Improve Learning and Teaching Activities by Incorporating Version Controlling Concepts to Learning Management Systems

Amarasinghe N.G.C.M.
158750H

Faculty of Information Technology
University of Moratuwa
June 2018

Improve Learning and Teaching Activities by Incorporating Version Controlling Concepts to Learning Management Systems

Amarasinghe N.G.C.M
158750H

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the fulfilment of the requirements of Degree of Master of Science in Information Technology

June 2018

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Name of Student — Signature of Student

Amarasinghe N.G.C.M. — ……

Date

Supervised by

Name of Supervisor — Signature of Supervisor

Chaman Wijesiriwardana — …….

Date

Dedication

We dedicate the output of this research work and thesis to development team of existing Learning Management Systems. Also I specially dedicate this new features to all those who generously contributed their valuable time, advising and helping in doing this research, especially to my supervisor Mr. Chaman Wijesiriwardana. And I hope the research and the findings described below will provide a useful insight for applying version controlling mechanisms to existing Learning Management Systems.

Acknowledgement

First of all, I would Like to thank my research supervisor Mr. Chaman Wijesiriwardana who spent his valuable time in guiding this research to make it a success. Furthermore, my next big thank goes to Prof. Asoka Karunanandha who taught us Research Methodology and Literature Review and Thesis Writing subjects which were the basis for this research.

Not only that my thanks should go to all the lecturers in M.Sc in Information Technology degree program of Faculty of IT, who gave their hands to sharpen our knowledge and ideas throughout these two years as they were the illumination which lit up our pathways to success.

Apart from the people who were directly involved, many more helped to make this project successful. Especially the people who engage with two surveys I have done with this research at the beginning and at the end as the evaluation part. Finally, I would like to thank all the batch mates of the M.Sc in IT degree program who gave their valuable feedback to improve the results of the research.

Abstract

E-Learning is becoming a most needful part of our education system. To the combination of the teacher and the student is more important like teaching something. As one of the major features of the existing e learning systems, “uploading documents” have become to a major functionality. Within this research, the researcher focuses to do needful validations to this major upload functionality (When updating the uploads previously done by the same user). After doing some initiate survey among the stakeholders of LMSs, identified the requirements exactly what the researcher has to do. Also as a software developer, the researcher always took the experiences from the version controlling systems (software developers use) like Bitbucket and GIT, which are in his working environment.

For full fill the requirements, the researcher found sources of some existing LMSs (Media Share android 1.1 and Moodle 3.5) and tried to apply the solutions on that. As a critical step, take a time to compare two existing systems like GIT and moodle (a version controlling system and an LMS), summarized the features, nice to have in LMSs while uploading. Then could generate own mechanism to compare two files and give a good validation for the files while uploading contents to the same place. And the researcher could process a SUS base survey among the lecturers and students about the new functions and took the SUS final result for that. And as the final result, the survey gives "Ok, we can improve more". And could handle a performance test for whole new features and could prove the performance are an acceptable level after the new implementation.

The reason behind the result is the researcher could give a good outcome from the research which could keep the existing patterns of the existing system and the new methods could stay with existing interfaces and keep the other functions as it is. Users of LMS got the new features without not taking any difficulties. And could give an answer to an existing problem which was in uploading files. This research didn’t focus on PDF file uploading, as a future task to do, the researcher like to specially mention that. And this is only focused on texts. In future, researchers who find solutions for content comparison and uploading can think about the files which contain videos, audios, and images also.

Table of Contents

Table of Contents for Figures

Table of Contents for tables

Chapter 1

Introduction

1.1 Introduction

This chapter gives an introduction to overall project on Integrate Version Controlling to LMS. Learning is the act of acquiring new or modifying and reinforcing existing, knowledge, behaviours, skills, values, or preferences which may lead to a potential change in synthesizing information, depth of the knowledge, attitude or behaviour relative to the type and range of experience. The ability to learn is possessed by humans, animals, plants and some machines. Progress over time tends to follow a learning curve. Learning does not happen all at once, but it builds upon and is shaped by previous knowledge. To that end, learning may be viewed as a process, rather than a collection of factual and procedural knowledge. Learning produces changes in the organism and the changes produced are relatively permanent [7]. There is a different kind of learnings. The Past to present people uses a lot of strategies to learn.

In present, it is promoting with the internet. We call it as e-learning. That is a trend now. Not only the students who are in Higher studies, even the students in primary education also use the internet as their learning path. It is the best way for the shoot ideas from someone to another, from one place to another with speed.

So taking this as a good learning way, a lot of universities using e-learning as their main education system. Not only for the reading, they are using the systems based on the internet to a lot of activities. Some of them share the articles and learning materials with students, online assignments, for good communication between teachers and students ext.

In that case, In the e-learning systems, teachers do a lot of activities. They are adding, deleting, submitting the new notice, publishing reading materials, Adding new assignments with the deadline. And they updating and deleting. Because they are humans they also do mistakes and regularly they are changing, change their ideas. They

have to update and delete.

Within this research, the researcher focusses the communication areas in the e-learning systems. If going through the famous e-learning systems in the world, normally we can see course panel which interacts the students and the course related lecturer. And the lecturer adding their submissions to assignment creating area in that panel. I do some modification with the version controlling that is the most suitable place in the system.

Version Controlling! Its big word in software engineering fields. According to some facts reported on the web, one of most needful part for the software developers is this [8]. Within this research, the researcher focuses on to the applying the version controlling features to the e-learning systems. To do that I use the existing UI functionalities, the interactive area between the lecturer and the students. Because I identified these are the requirements which came from the users of e-learning platforms accordance with pre-research.

The users of the e-Learning systems, need version controlling. But not the 100% same from the famous version controlling systems. Normally the developers, software engineers in the software engineering field, do the same for store their versions, modifications and changes time to time. Sometimes this software engineering process keep the whole project protection. It will keep the outcome of the product. Considering these heavy expectations, the professional version control systems like BitBucket, perforce ext. maintain their complex logic and features.

Do we need to implement all the version controlling features in e-learning version controlling areas? Before giving an answer to that, need to know what the users of the e-learning system expect from the version control. So as the first step of the research I did some small sub research on e-learning system's user's version controlling expectations. And I gathered the features what the popular version controlling system use day today. The sub research I execute Mainly targeting the students of the universities and their lectures. Identified what they really need, under the version controlling domain. Then went through the pre findings and understand the capacity what we can really apply here. Considered what should be the output.

1.2 Aim and Objectives

- Aim of the research

Aim of this research is to integrate the concept of traditional Version Controlling Systems to Learning Management Systems and designate the limitations in file uploading.

- Objectives:

- Study the limitations of Learning Management Systems.
- Identify the features selected from Version Controlling Systems.
- Integrate the Version Controlling System's features into a Learning Management Systems.
- evaluation

1.3 Background and Motivation

Use of version control systems is normal for Software developers/engineers. Every day at the end of the day or at a milestone of the development, the developers push their codes to Bitbucket. It is a best practice. And the functions of the Bitbucket, are Merging, pull requests, commit and push the codes, code reviews, see the history of the code, conflict resolving and taking alert messages ext. These things, Developers/Software Engineers do every day in their work life.

And when the researcher uses the Moodle when follow his university-related works, I feel completely different idea. And sometimes, researcher, I thought as a developer. I do mistakes and Bitbucket helps me to catch up those things and it makes me do my developing things with very clean. The same human, when touching the Moodle at the university for learning functions and do mistakes regularly but Moodle is not acting as Bitbucket. It takes everything that I do. As the user, I have to again log to the system and re-correct what I mistakenly did. So I thought if Moodle has version controlling features, it will be great. And I discussed with some lecturers and students also. I wanted to know about their feelings. Yes, I was correct. They had the same feeling and same expectations from the LMSs. That shows me the way to start this research.

When I do my survey among the stakeholders of LMSs. I needed to know, what is the major expectation from their side. And I choose some major features to implement my research. That will be a first step to think version controlling apply in Learning Management Systems.
There are a lot of researchers that have been done for the version control. And also I could found research documents and blogs for the e-learning systems. And according to some articles version controlling follows new trends to polish existing features [7][8]. Some software companies have done some experiments on version controlling but they have not published the logic and algorithms because of the competition of the companies which develop version control systems [8].

As an example, GIT has the huge client base. They give interface them to work not give any logic to use individually, and since its release, become an increasingly common version control system used by software projects. The service provider Github hosts approximately 6,100,000 open source repositories. Gaining access to these repositories would increase the code basis on which researchers can perform software analysis[10].

In that case, if apply the mechanisms of version controlling in our systems, we have to find out and build our own algorithms and logic. Some of the logic we can take from the developing language. As an example, if we need to compare two documents and find the differences, we can just use removeAll() predefine function in Java. But the difficulty is maintaining the performance. We are going to compare two documents which can be with the huge amount of words uploading by the students and the lecturers. There can be more than 100000 words (can be more). So I had to find out new methodologies and functions to keep high performance

1.4 Problem in Brief

There is a big question on applying version controlling to the e-learning software platforms. Considering and referring existing version control systems, is it needful to have that kind of high-level scenarios here? But some far those facts must be applying.

It is a requirement coming from the users. So what kind of features we can apply in the learning systems? What are the needful facts to consider when applying them to performance? We have to focus on performance and accuracy about the new implementations.

BitBucket	**Moodle**
Can Merging	Not need Merging
Can Branching	Not need Branching
Having approval process	Not Having approval process
Can have conflicts	No conflicts at all
Multiple users will work on one moment	One user use at a time
Need to see history deeply	Need to see history
Need Validating before upload	Need Validating before upload
Need Notification	Need Notification

Table 1.1 comparison the requirements with current Bitbucket

There are a lot of features in Version Controlling Systems. As of the pre-research I have done, I filtered out what are the features really need for Learning Management Systems. As the comparison, I took Moodle and Bitbucket.

1.5 Proposed Solution

We proposed, apply version controlling features, when the users of the eLearning system, upload or adding new items like assignments, notices next to the system, the users should be able to remind what he has done so far by reviewing his history. And when updating something he already adds, the system should revise the changes and detect the user's mistakes.

As the main focus of this research, I do some validating and reviewing to uploading and uploaded files. Some example scenarios are mentioned below.

Give notifications for following scenarios.

- Only change the content of the file, no changes in the file name.
- Only change the file name of the file, no changes in file content.
- Change the title and the content.
- Updated the content with adding more lines to existing file.
- Updated the content with removing some lines.
- Updated the lines with adding new lines and removing some lines.
- Updated the content's line by changing the positions.

And done an initiative for file bulk uploading and prevent upload same files again and again.

1.6 Structure of the Thesis

The overall thesis is structured as follows, Chapter gives an introduction to fill project with the objectives, background, problem, and solution. Chapter 2 critically reviews the literature in the eLearning system with version control features. Chapter 3 is about details of technology by showing it's relevant to real Assignment and for tutorials given by a university lecturer. Chapter 4 present our approach with users, inputs, outputs, process and features. Chapter 5 is the design of the version controlling which are famous eLearning systems, while chapter 6 implementation of the solution. Chapter 7 reports on the evaluation of the solution. Chapter 8 concludes the solution with a note on further work.

1.7 Identified Limitations.

- File comparison research done for text files.
  The introducing logic and algorithms have tested and verified for texts only.
  Not verified for audios videos and images.
- This document comparison has done with line wise.
  In this research, I followed line wise comparison for the documents. The same

logic can apply to word wise, letterwise or paragraph wise. That is depending on the product and its user base.

- When doing the bulk upload, the system compares the files in root level.

1.8 Summary

This chapter gives an introduction to the overview of the project on version controlling logics apply in eLearning system. We discussed main objectives and motivations factors to do the project in a broad manner with supported by citations. We defined the research problem and our proposed solution in briefly and finally defined the overall structure of the thesis. Next chapter will discuss the use of version controlling logic in eLearning system.

Chapter 2

Review of others works

2.1 Introduction

This chapter critically reviews the use of Applying version control mechanism in Learning management system and how the others have found the solutions to the problems in applying version controlling in software systems.

2.2 Use of content management in various systems

There are a lot of researches done for version controlling and its related mechanisms. Also same scenario to the Learning managements. But I couldn't find any thesis which directly done research on applying version controlling to LMS. But could happy with some researchers which have done for version controlling applying in some different systems like File management systems (File syncing system) ext.

The research named, A Version Control System for Everyone, done by Alexander Chumbley in Massachusetts Institute of Technology [1], Version control systems have, for many years, been applications that are developed and maintained with software engineers in mind. However, other less technical industries and endeavours can benefit immensely from the functionality these systems provide. Unfortunately, existing version control systems, such as Git[10][11] and Mercurial, have too steep a learning curve to make quick adoption feasible, especially for new or less technical users. And the researcher is focusing on File Syncing system's versioning. And he mentioned more. File syncing systems, on the other hand, such as Dropbox and Google Drive [7] are easier to learn and understand. But, they do not offer the same level of functionality as version control systems. We believe that the best from both of these types of systems can be combined to provide a simpler version control system, accessible to anyone. In his research, he has provided two main contributions. He outlines the steps needed to design a simpler, more powerful version control system by following Conceptual Design Theory. Then they have described the system, which they created to fulfil this

vision. The system, Snap store, is the result of all the goals and ideas described by that paper.

Other than that there was a summary report published with a research done by a group of Jisc Northern Ireland in the UK in 2015 [2] and they do file versioning and its benefits. As the text in the report, benefits of Version Control when used with the documents versioning are,

• There is an 'audit trail' of how a document developed during the drafting process.

• You can be confident that you have the most up to date version of a document

• You can prove which documents were 'in force' at a particular date – this might be crucial for appeals processes, for example.

• You can confidently delete draft or redundant versions of documents.

And researcher call Miettinen, Antti tried to apply version controlling to his all the using systems of his university Aalto, by research call Version Control Systems - development, comprehension and rationalization of usage: Case Aalto University [3]. From his research he was able to find out, how could the usage of version control systems be improved in Aalto University? What are the preparations prior a version control system project? and he had focused the security of the existing and suggesting process. And according to his documents, his all university systems, which were talking, linked with famous version controlling systems like GIT, SVN and Bitbucket. The most common uses were coding and the writing process of a research paper. The Department of Computer Science of his university also used their VCS for their courses and kept all the learning material stored in the system. The learning material was distributed with VCS and students could also return their assignments via VCS. And one of academic base systems, which is mainly used by the lecturers of his university, also already linked with existing version controlling system GIT. Mainly for research paper writing they need this versioning. As he mentions, when a group is writing a research paper they use this VCS. Research paper can have more than one author and as a group they need to develop a research paper. In that scenario also his system used a link to GIT. So using a third party tools they do versioning in his university systems. As the improvement, he suggests and tries to focus his solution to move "centralized version controlling" for his university. But that is a one system for whole systems in his university. It is complex. No point to merge developer's code's base systems to lecturers learning base systems together for this versioning. Other than that he was focused to the data existing. That is good. We have to think about existing details,

already saved in databases. And also as his third view, he focused to app's security. For my research, "Version Controlling Logics Apply in Learning Management Systems", I used existing security features. With the development of new features for versioning, I always get this as a responsible thing and focused to exist with existing security features of particular LMS.
Other than these documents, I referred a lot of thesis of related researches done for version control. Most of them basics on the Software develop versioning.

2.3 Summary of Challenges

Our discussion of the previous section has identified a large number of version controlling problems in Learning Management Systems and lack of solutions for in build version controlling logic based solutions. Showing below, a summary for those applications.

2.4 Problem Definition

As per literature, all detailed information could found, there is no proper LMS app which is within build version controlling.

Therefore, it was identified the critical need to develop and integrate version controlling methods to existing LMSs to overcome above circumstances.

2.5 Summary

This chapter presented a comprehensive critical review use of version controlling methods use in Learning Management Systems. I reported developments in version controlling around LMSs. We defined the research problem and also identified the possible technology addressing the research problem. Also, I identified how do apply this to already in use systems. Next chapter will discuss the technologies adapted for solving our problem.

Chapter 3

Technology

3.1 Introduction

Chapter 2 presented the previous projects and researches done by various researchers and how far they are tally with this proposing system. Here we are going to discuss the technology for proposing a system.

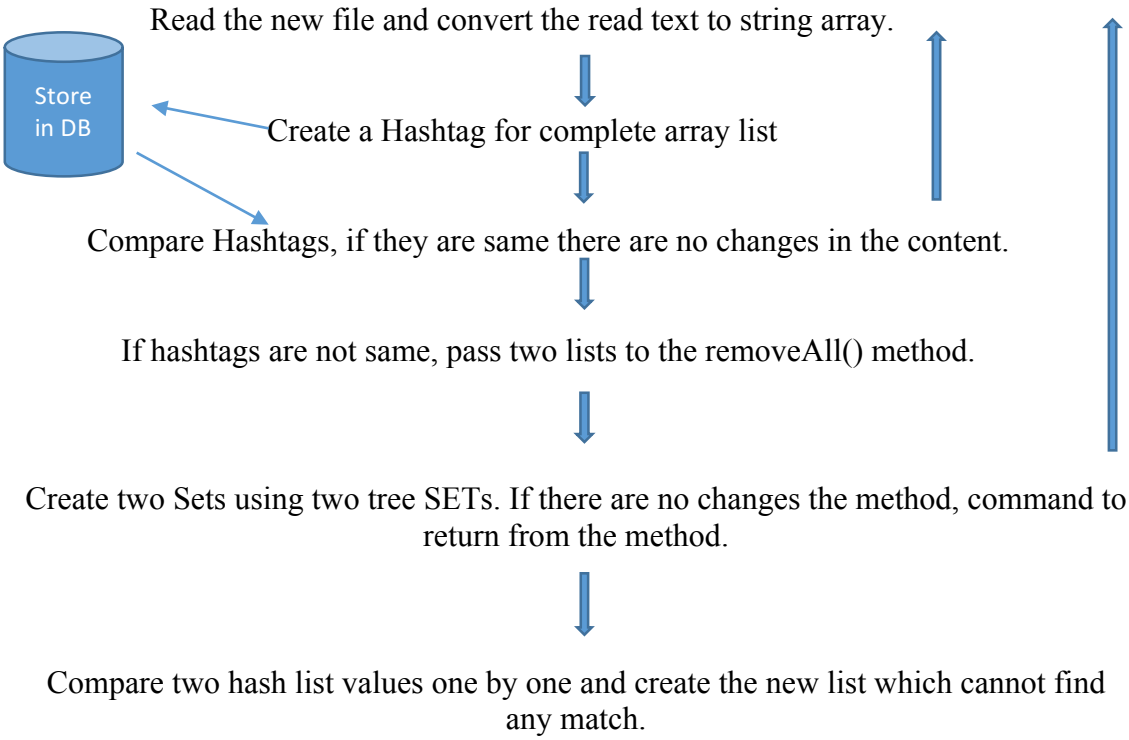
3.2 Technology basics

Here these technologies applying into the existing developed already using eLearning systems. Especially focus to the course panel that interacting the students and the course lecturer. And mainly, the researcher applied the findings with tallying existing development languages. As an example one of the famous eLearning systems, Moodle developed by a framework of PHP. Once applying some parts there, the researcher I had to follow the architectural patterns exist with selected system. And had to implement a separate database to store researcher's data because the database has not been provided to outsiders by the Moodle development team. So I had to make a separate channel to store my required data in MySQL database.

And for the testing purposes, brave enough to apply some expected features on mobile eLearning apps (Pearson Mediashare android app). And when developing and research on new logics always thought about the code complexity and run-time performance of the logic in the Android code I added. The pseudocode of the logic which introduced included at the end of this section. And when use some famous technologies like hashtagging on the saving files, did the small research on the performance. (Regarding the sub research, included more details and the test and test result in the Appendix part of the thesis). Hence Used some JSON APIs to transfer data between the MySQL database and the selected eLearning platform.

As the main technology of file changes detection, I used “hashing”. Using android java and PHP I could generate hashtags and use them at the code level, and its use to find out the changes in the files.

3.3 Flow chart for the content comparison in file upload validation



Listing 1: this is the logic for compare two documents with the uploading

3.3 Pseudocode for the content comparison validation

```
initialize passes SET A

initialize passes SET B

If {

        find – SET A !=  SET B

        SET intersection SET A and SET B

        print the strings of passes

}

else {

        print the there are no changes

}
```

Listing 2: this is the Pseudocode for compare two contents

3.4 Summary

This chapter presented technologies for conducting our thesis. For the version controlling development, we use existing mobile and web application to apply the logic. And especially focus on the requirement, it's performance and the accuracy. And the final part of the chapter contains the pseudocode for the development of file content comparison in "Media Share", I added with the research.

Chapter 4

Approach

4.1 Introduction

Chapter 3 presented the technology to be used to solve the research problem. This chapter described our approach to address the problem of applying Version controlling mechanism into existing Learning Management Systems. And this chapter highlights the key features that distinguish this novel approach from existing approach for version controlling use in Learning Management Systems.

4.2 Hypothesis

Using the existing version controlling systems can identify what we can expect from the version control. If we follow one of major VCS, there are multiple activities absorb LMSs. If we apply the same heavy logics which are existing in the existing VC systems, then the e-learning LMSs will become more complex which we don't want to get. If we apply our own logic to take good results from the new introducing features, then we can handle them easily in our expected way.

4.3 Input

As the main input of this research applied existing key features of version controlling to Learning Management. As an example, view the history of the uploading contents, Different between the updated document and previous document. As the first step, I had to find out what exactly the users need. I did a pre-research and analyzed the results.

Within that pre-research got the idea about, the features and other requirement users need. As the features, when updating the uploading documents, has to show the changes with compare to the previous upload what exactly user has changed with new uploading. And there should be a validation. Example, if they have just changed only

the name of the document but the content is same, even that also has to notify before uploading. And user should be able to see uploading history. And has to do a proper validating before uploading files. The system should prevent unnecessary file uploading. Save data and time. And performance should be high with these new features. Without law rate performance, these new things will fail from the users.

4.4 Output

Introducing version controlling features to the existing LMSs is the main output of the research. The features should especially have filtered for LMSs. And one of another output is, introduce those logics, algorithms and methods with version controlling with an easily applicable way to the LMSs.

4.4.1 UI Changes

As the UI changes, I introduced a new button and a new UI layer for the file uploading history. And notification layers for validating all the file uploading. If a user uploads a new file for the same place in the system, the notification layer shows the changes of two files. And if the user has changed the due date of the assignment, that message also shows to the user by the system and system generate notifications via email to the students.

And Could do a file uploading validate the process. Ex: when uploading a set of documents at once, and when the user is going to upload the same set again with changing one document, the system is taking only that file to upload. It prevents unnecessary uploads.

4.4.2 Performance improvement

To the catch up the changes between two files, researcher I, didn’t use Jave predefine removeAll() function because of the law performance. And MD5 and SHA 1 used for the file tags. Didn’t move to SHA2 and other higher level SHA1 file tagging methods because of the performance issues. To prove these technical things, regarding file tagging, I did a sub research regarding the tags generating method's performance. Attached the details of the results and the research of that, in the discussion area of this thesis.

4.5 Users

The users will be all the LMS users, university lecturers, instructors, system admins and students who use LMSs for academic purpose. This will make a better connection between Lecturers and the students.

4.6 Summary

This chapter presented the main approach to this research. Within the chapter mainly discussed the input when beginning of the research and the output I expect from this research at the end of development and the target users of this features. With next chapter, we discuss the design.

Chapter 5

Design and Implementation

5.1 Introduction

Chapter 4 presented the approach that we consider with this project. This chapter described the design and Implementation we done and we have already planned.

5.2 Implementation overview

If consider the major requirements of this research, we could implement this as in build modules. I am not suggesting to use these suggested components as a library. Because we cannot make a public library which can use with all the existing LMSs. Because they are built with different technologies. As examples Moodle uses PHP, Media Share uses Java, Angular and javascript, Learning Studio uses .net.
So, that is the major reason, that I suggest with this research, introducing the logic and mechanisms which can directly follow in existing products.

5.1.1 Implement the functionalities

I had to follow the same implementation language and the same framework with these developments.

- More performance for –diff command

To the fulfil of the proposal given by me at the beginning, I had to develop a new logic which has better performance to find out the differences of two large string values.

The development, I did for the “MediaShare” mobile app I followed the android as the developing language. Android is a framework of Java.

In java removeAll(), it has n square complexity. Because, if I use an example for explain java removeAll,

Ex:- I have 2 ArrayLists A and B of the same data structure C (hashCode() and equals() overridden). C represents a student's record. The two lists are of the same size and represent new student records and old ones respectively (the students are the same in both the lists, ordering might be different). I wish to keep only those records in A that have been changed. As such, I do :

A.removeAll(B)

As per the Javadocs, this would take each record of A and compare with each record of B, and if it finds both equal, it will remove the record from A. If a record of A is not found to be equal to any record in B, and since all students in A are also in B, it means that that record of A has changed. The problem is that it easily of n square complexity.

Anyway, I introduce, another removeAll() method home build instead of Java removeAll() method and used java HashSet.

I have encountered a performance bottleneck in member removeAll in some instances For ArrayList as mentioned above, just use standard removeAll, but if A is, for instance, an EList, n^2 can be encountered.

Hence, avoid relying on hidden good properties of specific implementations of List< T > ; Set.contains() O(1) is a guarantee (if we use a HashSet and have a decent hashCode, log2(n) for TreeSet with ordering relation), use that to bound algorithmic complexity.

I use the following code that avoids useless copies; the intention is that you are scanning a data structure finding irrelevant elements you don't want and adding them to "todel".

For some reason like avoiding concurrent modifications, you are navigating a tree etc..., you cannot remove elements as you are doing this traversal. So, we cumulate them into a HashSet "todel".

In the function, we need to modify "container" in place, since it is typically an attribute of the caller, but using remove(int index) on "container" might induce a copy because of a left shift of elements. We use a copy "contents" to achieve this.

Template argument is because, during the selection process, I often get subtypes of C, but feel free to use < T > everywhere.

```
public static <T> void removeAll ( List<T> container, Set<? extends T> todel ) {
   if (todel.isEmpty())
      return;
   List<T> contents = new ArrayList<T>(container);
   container.clear();
   int torem = todel.size();
   for (T elt : contents) {
      if ( torem==0 || ! todel.contains(elt) ) {
         container.add(elt);
      } else {
         torem--;
      }
   }
}
```

Listing 3: this algorithm gets two set of words as the input and generate differences of them

MD5 and SHA for track the changed file while uploading [4]

To the implementation of file uploading research, I used Hashtagging techniques. SHA1sum is a computer program that calculates and verifies SHA-1 hashes. It is

commonly used to verify the integrity of files. It is installed by default in most Unix-like operating systems. Variants include shasum, sha224sum, sha256sum, sha384sum and sha512sum, which use a specific SHA-2 hash function, and sha3sum.

The MD5 algorithm is a widely used hash function producing a 128-bit hash value. Although MD5 was initially designed to be used as a cryptographic hash function, it has been found to suffer from extensive vulnerabilities. It can still be used as a checksum to verify data integrity, but only against unintentional corruption.

A cryptographic hash function is a hash function which takes an input (or 'message') and returns a fixed-size alphanumeric string. The string is called the 'hash value', 'message digest', 'digital fingerprint', 'digest' or 'checksum'. [5]

Although SHA slower than MD5, this larger digest size makes it stronger against brute force attacks. MD5: MD5 was developed by Professor Ronald L. Rivest in 1994. Its 128 bit (16 byte) message digest makes it a faster implementation than SHA-1. This means that MD5 executes faster but is less secure than SHA1. [5]

And also I did a small sub research on MD5 and SHA. I wanted to see the outputs of MD5 and SHA comparison to the speed of MD5. The sub research done with samples documents and files use by lecturers who use LMSs.

From that sub research, I found MD5 gives 100% accurate output for uploading documents which can use in our suggesting components. And it is faster than SHA. SHA also give 100% correct and unique tags. But it is slower than MD5.

So decided we can use both ways, but when to consider timing I suggest to use MD5 hashing techniques here. The samples of the documents lecturers use are acceptable for MD5. And the files student's uploading can change time to time and it is depending on the lecturer's questions. So I suggest we can use SHA method in student's file uploading sections.

Anyway, if the system owners suggest they don't want to consider the time when uploading a document and there is a risk to generate equal tags when generating tags, they can use SHA hashing here.

To prove these things, I use MD5 hashing in lecturer's features and SHA1 in student's interfaces.

Also did this uploading research with the Moodle web application which has own PHP framework.

Used PHP code for reading the files and generate hash tag for each file. Then store the tag with the file name in MySQL database tables. When the user is uploading the same

file set with small changes the system's algorithm detect only the changed files and select them to upload. Exclude already uploaded files which have no change.

5.2 Implement the Features.

5.2.1 Track the changes of strings in lecturer's panel.

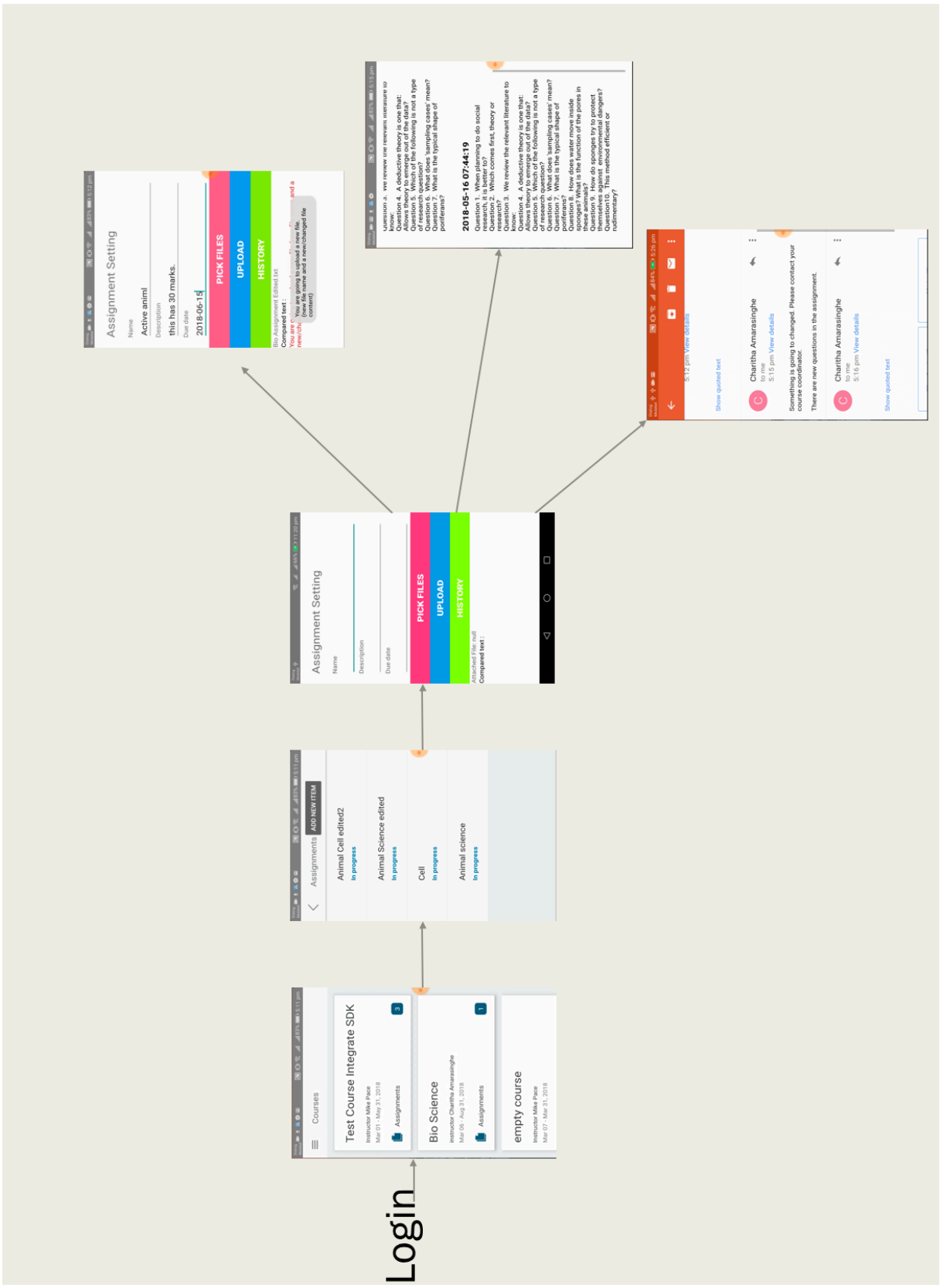
This implementation is done with the "Mediashare 1.1" android application. This helps to track the changes of the lecturer or the instructor. When they are changing the things they uploaded previously, the system will detect the changes and before uploading the changes will show to the user who is uploading.

And also here, with this research, I checked some scenarios when the user/instructor/lecturer update his previous commit when submitting assignments.

Give notifications for following scenarios.

- Only change the content of the file, no changes in the file name.
- Only change the file name of the file, no changes in file content.
- Change the title and the content.
- Updated the content with adding more lines to existing file.
- Updated the content with removing some lines.
- Updated the lines with adding new lines and removing some lines.
- Updated the content's line by changing the positions.

The architect of the main system, should not be changed. Because with the research I suggest only logic, algorithms and mechanisms how we can apply the version controlling. The final outcome should be with the existing systems. The designs I suggest after researching on both versions controlling and existing LMSs, all the suggestion should tally with the existing LMSs.



5.1 figure System diagram for Media Share integration

System detects those changes and alerting to the user what he is updating. Then this will help to reduce the mistakes while updating the assignment or Notice submitting. For the detection of these changes in the file content, I use MD5 hash tag techniques and found and used a logic which founded during the research.

5.2.2 Media Share android integrations involve with existing UIs

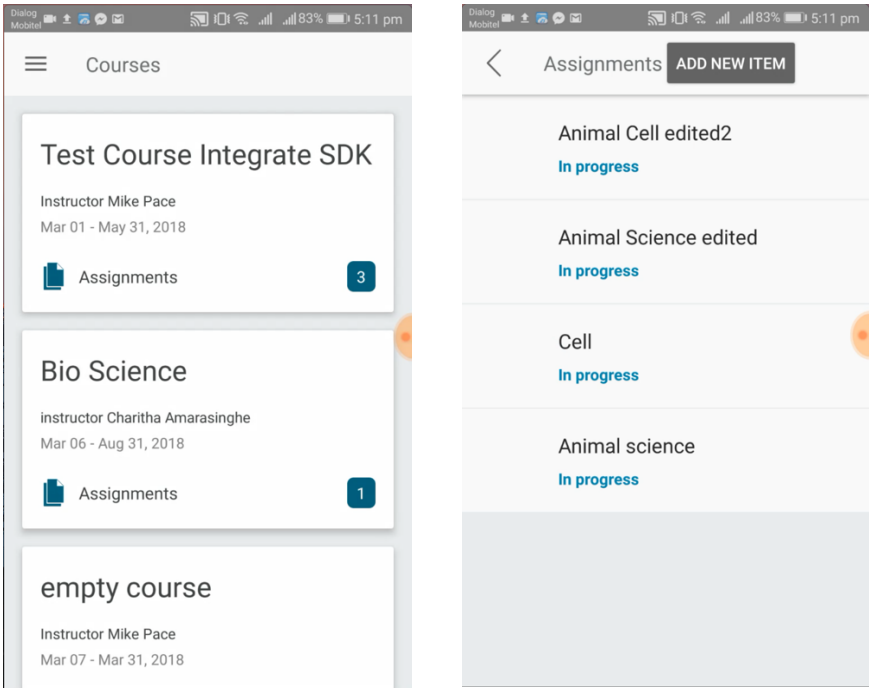


figure 5.2 – Selecting a course and an assignment. (this is from existing application and keep them as it is).

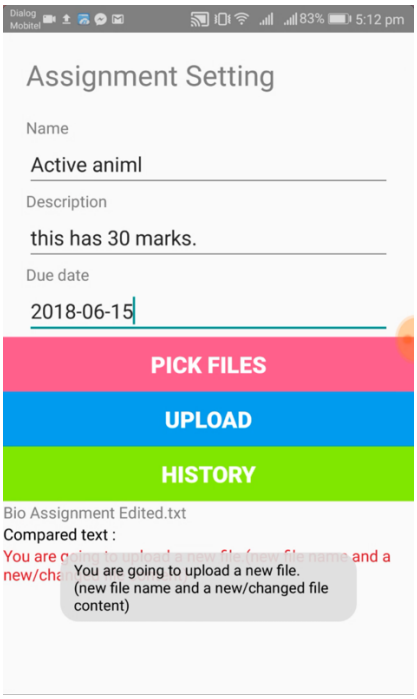


Figure 5.3 – suggested features in assignment creation area. When uploading a new file, the system gives alert, informing that.

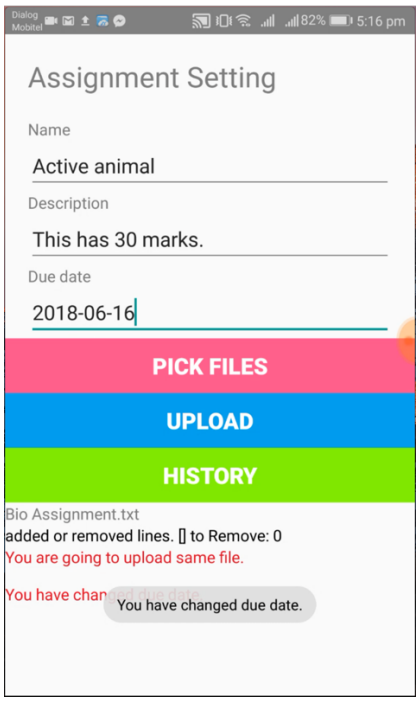


Figure 5.4 – suggested features in assignment creation area. When change the due date of the assignment, system alerting.

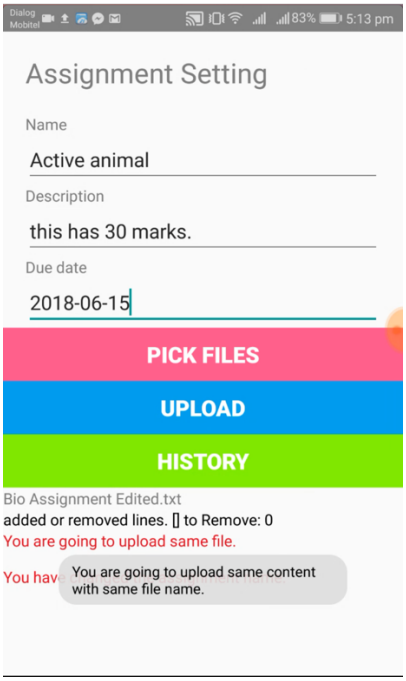


Figure 5.5 – suggested features in assignment creation area. When uploading the same file again, the system gives an alert, informing that.

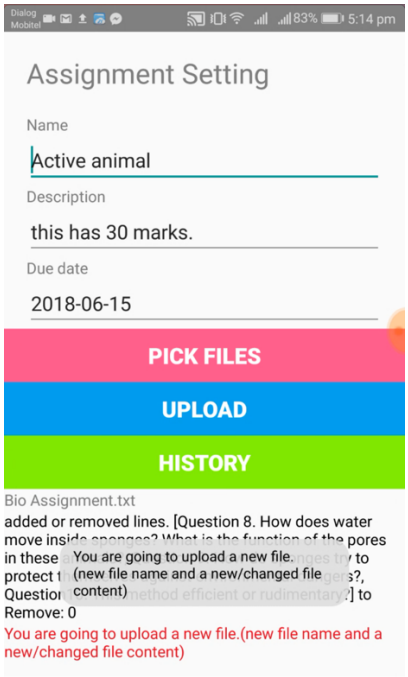


Figure 5.6 – suggested features in assignment creation area. When uploading a new file replacing the existing one, the system gives an alert, informing that.

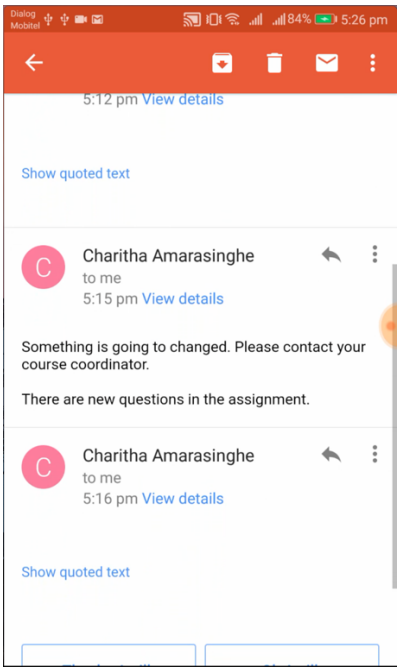


Figure 5.7 – suggested features in assignment creation area. Emailing the changes to the users/students

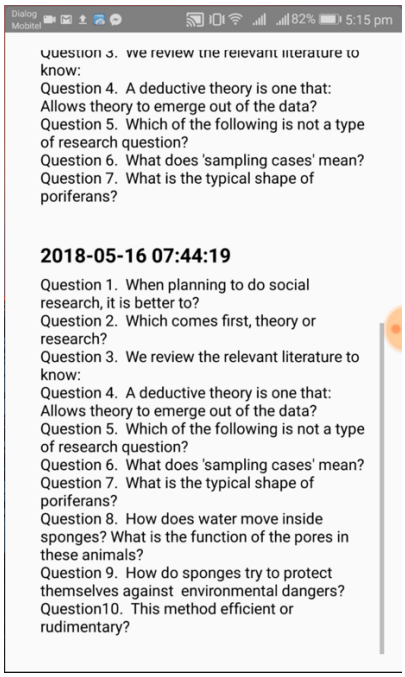


Figure 5.8– suggested features in assignment creation area. Users can see the history of the file uploaded.

5.2.3 API for the data retrieving

This part is not containing any research techniques. But it is needful. For the store data in MySQL database and those data should be pass to the application. For do that I implemented PHP base APIs. One of those APIs, directly involves with taking the versioning history of the changes.

5.2.4 File validates when student uploading multiple files at once.

With this scenario, the system tracks the files which are uploading by the students. When students try to upload the same file again and again mistakenly, system auto-detect that. As a valid real-world scenario, which I identify with pre researching steps, a university student mentioned, sometimes they do re-uploading again and again to same assignment. And if they have the bunch of files to upload, and he just changed

one file and truly he needs to upload only that changed file again, but he gives to upload the whole bunch of files. This time within current LMSs, uploading again the whole set. With this research implementation, the suggested feature is tracking all the files. It detects the changed files only. Only those files set to upload.

With this file detection feature, I suggest using SHA hashing. Because in that case students not consider timing when they upload compare to lectures. I needed to prove, we can use both scenarios SHA1 and MD5. But if some scenarios especially, consider efficiency we can use MD5.

5.3 Multiple file uploading feature - Integrate of "Moodle 3.5" web application.

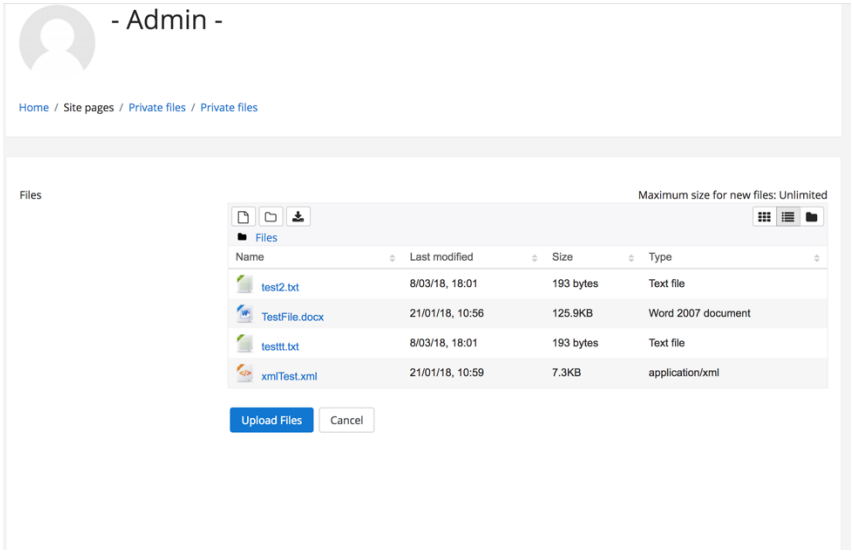


Figure 5.9 – The existing file uploading is in the Moodle 3.5 web application

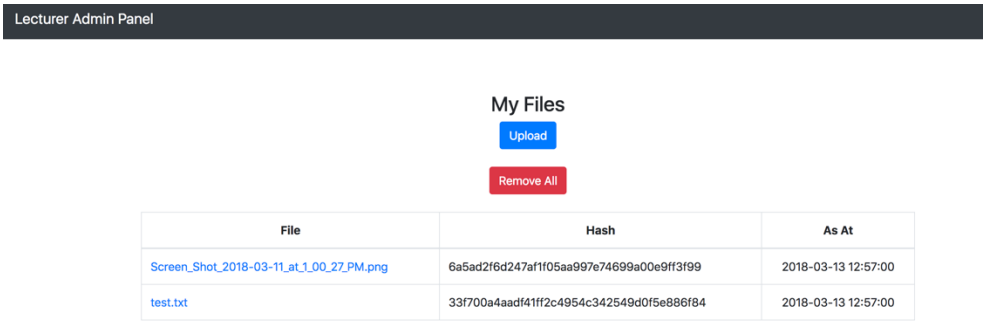


Figure 5.10 – Users can see what are the files he has already uploaded.

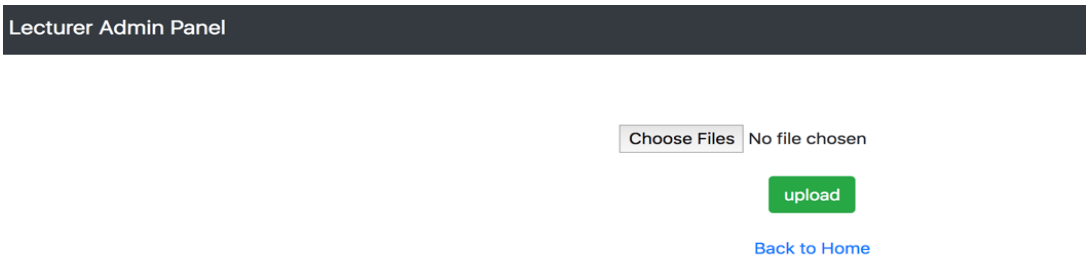


Figure 5.11– Suggested file uploading area in the app.

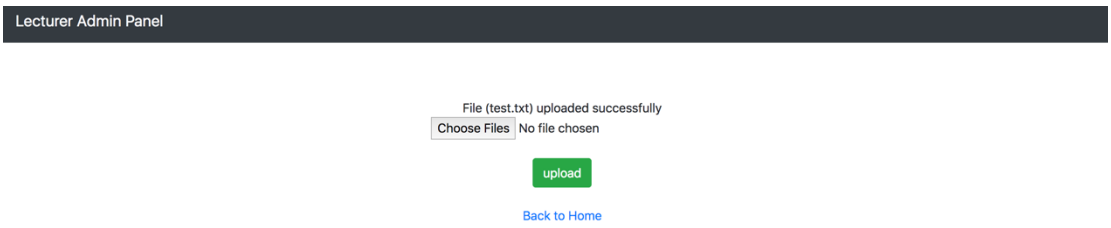


Figure 5.12 – When user uploading new files, system take them without no doubts.

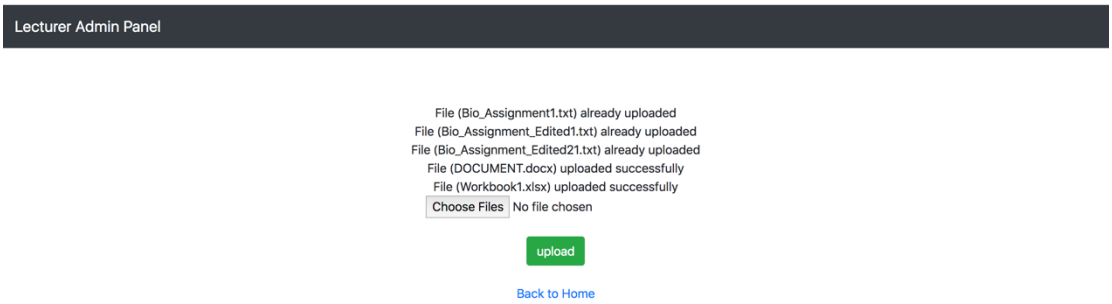


Figure 5.13 – If the user uploads the same file again, system detects that and not upload, reject the uploading, select only new files and inform it to the users.

5.4 Summary

This section presented the summary of what I have done as implementation and design. As the major implementation, I had to maintain a MySQL database and APIs which are retrieving those details from that database. These developments developed with the existing applications like ‘moodle’, ‘media share' ext. Moodle base development is done by the tally with PHP Moodle framework. The mediashare application is an Android based mobile app, all the algorithms developed using the Java. And I included the main UI's images of my designs in this chapter.

Chapter 7

Evaluation

7.1 Introduction

Chapter 6 presented the Implementations that we consider with this project. This chapter describes how testing strategies carried out for the research sub-question in terms of the evaluations results for the suggesting features in the Learning Management Systems.

7.2 Evaluation

This part of the document has divided into some areas. First I have mentioned how far I could fulfill the major requirements, using technical methodologies. Secondly I focused, how are the users take this new features to their existing app and how far it success.

7.2.1 Performance of the new features

For this I could calculate the time for various scenarios in the target app. As I mentioned in the technology chapter I used MySql as the database for this suggesting features. It is a popular database technique everybody can take as a neutral line. Ex: If MySql response speed is ‘x’ per second, LMS Blackboard use mongo DB we can guess how it would be.

API	Response time for 100 words document	Response time for 1000 words document	Response time for 10000 words document
api/assignment/create	623ms	1368ms	1971ms
api/assignment/update/{}	636ms	1172ms	2198ms

Table 7.1 evaluate performance for Introducing system

And after summarize the suggesting API results, I took existing feature's performance by calculating existing function's performance. Ex: for Moodle 3.5 file uploading function, taking this much of response time.

File uploading in..	Response time for 100 words document	Response time for 1000 words document	Response time for 10000 words document
Moodle 3.5	2280ms	3730ms	6440ms
Media Share 1.1.0	2430ms	3890ms	6710ms

Table 7.2 evaluate performance for existing system

After compare those two tables, we can come to the conclusion, the suggesting features and API structure will not be a performance wise problem to the existing systems.

7.2.2 Usability Scaling

For evaluation, I did a survey among the main users of LMSs. I ask 10 System Usability Scale questions from the target user base, lecturers and students. It was easy to find users who used LMSs in their lifetime or who is using LMSs his/her day today life cycle (students and lecturers). They have real feelings what they really need as version controlling in LMSs.

The Usability Scale questions I asked from the users. Users could answer like -> Strongly Agree (5 point), Agree (4 point), Neutral (3 point), Disagree (2 point) and Strongly Disagree (1 point)

1. I think that I would like to use this features frequently.

2. I found this version controlling features make the system unnecessarily complex.

3. I thought the new version controlling features was easy to use.

4. I think that I would need the support of a technical person to be able to use this system.

5. I found the various functions in this new Version Controlling features were well integrated.

6. I thought there was too much inconsistency in this system now. (After integrating Version controlling)

7. I would imagine that most lectures, students and other users would learn to use this new features very quickly.

8. I found the Learning Management System with new features very cumbersome to use.

9. I felt very confident using the version controlling in Learning Management System with new features.

10. I needed to learn a lot of things before I could get going with this new features a.

Strongly Agree (5 point), Agree (4 point), Neutral (3 point), Disagree (2 point) and Strongly Disagree (1 point)

As for the calculation,

Users have ranked each of the 10 templates questions above from 1 to 5, based on their level of agreement.

- For each of the odd numbered questions, subtract 1 from the score.
- For each of the even numbered questions, subtract their value from 5.
- Take these new values which you have found, and add up the total score. Then multiply this by 2.5.

User no	Calculated value
1	80
2	57.5
3	75
4	67.5
5	60
6	57.5
7	90
8	70
9	80
10	47.5
11	80
12	70
13	77.5
Avg:	912.5 / 13 = 70.19

Table 7.3 evaluate performance for Introducing system

More details and the calculated results are included in Appendix

Calculated of the SUS value = 70.19

(The SUS value calculation steps are mentioned in the discussion part of this thesis)

According to calculated results, 70.19 is belonging to C mark in the SUS measure. It means “You are doing OK but could improve”).

As a research, the new features took Ok from the audience.

7.2.4 Summery

After evaluating the results of post-research using the lectures and students, I could get to know the new version controlling features are ok with the users. There can be more improvements in the future. And technical vice I could prove the new technology perform with expected level, and sometimes exceeding the expectations.

Chapter 8

Conclusion and Future Work

8.1 Introduction

This chapter illustrates the results which are generated from the solution and the future improvements can be done to the solution. List of additional new features has been identified and these new features are listed in the future works subsection in this chapter.

8.2 Conclusion

The existing systems can integrate the version controlling features directly with their current implementations. And the system has to change their database structures to store the contents and hashtags for track the history. It is not a big deal with modern database technologies. I followed MySQL with my demonstration purpose implementations and it makes no barriers to store data.

The UI implementations were the critical part here. If the users feel mismatch compare to existing feature's performance, it will be a huge barrier. In my research, I always try to make the good performance for introducing algorithms and logic.

Under the point of good performance, when I implement the feature of filtering the new lines, that updated by the users to uploading contents, I could develop new methods which have best performance 0(1). And when make tagging to the contents of the files, I had to find out the quickest way to make that tags. I did a sub research to find out what is the best way for making tags. I used MD5 hashtags, SHA1 and other SHA tags for the experiment. There are no doubts about the output results. Problem was, if we focus on the performance of the system and if there is a considerable difference between tags we can use the best quickest way for make tags. It was MD5. In this thesis, I included details about the sub research in discussion area and included implementation methodologies related to these techniques in implementation chapters.

And in the Design chapter mainly focused on the UI implementations. I included new mobile features and their UI with new changes to design chapter.

8.3 Future Work

As the future development, I suggest to research on version controlling for PDF documents which cannot directly access to content. And if the system can show the changes in different colours same like GIT do in presents, it would a huge help to the users who use LMSs regularly.

8.4 Summary

As of the final chapter, including the conclusion and future work sections. In the conclusion section, concluded the major points which discussed whole over the thesis. And with the Future Work, I added major findings which can take the next steps of this research. The researcher needed to add them to new ideas which can develop next.

Key Word	Explanation
VC	Version Controlling
VCS	Version Controlling System
LMS	Learning Management System
MD5	Message Digest
SHA	Security Hash Algorithm
PDF	Portable Document Format
UI	User Interfaces
API	Application Programming Interface
DB	Data Bases
SUS	System Usability Scale
UUID	Universal Unique Identifier

Reference

[1] Chumbley, Alexander. *A Version Control System for Everyone*. 2016, Chumbley, Alexander. A Version Control System for Everyone. dspace.mit.edu/bitstream/handle/1721.1/112825/1014182419-MIT.pdf?sequence=1.

[2] https://www.cumbria.ac.uk/media/university-of-cumbria-website/content-assets/public/vco/documents/recordsmanagement/VersionControl.pdf

[3] http://www.theseus.fi/bitstream/handle/10024/118000/Miettinen_Antti.pdf;jsessionid=3B791BAE631440C64FAB718E4EEBD24F?sequence=1

[4] Gupta, Piyush, and Sandeep Kumar. "A Comparative Analysis of SHA and MD5 Algorithm ."

[5] *Secure Hash Algorithms*, en.wikipedia.org/wiki/Secure_Hash_Algorithms.

[7] Weber, Sandra. "Automatic Version Control System for Distributed Software Development." *Automatic Version Control System for Distributed Software Development*, Sept. 2012.

[8] Koc, Ali, and Abdullah Uz Tansel. "A Survey of Version Control Systems ." *A Survey of Version Control Systems*.

[9] Carlsson, Emil. "Mining Git Repositories, An Introduction to Repository Mining." *Mining Git Repositories, An Introduction to Repository Mining*, 2 Aug. 2013, pp. 1–43.

[10] Git, "git/git · GitHub," GitHub, [Online]. Available: https://github.com/git/git

Appendix A - Source code to “Media Share 1.1 android”

I could complete admin and lecturers uploading features using media share code. Especially I integrated admin panel to the mobile app implementation. Here I choose a mobile application ‘Media Share’, one of famous Learning Management System for this development.

From admin panel, I could complete the assignment uploading part to the system. It works like this. The lecturer can create an assignment for the related course. In an assignment, can include assignment name, date, description and the uploading file. There is an uploading area to upload the related file.

And as for this research, I added history button to see uploaded file content.

When creating an assignment for the first time, lecturer or the course owner is uploading a document. And the system auto takes hashtag for the file content and to the file name. When he re-uploading, the system generates hashtags for new content and for the file name. And then check these hashtags only, to detect the equality.

This is the Android java logic for reading the uploading file, send the read content to generate Hashtag and compare hashtags.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == Activity.RESULT_OK) {
        if (requestCode == PICK_FILE_REQUEST) {
            if (data == null) {
                //no data present
                return;
            }


// Select the file to read.
            Uri selectedFileUri = data.getData();
            selectedFilePath = UOMFilePath.getPath(this, selectedFileUri);

            File file = new File(selectedFilePath);
            currentFileName = file.getName();

// The Content turn to Strings
            fileToString = UOMReading.readDoc(file);
```

// Pass the String to the String Comparison method.

```
            stringComparisonForInstructor(fileToString);

            Log.i(TAG, "Selected File Path:" + selectedFilePath);
            pathList.add(selectedFilePath);
            for (int i = 0; i < pathList.size(); i++) {
                if (i == pathList.size() - 1) {
                    String fileName = new File(pathList.get(i)).getName();
                    fileListName = fileListName + "  " + fileName;
                    /*notificationLabel.setText(fileListName);*/
                }
            }
            hashValueForUploadingString = mdForString(fileToString);
            if (previousAssignmentHashValues != null) {
                String[] sp = previousAssignmentHashValues.split(" ");
                fileValidating(currentFileName, assignmentBackupString,
hashValueForUploadingString, previousAssignmentHashValues);
                assignmentNameCompare();

            } else {
                String previousFileName = "";
                fileValidating(name.getText().toString(),
assignmentBackupString, hashValueForUploadingString, "");
            }

            String fileName = new File(selectedFilePath).getName();
            notificationLabel.setText(fileName););*/
        }
    }
}
```

The logic for the read Files.

```
public static String readDoc(File f) {
    String text = "";
    int read, N = 1024 * 1024;
    char[] buffer = new char[N];

    try {
        FileReader fr = new FileReader(f);
        BufferedReader br = new BufferedReader(fr);

        while (true) {
            read = br.read(buffer, 0, N);
            text += new String(buffer, 0, read);

            if (read < N) {
                break;
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }

    return text;
}
```

// The String comparison method, take the previously stored strings (This has stored as an array and take the new String which has to compare and create another array from that Strings.

```
private void stringComparisonForInstructor(String value) {
    if (savedAssignmentFileString != null) {
        parts = new ArrayList <>(Arrays.asList(value.split("\n")));
        String savedValues = savedAssignmentFileString;// this string value
should comes from the Sever
        ArrayList <String> savedParts = new
ArrayList<>(Arrays.asList(savedValues.split("\n")));

// Prepare hash sets for comparison.
        Set <String> set = new HashSet <String>(savedParts);
        Set <String> partset = new HashSet <>(parts);

        removeAll(parts, set);
    }
}
```

// Prepare hash sets for comparison.

This is a one of famous scenario (Compare two strings an give the difference between them). As an example,

“I have 2 ArrayLists A and B of the same data structure C (hashCode() and equals() overridden). C represents a student's record. The two lists are of the same size and represent new student records and old ones respectively (the students are the same in both the lists, ordering might be different). I wish to keep only those records in A that have been changed. As such, I did: A.removeAll(B)”
As per the Javadocs, this would take each record of A and compare with each record of B, and if it finds both equal, it will remove the record from A. If a record of A is not found to be equal to any record in B, and since all students in A are also in B, it means that that record of A has changed. The problem is that its easily of **n square complexity**.

I have encountered a performance bottleneck in member removeAll (above-mentioned example, defined functionality use in Java or also –dff in GIT) in some instances.

Hence, avoid relying on hidden good properties of specific implementations of List< T > ; Set. Contains() O(1) is a guarantee, use that to bound algorithmic complexity.

I use the following code that avoids useless copies; the intention is that you are scanning a data structure finding irrelevant elements you don't want and adding them to "todel".

For some reason like avoiding concurrent modifications, you are navigating a tree etc..., you cannot remove elements as you are doing this traversal. So, we cumulate them into a HashSet "todel".

In the function, we need to modify "container" in place, since it is typically an attribute of the caller, but using remove (int index) on "container" might induce a copy because of left shift of elements. We use a copy "contents" to achieve this.

Template argument is because during the selection process, I often get subtypes of C, but feel free to use < T > everywhere.

```
public static <T> void removeAll(List <T> container, Set <? extends T>
toDelete) {
    if (toDelete.isEmpty())
        return;
    List <T> contents = new ArrayList <T>(container);
    container.clear();
 // since container contains no duplicates ensure |B| max contains() operations
    int toRemove = toDelete.size();
    for (T elt : contents) {
        if (toRemove == 0 || !toDelete.contains(elt)) {
            container.add(elt);
        } else {
            toRemove--;
        }
    }
    // Show the lists
    System.out.println("First List: " + container + toRemove);
    System.out.println("Second List: " + toRemove);
    HTMLVIewer.setTextOfATextView(comparisonLabel, "added or removed lines."
+ "\n" + container.toString() + "\n\n" + "to Remove:  " + toRemove);
}
```

// Prepare hashtags for comparison.

```
public static String mdForString(String input) {
    try {
```

// MD5 hashtags for comparison.
// MD5 was developed by Professor Ronald L. Rivest in 1994. Its 128 bit (16 byte) message digest makes it a faster implementation than SHA-1. This means that MD5 executes faster.

```
        java.security.MessageDigest md =
java.security.MessageDigest.getInstance("MD5");
        byte[] array = md.digest(input.getBytes("UTF-8"));
        StringBuffer sb = new StringBuffer();
        for (int i = 0; i < array.length; i++) {
            sb.append(String.format("%02x", array[i]));
        }
```

```
            return sb.toString();
        } catch (NoSuchAlgorithmException | UnsupportedEncodingException e) {
            return null;
        }
    }
```

// Compare two tags.

```
private boolean compareTag(String uploadingFilePath, ArrayList
savedHashList) {
    String hashValueForLoaclFile = generateMdFiveFlag(uploadingFilePath);
    for (int i = 0; i < savedHashList.size(); i++) {
        if (hashValueForLoaclFile.equals(savedHashList.get(i))) {
            return true;
        }
    }
    return false;
}
```

These are the major steps to validate Lecturer's updates. The lecturer can see his mistakes or what is he going to update before he does his uploading.

Basically, I focus on the major requirement came from their side. Users need accurate and quick results on this. If the system gives valid, correct results but it is performance vice week, it is not a good move. So the researcher always focuses on the performance of the application.

Appendix B - Integrated to “Moodle3.5”

I could integrate the suggested feature using Moodle 3.5 one of famous LMS in the world. It is developed by using PHP. I could attach my development especially thinking about student's functionality. Focusing to reduce uploading mistakes.

// Here I used hashing techniques to track file changes.

No problems with the SHA1. It can use for the track more deeply and performance vice it takes more times than MD5, but I suggest this time is not critical for students parts.

```
public function do_upload()

   {

      $this->multiple_upload();

      if (count($this->upload_errors) > 0) {

         $error = array_map(function ($k, $v) {

            return "$k => $v";

         }, array_keys($this->upload_errors),
array_values($this->upload_errors));

         $this->load->view('upload_form', ['error' => implode('', $error),
'success' => '']);

      }


      if (count($this->upload_data) > 0) {

         $error = "";

         $success = "";



         foreach($this->upload_data as $item){
```

```
            $sha1_hash = sha1_file($item['full_path']);
            $count = $this->count_hash($sha1_hash);
            if($count > 0){
                $error .= "File (".$item['file_name'].") already uploaded<br>";
            }else{
                $item['sha1_hash'] = $sha1_hash;
                $this->handle_file($item);
                $success .= "File (".$item['file_name'].") uploaded
successfully<br>";
        }
         }

        $this->load->view('upload_form', ['error' => $error, 'success' =>
$success]);
    }
  }
  public function handle_file($data)
  {
    $this->db->insert('attempts', [
        'file_name' => $data['file_name'],
        'hash' => $data['sha1_hash'],
        'as_at' => date('Y-m-d H:i A')
    ]);
  }
  private function count_hash($file_hash)
  {
```

```
        $this->db->where('hash', $file_hash);
        $this->db->from('attempts');
        return $this->db->count_all_results();
    }

    public function multiple_upload()
    {
        $this->load->library('upload');
        $number_of_files_uploaded = count($_FILES['userFiles']['name']);

        for ($i = 0; $i < $number_of_files_uploaded; $i++) {
            $_FILES['userfile']['name'] = $_FILES['userFiles']['name'][$i];
            $_FILES['userfile']['type'] = $_FILES['userFiles']['type'][$i];
            $_FILES['userfile']['tmp_name'] =
$_FILES['userFiles']['tmp_name'][$i];
            $_FILES['userfile']['error'] = $_FILES['userFiles']['error'][$i];
            $_FILES['userfile']['size'] = $_FILES['userFiles']['size'][$i];
            $config = array(
              //'file_name'     => <your ouw function to generate random names>,
                'allowed_types' => 'gif|jpg|png|pdf|ppt|doc|docx|txt|xls|xlsx|pptx',
                'max_size' => 10000,
                'overwrite' => FALSE,
                'upload_path' => './uploads/'

            );
            $this->upload->initialize($config);
```

```
                if (!$this->upload->do_upload()) {

                    $this->upload_errors[$_FILES['userFiles']['name'][$i]] =
        $this->upload->display_errors();

                } else {

                    $this->upload_data[] = $this->upload->data();

                }

            }

    }
```

// Using this, the system track changes of the uploading files.

If the user changes one of his uploading files and going to upload all the files, not only the changed one, the system detects only that changed one, and select to upload only that. It gives some benefits. Reduce time-consuming, Save the internet data.

Appendix C - Calculate the speed performance of the MD5 and SHAs.

```
import java.util.UUID;
import org.apache.commons.codec.digest.DigestUtils;
import org.apache.commons.lang.time.StopWatch;
public class Test {
   private static final int TIMES = 1_000_000;
   private static final String UUID_STRING = UUID.randomUUID().toString();
   public static void main(String[] args) {
      System.out.println(generateStringToHash());
      System.out.println("MD5: " + md5());
      System.out.println("SHA-1: " + sha1());
      System.out.println("SHA-256: " + sha256());
      System.out.println("SHA-512: " + sha512());
   }
   public static long md5() {
      StopWatch watch = new StopWatch();
      watch.start();
      for (int i = 0; i < TIMES; i++) {
         DigestUtils.md5Hex(generateStringToHash());
      }
      watch.stop();
      System.out.println(DigestUtils.md5Hex(generateStringToHash()));
      return watch.getTime();
```

```
        }
        public static long sha1() {
            System.out.println(DigestUtils.sha1Hex(generateStringToHash()));
            return watch.getTime();
        }
    public static long sha256() {
System.out.println(DigestUtils.sha256Hex(generateStringToHash()));
            return watch.getTime();
        }
        public static long sha512() {
      System.out.println(DigestUtils.sha512Hex(generateStringToHash()));
            return watch.getTime();
        }
        public static String generateStringToHash() {
            return UUID.randomUUID().toString() + System.currentTimeMillis();
        }
}
```

Several measurements were done. Two groups – one with smaller length string to hash and one with longer. Each group had following variations of generateStringToHash() method:

cached UUID–no extra time should be consumed
cached UUID + current system time – in this case, time is consumed to get system time
new UUID + current system time – in this case, time is consumed for generating the UUID and to get system time

5 Raw results

Five measurements were made for each case an average value calculated. Time is in milliseconds per 1 000 000 calculations. The system is 64 bits Windows 10 with 1 core Intel i7 2.60GHz and 16GB RAM.

generateStringToHash() with: return UUID_STRING;
Data to encode is ~36 characters in length (f5cdcda7-d873-455f-9902-dc9c7894bee0). UUID is cached and time stamp is not taken. No additional time is wasted.

Hash	#1 (ms)	#2 (ms)	#3 (ms)	#4 (ms)	#5 (ms)	Average per 1M (ms)
MD5	649	623	621	624	620	627.4
SHA-1	608	588	630	600	594	604
SHA-256	746	724	741	720	758	737.8
SHA-512	1073	1055	1050	1052	1052	1056.4

Table appendix 1 hashing performance part 1

generateStringToHash() with: return UUID_STRING + System.currentTimeMillis();

Data to encode is ~49 characters in length (*aa096640-21d6-4f44-9c49-4115d3fa69381468217419114*). UUID is cached.

Hash	#1 (ms)	#2 (ms)	#3 (ms)	#4 (ms)	#5 (ms)	Average per 1M (ms)
MD5	751	789	745	806	737	765.6
SHA-1	768	765	694	763	751	748.2
SHA-256	842	876	848	839	850	851
SHA-512	1161	1152	1164	1154	1163	1158.8

Table appendix 2 hashing performance part 2

generateStringToHash() with: return UUID.randomUUID().toString() + System.currentTimeMillis();
Data to encode is ~49 characters in length (*1af4a3e1-1d92-40e7-8a74-7bb7394211e01468216765464*).

New UUID is generated on each calculation so time for its generation is included in total time.

4	#1 (ms)	#2 (ms)	#3 (ms)	#4 (ms)	#5 (ms)	Average per 1M (ms)
MD5	1505	1471	1518	1463	1487	1488.8
SHA-1	1333	1309	1323	1326	1334	1325
SHA-256	1505	1496	1507	1498	1516	1504.4
SHA-512	1834	1827	1833	1836	1857	1837.4

Table appendix 3 hashing performance part 3

generateStringToHash() with: return UUID_STRING + UUID_STRING;

Data to encode is ~72 characters in length (*57149cb6-991c-4ffd-9c98-*

d823ee8a61f757149cb6-991c-4ffd-9c98-d823ee8a61f7). UUID is cached and time stamp is not taken. No additional time is wasted.

Hash	#1 (ms)	#2 (ms)	#3 (ms)	#4 (ms)	#5 (ms)	Average per 1M (ms)
MD5	856	824	876	811	828	839
SHA-1	921	896	970	904	893	916.8
SHA-256	1145	1137	1241	1141	1177	1168.2
SHA-512	1133	1131	1116	1102	1110	1118.4

Table appendix 4 hashing performance part 4

generateStringToHash() with: return UUID_STRING + UUID_STRING + System.currentTimeMillis();

Data to encode is ~85 characters in length (*759529c5-1f57-4167-b289-899c163c775e759529c5-1f57-4167-b289-899c163c775e1468218673060*). UUID is cached.

Hash	#1 (ms)	#2 (ms)	#3 (ms)	#4 (ms)	#5 (ms)	Average per 1M (ms)
MD5	1029	1035	1034	1012	1037	1029.4
SHA-1	1008	1016	1027	1007	990	1009.6
SHA-256	1254	1249	1290	1259	1248	1260
SHA-512	1228	1221	1232	1230	1226	1227.4

Table appendix 5 hashing performance part 5

generateStringToHash() with: final String randomUuid = UUID.randomUUID().toString();

return randomUuid + randomUuid + System.currentTimeMillis();

Data to encode is ~85 characters in length (2734b31f-16db-4eba-afd5-121d0670ffa72734b31f-16db-4eba-afd5-121d0670ffa71468217683040). New UUID is generated on each calculation so time for its generation is included in total time.

Hash	#1 (ms)	#2 (ms)	#3 (ms)	#4 (ms)	#5 (ms)	Average per 1M (ms)
MD5	1753	1757	1739	1751	1691	1738.2
SHA-1	1634	1634	1627	1634	1633	1632.4
SHA-256	1962	1956	1988	1988	1924	1963.6
SHA-512	1909	1946	1936	1929	1895	1923

Table appendix 6 hashing performance part 6

4.1 Aggregated results

Results from all iterations are aggregated and compared in the table below. There are 6 main cases. They are listed below and referenced in the table below:

Case 1 – 36 characters length string, UUID is cached

Case 2 – 49 characters length string, UUID is cached and system time stamp is calculated each iteration

Case 3 – 49 characters length string, new UUID is generated on each iteration and system time stamp is calculated each iteration

Case 4 – 72 characters length string, UUID is cached

Case 5 – 85 characters length string, UUID is cached and system time stamp is calculated each iteration

Case 6 – 85 characters length string, new UUID is generated on each iteration and system time stamp is calculated each iteration

All times below are per 1 000 000 calculations:

Hash	Case 1 (ms)	Case 2 (ms)	Case 3 (ms)	Case 4 (ms)	Case 5 (ms)	Case 6 (ms)
MD5	627.4	765.6	1488.8	839	1029.4	1738.2
SHA-1	604	748.2	1325	916.8	1009.6	1632.4
SHA-256	737.8	851	1504.4	1168.2	1260	1963.6
SHA-512	1056.4	1158.8	1837.4	1118.4	1227.4	1923

Table appendix 7 hashing performance part 7

Compare results

Some conclusions of the results based on two cases with short string (36 and 49 chars) and longer string (72 and 85 chars).

SHA-256 is faster with 31% than SHA-512 only when hashing small strings. When the string is longer SHA-512 is faster with 2.9%.

Time to get system time stamp is ~121.6 ms per 1M iterations.

Time to generate UUID is ~670.4 ms per 1M iterations.

SHA-1 is fastest hashing function with ~587.9 ms per 1M operations for short strings and 881.7 ms per 1M for longer strings.

MD5 is 7.6% slower than SHA-1 for short strings and 1.3% for longer strings.

SHA-256 is 15.5% slower than SHA-1 for short strings and 23.4% for longer strings.

SHA-512 is 51.7% slower that SHA-1 for short strings and 20% for longer.

Hash sizes

Important data to consider is hash size that is produced by each function:

MD5 produces 32 chars hash – *5f3a47d4c0f703c5d83265c3669f95e6*
SHA-1 produces 40 chars hash – *2c5a70165585bd4409aedeea289628fa6074e17e*
SHA-256 produces 64 chars hash –
b6ba4d0a53ddc447b25cb32b154c47f33770d479869be794ccc94dffa1698cd0
SHA-512 produces 128 chars hash –
54cdb8ee95fa7264b7eca84766ecccde7fd9e3e00c8b8bf518e9fcff52ad061ad28cae49ec3a09144ee8f342666462743718b5a73215bee373ed6f3120d30351

Purpose of use

In specific case this research was made for hashed string will be passed as API request. It is constructed from API Key + Secret Key + current time in seconds. So if API Key is something like 15-20 chars, Secret Key is 10-15 chars and time is 10 chars, total length of string to hash is 35-45 chars. Since it is being passed as request param it is better to be as short as possible.

Select hash function

Based on all data so far SHA-256 is selected. It is from secure SHA-2 family. It is much faster than SHA-512 with shorter stings and it produces 64 chars hash.

The conclusion of the sub research

The current post gives a comparison of MD5, SHA-1, SHA-256 and SHA-512 cryptographic hash functions. Important is that comparison is very dependant on specific implementation (Apache Commons Codec), the specific purpose of use (generate a secure token to be sent with API call). It is good MD5 and SHA-1 to be avoided as they are compromised and not secure. If their speed for given context is several times faster than secure SHA-2 ones and security is not that much important they can be chosen though. When choosing cryptographic hash function everything is up to a context of usage and benchmark tests for this context is needed.

Appendix D – SUS Calculation.

To calculate a score between 0 and 100 for the product:

1. Convert SUS responses to numbers, 1 for “Strongly Disagree”, and 5 for “Strongly Agree”.
2. For odd-numbered questions, subtract 1 from the response.
3. For even-numbered questions, subtract the response from 5.
4. Add the scores from each question and multiply the total by 2.5.
5. Remember to present the numbers as a SUS score, not a percentage.

If there are a small numbers of participants, consider calculating a confidence interval around the SUS score. This can help you understand the variability.

Here’s an overview of how the scores should measure:

80.3 or higher is an A. People love your site and will recommend it to their friends

68 or thereabouts gets you a C. You’re doing OK but could improve

51 or under gets you a big fat F. Make usability your priority now and fix this fast.

Table appendix 8 evaluate SUS calculation for Introducing system

SUS CALCULATION FOR THE RESEARCH AFETR DONE THE SUGGESTED FEATURES - VERSION CONTROLLING LOGICS APPLY IN LEARNING MANAGEMENT SYSTEMS													
Question No	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Sum**		**multiply by 2.5**
2nd Step	3	4	4	4	4	4	3	3	3	0	32	2.5	80
1st Step	4	1	5	1	5	1	4	2	4	5			
	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Disagree	Agree	Strongly Disagree			
2nd Step	3	2	3	2	2	2	2	3	2	2	23	2.5	57.5
1st Step	4	3	4	3	3	3	3	2	3	3			
	Agree	Neutral	Agree	Neutral	Neutral	Neutral	Neutral	Disagree	Neutral	Neutral			
2nd Step	3	3	3	3	3	3	3	3	3	3	30	2.5	75
1st Step	4	2	4	2	4	2	4	2	4	2			
	Agree	Disagree	Agree	Disagree	Agree	Disagree	Agree	Disagree	Agree	Disagree			
2nd Step	3	3	3	2	3	3	2	3	3	2	27	2.5	67.5
1st Step	4	2	4	3	4	2	3	2	4	3			
	Agree	Disagree	Agree	Neutral	Agree	Disagree	Neutral	Disagree	Agree	Neutral			
2nd Step	3	3	2	0	3	2	3	3	3	2	24	2.5	60
1st Step	4	2	3	5	4	3	4	2	4	3			
	Agree	Disagree	Neutral	Strongly Agree	Agree	Neutral	Agree	Disagree	Agree	Neutral			
2nd Step	3	2	3	4	3	1	1	2	3	1	23	2.5	57.5
1st Step	4	3	4	5	4	4	2	3	4	4			
	Agree	Neutral	Agree	Strongly Agree	Agree	Agree	Disagree	Neutral	Agree	Agree			
2nd Step	4	3	3	3	4	4	4	4	4	3	36	2.5	90
1st Step	5	2	4	2	5	1	5	1	5	2			
	Strongly Agre	Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree			
2nd Step	3	3	3	2	3	2	3	3	3	3	28	2.5	70
1st Step	4	2	4	3	4	3	4	2	4	2			
	Agree	Disagree	Agree	Neutral	Agree	Neutral	Agree	Disagree	Agree	Disagree			
2nd Step	3	3	3	3	3	3	4	4	3	3	32	2.5	80
1st Step	4	2	4	2	4	2	5	1	4	2			
	Agree	Disagree	Agree	Disagree	Agree	Disagree	Strongly Agree	Strongly Disagree	Agree	Disagree			
2nd Step	3	2	3	0	3	1	1	2	3	1	19	2.5	47.5
1st Step	4	3	4	5	4	4	2	3	4	4			
	Agree	Neutral	Agree	Strongly Agree	Agree	Agree	Disagree	Neutral	Agree	Agree			
2nd Step	3	3	3	3	3	3	4	4	3	3	32	2.5	80
1st Step	4	2	4	2	4	2	5	1	4	2			
	Agree	Disagree	Agree	Disagree	Agree	Disagree	Strongly Agree	Strongly Disagree	Agree	Disagree			
2nd Step	4	4	3	0	3	2	3	2	4	3	28	2.5	70
1st Step	5	1	4	5	4	3	4	3	5	2			
	Strongly Agre	Stongly Disagree	Agree	Strongly Agree	Agree	Disagree	Agree	Nutral	Strongly Agree	Disagree			
2nd Step	4	3	3	3	3	3	3	2	4	3	31	2.5	77.5
1st Step	5	2	4	2	4	2	4	3	5	2			
	Strongly Agre	Disagree	Agree	Strongly Disagree	Agree	Disagree	Agree	Nutral	Strongly Agree	Disagree			
				Strongly Agree (5 point), Agree (4 point), Neutral (3 point), Disagree (2 point) and Strongly Disagree (1 point)									912.5
													912.5 / 13
													70.19