# COMPARATIVE ANALYSIS OF OBJECT-RELATIONAL MAPPING (ORM) AND SQL QUERY IN THE IMPLEMENTATION OF BOOKING SERVICE *API* AT PT TUNAS DWIPA MATRA

**(Undergraduate Thesis)**

**By**

**NADAA AZHAR**
**NPM 2017051057**

**FACULTY OF MATHEMATICS AND NATURAL SCIENCES**
**LAMPUNG UNIVERSITY**
**BANDAR LAMPUNG**
**2025**

# COMPARATIVE ANALYSIS OF OBJECT-RELATIONAL MAPPING (ORM) AND SQL QUERY IN THE IMPLEMENTATION OF BOOKING SERVICE *API* AT PT TUNAS DWIPA MATRA

**By**

**NADAA AZHAR**
**NPM 2017051057**

**As**
**One of the requirements to attain a**
**Bachelor's degree in Computer Science**

**Within**

**The Computer Science Department,**
**on the Computer Science Bachelor's Program**


**FACULTY OF MATHEMATICS AND NATURAL SCIENCES**
**LAMPUNG UNIVERSITY**
**BANDAR LAMPUNG**
**2025**

## ABSTRACT

## COMPARATIVE ANALYSIS OF OBJECT-RELATIONAL MAPPING (ORM) AND SQL QUERY IN THE IMPLEMENTATION OF BOOKING SERVICE *API* AT PT TUNAS DWIPA MATRA

**By**

**Nadaa Azhar**

Information technology plays a vital role across all industries. PT Tunas Dwipa Matra recognizes this importance, understanding that an effective IT system is crucial for business operations and customer service. This study examines the performance of ORM and SQL queries in database operations, focusing on execution speed and memory efficiency across datasets of 10.000, 50.000, and 100.000 records. From this research, ORM showed significantly faster execution times, while maintaining comparable memory usage for both methods. Although, as data volume increased to 100.000 records, SQL's execution time grew substantially about 1,952s on average compared to ORM's more stable performance about 0,037s on average, with slightly 0,2 MiB more memory requirements. The study confirms ORM offers developer-friendly implementation and consistent memory efficiency, while SQL provides better scalability for complex queries. These findings suggest PT Tunas Dwipa Matra could optimize their system by using ORM for routine operations and SQL for large-scale data processing, achieving both performance efficiency and maintainability in their customer service applications.



**Keywords**: Object Relational Mapping (ORM); SQL Query; Performance Testing; Execution Speed; Memory Efficiency.

| | | |
|---|---|---|
| Thesis Title | : | **COMPARATIVE ANALYSIS OF OBJECT-RELATIONAL MAPPING (ORM) AND SQL QUERY IN THE IMPLEMENTATION OF BOOKING SERVICE API AT PT TUNAS DWIPA MATRA** |
| Student Name | : | Nadaa Azhar |
| Student Number | : | **2017051057** |
| Study Program | : | **Computer Science** |
| Department | : | **Computer Science** |
| Faculty | : | **Mathematics and Natural Sciences** |

**APPROVED BY**

1. Advisory Committee

| Advisor | Co-Advisor |
|---|---|
| **Bambang Hermanto, S.Kom., M.Cs** | **M. Iqbal Parabi, S.SI., M.T** |
| NIP. 19790912 200812 1 002 | NIP. 19901130 201504 1 002 |

2. Chairperson of Computer Science Department

**Dwi Saketki, S.Si., M.Kom**
NIP. 19680611 199802 1 001

**ADMITTED BY**

1. Examination Committee

| | | | |
|---|---|---|---|
| Chairperson | : | **Bambang Hermanto, S.Kom., M.Cs** | ..... |
| Secretary | : | **M. Iqbal Parabi, S.SI., M.T** | ..... |
| Examiner | : | **Dwi Sakethi, S.Si., M.Kom** | ..... |

2. Dean of Mathematics and Natural Sciences Faculty

**Dr. Eng. Heri Satria, S.Si., M.Si.**
NIP. 19711001 200501 1 002

Graduated on: **March 4th 2025**

**STATEMENT**

I, the undersigned below:

Student Name : Nadaa Azhar

Student Number : 2017051057

Hereby declare that my thesis entitled "Comparative Analysis of Object-Relational Mapping (ORM) and SQL Query in The Implementation of Booking Service API at PT Tunas Dwipa Matra" is my own work and not the work of others. All writings contained in this thesis have adhered to the academic writing regulations of Universitas Lampung. Should it be proven in the future that my thesis is the result of plagiarism or created by someone else, I am prepared to face consequences, including the annulment of the degree I have received.

Bandar Lampung, March 4th 2025


Nadaa Azhar
2017051057

## BIOGRAPHY

The researcher was born in Krui on October 14, 2002, as the third of five children to the late Mr. Arhas and Mrs. Identina Wati. Their educational journey began at SD Negeri 1 Krui, where they completed elementary education in 2014. They then progressed to SMP Negeri 2 Pesisir Tengah, graduating from junior high school in 2017. Continuing their studies at SMA Negeri 1 Pesisir Tengah, they successfully finished senior high school in 2020. That same year, they matriculated into the Computer Science program at the Faculty of Mathematics and Natural Sciences, University of Lampung, having secured admission through the SBMPTN national selection process. Throughout their university years, the individual actively participated in various academic and extracurricular pursuits, such as.

1. Member of Secretariat Bureau of Himakom (Computer Science Student Association) for the 2021/2022 term.
2. Secretary of Human Resource Development of KSE (Karya Salemba Empat) for the 2021/2022 term.
3. Secretary of Secretariat Bureau of Himakom (Computer Science Student Association) for the 2022/2023 term.
4. Secretary of Education, Research and Technology of KSE (Karya Salemba Empat) for the 2022/2023 term.
5. Lecturer assistant for Logic and System Operation courses in the Computer Science Department.
6. Awardee of Karya Salemba Empat Scholarship in 2022 and 2023.
7. Kampus Merdeka internship program at PT Tunas Dwipa Matra from March to August 2023.

8. Translator in Field Trip of Agri Lestari Nusantara in 2023.
9. Volunteer of Busa Pustaka in 2024.

## MOTTO

“Mohonlah pertolongan (kepada Allah) dengan sabar dan shalat. Sungguh, Allah beserta orang-orang yang sabar.”

(Q.S. Al Baqarah [2:153])

**DEDICATION**

*With heartfelt gratitude to Allah Subhanahu Wa Ta'ala, the Most Gracious and Merciful, and blessings upon our beloved Prophet Muhammad Shallallahu 'Alaihi Wasallam, I dedicate this work to those who matter most in my journey.*

***My dearest parents***
***and***
***My cherished family***

*To those with unwavering faith, endless patience, unconditional love, constant prayers, and their support have guided my every step toward this accomplishment.*

***My lovely friends***
***and***
***Department of Computer Science***
***University of Lampung***

*I am sincerely grateful for their presence and the impact they've had on both my education and personal development.*

## ACKNOWLEDGEMENT

With profound praise and gratitude to Allah Subhanahu Wa Ta'ala for His endless blessings, I humbly present this thesis entitled "Comparative Analysis of Object-Relational Mapping (ORM) and SQL Query in The Implementation of Booking Service API at PT Tunas Dwipa Matra". This undergraduate thesis fulfills the requirements for obtaining an undergraduate degree in Computer Science at the Department of Computer Science, Faculty of Mathematics and Natural Sciences, University of Lampung.

The journey of completing this thesis presented numerous challenges, both in research and writing. Although, through assistance and the unwavering support of many individuals, I was able to overcome these obstacles. Therefore, I extend my deepest gratitude to:

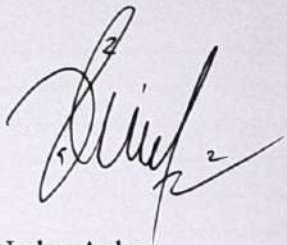
1. My heartfelt gratitude goes to my parents, Alm. Ayah and Ibu, as well as my siblings, Odang, Ingah, Amir, and Aqil. Their constant encouragement, whether through material support, helping hands, or heartfelt prayers, has been my source of strength throughout this journey.
2. My beloved nephews, Arshaka and Zio. Their radiant smiles and innocent wonder have filled my heart with boundless happiness and warmth.
3. Mr. Dr. Eng. Heri Satria, S.Si., M.Si., as the Dean of the Faculty of Mathematics and Natural Sciences.
4. Mr. Dwi Sakethi, S.Si., M.Kom., as the examiner and the Head of the Computer Science Department for his thorough review, suggestions and insightful feedback on this thesis.

5. Mr. Dr. rer. nat. Akmal Junaidi, M.Sc., for his guidance and support as my academic advisor.
6. Mr. Bambang Hermanto, S.Kom., M.Cs., for his role as the main supervisor and lecturer, for providing invaluable opportunities, advice, criticism, suggestions, assistance, and support throughout the thesis and study.
7. Mr. M. Iqbal Parabi, S.SI., M.T., as the second supervisor, for his critiques, constructive feedback, and thoughtful suggestions were instrumental in strengthening my thesis.
8. To all the lecturers for their profound expertise, unwavering dedication, and genuine commitment to teaching have been instrumental in shaping my intellectual growth and professional development.
9. To all the Computer Science Staff (Mrs. Nora, Mr. Nofal, Mr. Sam, and Mr. Jay). Their dedication and support have greatly helped me finish this thesis.
10. To my thesis partner, M. Hanif Pratama. His support, guidance, suggestions have been truly invaluable, making the process both productive and enjoyable.
11. To my best friends, Renna and Nisa, for their encouragement and support that have accompanied me on this journey and will continue to do so in the future.
12. To the Ihiiy members—Kayla, Karina, Dita, Dhavy, Mufid, Irfan, Rifqi, Naufal, and Ega—for their support and camaraderie
13. To my high school friends, Gita, Riska, Shiva, and Nesi for always being there with me.
14. To my junior school friends, Salma, Halimah, Angel, Ulik with their presence will always comfort me.
15. To my lovely cats, Haru, Kira, Mao, Mila, Tobi and Boni. Their presence brings so much joy and comfort into my life.

Lastly, I extend my gratitude to all my friends in the field of Computer Science.

The researcher acknowledges that this thesis may contain imperfections. However, we sincerely hope it will contribute meaningfully to academic discourse and practical applications. This study aspires to benefit not only the researcher but also researchers, students, and professionals in the field.

Bandar Lampung, March 4th, 2025



Nadaa Azhar
2017051057

# TABLE OF CONTENTS

# TABLE OF TABLES

# TABLE OF FIGURES

# I. INTRODUCTION

## 1.1. Background

In this digital era, information technology has become a key element in various aspects of life, including the service and reservation industry (Pattinama et al., 2023). Companies in this sector, including PT Tunas Dwipa Matra, have faced pressure to adopt the latest technology to meet the demands of an increasingly competitive market and ensure better customer experience. PT Tunas Dwipa Matra is committed to maintaining its competitiveness in the market by adopting the latest technological innovations.

One crucial aspect of digital transformation is the use of Application Programming Interface (API), which allows various applications and systems to communicate and interact with each other efficiently (Filiana et al., 2022). A study implementing API in the Odoo ERP system, specifically in the Customer Relationship Management (CRM) module with Couchbase as an offline storage solution, aims to manage and store customer data efficiently (Permatasari & Ariyani, 2019). From this research, it can be observed that the implementation of API is key to providing a fast, reliable, and user-friendly Booking Service for PT Tunas Dwipa Matra customers.

In the development of a database-based application, such as the API in this Booking Service, there is a debate regarding the technical approach to managing the database. One of the main debates is between the use of Object Relational Mapping (ORM) and SQL queries. ORM is a programming

technique that allows developers to interact with a relational database using an object-oriented programming language. ORM provides mapping between objects in the application code and tables in the database, enabling developers to perform database operations using object-oriented syntax and concepts (Gorodnichev et al., 2020). On the other hand, SQL queries are statements written in Structured Query Language (SQL) used to retrieve data from a relational database, allowing users to specify the desired data to be retrieved based on certain conditions and criteria (Alshemaimri et al., 2021).

The choice between using ORM or SQL queries can have a significant impact on application performance, scalability, and development complexity (Colley et al., 2018). Therefore, in the context of implementing the Booking Service API at PT Tunas Dwipa Matra, it is crucial to conduct an in-depth analysis of the differences, advantages, and disadvantages of each approach.

This research aims to provide a comprehensive understanding of how the use of ORM and SQL queries affects the implementation of the Booking Service API at PT Tunas Dwipa Matra. With a deep understanding of the comparison between these two approaches, PT Tunas Dwipa Matra can make informed decisions in designing an optimal system, ultimately improving service quality and operational efficiency.

Through this research, the author will analyze the performance and development complexity of each approach. The results of this research will serve as a strong foundation for PT Tunas Dwipa Matra in making the right decisions regarding the technical approach to implementing their Booking Service API.

## 1.2 Problem Statements

Based on the background description, the research problem formulation for this study is as follows:

1. The use of Object-Relational Mapping (ORM) significantly affects the implementation of the Booking Service API at PT Tunas Dwipa Matra.
2. The use of SQL queries has a notable impact on the implementation of the Booking Service API at PT Tunas Dwipa Matra.
3. Choosing a technical approach, whether ORM or SQL queries, significantly impacts the performance and operational efficiency of the Booking Service at PT Tunas Dwipa Matra.

## 1.3 Problem Constraints

The constraints of this study are as follows:

1. The analysis is limited to comparing the use of Object Relational Mapping (ORM) and SQL queries in the implementation of the Booking Service API at PT Tunas Dwipa Matra.
2. The evaluation is focused on the existing system at PT Tunas Dwipa Matra for the Booking Service, focusing in 'dms.uang.titipan.customer' table, 'dms.account.invoice' table, 'dms.booking.service' table.
3. The research will evaluate the speed performance and memory usage of the Booking Service API.
4. The research will be conducted on Mokita module.
5. The performance testing will be conducted in Python 2.7.

## 1.4 Research Objectives

The objective of this research is to analyze the comparison between the utilization of Object Relational Mapping (ORM) and SQL queries in the implementation of the Booking Service API at PT Tunas Dwipa Matra. The primary focus of this study is to comprehend the impact of employing ORM and SQL queries on the performance of the Booking Service system, identify the strengths and weaknesses of each approach in terms of speed performance, and memory usage of the Booking Service API.

## 1.5 Research Advantages

The benefits derived from this research are as follows:

1. For the author, this study provides a profound understanding of the comparison between ORM and SQL queries, research experience contributing to practical problem-solving, and the development of data analysis skills and scientific communication.
2. For the company, this research can assist in selecting the appropriate technology for the implementation of their Booking Service API, optimizing operational efficiency, and enhancing the company's competitiveness in the competitive service and reservation industry.
3. For practitioners, this research can offer practical guidance for professionals involved in the development of database-based applications.

# II. LITERATURE REVIEW

## 2.1. Previous Research

Several previous studies used as references in this research are as follows:

### 2.1.1. Object Relational Mapping Framework Performance Impact

This research was conducted by Dr. V. Sivakumar, T. Balachander, Logu, Ramu Jannali in the Turkish Journal of Computer and Mathematics Education Vol. 12 No. 7 (2021). This research article delves into the impact of various object-relational mapping frameworks on the performance of relational queries. By comparing 8 different frameworks across 4 programming languages, the study sheds light on the significant increase in query execution time when utilizing an object-relational mapping framework (Sivakumar et al., 2021).

The findings underscore the importance of considering database performance when making decisions about incorporating such frameworks, despite the potential lack of noticeable impact in most applications. The article's exploration of the limitations in using only select queries and a single database also highlights the need for further research to delve into additional factors that could influence the performance of object-relational mapping frameworks (Sivakumar et al., 2021).

### 2.1.2. The Impact of Object-Relational Mapping Frameworks on Relational Query Performance

This research was conducted by Derek Colley, Clare Stanier, Md Asaduzzaman in the International Conference on Computing, Electronics & Communications Engineering (iCCECE) (2018). The research explores the impact of object-relational mapping (ORM) frameworks on relational database query performance. It discusses the negative performance consequences of ORM tools, providing examples of suboptimal query performance patterns resulting from their use (Colley et al., 2018).

Additionally, the research suggests potential solutions to mitigate the performance impacts of ORMs and highlights the need for a database-centric approach to query performance tuning within the context of ORM frameworks. Furthermore, the article discusses the incorporation of a multi-schema model into relational database development as an alternative approach, emphasizing the limitations of NoSQL data stores and proposing the further development and augmentation of the relational model to address new challenges (Colley et al., 2018).

### 2.1.3. Exploring Object-Relational Mapping (ORM) Systems and How to Effectively Program a Data Access Model

This research was conducted by Mikhail Gorodnichev, Marina Moseva, Ksenia Poly, Khizar Dzhabrailov, Rinat in the Journal of Archaeology of Egypt/Egyptology Vol. 17(3) (2020). The research explores the concept of Object-Relational Mapping (ORM) and its application in software development. It delves into the challenges posed by the semantic gap between object-oriented programming and relational databases, and how ORM serves as a solution by providing an abstraction layer that enables developers to work with objects instead of tables. The study focuses on the Entity Framework

(EF) as an example of an ORM system and compares its functionality to manually written SQL queries. It discusses the advantages and drawbacks of ORM, including its impact on application performance, and presents findings from tests comparing EF and "pure" SQL operations (Gorodnichev et al., 2020).

The research also addresses the issue of performance optimization when using ORM libraries, demonstrating that ORM systems can be effectively utilized without significantly compromising application performance. Overall, the research concludes that ORM offers a convenient approach to working with relational databases, simplifying application development and maintenance.

### 2.1.4. Data-Oriented Differential Testing of Object-Relational Mapping Systems

This research was conducted by Thodoris Sotiropoulos, Stefanos Chaliasos, Vaggelis Atlidakis, in the International Conference on Software Engineering (ICSE) (2021). The research focuses on proposing and implementing a data-oriented testing approach, using a tool called CYNTHIA, for Object-Relational Mapping (ORM) systems. The authors address the challenges associated with differential testing of ORM systems, such as the lack of a common specification and input language, non-deterministic query results, DBMS-dependent results, and data generation. They also discuss the effectiveness of the solver-based approach for data generation in identifying mismatches between ORM outputs. The authors emphasize the importance of the quality of inserted data in testing ORM systems and highlight the potential of their differential testing approach in improving the reliability and robustness of ORM systems (Sotiropoulos et al., 2021).

### 2.1.5. Comparison of Eloquent ORM with Query Builder in Work Management System (Case Study: Muhammadiyah Lamongan Hospital)

This research was conducted by Febryan Akhdani and Danur Wijayanto, in the SENATIK Vol. 7 (2021). The research paper focuses on comparing the performance of Eloquent ORM and Query Builder within the context of a Work Management System at Muhammadiyah Lamongan Hospital. The study delves into various aspects such as the execution time of queries, page access, memory usage, and code readability to determine which method is more suitable for the system. The methodology employed in the research includes interviews, literature review, and testing, providing a comprehensive understanding of the comparison (Akhdani & Wijayanto, 2022).

## 2.2. Theoretical Framework Description

### 2.2.1. Database

A database is defined as a meticulously organized assembly of data that is electronically stored and retrieved. Within the realm of server-side web applications, it assumes a pivotal role, serving as an indispensable component for the storage and systematic processing of data generated or consumed by end users (Shao et al., 2020). In this research, the database system used is PostgreSQL.

PostgreSQL, often referred to as Postgres, is an advanced open-source Object-Relational Database Management System (ORDBMS). It places a strong emphasis on extensibility, creativity, and compatibility. As an open-source database system, PostgreSQL provides a robust and flexible platform for managing relational databases, allowing users to customize and extend its functionality according to their specific needs. The system is designed to support

a wide range of applications and promotes a collaborative and innovative approach to database management (Waruwu, 2019).

### 2.2.2. Object Relational Mapping (ORM)

An object-relational mapping (ORM) framework is a solution for the problem that arises when trying to map objects in programming languages to database tables. These frameworks provide a mapping from the object layer (object classes) to the relational database tables, allowing for an interface to generate SQL queries without the need for the application to contain any SQL code or have SQL capabilities (Sivakumar et al., 2021).

ORM tools significantly streamline the process of saving and retrieving objects within relational databases, providing an abstraction layer that allows developers to interact with database content primarily through object-oriented programming (OOP) entities instead of directly working with the database management system (DBMS). This approach abstracts the technical details of the DBMS, creating an interface that maps object-oriented structures, like classes and attributes, directly to relational tables and fields. By doing so, ORM reduces the complexity associated with database interactions, relieving developers from the need to write extensive SQL code for basic operations, such as inserts, updates, deletes, or queries. Instead, developers can handle data operations using familiar OOP principles, focusing more on application logic rather than the intricacies of SQL syntax or DBMS-specific features (Gorodnichev et al., 2020).

Alternatively, in brief, Object-Relational Mapping (ORM) defined as software solutions designed to address the object-relational impedance mismatch problem (Colley et al., 2018).

### 2.2.3. Structured Query Language (SQL)

Structured Query Language (SQL) refers to the standardized and widespread language used for working with SQL databases. It is utilized for executing complex queries, including a high number of join operations, and provides users with a high level of efficiency for storage and management of structured data. SQL is also chosen as the uniform language in the newly developed architecture for integrating different types of databases, and it is used for the definition of mappers to translate SQL queries into concrete languages used by specific databases that do not support the SQL standard (Bjeladinovic et al., 2020).

Structured Query Language (SQL) is a widely standardized language integral to managing SQL databases, particularly known for its ability to handle complex queries and multiple join operations that facilitate effective data organization and aggregation. SQL enables users to query data from multiple tables simultaneously, an essential function in relational databases where data is typically normalized to minimize redundancy. SQL is also a suitable language for integrating diverse database systems, as it can act as a common framework that translates SQL queries into the specific syntax of non-SQL databases, enhancing interoperability and operational efficiency across hybrid architectures (Ishaq Jound, 2020).

### 2.2.4. Application Programming Interface (API)

API stands for Application Programming Interface, is a set of rules and protocols that allows different software applications to communicate with each other. APIs define the methods and data formats that applications can use to request and exchange information (Pattinama et al., 2023).

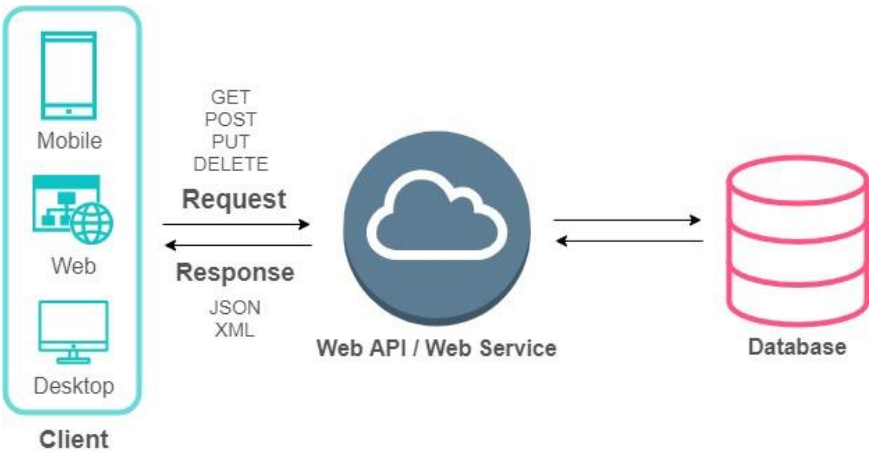



Figure 1. API Illustration

APIs have numerous applications in modern technology, facilitating the connection of various services and applications Author use in our daily lives. For instance, social media applications use APIs for content sharing, while payment APIs enable online transactions, among many other uses.

### 2.2.5. Python

Python stands as one of the most widely embraced programming languages globally, boasting a history spanning over two decades. Its prevalence is particularly pronounced in academic settings and enjoys extensive support as a platform for contemporary applications, notably in utilities, desktop applications, and web applications. Recognized as an interpreted language, Python prioritizes readability through its syntax while maintaining a compact size. Renowned for its extensive array of libraries, Python enables achieving more with concise code. The language's clean syntax proves apt for managing databases or utilizing tables (Patil, 2020).

### 2.2.6. Framework

A framework is an extensive and integrated software development platform meticulously designed to provide a well-organized and

unified environment. It comes equipped with a diverse array of tools, empowering developers to efficiently build, organize, and oversee the development of web applications. The framework serves as a foundational structure that streamlines the development process, facilitating the creation of robust and scalable web-based solutions (Endra et al., 2021).

### 2.2.7. Odoo

Odoo stands as an open-source Enterprise Resource Planning (ERP) software, presenting a multifaceted platform comprised of three fundamental components: a PostgreSQL database serving as its backend, an application server, and a web server. This comprehensive system encompasses basic modules designed to support various business functions, providing a foundation for extensive customization to cater to specific organizational needs. The PostgreSQL database serves as a robust and reliable backend, while the application server and web server collectively ensure seamless and efficient operations across the entire ERP ecosystem. The flexibility inherent in Odoo allows businesses to tailor their ERP solutions to match their unique requirements and operational workflows (Nurkhafidoh & Ariyani, 2019).

### 2.2.8. Enterprise Resource Planning (ERP)

Enterprise Resource Planning (ERP) systems represent sophisticated software solutions designed to integrate and synchronize information across an organization's diverse departments. Encompassing a wide array of modules, these systems efficiently manage key facets of organizational functions, including manufacturing, human resources, finance, and supply chain management. By automating essential corporate activities, ERP systems contribute to streamlining processes, expediting decision-

making processes, realizing cost reductions, and enhancing overall managerial control (Sambe et al., 2019).

# III. RESEARCH METHODOLOGY

## 3.1. Research Time and Place

The research will be conducted at PT. Tunas Dwipa Matra, located at Jl. Pramuka Number 01, Rajabasa District, Bandar Lampung City, Lampung Province 35144. This research will be conducted in the odd semester of the Academic Year 2023/2024.

## 3.2. Research Tools

The hardware and software specifications used in this research are as follows:

### 3.2.1. Hardware Tools

The hardware used in this research is a laptop with the following specifications:

a) System Manufacturer : MSI
b) System Model : MSI Modern 14 C11M
c) Processor : Intel Core i5-1155G7
d) RAM : 8.00 GB
e) System Type : 64 bits
f) Storage : SSD 512 GB

### 3.2.2. Software Tools

The software used in this research is as follows:

a) Operating System : Windows 11 Home
b) Integrated Development Environment : Visual Studio Code
c) Database Management System : PostgreSQL version 11

d) Framework : Odoo version 10
e) Programming Language : Python version 2.7
f) Version Control System : Git
g) Web Browser : Google Chrome

### 3.3. Research Stages

The research stages in this study are illustrated in Figure 1. The diagram represents the flowchart of the research process, which is divided into four main parts: problem identification, literature review, data collection, and performance testing.

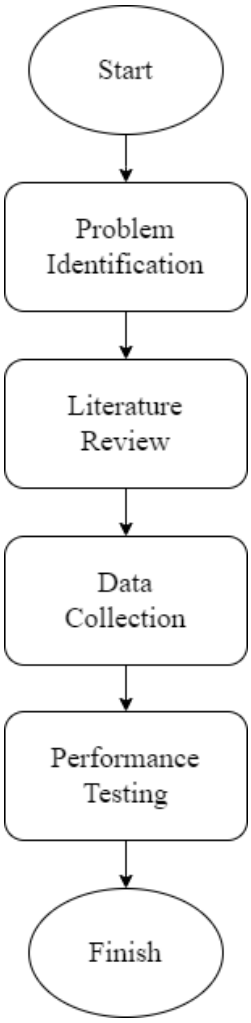



Figure 2. Research Flowchart

### 3.3.1. Problem Identification

The research addresses challenges in the booking service module of PT Tunas Dwipa Matra, where the existing code for database processing reveals inefficiencies. The core problem lies in deciding between Object-Relational Mapping (ORM) and SQL Query frameworks, aiming to identify the more effective framework for executing the program within the booking service module. This involves assessing performance bottlenecks, considering development speed and maintenance, evaluating scalability, examining user experience implications, ensuring seamless integration, and verifying alignment with business objectives. The research seeks to pinpoint specific issues affecting the current system's effectiveness and determine the optimal framework choice that aligns with both technical and business requirements.

### 3.3.2. Literature Review

A literature review involves gathering information from diverse sources like journals, books, and relevant research. The main goal is to cultivate a profound understanding of the research topic. This comprehensive review aids in formulating essential components that align with the forthcoming study, guiding the exploration of the chosen subject. By delving into the existing literature, this research can strategically determine its direction, identifying unexplored areas that warrant closer examination. The literature review serves as a valuable tool to gain insights into aspects that may not have been previously investigated, ensuring a thorough and informed approach to the research.

### 3.3.3. Data Collection

Data collection in this research is obtained from two sources: literature review and interview.

a) Literature Review

Literature review is the first step in the data collection process. In this stage, various information and data related to API, booking service, and Odoo framework will be gathered, including writings, photos, and electronic documents.

b) Observation

The next step in data collection for this research is observation. Observation is conducted to gain a more precise understanding of the current conditions of the booking service. This involves studying the database structure in the system to be integrated.

### 3.3.4. Performance Testing

The comparison used in this research is obtained by comparing the execution time between Object-Relational Mapping (ORM) and SQL Query. Since ORM and SQL Query differ in terms of code and data access programming techniques, a comparison is conducted with speed performance and memory usage as the parameters.

a) Data Dictionary

A data dictionary is used to specify and view the tables that will be utilized. In this booking service module, 18 tables from the internal company system are employed. All the specified tables will be used in creating data queries that will be sent to the API applications, namely Motoran, and vice versa. The tables used include:

1) The table "dms_booking_service" is used to store data related to booking service.
2) The table "dms_booking_service_line" is used to store data related to spare parts and service in the booking service menu.
3) The table "dms_service_advisor" is used to store data related to service advisors.

4) The table "dms_service_advisor_line" is used to store data related to spare parts and service in the service advisor menu.
5) The table "dms_service_order" is used to store data related to service orders.
6) The table "dms_service_order_line" is used to store data related to spare parts and service in the service order menu.
7) The table "res_partner" is used to store data related to customers.
8) The table "res_branch" is used to store data related to dealers.
9) The table "hr_employee" is used to store data related to employees.
10) The table "product_product" is used to store data related to service and spare part codes.
11) The table "product_template" is used to store data related to the price, name, and description of each spare part and service.
12) The table "dms_api_configuration" is used to store data related to API configuration.
13) The table "dms_api_log" is used to store data related to logs from received and sent APIs.
14) The table "stock_production_lot" is used to store data related to vehicles, such as chassis numbers, engines, and others.
15) The table "dms_account_invoice" is used to store data related to overall customer invoices.
16) The table "dms_account_payment" is used to store data related to notes or payments from customers.
17) The table "dms_account_payment_line" is used to store data related to lists of invoice notes or customer payments.

18) The table "dms_uang_titipan_customer" is used to store data related to customer down payment.

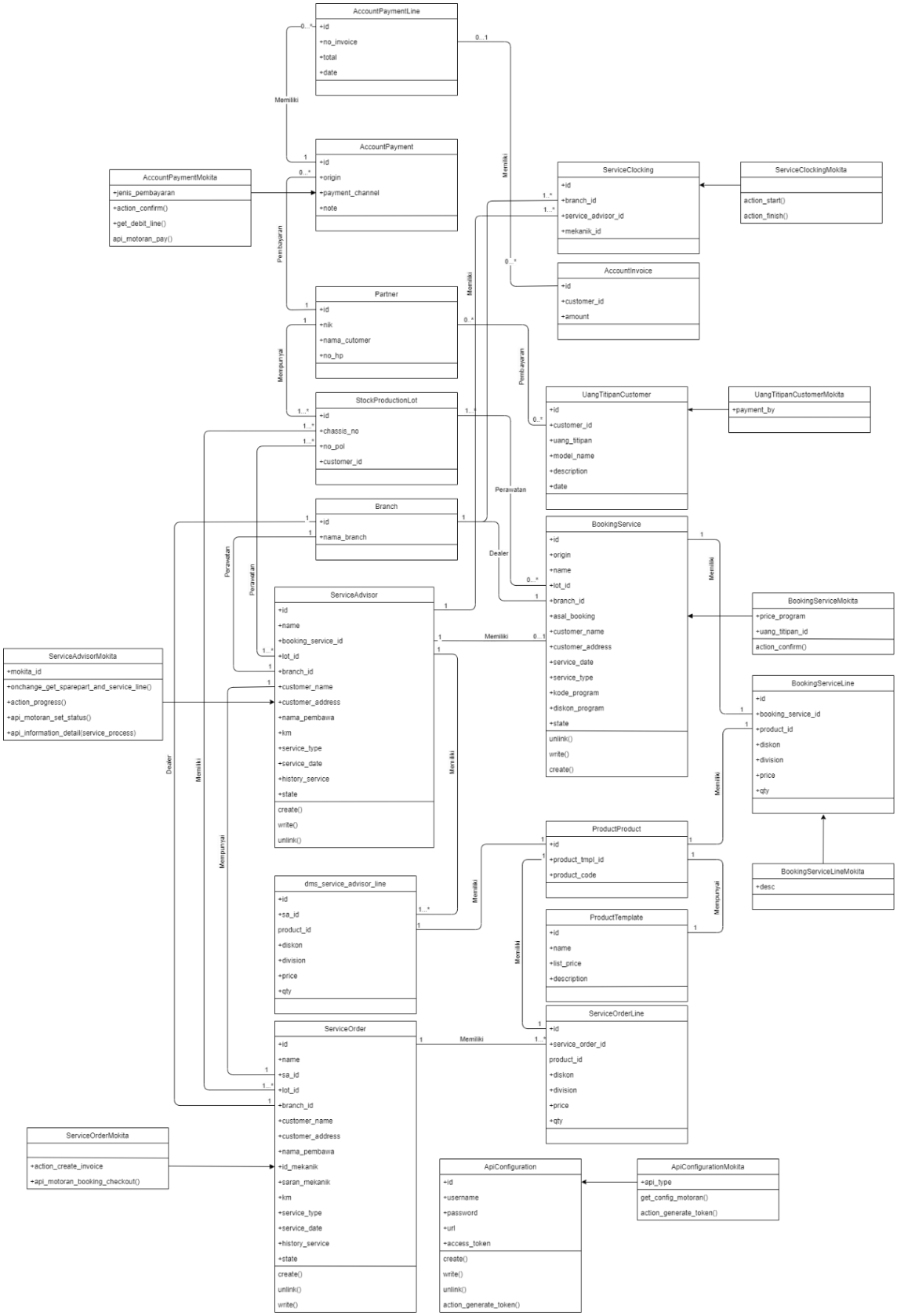



Figure 3. Class Diagram

b) Design of Performance Testing Comparison

At this stage, the testing design includes a comparison between ORM and SQL Query on a specific function within the related module. The required design elements are as follows:

1) Design Testing for Select Data *'dms.uang.titipan.customer'*

In this section, the part to be tested is the use of a query to create a new record in the 'dms.uang.titipan.customer' table.

Table 1. Select Data 'dms.uang.titipan.customer'

| **Syntax** | INSERT |
|---|---|
| **ORM** | uang_titipan_obj = request.env['dms.uang.titipan.customer'].suspend_security()<br>create_titipan = uang_titipan_obj.suspend_security().create({<br>'uang_titipan' : post.get('Amount',False),<br>'model_name' : 'dms.booking.service',<br>'customer_id' : customer_id.id,<br>'branch_id' : branch_id,<br>'division' : 'Sparepart',<br>'description': customer_desc,<br>'date' : datetime.now(),<br>'booking_service_name' : str(post.get('DmsBookingNumber','')),<br>'payment_by':'motoran'<br>}) |
| **Query SQL** | query = """<br>INSERT INTO dms_uang_titipan_customer (uang_titipan, model_name, customer_id, branch_id, division, description, date, booking_service_name, payment_by)<br>VALUES ({uang_titipan}, 'dms.booking.service', {customer_id}, {branch_id}, 'Sparepart', '{description}', '{date}', '{booking_service_name}', 'motoran')<br>RETURNING id<br>"""<br><br>query = query.format(uang_titipan = post.get('Amount',**FALSE**), customer_id = customer_id.id, branch_id = branch_id, description = customer_desc, **DATE** = datetime.**NOW**(), booking_service_name = str(post.get('DmsBookingNumber','')))<br><br>request._cr.execute(query) |

2) Design Testing for Update Data *'dms.account.invoice'*

In this section, the part that will be tested is the use of a query to retrieve data that meets several specified criteria on the `invoice_titipan_obj` object from the 'dms.account.invoice' model.

Table 2. Update Data 'dms.account.invoice'

| **Syntax** | SELECT |
|---|---|
| **ORM** | `invoice_titipan_obj = request.env['dms.account.invoice'].suspend_security().search(`<br>`  [('partner_id', '=', service_order.stnk_id.id),`<br>`  ('branch_id', '=', service_order.branch_id.id),`<br>`  ('state', '=', 'open'),`<br>`  ('type', '=', 'out_invoice'),`<br>`  ('model_name', '=', 'dms.uang.titipan.customer')])` |
| **Query SQL** | `query = """`<br>`  SELECT * FROM dms_account_invoice`<br>`  WHERE partner_id = {} AND branch_id = {} AND state = 'open'`<br>`  AND type = 'out_invoice' AND model_name = 'dms.uang.titipan.customer'`<br>`""".format(service_order.stnk_id.id, service_order.branch_id.id)`<br><br>`request._cr.execute(query)`<br>`invoice_titipan_data = request._cr.fetchall()` |

3) Design Testing for Insert Data *'dms.booking.service'*

In this section, the part to be tested involves the use of a query to write (update) several columns in the `booking_service_obj` object from the 'dms.booking.service' model.

Table 3. Insert Data 'dms.booking.service'

| **Syntax** | WRITE |
|---|---|
| **ORM** | `booking_service_obj.write({`<br>`'date_service' : booking_date,`<br>`'jam_service' : str(booking_time[0:2]),`<br>`'menit_service' :`<br>`str(booking_time[3:5]),`<br>`'service_pit_id' : pit_id,`<br>`'state' : 'reschedule'`<br>`})` |
| **Query SQL** | `query = """`<br>`UPDATE dms_booking_service`<br>`SET date_service = '{booking_date}',`<br>`jam_service = '{jam_service}',`<br>`menit_service = '{menit_service}',`<br>`service_pit_id = {service_pit_id},`<br>`state = 'reschedule'`<br>`WHERE id = {booking_service_id}`<br>`""".format(`<br>`booking_date=booking_date,`<br>`jam_service=booking_time[0:2],`<br>`menit_service=booking_time[3:5],`<br>`service_pit_id=pit_id,`<br>`booking_service_id=booking_service_id`<br>`)`<br>`request._cr.execute(query)` |

c) System Testing Scenario

The query testing conducted in this study utilizes performance testing and is followed by a comparative analysis of the test results. The following are the scenarios for testing the use of ORM and SQL Query in the Booking Service API. Performance

testing will be conducted with a dataset of 10,000; 50,000; and 100,000 data entries under optimal internet network conditions.

Table 4. System Testing Scenario

| **List of Tests** | **Description** | **Expected Outcome** |
|---|---|---|
| SELECT Data | Retrieve data that meets several specified criteria on '*dms.titipan.customer*' model. | Successfully retrieve data that meets the specified criteria. |
| WRITE Data | Write (update) object from the '*dms.account.invoice*' model. | The object updates successfully. |
| INSERT Data | Create a new record in the '*dms.booking.service*' table. | Successfully create a new record. |

# V. CONCLUSION AND RECOMMENDATION

## 5.1. Conclusion

The research on using ORM and SQL queries for the Booking Service API at PT Tunas Dwipa Matra demonstrates that ORM offers significant advantages in simplifying and optimizing development and improving maintainability. ORM performs efficiently in standard operations like data retrieval and updates, reducing complexity in the coding process and making it ideal for routine tasks. Its built-in validation, security features, and ease of use make it well-suited for scenarios where development speed and long-term maintainability are priorities. However, SQL queries excel in complex operations or performance-critical tasks, providing greater control over execution and faster processing for specialized or large-scale operations.

In conclusion, ORM is a favourable choice for most tasks within the Booking Service API, particularly for routine operations where ease of development and security are key. While SQL remains advantageous for highly optimized or complex queries, the overall findings suggest that ORM strikes a balance between efficiency, resource usage, and maintainability. By leveraging the strengths of both ORM and SQL, the implementation can achieve optimal performance and scalability while ensuring a secure and maintainable system.

## 5.2. Recommendation

It is recommended that PT Tunas Dwipa Matra adopts a hybrid approach, combining both ORM and SQL queries based on the specific needs of each task. For routine operations such as data insertion, updates, and simpler queries, ORM should be preferred for its simplicity and ease of maintenance. However, for more complex or critical tasks, direct SQL queries should be employed to improve speed and optimize resource usage.

# REFERENCES


Akhdani, F., & Wijayanto, D. (2022). Comparison of Eloquent ORM with Query Builder in Work Management System (Case Study: Muhammadiyah Lamongan Hospital). *Conference SENATIK STT Adisutjipto Yogyakarta*, *7*. https://doi.org/10.28989/senatik.v7i0.449

Alshemaimri, B., Elmasri, R., Alsahfi, T., & Almotairi, M. (2021). A survey of problematic database code fragments in software systems. *Engineering Reports*, *3*(10), e12441. https://doi.org/10.1002/eng2.12441

Bjeladinovic, S., Marjanovic, Z., & Babarogic, S. (2020). A proposal of architecture for integration and uniform use of hybrid SQL/NoSQL database components. *Journal of Systems and Software*, *168*, 110633. https://doi.org/10.1016/j.jss.2020.110633

Colley, D., Stanier, C., & Asaduzzaman, M. (2018). The Impact of Object-Relational Mapping Frameworks on Relational Query Performance. *2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, 47–52. https://doi.org/10.1109/iCCECOME.2018.8659222

Endra, R. Y., Aprilinda, Y., Dharmawan, Y. Y., & Ramadhan, W. (2021). Analisis Perbandingan Bahasa Pemrograman PHP Laravel dengan PHP Native pada

Pengembangan Website. *EXPERT: Jurnal Manajemen Sistem Informasi dan Teknologi*, *11*(1), 48. https://doi.org/10.36448/expert.v11i1.2012

Filiana, A., Rini, M. N. A., Prabawati, A. G., & Samat, R. A. (2022). Pengembangan Rest API Untuk Informasi Pasar Tradisional di Kota Yogyakarta dengan Metode Incremental. *SINTECH (Science and Information Technology) Journal*, *5*(1), 10–23. https://doi.org/10.31598/sintechjournal.v5i1.1060

Gorodnichev, M., Moseva, M., Poly, K., & Dzhabrailov, K. (2020). *Exploring Object-Relational Mapping (ORM) Systems and How to Effectively Program A Data Access Model*.

Ishaq Jound, H. H. (2020). *Comparison of performance between Raw SQL and Eloquent ORM in Laravel*.

Nurkhafidoh, S., & Ariyani, N. F. (2019). *Rancang Bangun API untuk Odoo ERP pada Modul Sales*. *8*(2).

Patil, P. R. (2020). *Smart Forest: An IoT Based Forest Safety And Conservation System*. *9*(03).

Pattinama, Y. L., Susanti, I., Luhur, U. B., Raya, J. C., & Utara, P. (2023). *Implementasi Rest API Web Service Dengan Otentifikasi JSON Web Token Untuk Aplikasi Properti*.

Permatasari, R. D., & Ariyani, N. F. (2019). *Rancang Bangun API untuk Odoo ERP pada Modul CRM (Customer Relationship Management)*.

Sambe, T., Maag, S., & Cavalli, A. (2019). A Methodology for Enterprise Resource Planning Automation Testing Application to the Open Source ERP-ODOO: *Proceedings of the 14th International Conference on Software Technologies*, 407–415. https://doi.org/10.5220/0007923004070415

Shao, S., Qiu, Z., Yu, X., Yang, W., Jin, G., Xie, T., & Wu, X. (2020). Database-Access Performance Antipatterns in Database-Backed Web Applications. *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 58–69. https://doi.org/10.1109/ICSME46990.2020.00016

Sivakumar, D. V., Balachander, T., & Jannali, R. (2021). *Turkish Journal of Computer and Mathematics Education Vol.12 No. 7 (2021), 2516-2519 Research Article*.

Sotiropoulos, T., Chaliasos, S., Atlidakis, V., Mitropoulos, D., & Spinellis, D. (2021). Data-Oriented Differential Testing of Object-Relational Mapping Systems. *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, 1535–1547. https://doi.org/10.1109/ICSE43902.2021.00137

Waruwu, T. S. (2019). *Implementasi Postgresql Sebagai Sistem Manajemen Basis Data Pada Pendaftaran Mahasiswa Baru Berbasis Web*.