# Analysis

## Literature Review

### Database Approach: SQL vs noSQL

When first deciding to undertake this project, this question arose to the forefront of the data management and storage options, Bhogal and Choksi *(2015)*, state *“With the emergence of Big Data, the use of NoSQL (Not only SQL) technology is rising rapidly among internet companies and other enterprises. Benefits include simplicity of design, horizontal scaling and finer control over availability. NoSQL databases are increasingly considered a viable alternative to relational databases, as more organizations recognize that its schema less data model is a better method for handling the large volumes of structured, semi structured and unstructured data, being captured and processed today”.* This posed the question when creating this application which would be better for the storage of data at the scale required by this application.

#### Types of NoSQL Databases

There are also several different types of NoSQL databases that can be categorized into the following categories, Key-Value Store Databases, Column-Oriented Databases, Document Store Databases, Graph databases, and Object-Oriented databases. Nayak et al. states the following on key value store databases “The key-value data stores are pretty simplistic but are quite efficient and powerful model. It has a simple application programming interface (API). A key value data store allows the user to store data in a schema less manner” (2013), the stores used in Key-Value store databases share a similarity to hash tables where keys are used as the indexes, Neyak et al states “making it faster than RDBMS Thus the data model is simple” (2013), examples of key value store databases are Amazon DynamoDB and RIAK. Column-Oriented Databases are hybrid row/column stores, in column-oriented databases data is not stored in tables but rather in large, distributed architectures where each key is associated to a singular or multiple columns or attributes. Data is stored in a manner where it can be quickly aggregated with a lesser amount of I/O activity and offers large scalability in the storage of data. Examples of Colum-oriented databases are NoSQL applications such as Googles Big Table or Apaches Cassandra. Document Store Databases are databases that store data in a document format, this allows these databases to offer “great performance and horizontal scalability options” (Neyak et al, 2013), the documents stored in a document-oriented database are somewhat a resemblance to the role records play in traditional relational databases but add a large scale of flexibility due to being schema less, Document oriented databases can use formats such as XML, PDF and JSON, some of the variants of Document Store Databases are MongoDB which was briefly mentioned above in a performance comparison of NoSQL vs SQL performance and another version of a document based database is CouchDB. Graph databases are NoSQL databases which store the data in a graph format which consists of nodes and edge where the nodes act as objects and the edges act as the relation between the objects. Graph databases use a technique called index free adjacency where each node consists of a pointer which points to an adjacent node. Graph databases are schema less much like other types of NoSQL databases and efficient storage of semi structured data. Queries in Graph databases are “expressed as traversals, thus making graph databases faster than relational databases” (Neyak et al, 2013), The current notable providers of Graph databases are Neo4j, Amazon Neptune, AllegroGraph, ArangoDB, InfiniteGraph and OrientDB. The final type of NoSQL database is Object oriented databases these are databases where the data or information that will be stored are represented as objects, like object-oriented programming objects, this means that OODB (Object oriented databases) could be considered a combination of OOP and database principles. OODB’s can offer various features of OOP such as encapsulation, polymorphism and inheritance. In an OODB the class, objects and attributes are comparable to a “table, tuple and columns in a tuple in RDBMS respectively” (Neyak et al., 2013). Each object has a unique identifier which is used to represent itself. An example of an Object-Oriented Database is db4o (database for objects) but this solution became end of life in 2014. After reviewing the different type of NoSQL databases if the application was to use a NoSQL database the type of NoSQL Database that would be best suited to the application would be a Document Store Database such as CouchDB or MongoDB, the next section will look into the performance of Document Store/Document-Oriented databases versus traditional relational databases.

#### Performance of NoSQL Document-Oriented Databases Vs Relational SQL Databases

Upon researching further into the performance difference between SQL and NoSQL it became apparent that both SQL and NoSQL have their advantages and disadvantages and their own use cases, In a study completed by Li and Manoharan (2013) of the university of Auckland they tested the performance between various NoSQL solutions as well as Microsoft SQL Express the study found that most operations using SQL were close to or as fast as the NoSQL alternatives tested the weakest area for SQL was with write operations placing it in the bottom of the pecking order versus most of the NoSQL alternatives. These findings are further solidified in another study completed by Parker, Poe and Vrbsky (2013), of The University of Alabama whose report focused on the performance comparison of MongoDB a NoSQL alternative and MySQL Database their study found that “MongoDB has better runtime performance for inserts, updates and simple queries” (Parker, Poe and Vrbsky, 2013) whereas SQL had better performance performing tasks querying non-key attributes of a table as well as aggregate queries. They also state in their findings that a NoSQL database would be a better solution for larger data sets with everchanging schema or in cases where the “queries performed will be less complex” (Parker, Poe and Vrbsky, 2013).

#### Conclusion

Upon completing the review of this topic the application will be using a MySQL database to store data as the data that is to be stored by the application although not complex in theory may require complex queries to be retrieved and modified due to the nature of the application especially in the use case where a group booking would occur as this would mean to retrieve the data multiple tables would have to be joined and queried to return the data required by the application namely the details of the participants such as name and email address as well as location, date and time of the booking for the system to generate a .ics file to send to the required participants with the correct information. This decision has also been partially made due to being more knowledgeable on MySQL databases versus NoSQL alternatives and also due to possible limitations of the server if newnumyspace is to be used due to restricted access to modify the server and install the packages required to use a NoSQL database, also in using a NoSQL database it would require that new knowledge is gained during the course of the project which may cause time spent on other key areas to be reduced and in turn reducing the overall quality of the finished product.

### User Experience and Accessibility

This topic was included in the literature review due to the wide range of users that could possibly use the web application once it is developed as users that could be able to use the system may not have the same ability whether that be technical ability or other challenges using the system because of disabilities. According to a study performed by Johnson Et Al. (2007) Both users with disabilities and elderly web users are likely to be faced with difficulties using the application due to usability issues. Their study also found that users from the elderly/disabled group that were tested in the study preferred a design that featured descriptive links and buttons as this “enabled them to find where they needed to go much more quickly and easily” (Johnson and Kent, 2007), They also found that restricting the input methods used and on screen explanations made the application easier to understand and one of the members of the elderly/disabled test group in the study stated “it was obvious what I had to do as it was well explained with the links and instructions” (Johnson and Kent, 2007), The measures used to evaluate usability are Time to learn, Speed of performance, Rate of errors by users, Retention over time and Subjective satisfaction which focus on the goals of effectiveness and efficiency (Shneiderman, Plaisant, Cohen and Jacobs, 2010.). In the book Web Accessibility: Web Standards and Regulatory Compliance Thatcher et al (2006) suggests recruiting users with disabilities and include users with different disabilities when focusing on the accessibility of an application. The common theme in the literature reviewed seems to be that to create a good user experience for the intended users to include those users in testing of the system, it is planned to undertake user testing on the application once it is in a workable state and the testers will be asked for feedback not only on the intended use of the application but also identify if the application is simple to use not only for those who are able but also those how have some form of disability, throughout the development process the W3 Web Accessibility Initiative (WAI) Web Content Accessibility Guidelines (WCAG) success criterion and techniques will be studied and followed rigidly to ensure that the standards of the WCAG are met in line with the W3 WAI Guidelines, the WCAG 2.1 requirements will be followed for this project when it comes to User Experience and accessibility of the application.

### Scheduling

#### What is scheduling and how is It relevant to the project?

“Scheduling is a decision-making process that is used on a regular basis in many manufacturing and services industries.” (Pinedo, n.d.), scheduling manages the allocation of resources to tasks over specified time periods with the goal of optimizing one or more objectives. Resources and tasks in an organisation can take various forms, in the instance of the application the resource could be the web server and the task could be storing a lecturer’s availability in the database. Another example could be to display a requested lecturers schedule to a student to allow a tutorial booking to be made. Some form of scheduler will need to be used in the application in order to send email reminders to the required parties that are attached to a booking, the scheduler would have to consider the priority of the task, for instance if the scheduler had a task to run once daily say at 7AM in the morning if any more bookings are made for that day then that would mean that no reminder would be sent for a booking made on the same day for later that day. With scheduling each task may feature a start time, due date and priority level tied to it and the aim of the task could be to reduce the completion time of tasks or to reduce the number of tasks completed after their due date. Scheduling plays an important role in many different settings such as manufacturing and production as well as in the majority of information processing environments. Scheduling also plays an important role in distribution and transportation settings as well as other variants of service industries. It plays such a key role as a decision-making process in the aforementioned industries, it’s stated that it may not be immediately clear the impact that schedules may have on key objectives, in practice it is often found that the choice of schedule has significant impact on a systems performance rather than choosing a schedule at random it would make more sense to invest yourself in identifying a suitable schedule. It is stated “Scheduling can be difficult from a technical as well as from an implementation point of view” (Pinedo, n.d.). Some of the difficulties encountered on the technical side of things are “similar to the difficulties encountered in other forms of combinatorial optimization and stochastic modelling” (Pinedo, n.d.). Whereas the difficulties on the implementation of scheduling are reliant on the accuracy of the analysis model used of the proposed scheduling problem and the reliability of the input data that is required.

#### Task Scheduling Algorithms

There are three different categories of heuristic, hybrid, and energy efficient algorithms in this review the focus will be on heuristic and hybrid scheduling approaches as these are the most appropriate to the system being built. Heuristic task scheduling supposedly provides an optimal solution as it uses knowledge bases for making the scheduling decisions, there can be either static or dynamic heuristic approaches. Static scheduling presumes that the tasks arrive at the same moment of time and they are independent of the systems current resource states and availability whereas dynamic scheduling the tasks are described as “dynamic in nature” (Mathew, Sekaran and Jose, 2014) as tasks can arrive at different times and are also dependent on the systems current state.

The static scheduling algorithms include First Come First Serve (FCFS) this method retrieves the tasks and queues them until the resources required to complete the task are available the resources are then assigned to the tasks based on the arrival time of the task, this is a les complex algorithm and does not consider anything other than the arrival times of the tasks to schedule tasks to machines. Another static scheduling method is Round Robin (RR) this uses a similar FCFS technique for the scheduling of tasks but reserves resources for each task for a set period the task is then pre-empted and queued until its next opportunity of execution. Minimum Completion Time (MCT) and Minimum Execution Time (MET) are another two heuristic scheduling strategies MET assigns machines to tasks based on which resource will complete the task the fastest but does not take into consideration resource availability at the point of scheduling which in turn can cause load imbalance. MCT Algorithms select machines to schedule tasks based on the expected completion time of the task across all available machines, it also considers the current load of the machine before scheduling a task on said machine. Another two heuristic algorithms for task scheduling are Min-Min and Max-Min, a Min-Min algorithm identifies the smallest task first from the job queue and assigns it to a machine that provides the shortest or minimum completion time for the proposed task, although this overall increases the total completion time of all tasks as it does not consider the load of the available machines before scheduling the task and simply assigns the smaller tasks to the machines with the fastest completion time. Max-Min on the other hand prioritises tasks with the largest or maximum completion time and assigns it to a machine with the smallest or minimum completion time for the selected task, this means smaller tasks are left until later and the algorithm also does not consider the current load of the systems. Although Max-min increases the throughput of the system compared to the min-min algorithms and the length of time from the start of tasks in the queue to the end of the queue often referred to as makespan, although Max-Min allows for tasks to be completed in parallel therefore smaller tasks can be executed on machines when they are free which in turn results in a better makespan and load balancing than the Min-Min algorithms.

The dynamic scheduling algorithms can be broken down into two different modes one of which is online mode which tasks are assigned and scheduled instantly upon arrival and after this the scheduling result cannot be changed, the online heuristic is suitable in cases where the arrival rate of tasks are low. Some of the online mode algorithms are Opportunistic Load Balancing (OLB) which assigns each task arbitrarily in order to the next machine that is expected to be available without taking into consideration of the expected execution time on the first available machine. There is also a variant of MET and MCT in online mode which works very similarly to the static scheduling heuristics MET and MCT methods, the final algorithms of online mode scheduling are Switching Algorithm (SA) which “uses MCT and MET heuristics in a cyclic fashion depending on the load distribution across all the machines” (Nagadevi, Satyapriya and Malathy, 2013). MET allows the SA to choose the best machine for tasks but could cause load imbalance by assigning large number of tasks to the same machines whereas MCT can balance the load more efficiently but may not assign tasks to machines that provide the tasks with a Minimum execution time. SA allows for MET to be used if tasks are arriving in random mixed order at the expense of load balance to a certain threshold then switch to MCT to balance the load across the machines. The final online mode algorithm is P-Percent Best (KPB) which considers a certain subset of machines when scheduling a task of which are formed by picking the k best machines based on the execution times of the tasks. This means that a “good value of k schedules a task to a machine only within a subset formed from computationally superior machines” (Nagadevi, Satyapriya and Malathy, 2013). The main aim of KPB is to avoid scheduling the current task to a machine which could be more suited to tasks that have not yet arrived this leads to a shorter makespan compared to the MCT heuristic. The batch mode algorithms are Min-Min and Max-Min which are similar to the static scheduling Min-Min and Max-Min algorithms, the final batch mode algorithm is a Sufferage heuristic which assigns tasks to a machine that would suffer most if that specific machine was not assigned to that task, in each scheduling event a suffrage value is calculated which comprises of the “difference between the first and second earliest completion time” (Nagadevi, Satyapriya and Malathy, 2013). The sufferage heuristic also allows rescheduling if it is found a task would be better suited to a machine than a task that is already assigned it will be unassigned and added back to the job queue.

#### Conclusion

In the current iteration of the system cron will be used to schedule jobs which is based on a FCFS algorithm which checks for existing jobs and determines the next time that the command must run, cron will need to be used in order to check the database for any upcoming bookings and then execute a php file which will allow for email reminders to be sent to the relevant parties, in the future a different task scheduling heuristic may be better suited for the system as traffic to the application increases and resources are less readily available a dynamic priority strategy that Is often used in cloud computing as it allows the processer to be fully utilised for the most processing tasks, this means that upon growth of the system an Earliest Deadline First algorithm may be better suited as this assigns task to the absolute deadline of active jobs.

### Commentary on Key Points in Requirements specification

## References

1. Abramova, V. and Bernardino, J., 2013. NoSQL databases. Proceedings of the International C\* Conference on Computer Science and Software Engineering - C3S2E '13, [online] Available at: <https://dl.acm.org/doi/abs/10.1145/2494444.2494447> [Accessed 14 December 2020].
2. Bhogal, J. and Choksi, I., 2015. Handling Big Data Using NoSQL. *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*, [online] pp.393-398. Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7096207> [Accessed 27 November 2020].
3. Braun, T., Siegel, H., Beck, N., Bölöni, L., Maheswaran, M., Reuther, A., Robertson, J., Theys, M., Yao, B., Hensgen, D. and Freund, R., 2001. A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. Journal of Parallel and Distributed Computing, [online] 61(6), pp.810-837. Available at: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.25.4721&rep=rep1&type=pdf> [Accessed 20 December 2020].
4. Fernandes, D. and Bernardino, J., 2018. Graph Databases Comparison: AllegroGraph, ArangoDB, InfiniteGraph, Neo4J, and OrientDB. Proceedings of the 7th International Conference on Data Science, Technology and Applications, [online] Available at: <https://pdfs.semanticscholar.org/322a/6e1f464330751dea2eb6beecac24466322ad.pdf> [Accessed 14 December 2020]. Johnson, R. and Kent, S., 2007. Designing universal access: web-applications for the elderly and disabled. *Cognition, Technology & Work*, [online] 9(4), pp.209-218. Available at: <https://link.springer.com/article/10.1007/s10111-007-0063-2> [Accessed 28 November 2020].
5. Jing Han, Haihong E, Guan Le and Jian Du, "Survey on NoSQL database," 2011 6th International Conference on Pervasive Computing and Applications, Port Elizabeth, 2011, pp. 363-366, doi: 10.1109/ICPCA.2011.6106531.
6. Li, Y. and Manoharan, S., 2013. A performance comparison of SQL and NoSQL databases. *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, [online] pp.15-19. Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6625441> [Accessed 27 November 2020].
7. Mathew, T., Sekaran, K. and Jose, J., 2014. Study and analysis of various task scheduling algorithms in the cloud computing environment. 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), [online] pp.658-664. Available at: <https://ieeexplore.ieee.org/abstract/document/6968517> [Accessed 3 January 2021].
8. Nagadevi, S., Satyapriya, K. and Malathy, D., 2013. A Survey On EconomicCloud Schedulers For Optimized Task Scheduling. *International Journal of Advanced Engineering Technology*, [online] pp. 58-62. Available at: <http://www.technicaljournalsonline.com/ijeat/VOL%20IV/IJAET%20VOL%20IV%20ISSUE%20I%20JANUARY%20MARCH%202013/IJAET%20IV%20I%20Article%2015.pdf> [Accessed 29 December 2020].
9. Neyak, A., Poriya, A. and Poojary, D., 2013. Type of NOSQL Databases and its Comparison with Relational Databases. *International Journal of Applied Information Systems*, [online] 5(4), pp.16-19. Available at: <https://www.researchgate.net/profile/Dikshay_Poojary/publication/302557703_Article_Type_of_nosql_databases_and_its_comparison_with_relational_databases/links/5aeaa2b50f7e9b837d3c40e7/Article-Type-of-nosql-databases-and-its-comparison-with-relational-databases.pdf> [Accessed 14 December 2020].
10. Parker, Z., Poe, S. and Vrbsky, S., 2013. Comparing NoSQL MongoDB to an SQL DB. *Proceedings of the 51st ACM Southeast Conference on - ACMSE '13*, [online] pp.1-6. Available at: <https://dl.acm.org/doi/pdf/10.1145/2498328.2500047> [Accessed 28 November 2020].
11. Pinedo, M., n.d. *Scheduling*. 5th ed. Springer, pp.1-674.
12. Pokorny, J., 2013. NoSQL databases: a step to database scalability in web environment. International Journal of Web Information Systems, [online] 9(1), pp.69-82. Available at: <https://www.emerald.com/insight/content/doi/10.1108/17440081311316398/full/html?journalCode=ijwis> [Accessed 15 December 2020].
13. Shneiderman, B., Plaisant, C., Cohen, M. and Jacobs, S., n.d. *Designing The User Interface*. 5th ed. Harlow : Pearson Education Limited, pp.1-14.
14. Thatcher, J., 2006. *Web Accessibility*. 1st ed. Berkeley, Calif.: Friends of ED, pp.1-83.
15. Tudorica, B. and Bucur, C., 2011. A comparison between several NoSQL databases with comments and notes. 2011 RoEduNet International Conference 10th Edition: Networking in Education and Research, [online] pp.1-5. Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5993686> [Accessed 15 December 2020].