Woody International Publish Limited
An Multidisciplinary Academic Journal Publisher

Journal of Theory and Practice of Humanities Science, Volume x, Issue x, 2024
https://www.woodyinternational.com/

A Brief Analysis of the Application of MyBatis in Java Development

Chao Jiang, Zhengde Bao*, Chenxi Li

School of Computer Software, Jincheng College of Sichuan University, Chengdu 611731, Sichuan, China
**Author to whom correspondence should be addressed.*

Abstract: ***MyBatis is a high-performance JavaEE persistence framework, so in large-scale JavaEE development, developers usually choose it as the persistence framework for their development projects; This article introduces the basic application and working principle of MyBatis in the development of project persistence layer through the case of automatic reply robot. This case uses the MyBatis framework to achieve two basic functions: automatic reply to page messages and information maintenance of backend pages; Through the project case in this article, readers can understand that MyBatis, as a persistence framework, has many benefits: basic ORM implementation, unified management of SQL statements, and implementation of dynamic SQL statements.***

Keywords: Java; Persistent layer; MyBatis; JDBC.

Cited as: Jiang, C., Bao, Z., & Li, C. (2024). A Brief Analysis of the Application of MyBatis in Java Development. *Journal of Theory and Practice in Engineering and Technology, 1*(3), 25–32. Retrieved from https://woodyinternational.com/index.php/jtpet/article/view/74

1. Introduction

Today, java language is still the most popular programming language in the internet field. However, in large-scale Java project development, if developers directly perform database related operations through traditional JDBC technology, it will generate a large amount of duplicate code and the steps will be cumbersome, which will reduce the work efficiency of developers. So in Java enterprise development, developers often use persistence layer frameworks that fully encapsulate JDBC to develop persistence layers, and MyBatis is a high-performance persistence layer framework [1-3].

MyBatis is an excellent persistence layer framework. It was originally an open-source project iBatis from Apache, later renamed MyBatis, and migrated to Github. MyBatis encapsulates JDBC internally, simplifying the complex process of loading drivers, creating connections, and creating statements. Developers only need to focus on the SQL statements themselves. It supports customized SQL, stored procedures, and advanced mapping, and can establish mapping relationships between entity classes and SQL statements [4-6]. It is a semi automated ORM implementation. MyBatis can use simple XML or annotations to configure and map native information, mapping interfaces and Java POJOs (Plain Ordinary Java Objects) to records in the database [7-8]. Its encapsulation is lower than Hibernate, but it has excellent performance, is compact, easy to learn, and has a wide range of applications. MyBatis is an excellent persistence layer framework [8]. Its advantages include: ease of use: by configuring SQL statements through simple XML or annotations, developers only need to focus on business logic [9]. Flexibility: Allow the use of dynamic SQL to generate different SQL statements based on different conditions. Superior performance [10]: It combines well with Java collection classes, reduces N+1 issues, and has high performance. Easy to integrate: can seamlessly integrate with mainstream frameworks such as Spring. Simple Mapping: Provides a simple mapping method that supports mapping the ORM field relationship between objects and databases [11]. Its drawbacks include: learning costs: the need to understand concepts such as SQL statements and mapping files. Dependency on databases: Many functions and mechanisms are related to specific databases and have poor portability [12]. Manually Writing SQL: Developers need to manually write SQL statements, which increases development costs [13].

2. Basic Introduction to MyBatis

2.1 Persistent Layer

In general large-scale software development, developers usually establish a dedicated layer for interacting with databases in the development project to achieve data persistence, which is generally referred to as the persistence layer [14].

2.1 MyBatis

MyBatis, as the preferred persistence framework for developers, encapsulates traditional JDBC within it. Compared to traditional JDBC, myb atis has many advantages: unified management of SQL statements, dynamic concatenation of SQL statements, ORM mapping, and so on [15].

3. Comparison between MyBatis and JDBC

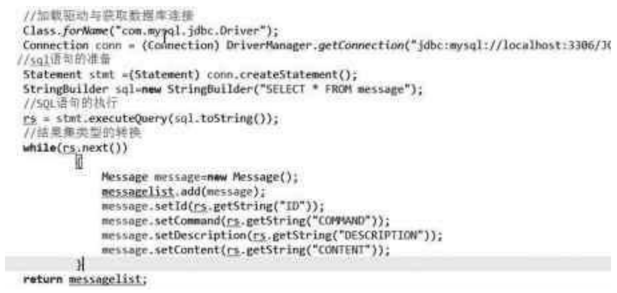
3.1 Introduction to JDBC

Due to the cumbersome database access process of JDBC, in actual project development, developers generally choose a persistence layer framework that encapsulates JDBC to interact with the database. Compared to the tedious database operation steps of traditional JDBC, the MyBatis framework supports customized SQL, stored procedures, advanced mapping, avoiding manual and JDBC code setting parameters and obtaining result sets, and many other functions [16].

3.2 A Simple Case Study of Using JDBC to Interact with a Database

This case is a functional module in the automatic reply robot project, which queries specified information based on multiple key fields on the front-end page and returns the results to the front-end page. Here, jd bc is used to complete this query function in order to introduce the general process of operating a database with jdbc:

Step 1: Load the MySQL driver and obtain the corresponding database for this case; Step 2: Preparation of SQL statements (initialize SQL query statements, and then use Java code to place relevant SQL conditional statements and query parameters after the SQL query statements) [17]; Step 3: Call the SQL statement to obtain the return result and assemble the returned data into the desired object; Step 4: Close the relevant resources. In this case, the design of related entity classes and database tables can refer to the chart design of the project later in this article.



```
//加载驱动与获取数据库连接
Class.forName("com.mysql.jdbc.Driver");
Connection conn = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/J
//sql语句的准备
Statement stmt =(Statement) conn.createStatement();
StringBuilder sql=new StringBuilder("SELECT * FROM message");
//SQL语句的执行
rs = stmt.executeQuery(sql.toString());
//结果集类型的转换
while(rs.next())
    {
        Message message=new Message();
        messagelist.add(message);
        message.setId(rs.getString("ID"));
        message.setCommand(rs.getString("COMMAND"));
        message.setDescription(rs.getString("DESCRIPTION"));
        message.setContent(rs.getString("CONTENT"));
    }
return messagelist;
```

Figure 1: Code snippet for implementing information query function in JDBC

3.3 Comparison between JDBC and MyBatis

In the above simple case, the traditional process of JDBC interacting with databases has many flaws:

The first point is that in JDBC, the acquisition and release of data sources are too frequent, and developers cannot easily configure data source information; In my Batis, developers can easily configure data source information in the core configuration file as a configuration file.

Secondly, in JDBC, SQL statements are not separated from program code, which makes it difficult to maintain the program code; In the my Batis framework, developers can configure SQL statements together in the mapping file, allowing them to focus on business logic without having to expend energy on SQL statements;

Thirdly, in JDBC, due to the lack of separation between program code and SQL statements, developers face great difficulty in concatenating SQL statements;

Fourthly, in the aforementioned JDBC instance, developers need to extract results from the Result Set and convert them to the specified type; In My batch, developers can map database table objects to JavaBeans through configuration files without the need for additional type conversion operations;

The many advantages of MyBatis will be reflected in the specific project code later, and will not be elaborated here.

4. Basic Principles of MyBatis Framework Operation

4.1 Core Objects and Corresponding Configuration Files in MyBatis:

Core configuration file: In this file, developers mainly configure the data source and mapping file path of the project [18].

Mapper.xml: In the mapping file, the relevant configuration of ORM mapping and all SQL statements are mainly placed;

The name of this object indicates that it is the producer of the SQL session instance. During the runtime from project initiation to project completion, this object provides all the SQL sessions that the entire project needs to use. At the start of the project, the object is created by the data source configuration class by reading the core configuration file of MyBatis. After the object is created, developers can use it to obtain a continuous stream of SQL sessions and interact with the database through sqls sessions. Generally speaking, if a development project involves multiple databases, it is necessary to obtain multiple SQL session scripts.

SQL Session: The database operations of the project are executed through this object.

Executor: Executor is a sub object in SQL session that performs specific database operations during each SQL session execution.

Mapped Statement: Mapped Statemet is a parameter in the execu tor object. This parameter is used to assemble all the information in the SQL statements in the mapping file, preparing for the specific database operations of the Executor object.

4.2 MyBatis Execution Flowchart:

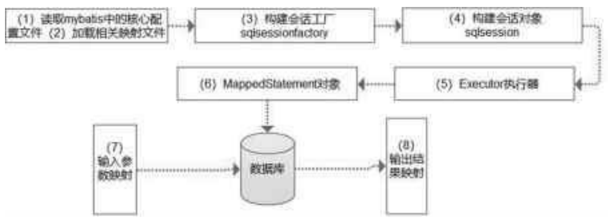

(1) 读取mybatis中的核心配置文件 (2) 加载相关映射文件
(3) 构建会话工厂 sqlsessionfactory
(4) 构建会话对象 sqlsession
(5) Executor执行器
(6) MappedStatement对象
(7) 输入参数映射
数据库
(8) 输出结果映射

Figure 2: Schematic diagram of MyBatis execution

5. The Application of MyBatis in the Development of Automatic Reply Robot Projects

5.1 Structure of the Entire Project

(1) Project Structure Diagram:

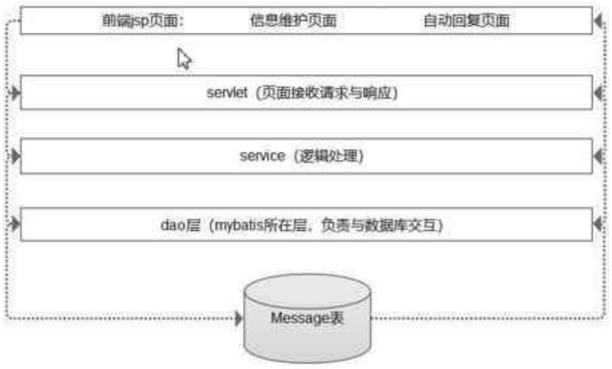

前端jsp页面:
信息维护页面
自动回复页面
servlet（页面接收请求与响应）
service（逻辑处理）
dao层（mybatis所在层，负责与数据库交互）
Message表

Figure 3: Project Structure Diagram

(2) Introduction to the functions of the project

As the focus of the automatic reply robot project case is to introduce the specific application of the MyBatis framework in the development of the project persistence layer, based on the interface planning of the dao layer, the author can generally divide the automatic reply robot case into two functional blocks:

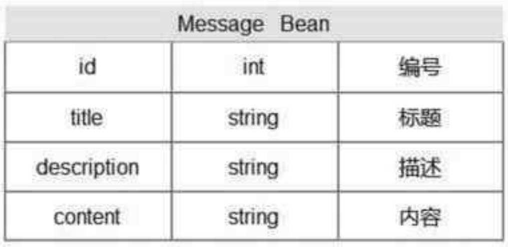
The first functional block: The corresponding servlet receives a request from the front-end information management page and passes it down to the DAO interface call class. Then, the DAO layer performs add, delete, modify, and query operations on the corresponding tables in the database, and sends the resulting dataset back to the front-end page step by step [19];

Second functional block: The corresponding servlet receives the keywords of the previous reply page request and passes them down to the DAO layer interface calling class. Afterwards, the DAO layer performs a query operation on the Message table of the database and returns the query result to the front-end reply page [20].

5.2 Specific Implementation of the Project

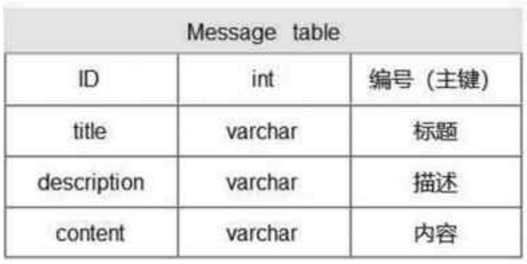
(1) Related entity class design, database table structure design, configuration file content:

Message class design:



Message Bean		
id	int	编号
title	string	标题
description	string	描述
content	string	内容

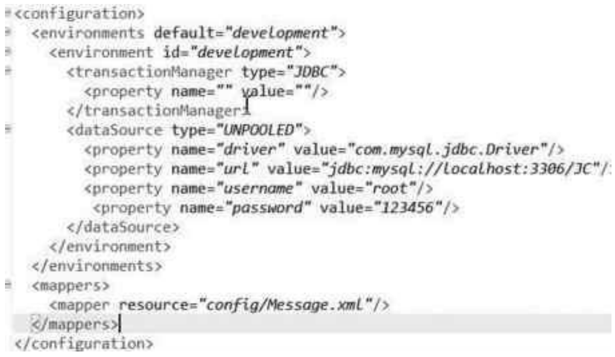
Figure 4: Database Table Structure Design for Project Entity Class Design:



Message table		
ID	int	编号（主键）
title	varchar	标题
description	varchar	描述
content	varchar	内容

Figure 5: Design of Project Database Tables

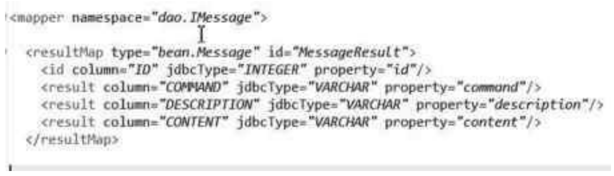
The content of the MyBatis configuration file in this project case includes the writing of the project's data source and mapping file path.



```
<configuration>
  <environments default="development">
    <environment id="development">
      <transactionManager type="JDBC">
        <property name="" value=""/>
      </transactionManager>
      <dataSource type="UNPOOLED">
        <property name="driver" value="com.mysql.jdbc.Driver"/>
        <property name="url" value="jdbc:mysql://localhost:3306/JC"/
        <property name="username" value="root"/>
         <property name="password" value="123456"/>
      </dataSource>
    </environment>
  </environments>
  <mappers>
    <mapper resource="config/Message.xml"/>
  </mappers>
</configuration>
```

Figure 6: MyBatis Core Configuration File Content

Configuration of important information in Mapper.xml file: Mapping of bean objects to message table:

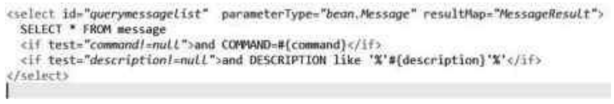


```
<mapper namespace="dao.IMessage">

  <resultMap type="bean.Message" id="MessageResult">
    <id column="ID" jdbcType="INTEGER" property="id"/>
    <result column="COMMAND" jdbcType="VARCHAR" property="command"/>
    <result column="DESCRIPTION" jdbcType="VARCHAR" property="description"/>
    <result column="CONTENT" jdbcType="VARCHAR" property="content"/>
  </resultMap>
```

Figure 7: Mapping configuration between entity classes and database tables

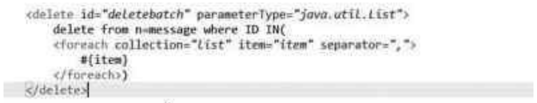
The SQL statements corresponding to the information query on the information maintenance page were concatenated using OGNL expressions:



```
<select id="querymessagelist"  parameterType="bean.Message" resultMap="MessageResult">
  SELECT * FROM message
  <if test="command!=null">and COMMAND=#{command}</if>
  <if test="description!=null">and DESCRIPTION like '%'#{description}'%'</if>
</select>
```

Figure 8: SQL statement for information query

Batch deletion of information on the information maintenance page using SQL statements: Conducted corresponding ID traversal using OGNL expressions:



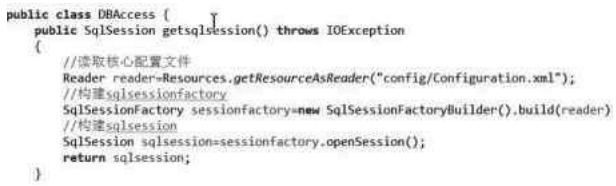
```
<delete id="deletebatch" parameterType="java.util.List">
    delete from n=message where ID IN(
    <foreach collection="list" item="item" separator=",">
        #{item}
    </foreach>)
</delete>
```

Figure 9: SQL statement for information deletion

... Similar SQL statements will not be repeated one by one.

(2) Design of data source acquisition class and interface design for persistence layer:

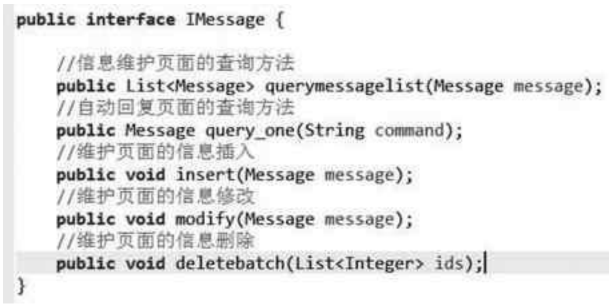
Data source retrieval class: Read the configuration file to build the SQL session factory, obtain the SQL session, and return it to the calling object:



```
public class DBAccess {
    public SqlSession getsqlsession() throws IOException
    {
        //读取核心配置文件
        Reader reader=Resources.getResourceAsReader("config/Configuration.xml");
        //构建sqlsessionfactory
        SqlSessionFactory sessionfactory=new SqlSessionFactoryBuilder().build(reader)
        //构建sqlsession
        SqlSession sqlsession=sessionfactory.openSession();
        return sqlsession;
    }
```

Figure 10: Design of Data Source Acquisition Class

DAO layer interface: Based on the two major functional blocks of the previous project and the SQL statements written in the configuration file, the following functions were designed in the DAO layer interface design:



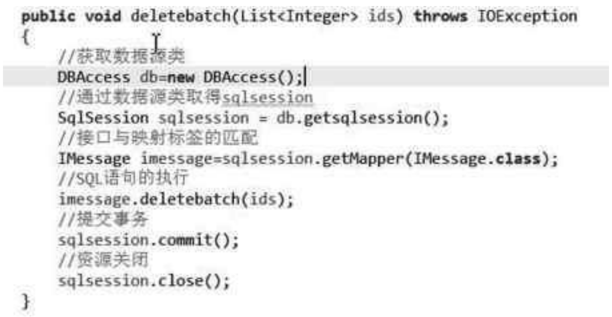
```
public interface IMessage {

    //信息维护页面的查询方法
    public List<Message> querymessagelist(Message message);
    //自动回复页面的查询方法
    public Message query_one(String command);
    //维护页面的信息插入
    public void insert(Message message);
    //维护页面的信息修改
    public void modify(Message message);
    //维护页面的信息删除
    public void deletebatch(List<Integer> ids);
}
```

Figure 11: Design of DAO layer interface

(3) Implementation of corresponding interface functions

Implementation of corresponding interface functions (taking batch deletion of information pages as an example here): First, use the data source class to obtain the SQL session; The 'interface' of the DAO layer is then used to obtain the corresponding 'mapper. class' through the' get mapper 'method of the session object; Finally, execute the corresponding SQL statement through the persistence layer interface and commit the transaction:



```
public void deletebatch(List<Integer> ids) throws IOException
{
    //获取数据源类
    DBAccess db=new DBAccess();
    //通过数据源类取得sqlsession
    SqlSession sqlsession = db.getsqlsession();
    //接口与映射标签的匹配
    IMessage imessage=sqlsession.getMapper(IMessage.class);
    //SQL语句的执行
    imessage.deletebatch(ids);
    //提交事务
    sqlsession.commit();
    //资源关闭
    sqlsession.close();
}
```

Figure 12: Implementation of Information Deletion Function

5.3 Project Summary

The front-end part of this project case involves two JSP pages; There are three vertical sections in the backend. At the control layer, a servlet is used to receive requests from front-end pages and return the corresponding result set; The significance of the logical layer in this case is not significant; In the DAO layer of the project, MyBatis was used to complete the interaction with the database [5]. In the design of the persistence layer, an interface oriented programming approach is adopted to match interfaces with related mapping labels, and then execute SQL statements by calling interfaces.

The above robot automatic reply project case introduces the basic principles of MyBatis framework operation, and uses important technologies such as interface programming and dynamic SQL statements to complete the functional block implementation of the entire persistence layer and database interaction.

6. The Advantages, Disadvantages, and Application Scenarios of MyBatis

6.1 Advantages and Disadvantages of MyBatis:

In Java development, compared to other DAO layer frameworks, Myba tis has many advantages:

Simple and easy to learn, for many beginners in Java development, using myba tis as a DAO layer framework is a good decision;

Flexibility: In the MyBatis framework, all SQL statements are placed in XML files, and the separation of SQL statements from logical code improves project development efficiency; In JDBC, the concatenation of SQL statements is quite cumbersome, while in MyBatis, the concatenation of SQL statements is very easy to implement;

Eliminating the coupling between SQL and program code: MyBatis is located in the persistence layer, focusing on database interaction and not participating in business logic, which makes the project system more clear and conducive to improving development efficiency; The measure of placing SQL statements in configuration files allows developers to focus on Java object-oriented development without having to spend too much energy on SQL statements.

By implementing the same database access actions, MyBatis can reduce code volume by more than half compared to traditional JDBC.

The shortcomings of MyBatis:

In MyBatis, SQL statements are uniformly configured in XML files, and writing a large number of SQL statements in the configuration file can be cumbersome;

The MyBatis framework focuses on SQL statements, which can make database portability very poor.

6.2 Application Scenarios of MyBatis:

Compared to other persistence frameworks, MyBatis focuses more on SQL, allowing users to implement ORM functionality more through SQL statements. Although this model is flexible, it can reduce development efficiency in developing a relatively mature project. Therefore, if an enterprise uses MyBatis, it needs to encapsulate it again according to the actual situation of the development project, making database access more concise and applicable, thereby improving development efficiency.

References

[1] Wu, Z. (2024). An Efficient Recommendation Model Based on Knowledge Graph Attention-Assisted Network (KGATAX). arXiv preprint arXiv:2409.15315.
[2] Zhu, Z., Wang, Z., Wu, Z., Zhang, Y., & Bo, S. (2024). Adversarial for Sequential Recommendation Walking in the Multi-Latent Space. Applied Science and Biotechnology Journal for Advanced Research, 3(4), 1-9.
[3] Shen, Z. (2023). Algorithm Optimization and Performance Improvement of Data Visualization Analysis Platform based on Artificial Intelligence. Frontiers in Computing and Intelligent Systems, 5(3), 14-17.
[4] Wang, Z., Zhu, Y., Li, Z., Wang, Z., Qin, H., & Liu, X. (2024). Graph neural network recommendation system for football formation. Applied Science and Biotechnology Journal for Advanced Research, 3(3), 33-39.
[5] He, C., Yu, B., Liu, M., Guo, L., Tian, L., & Huang, J. (2024). Utilizing Large Language Models to Illustrate Constraints for Construction Planning. Buildings, 14(8), 2511. https://doi.org/https://doi.org/10.3390/buildings14082511
[6] Wu, Z. (2024). Deep Learning with Improved Metaheuristic Optimization for Traffic Flow Prediction. Journal of Computer Science and Technology Studies, 6(4), 47-53.
[7] Zhu, Z., Wang, Z., Wu, Z., Zhang, Y., & Bo, S. (2024). Adversarial for Sequential Recommendation Walking in the Multi-Latent Space. Applied Science and Biotechnology Journal for Advanced Research, 3(4), 1-9.
[8] Tang Quan. (2017). Application and practice of Ssm framework in javaee teaching. Fujian Computer, 33(12), 3.
[9] Xu, Y., Shan, X., Guo, M., Gao, W., & Lin, Y. S. (2024). Design and Application of Experience Management Tools from the Perspective of Customer Perceived Value: A Study on the Electric Vehicle Market. World Electric Vehicle Journal, 15(8), 378.
[10] Ren, Z. (2024). A Novel Topic Segmentation Approach for Enhanced Dialogue Summarization. World Journal of Innovation and Modern Technology, 7(4), 42-49.
[11] Rong Y D. (2015). Application research on mybatis persistence layer framework. Information Security and Technology (12), 3.
[12] Wang, Z., Yan, H., Wang, Y., Xu, Z., Wang, Z., & Wu, Z. (2024). Research on autonomous robots navigation based on reinforcement learning. arXiv preprint arXiv:2407.02539.
[13] Cook for the army. Comparative study on Web framework and its application in Vehicle System. (Doctoral dissertation, Jinan University).
[14] He, C., Liu, M., Wang, Z., Chen, G., Zhang, Y., & Hsiang, S. M. (2022). Facilitating Smart Contract in Project Scheduling under Uncertainty—A Choquet Integral Approach. Construction Research Congress 2022, 930 – 939. https://doi.org/10.1061/9780784483961.097

[15] Huang Jincong, Huang Yanbo, Ye Haobin, & Huang Yanfei. (2021). A configuration method of SQL statement based on MyBatis, system. CN201711143542.4.
[16] Qiao LAN. (2017). Research and application of javaee Data persistence layer based on mybatis and spring. Information and Computers (8), 4.
[17] Li Shan, Jia Yanping, & Da Hu. (2017). Application of Mybatis reverse engineering in javaee. Communication World (24), 1.
[18] Ji, H., Xu, X., Su, G., Wang, J., & Wang, Y. (2024). Utilizing Machine Learning for Precise Audience Targeting in Data Science and Targeted Advertising. Academic Journal of Science and Technology, 9(2), 215-220.
[19] Ren, Z. (2024). Enhanced YOLOv8 Infrared Image Object Detection Method with SPD Module. Journal of Theory and Practice in Engineering and Technology, 1(2), 1-7. Retrieved from https://woodyinternational.com/index.php/jtpet/article/view/42
[20] Wang, Y., Lu, Y., Xie, Z., & Lu, G. (2021, October). Deep unsupervised 3d sfm face reconstruction based on massive landmark bundle adjustment. In Proceedings of the 29th ACM International Conference on Multimedia (pp. 1350-1358).