DOI 10.34185/1562-9945-6-143-2022-10
УДК 004.9:303.732.4

М.В. Андрюхіна, Т.В. Селівьорстова

АРХІТЕКТУРНЕ РІШЕННЯ ДЛЯ ВЕБ-ДОДАТКУ DDP (DIPLOMA DEFENSE PROJECT) ДЛЯ ДОКУМЕНТУВАННЯ ПРОЦЕСУ ЕКЗАМЕНУВАННЯ

Анотація. Стаття присвячена розробці архітектури програмного забезпечення. Голов ним завданням запропонованої архітектури є цифровізація роботи членів екзаменаційної комісії, більш швидкої обробки документації в процесі захисту. Технічна можливість зме ншити навантаження на секретаря комісії існує вже сьогодні. В міру збільшення кількос ті студентів випускників, важливості вчасного документування та перевірки інформації актуальним постало питання формування відповідної документації в процесі захисту за допомогою певного програмного забезпечення.

Як свідчить практика розвинених країн, саме широке використання цифрових технологій (у сферах виробництва, фінансів, державного управління, соціального обслуговування тощо) забезпечує суттєве підвищення ефективності економічної діяльності та якості суспільного життя. Україна також стала на шлях цифровізації, про що свідчить Прийн яття у 2018 році Концепції розвитку цифрової економіки та суспільства України на 2018 2020 роки та затвердженому плані заходів щодо її реалізації.

Робота секретаря екзаменаційної комісії являє собою рутину з ведення документації: ве дення протоколів, формуванню звітів, підрахунок статистичних даних. Щоб спростити роботу членів комісії є можливість практичного застосування такої інновації як про єкт DDP (diploma defense project). Проєкт у статті запропоновано розробити за допомо гою фреймворка Ruby on Rails. Ruby входить до складу більшості дистрибутивів ОС Linux, поставляється з Mac OS X і доступна для користувачів інших операційних систем. Ста ном на серпень 2022 року Ruby входить до топ 20 найпопулярніших мов програмування за версією авторитетного спеціалізованого рейтингу Tiobe.

У статті також описаний процес створення архітектурного рішення для проєкту DDP, описані рекомендаційні технології для його створення, а також наведені UML діаграми, що більш детально описують архітектуру проєкту.

Ключові слова: Ruby on Rails, члени екзаменаційної комісії, UML діаграми, нефункціональні вимоги, MVC, веб сервіс.

Постановка проблеми. Цифровізація - насичення фізичного світу електронно-цифровими пристроями, засобами, системами та налагодження елект-

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

ронно-комунікаційного обміну між ними, що фактично уможливлює інтегральну взаємодію віртуального та фізичного, тобто створює кіберфізичний простір.

При системному державному підході цифрові технології будуть значно стимулювати розвиток відкритого інформаційного суспільства як одного з істотних факторів розвитку демократії в країні, підвищення продуктивності, економічного зростання, а також підвищення якості життя громадян України [1].

Поставлено завдання розробити програмну архітектуру для цифровізації діяльності членів екзаменаційної комісії та прискорення обробки документації під час захисту. Проблема полягає в надмірному навантаженні на секретаря екзаменаційної комісії з документуванні процесу захисту, формуванні звітів та ін.

Аналіз останніх досліджень і публікацій. На просторах інтернету запропоновано багато статей із розробки веб-сервісів за допомогою Ruby on Rails. Але найголовнішим джерелом інформації є сайт з офіційною документацією RoR.

У третьому виданні Software Architecture in Practice Лен Басс, Пол Клементс та ін. знайомлять нас із концепціями та найкращими практиками архітектури програмного забезпечення — як структурована система програмного забезпечення та як елементи цієї системи мають взаємодіяти. У 2021 році вийшло четверте видання Software Architecture in Practice, яке оновлене одинадцятьма новими розділами, де детально пояснює, що таке архітектура програмного забезпечення, чому вона важлива та як її проектувати, створювати, аналізувати, розвивати та керувати нею дисциплінованими та ефективними способами.

Цікавою є робота [2], де представлено інформаційну систему документаційного забезпечення роботи секретаря екзаменаційної комісії Нац. ун-ту “Львівська політехніка”, обґрунтовано необхідність та доведено ефективність такого впровадження.

У вищезгаданих наукових працях та підручниках з архітектури детально описані теоретичні аспекти, які потрібно враховувати у розробці веб-сервісів. Автори приводять свої доводи, щодо доцільності використання тих, чи інших технологій після визначення функціональних та нефункціональних вимог клієнта. Але досі немає єдиного програмного рішення, щодо автоматизації робочого місця секретаря екзаменаційної комісії та членів комісії в цілому.

Мета роботи - дослідження наявних архітектурних рішень для автоматизації роботи екзаменаційної комісії та розробка архітектурного рішення для подальшого створення програмного продукту на його основі для збільшення ефективності та покращення якості процесу захисту. Це важливий практичний зв'язок з потребами сучасної освітньої системи.

Викладення основного матеріалу дослідження. Для кожного архітектурного стилю можна знайти еталонну архітектуру, яка описує основні частини конструювання, призначення та залежності між ними. Було обрано REST. Архітектурний стиль REST дуже популярний під час побудови веб сервісів.

Важливим етапом є визначення Architecture Significant Requirements (ASRs), тобто обрання патернів для виконання вимог щодо якості, обмежень і деяких функціональних вимог [3].

Передусім треба визначити стейкхолдерів, тобто, дізнатись, хто зацікавлений у проєкті. Необхідно почути потреби всіх людей, які дотичні до системи, адже комусь будуть вкрай важливі квартальні звіти, а комусь - документація на систему тощо.

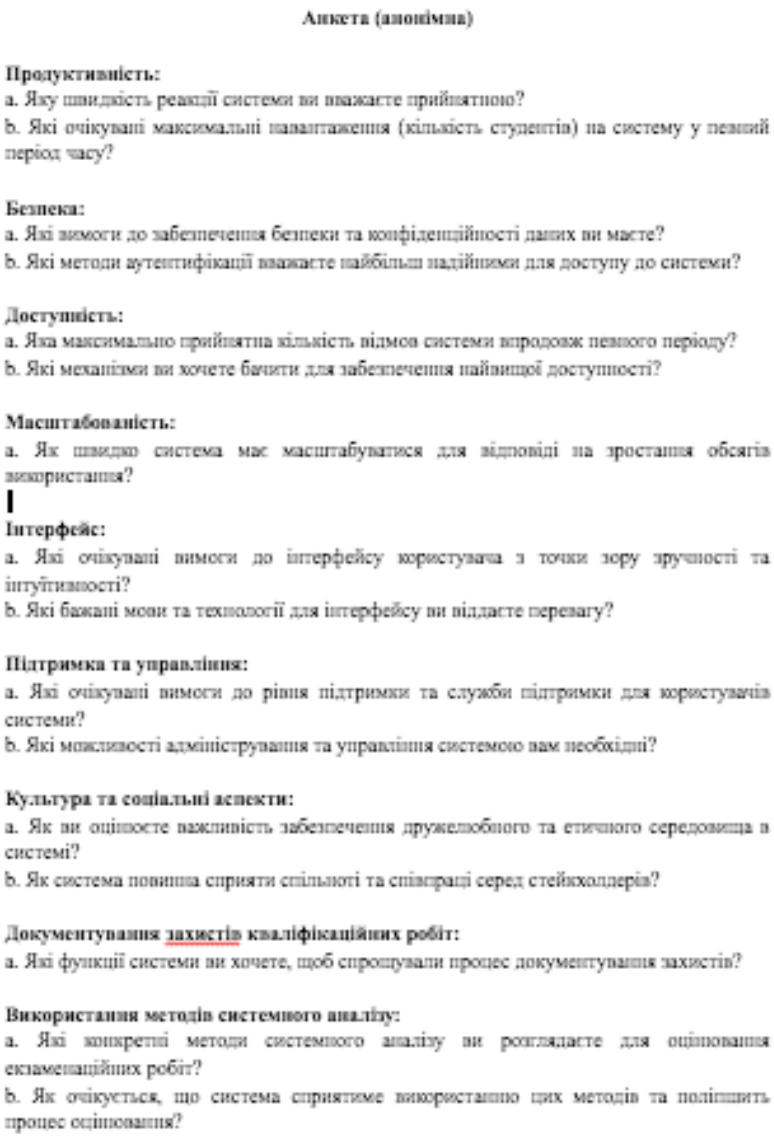
В нашому випадку визначені наступні - *секретар комісії, члени комісії, голова комісії, науковий керівний, студент, який проходить процедуру захисту.* Клієнт-замовник вважається кафедра.

Далі спробуємо провести аналіз нефункціональних вимог. Є багато способів збору нефункціональних вимог (вимог до якості): інтерв’ю та опитувальники для клієнтів, quality attributes workshops, реверс-інжинірінг систем, що вже існують.

Було сформовано анкету для викладачів, які мали досвід членства в комісії. Питання в анкеті спрямовані на отримання додаткової конкретної інформації про очікування та потреби членів комісії щодо документування та методів оцінювання (Рис. 1). Це дозволило краще врахувати їхні потреби та вимоги при розробці проєкту.

По завершенню збору вимог ми стикнулись із деякими розбіжностями у відповідях, які були враховані. Результати опитування будуть наведені в дисертаційній роботі.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)



Анкета (анонімна)

Продуктивність:
a. Яку швидкість реакції системи ви вважаєте прийнятною?
b. Які очікувані максимальні навантаження (кількість студентів) на систему у певний період часу?

Безпека:
a. Які вимоги до забезпечення безпеки та конфіденційності даних ви маєте?
b. Які методи аутентифікації вважаєте найбільш надійними для доступу до системи?

Доступність:
a. Яка максимально прийнятна кількість відмов системи впродовж певного періоду?
b. Які механізми ви хочете бачити для забезпечення найвищої доступності?

Масштабованість:
a. Як швидко система має масштабуватися для відповіді на зростання обсягів використання?

Інтерфейс:
a. Які очікувані вимоги до інтерфейсу користувача з точки зору зручності та інтуїтивності?
b. Які бажані мови та технології для інтерфейсу ви віддасте перевагу?

Підтримка та управління:
a. Які очікувані вимоги до рівня підтримки та служби підтримки для користувачів системи?
b. Які можливості адміністрування та управління системою вам необхідні?

Культура та соціальні аспекти:
a. Як ви оцінюєте важливість забезпечення дружелюбного та етичного середовища в системі?
b. Як система повинна сприяти спільноті та співпраці серед стейкхолдерів?

Документування захистів кваліфікаційних робіт:
a. Які функції системи ви хочете, щоб спрощували процес документування захистів?

Використання методів системного аналізу:
a. Які конкретні методи системного аналізу ви розглядаєте для оцінювання екзаменаційних робіт?
b. Як очікується, що система сприятиме використанню цих методів та поліпшить процес оцінювання?

Рисунок 1 - Анонімна анкета для членів комісії

Проведений аналіз функціональних вимог (architecturally significant requirements). У вимогах визначені фактори, що суттєво впливають на вибір архітектурного стилю та інструментів (наведені нище).

Аналіз обмежень, які має клієнт. Розробка на часній ініціативі підтримується обмеженими людськими та технічними ресурсами. Це може вплинути на швидкість та масштабність реалізації проєкту.

Отже, кафедра, яка визначена як клієнт проєкту, має наступні обмеження:

- бюджетні обмеження;

- часові обмеження; це може вплинути на обсяг та складність функціоналу, який може бути реалізований;

- технічні обмеження; проєкт повинен відповідати можливостям та обмеженням Ruby on Rails. Прив'язка до обладнання - вузьке місце. Обмежений обсяг пам'яті, обмеження часу виконання операцій, обмежені можливості вводу/виводу, нетрадиційні інтерфейси користувача та ін. У випадку системи DDP прив'язка до обладнання користувачів відсутнє, бо для використання сайту користувач має змогу сам обирати технічне обладнання;

- людські ресурси; ініціатива на часну відповідальність також вимагає відчуття обмежених ресурсів у вигляді робочого часу та кількості команди. Це вплине на обсяг та темп розробки;

- масштабність; фінансові обмеження впливають на можливість масштабування проєкту в майбутньому. Розширення функціоналу або інтеграція додаткових можливостей можуть бути обмеженими.

Отже, обмеженнями є фінансові та ресурсні аспекти, а також технічні та організаційні обмеження, які можуть вплинути на обсяг, швидкість та можливість майбутнього розвитку проєкту.

Для системи DDP було обрано наступний стек технологій:

- для бекенду та фронтенду - Ruby on Rails;

- для бази даних використовується Postgres.

Також на сервісі HOSTIA було придбано домен - lildoc.hhos.net (трафік домену необмежений, наразі дійсний до 22.08.2025), надалі планується придбання місця для БД.

Для контролю версій було обрано Github.

В процесі роботи виконується своєчасне документування процесу розробки, імплементацій розробника. Здійснюється ретельна перевірка залежних сервісів та дотримання безпекових обмежень.

Прецеденти - це функціональні можливості або послуги, які надає система (у випадку системи DDP, цифрова система для секретаря комісії). Нище наведені ідентифіковані основні прецеденти та їх зв'язки.

Збереження інформації та обробка даних:

– Опис: Система зберігає та обробляє певний обсяг інформації про кваліфікаційні роботи та оцінки.

– Актори: Секретар комісії.

– Зв'язки: Секретар може вводити, редагувати та переглядати дані кваліфікаційних робіт та оцінок.

– Визначення оцінки за різними критеріями:

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

– Опис: Система надає можливість члену комісії точніше визначити оцінку кваліфікаційних робіт, використовуючи різні критерії.

– Актори: Член комісії.

– Зв'язки: Член комісії може переглядати критерії та виставляти оцінки.

– Формування протоколів захистів:

– Опис: Система дозволяє швидко формувати протоколи захисту кваліфікаційних робіт.

– Актори: Секретар комісії.

– Зв'язки: Секретар комісії може створювати та редагувати протоколи захистів.

На рисунку 2 представлена діаграми прецедентів для системи DDP. Ця діаграма представляє основні функціональні можливості системи та взаємодію з акторами, які будуть взаємодіяти з цією системою.

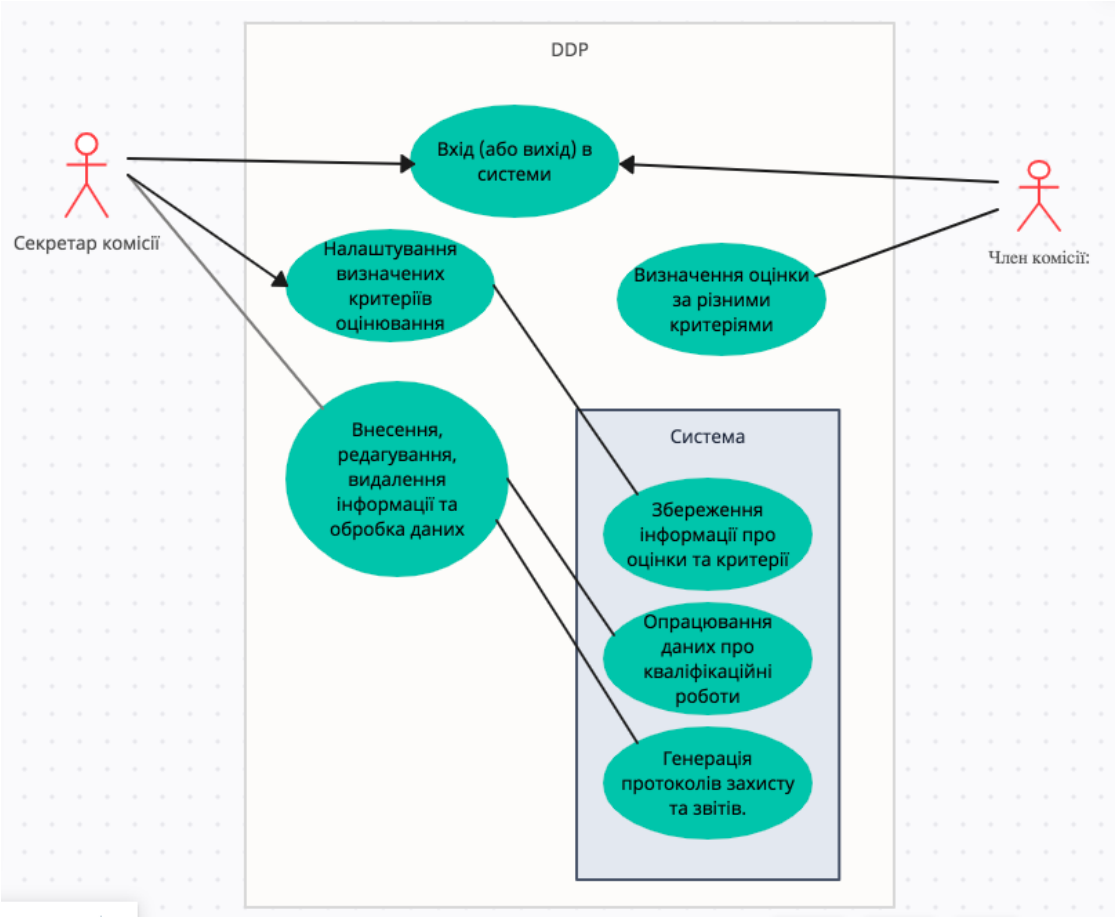


Рисунок 2 - Діаграма прецедентів системи DDP

Для проектування DDP проєкту розроблено ще декілька типів діаграм, які допоможуть краще зрозуміти та візуалізувати різні аспекти системи та її функціональності. Нижче представлені:

- діаграма класів (рис. 3): дозволяє представити основні класи системи та їх взаємозв'язки; включає об'єкти, такі як кваліфікаційні роботи, оцінки, користувачі, протоколи тощо.

- діаграма послідовності (рис. 4.1-2): дозволяє відобразити послідовність виконання операцій та взаємодій між системними компонентами. В розробці буде використовуватись розширення AppMap для редактору коду VSCode, яке відображає дані AppMap як інтерактивні діаграми;

- діаграма активності (рис. 5): дозволяє відобразити потоки роботи та процеси у вашій системі;

- діаграма сутності-зв'язку (рис. 6): дозволяє змоделювати структуру бази даних, сутності та їх взаємозв'язки.

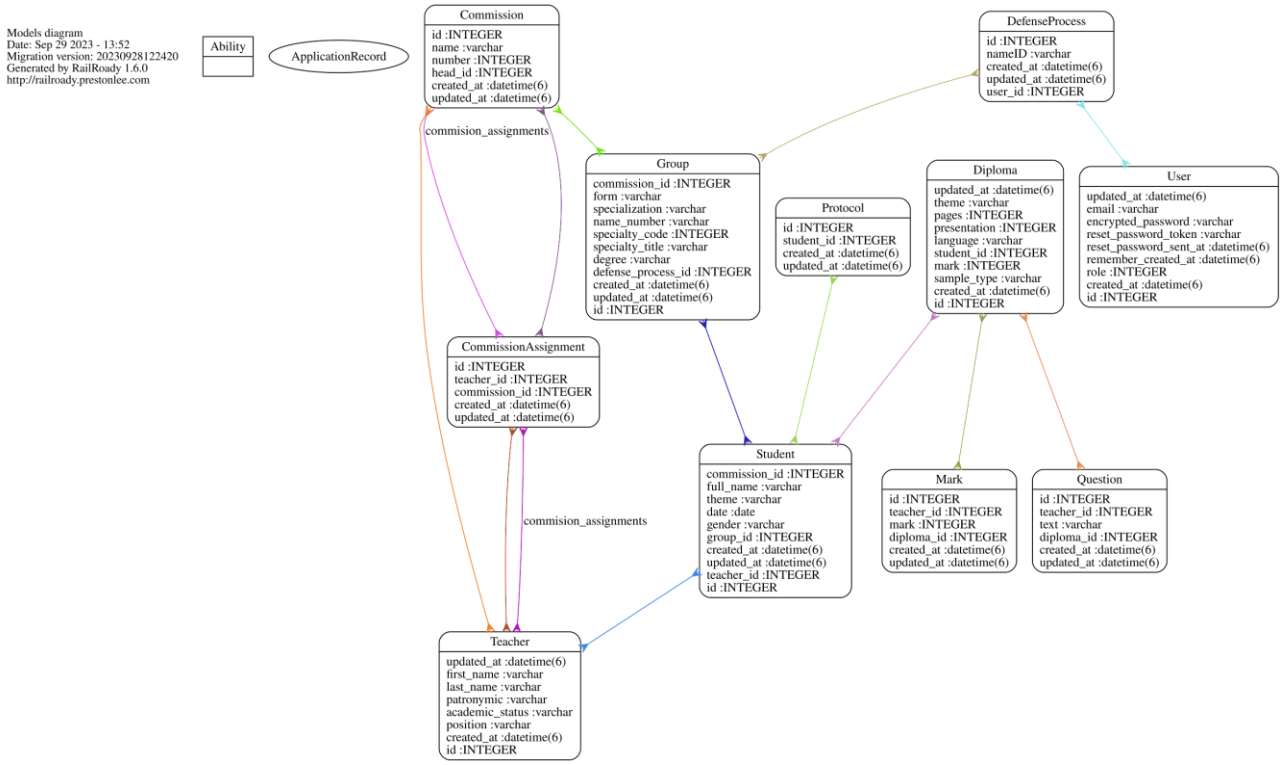


Рисунок 3 - Діаграма Класів (Class diagram) проєкту DDP

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Рисунок 4.1 - Фрагмент діаграми послідовності проєкта DDP, перегляд групи студентів (1)

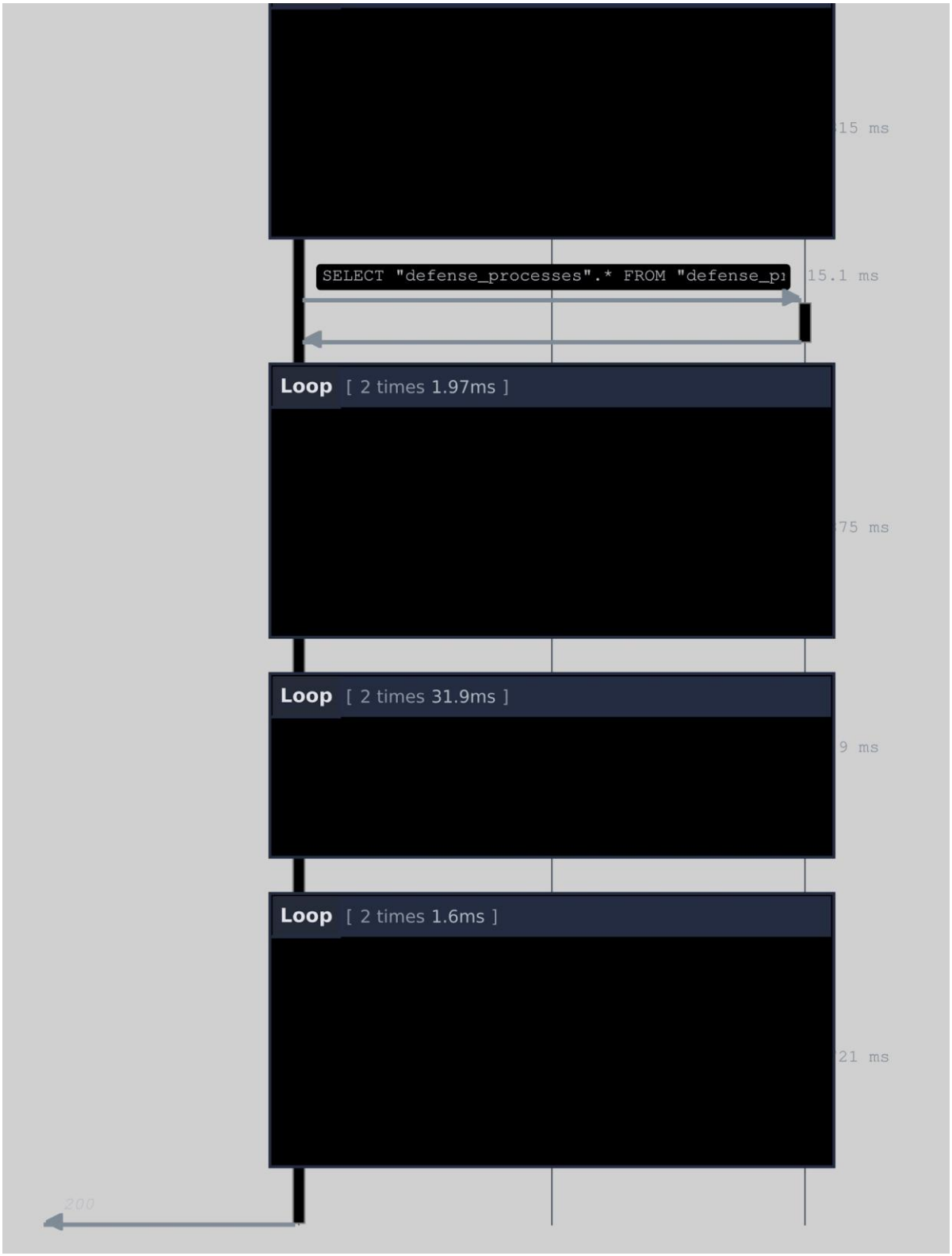


Рисунок 4.2 - Фрагмент діаграми послідовності проєкта DDP, перегляд групи студентів (2)

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

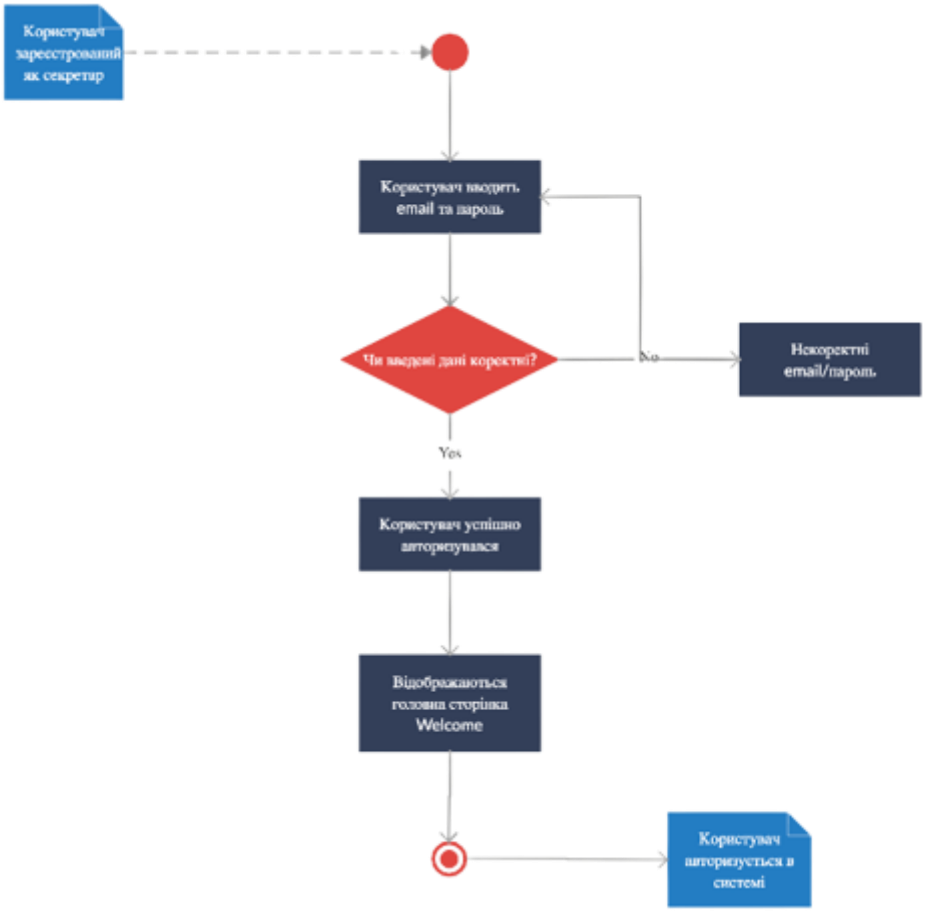


Рисунок 5 - Діаграма активності проєкту DDP,
авторизація користувача-секретаря

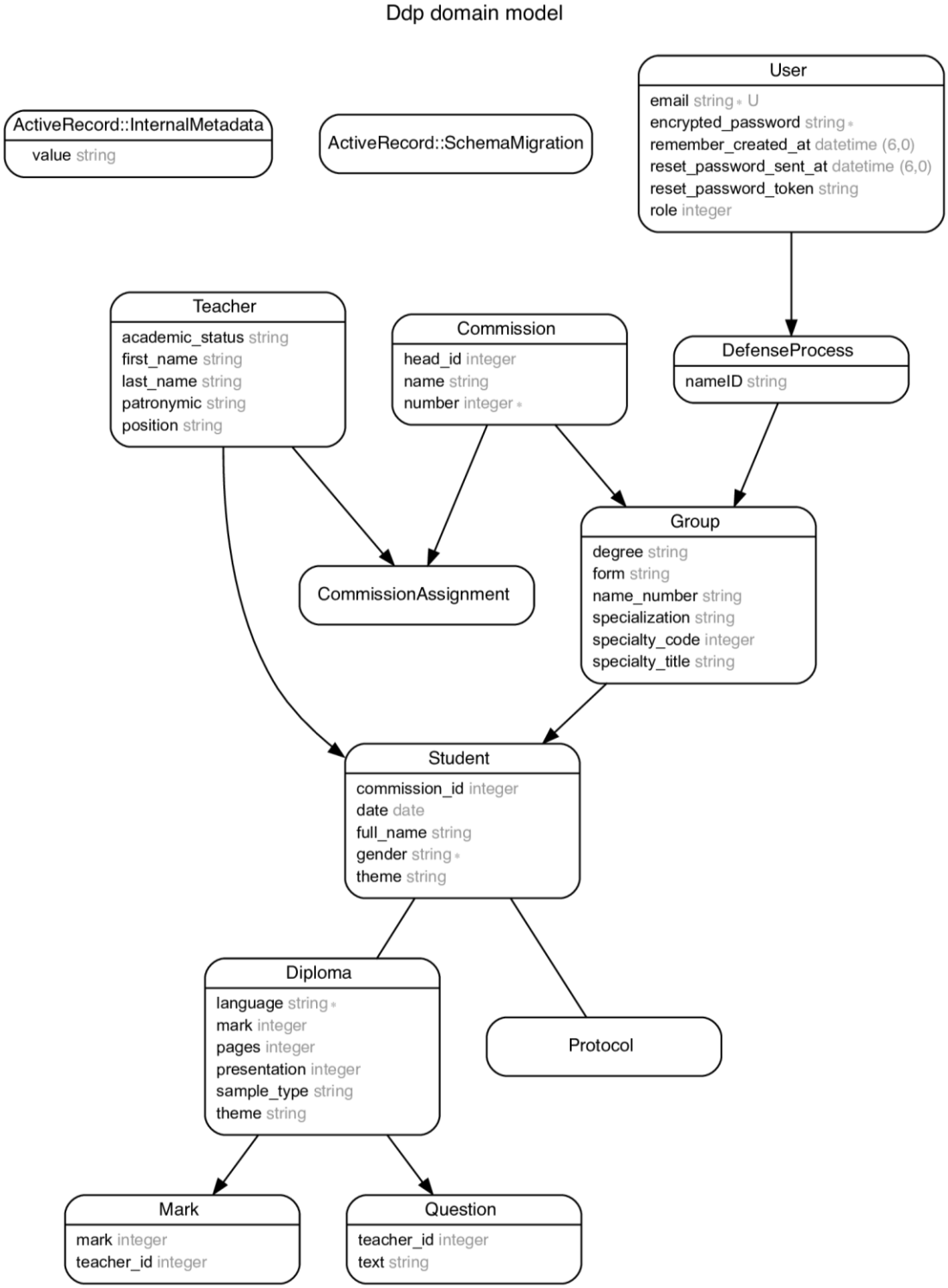


Рисунок 6 - Діаграма сутності-зв'язку (ER Diagram) проєкту DDP

Варто зазначити, що розробка веб-додатку на Ruby on Rails має свої особливості, які варто врахувати при проектуванні архітектури [4]. Нижче наведено загальний опис архітектури веб-додатку DDP проєкту, що буде побудований на Ruby on Rails.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Архітектура за моделлю-вид-контроллер (Model-View-Controller, MVC):

1. Моделі (Models):

- визначення логіки доступу до бази даних та обробки даних.

2. Види (Views):

- визначаєте представлення даних для користувача (HTML, CSS, JavaScript);

- видача даних користувачу в зручному для сприйняття вигляді.

3. Контролери (Controllers):

- керують взаємодією між моделями та видами;

- обробка HTTP-запитів від користувачів та визначають, яка дія повинна відбутися.

Інші ключові складові:

4. Маршрутизація (Routes):

- вказує, які URL-шляхи відображати на які контролери та дії;

- визначає маршрути для користувачів та URL-адреси додатку.

5. База даних (PostgreSQL):

- зберігання даних про кваліфікаційні роботи, оцінки, користувачів тощо.

6. Інтерфейс користувача:

- розробка інтерфейсу користувача з використанням HTML, CSS та JavaScript, можна використовувати фреймворки або бібліотеки, такі як Bootstrap або jQuery.

7. Система аутентифікації та авторизації:

- використання готових бібліотек для реалізації аутентифікації та авторизації (гем Devise) [5].

8. Тестування:

- реалізація тестування, включаючи модульні та інтеграційні тести, щоб переконатися у правильності роботи додатку та його компонентів.

Висновки з даного дослідження і перспективи подальших розробок за даним напрямом. Отже, в статті досліджені існуючі рішення для автоматизації робочого місця секретаря комісії, показані результати розробки загальної архітектури DDP проєкту, яка базується на принципах Ruby on Rails та паттерну MVC.

Представлений новий варіант архітектурного рішення веб-сервісу для автоматизації роботи членів екзаменаційної комісії. Наведені діаграми допо-

можуть краще розібратися з архітектурою та функціональністю DDP проєкту під час написання коду, спростять комунікацію в команді та сприятимуть ефективній розробці та впровадженню продукту.

ЛІТЕРАТУРА

1. Концепція розвитку цифрової економіки та суспільства України на 2018-2020 роки. – URL: https://zakon.rada.gov.ua/laws/show/67-2018-%D1%80#n14
2. Романовська І. Автоматизація роботи секретаря екзаменаційної комісії національного університету “Львівська політехніка” / Романовська І., Пелещишин А. Вісник Національного університету “Львівська політехніка”, № 879, 2017. – с. 100.
3. Software Architecture in Practice 3th Edition Len Bass, Paul Clements, Rick Kazman Addison-Wesley, 25 вер. 2012 р. - 624 стор.
4. Using Rails for API-only Applications [Електронний ресурс]. –
Режим доступу: https://guides.rubyonrails.org/api_app.html– 12.05.2022р
5. Ruby on Rails Technology Stack [Електронний ресурс]. –
Режим доступу: https://www.railscarma.com/technology-stack/ – 05.05.2022р.

REFERENCES

1. Concept of development of the digital economy and society of Ukraine for 2018-2020. – URL: https://zakon.rada.gov.ua/laws/show/67-2018-%D1%80#n14
2. Romanovska I. Automation of the work of the secretary of the examination board of the National University "Lviv Polytechnic" / Romanovska I., Peleshchyshyn A. Bulletin of the National University "Lviv Polytechnic", No. 879, 2017. - p. 100.
3. Software Architecture in Practice 3th Edition Len Bass, Paul Clements, Rick Kazman Addison-Wesley, 25 September 2012 p. - 624 pages.
4. Using Rails for API-only Applications [Electronic resource]. –
Access mode: https://guides.rubyonrails.org/api_app.html– 12.05.2022p
5. Ruby on Rails Technology Stack [Electronic resource]. –
Access mode: https://www.railscarma.com/technology-stack/ – 05.05.2022p.

Received 28.11.2022.
Accepted 30.11.2022.

Architectural solution for the ddp (diploma defense project) web application to document the examination process

Analysis of recent research and publications. The primary source of information about using Ruby on Rails is the official RoR documentation website.

After researching scientific papers and textbooks on architecture, theoretical aspects that should be taken into account when developing web services were collected.

Research objective. The aim of this work is to investigate existing architectural solutions for automating the work of the examination committee and to develop an architectural solution for creating a software product based on it to increase efficiency and improve the quality of the defense process.

Presentation of the main research material. The main stakeholders were identified - the secretary of the commission, commission members, commission chair, academic

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

supervisor, student undergoing defense procedures. The client-customer is considered the department.

A questionnaire was proposed for teachers to determine non-functional requirements. This allowed us to better consider their needs and requirements in project development.

Analysis of functional requirements (architecturally significant requirements) has been conducted. The requirements define factors that significantly influence the choice of architectural style and tools.

The constraints include financial and resource aspects, as well as technical and organizational constraints, which can impact the volume, speed, and possibility of future project development.

For the DDP system, the following technology stack was chosen: Ruby on Rails for backend and frontend; Postgres for the database.

Additionally, a domain was purchased on the HOSTIA service - lildoc.hhos.net (domain traffic is unlimited, currently valid until 08/22/2025), andplans are in place to purchase database hosting.

Github was chosen for version control. The design, class, sequence, activity, entity-relationship diagrams for theDDP system were formed and presented. The defined architecture of the DDP project:

- Follows Model-View-Controller (MVC) pattern.

- Components: Models, Views, Controllers, Routing, Database (PostgreSQL), User Interface, Authentication and Authorization System, Testing.

Conclusions from this study and prospects for further developments in this direction. The article examines existing solutions for automating the secretary of the commission's workplace, presents the results of developing the general architecture of the DDP project based on Ruby on Rails principles and the MVC pattern.

A new architectural solution for a web service to automate the work of the examination commission members is presented.

Селівьорстова Тетяна Віталіївна - кандидат технічних наук, доцент, Кафедра інформаційних технологій і систем, ННІ «Інститут промислових та бізнес технологій», Український державний університет науки і технологій.

Андрюхіна Маргарита Василівна - аспірант, асистент кафедри інформаційних технологій і систем, ННІ «Інститут промислових та бізнес технологій», Український державний університет науки і технологій.

Selivyorstova Tatjana - candidate of technical science, assistant professor, Department of information technology and systems, Scientific and educational institute «Institute of industrial and business technologies», Ukrainian state university of science and technologies.

Andriukhina Marharyta Vasylivna - Postgraduate student, Assistant of the Department of Information Technologies and Systems, Institute of Industrial and Business Technologies, Ukrainian State University of Science and Technology.