МОДУЛЬ ІНФОРМАЦІЙНО-УПРАВЛЯЮЧОЇ СИСТЕМИ ГОТЕЛЬНО-РЕСТОРАННОГО КОМПЛЕКСУ НА ОСНОВІ REST API

Стукаленко М. М., Мельник І.Ю.
Київський столичний університет імені Бориса Грінченка, м. Київ

ВСТУП

Актуальність і постановка проблеми. Готельно-ресторанний комплекс (ГРК) є багатофункціональним підприємством, що поєднує два операційно відмінні, але інформаційно взаємозалежних блоки: засіб розміщення та заклад харчування. Відсутність централізованої інформаційної системи призводить до дублювання записів, неузгодженості статусів між підрозділами та неможливості отримати консолідовану картину стану підприємства в режимі реального часу. Наявні на ринку рішення вирішують лише одну з цих задач: провідні PMS-системи — такі як Oracle Hospitality [1] та Cloudbeds [2] — орієнтовані виключно на готельний блок, тоді як спеціалізовані ресторанні POS-системи — Lightspeed Restaurant [3] та Toast POS [4] — не мають жодної інтеграції з готельним обліком. Повноцінні інтегровані рішення є економічно недоступними для підприємств малого та середнього масштабу, що обумовлює доцільність розробки гнучкого модульного рішення.

Мета дослідження. Метою роботи є проєктування та реалізація модуля інформаційно-управляючої системи (ІУС) ГРК у вигляді REST API на базі фреймворку Django. Система повинна забезпечувати управління ключовими бізнес-процесами готельного та ресторанного блоків через уніфікований програмний інтерфейс з автоматичною документацією у форматі OpenAPI 3.0.

Аналіз останніх досліджень і публікацій. Аналіз ринкових рішень демонструє чіткий поділ між готельним і ресторанним ПЗ: Oracle Hospitality [1] та Cloudbeds [2] пропонують розвинений номерний облік, але обмежену ресторанну функціональність; Lightspeed Restaurant [3] і Toast POS [4] забезпечують повноцінне управління замовленнями, столиками та меню, проте позбавлені будь-якого зв'язку з готельним блоком — фінансова звітність формується відокремлено. Порівняльний аналіз архітектурних стилів REST, GraphQL та gRPC [5] показує, що для CRUD-орієнтованих систем із чітко визначеними сутностями REST забезпечує оптимальне співвідношення функціональності та складності реалізації. Офіційна документація Django [6] та Django REST Framework [7] визначають декларативний підхід до побудови REST API, що суттєво скорочує обсяг шаблонного коду.

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Як результат проведеної роботи, розроблено модуль, реалізований у вигляді Django-проєкту із трьома незалежними застосунками: Hotel,

Restaurant та Dashboard. Такий підхід відповідає принципу розділення відповідальності: кожен застосунок має власні моделі, серіалізатори, view-функції та сервісний шар, що дозволяє незалежно розробляти та розширювати кожен блок. На відміну від Lightspeed [3] і Toast [4], що є закритими SaaS-платформами з фіксованою логікою, розроблений модуль має відкриту архітектуру та повністю адаптується під специфіку конкретного підприємства.

Технологічний стек включає: Django — основний веб-фреймворк з вбудованою ORM та адміністративною панеллю [6]; Django REST Framework— для побудови REST API з декларативними серіалізаторами та декоратором @api_view [7]; drf-spectacular — для автоматичної генерації OpenAPI 3.0 специфікації та Swagger UI; pandas — для читання та трансформації платіжних реєстрів у форматі Excel; SQLite — реляційна СУБД для середовища розробки.

Архітектура системи побудована за трирівневою схемою: HTTP-рівень приймає запити та формує відповіді; рівень серіалізації виконує двосторонню трансформацію між JSON і Python-об'єктами та валідацію вхідних даних; сервісний рівень містить бізнес-логіку, ізольовану від HTTP-контексту. Порівняно з GraphQL та gRPC, REST забезпечує мінімальний поріг входу, широку підтримку інструментарію та відповідність предметній області ГРК [5].

Для готельного підмодуля спроектовано 8 сутностей: WorkPose, Personal, RoomType, Room (зі статусною моделлю AV/NC/NA), Client, Booking, Stuff, Payment. Для ресторанного — 8 сутностей: WorkPose, Personal, Table, Dish, Order (з ManyToMany-зв'язком на Dish через автоматичну проміжну таблицю), Reserve, Raw, Payment. Усі зовнішні ключі визначено з поведінкою SET_NULL при видаленні для збереження фінансової та операційної історії.

Ключовим результатом проведеного дослідження є реалізація двох алгоритмів автоматичного розподілу ресурсів, що становлять ядро бізнес-логіки системи і відрізняють її від стандартних POS-рішень [3; 4].

Алгоритм пошуку вільного номера готелю (find_room()) базується на перевірці перетину часових інтервалів. Зайняті номери визначаються фільтрацією активних бронювань з умовою date_start__lt=date_end AND date_end__gt=date_start. Перший доступний номер заданого типу повертається через .exclude().first(). Алгоритм виконується двома SQL-запитами незалежно від розміру бази даних. Функція create_booking() розраховує вартість як добуток кількості ночей на ціну номера та автоматично підтверджує оплату при знаходженні відповідного платежу в системі.

Алгоритм резервування столика ресторану (create_reserve()) використовує часовий буфер ±2 години: зайняті столики визначаються фільтрацією активних резервувань у діапазоні [time − 2 год, time + 2 год],

перший вільний столик призначається автоматично. Серіалізатор CreateReserveSerializer додатково перевіряє, що запит подається не менш ніж за 2 години до бажаного часу.

У таблиці 1 наведено порівняння функціональних можливостей розробленого модуля з провідними ринковими рішеннями.

Таблиця 1

Порівняння функціональних можливостей систем

Функціональність	**Oracle Hospitality [1]**	**Lightspeed [3] / Toast [4]**	**Cloudbeds [2]**	**Розроблений модуль**
Управління номерним фондом	+	-	+	+
Бронювання кімнат	+	-	+	+
Управління замовленнями	частково	+	частково	+
Резервування столиків	-	+	-	+
Облік меню та страв	-	+	-	+
Консолідований дашборд	-	-	-	+
Імпорт платіжних реєстрів	-	-	-	+
OpenAPI-документація	-	-	-	+
Відкрита архітектура	-	-	-	+

Аналітичний модуль включає три типи ендпоінтів. Функція statistic() формує агреговану фінансову статистику (Avg, Sum, Count платежів) за довільним діапазоном дат через єдиний SQL-запит із агрегатними функціями Django ORM [6]. Функції hotel_dashboard() та restaurant_dashboard() повертають оперативні показники: розподіл ресурсів (GROUP BY через .values().annotate()), відсоток завантаженості, статистику замовлень і бронювань поточного дня, фінансові показники місяця. Консолідований ендпоінт /api/dashboard/ об'єднує показники обох блоків та розраховує загальну виручку комплексу — функціонал, відсутній у жодному з розглянутих ринкових рішень [1–4].

Механізм імпорту платіжних реєстрів (exel_parsing()) зчитує файл .xlsx через pandas, перейменовує стовпці з банківського формату та

виконує зіставлення (reconciliation) платежів із записами бронювань і замовлень. При знаходженні відповідного pay_id автоматично встановлюється прапорець is_pay=True — що автоматизує підтвердження оплати без ручного втручання оператора.

Валідація вхідних даних реалізована на двох рівнях: рівні поля (автоматична перевірка типів через DRF [7]) та рівні об'єкта (метод validate() — перевірка формату ІПН, телефону, діапазону дат, позитивності цін). Помилки повертаються у структурованому JSON-форматі з унікальними кодами (01.x — готельний блок, 02.x — ресторанний).

Тестування проводилося за допомогою автоматичної документації Swagger та спеціалізованого інструмента для тестування API Postman. У ході тестування методом чорного ящика за допомогою Swagger протестовано всі можливі сценарії, у тому числі й помилкові ендпоінти, що працюють з моделлю Room, а за допомогою Postman після написання автоматизованих тестів для всіх ендпоінтів було перевірено коректність їх роботи.

ВИСНОВКИ

В ході дослідження було встановлено що готельно ресторанний комплекс є складною системою, в якій готельний і ресторанний блоки повинні функціонувати як операційно відмінні, проте інформаційно взаємозалежні системи. Було визначено ключові бізнес процеси, підлягаючих автоматизації для обох блоків та ті, що потребують їх інформаційної кооперації для управлінської звітності.

У роботі спроектовано та реалізовано модуль ІУС готельно-ресторанного комплексу як REST API на базі Django та Django REST Framework [6; 7]. Порівняльний аналіз [5] підтвердив доцільність вибору REST для CRUD-орієнтованих систем із чітко визначеними сутностями предметної області.

Аналіз ринкових рішень [1–4] показав, що жодне з них не забезпечує одночасно управління готельними і ресторанними блоками в єдиній системі з консолідованою звітністю. Розроблений модуль усуває цей недолік: API-ендпоінти покривають повний спектр операцій обох блоків, алгоритм пошуку вільного номера та резервування столика виконуються двома SQL-запитами незалежно від розміру БД, консолідований дашборд агрегує показники обох підмодулів у реальному часі. Модульна архітектура забезпечує розширюваність без змін в існуючому коді, а відкрита реалізація, на відміну від закритих SaaS-платформ [3; 4] дозволяє повністю адаптувати систему під специфіку конкретного підприємства.

ДЖЕРЕЛА

1. Oracle Corporation. Oracle Hospitality. Hotel Property Management System. Режим доступу: https://www.oracle.com/hospitality/(дата звернення: 26.04.2026).

2. Cloudbeds, Inc. Cloudbeds Hospitality Management System. Режим доступу: https://www.cloudbeds.com/(дата звернення: 24.04.2026).

3. Lightspeed Commerce Inc. Lightspeed Restaurant POS System. Режим доступу: https://www.lightspeedhq.com/pos/restaurant/(дата звернення: 26.04.2026).

4. Toast, Inc. Toast POS — Restaurant Point of Sale System. Режим доступу: https://pos.toasttab.com/ (дата звернення: 26.04.2026).

5. Порівняльний аналіз ефективності архітектурних стилів REST, GraphQL та gRPC для масштабованих мікросервісних систем. Вісник Хмельницького національного університету. Серія: Технічні науки. – 2025. – Том 355, № 4.Режим доступу: https://doi.org/10.31891/2307-5732-2025-355-79 (дата звернення: 26.04.2026)

6. Django Software Foundation. Django documentation. Version 4.2 LTS. – 2023. – Режим доступу: https://docs.djangoproject.com/en/4.2/ (дата звернення: 26.04.2026).

7. Christie T. Django REST Framework Documentation [Електронний ресурс]. – 2024. – Режим доступу: https://www.django-rest-framework.org/ (дата звернення: 26.04.2026).

8. Abramov, V., Astafieva, M., Boiko, M., Bodnenko, D., Bushma, A., Vember, V., Hlushak, O., Zhyltsov, O., Ilich, L., Kobets, N., Kovaliuk, T., Kuchakovska, H., Lytvyn, O., Lytvyn, P., Mashkina, I., Morze, N., Nosenko, T., Proshkin, V., Radchenko, S., & Yaskevych, V. (2021). Theoretical and practical aspects of the use of mathematical methods and information technology in education and science. https://doi.org/10.28925/9720213284km

9. Носенко, Т. І., Машкіна, І. В., & Яскевич, В. О. (2025). Застосування алгоритмів та структур даних у штучному інтелекті: навчальний посібник . Київ : Київський столичний університет імені Бориса Грінченка, 2025. – 201 С

10. Хлиста, І., & Бондарчук, А. (2023). Вирішення проблеми часткового оновлення вмісту веб-сторінки шляхом оптимізації параметрів SPA. Комп’ютерне моделювання та інформаційні технології, 286-291.

11. Печериця, В. В., Бондарчук, А. П., & Замрій, І. В. (2021). Розробка методики прототипування об'єктів інформаційної системи на базі технології Java Script, Node. JS. Телекомунікаційні та інформаційні технології, (4), 12-19.

12. Вембер, В. П., Машкіна, І. В., Носенко, Т. І., & Яскевич, В. О. (2025). Можливості та виклики використання штучного інтелекту у навчанні фахових дисциплін студентів спеціальностей «Комп’ютерні науки» та «Інженерія програмного забезпечення». Open educational e-environment of modern University, (19), 1-16.