UNIVERSITY OF HERTFORDSHIRE

School of Computer Science

MASTER OF SCIENCE IN COMPUTER SCIENCE

MCOM0088

Project Report

ONLINE LECTURER APPOINTMENT SYSTEM FOR PROJECT STUDENTS

M. Alqahtani

ID:07171945

Supervisor's Name: Dr.Steve, Bennett

Second Marker's Name: Dr. Dawkins, Keith

May 2010

Abstract

In the University of Hertfordshire, the lecturer for MSc projects has to schedule individual appointments at a specific time, date and location. Each appointment is for an individual student to present his MSc project to the supervisor and second marker. This procedure is implemented manually by the MSc project tutor. The programme leader has to schedule appointments for each student with specific examiners, taking into consideration time, date and location. Then, all appointments are published as a list via the study net. Following this, each student receives an announcement on his UH email, then has to open study net and read the details from the appointment list.

This project studies the existent solutions of online appointment booking systems, and then produces a new online lecturer based on criteria such as usability, light login system, and synchronisation with existing calendar system such as Google, Outlook and email integration. This system is evaluated against these terms.

Acknowledgements

Sincere acknowledgement is due to my project supervisor Dr. Steve Bennett for his guidance throughout each stage of the project. I would also like to thank him for his massive support throughout my project. I am indebted to him for his abundant patience and tolerance.

Sincere appreciation is due to Dr. Nasser, the programme tutor, for having confidence in me and supporting me throughout the different stages of the project.

Very special and heartfelt thanks are also due to my friends for their valuable contributions in the experiment and gathering the data.

Lastly, very special and heartfelt thanks for my family, especially my mother and my wife.

TABLE OF CONTENTS

1. Introduction

The introduction chapter will present the research overview, which includes the description of the research questions, project goal and the report outline.

1.1.Overview of the problem

Each semester in the University of Hertfordshire, the MSc project lecturer has to schedule individual appointments at a specific time, date and location. Each appointment is for an individual student to present his MSc project to both the project supervisor and the second marker. For example, if there are 60 students enrolled on the MSc project course, the lecturer has to schedule 60 individual appointments.

This procedure is implemented manually The programme leader has to schedule appointments for each student with specific examiners, taking into consideration time, date and location. Then, all appointments are published as a list via the study net. Following this, each student receives an announcement on his UH email, then has to open study net and read the details from the appointment list.

Therefore, there are many drawbacks in this procedure. The first disadvantage is time wasted by lecturer in manually scheduling the appointments list. The second is the difficulty of assigning an appointment time to a student without giving them a choice, which increases the chance of an inappropriate slot being arranged. Another weakness of this system is the difficulty to amend the list of appointments, such as changing the location or time for 60 students. This is related to the final drawback of this procedure, which is its inflexibly in terms of notifications of student via e-mail for publishing new appointments, cancellation or changing in appointment details.

The appointment system was first used in the health services as a paper work system in the 1970s in England. The aim of this system was to reduce the waiting time of the patients and distribute those patients to available consultants. After that, it was improved to be a computer system because of the benefits that go to both patients and consultants, such as reduction of waiting time. In this system, a patient can make his appointment on a specific day by phoning the health centre to check the availability and reserve a time with his doctor. The advantages can be taken from the health appointment system for both consultants and their clients are: effective organisation of workload, more efficient management of time, it limits the number of patients seen in one session, less waiting-room space is required, the duration of the consultation can be varied according to predetermined need and patients are able to plan their day. This view has been supported in the work of Robbins (2006, p55). The latest booking system used by the NHS is called “choose & book”, which is a service that enables the patient to choose via http://www.chooseandbook.nhs.uk/

a hospital and a specialist of his first appointment after his GP agrees that he needs to see a specialist (Choose and Book, 2009).

Currently, there are many different solutions for online appointment systems, depending on the purpose of the businesses, whether these be barber's shops or business consultations. The Genbook website is an example of a general Appointment Booking System (ABS) which can be a part of any existing website in order to manage the appointment. This system has very powerful features such as online booking, automatic appointment confirmation, flexible appointment scheduling, online scheduling pages, and the ability to synchronise to other calendars such as Google calendar (Genbook, 2010). Other examples of exiting solutions for scheduling appointment are Moodle Scheduler and Jiffle, which are online web services that dramatically accelerate the process of scheduling meetings. However, many of these existing solutions systems have some functions which are inapplicable with the online lecturer appointment system and some even lack important functions such as…

The goals of this investigation were to focus on producing a web application that enables a lecturer to plan scheduled consultation appointments for his students, in order to enable each student to book a mutually suitable appointment time. This application should incorporate the best practices from existing online appointment systems and also focus on how to eliminate a new login username and password for students. Then, this application will be examined and evaluated, and the results of this stage gathered, analysed and presented.

In an attempt to find an answer for this problem, this investigation will focus on how to implement OpenID technique, email integration and synchronise this with other calendars, such as Outlook and Google calendar.

1.2.Project objectives

The project aims to write an online lecturer appointment system for project students. The lecturer will choose times of availability and length of consultations in a calendar view, which will then become "bookable" by students. As an advanced objective, the resulting feed should then be capable of being viewed in popular calendaring systems such as Google calendar or Microsoft Outlook. This will involve writing a web application in aspx and evaluating the usability of the system.

1.2.1. Core objectives:

- Produce an online lecturer appointment system for project students. This system must have at least the following functions:
  1. Two roles: a lecturer as administrator and a student.
  2. An appointment manager: this creates a schedule for appointment and sets up the duration and location of the consultation.
  3. The appointment: this represents an appointment that has been made in the system.
  4. Availability checker: This shows the available appointments.
  5. Booking appointment.
  6. Appointment confirmation: this sends the confirmation to the user.
  7. User commitment evaluation: this shows the user's commitments of attending appointments.
- Study some existence online systems in terms of the appointment manager, booking appointment, appointment confirmation, availability checker and ability to synchronize to other calendars.
- Determine the positive points and the drawbacks of the usability and reliability of the system in terms of the above functions.

1.2.2. Advanced objectives:

- Avoid using new usernames and passwords in this system.
- The system can synchronize to Google Calendars or Microsoft Outlook Calendar.
- The system can demonstrate the location of the consultation room, for example, using Google Map.

1.3. Outline of Report

The specific contents and aims of each chapter of this report are as follows:

Chapter 2 will present an overview of existing solutions of Online Appointments booking systems, which are evaluated in terms of their system functions, usability, advantages and drawbacks.

Chapter 3 will demonstrate the use-cases of the Online Lecturer Appointment Booking System (OLABSPS).

Chapter 4 will discuss the available systems for trust based login, examples of which are Shibboleth, Athens, and OpenID. These systems are explained and evaluated against usability, and efficiency.

Chapter 5 will provide a detailed explanation of the implementation of OpenID in OLABSPS.

Chapter 6 will describe the design of OLABSPS.

Chapter 7, will provide a detailed presentation and analysis of the data structure of the system.

Chapter 8 will explain the experiments that have been conducted in order to evaluate the functionality of the system.

Chapter 9 will focus on the results of the experiments and discuss the data they have generated.

Chapter 10 will be a concluding section, containing an evaluation against the original objectives stated at the beginning of the project.

2. An Evaluation of Existing Solutions

Currently there are many online scheduling appointment systems available, which might be used to provide an alternative lecturer appointment system. In this chapter, some existing systems will be explained and then critically evaluated in terms of usability, the registration system, the announcement system and ability to synchronise with other calendars.

Two of the most popular systems, GenBook and Jiffenow, have been chosen for this evaluation.

2.1. GenBook

The Genbook website is an example of a general Appointment Booking System (ABS) which can be a part of any existing website in order to manage the appointment. This system has very powerful features such as online booking, automatic appointment confirmation, Flexible appointment scheduling, online scheduling page, Customer Reviews and, synchronise to other calendars such as Google's calendar (Genbook, 2010).

In order to understand how the GenBook works, a new account has been created in GenBook system and my GenBook website appointment system is (http://math-solutions.genbook.com).

This system has two roles administrator and customer, which are visible below:

The administrator creates a new service with its details as shown in Figure 1.

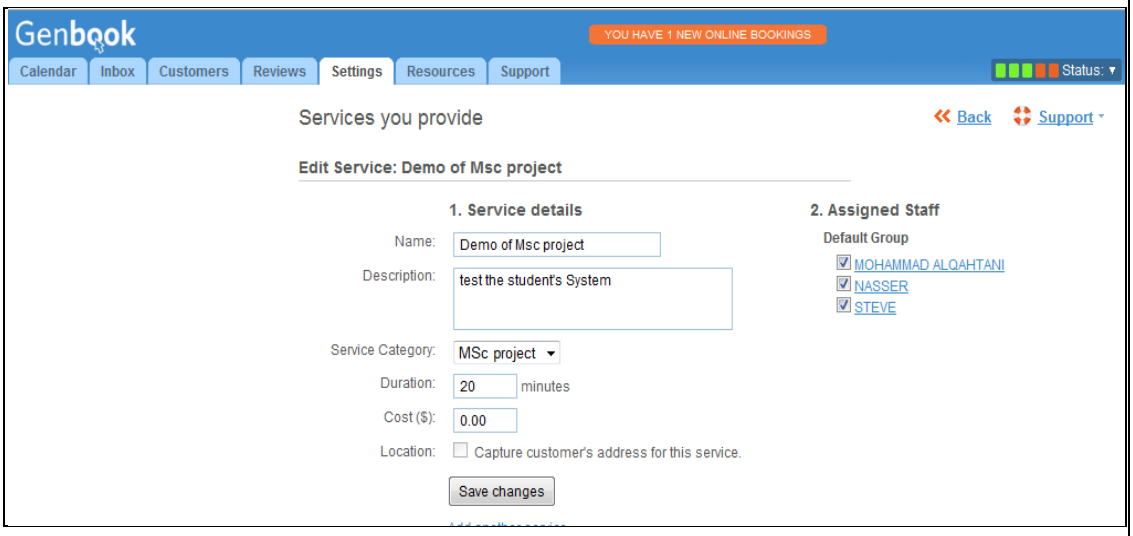


Figure 1: (Setting-Service)

After that, he assigns consultants to this service.

Then the client visits the website of Administrator and then must complete five stages to book an appointment. These stages are Service and Staff, Date and time, Your Details, Confirm and Finished.

For example, the client selects the service he wants such as “Demo Of MSc Project” .

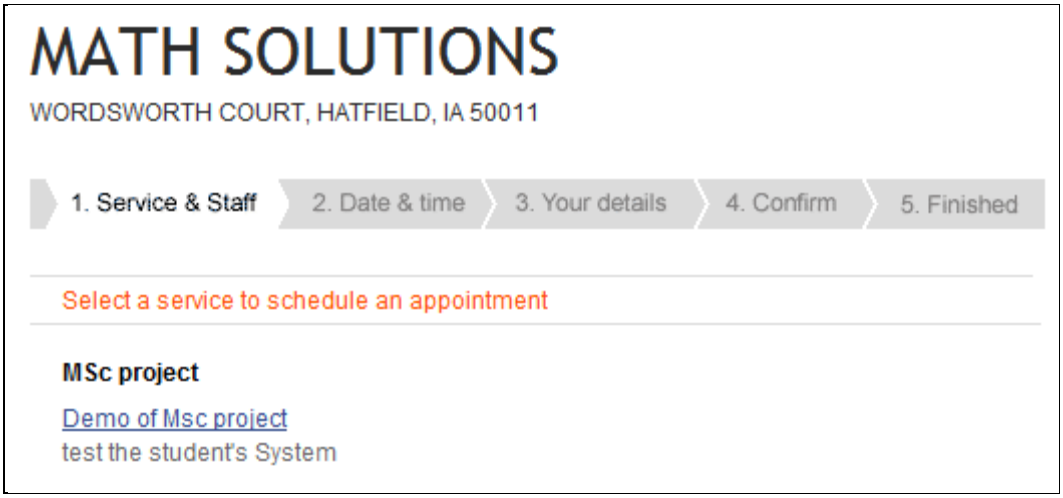


Figure 2: (Math Solutions)

The customer then selects one of the consultants, and chooses the date and time of appointment from the available options.

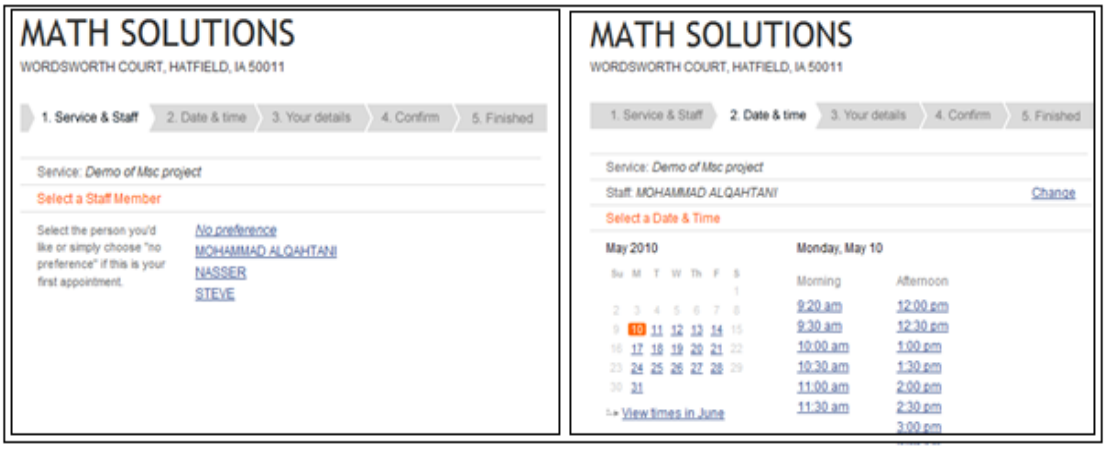


Figure 3: (Math Solutions)

In stage three, “Your details”, the customer must enter his details (first name, sir name, telephone and e-mail) before clicking on “Continue”.

Figure 4: (Math Solutions)

Finally, the client confirms the booking of this appointment and by the end of this stage; he has finished the booking procedures.

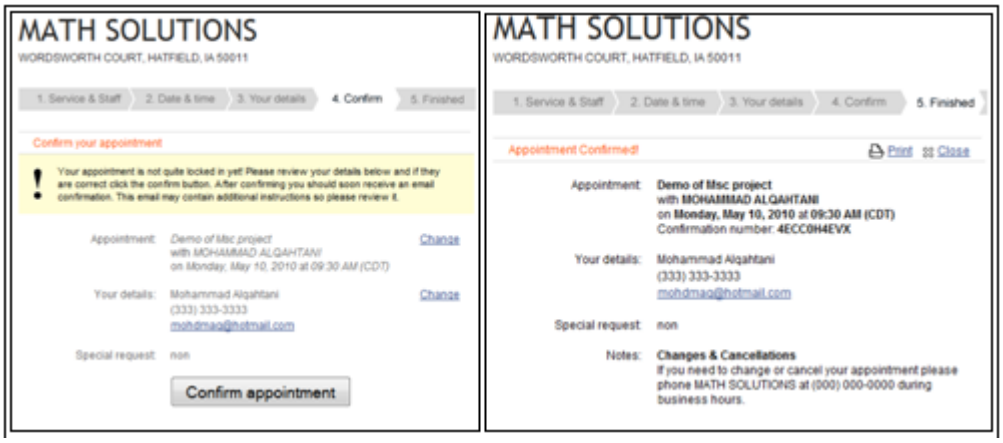


Figure 5: (Math Solutions)

After the booking procedure is completed, the system sends the client and the administrator the appointment's details and the calendar card by email.

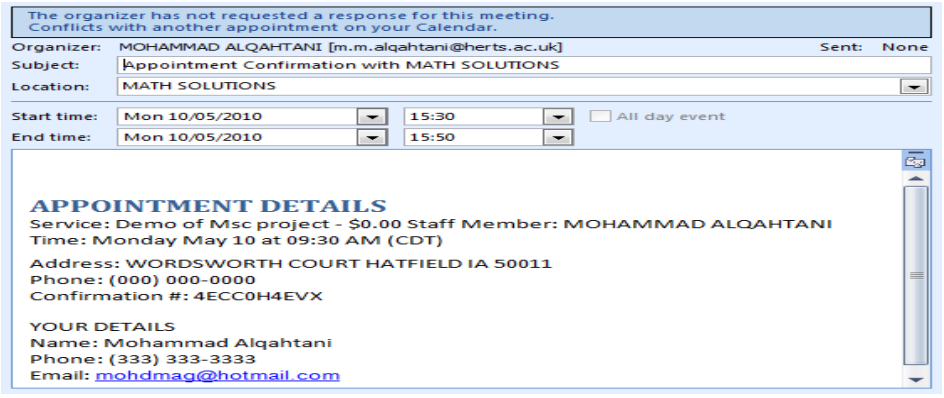


Figure 6

Additionally, the administrator can manage the booked appointments, as shown in figure7.

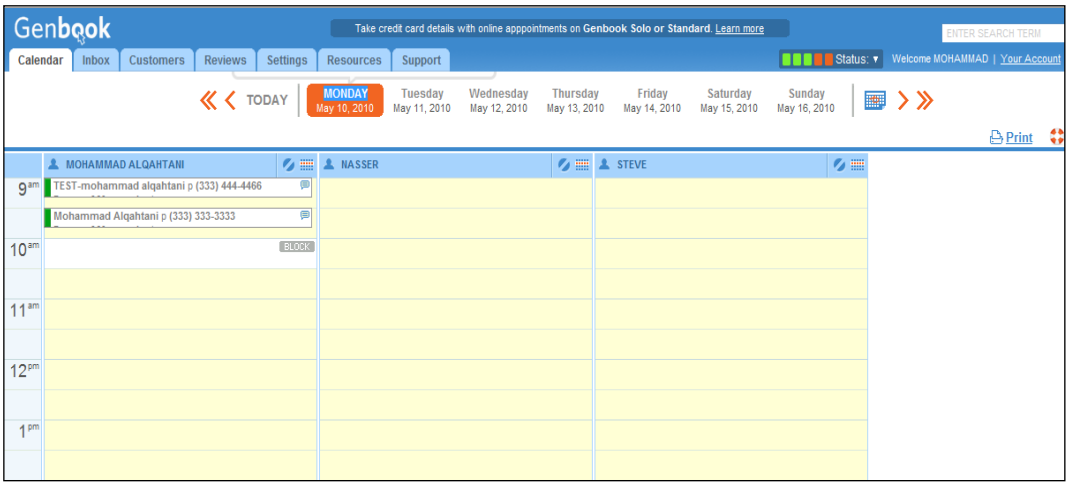


Figure7 :(Calendar)

The GenBook online appointment system has a number of very powerful features, namely:

- Easy to schedule the appointment for an administrator and also easy to book an appointment for client.
- Has a good correspondence system for both client and user.
- Good for use in small businesses, as an online booking appointment system.
- Synchronises with different calendars, such as Google calendar and Outlook.

However, this system also has some weaknesses:

- No start date and end date for appointments. In another words, the appointment runs every working day.
- Client has to enter his details with every booking he makes.
- The system does not check the authentication of the user.
- The client cannot cancel any of his appointments in this system,
- This online booking appointment system is open to anyone, not just a specific group of users can only make booking.
- No closed date for each appointment

2.2. Jifflenow

The second solution is **Jiffle,** which is an online service that speeds the process of scheduling appointments. Jiffle's technology allows a client to selectively share his availability calendar with his contacts and eliminate the time-consuming back-and-forth process (seven e-mails/voicemails on average) required to set up meetings with his colleagues and clients (Jiffle, 2008).

The JiffleNow online appointment system works as following:

A user creates an account in JiffleNow system and then creates a new group e.g. “MSc project”.

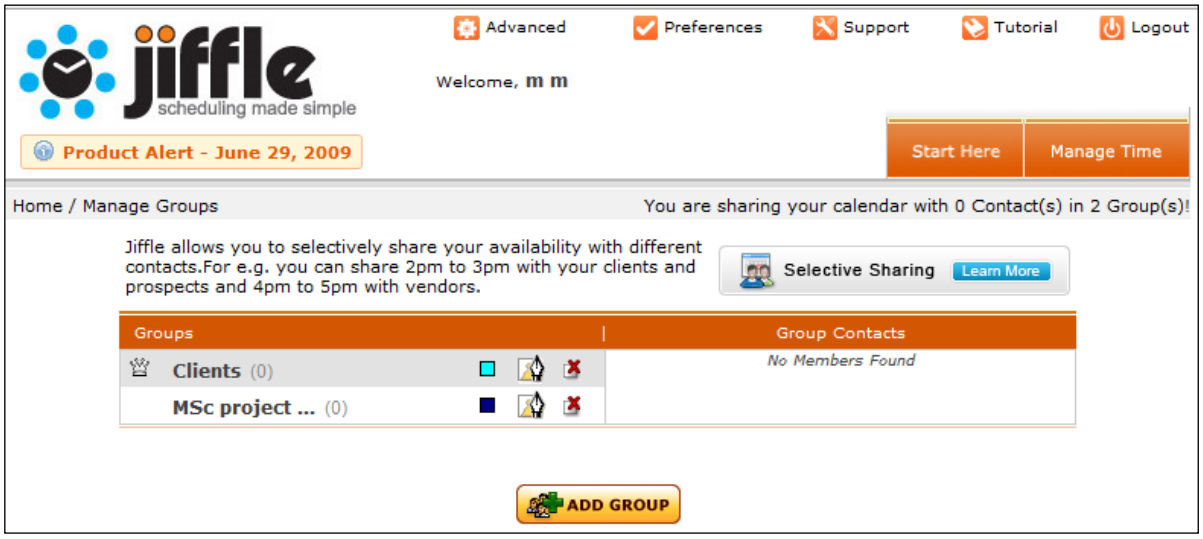


Figure 8 :(Manage Group, 2009)

After this, the user selects the “Manage Time” tab to create an available time meeting by selecting the available time from the calendar and specify the group that can book these appointments as shown in figure. For example, the user selects Friday, 2010-05-10 from 8:00 am to 1:30 p.m. for the MSc project group.

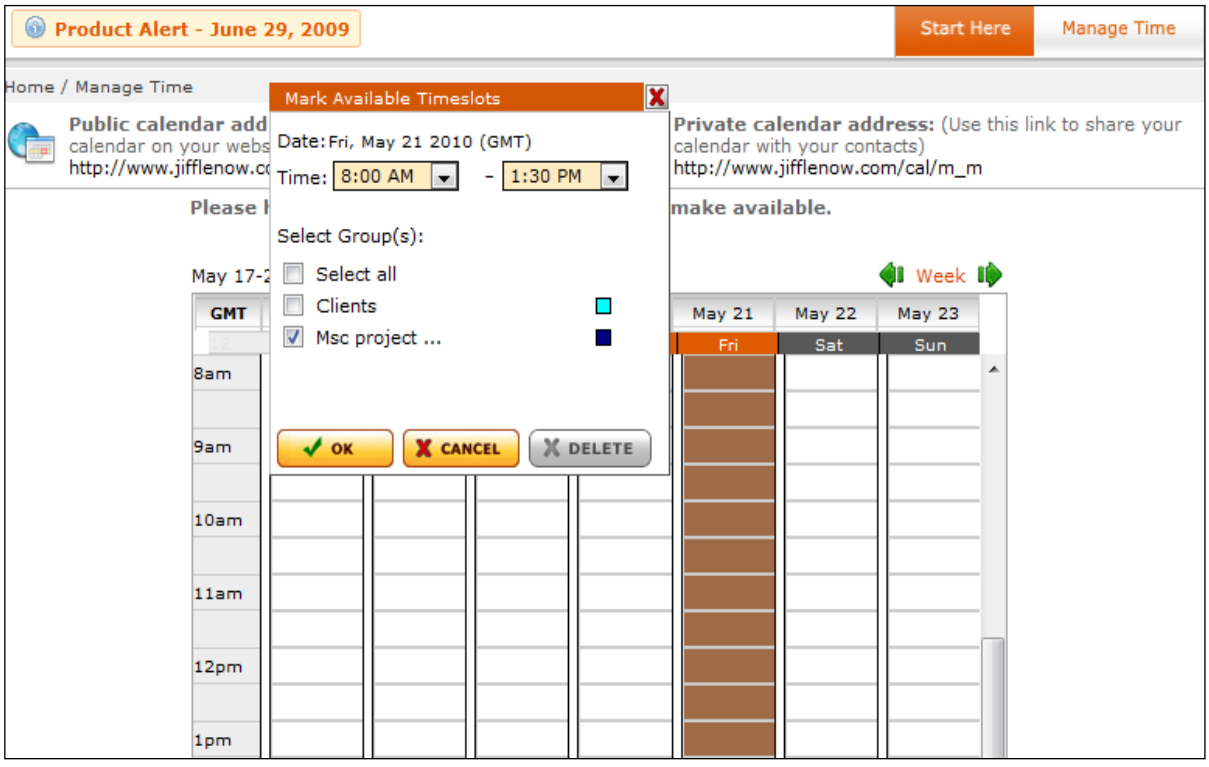


Figure 9 :(Manage Time, 2010)

Then, the user selects "Send Invites" from "Advance" in the main menu.

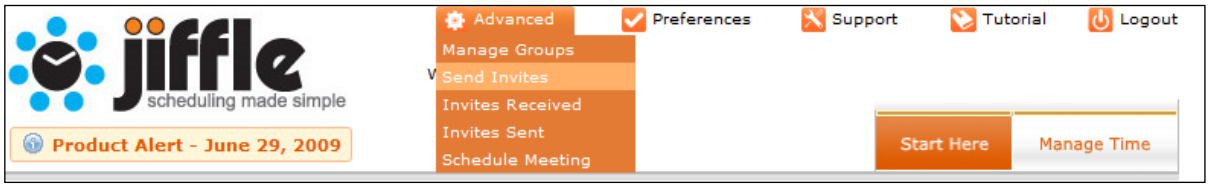


Figure 10 :(Send Invites, 2010)

Then he enters the details of his clients, group, start date and duration, before clicking on "SEND invites". For example, (first name: Mohammad, Last name: Alqahtani, Email: malqahtani@gmail.com, Group: Msc Project, Start Date: 20/05/2010, for one week).

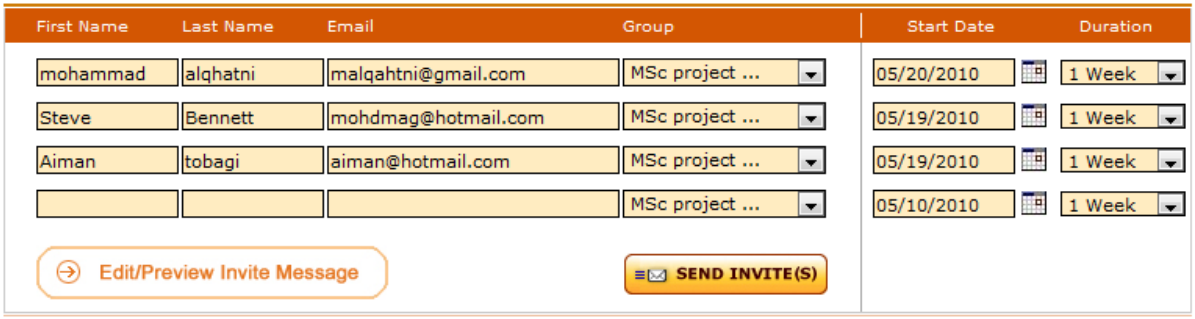


Figure 11 :(Send Invites, 2010)

Each client will receive invitation, including a link to the user's calendar. After the client opens the invitation and clicks on the calendar link, he sees on the user's calendar only the appointments that are assigned for him by the user. Then this client selects one of the available times from this calendar, completes it and sends the meeting confirmation as the following figure.

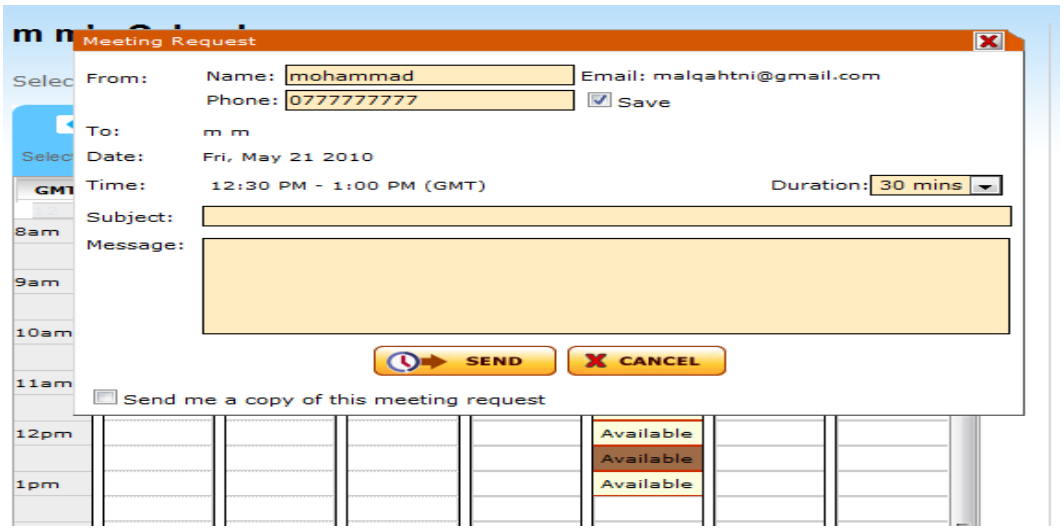


Figure 12 :(m m's Calendar, 2010)

This booked appointment will be pending until the owner of the calendar accepts this meeting as shown in figure13.

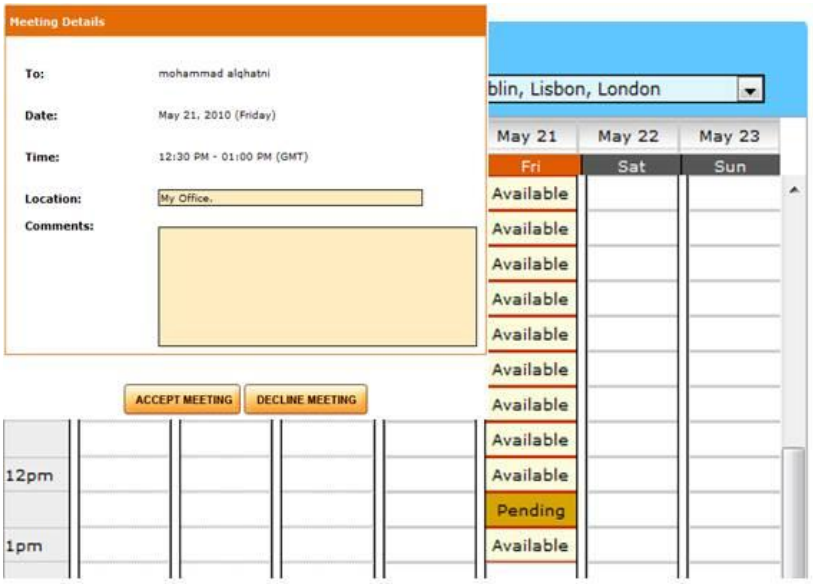


Figure 13 :(m m's Calendar, 2010)

The client will receive the confirmation of the meeting booking after the user accepts this meeting time slot.

The Jiffle online appointment system has a number of very powerful functions, which are:

- Easy to schedule the appointment for administrator and also easy to book appointment for client.
- Has good correspondence system for both client and user.
- Synchronise with different Calendars such as Google's calendar and Outlook.

However this system has some disadvantages which are:

- User has to enter his clients' details manually.
- User can invite only four clients each time.
- Client cannot cancel any of his appointments in this system.
- Long procedure to confirm the booking of meeting.
- User cannot give the subject for appointments, where his clients enter the subject of each meeting during booking processes.
- No deadline date for appointments.

3. Online Lecturer Appointment System for Project Students (OLASPS) use cases

In the online lecturer appointment booking system, there are three main roles: administrator, lecturer and student. Each of these roles has a distinct function within the system, which works as follows:

Administrator: set the activation code for lecturer and another activation code for students and then sends both lecturers and students the activation codes.

Lecturer: creates a new appointment window which has a start date, end date and the duration for each appointment. For example: (Demonstration of MSc Project between Monday 12 -May 2010 and Tuesday 13- May 2010, duration: 20 minutes for each appointment, the location: room E305 college Lane, the name of an examiner:: Steve and the close date of booking: Friday 9-May-2010).

System: creates the appointments for students to book and then emails the details of the appointment window to all students in the system.

Student: receives the email, opens the system, selects the appointment window, then books a preferred appointment time.

System: sends the booked appointment details and Calendar Card as an attachment to the student's e-mail.

The new features in this system are:

- Students do not need to register in the system to get new user name and password. This is handled by using OpenID to login to the system.
- Student can book an appropriate appointment time that is suitable for him.
- Student can save the booked appointment to his calendar easily by save the calendar card in his calendar.
- The use-cases diagram of the OLASPS as the following diagram, Figure 14

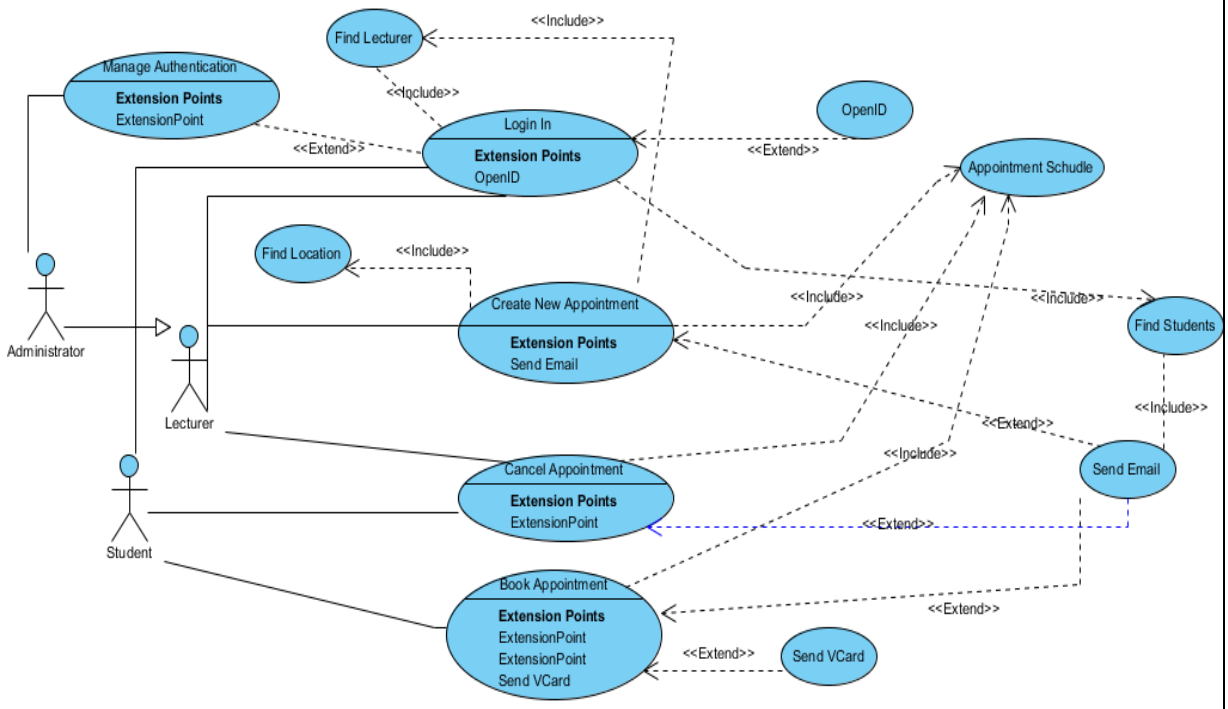


Figure 14: Use-case of (OLASPS)

4. Systems for trust based login (Shibboleth, Athens, openID)

There are some existent solutions to solve the problems of multiple login to different websites resources. The most popular techniques are Athens, Shibboleth, and OpenID.

4.1. Athens

Athens is a centralized Access Management System which is a single sign on protocol. This system has manipulated the problems of multiple login to different websites resources. This service is supplied by Eduserv since 1996 and then it has been become accustomed by JISC in 2000 (Kerry, 2009). Athens is used as a centralized Access Management system for the UK education institutes. In other words, this system has a database that includes the details of the institutions that use Athens.

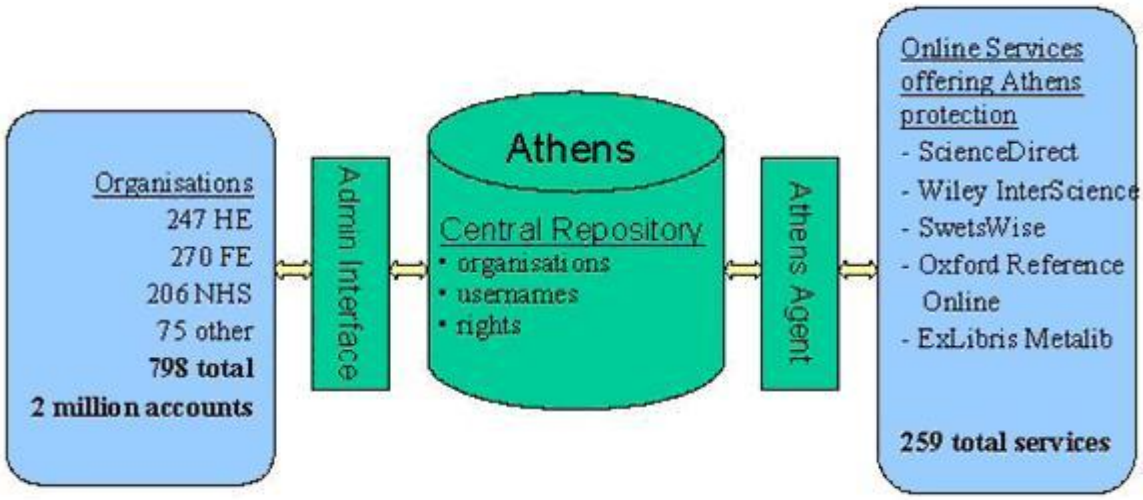
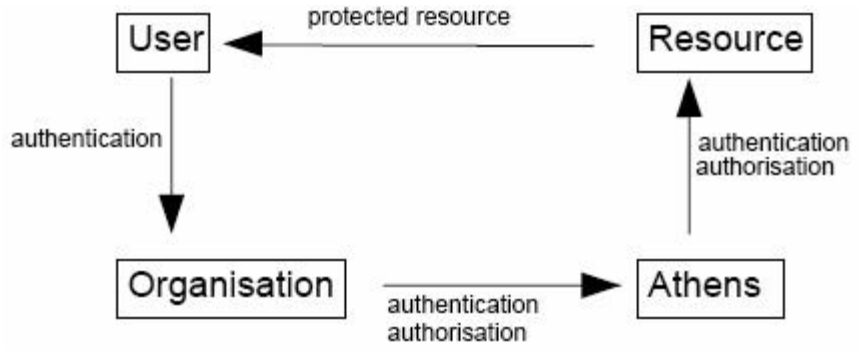


Figure15: Schematic of Athens Access Management, Source: Ubomoibhi, C., Masson, A., & Norris, L., (2001)

This is to allow a user from any institution to access any of other institutions' resources using single sign on. For example as described by Figure16, when a user visits the library homepage there is a link to the resources of another institution. By clicking on this link, the user is directed to a login. Once he provides his username and password, this is checked against the central repository and, upon successful authentication, the Athens agents interact with the online services validating the user (Layla, 2008).



Figuer16: AthensDA Data Flows (TrustDR . Avialable at: http://trustdr.ulster.ac.uk/work_in_progress/workpackages/WP2-1/theThreeAsWP2_1_29.php, 2006)

Despite Athens being a single sign-on protocol, it is relatively unused outside the UK and has difficulties in interoperability because Athens is not standards based.

4.2. Shibboleth

The second Single Sign On protocol is Shibboleth which is a centralised Access Management system. Shibboleth provides a mechanism in which a user can only log on once and then gains the right to use multiple forms of content protected in the different institutions who have agreed to work together, which is called a Shibboleth federation (David, John & Jackie, 2006).

Shibboleth exists in each server of all institutes using this technique. When a protected resource is requested, the shibboleth launches the "Where are You From" program which asks the user to logon and validate whether or not they can see the resource

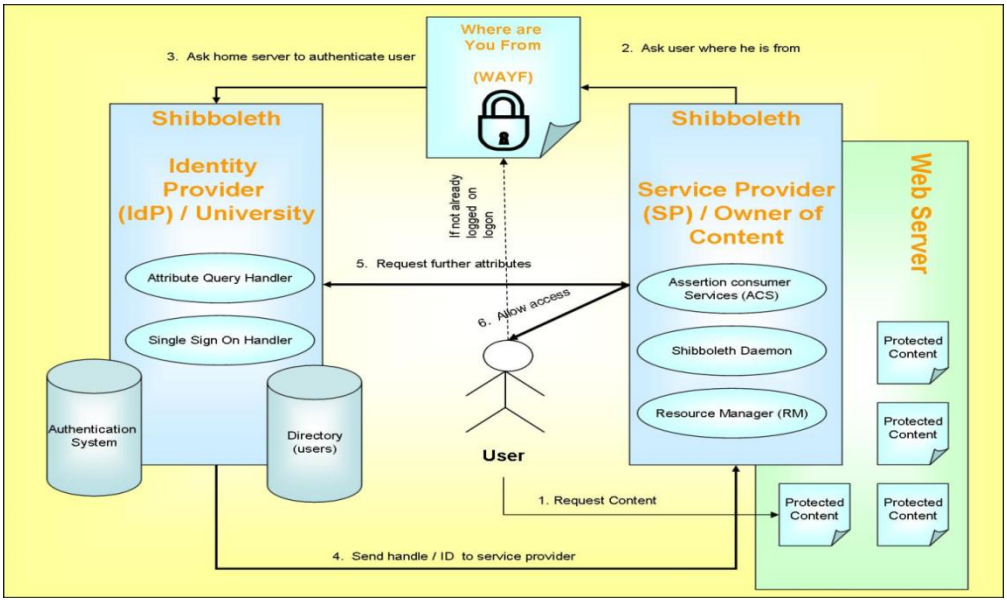


Figure17: An overview of the Shibboleth System (TrustDR . Avialable at: http://trustdr.ulster.ac.uk/work_in_progress/workpackages/WP2-1/theThreeAsWP2_1_29.php, 2006)

In spite of Shibboleth having the advantage of a single sign on protocol, it has many recognised disadvantages. Firstly, there is no global logout function in which a user has the ability to log out of all services simultaneously. The other drawback is complexity of implementing Shibboleth. Shibboleth relies on large numbers of assumptions to work, such as JavaScript being enabled on the web browser and institutions being able to support the administration requirements (Pollock and Cornford, 2000).

4.3. OpenID

Last a Single Sign-On protocol is OpenID. This solves the issue of having an individual user name and password for every web site supports OpenID. The OpenID works as a decentralized and an open framework for user-centric identity. There are some terms for the procedure of authentication in OpenID. First, the website server (WS) is where the website is hosted and the user tries to log in. Second, OpenID provider (OP) is an Identity Provider which authenticates the user instead of the WS. Last, the user is a person who wants to use a service of WS by authenticating with OpenID (Hyun-Kyung & Seung-Hun, 2008).

The OpenID authentication's process needs the following steps as shown in bellow:

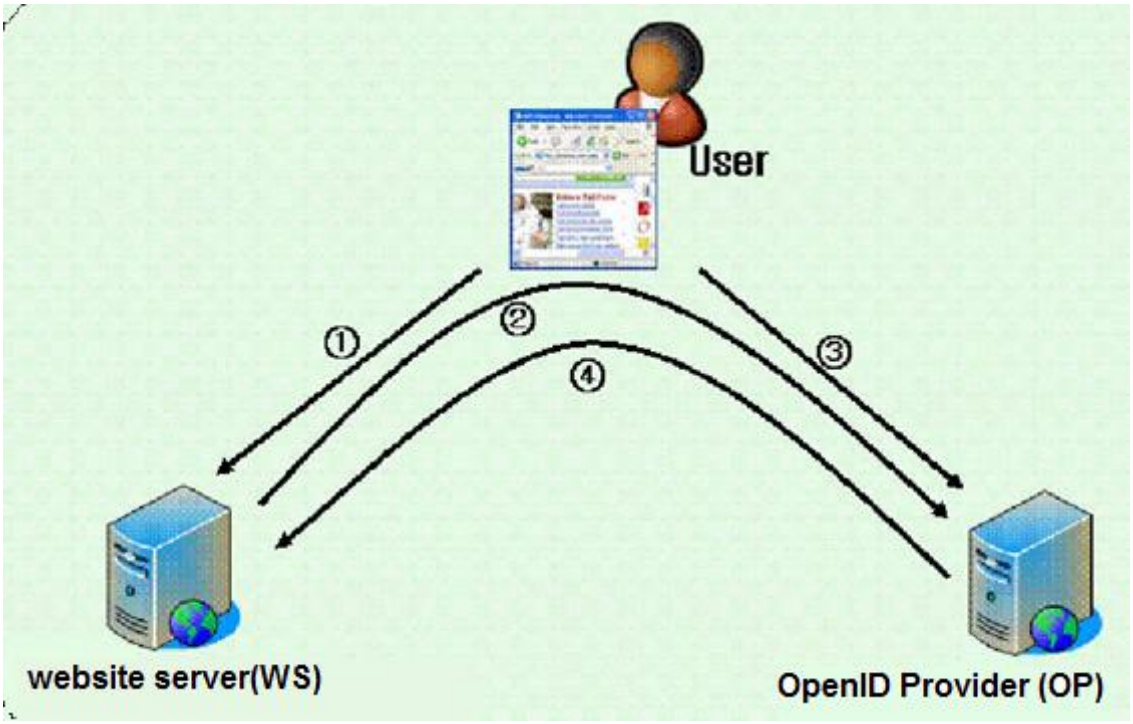


Figure18: The process of authentication in OpenID (Hyun-Kyung & Seung-Hun, 2008).

I. When a user tries to sign in to the WS, the WS asks the user to select his OpenID Provider.
II. The WS sends the session id to the user's OP.
III. The OP sets a session id to the user and the user enters his /her password into the field on the OP's page.

IV. If the account is verified, then the OP sets a secure session id to the user and sends authentication token to the WS. After that, the OP redirects the user to the WS. The WS shows the page for authenticated user via the browser.

Therefore, using OpenID has several advantages such as the user avoiding having a new user name and password. Secondly, OpenID is decentralised which means that the user does not have to trust a single authority such as Google or Microsoft. Another advantage is that websites can have stronger authentication by redirecting users to sign in with a strong credential. Finally, OpenID mechanism encourages many users to sign on websites that use OpenID because these users do not have to register.

5. Implementation of OpenID in Online Lecturer Appointment System

In order to solve the problem of having a new username and password of the new system, RPX technology was used. This technology is supplied by a corporation called Janrain, which allows users to sign into a website application by means of their existing Facebook, Twitter, Hotmail, Open ID or Google account. The advantage of this is that the user does not have to get a new user identification to login to this website **(**JanRain, 2009 **).** RPX is a proxy between a website and the OpenID provider, and is completely obvious to the end user as shown in figure19.

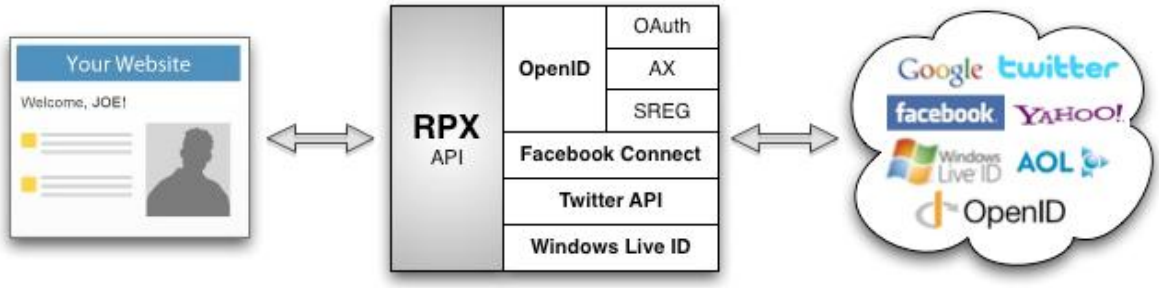


Figure19: The process of authentication in RPX (JanRain, 2009)

I used the following steps to implement the RPX in online lecturer appointment web application:

1. Register for an RPX developer account (https://rpxnow.com/get), and I was given secret API key and my RPX domain is ("https://online-appointment.rpxnow.com/") for my application. This data must be used in all communications with the RPX server. In this step I entered my domain name which is (mohammad-alqahtani.com) and the application name which is ("online-appointment").
2. Download the RPX class which is written in C# from the RPX website and then I converted the code to VB.NET. After that, the new class were included into the ASP.NET project (Example of RPX).
3. Create a Login.aspx page in ASP.NET project and add the following Hyperlink control provided by RPX service (JanRain, 2009).

```
<asp:HyperLink ID="lnkSignIn" runat="server" CssClass="rpxnow"
onclick="return false;" Text="Sign In" />
```

4. Add javascript in the top of <body> tag:

```
<script src=https://rpxnow.com/openid/v2/widget
type="text/javascript"></script>
```

```
<script type="text/javascript">
   RPXNOW.overlay = true;
```

```
    RPXNOW.language_preference = 'en';
</script>
```

5. In the page_load event of Login.aspx page, following code is added. This sets the Navigation URL of the Hyperlink control to point to the RPX authentication service:

```
Me.lnkSignIn.NavigateUrl = "https://online-
appointment.rpxnow.com/openid/v2/signin?token_url=http://" & _
Request.Url.Authority & "/RpxResponse.aspx"
```

RpxResponse.aspx page is added because the link in the code above informs the RPX service to redirect the user back to this page (*Neil, 2009*).

6. In RpxResponse.aspx.vb , the following methods are added:

6.1. `loginUser(ByVal authInfo As XmlElement)`. The porpuse of this function is to get the user's unique identifier and details from the returned XML authintication file of user . this file is returned by identifier provider that is used by the user at login time. the parameter of this function (authInfo) is the user's information including (unique identifier, names, email, provider).

6.2. `checkUser(ByVal identifier As String)`. This is to check if the user is already exist in the user table in the system's database or not. If the user is exist, then allow user to sign into the website. Otherwise, the funtion ask user to enter the activation code.

6.3. `addnewusers(ByVal id As String, ByVal disName As String, ByVal em As String, ByVal fname As String, ByVal sname As String, ByVal ftname As String)`. After the user enter the correct activation code, the `addnewusers()` adds this user to `user` table in the database or update the user's data if he is exist.

To sum up the above steps, the online lecturer appointment system login is shown as the diagram below:

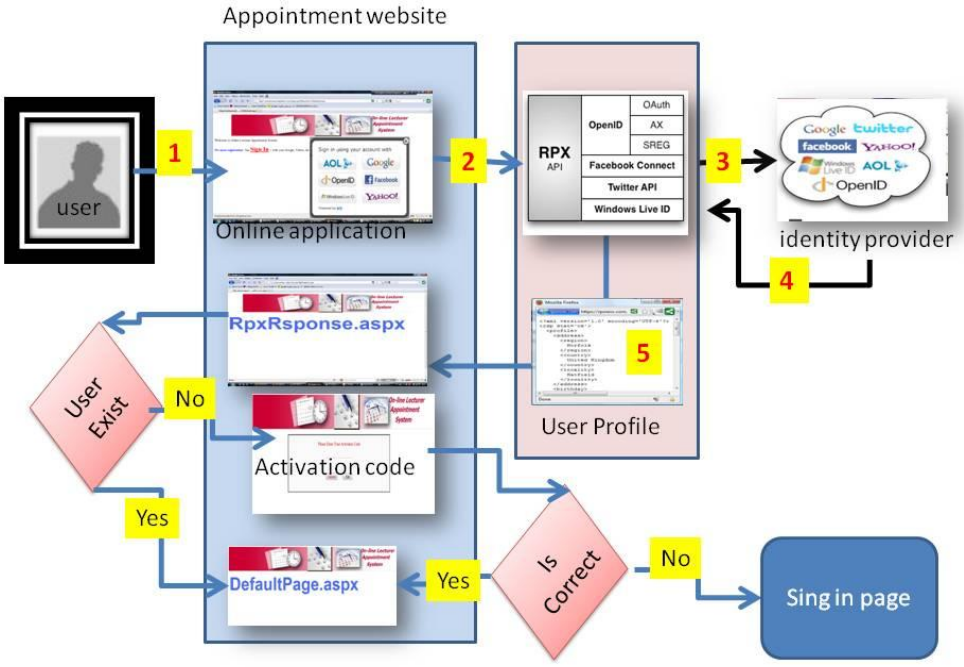


Figure20: online lecturer appointment system Authentication

1. A user signs up to the website by selecting any Identity Providers such as (Google, Yahoo, Windows Live, FaceBook, or OpenID).
2. RPX redirects the user to the Identity provider to login using his username and password.
3. The Identity Provider verified the user .
4. The Identity Provider returned the user's profile to RPX.
5. RPX send XML file which contained the user's details to the website system
6. System checked if the user exists in the system's database and if the user exists then he/she logged in. Otherwise, the system will ask the user to enter his/she activation code.

6. Online Lecturer appointment booking system Design (OLABS)

6.1. Design tools

OLABS is designed and implemented using the following software and hardware:

- Microsoft Visual Studio 2008 for writing the project system in aspx, VB.NET, and Java scripts
- SQL Manager Lite for SQL Server for design and Manage the database of the system
- Net-Beans 6.5 for designing the Use-cases of the system
- SQL Server as database provider
- Windows Server 2008 for hosting the OLABS

6.2. Publish the Online Lecturer appointment booking system (OLABS)

After the OLABS is implemented completely, OLABS is published under this website (http://mohammad-alqahtani.com). The aim of this publication is to enable the examiners and tester to evaluate and test the functionality of the system, as well as to run many experiments.

6.3. Class diagram

Based on the use-cases diagram of the design of OLABS there are five main functions that are provided by this system: Login function, Manage Authentication function, Create New Appointment, Book Appointment, Cancel Appointment

Firstly, the login function is used by two roles: lecturer and student. This function consists of five classes as shown in the figure21. The responsibility of the login function is to verify the identity of the user when he/she wants to use the website system.

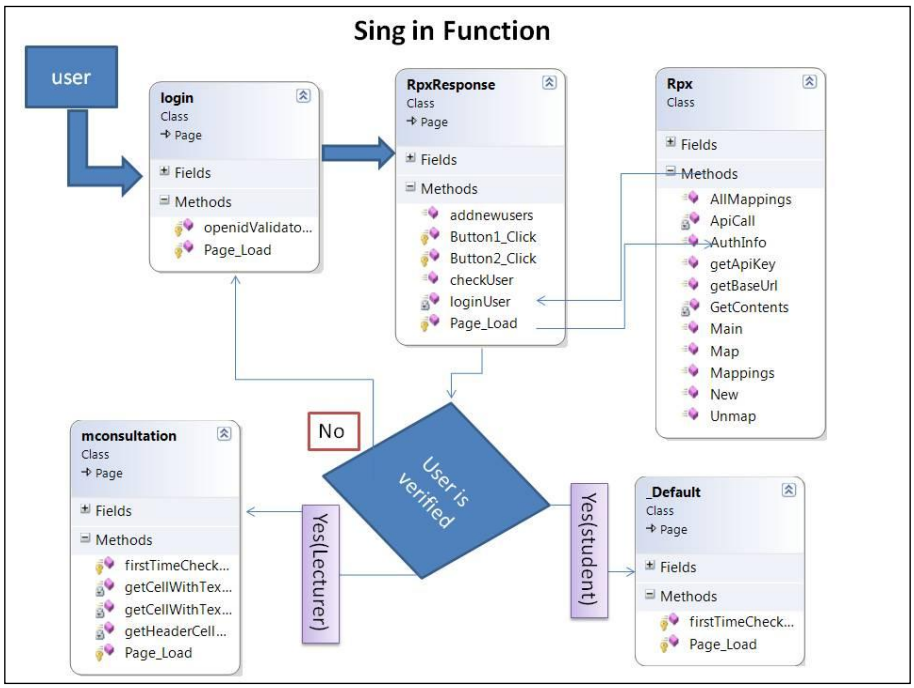


Figure21

The second function is Manage Authentication which is only used by the administrator role. This function is used for setting the activation code, adding new user, or cancelling an Appointment Window.

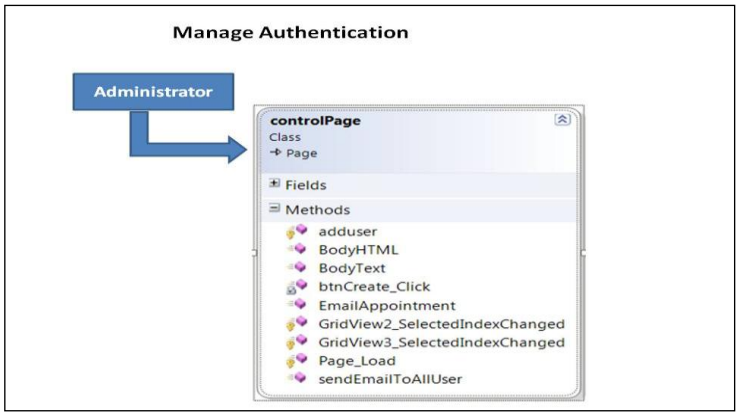


Figure21

Another function available is Create New Appointment which is used by both lecturer and administrator roles. This function performs the creation of a new appointment window and its appointment times, then sends email to all students in the system.

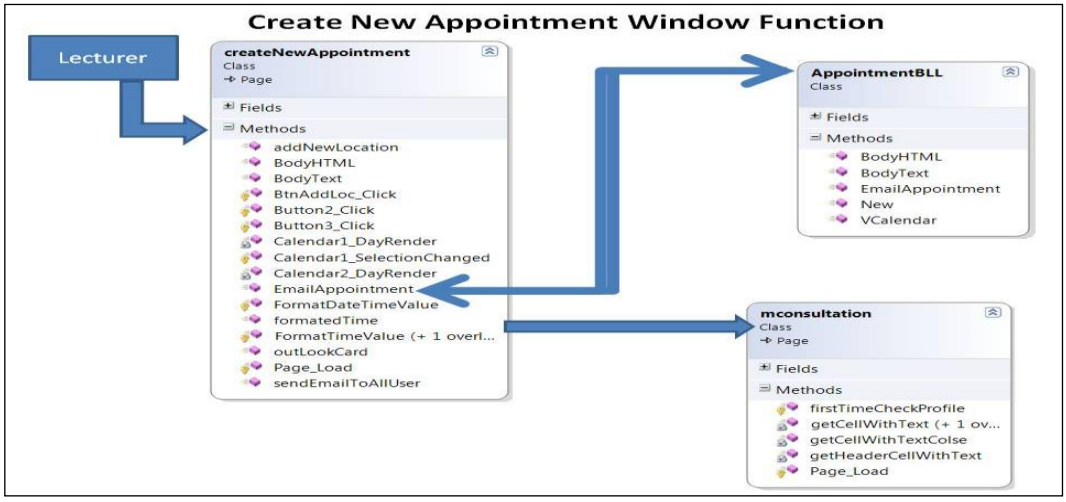


Figure23

The next function is Book Appointment, which is used by the student role. This function allows students to book appointment times, receive email containing the appointment details, receive calendar cards, and sees his booked appointments window.

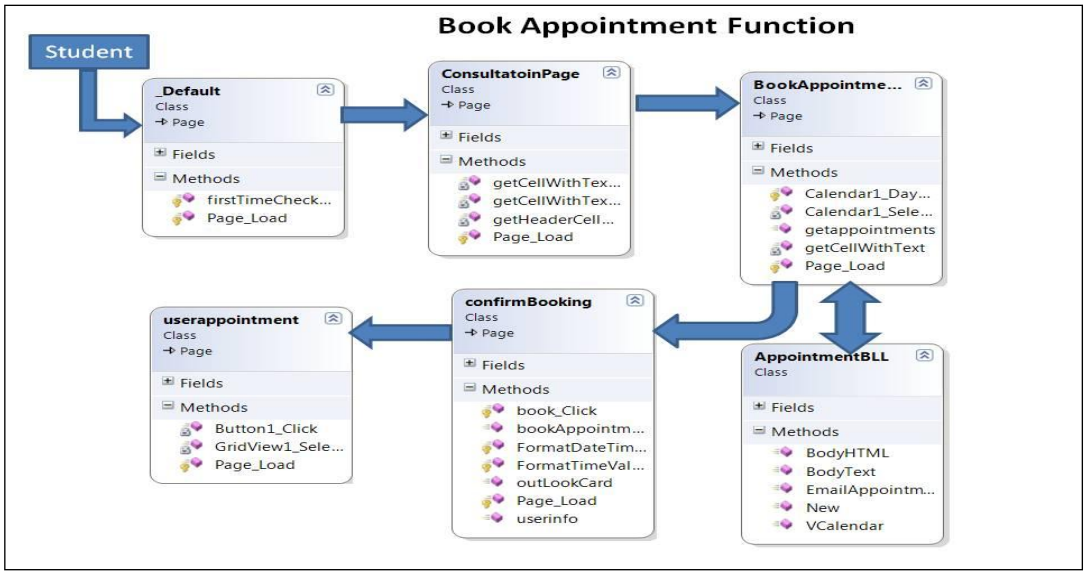


Figure24

The next function is the Cancel Appointment window which is used by the lecturer role and administrator. This function allows these roles to cancel any Appointment Window and send cancellation notifications to all students enrolled in this system.

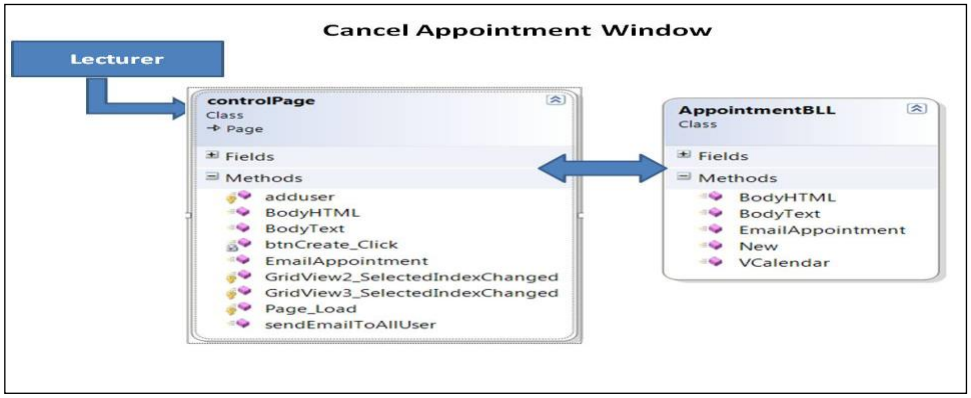


Figure25

The last available function is the Maintain Appointment windows which are used by both administrator and lecturer roles. The function of this option is to provide these roles the status of each appointment window, number of booked and booked time. Additionally, this function allows the lecturer role to cancel any student's booked time and then send a cancellation message via email to these students.

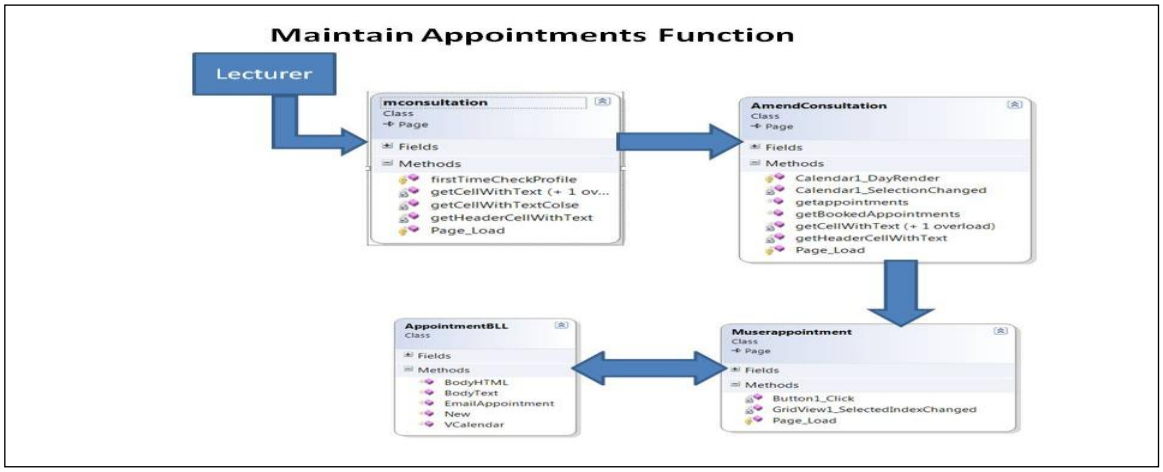


Figure26

7. System Data Structure

The database for the OLABS contains five main tables and one junction table as shown in figure27.
The main tables are consultation, appointment, user, location and activation code, while the junction table is UserAppointment.

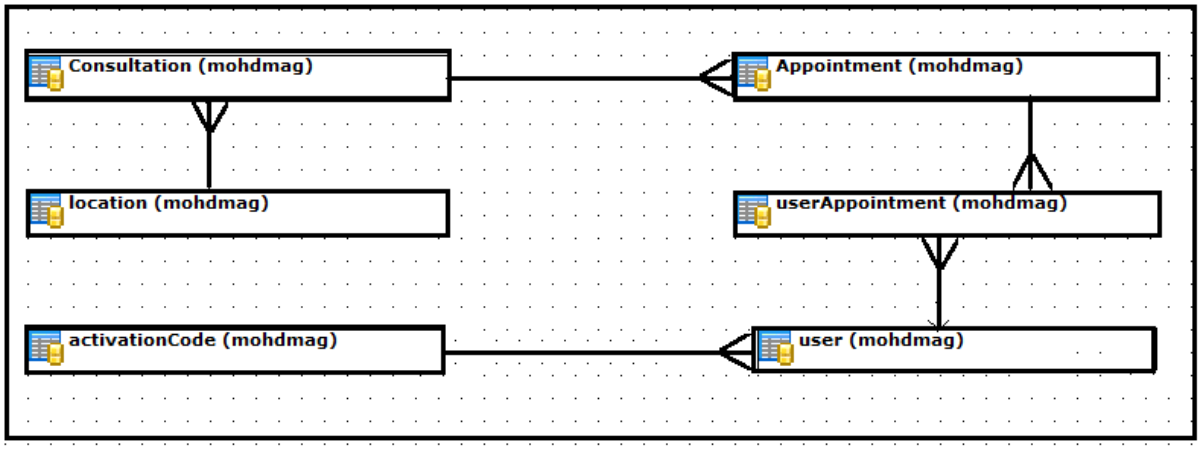


Figure27: Entity Diagram

The Consultation table stores the appointment windows which are created by the lecturer. Each consultation can have many appointments. Both Consultation and Appointment tables are used by the `CreateNewAppointment Class` to store the details of new appointment window and its appointments' times as shown in Figure 27.

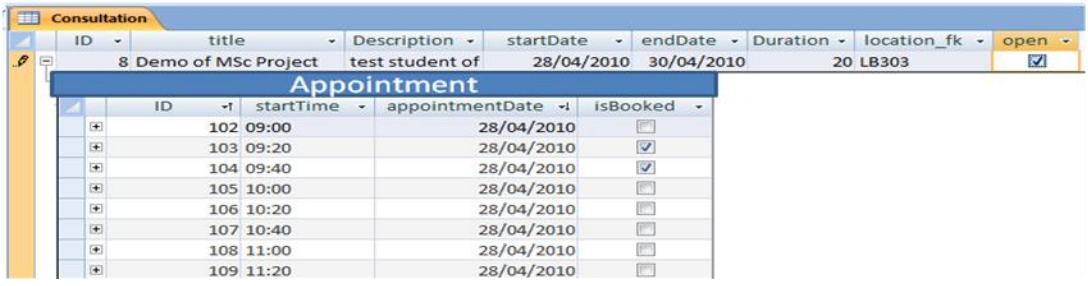


Figure28: Relation between Consultation and Appointment table

The location table is used to store the location of each consultation (appointment window). Thus, each location is used by at least one or more consultation(s). For example, both consultation1's appointments and consultation2's appointments take place in location 1. Additionally, Location table are used by `CreateNewAppointment Class.`

location

ID	Room	Building	Street	City	County	PostCode
1	LB303	Right	college Lane	hatfield	Hertfordshire	AL10 0EQ

Consultation

ID	title	startDate	endDate
8	Demo of MSc Project	28/04/2010	30/04/2010
9	MSC project meeting 4	03/05/2010	05/05/2010
10	Final Meeting of MSc Project	10/05/2010	12/05/2010

Figure29: Relation between Location and Consultation table

The user table is used to store the user's details and each user related to specific group either student group or lecturer group. Thus each activationCode has one or many user(s) and user must relate to one activationCode only. User table is used by the RPXResponse Class.

The database relational diagram is demonstrated by the bellow figure.

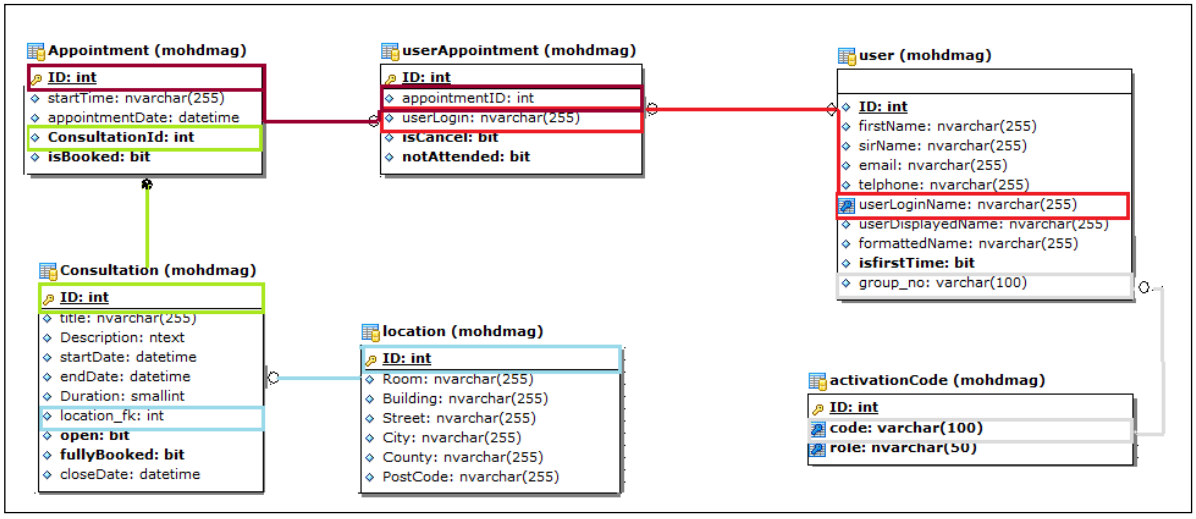


Figure30: rational schema of OLABS's Databse

8. Using the Online Lecturer appointment booking system (OLABS)

This chapter demonstrates how the OLABS is used by the three roles existing in this system.

An administrator sets up the activation code for students' accounts and another activation code for the lecturers' accounts. After that, the administrator sends the activation codes to students and lecturers by email.

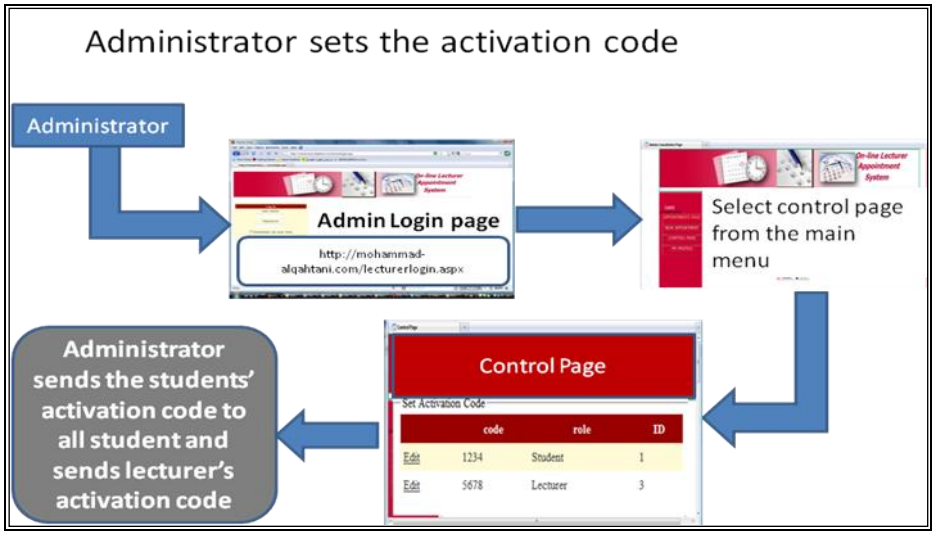


Figure31

After the users receive the activation code via email, the user signs into the system using his OpenID. The system will then asks for an activation code. After that, this user enters the right activation code. After this step, the system will activate this user's account and allow him to login to the system.

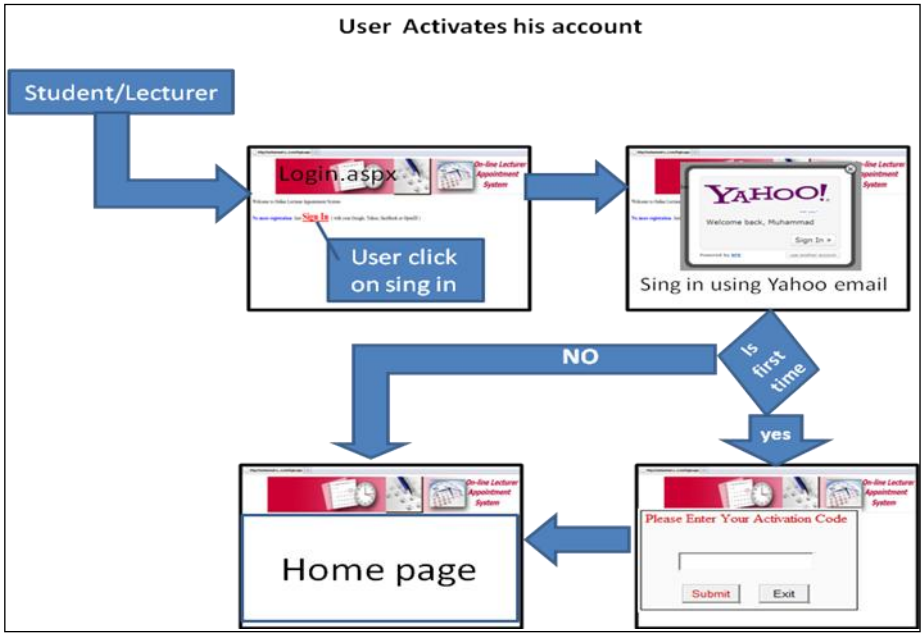


Figure32

After a lecturer signs into the system, he clicks on "new Appointment window" link to create new appointment window. In this page, he enters the appointment's title, details, duration, and location, start date, end date, close date and then he selected the start time of date and end time. For example, appointment Title: Demo of MSc Project , between 10/5/2010 and 11/5/2010, location: college Lane room F-305, each appointment lenght:20 minutes, start time of the day: 9:00 to 12:00). After that, the lecturer clicked on "Save" Button. As a result, the system sent e-mail with the details of this new appointment window to all students.

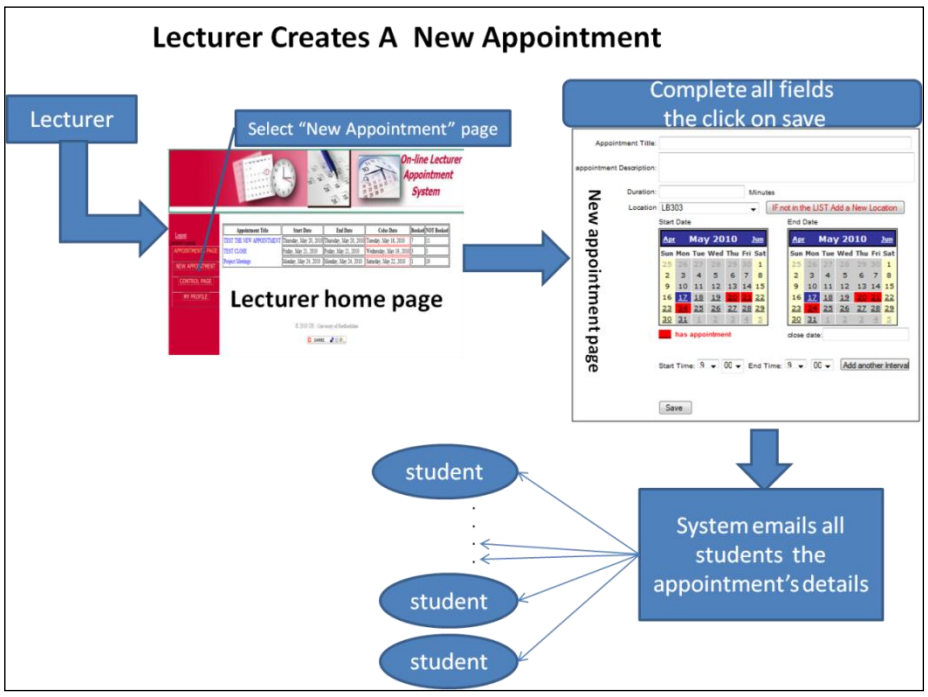


Figure33

After the Lecturer has created a new appointment and all students received thedetails for this, each student logs in to the system and goes to “book appointment” page. From this page the student clicks on the proper appointment window link. Next, a list of available appointments times appears and the student clicks on the suitable appointment time from the list of appointments. After that, he sees the full appointment’s details and then he clicks on the Book button to complete the booking process. Consequently, the system emails him the details of this appointment and the calendar card.

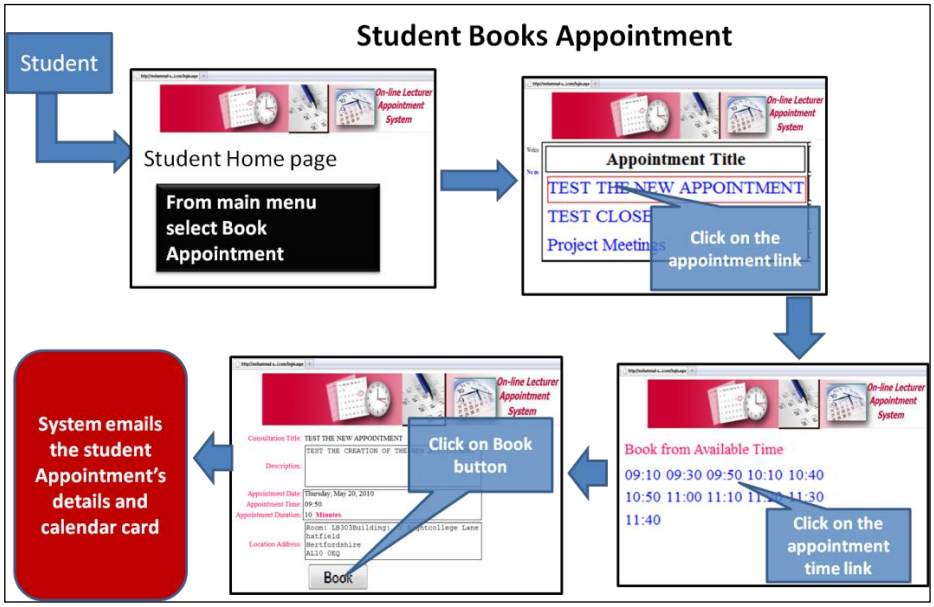


Figure34

A student goes to their appointment page. After that, from the appointment list, he selects an appointment that should be cancelled and then presses the “cancel selected appointment button”. As a result, this cancelled appointment appears in the list of “cancelled appointment”.

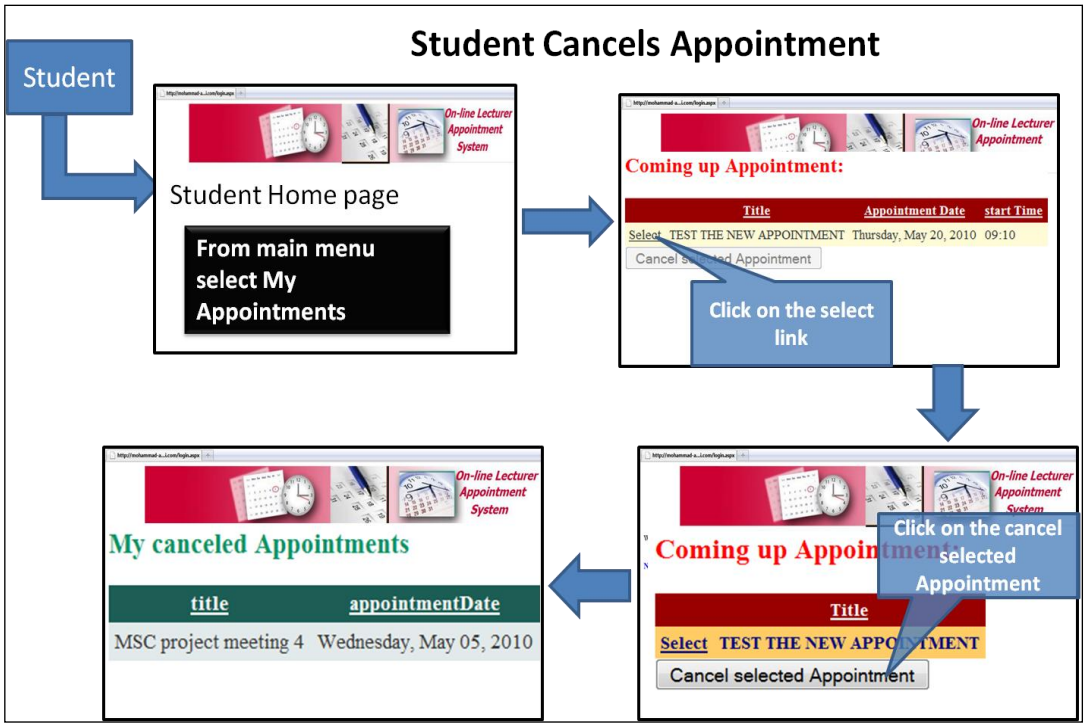


*Figure*35

9. Experimental Test

The experiment was implemented at the University of Hertfordshire on 10th of May 2010 for one week. In this experiment, four MSc computer science students were involved. One of the contributing students played the system administrator role, another one was the lecturer and the rest played the student role.

The aim of this experiment was to test if the produced web application system meets research requirements. Additionally, this experiment tested the functionality of the system and examine users.

In order to gather and document the data, the experiment was recorded and documented using:

- Microsoft Expression Encoder 3 to record voice and the web application's screens that were visited by the examiners.
- Google's analytic: a java scripts are added to all the web application's WebPages to record the behaviour of the website by Google's analytic in terms of visitor's activities and the traffic on the website.

The involved students were asked to complete the scenario of this experiment. The aim of this experiment's scenario is to meet the following goals:

- The authorisation roles which are administrator, lecturer and student.
- Sign in using OpenID.
- Creation of a new appointment.
- The eEmail integration.
- Booking appointment process
- Cancellation process

In this scenario, there were six tasks completed by the five testers as the following:

Task 1: test the authorization in the system

An administrator sets up the activation code for students' accounts and another activation code for the lecturers' accounts. After that, the administrator sends the activation codes to the students and lecturers by email. The aim of this step is to enable the students and lecturer to log into the system and activate their accounts using these activation codes.

Task 2: test the sign in system using openID and then activating the accounts

In the first part of task 2, a user who is in the role of student or lecturer signs into the system using his/her OpenID and then the system asks him to enter his activation code. After that, this user enters an incorrect activation code. The system will not allow this user to activate his account.

In the second part of this task, the user signs into the system again using his OpenID and then the system will asks him to enter the activation code. After that, this user enters the right activation code. The system will activate this user's account and allow him to login to the system.

The last part of this task, the user logs out the system and then logs in again using his OpenID to the system. As a consequence, the system must not ask the user to enter the activation code for a second time.

Task 3: create a new appointment window which includes the appointment's slots time.

After a lecturer signs into the system, he clicks on "new Appointment window" link to create new appointment window. In this page, he entered the appointment's title, details, duration, and location, start date, end date, close date and then he selected the start time of date and end time. For example, appointment Title:Demo of MSc Project , between 10/5/2010 and 11/5/2010, location: college Lane room F-305, each appointment lenght:20 minutes, start time of the day: 9:00 to 12:00). After that, the lecturer clicked on "Save" Button. As a result, the system sent e-mail with the details of this new appointment window to all students.

Task 4: book appointments.

After the completion of task 3, each student logs in to the system and goes to the "book appointment" page. From this page the student clicks on the proper appointment window link. In the next step, a list of available appointments times appears and the student clicked on the suitable appointment time from the list of appointments. After that, he sees the full appointment details and clicks on the "book" button to complete the booking process. Consequently, the system emailed him the details of this appointment and the calendar card.

Task 5: cancel appointments.

In this task a student goes to my appointment page. After that, when the appointment comes up, he selects the appointment that should be cancelled and presses the "cancel selected appointment" button. As a result, this cancelled appointment appeared in the list of "cancelled appointment".

Task 6: lecturer cancel appointment window.

A lecturer goes to "Control Page" and then cancels one of the appointments from the list of the unclosed appointment window. As a consequence, the system emails all students to tell them that this appointment has been cancelled.

Task 7: test the all system's functions

In this tasks all examiner visited all pages in this web application system in order to evaluate in general the usability and reliability of the system's functions. For example, fields validations, date validation, the system's failures and faults.

After the experiment was completed by the involved students, all videos and voice recordings were collected and gathered. After that, the information from the documentations of this experiment was collected and then reported in percentages, charts and figures. Additionally, the results from Google's analysis were collected in the form of chart and figures.

10.Results

In this chapter, the results from the Google analysis and experiment will be presented. The data is gathered form Google's Analytic from first of May to 15th of May 2010

Browser	Visits	Visits
Firefox	20	48.78%
Internet Explorer	17	41.46%
Chrome	4	9.76%

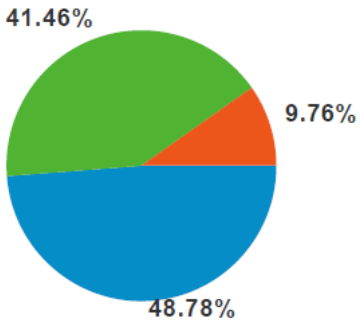


Figure36: Browser used by users

Figure36 shows the number of visits using different browsers. Only 10% of visitors used the chrome browser while 41.46% of visits is browsed by Internet Explorer. Interestingly, nearly half of visits are browsed by Firefox. In other words, this website is successfully browsed by different browsers.

Length of Visit May 1, 2010 - May 16, 2010

Duration of visit	Visits with this duration	Percentage of all visits
0-10 seconds	**13.00**	31.71%
31-60 seconds	**4.00**	9.76%
61-180 seconds	**2.00**	4.88%
181-600 seconds	**11.00**	26.83%
601-1,800 seconds	**11.00**	26.83%

Figure37: Length of visits

Figure37 illustrates that the length of visits as the number of visits per specific intervals of time. As can be seen, the majority of visits are in intervals 0-10 seconds, intervals

181-600 seconds and 601-1800 seconds. However, the minimum number of visits was in interval 51-180 seconds. In terms of functionality, 13 visits failed to sign into the website while around 24 visits succeed in singing into this website.

Figure38: Site Usage

This figure38 shows the site usage in general overview in terms of visits and page views. The average time spent on the site is nearly seven minutes per visit and the average number of pages visited is around eight pages per time.

task-2	part 1	signs in the system using wrong activation code first attempt	
Name	Role	login Identity Provider	does Login with wrong Activation code
Farahan	Lecturer	Windows Live	No
Sareh	Student	FaceBook	No
Aiman	Student	Google	No
Mohammad	Student	Yahoo	No

Tble1: Signs in the system using wrong activation code first attempt.

Table1 shows the number of student who signed into the system using the wrong activation code and the number who did not. Surprisingly, all students failed to log into the system by entering the wrong activation code at their first attempt. This indicates that the website succeed to prevent any failed authentication.

Task-2: Log into the system first time and second time					
Name	role	IDP	login 1st time to system	length of login 1st time(sec)	second time (sec)
Farahan	lecturer	Windows Live	Yes	37	23
Sareh	student	FaceBook	Yes	43	17
Aiman	student	Google	Yes	15	5
Mohammad	student	Yahoo	Yes	18	14
			average time in second	28.25	14.75

Tble2: Log into the system first time and second time

Table2 indicates the time needed to sign into the system by all students at the first-time attempt which is before use activated his/her account and at second time after activation the accounts. Interestingly at first attempt, the longest time was needed to sign into the system was 43 seconds while the shortest time was 18 seconds. The average time at first attempt was 28 seconds. However, at second attempt, the longest time was needed to sign into the system was 23 seconds while the shortest time was 5 seconds. The average sign in time at second attempt was 14 seconds. In other words, the average time for signing into the system after the activation of the user account is around 14 seconds. Finally, all users login to the system by their Google, Yahoo, FaceBook and Windows Live successfully.

Task-3: Lecturer Creates a New Appointment				
name	role	login Identity Provider	create appointment time	recive email after creation
Farahan	lecturer	Windows Live	84 sec	Yes
Sareh	student	FaceBook	N/A	Yes
Aiman	student	Google	N/A	Yes
Mohammad	student	Yahoo	N/A	Yes

Tble3: Time per Creation of a New Appointment

Table3 shows the time was needed to create a new appointment by a lecturer and the number of students who received e-mail and who did not. As can be seen, lecturer

created a new appointment successfully in 84 second. Additionally, all students received the appointment's details via their e-mails.

Task-4: Student Books Appointment							
Name	role	login Identity Provider	can book	length time of booking	receive e-mail	Receive V-card	can open email
Sareh	student	FaceBook	Yes	64	yes	yes	no
Aiman	student	Google	Yes	78	yes	yes	yes
Mohammad	student	Yahoo	Yes	82	yes	yes	yes
			Average	74.67			

Tble4: Time per booking Appointment

Table4 shows the time was needed to book an appointment by student and the number of students who received e-mail and who did not. As can be seen, all students booked appointment successfully. The maximum time was needed to complete the appointment booking was 82 seconds while the minimum was 64 seconds. Moreover, the average time was needed to finish booking was 74.6 seconds. All student received e-mails with calendar cards as attachments and all of these emails were accessible except the email that was sent to a student who used hotmail as an email provider.

task-5: Students cancelled their appointments				
Name	role	login Identity Provider	cancel appointment time	is cancelled
Sareh	student	FaceBook	44	Yes
Aiman	student	Google	47	Yes
Mohammad	student	Yahoo	43	Yes
		Avarage	44.67 seconds	

Tble5: Time per Cancellation of an Appointment

Table5 describes the time was needed by a student to cancel an appointment and this appointment where or not cancelled. As can be seen, all students cancelled their appointments successfully. After the cancellation, all the cancelled appointments were

bookable again. Another result from this table, the maximum time to cancel an appointment was 47 seconds while the minimum time was 43 seconds. Finally, the average time to cancel an appointment was 44.6 seconds.

11.Conclusion and Evaluation

11.1 Results Evaluation

The main objectives of this project were to produce an online lecturer appointment system for MSc project students in terms of usability, efficiency and reducing the time spending to schedule the appointments by the MSc programme tutor.

In this section, the results were achieved from Google Analytic and the experiment tests evaluated against the main project's objectives.

11.1.1. Evaluation of Google Analytic's results

As can be seen from these results, the produced web application is accessible by the most famous internet browsers such as Internet Explorer, Firefox and Chrome. Additionally, the majority of this application's visits previewed at least 8 pages per single visit. This main, a user can move around the website's pages smoothly. Finally, some website's visits were denied. This is because the

However, Google Analytic's experiment results are not very accuracy because this experiment was ran only for two weeks and a few number of users was involved.

11.1.2. Evaluation of experimental test results

In this part, the results will be evaluated against web application's requirements. In general, these requirements are the authentication of users, booking function, email integration, creation of new appointment function, cancellation function, sending appointment to Outlook calendar and Google's calendar.

In Online Lecturer Appointment Booking System, OpenID is used as an authentication technique which allows any user to sing into the website by his/her openID identification. Additionally to this technique, the Online Lecturer Appointment System verifies if the user is enabled to sing in or not by using Activation code. As can be seen from test results, the web application denies any unauthenticated user to sing into the system unless he/she provides the correct activation code at the first time. Moreover, if this user succeeded, he/she can access the system again without activation code. In other word, the amended OpenID authentication technique works successfully.

Another fact that is achieved form this results, the new appointment function works effectively in which a lecturer publishes a new appointment in a short time less than two minutes and then the system invites all students to book appointment via their emails.

The next achievement is that a student can smoothly book appointments in approximately 75 seconds. This means that the booking function works successfully. Despite students receiving confirmation emails of their appointments; a student using windows Live email cannot open the calendar card.

The cancellation appointment function was executed successfully by all students. This process was completed in average time 44 seconds.

This experimental test was run for only one week by a few contributing students. As a consequence, this test might not discover any serious structural weakness or any lack of functionality in this system. In order to increase the efficiency of this experimental test, there should be at least two lecturers and more than 10 students. All of those testers must run this experiment for one month.

11.2. Evaluate the web application

This web application meets the entire core and advanced project's objectives. These objectives are new web online lecturer appointment booking system, light login system, e-mail integration and calendar synchronisation.

In this system, the lecturer can create a new appointment or cancel an existing appointment. Moreover, he can also see the appointment's status such as a number of booked appointments and number of not booked appointment. Additionally, this lecturer can see the time-table of the appointment day. Furthermore, a student can book an appointment time or cancel his booking, he has before.

The email integration in this system notifies system's users the publishing of a new appointment, cancellation existing appointment and calendar card. As a consequence, these users do not have to check the system regularly because this powerful feature makes the users interact with any new or changed event in this system through emails.

However, the produced web application has to improve some system's functions. These functions are the roles in the system, User interface, notification e-mails.
The existing roles in this system are an administrator, a lecturer and students. However, the both administrator and lecturer roles work like one role. This mean, lecturer has the same authority of the administrator in which they do the same work.

11.3. Future work

The Online Lecturer Appointment Booking system for Project students might be more efficient and reliable for the users by enhancing and adding some functions. These

functions are an authorization in the system, notification e-mail system, and involvement of mobile application.

11.3.1. System Authorisation

The current main roles in the system are one lecturer and many students. Additionally, the lecturer can play the administrator role as well. Thus the both administrator and lecturer roles must have different authority and responsibilities. This will enable this system to have more than one user has lecturer authority.

11.3.2. Notification e-mail

This function sends notification email, which is the details of the appointment to a student or lecturer who has an appointment next day. The aim of this function is to remand a user to attend his appointment in order to avoid the absence.

11.3.3. Mobile Application

Currently, the revolution of mobile technology plays an important factor in facilitating access to the most of the systems from most places. Therefore, enable the user to book, cancel, and create an appointment through his/her mobile will increase the efficiency of this application. Additionally, notification messages via SMS (short message service) will be more efficient than e-mail messages.

11.4. Conclusion

The online appointment in general can effective organisation of workload, more efficient management of time, and users are able to plan their day. In this project, the investigation of this project was on creating a web application based on existing solutions, how to eliminate a new username and password for a user, and email integration system with calendar card which can be save in Outlook or Google's calendar.

Finally, the achievement in the project was the combination of OpenID technology and Activation code that were used to eliminate a new username and password for a user. Additionally, other achievement is the V.card which is used to be saved in Outlook or Google's calendar.

Bibliography

Calendar, GenBook.[online] Available at: http://www.genbook.com/manager/serviceprovider/30079653/cal /. [Accessed 11 April 2010].

Choose and Book. *Choose and Book.* [Online] [Cited: 02 27, 2009.] http://www.chooseandbook.nhs.uk/patients .

David, John & Jackie, 2006. *The Three A's: Authorisation, Authentication, and Access – information and guidance.* [Online] TrustDR. Available at: http://trustdr.ulster.ac.uk/work_in_progress/workpackages/WP2-1/theThreeAsWP2_1_29.php . [Accessed 12 May 2010].

***Example of RPX*. [Online] RPX. Available at: https://rpxnow.com/examples/Rpx.cs . [Accessed 14 April 2010].**

Hyun-Kyung & Seung-Hun, 2008. The Security Limitations of SSO in OpenID. [Online] Available at: http://mnet.skku.ac.kr/data/2008data/ICACT2008/pdf/tech/08F-03.pdf [Accessed 12 April 2010]

Imar Spaanjaars, Paul Wilton, and Shawn Livermore. 2006. *ASP.NET 2.0 Instant Results.* Indianapolis : Wiley Publishing, Inc., 2006.

JanRain, *2009. Introduction to RPX*. [Online] RPX. Available at: https://rpxnow.com/docs#overview [Accessed 26 April 2010]

Kerry, 2009. *Authentication, Authorisation and Accounting programme*. [Online] JISC. . Available at: http://www.jisc.ac.uk/whatwedo/programmes/aaa.aspx. [Accessed 1 May 2010].

M m's Calendar, Jiffle.[online] (29 June 2009)Available at: http://www.jifflenow.com/cal/m_m .[Accessed 11 April 2010].

MacDonald, Matthew. 2007. *Beginning ASP.NET 3.5 in VB 2008: From Novice to Professional, Second Edition.* United States of America : apress, 2007.

Manage Time, Jiffle.[online] (29 June 2009)Available at: http://www.jifflenow.com/markmeeting .[Accessed 11 April 2010].

Math Solution, GenBook.[online] Available at: http://math-solutions.genbook.com/. [Accessed 11 April 2010].

***Neil Donkin, 2009 . Using RPX with ASP.NET*. [Online] Softwarenation. Available at: http://www.softwarenation.co.uk/ShowArticle.aspx?id=5542847160244501962). [Accessed 14 April 2010].**

Paul J. Deitel, Harvey M. Deitel. 2008. ***AJAX, Rich Internet Applications, and Web Development For Programmers.*** **Boston : Pearson Education, 2008.**

Peiris, Chris. 2001. 15 Seconds : Creating a .NET Web Service. ***Internet.com Network.*** **[Online] 30 04 2001. [Cited: 03 02 2010.] http://mediakit.internet.com/icom_cgi/print/print.cgi?url=http://www.15seconds.com/Issue/010430.htm**

Pollock, N. & Cornford, J. 2000. Theory and Practice of the Virtual University: report on UK universities use of new technologies. [Online] In ARIADNE issue 24. Available at: http://www.ariadne.ac.uk/issue24/virtual-universities/. [Accessed 12 May 2010].

Robbins, Mari. 2006. ***Medical Receptionists and Secretaries Handbook.*** **fourth. Abingdon : Radcliffe Publishing Ltd, 2006. pp. 55-57.**

Sanderson, Steven. 2009. ***Pro ASP.NET MVC Framework.*** **USA : Apress, 2009.**

Send Invites, Jiffle.[online] (29 June 2009) Available at: http://www.jifflenow.com/invitation/send_inv .[Accessed 11 April 2010].

Setting-Service, GenBook.[online] Available at: http://www.genbook.com/manager/serviceprovider/30079653/settings/services. [Accessed 10 April 2010].

Ubomoibhi, C., Masson, A., & Norris, L., (2001) Integrating VLE and Library Systems: Opportunities and Challenges, Informatica 25 (2001) P1-10

Appendix A – OpenID Code

OpenID code includes these pages and Classes:

- RpxResponse.aspx
- RpxResponse.aspx.VB
- RPX Calss
- Login.aspx
- Login.aspx.vb

RpxResponse.aspx

```
<%@ Page Language="vb" AutoEventWireup="false"
CodeBehind="RpxResponse.aspx.vb" Inherits="online_appointment.RpxResponse" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title></title>
    <script type="text/javascript">

      function pageLoad() {
      }

    </script>
    <style type="text/css">
        .style1
        {
            text-align: center;
        }
        .style2
        {
            color: #CC0000;
        }
    </style>
</head>
<script type="text/javascript">
var gaJsHost = (("https:" == document.location.protocol) ? "https://ssl." :
"http://www.");
document.write(unescape("%3Cscript src='" + gaJsHost + "google-
analytics.com/ga.js' type='text/javascript'%3E%3C/script%3E"));
</script>
<script type="text/javascript">
try {
var pageTracker = _gat._getTracker("UA-16291579-1");
pageTracker._trackPageview();
} catch(err) {}</script>
<body>
    <form id="form1" runat="server">

    <div class="style1">
    <img src="Images/Logo.jpg" />
        <asp:Panel ID="Panel1" runat="server" BackColor="#FAFAFA"
BorderStyle="Groove"
            Height="188px" Width="545px">
            <span class="style2">
            <br />
            Please Enter Your Activation Code</span><br class="style2" />
            <br />
            <br class="style2" />
            <asp:TextBox ID="TextBox1" runat="server"
    CssClass="style2" Height="26px" TextMode="Password"
Width="143px"></asp:TextBox>
            <br />
            <br />
            <asp:Button
```

```
            ID="Button1" runat="server" Text="Submit"
    CssClass="style2" />
                
            <asp:Button ID="Button2" runat="server" Text="Exit"
    Width="53px" />
        </asp:Panel>
        <asp:ScriptManager ID="ScriptManager1" runat="server" />
        <div class="style1">
            <br class="style2" />
        <br class="style2" />

        </div>
    </div>
    <asp:HiddenField ID="HF1" runat="server" />
    <asp:HiddenField ID="HF3" runat="server" />
    <asp:HiddenField ID="HF6" runat="server" />
    <asp:HiddenField ID="HF4" runat="server" />
    <asp:HiddenField ID="HF5" runat="server" />
    <asp:HiddenField ID="HF7" runat="server" />
    <asp:HiddenField ID="HF2" runat="server" />
     <asp:HiddenField ID="HF8" runat="server" />
    </form>
</body>
</html>
```

RpxResponse.aspx.vb

```
Imports System.Xml
Imports System.Web.Security
Imports System.Data.OleDb

Partial Public Class RpxResponse
    Inherits System.Web.UI.Page

    Dim userProvidersUniqueID As String
    Dim displayNameNodeList As XmlNodeList
    Dim displayName As String
    Dim emailAddressNodeList As XmlNodeList
    Dim emailAddress As String
    Dim usernameNodeList As XmlNodeList
    Dim firstname As String
    Dim sirname As String
    Dim formatedName As String
    Dim prefernameNodeList As XmlNodeList
    Dim formattedNodeList As XmlNodeList
    Dim str(2) As String


Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
        If Not IsPostBack Then


        Const apiKey As String = "046a5d24da04d0b89407dc2993d42fb465074453"
        Const paramToken As String = "token"

        ' Get the login token passed back from the RPX authentication service
            Dim loginToken As String = Request.Form(paramToken)

            ' Create an RPX wrapper to get the user's data
        Dim rpx As New Rpx(apiKey, "https://online-appointment.rpxnow.com/")

            ' Get the user's details
        Dim authInfo As XmlElement = rpx.AuthInfo(loginToken)

            ' Log the user in
```

```
            Me.loginUser(authInfo)

        End If
    End Sub

    Private Sub loginUser(ByVal authInfo As XmlElement)

' Get the user's unique identifier (this will ALWAYS be returned
' regardless of the login provider
 userProvidersUniqueID = _
 authInfo.GetElementsByTagName("identifier")(0).InnerText

 ' See if the user's display name is provided (not supplied by some providers

     displayNameNodeList = authInfo.GetElementsByTagName("displayName")

If displayNameNodeList IsNot Nothing AndAlso displayNameNodeList.Count > 0
Then
            ' Got a display name
            displayName = displayNameNodeList(0).InnerText
Else
            ' No display name
        End If
        ' See if the user's email address is provided
        emailAddressNodeList = authInfo.GetElementsByTagName("email")

 If emailAddressNodeList IsNot Nothing AndAlso emailAddressNodeList.Count > 0
Then
            ' Got an email address
            emailAddress = emailAddressNodeList(0).InnerText
        Else
            emailAddress = "error@error.com"
        End If

                usernameNodeList = authInfo.GetElementsByTagName("name")

        If usernameNodeList IsNot Nothing AndAlso usernameNodeList.Count > 0
Then
            ' Got an email address
            firstname = usernameNodeList(0).InnerText


        Else
            firstname = "no name"

            formatedName = "no name"
            ' No email address
        End If

        prefernameNodeList =
authInfo.GetElementsByTagName("preferredUsername")

    If prefernameNodeList IsNot Nothing AndAlso prefernameNodeList.Count > 0
Then
            sirname = prefernameNodeList(0).InnerText
        Else

            sirname = "no name"
        End If
        formattedNodeList = authInfo.GetElementsByTagName("formatted")
        If formattedNodeList IsNot Nothing AndAlso formattedNodeList.Count > 0
Then
            formatedName = prefernameNodeList(0).InnerText
        Else

            formatedName = "no name"
        End If
```

```
        HF1.Value = userProvidersUniqueID
        HF2.Value = displayName
        HF3.Value = emailAddress
        HF4.Value = firstname
        HF5.Value = sirname
        HF6.Value = formatedName


        Dim oleDbConn As New
OleDb.OleDbConnection(ConfigurationManager.ConnectionStrings("AppointmentOnlin
e").ConnectionString)

        oleDbConn.Open()
        Dim sqlactivationCode As String = "Select * From [activationCode] "

        Dim cmdactivationCode As OleDbCommand = New
OleDbCommand(sqlactivationCode, oleDbConn)
        Dim dbread = cmdactivationCode.ExecuteReader()
        Dim i As Integer = 0
        While dbread.Read
            str(i) = dbread.Item("code").ToString
            i = i + 1
        End While

        dbread.Close()
        oleDbConn.Close()
        HF7.Value = str(0).ToString
        HF8.Value = str(1).ToString



        checkUser(userProvidersUniqueID)


    End Sub

    Sub addnewusers(ByVal id As String, ByVal disName As String, ByVal em As
String, ByVal fname As String, ByVal sname As String, ByVal ftname As String)
        Dim oleDbConn As New
OleDb.OleDbConnection(ConfigurationManager.ConnectionStrings("AppointmentOnlin
e").ConnectionString)

        oleDbConn.Open()
        Dim sqluser As String = "Select * From [user] Where userLoginName=" &
"'" & id & "'"

        Dim cmduser As OleDbCommand = New OleDbCommand(sqluser, oleDbConn)
        Dim dbread = cmduser.ExecuteReader()
        If Not dbread.Read() Then
            dbread.Close()
            cmduser.Dispose()

            If em = Nothing Then
                em = "error@error.com"
            End If

            Dim sqladduser As String = "INSERT INTO [user] (
[userLoginName],[email],
[userDisplayedName],[firstName],[sirName],[formattedName],[isfirstTime],[group
_no]) VALUES(" & "'" & id & "'," & " '" & em & "', " & "'" & disName & "', " &
"'" & fname & "', " _
            & "'" & sname & "', " & "'" & ftname & "', 1," & "'" &
(TextBox1.Text.ToString) & "')"
            Dim newusercmd As OleDbCommand = New OleDbCommand(sqladduser,
oleDbConn)


            newusercmd.ExecuteNonQuery()
```

```
            newusercmd.Dispose()
        Else

            dbread.Close()
            cmduser.Dispose()

            Dim sqladduser As String = "UPDATE  [user] SET [group_no]= '" &
(TextBox1.Text) & "' WHERE [userLoginName] =" & "'" & id & "'"
            Dim newusercmd As OleDbCommand = New OleDbCommand(sqladduser,
oleDbConn)
            newusercmd.ExecuteNonQuery()
            newusercmd.Dispose()

        End If

        oleDbConn.Close()

    End Sub

    Protected Sub Button1_Click(ByVal sender As Object, ByVal e As EventArgs)
Handles Button1.Click
        Dim result As String = Nothing
        If TextBox1.Text.Equals(HF7.Value.ToString) Then
            FormsAuthentication.SetAuthCookie(HF1.Value, False)
            addnewusers(HF1.Value, HF2.Value, HF3.Value, HF4.Value, HF5.Value,
HF6.Value)
            ' Set the authentication cookie and go back to the home page



            Response.Redirect("~/Default.aspx")
        ElseIf TextBox1.Text.Equals(HF8.Value.ToString) Then

            Membership.CreateUser(HF1.Value, "123456789-", HF3.Value,
"lecturer", "yes", True, result)
            Roles.AddUserToRole(HF1.Value, "Admin")
            FormsAuthentication.SetAuthCookie(HF1.Value, False)
            addnewusers(HF1.Value, HF2.Value, HF3.Value, HF4.Value, HF5.Value,
HF6.Value)
            ' Set the authentication cookie and go back to the home page



            Response.Redirect("~/Default.aspx")
        Else
            FormsAuthentication.SignOut()
            Response.Redirect("login.aspx")

        End If
    End Sub

    Sub checkUser(ByVal id As String)
        Dim oleDbConn As New
OleDb.OleDbConnection(ConfigurationManager.ConnectionStrings("AppointmentOnlin
e").ConnectionString)

        oleDbConn.Open()
        Dim sqluser As String = "Select * From [user] Where userLoginName=" &
"'" & id & "'"

        Dim cmduser As OleDbCommand = New OleDbCommand(sqluser, oleDbConn)
        Dim dbread = cmduser.ExecuteReader()
        If Not dbread.Read() Then
            dbread.Close()
            cmduser.Dispose()
        Else
```

```
            If dbread.Item("group_no").Equals(str(0).ToString) Or
dbread.Item("group_no").Equals(str(1).ToString) Then
                dbread.Close()
                cmduser.Dispose()
                FormsAuthentication.SetAuthCookie(userProvidersUniqueID,
False)

                Response.Redirect("~/Default.aspx")
            Else
                dbread.Close()
                cmduser.Dispose()

            End If
        End If

    End Sub

    Protected Sub Button2_Click(ByVal sender As Object, ByVal e As EventArgs)
Handles Button2.Click
        Response.Redirect("login.aspx")
    End Sub
End Class
```

RPX Class

```
Imports System
Imports System.Collections
Imports System.Collections.Generic
Imports System.IO
Imports System.Net
Imports System.Text
Imports System.Web
Imports System.Xml
Imports System.Xml.XPath

Public Class Rpx
    Private apiKey As String
    Private baseUrl As String

    Public Sub New(ByVal apiKey As String, ByVal baseUrl As String)
        While baseUrl.EndsWith("/")
            baseUrl = baseUrl.Substring(0, baseUrl.Length - 1)
        End While

        Me.apiKey = apiKey
        Me.baseUrl = baseUrl
    End Sub

    Public Function getApiKey() As String
        Return apiKey
    End Function
    Public Function getBaseUrl() As String
        Return baseUrl
    End Function

    Public Function AuthInfo(ByVal token As String) As XmlElement
        Dim query As New Dictionary(Of String, String)()
        query.Add("token", token)
        Return ApiCall("auth_info", query)
    End Function

    Public Function Mappings(ByVal primaryKey As String) As ArrayList
        Dim query As New Dictionary(Of String, String)()
        query.Add("primaryKey", primaryKey)
        Dim rsp As XmlElement = ApiCall("mappings", query)
        Dim oids As XmlElement = DirectCast(rsp.FirstChild, XmlElement)
```

```
        Dim result As New ArrayList()

        For i As Integer = 0 To oids.ChildNodes.Count - 1
            result.Add(oids.ChildNodes(i).InnerText)
        Next

        Return result
    End Function

    Public Function AllMappings() As Dictionary(Of String, ArrayList)
        Dim query As New Dictionary(Of String, String)()
        Dim rsp As XmlElement = ApiCall("all_mappings", query)

        Dim result As New Dictionary(Of String, ArrayList)()
        Dim nav As XPathNavigator = rsp.CreateNavigator()

        Dim mappings As XPathNodeIterator =
DirectCast(nav.Evaluate("/rsp/mappings/mapping"), XPathNodeIterator)
        For Each m As XPathNavigator In mappings
            Dim remote_key As String = GetContents("./primaryKey/text()", m)
            Dim ident_nodes As XPathNodeIterator =
DirectCast(m.Evaluate("./identifiers/identifier"), XPathNodeIterator)
            Dim identifiers As New ArrayList()
            For Each i As XPathNavigator In ident_nodes
                identifiers.Add(i.ToString())
            Next

            result.Add(remote_key, identifiers)
        Next

        Return result
    End Function

    Private Function GetContents(ByVal xpath_expr As String, ByVal nav As
XPathNavigator) As String
        Dim rk_nodes As XPathNodeIterator =
DirectCast(nav.Evaluate(xpath_expr), XPathNodeIterator)
        While rk_nodes.MoveNext()
            Return rk_nodes.Current.ToString()
        End While
        Return Nothing
    End Function

    Public Sub Map(ByVal identifier As String, ByVal primaryKey As String)
        Dim query As New Dictionary(Of String, String)()
        query.Add("identifier", identifier)
        query.Add("primaryKey", primaryKey)
        ApiCall("map", query)
    End Sub

    Public Sub Unmap(ByVal identifier As String, ByVal primaryKey As String)
        Dim query As New Dictionary(Of String, String)()
        query.Add("identifier", identifier)
        query.Add("primaryKey", primaryKey)
        ApiCall("unmap", query)
    End Sub

    Private Function ApiCall(ByVal methodName As String, ByVal partialQuery As
Dictionary(Of String, String)) As XmlElement
        Dim query As New Dictionary(Of String, String)(partialQuery)
        query.Add("format", "xml")
        query.Add("apiKey", apiKey)

        Dim sb As New StringBuilder()
        For Each e As KeyValuePair(Of String, String) In query
            If sb.Length > 0 Then
                sb.Append("&"c)
```

```
            End If

            sb.Append(System.Web.HttpUtility.UrlEncode(e.Key, Encoding.UTF8))
            sb.Append("="c)
            sb.Append(HttpUtility.UrlEncode(e.Value, Encoding.UTF8))
        Next
        Dim data As String = sb.ToString()

        Dim url As New Uri((baseUrl & "/api/v2/") + methodName)

        Dim request As HttpWebRequest = DirectCast(WebRequest.Create(url),
HttpWebRequest)
        request.Method = "POST"

        request.ContentType = "application/x-www-form-urlencoded"
        request.ContentLength = data.Length

        ' Write the request
        Dim stOut As New StreamWriter(request.GetRequestStream(),
Encoding.ASCII)
        stOut.Write(data)
        stOut.Close()

        Dim response As HttpWebResponse = DirectCast(request.GetResponse(),
HttpWebResponse)
        Dim dataStream As Stream = response.GetResponseStream()

        Dim doc As New XmlDocument()
        doc.PreserveWhitespace = False
        doc.Load(dataStream)

        Dim resp As XmlElement = doc.DocumentElement

        If Not resp.GetAttribute("stat").Equals("ok") Then

            Throw New SystemException("Unexpected API error")
        End If

        Return resp
    End Function

    Public Shared Sub Main(ByVal args As String())
        Dim r As New Rpx(args(0), args(1))

        If args(2).Equals("mappings") Then
            Console.WriteLine("Mappings for " & args(3) & ":")
            For Each s As String In r.Mappings(args(3))
                Console.WriteLine(s)
            Next
        End If

        If args(2).Equals("all_mappings") Then
            Console.WriteLine("All mappings:")

            For Each pair As KeyValuePair(Of String, ArrayList) In
r.AllMappings()
                Console.WriteLine(pair.Key & ":")
                For Each identifier As String In pair.Value
                    Console.WriteLine("  " & identifier)
                Next
            Next
        End If

        If args(2).Equals("map") Then
            Console.WriteLine((args(3) & " mapped to ") + args(4))
            r.Map(args(3), args(4))
        End If
```

```
        If args(2).Equals("unmap") Then
            Console.WriteLine((args(3) & " unmapped from ") + args(4))
            r.Unmap(args(3), args(4))

        End If
    End Sub
End Class
```

Login.aspx

```
<%@ Page Language="vb" AutoEventWireup="false" CodeBehind="login.aspx.vb"
Inherits="online_appointment.login"  %>


<script src="https://rpxnow.com/openid/v2/widget"
type="text/javascript"></script>

<script type="text/javascript">
    RPXNOW.overlay = true;
    RPXNOW.language_preference = 'en';
</script>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head id="Head1" runat="server">
    <title></title>

    <style type="text/css">
        .rpxnow
        {
            text-align: center;
            font-weight: 700;
            font-size: xx-large;
            color: #FF0000;
        }
        .style1
        {
            color: #0000FF;
            font-weight: bold;
        }
    </style>

</head>
<script type="text/javascript">
var gaJsHost = (("https:" == document.location.protocol) ? "https://ssl." :
"http://www.");
document.write(unescape("%3Cscript src='" + gaJsHost + "google-
analytics.com/ga.js' type='text/javascript'%3E%3C/script%3E"));
</script>
<script type="text/javascript">
try {
var pageTracker = _gat._getTracker("UA-16291579-1");
pageTracker._trackPageview();
} catch(err) {}</script>
<body>
    <form id="form1" runat="server">
    <div>


            
      <img
            alt="ONLINE APPOINTMENT SYSTEM" src="Images/Logo.jpg"
            style="width: 970px; height: 125px"
/>            &nbs
p;    
```

```
        <br />
        Welcome to Online Lecturar Appointment System<br />
        <br />
        <span class="style1">No more registration</span>  Just
        <asp:HyperLink ID="lnkSignIn" runat="server" CssClass="rpxnow"
onclick="return false;" Text="Sign In" />
         ( with your Google, Yahoo, faceBook or OpenID )<br />
    </div>

    </form>
</body>
</html>
```

Login.aspx.vb

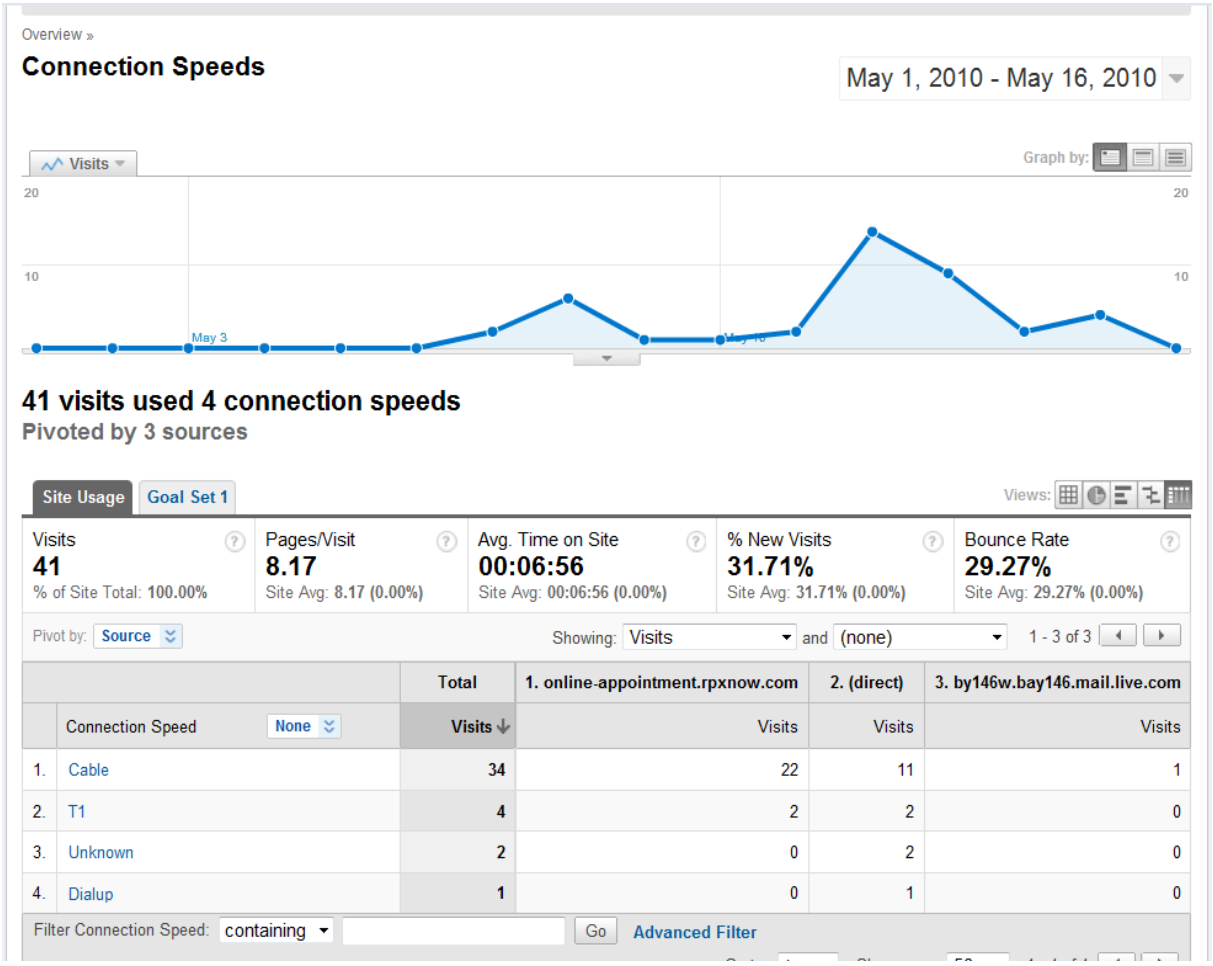
```
Imports DotfuscatorAttribute
Imports System
Imports System.Net
Imports System.Web.Security
Imports System.Web.UI
Imports System.Web.UI.WebControls
Imports System.Xml
Imports System.IO
Imports RPXLib

Partial Public Class login
    Inherits System.Web.UI.Page

Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
        Me.lnkSignIn.NavigateUrl = "https://online-
appointment.rpxnow.com/openid/v2/signin?token_url=http://" &
Request.Url.Authority & "/RpxResponse.aspx"
End Sub

End Class
```

Appendix B – Google Analytic Results

Week	Pageviews
May 2, 2010 - May 8, 2010	76
May 9, 2010 - May 15, 2010	259

Page	Pageviews	Unique Pageviews	Avg. Time on Page	Bounce Rate	% Exit
/Default.aspx	45	19	39.45	0.00	0.11
/RpxResponse.aspx	36	13	43.09	0.33	0.03
/userappointment.aspx	36	10	59.42	0.00	0.08
/login.aspx?ReturnUrl=/default.aspx	25	21	176.14	0.19	0.16
/controlPage.aspx	23	4	33.77	0.00	0.04
/ConsultatoinPage.aspx	20	11	9.00	0.00	0.05
/login.aspx?ReturnUrl=/Default.aspx	20	14	104.57	0.56	0.30
/login.aspx	18	8	47.86	0.00	0.22
/mconsultation.aspx	18	10	58.94	0.00	0.00
/BookAppointment.aspx?ID=45	10	4	24.44	0.00	0.10
/createNewAppointment.aspx	10	2	73.60	0.00	0.00
/login.aspx?ReturnUrl=/mconsultation.aspx	10	8	107.40	0.00	0.50
/BookAppointment.aspx?ID=40	6	4	17.67	0.00	0.00
/login.aspx?ReturnUrl=/userappointment.aspx	6	4	55.25	0.00	0.33
/lecturerlogin.aspx	5	2	99.00	0.00	0.00
/login.aspx?ReturnUrl=/controlPage.aspx	5	4	49.50	1.00	0.60
/userProfile.aspx	5	5	42.20	0.00	0.00
/userProfile.aspx?act=1	4	4	18.75	0.00	0.00
/confirmBooking.aspx?ID=856	3	2	14.67	0.00	0.00
/AmendConsultation.aspx?ID=40	2	2	27.50	0.00	0.00
/BookAppointment.aspx?ID=39	2	1	6.50	0.00	0.00
/Muserappointment.aspx?ID=20	2	1	10.00	0.00	0.00
/confirmBooking.aspx?ID=853	2	1	15.00	0.00	0.00
/confirmBooking.aspx?ID=854	2	1	23.00	0.00	0.00
/helppage.htm	2	2	42.50	0.00	0.00
/login.aspx?ReturnUrl=/BookAppointment.aspx?ID=45&ID=45	2	2	0.00	1.00	1.00
/login.aspx?ReturnUrl=/createNewAppointment.aspx	2	1	270.00	0.00	0.00
/login.aspx?ReturnUrl=/userProfile.aspx	2	2	0.00	0.00	1.00
/BookAppointment.aspx?ID=43	1	1	12.00	0.00	0.00
/confirmBooking.aspx?ID=784	1	1	5.00	0.00	0.00
/confirmBooking.aspx?ID=785	1	1	3.00	0.00	0.00
/confirmBooking.aspx?ID=789	1	1	3.00	0.00	0.00
/confirmBooking.aspx?ID=796	1	1	17.00	0.00	0.00
/confirmBooking.aspx?ID=799	1	1	28.00	0.00	0.00
/confirmBooking.aspx?ID=801	1	1	9.00	0.00	0.00
/confirmBooking.aspx?ID=835	1	1	3.00	0.00	0.00

/login.aspx?ReturnUrl=/AmendConsultation.aspx?ID=10&ID=10	1	1	585.00	0.00	0.00
/login.aspx?ReturnUrl=/AmendConsultation.aspx?ID=40&ID=40	1	1	0.00	0.00	1.00
/login.aspx?ReturnUrl=/confirmBooking.aspx?ID=856&ID=856	1	1	13.00	0.00	0.00
/login.aspx?ReturnUrl=/lectuerlogin.aspx	1	1	10.00	0.00	0.00



Most visits tracked: 1 pageviews

Pageviews in the visit	Visits with this many pageviews	Percentage of all visits
1 pageviews	**12.00**	29.27%
2 pageviews	**3.00**	7.32%
3 pageviews	**2.00**	4.88%
4 pageviews	**2.00**	4.88%
5 pageviews	**2.00**	4.88%
8 pageviews	**1.00**	2.44%
9 pageviews	**1.00**	2.44%
10 pageviews	**2.00**	4.88%
11 pageviews	**3.00**	7.32%
12 pageviews	**5.00**	12.20%
14 pageviews	**2.00**	4.88%
17 pageviews	**2.00**	4.88%
18 pageviews	**1.00**	2.44%
20+ pageviews	**3.00**	7.32%

Most visits repeated: 1 times

Count of visits from this visitor including current	Visits that were the visitor's nth visit	Percentage of all visits
1 times	**13.00**	31.71%
2 times	**8.00**	19.51%
3 times	**3.00**	7.32%
4 times	**2.00**	4.88%
5 times	**2.00**	4.88%
6 times	**2.00**	4.88%
7 times	**2.00**	4.88%
8 times	**2.00**	4.88%
9-14 times	**7.00**	17.07%

Appendix C – Experiment Results

task-2	part 1	sings in the system using wrong activation code first attempt	
Name	Role	login Identity Provider	does Login with wrong Activation code
Farahan	Lecturer	Windows Live	No
Sareh	Student	FaceBook	No
Aiman	Student	Google	No
Mohammad	Student	Yahoo	No

Task-2: Log into the system first time and second time					
Name	role	IDP	login 1st time to system	length of login 1st time(sec)	second time (sec)
Farahan	lecturer	Windows Live	Yes	37	23
Sareh	student	FaceBook	Yes	43	17
Aiman	student	Google	Yes	15	5
Mohammad	student	Yahoo	Yes	18	14
average time in second				28.25	14.75

Task-3: Lecturer Creates a New Appointment				
Name	Role	login Identity Provider	create appointment time	recive email after creation
Farahan	lecturer	Windows Live	84 sec	Yes
Sareh	student	FaceBook	N/A	Yes
Aiman	student	Google	N/A	Yes
Mohammad	student	Yahoo	N/A	Yes

Task-4: Student Books Appointment							
Name	role	login Identity Provider	can book	length time of booking	receive e-mail	Receive V-card	can open email
Sareh	student	FaceBook	Yes	64	yes	yes	no
Aiman	student	Google	Yes	78	yes	yes	yes
Mohammad	student	Yahoo	Yes	82	yes	yes	yes
			Average	74.67			

task-5: Students cancelled their appointments				
Name	Role	login Identity Provider	cancel appointment time	is cancelled
Sareh	student	FaceBook	44	Yes
Aiman	student	Google	47	Yes
Mohammad	student	Yahoo	43	Yes
		Avarage	44.67 seconds	

Appendix D – Extended Project Proposal

Computer Science Modular Masters Project May 2010

Extended Project Proposal

Student Family Name: Mohammad	**Student First Name: Alqahtani**
Student Registration Number: **07171945**	**Supervisor Name:** Bennett, Steve
Award Title (e.g. Software Engineering) **Computer Science**	**Second Marker Name:** Dawkins, Keith
Project Title/Research Question: Write an online lecturer appointment system for project students. In this, in a calendar view the lecturer will choose times of availability, and length of consultations. The system will then become "bookable" by students. As an advanced objective, the resulting feed should then be capable of being viewed in popular calendaring systems such as Google calendar or Microsoft outlook. This will involve writing a web application in aspx/jsp/php and evaluating the usability of the system	
Student Signature:	
Date:	

1. Introduction

1.1 Brief Description of the Project

An online lecturer appointment system for project students is a web application that can be used by a lecturer and his students to make and book an appointment. In this system, the lecturer chooses times of availability from a calendar view, and estimates a length and a location of consultations. Then, the system will become "bookable" by students.

The aim of this project is to produce a web application that enables a lecturer to plan scheduled consultation appointments for his students, in order to enable each student to book a mutually suitable appointment time. This application should encorporate the best practices from the existing online appointment systems. Then, this application will be evaluated in term of usability and reliability.

1.2 Background to the project

The appointment system was first used in the health services as a paper work system in the 1970s in England. The aim of this system was to reduce the waiting time of the patients and distribute those patients to available consultants. After that, it was improved to be a computer system because of the benefits that go to both patients and consultants, such as reduction of waiting time. In this system, a patient can make his appointment on a specific day by phoning the health centre to check the availability and reserve a time with his doctor. The advantages can be taken from the health appointment system for both consultants and their clients are: effective organisation of workload, more efficient management of time, it limits the number of patients seen in one session, less waiting-room space is required, the duration of the consultation can be varied according to predetermined need and patients are able to plan their day. This view has been supported in the work of Robbins (2006, p55). The latest booking system used by the NHS is called “choose & book”, which is a service that enables the patient to choose via http://www.chooseandbook.nhs.uk/ a hospital and a specialist of his first appointment after his GP agrees that he needs to see a specialist (Choose and Book).

Currently, there are many different online appointment systems, depending on the purpose of the businesses such as a haircut shop or business consultations. The

Genbook website is an example of a general Appointment Booking System (ABS) which can be a part of any existing website in order to manage the appointment. This system has very powerful features such as online booking, automatic appointment confirmation, Flexible appointment scheduling, online scheduling page, Customer Reviews and, synchronise to other calendars such as Google calendar (Genbook) .

The main futures of a general online system are consists of two parts: the public end-user section and the

maintenance section (administrator). The public user section is where end-users can sign up for an account, check availability, and make appointments (Imar Spaanjaars, Paul Wilton, and Shawn Livermore, 2006). The second part is the maintenance section where the administrator can plan and manage the appointments.

2. The nature of the proposed work and the methodology

The methodology used for this project is to study the current online appointment systems in terms of the functions of these systems, weaknesses and good practices. The aim of this study is to have clear ideas about the good and bad practices in these systems. Then based on the results of the study, the Online Lecturer Appointment System for Project Students will be produced. After that, the new produced system will be tested by some professionals based on a test plan . After the test of the system, the feedback is collected and then the comments and errors are fixed and are taken in consideration. Finally, this application will be evaluated in term of usability and reliability.

3. The main objectives of the project

The general objective of the project is to write an online lecturer appointment system for project students. In this, using a calendar view, the lecturer will choose times of availability and length of consultations. The system will then become "bookable" by students. As an advanced objective, the resulting feed should then be capable of being viewed in popular calendaring systems such as Google calendar or Microsoft outlook. This will involve writing a web application in aspx and evaluating the usability of the system.

3.1 Project Objectives:

3.1.1 Core Objectives:

- Produce an online lecturer appointment system for project students. This system must have at least the following functions:
    1. Two roles: a lecturer as administrator and a student
    2. Appointment Manager which creates a schedule for appointment and sets up the duration and location of the consultation.
    3. The Appointment: represents an appointment that has been made in the system.
    4. Availability Checker: shows the available appointments.
    5. Booking appointment
    6. Appointment confirmation: sends confirmation to the user
    7. User commitment evaluation: shows the user's commitments to attending appointments.
    8. Study some existence online systems in terms of Appointment Manager, Booking appointment, Appointment confirmation, Availability Checker and Synchronize to other Calendars.
- Determine the positive points and the drawbacks of the usability and reliability of the system in terms of the above functions.

3.1.2 Advanced objectives:

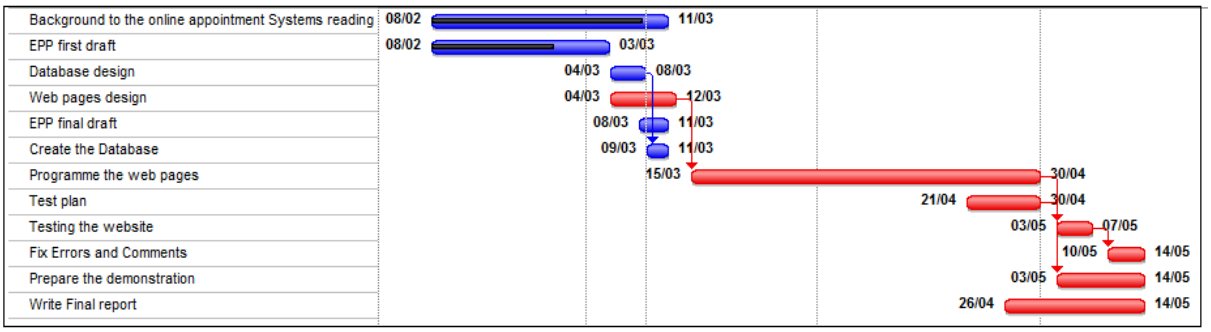
- The system can synchronize to Google Calendars or Microsoft Outlook Calendar.
- The system can demonstrate the location of the consultation room, for example, using Google Map.
- The system automatically can plan, schedule, cancel, and reschedule the appointments.

4. Working plan for the project

task#	Task	duration	start date	end date
1	Background to the online appointment	24 days	08/02/2010	11/03/2010

	Systems			
2	EPP first draft	18 days	08/02/2010	03/03/2010
3	Database design	3 days	04/03/2010	08/03/2010
4	Web pages design	7 days	04/03/2010	12/03/2010
5	EPP final draft	4 days	08/03/2010	11/03/2010
6	Create the Database	3 days	09/03/2010	11/03/2010
7	Programme the web pages	35 days	15/03/2010	30/04/2010
8	Test plan	8 days	21/04/2010	30/04/2010
9	Testing the website	5 days	03/05/2010	07/05/2010
10	Fix Errors and Comments	5 days	10/05/2010	14/05/2010
11	Prepare the demonstration	10 days	03/05/2010	14/05/2010
12	Write Final report	15 days	26/04/2010	14/05/2010

Gantt chart of the project plan:



5. particular hardware or software requirements

Software:

- Microsoft visual studio 2008
- Microsoft Access 2007
- Microsoft project 2007
- Net beans 6.5

Hardware

- PC

Bibliography

Choose and Book. *Choose and Book.* [Online] [Cited: 02 27, 2009.] http://www.chooseandbook.nhs.uk/patients.

Genbook. Genbook. *Genbook.* [Online] [Cited: 03 01, 2010.] http://www.genbook.com/index.html.

Imar Spaanjaars, Paul Wilton, and Shawn Livermore. 2006. *ASP.NET 2.0 Instant Results.* Indianapolis : Wiley Publishing, Inc., 2006.

MacDonald, Matthew. 2007. *Beginning ASP.NET 3.5 in VB 2008: From Novice to Professional, Second Edition.* United States of America : apress, 2007.

Paul J. Deitel, Harvey M. Deitel. 2008. *AJAX, Rich Internet Applications, and Web Development For Programmers.* Boston : Pearson Education, 2008.

Peiris, Chris. 2001. 15 Seconds : Creating a .NET Web Service. *Internet.com Network.* [Online] 30 04 2001. [Cited: 03 02 2010.] http://mediakit.internet.com/icom_cgi/print/print.cgi?url=http://www.15seconds.com/Issue/010430.htm.

Robbins, Mari. 2006. *Medical Receptionists and Secretaries Handbook.* fourth. Abingdon : Radcliffe Publishing Ltd, 2006. pp. 55-57.

Sanderson, Steven. 2009. *Pro ASP.NET MVC Framework.* USA : Apress, 2009.