

УДК 004.794

Льовкін В.М.<sup>1</sup>, Радєв П.А.<sup>2</sup>

<sup>1</sup> канд. техн. наук, доц. ЗНТУ

<sup>2</sup> студ. гр. КНТ-136сп ЗНТУ

## **РИЗИК-ОРІЄНТОВАНИЙ ПІДХІД ДО РОЗРОБКИ ПРОГРАМНОЇ СИСТЕМИ ДЛЯ АВТОМАТИЗАЦІЇ КЕРУВАННЯ ТЕПЛИЧНИМИ КОМПЛЕКСАМИ**

Одним з очевидних світових трендів останніх років є «інтернет речей» (IoT), за якого безліч різноманітних пристроїв різного розміру та призначення продукують величезну кількість інформації про навколишнє середовище, що при доцільному використанні може приносити велику користь та автоматизувати процеси, задля яких раніше вимагалися значні витрати робочої сили, таким чином приносячи економічний ефект [1]. Однією з багатьох сфер людської діяльності, у яких доцільно застосовувати подібні технології, є сільське господарство. Наприклад, детальний моніторинг параметрів оточуючого середовища може попередити велику кількість людських помилок, що стаються через брак інформації.

Новизна розробленої системи полягає в тому, що на відміну від відомих агрегаторів даних IoT вона спрямована не тільки на збір даних, а й їх зворотнє відправлення (наприклад, керуючих сигналів), а також в тому, що на відміну від вже згаданих агрегаторів мікрокомп'ютери (МК) мають змогу самостійно реєструвати себе у системі, без необхідності втручання користувача у цей процес.

Користувальська частина додатку є браузерно-орієнтованою, отже може бути однаково успішно запущена на будь-якому пристрої. Інтерфейс панелі керування є адаптивним відносно розмірів екрану та інтуїтивно зрозумілим (рис. 1, 2). Головними принципами розробки були наступні:

- серверний додаток повинен працювати з різними видами МК (за умови, що інформація, яка від них надходить, відповідає певним вимогам);
- серверний додаток повинен працювати з великою кількістю пристроїв та запитів без впливу на загальну швидкість роботи;

– серверний додаток та база даних повинні бути легко переносними між пристроями та платформами без змін в архітектурі, щоб у випадку відмови сервера мати змогу перейти на резервний без значних втрат у часі та швидкості;

– додаток для МК має бути відмовостійким настільки, щоб у випадках відсутності інтернету не втрачати дані, а у випадку відсутності енергопостачання мати змогу самостійно розпочати роботу;

– додаток для МК мусить мати легко розширювану архітектуру, яка б дозволяла без складностей додавати нові датчики чи керуємі елементи;

– веб-сайт мусить мати інтуїтивно зрозумілий інтерфейс і надавати персоналу доступ до потрібної інформації без попереднього навчання.

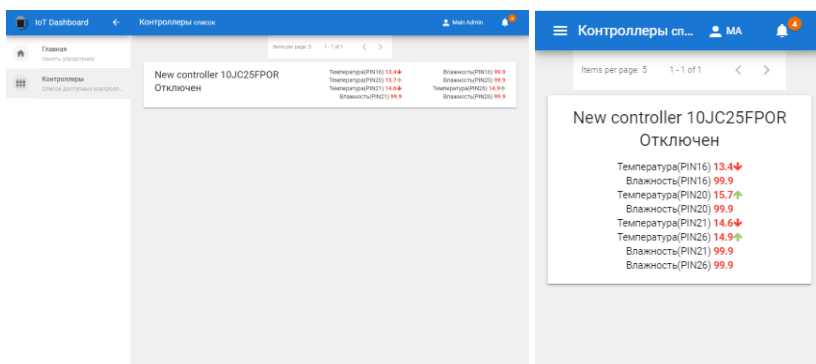


Рисунок 1 – Демонстрація інтерфейсу списку МК

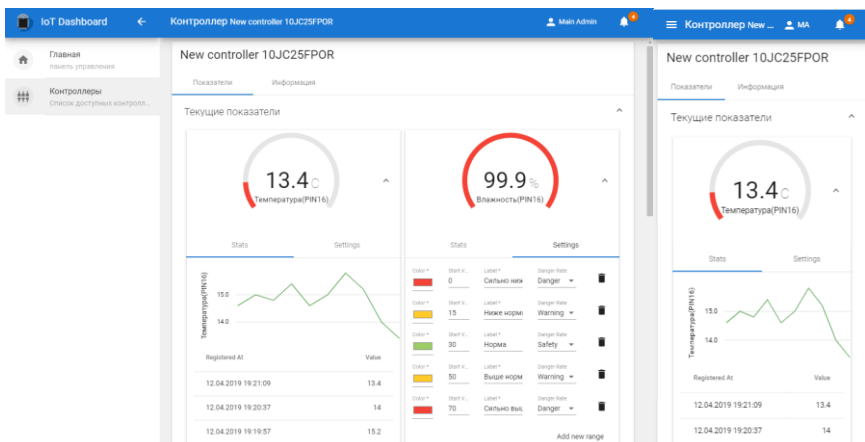


Рисунок 2 – Демонстрація інтерфейсу деталей МК

У різних частинах система використовує такі технології як WebApi2 (C#, сервер), UWP (C#, МК), Prism (C#, МК), SignalR (C#/JS, сервер/МК/клієнт), Entity Framework Core (C#, сервер/МК), Angular (JS, клієнт).

Прийняті протягом всього життєвого циклу рішення дозволили управляти ризиками проекту та досягти запланованого результату в короткі строки з обмеженими ресурсами, а отримана у результаті система виявилася придатною на практиці за результатами тестування.

### **СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

1. Kranz, M. Building the Internet of Things: Implement New Business Models, Disrupt Competitors, Transform Your Industry [Текст] / M. Kranz. – Wiley, 2016. – 272 p.