УДК 656.96

# ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНА ТЕХНОЛОГІЯ РОЗРОБЛЕННЯ ТРАНСПОРТНО-ІНФОРМАЦІЙНОГО ПОРТАЛУ

**Алексієв В.О.**, д.т.н., проф. каф. інформаційних систем ХНЕУ ім. С. Кузнеця, **Наумов В.С.,** д.т.н., зав. каф. інформаційних технологій та мехатроніки ХНАДУ, **Суховаров М.А.**, інженер НВП «Система+Сервіс», **Васютіна Г.О.**, інженер каф. інформаційних технологій та мехатроніки ХНАДУ

Стрімкий розвиток Інтернет-технологій є одним з впливових важелів, що сприяє трансформації сучасного ринка транспортних послуг. Процеси перевезення вантажів стають більш прозорими для користувачів логістичних сайтів та відповідних сервісів транспортно-інформаційних порталів. В свою чергу, зазначені сервіси перетворюються на більш зручні, набувають значнішого рівня інформативності, що дозволяє швидко узгодити сумісну роботу перевізників, експедиторів та вантажовласників [1]. На сьогоднішній день особливої актуальності набуває проблема розробки нових або вдосконалення існуючих Інтернет-ресурсів та сервісів, що обслуговують суб'єктів транспортного ринку. Це обумовлено потребами користувачів у наявності нових інформаційних систем, які б відповідали сучасному рівню розвитку Інтернет-технологій [2]. Першим етапом вирішення описаної проблеми є визначення оптимальної архітектури для побудови відповідних рішень рівня транспортного-інформаційного порталу.

Організація логістичного Інтернет-центру на базі транспортно-інформаційного порталу потребує визначення певних архітектурних рішень щодо застосування спеціалізованої програмно-апаратної платформи, яка відповідатиме вимогам: масштабованості, спрямованості на користувача, надійності, зручності розгортання та подальшого супроводження системи на етапі її функціонування. Слід відзначити, що такий транспортно-інформаційний портал повинен поєднувати декілька функцій. Його основним завданням є здійснення обробки даних про вантажі, що переміщуються за певними маршрутами та напрямками, про наявний рухомий склад і додаткової знань про відповідні транспортні процеси. Непрямим завданням порталу є підвищення цінності ресурсу для користувачів завдяки наданню додаткових інформаційних сервісів та їх соціалізації.

На основі аналізу типових завдань транспортно-інформаційного порталу можна визначити основні рівні застосування відповідної інформаційно-комунікаційної технології: мережевий рівень та фізичний сервер, процеси та програмні рішення рівня операційної системи серверної платформи, а також рівень комунікацій та взаємодії із сервісами порталу на базі персональних комп'ютерних систем, планшетів та смартфонів. Звичайним підходом у проектуванні архітектури відповідної системи є розробка монолітного рішення на базі веб-серверу та бази даних. Однак, це обмежує розробників

щодо подальшого супроводження та вдосконалення рішення рівня Інтернет-порталу.

Монолітна архітектура у більшості рішень основується на модульному принципі побудови програмних систем в основі якого застосовується об'єктно-орієнтований підхід. Це дозволяє проектувати складні та надійні веб-орієнтовані системи. Для застосування цього підходу слід враховувати, що вдосконалення коду, навіть незначної частини відповідної системи, потребує виконання процедур збирання (компонування) всього проекту, а також проведення відповідних автоматичних та ручних тестів проекту. Тільки після виконання наведених процедур веб-рішення розгортається на сервері проекту. Тому, більш доцільним, у якості методології розроблення програмної архітектури транспортно-інформаційного порталу, є вибір концепції мікросервісів [3], що дозволить уніфікувати та спростити процес розробки.

Мікросервісна архітектура веб-рішення базується на визначенні основних бізнес-процесів, для здійснення яких створюється програмне рішення, та відповідного поділу системи на окремі компоненти (мікросервіси) із слабкими зв'язками. Фактично взаємодія між всіма мікросервісами реалізується на основі програмного інтерфейсу API (Application Programming Interface), який імплементація якого здійснюється завдяки REST-інтерфейсу. Такий підхід дозволяє розгалузити процес розробки веб-рішення, виконувати розробку всіх сервісів окремо від додаткових та, у разі обмежених ресурсів, – сконцентруватися тільки на розробці основних сервісів.

Для транспортно-інформаційного порталу у якості основного сервісу можна виділити виконання логістичних завдань. Для цього слід обрати відповідні інструментальні засоби розробки та серверну інфраструктуру проекту. Оскільки мікросервіси є автономними та фактично не залежать один від одного, для додаткових систем можна обрати зовсім протилежні рішення щодо застосування засобів розробки та технологій побудови веб-компонентів. Відзначимо, що єдиним рішенням, що пов'язує сервіси, у обраній архітектурі, є серверна платформа.

Для формування ефективного підґрунтя масштабованої платформи мікросервісів доцільно обрати технологічні рішення на базі хмарних обчислень. Це дозволить виключити залежність від ресурсів окремого серверу, застосовувати абстрактні ресурси, що будуються на основі технологій віртуалізації, та мати можливості для гнучкої трансформації всієї програмно-апаратної інфраструктури системи, що розроблюється.

Раціональним вибором, у разі наявності фізичних обчислювальних ресурсів, є створення приватного хмарного рішення (Private Cloud Computing) на базі розгортання кластеру Proxmox VE. Це відкрита система управління віртуальними машинами та віртуальною мережевою інфраструктурою. У якості платформи віртуалізації застосовуються: Linux-контейнери (LXC) – для запуску виключно операційних систем на базі Linux та KVM – для запуску віртуальних машин незалежно від їх операційної

системи (рис. 1). Користувачу Proxmox VE також мають змогу обрати комерційну передплату, яка відрізняється додатковими сервісами та якістю обслуговування від компанії-розробника (Proxmox Server Solutions).

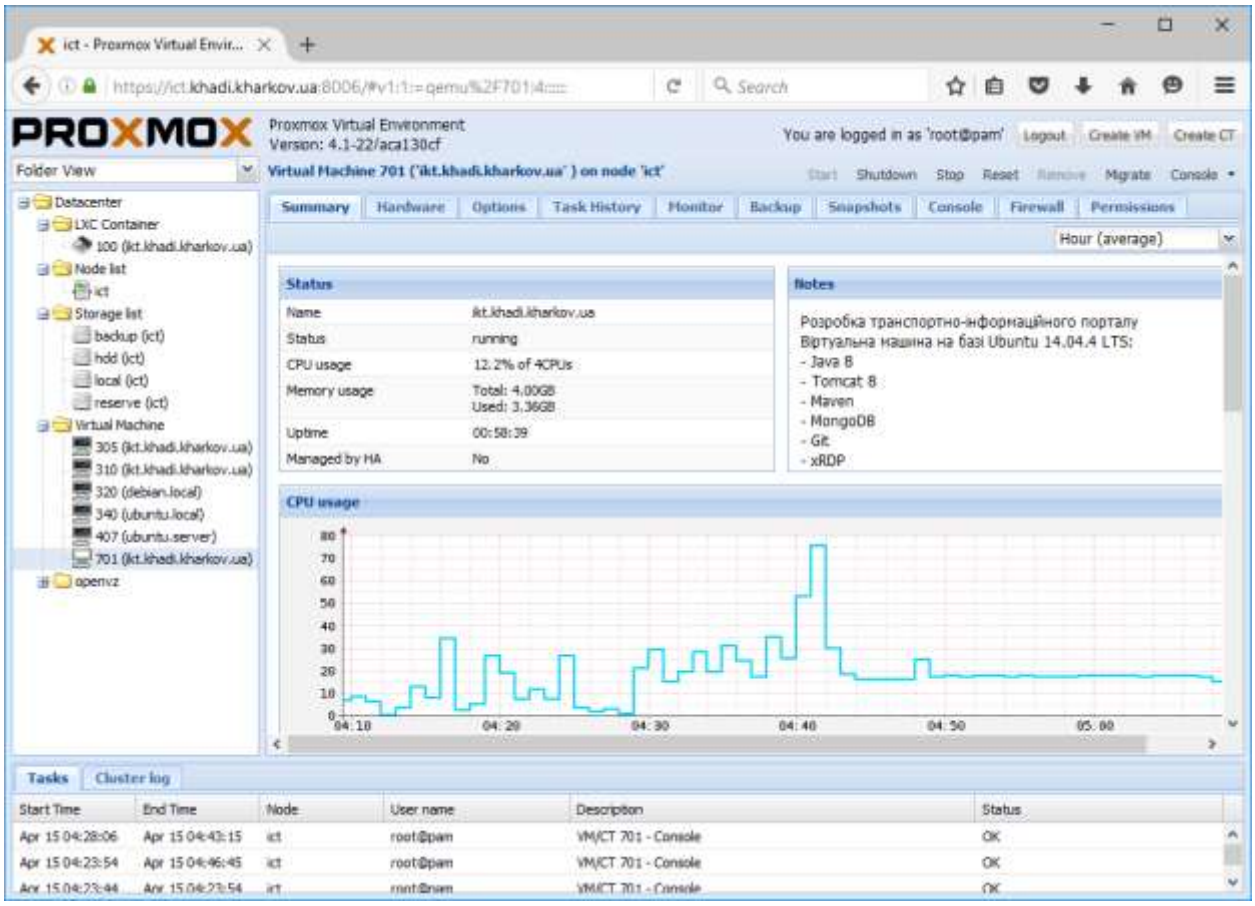


Рисунок 1 – Приклад роботи у середовищі Proxmox VE

Оскільки Proxmox VE є рішенням рівня «інфраструктура як сервіс» (IaaS – Infrastructure as a Service), то у разі потреби у додаткових ресурсах, їх завжди можна залучити завдяки послугам публічних хмарних сервісів (Public Cloud Computing) та, відповідно, – отримати архітектуру рівня гібридного рішення (Hybrid Cloud Computing). Відзначимо, що перевагою застосування приватної хмари є можливість повного контролю за наявними комп'ютерними ресурсами та даними.

На підґрунті визначення платформи розгортання віртуального середовища, що поєднує сервери додатків та віртуальні робочі місця розробників, слід визначити головний напрям у розробці – мову програмування (рис. 2). Це рішення продиктує умови до екосистеми проекту та визначить методологію розробки програмних рішень.

Одним з перспективних рішень на сьогодні є вибір Java як основної мови для програмування веб-систем. Це обумовлено надійністю та простотою застосування конструкцій об'єктно-орієнтованої мови, наданням безпечних засобів розробки веб-додатків, а також їх здатністю до перенесення, завдяки якій Java-програми можуть виконуватися за

управлінням будь-якої операційної системи за наявністю середовища виконання [4]. Також слід відмітити строгу типізацію змінних та виконання відповідних перевірок протягом роботи додатка, що має певне значення для розробки логістичних сервісів, які у свою чергу, передбачають застосування алгоритмів на обчислення даних.

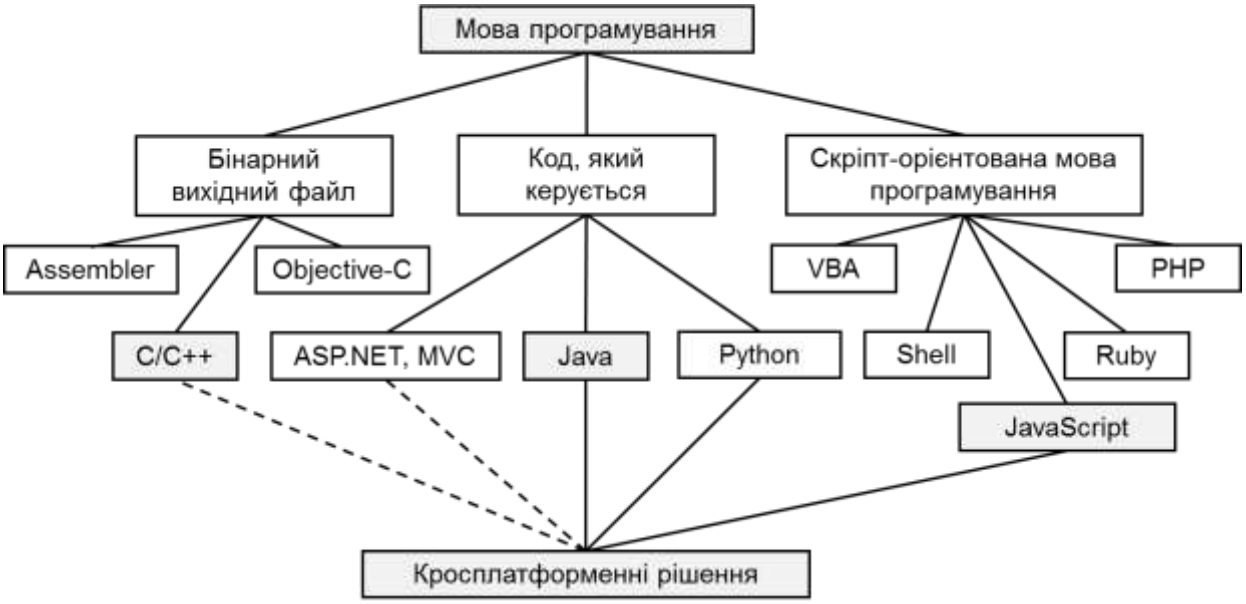


Рисунок 2 – Визначення мови програмування

Для підвищення ефективності розробки на мові Java доцільно застосувати фреймворк Spring. Цей фреймворк спрощує розробку та додає можливості впровадження залежностей (DI – Dependency Injection) в додатку та можливості аспектно-орієнтованого програмування (AOP – Aspect-Oriented Programming) [5]. Впровадження залежностей дозволяє вирішити проблему зв'язків класів. Завдяки DI об'єкти при створенні отримують свої залежності від складової, що координує роботу кожного об'єкта в системі. Об'єкти не створюють та не отримують свої залежності самостійно – залежності до них впроваджуються. З іншого боку – підхід AOP дозволяє оформити функціональність, використовувану в додатку, у вигляді багаторазово використовуваних компонентів.

Поруч з цим фреймворк Spring додає на стороні сервера можливість розробки відповідно моделі MVC (Model-View-Controller – архітектурного шаблону проектування «модель-вигляд-контролер»). Такий підхід дозволяє розгалужувати код веб-додатку на пов'язані складові, які можна розробляти окремо один від одного.

Поруч із вибором платформи програмування, для проекту створення транспортно-інформаційного порталу, не менш важливим є вибір середовища збереження даних (рис. 3). На сьогодні у якості оптимального рішення для організації сховища даних можна визначити систему управління базами даних MongoDB. Це нереляційна база даних, яка характеризується простотою розгортання, масштабування та супроводження рішень на її основі [6].

Завдяки документ-орієнтованій моделі збереження даних у MongoDB, можна швидко розпочати розроблення програмної складової проекту та сконцентрувати зусилля розробників на роботі з даними, але не виконувати проектування схеми бази даних, як це звичайно виконується у проектах на основі реляційних рішень.

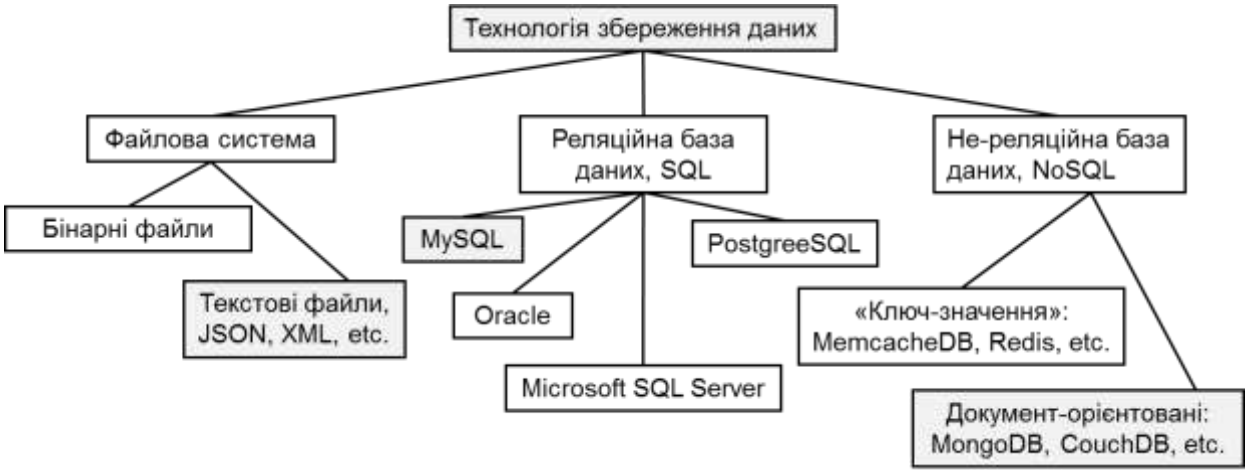


Рисунок 3 – Визначення технології збереження даних

Сучасний веб-додаток прийнято поділяти на дві складові: серверна складова (Back-end) та інтерфейс користувача (Front-end). Якщо вибір засобів розробки серверної складової транспортно-інформаційного порталу цілком базується на екосистемі Java-коду, то для інтерфейсу користувача доцільно обрати традиційні рішення на базі HTML5, CSS3 та JavaScript. Для уніфікації розробки та спрощення виконання верстання статичної складової сайту можна обрати CSS-фреймворк, наприклад, найбільш поширений на сьогодні – Bootstrap.

Для ефективної взаємодії з користувачами порталу, подолання проблем із якістю Інтернет-зв'язку та полегшення інтерфейсу сервісів порталу доцільно проводити розроблення за так методологією SPA – Single-Page Application [7]. Вона передбачає, що сторінка веб-сайту не перезавантажується кожного разу після запиту користувача, а тільки до вмісту ресурсу додаються дані, що довантажуються з серверу. Оптимізувати розробку SPA дозволяє застосування JavaScript-фреймворку AngularJS [8].

Наступною задачею визначення інформаційно-комунікаційної технології розроблення транспортно-інформаційного порталу є узгодження розробки серверної та інтерфейсної складових. Слід відзначити, що обидві складові базуються на веб-фреймворках, які застосовують підхід MVC – Model-View-Controller. Це реалізація парадигми – модель, представлення й контролер, де модель відповідає за структуру даних, представлення – за відображення інформації для користувачів, а контролер – реалізує бізнес-логіку програмної системи. Тому оптимальним є застосування REST-інтерфейсу (рис. 4), який забезпечить ще більшу модульність веб-рішення [9].

У розглянутій схемі взаємодії серверної та інтерфейсної частин транспортно-інформаційного порталу, можна виділити, що складові View та Controller у фреймворці Spring фактично перетворюються на компоненти, що реалізують REST-протокол. Це не задовольняє вимогам щодо модульності серверної частини та підвищує гнучкість всього програмного рішення завдяки можливості розширення цього API на інші мікросервіси, що згодом доповнять загальне веб-рішення.

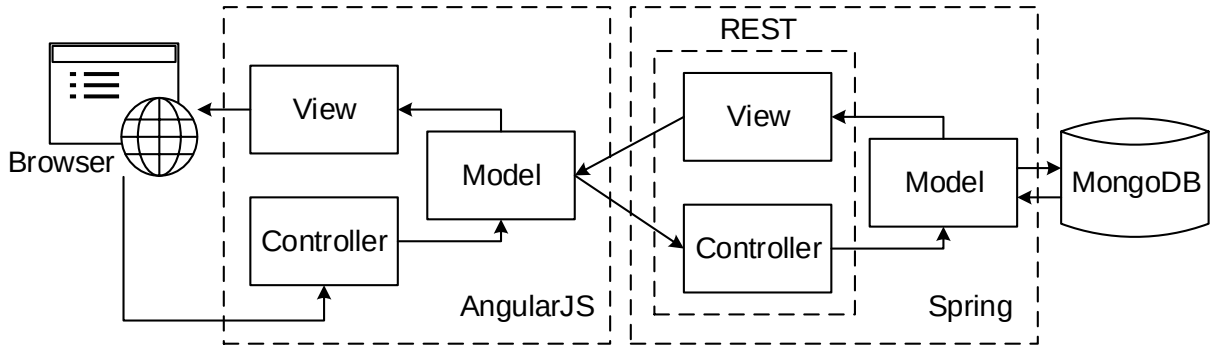


Рисунок 4 – Структура веб-рішення, на основі AngularJS та Spring

На сьогодні можна стверджувати, що всі етапи розробки програмного продукту поєднуються на рівні застосування інтегрованого середовища розробки (IDE – Integrated Development Environment). Для проекту, який основується на технології Java EE та JavaScript доцільно обрати рішення між вільними середовищами: NetBeans IDE, Eclipse IDE та комерційною системою IntelliJ IDEA Ultimate.

Аналіз Інтернет-форумів й спільнот [10], дошок об’яв, таких як Quora, Stack Overflow, DOU.ua та досвід авторів дозволяє визначити переваги рішення IntelliJ IDEA Ultimate (рис. 5), яке ґрунтується на вимогах цієї системи до порівняно невеликого об’єму ресурсів (оперативна пам’ять), що забезпечує зручну роботи у середовищі; більш стабільну роботу IDE; зручної у настройці плагінів для застосування з Spring і Tomcat; наявності інформативної підказки щодо введення коду; покращеного текстового редактору для JavaScript; зрозумілого інтерфейсу настроювання режиму налагодження та ін.

Вибір IDE слід здійснювати не тільки на рівні вирішення завдань організації середовища розробки, а також спільно з визначенням засобів автоматизації розгортання програмного рішення. Такий підхід, поруч із визначенням завдань тестування та контролю якості веб-рішення, відповідає методології DevOps (Development and Operations). Підґрунтям застосування такого підходу є технології віртуалізації. У якості серверної платформи транспортно-інформаційного порталу обрано Proxmox VE, що узгоджується із застосуванням технології віртуалізації VirtualBox як технології налаштування робочого простору розробника.

На рівні організації робочого середовища веб-програміста можна розробити віртуальну машину, яка буде містити всі компоненти розробки. На

цьому рівні фактично не має різниці у виборі середовища віртуалізації. Наступним кроком доцільно застосувати VirtualBox, який підтримує різні формати файлів для віртуальних машин, та конвертувати образ до відкритого формату OVF (Open Virtualization Format). Потім віртуальну машину можна конвертувати до формату, що застосовується у Proxmox VE із KVM (рис. 5).

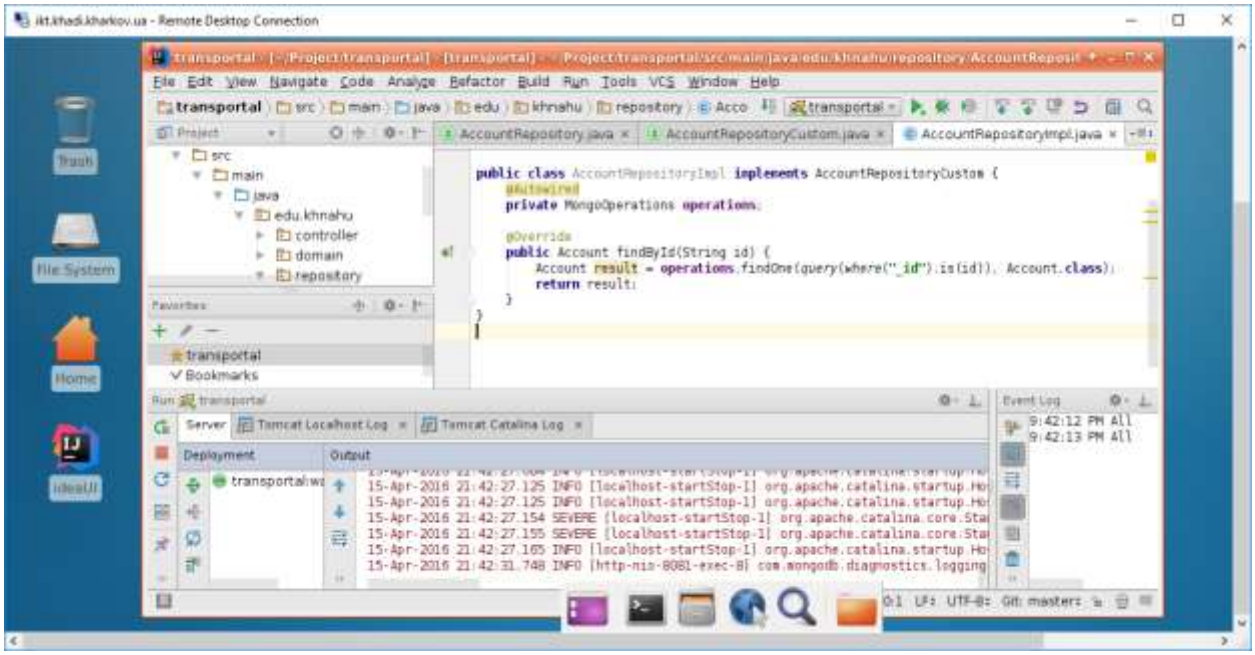

Рисунок 5 – Розробка у віртуальній машині KVM на базі Ubuntu 14.04

Завдяки застосуванню віртуалізації, як на рівні серверних рішень, так й організації робочого простору розробника, досягається злагоджена й прозора конфігурація середовища виконання веб-додатку. Наступним кроком у вдосконаленні цього рішення є вилучення середовища розробки із віртуальної машини та залучення засобів Vagrant. Таке рішення дозволяє отримати майже однакову конфігурацію серверного оточення та середовища розробки, включно із застосуванням усіх ресурсів комп'ютеру розробника для запуску IDE. Однак, у цьому випадку доцільно мати віртуальні машини на стороні серверу для виконання віддаленої розробки. Це дозволить програмувати завдання порталу з комп'ютеру, що має будь-які апаратні властивості (необхідним є тільки якісний доступ до мережі Інтернет).

У підході DevOps важливе місце займає рішення завдань організації репозиторію вихідних кодів проекту. Для цього доцільно застосувати рішення на основі розподіленої системи контролю версій Git [11]. Більш зручним та ефективним рішенням є застосування сервісів, наприклад: GitHub, Bitbucket або у разі самостійного розгортання серверної інфраструктури розробки – GitLab. На цій основі можна організувати розгортання веб-складової порталу за зміною вихідних кодів проекту, тобто вирішити завдання щодо неперервної інтеграції та розгортання веб-рішення.

Слід відзначити, що для визначення складових інформаційно-комунікаційної технології розроблення транспортно-інформаційного порталу

слід враховувати потреби сучасних користувачів у доступі до Інтернет-ресурсів за допомогою портативних пристроїв. Одним із рішень у цьому напрямку є застосування CSS-фреймворку, який забезпечить адаптивність веб-сайту. Однак, для вирішення такої складної задачі, як розроблення логістичного сервісу, слід передбачити створення кросплатформеного мобільного додатку. Найкращім рішенням цього завдання є застосування технологій Apache Cordova та PhoneGap [12], що узгоджується із вибором AngularJS.

Таким чином, визначено основні архітектурні рішення щодо розробки на базі сучасних інформаційно-комунікаційних технологій розподіленої системи транспортно-інформаційного порталу. Пропонуються гнучкі засоби розроблення поруч із залученням формалізованих підходів Java-програмування, а також застосування нереляційної бази даних MongoDB, веб-фреймворків, технологій віртуалізації, хмарних рішень та розроблення мобільних додатків на базі кросплатформених рішень.

## ПЕРЕЛІК ПОСИЛАНЬ


1. Наумов В.С. Информационные системы поддержки принятия решений при транспортном и экспедиторском обслуживании: Монография / В.С. Наумов. – Харьков: ХНАДУ, 2015. – 148 с.
2. Ніконов О.Я. Розроблення та впровадження інтернет-технологій для підвищення ефективності використання транспортних засобів / О.Я. Ніконов, В.О. Алексієв, В.Ю. Улько, Г.І. Середіна // Вісник СевНТУ. – 2013. – Вип. 142. – С. 69–72.
3. Ньюмен С. Создание микросервисов / С. Ньюмен. – СПб.: Питер, 2016. – 304 с.
4. Шилдт Г. Java 8: руководство для начинающих: пер. с англ. / Г. Шилдт. – М.: Вильямс, 2015. – 720 с.
5. Уоллс К. Spring в действии / К. Уоллс. – М.: ДМК Пресс, 2013. – 752 с.
6. Бэнкер К. MongoDB в действии: пер. с англ. / К. Бэнкер – М.: ДМК Пресс, 2012. – 394 с.
7. Monteiro F. Learning Single-page Web Application Development 7 / F. Monteiro. – Packt Publishing, 2014. – 214 p.
8. Козловский П. Разработка веб-приложений с использованием AngularJS / П. Козловский, П. Б. Дарвин. – М.: ДМК Пресс, 2014. – 394 с.
9. Williamson K. Learning AngularJS: A Guide to AngularJS Development / K. Williamson. – O'Reilly Media, 2015. – 325 p.
10. Почему IDEA лучше Eclipse [Електронний ресурс]. Режим доступу: https://habrahabr.ru/post/112749/.
11. Чакон С. Git для профессионального программиста / С. Чакон, Б. Штрауб. – СПб.: Питер, 2016. – 496 с.
12. Liang Y.E. PhoneGap and AngularJS for Cross-platform Development / Y.E. Liang. – Packt Publishing, 2014. – 122 p.