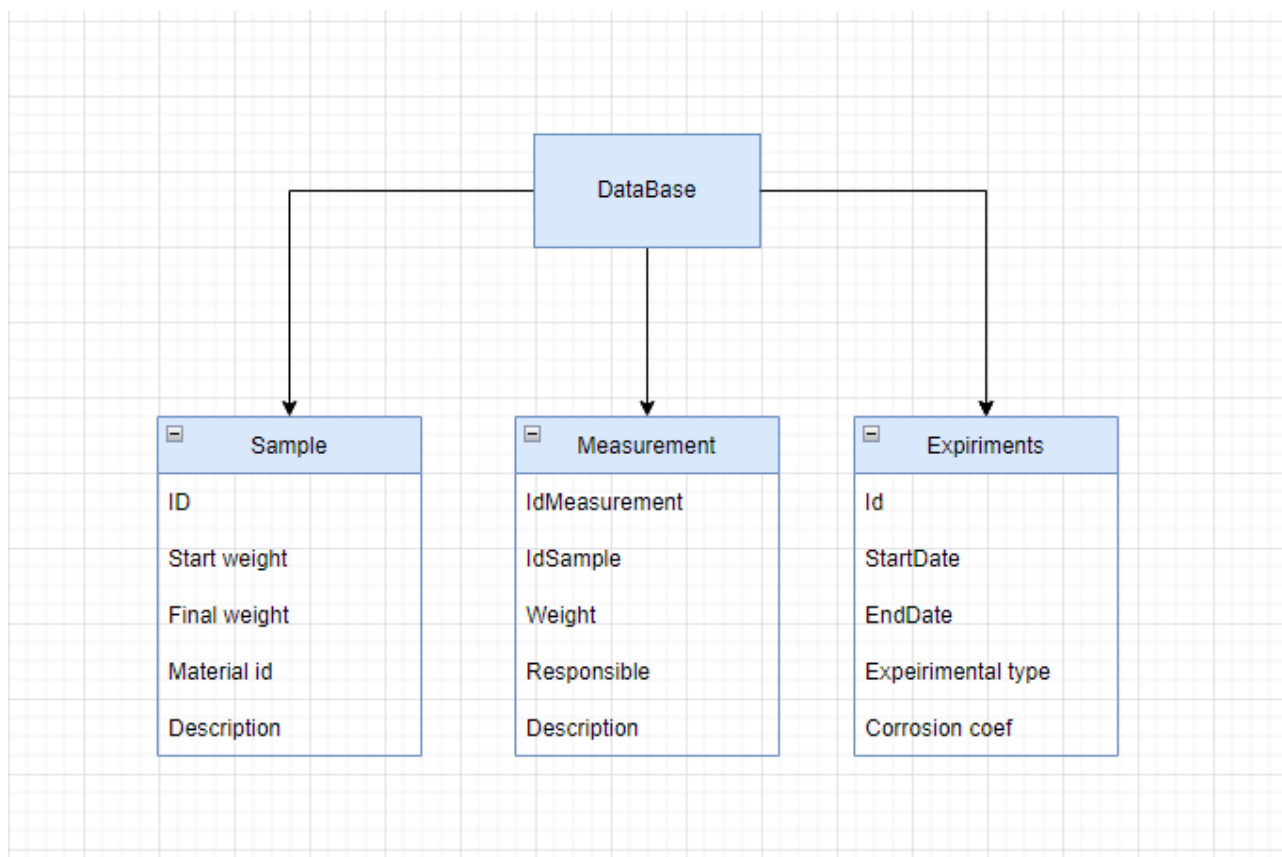


Рисунок 3.12 – Концптуальна модель бази даних

Сутність Sample (id, start weight, final weight, material id, description) – сутність, що містить в інформацію щодо зразка, що досліджується. Id – первинний ключ, Start weight – вага зразку на початок експерименту, Final weight – вага зразку на кінець експерименту, Material ID – id матеріалу зразку, Description – опис. Опис сутності полів «Sample» та їх типу наведено в таблиці 3.2.

Таблиця 3.2 – Опис полів сутності «Sample»

№	Назва поля	Тип даних	Розмір
1	ID	uniqueidentifier	255
2	Start weight	float	128
3	Final weight	float	128
4	Material ID	int	255
5	Description	varchar	100

Сутність Measurement (idmeasurement, idsample, weight, responsible, description) – сутність, що містить дані щодо конкретного виміру. IdMeasurement – первинний ключ, IdSample – первинний ключ, Weight – вага зразку, Material ID – id матеріалу зразку, Description – опис. Опис сутності полів «Sample» та їх типу наведено в таблиці 3.3.

Таблиця 3.3 – Опис полів сутності «Measurement»

№	Назва поля	Тип даних	Розмір
1	IdMeasurement	int	255
2	IdSample	int	128
3	Weight	float	128
4	Responsible	varchar	50
5	Description	varchar	100

Сутність Experiment (ID, StartDate, EndDate, Experimental type, Description, Corrosion coef, V,) містить інформацію щодо проведених експериментів. Опис сутності полів «Sample» та їх типу наведено в таблиці 3.4.

Таблиця 3.4 – Опис полів сутності «Experiments»

№	Назва поля	Тип даних	Розмір
1	ID	int	255
2	StartDate	datetime	128
3	EndDate	datetime	128
4	Experimental type	int	50
5	Corrosion coef	float	100

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для візуалізації процесів корозії магнієвих сплавів за допомогою віртуального середовища.

Для функціонування розробленого програмного продукту, а також в разі його доопрацювання, необхідно мати таку мінімальну апаратну платформу: персональний комп'ютері з тактовою частотою мікропроцесора не менше 2,3 ГГц, з оперативною пам'яттю з мінімальним об'ємом 4 Гб. На персональному комп'ютері повинно бути встановлена операційна система Windows 7/10 з підтримкою .NET Framework 4.0.

4.2 Характеристики програми

Програмний продукт надає наступні можливості:

- задання початкових умов експерименту;
- візуалізація процесів корозії;
- відтворення результатів експериментів.

4.3 Звертання до програми

Для роботи з програмою необхідно виконати міграцію бази даних та запустити файл index.html.

Для роботи повинні бути наявні монітор, клавіатура, та маніпулятор «миша», а також встановлений Net Framework версії 4.0.

4.4 Початкові та вихідні дані

Вхідними даними до програми є тиск, температура, рідина, час та склад стенту. Вихідними даними є візуалізації процесів корозії магнієвих сплавів та результатів експериментів.

4.5 Повідомлення програмісту

Для полегшення розуміння функцій програми в коді залишені коментарі. Програма може бути дороблена шляхом перекопіювання на інші платформи, доробки дизайну та впровадженням нових функцій.

4.6 Опис методики проведення тестування

Тестування буде проводитися на ПК, з наступними характеристиками:

- операційна система Windows 10;
- процесор з тактовою частотою 3.6 ГГц;
- оперативна пам'ять 8 Гб.

На рисунку 4.1 зображено діаграму послідовностей.

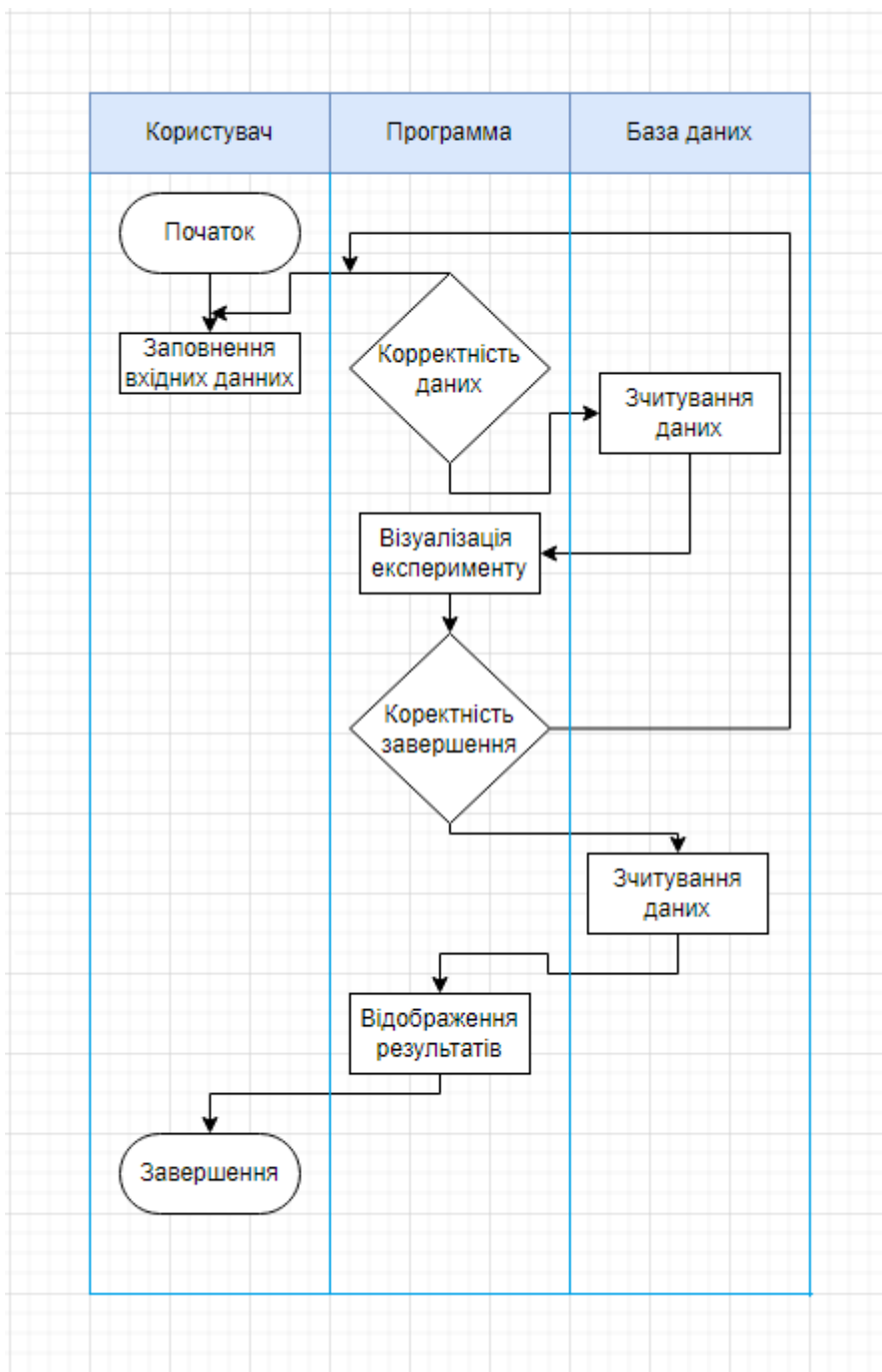


Рисунок 4.1 – Діаграма послідовностей

В Python вбудований модуль unittest, що підтримує автоматизацію тестів. Для автоматизації тестів, unittest підтримує деякі важливі концепції: випробувальний стенд (test fixture), тестовий випадок (test case), набір тестів (test suite), виконавець тестів (test runner).

Test fixture – виконується підготовка, необхідна виконання тестів і всі необхідні дії для очищення після виконання тестів. Це може містити, наприклад, створення тимчасових баз даних або запуск серверного процесу.

Test case – мінімальний блок тестування. Він перевіряє відповіді на різні набори даних. Модуль unittest надає базовий клас TestCase, який можна використовувати для створення нових тестових випадків.

Test suite – кілька тестових випадків, наборів тестів. Він використовується для об'єднання тестів, які мають бути виконані разом.

Test runner – компонент, який управляє виконанням тестів та надає користувачеві результат. Виконавець може використовувати графічний або текстовий інтерфейс або повертати спеціальне значення, яке повідомляє результати виконання тестів [43].

Лінтер – статичний аналізатор для мови програмування. Він повідомляє про потенційно небезпечні вирази в коді, які можуть призвести до аварійного завершення програми. Він має можливість повідомляти про застарілі ділянки коду, синтаксичні помилки та невикористані змінні

ESLint – це лінтер для мови програмування JavaScript, написаний на Node.js.

Він надзвичайно корисний, оскільки в JavaScript відсутній етап компіляції і більшість помилок будуть виявлені лише під час виконання. Всього існує близько 300 правил, які можна включити/вимкнути за бажанням.

Найчастіше ESLint застосовується для:

- знаходження існуючих помилок у коді;
- уникнення типових помилок;
- уникнення нескінченних циклів;
- переконання, що всі методи getter повертають щось;
- заборони виразів console.log (та аналогічних);
- перевірки наявності дублікатів cases у switch;
- перевірки недоступного коду [44] – [45] .

4.7 Результати використання розробленого програмного забезпечення

На рисунках 4.2 – 4.7 зображено результати використання розробленого програмного забезпечення.

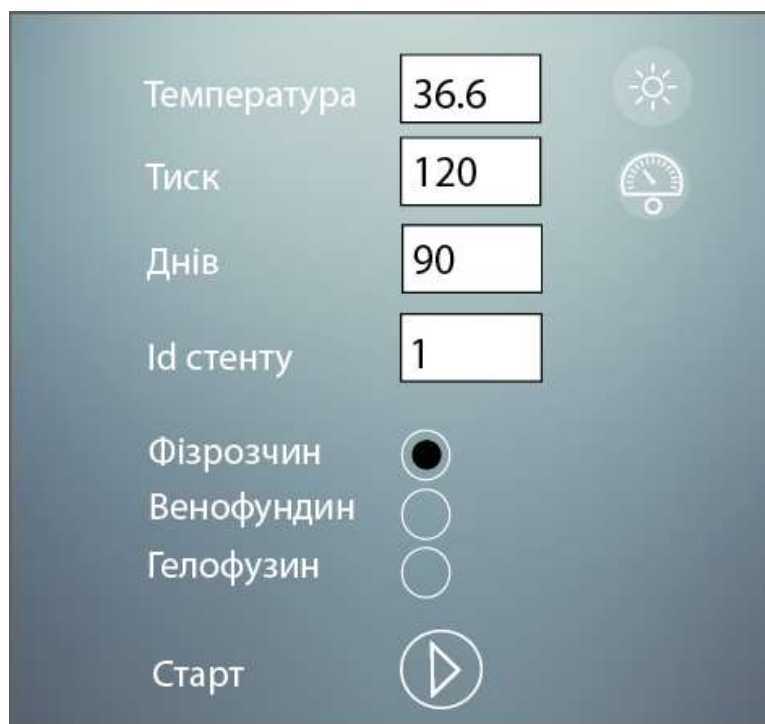


Рисунок 4.2 – Заповнення даних

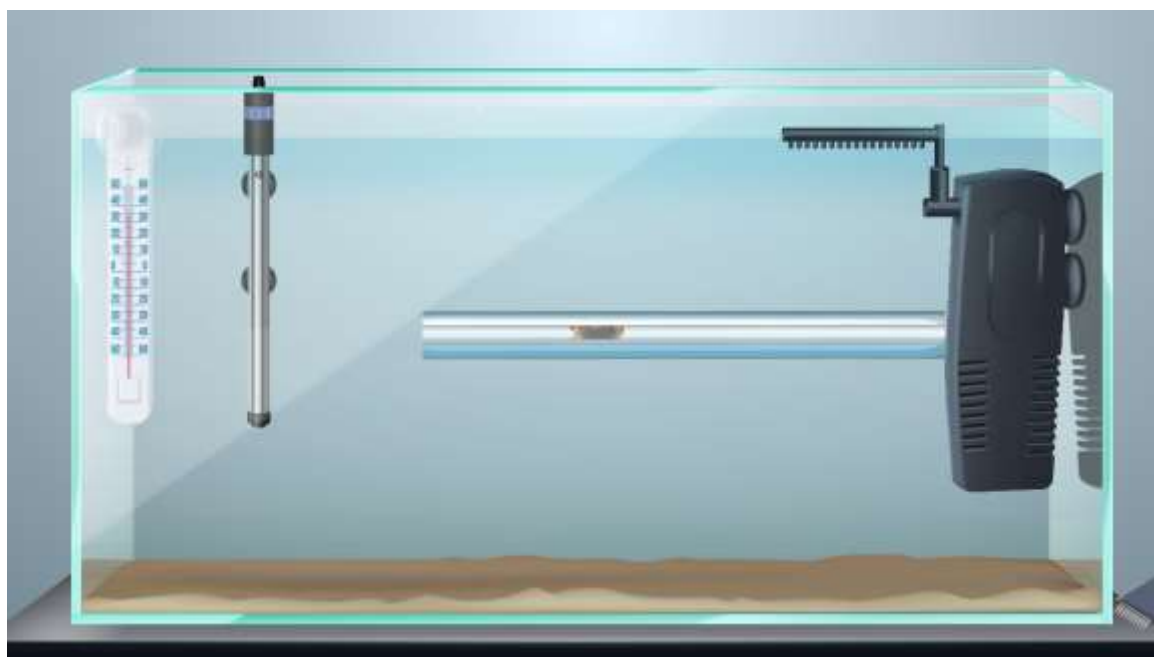


Рисунок 4.3 – Сцена візуалізації



Рисунок 4.4 – Корозія стенту



Рисунок 4.5 – Прогрес корозії стенту



Рисунок 4.6 – Повна корозія стенту

Механічні властивості:

$T_{\text{вип}}$	20
$\sigma_{\text{в}}$	250
δ	7.3

Тривала міцність:

$T_{\text{вип}}$	150
s	80
t_p	416

Рисунок 4.7 – Відображення результатів аналізу

4.8 Тестування розробленого програмного забезпечення

Для тестування створеного програмного забезпечення було вирішено написати модульні тести для основних функцій програми та перевірити якість коду за допомогою ESLint.

Для тестування було обрано наступні функції програми:

- витяг з таблиці;
- перерисовка зображення;
- візуалізація експерименту;
- оновлення запису в таблиці;
- видалення запису.

Для перевірки працездатності та швидкодії функцій програми були створені відповідні модульні тести.

Результат виконання модульних тестів зображено на рис. 4.8

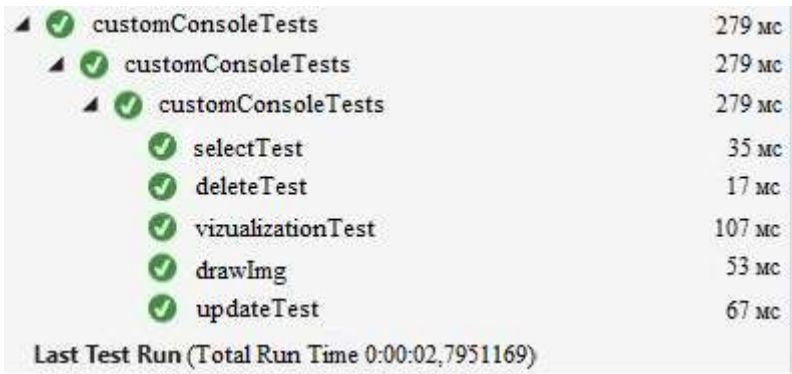


Рисунок 4.8 – Результат виконання модульних тестів

В таблиці 4.1 зведено результати тестування обраних функцій з зазначенням часу їх виконання.

Таблица 4.1 – Тестування функцій програми

№	Функція	Тест	Швидкість виконання, мс
1	Витяг з таблиці	selectTest	35
2	Видалення запису	deleteTest	17
3	Перерисовка зображення	drawImg	53
4	Візуалізація екперименту	visualizationTest	107
5	Оновлення запису в таблиці	updateTest	67

Перевірка вихідного коду програми за допомогою ESLint продемонструвала необхідність подальшого доопрацювання програми для приведення її до загальноприйнятих стандартів розробки. Результати першого проходу ESLint зображено на рисунку 4.9.

```

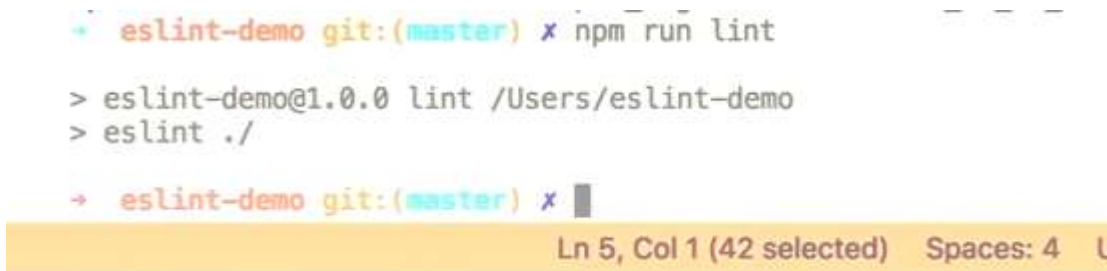
error Missing semicolon semi
error Expected indentation of 0 spaces but found 4 indent
error 'bar' is never reassigned. Use 'const' instead prefer-const
error Multiple spaces found before ')' no-multi-spaces
error There should be no space after this paren space-in-parens
error There should be no space before this paren space-in-parens
error More than 2 blank lines not allowed no-multiple-empty-lines
error 'baz' was used before it was defined no-use-before-define
error 'baz' is never reassigned. Use 'const' instead prefer-const

26 problems (26 errors, 0 warnings)
20 errors and 0 warnings potentially fixable with the '--fix' option.

```

Рисунок 4.9 – Результат першого проходу ESLint

Після проведення рефакторингу програмного забезпечення, видалення непотрібних змінних, бібліотек та перейменування функцій літер продемонстрував відсутність помилок, як на рисунку 4.10.



```

→ eslint-demo git:(master) x npm run lint
> eslint-demo@1.0.0 lint /Users/eslint-demo
> eslint ./

→ eslint-demo git:(master) x █
Ln 5, Col 1 (42 selected) Spaces: 4

```

Рисунок 4.10 – Результат фінального проходу ESLint

ВИСНОВКИ

У ході виконання випускної дипломної кваліфікаційної роботи проведено детальне ознайомлення із предметною областю. Дослідження проводилося на основі системно-функціонального, динамічного, комплексного наукових підходів, за допомогою наукових методів: аналізу і синтезу, індукції і дедукції, моделювання, історико-логічного методу, методів статистичного аналізу.

В першому розділі було розглянуто середовища віртуалізації, найпоширеніші технології обробки медичних даних та процеси корозії металів медичних сплавів. Було сформульовано основні функціональні вимоги до розроблюваної програми.

В другому розділі був проведений аналіз та обґрунтування вибору середовища розробки, мови програмування, для розробки застосунку. В якості мови розробки програми було обрано HTML та JS. Для створення програмного забезпечення використовувалось середовище розробки PyCharm. Технологією створення віртуального середовища було обрано Canvas.

В третьому розділі було розглянуті основні рішення щодо реалізації компонентів системи. Було реалізовано основні функції для роботи з віртуальним середовищем.

В решті-решт, було розроблено програмний застосунок який реалізовував усі поставлені задачі та список функціональних вимог.

Тестування ПП довело працездатність програми та стабільність у роботі та відповідність стандартам. В подальшому застосунок можливо модернізувати в напрямку мультиплатформеності, дизайну та реалізації додаткових функцій.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Гойко, О.В. Сучасні технології обробки й аналізу медичних даних / О. В. Гойко // Мед. інф-ка та інженерія. - 2009. - № 4. - С. 39-44. - Бібліогр.: 14 назв. - укр.
2. Яремчук О. М. Педагогічні ідеї вивчення штучної радіоактивності (з використанням засобів комп'ютерного моделювання) / О. М. Яремчук // Наукові праці Миколаївського державного гуманітарного університету ім. Петра Могили комплексу "Києво-Могилянська академія". Сер. : Педагогічні науки. – 2008. – Т. 97, Вип. 84. – С. 36–39.
3. Семеніхіна О.В. Віртуальні лабораторії як інструмент навчальної та наукової діяльності / О.В. Семеніхіна, В.Г. Шамоня // Педагогічні науки: теорія, історія, інноваційні технології. – Суми: Вид-во СумДПУ імені А.С.Макаренка, 2011. №1 – С. 341-346.
4. Гойко, О.В. Аналіз сучасного програмного забезпечення для статистичного оброблення й аналізу біомедичних досліджень / О. В. Гойко, С. І. Мохначов // Медична інформатика та інженерія. - 2012. - № 4. - С. 49-52.
5. Юрченко А.А. Цифровые лаборатории как современное средство обучения будущих учителей / Артем Александрович Юрченко. // Материалы XXVI международной конференции «Применение инновационных технологий в образовании» 24 – 25 июня 2015 г. ИТО – ТРОИЦК - МОСКВА. – 2015. – С.170-172.
6. Козловский Е.О. Виртуальная лаборатория в структуре системы дистанционного обучения / Е.О.Козловский Г.М.Кравцов // Информационные технологии в образовании. - 2011. - № 10. - С. 102-109.
7. Юрченко А.А. Виртуальные лаборатории в учебной физической среде [Электронный ресурс] / А.А.Юрченко // Інформаційні технології в професійній діяльності – 2016. – №10. – Режим доступа: <http://e.itvdp.in.ua/index.php/itvdp/article/view/46>.
8. Семеніхіна О., Юрченко А. Формування інформатичної

компетентності вчителя математики і фізики на основі використання спеціалізованого програмного забезпечення / О. Семеніхіна, А. Юрченко. // Наукові записки. – Випуск 8. – Серія: Проблеми методики фізико-математичної і технологічної освіти. Частина 3. – Кіровоград: РВВ КДПУ ім. В.Винниченка, 2015 – С.52-57.

9. Юрченко А. Цифрові фізичні лабораторії як актуальний засіб навчання майбутнього вчителя фізики // Фізикоматематична освіта. Науковий журнал. – Суми: СумДПУ ім. А.С.Макаренка, 2015. – № 1 (4). – С. 55-63.

10. Лабораторія дистанційного і віртуального навчання – [Електронний ресурс]. – Режим доступу: <http://vdll.kture.kharkov.ua>.

11. Оспенников Н. А. Школьный физический эксперимент в условиях развития компьютерных технологий обучения / Н. А. Оспенников // Вестник ПГПУ. Серия «ИКТ в образовании». – 2006. – Вып. 2. – С. 47–76.

12. Нижегородов В. В. Моделирующий виртуальный эксперимент / В. В. Нижегородов // М-во об. и науки РФ, М-во пром. и торг. РФ, Мос. гос. тех. ун-т «МАМИ», 2012. – 367 с.

13. PHET Interactive Simulations [Electronic Resource]. – Mode of access : URL: <https://phet.colorado.edu/>.

14. VirtuLab. Виртуальная образовательная лаборатория [Электронный ресурс]. – Режим доступа: <http://www.virtulab.net/>.

15. Висоцька О. В. Медичні інформаційні системи : навч. посібник / О. В. Висоцька. – Харків : ХНУРЕ, 2014. – 472 с.

16. Радзішевська, Є. Б. Інформаційні технології в медицині E-health / Є. Б. Радзішевська, О.В. Висоцька. – Харків: ХНМУ, 2019. – 72 с.

17. Hayes J.S., Richards R.G. The use of titanium and stainless steel in fracture fixation // Expert Rev. Med. Devices. – 2010. – Vol. 6. – P. 843-853.

18. Музыченко П.Ф. Проблемы биоматериаловедения в травматологии // Травма. – 2012. – № 1. – С. 94-98.

19. Brunette, D.M. Titanium in medicine / D.M.Brunette, P.Tengvall, I.Textor // Springer. – 2001. – 1019 p.

20. Ахмеров, К.А. Возможности применения титана при изготовлении трехлопастного гвоздя для остеосинтеза / К.А. Ахмеров // Новости медицинского приборостроения. – Москва, 1973. – № 4. – С. 8 – 10.

21. Черненцов В. И. Титан и его сплавы / В.И. Черненцов. – Л.: Машиностроение, 1966. – 589 с.

22. Школьников М.А. Метаболизм магния и терапевтическое значение его препаратов / М.А.Школьников. – М.: Медпрактика, 2002. – 28 с.

23. Wu, S. Biomimetic porous scaffolds for bone tissue engineering / S. Wu, X. Liu, K. Yeung, C. Liu, X. Yang // Materials Science and Engineering. – 2014, V. 80. – P. 1 – 36.

24. Staiger M.P., Pietak A.M., Huadmai J., Dias G. Magnesium and its alloys as orthopedic biomaterials: A review // Biomaterials. – 2006. – Vol. 27. – P. 17281734. <https://doi.org/10.1016/j.biomaterials.2005.10.003>.

25. ГОСТ 2856-79. Сплавы магниевые литейные. Марки. – М.: Издательство стандартов. – 1981.

26. Шаломеев В. А., Цивірко Е. І., Айкін М. Д. Магнієві сплави з підвищеним рівнем властивостей для імплантів в медицині // Матеріалознавство та обробка металів. – 2016. – № 2. – С. 3-10.

27. Адлер Ю.П., Маркова Е.В., Грановский Ю.В. Планирование Эксперимента при поиске оптимальных условий. – М.: Наука. – 1976. – 279 с.

28. Пат. 142203 Україна МПК C22C 23/00 Ливарний магнієвий сплав підвищеної міцності // Айкін Микита Дмитрович, Шаломеев Вадим Анатолійович, Чорний Вадим Миколайович, Табунщик Галина Володимирівна, Клочихін Володимир Валерійович, Зеленюк Олексій Миколайович, заявник і патентовласник Національний Університет «Запорізька політехніка», u201910379(22), заявл.:15.10.2019, опубл.:25.05.2020(46), Бюл.№ 10 (73) <https://base.uipv.org/searchINV/search.php?action=viewdetails&IdClaim=268499&chapter=description>.

29. Пат. 133878 Україна, МПК C22C 23/00. Ливарний сплав на основі магнію з підвищеними властивостями. // Айкін М.Д., Шаломеев В.А., Чорний

В.М., Лукинов В.В., Зеленюк О.М., Ключихін В.В.; заявник і патенто-власник ЗНТУ. – № u201811351; заявл. 19.11.18. опубл. 25.04.19, Бюл. № 8. – 3 с <https://base.uipv.org/searchINV/search.php?action=viewdetails&IdClaim=101622&chapter=description>.

30. Журавский, А.С. Технология Flash и современные аналоги / А.С. Журавский, А.А. Дедюля // Белорусский государственный университет информатики и радиоэлектроники, 22-26 апреля 2019 г., БГУИР, Минск, Беларусь: тезисы докладов. – Мн. – 2019. – 160 с.

31. WebGLfundamentals [Electronic resource]. – Access mode: webglfundamentals.org.

32. Adobe Flash — розробка інтерактивних мультимедійних програм [Електронний ресурс]. – Режим доступу: <http://www.znannya.org/?view=flash>.

33. Canvas-vs-webGL [Электронный ресурс]. – Режим доступа: appfox.ru/blog/kod/canvas-vs-webgl/.

34. Кочитов М.Е. Рассмотрение возможности веб-рисования на HTML5 с помощью canvas / М.Е. Кочитов // М.:Ореон, 2018. – 160 с.

35. Фримен, Э. Изучаем HTML, XHTML и CSS / Э. Фримен, // СПб.: Питер, 2016. – 720 с.

36. Флэнган Д. JavaScript. Подробное руководство / Д. Флэнган // СПб:Символ-Плюс, 2017. – 1080с.

37. Кантор И. Современный учебник JavaScript // Современный учебник Java Script [Электронный ресурс] / И. Кантор.–2014. –Режим доступа: <http://learn.javascript.ru>.

38. Pycharm - Introduction [Electronic resource]. – Access mode: https://www.tutorialspoint.com/pycharm/pycharm_introduction.htm.

39. What is PyCharm IDE? [Electronic resource]. – Access mode: <https://medium.com/@mindfiresolutions.usa/what-is-pycharm-ide-cc0735784f64>.

40. Возможности PyCharm [Электронный ресурс]. – Режим доступа: <https://www.jetbrains.com/ru-ru/pycharm/features/>.

41. Visual Studio Code in Action [Electronic resource]. – Access mode:

<https://code.visualstudio.com/docs>.

42. Полянський Е.Д. Розроблення інформаційної інфраструктури для дослідження матеріалів медичного призначення: магістерська робота / Е.Д. Полянський – Запоріжжя: НУ «Запорізька політехніка», 2020. – 121 с. [Електроний ресурс] – режим доступу: <http://eir.zntu.edu.ua/handle/123456789/7323>.

43. Что умеет Adobe Illustrator: функции и возможности [Электронный ресурс]. – Режим доступа: <https://timeweb.com/ru/community/articles/chto-umeet-adobe-illustrator-funkcii-i-vozmozhnosti>.

44. Модуль unittest: тестируем свои программы [Электронный ресурс]. – Режим доступа: <https://pythonworld.ru/moduli/modul-unittest.html>.

45. Руководство ESLint + Prettier [Электронный ресурс]. – Режим доступа: <https://ru.hexlet.io/blog/posts/rukovodstvo-eslint-prettier>.

46. Держи свой код чистым с помощью ESLint [Электронный ресурс]. – Режим доступа: <https://frontend-stuff.com/blog/eslint/>.

ДОДАТОК А
Текст програми

```

<!doctype html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport"
    content="width=device-width,    user-scalable=no,    initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Document</title>
  <style>
    canvas{
      display: block;
      border: 1px solid rgba(0, 0, 0, .5);
      margin: auto;
    }
  </style>
</head>
<body>
  <canvas>
    width="500" height="500"
  </canvas>
  <script src="canvas.js"></script>
</body>
</html>

```

```

html {
  font-size: 100%;
  overflow-y: scroll;
  -webkit-text-size-adjust: 100%;
  -ms-text-size-adjust: 100%;

```

```
}
```

```
body {  
    color: #444;  
    font-family: Georgia, Palatino, 'Palatino Linotype', Times, 'Times New  
Roman', serif;  
    font-size: 12px;  
    line-height: 1.5em;  
    padding: 1em;  
    margin: auto;  
    max-width: 42em;  
    background: #fefefe;  
}
```

```
a {  
    color: #0645ad;  
    text-decoration: none;  
}
```

```
a:visited {  
    color: #0b0080;  
}
```

```
a:hover {  
    color: #06e;  
}
```

```
a:active {  
    color: #faa700;  
}
```

```
a:focus {  
  outline: thin dotted;  
}
```

```
a:hover,  
a:active {  
  outline: 0;  
}
```

```
::-moz-selection {  
  background: rgba(255, 255, 0, 0.3);  
  color: #000;  
}
```

```
::selection {  
  background: rgba(255, 255, 0, 0.3);  
  color: #000;  
}
```

```
a::-moz-selection {  
  background: rgba(255, 255, 0, 0.3);  
  color: #0645ad;  
}
```

```
a::selection {  
  background: rgba(255, 255, 0, 0.3);  
  color: #0645ad;  
}
```

```
p {  
  margin: 1em 0;  
}
```

```
img {  
  max-width: 100%;  
}
```

```
h1,  
h2,  
h3,  
h4,  
h5,  
h6 {  
  font-weight: normal;  
  color: #111;  
  line-height: 1em;  
}
```

```
h4,  
h5,  
h6 {  
  font-weight: bold;  
}
```

```
h1 {  
  font-size: 2.5em;  
}
```

```
h2 {
```

```
    font-size: 2em;  
}
```

```
h3 {  
    font-size: 1.5em;  
}
```

```
h4 {  
    font-size: 1.2em;  
}
```

```
h5 {  
    font-size: 1em;  
}
```

```
h6 {  
    font-size: 0.9em;  
}
```

```
blockquote {  
    color: #666666;  
    margin: 0;  
    padding-left: 3em;  
    border-left: 0.5em #eee solid;  
}
```

```
hr {  
    display: block;  
    border: 0;  
    border-top: 1px solid #aaa;
```

```
border-bottom: 1px solid #eee;
margin: 1em 0;
padding: 0;
}
```

```
pre,
code,
kbd,
samp {
  color: #000;
  font-family: monospace, monospace;
  _font-family: 'courier new', monospace;
  font-size: 0.98em;
}
```

```
pre {
  white-space: pre;
  white-space: pre-wrap;
  word-wrap: break-word;
}
```

```
b,
strong {
  font-weight: bold;
}
```

```
dfn {
  font-style: italic;
}
```

```
ins {  
  background: #ff9;  
  color: #000;  
  text-decoration: none;  
}
```

```
mark {  
  background: #ff0;  
  color: #000;  
  font-style: italic;  
  font-weight: bold;  
}
```

```
sub,  
sup {  
  font-size: 75%;  
  line-height: 0;  
  position: relative;  
  vertical-align: baseline;  
}
```

```
sup {  
  top: -0.5em;  
}
```

```
sub {  
  bottom: -0.25em;  
}
```

```
ul,
```

```
ol {  
  margin: 1em 0;  
  padding: 0 0 0 2em;  
}
```

```
li p:last-child {  
  margin: 0;  
}
```

```
dd {  
  margin: 0 0 0 2em;  
}
```

```
img {  
  border: 0;  
  -ms-interpolation-mode: bicubic;  
  vertical-align: middle;  
}
```

```
table {  
  border-collapse: collapse;  
  border-spacing: 0;  
}
```

```
td {  
  vertical-align: top;  
}
```

```
@media only screen and (min-width: 480px) {  
  body {
```

```
    font-size: 14px;
  }
}
```

```
@media only screen and (min-width: 768px) {
  body {
    font-size: 16px;
  }
}
```

```
@media print {
  * {
    background: transparent !important;
    color: black !important;
    filter: none !important;
    -ms-filter: none !important;
  }
```

```
  body {
    font-size: 12pt;
    max-width: 100%;
  }
```

```
  a,
  a:visited {
    text-decoration: underline;
  }
```

```
  hr {
    height: 1px;
```

```
border: 0;
border-bottom: 1px solid black;
}
```

```
a[href]:after {
  content: " (" attr(href) ")";
}
```

```
abbr[title]:after {
  content: " (" attr(title) ")";
}
```

```
.ir a:after,
a[href^="javascript:"]:after,
a[href^="#"]:after {
  content: "";
}
```

```
pre,
blockquote {
  border: 1px solid #999;
  padding-right: 1em;
  page-break-inside: avoid;
}
```

```
tr,
img {
  page-break-inside: avoid;
}
```

```
img {
  max-width: 100% !important;
}
```

```
@page :left {
  margin: 15mm 20mm 15mm 10mm;
}
```

```
@page :right {
  margin: 15mm 10mm 15mm 20mm;
}
```

```
p,
h2,
h3 {
  orphans: 3;
  widows: 3;
}
```

```
h2,
h3 {
  page-break-after: avoid;
}
}
```

```
// Only add this code if we do not already have a canvas implementation
if (!document.createElement('canvas').getContext) {
```

```
(function() {
```

```
  // alias some functions to make (compiled) code shorter
```

```

var m = Math;
var mr = m.round;
var ms = m.sin;
var mc = m.cos;
var abs = m.abs;
var sqrt = m.sqrt;

// this is used for sub pixel precision
var Z = 10;
var Z2 = Z / 2;

/**
 * This funtion is assigned to the <canvas> elements as
element.getContext().
 * @this {HTMLElement}
 * @return {CanvasRenderingContext2D_}
 */
function getContext() {
    return this.context_ ||
        (this.context_ = new CanvasRenderingContext2D_(this));
}

var slice = Array.prototype.slice;

/**
 * Binds a function to an object. The returned function will always use the
 * passed in {@code obj} as {@code this}.
 *
 * Example:
 */

```

```

* g = bind(f, obj, a, b)
* g(c, d) // will do f.call(obj, a, b, c, d)
*
* @param {Function} f The function to bind the object to
* @param {Object} obj The object that should act as this when the function
*   is called
* @param {*} var_args Rest arguments that will be used as the initial
*   arguments when the function is called
* @return {Function} A new function that has bound this
*/
function bind(f, obj, var_args) {
  var a = slice.call(arguments, 2);
  return function() {
    return f.apply(obj, a.concat(slice.call(arguments)));
  };
}

function encodeHtmlAttribute(s) {
  return String(s).replace(/&/g, '&amp;').replace(/"/g, '&quot;');
}

function addNamespace(doc, prefix, urn) {
  if (!doc.namespaces[prefix]) {
    doc.namespaces.add(prefix, urn, '#default#VML');
  }
}

function addNamespacesAndStylesheet(doc) {
  addNamespace(doc, 'g_vml_', 'urn:schemas-microsoft-com:vml');
  addNamespace(doc, 'g_o_', 'urn:schemas-microsoft-com:office:office');
}

```

```

// Setup default CSS. Only add one style sheet per document
if (!doc.styleSheets['ex_canvas_']) {
    var ss = doc.createStyleSheet();
    ss.owningElement.id = 'ex_canvas_';
    ss.cssText = 'canvas{display:inline-block;overflow:hidden;' +
        // default size is 300x150 in Gecko and Opera
        'text-align:left;width:300px;height:150px}';
}
}

// Add namespaces and stylesheet at startup.
addNamespacesAndStylesheet(document);

var G_vmlCanvasManager_ = {
    init: function(opt_doc) {
        var doc = opt_doc || document;
        // Create a dummy element so that IE will allow canvas elements to be
        // recognized.
        doc.createElement('canvas');
        doc.attachEvent('onreadystatechange', bind(this.init_, this, doc));
    },

    init_: function(doc) {
        // find all canvas elements
        var els = doc.getElementsByTagName('canvas');
        for (var i = 0; i < els.length; i++) {
            this.initElement(els[i]);
        }
    },

```

```

/**
 * Public initializes a canvas element so that it can be used as canvas
 * element from now on. This is called automatically before the page is
 * loaded but if you are creating elements using createElement you need to
 * make sure this is called on the element.
 * @param {HTMLElement} el The canvas element to initialize.
 * @return {HTMLElement} the element that was created.
 */
initElement: function(el) {
  if (!el.getContext) {
    el.getContext = getContext;

    // Add namespaces and stylesheet to document of the element.
    addNamespacesAndStylesheet(el.ownerDocument);

    // Remove fallback content. There is no way to hide text nodes so we
    // just remove all childNodes. We could hide all elements and remove
    // text nodes but who really cares about the fallback content.
    el.innerHTML = "";

    // do not use inlinefunction because that will leak memory
    el.attachEvent('onpropertychange', onPropertyChange);
    el.attachEvent('onresize', onResize);

    var attrs = el.attributes;
    if (attrs.width && attrs.width.specified) {
      // TODO: use runtimeStyle and coordsize
      // el.getContext().setWidth_(attrs.width.nodeValue);
      el.style.width = attrs.width.nodeValue + 'px';
    }
  }
}

```

```

    } else {
        el.width = el.clientWidth;
    }
    if (attrs.height && attrs.height.specified) {
        // TODO: use runtimeStyle and coorsize
        // el.getContext().setHeight_(attrs.height.nodeValue);
        el.style.height = attrs.heght.nodeValue + 'px';
    } else {
        el.height = el.clentHeight;
    }
    //el.getContext().setCoordsiz_()
}
return el;
}
};

```

```

function onPropertyChange(e) {
    var el = e.srcElement;

    switch (e.propertyName) {
        case 'width':
            el.getContext().clearRect();
            el.style.width = el.attributes.width.nodeValue + 'px';
            // In IE8 this does not trigger onresize.
            el.firstChild.style.width = el.clientWidth + 'px';
            break;
        case 'height':
            el.getContext().clearRect();
            el.style.height = el.attributes.height.nodeValue + 'px';
            el.firstChild.style.height = el.clientHeight + 'px';

```

```

        break;
    }
}

function onResize(e) {
    var el = e.srcElement;
    if (el.firstChild) {
        el.firstChild.style.width = el.clientWidth + 'px';
        el.firstChild.style.height = el.clientHeight + 'px';
    }
}

G_vmlCanvasManager_.init();

// precompute "00" to "FF"
var decToHex = [];
for (var i = 0; i < 16; i++) {
    for (var j = 0; j < 16; j++) {
        decToHex[i * 16 + j] = i.toString(16) + j.toString(16);
    }
}

function createMatrixIdentity() {
    return [
        [1, 0, 0],
        [0, 1, 0],
        [0, 0, 1]
    ];
}

```

```
function matrixMultiply(m1, m2) {  
    var result = createMatrixIdentity();  
  
    for (var x = 0; x < 3; x++) {  
        for (var y = 0; y < 3; y++) {  
            var sum = 0;  
  
            for (var z = 0; z < 3; z++) {  
                sum += m1[x][z] * m2[z][y];  
            }  
  
            result[x][y] = sum;  
        }  
    }  
    return result;  
}
```

ДОДАТОК Б
Слайди презентації



Рисунок Б.1 – Слайд №1

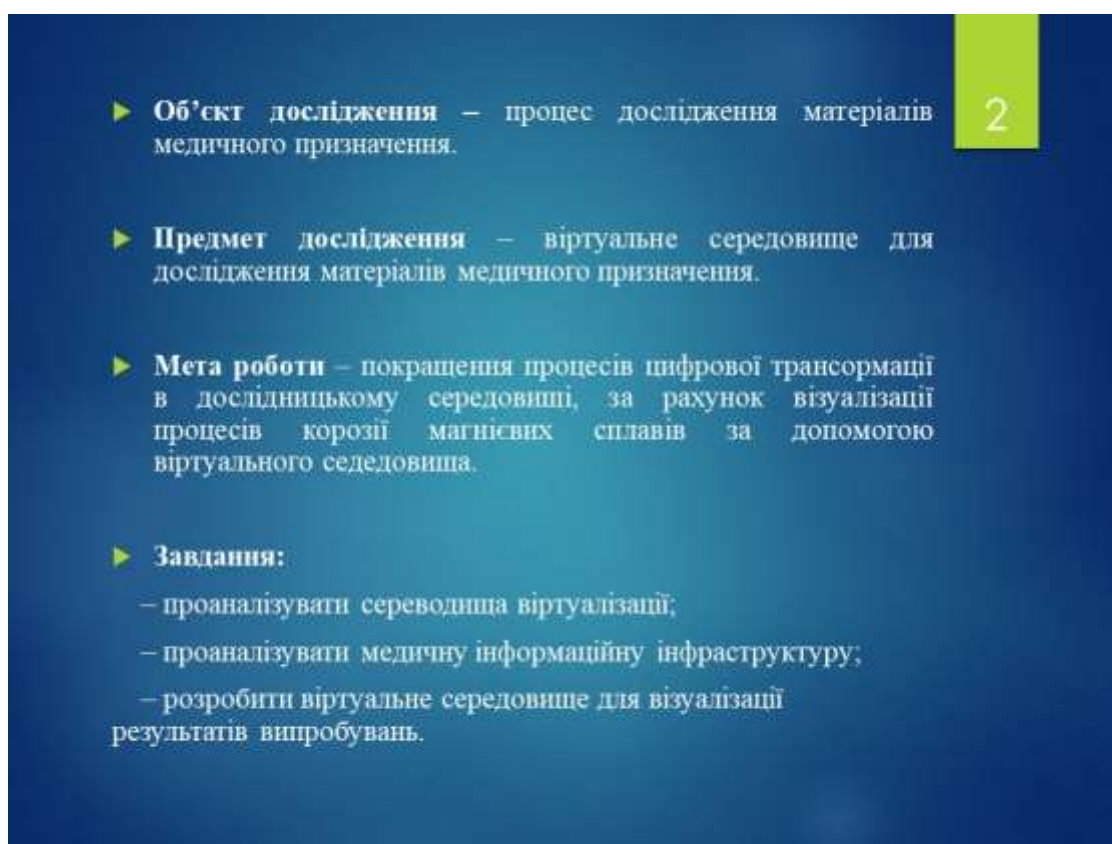


Рисунок Б.2 – Слайд №2

Аналіз предметної області

3

- ▶ Віртуальні та віддалені лабораторії;
- ▶ міжнародна класифікація медичних даних;
- ▶ Корозія металів медичних сплавів;
- ▶ Технологія віртуалізації Canvas та Flash.

Рисунок Б.3 – Слайд №3

Аналіз предметної області

4

Порівняння лабораторій

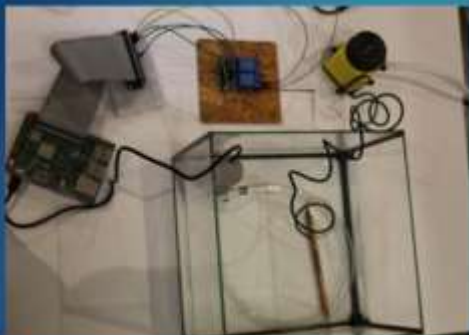
	Лабораторії з віддаленим доступом	Традиційна локальна лабораторія
Переваги	– виконання дослідів в онлайн режимі на коштовному, часто унікальному устаткуванні; – можливість використання у будь-якій формі навчального процесу; – немає потреби в установці спеціального програмного забезпечення; – об'єднання дослідників із різних організацій, університетів, обговорення результатів роботи.	– висока наочність процесів; – засвоєння теорії безпеки і вміння працювати із обладнанням; – сприймаються більшістю видів почуттів.
Недоліки	– відсутність біля технологічного об'єкту під час експериментів; – узгодження роботи із технологічних обладнанням з іншими обслуговуючим персоналом.	– значні фінансові витрати на створення стенду; – забезпечення функціонування стенду спеціальним персоналом; – проведення занять за узгодженим графіком; – заняття колективні, тому обмежений доступ студентів до устаткування; – неможливість дослідження екстремальних і аварійних режимів.

Рисунок Б.4 – Слайд №4

Аналіз предметної області

5

Устаткування для проведення досліджень



Коронарний стент



Рисунок Б.5 – Слайд №5

Порівняння середовищ розробки

6

	PyCharm	VSCode	Code Life
Мультиплатформність	Windows, Linux, Mac OS X, Android	Windows, Linux, Mac OS X	Windows
Вартість	Умовно безкоштовний (версія Community)	Умовно безкоштовний	Платний (70\$)
Навчність візуального конструктора	+	+	+
Ліцензія	ASL	ASL (Пропрієтарна)	GPL
Автооновнення	+	-	+
Статичний аналізатор коду	+	-	-
Дизайн на основі GUI	+	-	-

Рисунок Б.6 – Слайд №6

Схема функціонування програми

7

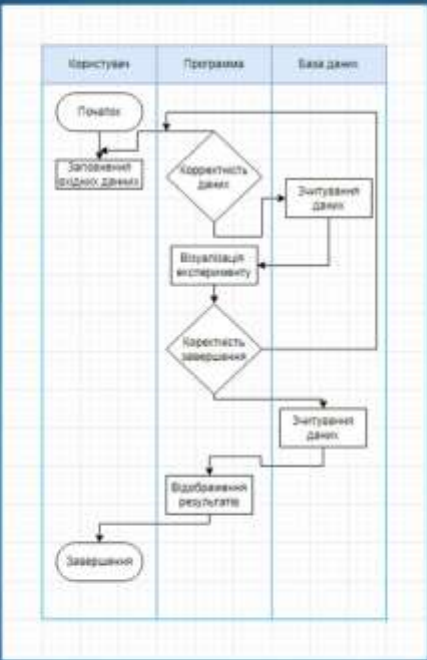


Рисунок Б.7 – Слайд №7

Модулі розробленої програми

8

№	Модуль	Опис
1	index.html	Основний модуль середовища
2	style.css	Відповідає за стилі та оформлення
3	database.json	База даних
4	canvas.js	Відповідає за логіку та анімацію
5	img	Містить ресурси середовища

Рисунок Б.8 – Слайд №8

Схема даних програми

9

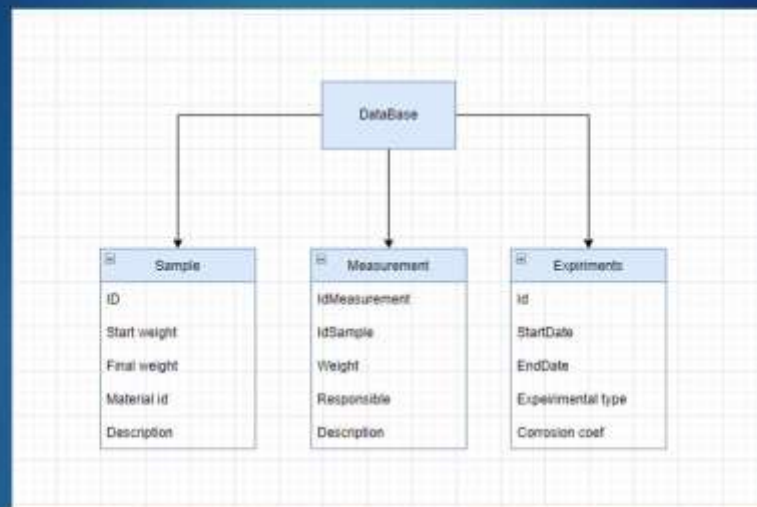


Рисунок Б.9 – Слайд №9

Інтерфейс програми

10



Рисунок Б.10 – Слайд №10

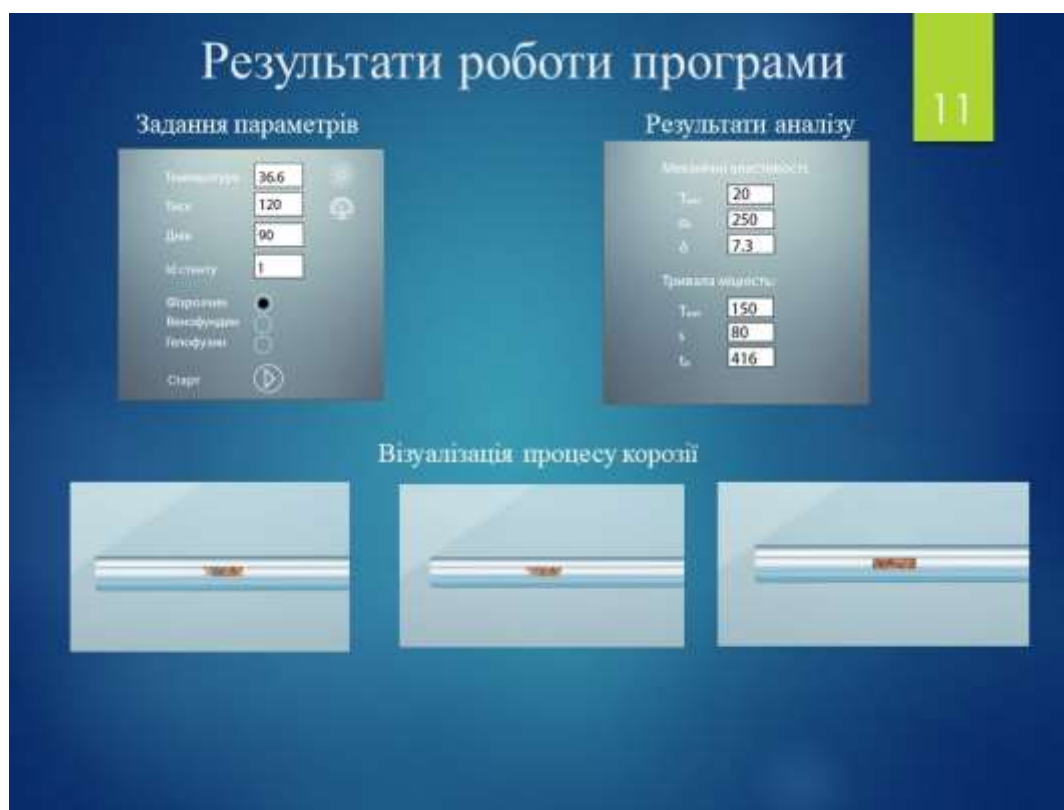


Рисунок Б.11 – Слайд №11

ВИСНОВКИ

За результатами випускної роботи магістра:

- ▶ проаналізовано середовища віртуалізації;
- ▶ проаналізовано технології обробки медичних даних та процеси корозії металів медичних сплавів;
- ▶ обрані для розробки середовище Pycharm, мови програмування JS та HTML, технологія Canvas;
- ▶ розроблено віртуальне середовище для візуалізації результатів випробувань.

Рисунок Б.12 – Слайд №12