Обробка інформації в складних організаційних системах

УДК 621.382 DOI: 10.30748/soi.2019.156.03

В.В. Голян, Д.І. Самойленко

Харківський національний університет радіоелектроніки, Харків

СТВОРЕННЯ АРХІТЕКТУРИ ПРОГРАМНОЇ СИСТЕМИ МОНІТОРИНГУ ІНФОРМАЦІЇ ПРО СТАН ЗДОРОВ'Я ЛЮДИНИ

У статті обґрунтована актуальність дослідження програмних систем моніторингу інформації про стан здоров'я людини, з допомогою розумних пристроїв IoT. Проведен аналіз існуючих пристроїв, які дозволяють стежити за станом здоров'я людини, програмних продуктів, які допомагають обробляти інформацію, зчитану з розумних пристроїв. А також спроектована архітектура програмної системи і схеми бази даних. У роботі були розглянуті сучасні технології для створення веб-сервісів, нову, набираючу оберти, концепцію інтернет речей, яка дозволяє робити моніторинг практично будь-якого об'єкта, відстежувати і керувати ним, а також включати інформацію про ці об'єкти в мережу.

Ключові слова: *програмна система, здоров'я, інтернет речей, архітектура систем, база даних.*

Вступ

Постановка проблеми. Тема вивчення здоров'я людини була і буде цікава протягом усього існування людства. Люди намагаються стежити за своїм здоров'ям кожен день. Хтось контролює кількість калорій, що поїдаються за день, хтось робить зарядку, хтось просто не має шкідливих звичок. Також останнім часом з'явилося багато різних "розумних помічників", які допомагають стежити за здоров'ям. Наприклад, велику популярність зараз набирають фітнес-трекери. Фітнес-трекери порівняно недавно прийшли на ринок гаджетів, але вже отримали свою частку популярності. Фітнес-трекер (fitness tracker) або "розумний браслет" – пристрій, що зчитує біологічні показники свого користувача. Фітнес-трекер являє собою, як правило, браслет або затиск з датчиком, який якраз і зчитує різні параметри свого власника. На даний момент існують системи, які дозволяють постійно відстежувати такі параметри стану людини як: пульс, температура, кількість споживаних калорій, кількість вдихів, видихів і т.д. Потрібно так само відзначити, що більшість трекерів не володіє програмним забезпеченням для аналізу даних, а може тільки їх фіксувати.

Ще одним корисним пристроєм може бути прилад, який вимірює кількість вітамінів та мінералів в організмі. Найвідомішим прикладом є Vitastiq. Вітаміномометр зовні схожий на олівець. Він підключається до смартфону через роз'єм для навушників. На смартфоні потрібно запустити спеціальний додаток. Далі користувачеві потрібно прикладати олівець Vitastiq до різних частин тіла людини. Результат перевірки будуть відображатися на екрані смартфона.

Можна зробити висновок, що вже на даний момент існує багато приладів, які допомагають вимірювати різні показники стану здоров'я людини, але ринок програмного забезпечення для подібних пристроїв поки малий. .

Аналіз останніх досліджень і публікацій. У зв'язку з описаною проблемою, необхідно зробити аналіз існуючих систем програмного забезпечення для моніторингу інформації про стан здоров'я людини. Наприклад: існує два мільтиплатформених додатки (платформи Android та iOS) для дослідження за станом свого здоров'я Moves. Інтерфейс програми недостатньо легкий. Перевагою Moves є наявність додатку для декількох платформ, але це рішення не націлене на дослідження здоров'я іншої людини, в ньому немає системи повідомлень, а також у додатку Moves відсутня функція зчитування артеріального тиску, що є досить важливою функцією. Також, один з найпопулярніших веб-сервісів – Strava, надає можливість здійснювати повноцінне керування своїми тренуваннями, отримувати статистику та ділитися своїми досягненнями з друзями. Додаток надає багато можливостей для соціальної активності користувача: він може публікувати фотографії з тренувань у свій профіль Instagram, ділитися своїми досягненнями в Facebook або Twitter, мотивувати друзів, залишаючи коментарі до опублікованих тренувань. Але він більше направлений на заняття спортом, ніж на дослідження стану здоров'я

людини. В ньому немає функціоналу, який допоможе при поганих показниках стану здоров'я, також він не має можливості створювати графіки показників стану здоров'я.

Метою статті є дослідження та актуалізація рішень за моніторингом та аналізом стану здоров'я людини, а також проектування архітектури програмної системи.

Виклад основного матеріалу

Щоб виявити проблеми в даній предметній галузі, треба розглянути звичайну людину, яка хоче наглядати за здоров'ям свого родича, наприклад дитини. Вже існує багато пристроїв, які ви можете дати йому в школу, а після того, як він повернеться, подивитися, як він себе почував на протязі всього дня. В залежності від результатів користувач розумних пристроїв може вчасно відвести свою дитину до лікаря або визначити, який комплекс вітамінів слід прийняти. Але що робити, якщо ви хочете переглянути стан здоров'я на відстані? Адже нині існуючі системи можуть обробляти і відправляти результати на відстані всього декількох метрів.

На допомогу приходять сучасні технології. Для того, щоб зробити рішення актуальним протягом багатьох років, було вирішено використовувати концепцію Internet of Things. Концепція IoT дозволяє робити моніторинг практично будь-якого об'єкта, відстежувати і керувати ним, а також включати інформацію про ці об'єкти в загальний "цифровий всесвіт".

Інтернет речей – це концепція фізичних об'єктів (речей), об'єднаних в обчислювальну мережу (в загальному випадку мається на увазі об'єднання через інтернет). Основна ідея полягає в тому, що речі в цій мережі будуть "розумні" і матимуть можливість взаємодії між собою і зовнішнім світом.

Сама концепція IoT була сформована в 1999 році як результат розвитку засобів радіочастотної ідентифікації (RFID) (для взаємодії пристроїв між собою і оточенням) [1–3]. Однак, протягом першого десятка років інтернет речей залишався долею в значній мірі ентузіастів і вузьких фахівців. Значним поштовхом для розвитку інтернету речей послужив розвиток технологій, а також інтерес до даної тематики з боку більшості світових концернів. Що по суті є взаємодоповнюючими і взаємостимулюючими факторами.

На даний момент практично всі великі компанії зі світовим ім'ям усвідомили, що дана концепція буде затребувана в найближчі кілька років. З цієї причини в їх складі з'явилися робочі групи, які займаються даною проблематикою.

На найвищому рівні абстракції можна розділити інтернет речей на 2 складові:

– програмну;

– апаратну.

У свою чергу програмна складова ділиться, як мінімум, на наступні частини:

– ПЗ для звичайного виконавчого пристрою ("речі");

– ПЗ для пристроїв каналів зв'язку, які забезпечують комунікацію "речей" між собою;

– ПЗ, яке забезпечує безпосередньо корисне навантаження (функціональність), тому що сама по собі можливість спілкуватися пристроїв між собою не несе користі для людини.

Програмна система має складатися з чотирьох частин: сервер, веб-клієнт, мобільний додаток, IoT. Сервер має надавати публічне API для обробки запитів з веб-клієнту та мобільного додатку. Це API має мати методи для реєстрації, авторизації, пошуку та підписки на інших користувачів. Також серверна частина повинна тісно взаємодіяти з базою даних: зберігати, видаляти та змінювати дані. На сервері баз даних зберігаються дані користувачів у базі даних PostgreSQL.

Дані, оброблені на серверах, передаються власне клієнтам. Емулятор IoT має бути розроблений таким чином, щоб кожний період часу дані про стан здоров'я людини оновлювались і користувач бачив лише найсвіжіші результати вимірів.

Для проектування програмної системи була обрана трирівнева архітектура (рис. 1). Одним з важливих чинників вибору даної архітектури є можливість роботи та зміни рівнів незалежно одне від одного [4].

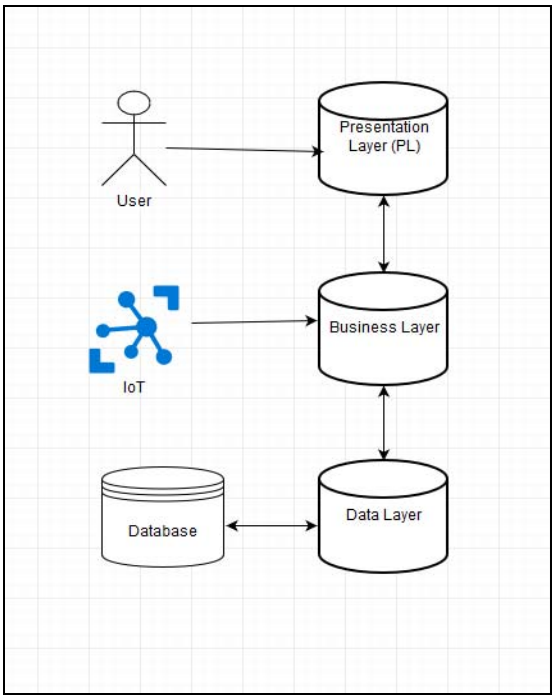

Рис. 1. Архітектура програмної системи

Під час планування та моделювання програмного продукту було визначено, що проект має складатися з наступних частин:

а) інтерфейс користувача забезпечує шар представлення (Presentation Layer, PL). Завдяки використанню графічного інтерфейсу, користувач безпосередньо взаємодіє зі всією системою;

б) шар бізнес-логіки (Business Layer, BL) є посередником між графічним інтерфейсом та даними, які зберігаються на сервері. Він містить всі необхідні функції, які необхідні системі для функціонування. Шар бізнес-логіки має відповідати патернам та принципам проектування програмного забезпечення. Оскільки шари є незалежними один від одного, шар бізнес-логіки може бути використаним як мобільним, так і веб додатком, завдяки публічному API;

в) шар доступу до даних (Data Access Layer, DAL) відповідає за зберігання та взаємодію даних з СУБД. Він відправляє запити на додавання, зміну та видалення даних, після чого трансформує отриманні результати у Java об'єкти.

Взаємодія користувача з програмною системою йде наступним чином: користувач, завдяки клієнту, працює с шаром представлення, який відображає елементи графічного інтерфейсу за допомогою JavaScript та HTML. Шар представлення виконує запити до шару бізнес-логіки, завдяки публічному API, який є на сервері. Після необхідних операцій шар доступу до даних працює з СУБД та відсилає відповідь на back-end. Після чого відповідь обробляється бізнес-логікою, та результат відсилається клієнту. IoT працює тільки з шаром бізнес-логіки.

При проектуванні програмної системи було прийняте рішення використовувати PostgreSQL в якості СКБД. PostgreSQL ґрунтується на використанні індексів, інтелектуальному планувальнику запитів, тонкій системі блокувань, системі управління буферами пам'яті і кешування, чудовою масштабованості при конкурентній роботі, що забезпечує високу продуктивність [5–8]. У PostgreSQL є вбудовані засоби із забезпечення шардінга (розподіл набору даних по серверах на основі певного ключа), комбінуючи який реплікацією даних можна побудувати горизонтально масштабований кластер зберігання, в якому відсутня єдина точка відмови (збій будь-якого вузла не позначається на роботі БД), підтримується автоматичне відновлення після збою і перенесення навантаження з вузла, який вийшов з ладу.

Основними таблицями у базі даних є: "User", "HealthStatus", "Subscription". Також є асоціативна сутність "User_Subscription", яка зв'язує користувача з його підписками. Сутності "UserRole" та "Status" є перераховуємого типу. Проаналізувавши вимоги до програмної системи, була спроектована схема бази даних (рис. 2).

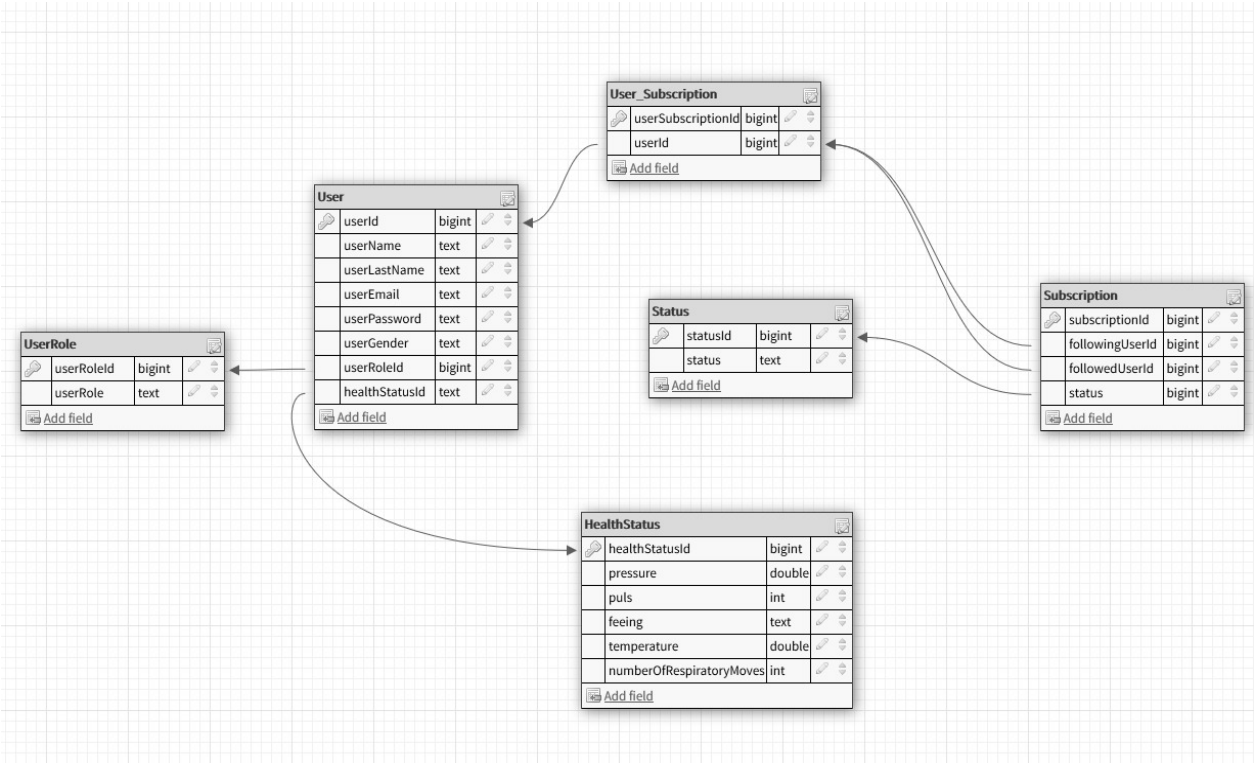

Рис. 2. Схема бази даних

Для розробки серверу рекомендовано обрати технологію створення Web-застосувань – Java з використанням SpringMVC фреймворку. Java є платформо-незалежною мовою, тому вона є досить поширеною та популярною мовою програмування. Також для розробки обрано такі основні технології як Spring Tool Suite та платформа Java. Для виклику методів сервера використовуються REST контроле-

ри з фреймворку Spring, що дозволяє отримувати та відправляти різноформатні дані по протоколу HTTP.

На сьогоднішній день прийнято використовувати REST (Representational State Transfer) метод взаємодії компонентів розподіленого додатка в мережі Інтернет, при якому виклик віддаленої процедури являє собою звичайний HTTP-запит, а необхідні дані передаються як параметри запиту [9–10]. Для ідентифікації ресурсів HTTP використовує глобальні URI. HTTP – протокол прикладного рівня, схожими на нього є FTP і SMTP.

Цей протокол використовується в комп'ютерних мережах. Назва скорочена від Hyper Text Transfer Protocol, протокол передачі гіпер-текстових документів. Обмін повідомленнями йде за звичайною схемою "запит-відповідь".

На відміну від багатьох інших протоколів, HTTP не зберігає свого стану. Це означає відсутність збереження проміжного стану між парами "запит-відповідь" [11–12].

Діаграма класів (рис. 3) включає в себе серверні класи User, як модель користувач; HealthStatus, як модель стану здоров'я людини; Subscription, як модель підписок користувачів. UserDAO, HealthStatusDAO, SubscriptionDAO – інтерфейси, що визначають всі операції, що повинні бути реалізовані для роботи з базою даних, за допомогою цих інтерфейсів забезпечується розширяємість системи, тому що є можливість використовувати іншу базу даних. Набір сервісів UserService, SubscriptionService, HealthStatusService, які оперують з даними та виконують усю логіку програмної системи. Доступ з графічного інтерфейсу до серверної частини системи здійснюється завдяки контролерам UserController, SubscriptionController, HealthStatusController.

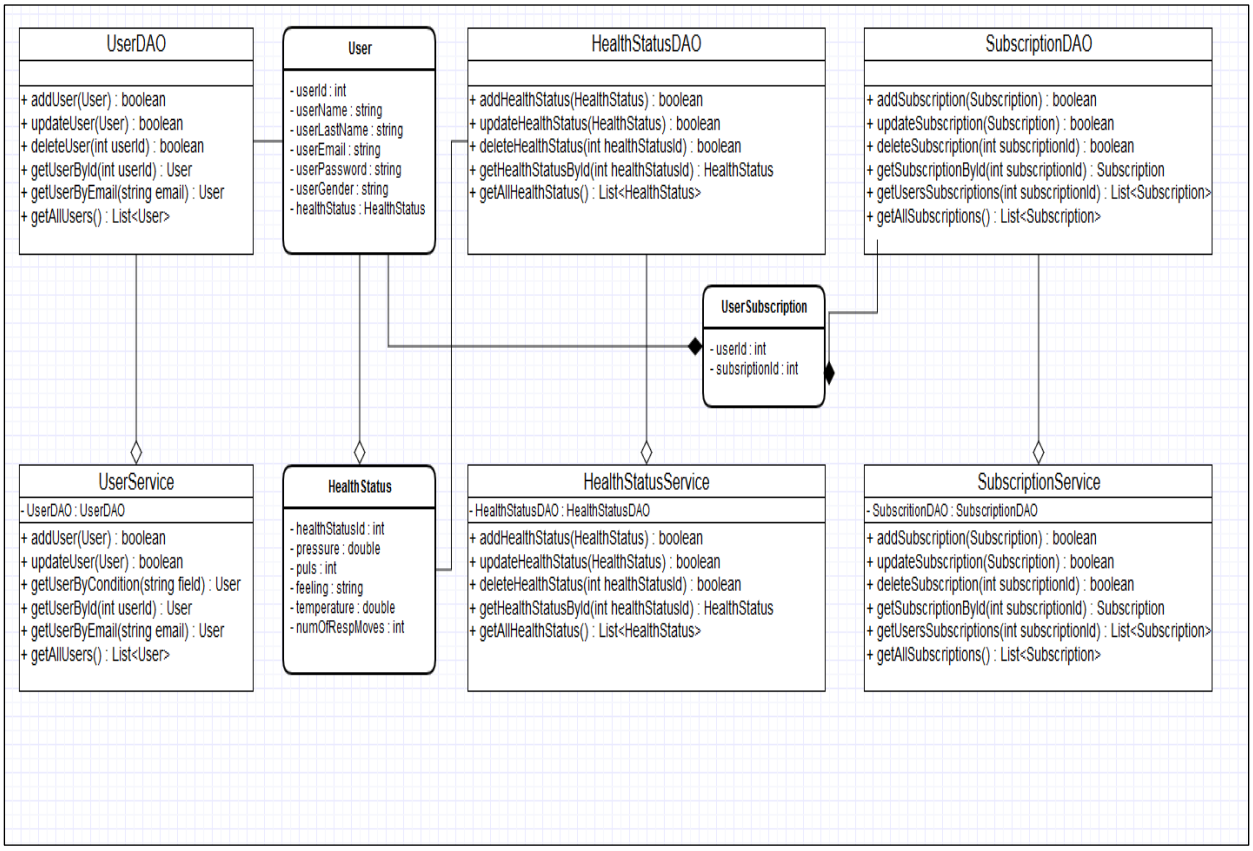

Рис. 3. Діаграма класів

Висновки

Проведені дослідження показали, що ринок розумних пристроїв зростає, а отже, і ринок програмного забезпечення для цих пристроїв не стоїть на місці. У результаті виконання роботи був проведений аналіз предметної області, проведена постановка задачі та спроектована архітектура програмної системи, що дозволяє доглядати за станом здоров'я людини. Програмна система базується на сервісно-орієнтованій архітектурі, тому складається з окремих частин: сервера, веб-клієнта та мобільного додатку. Також, в ході виконання роботи, вдалося актуалізувати рішення у сфері моніторингу та аналізу стану здоров'я людини, завдяки використанню нової технології інтернету речей та вибору сучасних технологій.

Список літератури

1. Chen M. RFID Technologies for Internet of Things / M. Chen, S. Chen. – Cham: Springer, 2016. – 95 p.

2. Rolf H.W. Internet of Things: Legal Perspectives / H.W. Rolf, R. Weber. – Zurich: Springer, 2010. – 129 p.

3. Uckelmann D. Architecting the internet of things / D. Uckelmann, M. Harrison, F. Michahelles. – New York: Springer, 2011. – 351 p.

4. Назаров С.В. Архитектура и проектирование программных систем: монография / С.В. Назаров. – 2-е изд., перераб. и доп. – Москва: ИНФРА-М, 2016. – 374 с.

5. PostgreSQL docs [Електронний ресурс] / PostgreSQL. – Режим доступу: www/URL: http://docs.postgresql.org/manual/ 08.05.2017. – Загол. з екрану.

6. PostgreSQL server programming : extend PostgreSQL using PostgreSQL server programming to create, test, debug, and optimize a range of user-defined functions in your favorite programming language / U. Dar, H. Krosing, J. Mlodgenski, K. Roybal. – Birmingham: Packt Publishing, 2015. – 312 p.

7. Schonig H-J. Mastering PostgreSQL 9.6: A comprehensive guide for PostgreSQL 9.6 developers and administrators / H-J . Schonig. – Birmingham: Packt Publishing, 2017. – 416 p.

8. Juba S. Learning PostgreSQL / S. Juba, A. Vannahme, A. Volkov. – Birmingham: Packt Publishing, 2015. – 464 p.

9. Varanasi B. Spring REST / B. Varanasi, S. Belida. – New York: Apress, 2015. – 208 p.

10. Allamaraju S. RESTful Web Services Cookbook: Solutions for Improving Scalability and Simplicity / S. Allamaraju. – New York: O' Reilly, 2010. – 448 p.

11. Олифер В. Компьютерные сети. Принципы, технологии, протоколы: пер. с англ. / В. Олифер, Н. Олифер. – СПб.: Питер, 2016. – 992 с.

12. Stallings W. Data and Computer Communications / W. Stallings. – New Jersey: Pearson, 2010. – 912 p.

References

1. Chen, M. and Chen, S. (2016), *RFID Technologies for Internet of Things*, Springer, Cham, 95 p.

2. Rolf, H.W. and Weber, R. (2010), *Internet of Things: Legal Perspectives*, Springer, Zurich, 129 p.

3. Uckelmann, D. and Harrison, M. (2011), *Architecting the internet of things*, Springer, New York, 351 p.

4. Nazarov, S.V. (2016), "*Arhitektura i proektirovanie programnih sistem: monografiya*" [*Architecture and design of software systems: monograph*], INFRA-M, Moscow, 374 p.

5. *PostgreSQL docs*, available at www.docs.postgresql.org/manual/ (accessed 8 May 2017).

6. Dar, U., Krosing, H., Mlodgenski, K. and Roybal, K. (2015), *PostgreSQL server programming : extend PostgreSQL using PostgreSQL server programming to create, test, debug, and optimize a range of user-defined functions in your favorite programming language*, Packt Publishing, Birmingham, 312 p.

7. Schonig, H-J. (2017), *Mastering PostgreSQL 9.6: A comprehensive guide for PostgreSQL 9.6 developers and administrators*, Packt Publishing, Birmingham, 416 p.

8. Juba, S., Vannahme, A. and Volkov, A. (2015), *Learning PostgreSQL*, Packt Publishing, Birmingham, 464 p.

9. Varanasi, B. and Belida, S. (2015), *Spring REST*, Apress, New York, 208 p.

10. Allamaraju, S. (2010), *RESTful Web Services Cookbook: Solutions for Improving Scalability and Simplicity*, O' Reilly, New York, 448 p.

11. Olipher, V. and Olipher, N. (2016), "*Kompiuternie seti. Principy, tehnologii, protokoli*" [*Computer network. Principles, technologies, protocols*], Publishing House Piter, Sankt-Petersburg, 992 p.

12. Stallings, W. (2010), *Data and Computer Communications*, Pearson, New Jersey, 912 p.

Надійшла до редколегії 4.01.2019
Схвалена до друку 22.01.2019

Відомості про авторів:

Голян Віра Володимирівна
кандидат технічних наук доцент
доцент кафедри Харківського національного
університету радіоелектроніки,
Харків, Україна
https://orcid.org/0000-0001-5981-4760

Самойленко Денис Ігорович
бакалавр Харківського національного
університету радіоелектроніки,
Харків, Україна
https://orcid.org/0000-0001-9077-7910

Information about the authors:

Vira Golian
Candidate of Technical Sciences Associate Professor
Senior Lecturer
of Kharkiv National University of Radio Electronics,
Kharkiv, Ukraine
https://orcid.org/0000-0001-5981-4760

Denys Samoilenko
Bachelor
of Kharkiv National University of Radio Electronics,
Kharkiv, Ukraine
https://orcid.org/0000-0001-9077-7910

СОЗДАНИЕ АРХИТЕКТУРЫ ПРОГРАММНОЙ СИСТЕМЫ МОНИТОРИНГА ИНФОРМАЦИИ О СОСТОЯНИИ ЗДОРОВЬЯ ЧЕЛОВЕКА

В.В. Голян, Д.И. Самойленко

В статье обоснована актуальность исследования программных систем мониторинга информации о состоянии здоровья человека с помощью умных устройств IoT. Был проведен анализ существующих устройств, которые позволяют следить за состоянием здоровья человека, а также программных продуктов, которые помогают обрабатывать полученную с устройств информацию. Целью работы является проектирование архитектуры приложения, а также схемы базы данных, так, чтобы приложение могло удовлетворять всем требованиям пользователей, желающих использовать систему мониторинга информации о здоровье человека. Были рассмотрены методы создания инструментария, основанные на разработке веб-приложений на платформе Java и протоколе передачи данных HTTP. PostgreSQL была выбрана в качестве СУБД для системы. Преимущества данной СУБД описаны в статье. Помимо этого, для работы с базой данных были разработаны таблицы и связи между ними. Также была спроектирована архитектура программной системы. В работе были рассмотрены современные технологии для создания веб-сервисов, а также новая, набирающая популярность концепция интернета вещей, которая позволяет производить мониторинг практически любого объекта, следить и управлять им, а также включать информацию об этих объектах в сеть. В статье была описана схема взаимодействия всех компонентов системы. Также в публикации представлена диаграмма классов для приложения для мониторинга состояния здоровья человека. В ней представлены основные сущности и связи между ними, которые потребуются для реализации приложения. Как результат работы, была разработана архитектура и схема базы данных для программной системы мониторинга состояния здоровья человека.

Ключевые слова: *программная система, здоровье, интернет вещей, архитектура систем, база данных.*

CREATION OF ARCHITECTURE OF SOFTWARE MONITORING SYSTEM INFORMATION ON THE CONDITION OF HUMAN HEALTH

V. Golian, D. Samoilenko

The article substantiates the relevance of the study of software systems for monitoring information about the state of human health using smart devices IoT. The analysis of existing devices that allow to monitor the state of human health, software products that help to process the information received from the devices was carried out. The aim is to design and development of intelligent software system for monitoring on human health. Functional systems must meet all the requirements of users who want to use the system for monitoring information on human health. The methods of making instruments based on the development of web applications on the platform Java, data transfer protocol HTTP. Also next technologies were analysed: Apache Maven, Apache Tomcat servlet container and Spring framework for creating web applications on the platform Java EE. PostgreSQL was selected as DBMS. The advantages if this DBMS are described in the article. Tables and relationships in the database were designed. And also considered a new, gaining popularity concept of the Internet of things, which allows you to monitor almost any object, monitor and manage it, as well as include information about these objects in the network. The scheme of interaction of all system components was also described. Also, the publication presents a class diagram for the application for monitoring the state of human health. It presents the basic entities and relationships between them that will be required to implement the application. As a result of this work, the architecture and database scheme of the human health information monitoring system was designed, which is a web application based on the SpringMVC framework, so the aim of the work was done.

Keywords: *software system, health, internet of things, software architecture, database.*