

C O V E N T R Y

U N I V E R S I T Y

Faculty of Engineering and Computing
Department of Computing and the Digital Environment

EC019 – European Direct Entry
300CDE – Individual Project (Mechatronics)

Ticketing system

Author: Karol Gornicki

SID: 3417562

Supervisor: Dr Norlaily Yaacob

Submitted in partial fulfilment of the requirements for the Degree of Bachelor of Engineering Subject Informatics

Academic Year: 20010/11

Office Stamp
Declaration of Originality

This project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet, etc) has been acknowledged within the main report to an entry in the References list.

I agree that an electronic copy of this report may be stored and used for the purposes of plagiarism prevention and detection.

I understand that cheating and plagiarism constitute a breach of University Regulations and will be dealt with accordingly.

Signed:

Date:

Copyright

The copyright of this project and report belongs to Coventry University.

Abstract

This document refers to development process of web based ticket selling system. Firstly, work has been preceded by recognition of problem, analysis of it. After taking outcomes from these stages into consideration, objectives of newly designed system were established. At this point the strategy of development was set, and was followed by precise design of entire application, implementation, tests, and drawing report that documents the whole process of creation and final result of the project - a working system.

Research was based on online materials and several books, and existing systems that perform similar activities. Draft of application has been prepared in accordance with UML 2.1 standards which established backbone of application and scheduled implementation stage, which was broken into series of smaller tasks in order to better manage work.

System was created in .NET framework, both 3.5 and 4 version, and different parts cooperate without any conflicts. Each project relies on Microsoft SQL Server 2008, which hosts its database. To manage data sets LINQ has been at play, which showed how easy it can be to compare to traditional SQL statements.

Entire process has been scheduled at the beginning and followed to the end, with little deviation, which did not disturb the process in general.

Table of contents

Abstract.....	4
Additional materials on the accompanying CD.....	6
Acknowledgements.....	7
1. Introduction.....	8
1.1. Background of the project.....	8
1.2. Project objectives.....	8
1.3. Overview of this project.....	9
2. Methodology.....	12
3. Analysis of problem.....	14
3.1. Investigated web applications.....	14
3.2. Draft of application.....	15
4. Literature review and specifying tools.....	17
5. Design.....	21
5.1. Overall design of application's structure.....	21
5.2. Design of order's database.....	23
5.3. Design of Web Services.....	24
5.4. Schedule of implementation.....	25
6. Implementation.....	26
6.1. Main application.....	26
6.2. PDF printer.....	30
6.3. Web Services.....	31
7. Testing and results.....	32
8. Project management.....	33
8.1. Project schedule.....	33
8.2. Risk management.....	33
8.3. Quality management.....	34
9. Critical appraisal.....	35
10. Conclusions.....	37
10.1. Achievements.....	37
10.2. Recommendations for the future work.....	37
11. Student reflections.....	39
Bibliography and references.....	40

Additional materials on the accompanying CD

- Project's source code:
 - Ballet Web Service
 - Opera Web Service
 - Theatre Web Service
 - Main application
 - ReadMe file with explanation how to run project
- Report
- Diagrams

Acknowledgements

I would like to thank Dr Norlaily Yaacob and Tomasz Siekierda for the support and guidance I received from them.

1. Introduction

The first chapter covers the induction into the project. Its background, in terms of reasons why it has been created, and requirements that common existing web application systems face. Dealing with the project has been drafted by setting the objectives which are specified in the subsequent sub-chapter. They determined the scope of this project, indicated the most important aspects of it and affected the way the application has been created and before that, established main direction of research that has been taken. Last stage of this chapter is dedicated to overview of this report which outlines the structure of the entire document and briefly describes each of the following chapters.

1.1. Background of the project

The aim of this project, Ticketing system, is to create a web application of ticketing system. Each institution that performs some kind of entertainment has already created a web site which enables the user to gather information about repertoire and furthermore book seats for certain events. Those applications share similarities with most on-line shops. However, the majority of them are oriented only to one organisation, or sometimes few members of one consortium.

This circumstance has brought a challenge to create a web application that would be eligible for an agency that desires to aggregate the repertoire from different places and deliver those resources to the users in order to facilitate the decision making process. The application had to be designed that fully automatically handles with it in order to gathering and displaying relevant information. Since data are coming from different facilities to simulate this process, the perfect scenario was to design distributed application structure, where each institution is represented by different Web Service which can be constituted as gateway for exchanging data with external agency web site. This approach enables the application to be always updated, and give the whole power of fulfilling the contents of the repertoire to the original supplier. Since they administrate Web Services on their own, although standardising data, and messages that are exchanged over the Internet is crucial to keep the structure of application as easy as it can be.

To achieve success of newly designed web application, the application must meet all the requirements that already existing systems face, and moreover add new value to the buying process that gives a chance to compare on the market with active ticketing systems, because newly introduced companies do not have any chance to compare with others only when they duplicate patterns.

1.2. Project objectives

The objectives of this project directed the way the research had been taken and later, affected the development process. The general aim was to create a competitive to present once ticketing system, which share similarities with some and extend the functionality area by applying new patterns that improve buying process which would be appreciated by

users. Detailed objectives are pointed out below.

- Distributed application architecture – by applying this structure project can truly imitate the environment that agent's web site would have to face in a real business. What is more, this scenario forces to establish standard of data and messages that are exchanged along this process in term of simplifying as much as it is possible the structure of application and algorithms that were implemented to meet required functionalities.
- Automated process of gathering and displaying relevant information – the preconceived idea was, that Web Services are not the property of agency, they are ran by the organization which they represent. Therefore, agent's web site has to gather information automatically, not transfer it manually. Agent's web site only interferes with data stored in Web Services in case of booking or cancelling orders (details that lay down to this process depends on agreement between two interested sides). Same approach is related to processing that data, and presenting to the end-user – flexibility of the system depending on the contents is crucial to satisfy average user.
- Easy to navigate interface – successfulness of the web site has two faces: functionality and interface. Well designed, easy to follow interface is a key to possessing customers.
- Informing users about upcoming events – nowadays e-commerce web site has to go in front of the customers needs, thus has to propose some events that user might be interested to attend.
- Managing shopping cart – this is necessary aspect of all on-line shops. Buying one ticket at a time would simply turn off users from purchasing at this web site. All common functionalities used at different shops should be applied as well.
- Ordering tickets for certain entertainments – the main functionality of the application. It would be managed by updating database stored on proper Web Service, through specially designed method.
- Cancelling orders – similar to ordering, although this process requires deeper understanding. To increase flexibility user should be able to cancel the whole order that he made once, but also just a single ticket. Detailed conditions whenever user is allowed to do that, as well, repayment issue has to be discussed by interested sides. Nevertheless system should easily cope with it.
- Printing the tickets – should be done automatically, based on order to prove the purchasing, and optionally to be a foundation for future cancelling.
- Additional functionalities – functionalities that can sharply increase interests of the web site, attract other business entities to cooperate with, or simplify decision making process to the end-user which may benefit with his or her trust to the company. For instance recalculating the sub-total into different currencies.

Apart from issues that are related to the content of the system the natural objective of report is to present how technology that has been chosen can facilitate the process of creating that sort of software as well as how it deals with it in term of implementation.

1.3. Overview of this project

The report has been structured into several chapters which describe in a chronological

order processes involved in designing and creating the application that was the main object of this project as well as supplement comments that can give wider view on investigated stage or draft for future development.

Chapter 2, "Methodology". Second chapter covers methodology which defines the process that has been applied to complete this project: from research stage, through designing, implementation and ending on evaluation and drawing the conclusions. Also determines timetable, that has been followed through the whole process.

Chapter 3, "Analysis of problem". This chapter is concentrated on analysis of existing ticketing systems, pointing out strengths and pitfalls of patterns that have been implemented, appraisal each of them in terms of usability, complexity, and sense of righteousness.

Chapter 4, "Literature review". This stage of the report presents outcomes of research that has been taken in order to gather information of technology issues – how to implement certain functionalities, as well as determine which approaches are feasible, and support each other in terms of syntax and standardising data. Conclusions derived from this chapter truly affected the design and implementation stage even in order to make fundamental decisions.

Chapter 5, "Design". This chapter is preceding the implementation stage and builds fundamentals for it. Ships diagrams, justifies patterns that would be used to carry out the task, and explains the data flow alongside the concept of whole application. The conclusions of this section inherit from outcomes of two previous chapters.

Chapter 6, "Implementation". This phase of the report testifies the implementation, by describing of how, from technical point of view each part has been done, as well as by screen shots which document final effort.

Chapter 7, "Testing and evaluation". This chapter proofs that created application truly meet the requirement that has been established at the beginning of the process.

Chapter 8, "Project management". This stage explains how project was managed throughout development process. Adopted standards and techniques crucial to the process are emphasised.

Chapter 9, "Critical appraisal". This chapter presents critical appraisal about this project.

Chapter 10, "Conclusions". This chapter summarise the project in order to development process as well as created application, and presents challenges on the horizon for further development.

Chapter 11, "Student reflection". Last stage is emphasises author's reflections about entire report.

2. Methodology

In order to track the progress of work for this report, waterfall model of organising the work has been adopted. The biggest strength of this approach is highly structured workflow divided into stages in which one inherits outcomes from the previous stage. During the process it is not advisable to go back and modify already completed stages, because they might affect previous outcomes and the process of refactorisation might call for additional overwhelming work. Therefore model applied here had slight modification in comparison with traditional one – project planning stage which was oriented on determining the scope of research as well as creating predesigned model of application, which may be changed a bit because of outcomes of research stage.

Since requirements for this project have been established at the very beginning there was no obligation to abandon this approach, especially when project was developed single-handedly. In case of cooperation with real client or team. A more efficient approach would be an interactive and incremental development model which would offer more contact with the client, who can track the progress and propose improvements with immediacy.

To coordinate all stages, and certainty that everything is done in right time each stage of the model has been assigned with certain period of time and to each estimated time of work that is required. Gantt char and other related materials are enclosed in Appendix A – Specification of the project.

First stage covered the basic needs of the project – establishing the requirements and objectives that the application has to meet alongside learning objectives which are highly related to the main topic. These are included in Specification of the project (Appendix A).

Next stage was related to gathering information about running ticketing systems, with focus on those which managed entertainment. The final draft was created capitalizing on their strengths and drawbacks which was the backbone for development of the application. The task has taken around 10 hours, and was performed in time. The outcomes of this stage are contained in Chapter 3.

Following stage concentrated on project planning which was related to drawing diagrams determining the functionalities of this application, and marked out objectives for the literature review stage, in terms of capabilities of technology of handling with certain features. It was predesign of the application, less detailed which allows for modification before final design of the project. The reason for applying this approach was unconcern of the outcomes of literature review, that some ideas might be too hard or complicated to implement or required commercial software solutions. Estimated time for this one was 20 hours, and was performed in time.

Subsequence phase was dedicated for literature review and specifying the tools that would be used in development process. This level of the model highly relies on previous stage

which determined objectives around which research should be taken. Twenty hours was reserved for this phase and was sufficient; although during implementation stage additional research was made.

Next stage was dedicated for designing the draft of application. Outcomes from two previous stages were groundwork for that. Most of preconceived ideas were maintained. All ideas had foundations, and draft/design outcome was feasible in terms of implementation. The conclusion of it were diagrams which lay background for creating databases, and menu of application alongside data flow across whole application. Moreover, it provided a plan of developing application (the order of creating each part). Estimated 10 hours was enough to deal with this task.

The most time-consuming part was implementation, the estimated 100 hours time limit was adequate. Almost each task scheduled in this stage was outsourced by previously done research

The very last stage was destined for testing, evaluation, drawing the conclusions and writing final report. Summarised time for all mentioned activities amounted to 40 hours.

3. Analysis of problem

This chapter covers analysis of existing ticketing systems that are administered by one of the biggest organisations in entertainment's business. Most attention was focused on usability of implemented solutions, navigation throughout web sites, potential data flow, shopping cart management, and process of buying tickets (also visualisation of the audience).

Mariinsky Theatre (Saint Petersburg) [17], The Bolshoi Theatre of Russia (Moscow) [18], New York City Opera [19], and Royal Opera House (London) [20] were investigated in order to recognise the problem. All of them are worldwide known institutions, and establish trends in entertainment's discipline. All of them enable users to browse the schedule of events, but also book tickets for them. Thereby in general characteristic they do not differ much from online shops.

First sub-chapter outlines research that was made, next drafts an ideal web site based on outcomes from previous stage, by combining good approaches noticed in investigated facilities.

3.1. Investigated web applications

Mariinsky Theatre's web site is very well organised. It is overfilled by classic spirit, which was achieved by certain design, and good colour combination. From first hit user is informed about upcoming events or can choose a date from the calendar. The interface is very easy to navigate, intuitive – user does not need any hint to understand the flow. Playbill is scheduled only two months in advance, but it probably depends on the policy of the company. Results of searching can be filtered by stage or type of performance. Superbly well developed system of selecting seats, comprehensive and intuitive. As was discovered in further research, it was created by support of JavaScript. Orders are organised in a basket. Moreover, web site ships information about changes that had been made lately. Pitfalls of this application are: poor quality of printed documents (they are not formatted into PDF), registration required to proceed with purchasing the tickets (probably depends on policy of the company). Overall the web site is very well organised in order of usability and fully meets requirements for this sort of application.

The Bolshoi Theatre of Russia is another example of a web site, quite well developed for browsing and purchasing tickets for events. Very stylish layout is a hallmark of this application. Unfortunately, there are few drawbacks to it, the information is not presented in a transparent way, there are too many sections on main page. What's more, the calendar redirects user to the schedule for the whole month which is obscure. Same approach was adopted as in Mariinsky Theatre – user firstly has to register, afterwards they are able to choose a seat for particular entertainment through graphic representation of the audience. Orders are organised in a basket. The drawbacks of this application are: web site is not fully translated into English (section marked as English), tickets are not printable (this aspect highly depends on company's policy), printable version of

entertainments are not formatted into PDF. To sum up, application was well-organised but not as good as previously analysed example.

In contrast to well developed Russian web sites, New York Opera's presents itself poorly. The design of layout is strange, contrastive colours were used. The interface is not easy to follow. It is allowed to enter booking system for past events. User is not able to chose which seat will be booked, can only set up one of few options (e. g. "best available", by price). Orders are managed by shopping cart, although company charge extra for purchasing through the Internet. Html is formatted incorrectly. Poor quality and non user-friendly interface might discourage potential customers from purchasing at this web site.

The last investigated institution was Royal House Opera's web site. It is well designed application in term of data-flow and transparency, lacks only few details. Menu is located on the bottom of the web-site. It is highly discouraging approach, and calendar is marked as a link, not the actual control (although it might not fit well to the overall design of the web site, and the pattern used later would not be suitable for the first page).

3.2. Draft of application

Desired application should combine all features pointed out as valuable in previously investigated examples, joined to cooperate with each other through clear and easy to navigate interface. Web site should be highly readable, not overfilled by information, and in well-stylised design. Well organised databases working on the server side would be crucial to keep the code transparent.

Main page should welcome customer with information about upcoming events, non-confusingly present navigation menu (should be interactive, not a group of passive links) and calender. Number of displayed events should be limited to increase readability, especially when agent's web site is outsourced to more then one Web Service. Selecting a day on the calender must immediately redirect to the schedule for chosen day and mark that day. Colouring of the application's interface should be interesting for the user, and focusing his attention. It can be archived be using light versions of colours. Fonts which are not used in daily newspapers (but still highly readable) are advised. Modern design often well match when elements are trimmed to indicate boundaries.

Navigation menu should encapsulate links to pages which contain:

- Schedule for a month – automatically generated set of events (which would be acquired from Web Services) despite the amount of date.
- Scheduled entertainments by particular organisation – same as point above, though information are gathered only from one Web Service.
- Shopping cart – displays already chosen elements, optionally can be removed. Since that sort of performance attract international audience recalculation of the total price might be welcome.
- About page.
- Navigation back to home page.

Moreover, there has to exist a page for particular entertainment, but it's content has to be generated by algorithm, creating each page for different event would be too costly and time-consuming.

4. Literature review and specifying tools

This chapter covers research that has been taken in order to gather information about technical capacities, implementation issues and software that were relevant to objectives specified earlier. Presented materials are structured, by introducing the theoretical image of functionality in the context of entire application, and after, research outcomes in terms of implementation, approaches (few times occurred a situation where a few different techniques to develop same feature of the system were conceived).

It is first necessary to consider in which technology Web Services would be created. Since they only interact by corresponding messages, which are standardised, it does not matter to the rest of the application, in which programming language would they be written. Because of solid background in C#, personal interests and superbly developed Visual Studio, this platform has been chosen as language of creation of the Web Services. The other eventualities were Python (basic experience), and Java (average). Moreover, the preconceived idea of the structure included implementing database within Web Service. For fully understanding of distributed applications, their structures and purposes materials [1] from 320CT Distributed Applications Development module were used, as well as Chapter 31, "Working with Services" from "Professional ASP.NET 4 in C# and VB 2010" [2]. Organising work within Web Service application is done in the same way as all other projects with one exception, all methods reachable by external requester have to be followed by [Web Method] statement. Additional information of how to create and consume Web Service has been found on tutorial at YouTube [3].

Very important advantage of C# platform is Microsoft Visual Studio as programming environment which fully supports creating Web Service applications (ASP.NET) and sharply improves development process in comparison to other, like NetBeans, which is not as well expanded.

For drawing use case diagrams UML Studio has been used because of it's easiness of managing.

Next challenge was to investigate the way Web Services can communicate with agent's web site. Context of this project required synchronous correspondence. Fortunately, default implemented solutions in ASP.NET platform are all synchronous [2]. Approaches of how to consume them are included both, in previously mentioned book [2] and on-line tutorials [3]. Because of enormously simplicity of syntax over jQuery scripts that were advised firstly, default ASP.NET solutions was used in this project. Moreover, they rely on SOAP messages, which are widely explained in 320CT module materials [1].

Subsequent stage of investigation was understanding of databases that are hosted on Microsoft SQL Server 2008 (already included in Visual Studio 2010). Most research in this area was done by exploring MSDN Visual Studio 2005 Developer Centre web site [4]. Though it is concerned with version 2005 it can still be easily transposed into 2008

version. Difference that were introduced occurred with connecting to the database. It was concerned with creating LINQ to SQL class (dbml file) which mapped relational objects. Based on it and by exploring ConfigurationManager class data context was created, and querying could be done in LINQ. Moreover, since project would be tested on localhost, this approach does not require changing connection string in Web.config whenever the location of the project change. This technique of implementing database is supported in Web Service projects with no exceptions.

Querying database was obvious forward step. Choice had to be made between two widely used techniques: SQL and LINQ. Reason to support LINQ over SQL (in my judgement) are:

- LINQ is able to recognise the type of data that it is manipulating with. It cuts down debugging time. On the opposite side, SQL plays with strings, and only during execution of queries raises exceptions and errors. Objects mapped earlier in dbml file benefits that Visual Studio is supporting LINQ queries by hints, which sharply accelerate development process.
- LINQ enables writing queries by using lambda expressions (same as mathematical approach) – which are preferred by large community of developers, because of their simplicity (looks like working with objects).
- LINQ is integrated with C#, thereby eliminating the impedance mismatch between programming languages and databases. [5]
- In context of the whole project, LINQ can be applied to any type of queries for enumerable objects, therefore better suited to parsing XML file

Most of the information on how to play with LINQ has been gathered from “Pro LINQ” [6], especially chapters 3 - 5, 7, 10, 11 and on-line tutorial [7].

Organising web site raised few aspects as well. The major challenges were: master page, passing data between pages, managing interactive menu, and dynamic displaying of information, depending on the content of database. All those techniques put together complete main menu, flow of data within application, organise shopping cart and presents data in an interactive and intuitive way.

Passing data between pages highly depends on how sensitive they are. Less important can be transferred by query string in URL (most widely used with data which does not interfere with buying process). In case of making orders it is a good approach to hide them from user to avoid any sort of manipulations (purposeful or not). To do that ASP.NET support Session object which refers to the session created by browser, and can pass data as long as the browser is uninterruptedly running, though it also supports time limitation. Same functionality can be provided by 2 classes. One represents the item (for instance entertainment data) and second list of items with information about current priority of searching and other common to the process if needed. However there is no need to reinvent the wheel since it provides same functionalities as Session object, and requires much more work to do. Additional research in order to gather more information about managing Session object was made, most helpful turned out to be [2], chapters 11 and 21 and MSDN web site [8] and [9].

Menu can be designed in many ways. Priority of this task is to make it clear to navigate and to have a nicely looking set of controls. They should focus the attention of the users. Transparency is a necessity as well. Because of all mentioned reasons applying master page looked like a natural consequence. In order to gain solidified knowledge in this area online tutorials [10], [11] turned out to be very supportive, with various examples of master pages which they contained. Master page is able to handle more complex interaction between controls, and code which works on a server side, to redirect in correspondence to the calendar control.

Navigation menu should be consolidated with master page to follow through the whole application. Two approaches were investigated, ASP.NET controls and jQuery scripts [12], [13]. The advantages of the first were: simple implementation (drag and drop technique in Visual Studio) and high level of integrity with ASP. Second approach was characterised by far more complex syntax but better outlook.

One of the very last condition that needs to be investigated was a method for displaying data in a dynamic way, insensitive of amount of data that had to be processed. Since one of the first releases .NET framework encourages to use GridView. After analysis of usability (the biggest advantage was automatic populating control from data source based on names of bounded data fields) and effortless of associating event "on click button" with proper method. Fundamentals understanding for this approach was provided by book Pro ASP.NET 3.5 in C# 2008, chapter 10 [14]. jQuery community also share few templates for presenting group of data, unless they required big amount of modification in order to facilitate it into the project's scenario.

The way Microsoft designed GridView was not sufficient to use in visualisation of the audience. ASP.NET supports coding the table from CS file which works on server side and in runtime displaying the results on ASPX page. Thereby it can be populated with appropriate data on the fly (within Load_page method), and designed the way it imitates the view of the audience from the top. The outcome of this draft would not be as sophisticated and smooth as graphic interfaces implemented by big stages, mostly in javascript or flash animation technology.

Recalculation of the value of the order can be handled in many ways, which are concerned with different approaches of gathering data. The two easiest ways were: use one of the existing Web Services, or download the data from trusted source. Second idea is more safe. European Central Bank publishes daily rates of currencies in valid XML file. .NET framework richly supports retrieving XML files by external classes and LINQ to XML. Querying (by LINQ) data stored as XML document is done the same way as querying database that was previously investigated.

Very good approach, which can save lot of time during development process, is using cascade style sheet (CSS), which define set of styles that can be assigned to particular entity of the application.

Generating PDF files was proposed to be handled by iTextSharp library, which is open source, designed for C# platform. Significant impact in order to gather understanding of it had two blogs: Mikesdotnetting [15] and Massoud Mazar [16], which explained basics of populating the documents by various of forms, and handling with header and footer in fully automatic way.

5. Design

Current section describes the final draft of application. It was done by drawing diagrams which presents functionality of menu, and more, data flow across the entire web site and common Web Services. Each sub-project contains diagram which depicts set of features (although all 3 Web Services resemble each other, because of similar tasks that they perform) which constituted as a backbone for the draft. Each detailed feature contained within diagrams is described later in order to verify technique that is the most adequate to implement it.

Last sub-section schedules order in which tasks should be performed. It is good approach to sequence them, because of early recognition of any incompatibles that can be released through anterior tests after very first assemblies.

5.1. Overall design of application's structure

According to the project planning stage, and afterwards realising capabilities of .NET framework in implementation of use case diagram has been sketched. The aim of it was to illustrate the functionality of menu, emphasise the most important issue, and determine flow of the data across the whole system. Certain entities represent particular pages and relations “extend” or “include” distinctly differentiate direction where data has to be processed as well as classify the sensitiveness of the information that is passing by. Figure 1. presents high level use case diagram for entire application.

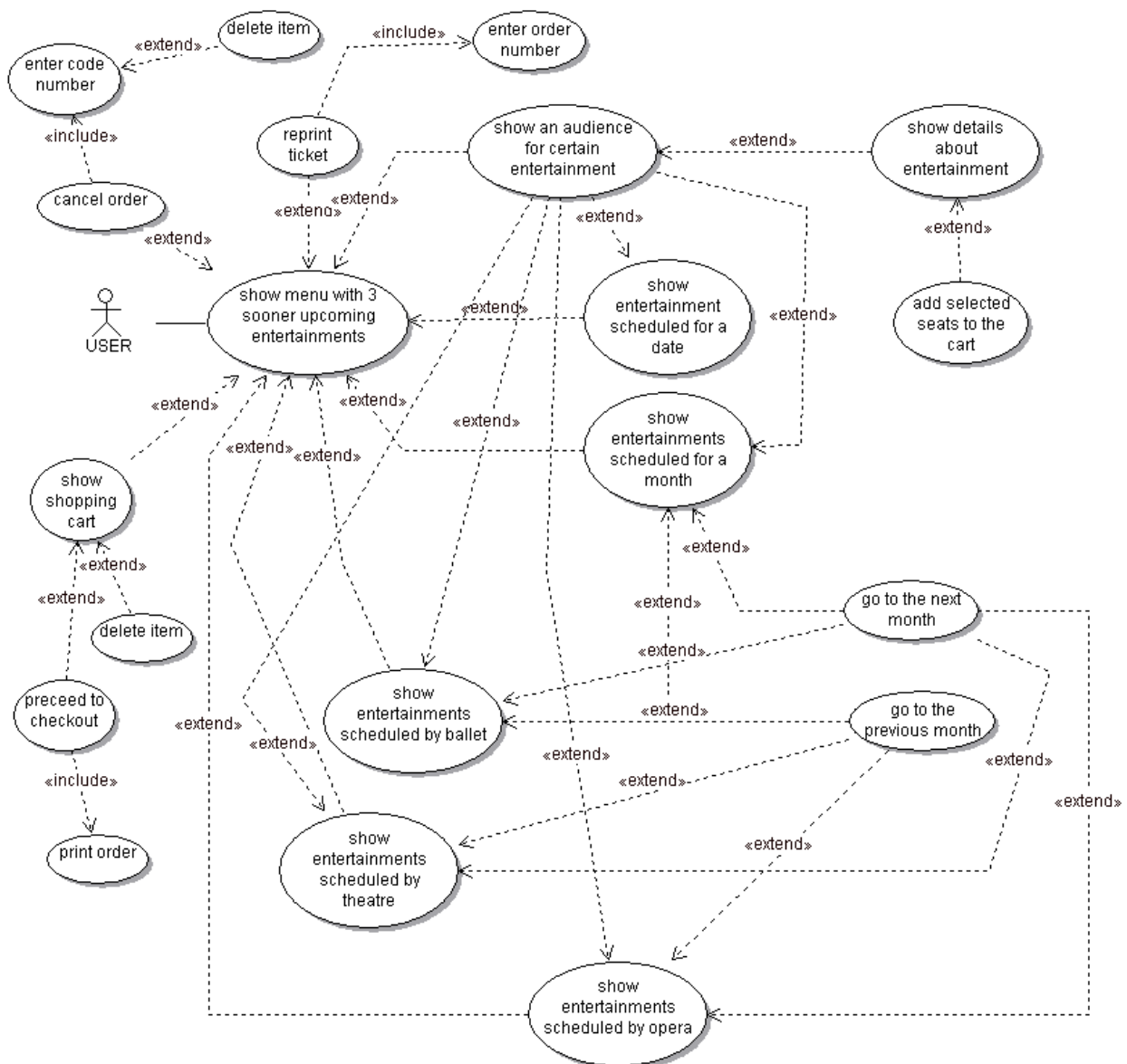


Figure 1. High level use case diagram for entire application.

As it is shown, user has right to choose one from couple of options, after displaying home page which informs him also about 3 sooner upcoming events that has been scheduled in connected to agent's web site organisations. This explains "select entertainment" entity shown on the diagram, which enables explicit redirect to page where seats for entertainment are selected, and get view on details about chosen performance.

"Show entertainments scheduled for a date" is navigated through selecting a date on a calendar. After that automatic redirection takes place. A page pops up with a list of all events scheduled for selected day from all facilities associated with the application. User can select one of the listed shows and go to the page which visualises audience. User

faces multiple choice, of selecting seats. Afterwards they can add selected items to the shopping cart or abandon it.

“Show entertainments scheduled for a month” presents aggregated repertoire from every associated organisation on one page. Shows are limited by date, only those from currently investigated month are showed.

“Show scheduled by” is responsible for shipping information for particular month (current is default). User can select one, and advance to seat selection stage, or swap one month ahead or backward. This option it tripled, for each associated Web Service source.

“Show shopping cart” displays added items. User can either delete few or proceeded to checkout, which is followed by procedure of fulfilling the form with payment details. At the end of the process customer receives PDF file with unique order number (not generated based on session ID), and itemised seats that have been booked.

To cancel part of, or whole order user has to follow through pages, firstly has to enter valid order code, then a generated list of items that belongs to specified order would show up. User can delete as much as possible (except those, which are to close to premiere). After selection finishing updating the cart.

“Reprint the ticket” provides reprinting the ticket based on order’s code.

5.2. Design of order's database

All orders that have been made are stored in a database which is managed by main application. Main goal is to track orders, but also to provide reference to certain orders to cancel some or reprint the ticket, because in one time customer may purchase tickets from different facilities. Calling Web Service each time would be too time-consuming. Since application does not require registration of the customers, their names are each time assigned to the particular order. Figure 2. presents entity relationship diagram which models database.

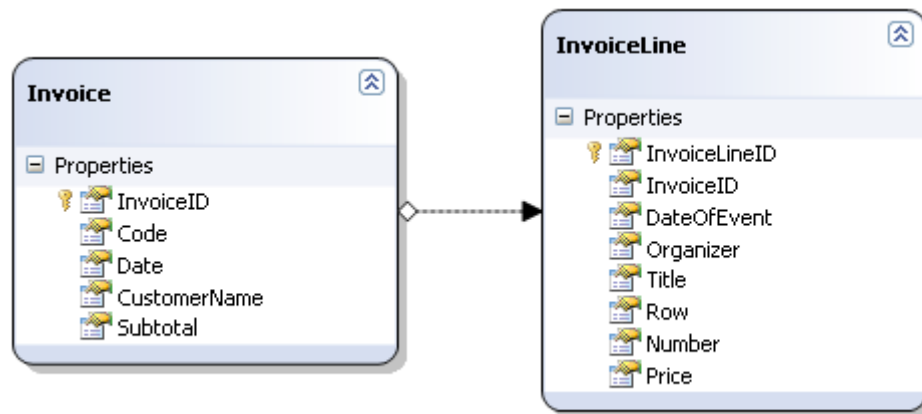


Figure 2. Entity relationship diagram.

Rows in the tables has been named according to the function or information assigned to not confuse. Therefore “CustomerName” contains name of the customer that particular order is assigned to.

5.3. Design of Web Services

Since all Web Services are responsible for similar activities they were designed in a same way with slight differences between each other. Dissimilarities depended on character of performance that were represented by each Web Service, but they only occur in description of the entertainment. Core of database, which is structure of entities as well as structure of tables “Seats” and “Entertainment” were unchanged. Figures 3, 4, and 5 shows entity relationship diagrams for 3 associated Web Services.



Figure 3. Entity relationship diagram for ballet Web Service.

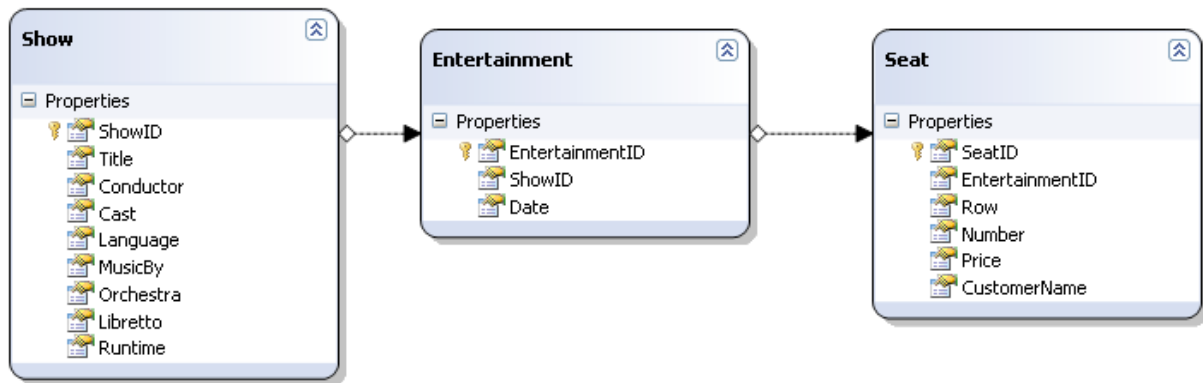


Figure 4. Entity relationship diagram for opera Web Service.



Figure 5. Entity relationship diagram for theatre Web Service.

Rows in the tables were named accordingly to their function within Web Service. If seat is open is populated by null value, and similarly during cancellation process - customer's name is substituted by null.

5.4. Schedule of implementation

Approach of implementing satellite facilities first was adopted. Firstly, Web Services had to be created, and classes that are not the main core of application, but support primary functionalities. The reason for applying this strategy was recognition of any failures on early stages and reacting by immediate fixing them. It was important to correctly create parts which were outsourcing main activities.

In the middle of development process all pages and navigation between them was carried out, including schema of transporting data in Session object, and appropriate points to control content and cleaner to provide appropriate tracking and display data correctly even when user is switching controls absurdly.

Last stage was oriented on consuming data shipped by Web Services and properly displaying on proper pages. Moreover, they become interactive with their equivalent in database stored on application side. Process of printing PDF file relies on content of database.

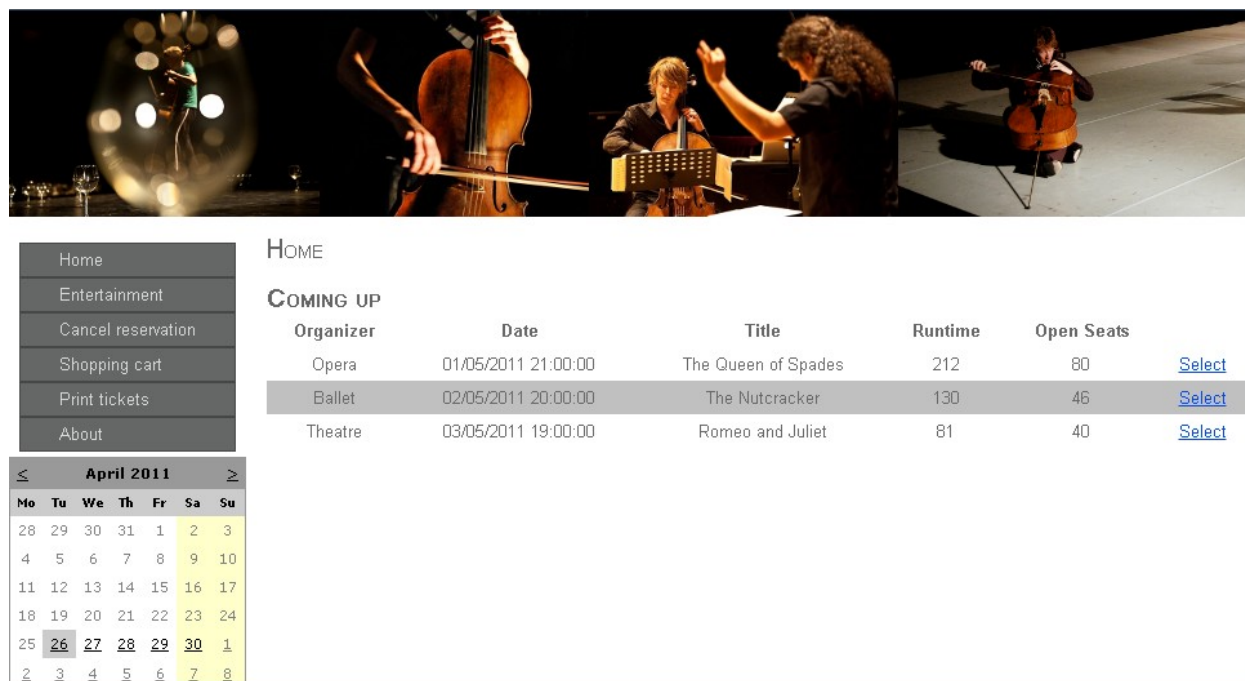
6. Implementation

During the process of implementation no major problems occurred, but few times unravel cases which called for slight modification of drafts. This chapter covers brief revision of aspects mentioned by preceding chapter, and complements by explanation of approaches that does not attract at first but are significant to the flow and are crucial to achieve high level of customer's satisfaction (those which perform behind the scene actions and make things work). To emphasise the efforts, last stage of this chapter is accompanied by screen shot which proofs implementation of designed earlier drafts.

In nearly all cases research outcomes pave the way to how application was implemented. Thereby there is no need to duplicate same patters that were adduced in the "Literature review" chapter.

6.1. Main application

Figure 6. presents welcome page, on the left side navigation menu is located, below calender. Main content of the page are 3 upcoming events. They are generated automatically. In each Web Services a method has been implemented that informs about 3 earliest events since date that is passed as an parameter of this method. System temporary aggregates results from all associated Web Services, extracting 3 earliest and populate a GridView control with them, from another temporary source - DataTable.



The screenshot shows the home page of a ticketing system. At the top is a banner image of a musical performance. Below it, on the left, is a vertical navigation menu with links: Home, Entertainment, Cancel reservation, Shopping cart, Print tickets, and About. To the right of the menu is a calendar for April 2011, with the 26th, 27th, 28th, 29th, and 30th highlighted. Further right is a section titled 'COMING UP' which contains a table of upcoming events.

Organizer	Date	Title	Runtime	Open Seats	
Opera	01/05/2011 21:00:00	The Queen of Spades	212	80	Select
Ballet	02/05/2011 20:00:00	The Nutcracker	130	46	Select
Theatre	03/05/2011 19:00:00	Romeo and Juliet	81	40	Select

Figure 6. Welcome view on the home page.

Displaying events dependent on their date or location is done in very similar way as upcoming section; request for certain month is sent, and the results are populated into `DataTable` objects, which outsourcing `GridView` objects. Each row includes "Select" button. Most importantly, each button has to refer to the row from where it was called. To do that `GridView` objects have specified `OnRowCommand` function, which catch `GridViewCommandEventArgs` object, from which through `CommandArgument.ToString()` method string object is generated, which is later converted into int. This technique enables the app to get row from which button was called. After that data is found in same row they are used to specify `Session` attributes and redirect to `Entertainment.aspx`. Listing of mentioned method is enclosed in Listing 1.

Listing 1. Example of RowCommand method.

```
protected void GvEntBallet_RowCommand(object sender, GridViewCommandEventArgs e)
{
    if (e.CommandName.Equals("Select"))
    {
        int rowNumber = Convert.ToInt32(e.CommandArgument.ToString());
        Session["date"] = GvEntBallet.Rows[rowNumber].Cells[1].Text;
        Session["stage"] = "Ballet";
        Session["title"] = GvEntBallet.Rows[rowNumber].Cells[0].Text;
        Response.Redirect("Entertainment.aspx");
    }
}
```

From implementation point of view process of managing audience (throughout the entire browser's lifecycle of application) is very interesting. System for communication between pages is based on `Session` object, expiration time of which was set for 30 minutes. Figure 7. presents view on the audience.

The stage picture was originally downloaded form [22].



Home

Entertainment

Cancel reservation

Shopping cart

Print tickets

About

STATGE FOR THE NUTCRACKER AT MAY 02, 2011 08:00

[Details about entertainment](#)



Price: £55	Price: £55	Price: £55	Price: £55	Price: £55	Price: £55		Price: £55	
Select	Select	Select	Select	Select	Select		Select	
Price: £55		Price: £55		Price: £55	Price: £55	Price: £55	Price: £55	Price: £55
Select		Select		Select	Select	Select	Select	Select
	Price: £55	Price: £55	Price: £55					Price: £55

Figure 7. View on the audience.

Selecting or reselecting the spots only the background of them changes. Add to the cart button render all cells of the table that visualise the audience, and inserts into the cart. In case when customer reenter same entertainment all previously checked (added to the cart) seats have to be marked as taken (Figure 8., marked by blank yellow colour). This happens by verification of contents of shopping case and is handled at first stage, when table is generated.

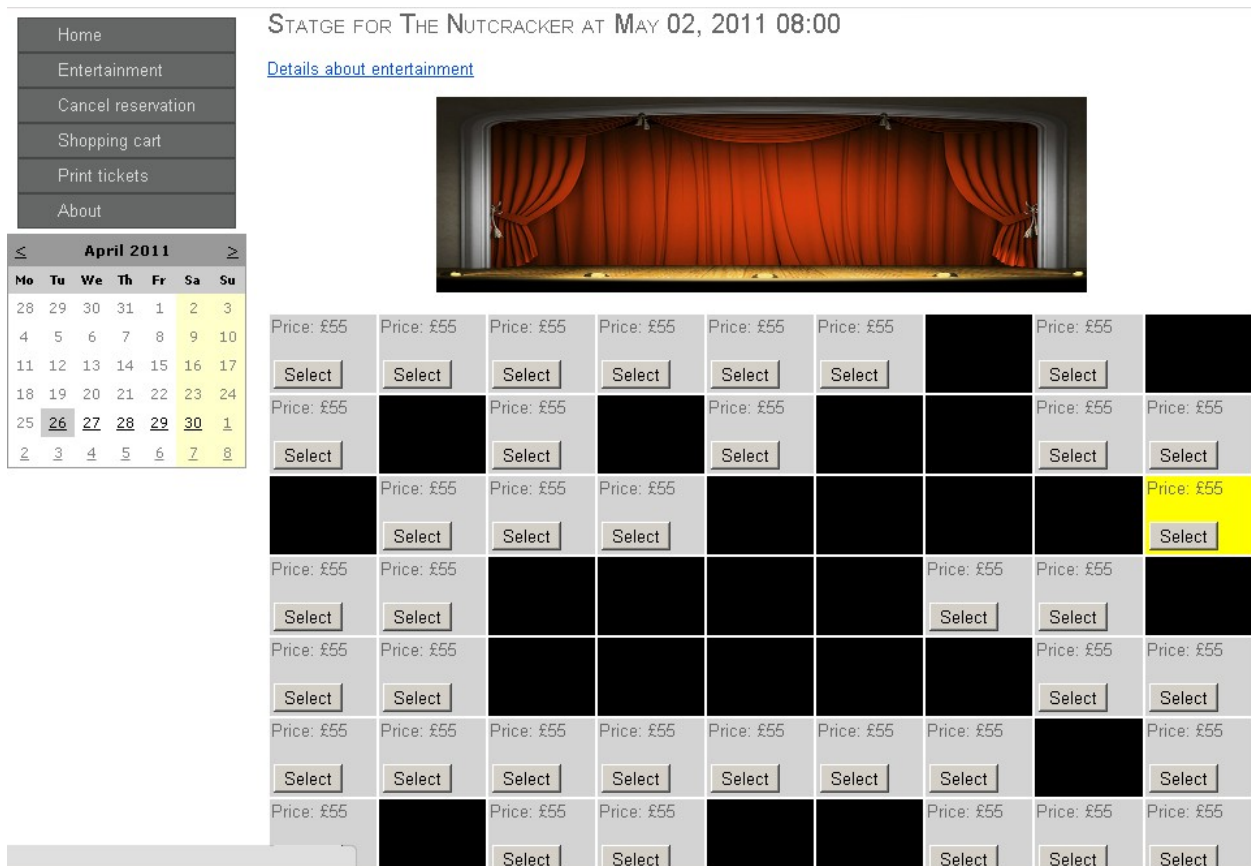


Figure 8. Visualisation of an audience

Visualisation of the audience keeps more secrets. All the buttons that are appearing were created by single line (in a loop), and each are handled by one function. To make that work method has to recognise the ID of the button (which was assigned to every key by incrementing a variable). Listing 2 shows how ID is assigned to the button manually and later data from sender object are extracted.

Listing 2. Creation and serving button.

```
BtSelectT.ID = "BtSelectT_" + row + "-" + number;
BtSelectT.Click += new EventHandler(BtSelectT_Click);
...
protected void BtSelectT_Click(object sender, EventArgs e)
{
    int row = Convert.ToInt32(((Button)sender).ID.Substring(10, 1)) - 1;
    int number = Convert.ToInt32(((Button)sender).ID.Substring(12, 1)) - 1;
    ...
}
```

Data added to the shopping cart are stored in the Session object throughout application's lifecycle. Despite the fact that session transport only plane text, it is easy to pass date even in different format, because during reading process data can be converted to the original format by simple casting. Code responsible for that is enlised in Listing 3.

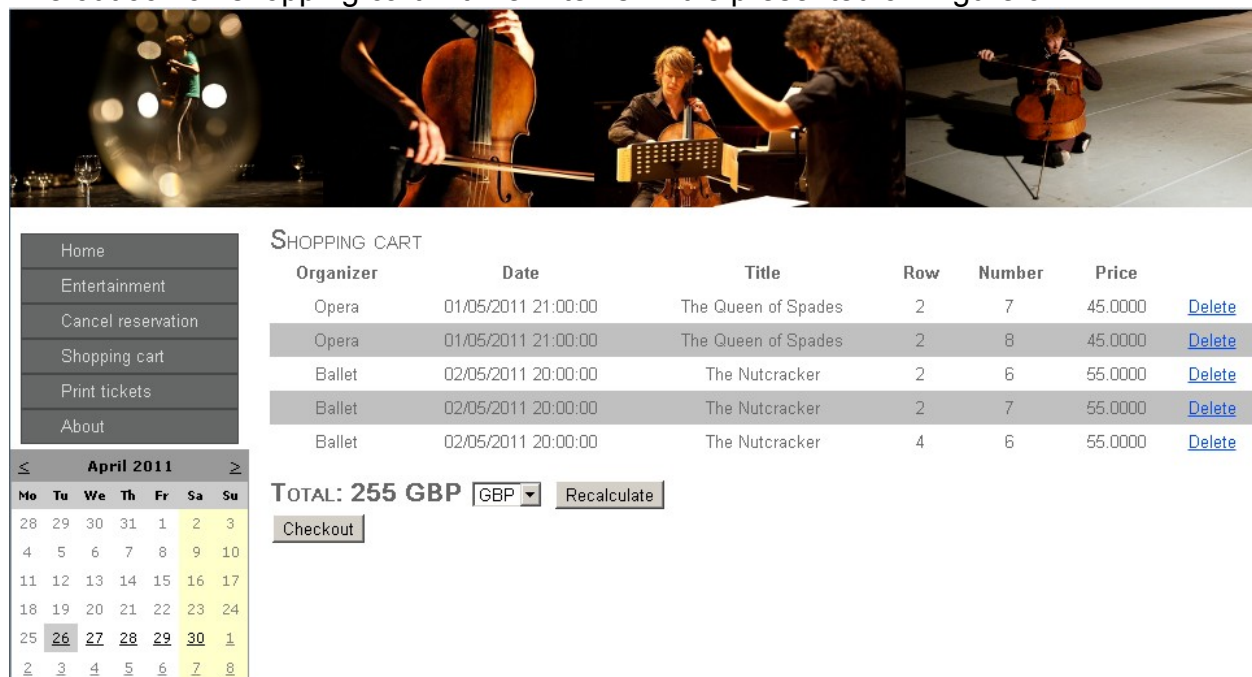
Listing 3. Passing shopping cart in Session object.

```

DataTable DtShoppingCart = new DataTable();
Session["ShoppingCart"] = DtShoppingCart;
...
DataTable DtShoppingCart = (DataTable)Session["ShoppingCart"];

```

The outlook on shopping cart with few items in it is presented on Figure 9.



The screenshot displays a web interface for a ticketing system. At the top, there is a navigation menu with links: Home, Entertainment, Cancel reservation, Shopping cart, Print tickets, and About. Below the menu is a calendar for April 2011, showing the days of the week and the dates. The main content area is titled 'SHOPPING CART' and contains a table with the following columns: Organizer, Date, Title, Row, Number, Price, and a Delete link. The table lists five items: two Opera tickets for 'The Queen of Spades' and three Ballet tickets for 'The Nutcracker'. Below the table, there is a summary section showing 'TOTAL: 255 GBP' with a currency dropdown set to 'GBP' and a 'Recalculate' button. A 'Checkout' button is also present.

Organizer	Date	Title	Row	Number	Price	Delete
Opera	01/05/2011 21:00:00	The Queen of Spades	2	7	45.0000	Delete
Opera	01/05/2011 21:00:00	The Queen of Spades	2	8	45.0000	Delete
Ballet	02/05/2011 20:00:00	The Nutcracker	2	6	55.0000	Delete
Ballet	02/05/2011 20:00:00	The Nutcracker	2	7	55.0000	Delete
Ballet	02/05/2011 20:00:00	The Nutcracker	4	6	55.0000	Delete

TOTAL: 255 GBP GBP Recalculate

Checkout

Figure 9. Outlook on shopping cart.

6.2. PDF printer

Out of the main core of application, in classes responsible for printing PDF files a non-trivial approach was adopted. It was concerned with handling header and footer, on each page of the document. The structure of the paper created by iTextSharp library is that firstly everything is adding to the one object (orders affect final effect), which is later transformed into actual PDF file. To cope with that HeaderFooter class was created, which inherits from PdfPageEventHelper class and override OnOpenDocument, OnStartPage, OnEndPage, and OnCloseDocument methods to add information on the top and bottom side of the document, and moreover, trace the number of pages which are included into footer.

Example of how sample ticket look like shows Figure 10.

Ticketing system

Orderer name: Richard J Right

Orders code: KLVDEQIYPH

Date: April 26, 2011 10:23:57

Organizer	Date	Title	Row	Number
Opera	May 01, 2011 21:00	The Queen of Spades	2	7
Opera	May 01, 2011 21:00	The Queen of Spades	2	8
Ballet	May 02, 2011 20:00	The Nutcracker	2	6
Ballet	May 02, 2011 20:00	The Nutcracker	2	7
Ballet	May 02, 2011 20:00	The Nutcracker	4	6

Figure 10. Sample of ticket.

6.3. Web Services

Each Web Service was created in accordance to the drafts. To smooth the way on few tasks additionally (it touches every Web Service, since they all share the same logic) method to OpenSeats which returns the number of seats which are still available for particular entertainment. This method is not preceded by [Web Method] statement, therefore could be declared as private, only to use in local namespace of Web Service. Figure 11. presents set of methods available from Opera Web Service for external requester.

Service1

The following operations are supported. For a formal definition, please review the [Service Description](#).

- [BookSeat](#)
- [CancelReservation](#)
- [Get3EarliestEntertainment](#)
- [GetScheduleForAMonth](#)
- [GetScheduleForDate](#)
- [GetSeats](#)
- [GetShowDetails](#)

Figure 11. Opera Web Service, interface.

7. Testing and results

When implementation stage was over the tests have begun. The purpose of this was to revise if application fully meets the requirements that were established at the beginning. The system was tested in order to verify if it performs all activities correctly, then subjective appraisal of how much satisfying investigated features are from hypothetical customer's point of view.

Tests were made on Windows XP operating system, by using Chrome, Firefox and Internet Explorer web browsers.

All pages are rendered and displayed correctly, in order to set up the controls on the page as well as fidelity of colours interpretation. Data between pages were transferred properly, which was reckon based on appropriate communicates and actions performed by each page. Dynamic menu was working fine on each browser, same as GridView and tables which were generated automatically based on data sources. In different browsers buttons look a bit different but it depends on the software and ASP.NET has nothing to do with it.

Through the navigation menu set up on master page, which is related with every page. Simulated behaviour of typical customer does not certify any unwelcome circumstances. Also simulation of irrational user proceed successfully, no critical errors occurred.

Customers would be satisfied based on functionalities of presented application, although demanding agent's company would ask for more sophisticated graphical interface.

8. Project management

This chapter provides explanations of how this project has been handled, according to the plan that was drafted at the very beginning. It takes into account risks and quality of management.

8.1. Project schedule

Figure 12. shows Gantt chart which reflect my schedule of work that was applied at first stage.

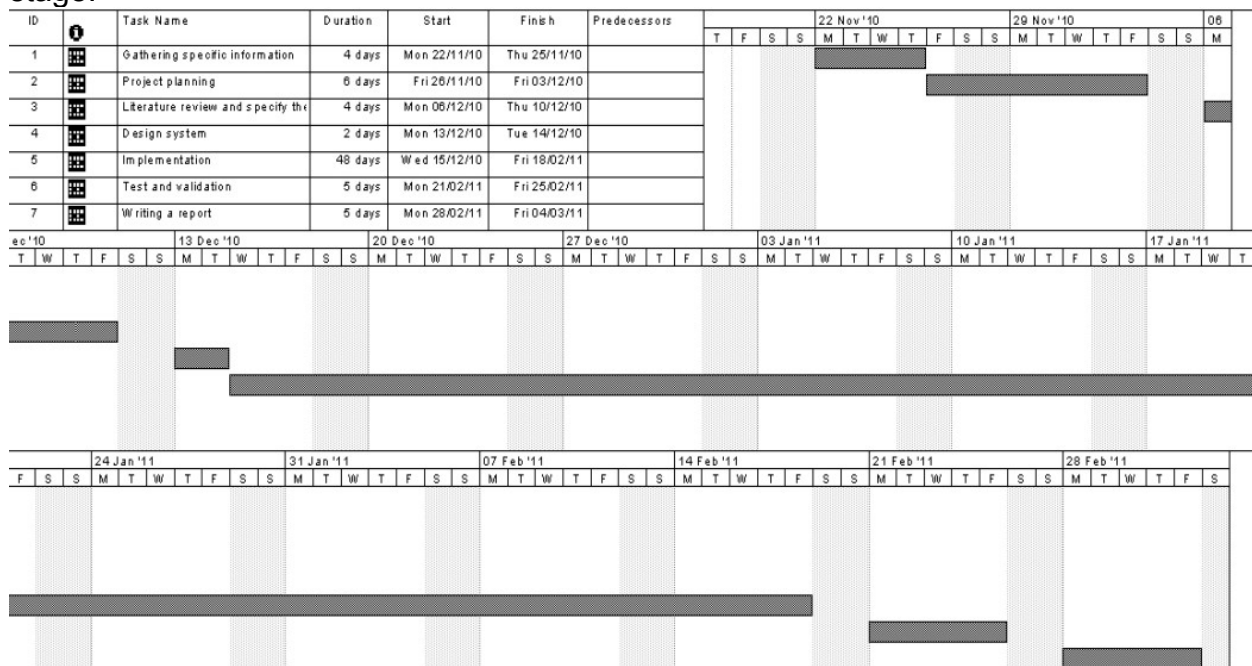


Figure 12. Gantt chart

Throughout the development process I was accomplishing tasks in time, only writing report exceeded the planned period, since there was no rush about it. Some early stages were started a little earlier than it is showed on figure above. The literature review has been done mostly in time that was established for this task, however implementation stage occurred few ambiguity which called for extra research. Outcomes of that has been included into research chapter.

8.2. Risk management

Challenge that has caused the biggest risk was concerned with technology and my ability to implement all aspects of application, mostly those which are widely used in commercial applications, because best solutions are licensed, like managing of the data flow in order to track orders from the moment they were added to the cart up to checkout. Fortunately, I coped with that, moreover, data are not reachable through this process to the user, so any unplanned modification cannot occur.

Few minor circumstances require more time than I was expecting, but mostly it yields results as much shorter code than my predictions were.

I did not experience any physical pain because of long time spent in front of the computer or books.

8.3. *Quality management*

In order to track the progress after each phase of development several approaches were adopted (only exceptions are implementation and tests, they required different attitude). After completing every stage (except implementation and tests) I was preparing notes with essential outcomes that affect further stage, another for learning outcomes, what I have learned because of that, and how already done achievements may be developed to enrich the application in many fields: transparency of code, reliability and improvements of work process.

Implementation and tests stage were played with different rules. Coding of entire application was divided into series of smaller tasks, after each was done, small tests validate if created part behaviour appropriate and meet requirements. At the end of implementation stage general tests were performed to imitate behaviour of typical (but also unusual) customers in order to find any errors and simultaneously encounter new ideas.

Throughout entire implementation stage widely used in programmers community standard of coding had been at play. It increases readability of the code, especially when is accompanied by numerous of comments.

Official meetings with my supervisor were not mandatory to track the progress of work, since we have seen each other during lectures and laboratory sessions (I was attempting to 2 modules that were taught by her) and all minor uncertainties were handled before or after classes. Nevertheless, we had few meetings and as a result of long discussions during them I have received feedbacks and further work was mapped due to objectives that were ahead.

9. Critical appraisal

This section is dedicated to dispassionate analysis of the work and its outcomes. Demonstration of the way the project affected my skills and helped develop them.

At the very beginning of creation of this project the whole work process was scheduled. Waterfall model was adopted. The aim of that was to estimate time that was needed, specify flow of information between stages to organise it in order to increase productivity and pleasure of working. To every task was assigned estimated time and deadline until when each stage should be completed, to track if the process does not stuck on some point.

Personally I prefer to work with more flexible model, like an interactive and incremental development approach. It enables to introduce changes in drafts prepared at very beginning and response for clients demands that may change during development process.

Furthermore analysis of the problem was performed. This stage allowed me to gain understanding in customers' needs in order to get comprehensive information about entertainments and go through non-complex decision process where users are fully supported by organisation and feel secure. Moreover to see, how important gathered information is with reference to the widen picture of whole organisation that is about to be created based on them. The correct recognition on this level is crucial. The outcomes of it affected design stage and all further work process.

Throughout analysis stage I got understanding of the biggest challenges that nowadays web applications face, in order to manage relationship with customers, communicate with them, and advance the buying process during application lifecycle.

Analysis part was very exciting but also, there occurred a few pitfalls. First refers to requirements, in real business those would be clearly specified by organisation, what are they looking for, emphasised they critical aims. Also in the middle of development process they might support by accepting drafts or suggesting different approaches, and assume previews. Interaction with the client would make the application flourish, make the process more vital and put much bigger pressure on the deadlines.

As it was mentioned above, analysis part was then projected onto the final design. Because of the adopted model of managing the work for this project, once established drafts should not be modified later on. Only slight changes might occur, which not affect the entire structure of application. This encourage to prepare the most reliable diagrams and forecast satisfaction and correctness of designed solutions.

The technology that was chosen to implement drafts affect my skills as well. I became more familiar with commercial aspects of ASP.NET, realised which approaches are acceptable in order to sensitiveness of the data that they manage, and gather knowledge of presenting data automatically careless about their numbers, as well as referring to automatically generated controls which act individually according to their unique ID and different methods.

The process of implementation was much simplified by using the same convention of naming the objects in entire application as well as comments, which increased its readability and transparency. That was very effective method for debugging.

The last stage, report writing, deepened my understanding in clear expressing myself, deductive argumentation, and proper document formatting, which follows international standard.

10. Conclusions

This section presents achievements of this project and recommendation for the future work. Project was developed in account of few directions, and had to met requirements on different field.

10.1. Achievements

The biggest achievement of this project is a working application that fully meet objectives established at the beginning and report which documents process of creation alongside research area that increased my understanding of this sort of application in order to implement and business purposes of it.

In details, application manages browsing and buying tickets process, and supports decision making process by suggesting upcoming events. Implemented system is very flexible and reliable in terms of cancellation orders, or modifying them. Customers are moving throughout clear and intuitive menu that needs no explanation. Every time user is explicitly informed where he currently is, and in which process are they involved. Even irrational actions do not break the system.

During browsing the content of repertoires user faces automatically generated sets of data, that perform properly, both presenting data about events as well as audience of certain performance.

System copes with automatic printing of PDF documents which imitate tickets (contain information about purchased seats, dates, events etc.).

Report documents work that has been done through development process, explain reasons of adopting particular approaches, how the information was researched, system implemented, and provides a comprehensible summation.

10.2. Recommendations for the future work

Although the project was completed new improvements are very welcome. The challenge on the horizon would be implementing system of hints that will suggest user events, based on already viewed ones, and adding to the cart different events that customer might be interested in booking. This sort of solutions are very time-consuming. Nowadays patterns are based on connection graphs, which imitate network of events related to each other. The complexity level of those algorithms is very high, and today mostly big corporations can afford them. The approach for next days that could meet this challenge are ontologies, which are based on XML documents and use syntax readable by human beings.

Other critical circumstance is backup of order's database. Additional parallel system should work in the background and be launched to act only in case of failure of the primary.

Graphic might be changed into more attractive to the user, but it mostly depends on company's demands. At this moment they are highly communicative but different layout may attract more people, or transpose web site into more classic or mature design.

Systems which are updated numerous of times, spread information about upcoming events by newsletters. This option could be adopted as it might help stay in touch with potential customers and keep company's name in their memory. This activity might be very effective, but the strategy of that should be designed by someone with strong management experience.

11. Student reflections

I am very pleased about this project, the way of development process as well as documentation. It gave me an opportunity to simulate commercial application which shares similarities with two different areas - distributed application approach and online shop.

Challenges that I have faced during development deepen my understanding of processing data, and importance of it according to it's sensitiveness. Managing e-commerce application turned out not as hard as I was expecting, although application that I have developed can be modified in order to implement new features that may increase their attractiveness and add value to the major process.

To solve all problems that I have faced Internet sources and 3 books that I had were enough and guided me to accomplish tasks.

Bibliography and references

- [1] Norlaily Yaacob, Nazaraf Shah (2011) *320CT - Distributed application development (module lectures)*, Coventry University
- [2] B. Evjen, S. Hanselman, D. Rader (2010) *Professional ASP.NET 4: in C# and VB*, Wiley Publishing, Inc.
- [3] YouTube (2010) *Create and Use ASP.NET Web Service in Visual Studio 2008* [online] available from <<http://www.youtube.com/watch?v=qOqEKpYbTzw>> [1 December 2010]
- [4] MSDN (2010) *SQL Server Express - Learn* [online] available from <<http://msdn.microsoft.com/hi-in/express/aa718391.aspx#1>> [6 December 2010]
- [5] LINQPad (2010) *Why LINQ beats SQL* [online] available from <<http://www.linqpad.net/WhyLINQBeatsSQL.aspx>> [7 December 2010]
- [6] A. Freeman, J. C. Rattz Jr. (2010) *Pro LINQ: Language Integrated Query in C# 2010*, Apress, ISBN-13 (pbk): 978-1-4302-2653-6, ISBN-13 (electronic): 978-1-4302-2654-3
- [7] Microsoft ASP.net (2010) *LINQ Videos: The Official Microsoft ASP.NET Site* [online] available from <<http://www.asp.net/linq/videos>> [17 January 2011]
- [8] MSDN (2010) *ASP.NET Session State Overview* [online] available from <<http://msdn.microsoft.com/en-us/library/ms178581.aspx>> [17 January 2011]
- [9] The Code Project (2011) *Exploring Session in ASP.Net* [online] available from <<http://www.codeproject.com/KB/aspnet/ExploringSession.aspx>> [20 January 2011]
- [10] MSDN (2010) *ASP.NET Master Pages* [online] available from <<http://msdn.microsoft.com/en-us/library/wtxbf3hh.aspx>> [17 January 2011]
- [11] Microsoft ASP.net (2011) *ASP.NET Master Pages Tutorials: The Official Microsoft ASP.NET Site* [online] available from <<http://www.asp.net/master-pages/tutorials>> [17 January 2011]
- [12] A List Apart, Nick Rigby (2004) *Drop-Down Menus, Horizontal Style* [online] available from <<http://www.alistapart.com/articles/horizddropdowns/>> [9 December 2010]
- [13] Hv-designs, Richard Carpenter (2009) *Sliding JQuery Menu* [online] available from <<http://www.hv-designs.co.uk/2009/02/17/sliding-jquery-menu/>> [9 December 2010]
- [14] M. MacDonald, M. Szpuszta (2008) *Pro ASP.NET 3.5 in C# 2008, Second Edition*, Apress, ISBN-13 (pbk): 978-1-59059-893-1, ISBN-10 (pbk): 1-59059-893-8
- [15] Mikesdotnetting (2011) *Create PDFs in ASP.NET - getting started with iTextSharp* [online] available from <<http://www.mikesdotnetting.com/Article/80/Create-PDFs-in-ASP.NET-getting-started-with-iTextSharp>> [14 Jan 2011]
- [16] Massoud Mazar (2008) *Code sample for using iTextSharp PDF library* [online] available from <<http://www.mazsoft.com/blog/post/2008/04/30/Code-sample-for-using-iTextSharp-PDF-library.aspx>> [14 Jan 2011]
- [17] Mariiyski Theatre (2011) [online] available from <<http://www.mariinsky.ru/en/>> [22 January 2011]
- [18] The Bolshoi Theatre of Russia (2011) [online] available from <<http://www.bolshoi.ru/en/>> [22 January 2011]
- [19] New York City Opera (2011) [online] available from <<http://www.nycopera.com/>> [22 January 2011]
- [20] Royal Opera House (2011) [online] available from <<http://www.roh.org.uk/>> [22 January 2011]

January 2011]

[21] European Central Bank (2011) Reference rates [online] available from
<<http://www.ecb.int/stats/eurofxref/eurofxref-daily.xml>> [14 January 2011]

[22] Flickr (2011) [online] available from
<http://farm3.static.flickr.com/2330/2416100250_aa478b3496.jpg> [28 January 2011]