CE301 Final Year Project

Database For Study Abroad

Supervisor: Dr Anthony Vickers
Second Assessor: Mr Keith Primrose

By Louis Jade

Registration Number: 0823084
Login: ljade

Contents

Abstract

The Study Abroad office currently has a paper based system for dealing with the transfer of study abroad placements, incoming and outgoing. Some information is already on the main university database however most of the work is done on paper and can be time consuming and inefficient when trying to generate information.

My Project will extend the current database at the University for the Needs of the study abroad office; this has been written in SQL Server for compatibility

The study abroad office will need to have a user interface to add, update, delete and search records as well as preforming management function for the quick manipulation of data. The interface has been produced in ASP.NET

Section 1: Introduction

Section 1: Introduction

1.1) Introduction Overview

The final project that I chose to create was “A database for the study abroad office” which is a database to store information about incoming/outgoing students. This led me to think about previous techniques I had learnt in the Databases and information retrieval CE-235 module [1] with the skill set I learnt from this module I should be able to make the database that is suitable. I also wanted to create a front end interface for the study abroad team so they had an end to end system. I had previously done some work creating interfaces in the Mobile and web application – CE311 module [2]

In taking this project I have learnt many things mostly how to transfer a client’s needs into an end system and trying to get the important balance of what a programmer thinks the client wants and what the client really wants. Whilst working on the project I worked a lot with SQL Server and ASP.NET these two disciplines are the strongest in my skill set to take to employers after my degree.

This document will describe my project so that it can be understood at a glance then go into technical depth in the latter sections. I begin by explaining the background information on my client and the way the current MIS system is run at Essex University

1.2) Acronyms

Study abroad office – SAO

Structured Query Language - SQL

Active Server Pages – ASP

Management Information System – MIS

European Region Action Scheme for the Mobility of University Students – Erasmus

Operating System – O/S

Unified Modelling Language – UML

Academic stay record – ASR or ASR key

Entity Relationship - ER

1.3) Background Information

The background research that I have carried out for this project can be seen in this section

My initial thought when reading the description of the Final Project Hand-out Extract [Appendix A] is that I would be creating a database for SAO that would allow them to add, update and delete data related to students studying abroad from Essex University and incoming students from other institutions.

From meeting my supervisor Anthony Vickers, who has being doing a lot of work with SAO and ERSAMUS as well as having a broad knowledge of how the university works. He discussed how our university holds all of its information in the MIS system managed by John Fell. This led me to look into what MIS systems before arranging to meet John Fell.

I went on to meet, Angela Turton the manager of SAO. She explained that currently they have a mainly paper based system however they still have to get a lot of information from MIS about students to send to other universities as well as using the information for their own purposes.

Angela is asked frequently to compile data on incoming/outgoing students to produce reports for staff from other departments of the university this all has to be done manually and can be very time consuming. Angela showed me how they currently search for information on the MIS system which provides them no unique facility to show information related to study abroad.

Before organizing the meeting with John Fell I did some research into what a MIS is as I had no great understanding of what it is or how it worked.

After some research [4], the best example I found of what a MIS provides was of a car dealership. For Example Toyota had the problem of producing too many cars that people did not want. They were sending each show room the same stock. They started to monitor what cars were selling and they found out (example) That the Colchester branch was selling 7 Black cars a month and 1 Red, whereas the surrey branch was only selling 2 black and 2 red. From this information they could start to Taylor the cars they were building to the each dealerships needs. From this I understood that they may have already had some of that data but by analysing at it and interrupting it to a useful function that it could be used to profit the business.

By meeting with John Fell the MIS manager he began to explain that the university had a vast amount of databases that held all kinds of information, some of it was input via web form such as accommodation details and management functions could be processed by using the MIS interface. John went on to say that I could use part of the universities backbone database to extend and build upon. After the meeting went back to re-read the Final Project Hand-out Extract [Appendix A] and it does state that some data should originate from the university database. By using the schema for Essex Universities own MIS I would be meeting this criteria.

1.4) Project Description

1.4.1) Project Type

A management information system

1.4.2) The Problem

My Project will be dealing with the problem of understanding and processing the needs of my client the SAO and transferring them into technical deliverables to facilitate the SAO needs. My project will store the information needed for the study abroad in a manner that it can be sorted and queried. From there the user will need an interface to interact with to provide the user qualitative data.

"Our marketing effectiveness leads to our sales effectiveness, which leads to our service effectiveness. Data quality is key to the success of that. If you don't have quality data, that whole chain breaks down." Chuck Scoggins, Hilton Hotels [5]

Agile development of a database in Microsoft Access would logically be simpler, as well as faster way to create a whole new database for SAO. By doing this it would be tailored to their needs without having any extra baggage. However integrating it with the existing MIS database would add more complexity because of having to learn the existing architecture but in the long run would be more maintainable as well as not having the issue of duplicate data.

If I want to continue to design the End to End system I will need a developing suite and a language to write it in. Originally I started to look at Uniface (what MIS is currently written in). After much deliberation I decided to develop the front end system in ASP.NET C# using Microsoft's visual studio, I feel this was a better choice as the front-end then become portable.

1.4.3) Aims and Objectives

Aim
To create a database and interface that caters for all the needs of SAO manager Angela Turton, which gives her more flexibility than the current MIS system.

Objectives

- Background Research
  - Looking at current MIS systems mainly in the educational stream.
- Future Idea's
  - Creating new features that are intuitive to the way the SAO team work based on what I have learnt from them.
- Developing A Database
  - Developing a database in the most efficient way for the SAO.
- A interface
  - Developing an interface that works with the database for adding, updating, deleting and manipulating data.
- Testing
  - Test the Database and interface to test functionality and remove bugs.
- Evaluation
  - Evaluating the effectiveness of my database and interface with SAO manager Angela Turton and MIS manager John Fell as well as my own bug testing.

Section 2: Planning

Section 2: Planning

2.1 Overview of this section

This section describes all the planning that took place for me to undertake my project.

2.2 Methodology

To create my project I needed a framework to start upon; at first I needed to research what a MIS is and how other people use them after this I looked into what software components comprised of a MIS. At this point I knew it would consist of a database and user interface to perform functions on the database to provide qualitative information to the user. I will have to use several different pieces of software which I mention later in section [3]. I will test my project on my client Angela Turton and get feedback from John Fell the MIS manager on the database that I have created as well as my own testing.

I started to carry out the background research into the MIS system which I put into my initial report [13] it gave me a much clearer understanding on what I need to provide, I then began to look into what would be the most appropriate type of language to use for the database and user interface. Currently the university has multitude databases mostly running on SQL server. The user interface was written in a language called Uniface, which is accessible to staff depending on their privilege level.

I am going to use SQL Server for the database creation and management of the database as I feel that is most fitting with the current university fittings I am then going to use visual studio ASP.NET C# to create the user interface. I will discuss why I use visual studio later instead of Uniface I printed the notes from CE311 and CE235 as they are directly relevant to using both of those languages.

For testing I will use Angela as the test subject because the program is designed for her and her team (SAO) I will be able to make any amendments that she needs. I will also like to get John Fell MIS manager to look at the project/ER diagram as I would like him to make use of my system within the universities already large database. The final testing for bugs I shall carry out myself.

To reach the goals of my project I am going to:

1. Research MIS
2. Research MIS software components
3. Meet with John Fell the database manager to discuss the current system
4. Meet with Angela Turton to discuss user needs
5. Begin development/implementation
6. Test
7. Changed based on updated needs

2.3 Development methods/models

Development models are designed to show the process of work, allowing you to indicate the age of the product and also the work flow process.

The models give a visual representation of each stage in the work flow process. Some examples of different development methods are show below.

2.3.1 Waterfall model

The waterfall model is what I had mainly used over previous assignments. Each section has to be completed before moving down to the next section. When issues occur the project must loop back to complete the problem before moving on again. As you can see from the diagram below that once the project is finished you then go back to the top section and starts again with the next iteration, until the project is complete.

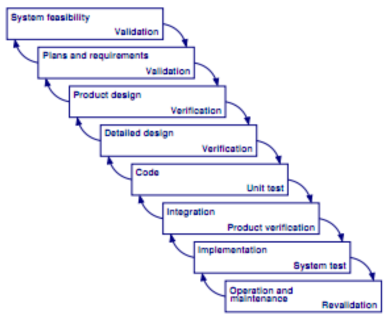


Image – waterfall [7]

2.3.2 Agile development

I have started to use this in the last semester of the 3rd year, for the module CE320 Large Scale Software Systems [9] the idea of agile development is that you move straight into programming using techniques such as extreme programming (XP). A customers needs would be something like a rapid design (something up and running very quickly). We had a Mark Anning from BT who spoke in depth about Agile design [8] and the way BT are using it to develop equipment and software.

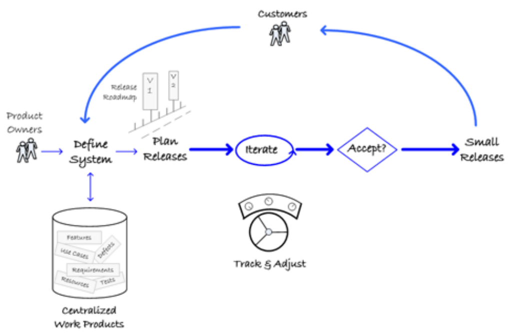


Image – Agile devolpment [7]

I thought about changing my design process to agile after the lecture with Mark however I didn't think it was suitable in the end as I will end up back tracking when making errors. When I looked into agile design even further it requires a lot of decoupling to make system wide changes quickly, which I didn't want to do.

2.3.3 Unified process

The unified process is a more heavy weight approach to the modelling system most models are built around UML designs, They are usually case driven for a client's individual needs and can specify each step of the user process, going into great depth if necessary.

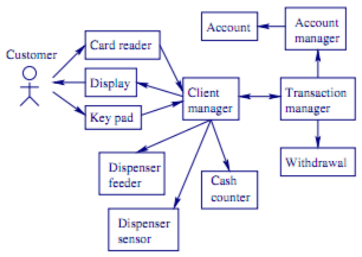


Image – Unified process [7]

For considering SAO's needs I needed to create a UML diagram for each relevant area/function of their work flow, by splitting up the needs of SAO I was able to create a ER diagram that I used to create their database.

2.4 Methodology conclusion

I have found that each model has its strengths and weaknesses, after the lecture with mark Anning I thought about switching to agile development because I get something running very quickly then build upon it, which would get results quickly.

After some deliberation I feel that I have taking techniques from two models and by split my project into two sections. For understanding my client's needs to be able to design the steps I need for my waterfall model I used the unified process for splitting the client's needs into relevant sections.

Then I used waterfall development for the actual program, taking each stage step by step then going over the whole project several times to discover bugs and make changes.

2.5 Project Development

As mentioned earlier I have split my project into two sections. First I had to take the needs of my client, having a finite set of user requirements this was key to getting the project right. Then I had the implementation phase which was structured into two halves, the database design in SQL and the User interface in ASP.NET C#.

2.6 Project Languages

I knew that the current MIS system was built in Uniface, after carrying out some research I could find very little about Uniface, it wasn't an off the shelf package such as Microsoft products. If I wanted to use Uniface[3] I was going to have to ring the software company, the result of the conversation was: The version of Uniface our university use is so old even they don't have a copy of it anymore and it's a very hard language to learn. I went to see John Fell the manager of MIS and he confirmed what Uniface was a hard language to learn and only one person at Essex has some basic training of how to use it. He suggested that I use something more portable (web based) such as developing it in ASP.NET C# which is still a programming language that will do everything I needed for the project just a modern web based version.

I used ASP.NET C# in a previous assignment [14] and I got on with ASP.NET well, the interface is very friendly and the error handling is also quite intuitive, it gives you slightly more detailed messages than other languages, out of all my choices of language I probably would have chosen this, it just didn't occur to me at first that this language was even an option.

For the database section I started using Microsoft Access for the prototype, knowing full well that it wasn't suitable, but was a quick way of testing some functionality. From this I moved onto SQL Server using SQL Server management studio which is a good software package to use with the language because it gives a nice GUI instead of command line and a visual representation of a database is a good feature. The language its self is very similar to MYSQL and was the best choice to use as the university already use it on their system and my designs would be compatible with their system, which is a key requirement.

2.6.1 Development tools

I list everything I used for my project below.

2.6.1.1 Software

- Operating Systems

  The O/S that I have used throughout the project will be windows 7, Both the Lab computers and my home machine run windows 7, this environment will also run all of the other software instruments necessary to carry out my project.

- Web Browsers
  I tested my project on various browsers, Internet Explorer, Mozilla Firefox and Google Chrome I have used the browsers to carry out background research and also to test various aspects of project to see the functionality as well as the design.

- Visual Studio
  Visual Studio 2010 is user friendly software developing tool that has allowed me to visually see the design as well as being able to code in the application. It was the best choice for me because it was amongst the best applications to code in ASP.NET and I could also obtain a free student licence.

- SQL Server Management studio
  SQl allows a user to create a large mainstream database and manage the database within the same suite. This software package has advanced features compared Microsoft Access, for example I can attach and detach a database, this is especially useful when Visual Studio needs to use it. SQL server manager also allowed the input of code into the execution tool, which allowed me to pre-write all of my tables and relationships then build them all with one button.

- Microsoft office:
  - Visio
    Visio has provided me with an easy way of creating ER diagrams. Visio allow you to connect objects together and move them but still keep them connected, it's a simple idea but not many other software packages offer it.

- Word
  Microsoft word an industry standard tool for word processing. All of my documentation will be written in Microsoft word however all notes will be made in the Log Book. End Note is also compatible which will be used for referencing.

- Excel
  Excel was used to view the sandbox version of the 'studb' & 'padbas' databases that I have been given by john fell. They contained a lot of information sometimes it was quicker to view them this way. I also used excel to export

- Access
  Access is a database creation tool very popular for creating usually small database. I used this to create a ER Diagrams at first, then moved onto Visio

- PowerPoint
  Weekly meetings with my project supervisor were shown on PowerPoint it's a good tool to use for demonstrating on a project as it allows flexibility in terms of design. I used PowerPoint again for designing my project poster.

- Project
  Microsoft Project again is an industry standard tool for documenting projects in professional formats such as Gantt charts, to show the time taken on each section of the project as well as showing key milestones.

- Notepad
  Notepad was briefly used to remove anomalies when doing a bulk insert for the sql data. Notepad is a handy tool because it opens most text based file types.

- End Note
  End Note has been used on all reports that require referencing; it's an easy way to build a library of references that automatically format within the document.

2.6.1.2 Hardware

- My Computer
  Most of my project work was carried out on my home machine. It took some time setting up with all the various pieces of software (visual studio, SQL server) but once it was well worth it as I work longer hours when I am at home. Towards the end of the project I started become worried about my computer failing so I regularly made backup copies to a pen drive.

- University computers
  I rarely used the university computers as it was easy and more comfortable to do it

at home, however I did use the universities machines for demonstrations as my project can be run on them.

- Log book
  The Logbook was an informal way of documenting the whole project. This actually shows dates of completed milestones as well as problems, meetings or other necessary information related to my project. Looking back in the beginning I was confused to why a computer scientist needed a paper based method of recording documents, when I started to write reports I realised how important this document is.

2.7 System Functionality

The system functionality of the SAO MIS can be almost endless, once I have all the tables properly normalized in the database, the information can be queried in lots of different ways to suit not only my client (SAO) needs but also the needs of other in the university who need feedback for their purposes.

From a Database perspective the functionality, will be to store information and process requests from the interface of the MIS to provide queried feedback and statistics. The database will be kept on server so it can be accessed by the user interface. If the database needs amendments it can be seen in its own form, in SQL studio management.

From a user interface point of view the program should be accessible via a browser. If all goes to plan and my changes get accepted by John Fell, then SAO will be able to view it from the local intranet hosted on the server. It will have a web address internally something like "studyabroad.essex.ac.uk" as an example. This will open the home screen to which they can log into.

Once the client is logged into the interface they can add new records about an incoming student from another university. For outgoing students less detail can be taken because from the ASR key I can look up in the database the information that already available about them and use it for student record purposes (minimising duplicate data). Both incoming and outgoing students can have records searched and updated. As well as being able to generate statistical information which can be in some cases viewed in graphical format or exported to excel for other manipulation.

The Detailed version of all system functionality is in section 4 "Design"

2.8 Project planning

Project Planning discusses major/milestone task and how they have been undertaken and completed.

2.8.1 Task Description

The task description gives an overview, of each element of the project.

- Analysis and Requirements
  Ground knowledge of the project, setting the footings for the rest of the work.

  - Background reading
    The background reading was learning the introduction to the topic of MIS and the furthermore how MIS are constructed.

  - Meetings
    Regular meetings with my supervisor to track progress, I also met Jerome Robinson (CE235 Lecturer) to get comments on ER diagram.

  - Initial report
    the basic report on all work done to that point and the Gantt chart.

- Design
  All Design aspects of design related to the project.

  - UML/ER diagrams
    The UML/ER diagram will show the layout of tables so that i can create the real database from these designs.

  - Technical background reading
    Carry out basic background reading to see if my designs are achievable.

  - Prototype GUI
    The prototype GUI will be a basic demonstration of some features. Connecting to the database and style sheet.

  - Configure home machine
    I need to set up my home computer to run all the necessary programs to create my project.

  - Interim report
    All work to date, with the technical specification and updated Gantt chart.

- Implementation
  - Meetings
  - Technical Reading
  - Implement database
  - Implement GUI

- Project Poster Day
  The research and design of my poster.

- Testing
  - Meeting with Angela
    The meeting with Angela will to see if I have met the user requirements. Angela’s computer knowledge is limited so this will also count as a non-computer literate user.

  - Meeting with John
    The meeting with john will give a overview of my database, I would like a comment to include in this report.

  - Personal Testing
    I will test the program personally for all bugs and faults.

- Final Report
  This Document is the summary of my work; it is now finished and tested. Within this document I will give a critical summary of my work.

2.10 Schedule

Building my schedule/Gantt chart has been a important part of the project it need to make a clear summary of everything that I have undertaken over the last few months and plan for everything that I need to do. My Gantt chart show in [Appendix 2: Gantt chart] details all the mile stones and activities I needed to carry out to complete my project. I feel that I have planned well and my Gantt chart has been a well thought out and structured document.

I ordered my project my milestones and activity sequencing. Milestones was a major point or hand of the project which I had many of and I think I planned this accurately, activity sequencing was the order I carried out tasks again I stuck to this and planned this well which was a good indication I knew what I was doing from the planning stage to writing the final report.

From the interim report, not a lot changed other than the amount of time that I dedicated towards my implementation. Instead of taking 12 days for back end implementation (database) I took 5 days and then 17 for the front end system as I had done the database almost in the planning section of the project. I allowed slack time with every measurement I made because things never go to plan, this all worked out rather well and I finished just about it time. I would say that the overall planning of the gantt chart was a good representation to the time and dates i dedicated to my project.

Section 3: Requirements

Section 3: Requirements

3.1 Overview of this section

This section will discuss all the requirements of my project, then in more detail the requirements specification and finally the non-functional requirements.

3.2 Requirements

My project/program will meet the following requirements listed below [15]:

- The ability to store records in a logical manner due to normalized table and data field set with related tables that provide no duplicate data.
- The ability to add historic data and store it in the database by warehousing all information so that it can be indexed for quick searching.
- Incoming Students
  - Add
  - Modify
- Outgoing Students
  - Add via previous record
  - Modify study abroad data
- Query
  - The system will be able to provide advanced data definition using querying formula to produce a set relevant set of records to a user within a few steps of use.
  - Queries related to incoming students
  - Queries related to outgoing students
  - All of SAO's prebuilt queries [Appendix Section G: The List of Angela Turtons Requirements]
- Secure
  - All data input must be validated, against malicious attacks and mistakes.
  - A login page for staff, which will provide authentication, after this they will be redirected to a home page were a series of options to perform other tasks will be available.
- Maintainable
- User friendly
  - The users should not need a technical knowledge, however it would be assumed that users can navigate and use a normal webpage.
  - Navigation on all pages, that is similar to the Essex university theme at current (The header bar).
  - All pages will follow the same theme to keep layout similar.
- Portable
- Available & reliable

3.3 Requirements Specification

- Incoming Students
  - Add
    Add completely new information to the database for a given student.

  - Modify
    The ability to modify the records of a student in the database without having to remove them

- Outgoing Students
  - Add via previous record
    Add a student record by taking the ASR key of a current student which will take the default information from the already existing database and add the new necessary information to the study abroad section of the database.

  - Modify study abroad data
    The ability to modify the records of a student in the database without having to remove them

- Query
  - Queries related to incoming students
  - Queries related to outgoing students
  - All of SAO's prebuilt queries [Appendix Section G: The List of Angela Turtons Requirements]
    Manager of SAO Angela Turton wrote a list of functions that the software needs to be able to produce results for this can be found in: [Appendix Section G: The List of Angela Turtons Requirements]

- Secure
  The web based application needs to be secure, for example when this is hosted is someone access the page who isn't granted access we should ask for a user name and password to confirm the users identity and divert unwanted traffic. Secure also in terms of the database: The database should be stored in a secure environment (Essex University server farm would be suitable) and also the data that is put in to stop human error affecting the databases safety, For example a user should not be allowed to type in the command "dropdatabase;" or insert numerical information in to CHAR fields.

- Maintainable
  The system needs to be built In a way In which it can be easily updated for example instead of changing the theme on every page, having a style sheet so I can just change the information on page. This also includes the use of master pages.

- User friendly
  SAO have medium to limited computer knowledge so it needs to be simple, they have knowledge of Microsoft software such as: Excel, Word, Outlook and internet Explorer so any external attributes should use them components.

- Portable
  The software will be viewable via a web browser and if located on the server should be useable from any internet active computer in the world.

- Available & reliable
  If successful and it is hosted on the Essex University Server, it will be reliable as the server is maintained by a team of technicians and it will be available 24/7 as the server is constantly online.

3.4 Non-functional requirements

- Normalization
  Normalizing the database is a key requirement for its success but to fit in line with the current structure MIS is written in I would like the database to be normalized.

- Fitting with Essex University Business theme
  Essex university website has a “business theme” everything relating to Essex University is now displaying the logo and using the style sheet and necessary parameters to keep it again in line with the current theme.

3.5 Requirements Conclusion

I am going to use this section later in [Section 7: Conclusion] to critically asses if I have met the requirements of this project.

Section 4: Design

Section 4: Design

4.1 Overview of this section

This section will discuss the main design points in detail of the database and the user interface system from the initial design the design to implement phase. I will start to talk about the initial idea of the designs then the prototype of the database and user interface. Finally I will describe the final design for both user interface and database

4.2 Initial idea

The initial idea was to create the database in the prototype of the database in this section and only create designs in visual studio for the interface with no functionality (for the design stage).

4.2.1 The Database

the initial design was to test the functionality of the queries as I thought the hardest part of the database will to get the queries working, so I wanted to develop a prototype to test this. Then I can plan the ‘design’ of my real database schema.

4.2.2 Prototype Database

I started by making a prototype of a database in Microsoft Access database for testing to see if my actual design would work. I knew that I wanted to create the final system in SQL Server because studb is written in SQL Server and I wanted to use that database so it would limit the amount of duplicate data.

I took on Angela’s needs and created a database ‘purely’ for her needs, so this database has all new student information (which was a bad idea but I explain why later). I tested that some of Angela’s query’s worked; mining basic information from the database and the structure I created worked to the extent I could run basic procedures from it however I wanted to a database that interacted with the current database at MIS and this system built in Microsoft Access was not capable of doing that. I have attached a copy of the ER diagram in [Appendix 5: Initial Access ER Diagram]

4.2.3 Meeting with John

I went to show John Fell MIS manager, my ER Diagram of what I have made in Access as a basic structure and he said “the design looks a bit flat” we went on to talk about more about how Angela’s process worked and what she really wanted. The outcome of the meeting [Log book 9/04/2010] was that I should have normalized my current database in Access further, however I missed the boat slightly with a couple of her user needs and what I would be best doing is taking a sandbox version of the studb and padbas database that run at Essex University.

4.2.4 Studb and Padbas

Studb is a record of student information, has various tables inside and contains most of the information we hold as a university about students. Padbas is a database that holds mainly information about other university or non-hessa universities. I will go on to explain more how I use these databases later.

4.2.5 SQL Database 1

From this point it changed the whole way I needed to think about creating a database, from now I already had outgoing student information. My main focus was to look at the sandbox version of the database and then learn how it worked, to then integrate it with my plans, I would need to add and extend the current system for it to provide the output needed for the SAO. I have included a the ER diagram in [Appendix E: initial ER diagram] the main floor with the ER diagram is that I overlooked the fact that incoming and outgoing students have a different requirements as well as the fact the study abroad office is the first point of contact that an incoming student has with our university. From here registry add them to the main database once some financial and clerical information has been approved. From this it took me to my final iteration this is discussed below in the Main Database design section.

4.2.6 The User Interface

Initially I had thought about doing the interface in Uniface which I mentioned prior to this, but that became unachievable and inefficient, so I started to do my designs in visual studio, creating a master page and then using the style sheet within each page. I started creating a few basic pages just to see the structure and have a basic theme, a footing for the rest of my designs using the Essex university business theme. I wanted all the pages to look like the home page of the current theme for the university.

It began to show how quickly I could get something up and running that was aesthetically pleasing. I have attached a couple of screen shots from my initial design to [Appendix 7: Initial Designs]

4.3 Updated idea

When stepping into phase two of the design I was going to be doing 80% of the design work* for the database and 20% implementation* as this was mostly design based however the user interface was the complete opposite 20% design phase* 80% implementation*.
* = estimate values.

4.4 Main Database design

When designing the main database, the main thing I had to consider was table design and normalization from there I was able to design all of the queries SAO needed. I am going to be using [Appendix 6: Final ER diagram] as a reference throughout this section.

4.4.1 Table design and Normalization

I had to design each table so that it had relevant information, to that table. Most normalization critics say that the information should be based on the whole key and nothing but they key [10]. The way MIS is currently built is that information for one person all relates to their unique personal identifier PRID, if that person does a degree at Essex they will have an ASR key, which is an academic stay record. Later if I come back to do a PHD then I have a new ASR key but the same PRID. For the one ASR key to contain all the information the university needs about you in one table is quite a lot to ask so they have to split it into relevant sections using your ASR key and sometimes other keys like Exam_Key. For the purpose of my section of the database the ASR Key is going to be the main key however I will need to split my tables for outgoing students into 3 main sections. The reason I have split them into 3 main sections is because they are time specific:

sb_cf

This is the starting point for study abroad students outgoing (from essex) SAO take their ASR key which will then later provide us with all of that students details such as course, email, name and much more. SAO will also take their preference choices. This is the first part of the process. SAO then has to talk to the university about if this is all possible. This table will link to PR_MM with a 1 to many relationship, The ASR key being the primary key of this table.

sb_allocation

This is the 2nd stage in the SAO process. One university will defiantly be confirmed, this then collects information about that university (some of which It gets from non_hessa.Padbas database) this contains information such as Erasmus code, university name, country code etc.
This table is linked to the sb_cf table in a 1 to 1 relationship, as well as the padbas table for university lookups in a 1 to * relationship again the ASR Key is the primary Key.

sb_comments

The 3rd stage of SAO checking information, this mostly happens while the student is on the placement or just after they have returned. Checking information like certificates of education has arrived.
This links to the allocation table in a 1 to 1 relationship and uses the ASR Key as its primary key.

The incoming section of the table design is structured in a slightly different way, as we have no previous information about incoming students these two tables are only connected to each other; however they still would be part of the studb database group because of their data type.

sb_incoming

collects basic information such as name, address, current place of study etc. -This links to the sb_incoming_courses table in a 1 to 1 relationship using the inc_id as a primary key.

sb_incoming_courses

I have split this into its own table to store all the course information relating back to that student. If a student returns again to essex they will already have a PRID and not have to renew this process.

This links to the sb_incoming table in a 1 to 1 relationship using the inc_id as a primary key.

4.4.2 Data warehousing and clustered keys

I looked into data warehousing for when historic data needs to be put to one side and I found by using a cluster key I could solve this problem and not have to create a data warehouse, searching will remain at a good speed even when at high capacity. I added a clustered key to the each table using its primary key. This can be seen in [Appendix C: SQL Code]

4.4.3 Queries

SAO have asked for a list of queries which are attached in [Appendix 9: List of Angela's Query Requirements] but below I explain some of the designs of the queries and attach the designs in the appendix.

4.4.3.1 Incoming Students

- Fee Paying Students
  SAO need to calculate the data for the total fee paying students from each country that we receive in a certain year, for this I had to design a query that could take a count field, to calculate the total number of students from that country, SQL has a built in count function which I used. Then to determine which year that the search is calculating by I have designed it to link to a asp variable that it will receive the date from. Finally I group by everything that is not the 'count' function otherwise SQL will return errors. [Appendix 12: Fee paying students]

- Total Students Abroad
  SAO need to see the total students abroad for this I use the count function to calculate the total number of students abroad with a group by rollup feature that will then count the total number of counts, giving me a grand total of the number of students currently abroad. [Appendix 13: Total students abroad]

- Showing people who haven't listed all information
  Some students may not submit all the documents on time, this query runs a check on all incoming students to see if no information has been inserted into some of the fields relating to them, this is done by using the WHERE clause in SQL Where field X IS NOT NULL or field Z IS NOT NULL. This query checks all fields of blank data. I added a code snipped in [Appendix 14:incoming students who haven't listed their information]

- Who came to the university
  this was a basic select statement.

- Exchange difference
  the exchange difference worked at Design stage, but implementation failed. I was using an outer join of two SQL Select statements to count compare side by side the total incoming and the total outgoing for countries, however it didn't work quite as I planned and I will discuss this in further detail in implementation [section 5: implementation].

4.4.3.2 Outgoing students

- Students abroad over several years
  SAO needed to find out how many students went abroad over several given years i.e 2001 to 2005 in a particular semester i.e. AU, SP or FY for this I needed to select information from two different tables to do this I used an inner join as I didn't not want to create duplicate results I stopped the creating of duplicates by using the '=' operator in the from clause. From here I then needed to have a count field to count the number per country.

Then I had to go away and research how to select something using a range of dates instead of one fixed date. I found that I can use the DATEDIF operator available in SQL which lets me input two different dates or a range of year's i.e over the last 5 years. I spoke to Angela and she preferred the range idea. Finally to see the grand total I will implement with the roll up feature. I have the code in: [Appendix 15: outgoing students over several years]

- Annual report
  To produce an annual report of all the basic information on what student went to what country and university, then the basic information on that student as well. I need to select information from 3 different tables. Inner joins will be used to connect the tables together using the ASR KEY as the identifier to join them. Finally I will have to use an ASP variable to use a where clause i.e. where date = asp variable. [Appendix 16: outgoing annual report]

- Department and Year
  For outgoing students SAO needed to searcher slightly deeper and define what department a student came from and what semester, for this I had two asp variables and to show the results I was selecting from multiple tables using inner joins and counts statements, finally summarising it with the WHERE clause specifying the data must equal the variables. [Appendix 17: outgoing department year]

- Outgoing late report
  Very similar to the incoming late report just checking different field names.

4.4.4 Stored procedure

To implement the search facility I needed for write a procedure that could handle searching based on criteria from certain fields and then return the correct values. This is done using the 'LIKE' function in SQL. It works quite effectively as I am only searching basic things such as names. Once the search is completed the values will return to my ASP.NET data set which I discuss in the implementation stage.

4.4.5 Alter Function

I needed to create a primary key for the incoming students; I could add an auto number to the table which will just give an incremental value based on the last entry for every added member. I didn't feel this was the best way of making a primary key as it needed to have some reference to it being an incoming primary key so I added "INC" to the front of the key, so for example "INC00001" I did this by creating a ALTER function that incremented a my type of primary key based on a normal auto number [Appendix 10: SQL Alter Function]

4.5 Main User Interface design

When it came round to starting the main user design, it was more of the implementation stage, I took my designs from my initial design (the university of Essex home theme) and used them as a basis however I started a new project within visual studio and just looked back at them as a reference. I knew that I wanted all the pages to be clear and clean like the current styling of the Essex university website and this is what I did. My project was growing so big with the amount of pages I was creating it would have taken me along to design each page. Instead of designing each page I just used one master page which was the basis of every other page and a style sheet that that laid everything out on the given page depending on what it was. You could almost say that that after the initial designs, I created a main template in my new project file in visual studio “final project” and all the pages I created revolved around the default structure but I created then in a “agile” way, created them as I needed them. I will explain all the pages I created in the Implementation section along with the queries.

Section 5: Implementation

Section 5: Implementation

5.1 Overview of this section

The implementation section describes how the project was produced as a whole and how each part of the project fits together. This section will mainly discuss the user interface as this was the main part of the implementation.

5.2 Implementing the interface

Master Page – ASP uses a technology called master pages I used one master page which will be the default frame for every page, this will do things such as attach the style sheet to every linked page. This saved me lots of time and the master page was based/take from my designs. Part of the code in the master page is that it checks to see if the user is logged in; if the user is not logged in it diverts them to the login page.

I my style sheets is just a list of fonts that I have used, mostly taken from Essex universities style sheet [11]

Each ASP page that I create I have a default set of includes, which use a certain set of API's this is included in the [appendix 20: List of Includes] I need this for certain methods such as when communicating with SQL server.

Login page – The login page is the first point of call, if the user is not already signed in this uses ASP.NET Login facility which allows users to be added via the built in administrator page. I wanted to use this way because later if my project does get used by the university they can update the login page via the administrator page to take the Essex University logon credentials instead. If you fail to login it will the user will not be able to view any pages further than the login. If login is successful it will redirect you to page2.aspx

Page 2 – This is my index page so to speak, which has no real functionality other than to redirect you to the needed page. This was the first page I started to implement the style sheet. I had to tag everything correctly to keep the formatting of the Essex university theme. I also added a ASP.NET control that displays the login user name at the top of the page so the user can check that they are not accidently already logged in as someone else.

Incoming – The main incoming page has two main control buttons that use a sever.transfer to redirect you to other pages.

Add incoming

Add incoming is a data capture form page, it has a series of tick boxes, input boxes and drop down lists. This is to capture all the necessary data including the student's module choices in one page.

The dropdown list that lets you choose the country feeds back information to the asp page, so that it only shows you universities that are listed in that country. It does this by using the post-back method in the country drop down list then running a query and adding all the results to a data reader which is then fed into the drop down list for universities, this is in [appendix21: displaying information in HTML from data reader]

The on submit button, opens connections to, two different database tables adding all the information about the student to sb_incoming and the information about the courses to sb_incoming_courses. This then individually puts each record into the correct field. I have attached the code for the indentation into the [Appendix 19: inserting data into two tables]

If the submit button is pressed and the date has been incorrectly inserted it will generate a validation error and again if key fields are not input such as passport number.

Search incoming

The incoming search facility allows you to enter the student's name that you are searching for to find some results (however the search will give you a validation error message if nothing is inserted) the results will then display below the search bar. This is reading information from a data reader and displaying it in a table. The user can then click more information which takes the current users ID and forwards it to the next page were information is displayed based on that number. I have attached a copy of the data readers code in the [appendix21: displaying information in HTML from data reader]

Incoming results

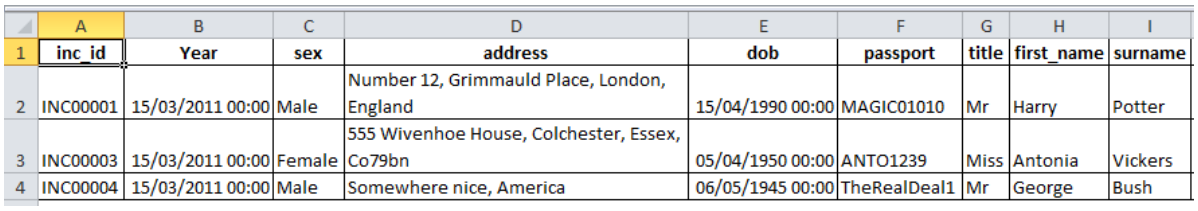
The incoming search results take the user ID, from the selected result on the previous page then run a query based on the users ID/primary key which displays all the information on the particular student. Originally I was going to manually make lots of text boxes that the data was attached to, however I found the details view tool which enabled me to generate all the data quickly and attach my style sheet to the table.

In main

In-main is a basic transfer page allowing the user to select which query they would like to run, so queries require information such as date which is entered on this page then transferred to the relevant query on the next page.

Who came

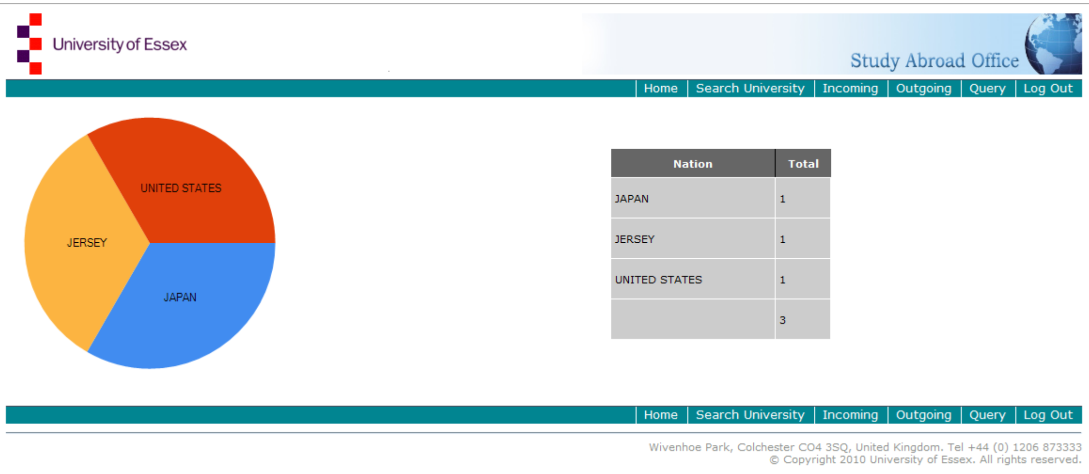
The who came page, display a list of people that came to our university. It shows information about that student i.e. were they were from, name etc. I wanted SAO to be able to export information to Excel like they previously could with some data they got from MIS. For me to export this to Excel I had use another include "using System.Collections.Generic;" which allowed me to interact with excel then write code for behind the 'export to excel' button which took all the information from gridview and converted to string method then passes in to Excel. Below is a screenshot of my export to excel



	A	B	C	D	E	F	G	H	I
1	inc_id	Year	sex	address	dob	passport	title	first_name	surname
2	INC00001	15/03/2011 00:00	Male	Number 12, Grimmauld Place, London, England	15/04/1990 00:00	MAGIC01010	Mr	Harry	Potter
3	INC00003	15/03/2011 00:00	Female	555 Wivenhoe House, Colchester, Essex, Co79bn	05/04/1950 00:00	ANTO1239	Miss	Antonia	Vickers
4	INC00004	15/03/2011 00:00	Male	Somewhere nice, America	06/05/1945 00:00	TheRealDeal1	Mr	George	Bush

Total abroad

Total abroad students who are/have been visiting us from abroad. For this page I coded the query in a SQL data source, and then linked a gridview display to the results. The reason I used grid is because it's such a user friendly way to build pages in asp. Instead of hardcoding the data source (which I do later for another query) grid view dynamically generates information based on a data source which then allows me to attach a style sheet to it. For this example I wanted to give SAO a graphically representation as well, so I linked the data source to pie chart and this was generated. Screen shot below.


University of Essex
Study Abroad Office
Home | Search University | Incoming | Outgoing | Query | Log Out
UNITED STATES
JERSEY
JAPAN
Home | Search University | Incoming | Outgoing | Query | Log Out
Wivenhoe Park, Colchester CO4 3SQ, United Kingdom. Tel +44 (0) 1206 873333
© Copyright 2010 University of Essex. All rights reserved.

Nation	Total
JAPAN	1
JERSEY	1
UNITED STATES	1
	3

Both country

The both country section was supposed to display a balance exchange in a table format but unfortunately I couldn't get this to work, This will be mentioned later in the problems section below. In the end I have to go for using to select queries to show to total students from both countries and display them in pie chart format next to each other. This is not particularly what I planned to do but still did the job reasonably well.

Fee paying

The way I designed fee paying was a bit more dynamic it had two queries on one page but they were both set to be hidden if they had no data in them. The information was passed from the page before in a server.transfer method + the year, this would display in the top address bar of your browser. This was a variable designed to be assigned to one of the labels on the next page. If a label has information attached to it than the relevant data source can read it generating the necessary table. However if no information is passed the page will display nothing.

show info of nulls

This runs the query mentioned in the designs that searches through all incoming students to make sure the documents are submitted then displays this in grid view to show what students have submitted their details.

Outgoing- The main outgoing page has two main control buttons that use a sever.transfer to redirect you to other pages.

Outgoing Choices

The outgoing choices page, wants you to insert the students ASR key if the students key is already in the database I have written a java script message to block the entry as it's already in the database. This is done with a try catch method on inputting it into the database. The insertion into the database is manual, taking each field and manually adding the entry. This page also allows you to input a student's disability and by clicking the link will open a pop up box with all the disability codes. Once submit is clicked it will take you to the outgoing profile page were more details are displayed.

Outgoing Allocation

Very similar to outgoing choices, entries already in the database and manually adds the information to the database, the drop down box will also change the list of universities based of which country is selected [Appendix 18: Nation Selected Change] this is done with a SQL query and data reader.

Outgoing Comments

Very similar to outgoing choices, entries already in the database and manually adds the information to the database

Outgoing Profile

At the end of each section (choices, allocation, comments) when submit is pressed it transfers you to outgoing profile page which is building a profile on the set student. It does this by referencing it to a student's ASR key. The top part of the profile page is the student information which is all attached from the database manually. This was very time consuming and I had to write a long function for this to work, it wasn't a particular hard task just tedious. This was one of the first things I did and from here I started using grid view which worked out a lot better. I am glad I did hard code this because you should learn the technique before using grid view, which I did. I have attached the code to the [Appendix 19: inserting data into two tables] . Once you can see the tables at the bottom it allows you to update the columns allowing the user to change information.

5.2.1 Query Outgoing

The outgoing section is mostly very similar to the techniques used in the incoming section so I will refer back to that somewhat.

Out main

Out main is just a linking page to all the other query pages for the outgoing section. It does allow for some user input i.e. choosing the year that the query is run from.

Department year

The department year uses a data source and grid view to output

Late report

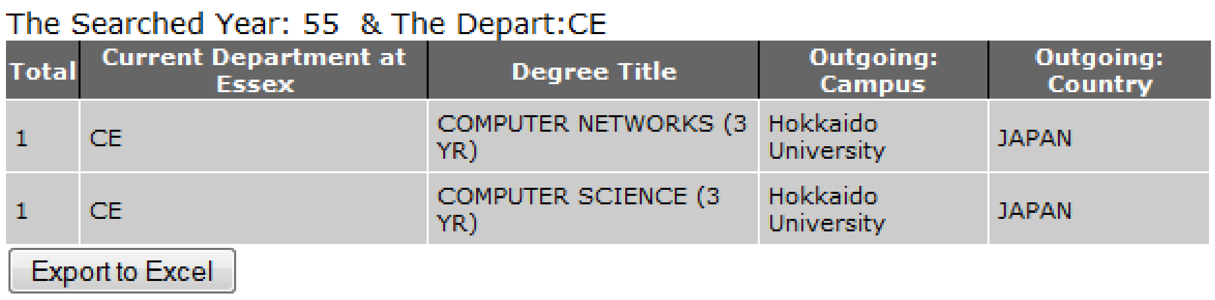
Like the late report for incoming students, just shows a list of students who haven't had the full amount of data inserted i.e. date back is left blank.

Search outgoing & outgoing results

Works in the same way as searching incoming students, using the SQL LIKE function but for outgoing students, once the page has loaded you have the ability to update information that are in the tables by clicking update this uses the update clause in grid view.

Student locations

A student location, display's a variety of things you can choose to see the amount of people from one department over one select year or over several. You can the export the grid to excel if the user wishes with the export function I mentioned earlier.


The Searched Year: 55 & The Depart:CE

Total	Current Department at Essex	Degree Title	Outgoing: Campus	Outgoing: Country
1	CE	COMPUTER NETWORKS (3 YR)	Hokkaido University	JAPAN
1	CE	COMPUTER SCIENCE (3 YR)	Hokkaido University	JAPAN

Annual report

Annual report is a year list of students that went away and there location, this is displayed in grid view.

Disability

The disability page is set to be a small popup window on certain pages to show what number is the reference for a disability

University search and university display.

This works in the same way as search incoming and incoming results. Using the SQL LIKE search facility to return results from the padbas database, of all non hessa universities. The user can then click more information to bring up all of the universities details.

5.3 The Database

The implementation of the database was fairly straight forward, as I mentioned previously most of the work that needed doing was in the design section. To save time I already had made my notes in SQL server.

When I created the database for the first time it failed because of a couple of syntax/spelling errors, but other than that loaded fine. I then had to insert all the test information that John Fell had given this was a lot of information and couldn't be done manually. I tried numerous ways of inserting it and most were giving me errors of some sort or were very time consuming such as inserting rows manually. I then started to research a bulk insert. This worked brilliantly in the end, I was able to insert hundreds of rows at a time. This was all done via SQL server using the text based compiler that it has built in.

I was now at the point where the database was ready to be deployed so I had to start getting the database working with Visual studio which was slightly trickier than I had hoped. For some reason it didn't like having SQL server running at the same time so I had to detach the database every time I wanted to use it in the visual studio environment. Eventually I kept it permanently attached to the visual studio interface were I could do most changes from there. After a while of using it visual studio I could see that there are some benefits it allowed me to view table data and edit it a lot quicker, with the interface instead of using the command line for SQL server.

I then went on to use the queries I had made in my designs to implement in visual studio, I done this in one of two ways. By hardcoding it into the ASP code or using the Microsoft grid view tool, which was a great feature and a much quicker and less error prone way of programming my queries.

I also had to insert the data into the database from the interface (new students etc), this was hard coded almost every time as it had to insert each field directly into the database. I am sure there is a quicker way of doing this but sometimes I wanted to change the order they were inserted and also to insert information into several tables from one form in some occasions.

Displaying an individual student's data was done in things like grid views by running queries based on the students ID.

5.4 The problems and solutions

When I showed John Fell my first database design he said it was a bit flat because I could have normalized a bit further 'decoupling' the database so to speak. When I changed my database to the SQL main version I separated things a lot more, but I was also using the studb database so I removed a lot of my student information tables. In the end this worked out fine and the database was suitable, I double checked my normalization and database layout with Mr Jerome Robertson a Database Lecturer at Essex University.

Inserting data via the bulk insert method provided to be a very good way of inserting data and also a very troublesome. When trying to insert 5,000 rows I would quite regularly see the message in the SQL console box '99 Max errors reached' which is always satisfying. Where I was inserting on a comma delimited method but parts of address for universities had a comma in the name, so I had to remove or replace the comma for the data to be processed. Around 95% of the time most entries were fine and this was by far the quickest way.

SQL server and Visual studio working together, became a frustrating task because at first it would not connect to the database then after a while it wouldn't connect to the database If SQL server was still attached. In the end I had to write the code to attach the database to set a location from visual studio and it seemed to work fine after that point.

Stored procedures were fairly problematic as for quite a while I had one version of my search working (for universities) and the search for students working incorrectly. I spent ages trying to reconfigure and work out what the issue was and I finally realised I had a extra '%' symbol a small error that took me a long time trying to figure it out. One of the best things I have learnt from the project sometimes it is best to walk away from the problem in hand and have a break instead of trying most method and still getting it wrong due to it being a smaller error than I originally thought.

The Alter function creating a unique number i.e. INC0001 was a very strange thing to create because I didn't think it would be as complex as It was, to create a number like sequence like 1.2.3.4.5 is fairly straight forward as you can use an auto-number but for the INC0001 it was harder to create because it needed an auto-number to reference its numbering structure to. I found out that if I had used MYSQL it would have been simpler.

The most time consuming errors in ASP.NET were compilation errors. For example if I change a setting on page 1 I then for some reason cannot load page 2 after I have changed something there. I end up fiddling with page 2 and the error message wasn't very clear. After spending some time with ASP I started to realise that message sometimes meant slightly different things which was ok after the initial working out what they meant, however when I started I did have to delete a few pages and start again because of a simple error it just kept failing to load.

Section 6: Testing

Section 6: Testing

6.1 Overview of the section

To test my project for SAO, I carried out testing while implementing then tested the whole project again as a final piece.

6.2 My own Testing

I carried out testing while implementing then tested my entire project over again to test for bugs and leaks. Most of the real problems were discussed earlier in the project section

I went on to test every individual pages and forms loading correctly everything seemed to be in order and if I find any more bugs I can amend them over the rest of the Easter break.

As well as testing things generally shown in the table below, this gives comments on everything I have tested as a category. I have taken all these categories from the requirements.

	Yes/No/comment
• Incoming Students	
○ Add	I was able to add an incoming student
○ Modify	I can modify info
• Outgoing Students	
○ Add via previous record	This works perfectly and also stops duplicates from being created.
○ Modify study abroad data	
• Query	
○ Queries related to incoming students	Everything working
○ Queries related to outgoing students	Everything working
○ All of SAO's prebuilt queries [Appendix: 9 Angela Turtons query requirements]	Everything working
• Secure	The interface is only accessible with a password
• Maintainable	The system can be maintained with studb and interface could also be upgraded easily.
• User friendly	When testing the layout is very clear and friendly Angela also was getting to grips with it quickly. So yes this was a sucess
• Portable	You will be able to access this from any web enabled device to use anywhere on the campus network. So yes it is.
• Available & reliable	This cannot be tested unless it is on the server, however if it is on the server there should be no problems.

6.3 Testing with Angela Turton

Angela came to the lab to look at my project and have a sit down test/demonstration of how the project works. Angela is a non-technical user, so she has come to see how the project looks and feels as well as testing the functionality she knows she will need. Angela worked her way through the project adding new students, searching for current students as well as running queries. Her feedback was: overall very impressed. Angela likes how the reports can be exported to excel however need to make some changes on the following things:

- Exchange balance for exchange students
- Incoming year query not working on demo
- Full year au,sp,fy option for incoming students option
- Spelling mistake on outgoing student page

6.4 Test Conclusion

When I am talking about testing, the actual main part was done while I was building the system. I spent around half a day at the end of the program to check for bugs and anomalies, I think more things can be put in place to stop any bugs like exception and error handling on every possible link or function however I didn't get time to implement all of this and the project seems to be stable as it is with the testing I have done.

Section 7: Conclusion

Section 7: Conclusion

7.1 Overview of the section

This section of the report gives an overview on everything that has happened both positive and negative allowing me to critically asses the main focus points of the project as well as giving plans for a future iteration and a conclusive summary.

7.2 Task analysis

Task Analysis will critically assess the main sections of the project.

7.2.1 Design

The design was started straight after my initial planning, I knew that my designs for the front interface would be brief or actual implementation on visual studio which is what I did; this saved me a bit of time for the implementation stage as I already had the software working and installed. As for the database the main section was the design, I planned it with various ER diagrams laying it out then designed the table design in SQL server manager were I just had to run my designed code to implement for the implementation stage.

In summary I think the design of the database was well planned and structured but I could have spent more time on the interface, as I didn't know precisely how many pages there would be and what each page would contain.

7.2.2 Implementation

The main part without doing the previous assignment that I mentioned I would have had no real suitable language to develop this in so I was very lucky within a way. ASP.NET as a language was actually pretty enjoyable as far as programming goes, so much so I think it should be taught the 2nd year students as one of the optional modules for computer science students. The implementation its self was the toughest bit as little errors sometimes were hard to spot and took up a lot of time but saying that once I got used to the way ASP.NET operates I was getting a lot quicker towards the end. The development of the database was mainly designed based and the biggest problem I faced with SQL was some of the sizes on my fields were too small from my original designs.

In summary, I am very happy I used ASP.NET and I feel that I have really gained a good skill by doing this, especially on integrating a development with an ASP.NET database.

7.2.3 Testing

For my testing I felt that when I let someone use it might crash or fail, however I have been using my project for running report and gradual testing the whole way through my project then finally did to large test runs at the end. I think my paranoia of it crashing comes from knowing how much time effort and programming goes into most main stream products before they go live and even then we still see numerous crashes in normal life for example at ATM machines. To summarise the testing I would like to have had a more set out criteria to do some black and white testing on or even had another company test my project.

7.2.4 Gantt chart

I am a very visual person I plan a lot of my work on my white board I have at home. I make notes and lists about almost everything. However with the Gantt chart I didn't refer back to it as much as I would like to have done. I would like to try and explore new software that would suit my needs better. For example: I would like an application that splits each section into multiple layers or mile stones then for me to be able to extend a section to add more goals / jobs. Then each goal/job can have a comment and a progress bar, by clicking the milestone's you can also open the relevant document or file so that it is all kept together a bit more neatly. I feel that if the project management software was a bit more extensive I would have used it more and my Gantt chart would have been better.

In summary I just ended up using my logbook more to make notes and kept referring back to that as my reference instead of using my Gantt chart as much.

7.3 Final Conclusion

I have thoroughly enjoyed my project and working my supervisor Anthony Vickers. Overall I would say the product was a success as I have met my user requirements, I was especially pleased with the export to excel functionality that I put in. After demonstration day I was also a lot happier about my design as a lot of people had focused on the functionality were as I have made my project aesthetically pleasing as well.

Last words: I do believe the project was a success in terms of my ability and requirements however I would like attempt another iteration to add more components and to carry out more testing,

7.4 The Next Iteration

Looking forward, if I could plan the next iteration for my project I would like to carry out some more tasks:

- Bulk email capture

Instead of SAO having to individually find names, and ask departments to send out email mailshots. I believe it would be a good feature if they can email those who have already showed interest in placements or have already undertaken placements. For this you would need to create another table for the Studb database, recording all interested students, you would also have to ask permission to keep a record of them I believe.

- Add a better logon facility

The logon I currently use for the project is just a basic system in ASP.NET; it would be nice to use Essex university Logon system with group permissions.

- Decouple the interface

Some of the ways I programmed the interface wasn't the most efficient, I could have used more advanced programming techniques, which wouldn't have been a lot harder to do but time consuming after I had an already done the previous method. However some aspects did show decoupled programming styles, like the master pages, style sheets (detached) and references to data stores in the web.config file.

- Set reminders within the program of up and coming dates.

Some small features go a long way, when something is about to become overdue the system could display reminders for certain users giving them helpful messages.

- Email students automatically to chase them

Wouldn't life be easier if you didn't have to chase people for information? I think so. Building an automatic emailing system for items that are going to expire wouldn't be an easy task but very achievable, it would require alterations to the database as well as a new function being implemented.

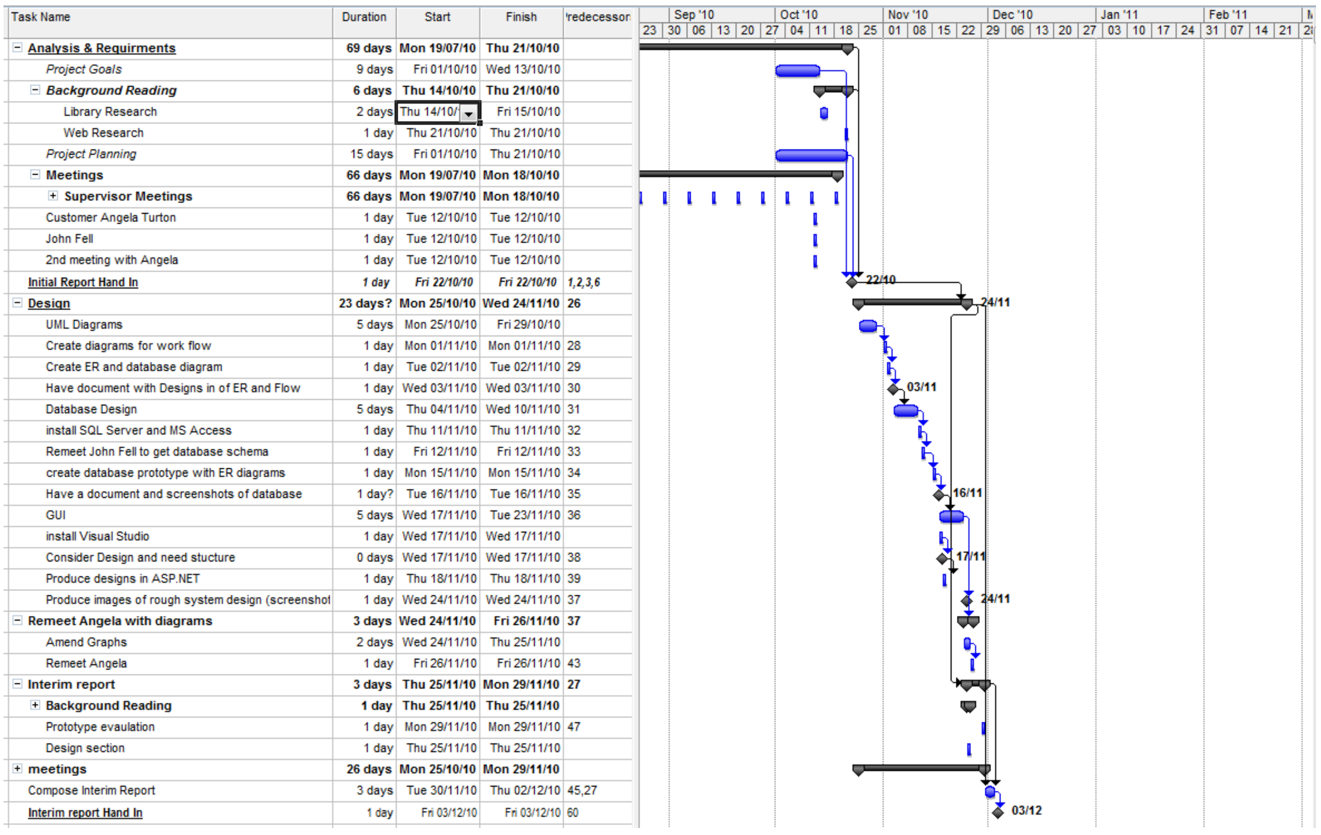
Section 8: Appendix

Section 8: Appendix

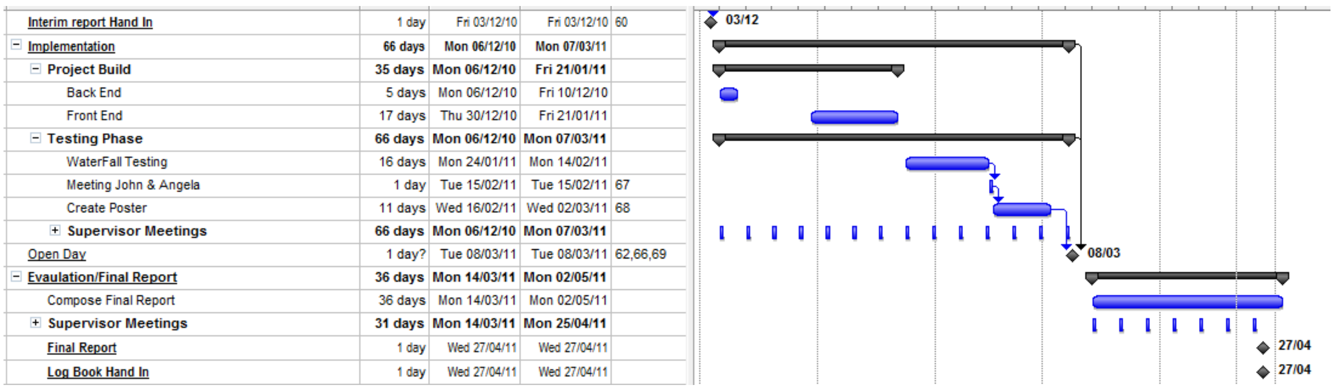
Appendix 1: Final Project Hand-out Extract

The project will be to design and construct a database from the following specification which has been produced by the University of Essex Study Abroad Office.
The system will permit the efficient production of management information and should interface with the University's student records system. Some information would be entered into the new database and transferred to student records, and some data would originate in the student records system and be transferred to the new database. The database would need to encompass both students coming to Essex for study abroad and Essex students going out on study abroad. Fields must include: name, nationality, fee status, degree scheme, home/host university, whether exchange or feepaying, duration of study abroad, year of study, amount of grant/scholarship, and agent used. [I would elaborate on all these.] It will need to be possible to sort data by any of these fields, and to search for both individual records and sets of records including/excluding particular data. The database should be able to generate simple reports.

Appendix 2: Gantt chart



Task Name	Duration	Start	Finish	'redecessor
Analysis & Requirments	**69 days**	**Mon 19/07/10**	**Thu 21/10/10**	
Project Goals	9 days	Fri 01/10/10	Wed 13/10/10	
Background Reading	**6 days**	**Thu 14/10/10**	**Thu 21/10/10**	
Library Research	2 days	Thu 14/10/	Fri 15/10/10	
Web Research	1 day	Thu 21/10/10	Thu 21/10/10	
Project Planning	15 days	Fri 01/10/10	Thu 21/10/10	
Meetings	**66 days**	**Mon 19/07/10**	**Mon 18/10/10**	
Supervisor Meetings	**66 days**	**Mon 19/07/10**	**Mon 18/10/10**	
Customer Angela Turton	1 day	Tue 12/10/10	Tue 12/10/10	
John Fell	1 day	Tue 12/10/10	Tue 12/10/10	
2nd meeting with Angela	1 day	Tue 12/10/10	Tue 12/10/10	
Initial Report Hand In	*1 day*	*Fri 22/10/10*	*Fri 22/10/10*	*1,2,3,6*
Design	**23 days?**	**Mon 25/10/10**	**Wed 24/11/10**	**26**
UML Diagrams	5 days	Mon 25/10/10	Fri 29/10/10	
Create diagrams for work flow	1 day	Mon 01/11/10	Mon 01/11/10	28
Create ER and database diagram	1 day	Tue 02/11/10	Tue 02/11/10	29
Have document with Designs in of ER and Flow	1 day	Wed 03/11/10	Wed 03/11/10	30
Database Design	5 days	Thu 04/11/10	Wed 10/11/10	31
install SQL Server and MS Access	1 day	Thu 11/11/10	Thu 11/11/10	32
Remeet John Fell to get database schema	1 day	Fri 12/11/10	Fri 12/11/10	33
create database prototype with ER diagrams	1 day	Mon 15/11/10	Mon 15/11/10	34
Have a document and screenshots of database	1 day?	Tue 16/11/10	Tue 16/11/10	35
GUI	5 days	Wed 17/11/10	Tue 23/11/10	36
install Visual Studio	1 day	Wed 17/11/10	Wed 17/11/10	
Consider Design and need stucture	0 days	Wed 17/11/10	Wed 17/11/10	38
Produce designs in ASP.NET	1 day	Thu 18/11/10	Thu 18/11/10	39
Produce images of rough system design (screenshot	1 day	Wed 24/11/10	Wed 24/11/10	37
Remeet Angela with diagrams	**3 days**	**Wed 24/11/10**	**Fri 26/11/10**	**37**
Amend Graphs	2 days	Wed 24/11/10	Thu 25/11/10	
Remeet Angela	1 day	Fri 26/11/10	Fri 26/11/10	43
Interim report	**3 days**	**Thu 25/11/10**	**Mon 29/11/10**	**27**
Background Reading	**1 day**	**Thu 25/11/10**	**Thu 25/11/10**	
Prototype evaulation	1 day	Mon 29/11/10	Mon 29/11/10	47
Design section	1 day	Thu 25/11/10	Thu 25/11/10	
meetings	**26 days**	**Mon 25/10/10**	**Mon 29/11/10**	
Compose Interim Report	3 days	Tue 30/11/10	Thu 02/12/10	45,27
Interim report Hand In	1 day	Fri 03/12/10	Fri 03/12/10	60

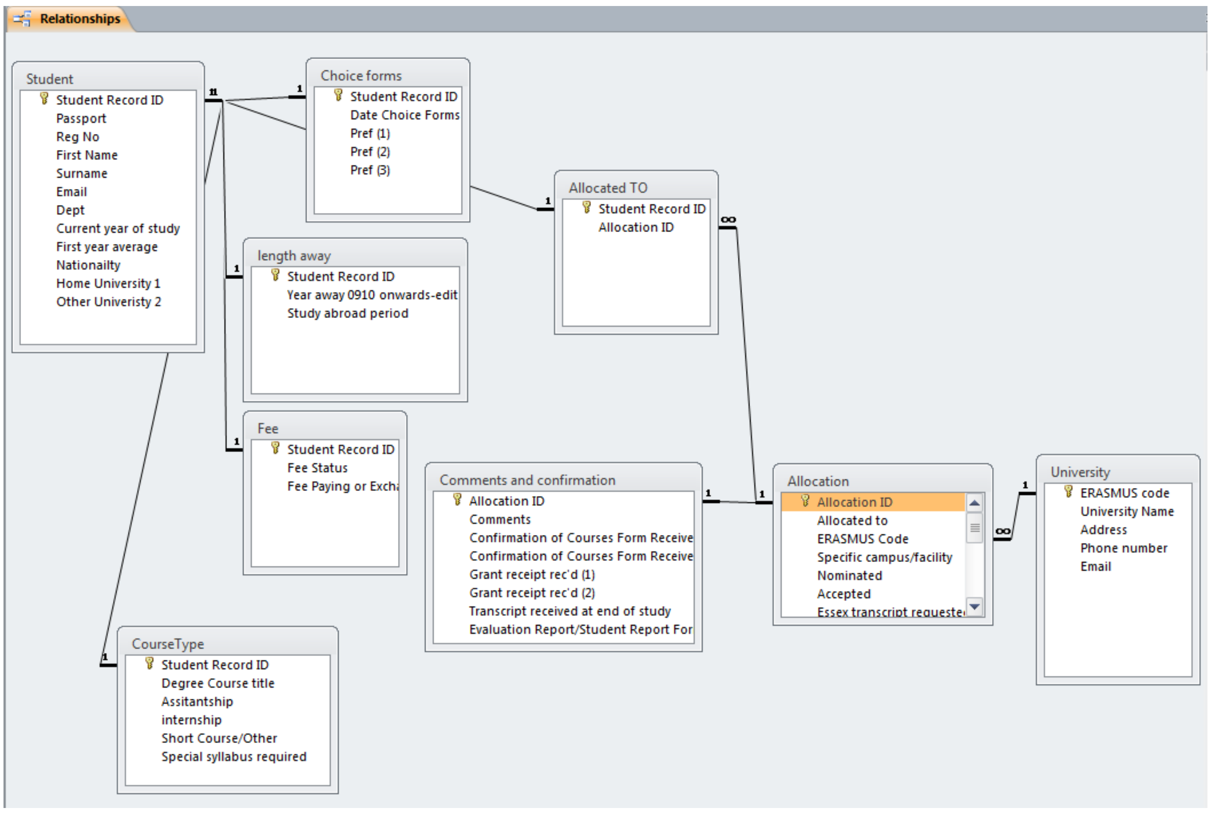


Interim report Hand In	1 day	Fri 03/12/10	Fri 03/12/10	60
Implementation	66 days	Mon 06/12/10	Mon 07/03/11	
Project Build	35 days	Mon 06/12/10	Fri 21/01/11	
Back End	5 days	Mon 06/12/10	Fri 10/12/10	
Front End	17 days	Thu 30/12/10	Fri 21/01/11	
Testing Phase	66 days	Mon 06/12/10	Mon 07/03/11	
WaterFall Testing	16 days	Mon 24/01/11	Mon 14/02/11	
Meeting John & Angela	1 day	Tue 15/02/11	Tue 15/02/11	67
Create Poster	11 days	Wed 16/02/11	Wed 02/03/11	68
Supervisor Meetings	66 days	Mon 06/12/10	Mon 07/03/11	
Open Day	1 day?	Tue 08/03/11	Tue 08/03/11	62,66,69
Evaulation/Final Report	36 days	Mon 14/03/11	Mon 02/05/11	
Compose Final Report	36 days	Mon 14/03/11	Mon 02/05/11	
Supervisor Meetings	31 days	Mon 14/03/11	Mon 25/04/11	
Final Report	1 day	Wed 27/04/11	Wed 27/04/11	
Log Book Hand In	1 day	Wed 27/04/11	Wed 27/04/11	

Appendix 3: SQL Code for main database

The Sql code is very long for the main structure of the report and I would please ask if you could refer to project CD which has been handed in. On this CD a file called SQL.SQL will be in the main directory, this contains all the code for the database (but not the queries, this is later.)

Appendix 4: Initial Access ER Diagram



Student
- Student Record ID
- Passport
- Reg No
- First Name
- Surname
- Email
- Dept
- Current year of study
- First year average
- Nationailty
- Home University 1
- Other Univeristy 2

Choice forms
- Student Record ID
- Date Choice Forms
- Pref (1)
- Pref (2)
- Pref (3)

Allocated TO
- Student Record ID
- Allocation ID

length away
- Student Record ID
- Year away 0910 onwards-edit
- Study abroad period

Fee
- Student Record ID
- Fee Status
- Fee Paying or Excha

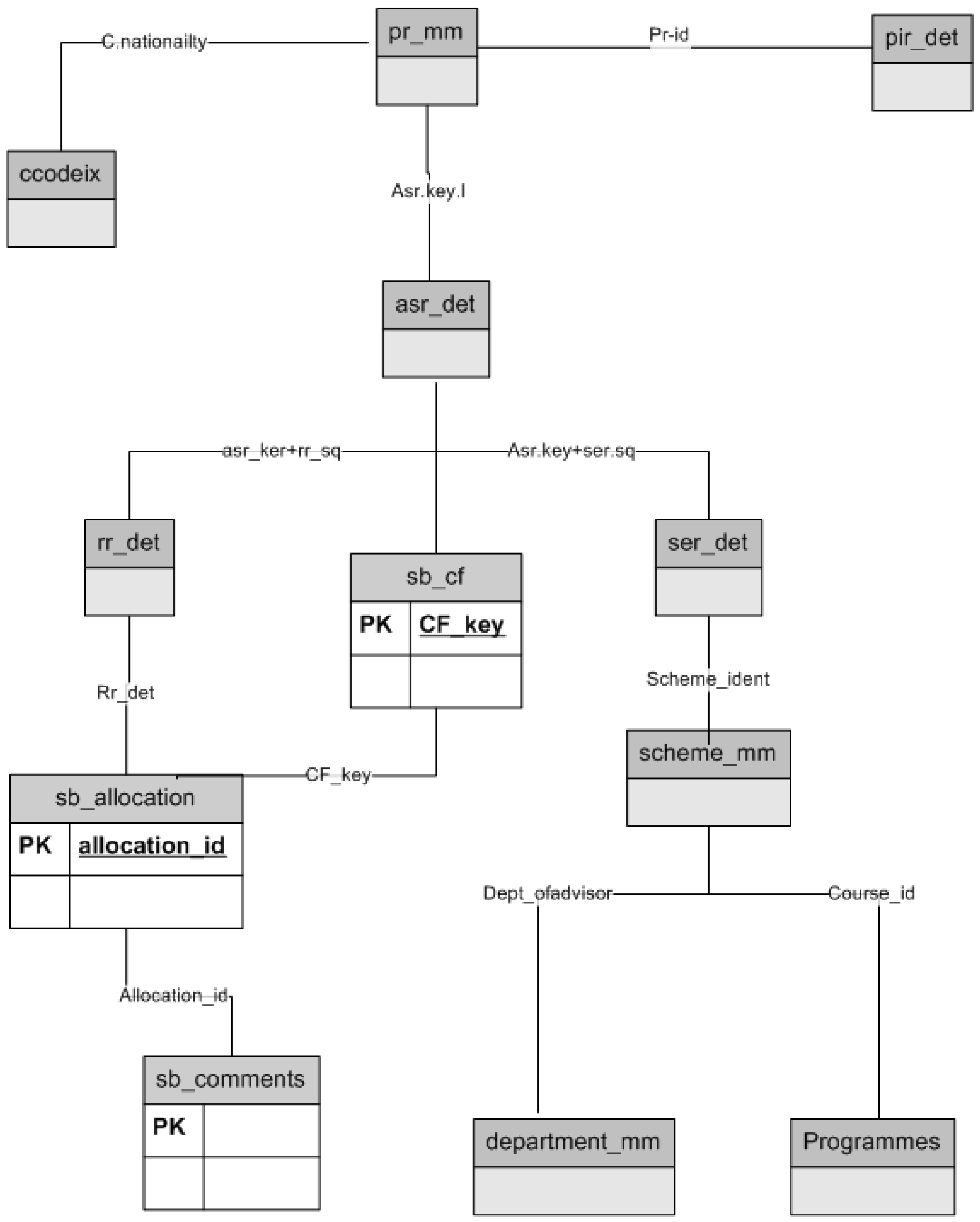
Comments and confirmation
- Allocation ID
- Comments
- Confirmation of Courses Form Receive
- Confirmation of Courses Form Receive
- Grant receipt rec'd (1)
- Grant receipt rec'd (2)
- Transcript received at end of study
- Evaluation Report/Student Report For

Allocation
- Allocation ID
- Allocated to
- ERASMUS Code
- Specific campus/facility
- Nominated
- Accepted
- Essex transcript requeste

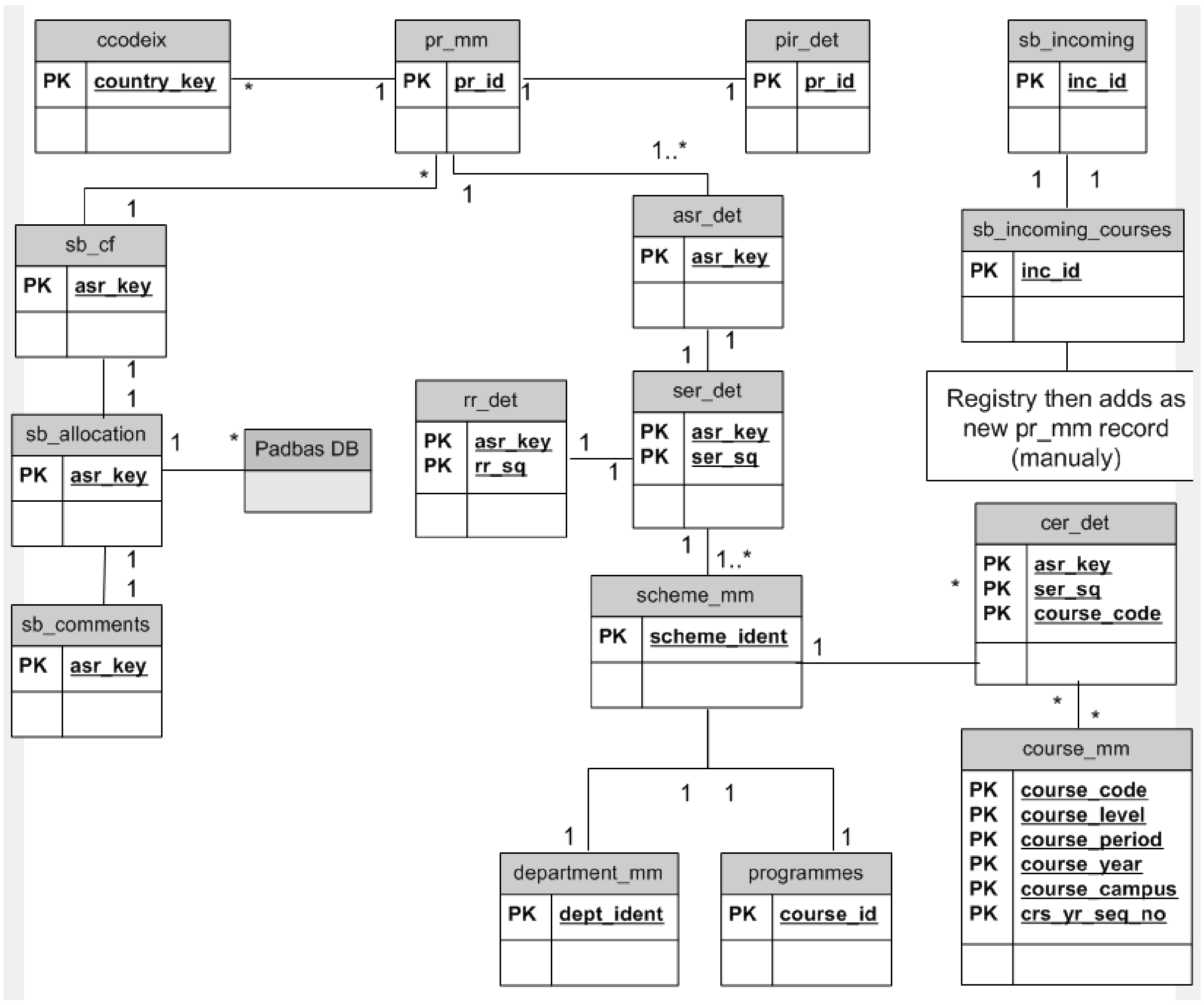
University
- ERASMUS code
- University Name
- Address
- Phone number
- Email

CourseType
- Student Record ID
- Degree Course title
- Assitantship
- internship
- Short Course/Other
- Special syllabus required

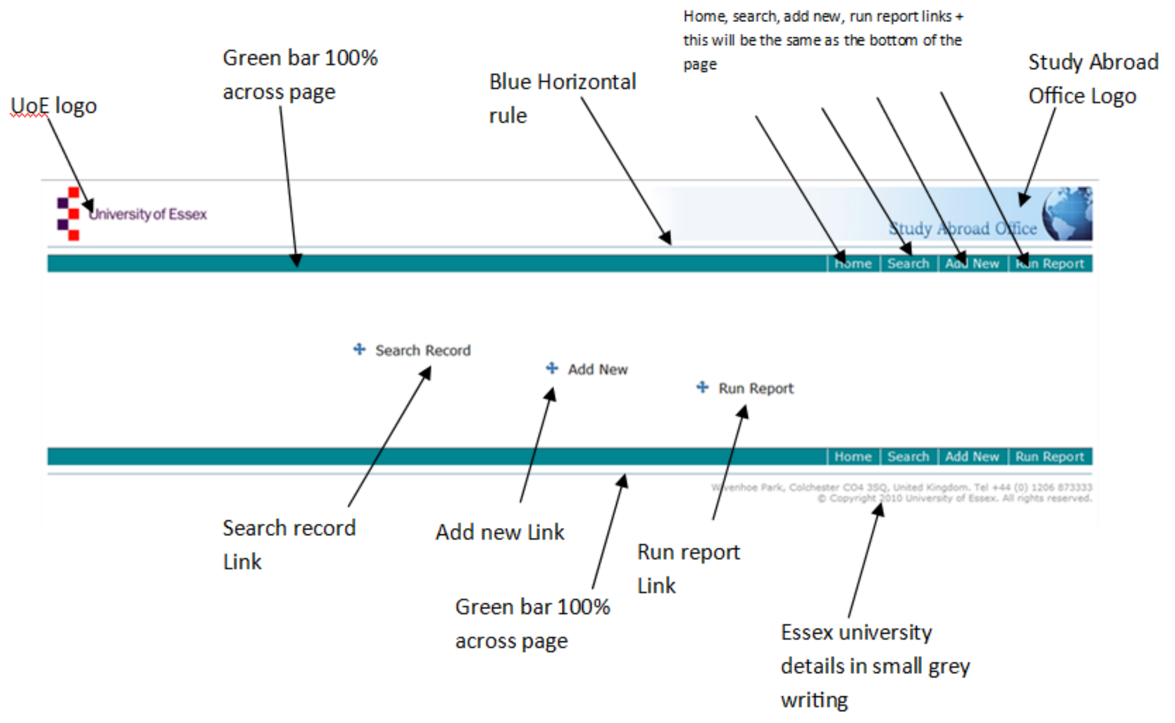
Appendix 5: Initial ER diagram



Appendix 6: Final ER diagram



Appendix 7: initial Designs


UoE logo
Green bar 100% across page
Blue Horizontal rule
Home, search, add new, run report links + this will be the same as the bottom of the page
Study Abroad Office Logo
University of Essex
Study Abroad Office
Home | Search | Add New | Run Report
Search Record
Add New
Run Report
Home | Search | Add New | Run Report
Wivenhoe Park, Colchester CO4 3SQ, United Kingdom. Tel +44 (0) 1206 873333
© Copyright 2010 University of Essex. All rights reserved.
Search record Link
Add new Link
Green bar 100% across page
Run report Link
Essex university details in small grey writing

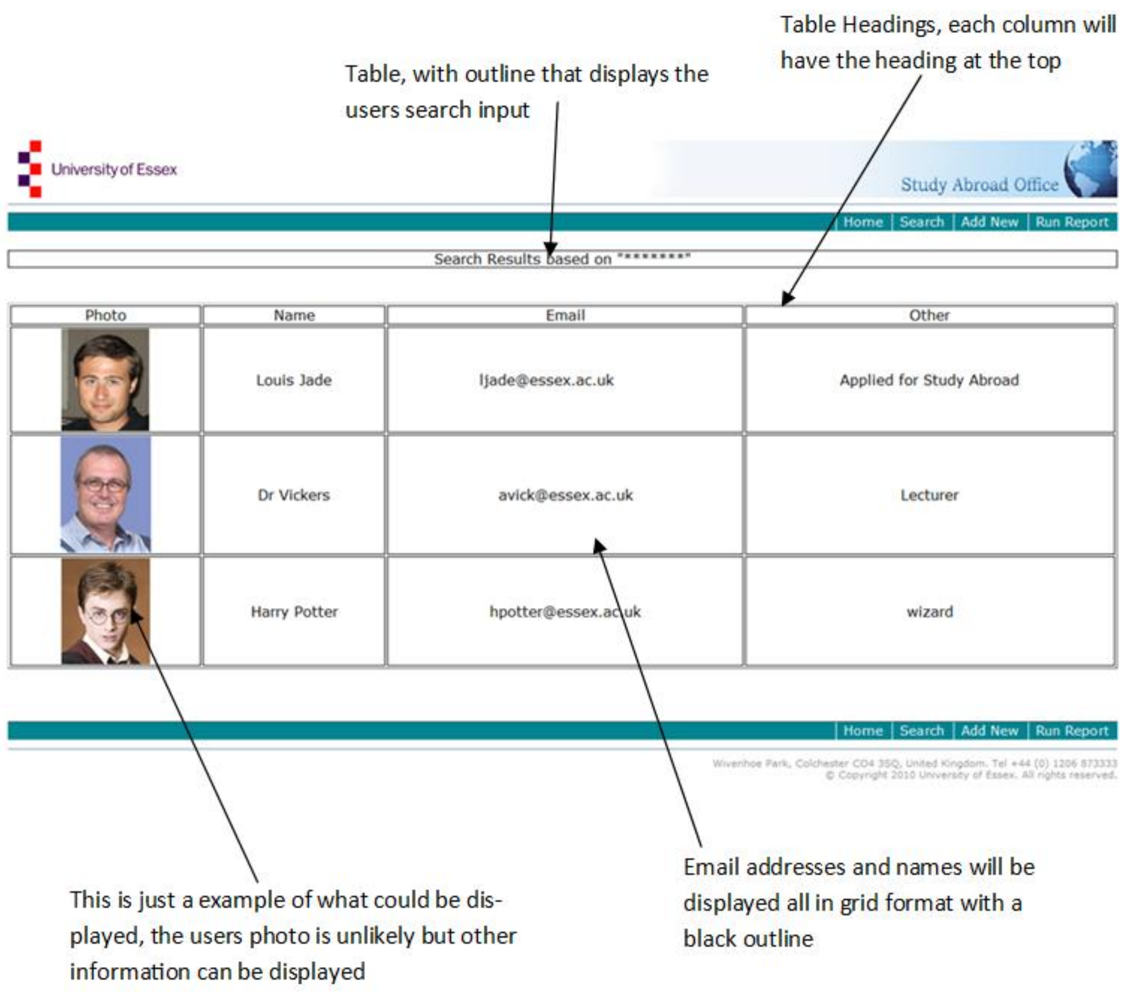
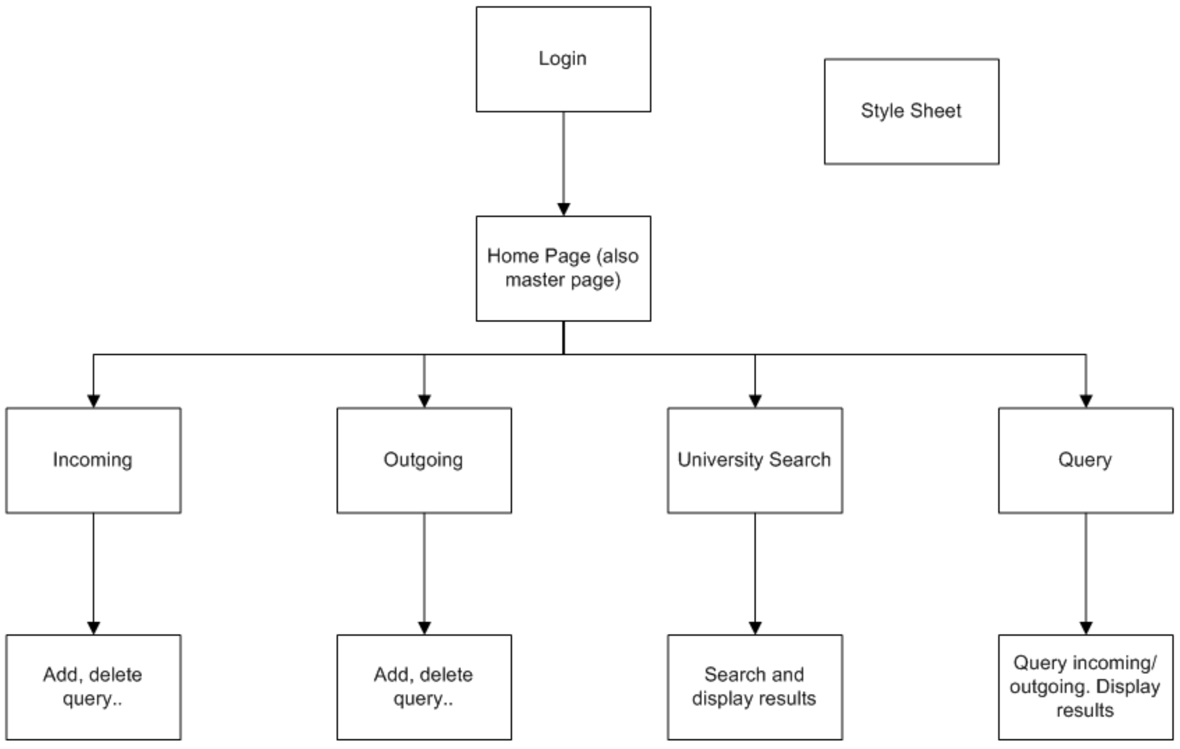


Table Headings, each column will have the heading at the top
Table, with outline that displays the users search input
University of Essex
Study Abroad Office
Home | Search | Add New | Run Report
Search Results based on "*********"
Home | Search | Add New | Run Report
Wivenhoe Park, Colchester CO4 3SQ, United Kingdom. Tel +44 (0) 1206 873333
© Copyright 2010 University of Essex. All rights reserved.
This is just a example of what could be displayed, the users photo is unlikely but other information can be displayed
Email addresses and names will be displayed all in grid format with a black outline

Photo	Name	Email	Other
	Louis Jade	ljade@essex.ac.uk	Applied for Study Abroad
	Dr Vickers	avick@essex.ac.uk	Lecturer
	Harry Potter	hpotter@essex.ac.uk	wizard

Appendix 8: Page Structure


Login
Style Sheet
Home Page (also master page)
Incoming
Outgoing
University Search
Query
Add, delete query..
Add, delete query..
Search and display results
Query incoming/ outgoing. Display results

Appendix 9: List of Angela Turton's Query Requirements

a) How many students who went abroad in a given year, or across several years, were home, how many were EU, and how many were overseas?

b) Which countries did students in each of those categories go to?

c) How many students in each category and each destination went for a semester, and how many for a year?

d) In a given year or period of years, which universities sent us fee-paying students, and how many, and for which period?

e) In a given year or period of years, how many fee-paying students did we receive? How many were EU and how many were overseas? How many students in each fee category came for the Autumn Term, how many for Spring/Summer, and how many for the full year?

f) How many students did we send/receive, and for what period (semester or year), to/from each exchange partner university in a given year?

g) What is the cumulative exchange balance? (That is, how many students have we sent/received and for how many semesters to/from a particular exchange partner over the entire history of the exchange?)

h) Obvious things, such as being able to search for a particular student.

i) We would like to be able to run reports to show which incoming students have not submitted an English language test score, which students have not submitted their acceptance form, etc. (We would like to do this for any of the incoming student items.)

j) For outgoing students, it would be useful to be able to run reports showing who has not submitted their passport, or their special syllabus form, etc.

k) We need to run an annual report showing the name of every outgoing student and which country he/she went to, as well as whether it is a semester or a year abroad.

l) How many students from each department went abroad in a given year or across several years? It would also be useful to be able to see that by course.

m) In a given year, how many incoming study abroad (both fee-paying and exchange) students were there for the Autumn Term, how many for Spring/Summer, and how many for full year?

Appendix 10 : SQL Alter Function

```
ALTER function incomingnumber (@id int)
returns char(8)
as
begin
return 'INC' + right('00000' + convert(varchar(10), @id), 5)
end
```

Appendix 11: Stored Procedure

```
ALTER PROCEDURE dbo.StoredProcedure1
@SearchString varchar(50)
AS
SELECT *
FROM sb_incoming
WHERE (first_name LIKE '%' + @SearchString + '%' OR surname LIKE '%' + @SearchString +
'%' OR inc_id LIKE '%' + @SearchString)
       RETURN
```

Appendix 12: Fee Paying Student Query

```
SELECT COUNT(nation) AS Total, sb_period, pay_exchange, home_uni, nation FROM
sb_incoming WHERE (Year = @year) GROUP BY sb_period, pay_exchange, home_uni, nation
```

Appendix 13: Total Students Aboard

```
SELECT       nation, COUNT(nation) AS total
FROM          sb_incoming
GROUP BY nation WITH ROLLUP
```

Appendix 14: incoming students who have not submitted information

```
prev_edu1, disability FROM sb_incoming WHERE (sex IS NULL) OR (address IS NULL) OR
(dob IS NULL)
```

Appendix 15: outgoing students over several years

```
SELECT       COUNT(sb_comments.asr_key) AS Total, sb_allocation.country
FROM          sb_comments INNER JOIN
                   sb_allocation ON sb_comments.asr_key = sb_allocation.asr_key
WHERE        (DATEDIFF(year, sb_comments.year_away, GETDATE()) <= @year)
and sb_comments.study_ab_period = @misc
GROUP BY sb_allocation.country WITH ROLLUP
```

Appendix 16: outgoing annual report

```
SELECT pr_mm.title, pr_mm.e_mail, pr_mm.asr_key_l, sb_allocation.country,
sb_comments.year_away, sb_comments.study_ab_period, pr_mm.other_names,
pr_mm.surname FROM sb_allocation INNER JOIN sb_cf ON sb_allocation.asr_key =
sb_cf.asr_key INNER JOIN sb_comments ON sb_allocation.asr_key = sb_comments.asr_key
INNER JOIN asr_det ON sb_allocation.asr_key = asr_det.asr_key INNER JOIN pr_mm ON
asr_det.pr_id = pr_mm.pr_id WHERE (sb_comments.year_away = @year)
```

Appendix 17: Outgoing Department and year

```
SELECT COUNT(programmes.current_dept) AS Number, programmes.current_dept,
programmes.sch_med_title, sb_allocation.spec_campus, sb_allocation.country FROM sb_cf
INNER JOIN sb_allocation ON sb_cf.asr_key = sb_allocation.asr_key INNER JOIN ser_det ON
sb_cf.asr_key = ser_det.asr_key INNER JOIN programmes ON ser_det.course_id =
programmes.course_id INNER JOIN sb_comments ON sb_allocation.asr_key =
sb_comments.asr_key WHERE (sb_comments.year_away = @yeartime) AND
(programmes.current_dept = @dept) GROUP BY programmes.current_dept,
sb_comments.year_away, programmes.sch_med_title, sb_allocation.spec_campus,
sb_allocation.country
```

Appendix 18: Nation Selected Change

```
        protected void nation_SelectedIndexChanged(object sender, EventArgs e)
        {

            SqlConnection conn1 = new
SqlConnection("Server=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\studb.mdf;Integr
ated Security=True;User Instance=True");
            conn1.Open();
            string strSQLCommand = "SELECT country_key FROM ccodeix WHERE
ccodeix.country_dsc ='" + nation.SelectedValue + "'";
            SqlCommand command = new SqlCommand(strSQLCommand, conn1);
            string key = (string)command.ExecuteScalar();
            conn1.Close();
            DataTable dt = new DataTable();
            SqlConnection connection = new
SqlConnection("Server=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\padbas.mdf;Integ
rated Security=True;User Instance=True");
            try
            {
                connection.Open();
                string sqlStatement = "SELECT instit_h_ed_name FROM non_hesa_mm WHERE
non_hesa_mm.country_key ='" + key + "'";
                SqlCommand sqlCmd = new SqlCommand(sqlStatement, connection);
                SqlDataAdapter sqlDa = new SqlDataAdapter(sqlCmd);

                sqlDa.Fill(dt);
                if (dt.Rows.Count > 0)
                {
                    home.DataSource = dt;
                    home.DataTextField = "instit_h_ed_name"; // the items to be
displayed in the list items
                    home.DataValueField = "instit_h_ed_name"; // the id of the items
displayed
                    home.DataBind();
                }
            }
            catch (System.Data.SqlClient.SqlException ex)
            {
                string msg = "Fetch Error:";
                msg += ex.Message;
                throw new Exception(msg);
            }
            finally
            {
                connection.Close();
            }
        }
```

Appendix 19: inserting data into two tables

```
        protected void submit_Click(object sender, EventArgs e)
        {
            string strDate = DateTime.Now.ToLongDateString();
            SqlConnection connection2 = new
SqlConnection("Server=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\studb.mdf;Integr
ated Security=True;User Instance=True");
            SqlCommand myCommand2 = new SqlCommand("INSERT INTO sb_incoming
(sex,address,dob,passport,passport_img,title,first_name,surname,email,sb_period,home_u
ni,nation,feeamount,pay_exchange,current_crs_tit,current_study_sc,lang_score,ac_ref1,a
c_ref2,prev_edu1,prev_edu2,notes,disability,year) " +
                                             "Values ('" + mf.Text + "','" + address.Text +
"','" + d2.Text + "-" + d1.Text + "-" + year.Text + "','" + passportno.Text + "','" +
passportimage.Text + "','" + title.Text + "','" + firstname.Text + "','" +
surname.Text + "','" + email.Text + "','" + sb_period.Text + "','" + home.Text + "','"
+ nation.Text + "','" + feeamount.Text + "','" + payex.Text + "','" +
currentcourse.Text + "','" + score.Text + "','" + lang_score.Text + "','" + ac1.Text +
"','" + ac2.Text + "','" + previousedu1.Text + "','" + previousedu2.Text + "','" +
notes.Text + "','" + disability.Text + "', '" + strDate +"')", connection2);

            connection2.Open();
            myCommand2.ExecuteNonQuery();
            connection2.Close();




            SqlConnection connection4 = new
SqlConnection("Server=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\studb.mdf;Integr
ated Security=True;User Instance=True");
            SqlCommand myCommand4 = new SqlCommand("SELECT inc_id FROM sb_incoming",
connection4);

            connection4.Open();
            SqlDataReader r = myCommand4.ExecuteReader();
            string inc_id = "";
            while (r.Read())
            {
                inc_id = (string)r["inc_id"];
            }
            r.Close();

            connection4.Close();



            SqlConnection connection3 = new
SqlConnection("Server=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\studb.mdf;Integr
ated Security=True;User Instance=True");
            SqlCommand myCommand3 = new SqlCommand("INSERT INTO sb_incoming_courses
(inc_id,course_code1,course_title1,credits1,course_code2,course_title2,credits2,course
_code3,course_title3,credits3,course_code4,course_title4,credits4,course_code5,course_
title5,credits5,course_code6,course_title6,credits6,course_code7,course_title7,credits
7,course_code8,course_title8,credits8) " +
                                             "Values ('" + inc_id + "','" + crs1.Text + "','"
+ crt1.Text + "','" + cc1.Text + "','" + crs2.Text + "','" + crt2.Text + "','" +
cc2.Text + "','" + crs3.Text + "','" + crt3.Text + "','" + cc3.Text + "','" +
crs4.Text + "','" + crt4.Text + "','" + cc4.Text + "','" + crs5.Text + "','" +
crt5.Text + "','" + cc5.Text + "','" + crs6.Text + "','" + crt6.Text + "','" +
cc6.Text + "','" + crs7.Text + "','" + crt7.Text + "','" + cc7.Text + "','" +
crs8.Text + "','" + crt8.Text + "','" + cc8.Text + "')", connection3);

            connection3.Open();
            myCommand3.ExecuteNonQuery();
            connection3.Close();

            Server.Transfer("page2.aspx");
        }
```

Appendix 20: List of includes

```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Data.SqlClient;
```

Appendix 21: displaying information in HTML from data reader

```
<tr><td align="center"><%#DataBinder.Eval(Container.DataItem, "first_name")%></a></td>
<td align="center"><%#DataBinder.Eval(Container.DataItem, "surname")%></td>
<td align="center"> <asp:HyperLink ID="l1" runat="server" NavigateUrl='<%#
DataBinder.Eval(Container.DataItem, "inc_id", "incomingsearchresults.aspx?item={0}")
%>'><span>More<span></asp:HyperLink></a> </td></tr>
```

Appendix 22: Excel Transfer Code

```
        public override void VerifyRenderingInServerForm(Control control)
        {

            //This overides the runat=server method, so i can use it within a main
        }

        protected void BtnExport_Click(object sender, EventArgs e)
        {
            Response.Clear();
            Response.AddHeader("content-disposition",
"attachment;filename=STUDYABROAD.xls");
            Response.Charset = "";
            Response.ContentType = "application/vnd.xls";
            System.IO.StringWriter stringWrite = new System.IO.StringWriter();
            System.Web.UI.HtmlTextWriter htmlWrite = new HtmlTextWriter(stringWrite);
            GridView1.RenderControl(htmlWrite);
            Response.Write(stringWrite.ToString());
            Response.End();
        }
```

Appendix 23: Outgoing Profile

```
   using (SqlConnection conn = new SqlConnection("Data
Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\studb.mdf;Integrated
Security=True;User Instance=True"))
            {
                SqlCommand selectCommand = new SqlCommand("SELECT * FROM pr_mm where
asr_key_l='" + item + "'", conn);
                conn.Open();
                SqlDataReader r = selectCommand.ExecuteReader();
                //string inc_id = "";
                while (r.Read())
                {
                    try
                    {
                        title.Text = (string)r["title"];
                    }
                    catch (Exception)
                    {
                        //
                        title.Text = "NULL";

                    }

                    try
                    {
                        firstname.Text = (string)r["other_names"];
                    }
                    catch (Exception)
                    {
                        //
                        firstname.Text = "NULL";

                    }

                    try
                    {
                        Name.Text = (string)r["surname"];
                    }
                    catch (Exception)
                    {
                        //
                        Name.Text = "NULL";

                    }

                    try
                    {
                        email.Text = (string)r["e_mail"];
                    }
                    catch (Exception)
                    {
                        //
                        email.Text = "NULL";

                    }
                    try
                    {
                        sex.Text = (string)r["sex"];
                    }
                    catch (Exception)
                    {
                        //
                        sex.Text = "NULL";

                    }
                }

                r.Close();
                conn.Close();
            }
```

Appendix 24: Database insert and java script blocking

```
        protected void Submit_Click(object sender, EventArgs e)
         {
             try
             {
                 SqlConnection connection2 = new
SqlConnection("Server=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\studb.mdf;Integr
ated Security=True;User Instance=True");
                 SqlCommand myCommand2 = new SqlCommand("INSERT INTO sb_cf
(asr_key,pref1,pref2,pref3,disability) " +
                                         "Values ('" + asrkey.Text + "','" +
pref1.Text + "','" + pref2.Text + "','" + pref3.Text + "','" + disability.Text + "')",
connection2);

                 connection2.Open();
                 myCommand2.ExecuteNonQuery();
                 connection2.Close();
                 Server.Transfer("outprofile.aspx?item=" + asrkey.Text);


             }
             catch (SqlException sqlEx)
             {
                 Response.Write("<script type=\"text/javascript\"
language=\"javascript\">");
                 Response.Write("alert('" + "That Student is already in the database."
+ "');");
                 Response.Write("</script>");

             }
             finally
             {

             }
         }
         protected void Lookup_Click(object sender, EventArgs e)
         {
              Lookup.Attributes.Add("OnClick",
"window.open('disability.aspx',null,'Height=250;left=20;Width=20;top=50;menubar=no;too
lbar=no;scrollbars=no');");
```

Section 9: Bibliography

Section 9: Bibliography

[1] http://courses.essex.ac.uk/ce/ce235/ Course repository for CE235

[2] http://courses.essex.ac.uk/ce/ce311/ Course repository for CE311

[3] http://www.compuware.com/products.asp Uniface vendors

[4] management information systems, managing the digital economy 10th edition by Kenneth C Laudon and Jane P Plaudon

[5] http://www.informationweek.com/news/global-cio/showArticle.jhtml?articleID=187200771 Hilton article

[6] Bednar A. James .- http://www.inf.ed.ac.uk/teaching/courses/sapm/2005-2006/slides/methodologies_4up.pdf Images for waterfall development and EP

[7] http://www.maverickconceptions.com/wp-content/uploads/2008/12/agile_development_training.gif Images for Agile development

[8] http://courses.essex.ac.uk.proxy.essex.ac.uk/ce/ce301/restricted/Scaling_Agile_22_Oct_10.pdf BT, mark Anning, scaling agile delivery

[9] http://courses.essex.ac.uk.proxy.essex.ac.uk/ce/ce320/restricted/ Course repository for CE320

[10] Codd, E.F. "Further Normalization of the Data Base Relational Model." (Presented at Courant Computer Science Symposia Series 6, "Data Base Systems," New York City, May 24th–25th, 1971.) IBM Research Report RJ909 (August 31st, 1971). Republished in Randall J. Rustin (ed.), Data Base Systems: Courant Computer Science Symposia Series 6. Prentice-Hall, 1972.
http://en.wikipedia.org/wiki/Third_normal_form#cite_note-Codd-0 Primary Key citation from Codd

[11] http://www.essex.ac.uk/style/essex.css Essex University style sheet

[12] My initial report (which can be found on the online coursework submission server)

[13] http://courses.essex.ac.uk.proxy.essex.ac.uk/ce/ce301/restricted/PROJECT_Street.pdf Phillip street's final report

[14] http://courses.essex.ac.uk.proxy.essex.ac.uk/ce/ce311/restricted/lectures/slides/314.html Web assignment using ASP.NET

[15] My interim report (which can be found on the online coursework submission server)

Project poster