The Implementation of a Filestream based English Language Learning System

Asma Tawil, Mohamed Mhereeg

Faculty of Engineering Technology
Tripoli, Libya
a.g.altawil@gmail.com

University of Tripoli
Tripoli - Libya
mmhereeg@msn.com

ABSTRACT

The paper discusses the feasibility of constructing a SQL Server FILESTREAM based English Language Learning System (ELLS). The paper focuses on the Implementation phase of the system. It explains the prospect of storing and managing unstructured data (e.g. Images, video, Word, Excel, PDF, MP3, etc) for educational purposes utilizing FILESTREAM technique provided by SQL Server 2012, and explains how to maintain efficient storage and access to BLOB data. However, Storing unstructured data posed many challenges, such as how to maintain transactional consistency between the structured and unstructured data, how to manage backup, restore, and storage performance. The paper seeks to utilize the combination of SQL Server 2012 features and the NTFS (New Technology File System) to improve the efficiency and performance of the ELLS system for children. The system also seeks to maintain the transactional consistency between the unstructured data and corresponding structured data. Furthermore, the system allows the user to customize some operations such as creating, updating and deleting photos, videos or audios in the database. The system supports some maintenance operations such as backup, restore, and consistency checking.

KEYWORDS

BLOB, ELLS, Filestream, NTFS, Unstructured Data.

1 INTRODUCTION

In recent years, there has been an explosion in the volume of digital data created and stored by both individuals and organizations [1].

Historically, businesses have used computer systems and databases to store most of their business data in structured formats such as relational tables or fixed format files, and software applications have used these structured data stores to perform business tasks[2]. Today however, a large proportion of an organization's data is typically stored in documents created with productivity tools such as Microsoft Office Excel and Microsoft Office Word, and advances in digital photography, document scanning, video production, and audio formats have further extended the range of unstructured data formats that are used for business data [1], [3]. For example, in the context of relational database systems, it refers to data that can't be stored in rows and columns. Additionally, dramatic reductions in the cost of hardware storage and memory have significantly affected the amount and type of data that is stored in computer systems, and led to the emergence of a new generation of business application that merges traditional relational data structures with unstructured digital content [3], [4].

This profusion of digital content means that organizations are now seeking to manage both relational data and unstructured data at the enterprise scale, and require a solution that comprehensively meets the needs of relational and non-relational data storage while reducing the cost of managing and building applications for that data [5].

Storing unstructured data such as text documents, images, and videos posed many challenges, such as how to maintain transactional consistency between the structured and unstructured data, how to manage backup and restore, and storage performance and scalability[6]. Architects of applications that required the storage of binary large objects (BLOB) data could either store the data in the database or store it outside of the database with a reference stored in the database [6], [7].

This paper focuses on the implementation of a SQL Server FILESTREAM based English Language Learning System (ELLS). The system is implemented taking into consideration the above mentioned issues. Thus these issues are solved throughout the development of the project. The ELLS utilizes the FILESTREAM feature provided by SQL Server 2008/2012, which allows the storage and efficient access to BLOB data using a combination of SQL Server and the NTFS (New Technology File System). The ELLS has been implemented in which these features have been applied. The system is intended to offer an excellent supporting material for in-class teaching, developing some language skills like reading, listening and speaking at a certain level. The system will also allow undertaking tests and providing automatic marking and feedback to students. This paper is based on the previous paper entitled "Analysis and Design of a Filestream based English Language Learning System"

2 IMPLEMENTATION OF "ENGLISH LANGUAGE LEARNING SYSTEM" SYSTEM

The "ELLS" system is implemented using the Windows Forms as an application platform to develop the application. Windows Forms Application can be written either in C# or VB.net. It is a common knowledge that VB.NET and C# are functionally equivalent. Hence a frequently-heard argument for choosing one over the other is because they are functionally equivalent, neither poses a clear advantage. Thus, the really important differences between the languages are not to be found in their interaction with the runtime system but in the support they offer to the programmer. In short, these differences exist in the syntax, structure and clarity. There was actually no grave issue to consider in terms of their compilation and execution efficiency or their editing, design and debugging facilities. Ultimately, we decided to use C#.

2.1 Access page

When using the "ELLS" system for the first time, the user has to register as New User by inserting the user details through the administrator (teacher) of the system, he/she will log into the system using a valid username and password. When the system validates the data from the database, the user will be directed to the Home Page depending on the type of user, otherwise message (Invalid Username /Password) will be displayed.

2.2 Login as Admin (Teacher)

Figure 1 shows the admin page after a successful login, where the administrator can add students, Levels, Questions, diverse lessons, tests for each level with answers, and edit all features. It is the page where all tasks of the system can be performed as they appear after login.

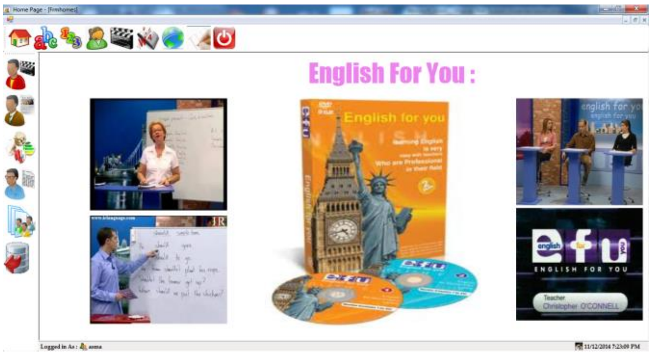


Figure 1. Login as Admin (Teacher)

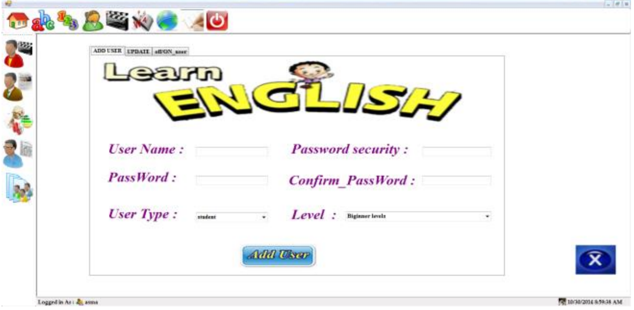
2.3 Login as a student

Figure 2 displays the home page of the student after a successful login. The page contains a menu of options, the user can perform operations like; view the numbers, view the letters, view the lessons (video or audio), and do a test and get the results.

Figure 2. Login as Student

2.4 Add a student by the admin

In this page (Figure 3) the admin (teacher) can add students to the system.



User Name :

Password security :

PassWord :

Confirm_PassWord :

User Type : student

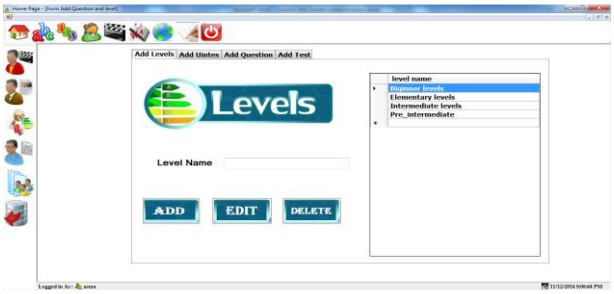
Level : Biginner levels

Figure 3. Add Student Page

The figure above shows the way to register a new student, This page presents the "add user" tab, a text box for the username and its label, a text box for the Password security and its label, a text box for the password and its label, a text box for the confirmation of the password and its label, a dropdown list contains a number of Levels and its label, a text box for the user type and its label, and a button to continue the registration process and save information submitted in the form into the database.

2.5 Add level by Admin

In this page (Figure 4) the admin (Teacher) can add an educational level to the system, with the ability to modify and delete data from the same page. The user can search for a particular level and then click on the found level in order for the system to display the level's name inside the text box provided.

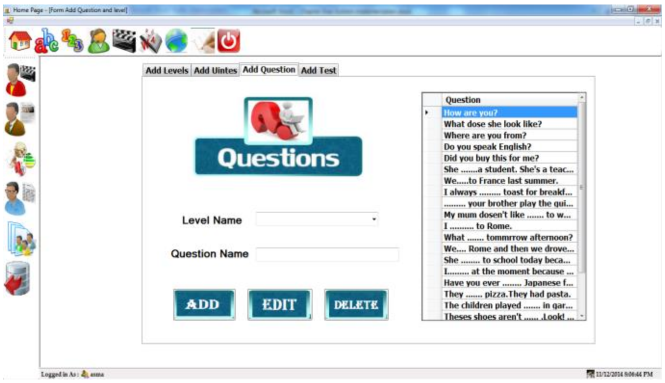


level name
Biginner levels
Elementary levels
Intermediate levels
Pre_intermediate

Figure 4. Add level by Admin

2.6 Add questions by Admin

In this page (Figure 5) the admin can add questions to each level in the system, with the ability to modify and delete data from the same page, to search for a particular question, the user can choose the question from the table provided. The system will then display the questions inside the text box provided.

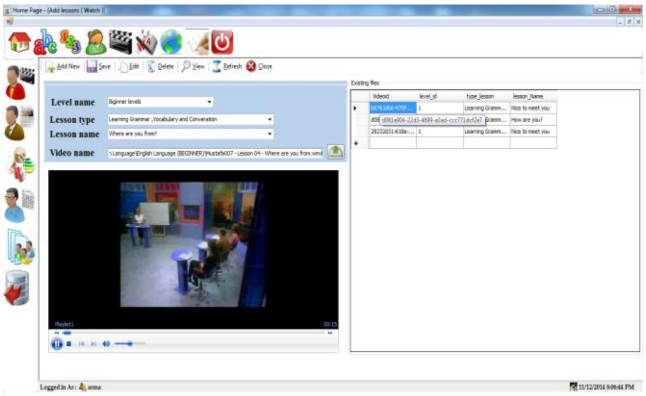


Question
How are you?
What dose she look like?
Where are you from?
Do you speak English?
Did you buy this for me?
She ....a student. She's a teac...
We.....to France last summer.
I always .... toast for breakf...
.... your brother play the gui...
My mum dosen't like .... to w...
I .... to Rome.
What .... tommrow afternoon?
We.... Rome and then we drove...
She .... to school today beca...
I......... at the moment because ...
Have you ever .... Japanese f...
They .... pizza.They had pasta.
The children played .... in gar...
Theses shoes aren't ...... .Look! ...

Figure 5. Add questions by Admin

2.7 Add Lessons (video) by Admin

In this page (Figure 6) the admin (teacher) can add lessons to each educational level in the system. These lessons contain videos explaining the English language lessons for each level. The lessons are saved in the database and retrieved when needed. The system will display all data inside the text boxes with the ability to modify and delete data from the same page. The user can search for lesson names by clicking on the DataGridView.



Level name: Biginner levels
Lesson type: Learning Grammar ,Vocabulary and Converation
Lesson name: Where are you from?
Video name: n Language\English Language (BEGINNER)\Mustafa007 - Lesson 04 - Where are you from.wmv

Videoid	level_id	type_lesson	lesson_Name
b8761d66-975f-...	1	Learning Gramm...	Nice to meet you
d08 d081e904-22d3-4899-a1ad-ccc771dcf2e7		Gramm...	How are you?
29232d31-618a-...	1	Learning Gramm...	Nice to meet you

Figure 6. Add Lessons (Video) by Admin

All educational classes are accessed in this page including levels, types of lessons (grammar – full explanation, grammar – conversation – listening, conversation), and name of the lesson, that is by using FILESTREAM feature to store structured and unstructured data. There are a few steps to follow when writing the code to access the data through the streaming APIs when adding, editing, searching and viewing lessons.

1. Start a transaction**:** A transaction is needed to gain a streaming access to the FILESTREAM data.
2. Retrieve the path name and transaction context: A logical path is needed to obtain a name to the FILESTREAM data file and a pointer to the current transaction context to access the FILESTREAM data.
3. Open the FILESTREAM data file**:** Gain access to the FILESTREAM data file either using the .NET wrapper class or the Win32 FILESTREAM API.
4. Read/Write FILESTREAM data**:** Read or write FILESTREAM data using the file I/O functions.
5. Close the FILESTREAM data file**:** Gracefully close the FILESTREAM data file.

6. Commit/rollback the transaction**:** Commit or roll back the transaction when you have finished.

   These steps are required to be followed in order to store videos using a combination of both the database and the files system. The following code explains how to add lessons to the database, there are several functions which are:

   - Upload the lesson using the Open Dialogue function (Figure 7).

```
OpenFileDialog dlg = new OpenFileDialog();
dlg.Title = "Select a file to upload";
dlg.Filter = "Video files
(*.avi;*.qt;*.wmv,*.mov,*.mp4)|*.avi;*.qt;*.wmv;*.mov;*.mp4|"
+ "All files (*.*)|*.*";   dlg.Title = "";
if (dlg.ShowDialog() == DialogResult.OK){
UploadVideo.Text = dlg.FileName;
axWindowsMediaPlayer1.URL = dlg.FileName;}
```

Figure 7. Upload File (Video)

- After writing the lessons' data, the lessons are stored using the code as explained in Figure 8:

```
FileStream FileStream = new System.IO.FileStream(UploadVideo.Text,
System.IO.FileMode.Open, System.IO.FileAccess.Read);
Reader = new System.IO.BinaryReader(FileStream);
FileData = Reader.ReadBytes(((int)(FileStream.Length)));
double FileSize = FileData.Length;
FileStream.Close();
TextBox2.Text = FileSize.ToString();
fsDB.AddFiles(this.level.SelectedValue.ToString(), comtypelesson.Text,
this.txtVideoname.Text.ToString(), this.textBox3.Text.ToString(),
FileData);
MessageBox.Show("Saved the file");
```

Figure 8. Call add lesson procedure

- The "AddFile " function is called as in the following code and it includes (Level number, Type of lesson, name of lesson, Type of video, video path). Figure 9 shows how this procedure can be done:

```
tx = con.BeginTransaction();
SqlCommand cmd = new SqlCommand("FSInsertVideo", con, tx);
cmd.CommandType = CommandType.StoredProcedure;
cmd.Parameters.Add(new SqlParameter("level_id", level));
.
.
SqlCommand cmd2 = new SqlCommand("SELECT
GET_FILESTREAM_
TRANSACTION _CONTEXT()", con, tx);
object obj = cmd2.ExecuteScalar(); byte[]   txCtx = ((byte[])(obj));
SqlFileStream fs = new SqlFileStream(path, txCtx, FileAccess.Write);
```

Figure 9. Add Lessons (Video) by Admin

Note: Adding file -This function is used to store data in the database with storing the video in the FILESTREAM folders to allow the user to retrieve it anytime.

Retrieving lessons: The search function is called first to view the data before calling the amendment and deletion function, there are three functions have been used to retrieve data and these functions are:

1- The application uploads all data from the database and displays it in a DataGridView (Figure 10).

```
SqlCommand cmd = new SqlCommand("SELECT [Videoid],[level_id],
[type_lesson], [lesson_Name]  FROM Lesson_Video", con); con.Open();
   return cmd.ExecuteReader(CommandBehavior.CloseConnection);
```

Figure 10. Data display in a DataGridView

2- When clicking on a DataGridview cell it then displays all data in text boxes (Figure11).

```
TextBox1.Text = grdFiles.CurrentRow.Cells[0].Value.ToString();
level.SelectedValue  = grdFiles.CurrentRow.Cells[1].Value.ToString();
comtypelesson.Text = grdFiles.CurrentRow.Cells[2].Value.ToString();
txtVideoname.Text = grdFiles.CurrentRow.Cells[3].Value.ToString();
```

Figure 11. Data display in a text boxes

3- The video is displayed in the AxWindows-MediaPlayer (Figure 12).

```
SqlTransaction txn = con.BeginTransaction();
string sql = ("SELECT VideoData.PathName(), GET_FILESTREAM_
TRANSACTION _   CONTEXT(),VideoName FROM Lesson_Video
WHERE VideoId= '" + TextBox1.Text +   "'");
.
.
SqlFileStream sfs = new SqlFileStream(filePath, objContext,
System.IO.FileAccess
```

```
.Read); byte[] buffer = new byte[(int)sfs.Length];
sfs.Read(buffer, 0, buffer.Length);
FileStream fs = new System.IO.FileStream(filename, FileMode.Create,
FileAccess.
Write, FileShare.Write);
axWindowsMediaPlayer1.URL = filename;
```

Figure 12. Retrieve Video form the database

Editing lessons: After confirming the validity of all the data in the database and viewing it, the Administrator (Teacher) can edit the data by selecting a new lesson using the *Upload* function as shown in the above figure 7. Figure 13 shows how the procedure *Edit* can be done:

```
Guid ID = new Guid(TextBox1.Text);
fileData = System.IO.File.ReadAllBytes(UploadVideo.Text);
FileSize = fileData.Length;
TextBox2.Text = FileSize.ToString();
fsDB.UPWriteFileStream(ID, this.level.SelectedValue.ToString(),
comtypelesson.Text, this.txtVideoname.Text.ToString(),
this.textBox3.Text.ToString(),
fileData);
```

Figure 13. Call edit Lesson (Video)

After downloading the lesson and filling in the data, the "*UPWriteFileStream()*" function is called as shown in the following Figure 14.

```
SqlCommand cmd = new SqlCommand("Update  Lesson_Video set
level_id=@level_id,
type_lesson=@type_lesson,Lesson_name=@Lesson_name,VideoName=
@VideoName,
VideoData=0x where Videoid='" + pID + "'", con);
```

```
cmd.Parameters.Add("@VideoId", SqlDbType.UniqueIdentifier).Value =
pID;
cmd.ExecuteNonQuery();
```

Figure 14. Edit Lessons (Video) by Admin

Figure 15 shows how to retrieve the current transaction context and the logical path to the FILESTREAM data value prior to requesting streaming access from C#.

```
SqlTransaction trx = con.BeginTransaction();
cmd = new SqlCommand("SELECT VideoData.PathName(),
GET_FILESTREAM _
TRANSACTION_CONTEXT() " + "FROM lesson_video " + "WHERE
VideoId = @VideoId",
con); cmd.Transaction = trx;
```

Figure 15. Retrieve the transaction context and logical path name of a FILESTREAM data value.

Figure 16 shows how to open a FILESTREAM data file for reading.

```
rdr.Read();
string filePathInServer = rdr.GetString(0);
byte[] transactionContext = (byte[])rdr[1];
rdr.Close();
SqlFileStream fs = new SqlFileStream(filePathInServer,
transactionContext, FileAccess
.ReadWrite);fs.Write(fileData, 0, fileData.Length);   fs.Close ();
trx.Commit();
```

Figure 16. opening a FILESTREAM file for read access.

Deleting lessons: As explained previously, the FILESTREAM function deals with the database and the system files, therefore the deletion function can be called, this function calls on two functions to clear the lesson data from within and beyond the database as shown in the following Figure 17.

```
Guid ID = new Guid(TextBox1.Text);
fsDB.DeleteFile(ID);
```

Figure 17. Delete lesson (Video)

The following code shows how to delete data from within the database (Figure 18).

```
string sqll = ("DELETE FROM Lesson_Video where videoid = '" +
intGUID + "'");
SqlCommand cmdd = new SqlCommand(sqll, con);
cmdd.ExecuteNonQuery();
```

Figure 18. Delete lesson (Video) data from database

The following code illustrates the FILESTREAM access to the files within the system files (Figure 19).

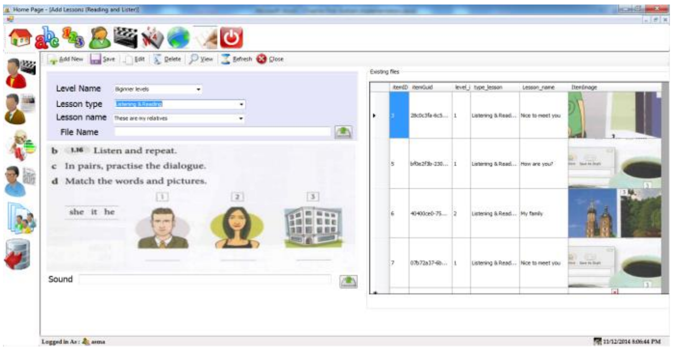
```
cmdd.CommandText = "CHECKPOINT";
cmdd.ExecuteNonQuery();
```

Figure 19. Delete video from Folder Filestream

Note: Every table having a FILESTREAM column should have a UNIQUEIDENTIFIER column with ROWGUIDCOL and UNIQUE attributes.

2.8 Add Lessons (Listening) by Admin

In this page (Figure 20) the admin can add lessons to each level in the system. These lessons contain images and sound to explain the English language lessons for each level of education. These lessons are saved in the database and retrieved when needed with the ability to modify and delete data from the same page. Users can search for particular lessons by choosing the lesson from the table provided. The system will then display all data inside the text boxes provided.



Level Name: Biginner levels
Lesson type: Listening & Reading
Lesson name: These are my relatives
File Name:

b 1.16 Listen and repeat.
c In pairs, practise the dialogue.
d Match the words and pictures.
she it he

Sound:

itemID	itemGuid	level_i	type_lesson	Lesson_name	ItemImage
3	28c0c3fa-6c5...	1	Listening & Read...	Nice to meet you	
5	bf0e2f3b-230...	1	Listening & Read...	How are you?	
6	40400ce0-75...	2	Listening & Read...	My family	
7	07b72a37-6b...	1	Listening & Read...	Nice to meet you	

Logged in As : asma

11/12/2014 8:06:44 PM

Figure 20. Add Lessons (Listening) by Admin

According to the previously mentioned steps required to store lessons containing (Images and Audio) using a combination of database and the files system features, the following code explains how to add lessons to the database. There are several functions that can be applied which are:

- Upload the lesson (Reading) using the Open Dialogue as in Figure 21.

```
OpenFileDialog dlg = new OpenFileDialog();
dlg.Filter = "Images
(*.BMP;*.JPG;*.GIF,*.PNG,*.TIFF)|*.BMP;*.JPG;*.GIF;*.PNG;*.TIFF|"
+ "All files (*.*)|*.*"; dlg.Title = "";
if (dlg.ShowDialog()== DialogResult.OK) {
UploadImage.Text = dlg.FileName;
PictureBox1.Image = Image.FromFile(dlg.FileName); }
```

Figure 21. Upload Image File

- Upload the lesson (Listening) using the Open Dialogue as in Figure 22.

```
OpenFileDialog dlg = new OpenFileDialog();
dlg.Filter = "Audio files  (*.wav;*.mpa;*.mp2,*.mp3,*.au,*.wma)|
*.wav;*.mp2; *.GIF;
*.mp3;*.au;*.wma|" + "All files (*.*)|*.*"; dlg.Title = "";
if (dlg.ShowDialog() == DialogResult.OK) {
UploadAudia.Text = dlg.FileName;
axWindowsMediaPlayer1.URL  = dlg.FileName; }
```

Figure 22. Upload audio File

- After writing the lessons' data, the lessons are stored using following code in Figure 23.

```
fileData = System.IO.File.ReadAllBytes(UploadImage.Text);
fileData1 = System.IO.File.ReadAllBytes(UploadAudia.Text);
fsDB.AddFilesImage(this.level.SelectedValue.ToString(),
this.txtItemNumber.
this.txtItemDescription.Text,textBox3.Text , fileData, fileData1);
```

Figure 23. Call edit Lesson

- The function “AddFilesImage“ is called that includes the attributes (Level number, Type of lesson, name of lesson, Type of audio, Image path and audio path). Figure 24 shows how this procedure can be done:

```
tx = con.BeginTransaction();
SqlCommand cmd = new SqlCommand("FSInsertimage", con, tx);
cmd.CommandType = CommandType.StoredProcedure;
cmd.Parameters.Add(new SqlParameter("level_id", level));
.
SqlCommand cmd2 = new SqlCommand("SELECT
GET_FILESTREAM_TRANSACTION_
CONTEXT(),GET_FILESTREAM_TRANSACTION_CONTEXT()",
con, tx);
SqlFileStream fs = new SqlFileStream(path, txCtx, FileAccess.Write);
SqlFileStream fs1 = new SqlFileStream(path1, txCtx1, FileAccess.Write);
```

Figure 24. Add Lessons (Listening) by Admin

Note: Adding file function -This function is used to store data in the database together with storing the Image and audio in FILESTREAM folders to allow the user to retrieve them anytime.

Retrieving lessons: The Search function is called first to view the data before calling the amendment and deletion functions, there are two functions used to retrieve data and these functions are:

1. The application uploads images from the database and displays them in a DataGridView (Figure 25).

```
SqlCommand cmd = new SqlCommand("SELECT [itemID],[itemGuid],
[level_id],
[type_lesson],[Lesson_name],[ItemImage] FROM Lesson_listening",
con);
con.Open();
return cmd.ExecuteReader(CommandBehavior.CloseConnection);
```

Figure 25. Data Display (Image) in a DataGridView

2. When clicking on the DataGridview cell, this action will display the image in the picture box (Figure 25).

```
TextBox1.Text = grdFiles.CurrentRow.Cells[0].Value.ToString();
cmd.CommandText = ("SELECT ItemImage FROM Lesson_listening
WHERE Itemid  =" + (TextBox1.Text + ""));
byte[] buffer = (byte[])cmd.ExecuteScalar();
MemoryStream ms = new MemoryStream(buffer);
Bitmap bmp = new Bitmap(ms);
PictureBox1.Image = bmp;
con.Close();
```

Figure 25 Retrieve Image form database

Editing lessons: After confirming the validity of the data in the database and viewing it, the Administrator can edit the data by selecting a new lesson using the Upload function as shown in the above Figure 21, 22. Figure 26 shows how the editing procedure can be accomplished.

```
Guid ID = new Guid(textBox2.Text);
fileData = System.IO.File.ReadAllBytes(UploadImage.Text);
fileData1 = System.IO.File.ReadAllBytes(UploadAudia.Text);
fsDB.UpdateFileStreamnew(ID, this.level.SelectedValue.ToString(),
this.txtItemDescription.Text.ToString(), txtItemNumber.Text,
textBox3.Text, fileData,
fileData1);
```

Figure 26. Call edit Lesson (Listening)

After downloading the lesson and filling in the required data, the user can call on the “UpdateFileStreamnew“ function as shown in the following Figure 27.

```
SqlCommand cmd = new SqlCommand("Update  Lesson_listening set
level_id=@level_id,
type_lesson=@type_lesson,Lesson_name=@Lesson_name,ItemDescripti
on=   @ItemDescription, ItemImage=0x  where ItemGuid='" + pID + "'",
con);
cmd.Parameters.Add("@ItemGuid", SqlDbType.UniqueIdentifier).Value
= pID;
.
cmd.ExecuteNonQuery();
```

Figure 27. Edit Lessons (Listening) by Admin

Figure 28 shows how to retrieve the current transaction context and the logical path to the FILESTREAM data value prior to requesting streaming access from C#.

```
SqlTransaction trx = con.BeginTransaction();
cmd = new SqlCommand("SELECT
ItemImage.PathName(),GET_FILESTREAM_
TRANSACTION_CONTEXT(),Itemsound.PathName(),
GET_FILESTREAM_
TRANSACTION_CONTEXT() " + "FROM Lesson_listening " +
"WHERE ItemGuid =
@ItemGuid", con); cmd.Transaction = trx;
```

Figure 28. Retrieving the transaction context and logical path name

Figure 29 shows how to open a FILESTREAM data file for reading.

```
string filePathInServer = rdr.GetString(0);
byte[] transactionContext = (byte[])rdr[1];
string filePathInServer1 = rdr.GetString(2);
byte[] transactionContext1 = (byte[])rdr[3];
SqlFileStream fs1 = new SqlFileStream(filePathInServer,
transactionContext,
FileAccess.Write); fs1.Write(fileData, 0, fileData.Length);
SqlFileStream fs = new SqlFileStream(filePathInServer1,
transactionContext1, FileAccess.
Write);
```

Figure 29. opening a FILESTREAM file for reading access.

Deleting lessons: As explained previously, the FILESTREAM function deals with the database and system files, therefore the user can call on the deletion function. This function calls two functions to clear the lessons’ data from within and beyond the database as shown the following Figure 30.

```
string sqll = ("DELETE FROM Lesson_listening where ItemGuid = '" +
intGUID + "'");
SqlCommand cmdd = new SqlCommand(sqll, con);
cmdd.ExecuteNonQuery();
```

Figure 30. Delete data (Listening) from database

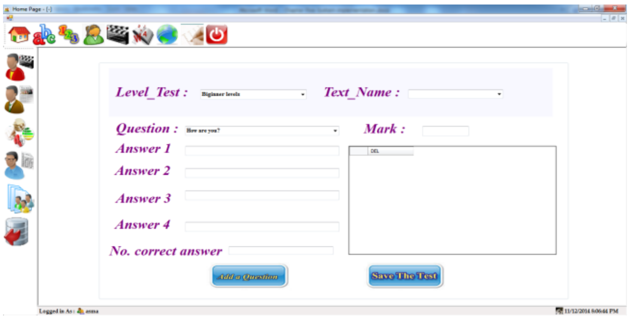
The following code illustrates the FILESTREAM access to the files within the system files point (Figure 31).

```
cmdd.CommandText = "CHECKPOINT";
cmdd.ExecuteNonQuery();
```

Figure 31. Delete Listening from Folder Filestream

2.9 Add the Test/Answers

This page (Figure 32) allows the administrator to add questions to the system. Teachers must type the question and at least four options for the answers. The correct answer for the question must be the number provided in front of the option. This page contains a dropdown list that contains the level's name and its label, a name of the test and its label, a name of the question and its label, a text box for the correct answer and its label, a text box for each wrong answer and its label, a text box for the mark and its label, a text box for the number of the correct answers and its label. The teachers can add as many questions as required. Then, press finish when the task is accomplished.



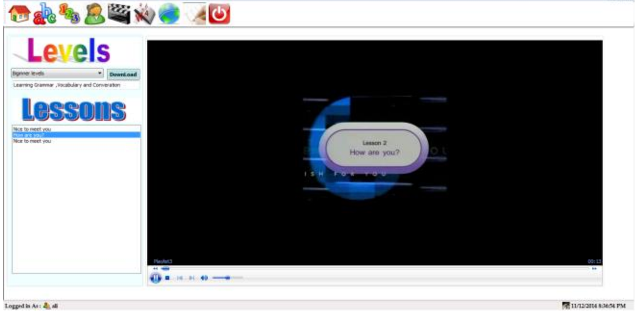
Level_Test : Biginner levels

Question : How are you?

Figure 32. Add the Test/ Answers

2.10 View the Lessons (Video) by student

Students can view the levels, and choose the required lesson from the dropdown menu that views all lessons. To view lessons just click on the lesson's name from the list-Box provided as shown in Figure 33.



Home Page - [View thw lessons (Watch)]

Biginner levels

Learning Grammar ,Vocabulary and Converation

Nice to meet you
How are you?
Nice to meet you

Figure 33. View the Lessons (Video)

The user can display all the lessons for each level by clicking on the "Download" Button that is in order to choose the required level and lesson from the dropdown list, as shown the following Figure 34.

```
con.Open();
string sql = ("select * from Lesson_Video where level_id=" +
TextBox1.Text + "");
SqlCommand cmd = new SqlCommand(sql, con);
SqlDataReader rd1 = cmd.ExecuteReader();
while (rd1.Read()){
if (rd1["type_lesson"].ToString() != "Grammar"){
ListBox1.Items.Add(rd1["Lesson_Name"]); }}
```

Figure 34. Views the lessons

When choosing the lesson name from the list-box, the video will be displayed in the AxWindows-MediaPlayer, as shown in Figure 35.

```
SqlTransaction txn = con.BeginTransaction();
string sql = ("SELECT VideoData.PathName(),
GET_FILESTREAM_TRANSACTION_
CONTEXT(), VideoName FROM Lesson_Video WHERE VideoId= '" +
TextBox2.Text + "'");
.
.
FileStream fs = new System.IO.FileStream(filename, FileMode.Create,
FileAccess.Write,
FileShare.Write); fs.Write(buffer, 0, buffer.Length);
AxWindowsMediaPlayer1.URL = filename;
fs.Flush(); fs.Close();
```

Figure 35. View the Lessons (Video)

2.11 View the Lessons (Listening)

In this page, students can view all the lessons, the mechanism is that they need to choose the name of the level from the dropdown menu, view the lessons of any level and then choose the desired lesson. To view lessons, click on the lesson's name from the dropdown menu, to listen to a lesson, click on the "listen" button.

To choose the desired level from the dropdown list, display all the lessons of any level by clicking on the "Download" Button, see Figure 36.

```
string sql = ("select * from Lesson_listening where level_id=" +
(TextBox1.Text + ""));
SqlCommand cmd = new SqlCommand(sql, con);
SqlDataReader rd = cmd.ExecuteReader();
while (rd.Read()) {
ListBox1.Items.Add(rd["Lesson_Name"]);  }
```

Figure 36 Views the lessons

When choosing a lesson from the list-box, an image is displayed in the picture box as shown the following Figure 37.

```
sqlCommand.CommandText = ("SELECT ItemImage FROM
Lesson_listening WHERE
Itemid ="+ (TextBox2.Text + ""));
byte[] buffer = (byte[])sqlCommand.ExecuteScalar();
MemoryStream ms = new MemoryStream(buffer);
Bitmap bmp = new Bitmap(ms);
PictureBox3.Image = bmp;
```

Figure 37. View the Lessons (Reading)

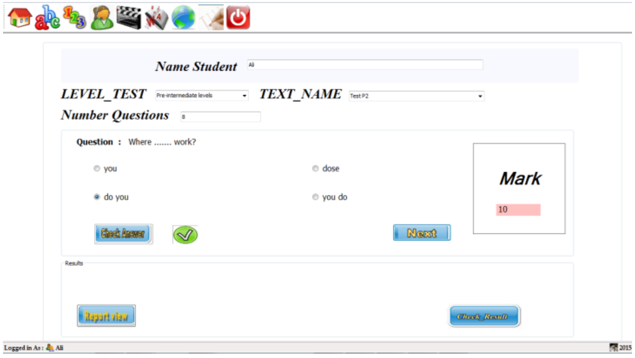
To listen to the lesson, click on the "Listen" button as shown the following Figure 38.

```
SqlTransaction txn = con.BeginTransaction();
string sql = ("SELECT Itemsound.PathName(), GET_FILESTREAM_
TRANSACTION_
CONTEXT(),itemDescription FROM Lesson_listening WHERE itemID="
+TextBox2.Text+"");
.
SqlFileStream sfs = new SqlFileStream(filePath, objContext,
System.IO.FileAccess.Read);
byte[] buffer = new byte[(int)sfs.Length]; sfs.Read(buffer, 0,
buffer.Length);
FileStream fs = new System.IO.FileStream(filename, FileMode.Create,
FileAccess.Write,
FileShare.Write); fs.Write(buffer, 0, buffer.Length);
AxWindowsMediaPlayer1.URL = filename;
```

Figure 38. View the Lessons (Listening)

2.12 Doing the Test

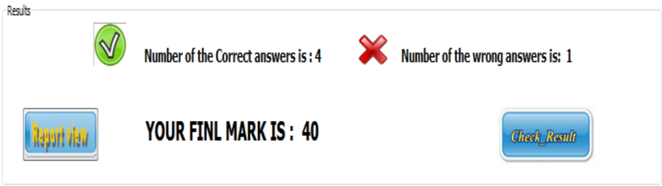
In this page (Figure 39), students can take the test, they have to choose the name of the level from the dropdown menu and then select the name of the desired test, the next step is to click on the "Start" button to start the test and check the answers at the end of the test. Students can check the results immediately after completing the test via clicking on the "Check_Result" button provided.



Name Student: Ai
LEVEL_TEST: Pre-intermediate levels
TEXT_NAME: Test P2
Number Questions: 8
Question : Where .... work?
○ you
○ dose
● do you
○ you do
Mark: 10
Logged in As : Ali

Figure 39. Do Test

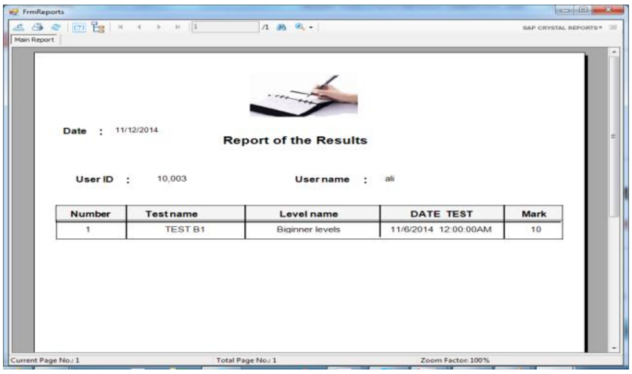
After the completion of the exam, the students can display and store the results as shown in Figure 40.



Number of the Correct answers is : 4
Number of the wrong answers is: 1
YOUR FINL MARK IS : 40

Figure 40. Display and store the results

The students can view a report of the results of all undertaken tests, as shown in Figure 41.



Report of the Results

Date : 11/12/2014

User ID : 10,003

User name : ali

Number	Test name	Level name	DATE TEST	Mark
1	TEST B1	Biginner levels	11/6/2014 12:00:00AM	10

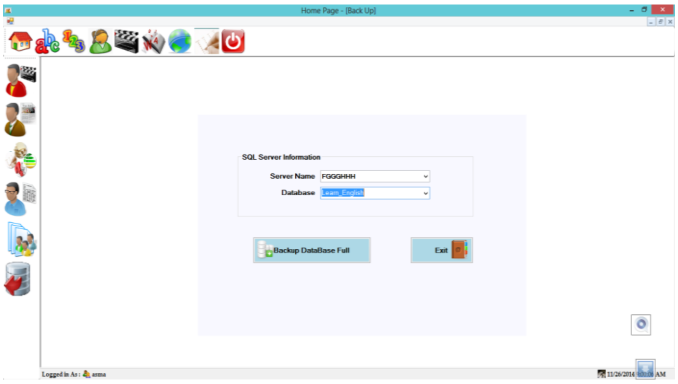
Current Page No.: 1
Total Page No.: 1
Zoom Factor: 100%

Figure 41. Reporting the result

3 BACKUP AND RESTORE FOR FILESTREAM DATABASE

3.1 Backup page:

In this page (Figure 42), the admin can take a full backup of the English learning database. To perform this operation, the admin chooses the name of the server and the name of the database from the dropdown list, and then clicks on the "Backup database" button to start the Backup.



SQL Server Information
Server Name: FGGGHHH
Database: Learn_English
Logged in As : asma
11/26/2014

Figure 42. Backup database

When choosing the name of the server from the Drop down- list box, all databases in the system are displayed as shown in Figure 43.

```
con = new SqlConnection(("Data
Source="+(cmbserver.Text.Trim()+";
Database=Master;integrated security=SSPI;")));
cmd = new SqlCommand("select * from sysdatabases", con);
dread = cmd.ExecuteReader();
while (dread.Read()){ cmbdatabase.Items.Add(dread[0]);}
```

Figure 43. Display database into Drop Down list

The teacher can take a full back up by calling the *("backup")* function. Figure 44 shows how this procedure can be done:

```
if (((cmbserver.Text == "")|| (cmbdatabase.Text == ""))){
MessageBox.Show("Server Name & Database Blank Field");
return;}
else if ((str == "backup")){
SaveFileDialog1.FileName = cmbdatabase.Text;
SaveFileDialog1.ShowDialog();
```

```
string s = SaveFileDialog1.FileName;
query(("backup database"+ cmbdatabase.Text+" to
disk='"+s+"'"));
```

Figure 44. Backup database

When the "Backup Database" button is clicked, a dialog box will appear to allow the admin to choose the location of the storage in which the backup will be stored.

Note: "*A full backup*“ of a FILESTREAM enabled database will also include the FILESTREAM data.

3.2 Restore page:

In this page (Figure 45), the admin can recover a full backup of the Learn_english database. To perform this procedure, the admin chooses the name of the server and the name of database from the dropdown list, and then clicks on the "Restore database" button to start the Backup.

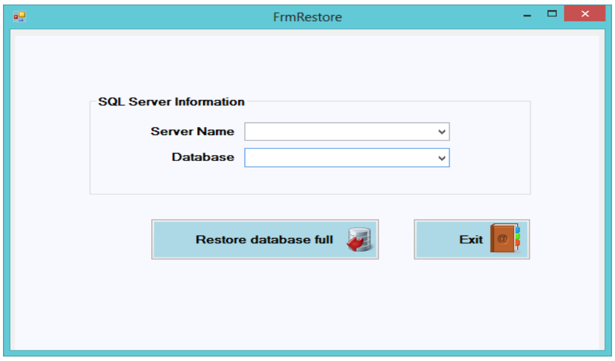


Figure 45. Restore database page

The admin can take a full back up by calling on the Restore Database function. Figure 46 shows how this procedure can be done.

```
query(("RESTORE DATABASE " + ComboBoxDatabaseName.Text + "
FROM
DISK ='"+ OpenFileDialog1.FileName +"'WITH REPLACE,RECOVERY
"));
label3.Text="Database Backup file has been restored successfully";
```

Figure 46. Restore database

When the admin clicks on the "Restore Database" button, a dialog box will appear in order for the admin to choose the desired database.

4 CONCLUSION

This paper has presented a novel approach of implementing the ELLS system via utilizing the Filestream technique supported by SQL server 2012. The system proved to be successful in allowing an efficient storage and management of unstructured data (video, sound and pictures).

A C# based Windows Forms have been created that contain all commands that deal with FILESTREAM applications that is to allow the user to create a new photo, video or audio in the database, and also to support the maintenance operations such as backups, restore, and consistency checking.

The outcome of this work is an attempt to maintain an efficient and effective storage and access to BLOB data. This would in turn allow the ELLS to operate smoothly by enhancing the storage and performance of the unstructured content in the database by leveraging the NTFS file system. This was performed while maintaining logical integration between the database and the file system that includes the transactional support feature. The OpenSqlFilestream native client API function has been used to deliver high-performance streaming of BLOB content between the file system managed by SQL Server and Windows applications.

REFERENCES

[1] Managing Unstructured Data with SQL Server (2008, July). Retrieved June 19, 2014, from http://download.microsoft.com/ download/a/c/d/acd8e043-d69b-4f09-bc9e-4168b65aa a71/SQL2008UnstructuredData.doc.

[2] Bhide, A., & Bhide, S. (n.d.). Big Data Expertise. Retrieved November 9, 2014, from http://calsoftinc.com/domain-expertise/storage-expertise/big-data-expertise/

[3] Jasmin Azemović. Varbinary vs. Filestream and other BLOB issues. Retrieved May 1, 2014 from http://sqltales.wordpress.com/2012/05/15/varbinary-vs-filestream-and-other-blob-issues-3.

[4] Paul S. Randal (SQLskills.com) FILESTREAM Storage http://nousinfo.wordpress.com /2010/09/07/ filestream-in-sql-server-2008/FILESTREAM Storage.docx .

[5] Sebastian, J., & Aelterman, S. (2012). The Art of SQL Filestream (p. 498). Simple Talk Publishing.

[6] Shaun Tinline-Jones. (2011, February). SQL Server 2008 R2 *FILESTREAM Design and Implementation Considerations*. Microsoft.

[7] SQL2008 Unstructured Data.[Online] microsoft, 2012.http://download.microsoft.com/ download/a/c/d/acd8e043-d69b-4f09-bc9e-4168b65aa a71 /SQL2008UnstructuredData.doc