1 Introduction

The Software Design document is a paper that provides detailed documentation on the design of the proposed prototype, *Document Submission System,* to aid in software development. The system shall support text-based analysis on the submission. The system is able to perform text extraction from the report and thus, generate five MCQ questions for the student to complete the submission. Failure to do so could result in incomplete submission and may be reflected in the submission. Web search against the assignment help portal can be carried out by the system to look for identical submissions. The system is capable of summarizing the report submitted with analysis result that includes reference summary, sentiment analysis, word count and frequency count, etc. The prototype is designed to have a backend MySQL server to support data storing. All application data is to be kept permanently in the persistent storage and the system provides an admin interface for data management.

The main idea to be addressed in this document would be the software architecture adopted in the design and describing how the software should be built. Contained by the document are graphical documentation of the software design of the project including class diagrams, high-level architectural diagram, use case diagrams, collaboration models, software design pattern and other supporting requirement information.

Apart from software design, there is also a research report embedded in this document. Various research is to be carried out by the team to develop certain technology that are requested by the client to satisfy the functional requirements of the system. Relevant technologies include Google NLP (text analysis tool), PDF-based text extraction, sentiment analysis and AWS EC2, S3 and RDS cloud services.

The purpose of this document is to provide a comprehensive description of the system design in a graphical way to deliver an understanding of what is to be built and how it is planned to build. The document ensures that the software developers are on the same page of the software design before proceeding with the implementation and it also serves as a blueprint for communicating ideas. The target readers and audiences of this document are the client (Caslon Chua) and the software developer team.

This software design and research report document serves as a proof of concept for the use of system building that delivers a base level of functionality to determine if the project is feasible to implement these technologies. It is basically focused on the design of critical components of the system. There will be no code implementation covered in this document.

1.1 Overview

This document serves as a purpose to conduct a thorough analysis on the software problems and aims to detail the design without any implementation yet. It is intended for client and developers of the system to communicate ideas within the system context. The proposed solution in this paper describes a functional document submission system with backend server database. The scope of the analysis mainly reviews object-design adopted by the system and incorporates problem analysis, high-level system architecture, design patterns, design heuristics as well as detailed design (class diagram).

1.2 Definitions, Acronyms and Abbreviations

- MySQL Open-source relational database management system.
- PDF Portable Document Format
- EC2 Amazon Elastic Computing Cloud
- AWS Amazon Web Services
- RDS Amazon Relational Database Service
- MCQ Multiple Choice Question
- OOD Object-Oriented Design
- NLP Natural Language Processing
- FAQ Frequently Asked Questions
- AMI Amazon Machine Image
- UML Unified Modelling Language

2 Problem Analysis

This software design document will analyse the requirement of the document submission system stated in the Software Requirement Specification. All the functional and quality requirements will be discussed and analysed to produce design solutions for software. The requirement in SRS reveals all the essential functionalities that the document submission system needs to achieve business goals and every use case scenario.

2.1 System Goals and Objectives

~~Based on~~ the Software Requirements Specification~~, it~~ shows a list of functionalities that the system needs to perform to meet the use cases carried out in the acceptance criteria.

- Accept PDF submission from student
- Perform text extraction from submitted document
- Perform web search against assignment help or code repository website
- Generate MCQ questions from document submitted for student
- Analyze submitted document from student
- Record the information of:
    - The admin / convenor
    - The student
    - The student's document submission
    - The multiple-choice questions generated from the submitted document
    - The student's selected answers for the multiple-choice questions
    - The analysis results
        - Summarized extraction of the document
        - Extracted keywords
        - Web search results from the title of the document (reference summary)
        - Sentiment analysis results
        - Keyword frequency count
- Produce analysis report from document submitted
    - Text analysis report
- Deliver analysis report to the convenor by sending email notification

2.2 Assumptions

A1. Only written documents and reports are submitted (research reports, literature reviews, and other essay-like documents).

A2. Only document related file extensions are submitted (.docx, .doc, .pdf).

A3. Codes, spreadsheets, slides, images, videos, and zipped files are not being submitted.

A4. Each analysis report and the multiple-choice questions generated is for exactly one document, one student and directed to one selected convenor.

A5. All users (students) will have a unique id, name and email address.

A6. A user (student) can only submit 1 document at a time for analysis and multiple-choice question generation.

A7. A user (student) will only be able to answer the MCQ questions after they have submitted their document.

A8. The report analysis will be sent to the convenor only via email notification.

3 High–Level System Architecture and Alternatives

3.1 System Architecture

3.1.1 Component-and-connector view

Figure 1 below presents the high-level design of the document submission system:

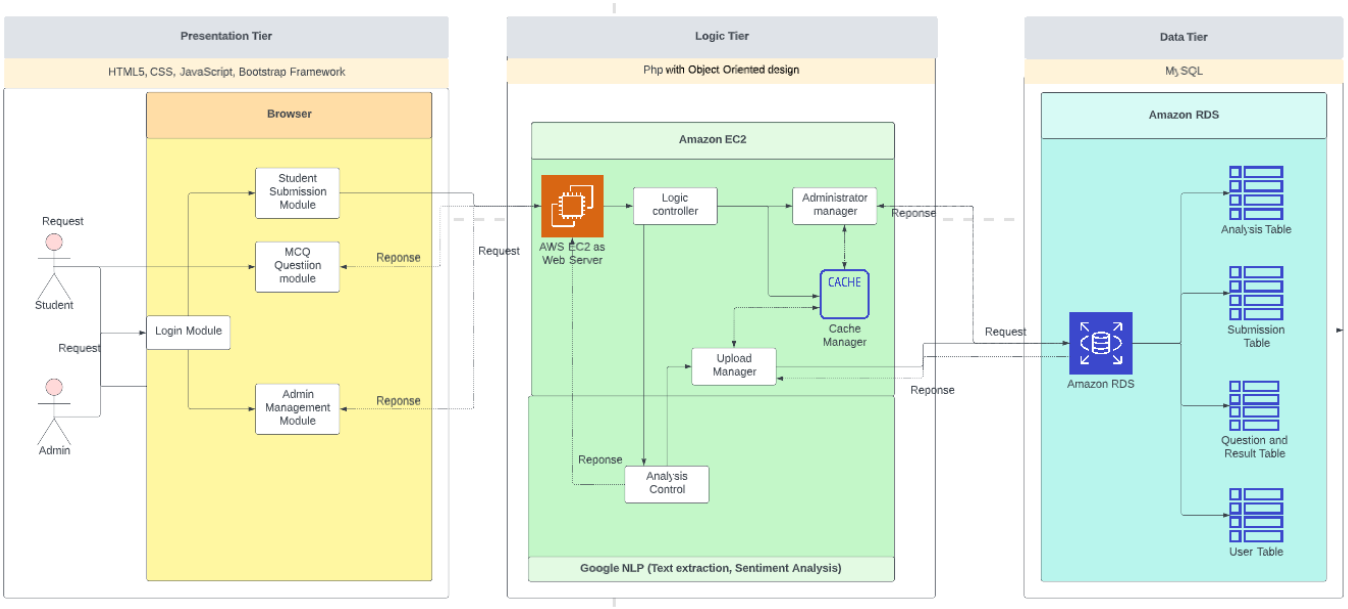


Figure 1 Component-and-connector view of architecture

The design is presented in a component and connector, arranged in a tiered architecture consisting of 3 ~~layers:~~ the **presentation tier** dashboard accessed by the data administrator and students via the login module. It is important to specify that in this instance the unit convenors will have access to submissions like the data administrator as they are both considered as the role of admins in this context; the only difference would be that the convenor will only be able to access all information related to their corresponding unit, whereas the data administrator would have complete access to all records in the database. The dashboard will then allow the students to interact with the login module of the presentation tier, where they would be able to login to the system and submit documents in the student submissions module.

Furthermore, once the student submits the document, the MCQ module will be available for them to answer the questions provided and submit. The admin component will interact such that it can access the submissions in the admin management module, where the records will be available according to the type of admin the individual is (i.e., convenor or data administrator). Inside of the admin management module it would be interactive such that every analysis report and the questions generated for each submission by each student would be available for viewing and changes. The data administrator in the admin component is allowed to access and maintain the data-store, where management access will be provided in the logic layer.

The **logic tier** allows data sent by the presentation layer to be received by the analysis control component via the logic controller component: the logic controller component is responsible for the validation of data submitted in PHP. The administrator manager component allows the data administrator to request access for the maintenance of the database components. The upload manager component has the functionality of ensuring that successful uploading of information is made when analysis control component has conducted analysis and found analytical information that would need to be stored in the database. Finally, the cache manager that is connected to the web server is responsible for caching user data arriving from web server in the instance where the total amount of requests or updates exceed the throughput of the database. All the components detailed will be located within the Amazon EC2 instance.

The analysis control component consists of a machine-learning model with natural language capabilities to extract, conduct writing style analysis and summarize the content in a submitted document, where then onwards it would proceed to generate MCQs for the students to answer and the convenors to check with the utilization of Google NLP combined with other Python-related libraries; this analysed information would be passed to the database located in the data layer. The analysis control component will be the only component to exist outside of the Amazon EC2 instance from the logic tier; instead, the information would be handled via Google NLP.

The **data tier** consists of the database component which is connected to the cache manager, administrator manager and analysis control components. The database consists of connections to the analysis, questions, submit and user tables where queries can be passed from the database component to retrieve data corresponding to each query, which will be presented in a visual manner in the dashboard component for the student and admin components to view and utilize for their tasks. These components also allow the storage of new information pertaining to a new submission.

3.1.2 Deployment View

The figure 2 below is a diagrammatical representation of the document submission system from a deployment perspective:

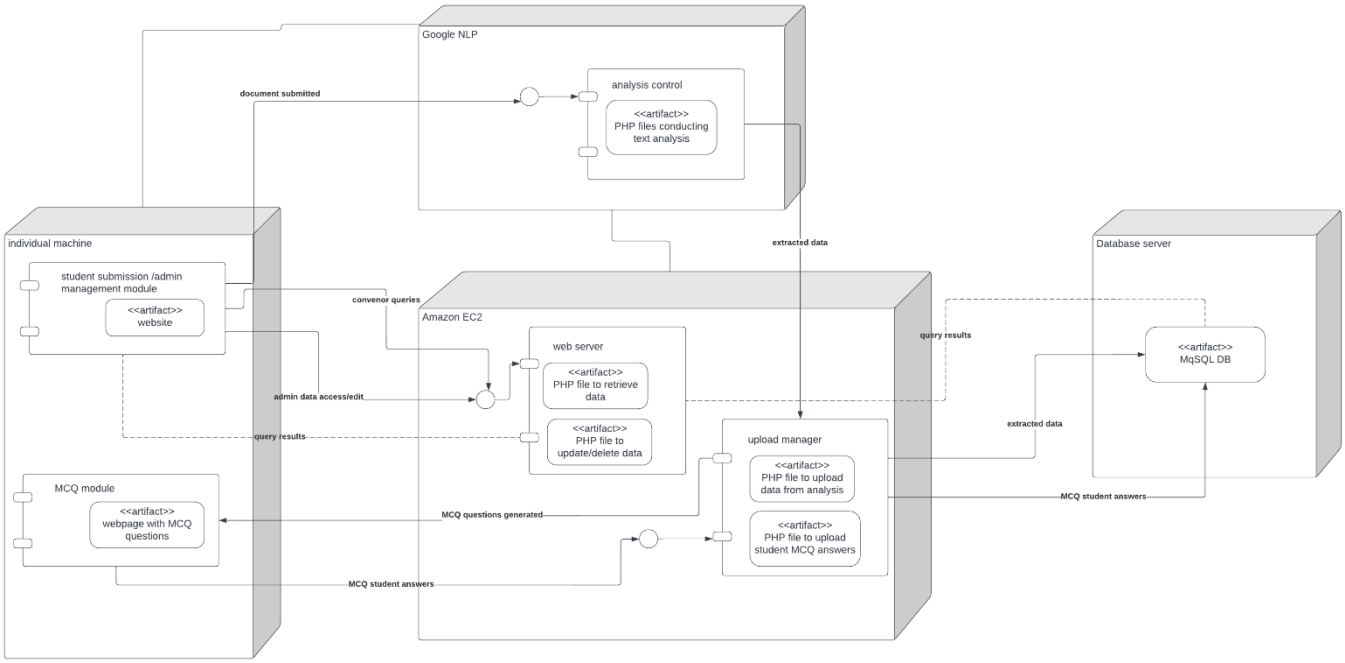


Figure 2 Deployment view of the solution

The student will be able to upload files to the submission system, which will be compiled and submitted via the student submission component that will be home page of the developed website that the student can access with an internet-connected device. There onwards it will be passed to the analysis control component under Google NLP where the document content will be extracted for text analysis. The text analysis will include the following:

1. A reference summary extraction from the document submitted by the student(if a list of references is available)
2. Keywords and key phrases identified in the document
3. MCQs generated with answers
4. Web search results from the title of the document with the result headings and their corresponding links

Once the text analysis is completed, all information will be passed to the upload manager component which will handle uploading all information provided onto their corresponding tables for safe storage. The uploaded manager will also simultaneously pass MCQ questions generated will be presented for the student in the MCQ module for the student to answer, and also store the questions with the actual correct answers. Once the questions are answered by the student, they will submit them again, which will be passed back to the upload manager for storing student answers in the database.

The admin management component will be accessible to convenors and data administrators; the convenor will be able to login to the dashboard and access submissions made by students that are involved with their corresponding unit. This will be done by navigating the website to access student submissions, which will result in a query to be executed that will return all relevant submissions made. By selecting a particular submission, it will trigger another query that would retrieve the relevant information regarding the submission. The retrieved information will then be added to form an analysis report via a PHP file that will be located in the web server. Along with the analysis report, the MCQ questions generated, including the student's answers and the actual answers are displayed in the admin management component.

The data administrator will get access to all data in the database for data to be updated and managed. The data administrator will also be able to access the data management component using an internet connected device. Once logged in, all information will be loaded to the admin interface where the data admin can edit or delete records in the database. This will be done by the corresponding PHP file applicable for the type of user logged in, which in this instance is the data administrator. The PHP file artifact will be located in the web server component of the Amazon EC2 instance.

3.2 Other Alternative Architectures Explored

3.2.1 Peer-to-Peer Architecture

The process below shows a high-level view of the document submission system arranged in a Peer-to-peer Architecture.

In the document submission system when a student logs into the system. The student establishes a connection to a workstation (node). The workstation will be responsible for collecting the documents and the MCQ answers from the student, send the generated questions to the student, and sends them all to the convenor (another node) via email. The requests that perform the actions mentioned above will be directly connected between the student's computer and the workstation.

However, the Peer-to-peer Architecture uses different computer system to store data. If many students submit their documents, then the storage and processing power needs to be upgraded more frequently when compared to having a centralised server. The student might also need to take longer time to send the documents and the MCQ answers, as well as receiving the MCQ questions as the connection bandwidth is limited on both ends of the node.

3.2.2 Event Driven Architecture

The process of the document submission system can be explained using the event driven architecture as follows.

In the document submission system when a student accesses the login to the system, a login event is published. Then the document submission system receives this event and accepts the login, if the credentials match. The student then makes a submission and uploads a document, which will publish a new upload event. A submission event is published when the student clicks on the 'submit' button after he/she

finishes uploading the document/s. The student will then be required to answer the MCQ questions generated, which the system will publish an answer event. An event is triggered when the student answers each of five questions. Then 'submit' will complete submission process. After the student 'submit' the MCQ questions, they are sent to the convenor via email. Another email event is published in that instance.

However, development of the system from an event driven architecture is much more complex as event driven architectures backend does all the work to deliver the end user experience. Duplicates events may also occur in event driven environment, and this will result in increased work for error handling and trouble-shooting these errors.

4 Detailed Design (using Object Orientation or alternative)

4.1 The Detailed Design and Justification

The proposed design method of *Document Submission System* is **Object-Oriented Design (OOD)** which takes a **responsibility-driven** approach. The rationale behind the adoption of OOD in the system design is due to the strong encapsulation supported by OOD, thereby enhancing the reusability, maintainability, testability, and expendability of the software (Wirfs-Brock, R. and Wilkerson, B., 1989). Improving these software metrics can be achieved by managing the complexity of the software effectively. OOD method separates or decomposes the system based on the objects of the system to increase the modularity of the system and thus achieve *Weak Coupling and Strong Cohesion*. Each object in the system will follow the approach of **Responsibility-Driven Design (RDD)**. The main idea of this design focuses on assigning a certain role to each object in the system to evenly distribute the system intelligence. What are the responsibilities of this object? What information does this object share and who are the collaborators? (Wirfs-Brock, R. and Wilkerson, B., 1989) Behaviours are kept together with any relevant data in the individual object.

The system objectives outlined in 2.1 reveals a list of candidate classes that may be included into the system to perform those operations. Below shows all the required candidate classes:

- User – Parent class
- Unit
- Student – Child class
- StudentTable
- Admin – Child class
- Convenor – Child class
- Submission
- SubmissionTable
- Question
- QuestionTable
- Question List
- Analysis
- AnalysisTable
- Sentiment Analysis
- Database

All the candidate classes above will be tied with at least one collaborator and assigned with certain role to attain the system objectives. Overall, the system is designed to have a set of these interacting objects, each with at least one role. The high-level view of the object design is illustrated via a UML Class Diagram below.

Data storing will be managed by a database in the backend cloud server. Each of the classes act as a software component which encompasses a collection of data and relevant responsibilities, and they are tied together

in an inter-dependency relationship to form the system. The classes will be the interface between the application and the database which governs the data flow.

In terms of the database schema, it is designed to normalize the database tables to third normal form without any data duplication, transitive dependency, and data anomalies. This is to simplify the data management and ensure referential integrity.

4.1.1 UML Class Diagram

Figure 3 below represents the UML class diagram for the document submission system to be implemented by group 28:

Figure 3 UML diagram for document submission system

4.1.2 Design Patterns

Model-View-Controller Pattern

MVC Design Pattern is applied in the development of Document Submission System. It is widely adopted in most of the software implementation as it practices the goals of '***separation of concerns***' and is very good in concern separation for user interaction. Since the system itself is performing a high abundance of data and handling lots of object interactions, MVC aids in separating user interaction from data processing and enables them to behave independently (Curry, E. and Grace, P., 2008). In other words, MVC decomposes the system intelligence and ease the complexity management which is beneficial to our system. The MVC proposes an effective method for viewing and varying data. It allows a model to have several views and controllers, which can be constructed and reworked independently of the model (Curry, E. and Grace, P., 2008).

There are three different components produced by decoupling data-processing logic, data access, data presentation and user interaction tasks to achieve independency. **Model**, **View** and **Controller**. Table 1 shows their individual responsibility and rule. Figure 4 illustrates the relationship between each MVC component.

Model	Model objects are responsible for storing, encapsulating and abstracting the data. There should not be any methods or application logic.
View	View components controls how the information from the model object is displayed to the user. View class design goes from the very basic to the very precise; generic view classes are supplied by the framework to present most kind of string, number, or image, while you are expected to implement very specific view objects designed for the application. Furthermore, view components also interpret and respond to user-initiated events including mouse click and keystrokes. Subsequent actions are instantiated in respond to the events that are passed to the controller object for execution. The event and resulting action are often very simple; clicking the mouse over a button object will send an action message to a controller.
Controller	Controller objects interacts and handles application actions from users. It merely serves as an intermediary between Model and View to facilitate the communication between them. Actions are usually invoked by view objects in response to user events.

Table 1: Model-View-Controller Components (Curry, E. and Grace, P., 2008)

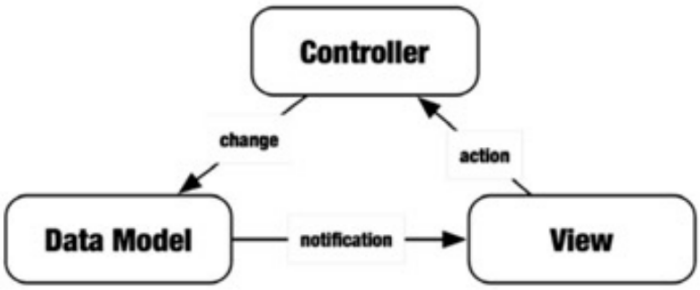


Figure 4: MVC Component Relationship (Bucanek, J n.d.)

Advantages of MVC Pattern in Software Quality

Modularity - The MVC Design Pattern is a good practice of computer science principles, separation of concern, and encapsulation. Applying the pattern enables the developers to identify the functional requirements of the document submission system and convert them into roles and compartmentalize those roles into distinct objects (Bucanek, J n.d.).

Interconnected functionalities like submitting assignment, answering questions are localized into containers with particular boundaries. When the interface to the class is well specified, it is less difficult to identify errors or analyse how the classes are affected when there are changes in the classes (Bucanek, J n.d.). Separating these components allows future modifications like replacing, subclassing and reusing classes without any impact on the other design components.

Flexibility - Model-View-Controller design pattern offers high flexibility. This is because MVC supports the features of effortlessly replacing view objects or use several view objects without worrying the cause of impact to the controller or data model. Any object (data model, controller, or view) is compatible with any functionally same object (Bucanek, J n.d.). Exchanging objects, or affixing multiple objects, does not disrupt the rest of the design. The mantra of MVC is to construct complicated applications by interlinking simple objects.

For instance, in the view object of QuestionList class, we can easily define multiple views of displaying the questions. The system is able to present the questions in different ways but yet still maintain the integrity of the data model.

Reuse - Reuse has a close association to flexibility. For instance, extracting the common abstractions of the classes allows the team to reuse the objects in other applications (if there is any) for some purposes. View and model components are the two most commonly reused objects (Bucanek, J n.d.).

Scalability - The Model-View-Controller design pattern has the effect of low coupling. MCV framework introduces low coupling among models, views or controllers which ease the modification. As MVC pattern separates the responsibilities, it is less complex to maintain for future development and modification. Thus, scalability increases.

Observer Behavioural Patterns

The observer design pattern has been considered and applied in our document submission system project. The observer, also known as the event subscriber or listener, is responsible for listening and waiting for updated data. If the data changes, the observers will notify immediately. This pattern suggests adding a subscription mechanism that observers have the choice to either subscribe or unsubscribe to from a stream of the event (Refactoring.guru 2022).

The advantages of an observer design pattern are providing a loose coupling between objects and provide scalability. The observers can be added or removed without affecting other objects. Below is the design of observer patterns implemented in our UML class diagram:

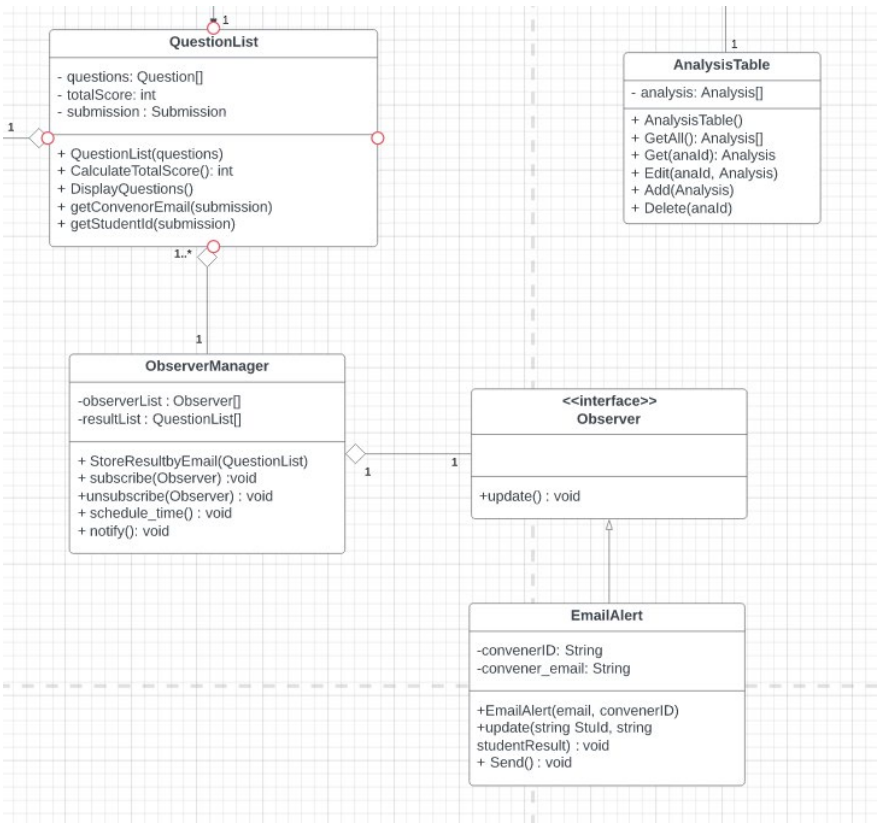


Figure 5 Observer patterns implemented

The ObserverManager will act as the **publisher of the student MCQ result to the convener**. The observer will serve as the base class that can derive types of observers. The derived class of observers will act as **subscribers of the observerManager (publisher).** The observerManager (publisher) can add many kinds of subscribe sub-class under the observer base class, such as phone calls and SMS. It provides scalability to the system.

ObeserverManager pseudocode:

```
//pseudocode
public class ObserverManager(){
    //store list of observer/convener that wants to receive email
    List<Observer> observerList = new List<Observer>();
    List<QuestionList> resultList =new List<QuestionList>();

    //add and remove the Observer(for now, is the convener that wants/do not want
to receive schedule email)
    public void subscribe(observer o){
        emailList.add(o);
    };
    public void unsubscribe(observer o){
        emailList.remove(o);
    };

    public void schedule_time(){
        DateTime nowTime = DateTime.Now;
        DateTime scheduledTime = new DateTime(nowTime.Year, nowTime.Month,
nowTime.Day, 9, 0, 0, 0); //every day 9am
        if (nowTime >= scheduledTime)
        {
            nofity();
        }
    };

    public void notify(){
        for each (observer o in observerList){
            for each (QuestionList r in resultList){
                if(o.convener_email =
resultList[r].QuestionList.submission.convener_email){
                    o.update(string resultList[r].QuestionList.submission.stuId,
string resultList[r].QuestionList.totalScore);
                }
            }
            }
        }
    };
};
};
```

Figure 6 observerManager pseudocode

The observerManager(Publisher) will have a list of observers (subscribers) that want to receive the student result. Whenever there is a new submission from the student, the observerManager will keep the student MCQ result in a list. An email will be composed by the end of the day (24 hours), with the student Id, student MCQ result send to convenor email (observer that have subscribed).

Observer and its sub-classes pseudocode:

```
//pseudocode
//base class
public abstract class Observer(){
    public abstract void update(){};
};

//derived class
public class Observer: EmailAlert
{
    private string email;
    private string convenerID;

    //constructor for creating convener that wants to receive schedule email
    public EmailAlert(email,convenerID){
        this.email = email;
        this.convenerID = convenerID;
    }

    //perform email sending function
    public override void update(string StuId, string studentResult){

        Console.WriteLine("Below is the schedule email for submission of student
with their result");
        //populate the email format and send
    };

};
```

Figure 7 observer and the pseudocode of its sub classes

When the **update** has been sent to the observer object, the update function will send an email query to the convenor with a list student Id and student result. Convenor can log in to the submission system web page to have a completely view of the student result report.

4.2 Design Verification

The proposed design solution is objectively verified by the following five non-trivial business use scenarios. Each of the scenario is illustrated using UML Sequence Diagram. In this section, these scenarios will demonstrate how the proposed solution handles them.

4.2.1 Student Login

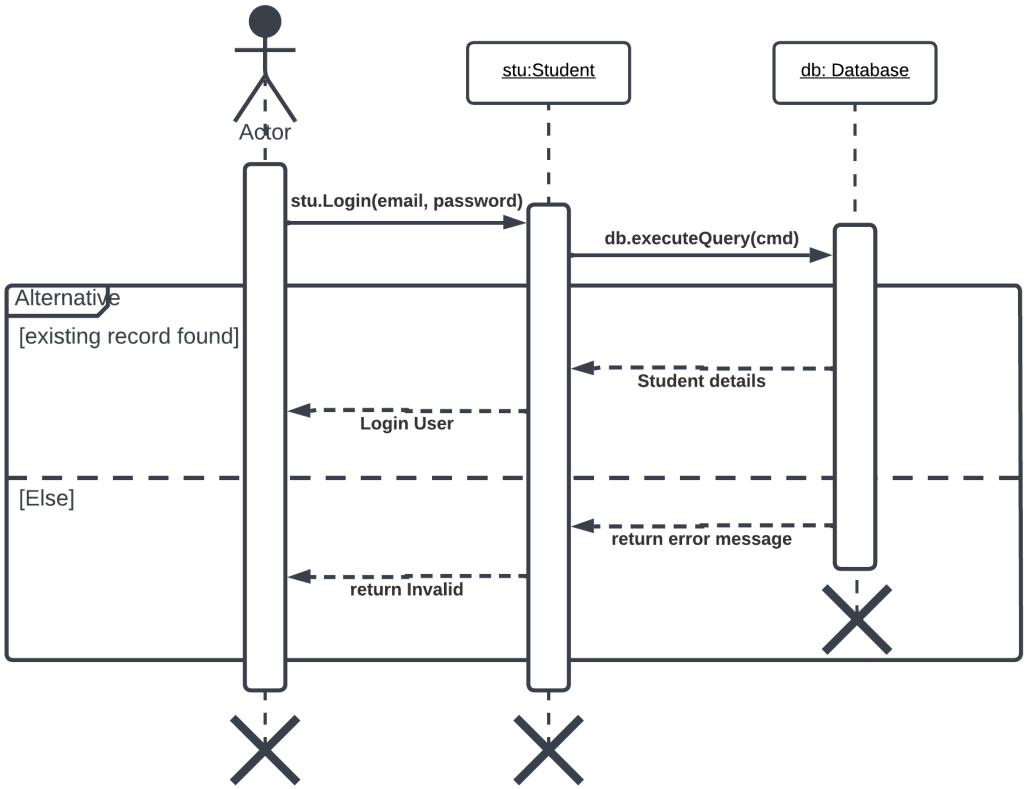


Figure 8 Sequence diagram for student login

After the user has entered the login credentials (email and password), the user will click on the Login button which invokes the Login method encapsulated in the Student class. The student class then calls the Database class to execute the search query to search for the existing record of the student. Invalid login credentials will be returned to user if no existing record is found. Otherwise, the system will grant access to the student to access the system.

4.2.2 Student Submits an Assignment

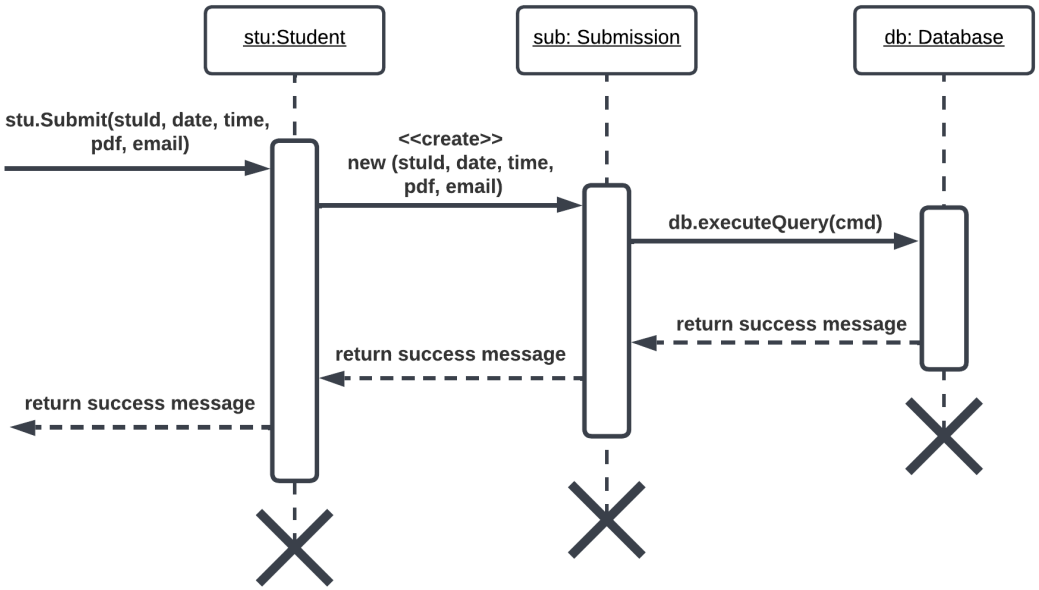
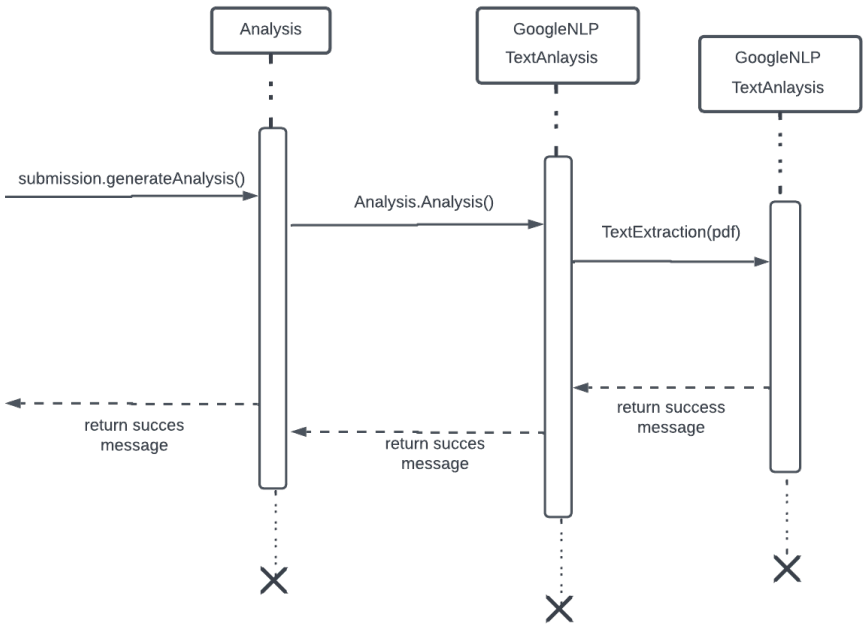


Figure 9 Sequence diagram for document submission

The caller calls the Student class to carry out the Submit method and passes in certain arguments which include student Id, pdf file and current date and time. A new submission object is created to store these data. The database class is then invoked to execute the insert query which stores the data from the submission object to the database. A return message will be returned by the database to the submission which passed it to the student.

4.2.3 Extract Text from PDF file



Once the document has been submitted for analysis, an Analysis object will be created through generateAnalysis(). This method will analysis the text through methods in Google NLP; Natural Language Processing. Once the text analysis has been completed, it will extract the keywords or phrases and it will be deemed a success if a “return success message” has appeared.

4.2.4 Report Analysis

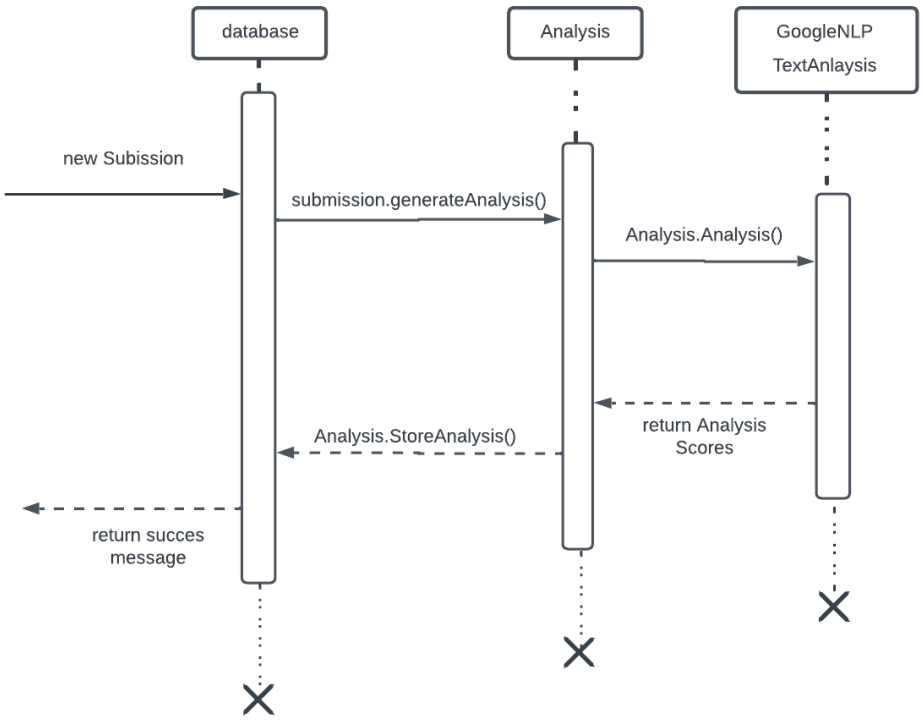


Figure 10 Sequence diagram for conducting report analysis

After a new submission is made by the student, the submission is stored in the database. Then the method generateAnalysis() is called to create a new Analysis object. The method Analysis is called to carry out the text analysis which is done using the methods in Google NLP. Several Analyses like syntax Analysis, Sentiment Analysis, Entity Analysis, Entity Sentiment Analysis and text Classification are carried out in this process. After the analysis is completed, Google NLP will return the scores of the Analyses performed. Then the method storeAnalysis() is called to store the scores in the database. After the scores are stored successfully, a success message will be returned.

4.2.5 Admin view Submissions from Database

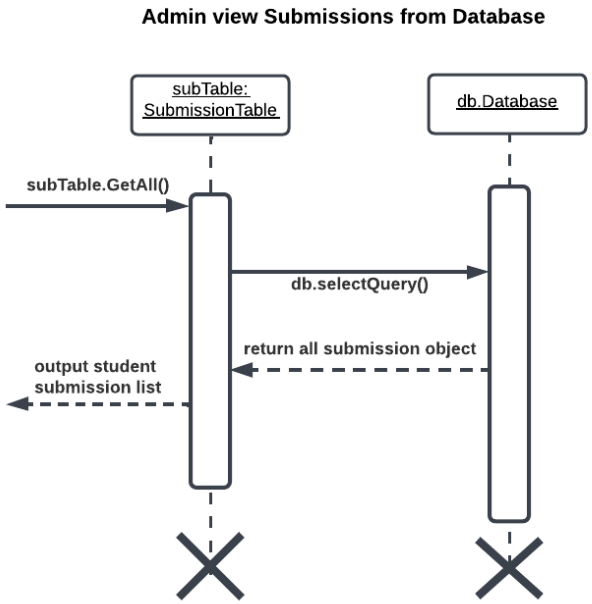


Figure 11 Sequence diagram for admin viewing submissions

When admin successfully login to document submission system webpage, they will be directed to students' submission page for checking total submission of students. The submission table object will perform GetAll() function to retrieve all students submission entries in the database to user interface as a list of table entries.

Besides, the filter for particular student Id and date will be done by front-end filter function, controlled by HTML. The student Id filter will get admin input for particular student Id whereas date filter will accept data input from admin. Admin have the freedom to provide input in both student Id and date filter. Below are the 4 scenario that will be examine by system:

- Admin provides Student Id and particular date – Filter by **certain** student submission in particular date
- Admin provides Student Id but did not provide particular date - Filter by **certain** student submission in **ALL** date
- Admin did not provide Student Id but provide particular date - Filter by **ALL** student submission in **particular** date
- Admin did not provide Student Id and particular date - Filter by **ALL** student submission in **ALL** date

5 Research and Investigations

5.1 Amazon Web Services

Document Submission System uses the cloud-computing Infrastructure-as-a-Service (IaaS) offered by AWS to host the application over the Internet. It serves as a platform built for accessing computing resources. These cloud applications comprise of huge data centres with powerful servers that host the web services and applications in the cloud. This amazing software platform offers convenience to the application user because anyone can access to the application as long as there is stable connection established and standard browser. This section discusses and reviews EC2, S3 and RDS.

5.1.1 Elastic Compute Cloud (EC2)

Amazon Elastic Compute Cloud refers to one of the Amazon Web Services that supports resizable compute capacity in the cloud which enables developers to effortlessly web-scale the capacity of computing to suit the surfing demand. With the help of the AWS EC2 interface, the developers can adjust and configure the computing capacity with minimum friction. From the perspective of a software developer, there are various benefits offered by EC2 (Kulkarni, G., Sutar, R. and Gambhir, J., 2012).

- Lessens the time needed to boot new server instances in minutes
- Scaling capacity can be accomplished very quickly either vertically or horizontally
- Customize the capacity according to the requirements by altering the computing economics. The software developer only pays for capacity that is actually used.
- Developers are offered various necessary tools to develop failure resilient applications. These applications are then separated from common failure scenarios (Kulkarni, G., Sutar, R. and Gambhir, J., 2012)

Amazon EC2 offers a true computing virtual environment which enables developers to launch instances using AWS interfaces in a variety of operating systems such as Amazon Linux, Ubuntu, Windows Server, Red Hat Enterprise Linux, SUSE Linux Enterprise Server, Fedora, Debian, CentOS, Gentoo Linux, Oracle Linux, and FreeBSD. The launched instances can be loaded with custom application environment and configured network's access permissions. Before utilizing AWS EC2 services, there are a list of procedures needed to be followed and satisfied:

1. Choose a pre-configured, ***launch template image*** to be created and run instantly. For example, a launch template can contain the AMI (Amazon Machine Image) ID, instance type, and network settings that you typically use to launch instances. The AMI comprises of the applications, libraries, data and other connected configuration settings.

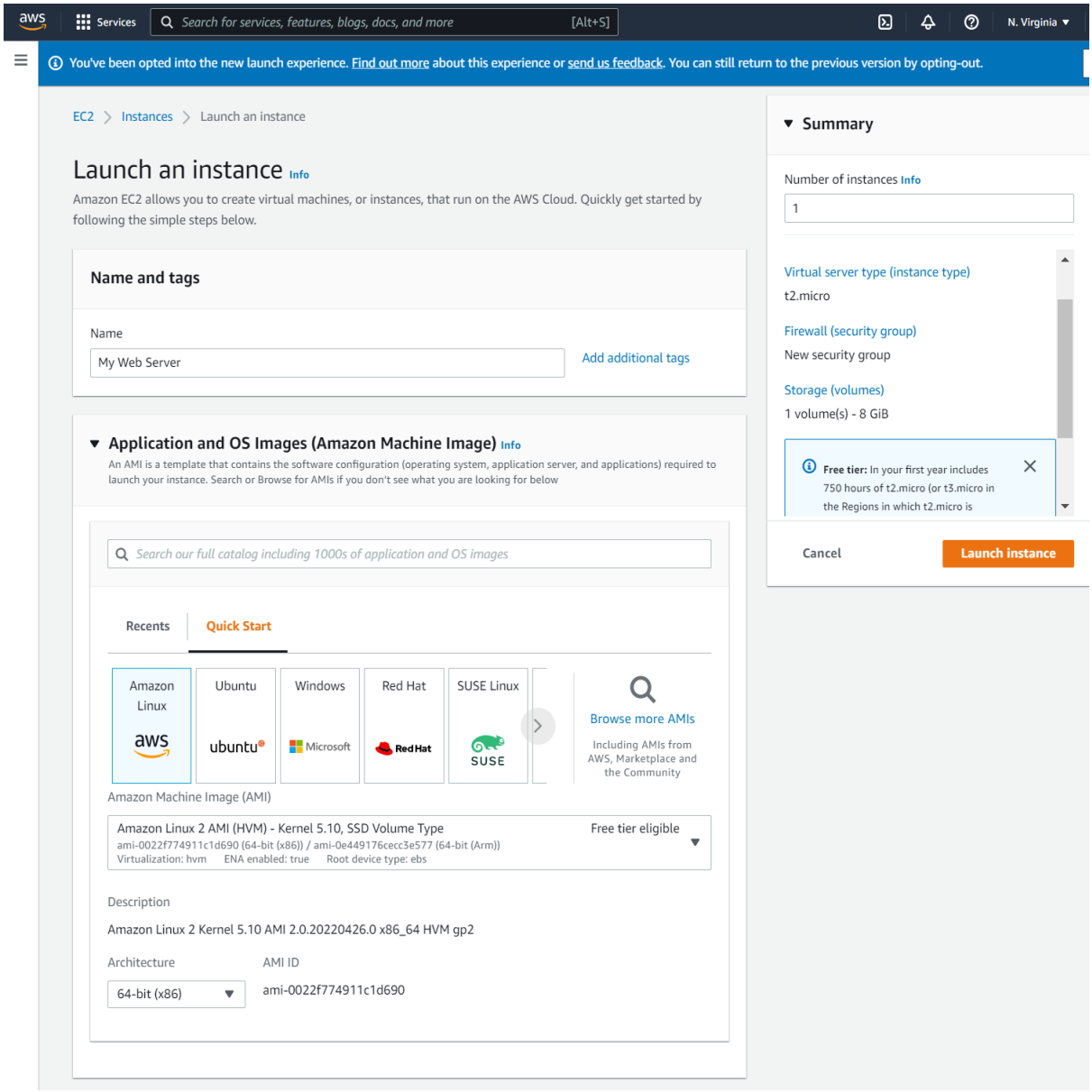


Figure 12 Interface for launching an Amazon EC2 instance

2. Configure and setup the ***security and network access*** on the Amazon EC2 instance.

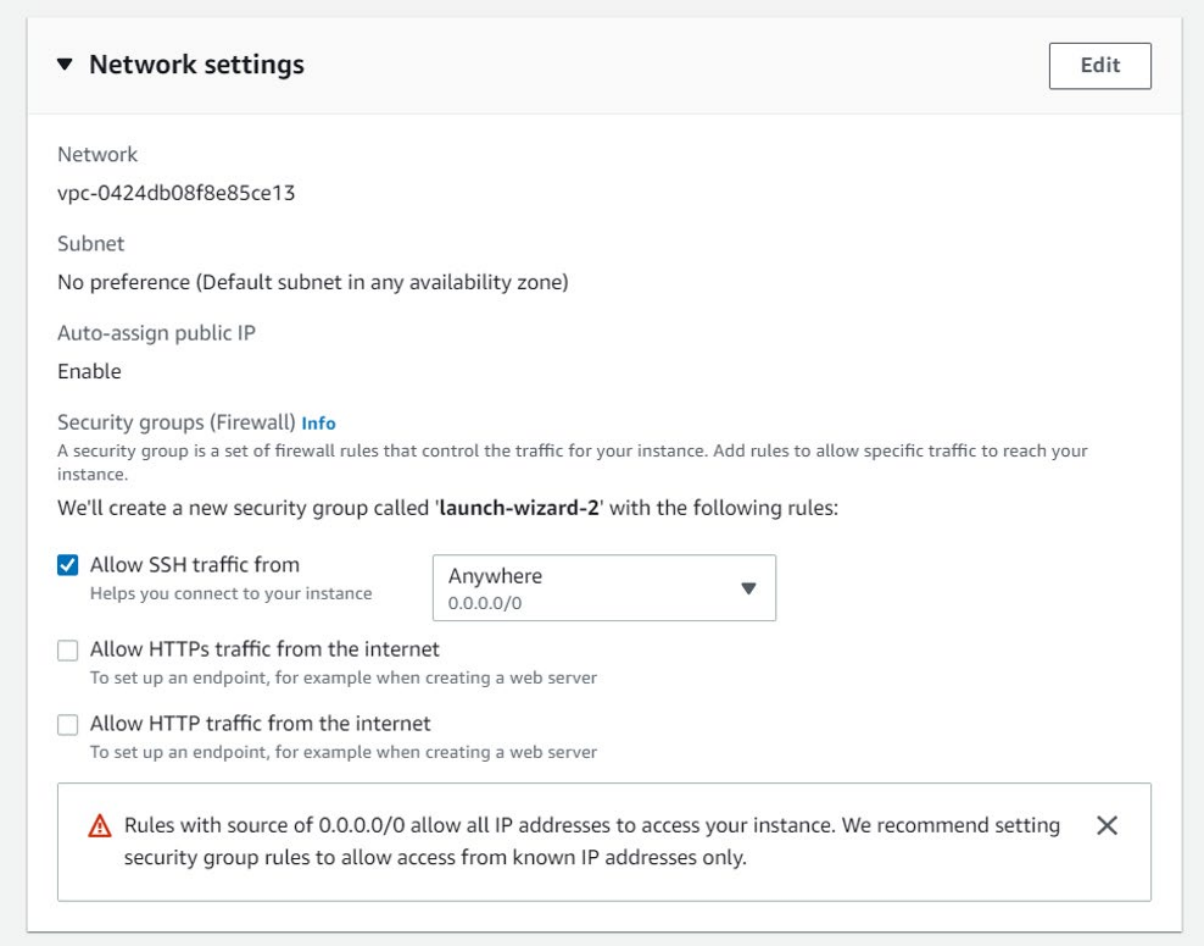


Figure 13 Configuration for Amazon EC2 instance

3. Select the type of instance, operating system needed. Start, terminate and monitor the launched instances needed with the help of web service APIs or other offered management tools.

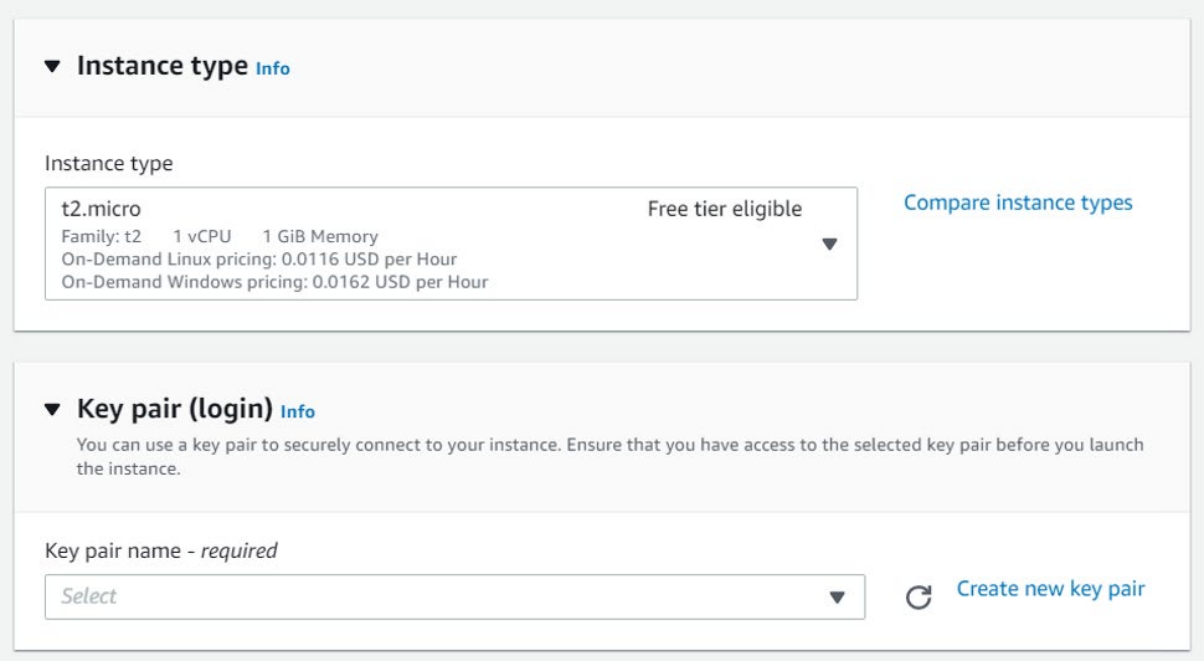


Figure 14 Instance type selection for Amazon EC2

4. Ascertain if the instance is intended to run in various locations by utilizing static IP endpoints or attach permanent block storage to the instances.
5. Pay for the resources that is actually consumed. Calculations are based on ***instance-hours*** or data transfer (Kulkarni, G., Sutar, R. and Gambhir, J., 2012).

5.1.2 Amazon S3 Storage Model

Amazon's Simple Storage Service (S3) provide cloud storage to store data which named as 'objects' and grouped in a container called 'buckets'. **This is explicitly beneficial to our prototype as it stores the project folder which are to be loaded by the EC2 instance later.**

To achieve the storage model, buckets have to be created initially prior to using it, each AWS user is able to create up to 100 buckets. The bucket names are global and only uniquely identified name is allowed in AWS to avoid name duplication. Every bucket can be configured to control the network access such as read/write permission (Garfinkel, S., 2007). Amazon states that S3 storage is designed to store huge objects and several tests have been conducted to verify that S3 storage offers high throughput dramatically on large objects than small objects due to high overhead per transaction (Garfinkel, S., 2007).

5.1.3 Amazon Pricing

The pricing for AWS EC2 is set at 10 cents/hour for each instance using rounded up fractional hours. Each running instance must be shut down with ***ec2-terminate-instances*** command to avoid additional charges.

Crashed instance or instance that haven't reboot spontaneously will keep on acquiring charges (Garfinkel, S., 2007).

AWS S3 Storage is charged based on the size of data storage. It is charged on a flat basis of 15 cents/GB stored for one month. The data size will be worked out twice on a daily basis. Amazon has made a decision that starting from 1st of June 2007, 1 cent for each transaction fee will be charged for every 1000 PUT requests and 1 cent for every 10,000 GET requests (Garfinkel, S., 2007). Deleting requests do not incur any charges.

5.1.4 Relational Database System (RDS)

Amazon Relational Database Service (Amazon RDS) is a web service that helps users establish and run a relational database in the cloud. It delivers scalable and cost-effective storage for users while handling database management responsibilities (Jinesh. V, Sajee.M, 2014). It allows users to concentrate on the application and free up the stress of managing the database. Amazon RDS will automatically help users to secure the database including update patches and backup database.
In our project, we will use AWS Academy learner lab provided by Swinburne university to launch our Amazon RDS service. Below is the list of procedure to create a database (DB) instance:

1. Sign into AWS Academy Learner Lab and direct to AWS management console
2. Choose the Amazon RDS under AWS service panel
3. Choose the Database option to create (Our project will use MySQL as database engine)

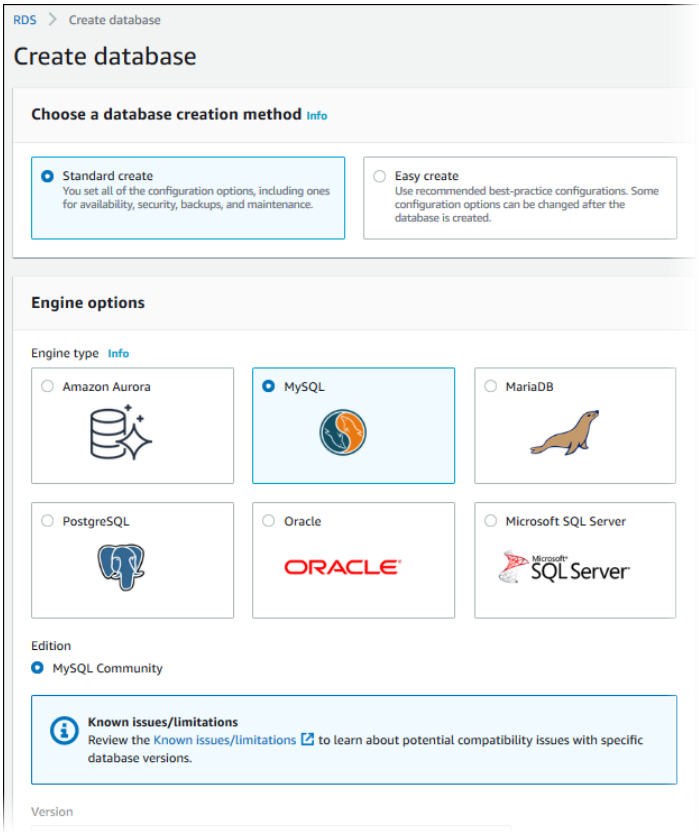


Figure 15 Create database feature in Amazon EC2

Example of create database page (Amazon Web Service, 2022)

4. Choose the 'Free Tier' for database template
5. In setting section, set the values based on project needs

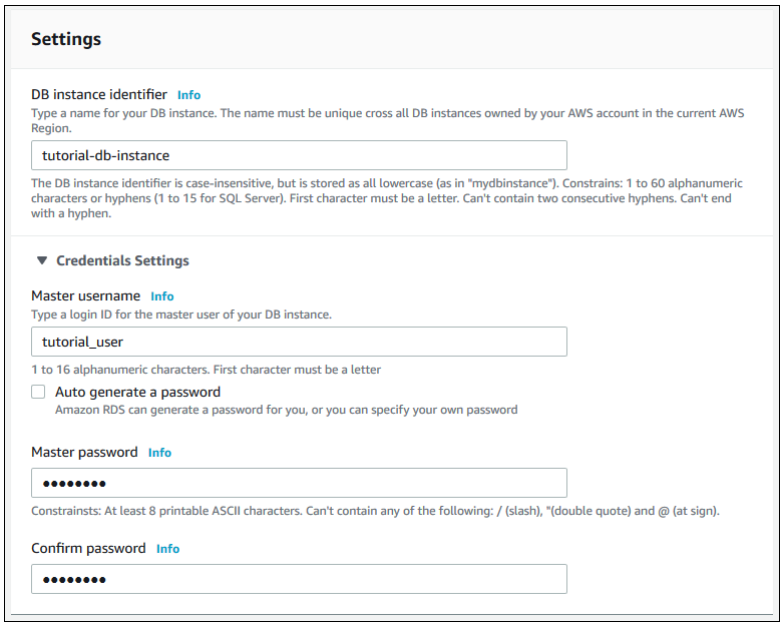


Figure 16 Example of setting option in create database page (Amazon Web Service, 2022)

   a. DB instance identifier is the unique name for database
   b. Master username is the admin name for managing this database

6. After finishing the settings, user will have to choose DB instance size. Our project will adopt **db.t2.micro** instance size which have 1vCPU and 1 GiB RAM, to coordinate to our project needs.

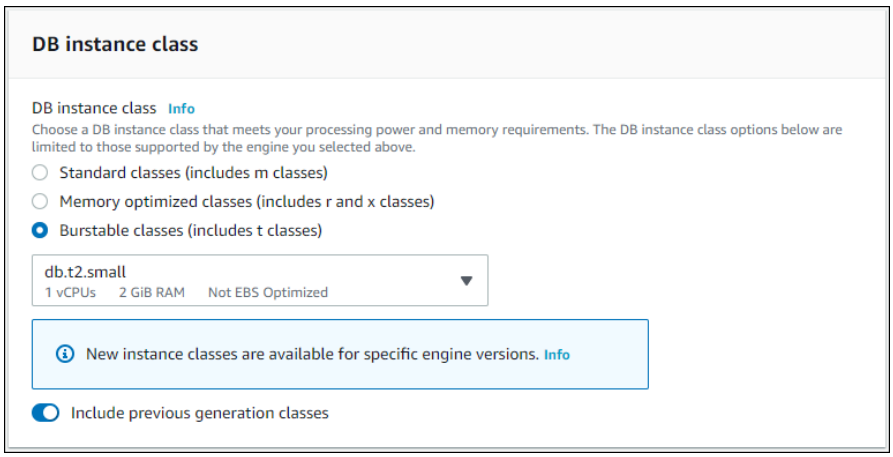


Figure 17 Example of DB instance size in create database page (Amazon Web Service, 2022)

7. Use the default value set in this Storage and Availability & durability sections
8. Heading to connectivity section, set the network value based on the project.

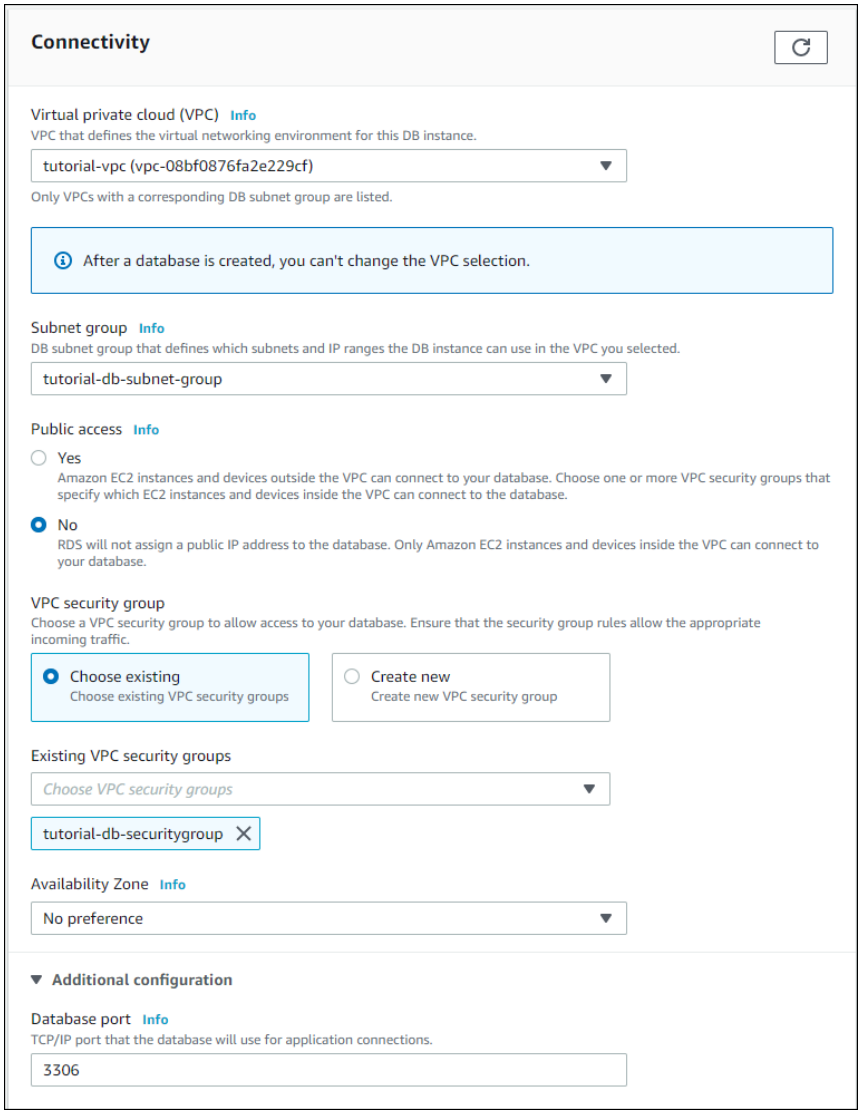


Figure 18 Example of connectivity setting in create database page (Amazon Web Service, 2022)

a. Virtual Private Cloud (VPC) – choose the default VPC/ VPC that are same as EC2 instance
    i. Additional Connectivity configuration
        1. Subnet group – Choose default
        2. Public accessible – Choose **No** for not assign public IP address to RDS instance. Only EC2 instance or device inside the VPC can connect to database.
    ii. VPC security group – Choose the existing security group that have created in EC2 instance.
    iii. Availability Zone – Choose No preference
    iv. Database Port – Make sure use the default database port value 3306

9. Go to Database authentication after finishing the connectivity setting.
10. Enable Password authentication in database.
11. Keep all the setup by default in Additional configuration section.
12. Review all the setting and click Create Database once complete.

13. After successfully setup the database, view and copy the Endpoint and Port of the DB instance (both information is for connecting web server EC2 to database)

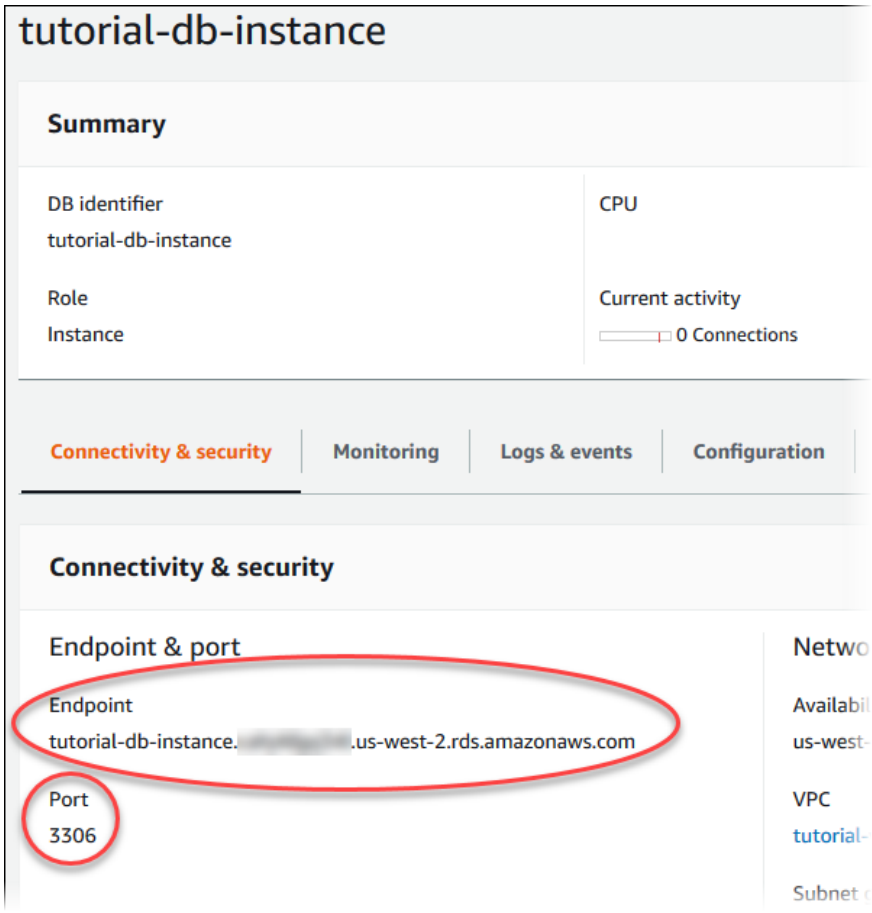


Figure 19 Example of Endpoint and Port information (Amazon Web Service, 2022)

14. Go back to EC2 instance and set Inbound rules for database port (3306)

5.2 Text Analysis Tool

5.2.1 Generate Question

The relevance of question generation to this context is a software requirement where the software solution is expected to generate five MCQ questions including answers from the text extracted. The generation of questions and answers is expected to be conducted using NLP. Although the main text analysis tool used is Google NLP, the research conducted concluded that there is no efficient way to generate questions with answers. Therefore, another library or tool would have to be utilized in order to generate questions and answers.

According to the extended research conducted, a library named "Questgen.ai" which can generate questions via NLP algorithms (Kandi X-RAY Questgen.ai Summary, 2022). It currently supports the following types of questions and capabilities:

1. MCQs
2. Boolean questions ("Yes" or "no")
3. Question answering
4. Question paraphrasing
5. General FAQs

However, for our software requirement fulfilment, point number 1 will only be used; prior to question generation, the "Sense2vec" Python package would need to be downloaded and extracted to generate the MCQs. Once this is accomplished, the MCQs can be generated in Python using the Questgen.ai library as follows:

```
qg = main.QGen()
output = qg.predict_mcq(payload)
pprint (output)
```

Figure 20 Question generation using Questgen.ai library

The payload in this context would be the text extracted from the document where it will be passed to "predict_mcq" method to generate the MCQs. To be more precise, figure below shows an example payload which can be run be used as an argument for the question prediction method:

```
payload = {
        "input_text": "Sachin Ramesh Tendulkar is a former
international cricketer from India and a former captain of
the Indian national team. He is widely regarded as one of
the greatest batsmen in the history of cricket. He is the
highest run scorer of all time in International cricket."
    }
```

Figure 21 Example payload (Goutham Golla, 2022)

An example of the output with the above payload is presented below in figure 22:

```
{'questions': [{'answer': 'cricketer',
                'context': 'Sachin Ramesh Tendulkar is a former international '
                           'cricketer from India and a former captain of the '
                           'Indian national team.',
                'extra_options': ['Mark Waugh',
                                  'Sharma',
                                  'Ricky Ponting',
                                  'Afridi',
                                  'Kohli',
                                  'Dhoni'],
                'id': 1,
                'options': ['Brett Lee', 'Footballer', 'International Cricket'],
                'options_algorithm': 'sense2vec',
                'question_statement': "What is Sachin Ramesh Tendulkar's "
                                      'career?',
                'question_type': 'MCQ'},
               {'answer': 'india',
                'context': 'Sachin Ramesh Tendulkar is a former international '
                           'cricketer from India and a former captain of the '
                           'Indian national team.',
                'extra_options': ['Pakistan',
                                  'South Korea',
                                  'Nepal',
                                  'Philippines',
                                  'Zimbabwe'],
                'id': 2,
                'options': ['Bangladesh', 'Indonesia', 'China'],
                'options_algorithm': 'sense2vec',
                'question_statement': 'Where is Sachin Ramesh Tendulkar from?',
                'question_type': 'MCQ'},
               {'answer': 'batsmen',
                'context': 'He is widely regarded as one of the greatest '
                           'batsmen in the history of cricket.',
                'extra_options': ['Ashwin', 'Dhoni', 'Afridi', 'Death Overs'],
                'id': 3,
                'options': ['Bowlers', 'Wickets', 'Mccullum'],
                'options_algorithm': 'sense2vec',
                'question_statement': 'What is the best cricketer?',
                'question_type': 'MCQ'}]}
```

Figure 22 Example output based on the example payload (Goutham Golla, 2022))

According to figure 3, there are 3 MCQs generated from the given input payload, which would be the text extracted from the diagram. For each MCQ generated, four elements would be presented: the answer

element shows the answer to the MCQ, the context shows the statement from where the question was derived from, the options show the other options for the user to select that are generated, the options algorithm shows the algorithm used to generate the other options and finally the id element gives the question number. If these results are generated successfully, these elements can be used to present the MCQ questions to the student for them to answer and submit via the dashboard component.

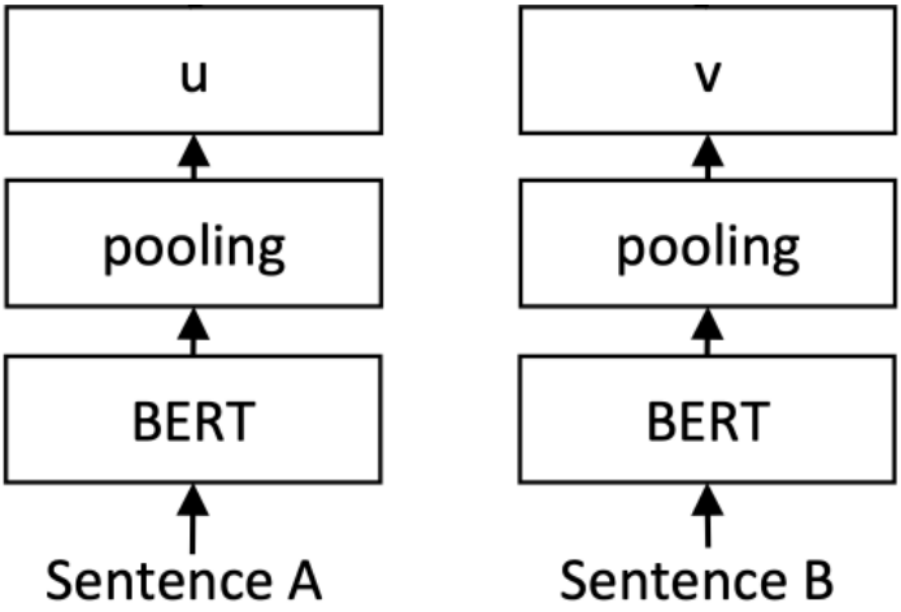
5.2.2 Extract sentence/Keyword

Keyword extraction is an automated method which extracts key information from pieces of text by looking for certain words or phrases.

There are various methods to extracting keywords which are:

- **KeyBert –** A relatively simple "keyword extraction method" which takes advantage of the "SBERT embeddings" to find keywords and phrases from a document and comparing it to another similar document.

  Essentially, how it works is that it will first embed a document generated using the sentence-BERT model. Afterwards, the embedded words will be extracted for N-gram phrases, a continuous sequence of words within a document.



Example of sentence-BERT Model (Viktor Karlsson 2020)

To install KeyBert, users must use the example below.

```
pip install keybert
from keybert import KeyBERT
```

Figure example of KeyBert (Ali Mansour, 2022))

After installing KeyBert, write down the functionality for extracting keywords using several parameters such as the text, number of words or phrases, number of keywords to be found and highlighting the keyword as shown the figure below.

```
keywords = kw_model.extract_keywords(full_text,
                                     keyphrase_ngram_range=(1, 3),
                                     stop_words='english',
                                     highlight=False,
                                     top_n=10)
keywords_list= list(dict(keywords).keys())
print(keywords_list)
```

Figure example of KeyBert (Ali Mansour, 2022))

Once it has been completed, KeyBert will show the N-gram range results as shown below.

```
['clustering word vectors',
 'text vectorization methods',
 'text mining',
 'concepts clustering word',
 'text mining tasks',
 'new text vectorization',
 'text vectorization',
 'text vectorization method',
 'weighting concepts based',
 'traditional text vectorization']
```

Figure example of KeyBert N-gram results (Ali Mansour, 2022))

Furthermore, if highlight function is set to true, we can see words that are highlighted.

VECTORIZATION OF TEXT USING DATA MINING METHODS, In the text mining textual representation be not only efficient but also interpretable, as this enables an understanding of the operational logic underlying the data mining models. Traditional text vectorization such as TF-IDF and bag-of-words are effective and characterized by intuitive interpretability, but suffer from the «curse of dimensionality», and they are unable to capture the meanings of words. On the other hand, modern distributed methods effectively capture the hidden semantics, but they are computationally intensive, time-consuming, and uninterpretable. This article proposes a new text vectorization called Bag of weighted Concepts BoWC that presents a document according to the concepts' information it contains. The proposed method creates concepts by clustering word (i.e. word embedding) then uses the frequencies of these concept clusters represent document vectors. To enrich the resulted document representation, a new modified weighting function is proposed for weighting concepts based on statistics extracted from word embedding information. The generated vectors are characterized by interpretability, low dimensionality, high accuracy, and low computational costs when used in data mining tasks. The proposed method has been tested on five different benchmark datasets in two data mining tasks; document clustering classification, and compared with several baselines, including Bag-of-words, TF-IDF, Averaged GloVe, Bag-of-Concepts, and VLAC. The results indicate that BoWC outperforms most baselines and gives 7% better accuracy on average

Figure example of KeyBert N-gram results with highlights (Ali Mansour, 2022))

- **YAKE! (Yet Another Keyword Extractor) –** Uses "text statistical features" which are extracted from single documents for the purpose of finding the most important words within the text. YAKE does not require to trained and

To install YAKE, users must write down the following code and then import it like below.

```
pip install git+https://github.com/LIAAD/yake
import yake
```

```
kw_extractor = yake.KeywordExtractor(top=10, stopwords=None)
keywords = kw_extractor.extract_keywords(full_text)
for kw, v in keywords:
  print("Keyphrase: ",kw, ": score", v)
```

Figure example of YAKE! (Ali Mansour, 2022))

Once YAKE has been imported, a "KeywordExtractor object" must be build. In the example above, the parameters used in this case are the numbers of words retrieved is set to 10; kw_extractor = yake.KeywordExtractor(**top=10,** stopwords=None) and a list of stop words will also pass.

Once that has been completed, using the **extract_keywords** function, it will return the keyword score depending on the length range from 1 to 3.

```
Keyphrase:  operational logic underlying : score 0.008502958451052589
Keyphrase:  text vectorization methods : score 0.015613284939549285
Keyphrase:  text vectorization : score 0.02310717508615897
Keyphrase:  Traditional text vectorization : score 0.02325791341228692
Keyphrase:  data mining models : score 0.02830809004349318
Keyphrase:  data mining tasks : score 0.033863083795882626
Keyphrase:  DATA MINING : score 0.03618462463953267
Keyphrase:  text mining tasks : score 0.037652251074155374
Keyphrase:  enables an understanding : score 0.04036782511075581
Keyphrase:  operational logic : score 0.04036782511075581
```

Figure Example of YAKE! Result output (Ali Mansour, 2022))

From this example, the most common used words "text mining", "data mining" and "text vectorization records".

- **TextRank – "**A graph-based ranking model for text processing" to find the most relevant words within the sentence as well as keywords in the text. Words are represented by a node and edges represent the relationship between words which are formed by "defining the co-occurrence of words".

To use TextRank, we must first install and import as usual like in the example below.

```
pip install summa
from summa import keywords
```

```
TR_keywords = keywords.keywords(full_text, scores=True)
print(TR_keywords[0:10])
```

Figure Example of TextRank (Ali Mansour, 2022))

Once the installation and importing has been complete, using the keyword functionality shown above, the text will show up and additionally, using the print function will also print the scores of each relevant word of the resulting keyword like in the figure below.

```
[('methods', 0.29585314188985434),
 ('method', 0.29585314188985434),
 ('document', 0.29300649554724484),
 ('concepts', 0.2597209892723852),
 ('concept', 0.2597209892723852),
 ('mining', 0.20425273810869513),
 ('vectorization', 0.20080655873686565),
 ('word vectors', 0.18267366210822228),
 ('computationally', 0.16718186386765732),
 ('computational', 0.16718186386765732)]
```

Figure Example of TextRank Result output (Ali Mansour, 2022))

- **Rake (Rapid Automatic Keyword Extraction) –** An extraction method which is effective for "multiple types of documents" with "specific grammatical conventions. Using Rake would be able to detect the most frequently used words or phrases in the written piece of text.

  For Rake to work, we must install and the package "multi_rake" and then import as shown below.

```
pip install multi_rake
```

```
from multi_rake import Rake
rake = Rake()
keywords = rake.apply(full_text)
print(keywords[:10])
```

Figure example of Rake (Ali Mansour, 2022))

Once we run the code, the most used words will show up and as shown in the example below, we can see that the most frequently used word is “text mining” and “data mining”.

```
[('data mining methods', 9.0),
 ('operational logic underlying', 9.0),
 ('data mining models', 9.0),
 ('modified weighting function', 9.0),
 ('weighting concepts based', 9.0),
 ('data mining tasks', 9.0),
 ('weighted concepts bowc', 8.5),
 ('low computational costs', 8.5),
 ('text mining tasks', 8.0),
 ('represent document vectors', 7.916666666666666)]
```

Figure Output of key phrases extracted from a document

The purpose of using a keyword extraction method is to describe what the written text is about by identifying the keywords or phrases within the document.

5.2.3 Perform Analysis and Send Result

The **Google Natural Language API** consists of 5 different services that can perform text analysis and shows the results for each type of analysis.

Syntax Analysis

In each sentence, Google Natural Language API can break down all the words with a rich set of linguistic information for each token. Basically, it finds the grammatical information of each word within the sentence, such as its type, gender, grammatical case, tense, and grammatical mood.

A	tag: DET
computer	tag: NOUN number: SINGULAR
once	tag: ADV
beat	tag: VERB mood: INDICATIVE tense: PAST
me	tag: PRON case: ACCUSATIVE number: SINGULAR person: FIRST
at	tag: ADP
chess	tag: NOUN number: SINGULAR
.	tag: PUNCT

Table 2 Syntax Analysis results provided by Google Natural Language API (Toptal Engineering, 2022).

Sentiment Analysis

Google's sentiment analysis can detect the emotions that are conveyed within a sentence given. The API returns 2 values, the score describes the positivity of the text, ranging from –1 (negative) to +1 (positive), with 0 being neutral. The magnitude measures how striking the emotion is being delivered. The more bombastic words included, the greater the magnitude.

Input sentence	Sentiment Results	Interpretation
"I go to school today for lunch."	Score: 0.0 Magnitude: 0.0	A completely neutral statement. Does not contain any emotion at all
"This meal is good."	Score: 0.7 Magnitude: 0.7	A positive sentiment, but not expressed very strongly
"This meal is very delicious. Whoever made this deserves a raise"	Score: 0.7 Magnitude: 2.3	A positive sentiment but expressed much stronger.
This meal is very delicious. But for some reason the ingredients are not very fresh.	Score: 0.0 Magnitude: 1.6	There are emotions expressed in this sentence based on the magnitude. However, the sentiment shows that they are mixed and not clearly positive or negative

Table 3: Sentiment Analysis results provided by Google Natural Language API (Toptal Engineering, 2022).

Entity Analysis

Google Natural Language API can also detect the entities that are present in each sentence such as a person, landmark or any object that can be seen and felt by humans. Some information in the form of a Wikipedia link and a salience score is generated based on the entity detected. The salience score is calculated based on the importance of the entity in the document, ranging from 0 being less salient, and 1 being highly salient.

"Robert DeNiro spoke to Martin Scorsese in Hollywood on Christmas Eve in December 2011"

Detected entity	Entity information
Robet De Niro	type : PERSON salience : 0.5869118 wikipedia_url : https://en.wikipedia.org/wiki/Robert_De_Niro
Hollywood	type : LOCATION salience : 0.17918482 wikipedia_url : https://en.wikipedia.org/wiki/Hollywood
Martin Scorsese	type : LOCATION salience : 0.17712952 wikipedia_url : https://en.wikipedia.org/wiki/Martin_Scorsese

Christmas Eve	type : PERSON salience : 0.056773853 wikipedia_url : https://en.wikipedia.org/wiki/Christmas
December 2011	type : DATE Year: 2011 Month: 12 salience : 0.0 wikipedia_url : -
2011	type : NUMBER salience : 0.0 wikipedia_url : -

Table 3: Entity Analysis results provided by Google Natural Language API (Toptal Engineering, 2022).

Entity Sentiment Analysis

Google Natural Language API can also combine both Sentiment Analysis and Entity Analysis to form Entity Sentiment Analysis. It will provide the salience score for the entity, magnitude, and score based on the sentence.

"The author is a horrible writer. The reader is very intelligent on the other hand"

Detected entity	Entity information
Author	Salience: 0.8773350715637207 Sentiment: magnitude: 1.899999976158142 score: -0.8999999761581421
Reader	Salience: 0.08653714507818222 Sentiment: magnitude: 0.8999999761581421 score: 0.8999999761581421

Table 4: Entity Sentiment Analysis results provided by Google Natural Language API (Toptal Engineering, 2022).

Text Classification

Google Natural API also provides a text classification model. It classifies the input documents into large sets of hierarchical categories, each of them has several sub-categories. A confidence score is given to determine how close the sentence belongs inside the category, the higher the relativity, the greater the confidence score.

"The D3500's large 24.2 MP DX-format sensor captures richly detailed photos and Full HD movies—even when you shoot in low light. Combined with the rendering power of your NIKKOR lens, you can start creating artistic portraits with smooth background blur. With ease."

Category	Confidence
Arts & Entertainment/Visual Art & Design/Photographic & Digital Arts	0.95
Hobbies & Leisure	0.94

Computers & Electronics/Consumer Electronics/Camera & Photo Equipment	0.85

Table 5: Text Classification results provided by Google Natural Language API (Toptal Engineering, 2022).

6 References

Wirfs-Brock, R. and Wilkerson, B., 1989. Object-oriented design: A responsibility-driven approach. ACM sigplan notices, 24(10), pp.71-75.

Toptal Engineering Blog. 2022. Looking for Meaning - A Google NLP Tutorial | Toptal. [ONLINE] Available at: https://www.toptal.com/machine-learning/google-nlp-tutorial. [Accessed 04 May 2022].

Goutham Golla, R., 2022. *Questgen AI*. [online] Github. Available at: <https://github.com/ramsrigouthamg/Questgen.ai> [Accessed 5 May 2022].

Mansour Ali, 2022. Four of the easiest and most effective methods to Extract Keywords from a Single Text using Python. Available at: < https://www.analyticsvidhya.com/blog/2022/01/four-of-the-easiest-and-most-effective-methods-of-keyword-extraction-from-a-single-text-using-python/> [Accessed 10 May 2022].

Kandi. 2022. Kandi X-RAY Questgen.ai Summary. [online] Available at: <https://kandi.openweaver.com/python/ramsrigouthamg/Questgen.ai> [Accessed 25 April 2022].

Karlsson Viktor, 2020. SentenceBERT – Semantically meaningful sentence embeddings the right way. Available at: < https://medium.com/dair-ai/tl-dr-sentencebert-8dec326daf4e> [Accessed 11 May 2022].

Bucanek, J n.d., 'Model-View-Controller Pattern', Learn Objective-C for Java developers, Apress,, [New York] :, pp. 353–402.

Curry, E. and Grace, P., 2008. Flexible self-management using the model-view-controller pattern. IEEE software, 25(3), pp.84-90.

Observer 2022?, Refactoring.guru, viewed 5 May 2022, <https://refactoring.guru/design-patterns/observer>.

Varia, J. and Mathew, S., 2014. *Overview of amazon web services*. Amazon Web Services, *105*, viewed 9 May 2022, < http://cabibbo.dia.uniroma3.it/asw-2014-2015/altrui/AWS_Overview.pdf >.

Amazon Web Services Inc, 2022, *Amazon Relational Database Service User Guide*, pp. 171-176, viewed 10 May 2022, <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/rds-ug.pdf#CHAP_Tutorials.WebServerDB.CreateDBInstance>

Kulkarni, G., Sutar, R. and Gambhir, J., 2012. Cloud computing-Infrastructure as service-Amazon EC2. International journal of Engineering research and applications, 2(1), pp.117-125.

Garfinkel, S., 2007. An evaluation of Amazon's grid computing services: EC2, S3, and SQS.

7. Appendix

Client Research Report

Document Submission System

Client Research Report

Software Engineering Project | 28

Name	Position	Email	Phone
Jun Wee Tan	Leader	101231636@student.swin.edu.au	0493461576
Xin Zhe Chong	Member	103698851@student.swin.edu.au	0406259449
Adrian Sim Huan Tze	Member	101225244@student.swin.edu.au	0474642003
K.M. Yovinma M. Konara	Member	192426323@student.swin.edu.au	0411101072
Sandali T. Jayasinghe	Member	102849357@student.swin.edu.au	0470617473
Richard Ly	Member	103340644@student.swin.edu.au	0424391144

SWE40001 EAT40003, Software Engineering Project A, Semester 1 AND 15/04/2022

DOCUMENT CHANGE CONTROL

Version	Date	Authors	Summary of Changes
0.0.1	15/4/2022	Jun Wee Tan	• Append introduction section • Modify definition of plagiarism • Completed Metrics of plagiarism
0.0.2	15/4/2022	Xin Zhe Chong	• Modified current state of plagiarism
0.0.3	15/4/2022	Sandali Jayasinghe	• Modify current state of plagiarism • Modify Existing software that can support text analysis
0.0.4	15/4/2022	Richard Ly	• Added more information about current state of contract cheating. • Added more information about metrics of contract cheating
0.0.5	15/4/2022	K.M. Yovinma M. Konara	• Reviewing and formatting of document content • Written introduction to plagiarism and contract cheating and statistics related
0.0.6	15/4/2022	Adrian Sim Huan Tze	• Changes font style and makes adjustment to font size. • Added Problem Statement • Added definition of Contract Cheating • Added Prevalence of Contract Cheating

DOCUMENT SIGN OFF

Name	Position	Signature	Date
Jun Wee Tan	Team Leader		15/04/2022
Adrian Sim Huan Tze	Member		15/04/2022
Xin Zhe Chong	Member		15/04/2022
K.M. Yovinma M. Konara	Member		15/04/2022
Sandali T. Jayasinghe	Member		15/04/2022
Richard Ly	Member		15/04/2022

Table of Contents

Introduction

This research report aims to peruse the topic of "plagiarism and contract cheating", including its associated definitions, metrics and the state of plagiarism and contract cheating the current world.

With the informative research findings, the team of developers for the project "Document Submission System" can utilize these findings to conduct a more focused and thorough text analysis of the document submitted by the individual. Furthermore, the report aims to provide more information for the client on contract cheating and plagiarism in a more specific context, such as the types of cheating and popular trends in cheating and plagiarism.

Problem Statement

Academic plagiarism has been a crucial problem for decades; with the current widespread distribution of advanced information technology such as the Internet, an individual can commit plagiarism much more effortlessly. The definition of plagiarism and contract cheating have drawn much attention from the research academy in the past two decades, where it was vastly discussed in terms of defining and solving both plagiarism and contract cheating. There are a wide variety of papers that define plagiarism and contract cheating, which will be introduced and reviewed comprehensively in this report. The main focus this report is to study the current state of plagiarism/contract cheating and give an idea of how frequently it occurs in the academic industry and discuss solutions that help detect plagiarism and prevent it from occurring in an academic instance.

Definition and metrics of plagiarism and contract cheating

Definition of plagiarism

Plagiarism is defined as an individual who submits or presents their works without acknowledging the original source and author of the work (University of Sydney) and according to (Cambridge dictionary) plagiarism is the act of stealing another individual's ideas and claiming them as one's own without due credit. It is a severe academic issue and makes no difference regardless of the individual's motivation behind committing the act, whether purposeful or not.

Definition of contract cheating

According to International Centre for Academic Integrity, contract cheating is defined as the outsourcing of academic work intended for students to third parties such as family and friends, assignment assistance services (ICAI).

(Clarke and Lancaster, 2006) initially coined the phrase 'contract cheating' and further elaborated that in the early stage of contract cheating, it is an issue which is more likely to be limited to computer coding work. However, this academic issue has recently been expanded and emerged to all disciplines across the higher education sector. Whether these third parties are reputable providers or not, contract cheating remains to be defined as series of practices relating to the outsourcing of student's assessment work to third parties (Bretag et.al, 2019).

Both these acts are highly popular in the academic sector; academics and students are consistently advised to reference findings that are not their own. If not, they would have to face consequences such as penalties or failed assessments, as a result of failure to comply under academic standards.

Metrics of plagiarism

Plagiarism can be conducted by copying any content without adequately referencing it, regardless of the medium in which the original information was published in. The following shows a list that is popular yet not limited types of media that may be vulnerable to plagiarism:

- Print books / e-books
- Research reports
- Newspaper articles
- Magazine articles
- Journals / e-journals
- Theses
- Internal materials
- Email
- Images
- Audio
- Video (such as YouTube video)

Process of plagiarism checking

For most plagiarism tools, plagiarism checking will be conducted by advanced scans that will peruse for any similarities between documents and existing text using complex database tools. To be more specific, the plagiarism tool will scan all text in the document and cross-check against a database of existing content including web sources, scholarly journals, scientific publications, and books (Tegan & Jack 2022).

According to (Turnitin), there are four methods to conduct the checking**.** The first will be to go through a 'keyword analysis' where any exact matches with keyword will be identified and highlighted, similar to a search engine. The second method will be to analyse strings of words which is usually a combination of three or four words that can be formed into a sentence. The third method will be scanning and analysing sentence writing styles where the system will examine word sequences such as phrases used and compare them to other existing documents. The final method is using what is defined as 'fingerprinting analysis'; this is where plagiarism tools will scan and identify the unique fragments such as a word or sentence and the order of word fragments placed in a document.

Both the first and second methods of checking help identify an exact copy of original content but are weak in detecting plagiarism in a sentence that has been paraphrased. The third method can address issues related to paraphrasing issues but does not work on checking exact word matches. Fingerprinting analysis can manage all the issues that the previous three issues encounter. It analyses the document's unique fingerprints, such as tone, style, and phrasing and the plagiarism system and will put up a red flag if the examined document consists of unoriginal fingerprints.

After the completion of scanning and analysis process, the plagiarism detection system will then send a report that defines results such as plagiarism percentage, highlights of plagiarized sentences, and lists of sources to the user.

Different plagiarism checkers have their own operational methods and are limited by the following factors:

- Checking Algorithm
- Database size – the larger the database, the more precise the plagiarism check
- Quality of scanning – High quality plagiarism checker will perform fingerprinting analysis to discover non-exact matches among paraphrased or changed texts. Each fragment of sentences will be checked for similarity.

Acceptable plagiarism percentage for a document

The acceptable plagiarism percentage for a document is ambiguous and lacks consensus on a specific threshold for what is considered as acceptable in plagiarism. The main reason is because each plagiarism tool has its own standard and not all universities have a strict requirement on the exact value considered

as the acceptable percentage of plagiarism. Different papers will also have varied standards, such as research reports and journals.

However, the rule of thumb is that the plagiarism score is generally maintained below 20% of similarity, excluding references and try to sustain zero plagiarism to avoid any issues.

According to (Turnitin), a web plagiarism detection service, the similarity of document can be categorized to five categories, which is blue, green, yellow, orange and red.

Similarity	Definition
Blue	No matching text
Green	Below 24% matching text
Yellow	Between 25-49% matching text
Orange	Between 50-74% matching text
Red	Between 75-100% matching text

Table 1 Turnitin similarity score table (Turnitin)

Any document submitted that results in a similarity report below 25% is considered acceptable and as original work. But the similarity reports should be as a reference in identifying the potential plagiarism score and not the level of plagiarism committed by an individual (Jessica 2020).

Metrics of contract cheating

Contract cheating can be measured through either survey or observation. From a surveying perspective, students can undergo a survey regarding academic dishonesty, demographic and other activities whereas in an observatory perspective it is much harder since students who are aware of them being monitored will change their strategy or behaviour to prevent getting caught in the act.

Additionally, contract cheating can be measured by the instructor examining the contents of the submitted work by identifying the key components such as the name of the author, the language utilised and the general structure of the work and if the submitted work is relevant to the topic at hand (TEQSA 2022).

Instructors may also be able to measure 'contract cheating' by comparing student's past submissions and seeing if there are any discrepancies in the language style used in the current submission in comparison to prior works (TEQSA 2022).

Process of determining activities of contracting cheating

(Fendler and Godbey 2018) designed a test specifically to punish cheaters and give honest students the credit they deserve. The test consists of 2 identical papers but slightly different details to fool the cheater into thinking it was the same question. If both the student and the cheater answer the question correctly, then both answers must be different, which means no cheating has occurred. If both the cheater and student's answer are the same, it is an indicator of cheating, and the cheater can be identified easily based on what answer matches more to the question defined in the papers. The regular and F/G systems used to identify all instances of cheating are represented in Table 2 below as follows:

Regular System		
Possible outcomes	**Cheating indication**	**Cheater?**
Student A and Student B does not match	No cheating indicated	-
Student A and B match, but both are correct	No conclusion	**No conclusion**
Student A and B match, but both are incorrect	Suggests possible cheating	No conclusion
F/G System		
Possible outcomes	**Cheating indication**	**Cheater?**
Student A and Student B does not match	No cheating indicated	-
Student A and B match, but only Student B is incorrect	Suggests possible cheating	**Student B**
Student A and B match, but only Student A is incorrect	Suggests possible cheating	**Student A**
Student A and B match, but both are incorrect	Suggests possible cheating	No conclusion

Table 2 Regular vs F/G System for Identifying Cheaters (Fendler, Godbley, 2018)

Current state of plagiarism and contracting cheating

Current state of Contracting Cheating

There will be instances where students will essentially take credit for completing work or assignments they have not done themselves, in the event if plagiarism or contract cheating is committed (Lee 2019).

Forms of contract cheating remains to be the same but has now given rise to different virtual and easily accessible services available online such as downloading a written papers on various free essay sites, sites that allow the downloading of documents related to their course such as StuDoc, working with other students on exams that are meant to be individually conducted, and allowing other students to use your work and submitting it as their own in exchange for money as often, contract cheating requires a payment fee for the work to be used (Billa 2022).

Recently, laws were introduced regarding contract cheating services where in the instance that they are found guilty of providing students the means to use their services, the providers themselves can face up to 2 years in jail as well as a fine of $110,000 (Taylor 2021).

According to (Newton, 2022), 65 full assessed studies from 1978 to 2014 showed there was a 12% increase in contract cheating, this would represent over 31 million students as of post-2014 using contract cheating to completing their assessments. As for plagiarism, according to ARGA (n,d.), a study performed by Kessler showed that over 76% university and college students have copied assignments content word-for-word.

It was also recorded that COVID-19 had affected contract cheating and plagiarism, with plagiarism incrementing by 10% post-pandemic, based on a study with 51,000 participants conducted by CopyLeaks, according to (Schaffhauser, 2022). (Hill, Mason and Dunn, 2021) also states that COVID-19 has benefitted many contract-cheating services, with some offering COVID-19 discounts on their services.

This is also supported by another research study conducted by the University of Western Australia in 2021 during the height of COVID-19 (Prakaash 2022). According to Curtis (2021), it was found that 8-11% of students submitted assignments and assessments that was written by a third-party and “over 95% of students who cheat in this way are not caught”.

Furthermore, due to COVID-19, it has enabled many contract cheating service providers to flourish as more universities look to online teaching, thereby allowing students to complete their online exams without supervision (Prakaash 2022). Table 3 shows the results of the survey as follows:

As a result, the market for these contract cheating providers has increased greatly as they are able to provide students with access to a wider variety of subjects and topics and even display advertisements on Facebook, Twitter etc. to further entice students to pay for their services (Prakaash 2022). An example of such an advertisement is presented below:

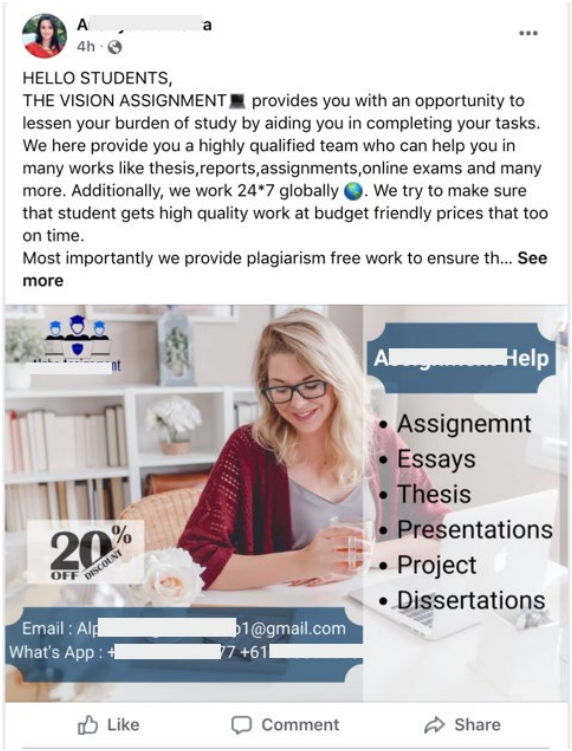


Figure 1 Advertisement for contract cheating (thepienews)

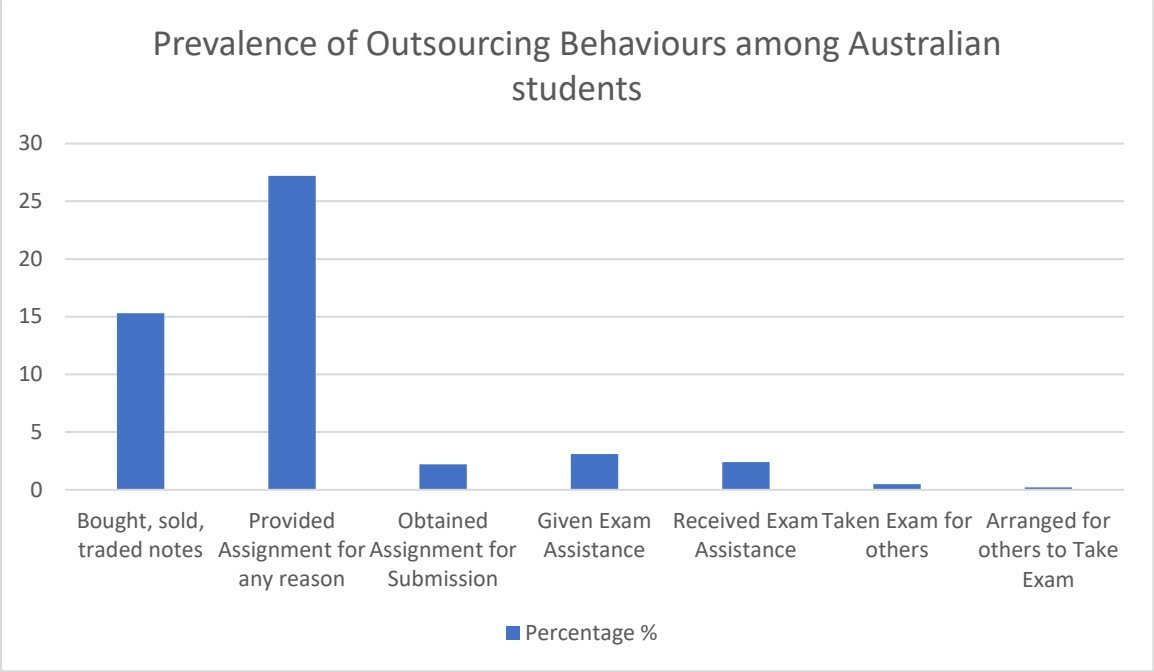


Figure 2: Prevalence of Outsourcing Behaviours among Australian students (Bretag et.al 2019)

With the rise of high frequency in committing contract cheating in the current world, many educators have shown concern regarding students' natural tendency to outsource other learning attributes like notetaking and text paraphrasing. There are several outsourcing websites of file-sharing and peer-to-peer networking

like ThinkSwap, Course Hero and Baidu Library that allow students to trade notes and other course-related materials. Online paraphrasing tools such as GoParaphrase or Paraphrasing Online can be used to spontaneously generate different phrasing for any entered text, making the marker of the plagiarism detection system a harder time to determine if the work is plagiarised.

Current state of Plagiarism

In the recent years plagiarism among students increased by 10 % after the COVID-19 pandemic, when almost all the classes went online. There was an increase in the average rate of copying in student work from 35% to 45%. (Schaffhauser 2021). In a study conducted by Christian Hopp & Alexander Speil (2021) on the prevalence of plagiarism, it was found that there are significant levels if plagiarism among a group of first semester students in university.

The rapid evolution of technology in the past two decades is said to have an impact on plagiarism in universities. With the help of digital technologies students can find an unlimited amount of content regarding the respective subjects online, which allows the students to locate the resources they want to use in their work. To use the content found in these sources the student must provide references. (Viper Blog 2019). McKenzie even quoted that the modern-day plagiarism requires little effort and is far more powerful than the plagiarism methods of yesteryear. Students can now simply download and obtain hundreds of pages per via the internet (McKenzie, 1998). In the present, plagiarism occurs in different stages of publishing (planning, research, writing) and applies to both electronic and printed versions (Agrawal 2020).

Students search for the answers by simply looking up the internet and using sections of the questions as the keywords for their search. As an example, if an assignment requires a student to produce a report regarding the impact of COVID-19, the student only needs to open their browser and enter “COVID-19 impact research”. Many results of similar reports will appear, and the student will select and copy the reports that matches the most to his assignment requirement. The answers could come in different shapes and forms depending on the content requirement of the assignment or the cleverness of the student. This includes but not limited to:

- Copying other student’s paper (current or graduated)
- Copying own paper (previous works)
- Copying segments/entirety of report/journal/article from the internet
- Copying codes (entirely) from online repositories or forums

If the student understood the requirements of the assignment and are able to generate keywords from it, the internet will be helpful for them.

Existing software that can support text analysis

Text analysis software, text analytics or text mining software helps users gain understandings from structured and unstructured data using Natural Language Processing (NLP). These insights include key phrases, sentiment analysis, language, patterns, and entities. Visual representations are prepared from

these insights with the influence of NLP and Machine Learning for easier data representation (G2-Business Software Reviews). Some common features of Text Analysis Software include:

- Ability to understand the language of the text written in,
- Syntax Parsing – breaking down a sentence and understanding how and why it was constructed.
- Key phrase extraction – helps users understand patterns and themes within the text.
- Sentiment Analysis – Helps classify the text as positive, happy, negative, sad or neutral.

The Top 5 text analysis software ranked by G2 Business Software Reviews based on user satisfaction are:

- RapidMiner
- Confirmit
- Chattermill
- Amazon Comprehend
- Relative Insight

The main users of text analysis software are as follows:

- Finance Teams - To analyze business performance and discover trends in free-text data
- Data Analysts - To set up the software for other employees or teams and create queries to get an understanding of business-critical data
- Sales and Marketing Teams - Gain insights on sales performance and optimize the sales revenue and
- Customer Service teams - Gain understanding of insights of messages and respond to messages in a targeted manner

Research Methods

Jun Wee Tan:

Google Chrome browser was utilized for finding any research report or article related to both contract cheating and plagiarism. Below are my search keywords when doing this research report:

1. Contract cheating Swinburne
2. Difference between contract cheating and plagiarism
3. How to measure plagiarism
4. How measure plagiarism work
5. Acceptable percentage of plagiarism report

Adrian Sim:

Research papers were searched with the help of Google Scholar; these papers were searched such that they are relevant to the topic of plagiarism and contract cheating. More than 100 results are returned with the keywords included in the title of research papers for each search. All the results returned are relevant study

conducted by other researchers about contract cheating and plagiarism. Below listed down the keywords from my searches:

1. Contract cheating
2. Plagiarism
3. Detecting plagiarism
4. Detecting contract cheating
5. Metrics of contract cheating

Xin Zhe Chong -

Google search in Microsoft Edge was utilized for finding the research reports that previously conducted studies on plagiarism. This includes the prevalence of plagiarism as well as the type of plagiarism. There was also an interrogation of friends and colleagues from other universities regarding their take on plagiarism.

1. Current state of plagiarism
2. Metrics of contracting cheating

K.M. Yovinma M. Konara:

Google search and Google Scholar was utilized to search for the definitions and all statistics related to both plagiarism and contract cheating. As most papers were limited from public use without purchase, the Swinburne Library Database was used to get access to the papers mentioned in the report. Some of the searches conducted were:

1. Definition of plagiarism
2. Definition of contract cheating
3. Recent contract cheating statistics
4. Recent plagiarism statistics

Sandali T. Jayasinghe:

The Swinburne Library Database and Google search platform were utilized to find articles and research papers on the topics current state of plagiarism and text analysis software.

Richard Ly

Google Chrome was used as the main browser for finding research reports and articles on past studies on plagiarism and contract cheating as well as what is considered plagiarism and contract cheating. The searched terms are as follows:

1. Current State of Contract Cheating in Australia
2. What is contract cheating
3. How do contract cheating providers promote their site
4. Metrics of Contract Cheating