DOI 10.31891/2307-5732-2024-333-2-52
УДК 004.77:796.6

ЖУРАВСЬКА ІРИНА
Чорноморський національний університет ім. Петра Могили
https://orcid.org/0000-0002-8102-9854
e-mail: iryna.zhuravska@chmnu.edu.ua
ПИЛИПЧУК БОГДАН
Чорноморський національний університет ім. Петра Могили
https://orcid.org/0009-0003-3893-5311
e-mail: pylypchyk50@ukr.net

ТЕХНОЛОГІЇ ІНТЕРНЕТУ РЕЧЕЙ ДЛЯ МОНІТОРИНГУ ФІЗИЧНИХ НАВАНТАЖЕНЬ ВЕЛОСПОРТСМЕНІВ

*В роботі розкрито принципи застосування технології Інтернет речей у сучасному спорті з акцентом на організацію режиму тренувань та змагань у велоспорті. Ця технологія є надійним способом збору даних, однак передбачає вибір певних шкал оцінювання фізичних навантажень спортсменів та втручання тренера в коригування системи тренувань, обраної за допомогою кіберфізичної системи. Найбільш розповсюдженими серед спортсменів є шкала FTP та шкала Вамса (VAMs). Найбільш придатними для поєднання особливостей застосування технології IoT та зазначених шкал є давачі, вбудовані в спортивні гаджети виробництва компанії Garmin – годинники, фітнес-браслети, GPS-пристрої тощо, які фіксують різноманітні дані під час активності.. Доступ до необхідної інформації, яка зберігається вебзастосунком Garmin Connect, можливо отримати шляхом експорту з сайту файлу з даними у форматі *.csv, які підпадають під категорію Big Data. Для розуміння спортивного прогресу розроблено програмне забезпечення (ПЗ), що включає клієнтський застосунок та прошарок роботи з даними, реалізовані мовою програмування Python з використанням відповідних бібліотек. ПЗ націлено на обробку та візуалізацію статистичних даних задля полегшення оцінювання фізичної підготовки та корегування тренувань велоспортсменів. У запропонованому рішенні враховано, що технології IoT мають передбачати підключення різноманітних спеціалізованих модулів та давачів, у т. ч. власної розробки, а не тільки розповсюджених серед велоспортсменів датчиків фірми Garmin. Тому в основі розробленого ПЗ лежать такі можливості, як масштабованість та гнучкість, які у подальшому забезпечать обмін даними між давачами спорядження велоспортсменів та комп'ютерною системою в автоматичному режимі з використанням стандартних протоколів зв'язку відповідно до концепції IoT.*

Ключові слова: Інтернет речей, кіберфізична система, датчики, велоспорт, моніторинг фізичних навантажень, інформаційна система, Big Data.

ZHURAVSKA IRYNA, PYLYPCHUK BOHDAN
Petro Mohyla Black Sea National University

THE INTERNET OF THINGS TECHNOLOGIES FOR MONITORING THE PHYSICAL LOADS OF CYCLISTS

*The work reveals the principles of using the Internet of Things technology (IoT) in modern sports with an emphasis on the organization of training regimes and competitions in cycling sport. This technology is a reliable way of collecting data, but it involves the selection of certain scales for evaluating the physical loads of athletes and the correction of the coach in adjusting the training regimes selected with the help of the cyber-physical system. The most common among athletes are the FTP scale and the VAMs scale. The most suitable for combining the features of the application of IoT technology and the selected scales are sensors built into sports gadgets manufactured by Garmin – watches, fitness bracelets, GPS devices, etc., which record various data during cyclist activity. This data have stored by the Garmin Connect web application and can been classificated as Big Data. Access to the necessary information can be obtained by exporting a data file in *.csv format from the site. The architecture of the software includes a client application and a layer of working with data, implemented in the Python programming language using appropriate libraries. The software is aimed at processing and visualizing statistical data to facilitate the assessment of physical fitness and training adjustments of cyclists. The proposed solution takes into account that IoT technologies should provide for the connection of various specialized modules and sensors, including those of our own development, and not only Garmin sensors populared among cyclists. Therefore, the basis of the developed software are such possibilities as scalability and flexibility, which will in the future ensure the exchange of data between the embedded sensors into cyclists' equipment and the computer system in automatic mode using standard communication protocols in accordance with the concept of IoT.*

Keywords: Internet of Things, cyber physical system, sensors, cycling sport, physical loads monitoring, information system, Big Data.

Постановка проблеми

Впровадження технологій Internet of Things (IoT) в процес оцінювання фізичного навантаження велоспортсменів – один із пріоритетів сучасного розвитку цієї галузі. Інтернет речей в спорті спроможний забезпечити оптимізацію процесу на всіх етапах: від моменту передачі даних від спортсмена під час тренування до фіксації показників його стану під час змагань. За допомогою технології Інтернету речей показники фізичного стану спортсмена збирають через мережу підключених до інтернету пристроїв. Така мережа побудована на основі кіберфізичного підходу: пристрої, які мають вбудовані давачі, збирають інформацію і обмінюються нею між собою і хмарним сховищем, а також надають спортсмену команди щодо зміни режиму тренування або поведінки на трасі. Всі ці пристрої взаємопов'язані між собою, мають можливість зчитування та відображення параметрів, функцію програмування та ідентифікації, а також дозволяють виключити необхідність участі людини за рахунок використання інтелектуальних інтерфейсів. Але для розуміння спортивного прогресу необхідно розробити спеціалізоване програмне забезпечення (ПЗ), що оброблює велику кількість даних, які підпадають під категорію Big Data. До того ж, отримані дані

потребують якісної візуалізації для можливості швидкого коригування з боку тренера режиму тренувань / стратегії виступів на змаганні.

Об'єктом роботи (розробки) є процеси моніторингу фізичних навантажень велоспортсмена з використанням технологій IoT.

Предмет роботи – методи та засоби створення кіберфізичної системи, що виконує функції обробки даних з давачів, вбудованих у спорядження велоспортсмена, щодо фізичного навантаження та коригування плану тренувань велоспортсменів.

Мета – оцінювання фізичних навантажень велоспортсмена шляхом обробки статистичних даних, отриманих з сенсорів велоспорядження за допомогою технологій Інтернету речей, та подальша візуалізація даних засобами розробленого програмного забезпечення (ПЗ) для можливості коригування режимів тренувань велоспортсмена.

Аналіз останніх джерел

Спортивні технології відразу підхопили тенденцію до цифровізації та використання комунікаційних технологій. Особливість застосування IoT у спорті – технологія використовується на всіх рівнях підготовки спортсменів: від отримання його фізичних показників до планування його досягнень [1]. Ось приклади, як впроваджують Інтернет речей у велоспорт [2]:

давачі, встановлені з безпосереднім контактом з тілом спортсмена та на велосипеді, відправляють дані обліковій системі, що виключає людський фактор;

стан спортсмена та велосипеда перебуває під постійним контролем;

система датчиків дає змогу відстежувати й контролювати навантаження на спортсмена.

інформаційна система віддалено відстежує фізичні показники спортсмена та реєструє їх;

інформаційна система розраховує відповідні показники згідно з обраною шкалою оцінювання та на їх основі планує режим тренувань;

тренер віддалено керує обладнанням системи й оперативно коригує проблеми й усуває помилки у призначеному режимі тренувань.

Фізичні навантаження можуть бути різної інтенсивності та тривалості, тому контроль за ними є дуже важливим завданням для тренерів та спортсменів [3]. Для вирішення таких задач доцільно використання технологій Інтернету речей, які дозволяють реєструвати та передавати стандартними протоколами зв'язку параметри моніторингу навантажень велоспортсмена з давачів, які вбудовані в найбільш поширені для велоспорядження пристрої від компанії Garmin [4]. Вірний підхід до тренувань дозволяє досягнути максимальних результатів та зменшити ризик отримання травм.

Враховуючі те, що велоспорт – це загальний термін, який використовується для багатьох окремих змагальних дисциплін (велоспорт на треку, гірський велосипед, шосейний велоспорт, гонку на час, велокрос, велоспорт на гравії, вільний стиль BMX, перегони BMX, паравелоспорт, артистичний велоспорт тощо), для оцінювання фізичної підготовки велоспортсмена використовуються різні шкали оцінювання та параметри моніторингу, що спричиняє необхідність обробки великої кількості даних, які підпадають під категорію Big Data. Основними шкалами є [5; 6]:

- шкала FTP;
- шкала Вамса (VAMs);
- шкала Фостера (FTS);
- шкала Ратінга спроможності (RPE);
- шкала Борхардта.

Найбільш поширеними серед професійних велосипедистів є шкала FTP та Шкала Вамса (VAMs – скорочення від «Velocita Ascensionale Media», що означає середню підйомну швидкість італійською) [5]. Тому для обробки параметрів, використовуваних зазначеними шкалами, окремих досліджень потребують кінцеві функції для роботи зі шкалами. В таких функціях мають бути реалізовані всі необхідні методи для роботи з обраною шкалою, а саме функція оцінки фізичної підготовки спортсмена, візуалізація даних та їх експорт – все це передбачає програмну реалізацію методів опрацювання великих даних (Big Data). Big Data, зібрані від Garmin-давачів, – це дуже великі та складні набори даних, які не можуть бути оброблені або проаналізовані за допомогою традиційних технік обробки даних [7].

Три особливості великих даних, які мають бути враховані при їх обробці спеціалізованим ПЗ, це обсяг даних, швидкість їх реєстрації та їх різноманітність.

Виклад основного матеріалу

Проєкт розробки програмного забезпечення (ПЗ) націлений на створення інформаційної системи для обробки та візуалізації статистичних даних, спрямованої на полегшення оцінювання фізичної підготовки та корегування тренувань велоспортсменів. Застосовані бібліотеки Pandas та NumPy для ефективної обробки даних і вирішення недоліків аналогів, таких як відсутність візуалізації та глибокого аналізу фізичної підготовки.

Архітектура інформаційної системи включає клієнтський застосунок та прошарок роботи з даними, реалізовані мовою програмування Python з використанням відповідних бібліотек. Інтерфейс користувача відповідає стандартам UI та UX, сприяючи ефективній роботі тренерів та покращенню фізичної підготовки велоспортсменів через аналіз та візуалізацію статистичних даних.

На рис. 1 показано архітектуру програми для роботи з Big Data, яка адаптована до вимог програмного забезпечення з аналізу даних для оцінки фізичних показників [8].

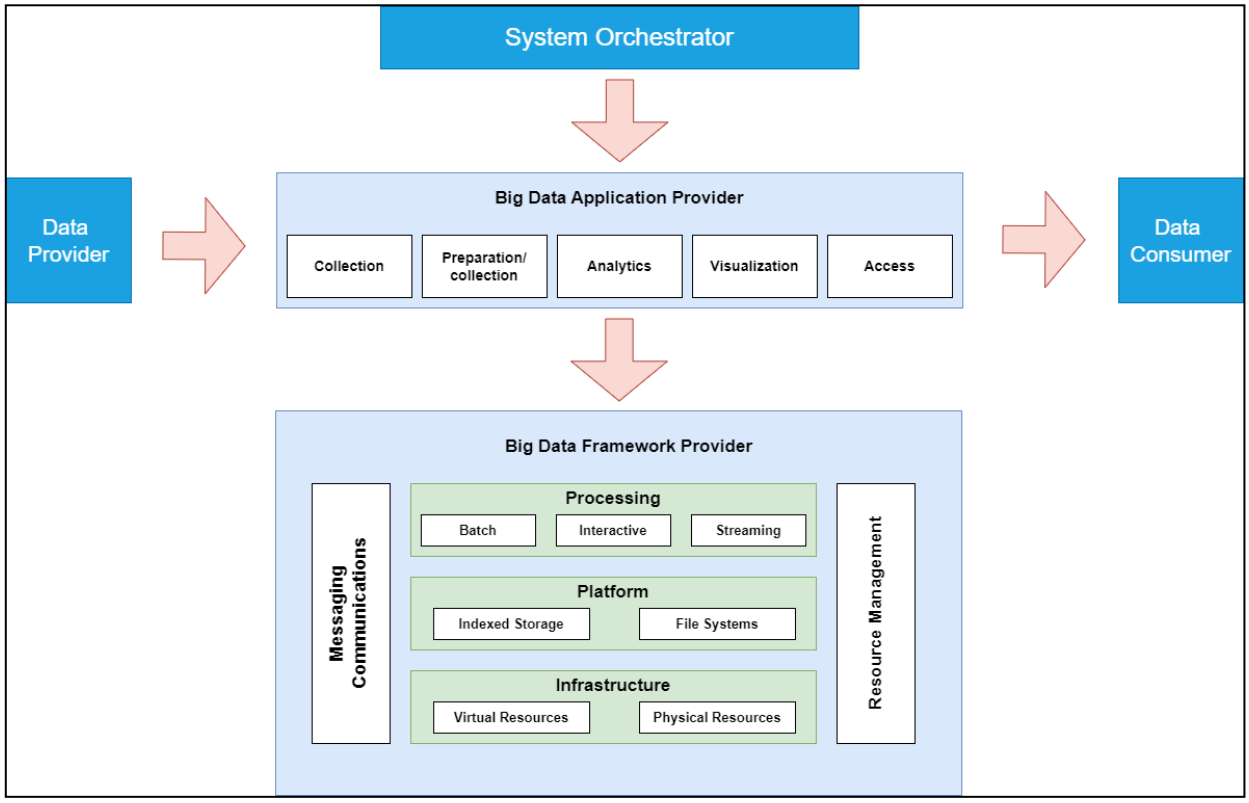


Рис.1. Архітектура ПЗ для роботи з Big Data

Опис головних компонентів схеми, наведеної на рис. 1:

System Orchestrator гарантує, що різні програми, дані та компоненти інфраструктури працюють разом;

Data Provider надає нові дані, що надходять з різних джерел, у систему великих даних для перетворення їх у оброблювані Data Set-и [9];

Big Data Application Provider містить бізнес-логіку та функціональні можливості, необхідні для перетворення даних у цінні знання за допомогою п’яти основних дій:

– збір;
– підготовка;
– аналітика;
– візуалізація;
– доступ.

Big Data Framework Provider має ресурси та сервіси для зберігання та обробки даних;

Data Consumer використовує інтерфейси сервісів, наданих Big Data Application Provider для доступу до необхідної інформації.

Garmin – відомий виробник спортивних гаджетів, таких як годинники, фітнес-браслети та GPS-пристрої, які фіксують різноманітні дані під час активності. Ці дані, такі як серцевий ритм, відстань, швидкість та багато інших параметрів, збираються і зберігаються вебзастосунком Garmin Connect. Завдяки обладнанню Garmin та вебзастосунку Garmin Connect можливо отримати доступ до необхідної інформації шляхом експорту файлу з даними з сайту, який після обробки та аналізу розробленим ПЗ допоможе в розумінні свого спортивного прогресу та прийнятті більш обґрунтованих рішень щодо покращення своїх тренувань та здоров'я.

Також у майбутньому планується відійти від залежності у використанні модулів від побічних виробників обладнання та розробити власні спеціалізовані модулі та датчики. Тому у основі розроблюваного ПЗ лежать такі можливості, як масштабованість та гнучкість, які у подальшому допоможуть розвивати та оновлювати функціонал та можливості застосунку й проєкту в цілому.

Отримані дані є файлами з розширенням *.csv* – тут зібрані дані велосипедиста протягом тренувального сезону (12 місяців). В даному файлі записані показники: Activity Type, Date, Favorite, Title, Distance, Calories, Time, Avg HR, Max HR, Aerobic TE, Avg Speed, Max Speed, Total Ascent, Total Descent, Avg Stride Length, Avg Vertical Ratio, Avg Vertical Oscillation, Avg Ground Contact Time, Avg Bike Cadence, Max Bike Cadence, Normalized PowerB (NPB), Training Stress ScoreB, Max Avg Power (20 min), Avg Power, Max Power, Grit, Flow, Total Strokes, Avg Swolf, Avg Stroke Rate, Total Reps, Dive Time, Min Temp, Surface Interval, Decompression, Best Lap Time, Number of Laps,Max Temp, Avg Resp, Min Resp, Max Resp, Moving Time, Elapsed Time, Min Elevation, Max Elevation.

Pandas – це бібліотека для аналізу, обробки і маніпуляції з даними [10].

Matplotlib – це потужна бібліотека для створення графіків у Python, яка дозволяє створювати статичні, анімовані та інтерактивні візуалізації [11]. Вона легко інтегрується з NumPy та різними графічними бібліотеками, такими як Tkinter або PyQt [12–14]. За допомогою цієї бібліотеки можна побудувати різноманітні типи графіків.

NumPy – це бібліотека, яка розширює стандартні структури даних шляхом додавання багатовимірних масивів та матриць, а також надає велику колекцію математичних функцій для роботи з цими масивами [12].

При розробці застосунків на мові програмування Python, правильна архітектура відіграє важливу роль у створенні ефективного і легкозмінного програмного рішення.

Основна ідея полягає в тому, щоб розділити функціональність програми на окремі модулі, які виконують специфічні завдання (рис. 2). Це дозволяє покращити читабельність, підтримку і розширюваність проєкту, враховуючи всі його потоки даних (рис. 3).

```
def plot_distance_by_month():...

def FTP():...

def export_to_excel(data):...


def VAMs():...

def export_to_excel(data):
    save_path = filedialog.asksaveasfilename(defaultextension='.xlsx')
    if save_path:
        with ExcelWriter(save_path) as writer:
            data.to_excel(writer, index=False)
        messagebox.showinfo("Export Successful", "Data exported to Excel successfully.")
```

Рис. 2. Розбиття коду на модулі (функції)

Саме таких принципів та шаблону структури проєкту було дотримано під час розробки застосунку з моніторингу фізичних навантажень велоспортсмена.

Основною задачею даного прошарку є різноманітна робота та оперування даними, аби не ускладнювати розробку інтерфейсу зайвою логікою. Вирішення досягається шляхом створення функцій – методів для кожного методу оцінки фізичної підготовки, які б містили усі необхідні методи для читання, запису, перетворення даних тощо. Загальна логіка даного шару прописана в базовому класі і перевизначається у дочірніх класах. При цьому можна легко створити різні допоміжні функції всередині похідних методів, які б були локально корисними, без залежності від зовнішніх факторів.

Першою і основною функцією є читання даних. Якщо сталася помилка при завантаженні файлу (наприклад, файл не має розширення *.csv* або має неправильний формат), з'являється діалогове вікно повідомлення про помилку *(messagebox.showerror)* з повідомленням «Помилка» та текстом «Не вдалося завантажити файл. Перевірте, чи файл має розширення *.csv* та спробуйте ще раз».

Наступною важливою функцією є *def format_file*. Дана функція призначена для форматування файлу з даними. Вона приймає один аргумент – об'єкт *DataFrame* під назвою *df*. Видаляється певний набір стовпців з датафрейму *df*, які не потрібні для реалізації методів оцінки фізичної підготовки. Стовбці, що видаляються, мають такі назви: *Activity Type, Favorite, Aerobic TE, Avg Stride Length, Avg Vertical Ratio, Avg Vertical Oscillation, Avg Ground Contact Time, Grit, Flow, Total Strokes, Avg. Swolf, Avg Stroke Rate, Total Reps, Dive Time, Min Temp, Surface Interval, Decompression, Best Lap Time, Number of Laps, Max Temp, Avg Resp, Min Resp, Max Resp, Moving Time, Elapsed Time, Min Elevation.* Ці стовбці перераховані у вигляді списку рядків і виключаються з датафрейму. В результаті цих операцій датафрейм *df* змінюється, що пришвидшує роботу застосунку.

Ключовими ж являються кінцеві функції для роботи зі шкалами. Нижче наведено код функції для розрахунку фізичної підготовки за **шкалою FTP**. В ній реалізовані всі необхідні методи для роботи з даною шкалою, а саме функція оцінки фізичної підготовки спортсмена, візуалізація даних та їх експорт. Дана функція FTP() виконує розрахунок FTP (функціональний поріг) на основі даних з файлу, вибраного користувачем (рис. 4), та відображає результати розрахунку у вікні програми.

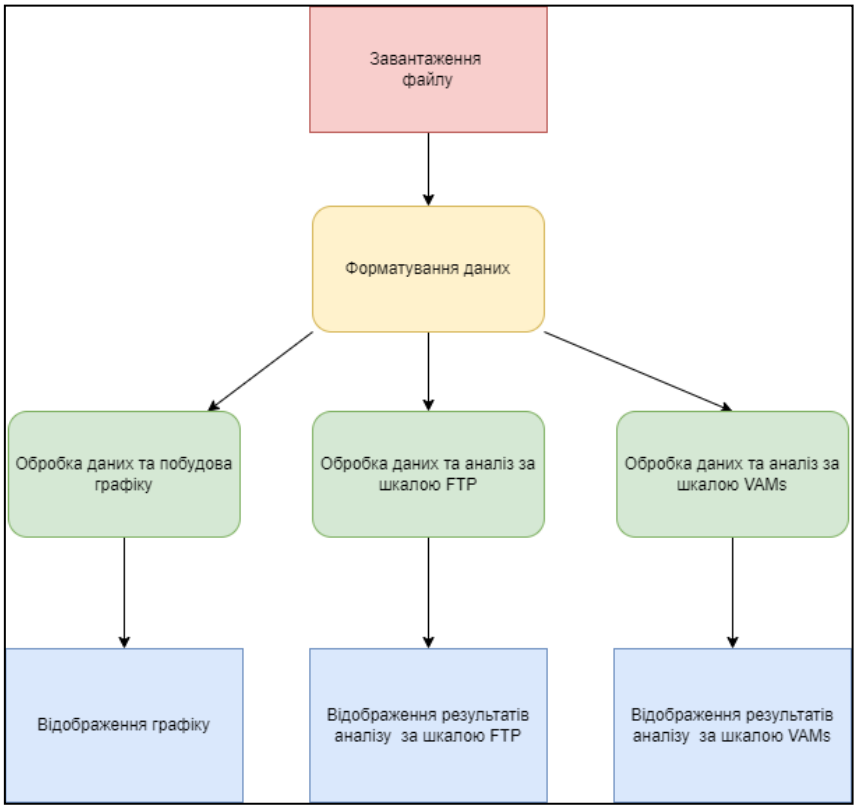


Рис. 3. Діаграма потоків даних

```
def FTP():
    """
    Фунуція обробки даних для аналізй за шкалою FTP
    """
    data = pd.read_csv(file_path)
    data = format_file(data)
    data['Avg Power'] = data['Avg Power'].str.replace(',', '').astype(float)  # Перетворення на float
    data['FTP'] = data['Avg Power'] / 60 * 0.95
    # Додавання нового стовбця для рейтингу FTP на основі значення FTP
    conditions = [
        (data['FTP'] < 2.23),
        (data['FTP'] >= 2.23) & (data['FTP'] <= 2.78),
        (data['FTP'] >= 2.79) & (data['FTP'] <= 3.92),
        (data['FTP'] >= 3.93) & (data['FTP'] <= 5.04),
        (data['FTP'] > 5.04)
    ]
    ratings = ['Непідготовлений', 'Задовільно', 'Добре', 'Відмінно', 'Чудово']
    data['Rating'] = np.select(conditions, ratings, default='Unknown')

    # Створення нового вікна верхнього рівня для результатів розрахунку FTP
    ftp_window = tk.Toplevel(root)
    ftp_window.title("FTP Calculation Results")

    # Створення рамки у вікні верхнього рівня за допомогою менеджера геометрії сітки
    table_frame = tk.Frame(ftp_window)
    table_frame.grid(padx=10, pady=10)

    # Створення віджету таблиці та відображення результату обчислення FTP
    table = Table(table_frame, dataframe=data[['Date', 'FTP', 'Rating']], showtoolbar=True, showstatusbar=True)
    table.grid(row=0, column=0, padx=10, pady=10)
    table.show()

    # Створення кнопки експорту
    export_button = tk.Button(ftp_window, text="Export to Excel", command=lambda: export_to_excel(data))
    export_button.grid(row=1, column=0, pady=10)

def export_to_excel(data):
    save_path = filedialog.asksaveasfilename(defaultextension='.xlsx')
    if save_path:
        with ExcelWriter(save_path) as writer:
            data.to_excel(writer, index=False)
        messagebox.showinfo("Export Successful", "Data exported to Excel successfully.")
```

Рис. 4. Лістинг функції розрахунку FTP (функціонального порогу)

Основні дії функції:

1) зчитується файл з даними за допомогою функції *pd.read_csv()* і зберігається у змінну *data;*

2) викликається функція *format_file(data),* яка форматує дані, виконуючи певні операції зі стовбцями датафрейму *data*. Оновлений датафрейм знову зберігається у змінну *data*;

3) виконується обробка стовбця даних. Кома в значеннях замінюється на порожній рядок за допомогою методу *str.replace(),* а потім значення перетворюються на числовий тип даних *(float)* за допомогою методу *astype()* та виконується збереження значень;

4) розраховується значення FTP;

5) додається новий стовбець «Rating» для оцінки FTP на основі значення FTP. Використовується функція *np.select()* для вибору відповідного рейтингу залежно від діапазону FTP. Рейтинги та умови розрахунку відповідають діапазонам FTP, і значення «Unknown» використовується як значення за замовчуванням, якщо жодна з умов не співпадає;

6) створюється нове вікно верхнього рівня *(tk.Toplevel)* для відображення результатів розрахунку FTP;

7) створюється фрейм всередині вікна за допомогою менеджера геометрії *'grid' (table_frame.grid(padx=10, pady=10));*

8) створюється віджет таблиці *(Table)* з бібліотеки *tkinter* у фреймі *table_frame* і відображаються результати розрахунку FTP з обраними стовпцями «Date», «FTP» та «Rating». Таблиця показує панель інструментів та рядок стану. Розташовується за допомогою *table.grid().* Відображається таблиця за допомогою *table.show();*

9) створюється кнопка експорту до Excel (Export to Excel), яка викликає функцію *export_to_excel(data)* при натисканні. Кнопка розташовується у вікні FTP за допомогою *export_button.grid().*

Функція для роботи зі **шкалою VAMs** є майже аналогічна, але з певними незначними особливостями. Дана функція VAMs() виконує обробку даних для аналізу за шкалою VAMs (Vertical Ascent Meters per Second) на основі даних з файлу, вибраного користувачем. Результати обчислень відображаються у вікні програми. Основні дії функції:

1) зчитується файл з даними за допомогою функції pd.read_csv() і зберігається у змінну data;

2) викликається функція *format_file(data)*, яка форматує дані, виконуючи певні операції зі стовпцями датафрейму *data*. Оновлений датафрейм знову зберігається у змінну *data*;

3) виконується обробка стовпця «*Max Elevation*». Кома в значеннях замінюється на порожній рядок за допомогою методу *str.replace()*, а потім значення перетворюються на числовий тип даних (*float*) за допомогою методу *astype()*. Результати зберігаються назад у стовпець *«Max Elevation»*;

4) стовпець *«Time»* перетворюється на тип *datetime*, використовуючи *pd.to_datetime()*, з форматом часу *«%H:%M:%S»*. Потім значення округлюються до найближчої години за допомогою методу *round()* та *dt.time*. Оновлені значення зберігаються назад у стовпець *«Time»;*

5) створюється стовпець *«Seconds»*, де обчислюється тривалість часу у секундах на основі значення стовпця *«Time»*. Для цього використовується метод *apply()* та *datetime.timedelta().* Результати зберігаються у стовпець *«Seconds»;*

6) обчислюється значення VAMs (метри вертикального підйому на секунду) шляхом ділення значення *«Max Elevation»* на значення *«Seconds».* Результати зберігаються у новому стовпці *«VAMs»;*

7) створюється нове вікно верхнього рівня *(tk.Toplevel)* для відображення результатів розрахунку VAMs;

8) створюється фрейм *table_frame* у вікні верхнього рівня, використовуючи менеджер геометрії *grid();*

9) створюється віджет таблиці *(Table)* та відображаються результати обчислення VAMs за допомогою передачі відповідних стовпців датафрейму *data* у параметр *dataframe* конструктора *Table*. Віджет таблиці розміщується у фреймі *table_frame* за допомогою *table.grid().* Панель інструментів та рядок стану також відображаються, оскільки параметри *showtoolbar=True* та *showstatusbar=True*;

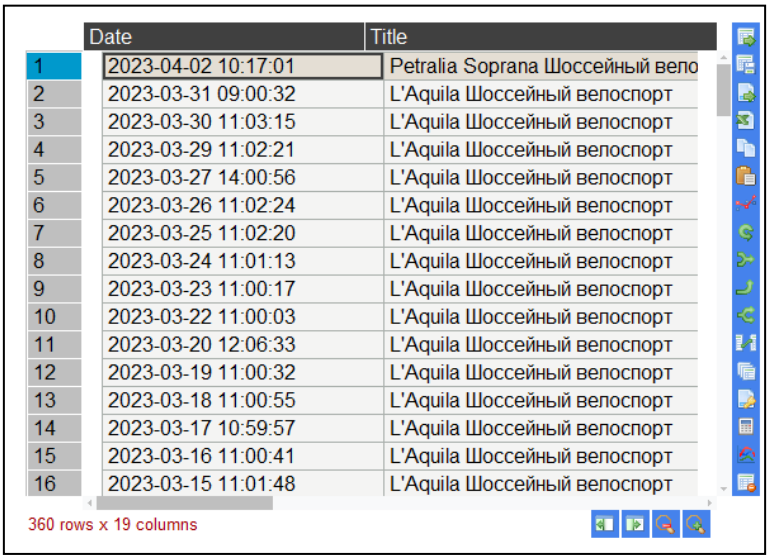
10) викликається метод *table.show()*, щоб відобразити таблицю з результатами;

11) створюється кнопка експорту до Excel з текстом «Export to Excel», яка викликає функцію *export_to_excel(data)* при натисканні. Кнопка розташовується у вікні *vams_window* за допомогою *export_button.grid().* Функція *export_to_excel(data)* виконує експорт даних у форматі Excel *(*.xlsx)*. Функція відкриває діалогове вікно для вибору шляху збереження файлу за допомогою *filedialog.asksaveasfilename().* Якщо шлях вибраний, то дані зберігаються у файл Excel за допомогою *data.to_excel().* Після успішного експорту виводиться повідомлення про успішне експортування за допомогою *messagebox.showinfo().*

Також було реалізовано метод вибору шкали оцінювання як варіанту списку з *combobox* (рис. 5).

```
def ok_button():
    if variable.get() == "Графік пройденої відстані по місяцях":
        plot_distance_by_month()
    elif variable.get() == "Шкала FTP":
        FTP()
    elif variable.get() == "Шкала VAMs":
        VAMs()
    else:
        root.destroy()
ok_button = tk.Button(frame, text="Розрахувати", command=ok_button)
ok_button.grid(row=0, column=2, padx=15, pady=15)
```

Рис. 5. Лістинг коду вибору варіанту списку з combobox



	Date	Title
1	2023-04-02 10:17:01	Petralia Soprana Шоссейный вело
2	2023-03-31 09:00:32	L'Aquila Шоссейный велоспорт
3	2023-03-30 11:03:15	L'Aquila Шоссейный велоспорт
4	2023-03-29 11:02:21	L'Aquila Шоссейный велоспорт
5	2023-03-27 14:00:56	L'Aquila Шоссейный велоспорт
6	2023-03-26 11:02:24	L'Aquila Шоссейный велоспорт
7	2023-03-25 11:02:20	L'Aquila Шоссейный велоспорт
8	2023-03-24 11:01:13	L'Aquila Шоссейный велоспорт
9	2023-03-23 11:00:17	L'Aquila Шоссейный велоспорт
10	2023-03-22 11:00:03	L'Aquila Шоссейный велоспорт
11	2023-03-20 12:06:33	L'Aquila Шоссейный велоспорт
12	2023-03-19 11:00:32	L'Aquila Шоссейный велоспорт
13	2023-03-18 11:00:55	L'Aquila Шоссейный велоспорт
14	2023-03-17 10:59:57	L'Aquila Шоссейный велоспорт
15	2023-03-16 11:00:41	L'Aquila Шоссейный велоспорт
16	2023-03-15 11:01:48	L'Aquila Шоссейный велоспорт

360 rows x 19 columns

Рис. 6. Фрейм з табличною візуалізацією даних

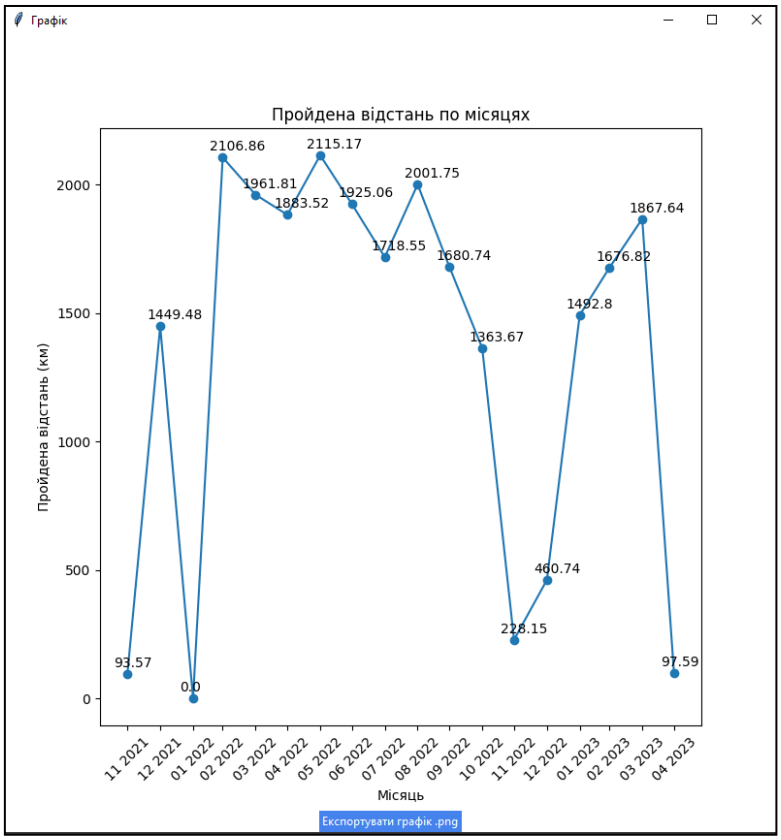


Рис. 7. Фрейм з візуалізацією даних

Дана функція необхідна для того, щоб можна було за значенням із combobox, що використовується для обрання методу оцінки, виконати обраний метод.

Наступним фреймом є таблична візуалізація (за можливістю) обраних даних, зображена на рис. 6. Фрейм з візуалізацією табличних даних, обраних користувачем, виконується одразу після вибору файлу з даними для аналізу.

Наступним фреймом є візуалізація даних з попереднього фрейму, зображена на рис. 7. Під зображенням реалізовано меню для роботи з ним, головною функцією в якому є саме можливість експорту зображення.

Візуалізація відбувається завдяки класам *Figure* та *FigureCanvasTkAgg* бібліотеки *matplotlib*. Перший відповідає за власне фігуру та її зображення, а другий дозволяє отримати віджет для подальшого відображення в фреймі. Приклад суміщення роботи цих класів для отримання результату, зображеного на рис. 11, наведено на лістингу нижче. Також на ньому вказано, як необхідно додавати зображення до фрейму. Відбувається це через отримання віджету завдяки функції *get_tk_widget()*, наведеної на рис. 8.

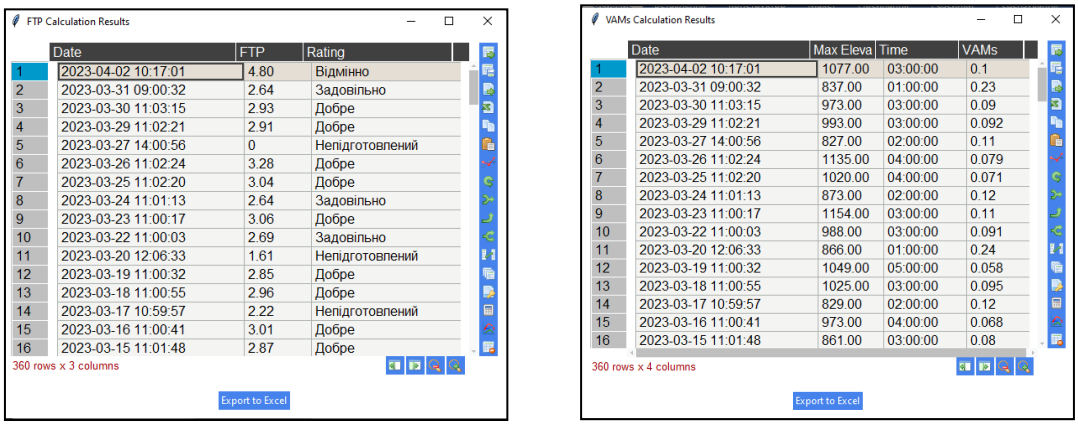
```
# Створюємо нове вікно (frame) для відображення графіка
   graph_window = tk.Toplevel(root)
   graph_window.title('Графік')

   # Створюємо об'єкт FigureCanvasTkAgg для відображення графіка в новому вікні
   canvas = FigureCanvasTkAgg(fig, master=graph_window)
   canvas.draw()
   canvas.get_tk_widget().pack()
   # Додавання кнопки експорту графіку
   def export_graph():
      save_path = filedialog.asksaveasfilename(defaultextension='.png')
      if save_path:
         fig.savefig(save_path)
         print("Графік експортовано у форматі .PNG")

   export_button = tk.Button(master=graph_window, text="Експортувати графік .png", command=export_graph)
   export_button.pack()
```

Рис. 8. Лістинг коду функції *get_tk_widget()*

Головними фреймами є фрейми з результатами оцінювання спортсмена за різними шкалами, як показано на рис. 9.



а) **б)**

Рис. 9. Фрейм з результатами оцінювання: а – за шкалою FTP; б – за шкалою VAMs

Можна було також зробити глобальний словник, але тут є певна проблема, що не дозволить реалізувати все через те, що елементи на початку інтерпретованого файлу нічого не знають про елементи після них. Це можна виправити імпортом анотацій з модулю __future__, але це спрацює лише всередині функцій. Поза ними дані маніпуляції будуть неуспішними і IDE буде видавати помилку, оскільки посилання на функцію чи клас є невизначеним.

Висновки

У роботі розглянуто проблеми та можливості застосування технології Інтернету речей (IoT) для збору даних з давачів, вбудованих у спорядження спортсмена з акцентом на проблеми сучасних видів велоспорту. передбачає вибір певних шкал оцінювання фізичних навантажень спортсменів Розроблено програмне забезпечення (ПЗ), що обробляє велику кількість даних, які підпадають під категорію Big Data. Архітектура ПЗ включає клієнтський застосунок та прошарок роботи з даними, реалізовані мовою програмування Python з використанням відповідних бібліотек. Проаналізовано особливості опрацювання отриманих даних із застосуванням шкал FTP та VAMs оцінювання фізичних навантажень спортсменів.

Розроблене ПЗ націлено на обробку та візуалізацію статистичних даних задля полегшення оцінювання фізичної підготовки та корегування тренувань велоспортсменів. У запропонованому рішенні враховано, що технології IoT мають передбачати підключення різноманітних спеціалізованих модулів та давачів, у т. ч. власної розробки, а не тільки розповсюджених серед велоспортсменів датчиків фірми Garmin. Тому в основі розробленого ПЗ лежать такі можливості, як масштабованість та гнучкість, які у подальшому забезпечать обмін даними між давачами велоспортсменів та комп'ютерною системою в автоматичному режимі з використанням стандартних протоколів зв'язку відповідно до концепції IoT.

Розробка має практичне значення для тренерів, дозволяючи швидше оцінювати фізичну підготовку спортсменів та поліпшувати коригування відновлювальних процесів.

References

1. Du M., Liu Z. Efficient feature recognition and matching technology for IoT-enabled sports training. *Internet Technology Letters*. Nov. 2023. DOI: 10.1002/itl2.490.

2. Thomas Fallon T., Heron N. A systematic review protocol of injuries and illness across all the competitive

cycling disciplines : preprint. Jan. 2024. 9 p. DOI: 10.21203/rs.3.rs-3909153/v2.

3. Municio E., Daneels G., De Brouwer M., Ongenae F., De Turck F. Continuous athlete monitoring in challenging cycling environments using IoT technologies. *IEEE Internet of Things Journal.* Sep. 2019. Vol. 99. P. 1–14. DOI: 10.1109/JIOT.2019.2942761.

4. Classification of Garmin heart rate monitors: analysis, tips, recommendations. URL: https://prostobzor.com/garmin-hrms-rating/

5. Global Recommendations on Physical Activity for Health, World Health Organization. Geneva, Switzerland, 2009. URL: http://www.who.int/ncds/prevention/physical-activity/en/

6. FTP (Functional Threshold Power). URL: https://velojournal.net/kak-izmerit-progress-v-velosporte-ftp-test-i-drugie-metody

7. Nunes F. P., Domingues P., Frade M. Post-mortem digital forensic analysis of the Garmin Connect application for Android. *Forensic Science International Digital Investigation*. Sep. 2023. Vol. 47. DOI: 10.1016/j.fsidi.2023.301624.

8. Big data architectures. URL: https://learn.microsoft.com/en-us/azure/architecture/data-guide/big-data/.

9. Data set. URL: https://www.techtarget.com/whatis/definition/data-set

10. Getting started – Pandas documentation. URL: https://www.w3schools.com/python/pandas/default.asp

11. Matplotlib: Visualization with Python. URL: https://matplotlib.org/

12. NumPy Introduction. URL: https://www.w3schools.com/python/numpy/numpy_intro.asp .

13. Tkinter – Interface Python до Tcl/Tk. URL: https://docs.python.org/uk/3/library/tkinter.html.

14. PyQt Documentation. URL: https://doc.qt.io/qtforpython-6/