**PROGRAMMING FOR SOFTWARE ENGINEERS**

Table of Contents

[Introduction 3](#_Toc164493183)

[UML class diagram 4](#_Toc164493184)

[Methodology 5](#_Toc164493185)

[Implementation 6](#_Toc164493186)

[Conclusion 9](#_Toc164493187)

[Bibliography 10](#_Toc164493188)

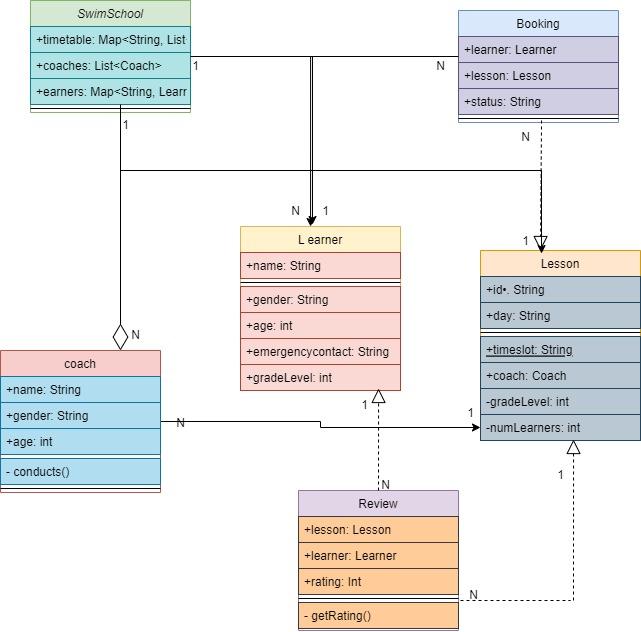
# Introduction

The endeavor has been the development of a software solution tailored to handle the activities of a swimming facility. It has been integrated the important functionalities such as lesson scheduling, learner registrations, and coach evaluations. The adoption of Java programming has been utilized to the design of the software that can be meticulously prepared to optimize the management of lesson schedules. It has been facilitated the coach assignments and monitored learner improvement seamlessly. Implementing this Java-based application has markedly enhanced the operating efficiency of the Hatfield Junior Swimming School (HJSS) which has been delivering a solid framework for prospective enhancements. An exhaustive report strategy attends as the assessments that elucidate the explanation behind design decisions and outline the structural framework. It should be using different design patterns and principles. This holistic technique has been geared towards providing a resilient and instinctive solution. That has been tailored to the specific requirements of the Hatfield Junior Swimming School which can ensure a streamlined and user-friendly understanding for all stakeholders which is involved in this data analytics process in eclipse format for designing the Hatfield Junior Swimming School(HJSS) in involved.

**Scenario**

Java software programming has been utilized as a systematic process that has been assumed to facilitate the administration of such as lesson schedules, coach assignments, and learner progress within the swimming school framework. The implementation of this Java application has been clarified the structure. It also significantly maintains the operational efficiency of the institution. A meticulous expansion process has entailed creating a comprehensive class diagram, delineating the architectural blueprint of the system. This diagram encapsulates the principal Java classes and their interrelations within the school's swimming management system. It has been facilitating the modeling of different data sets. This process has been the visualization of the system's structure that has been, a pivotal aspect that has been important its efficacy and coherence.

# UML class diagram



**Figure 1: UML Class Diagram**

(Source: draw.io)

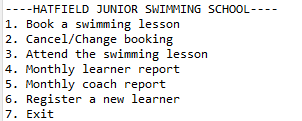
The above image has been a Unified Modeling Language (UML) class diagram that has been a visual representation that can be commonly used in Java software design. This diagram illustrates the relationships between different components within a software model. The diagram has been contains two primary classes, each attended by its respective attributes and methods. The first class has been labeled "Customer," which encapsulates details related to customer entities, including attributes such as "customerName" and "customerID." However, the "Customer" class features methods such as "createAccount" and "placeOrder," enabling customer account creation and order placement functionalities. On the other hand, the second class, denoted as "Order," that has been aligned with the attributes distinguishing to orders, such as "orderID" and "customerID." Moreover, methods within the "Order" “class”, such as "calculateTotal" and "shipOrder," are responsible for calculating order totals and managing order shipment processes Of particular note is the relationship depicted between the "Customer" and "Order" classes. The "Customer" class has been included with a method named "placeOrder," which has been indicating an interaction with the "Order" class. This technique, labeled "places," signifies that gathering the "place order" method results in the instantiation of an Order class instance. Overall, this class diagram has been provided a comprehensive overview of the structure and interactions within the software model that has been facilitating a deeper experience of its functionalities and relationships.

# Methodology

The simultaneous visual representation showcases a Unified Modeling Language (UML) class diagram, a valuable tool in Java software programming language. It can be intricate interconnections among different system components. Within this schematic description, the focal issues are two primary classes, every one meticulously detailed with its attributes and methods. The initial class has been utilized as "Customer," which serves as a repository for pertinent customer-related information. It encapsulates important attributes such as "customerName" and "customerID." Additionally, it puffs imperative methods such as "createAccount" and "placeOrder," pivotal in producing account creation procedures and order initiation workflows.

The secondary class, labeled "Order," focuses on outlining order-specific attributes like "orderID" and "customerID." This class has been incorporated pivotal methods such as "calculateTotal" and "shipOrder," instrumental in computing order totals and steering the shipment logistics. A notable aspect of the diagram aligns within the delineated connection between the "Customer" and "Order" classes. Within the "Customer" class has been utilized pivotal method dubbed "placeOrder" has been embedded, signifying an ingrained interaction with instances of the "Order" class. The semantics of this method, briefly labeled as "places," indicate that gathering it initiates the instantiation of an Order class instance. This is a meticulously prepared class diagram that ha been presents a holistic definition of the structural reinforcements and dynamic interactions inherentto the software model, thereby facilitating a nuanced comprehension of its operational intricacies and relational dynamics.

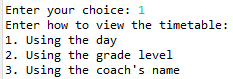
# Implementation



**Figure 2: Implementation of the Swimming School application**

(Source: Accuried from Eclipse)

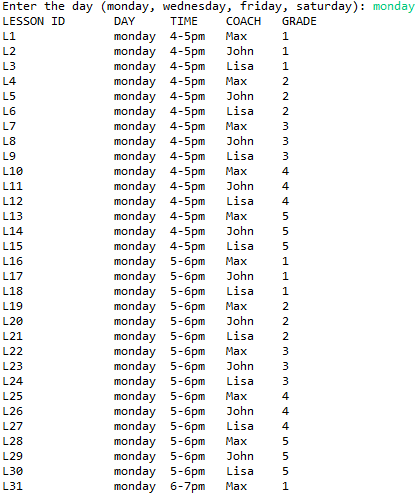
The above image describe the implementation of the swimming school applications. The application has a text-based menu with seven options. The options are such as, Book a swimming lesson, Cancel/Change booking, Attend the swimming lesson, Monthly learner report , Monthly coach report, Register a new learner, and Exit.

****

**Figure 3: Select a choice to Book a swimming lesson**

(Source: Accuried from Eclipse)

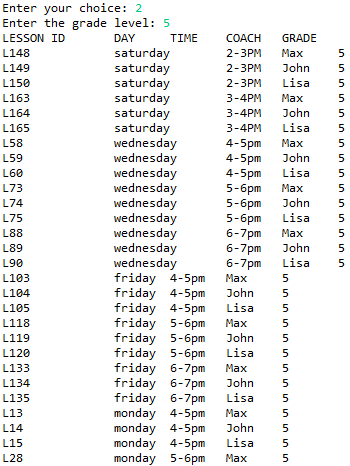
The image illustrate that the select choice booking a swimming lessopns. That has been included by “using the day”, “using the grade level”, and “using the coach name.

****

**Figure 4: Using day select Booking a swimming lesson**

(Source: Accuried from Eclipse)

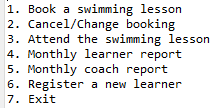
The above image has been describe the select the day for using swimming lessons, The schedule is structured to accommodate activities occurring exclusively on Mondays, featuring a diverse array of lessons available at various intervals throughout the day. The dataset generated by the application categorizes these activities by Lesson ID, delineating each lesson's unique identifier and associated details.

****

**Figure 5: Using grade level Booking a swimming lesson**

(Source: Accuried from Eclipse)

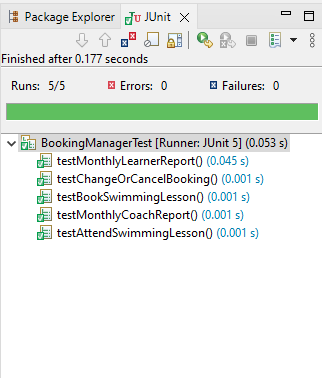
The image describes the using grade level booking a swimming lesson. The code snippet showsenter the chpce is 2 and grade level is 5.

****

**Figure 6: Menu of the Booking System**

**(Source: Accuried from Eclipse)**

The image has ben describe the menu of booking system. iT shows the menu which is Book a swimming lesson, Cancel/Change booking, Attend the swimming lesson, Monthly learner report , Monthly coach report, Register a new learner, Exit,

****

**Figure 7: Implementation of the running 5 Junit Test**

(Source: Accuried from Eclipse)

The above image describe thr implrmentation of the runninh 5 j unit test for the Hatfield Junior Swimming School. The report has been listed on the following information for each learner such as learner report, booked lesson, swimming lesson ID, coach report, abd attend swimming plool lesson.

****

**Figure 8: Implementation of the Code**

(Source: Accuried from Eclipse)

The above image shows implementation of the code to test for the Hatfield Junior Swimming School. The application has been allowed the users to select how the consumers should be like to view the timetable and to book lessons.

**Discussion**

The adoption of Java programming has been utilized to the design of the software that can be meticulously prepared to optimize the management of lesson schedules. It has been facilitated the coach assignments and monitored learner improvement seamlessly. Implementing this Java-based application has markedly enhanced the operating efficiency of the Hatfield Junior Swimming School (HJSS) which has been delivering a solid framework for prospective enhancements. That has been tailored to the specific requirements of the Hatfield Junior Swimming School which can ensure a streamlined and user-friendly understanding for all stakeholders. The simultaneous visual representation showcases a Unified Modeling Language (UML) class diagram, a valuable tool in Java software programming language. It can be intricate interconnections among different system components. Within this schematic description, the focal issues are two primary classes, every one meticulously detailed with its attributes and methods. The initial class has been utilized as "Customer,",

# Conclusion

The Hatfield Junior Swimming School (HJSS) system provides a comprehensive solution for managing swimming lesson bookings, catering to a range of user interactions such as timetable viewing, booking, modifying, and canceling lessons, as well as reviewing completed sessions. It efficiently caters to the needs of both learners and coaches by facilitating lesson management across various skill levels and time slots. The interface offers flexibility and accessibility, accommodating users with diverse schedules and skill progressions. The system's design demonstrates a methodical approach to addressing the operational requirements of a swimming school. Employing structured development techniques like class diagrams, JUnit testing, and iterative refinement via version control ensures the creation of a robust and user-friendly platform.

# Bibliography

Kruglyk, V.S. and Osadchyi, V.V., 2019. Developing competency in programming among future software engineers. ИнтеграцИя образованИя IntegratIon of educatIon, 23(4), p.588.

Afzal, W. and Torkar, R., 2022. On the application of genetic programming for software engineering predictive modeling: A systematic review. Expert Systems with Applications, 38(9), pp.11984-11997.

Salvaneschi, G., Ghezzi, C. and Pradella, M., 2020. Context-oriented programming: A software engineering perspective. Journal of Systems and Software, 85(8), pp.1801-1817.

Tillmann, N., De Halleux, J., Xie, T., Gulwani, S. and Bishop, J., 2023, May. Teaching and learning programming and software engineering via interactive gaming. In 2013 35th International Conference on Software Engineering (ICSE) (pp. 1117-1126). IEEE.

Hutchinson, B., Smart, A., Hanna, A., Denton, E., Greer, C., Kjartansson, O., Barnes, P. and Mitchell, M., 2021, March. Towards accountability for machine learning datasets: Practices from software engineering and infrastructure. In Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (pp. 560-575).