2017

Academic integrity violation detection for WVU CS101

Lakshman Chandrasekaran

Academic Integrity Violation Detection for WVU CS101

Lakshman Chandrasekaran

Problem Report submitted
to the Statler College of Engineering and Mineral Resources
at West Virginia University

in partial fulfillment of the requirements for the degree of

Master of Science in
Computer Science

Dr. Elaine Eschen, Ph.D., Chair

Dr. Alan Barnes, Ph.D.

Dr. Afzel Noore, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown WV

2017

Keywords: Computer Science 101, Plagiarism, Academic Integrity Violation, Microsoft Office, Microsoft Access, Microsoft Excel, Microsoft Word, Microsoft PowerPoint, DAO, OpenXML 2.6, Umbraco, Database, String Tokens

Abstract

Academic Integrity Violation Detection for WVU CS101

Lakshman Chandrasekaran

We developed a software system to detect academic integrity violations within the course of Computer Science 101 taught at West Virginia University. In this course students are instructed on how to use the Microsoft Office suite of applications, including Access, Excel, PowerPoint, and Word for data analysis.

Academic integrity is taken seriously at West Virginia University. The current system of checking for academic integrity violations has been effective, but there is room for improvement. The new software system for detecting academic integrity violations is user-friendly, computationally efficient, and time-saving for instructors of Computer Science 101. The underlying idea is to tag each file with a hidden user-specific identification. The software system creates unique starter files which identify the student to whom it was assigned. Unique string tokens are generated for each student and saved in a database. These tokens are then injected into the starter files in concealed locations. The tokens facilitate checking for violations at the content level, in addition to the existing file-level detection. When completed assignments are submitted by the student, the software extracts these unique string tokens to match with ones already in the database to detect violations.

Acknowledgements

There have been a lot of highly influential people in my life. This is in no way a comprehensive list.

To my first teacher of Computer Science who introduced me to the world of programming during my 10th grade.

To the Department of Computer Engineering at Sardar Vallabhbhai National Institute of Technology, Surat India for molding me into a Computer Engineer.

To Mr. Jitendra Wase, Mr. M M Prasad, Mr. P Arunkumar and many other people managers at HP Globalsoft Limited for giving me opportunities to learn and grow during my 8-year stint there.

To my brother Sriram for introducing me to computer games like Wolf and Prince of Persia. The fascination of playing these games is something that I will take myself to the grave.

To my mom and dad for bringing me into this world.

To my wife and two cute kids for putting up with my endless chatting.

Contents

List of figures

List of tables

List of acronyms

RDBMS: Relational Database Management System

XML: Extended Markup Language

HTML: Hypertext Markup Language

VBA: Visual Basic for Applications

DAO: Data Access Object

PDF: Portable Document Format

CMS: Content Management System

MVC: Model View Controller

1. Introduction

The Lane Department of Computer Science and Electrical Engineering belonging to the Benjamin M Statler College of Engineering and Mineral Resources at West Virginia University is currently offering a course CS101 – Introduction to Computer Applications to students of the university. The course is designed to teach students data analysis and problem solving skills using the Microsoft Office Suite of applications viz. Excel, Access, PowerPoint, and Word. Every semester, it is offered as a GEC and GEF compliant course for students from non-CS majors.

The instructors of this course are current graduate students of the department who are selected through interviews. These interviews consider technical knowledge, teaching style, language skills, and classroom presence among other factors. Each instructor is expected to teach two sections of CS101 with a maximum of 48 students each. Each section typically meets for 50 minutes twice a week.

The first half of the course targets Microsoft Excel with students completing three homework assignments based on the material covered. Each of these homework assignments offer instructions and a dataset that is to be used for the execution of the homework assignment. The students' ability to carry out the instructions is then measured by instructors who grade these assignments. The second half of the course targets Microsoft Access, Microsoft Word and Microsoft PowerPoint. Microsoft Access has two homework assignments based on it while the last homework assignment is based on a combination of all the four.

While most students choose to work honestly, there are some students who attempt to be dishonest by submitting others' work as their own. This is an academic integrity violation as outlined by the course policy and the student conduct code at the university. The course statistics show these students to be about 5% of the total students enrolled each semester. The responsibility of detecting and prosecuting these violations lies on the part of the instructor.

Currently, the method followed by instructors to detect these cases is based on the idea that when students attempt to cheat, they obtain the file from someone else and make minor changes on it before passing it off as their own. In such cases, the file creation timestamps (available from file properties) are identical. In addition to this, information about the author and company to which the version of Microsoft Office is registered are injected into the file when it is created for the first time by a Microsoft Office application. So, when a student copies a file as it is from a different student, these file properties are also copied. Instructors consider these file properties to decide about a possible violation when submitted homework assignments are graded.

The current method of detecting possible academic integrity violations relies heavily on the file properties as outlined above. This method does not consider instances, where the

attempt to copy work was made at the content level (a student copies content from another student's file into their own file). This is the challenge and motivation behind this project. This software system can detect academic integrity violations that involve plagiarism at the content level as well as the document level.

1.1 Motivation

The project is aimed at creating an environment that would foster increased productivity and reduced effort for future instructors of the course of CS101. For instructors of CS101, much valuable time is lost detecting academic integrity violations and prosecuting the students. This time could be spent on making the classes more enjoyable and engaged with the students so that they get a better learning experience. The software system can detect academic integrity violations efficiently and quickly thereby reducing the burden on instructors.

1.2 Contributions

This project contributes to the implementation of this software system. As part of the project, we also researched existing methods and approaches that can be used to tag a file with a specific user.

1.3 Structure of this report

The rest of this report is structured as follows:

Chapter 2 looks at the background behind the file formats used by the four applications considered for the scope of this project: Microsoft Access, Microsoft Excel, Microsoft Word and Microsoft PowerPoint.

Chapter 3 discusses the underlying problem and an approach to solve it.

Chapter 4 provides an overview of the approach and the design of the implementation.

Chapter 5 discusses the challenges encountered during implementation.

Chapter 6 discusses future work and improvements.

2. Background

This section will describe the various Microsoft Office files that are used by CS101.

2.1 Microsoft Access

Microsoft Access is a relational database management system available within the Microsoft Office suite of applications. It is an application that offers a graphical user interface to a traditional relational database management system (RDBMS) [1]. On homework assignments, students are assessed on their ability to create tables, relationships, queries, forms and reports.

2.2 Microsoft Excel

Microsoft Excel is a spreadsheet application offered by the Microsoft Office suite of applications. It is an application that supports mathematical calculations using internal formulas, graphing features and pivot tables. Students are assessed on their ability to format individual cells, create mathematical formulas, insert graphs and pivot tables on homework assignments.

2.3 Microsoft Word

Microsoft Word is a word processing application available within the Microsoft Office suite of applications. It offers features to add text, pictures, graphs, references, change themes and many more. On the homework assignment, students are assessed on their ability to format data, add graphs, pictures and references based on instructions given to them.

2.4 Microsoft PowerPoint

Microsoft PowerPoint is an application from the Microsoft Office suite for creating presentations. The application offers features to create slides based on different themes and to add pictures and graphs on these slides apart from textual data. Students are assessed on their ability to create slides, insert textual and graphical objects, and modify slide layouts while working on the homework assignment.

2.5 Microsoft Open XML

The Microsoft Open XML standards need to be mentioned on this list since it is the backbone of the file formats created with modern Microsoft Office applications. Starting with Microsoft Office 2007, files created with Excel, Word and PowerPoint are stored using Open XML. The format is based on the concept that each Office file is packaged as a zip file containing many XML documents and folders [2] [3]. XML uses HTML-like tags to identify data and metadata. The zip file contains separate files to store the information on the document. For example, there are

separate XML files for each of the sheets in an Excel workbook. There are also separate XML files for the formatting inside the workbook including cell styles, borders, number formatting etc.

3. Problem and a solution

The current instructions for homework assignments in CS101 are defined in such a way that the student is creating a new file from scratch using the desired application. The application saves the timestamp of creation, title of the document and the author of the application (based on the user name to which the application is registered) and injects these into the file properties. The company information is also injected, if it is available and defined during the installation of Microsoft Office. These four properties are useful to detect a case of academic integrity violation since a copied file will share the same timestamp of creation and possibly the company, author and title.

However, there are two practical situations when this mechanism would not work. One is when a file is created using the "Right-Click" menu in a Windows operating system. Options to create spreadsheets, word documents and presentation documents exist on this menu. By default, most Windows computers use a predefined system date/time as the timestamp of creation for such a file. For example, a Microsoft Excel workbook created in a computer running Windows 10 and Microsoft Office 2016 has a created date/time as "6/5/2015 2:17PM" when using the "Right-Click" method. The author, title and company information are not saved correctly either. The four vital file properties required to detect a violation are either blank or incorrect.

The other situation is when a student attempts to copy certain content from the file rather than the entire file itself. In these cases, the timestamp of creation, title and author differ even though violation is obvious. The current system cannot detect such a violation.

A solution to this problem is to use files that are unique to each student. But since all students start with the same set of instructions and data files and work in similar Microsoft Office environments, this is a challenge. The approach we have considered is to add user-specific file properties (document level) into the file so that the existing system of detecting violations can be effective uniformly. Similar user-specific information can also be added inside the document (content level) in such a way that it does not hinder the normal operation of the document. This will help detect violations on the content level.

The user-specific information that is added at the document level and content level cannot be explicit and open since that would result in students attempting to override the system. The string tokens need to be concealed at the document level and the content level. For the token, a unique combination of letters, numbers and special characters is used so that it is difficult for students to recreate them if they should see them. This approach is followed so that

students are discouraged from trying to remove these tokens. For a student who is trying to cheat, it will take more time to search and replace these tokens.

The motivation behind a student trying to cheat is to be considered here. Usually a student tries to cheat when the remaining time until the submission deadline is limited. So, the system has been designed in such a way that it would take more time for the student to override the system than working on the actual homework assignment itself. The motivation factor is therefore suppressed.

3.1 Underlying Idea

As discussed above, the underlying idea behind tagging a file with a user is to add a complex string of characters in a predefined location on the file and provide that as a starter file to students when they start working on an assignment. When the students work on the assignment and submit it later, the string tokens extracted from the file can be matched with the ones that were given to the student in the starter file. The location of these string tokens differ for each application since the file formats are widely different.

3.2 Microsoft Access

In Access, tokens could be placed at the document level using hidden tables and/or system tables. Tables can be made hidden by setting the hidden property, while system tables can be created by naming a table with a prefix “Usys”. A table name in Access cannot use special characters. The maximum length of a table name is 64 and with the prefix “Usys” being added to the table name to make it a system table, the length of the string token that needs to be generated is 60. This solution implements this method of creating a system table for document level tokens.

The database files created with Access have a discrete nature to them meaning that there are no continuous placeholders. This contrasts with Excel, Word and PowerPoint where there is a continuous range of cells, pages or slides, respectively. This makes it difficult to place the hidden string tokens inside the content unless the token is placed inside an existing object.

Each object that resides in an Access database has built-in properties that include the timestamp of creation and modification, and a description among many others. These three properties can be easily viewed by right-clicking on the desired object and clicking the menu option for “object properties”. These are commonly used properties for an object, but more advanced properties exist. These advanced properties can be viewed and edited while on the design view of an object, but are otherwise hidden from plain public view.

3.2.1 Microsoft Access - Tables

Here is a full list of the properties that are specific to tables within Access. As mentioned earlier, these properties can be viewed/edited from the design view of the table.

Property Name	**Type of Information**
Read Only When Disconnected	Object-specific Information
Subdatasheet Expanded	Object-specific Information
Subdatasheet Height	Object-specific Information
Orientation	Object-specific Information
Description	Textual Information
Default View	Object-specific Information
Validation Rule	Object-specific Information
Validation Text	Textual Information
Filter	Object-specific Information
Order By	Object-specific Information
Subdatasheet Name	Object-specific Information
Link Child Fields	Object-specific Information
Link Master Fields	Object-specific Information
Filter On Load	Object-specific Information
Order By On Load	Object-specific Information

Table 1: Access Table Properties

Description and Validation Text are the two property attributes that can accept text, while the others can accept only one of a designated range of values depending on the primary purpose of that property. Validation Text is a better choice for adding a string token since it can only be viewed/edited while on the design view of table. The maximum length of the string that can be set for this property is 256 characters. Our solution uses the “Validation Text” property.

3.2.2 Microsoft Access - Queries

Here is a table showing the list of properties that can be set for a query in Access.

Property Name	**Type of Information**
Description	Textual Information
Default View	Object-specific Information
Output All Fields	Object-specific Information
Top Values	Object-specific Information
Unique Values	Object-specific Information
Unique Records	Object-specific Information
Source Database	Object-specific Information
Source Connect Str	Object-specific Information
Record Locks	Object-specific Information
Recordset Type	Object-specific Information
ODBC Timeout	Object-specific Information
Filter	Object-specific Information
Order By	Object-specific Information
Max Records	Object-specific Information
Orientation	Object-specific Information
Subdatasheet Name	Object-specific Information
Link Child Fields	Object-specific Information
Link Master Fields	Object-specific Information
Subdatasheet Height	Object-specific Information
Subdatasheet Expanded	Object-specific Information

Filter On Load	Object-specific Information
Order By On Load	Object-specific Information

Table 2: Access Query Properties

For queries, Description is the only property attribute that can be set to a text value, but at the same time, it can be easily viewed/edited by a common user. Instead, the property "Subdatasheet Name" is used. This is a property that accepts the name of an existing table/query within the database. The only table/query that is unique to each student is the system table that was added as part of the document level token. Our solution uses the "Subdatasheet Name" property.

3.2.3 Microsoft Access – Other Objects

Tables and queries are not the only objects within an Access database file. Other objects such as relationships, forms and reports also exist and there is a possibility that students could potentially copy these objects from another student. However, these objects are relatively easier to create and are therefore excluded from the scope of this project.

3.3 Microsoft Excel

In the case of Microsoft Excel, individual objects do not exist, but there is a continuous range of cells in which to add data. Any approach to add string tokens into a spreadsheet must consider the primary purpose of creating on that spreadsheet. Since a spreadsheet is expected to contain user defined data in its cells, it might not be a good solution to add string tokens as the actual content of cells. However, two possible alternatives for using cells exist. One involves adding string tokens into a random collection of cells selected in the neighborhood of the last few rows and columns of a spreadsheet. This would only help with document level checking since all assignments involve adding formula and content to the first few columns and rows of the workbook. The other involves adding string tokens into specific cells with font color white such that the text appears invisible. This approach is better than the earlier one for content level tokens, but still might not yield good results since it relies on the probability of content being copied over from one of the cells where the hidden string tokens are placed. String tokens added into certain cells with a white colored font could still be used as an additional method of injecting tokens. This however is not considered for this project.

A better approach to adding string tokens in a spreadsheet is to add cell styles named using the string tokens. Cell styles can be created and defined in a Microsoft Excel workbook. The formatting behind the new cell styles created is like the default "Normal" cell style that is a part of every Excel workbook. This is done to ensure there is no difference in the appearance of the cells. These cell styles are saved inside the workbook even if they are not used on any of the cells

within the workbook. At the same time, they get created in a workbook when content is copied over from another workbook to this one. So, when a student attempts to copy content, these string tokens in the form of cell styles also get copied over. When applied to cells, cell styles can be used to detect copying at the content level. The mere presence of them inside the workbook can be used to detect copying at the document level.

This solution has been designed to add twelve unique cell styles to each file generated by the system. If content is present inside a cell, one of the cell styles is applied to it. If no content is present, the cell styles only exist in the document. The maximum length of the name of a cell style is 64 including the Office environment for Mac.

3.4 Microsoft Word

Like Microsoft Excel, text with a white colored font can be used to hide string tokens from plain view. Here is an example. This text is visible but this text is . String tokens thus added could be placed throughout the document to check for any content that might be copied over from a different document. However, this could get easily deleted when content is written or gets pasted over from a different document. Further, if string tokens are added with a white colored font, they cannot be added all over the document if the initial clean starter file has only a single page of text. Hence this might not be a good approach, but this could be used as an additional method of adding string tokens.

A better approach is to use hidden bookmarks with the unique string tokens defined as the name of the bookmark. A bookmark in a Microsoft Word document is like a hyperlink within the document. It can be used to define a location within the document that could be cross-referenced from within the document. For example, table of contents in Word uses bookmarks internally to cross-reference to different sections of the document. Bookmarks also have the option of being hidden, thus making it difficult for a naïve user to search and find the bookmark let alone delete it.

Hidden bookmarks are very useful in detecting copying at the content level. For the document level, user defined properties can be defined. The maximum length of the name of a bookmark is 40 characters. This is the case with Office for Mac also. These bookmarks have been used in this solution's design.

3.5 Microsoft PowerPoint

Like Microsoft Word and Microsoft Excel, PowerPoint is also a candidate for placing string tokens with a white colored font. Again, the risk of these getting deleted by a naïve user is high. Further these can be added only on existing slides. Assuming a single slide clean starter file to

start with, these white colored string tokens might not detect changes to all slides. So, this might not be a good approach.

A better approach is to use hidden textboxes. Textboxes can be created on a presentation slide and can be made hidden using the "Selection Pane". The names for these textboxes can be defined using string tokens or the content inside these textboxes can be the generated string token. For making the extraction process easier, it is better to add the string tokens as the names of these textboxes.

Hidden textboxes are useful in detecting copying at the document level and at the content level. The maximum length of the name of a textbox is 64. User defined properties can also be defined just like in Word. This solution's design uses the same.

3.6 Generic Approaches

A generic approach common to all the four Microsoft Office applications is to use macros. Macros are user defined scripts that are designed to exist in the actual files and are developed using VBA (Visual Basic for Applications). These macros could be designed in such a way that they would record all actions done on a specific file. This file can be offered to the student as a starter file for assignments. The main drawback with this approach is the explicit and technical nature of these macros although their implementation is comparatively easy. Students would need to explicitly turn on macros for this approach to be effective but macros are too technical for this course. Also, macros can be turned off by a student while working with a file and this causes the detection to fail. This can be tackled with an Office document that opens up correctly only when macros are enabled. But as noted earlier, this is an explicit attempt and should therefore be avoided unless this method is used in addition to an existing method.

4. Solution Design

The approaches to solve the problem for all the Microsoft Office applications are based on the creation of unique string tokens. These string tokens also need to be tied to an individual student. A database oriented design is needed because of the creation of string tokens and the need to tie them to students.

4.1 Database Design

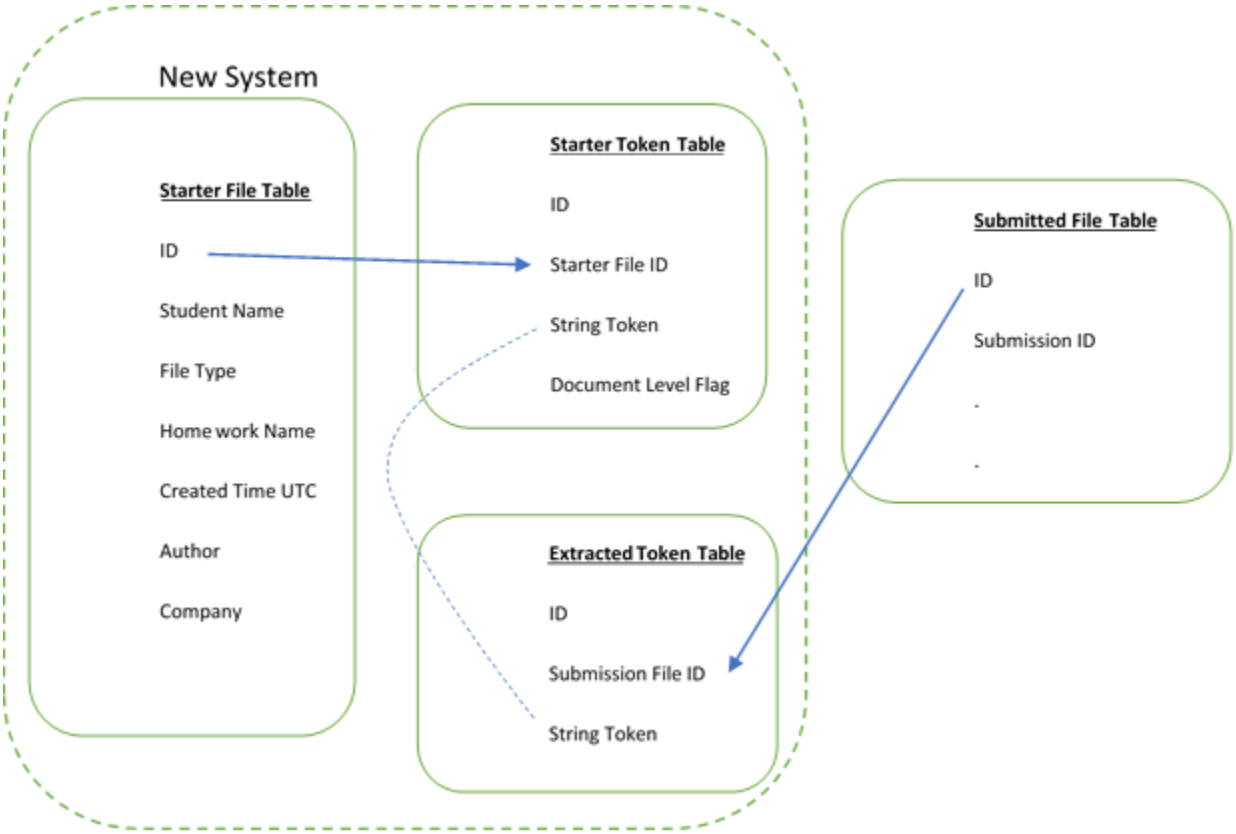

Figure 1: Database Design

The database for this solution should have the capability to directly relate a specific file and its set of string tokens with a specific student. It should also be able to retrieve the student for whom a specific token (specific file) was issued. Similarly, when a student submits the completed file, the system should be able to extract all the tokens found within the file and update that into the database. This way the tokens can be compared to see if they match. In this design, the three tables on the left are the new tables designed as part of the new system. The starter file table will have a record of each file generated through the system. So, it will have records for all students working on all homework assignments. The starter token table will have a record for each string token generated by the system. Since an assignment may contain multiple string tokens within it for the document level and the content level tokens, there is a one-to-many relationship between the starter file table and the starter token table. This is depicted on the diagram using a line. The arrow on the line depicts the "many" side of the relationship. On the submission side, the extracted token table will have a record for each string token extracted from a student's completed assignment file. The current system has a submission file table that stores the submission ID and student name among many others. Since each submitted file is bound to contain many string tokens, there is a one-to-many relationship. The line depicts this

relationship with the arrow pointing at the “many” side of the relationship. The dotted line on the diagram is not a relationship, but a pointer to show that the two fields must match on content.

4.2 System Design

4.2.1 Student logging in to access the homework assignment

With the current system, the students of CS101 are not authenticated before an assignment can be opened. The assignment for any homework typically contains two files. One is the instructions file available as a PDF and the other one is the data files available as a compressed archive.

With the proposed system, there will be a third file that will also be made available. This is where the starter file, injected with unique student specific string tokens, will be offered to the student. Since this one is specific to a student, the student will need to be authenticated into the CS101 site before this starter file is made available. Here is the process flow for the creation of the starter file expressed as a flowchart.

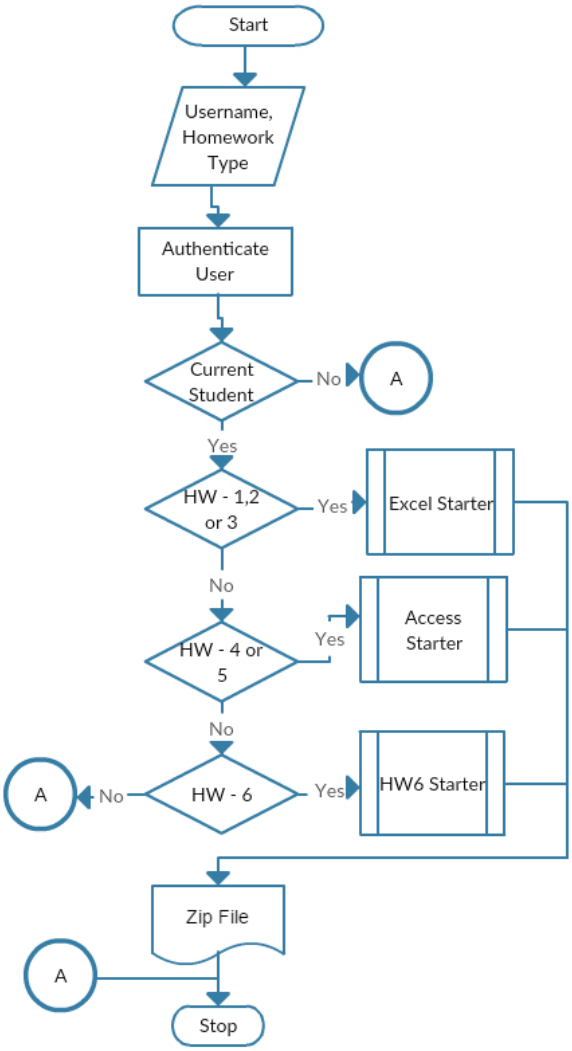

Figure 2: Flowchart for Starter File Generation

The individual sub-processes are herein expressed in pseudo-code.

<u>Excel Starter</u>

Input – username, homework type

Start

Check existing database to find the section of student

Check existing database to find the problem assigned to student's section for homework type

Check new database to find the clean starter file for the problem.

Clone the file and create a copy in temporary folder.

Open the cloned file for editing.

Set author to student name, company to “WVU CS101 Website” and title to homework type

Add twelve new cell styles named using unique string tokens into excel workbook.

Close the file for editing

Add twelve cell styles’ names into database along with student name.

Return file

Stop

Access Starter

Input – username, homework type

Start

Check existing database to find the section of student

Check existing database to find the problem assigned to student’s section for homework type

Check new database to find the clean starter file for the problem.

Clone the file and create a copy in temporary folder.

Open the cloned file for editing.

Set author to student name, company to “WVU CS101 Website” and title to homework type

Generate a string token and create a new system table using this token as the name

For all tables inside the access database file, create unique string tokens and assign them as “Validation Text” property of table.

For all queries inside the access database file, set the name of the system table created above as the “Subdatasheet Name” property of query.

Close the file for editing

In the database, update the generated tokens for the student.

Return file

Stop

Word Starter

Input – username

Start

Check existing database to find the section of student

Check existing database to find the problem assigned to student’s section for homework type

Check new database to find the clean starter file for the problem.

Clone the file and create a copy in temporary folder.

Open the cloned file for editing.

Set author to student name, company to “WVU CS101 Website” and title to homework 6

For each paragraph of text inside the document, generate a string token and use its name to define a hidden bookmark in the middle of the paragraph

Close the file for editing

In the database, update the generated tokens for the student.

Return file

Stop

PowerPoint Starter

Input – username

Start

Check existing database to find the section of student

Check existing database to find the problem assigned to student's section for homework type

Check new database to find the clean starter file for the problem.

Clone the file and create a copy in temporary folder.

Open the cloned file for editing.

Set author to student name, company to "WVU CS101 Website" and title to homework 6

For each slide inside the document, generate a string token and use its name to define a hidden textbox

Close the file for editing

In the database, update the generated tokens for the student.

Return file

Stop

HW6 Starter

Input – username

Start

Check existing database to find the section of student

Check existing database to find the problem assigned to student's section for homework type

Check new database to find the clean starter file for the problem.

Call the sub-procedures Excel Starter, Access Starter, Word Starter, PowerPoint Starter to generate separate files.

Return files

Stop

4.2.2 Student logging in to submit competed assignment

The current system of authenticating a user before submissions will continue after the implementation of the system. The existing process to retrieve the title, author and company from the file properties will continue as is. The new process to check and retrieve string tokens will kick start after that. Here is the process flow expressed in pseudo-code.

File Checker

Input – file, username

Start

Switch (file type)

If file type is excel workbook, retrieve all the user defined cell styles' names from the workbook.

If file type is access database, retrieve the string tokens from user-defined system table's entries, tables' validation text property and queries' subdatasheet name property.

If file type is word document, retrieve the string tokens from the hidden bookmarks defined in the document.

If file type is powerpoint presentation, retrieve the string tokens from hidden textboxes found on all slides on the presentation.

End Switch

In the database, update the extracted tokens for the student.

If no tokens are found, add a record of “TokensRemoved” for the student.

Stop

4.2.3 Instructors logging in to check academic integrity violations

Instructors currently detect violations using the following method. A real-time snapshot of the file properties extracted from all files in the current semester is made available. An Access database file containing information about file properties of all files submitted over the last few years is also available for instructors. The Access database file also contains queries defined on it. These queries sift through the file properties of files from the current semester and past semesters to detect a possible violation. Instructors then download the relevant students’ files and manually verify the violation. Appropriate steps are taken to prosecute the student thereafter.

With the new system, the method will be similar. The existing Access database has been enhanced with additional tables and queries so that they can work with string tokens in addition to file properties. A real-time snapshot of the information found on the starter file table, starter token table and the extracted token table will be available as a download so that instructors can import these into this Access database file and run the appropriate queries.

The queries on this database have been designed such that for each extracted token on a student’s submitted file, the starter file table and the starter token table are scanned to find which student was assigned the token. If this student is different from the student whose file is being reviewed, the query displays the record showing the name of the two students involved, the matching string tokens, the assignment information and the dates when the original starter files containing these string tokens were generated.

4.3 Technical Design

The current CS101 website is a custom ASP.NET MVC5 application. Therefore, all the components developed as part of this project were developed in C#.

The first crucial step is the creation of the string tokens since it is a step that is common to all applications. In the C# library, there is a class for defining and constructing unique numbers. This class and its methods are used to define and generate string tokens whenever they are needed.

DAO is a library designed by Microsoft for use in handling Access database files. The methods from this library are used extensively while working with the Access database starter files required for Homework assignments 4, 5 and 6. The documentation available on the web

for DAO is limited since it is an older technology. Hence, functionally working code was obtained through trial and error and extensive debugging.

Open XML 2.6 is the latest version of the Microsoft Open XML standards. Open XML is used to work with Excel, Word and PowerPoint files. The library is rich with a lot of methods and properties available to handle files from these three applications [4]. In instances, where methods were not available, the underlying XML was manipulated and the desired functionality was achieved.

4.4 Assumptions and Limitations

The system has been designed based on certain assumptions and limitations. Here are some of them.

- The primary purpose of this software system is to enable instructors to detect academic integrity violations easily and effectively. However, for the system to be effective, it is vital for students to download starter files from the system and work on them as part of the homework assignment. As part of this, instructions for all homework assignments would need to be changed to accurately reflect the need for students to download starter files from the system (rather than creating a new file as is used in the current system).
- The load on this software system is expected to be high on days when a homework assignment is due. We do not know how the system will react to these high levels of load since stress testing has not been performed on this software system. Unit testing of the various functionalities and numerous end-to-end tests were performed however.
- The software system has been designed and developed to be used by CS101. The current CS101 application is stable and so the introduction of these new components developed as part of this project might be done in a phase-wise approach.
- In the case of all the four applications, document level tokens are injected easily. However, for the content level, string tokens can be added, but for them to be effective in detecting cases of content level copying, the clean starter files for each assignment are expected have a minimum list of objects/content to start with.
- In the current system, there are no lookups into the database backend before a homework assignment is displayed to the student. With the new system, the number of database lookups completed before an assignment is available to the student is three. There are also two occasions when information is updated into the database backend. These database backend operations could introduce a lag on the CS 101 site.

- When a submitted file being checked for tokens returns with an empty list of string tokens, the system will not be able to match up string tokens with the ones that were present in the student's starter file(s). The solution has been designed to count these instances of no tokens as an explicit attempt to avoid detection.
- The solution for generating Excel files has been designed such that the unique cell styles that are added to the document are only applied to cells if content is already present. A more comprehensive approach would be to apply cell styles to all cells on the workbook. This approach was not considered for this solution since it could lead to a severe load on the server on days when a homework assignment is due.
- For the purposes of simplicity, the length of all string tokens generated for all the four types of files has been fixed at 32 with an additional constraint for Access. The string token generated for Access does not contain special characters.

5. Challenges

Most of the challenges with respect to the implementation of this project have been with respect to integrating the solution with the current CS101 website. The current CS101 website has been built using an ASP.NET MVC5 application that works with an Umbraco front end. The modules that have been built for this software system have been created in such a way that they exist as a stand-alone solution. This is to ensure the stability of the current CS101 website. Here are some of the functional requirements that were identified in the earlier stages of this project and how they were handled as a challenge.

- The software system has been designed to co-exist with the current CS101 website. The current website runs with an Umbraco 7.5.9 content management system (CMS) and is a custom ASP.NET MVC5 application on the administration side. The backend server is a Microsoft SQL Server. A separate project has been defined in the existing CS101 solution to house this software system.
- The current website was developed in C# using .NET Framework 4.5 and Entity Framework 5. This software system has also been developed in C# using both the framework technologies.
- The Umbraco CMS 7.5.9 component of CS101 took a lot of effort and time to understand the flow, methods and the actions triggered by it on the CS101 website. Special privileges were needed to get full control of it and understand its functionality.
- The full CS101 application runs smoothly on the existing production server. However, the application loaded up slowly on the desktop we used for the development of this project. This significantly slowed down testing and development.

- The original design of the software system assumed the same set of unique string tokens for a student. This meant that the same set of 10 or 12 string tokens would be used for every homework assignment by the same student. This was later changed to include unique string tokens for each time a file is generated by a student. The software system now allows a student to generate multiple starter files for the same homework assignment.
- The cloning of the clean starter file that is done as part of the file generation process has been defined such that a non-generic file name is used for the new cloned file. This ensures the smooth operation of the system when two or more students are executing the file generation process at the same time.
- The solution was originally designed to include the placeholder for the starter files for each assignment as part of the "Grades & Submissions" page of the CS101 site. This was later changed to include the placeholder on the assignments page itself.
- The starter files offered by the system to students has been made available on the current assignments page along with the instructions and data files. Assignment display templates (for homework assignments) has been changed in Umbraco to include a placeholder for the starter file.
- The clean starter files to be used for the various problems of each homework assignment are bound to be different. A table of the various problems assigned in CS101 along with the hyperlink to their clean starter files has been created for this project. This table is needed to ensure the flow of the starter file generation process. Originally the project was designed such that the instructors select the clean starter files when creating and publishing the homework assignment.
- A dropdown media type was created in Umbraco to handle the input to the starter file generation process. This dropdown would be used by instructors to select the problem that has been assigned to the instructor's section.
- The information extracted by the software system during the submission of file by the student is critical to prove/disprove a potential violation. Therefore, extreme caution has been exercised to make sure that no extra tokens are added by the software system during the extraction process. The submitted file is opened with a read-only mode so that no changes are made to the file by the system.
- A lot of CS101 students use Mac as their primary computing system. CS101 considers the Mac versions of Microsoft Office acceptable for use by students. Therefore, the choice of placeholders to inject string tokens were considered and tested successfully in a Mac environment with Microsoft Office. Of course, this is specific to Excel, Word and PowerPoint since Office for Mac does not support Access.
- The current CS101 website is stable. Therefore, changes made to the existing website have been kept minimal. The primary functions for generating new files

and analyzing submitted files have been designed to be self-contained within their own library so that the integration of these new components with the CS101 website can be smoothly completed.

- The full source code of this software system is available on GitHub and has been shared with the CS101 course coordinator.

6. Conclusion

The solution has been designed based on trial and error approaches made by the author. The emphasis of the approaches was to hide information in a Microsoft Office environment. The design of this solution is suited for the kind of assignments that are used by CS101. This solution is not meant to be a one-size-fits-all solution. Further, it assumes that most students of CS101 are beginner users of the Microsoft Office environment. Advanced users might find it easy to identify and possibly delete these string tokens. In the absence of string tokens, there would no longer be a way to uniquely tag a file with a student.

The placeholders for the starter file available on the assignments page could be changed in such a way that the required files for the assignment are compressed into a single file and offered to students. This would eliminate the need for students to download multiple files before they start working on an assignment.

The design of this solution for Access is not comprehensive like the others. This is because the file structure in Access has a discrete nature. Therefore, the approach for content level tokens was changed to include the tokens on the object level. Only tables and queries were considered for these object level tokens. This could be enhanced to include object level tokens for other objects within an Access database.

The design of the Excel solution is solely based on cell styles as discussed. Although this is a comprehensive method meaning that it will enable any sort of copying to be tagged, it is open for manipulation by students. For a future version of this software system, white colored font could also be considered as an additional method of tagging with string tokens. Further, the solution currently applies a cell style only if there is data prepopulated in the worksheet. A stronger solution would be to apply cell styles to all cells within the workbook. The design of this solution for Word could also use white colored font as an additional method of injecting string tokens into a document.

This project could be extended to create an automatic grading system that would grade students' submissions in real time.

7. References

[1] [Online]. Available: https://en.wikipedia.org.

[2] [Online]. Available: https://dev.office.com/.

[3] [Online]. Available: https://msdn.microsoft.com/en-us.

[4] [Online].Available: https://GitHub.com/OfficeDev/office-content/tree/master/en-us/OpenXMLCon.

[5] [Online]. Available: https://www.codeproject.com.

[6] [Online]. Available: http://stackoverflow.com.