УДК 004.75:37.018.43

ВИКОРИСТАННЯ DOCKER-КОНТЕЙНЕРИЗАЦІЇ ПІД ЧАС РОЗРОБКИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ СТУДЕНТСЬКОГО МІСТЕЧКА

І. М. Терновий, О. Г. Хамула

Національний університет «Львівська політехніка»,
вул. С. Бандери, 12, Львів, 79013, Україна

Стаття присвячена дослідженню використання технології Docker-контейнеризації для розробки, розгортання та управління інформаційною системою студентського містечка. У сучасних освітніх закладах інформаційні системи відіграють ключову роль у забезпеченні ефективного управління розкладами, ресурсами, комунікаціями та інноваційними технологіями, такими як доповнена реальність (AR). Однак традиційні методи розробки таких систем стикаються з викликами, зокрема складністю розгортання, низькою портативністю, високими вимогами до ресурсів і труднощами інтеграції з гетерогенними середовищами. У цьому контексті Docker-контейнеризація пропонує перспективне рішення, забезпечуючи модульність, масштабованість і спрощення процесів розробки та адміністрування.

В статті аналізується ефективності застосування Docker для створення інформаційної системи студентського містечка, з акцентом на оптимізацію процесів розробки, зниження витрат і забезпечення сумісності з освітніми задачами. Розглядаються ключові етапи управління проєктом: планування, координація роботи міждисциплінарної команди (програмістів, адміністраторів, дизайнерів), вибір технологічного стека (наприклад, Docker, Kubernetes, Unity для AR-компонентів) і контроль якості.

Дослідження демонструє, що Docker дозволяє створювати модульні системи на основі мікросервісної архітектури, що спрощує інтеграцію різноманітних компонентів, таких як управління розкладами, бібліотечними ресурсами чи AR-додатками для віртуальних турів кампусом. Стаття також аналізує виклики, пов'язані з безпекою контейнерів, необхідністю підготовки IT-фахівців і інтеграцією з legacy-системами. Пропонуються практичні рекомендації щодо використання інструментів моніторингу (Prometheus) і сканування вразливостей (Trivy) для забезпечення надійності системи. Результати дослідження підкреслюють, що Docker-контейнеризація сприяє створенню гнучких, портативних і економічно вигідних інформаційних систем для студентських містечок. Стаття буде корисною для розробників, адміністраторів і дослідників, які працюють над впровадженням сучасних технологій в освітнє середовище, а також для менеджерів проєктів, які прагнуть оптимізувати витрати та підвищити ефективність управління.

Ключові слова: *Docker-контейнеризація, інформаційна система, студентське містечко, розробка програмного забезпечення, масштабованість, управління проєктами, безпека контейнерів.*

У сучасному світі інформаційні технології відіграють ключову роль у забезпеченні ефективного функціонування освітніх закладів, зокрема студентських містечок, де складні інформаційні системи підтримують управління розкладами, ресурсами, комунікацією та іншими процесами. Розробка таких систем вимагає гнучких, масштабованих і надійних рішень, які дозволяють швидко адаптуватися до мінливих потреб користувачів. У цьому контексті технологія Docker-контейнеризації набуває особливого значення, пропонуючи інструменти для ізоляції, спрощення розгортання та управління програмними компонентами. Ця стаття присвячена дослідженню використання Docker-контейнеризації для створення інформаційної системи студентського містечка, з акцентом на її здатність забезпечувати модульність, портативність і оптимізацію процесів розробки та адміністрування.

Постановка проблеми. Сучасні інформаційні системи для студентських містечок стикаються з низкою викликів, пов'язаних із їхньою розробкою, розгортанням і підтримкою. Традиційні підходи до створення таких систем часто є громіздкими, потребують значних ресурсів для налаштування серверного середовища та забезпечення сумісності між різними компонентами програмного забезпечення. Студентські містечка, як комплексні освітні екосистеми, потребують інформаційних систем, які здатні інтегрувати різноманітні функції — від управління розкладами та ресурсами до підтримки комунікаційних платформ і, потенційно, інноваційних технологій, таких як доповнена реальність (AR). Однак ці системи мають бути гнучкими, масштабованими та легкими в адмініструванні, щоб відповідати динамічним потребам студентів і адміністрації.

Одним із ключових викликів є складність розгортання та управління програмними компонентами в умовах гетерогенних апаратних і програмних середовищ. Традиційні методи віртуалізації, такі як використання віртуальних машин, часто є ресурсоємними та ускладнюють швидке масштабування. Крім того, розробка інформаційних систем для студентських містечок потребує забезпечення високого рівня портативності, щоб система могла працювати на різних платформах без значних модифікацій. До цього додаються проблеми, пов'язані з витратами часу та ресурсів на тестування, оновлення та підтримку системи, а також необхідність інтеграції сучасних технологій, таких як мікросервісна архітектура чи AR-функціонал, для створення інтерактивного та інноваційного освітнього середовища.

Технологія Docker-контейнеризації пропонує потенційне рішення цих проблем завдяки легкій віртуалізації, яка забезпечує ізоляцію додатків, спрощення розгортання та підвищення портативності. Однак використання Docker у контексті інформаційних систем для студентських містечок залишається недостатньо дослідженим, особливо щодо оптимізації процесів розробки, зниження витрат і забезпечення сумісності з освітніми задачами. Таким чином, виникає потреба в дослідженні ефективності Docker-контейнеризації для створення модульних, масштабованих і економічно вигідних інформаційних систем, які відповідають вимогам студентських містечок, а також у визначенні оптимальних підходів до управління такими проектами та розрахунку їхньої вартості.

Аналіз останніх досліджень та публікацій. Docker-контейнеризація стала революційною технологією в розробці програмного забезпечення, зокрема для інформаційних систем, завдяки своїй легкості, портативності та ефективності порівняно з традиційною віртуалізацією. У контексті студентських містечок інформаційні системи мають вирішувати задачі управління розкладами, ресурсами, комунікаціями та, можливо, інтеграції інноваційних технологій, таких як доповнена реальність (AR). Останні дослідження підкреслюють, що Docker забезпечує: портативність (системи можна розгортати в різних середовищах (локальних, хмарних) без значних модифікацій) [1]; масштабованість (контейнери дозволяють швидко масштабувати окремі компоненти системи, що важливо для освітніх платформ із мінливим навантаженням); економію ресурсів (на відміну від віртуальних машин, Docker використовує менше ресурсів, що робить його привабливим для освітніх закладів із обмеженим бюджетом) [2].

Стаття [2] підкреслює, що контейнеризація трансформує освітні платформи, забезпечуючи ізоляцію, портативність і ефективність. Наприклад, системи управління навчанням (LMS), такі як Moodle чи Canvas, використовують Docker для забезпечення стабільної роботи та масштабування. У контексті студентського містечка це може стосуватися розгортання LMS, віртуальних лабораторій чи комунікаційних платформ. Ключові виклики включають обмежені бюджети, нестачу кваліфікованих кадрів і проблеми інтеграції з застарілими системами. Ця стаття деталізує, як Docker використовується для розгортання LMS, віртуальних класів (наприклад, Zoom, Microsoft Teams) і дослідницьких платформ. Вона пропонує практичні стратегії, такі як оцінка потреб, вибір інструментів (Docker, Kubernetes) і тестування контейнерів. Це прямо стосується інформаційної системи студентського містечка, де потрібна інтеграція різноманітних сервісів.

В наступній статті [3] проведено аналіз 57 досліджень, які показують, що Docker відіграє ключову роль у хмарних і розподілених системах. Вона підкреслює легкість Docker, що сприяє швидкому розгортанню мікросервісів, які можуть бути основою інформаційної системи студентського містечка. Наприклад, мікросервіси для управління розкладами, бібліотечними ресурсами чи AR-додатками можна ізолювати в окремих контейнерах, забезпечуючи модульність.

Автори у своєму дослідженні [4] описують веб-інтерфейс для розгортання Docker-контейнерів, що усуває потребу в локальній установці Docker. Це може бути корисним для студентських містечок, де користувачі (адміністратори чи студенти) можуть керувати системою без глибоких технічних знань, наприклад, через веб-портал для доступу до освітніх ресурсів.

Що стосується безпекової сторони та управління доступом, то в статті [5] наголошується на безпекові виклики Docker, такі як вразливості в образах контейнерів і потреба в моніторингу. Для студентського містечка це важливо, оскільки система може містити конфіденційні дані (наприклад, персональну інформацію студентів). Рекомендації включають використання сканерів вразливостей (Trivy) і налаштування доступу. А в дослідженні [6] описується система управління доступом на базі Docker, що може бути застосована до студентського містечка

для контролю ролей (студенти, викладачі, адміністратори). Це вирішує проблему складного розгортання та низької ефективності традиційних систем.

Питання, що стосуються розкриттю використання на практиці Docker та можливі виклики його використання описані в статтях [7, 8]. Так в статті [7] підкреслюється, що Docker забезпечує відтворюваність середовищ, що є критичним для дослідницьких платформ у студентських містечках. Наприклад, контейнери можуть використовуватися для розгортання однакових середовищ для лабораторних робіт чи AR-додатків. А в роботі [8] проведені дослідження показують що Docker на Linux має кращу продуктивність порівняно з Windows чи macOS через додатковий шар віртуалізації. Це важливо для вибору інфраструктури для інформаційної системи студентського містечка, особливо якщо бюджет обмежений.

Хоча контейнеризація, як бачимо з огляду літературних джерел, широко застосовується в хмарних і комерційних системах, її використання в освітніх інформаційних системах, зокрема для студентських містечок, залишається менш дослідженим. А вразливості в Docker-образах і потреба в постійному моніторингу ускладнюють впровадження в освітніх закладах, де бракує кваліфікованих спеціалістів. Багато кампусів використовують legacy-системи, які важко інтегрувати з контейнерами. Потрібні API або middleware для вирішення цих проблем. Впровадження Docker вимагає підготовки IT-команд, що може бути бар'єром для освітніх закладів.

Також, останні дослідження показують, що Docker-контейнеризація є потужним інструментом для розробки інформаційних систем, зокрема в освітньому контексті. Вона забезпечує модульність, портативність і економію ресурсів, що ідеально відповідає потребам студентських містечок. Однак успішне впровадження вимагає вирішення безпекових питань, інтеграції з наявними системами та навчання персоналу. Для студентського містечка Docker може спростити розгортання систем управління навчанням, комунікаційних платформ чи AR-додатків, але потребує ретельного планування та оцінки вартості, щоб забезпечити ефективність і безпеку.

Мета. Дослідити та продемонструвати ефективність використання технології Docker-контейнеризації для розробки, розгортання та управління інформаційною системою студентського містечка, забезпечуючи модульність, масштабованість і спрощення процесів інтеграції та тестування, а також оцінити її переваги для оптимізації роботи освітніх інформаційних систем.

Виклад основного матеріалу дослідження. Під час розробки та підтримки інформаційної системи для студентського містечка існує необхідність в стабільності, масштабованості та швидкому оновленні програмного забезпечення. Оскільки інформаційна система зазвичай складається з декількох взаємопов'язаних компонентів (веб-інтерфейс, API, база даних, тощо), то виникає потреба в уніфікованому підході, який би був гнучким у додаванні та налаштуванні нових компонент. Для того, щоб покращити ефективність локальної розробки та процесу розгортання, варто розглянути використання Docker-контейнеризації, яка дозволяє суттєво полегшити дані процеси, підвищити надійність системи, а також істотно знизити витрати часу та зусиль. Варто зазначити, що контейнеризація дозволяє створити середовище, яке буде ідентичним на різних комп'ютерах розробників, тестових серверах та

в інших середовищах, що суттєво зменшує кількість помилок, пов'язаних із відмінностями конфігурацій. Крім того, використання Docker дає можливість швидко масштабувати систему у разі збільшення кількості користувачів, що особливо актуально під час пікових навантажень, наприклад у період поселення студентів або початку навчального семестру.

Варто розглянути даний концепт більш технічно, тому Docker-контейнеризація - це інструмент для контейнеризації компонент програмного забезпечення, що дозволяє упаковувати певну компоненту разом з усіма залежностями у ізольоване середовище, яке називається контейнером. В свою чергу контейнер працює однаково на будь-якому сервері або комп'ютері, незалежно від операційної системи чи локальних налаштувань, що усуває проблему відмінності конфігураційних налаштувань поміж різними операційними системами та середовищами. За своєю суттю контейнери є легкими та швидкими у запуску, оскільки використовують ядро операційної системи спільно, на відміну від віртуальних машин, що потребують власної операційної системи. Для створення та налаштування контейнера використовується Dockerfile, де описуються залежності, середовище виконання та команди запуску. Docker також надає реєстр образів (Docker Hub), з якого можна завантажувати готові образи, наприклад для баз даних або веб-серверів. Загалом, контейнеризація з Docker спрощує процес налаштування середовища розробки та прискорює розгортання оновлень. Крім того, Docker легко інтегрується з системами автоматичного тестування та CI/CD, що дозволяє автоматизувати перевірку та розгортання змін у програмному забезпеченні. Завдяки ізоляції, контейнери захищають систему від впливу некоректної роботи окремих компонентів, підвищуючи стабільність. Варто розглянути приклад налаштування Dockerfile:

```
FROM ruby:3.3.7-alpine

ENV BUILD_PACKAGES="build-base" \
    DEV_PACKAGES="tzdata postgresql-dev g++ make nodejs yarn vim git gcompat imagemagick imagemagick-dev python3" \
    RUBY_PACKAGES="" \
    PAGER="more" \
    NODE_OPTIONS="--openssl-legacy-provider"

RUN apk update && \
    apk add --no-cache \
    $BUILD_PACKAGES \
    $DEV_PACKAGES \
    $RUBY_PACKAGES && \
    rm -rf /var/cache/apk/*

RUN gem update --system && gem install bundler -v '2.3.0'

RUN mkdir -p /app
WORKDIR /app

COPY Gemfile* ./
RUN bundle install

COPY . ./

RUN yarn install --check-files

RUN bundle exec rails assets:precompile

EXPOSE 80

CMD bundle exec rails db:migrate && bundle exec rails s -p 80 -b '0.0.0.0'
```

Можна побачити, що даний Dockerfile використовує ruby:3.3.7 на базі Alpine Linux дистрибутиву, згодом встановлюються необхідні робочі та системні залежності, визначається робоча директорія в межах Docker-образу і вже в робоче середовище копіюються файли програмного забезпечення і на їх основі запускається сервер на TCP порті номер 80. Отже, це свого роду шаблон, що містить програмний код та необхідне середовище для його подальшого запуску в контейнері, адже Dockerfile створює лише образ. Після того, як налаштувати Dockerfile, є можливість створити Docker-образ за допомогою команди: “docker build -t my_app .”. І вже для запуску програми на основі створеного образу, необхідно виконати команду: “docker run -p 3000:80 my_app”, що створить контейнер на базі згенерованого образу. Контейнер - це запущений екземпляр образу, що виконує програму у ізольованому середовищі, його можна зупиняти, запускати повторно або видаляти без впливу на початковий образ.

Додатково варто розглянути docker-compose, який використовується для одночасного запуску та управління кількома взаємопов’язаними контейнерами як єдиною системою. Це спрощує налаштування, запуск і масштабування складних проєктів (наприклад, програмний код + база даних) однією командою. Прикладом може слугувати наступний docker-compose.yml файл:

```
version: '3.7'

x-app: &app
  build: .
  tmpfs:
  - /tmp
  stdin_open: true
  tty: true
  volumes:
    - .:/app:cached
    - rails_cache:/app/tmp/cache
    - bundle:/usr/local/bundle
  environment:
    - NODE_ENV=development
    - POSTGRES_HOST=db
    - POSTGRES_USER=postgres
    - POSTGRES_PASSWORD=password
    - REDIS_URL=redis://redis:6379/0
    - LAUNCHY_DRY_RUN=true
    - BROWSER=/dev/null
```

```yaml
▷Run All Services
services:
  ▷Run Service
  app:
    <<: *app
    depends_on:
      - db
    ports:
      - "3000:80"

  ▷Run Service
  db:
    image: postgres:17-alpine
    environment:
      - POSTGRES_HOST_AUTH_METHOD=trust
    ports:
      - "5432:5432"
    volumes:
      - dorm-db-data:/var/lib/postgresql/data

  ▷Run Service
  sidekiq:
    <<: *app
    command: bundle exec sidekiq
    depends_on:
      - db

  ▷Run Service
  redis:
    image: redis:alpine
    ports:
      - "6379:6379"

volumes:
  rails_cache:
  dorm-db-data:
  bundle:
```

В даному docker-compose.yml файлі описано конфігурацію декількох окремих компонент, а саме: app, db, sidekiq та redis, які будуть запускатись одночасно з коректними взаємозалежностями. Для того, щоб запустити дані сервіси, необхідно спочатку згенерувати або перегенерувати Docker-образи, на які посилаються компоненти: “docker-compose build”. Після того можна запускати компоненти окремо за допомогою команди “docker-compose up ‘назва компоненти’” або єдиною командою запустити всі сервіси одночасно: “docker-compose up”. Таким чином, кожна із даних компонент буде працювати та необхідні взаємозв’язки будуть налаштовані між собою, що в свою чергу істотно полегшує процес розробки програмного забезпечення на різних середовищах.

Висновок. Використання Docker у розробці інформаційної системи для студентського містечка є сучасним і ефективним підходом, що відповідає вимогам стабільності, гнучкості та швидкого впровадження змін. Контейнеризація дозволяє уникнути проблем, пов’язаних з різними середовищами розробки та значно

полегшує налаштування і запуск програмного забезпечення. Додатково, усі компоненти інформаційної системи можна ізолювати і керувати ними окремо, що підвищує надійність і загальну безпеку. Таким чином, впровадження контейнеризації сприяє підвищенню якості обслуговування студентів та оптимізує витрати на підтримку. Тому можна констатувати, що Docker є доцільним і корисним рішенням для розробки сучасних цифрових рішень у студентському середовищі зокрема.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. What is Docker? URL: https://docs.docker.com/get-started/docker-overview/.
2. Containerization For Educational Platforms. URL: https://www.meegle.com/en_us/topics/containerization/containerization-for-educational-platforms.
3. W.M.C.J.T. Kithulwatta, K.P.N Jayasena, Banage T. G. S. Kumara, R.M. Kapila Tharanga Rathnayaka. Integration With Docker Container Technologies for Distributed and Microservices Applications: A State-of-the-Art Review. *International Journal of Systems and Service-Oriented Engineering*. January 2022. https://DOI:10.4018/IJSSOE.297136. URL: https://www.researchgate.net/publication/362669052_Integration_With_Docker_Container_Technologies_for_Distributed_and_Microservices_Applications_A_State-of-the-Art_Review.
4. Abhishek M. Nair, Sivaiswarya CK., Sidharth S., Visakh KK., Jibin Joy. Dockerized Application with Web Interface. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*. Volume 10, Issue 2. March-April-2024. P. 412-419. https://doi.org/10.32628/CSEIT243646.
5. Gogulakrishnan Thiyagarajan, Prabhudarshi Nayak. Docker under Siege: Securing Containers in the Modern Era. *International Journal of Scientific Research in Computer Science Engineering and Information Technology*. Issue Volume 11, Issue 1, January-February-2025, P. 3674-3719. https://DOI:10.32628/CSEIT25112773.
6. Li You, Hui Sun. Research and Design of Docker Technology-Based Authority Management System. *Comput Intell Neurosci*. 2023 Jul 26; 2023: 9767210. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC9020899/.
7. Jürgen Cito, Vincenzo Ferme, Harald C. Gall. Using Docker Containers to Improve Reproducibility in Software and Web Engineering Research. *Lecture Notes in Computer Science*. International Conference on Web Engineering (ICWE) 2016, LNCS 9671, pp. 609–612, 2016. https://DOI: 10.1007/978-3-319-38791-8 58.
8. Maciej Sobieraj and Daniel Kotyński. Docker Performance Evaluation across Operating Systems. *Electrical, Electronics and Communications Engineering*. 2024, *14*(15), 6672; https://doi.org/10.3390/app14156672. URL: https://www.mdpi.com/2076-3417/14/15/6672.

REFERENCES

1. What is Docker? URL: https://docs.docker.com/get-started/docker-overview/.
2. Containerization For Educational Platforms. URL: https://www.meegle.com/en_us/topics/containerization/containerization-for-educational-platforms.
3. W.M.C.J.T. Kithulwatta, K.P.N Jayasena, Banage T. G. S. Kumara, R. M. Kapila Tharanga Rathnayaka. Integration With Docker Container Technologies for Distributed and

Microservices Applications: A State-of-the-Art Review. *International Journal of Systems and Service-Oriented Engineering*. January 2022. https://DOI:10.4018/IJSSOE.297136. URL: https://www.researchgate.net/publication/362669052_Integration_With_Docker_Container_Technologies_for_Distributed_and_Microservices_Applications_A_State-of-the-Art_Review.

4. Abhishek M. Nair, Sivaiswarya CK., Sidharth S., Visakh KK., Jibin Joy. Dockerized Application with Web Interface. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*. Volume 10, Issue 2. March-April-2024. P. 412-419. https://doi.org/10.32628/CSEIT243646.
5. Gogulakrishnan Thiyagarajan, Prabhudarshi Nayak. Docker under Siege: Securing Containers in the Modern Era. *International Journal of Scientific Research in Computer Science Engineering and Information Technology*. Issue Volume 11, Issue 1, January-February-2025, P. 3674-3719. https://DOI:10.32628/CSEIT25112773.
6. Li You, Hui Sun. Research and Design of Docker Technology-Based Authority Management System. *Comput Intell Neurosci*. 2023 Jul 26; 2023: 9767210. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC9020899/.
7. Jürgen Cito, Vincenzo Ferme, Harald C. Gall. Using Docker Containers to Improve Reproducibility in Software and Web Engineering Research. *Lecture Notes in Computer Science*. International Conference on Web Engineering (ICWE) 2016, LNCS 9671, pp. 609–612, 2016. https://DOI: 10.1007/978-3-319-38791-8 58.
8. Maciej Sobieraj and Daniel Kotyński. Docker Performance Evaluation across Operating Systems. *Electrical, Electronics and Communications Engineering*. 2024, *14*(15), 6672; https://doi.org/10.3390/app14156672. URL: https://www.mdpi.com/2076-3417/14/15/6672.

doi: 10.32403/1998-6912-2025-1-70-83-92

USING DOCKER CONTAINERIZATION DURING THE DEVELOPMENT OF AN INFORMATION SYSTEM FOR A STUDENT CAMPUS

I. M. Ternovyy, O. H. Khamula

Lviv Polytechnic National University,
12, S. Bandery St., Lviv, 79013, Ukraine
ivanternovyi@gmail.com, Orest.H.Khamula@lpnu.ua

The article is devoted to the study of the use of Docker containerization technology for the development, deployment and management of a campus information system. In modern educational institutions, information systems play a key role in ensuring the effective management of schedules, resources, communications and innovative technologies, such as augmented reality (AR). However, traditional methods of developing such systems face challenges, in particular, the complexity of deployment, low portability, high resource requirements and difficulties in integrating with heterogeneous environments. In this context, Docker containerization offers a promising solution,

providing modularity, scalability and simplification of development and administration processes.

The article analyzes the effectiveness of using Docker to create a campus information system, with an emphasis on optimizing development processes, reducing costs and ensuring compatibility with educational tasks. The key stages of project management are considered: planning, coordination of the work of an interdisciplinary team (programmers, administrators, designers), selection of a technology stack (e.g., Docker, Kubernetes, Unity for AR components), and quality control.

The study demonstrates that Docker enables the creation of modular systems based on a microservices architecture, which simplifies the integration of diverse components, such as scheduling management, library resources, or AR applications for virtual campus tours. The article also analyzes the challenges associated with container security, the need for IT training, and integration with legacy systems. Practical recommendations are offered for the use of monitoring tools (Prometheus) and vulnerability scanning (Trivy) to ensure system reliability. The results of the study emphasize that Docker containerization contributes to the creation of flexible, portable, and cost-effective information systems for student campuses. The article will be useful for developers, administrators, and researchers working on the implementation of modern technologies in the educational environment, as well as for project managers seeking to optimize costs and improve management efficiency.

Keywords: *Docker containerization, information system, campus, software development, scalability, project management, container security.*

Стаття надійшла до редакції 02.06.2025.
Received 02.06.2025.