# Justification of Architecture Pattern

Upon undertaking this project for the Music School, a suitable architecture pattern needed to be selected for the project. This decision was made by taking into account the advantages and shortcomings of each available selection as well as its relevance in relation to the client’s requirements and end deliverable.

Three-tier architecture was the chosen pattern for this project as it comprises of three separate layers which are easily relatable to the business requirements of the Music School.

Firstly, there is a Client Layer available. Also referred to as the presentation layer, this sector contains the user interface of the application being developed. It is useful as it can present data in a globally acceptable format as well as take user input from fields such as: text boxes, buttons, and labels. This can be applied directly to pages such as the login portal, registration portal or student/staff and/or administration sections of the web based application being developed for the Music School.

The second layer within the three-tier architecture pattern is the business layer. This layer carries the business logic and validates the: business data, business rules and calculations and data input from users. Essentially, it acts as intermediary between the Client Layer and Data Layer to ensure that information is being delivered, received and updated quickly and accurately. This is essential for several business applications for the Music School, including: the integrity of data being sent and received, management of website traffic, exception handling and rules, as well as scalability; under the assumption that there is clientele/staff growth in the foreseeable future.

The final layer in the three-tier pattern is the data layer. This layer relates to the actual business database. The Database Layer allows the connection for the back-end database to the other layers. It also provides the ability to insert, update, delete or extract information from the data based on user interaction or input. ("Difference Between Two-Tier and Three-Tier Architecture?", 2018)

To reaffirm the decision for using a three-tier architecture pattern, the structure can be compared to single or two-tier architecture patterns. A 1-tier pattern comprises of each layer of a three-tier pattern, however this is combined in a single component. This creates difficulty for developers in the cohesion and clarity of their code. It also introduces issues in relation to identifying boundaries and the separation of programming logic in the web application.   
Two-tier separates the database layer however; the presentation (HTML) and business logic is still co-mingled. Despite being easier to maintain, modify and uphold apt communication speeds, this architecture remains cost-ineffective and as a rise in users increase performance can heavily be degraded as each function required for the web application is still not yet separated. (Marston, 2018)  
  
Juxtaposing the two aforementioned patterns there are distinguishable benefits which arise from the use of three-tier architecture. Despite, the initial increase in complexity and effort upon setup, the three-tier pattern provides: enhanced performance through the use of ‘lightweight persistent objects’, scalability, as well as, improved data integrity and security ("Difference Between Two-Tier and Three-Tier Architecture?", 2018). For these reasons, the three-tier pattern remained the most suitable and advantageous option.

# References

* *Difference Between Two-Tier and Three-Tier Architecture?*. (2018). *Software Testing Class*. Retrieved 17 April 2018, from <http://www.softwaretestingclass.com/what-is-difference-between-two-tier-and-three-tier-architecture/>
* Marston, T. (2018). *What is the 3-Tier Architecture?*. *Tonymarston.net*. Retrieved 17 April 2018, from <http://www.tonymarston.net/php-mysql/3-tier-architecture.html>