УДК 311.175:305

СТВОРЕНННЯ УНІВЕРСАЛЬНОЇ СИСТЕМИ БРОНЮВАННЯ ЗАСОБАМИ PHP-ФРЕЙМВОРКУ Yii2 ТА СЕРВЕРУ ЧЕРГ RabbitMQ

Танін Антон

Науковий керівник: канд. техн. наук, старший викладач кафедри прикладної математики, статистики та економіки, Нарадовий В.В.

Центральноукраїнський державний педагогічний університет імені Володимира Винниченка, м. Кропивницький, Україна

В статті проілюстровано підхід до вирішення проблеми очікування вільних місць у закладах різного типу, таких як кафе, ресторани, готелі, тощо. Проаналізовані основні можливості системи бронювання, яка допомагає зберегти час в очікувані вільних місць або номерів. Також описано можливості фреймворку ***Yii2*** *як основного засобу розробки системи бронювання. Додатково наведено опис серверу черг* ***RabbitMQ*** *та пошукової системи* ***Sphinx,*** *які використовувались для задоволення поставлених завдань та повноцінної роботи системи. В статті наведено основні поняття фреймворку та приклади роботи інструментів* ***RabbitMQ*** *і* ***Sphinx.***

Ключові слова: Yii2, бронювання, універсальність, черги, пошукова система, RabbitMQ.

Building the universal reservation system by PHP-framework YII2 and server of queues RABBITMQ

A. Tanin

Scientific supervisor: Candidate of Engineering Sciences, Professor of the Department of Applied Mathematics, Statistics and Economics, Naradovy V. V.

The Volodymyr Vynnychenko Central Ukrainian State Pedagogical University, Kropyvnytsky, Ukraine

The article illustrates the approach to solving the problem of waiting for vacancies in various types of institutions, such as cafes, restaurants, hotels, etc. The main features of the booking system are analyzed, which helps to save time in the expected vacancies or numbers. It also describes the capabilities of the Yii2 framework as the main tool for developing a reservation system. In addition, a description of the server of the RabbitMQ queues and the Sphinx search engine, which was used to meet the tasks and the full operation of the system, is described. The article outlines the basic concepts of the framework and examples of RabbitMQ and Sphinx tools.

Key words: Yii2, reservation, versatility, queues, search engine, RabbitMQ.

Постановка проблеми. Кожного дня велика кількість людей відвідує різні заклади, більшість з яких мають свої сайти або мобільні додатки, які дають змогу дізнатись певну інформацію про той чи інший заклад. Основною проблемою є те, що ці сайти або мобільні додатки не мають можливості показати кількість вільних місць або номерів (в залежності від типу закладу). Одним із пунктів досягнення цієї мети є спроба створити універсальну систему бронювання відповідно номерів або місць, яка дасть можливість споживачу виконати запит резервування, використовуючи веб-сайт або мобільний додаток.

Аналіз досліджень і публікацій. Дослідження проблеми електронного бронювання та резервування знаходяться в центрі уваги багатьох зарубіжних та вітчизняних спеціалістів, зокрема Л. І. Алабкіна, І. Енджейчик, І. Т. Балабанова, М. Б. Біржанова та інших, що свідчить про актуальність вирішення даної проблеми [1**].**

Використання універсальної системи, створеної засобами **Yii2**, **RabbitMQ** та пошукового механізму **Sphinx**, дозволяє зменшити навантаження на персонал закладу, що в свою чергу дасть можливість збільшити швидкість обслуговування клієнтів.

Постановка завдання. Дослідити підходи до створення електронної системи бронювання, визначити основні поняття засобів, які можуть використовуватись в розробці системи, оцінити їх можливості, функції та методи використання.

Yii – це високопродуктивний компонентний PHP-фреймворк, призначений для швидкої розробки сучасних веб-додатків. Слово Yii (вимовляється як Йі [ji:]) в китайській мові означає "простий та еволюційний". Також Yii може розшифровуватись як акронім для Yes It Is! Завдяки його компонентній структурі і відмінній підтримці кешування, фреймворк особливо підходить для розробки таких великих проектів як портали, форуми, системи керування вмістом (CMS), інтернет-магазини або RESTful-додатки.

Додатки Yii організовані згідно архітектурного шаблону Модель-Представлення-Контролер (MVC). Моделі являють собою дані, бізнес-логіку та

бізнес-правила; представлення відповідають за відображення даних моделей; контролери приймають вхідні дані від користувача і перетворюють їх у команди для моделей та представлень. Окрім MVC, Yii додаток також має наступні сутності:

- вхідні скрипти: це PHP-скрипти, які доступні напряму кінцевому користувачу додатка. Вони відповідають за запуск циклу обробки запиту.
- додатки: це глобально доступні об'єкти, які відповідають за коректну роботу різних компонентів додатка і їх координацію для обробки запиту.
- компоненти додатку: це об'єкти, зареєстровані в додатку і які надають різноманітні можливості для обробки запитів.
- модулі: це самодостатні пакунки, що включають в себе повністю всі ресурси для MVC. Додаток може бути організовано за допомогою декількох модулів.
- фільтри: це код, який повинен бути виконаний до і після обробки запиту контролерами.
- віджети: це об'єкти, які можуть бути вбудованими у представлення. Вони можуть містити логіку контролера і можуть бути повторно використаними у різних представленнях.

На даний момент існує дві основні версії **Yii:** 1.1 та 2.0. Версія 1.1 є попереднім поколінням і знаходиться у стані підтримки. Версія 2.0 - це повністю переписаний Yii, що використовує останні технології і протоколи, такі як Composer, PSR та простори імен.

RabbitMQ — це брокер повідомлень, який дозволяє взаємодіяти різним програмам за допомогою протоколу AMQP. RabbitMQ являє собою чудове рішення для побудови SOA(сервіс-орієнтованої архітектури) та розподілом відкладених ресурсомістких задач. Його головна задача — приймати та віддавати повідомлення. Його можна представити як поштове відділення: коли Ми кидаємо листа до ящику, Ми впевнені, що його буде доставлено до адресанта. В такій аналогії RabbitMQ є і скринькою, і поштовим відділенням, і

листоношою одночасно. В RabbitMQ, а також в обміні повідомленнями в цілому, використовується така термінологія:

- **Producer**(постачальник) — програма, яка відправляє повідомлення.
- **Queue**(черга) — ім'я "поштового ящика". Вона існує всередині RabbitMQ. Повідомлення проходять через сам сервер і додатки, але все одно вони зберігаються тільки в чергах.
- **Consumer**(приймач) — додаток, який приймає повідомлення. Приймач знаходиться в стані очікування повідомлень.

Сервер черг можна використовувати багатьма методами. Основними є:

- Hello World(using the php-amqplib Client)
- WorkQueues(using php-amqplib)
- Publish/Subscribe
- Routing
- Topics

Метод «**Hello World**» передбачає два PHP-додатки. Постачальник, який посилає одне повідомлення, і приймач, який отримує повідомлення і друкує їх. Цей найпростіший приклад, в якому не показані деталі API PHP-amqplib. Увага концентрується на прості речі, щоб зрозуміти як працює система серверу черг.

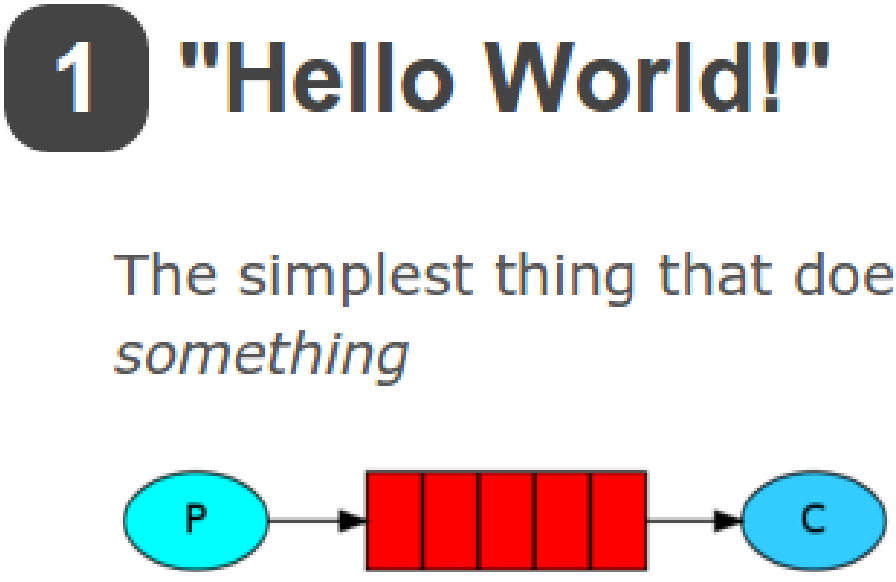

Рис. 1. Метод «Hello World»

Метод «**WorkQueues**». Основна ідея робочих черг - уникнути очікування завершення ресурсоємного завдання. Замість цього, нове завдання, яке надійшло до серверу черг, буде виконане пізніше. Це завдання буде відправлене в чергу у вигляді повідомлення. Додаток обробки працює у фоновому режимі. При запуску декількох додатків завдання будуть розподілені між ними. Ця концепція є особливо корисною в тих випадках, коли неможливо обробляти складну задачу протягом короткого запиту HTTP.

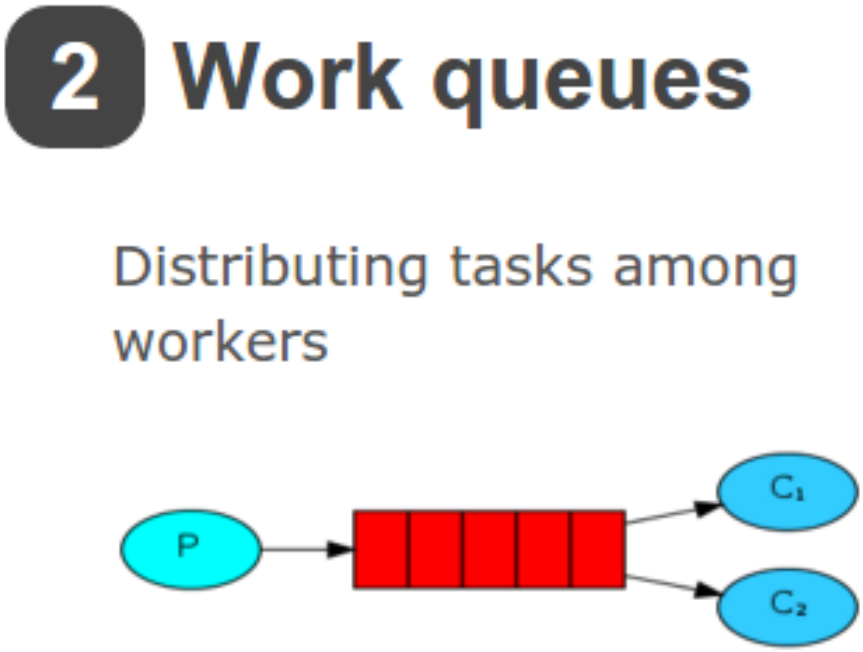

Рис. 2. Метод «WorkQueues»

Метод «**Publish/Subscribe**». Основна ідея в моделі відправки повідомлень Rabbit - Постачальник (producer) ніколи не відправляє повідомлення безпосередньо в чергу. Фактично, досить часто постачальник не знає, чи дійшло його повідомлення до конкретної черги. Замість цього постачальник відправляє повідомлення в точку доступу. У точці доступу немає нічого складного. Точка доступу виконує дві функції:

- Отримує повідомлення від постачальника
- Відправляє ці повідомлення в чергу.

Точка доступу точно знає, що робити з надійшли повідомленнями. Надіслати повідомлення в конкретну чергу, або в кілька черг, або не

відправляти нікому і видалити його. Ці правила описуються в типі точки доступу (exchange type).

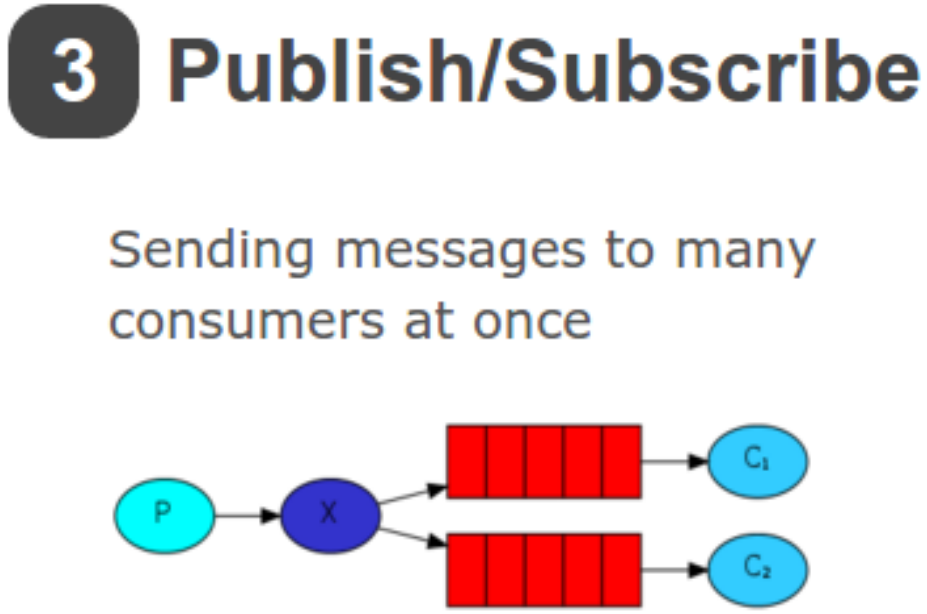

Рис. 3. Метод «Publish/Subscribe»

Метод «**Routing**». Система логування відправила всім отримувачам всі повідомлення. Для розширення системи та фільтрування повідомлень за ступенем важливості потрібно налаштувати систему так, щоб скрипт, що записує лог-файли на диск, не гаяв своє місце на повідомлення з типом warning або info. Даний метод використовує вбудований тип direct. Його алгоритм дуже простий – повідомлення йдуть в ту чергу, binding_key якої збігається з routing key повідомлення. На схемі зображена точка доступу X і дві пов'язані з нею черги. Перша черга пов'язана з binding key = orange, а друга черга має дві зв'язку. Одна з ключем binding key = black, а друга з ключем – green. Повідомлення з routing key = orange будуть направлятися в чергу Q1, а повідомлення з ключем black або green попрямують в чергу Q2. Всі інші повідомлення будуть видалені.

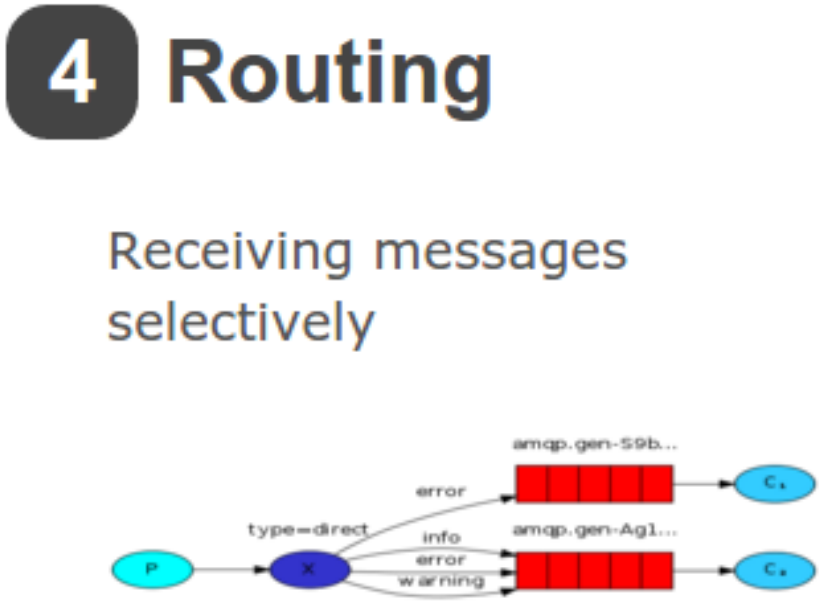

Рис. 4. Метод «Routing»

AMQP. AMQP (Advanced Message Queuing Protocol) - відкритий протокол для передачі повідомлень між компонентами системи. Основна ідея полягає в тому, що окремі підсистеми (або незалежні додатки) можуть обмінюватися довільним чином повідомленнями через AMQP-брокер, який здійснює маршрутизацію, можливо гарантує доставку, розподіл потоків даних, підписку на потрібні типи повідомлень. Архітектуру протоколу розробив John O'Hara з банку JP Morgan Chase & Co. AMQP заснований на трьох поняттях:

- Повідомлення (message) - одиниця переданих даних, основна його частина (зміст) ніяк не тлумачиться сервером, до повідомлення можуть бути причеплені структуровані заголовки.
- Точка обміну (exchange) - у неї відправляються повідомлення. Точка обміну розподіляє повідомлення в одну або кілька черг. При цьому в точці обміну повідомлення не зберігаються. Точки обміну бувають трьох типів: fanout - повідомлення передається в усі причеплені до неї черги; direct - повідомлення передається в чергу з ім'ям, що збігається з ключем маршрутизації (routing key) (ключ маршрутизації вказується при відправці повідомлення); topic - щось середнє між fanout і direct,

повідомлення передається в черзі, для яких збігається маска на ключ маршрутизації, наприклад, app.notification.sms. * - у чергу будуть доставлені всі повідомлення, відправлені з ключами, що починаються на app.notification.sms.

- Черга (queue) - тут зберігаються повідомлення до тих пір, поки не будуть забрані клієнтом. Клієнт завжди забирає повідомлення з однієї або декількох черг.

Основна ідея полягає в тому, що окремі підсистеми (або незалежні додатки) можуть обмінюватися довільним чином повідомленнями через AMQP-брокер, який здійснює маршрутизацію, можливо гарантує доставку, розподіл потоків даних, підписку на потрібні типи повідомлень. Як класичні приклади зазвичай наводяться фінансові додатки, пов'язані, наприклад, з доставкою споживачам інформації про курси цінних паперів в режимі реального часу, також можливо RPC-взаємодія двох підсистем, які не мають зв'язку один з одним (взаємодія через загальний протокол AMQP) і так далі і тому подібне.

Sphinx – система повнотекстового пошуку, розроблена Андрієм Аксьоновим і поширювана за ліцензією GNU GPL. Основні можливості:

- Висока швидкість індексації(до 10-15 Мб/с).
- Висока швидкість пошуку(150-250 запитів в секунду).
- Підтримка розподіленого пошуку.
- Підтримка СУБД MySQL та PostgreSQL.

Перед початком використання системи Sphinx потрібно підготувати базу даних для пошуку. Після створення таблиці та встановлення системи Sphinx виконується його оптимізація в файлах налаштувань системи (sphinx.conf.in). Далі налаштовується пошуковий демон.

Висновки. В результаті проведеного дослідження було виявлено, що проблема бронювання є досить актуальною і над її вирішенням працюють досвідчені вітчизняні та зарубіжні спеціалісти. Аналіз існуючих електронних систем бронювання показує що ці системи, як правило, мають вузькі напрямки

спеціалізації і не завжди містять в собі різні типи закладів. Результатом цього є те, що користувачу не завжди зручно шукати той чи інший заклад, тому питання додаткових досліджень вимагає розробки загальної системи резервування, яка поєднає в собі всі типи закладів, зручний пошук, та коректну можливість бронювання.

Список літератури

1. И. Енджейчик Современный туристский бизнес. – М.: Финансы и статистика, 2003 – 320 с.
2. Создание ознакомительного поискового движка на Sphinx [Електронний ресурс]. – 2010. – Режим доступу до ресурсу: https://habrahabr.ru/post/104690/.
3. Sphinx | Open Source Search Engine [Електронний ресурс] – Режим доступу до ресурсу: http://sphinxsearch.com/.
4. PHP: Sphinx - Manual [Електронний ресурс] – Режим доступу до ресурсу: http://php.net/manual/ru/book.sphinx.php.
5. Sphinx documentation contents [Електронний ресурс] – Режим доступу до ресурсу: http://www.sphinx-doc.org/en/stable/contents.html.
6. RabbitMQ - Messaging that just works [Електронний ресурс] – Режим доступу до ресурсу: https://www.rabbitmq.com/.
7. PHP, MySQL [Електронний ресурс] – Режим доступу до ресурсу: http://www.php.su/.
8. Yii Framework [Електронний ресурс] – Режим доступу до ресурсу: http:// https://yiiframework.com.ua/uk/doc/guide/2/.