

Functional specification

The context

Hounslow West Evangelical Church has recently ‘revamped’ their website. Users are now able to download audio files of the weekly sermons. Two of these sermons are recorded every week. Mr G Yuen, a senior software engineer working in defence, is an elder of the church as well as the church’s webmaster. He is a very busy man and therefore has delegated the task of digitising the sermon audio tape library to Mrs R Hollands.

Mrs R Hollands is the wife of the minister of the church and has an ‘aptitude’ for working with ICT and using software effectively. She has already been able to utilize a new sound card (that was a gift to her family from an anonymous giver) by using the software provided by the card’s manufacturer to convert a number of the sermon tapes into MP3. However, she does not know how to edit raw HTML code.

The nature of the problem

This means that when she has completed a conversion, she has to give the newly produced MP3 files to Mr Yuen, physically on a CD-ROM, at the Thursday evening prayer meeting. This enables him when he gets home; to upload the audio using FTP and update the raw HTML code himself.

This whole process is very inefficient and time consuming for both stakeholders.

Mrs R Hollands, when handing over the week’s audio files also gives a print out to Mr Yuen that she has created in Microsoft Word 2003. It contains the following information about each of the sermons:

- title
- date and time preached
- lesson (book, chapter and verse)

This information is required for the website. However, more information about each sermon would like to be retained by the church including a sermon summary, the preacher’s name and the name of the sermon series. Both stakeholders are fed up with the entire situation and considering scraping the whole venture.

A solution must be provided by writing a simple ‘front end’ interface with the website using JavaScript, HTML, PHP and MySQL or another solution would be to generate and store the HTML locally using a user input form to enter in new data, then FTP it to the server.

What the custom solution is required to do

Upon hearing about and understanding this problem – being the minister’s son – I will strive to build a customised solution that will eliminate the involvement of Mr Yuen and enable my mother to update the website quickly without seeing a shred of HTML code. I have chosen the latter solution as I am not an able JavaScript programmer.

The solution will have to be in Microsoft Word 2003 seeing as that is a very familiar application to Mrs R Hollands and the church cannot afford to pay out any money to procure a relational database application such as Microsoft Access or spreadsheet software such as Microsoft Excel.

Input

The final version of the customised solution will allow the user to enter the following information into a be-spoke Visual Basic form:

- date (DD, MM, YYYY)
- time of day (AM or PM)
- title
- lesson (book, chapter, beginning verse and ending verse)
- series name
- preacher
- MP3's location (example:
<http://hwec.org.uk/audio/mp3/YYYY/YYYYMMDDAM.mp3>)
- MP3's file size (bytes)
- duration (MM:SS)
- summary of the sermon

Processes and storage

The information inputted will then be stored in an appropriate way.

The solution will then be able to:

- retrieve this information
- navigate through the different sermons stored
- search for particular sermons via sermon title
- edit/update/amend information about the particular sermon found in the search
- append sermon data as new
- delete sermon data

The solution will perform the following processing of the data stored when requested:

- process sermon into three different orders (for HTML output)
 - by date (descending order)
 - by title (alphabetical order)
 - by book (alphabetical order)
- process into valid HTML
- process sermon data into valid iTunes podcast feed
- process a real simple syndication (RSS) feed

Output

The solution will output the following:

- valid HTML
- a valid iTunes podcast feed
- a real simple syndication (RSS) feed

Extra functionality

The customised solution will do the following upon request:

- clear the input form of sermon data
- upload the HTML produced to the HWEC web server
- upload iTunes podcast feed to the HWEC web server
- upload real simple syndication (RSS) feed to the HWEC web server

Measuring success

There are number of ways that I can measure the success of the customised solution:

1. Time taken to input, save and upload sermon data via form must be shorter than creating print out, waiting till Thursday to give print out to Mr Yuen, the time it takes Mr Yuen to update HTML code and upload the changes to the code to web server via FTP.
2. There must be less input mistakes made than before.
3. The solution must provide a secure front end and hide the processes and storage in the background.
4. The final solution must be easy to use and easy to navigate.
5. It must also notify the user of any errors or other problems.