Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА ЗАСТОСУНКУ ДЛЯ АНАЛІЗУ ЕФЕКТИВНОСТІ ПІДГОТОВКИ СПОРТСМЕНІВ
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-19-1

Гончарова О.В.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Кобилін О.А.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Гончаровій Оксані Володимирівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка застосунку для аналізу ефективності підготовки спортсменів

затверджена наказом університету від 15 травня 2023 року № 474 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 22 травня 2023 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, бібліотека JavaFX.

4. Перелік питань, що потрібно опрацювати в роботі

1. Аналітичний огляд ПО та інструменатрію для розробки десктопних застосунків з аналізу ефективності підготовки спортсменів.
2. Моделювання застосунку для аналізу ефективності підготовки спортсменів.
3. Програмна реалізація застосунку.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми аналізу ефективності підготовки спортсменів, постановка задачі, тестові зображення.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Творошенко І.С.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	10.04.2023	
2	Аналіз завдання, підбір літератури	11.04.23-17.04.23	
3	Аналіз літератури з досліджуваної проблеми	18.04.23-20.04.23	
4	Огляд ПО та інструментарію для розробки	21.04.23-30.04.23	
5	Моделювання застосунку	01.05.23-14.05.23	
6	Програмна реалізація	15.05.23-23.05.23	
7	Оформлення пояснювальної записки	24.05.23-26.05.23	
8	Перевірка на плагіат	27.05.23	
9	Рецензування	28.05.23	
10	Підготовка презентації та доповіді	29.05.23-30.05.23	
11	Занесення роботи в електронний архів	31.05.23	
12	Попередній захист кваліфікаційної роботи	31.05.23	

Дата видачі завдання 10 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ _____доц. Кобилін О.А._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 71 с., 3 табл., 31 рис., 1 дод., 38 джерел.

ЕФЕКТИВНІСТЬ ПІДГОТОВКИ СПОРТСМЕНІВ, JAVA FX, АНАЛІЗ ВІДСОТКОВОГО ВІДНОШЕННЯ ПІДГОТОВКИ, СТАТИСТИЧНИЙ АНАЛІЗ РЕЗУЛЬТАТІВ.

Об’єктом роботи є аналіз ефективності підготовки спортсменів.

Метою роботи є розробка застосунку для швидкого та точного аналізу відсоткового відношення підготовки спортсмена до змагань з урахуванням певних параметрів для оцінки його фізичного стану та прогресу в обраній галузі спорту.

У ході роботи проводиться аналітичний огляд існуючих платформ для аналізу ефективності спортсменів, а також аналіз засобів для створення застосунків мовою Java. Змодельована математична модель застосунку, визначено взаємозв’язок між параметрами і прогресом спортивних досягнень.

У результаті роботи здійснена програмна реалізація десктоп застосунку для відстеження прогресу тренувань спортсменів за допомогою фреймворку JavaFX.

EFFICIENCY OF TRAINING FOR ATHLETES, JAVA FX, ANALYSIS OF VID-CELL VISION TRAINING, STATISTICAL ANALYSIS OF RESULTS.

The object of the work is the analysis of the effectiveness of training athletes.

The purpose of the work is to develop an application for quick and accurate analysis of the percentage ratio of an athlete's preparation for competition, considering certain parameters for assessing his physical condition and progress in the chosen field of sports.

The study provides an analytical review of existing platforms for analyzing the performance of athletes is carried out, as well as an analysis of tools for creating applications in the Java language. The mathematical model of the application was simulated, the relationship between the parameters and the progress of sports achievements was determined.

As a result of the work, a software implementation of a desktop application for tracking the progress of athletes' training using the JavaFX framework was carried out.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 7
Вступ 8
1 Аналітичний огляд ПО та інструменатрію для розробки десктопних застосунків з аналізу ефективності підготовки спортсменів 9
1.1 Огляд існуючих застосунків для аналізу ефективності спортсменів 9
1.1.1 Sportlyzer 9
1.1.2 TrainingPeaks 11
1.1.3 AthleteMonitoring 11
1.1.4 Strava 12
1.1.5 Timing Events 12
1.1.6 Результати порівняння застосунків 13
1.2 Аналіз засобів для створення десктоп застосунків мовою Java 14
1.2.1 AWT 14
1.2.2 Swing 15
1.2.3 SWT 16
1.2.4 JavaFX 18
1.3 Постановка задачі 19
2 Моделювання застосунку для аналізу ефективності підготовки спортсменів 21
2.1 Визначення параметрів для аналізу ефективності підготовки 21
2.1.1 Визначення коефіцієнтів для спільних параметрів до кожного виду спорту 21
2.1.2 Специфічні параметри для окремих видів спорту 25
2.2 Моделювання алгоритму вирішення задачі 26

2.2.1 Збір даних та їх обробка 27
2.2.2 Класифікація отриманих даних 28
2.2.3 Обмеження результатів 32
2.2.4 Статистичний аналіз даних 34
2.2.5 Переваги та недоліки обраних алгоритмів та методів розрахунку 37
3 Архітектура застосунку 41
3.1 Загальний огляд використаних технологій 41
3.2 Програмна реалізація прототипу застосунку 42
3.2.1 Аналіз інформації для обробки 47
3.2.2 Інтерфейс користувача 51
3.3 Інструкція користувача 55
Висновки 61
Перелік джерел посилання 62
Додаток А Тестування застосунку 66

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ОС – операційна система

ООП – об’єктно-орієнтоване програмування

ПО – предметна область

API – Application Programming Interface (інтерфейс прикладного програмування)

AWS – Amazon Web Service (вебсервіс Amazon)

AWT – Abstract Window Toolkit (абстрактний віконний інтерфейс)

GUI – Graphical User Interface (графічний інтерфейс користувача)

GWT – Google Web Toolkit (вебінструментарій Google)

MVC – Model-View-Controller (модель-вид-контролер) – шаблон програмування, що розділяє архітектуру на три відповідних рівня

RFID – Radio Frequency Identification (радіочастотна ідентифікація)

SWT – Standard Widget Toolkit (стандартний інструментарій віджетів)

ВСТУП

У сучасному світі, де спортивні змагання та події з кожним роком набувають популярності та повсякчасної можливості доступу до записів змагань та ігор, фізична підготовка стає надзвичайно важливою для досягнення успіху та визнання в спортивній галузі. Ефективність підготовки спортсменів залежить від багатьох факторів, таких як вікова категорія, тривалість та інтенсивність тренувань, раціональність харчування, медичні показники і т.д.

Один з ключових аспектів успішної підготовки атлетів – це моніторинг їхньої ефективності та результативності. Цей процес необхідно проводити на постійній основі, щоб мати змогу адаптувати тренувальні програми та стратегії до потреб та можливостей кожного спортсмена.

Метою роботи є розробка застосунку для швидкого та точного аналізу відсоткового відношення підготовки спортсмена до змагань з урахуванням певних параметрів для оцінки його фізичного стану та прогресу в обраній галузі спорту. Він дозволить спростити та упорядкувати процес моніторингу, забезпечуючи більш точний та зрозумілий аналіз даних. Для цього будуть використані методи та інструменти комп’ютерної науки та математики, а також дані, зібрані з різних джерел в рамках наукового дослідження.

Практична новизна та актуальність роботи полягає в тому, що застосунок має забезпечити збір та обробку даних про фізичний стан спортсменів, їхні досягнення та інші важливі показники, та надаватиме тренерам можливість в режимі реального часу відслідковувати прогрес підготовки, а також вчасно коригувати тренування.

1 АНАЛІТИЧНИЙ ОГЛЯД ПО ТА ІНСТРУМЕНАТРІЮ ДЛЯ РОЗРОБКИ ДЕСКТОПНИХ ЗАСТОСУНКІВ З АНАЛІЗУ ЕФЕКТИВНОСТІ ПІДГОТОВКИ СПОРТСМЕНІВ

У підготовці до змагань найважливішим є забезпечення адекватного навантаження, що відповідає меті і фізичним можливостям спортсмена. Основні аспекти, які слід враховувати при плануванні тренувань, включають регулярність (для покращення здібностей і уникнення травм), поступовість (починають з малого навантаження, збільшуючи його, дозволяючи адаптуватися) та різноманітність тренувань (для бігу з метою покращення витривалості використовують спринт, інтервальні, темпові пробіжки) [1].

Наразі існують різноманітні застосунки та програми для аналізу вищеперелічених факторів, які базуються на зборі та аналізі даних про тренування, змагання, результатів фізичних вимірювань та інших параметрів. Тож в цьому розділі буде проведено детальний розбір вже існуючих платформ на ринку програмного забезпечення, а також розглянуто фреймворки та технології для створення власних десктопних застосунків [2, 3].

1.1 Огляд існуючих застосунків для аналізу ефективності спортсменів

У ході аналізу застосунків для аналізу ефективності атлетів було виділено Sportlyzer, TrainingPeaks, AthleteMonitoring, Strava і Timing Events.

1.1.1 Sportlyzer

Sportlyzer – це хмарна платформа для тренерів та команд, що допомагає вони управляти тренуваннями, комунікацією та аналізом даних. Застосунок

надає можливість спостерігати за тренувальним процесом спортсменів та вести аналіз їхньої продуктивності.

Sportlyzer має клієнт-серверну архітектуру та забезпечується за допомогою хмарних сервісів Amazon Web Services (AWS). Застосунок пропонує своїм користувачам доступ до вебверсії, а також до мобільних пристроїв для iOS та Android. Sportlyzer пропонує інтеграцію з різними фітнес-трекерами та платформами, такими як Garmin, Polar, Strava, яка дозволяє автоматично отримувати звіти про активність спортсменів зі сторонніх ресурсів, що є його безперечною перевагою.

Однак є й негативні аспекти при роботі з цим застосунком. Для належного функціонування необхідний стабільний доступ до Інтернету, що може створювати проблеми у випадку обмеженої доступності мережі. Більш того, платформа може бути досить дорогою у використанні, особливо для менших команд й індивідуальних тренерів. Функціональність безкоштовної версії дуже урізана та містить лише 5 можливостей (рис. 1.1) [4].

Рисунок 1.1 – Функціонал безкоштовної та платної версії Sportlyzer

1.1.2 TrainingPeaks

TrainingPeaks працює як онлайн-сервіс, тому для його використання необхідний доступ до Інтернету та браузер. Сервіс має багато функцій для моніторингу та аналізу тренувань, таких як можливість створювати та редагувати тренувальні плани, відслідковувати прогрес, аналізувати дані про тренування за багатьма параметрами і т.д. Так само, як і Sportlyzer, TrainingPeaks підтримується на iOS та Android пристроях, може інтегруватись зі сторонніми гаджетами та програмами.

Недоліками TrainingPeaks можуть бути висока вартість платної версії, а також складнощі в налаштуванні та використанні для новачків. Також, деякі користувачі відмічають недостатню інтуїтивність користувацького інтерфейсу та складність встановлення зв’язку з фітнес-трекерами [5].

1.1.3 AthleteMonitoring

AthleteMonitoring відрізняється від попередніх сервісів тим, що забезпечує аналіз більш ніж 400 параметрів, пов’язаних з фізичною підготовкою, що дозволяє отримувати детальну інформацію про виконання спортсменом різних вправ, показники його здоров’я та стану тренування. Цей застосунок забезпечує моніторинг у реальному часі, що дозволяє вчасно реагувати на зміни в тренувальному процесі та налагоджувати навантаження для досягнення максимальних результатів.

Головний недолік AthleteMonitoring спільний з попередньо проаналізованими платформами – він є досить недешевим та може не відповідати бюджету багатьох спортивних організацій та клубів. Більш того, безкоштовна пробна версія дуже обмежена у функціоналі, що може скласти додаткові труднощі для його впровадження та використання [6].

1.1.4 Strava

Strava доступний як вебсайт та мобільний застосунок на платформах Android та iOS, який дозволяє відстежувати і аналізувати тренування, спілкуватися з іншими спортсменами та змагатися з ними в різних види спорту. Платформу вирізняє серед конкурентів можливість створювати та приєднуватися до груп для спільного тренування та змагань з іншими користувачами; вбудована функція GPS-навігації, яка допомагає користувачам планувати та виконувати маршрути на велосипеді або пішки. Однак через останню функцію є потенційні проблеми з безпекою, оскільки деякі користувачі можуть дізнатися про приватні маршрути та тренування, які не повинні бути доступними для громадського перегляду. Ще одним недоліком являється обмеження на безкоштовний облік тільки для двох типів активностей – бігу та їзди на велосипеді [7].

1.1.5 Timing Events

Timing Events – це програмне забезпечення для спортивних подій та змагань, який дозволяє організаторам ефективно керувати своєю подією. Воно здійснює облік та аналіз результатів змагань і тренувань з використанням технології RFID, що дозволяє збирати дані про учасників, такі як час, швидкість, відстань, середня температура тіла тощо. Також застосунок має можливість інтеграції з іншими системами для отримання додаткових даних: з GPS-пристроями та біометричними датчиками.

Вочевидь, недоліками є те, що платформа не безкоштовна, а вартість використання достатньо висока (від $ 19,99/ 1 захід). З точки зору реалізації технології RFID досить складні та вимагають додаткових витрат на обладнання, що й зумовлює таку високу ціну [8].

1.1.6 Результати порівняння застосунків

Представлені застосунки для аналізу ефективності спортсменів є найбільш популярними, однак з огляду на активний розвиток цієї області, подібних платформ існує достатньо велика кількість. Проаналізувавши Sportlyzer, TrainingPeaks, AthleteMonitoring та Strava, було виявлено ряд спільних проблем (наведені у таблиці 1.1) [9].

Таблиця 1.1 – Проблематика розглянутих застосунків

Застосунок	**Вартість підписки**	**Безкоштовний функціонал**	**Інтерфейс**
Sportlyzer	від 24 € /міс. (залежно від кількості)	скорочено до 5 не аналітичних функцій	інтуїтивно зрозумілий
TrainingPeaks	19,95 € / міс.	немає відстеження прогресу, функції редагування минулих тренувань	складний та спрямований на професійних користувачів
AthleteMonitoring	платні плани від 10 $ / міс	аналогічно до TrainingPeaks	складний для новачків
Strava	від 7,99 $ / міс	зменшена кількість підтримуваних видів спорту	простий у розумінні
Timing Events	від 19,99 $ / 1 захід	немає	інтуїтивно зрозумілий

Таким чином видно, що застосунки мають високу вартість підписки, а безкоштовні версії мають дуже обмежені функціональні можливості, в деяких платформах є проблеми зі зрозумілістю інтерфейсів. Також, хотілося б

відмітити, що статистика за більшістю аналогів для аналізу спортивних досягнень вираховується за одним певним параметром, але мало де можна зустріти загальну оцінку тренувального прогресу.

1.2 Аналіз засобів для створення десктоп застосунків мовою Java

В цій частині розділу буде проведено порівняння таких відомих Java фреймворків та бібліотек, як Swing, AWT, SWT та JavaFX.

1.2.1 AWT

AWT – це API для створення віконних програм із графічним інтерфейсом користувача на Java. AWT надає набір класів та інтерфейсів для створення графічного інтерфейсу користувача (GUI) в програмах на мові Java. Вона містить примітивні компоненти, такі як кнопки, поля введення, списки та інші, які можна використовувати для створення інтерактивних застосунків. Приклад інтерфейсу, створеного за допомогою цієї бібліотеки можна побачити на рисунку 1.2.

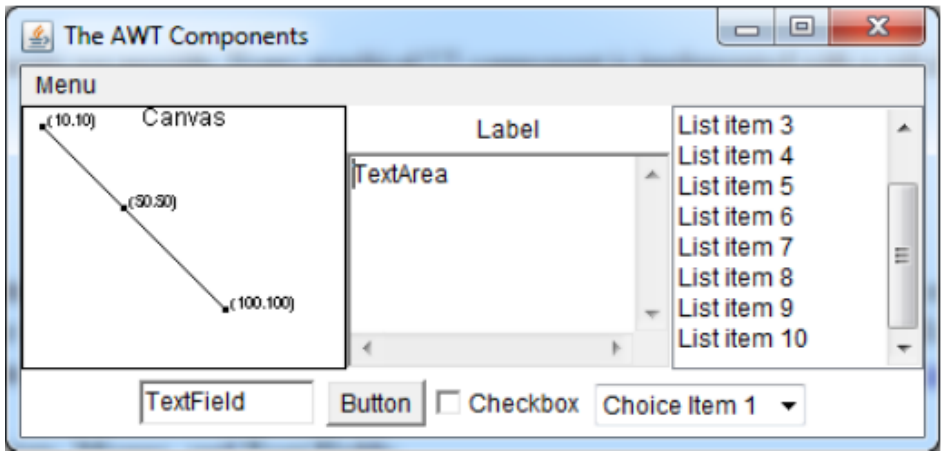


Рисунок 1.2 – Приклад інтерфейсу, створеного за допомогою AWT

Цей фреймворк, залежить від платформи та використовує нативні компоненти, тобто компоненти GUI, що належать AWT, не однакові для всіх ОС. Ще один недолік полягає в тому, що AWT не містить засобів для відображення іконок, утворення таблиць, а також тут повністю відсутні спливаючі підказки.

З часом AWT втратив свою популярність через те, що його компоненти мали обмежені можливості налаштування вигляду та стилю, а також потребується багато часу на виконання програми. Тому з’явився наступник AWT – Swing, який надавав більшу гнучкість та налаштування вигляду користувацького інтерфейсу [10].

1.2.2 Swing

Swing – це набір графічних інтерфейсів нового покоління, створений компанією Sun Microsystems для забезпечення корпоративної розробки на Java. Під корпоративною розробкою мається на увазі, що Swing може використовуватись для створення масштабних програм мовою Java із широким набором потужних компонентів. Крім того, доступна можливість легко розширити або змінити ці компоненти, керуючи їхнім виглядом і поведінкою. Однак, важливо пам’ятати, що Swing не є повною заміною AWT, він фактично створений на основі бібліотек свого попередника [11].

Swing пропонує безліч компонентів для створення інтерактивних програм, таких як кнопки, поля входу, списки, таблиці, панелі, діалогові вікна та інше. Архітектура цієї бібліотеки була орієнтована на забезпечення легкого впровадження, тож Swing дозволяє власноруч корегувати Look and Feel (L&F) розроблюваного застосунка. «Look» визначає зовнішній вигляд компонентів, а «Feel» – їх поведінку. Взагалі, для початку роботи з цим фреймворком користувачеві не потрібні знання, пов’язані зі створенням елементів

інтерфейсу користувача, анотування полів моделі вже достатньо, для використання розроблених форм на основі макету сітки [12].

Приклад інтерфейсу, розробленого зі Swing зображено на рисунку 1.3.

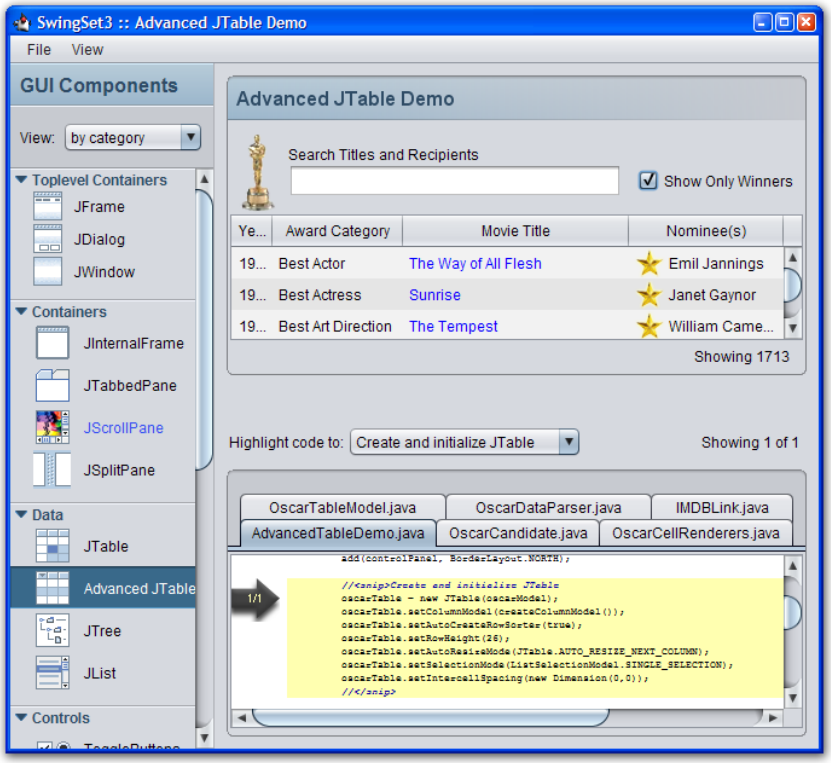


Рисунок 1.3 – Приклад інтерфейсу, створеного за допомогою Swing

Однак, найвагомішим недоліком цієї бібліотеки є нестабільність та значні затримки в роботі застосунків, особливо це помітно при використанні перших версій Swing. Окрім того, при роботі з менеджером компонування для більш складних інтерфейсів, процес розробки може бути ускладненим та не дуже зручним.

1.2.3 SWT

Будучи незадоволеними реалізаціями існуючих бібліотек для графічних інтерфейсів користувача, і потребуючи потужного рішення для своєї IDE

Eclipse, спільнота Eclipse вирішила розробити новий інструментарій GUI. На рисунку 1.4 представлено інтерфейс, розроблений за допомогою SWT.

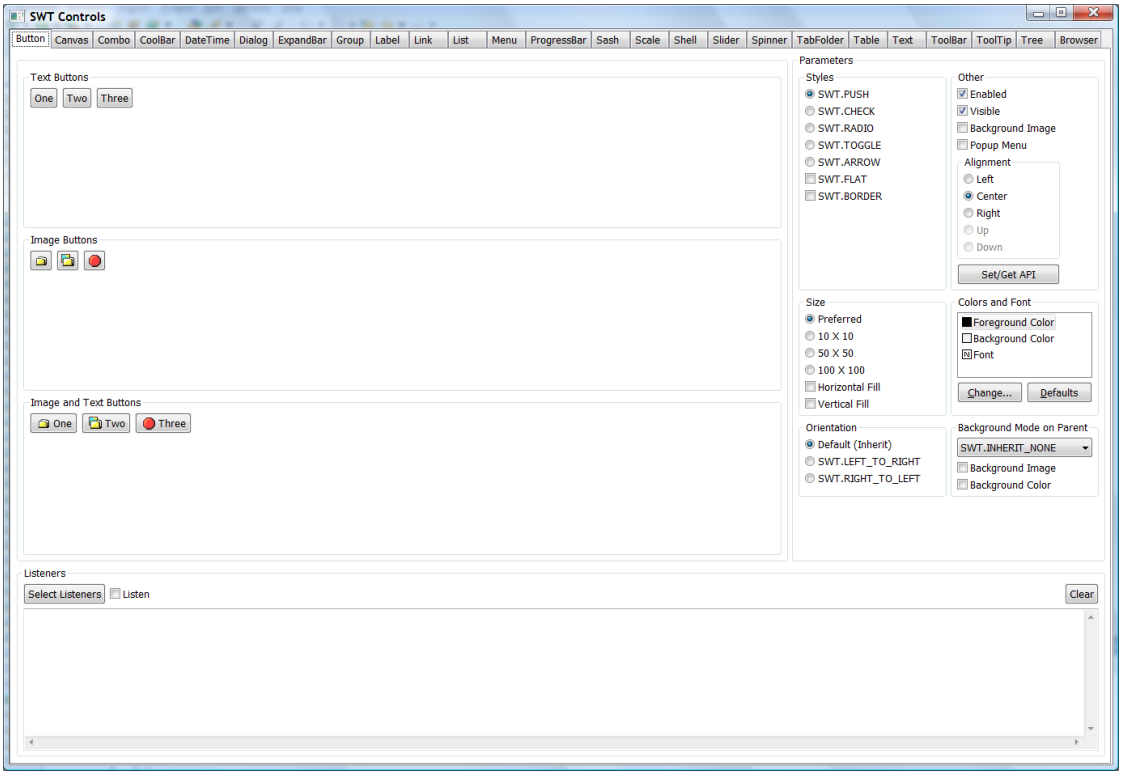


Рисунок 1.4 – Приклад інтерфейсу, створеного за допомогою SWT

SWT – це набір віджетів з відкритим кодом для Java, розроблений для забезпечення ефективного портативного доступу до засобів інтерфейсу користувача ОС, у яких він реалізований [13].

SWT використовує найкраще від обох попередників – AWT і Swing. SWT використовує вбудовані у ОС компоненти, так само як AWT, однак окрім цього вона має ще й власні інтерфейси взаємодії для кожної платформи. Також, через цю бібліотеку можливо використовувати як елементи з AWT, так і зі Swing. Ще одна перевага платформи полягає в тому, що замість програмування безпосередньо в API, програма може спілкуватися з API через JFace, рівнем абстракції поверх SWT, що забезпечує потужність шаблону MVC [14].

Очевидним недоліком SWT є потреба інсталяції окремих бібліотек для різних платформ, та необхідність відстеження використання ресурсів пам’яті під час їх звільненні, оскільки цей процес не автоматичний. Крім того, архітектура SWT є досить складною, що може ускладнювати процес розробки власного інтерфейсу.

1.2.4 JavaFX

JavaFX – це фреймворк для створення застосунків нового покоління з відкритим кодом для десктопних, мобільних і вбудованих систем, побудованих на Java. Він був розроблений компанією Sun Microsystems і опублікований у грудні 2008 р. та призначений для заміни Swing як стандартної бібліотеки GUI для Java SE [15, 16].

JavaFX пропонує інструменти для розробки динамічних інтерфейсів користувача, включаючи графіку, анімацію, медіа та 3D-графіку. Крім того, підтримується мультиплатформенність: застосунки працюють однаково на різних ОС, таких як Windows, macOS, Linux та мобільні пристрої [17].

JavaFX побудовано на багаторівневій архітектурі компонентів, які підтримують інтерфейс користувача. Діаграма на рисунку 1.5 показує архітектуру JavaFX API та компоненти, які підтримують JavaFX API.

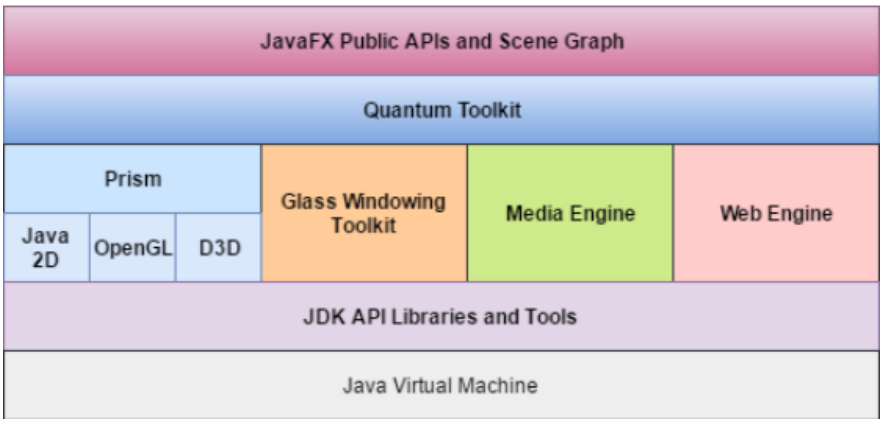


Рисунок 1.5 – Архітектура компонентів JavaFX

Верхній шар забезпечує завершений набір загальнодоступних API, які забезпечують гнучкість створення багатофункціональних клієнтських програм. Він також забезпечує граф сцени, який є ієрархічним деревом вузлів, який включає всі елементи та їхні зв’язки в інтерфейс програми. Quantum Toolkit – це абстракція над низькорівневими компонентами Prism, Glass, Media Engine і Web Engine. Він поєднує Prism (використовується для відтворення графіки) і GWT (надає можливості для керування вікнами, таймерами та чергами подій) і робить їх доступними для JavaFX. У шарах нижче – Prism (використовується для відтворення графіки в JavaFX). За допомогою Web Engine можна керувати вебвмістом із Java, а Media Engine підтримує відтворення відео та аудіо контенту. Віртуальна машина Java (JVM) розташована внизу та обробляє прикладні завдання, такі як керування стеком, завантаження та зберігання змінних, арифметика, виклик і повернення методу, перетворення типів, створення винятків та інше [18]. JavaFX має більш простий та легкий API, ніж попередньо розглянуті бібліотеки. Це дозволяє програмістам швидко створювати інтерфейси користувача з меншим обсягом коду. Порівняно зі Swing, AWT, SWT, фреймворк JavaFX має значно ширший спектр компонентів для використання, які дозволяють користувачу створювати більш графічно складні та привабливі інтерфейси [19].

1.3 Постановка задачі

Проаналізувавши існуючі рішення в області аналізу ефективності підготовки спортсменів та існуючі засоби для створення десктопних застосунків на Java, можна наголосити про актуальність проблеми та висунути ряд функціональних і технічних вимог щодо розроблюваного прототипу застосунку [20].

Об’єктом роботи є аналіз ефективності підготовки спортсменів.

Метою роботи є розробка застосунку для швидкого та точного аналізу відсоткового відношення підготовки спортсмена до змагань з урахуванням певних параметрів для оцінки його фізичного стану та прогресу в обраній галузі спорту.

Застосунок буде призначений для використання в особистих цілях користувача на десктопних пристроях. Необхідно забезпечити можливість вносити тренування за певним графіком. Наприклад, для найбільш швидкого прогресу будуть розглядатись щоденні тренування, однак ще важливішим критерієм буде їх результативність. Слід пам’ятати, що в залежності від отриманих обчислень, можливе як збільшення, так і для зменшення підготовки.

Для досягнення мети необхідно вирішити такі завдання щодо розробки прототипу застосунку:

– створення зручного та інтуїтивно зрозумілого інтерфейсу;

– забезпечення можливістю заповнювати журнал тренувань отриманими результатами показників;

– реалізація розрахунку формул за кожним типом тренувань;

– проведення перерахунку процентного прогресу ефективності підготовки після внесення інформації про нове тренування;

– забезпечення відображення статистики за результатами проведених тренувань;

– надання можливості експорту результатів статистичного аналізу даних (середнє відхилення, середнє значення показників тренувань).

2 МОДЕЛЮВАННЯ ЗАСТОСУНКУ ДЛЯ АНАЛІЗУ ЕФЕКТИВНОСТІ ПІДГОТОВКИ СПОРТСМЕНІВ

Для початку, треба визначити, які метрики будуть використовуватись для оцінки прогресу, наприклад, кількість повторень, вага, час або дистанція. Після можна розглянути кожен вид спорту окремо і визначити, мету і кількість тренувань необхідних для певного результату.

Наступним кроком є виставлення коефіцієнтів для параметрів. Коефіцієнти можуть використовуватись для корекції обчислень залежно від виду спорту, бути позитивними і негативними, використовуватись як для збільшення, так і для зменшення прогресу підготовки.

Останнім кроком є розрахунок формули для кожного тренування. Вона може відрізнятись в залежності від мети. Наприклад, якщо мета збільшити м’язову масу, то включатиметься більша кількість вправ з великими вагами та меншу кількість повторень. Якщо мета підготуватись до змагань, то формула буде орієнтована на показники швидкості, витривалості спортсмена, тощо.

2.1 Визначення параметрів для аналізу ефективності підготовки

2.1.1 Визначення коефіцієнтів для спільних параметрів до кожного виду спорту

Під час аналізу способів підготовки до змагань були розроблені коефіцієнти для кожного параметру тренування. Вони вказують на важливість кожного з показників при розрахунку ефективності тренування та мають свій вагомий внесок в загальну оцінку результату [21].

Розглянемо кожен з коефіцієнтів детальніше на прикладі бігу (для бігу, наприклад, було виділено такі параметри, як тривалість тренування,

максимальний та середній пульс, дистанція, самопочуття, максимальна та середня швидкість) [22]:

– durK (0,1) – відображає важливість тривалості тренування. Чим триваліше тренування, тим більший внесок він має у загальну оцінку результату;

– pulseMaxK (0,1) – цей коефіцієнт відображає важливість максимального пульсу, який спостерігався під час тренування. Він вказує на те, що тренування було інтенсивним, тобто спортсмен під час тренування досягав свого максимального серцевого ритму, але ритм не перебільшував референтні значення;

– pulseAverageK (0,15) – цей коефіцієнт відображає важливість середнього пульсу під час тренування. Чим вищий середній пульс, тим більше фізичне навантаження отримав спортсмен, але тим менше його витривалість;

– selfFeelingK (0,1) – цей коефіцієнт відображає важливість самопочуття після тренування. Чим краще спортсмен себе відчуває після тренування, тим більший внесок він має у загальну оцінку результату;

– distanceK (0,2) – цей коефіцієнт відображає важливість пройденої відстані під час тренування. Чим більшу відстань пройшов спортсмен, тим більший внесок він має у загальну оцінку результату;

– speedMaxK (0,3) – цей коефіцієнт відображає важливість максимальної швидкості, якої досягає спортсмен під час тренування;

– speedAverageK(0,05) – цей коефіцієнт відображає важливість середньої швидкості спортсмена під час тренування.

Коефіцієнти вказують на вагомість кожного з показників у загальній оцінці результату тренування. Наприклад, distanceK дорівнює 0,2, а це означає, що дистанція, пройдена під час тренування, є досить важливим фактором, але не настільки важливим, як максимальна швидкість (speedMaxK), якій дорівнює коефіцієнт 0,3. Максимальна швидкість (speedMaxK) є досить

важливою у бігу на короткі дистанції, а в середньому темпі (speedAverageK) важливішою є витривалість та пульс (pulseAverageK).

Параметри durK, pulseMaxK, pulseAverageK, selfFeelingK використовуються в усіх видах тренувань, адже у кожному виді спорту такі параметри мають значення. Проте у різних видах ці параметри мають різну вагу, а отже – різні значення коефіцієнтів.

Вищевказані коефіцієнти для плавання збігаються за винятком pulseAverageK, де для плавання значення складає 0,05 у порівнянні 0,15 – для бігу, оскільки спираючись на мій досвід та вивченні джерел відкритого типу, можна стверджувати, що значення середнього пульсу для тренувань з бігу має більшу вагу [23].

Для тренувань з боротьби значення наступні:

- durK – 0,3;
- pulseMaxK – 0,2;
- pulseAverageK – 0,2;
- selfFeelingK – 0,1.

Високий коефіцієнт для параметру тривалості тренування при підготовці до змагань з боротьби зумовлений тим, що цей параметр має прямий вплив на збереження та підвищення витривалості борця на ринзі. У цьому виді спорту часто відбуваються довготривалі змагання, під час яких спортсменові вкрай необхідно зберігати оптимальний рівень енергії та витривалості [24].

Для змагань з велоспорту або для підготовки до марафону було прийняте рішення використовувати наступні значення:

- durK – 0,3;
- pulseMaxK – 0,2;
- pulseAverageK – 0,1;
- selfFeelingK – 0,1.

Обґрунтування високого коефіцієнту для параметру тривалості тренування аналогічне до обґрунтування цього ж самого коефіцієнту для боротьби, оскільки для велоспорту витривалість є однією з найголовніших характеристик.

Для пауерліфтингу значення зазначених спільних параметрів будуть наступні:

– durK – 0,2;
– pulseMaxK – 0,1;
– pulseAverageK – 0,1;
– selfFeelingK – 0,2.

У кожного тренера можуть бути різні підходи до визначення коефіцієнтів, але в цьому випадку ці значення відображають важливість відповідних параметрів для підготовки до змагань з пауерліфтингу, враховуючи специфіку цього виду спорту за суб’єктивною експертною думкою.

Останній доступний вид тренувань – вільне тренування. Коефіцієнти наступні:

– durK – 0,2;
– pulseMaxK – 0,2;
– pulseAverageK – 0,15;
– selfFeelingK – 0,15.

Коефіцієнти майже однакові, що зумовлено великою кількістю видів фізичної активності, що можуть припадати до категорії «Вільне тренування». У застосунку буде можливість обрати 5 видів спорту для підготовки та ввести 6 видів тренувань: 5 профільних та 1 вільне. Будь-яка активність, відмінна від 5 профільних видів тренувань має заноситись до категорії «Вільне тренування».

2.1.2 Специфічні параметри для окремих видів спорту

Далі необхідно зазначити параметри та коефіцієнти, що є окремими для кожного виду тренувань.

Одним з найважливіших елементів підготовки до змагань з плавання є тренування на витривалість, яке допомагає підвищити стійкість до втоми і збільшити час перебування у воді.

Для досягнення цієї мети можна використовувати тренування на довгі дистанції, такі як плавання на відстань від 800 метрів і більше, а також тренування з використанням плавальних інструментів, таких як плавні круги, тощо.

Для підготовки до змагань з плавання окрім загальних окрім загальних параметрів та коефіцієнтів, було підібрано низку профільних параметрів, а також коефіцієнтів до них: час затримання дихання, час запливу на 100 метрів та дистанція [25]. Коефіцієнти для вищевказаних параметрів:

– distanceK (0,15) – дистанція. Відстань, яку проплив спортсмен під час тренування, є важливим показником його фізичної підготовки;

– breathHoldingK (0,2) – час утримання дихання. Даний параметр вказує на здатність спортсмена до контролю дихання та підготовку до пірнань;

– timeHundredMetersK (0,3) – час на 100 метрів. Даний параметр є ключовим для плавців на короткі дистанції і вказує на швидкість, з якою спортсмен здатний пропливати дистанцію.

Найважливішим та найвагомішим фактором для оцінки результатів обрано час запливу на 100 метрів, оскільки цей результат є ключовим у підготовці до змагань з плавання, тому він має найбільшу вагу – 0,3.

Для підготовки до змагань з боротьби обрано лише один параметр, адже параметрів за замовчуванням для цього виду тренувань достатньо для повноцінного аналізу. Цей єдиний критерій – перемога чи програш у спарингу. Він є виключенням і не має коефіцієнту, від фінального значення прогресу за

тренування віднімається 0,1 у разі програшу, та навпаки додається 0,1 у разі перемоги. За відсутності спарингів ніяких дій не відбувається.

Для велоспорту, аналогічно бігу чи плаванню, параметром обрано дистанцію, що долає користувач: distanceK – 0,3. Оскільки у велоспорті ключові характеристики – це сила ніг та витривалість, найголовніше у тренуваннях – практика, а отже більше проїхав – краще результат.

Для пауерліфтингу найголовнішим коефіцієнтом є піднята на тренуванні вага: liftedWeightK – 0,4. Коефіцієнт 0,4 найбільший серед усіх параметрів усіх тренувань, оскільки чим більше вагу може підняти людина – тим більше вірогідність її перемоги на змаганнях.

Для вільного тренування, з причини різноманітності активності, яку можна ввести як «вільне тренування» основним параметром обрано калорії, які спалив користувач не тренуванні: caloriesBurnedK – 0,3.

2.2 Моделювання алгоритму вирішення задачі

Проблематика проведення аналізу ефективності полягає в правильній структуризації та дослідженні введеної користувачем інформації. Тому процес оцінки прогресу атлета можна представити у вигляді загальних чотирьох кроків, наведених на рисунку 2.1.

Рисунок 2.1 – Загальні кроки для реалізації аналізу ефективності спортсменів

Далі більш детально розглядатиметься кожен з вищевказаних етапів реалізації.

2.2.1 Збір даних та їх обробка

Дані вносяться у застосунок користувачем після тренування. Для кожного виду тренування існують обов’язкові параметри. Користувач може ввести цей мінімальній набір, але для досягнення максимального прогресу необхідно заповнити усі поля, оскільки порожні ініціалізуються нулем.

Кожне значення приводиться до 100 (в залежності від еталонного значення (підпункт 2.2.2)) та множиться на коефіцієнт тренування 0,004, якщо це профільне тренування, або ж на 0,0004, якщо тренування непрофільне.

Приклад: для бігу маємо 7 полів із коефіцієнтами: тривалість (0,1), максимальний пульс (0,1), середній пульс (0,15), самопочуття (0,1), дистанція (0,2), максимальна швидкість (0,3), середня швидкість (0,05). Якщо привести усі дані до 100 та припустити, що значення тренування ідеальні. Отримуємо:

100*0,1 + 100*0,1 + 100*0,15 + 100*0,1 + 100*0,2 + 100*0,3 + 100*0,05 = 0 + + 10 + 15+ 10 + 20 + 30 + 5 = 100.

Припустимо, що біг – це профільне тренування, отже множимо на коефіцієнт профільного тренування 0,004 – 100*0,004 = 0,4. Отриманий результат – це максимальне значення, що можна отримати за одне тренування.

Тепер припустимо, що користувач вказав лише обов’язкові поля (тривалість, самопочуття, дистанцію). Маємо:

100*0,1 + 0*0,1 + 0*0,15 + 100*0,1 + 100*0,2 + 0*0,3 + 0*0,05 = 10 + 0 + 0 + + 10 + 20 + 0 + 0 = 40.

Оскільки тренування профільне множимо на 0,004: 40 * 0,004 = 0,16. Таким чином, можна переконатися, що неповне введення даних значно зменшить швидкість підготовки спортсмена, за алгоритмом застосунку.

2.2.2 Класифікація отриманих даних

Після внесення даних користувачем вони класифікуються за типом тренування та вносяться до списку усіх тренувань [26-30]. Спершу відбувається їх валідація. Для кожного параметру існує референтне значення наведене у таблиці 2.1.

Таблиця 2.1 – Референтні значення параметрів для усіх тренувань

Параметр	**Значення**
Тривалість будь-якого тренування(хв)	20-1440
Максимальний пульс для усіх тренувань	40-250
Середній пульс для усіх тренувань	40-250
Самопочуття	-
Дистанція бігу(метри)	0-110000
Максимальна швидкість(км/год)	0-50
Середня швидкість(км/год)	0-50
Дистанція запливу(метри)	0-30000
Затримка дихання(секунди)	0-1500
Час запливу на 100 метрів(секунди)	40+
Спалені калорії	0-4000
Піднята вага(кг)	0-1300
Дистанція заїзду на велосипеді(метри)	0-200000

Для параметру самопочуття, референтні значення не наведені, оскільки користувач має можливість обрати з випадаючого списку «ComboBox» обрати одне з п’яти значень [1,2,3,4,5], де 1 – дуже погано, 5 – дуже добре. Це значення при калькуляції множиться на 20, тим самим отримуємо 5 варіантів значень [20,40,60,80,100].

Якщо значення, введене користувачем не відповідає референтному значенню – виникає помилка, яку бачить користувач (у цьому випадку дані не додаються до списку тренувань та не впливають на прогрес). Якщо введені користувачем дані співпадають із референтними значеннями, то відбудеться калькуляція за (2.1).

$$v_1 \cdot k_1 + v_2 \cdot k_2 + \cdots + v_x \cdot k_x, \tag{2.1}$$

де v_x – приведене до 100, значення параметру тренування;

k_x – коефіцієнт для параметру v_x.

Значення v розраховується виходячи з еталонного значення, що є окремим для кожного параметру:

$$v_x = \frac{v \cdot 100}{E_x}, \tag{2.2}$$

де v – введене користувачем значення параметру;

E_x – еталонне значення параметру.

Для значень максимального та середнього пульсу (усі тренування), а також значення часу запливу на 100 метрів (плавання) ця формула була інвертована із деяким доопрацюванням, оскільки вищевказані параметри вважаються оптимальнішими, коли їх значення стають менші. Тоді для значення часу формула виглядатиме як $v_x = 200 - \frac{v \cdot 100}{E_x}$, а для значень пульсу:

$v_x = 300 - \frac{2v \cdot 100}{E_x}$. Відмінність цієї формули полягає в тому, що негативний ефект від перебільшення еталонне значення введеним, подвоюється.

Якщо значення v_x перебільшує 100 (у випадках, коли введене користувачем значення v перебільшує еталонне значення E_x) значення v_x автоматично зменшується до 100. Нижче приведено значення усіх еталонних значень для параметрів усіх тренувань (табл. 2.2).

Таблиця 2.2 – Еталонні значення для параметрів кожного тренування

Назва параметру	Значення
Біг	
Тривалість(хв)	120
Максимальний пульс	165
Середній пульс	130
Дистанція(метри)	5000
Максимальна швидкість(км/год)	35
Середня швидкість(км/год)	23
Плавання	
Тривалість(хв)	90
Максимальний пульс	150
Середній пульс	120
Дистанція(метри)	2000
Затримка дихання(секунди)	150
Час запливу на 100 метрів(секунди)	60
Боротьба	
Тривалість(хв)	90
Максимальний пульс	150
Середній пульс	130

Продовження таблиці 2.2

Вільне тренування	
Тривалість(хв)	90
Максимальний пульс	180
Середній пульс	120
Спалені калорії	2000
Велоспорт	
Тривалість(хв)	180
Максимальний пульс	150
Середній пульс	140
Дистанція(метри)	30000
Пауерліфтинг	
Тривалість(хв)	90
Максимальний пульс	160
Середній пульс	130
Піднята вага	400

Після підрахування значення v_x виконується обчислення прогресу (підпункт 2.2.1). Для обчислення використовуються 2 коефіцієнти: коефіцієнт вільного (0,0004) та профільного тренувань (0,04).

Підготовка до змагань вимагає ретельного контролю за процесом тренувань, зокрема регулювання інтенсивності та обсягу тренувань залежно від мети та фізичного стану спортсмена. Коефіцієнт прогресу є одним з ключових факторів, які впливають на результат за розрахунком програми.

Вибір коефіцієнту прогресу може бути обґрунтований на основі наукових досліджень та емпіричних даних про вплив різних непрофільних тренувань коефіцієнту на результати. Наприклад, використання великого значення коефіцієнту прогресу може призвести до швидкого збільшення прогресу, що може ввести спортсмена в оману, ніби він готовий до змагань. З

іншого боку, використання занадто малого значення коефіцієнту прогресу може призвести до відображення недостатнього прогресу у тренуваннях [31].

Використання значення 0,004 для профільних тренувань може вказувати на відносно високий рівень вагомості тренування до прогресу, оскільки більший коефіцієнт прогресу вказує на швидший ріст готовності за алгоритмом програми.

В свою чергу, використання значення 0,0004 для непрофільних тренувань може вказувати на менший рівень вагомості тренувань, оскільки менший коефіцієнт прогресу вказує на повільніший ріст рівня підготовки. За таких значень коефіцієнту може бути досягнуто балансу між досягненням оптимального прогресу підготовки та зменшенням впливу непрофільних тренувань на фінальний результат.

2.2.3 Обмеження результатів

Після обчислення прогресу відбувається збереження даних та відображення оновленого результату. Проте, припустимо, що спортсмен за день тренувався 3 рази із еталонними значеннями кожного з параметрів і ввів ці данні. Якщо, максимальний результат за тренування дорівнює 0,4, то його результат за день має становити 0,4*3 = 1,2. Саме з цієї причини застосунок має обмеження у розмірі 0,5 одиниць прогресу за день. Це означає, що для алгоритму не важлива кількість тренувань, оскільки при досягнені прогресу 0,5 за день будь-які розрахунки припиняться та подальші тренування внесені у цей день не впливають на прогрес.

Таке обмеження введене з ряду причин. По-перше, повинен бути повний контроль над темпом прогресу. Встановлення обмеження дозволяє в контролювати темп прогресу спортсмена та запобігати занадто швидкому наростанню навантажень. Це може бути особливо важливо, оскільки занадто

швидкий прогрес може призвести до перевантаження, травм та зниження результативності.

По-друге, дуже вагомою є об’єктивність оцінки. Обмеження встановлення максимального прогресу за день може допомогти забезпечити більш об’єктивну оцінку прогресу спортсмена, незалежно від кількості тренувань, які він проводить за день. Це може бути корисно, оскільки різна кількість тренувань може впливати на загальний прогрес, а встановлене обмеження може допомогти вирівняти ці впливи та забезпечити більш точну оцінку.

І звісно, контроль за тренувальним процесом. Встановлення обмеження на максимальний прогрес за день може також допомогти забезпечити контроль за тренувальним процесом і враховувати розподіл навантажень на тривалий період. Це може бути корисно в плануванні тренувальних програм та досягнення кращого результату в довгостроковій перспективі, забезпечуючи баланс між навантаженням та відпочинком [32].

Загалом, введення обмеження на максимальний прогрес за день допомагає забезпечити більш безпечний, ефективний та узгоджений тренувальний процес для спортсмена, дозволяє уникнути занадто швидкого наростання навантажень та ризику перевантажень [33]. Після аналізу прогресу, у випадку, якщо користувач у налаштуваннях активував порівняння із світовими рекордами – відбувається перевірка деяких параметрів на відповідність світовому рекорду. На етапі прототипу таких значень два:

– WORLD_RECORD_SWIM – 46 секунд, встановлений світовий рекорд з плавання на 100 метрів;

– WORLD_RECORD_LIFTING – 1275 кг, встановлений світовий рекорд з пауерліфтингу.

Якщо значення введене користувачем перебільшує світовий рекорд, а тренування є профільним – рівень прогресу автоматично встановлюється на рівні 100%.

Ще одним не менш важливим обмеженням є встановлення необхідного рівня тренувань для досягнення результату. Припустимо, що спортсмен кожного дня вносить дані профільного тренування з еталонними значеннями. Щоб досягти повної готовності по розрахунку алгоритму, йому необхідно 250 днів та 250 тренувань, відповідно (250*0,4 = 100).

Але якщо спортсмен виставив цільовим тренуванням біг, а кожного дня вносить еталонні значення з тренування по велоспорту – за 2500 днів і 2500 тренувань він отримає (2500*0,04) 100% готовності до змагань з бігу, що буде некоректним значенням.

Тому, щоб запобігти таким діям з богу спортсмена – було створено додаткове обмеження на кількість тренувань, що обчислюється за формулою:

$$R_{max} = count*0,4, \tag{2.3}$$

де R_{max} – максимальне значення прогресу;

count – кількість профільних тренувань.

Отже, якщо спортсмен протягом року буде вносити тренування з велоспорту, а потім введе дані лише стосовно одного тренування з бігу – його прогрес закріпиться на рівні 0,4 (R_{max} = 1*0,4).

Це обмеження контролює підготовку спортсмена до окремого змагання, та обмежує накопичення прогресу, що ніяк, або майже ніяк не впливає на готовність до змагань.

2.2.4 Статистичний аналіз даних

У користувача є можливість зберегти статистику своїх тренувань до файлу. У статистиці відображається середнє значення та середнє відхилення кожного параметру обраного тренування.

Статистика тренувань є важливою галуззю статистики, яка досліджує дані, пов’язані з тренувальними процесами, такими як фізична активність, спорт, навчання, технічна підготовка та багато іншого. Одним з ключових аспектів статистики тренувань є вимірювання та аналіз даних, щоб зрозуміти їх характеристики та ефективність [34]. Середнє значення та середнє відхилення є важливими метриками, які використовуються для оцінки центральної тенденції та розсіювання даних тренувань.

Середнє значення, також відоме як середнє арифметичне, є однією з основних метрик, яка використовується для оцінки центральної тенденції даних тренувань. Воно обчислюється шляхом додавання всіх значень даних тренувань та ділення на їх кількість:

$$a_{\text{сер}} = \frac{a_1+a_2+\cdots+a_n}{n}. \tag{2.4}$$

Середнє значення відображає «середнє» або «типове» значення тренувань і може бути використане для порівняння різних тренувань, виявлення змін в тренуваннях з часом та визначення загальної тенденції тренувань. Наприклад, при вивченні ефективності тренувань з бігу середнє значення швидкості на тренування може допомогти визначити швидкість в незалежності від інших параметрів. Це може допомогти спортсмену прийняти рішення щодо корисності своїх результатів, враховуючи типовий результат тренувань.

Середнє відхилення є ще однією важливою метрикою в статистиці тренувань, яка використовується для вимірювання розсіювання даних тренувань:

$$S = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}. \tag{2.5}$$

Воно відображає розбіжність або варіабельність значень даних тренувань відносно їх середнього значення. Чим вище середнє відхилення, тим більша варіабельність в даних тренувань.

Середнє відхилення може допомогти розуміти, наскільки різні значення відрізняються від середнього, тобто наскільки розподіл даних тренувань розподіляється навколо середнього значення. Це може бути важливо при вивченні стабільності тренувань, виявленні випадкових відхилень або аномалій у даних тренувань.

Середнє значення та середнє відхилення можуть бути використані в різних аспектах статистики тренувань, таких як:

− оцінка ефективності тренувань: Середнє значення може допомогти визначити середній рівень досягнень у тренуваннях. Середнє відхилення може дати уявлення про рівень коливань або варіабельність даних тренувань, що може вказувати на стабільність або незбалансованість тренувань;

− моніторинг змін у тренуваннях: Середнє значення та середнє відхилення можуть бути використані для відстеження змін у тренуваннях з часом. Порівняння середнього значення та середнього відхилення на різних етапах тренувань може допомогти виявити тенденції та зміни в результатах тренувань;

− виявлення аномалій: Середнє значення та середнє відхилення можуть бути використані для виявлення аномалій або випадкових відхилень у тренуваннях. Значення, які відхиляються від середнього на велику величину відносно середнього відхилення, можуть вказувати на потенційні проблеми або несподівані зміни в тренуваннях;

− порівняння між різними тренуваннями: Середнє значення та середнє відхилення можуть бути використані для порівняння результатів тренувань між різними групами, різними особами або різними часовими періодами. Це може допомогти в оцінці різних стратегій тренувань, визначенні ефективності підходів та прийнятті відповідних рішень на основі даних тренувань;

− прогнозування майбутніх результатів: Середнє значення та середнє відхилення можуть бути використані для прогнозування майбутніх результатів тренувань. На основі середнього значення та середнього відхилення можна встановити базовий рівень тренувань і прогнозувати, як можуть змінитися результати з додатковими тренуваннями. Це може допомогти тренерам або спортсменам встановити реалістичні цілі, визначити необхідний рівень зусиль та здійснювати відповідні корекції під час тренувального процесу.

Таким чином, середнє значення та середнє відхилення відіграють важливу роль у статистиці тренувань, надаючи об’єктивну оцінку результатів тренувань, допомагаючи відстежувати зміни, виявляти аномалії та варіабельність результатів.

Додатково, важливо зазначити, оскільки розроблюваний застосунок – прототип у ньому використані не всі методи статистичних метрик, середнє значення та середнє відхилення можуть бути використані в комбінації з іншими статистичними метриками, такими як медіана, мода, діапазон, коефіцієнт варіації та інші, для більш комплексного аналізу результатів тренувань. Кожна з цих метрик надає свої власні переваги та може доповнювати одна одну, дозволяючи отримати більш повну картину результатів тренувань.

2.2.5 Переваги та недоліки обраних алгоритмів та методів розрахунку

З попередніх підрозділів видно, що спортсмен використовує ряд коефіцієнтів для врахування різних аспектів тренувань, таких як тривалість тренування, дистанція, самочуття, тощо, а також коефіцієнти, що відповідають за фізичні показники. Однак, варто зазначити, що ці коефіцієнти встановлені на певному рівні можуть бути відповідно налаштовані в

подальшому в автоматичному порядку в залежності від особистих потреб і можливостей спортсмена, але ця функція на даний час знаходиться на етапі теоретичного обґрунтування.

Однак, можна виокремити декілька недоліків або потенційних проблем в цій програмі з наукової точки зору:

− недостатня наукова обґрунтованість: використовується коефіцієнт, такий як «самопочуття», який не має чіткої наукової бази або доказів його ефективності в тренуваннях. Важливо мати науково обґрунтовані підходи до визначення коефіцієнтів, що використовуються в тренувальних програмах. Підхід до визначення цього коефіцієнту був суб’єктивним. Проте, треба зазначити, що на власному досвіді автор переконався в тому, що тренування із кращім самопочуттям та мотивацією дають більше результатів;

− відсутність індивідуалізації: існує сталий набір коефіцієнтів для всіх спортсменів, без врахування їх індивідуальних особливостей, фізичної підготовки та поточного стану здоров’я. Це може призвести до недоцільного навантаження або недостатньої завантаженості окремих спортсменів;

− обмежена кількість врахованих факторів: в застосунку використовуються лише кілька коефіцієнтів, таких як тривалість тренування, пульс, самочуття та фізичні показники. Однак, фізична підготовка спортсмена може бути визначена багатьма іншими факторами, такими як рівень фітнесу, м’язова маса, вік, стать, рівень досвіду тощо. Відсутність врахування цих факторів може призвести до недостатньо точного аналізу та розрахунку прогресу;

− відсутність функцій моніторингу та адаптації: застосунок не містить функцій моніторингу та адаптації до змін в фізичній підготовці спортсмена. Наприклад, вона не враховує зміни в фітнесі, втому, травмах або інших факторах, що можуть вплинути на відповідь організму на тренування. Відсутність цих функцій може призвести до недоцільного навантаження або ризику перевантаження;

− відсутність персоналізації: програма не враховує індивідуальних мета-цілей, обмежень та можливостей спортсмена. Відсутність персоналізації може призвести до неефективного навантаження, недоцільного розподілу тренувального навантаження та неправильного розрахунку прогресу.

Усі вищевказані проблеми є проблемами застосунку-прототипу. Вирішення кожної з них потребує великих зусиль та часу, однак згодом продукт може бути доповнено додатковими функціями та алгоритмами. Для часткового вирішення цих проблем, було використано наукові дослідження з фізіології, біомеханіки, науки про тренування та фітнес, щоб врахувати більше факторів в застосунку. Також було проведено додаткове дослідження з використанням AI-технологій для покращення точності розрахунків та адаптації до змін в фізичній підготовці спортсмена.

Із переваг обраного алгоритму необхідно виділити наступні:

− висока швидкодія: застосунок має ефективну реалізацію алгоритму, що дозволяє отримувати результати швидко та ефективно. Це було досягнуто завдяки оптимізації алгоритмів, та обчислювальних ресурсів;

− інноваційний підхід: застосунок використовує новаторський підхід, алгоритм, який відрізняє її від інших аналогічних програм на ринку. Порівняно з існуючими аналогами, не потребується багато часу на запуск та проводяться обчислення у режимі реального часу;

− розширені можливості: програма має розширені можливості, які дозволяють використовувати її в різних контекстах для підготовки до різних змагань;

− висока автоматизація: використовуються автоматизовані процеси, що сприяють зручності та ефективності використання. Це включає динамічну валідацію даних, автоматичне збереження даних при виході, відмалювання усіх анімованих компонентів, що допомагає користувачу максимально ефективно використовувати програму;

− зручний інтерфейс: застосунок має зручний та інтуїтивно зрозумілий інтерфейс, що дозволяє користувачам легко взаємодіяти з програмою та налаштовувати її параметри. Зручний інтерфейс може сприяти швидкому освоєнню програми, зменшенню часу на налагодження та використання програми, а також забезпечити комфортні умови для роботи користувачів.

3 АРХІТЕКТУРА ЗАСТОСУНКУ

3.1 Загальний огляд використаних технологій

У даному розділі буде надано загальний огляд технологій, які були використані під час розробки застосунку для аналізу підготовки спортсменів до змагань. Застосунок був розроблений з використанням Java 18 та JavaFX, двох потужних та популярних інструментів для розробки мовою програмування Java [35].

Java 18 є однією з останніх версій мови програмування Java, яка надає численні покращення і нові можливості для розробників. Деякі з основних особливостей Java 18 включають:

– покращення шаблонів інтерполяції рядків, що дозволяє зручно комбінувати значення в рядках;

– введення шаблонів специфікації для полів, що полегшує роботу з аннотаціями та валідацією;

– методи для роботи зі строками, масивами, іншими структурами даних;

– удосконалені API для роботи з текстом та датами;

– покращення продуктивності та стабільності.

JavaFX є бібліотекою для побудови графічних інтерфейсів користувача (GUI) у програмах на мові Java. Ця технологія надає багатий набір інструментів та компонентів для створення модерних та естетичних інтерфейсів користувача. Деякі особливості JavaFX включають:

– можливість використання розмітки за допомогою файлів FXML, що дозволяє відокремлювати дизайн інтерфейсу від логіки застосунку;

– підтримка анімації та переходів, що дозволяє створювати більш інтерактивні та привабливі інтерфейси для користувачів;

– гнучкість та можливість розширення за допомогою CSS, що дозволяє налаштовувати зовнішній вигляд елементів інтерфейсу;

– підтримка багатомовності та міжнародного форматування.

Для розробки використовується інтегроване середовище розробки IntelliJ IDEA. Ця IDE надає потужні інструменти для написання, налагодження та збирання коду, а також допомагає в організації проєкту та використанні різних бібліотек та фреймворків.

Ці технології та інструменти використовуються разом для розробки застосунку для аналізу підготовки спортсменів до змагань. Вони дозволяють створювати потужну та функціональну програму, яка надає зручний інтерфейс для аналізу даних підготовки.

3.2 Програмна реалізація прототипу застосунку

Після попереднього моделювання, обрано стиль програмування ООП. ООП (об’єктно-орієнтоване програмування) є парадигмою програмування, яка спрямована на організацію програмного коду навколо об’єктів, що представляють реальні або віртуальні об’єкти з реального світу. У ООП, програмний код структурується у вигляді класів та об’єктів, які взаємодіють один з одним за допомогою повідомлень та методів [36, 37].

Основні концепції ООП включають:

− класи та об’єкти. Клас є шаблоном або описом, за яким створюються об’єкти. Він визначає характеристики та поведінку об’єктів. Об’єкти є конкретними екземплярами класу, які мають свої власні стан та поведінку;

− інкапсуляція. Інкапсуляція відноситься до засобів обмеження доступу до внутрішнього стану та функцій об’єкта. Це дозволяє захищати дані від несанкціонованого доступу та змін, а також забезпечує абстракцію та модульність коду;

− наслідування. Наслідування дозволяє створювати нові класи на основі існуючих. Клас, що успадковує інші класи, називається підкласом або

похідним класом, а клас, від якого успадковується, називається батьківським класом або суперкласом. Це дозволяє використовувати вже існуючий код, розширюючи його функціональність або змінюючи його поведінку;

– поліморфізм. Поліморфізм в ООП дозволяє використовувати одну назву для виконання різних дій. Це може мати місце завдяки можливості перевизначення методів у похідних класах або за допомогою інтерфейсів. Поліморфізм дозволяє звертатися до об’єктів різних класів через спільний інтерфейс або базовий клас, що спрощує роботу зі змінними більш абстрактним рівнем;

– абстракція. Абстракція в ООП дозволяє виділити важливі характеристики об’єктів та ігнорувати незначні деталі. Це дає можливість створювати абстрактні класи та інтерфейси, які визначають загальні характеристики та поведінку, а деталі реалізації визначаються в конкретних класах.

Застосування ООП у розробці програмного забезпечення дозволяє забезпечити модульність, повторне використання коду, підвищення ефективності розробки та підтримки, забезпечення зрозумілості та розширюваності коду. ООП також сприяє створенню більш організованої та структурованої розробки, що полегшує співпрацю між розробниками у командному проєкті.

У контексті застосунку, що розробляється (далі WarmUp), ООП було використано для створення класів, які представляють, змагання та інші сутності, які відповідають за окремі обчислення, а також для організації взаємодії між ними через методи та повідомлення. Це допомогло створити структуровану та модульну систему для ефективного аналізу та обробки даних спортсменів.

Структура класів WarmUp має наступний вигляд:

– класи тренувань – Training, TrainingType та інші:

1) абстрактний клас Training – клас, що містить в собі загальну інформацію для усіх тренувань, усі класи тренувань унаслідуються від нього;

2) перерахування TrainingType – перерахування, що містить у собі назви усіх тренувань для полегшення читабельності коду;

3) класи SwimTraining, RunTraining, StrengthTraining, FreeTraining, FightTraining, BicycleTraining – класи тренувань, що містять специфічні для кожного тренування змінні;

– клас DayData – клас, що зберігає у собі усі тренування та інформацію щодо прогресу окремого дня. Усі головні обчислення проводяться у цьому класі. Їх наведено нижче:

1) методи runCalculations, swimCalculations – окремі методи для кожного типу тренувань, що виконують окремі калькуляції, безпосередньо для свого типу тренувань (із своїми коефіцієнтами, при цьому враховуючи, чи це тренування на поточний момент часу є цільовим);

2) defaultCalculations – метод, що є загальним для тренувань. Проводиться калькуляція «імпакту» основних полів, які є загальними для усіх тренувань (поля, що містяться у класі Trainings);

3) checkTrainingAmountMaxAllowed – один з ключових методів, виконує перевірку кількості тренувань. Якщо непрофільних тренувань більше 50% – максимальний прогрес буде становити 0,4*кількість профільних тренувань. Цей метод відповідає за виставлення максимального значення;

4) два методи calculateProgress (метод 1) та перевизначений calculateProgress (метод 2), що приймає параметром одне тренування працюють у парі. Метод 1 бере усі тренування, та у циклі кожне з них передає методу 2. Метод 2, у свою чергу, визначає тип цього тренування, проводить усі необхідні розрахунки, зокрема, перевірку максимально допустимого прогресу (checkTrainingAmountMaxAllowed), та повертає значення, яке є оцінкою прогресу за окреме тренування;

− клас DayDataComparator – клас, який є нащадком інтерфейсу Comparator<?>, використовується для сортування тренувань;

− клас TimeSeriesAnalysis – клас, який використовується для збереження звітів. Він складається з методів:

1) calculateStatistics – метод, що отримує усі тренування, класифікує їх за типом, витягає усі необхідні параметри, та передає на обробку дані універсальні для кожного тренування, які згодом записуються до файлу;

2) writeRun, wrriteSwim і т.д. – методи, специфічні для кожного тренування, в яких відбувається калькуляція специфічних для тренування змінних та їх запис у файл;

3) calculateMean – метод, що викликається вищеперерахованими методами. Калькулює середнє значення для переданого набору значень (параметрів тренувань);

4) calculateStandartDeviation – метод, що викликається методами класу TimeSeriesAnalysis (окрім calculateMean). Калькулює стандартне відхилення для переданого набору значень (параметрів тренувань);

− клас Audio – клас, який відповідає за відтворення аудіо-файлів у форматі .wav: sound – єдиний метод, що використовується застосунком. Він відповідає за відтворення аудіо-файлів;

− клас BinaryFileHelper – клас, що відповідає за збереження даних тренувань у файл:

1) saveDataToFile – метод, що автоматично викликається під час закриття програми та зберігає дані до файлу;

2) loadDataFromFile – метод, що автоматично викликається при ініціалізації програми та завантажує дані тренувань;

− класи Controller – класи, що являють собою обробники подій, що прив’язані до відповідних форм. Наприклад, у класі MainController прописано дію, яка буде виконуватись при натисканні кнопки «Exit». IDE автоматично зв’язує .fxml файли з контролерами, отже при натисканні кнопки «Exit»

запуститься код, що прописаний у класі MainController у методі exitButtonPressed;

– клас WarmUp – клас ініціалізації програмного застосунку. В цьому класі прописано ініціалізацію JavaFx.

Структуру взаємодії між класами зображено на рисунку 3.1. Тут зображено саме структуру класів, зв’язок між їх елементами, а не ієрархію наслідування.

Об’єкт «Controller» являє собою усі контролери, що з метою покращення зовнішнього вигляду діаграми були об’єднані в один об’єкт.

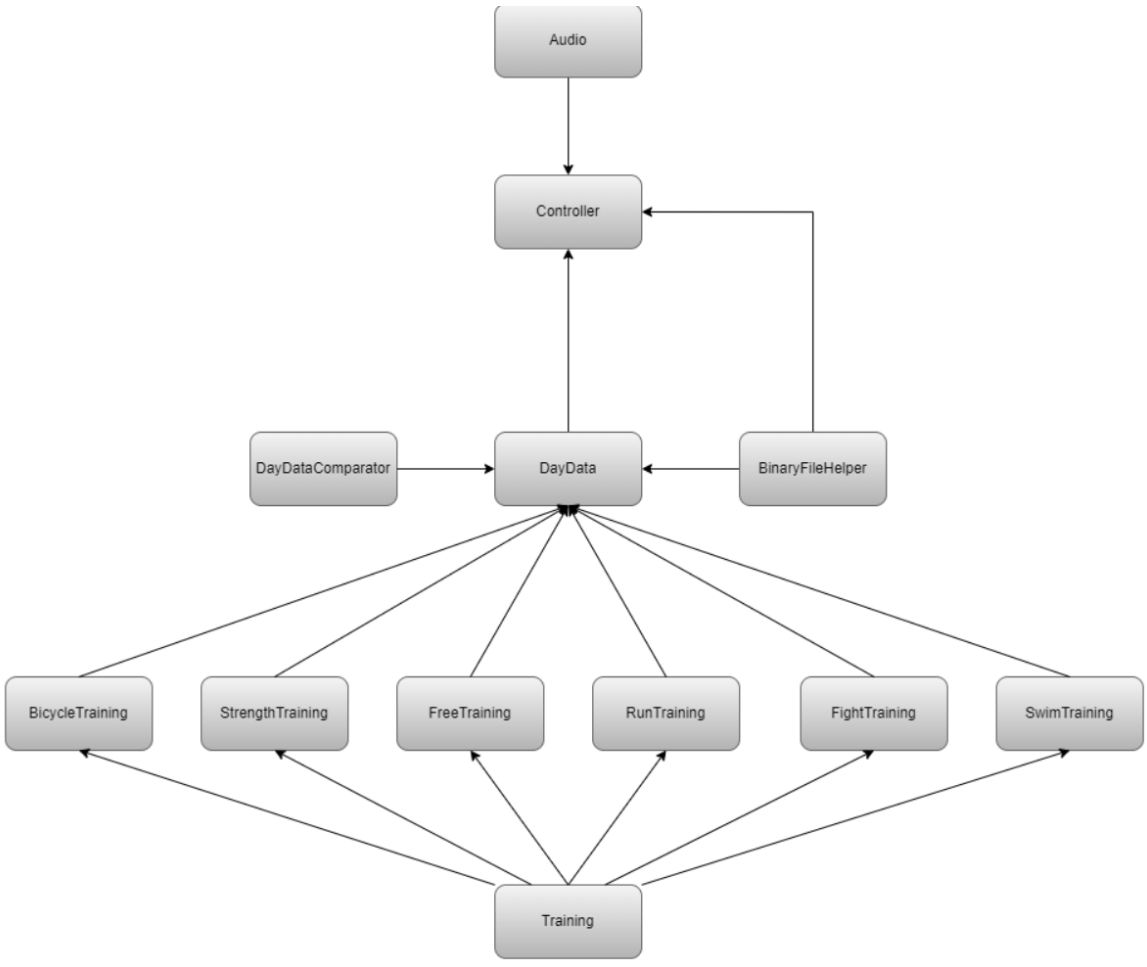


Рисунок 3.1 – Діаграма класів застосунку WarmUp

3.2.1 Аналіз інформації для обробки

Застосунок зберігає усі дані в бінарному вигляді у файлі, що знаходиться у директорії. Під час кожної ініціалізації ці дані завантажуються до системи, а під час кожного завершення роботи – перезаписуються в файлі. За процес роботи з файлом відповідає клас BinaryFileHelper.

У випадку, коли файл даних відсутній – відбувається ініціалізація застосунку без тренувань (рис. А.1), а налаштування, які підлаштовує під себе користувач, ініціалізуються значеннями за замовчуванням (рис. А.2), а саме:

– порівняння результатів із світовими рекордами за замовчуванням не відбувається;

– цільовим тренуванням за замовчуванням обрано поле «None», отже прогрес на обчислюється.

Після завантаження в систему, дані проходять обробку у класі DayData перед ініціалізацією графічного інтерфейсу користувача. Спочатку дані, що завантажені, проходять класифікацію згідно дати та типу тренувань і зберігаються у список об’єктів DayData.

Далі відбувається калькуляція прогресу у декілька кроків.

Крок 1. Завантажені дані класифікуються за типом тренування.

Крок 2. Для кожного тренування відбувається обчислення даних, що також ділиться на декілька етапів:

Етапі 1. Для тренування відбувається калькуляція значень, що є однаковими для кожного тренування.

Етап 2. Відбувається калькуляція специфічних значень, що є індивідуальними.

Крок 3. Далі, у випадку, якщо користувач увімкнув порівняння із світовими рекордами, відбувається порівняння даних із константними значеннями світових рекордів (лише для деяких тренувань та для деяких значень, значення рекордів актуальні на 20.04.2023).

Крок 4. Проводиться перевірка максимального значення прогресу.

Після проведення калькуляції виставляються значення шкали прогресу, які відображаються на головному екрані.

Для даних, що ввів користувач є можливість створити звіт. Після детального аналізу предметної області вдалося встановити, що середнє значення та стандартне відхилення є одними з найголовніших показників, які необхідно враховувати під час тренувань.

Під час створення звіту калькулюються середнє значення та стандартне відхилення для кожного параметру тренування. Користувач самостійно обирає тренування, по якому створюється статистика.

Створення звіту включає наступні кроки:

− вибір тренування для створення статистики;

− класифікація даних за обраним видом спорту;

− калькуляція середнього значення для кожного параметру тренування;

− калькуляція стандартного відхилення для кожного параметру тренування;

− форматування та відображення статистики у звіті.

Отриманий звіт міститиме інформацію про середнє значення та стандартне відхилення для кожного параметру тренування, що дозволить користувачу отримати уявлення про загальну ефективність та розподіл результатів своїх тренувань.

Окрім аналізу введених даних у застосунку реалізовано валідацію даних, що вводить користувач для запобігання введення помилкових даних та помилок в розрахунках. Валідація відбувається у реальному часі у момент введення символу користувачем під час введення нового тренування. За процес валідації, наприклад, значення максимального та середнього пульсу, відповідає функція valueBPMCheck класу AddWindowController (контролер вікна додавання тренування). Часткову реалізацію цієї функції надано в лістингу 3.1.

Лістинг 3.1 Функція valueBPMCheck:

```
public void valueBPMCheck() {
    if (maxBPMField.getText().equals(""))
        return;
    if (!maxBPMField.getText().matches("\\d+")) {
        return;
    }
   if (Integer.parseInt(maxBPMField.getText()) > 250) {
        errorLabel.setStyle("-fx-text-fill: red;");
        errorLabel.setText("Invalid MaxBPM value");
    } else if (Integer.parseInt(maxBPMField.getText()) >= 200) {
        errorLabel.setStyle("-fx-text-fill: red;");
        errorLabel.setText("Pulse too high, see a doctor");
    } else {
        errorLabel.setText("");
    }
}
```

Вище-зображена функція активується кожного разу, коли користувач вводить, видаляє або вставляє значення у поле пульсу. Функція перевіряє поточне значення та виводить на екран коментар стосовно введеного значення.

Якщо значення перебільшує 200 – виводиться коментар із порадою звернутися до лікаря, оскільки пульс занадто високий. Якщо значення перебільшує 250 – виводиться повідомлення про помилку, оскільки значення завелике.

Окрім перевірки значення, перевіряється і приналежність значення до цифрового діапазону (помилка «невірні дані», якщо у поле введено щось, окрім цифр).

Після первинної валідації даних та натискання користувачем кнопки зберегти відбувається валідація значень на приналежність їх до діапазонів, що вказані в таблиці 2.1.

У випадку, якщо усі значення пройшли валідацію – відбувається розрахунок нового значення прогресу.

Окрім додавання даних користувач також має можливість видаляти будь-яке тренування, що було ним внесене у програму раніше. За цей функціонал відповідає метод deleteKeyPressed (лістинг 3.2) класу MainWindowController.

Лістинг 3.2 Функція deleteKeyPressed:

```
public void deleteKeyPressed(KeyEvent keyEvent) {
    if (keyEvent.getCode() == KeyCode.DELETE) {
        Training selectedTraining = tableView.getSelectionModel().getSelectedItem();
        if (selectedTraining != null) {
            DayData dayData = MainController.getData().stream().filter(i->i.getDate().equals(selectedTraining.getCurrentDate())).findFirst().get();
            ArrayList<Training> list = dayData.getTrainings();
            list.remove(selectedTraining);
            dayData.setTrainings(list);
            tableView.getItems().remove(selectedTraining);
            tableView.refresh();
            tableView.getSelectionModel().clearSelection();
            DayData.calculateProgress();
            MainWindowController.updatePieChart(MainController.getData());
            initializeTrainingLabelText();
        }
    }
```

Ця функція перевіряє чи вибрано у поточний момент тренування із списку та, у випадку, якщо тренування вибрано, видаляє його із списку тренувань. Після видалення запускається процес оновлення таблиці із тренуваннями для видалення з неї зайвого елемента. Останнім етапом видалення є повний перерахунок значення прогресу, включаючи максимально допустиме значення для актуалізації даних.

3.2.2 Інтерфейс користувача

Після ініціалізації користувач бачить головний екран (рис. 3.2).

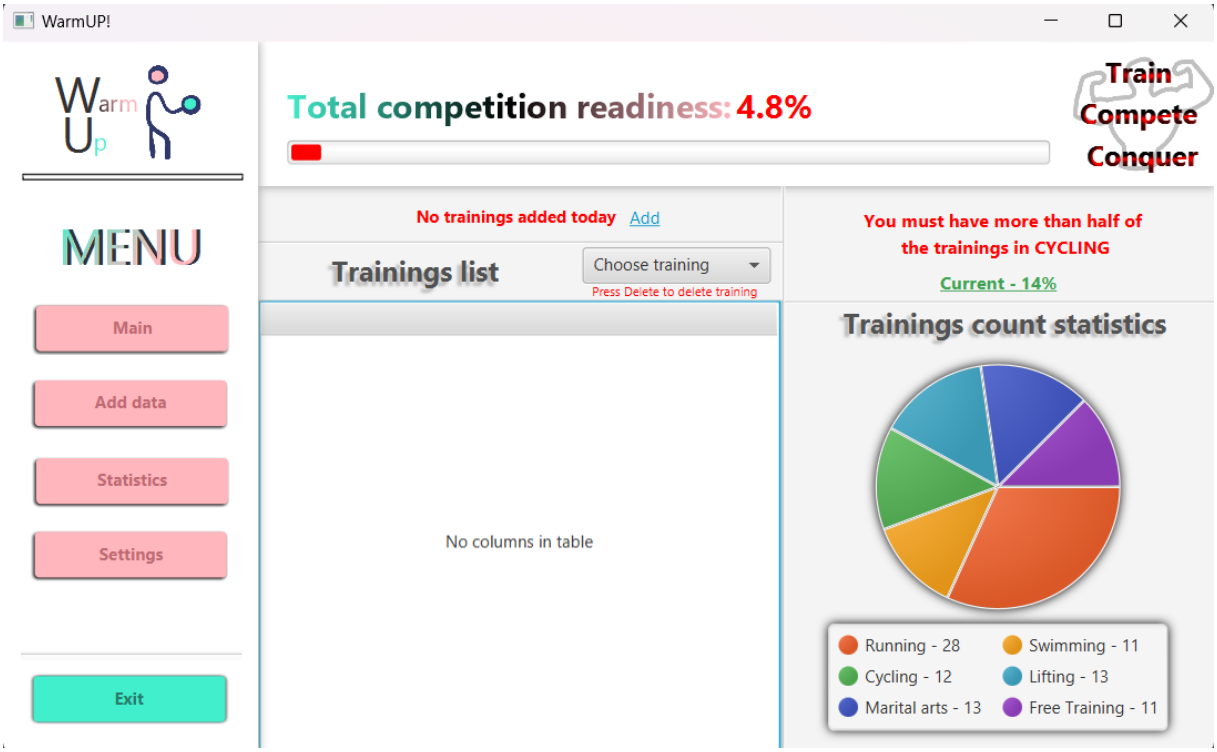


Рисунок 3.2 – Головний екран

Логотип та слоган було розроблено самостійно засобами графічної бібліотеки JavaFX (рис. 3.3).

а) б)

Рисунок 3.3 – Розроблені графічні елементи:

а) логотип;

б) слоган

На головному екрані, користувач обирає, список яких саме тренувань хоче побачити на екрані (рис. 3.4).

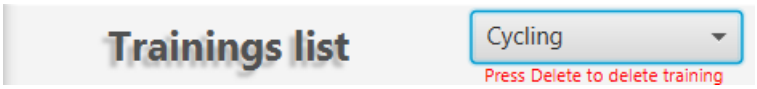


Рисунок 3.4 – Вибір типу тренувань для відображення

Після вибору в автоматичному режимі оновлюються дані та відображаються списком усі тренування обраного типу, які раніше вносив користувач до застосунку (рис. 3.5).

Duration (min)	Max BPM	Average BPM	Self-Feeling(1-5)
111	207	172	3
137	216	144	5
95	210	168	2
112	217	142	4
120	181	169	3
103	153	144	5
123	168	129	4
128	190	174	4
144	184	151	4
91	219	127	3
123	190	168	2
108	182	133	4

Рисунок 3.5 – Список тренувань

У правій частині головного екрану користувач бачить графік із розподілом кількості тренувань, які були внесені до програми, а також повідомлення, стосовно поточного відсоткового відношення тренувань (як було вказано раніше, у випадку, якщо кількість профільних тренувань менша, ніж 50% – максимальне значення прогресу буде обмежено за формулою, яка вказана у розділі 2). Цей випадок можна побачити на рисунку А.3, а на рисунку А.4 зображено зовнішній вигляд правої частини головного екрану у випадку, коли відсоток тренувань з бігу перевищує значення у 50%.

На екрані «Add Data» користувач має змогу додати нове тренування, яке після валідації даних автоматично записується до тренувань та відображається у списку на головному екрані (рис. 3.6).

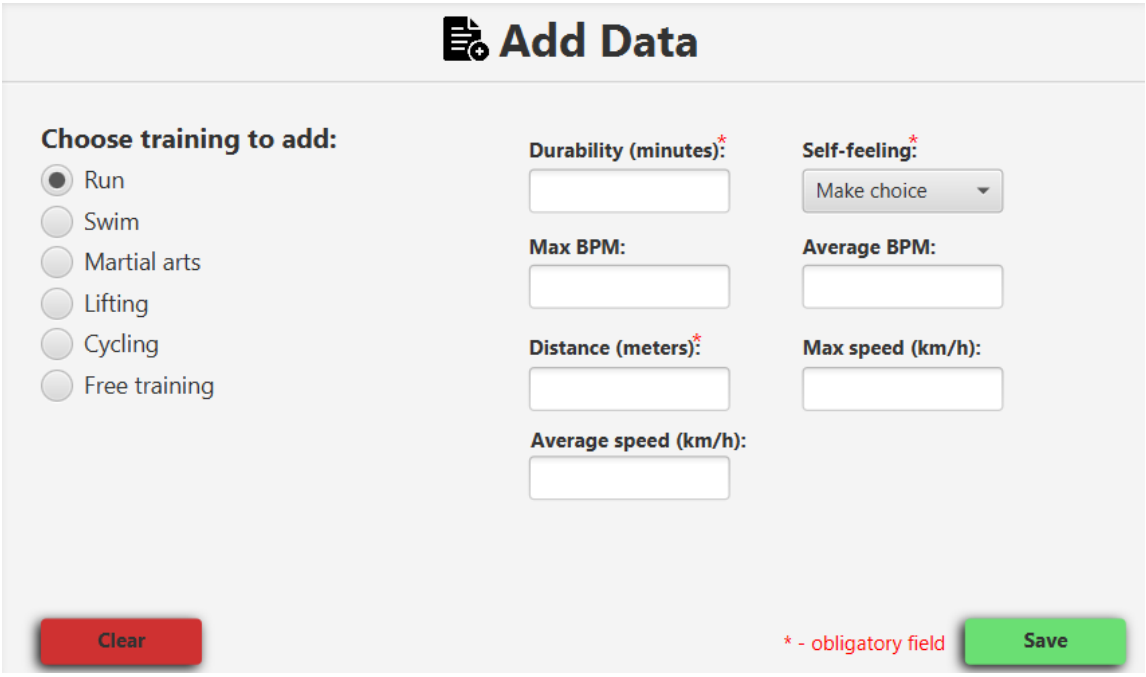


Рисунок 3.6 – Екран додавання тренування

Користувач має можливість додавати тренування за будь-яким напрямом, вказаним у лівій частині екрану. В залежності від обраного тренування список полів для заповнення буде відрізнятись (рис. А.5).

Також, необхідно звернути увагу, що не усі поля є обов’язковими. Обов’язкові поля зазначені червоною зірочкою. Але необхідно мати на увазі,

що незаповнені значення ініціалізуються нулем в автоматичному режимі, що вплине на значення прогресу в негативному ключі, тож за можливістю користувач має заповнити максимальну кількість полів. Таке поводження алгоритму зумовлено тим, що для повної оцінки продуктивності недостатньо двох-трьох значень, необхідно враховувати якомога більше факторів, тип паче, що спортсмен, який готується до змагань, має мати пристрій/ пристрої, що дозволять йому вимірювати пульс під час тренування, тощо.

У вікні «Statistics» користувачу доступний графік, на якому зазначено, як змінюється головна характеристика тренування, протягом підготовки до змагань. Наприклад, для бігу – це максимальна швидкість (рис. А.6).

Графік відображається в залежності від того, яке саме тренування обрано цільовим в налаштуваннях. Окрім графіку користувач зберігає звіт за тренуванням яке вибере в ChoiceBox (chose training type) при натисканні на кнопку «Save To File».

У випадку, якщо цільовим тренуванням обрано поле «None» (прогрес не рахується) у вкладці «Statistics» користувач побачить повідомлення, що тренування не обране, а на графіку буде відображатись зведення тривалості усіх тренувань, що раніше були додані (рис. А.7).

У вкладці «Settings» користувач має можливість обирати цільове тренування та вказувати, чи звіряти його результати із світовими рекордами (рис. 3.7), а у випадку співпадіння або перебільшення, значення прогресу автоматично буде встановлено на 100%, але тільки в тому випадку, якщо тренування в якому користувач досяг світового рекорду буде обрано як цільове (рис. А.8, рис. А.9).

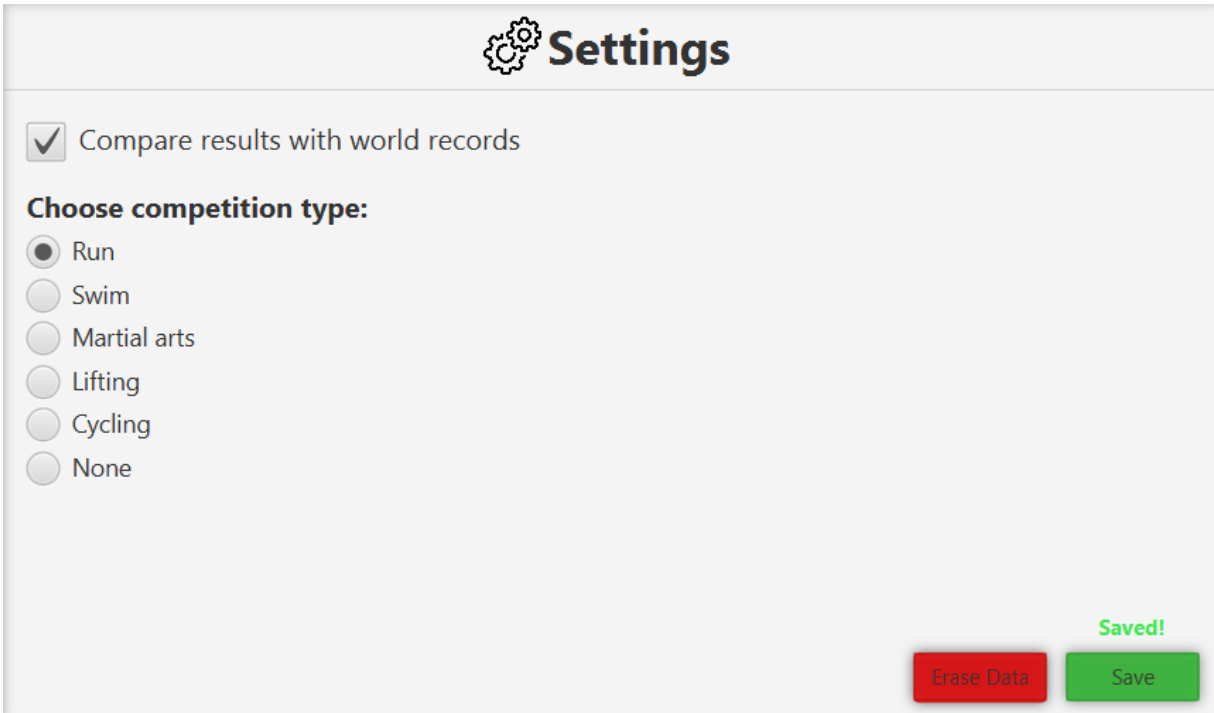


Рисунок 3.7 – Вікно налаштувань

Окрім налаштувань у цій вкладці є можливість стерти дані. При стиранні застосунок перезапуститься, а усі дані, що раніше були введені користувачем будуть видалені. Процес видалення буде детально описано у наступному підрозділі.

3.3 Інструкція користувача

Після ініціалізації застосунку користувач бачить перед собою головний екран (рис. 3.2). На головному екрані користувач має змогу відобразити усі тренування кожної категорії, що були внесені раніше у систему, вибираючи відповідний вид спорту в випадаючому списку.

Будь-яке тренування можливо видалити, натиснувши кнопку «delete» на клавіатурі. Тренування, що було видалено неможливо відновити, або додати знову(окрім сьогоднішнього тренування), оскільки реалізовано можливість

додавати тренування виключно поточною датою. Процес видалення зображено на рисунку А.10.

Усі дані застосунку зберігаються у файлі data.txt у бінарному вигляді в директорії src/main/resources/data/data.txt. Видалення файлу даних призведе до обнулення прогресу, видалення даних усіх тренувань та налаштувань. Для створення резервної копії достатньо скопіювати файл на диск, а у випадку необхідності відновлення – замінити файл у директорії зберігання на скопійований. Дані зберігаються у файл під час виходу із застосунку та завантажуються автоматично під час ініціалізації. Із метою збереження даних, що було додано під час поточної сесії користувач має закривати застосунок виключно кнопкою «Exit», або кнопкою закриття вікна на верхній рамці панелі. Закриття через диспетчер завдань та іншими схожими засобами призведе до втрати змін, які були внесені в поточній сесії.

Напис «You must have more than.....» у верхній правій частини головного екрану – це обмеження, що полягає в необхідності мати більше половини тренувань з виду спорту, який зазначено в написі, і, відповідно, обрано цільовим. Якщо таких тренувань менше – алгоритмом виставляється верхня межа значення «total competition readiness», яка дорівнюватиме (кількість тренувань з обраного спорту *0,4).

Пояснення обмеження:

– максимальний прогрес за день – 0,5%;
– за профільне тренування – 0,4%;
– за непрофільне 0,04%.

Тим самим 250 профільних тренувань * 0,4 = 100% прогресу, або ж 200 разів по 0,5% на день = 100%. Теоретично це можливо, проте 0,5% в день можна отримати додавши два профільні тренування або профільне і 3 звичайні тренування (0,4+0,04+0,04+0,04 = 0,52, в день максимум 0,5%, тому = 0,5%). Більш того, 0,4% – це максимальний бал, який виставляється, якщо користувач заповнить усі поля при додаванні тренування і вони будуть кращими за

еталонні значення. Звісно, таке навантаження є надто великим та можливе лише теоретично.

Окрім панелі тренувань на головному екрані користувач бачить коло-діаграму, на якій відображено відношення кількості усіх тренувань, що були внесені користувачем (рис. А.3) та інтерактивне поле, яке інформує користувача про внесення сьогоднішніх даних та пропонує натиснути на посилання для переходу до екрану додавання тренування (рис. 3.8), або інформує про додане тренування (рис. 3.9).

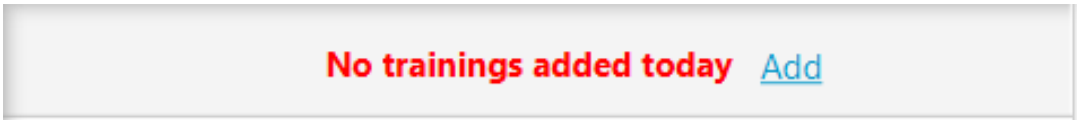


Рисунок 3.8 – Інтерактивне поле при не доданому тренуванні

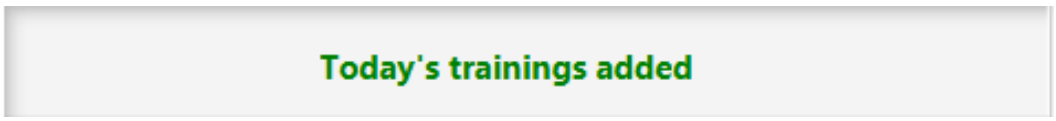
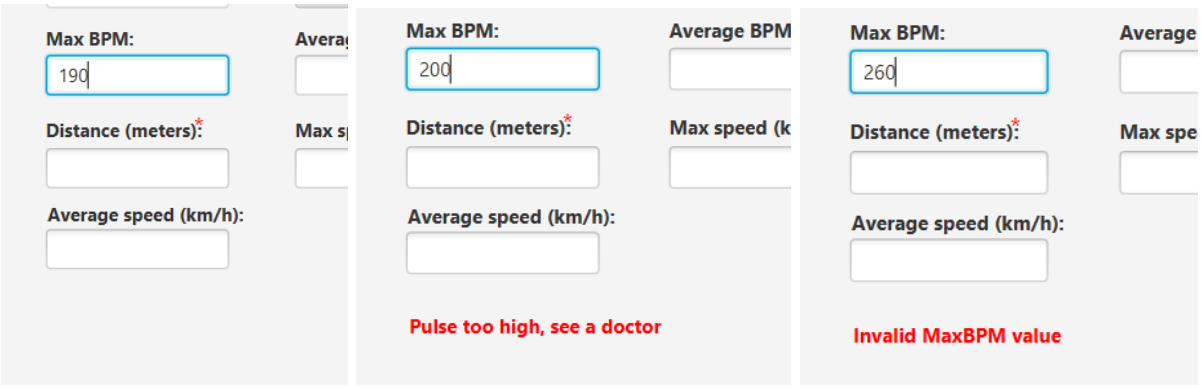


Рисунок 3.9 – Інтерактивне поле після додавання тренування

На екрані «Add data» користувач у лівій частині обирає тип тренування, яке збирається додати, після чого вводить дані у відповідні поля. Із референтними та еталонними значеннями можна ознайомитись у таблицях 2.1 і 2.2 відповідно. У випадку розбіжності даних із референтними – система валідації видасть повідомлення на екрані. У випадку успішного збереження користувач почує характерний сигнал та візуальне підтвердження збереження.

На рисунку 3.10 зображено процедуру валідації полів максимального та середнього пульсу. Валідація активується під час кожної дії із відповідним полем (введення, стирання, тощо.) та виводить поточний результат валідації.



а) б) в)

Рисунок 3.10 – Процедура валідації полів MaxBPM та AverageBPM:
а) референтне значення;
б) підвищене референтне значення;
в) нереферентне значення

У вікні статистики користувач завжди отримує графік того тренування, яке обрано цільовим. На екрані відображається зміна одного з ключових значень цього виду спорту протягом тренувань, що вводив користувач, або, якщо цільове тренування не обрано – користувач бачить графік із зміною тривалості усіх тренувань.

У вікні статистики також є можливість створити звіт, який буде збережено до файлу. Процес створення такого звіту продемонстровано на рисунку А.11.

У файл зберігаються середні значення та стандартні відхилення усіх полів, що містить тренування, а також сумарну кількість тренувань з цього виду спорту. Файл зберігається до директорії src/main/resources/data/Analysis/time_series_analysis.txt. Приклад інформації, що зберігається у файлі після створення звіту можна побачити на рисунку А.12.

Останнє вікно інтерфейсу – вікно «Settings». У налаштуваннях користувач обирає тип цільового тренування і активує (або не активує) функцію порівняння із світовими рекордами, яка полягає в автоматичному встановлені прогресу на рівні 100% у випадку досягнення спортсменом

результату, кращого за світовий рекорд (рекорди актуальні на 20.04.2023). Для збереження змін необхідно натиснути кнопку «Save», після чого відбудеться автоматичне перерахування поточного прогресу. Факт збереження супроводжується характерний звуком та візуальним сигналом.

Окрім зміни налаштувань, на екрані «Settings» користувач має можливість стерти дані застосунку (без можливості відновлення). Для стирання даних необхідно натиснути кнопку «Erase Data».

При спробі видалити дані програма видасть характерний звуковий сигнал, що зверне увагу користувача та попередить про невідворотність його дій (рис. 3.11).

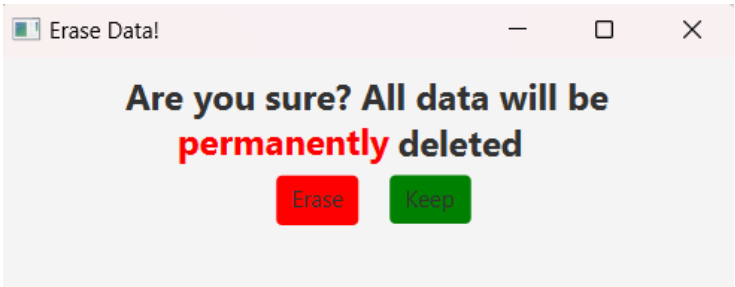


Рисунок 3.11 – Повідомлення про видалення

У випадку, якщо користувач дійсно має намір стерти дані – система запросить в нього ввести перевірочне словосполучення для підтвердження намірів (рис. 3.12).

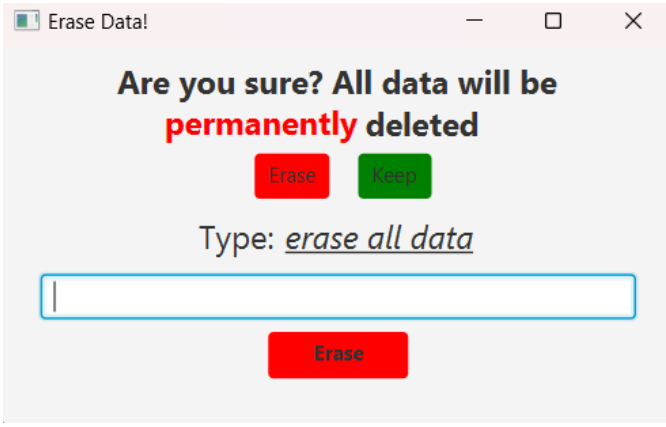
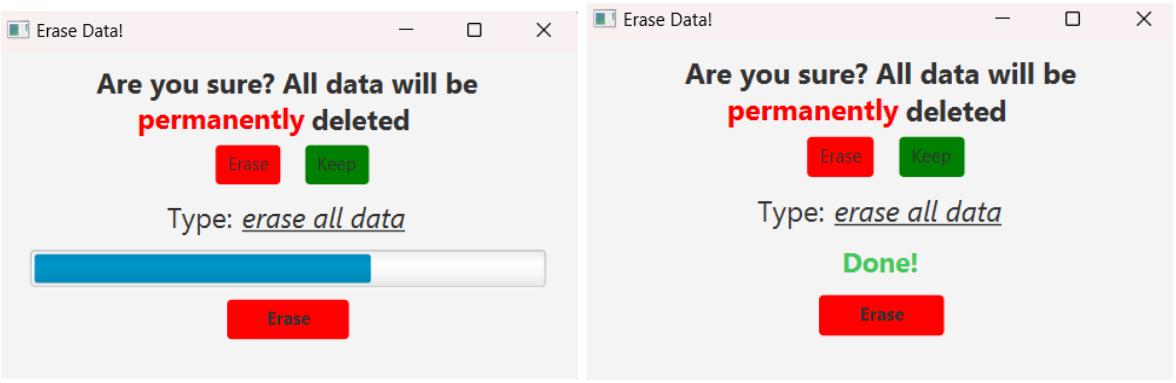


Рисунок 3.12 – Підтвердження намірів видалення даних

Тільки у випадку співпадіння програма почне стирання, про що повідомить користувача процесом видалення, після якого виведе підтвердження видалення (рис. 3.13).



а) б)

Рисунок 3.13 – Процес видалення даних:

а) видалення даних;

б) повідомлення про успішне видалення

Якщо видалення виявилось успішним – програма закриється автоматично, а при наступному старті її буде проініціалізовано із налаштуваннями за замовчуванням. Аналогічного результату можна дійти шляхом видалення файлу даних data.txt.

ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений і реалізований прототип сервісу для аналізу ефективності підготовки спортсменів до змагань з декількох видів спорту. Розроблений застосунок призначений для відстеження відсоткового тренувального прогресу. Він може бути корисним для тренерів, медичних працівників і власне атлетів з метою коригування тренувальних програм та навантажень.

Був проведений аналіз сервісів для аналізу ефективності підготовки, що дозволило виділити проблематику даної предметної області. Вже існуючі платформи надають лише функціонал для вираховування удосконалення у тренуваннях лише за певним параметром, але не мають змоги оцінити загальну картину прогресу в обраному виді спорту. Однак, найсуттєвішою проблемою цих рішень є те, що вони мають високу вартість підписки, в той час як у безкоштовній версії значно урізані функціональні можливості. Окрім того в деяких платформах є проблеми зі зрозумілістю інтерфейсів. Розроблена система спрямована на вирішення цих проблем, адже дозволяє отримати детальну статистику за тренувальним прогресом в лічені секунди. Правильність роботи застосунку було перевірено на тестових даних.

Результати роботи апробовано у вигляді тез доповідей під час 27-го Міжнародного молодіжного форуму «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ» [38].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Lieberman, D. (2020). Exercised: The science of physical activity, rest and health. Penguin UK.

2. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень.

3. Lyashenko, V., Mohammad, A., & Kobylin, O. (2015). Experiments with Fusion of Images with Use of Wavelet Transformation in Problems of the Text Information Analysis.

4. Sportlyzer. URL: https://www.sportlyzer.com/en/ (дата звернення 10.04.2023).

5. TrainingPeacks. URL: https://www.trainingpeaks.com/ (дата звернення 10.04.2023).

6. AthleteMonitoring. URL: https://www.athletemonitoring.com/ (дата звернення 11.04.2023).

7. Strava. URL: https://labs.strava.com/ (дата звернення 11.04.2023).

8. TimingEvents. URL: https://timingevents.com/ (дата звернення 13.04.2023).

9. Oleg, K., Sergii, M., & Mykhailo, S. (2017, October). Video Clustering via Multidimensional Time-Series Analysis. In *Proceedings of the 9th International Conference on Information Management and Engineering* (pp. 60-63). ACM.

10. Zain, S. B. M., & Zain, S. M. (2020). JAVA GUI BUNDLE pp. 4-23.

11. Loy, M., Eckstein, R., Wood, D., Elliott, J., & Cole, B. (2020). Java swing. " O'Reilly Media, Inc." pp. 4-10.

12. FRĄSZCZAK, D., BUGAJEWSKI, D., & MAGDZIARZ, K. (2022). Swing Dynamic GUI. Proceedings of the 40th International Business Information Management Association (IBIMA), 23, 24.

13. SWT on the Eclipse website. URL: https://www.eclipse.org/swt/ (дата звернення 19.04.2023).

14. Milošević, U., Spasić, M., Komarica, S., Kovačević, N., Mutavdžić, M., & Stanojević, M. Making NooJ Open Source and Platform Independent: Java Experience.

15. JavaFX documentation. URL: https://openjfx.io/ (дата звернення 21.04.2023).

16. Lyashenko V., Kobylin O., Selevko O. (2020) Wavelet Analysis and Contrast Modification in the Study of Cell Structures Images. International Journal of Advanced Trends in Computer Science and Engineering. 9(4). – 4701-4706.

17. Rosati, C., Weiss, G., Lund, S. K. K., & Smith, E. (2019). New Java frameworks for building next generation EPICS applications.

18. Alkhars, A., & Mahmoud, W. (2019). Cross-Platform Desktop Development (JavaFX vs. Electron) pp. 7-15.

19. Pedersen, S. B., & Jackson, S. (2019). Graphical User Interface Programming Challenges Moving Beyond Java Swing and JavaFX. Proc. ICALEPCS'19.

20. Kobylin, O. A., Gorokhovatskyi, V. O., Tvoroshenko, I. S., & Peredrii, O. O. (2020). The application of non-parametric statistics methods in image classifiers based on structural description components. Telecommunications and Radio Engineering, 79(10).

21. Rabotiahov, A., Kobylin, O., Dudar, Z., & Lyashenko, V. (2018, February). Bionic image segmentation of cytology samples method. In *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)* (pp. 665-670). IEEE.

22. Daniels, J. (2013). Daniels' running formula. Human Kinetics pp. 1-90.

23. Рожков, В. О. (2022). Особливості застосування навантажень під час занять здобувачів вищої освіти в секціях з легкої атлетики. Матеріали III Всеукраїнської науково-практичної конференції, С. 87.

24. Junaydulloevich, A. M., & Istamovich, A. K. (2021). Basic laws and descriptions of ways to develop technical skills in boxing. Web of Scientist: International Scientific Research Journal, 2(05), pp. 15-26.

25. Costill, D. L., Thomas, R., Robergs, R. A., Pascoe, D., Lambert, C., Barr, S., & Fink, W. J. (1991). Adaptations to swimming training: influence of training volume. Medicine & Science in Sports & Exercise, 23(3), 371-377.

26. Kobylin, O., Vyskrebentseva, S., & Petrova, R. (2019). Обробка даних, що містять пропуски в задачах кластеризації. *Системи управління, навігації та зв'язку. Збірник наукових праць*, *5*(57).

27. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2018, August). Representative Based Clustering of Long Multivariate Sequences with Different Lengths. In *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)* (pp. 545-548). IEEE.

28. Bodyanskiy, Y., Kobylin, I., Rashkevych, Y., Vynokurova, O., & Peleshko, D. (2018, February). Hybrid fuzzy-clustering algorithm of unevenly and asynchronously spaced time series in computer engineering. In *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)* (pp. 930-935). IEEE.

29. Bodyanskiy, Y., Vynokurova, O., Kobylin, I., & Kobylin, O. (2016). Adaptive fuzzy clustering of short time series with unevenly distributed observations in Data Stream Mining tasks. *Information Technology and Management Science*, *19*(1), 23-28.

30. Lyashenko, V., Matarneh, R., Kobylin, O., & Putyatin, Y. (2016). Contour detection and allocation for cytological images using Wavelet analysis methodology.

31. Гузій, О. В., Романчук, О. П., & Магльований, А. В. (2020). Сенсомоторні показники як критерії впливу інтенсивних фізичних навантажень на організм спортсмена. Український журнал медицини, біології та спорту, 5(3), С. 351-358.

32. Tursunov Oqil Amrilloevich. (2022). Methods of Increasing the Effectiveness of Training of Athletes Engaged in Primary Training Groups. Texas Journal of Multidisciplinary Studies, 6, pp. 134–139.

33. Kobylin, O., & Lyashenko, V. (2016). Contrast Modification as a Tool to Study the Structure of Blood Components.

34. Mashtalir, V., Ruban, I., & Levashenko, V. (Eds.). (2019). Advances in Spatio-Temporal Segmentation of Visual Data (Vol. 876). Springer Nature.

35. Kuzminska, O., Mazorchuk, M., Morze, N., & Kobylin, O. (2019, June). Digital learning environment of ukrainian universities: The main components to influence the competence of students and teachers. In *International Conference on Information and Communication Technologies in Education, Research, and Industrial Applications* (pp. 210-230). Springer, Cham.

36. Kinoshenko, D., Kobylin, O., Mashtalir, S., & Stolbovyi, M. (2019, March). Metric video retrieval speedup by irrelevant data elimination. In *Eleventh International Conference on Machine Vision (ICMV 2018)* (Vol. 11041, pp. 176-183). SPIE.

37. Gorokhovatskyi V., Tvoroshenko I., Kobylin O., and Vlasenko N. (2023) Search for visual objects by request in the form of a cluster representation for the structural image description, Advances in Electrical and Electronic Engineering, 21(1), pp. 19-27.

38. Гончарова О. В. Розробка застосунку для аналізу ефективності підготовки спортсменів: Радіоелектроніка та молодь у XXI столітті: 27 міжнародний молодіжний форум. (Харків 10-12 травня 2023 р.) Харків 2023. С. 60-61.