УДК 004.4:004.5:004.6

ВИКОРИСТАННЯ ХМАРНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ПРОГРАМНИХ ДОДАТКІВ

М. С. Пасєка

Івано-Франківський національний технічний університет нафти і газу, вул. Карпатська, 15, Івано-Франківськ, 76019, Україна

Розглянуто теоретичні та практичні засади послідовності розробки програмних додатків, складових великих інформаційних систем із використанням хмарних технологій, які забезпечують прискорення розробки та мінімізацію фінансових витрат на програмний проект. Наведений приклад розробки моделей бізнес-логіки взаємодії компонентів інформаційної системи, розробки схеми проекту і взаємозв'язку між модулями та структури необхідної бази даних, а також розроблений інтерфейс взаємодії користувача із додатком на базі хмарних технологій.

Ключові слова: *бізнес-логіка, розробка додатків, хмарні технології, інтерфейс користувача, бази даних, структурна схема проектування, архітектурний шаблон.*

Постановка проблеми. В умовах стрімкого розвитку інформаційно-комунікаційних технологій розширення безкоштовних хмарних сервісів актуальним є впровадження та використання у промисловості новітніх інформаційних сервісів на основі спільного користування конкурентними технологіями. У зв'язку з цим з'явилась актуальна потреба в аутсорсингу інформаційно-комунікативних технологій — хмарних послуг (IT-послуги) у системі розробки програмних додатків [1]. Поняття «Аутсорсинг IT» означає передавання компанією будь-якого IT-процесу (програми, функції, роботи) або якоїсь частини сторонній організації, що забезпечує професійні IT- послуги. Це, зокрема, підтримка функціонування інформаційно-довідкових, експертних систем, гарантування інформаційної безпеки баз та банків даних підприємств, зберігання й опрацювання значних обсягів даних, надання апаратних ресурсів. Аутсорсинг вирішує питання скорочення фінансових і часових витрат на впровадження, підтримку й модернізацію IT-інфраструктури [3, 4]. Він забезпечує конвергенцію інформаційно-комунікаційних середовищ, а саме: зближення різноманітних електронних технологій підвищення бізнес-вимог до стабільності і доступності IT-послуг.

Мета статті — проаналізувати сучасні підходи у використанні хмарних технологій для професійної розробки програмних додатків, а також дослідити особливості використання хмарних технологій у професійній діяльності програмістів.

Теоретичні відомості. Хмарні технології (cloud computing) визначають вільний спосіб доступу до зовнішніх інформаційно-комунікативних ресурсів у вигляді різноманітних інтернет-сервісів. Термін «Хмарні технології» запропонував Рамнат

Челлаппат (Ramnath Chellappa), визначивши його як «обчислювальну парадигму, за якої межі обчислювальних елементів залежатимуть від економічної доцільності, а не тільки від технічних обмежень». Із появою першої технології, з'явився доступ до додатка через веб-сайт, а саме – програмне забезпечення (ПЗ) як сервіс (Software as a Service [SaaS]) [6]. У наш час хмарні технології привертають велику увагу до розробки програмних додатків, а саме: розробки структурних схем, використанні архітектурних шаблонів та розробки баз даних.

Структурна схема — схема, яка визначає основні функціональні частини виробу, їх взаємозв'язки та призначення. Під функціональною частиною розуміють складову частину схеми: елемент, пристрій, функціональну групу, функціональну ланку. Структурна схема призначена для відображення загальної структури проекту, тобто його основних блоків, вузлів, частин та головних зв'язків між ними. Із структурної схеми має бути зрозуміло, як проект працює в основних режимах роботи, а також те, як взаємодіють його частини. Елементи структурної схеми можна позначити довільно, хоча потрібно дотримуватись загальноприйнятих правил виконання схем.

Модель-вигляд-контролер — архітектурний шаблон, який використовують під час проектування та розробки програмного забезпечення. Цей шаблон ділить систему на три частини: модель, вигляд та керування даними. Його застосовується для відокремлення даних (модель) від інтерфейсу користувача (вигляду), щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни у моделі даних могли здійснюватися без змін інтерфейсу користувача [8].

Мета шаблону — гнучкий дизайн програмного додатка, який повинен полегшувати подальші зміни чи розширення додатка, а також давати змогу повторно використовувати окремі компоненти додатка. Крім того, використання цього шаблону у великих системах приводить до впорядкованості структури додатка і спрощує сам проект.

База даних (БД) — упорядкований набір логічно взаємопов'язаних даних, що призначені для задоволення інформаційних потреб користувачів. Головне завдання БД — гарантоване збереження значних обсягів інформації (так звані записи даних) та надання доступу до неї користувачеві або ж прикладному додаткові [2, 5]. Отже, БД складається з двох частин: збереженої інформації та системи керування нею. Для забезпечення ефективності доступу записи даних організовують як множину фактів (елемент даних). У реляційних базах даних та плоских базах даних таблиця — це набір елементів даних (значень), які організовані завдяки моделі з різними іменами вертикальних стовпчиків (полів/доменів) і горизонтальних рядків (кортежів). Таблиця має визначену кількість стовпців, тоді як кількість рядків може змінюватися у різні моменти (обмеження лише розміром файла).

Проектування — процес створення проекту, прототипу, прообразу майбутнього додатка, стану та способів його розробки. У проектуванні застосовують системний підхід, який полягає у встановленні структури системи, типу зв'язків, визначенні атрибутів тощо.

Інтерфейс користувача — засіб зручної взаємодії користувача з інформаційною системою. Сукупність засобів для обробки та відображення інформації, максимально пристосованих для зручності користувача; у графічних системах інтерфейс користувача реалізовується багатовіконним режимом, змінами кольору, розміру, видимості (прозорість, напівпрозорість, невидимість) вікон, їхнім розташуванням, сортуванням елементів вікон, гнучкими налаштовуваннями як самих вікон, так і окремих їхніх елементів (файли, папки, ярлики, шрифти тощо), доступністю багатокористувацьких налаштувань. Можна виокремити такі види графічного інтерфейсу користувача (GUI — Graphical user interface): простий (типові екранні форми та стандартні елементи інтерфейсу, що забезпечує сама підсистема GUI); істинно-графічний, двовимірний (нестандартні елементи інтерфейсу та оригінальні метафори, що реалізовані власними засобами програми або сторонньою бібліотекою) [7].

Графічний інтерфейс є «дружнім» для користувачів, котрі розпочали знайомство з комп'ютером з графічного інтерфейсу. У додатках обробки графіки він найчастіше є єдино можливим. Інтерфейс має важливе значення для будь-якої програмної системи і є невід'ємною її складовою, орієнтовано, передусім на кінцевого користувача. Саме через інтерфейс користувач судить про додаток у цілому, більше того, часто рішення про використання додатку користувач приймає по тому, наскільки йому зручний і зрозумілий інтерфейс. Трудомісткість проектування і розробки інтерфейсу досить велика. За оцінками фахівців, у середньому вона становить більшу частину часу реалізації проекту.

Методи реалізації з використанням хмарних технологій

Метою розробки проекту є створення веб-сайта операцій із нерухомістю, а саме: продаж, купівля, оренда та обмін квартир чи приміщення для замовників. До основних вимог побудови структурної схеми належать: легкість впровадження і супроводу; ефективна маршрутизація запитів; розподіл бізнес-логіки та інтерфейсу користувача; автономність програмних компонентів; вибірка полів бази даних за кількома різними критеріями; можливість масштабування проекту. Згідно із проектом використовуватимемо архітектурний шаблон MVC (model-view-controller). Оскільки серед програмних вимог виникає необхідність вибірки з БД за багатьма критеріями, то у проект залучено реляційну базу даних MySQL. Легкості впровадження, супроводу, ефективної маршрутизації можна досягти завдяки популярному фреймворку Yii з зазначеною вище архітектурою MVC, а також використовуючи готові програмні рішення (компоненти, модулі). Цей фреймворк працює на одній з найбільш популярних у web мові програмування — PHP, дозволяє конфігурувати memcached-систему чи файловий кеш. Масштабування проекту таким чином стає більш швидким. Як альтернативний варіант розглядали фреймворк Laravel, проте кількість розробників, що працюють із ним, значно менша, а реалізація інтерфейсу частково не відповідає вимогам. Структурну схему проекту зображено на рис. 1.

Фронт-контролер забезпечує попереднє опрацювання запиту та передачу управління необхідному контролерові. Контролер відповідає за взаємодію між

моделлю і представленням. Модель працює з даними та правилами бізнес-логіки роботи. Представлення працює з користувацьким інтерфейсом. Модулі між собою взаємодіють таким чином: користувач переходить на сайт, сервер обробляє запит, запускаючи скрипт ініціалізації (index.php); скрипт ініціалізації (index.php) створює екземпляр фронт-контролерові і запускає його на виконання; фронт-контролер отримує детальну інформацію про запит від компонента Запит; фронт-контролер із запиту визначає потрібний *Контролер* та *Дію* за допомогою компонента *Маршрутизатор*; фронт-контролер створює екземпляр визначеного контролера для подальшої обробки запиту цим екземпляром. Контролер через знайдену *Дію* визначає необхідний *Метод*. Перевіряють фільтри, що зв'язані з цим *Методом*.

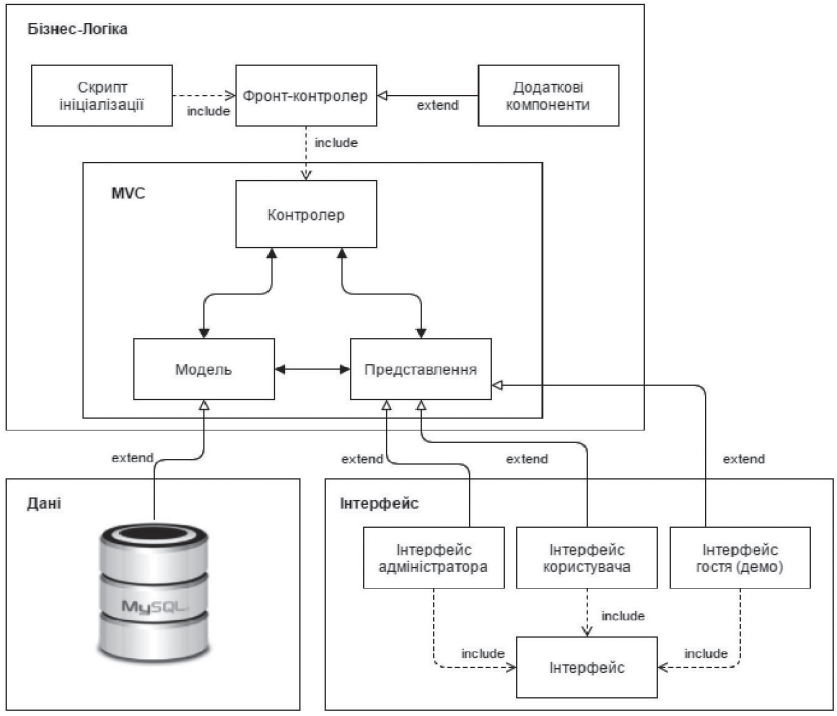

Рис. 1. Структурна схема проекту

Якщо фільтр дозволить — Метод виконується: метод дістає Модель згідно з запитом; метод підключає Представлення, передаючи в нього Модель; представлення відображає властивості Моделі; метод завершує формування представлення, виводячи результат. Взаємодію між модулями зображено на рис. 2.

Використовуємо хмарний сервіс моделювання баз даних «dbdesigner.net», основною перевагою якого є можливість створення SQL-коду на основі змодельованої структури бази даних (таблиць із полями та зв'язок між ними). При цьому можна визначити, під який тип бази даних буде створюватись код. Існує підтримка MySQL, MS SQL Server, SQLite, PostgreSQL, Oracle.

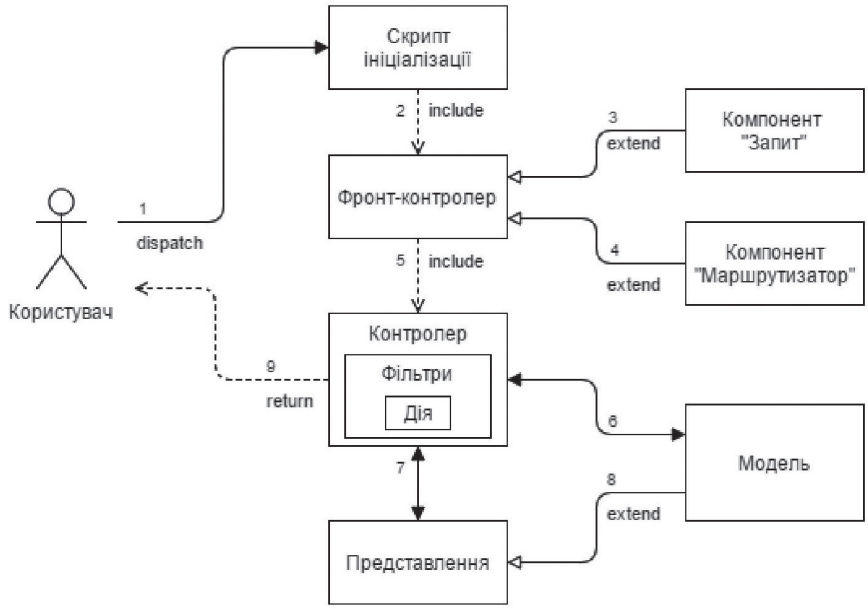

Рис. 2. Схема взаємозв'язку між модулями

Створити базу даних можна, вибравши пункт головного меню «Schema», підпункт «New…», після чого з'явиться нове діалогове вікно (рис. 3). У цьому вікні зазначають ім'я бази даних та її тип. Під час вибору типу «Generic» наприкінці моделювання створювати SQL-код можна відносно кожного типу. Якщо ж на початку вибрати конкретну базу даних, то й генерувати код можна буде лише для неї (рис. 3).

Отримаємо порожню робочу область. Для створення таблиці використовують пункт головного меню «Insert», підпункт «Table». Оскільки метою розробки програмного додатка веб-сайт є здійснення операцій із нерухомістю (продаж, купіля, оренда, обмін), то треба розпочати проектування саме з таблиці «Квартири», додавши потрібні поля за допомогою кнопки «Add field» (рис. 4).

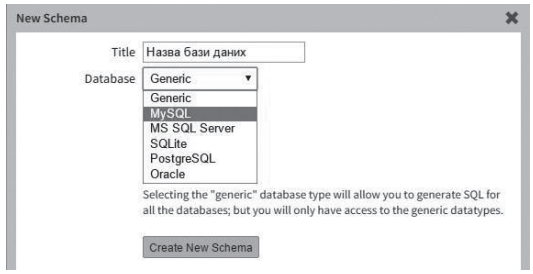

Рис. 3. Діалогове вікно створення нової моделі БД

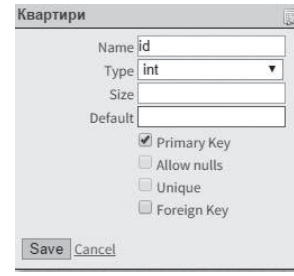

Рис. 4. Діалогове вікно додавання поля у таблицю

Зберегти поле можна, натиснувши кнопку «Save». Також не треба забувати виставляти індекси та вказувати залежності в разі потреби. У результаті таблиця «Квартири» отримала структуру (рис. 5).

Згідно з таблицею «Квартири» потрібно створити таблицю «Користувачі», що міститиме дані користувачів. Також має бути таблиця «Привілеї» з описом привілеїв користувачів (наприклад, демо, повний, заблокований, адміністратор). Пізніше моделюються таблиці «Міста», «Райони», «Тип стін квартир», «Тип оголошення» та «Фото квартир» із відповідним зв'язком. Результуючу базу даних зображено на рис. 6.

Рис. 5. Структура таблиці «Квартири»

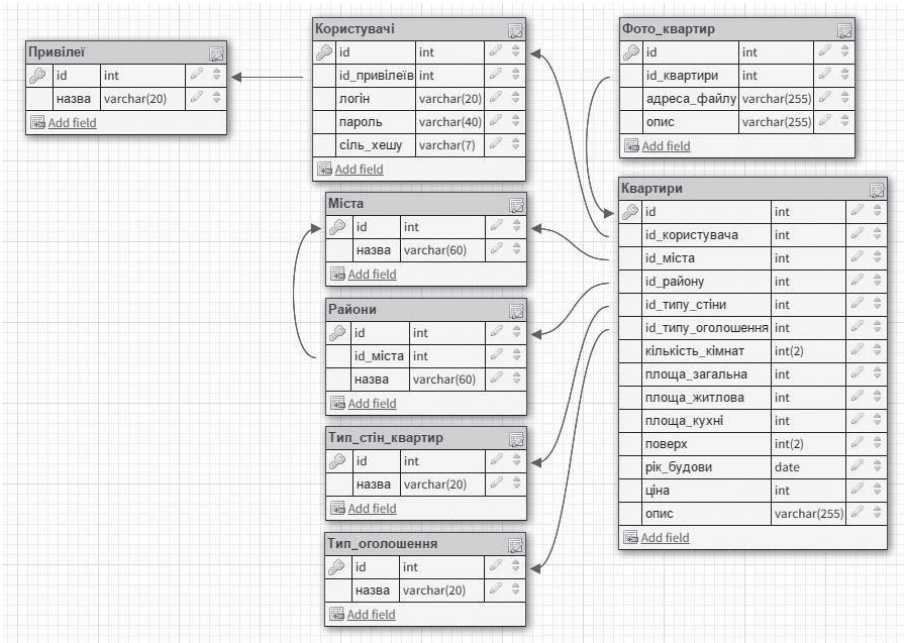

Рис. 6. Структура необхідної бази даних

Далі базу даних було збережено за допомогою пункту головного меню «Export», підпункту «Sql…» та збережено файл. Треба зазначити, що більшість користувачів сервісу «dbdesigner.net» зберігають схеми БД на самому сервісі за допомогою підпункту «Save», пункту головного меню «Schema».

Перед проектуванням користувацького інтерфейсу потрібно підготувати докладний перелік функціональності користувачів. Він дасть змогу врахувати всі функціональні вимоги і краще зрозуміти особливості майбутньої системи. На його основі потрібно робити висновок, які з функцій вимагають цілого процесу, які — просто окремої сторінки, а яким буде досить одної кнопки. Орієнтуючись на складений перелік, можна підготувати схему прототипу навігації по сайту. Після цього треба намалювати діаграми переходів між сторінками, які об'єднують сторінки системи у рамках конкретних процесів. Завдяки цьому видно, як користувачі працюватимуть із сайтом і як виконуватимуть конкретні завдання. Все це дасть змогу краще передбачити та спроектувати інтерфейс користувача. Проектування інтерфейсу може відбуватись за допомогою багатьом різноманітним сервісів. У нашому випадку було використано сервіс https://moqups.com/, оскільки він давно добре себе зарекомендував. Використовуючи сервіс, було спроектовано головну сторінку сайта, більшу частину дизайну якої використовуватимуть на усіх сторінках як шаблон (макет). Макет проектувався відносно поставлених специфікацій вимог з урахуванням відповідності стандартам ISO 9001 (рис. 7).

Також було спроектовано сторінки особистого кабінету, списку пропозицій та детального перегляду конкретної пропозиції (рис. 8), авторизації, реєстрації (рис. 9), відновлення пароля.

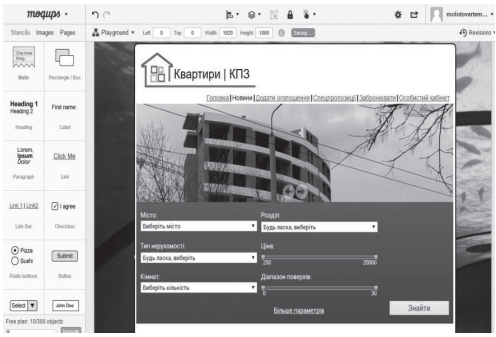

Рис. 7. Частина макету головної сторінки сайта

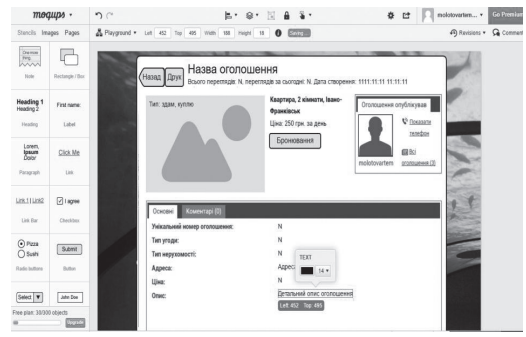

Рис. 8. Інтерфейс сторінки перегляду оголошення

У процесі проектування використовували різноманітні компоненти онлайн-сервісу (розміщені в лівому меню та зображені на рис. 9). Найчастіше з них використовувались текстові поля, кнопки, елементи меню, зображення та ін.

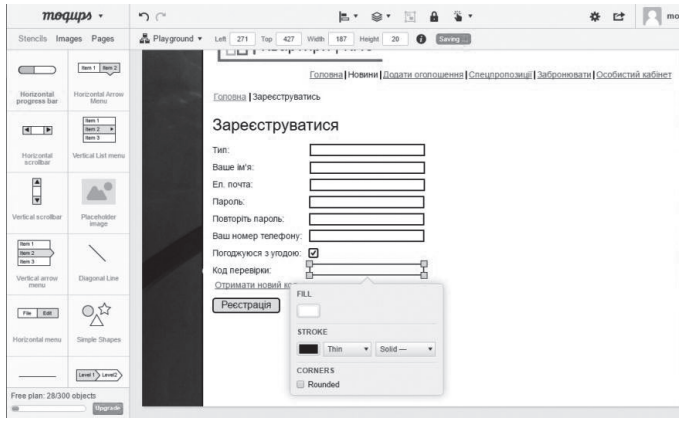

Рис. 9. Інтерфейс сторінки реєстрації користувача

Завдяки їм проектування інтерфейсу користувача здійснювалось значно швидше, ніж під час використання інших сервісів чи паперу. Також було присутнє зручне автоматичне розсташування при дублюванні компонента, розміщення та вирівнювання їх по сітці.

Висновки. Використання хмарних сервісів у розробці програмного додатка забезпечило пришвидшення розробки та економію грошових коштів на придбання ліцензійних програмних продуктів. За допомогою хмарного сервісу «Moqups» реалізовано макети (прототипи) інтерфейсу користувача, які в подальшому будуть реалізовані у програмному коді додатка. Інтерфейс спроектований згідно з вимогами ISO 9001, що забезпечує зручну, швидку та інтуїтивну роботу користувача з вебдодатком. Візуалізація структурної схеми відбувалась за допомогою хмарного сервісу draw.io, який описав взаємодію між модулями залежно від вимог проекту та застосовуваних рекомендацій. За допомогою хмарного сервісу dbdesigner.net було побудовано структуру бази даних, що використовуватимуть у подальшій розробці програмного забезпечення Встановлено взаємозв'язок між полями таблиць, а також згенеровано SQL — код створення таблиць.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Биков В. Ю. Хмарні технології, ІКТ-аутсорсинг і нові функції ІКТ підрозділів освітніх і наукових установ. Інформаційні технології в освіті. № 10. 2011. С. 8–23.
2. Проектування та використання сховищ даних для опрацювання результатів оцінювання знань студентів / Дурняк Б. В., Пасєка Н. М., Пасєка М. С., Ерстенюк О. В. Науковий вісник НЛТУ України: зб. наук.–техн. праць. Львів : РВВ НЛТУ України. 2015. Вип. 25.9. С. 365–373.
3. Кадемія М. Ю., Кобися В. М. Можливості, що надають хмарні технології. Хмарні технології в освіті : матеріали Всеукраїнського науково-методичного Інтернет-семінару. Кривий Ріг : Видавничий відділ КМІ, 2012. С. 66–67.
4. Литвинова С. Г. Хмарні технології як засіб розбудови інноваційної школи [Електронний ресурс]. URL: http://www.zoippo. zp.ua/pages/el_gurnal/pages/vip14.html.

5. Пасєка Н. М. Використання когнітивних методів для набуття компетентності педагогів у галузі інформатики. Науковий вісник НЛТУ України: зб. наук.-техн. праць. Львів : РВВ НЛТУ України. 2015. Вип. 25.6. С. 359–364.
6. Перспективи розвитку ринку хмарних обчислень в Україні: переваги та ризики [Електронний ресурс]. Аналітична записка [сайт]. URL: http://www.niss.gov.ua/articles/1191/#_ftn2.
7. Шиненко М. А., Сороко Н. В. Перспективи розвитку програмного забезпечення як послуги для створення документів електронної бібліотеки на прикладі Microsoft Office 365 [Електронний ресурс]. Інформаційні технології і засоби навчання: електронне наукове фахове видання [сайт]. 2011. Том 26, № 6 (2011). URL: http://www.nbuv.gov.ua/e-journals/ITZN/em5/emg.html.
8. Шишкіна М. П., Попель М. В. Хмарно орієнтоване освітнє середовище навчального закладу: сучасний стан і перспективи розвитку досліджень. Інформаційні технології і засоби навчання. 2013. № 5. С. 66–80.

REFERENCES

1. Bykov, V. Yu. (2011). Khmarni tekhnolohiyi, IKT-autsorsynh i novi funktsiyi IKT pidrozdiliv osvitnikh i naukovykh ustanov. Informatsiyni tekhnolohiyi v osviti,10, 8–23 (in Ukrainian).
2. Durnyak B. V., Pasyeka, N. M., Pasyeka, M. S. & Erstenyuk, O. V. (2015). Proektuvannya ta vykorystannya skhovyshch danykh dlya opratsyuvannya rezul'tativ otsinyuvannya znan' studentiv. Naukovyy visnyk NLTU Ukrayiny: zb. nauk.–tekhn. prats', 25.9, 365–373 (in Ukrainian).
3. Kademiya, M. Yu. & Kobysya, V. M. (2012). Mozhlyvosti, shcho nadayut' khmarni tekhnolohiyi. Khmarni tekhnolohiyi v osviti : materialy Vseukrayins'koho naukovo-metodychnoho Internet-seminaru, 66–67. (in Ukrainian).
4. Lytvynova, S. H. Khmarni tekhnolohiyi yak zasib rozbudovy innovatsiynoyi shkoly. Retrieved from http://www.zoippo. zp.ua/pages/el_gurnal/pages/vip14.html (in Ukrainian).
5. Pasyeka, N. M. (2015). Vykorystannya kohnityvnykh metodiv dlya nabuttya kompetentnosti pedahohiv u haluzi informatyky. Naukovyy visnyk NLTU Ukrayiny: zb. nauk.-tekhn. prats', 25.6, 359–364 (in Ukrainian).
6. Perspektyvy rozvytku rynku khmarnykh obchyslen' v Ukrayini: perevahy ta ryzyky. Analytychna zapyska. Retrieved from http://www.niss.gov.ua/articles/1191/#_ftn2 (in Ukrainian).
7. Shynenko, M. A. & Soroko, N. V. (2011). Perspektyvy rozvytku prohramnoho zabezpechennya yak posluhy dlya stvorennya dokumentiv elektronnoyi biblioteky na prykladi Microsoft Office 365. Informatsiyni tekhnolohiyi i zasoby navchannya: elektronne naukove fakhove vydannya, Vol. 26, 6 (2011). Retrieved from http://www.nbuv.gov.ua/e-journals/ITZN/em5/emg.html (in Ukrainian).
8. Shyshkina, M. P. & Popel', M. V. (2013). Khmarno oriyentovane osvitnye seredovyshche navchal'noho zakladu: suchasnyy stan i perspektyvy rozvytku doslidzhen'. Informatsiyni tekhnolohiyi i zasoby navchannya, 5, 66–80 (in Ukrainian).

THE USE OF CLOUD TECHNOLOGIES TO DEVELOP SOFTWARE APPLICATIONS

M. S. Pasyeka

Ivano-Frankivsk National tTechnical University of Oil and Gas,
15 Karpatska St., Ivano-Frankivsk, Ukraine,
pms.mykola@gmail.com

Theoretical and practical bases of the sequence of development of software applications as components of larger information systems using cloud technologies have been reviewed that accelerate the development and minimize the financial cost of a software project. An example of the development models of the business logic of the components interaction of the information system, the development of the schematic diagram showing the relationship between the modules and the structure of the required databases has been given as well as the interface for user interaction with the application based on cloud technologies has been developed.

Keywords: *business logic, application development, cloud technology, user interface, database, structural design pattern, architectural pattern.*

Стаття надійшла до редакції 12.12.2016.
Received 12.12.2016.