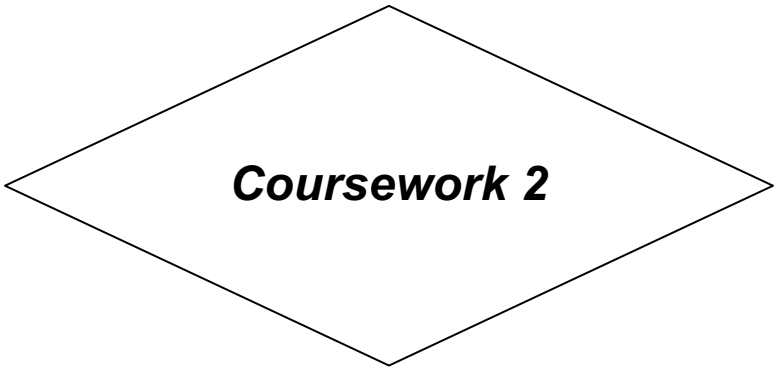
# MSc Business Computing

## *2CMMM01 – Database Systems*

## 1 Web Transactions




Student: Diamantopoulos Dionysis

ID: 01184485

Module: Database Systems

Code: 2CMMM01

Lecturer: Angelos Stefanidis

# Table of Contents

# 2 Introduction

The Internet is a world-wide collection of interconnected computer networks. The Web is a hypermedia-based 'point-click' system that provides a simple means to explore the information on the Internet. Information on the Web if stored in documents using HTML and displayed by a Web browser. The Web browser exchanges information with a Web server using HTTP.

In the Web environment, the traditional two-tier client-server model has been replaced by a three-tier model, consisting of a user interface layer (the client), a business logic and data processing layer (the application server), and a DBMS (the database server), distributed over different machines.

The advantages of the Web as a database platform include DBMS advantages, simplicity, platform independence, GUI, standardization, cross-platform support, transparent network access, and scalable deployment. The disadvantages include reliability, poor security, cost, poor scalability, limited functionality of HTML, statelessness, bandwidth, performance, and immaturity.

The Common Gateway Interface (CGI) is a specification for transferring information between a Web server and a CGI script. It is a popular technique for integrating databases into the Web. Fast CGI overcomes most of the problems suffered by CGI and includes the best features of CGI and server APIs.

An alternative approach to CGI is to extend the Web server, typified by the Netscape API (NSAPI) and Microsoft Internet Server API (ISAPI). Using an API, the traditional functionality is linked into the server itself.

Scripting languages such as JavaScript and VBScript can be used to extend both the browser and the server. Scripting languages allow the creation of functions embedded within HTML.

Some other techniques related to accessing database via the Web includes: the JDBC, JSQL, Microsoft Active Platform and Oracle's Network Computing Architecture.

Security in a Web environment consists of proxy servers, firewalls, digital signatures, digital certificates, Kerberos, Secure Sockets layer (SSL), Secure HTTP (S-HTTP), Secure Electronic Transactions (SET), and Secure Transaction Technology (STT).

# 3 Web Functionality

The online transaction is essential to a successful business on the web. In the broadest sense, an online transaction is any web-based interaction that involves the two-way exchange of information. Given the sort of complex information flow that may be embodied within any one transaction, the importance of ease-of-use hardly needs to be emphasized. But perhaps precisely because they place such demands on the user, online transactions are often the cause of serious usability problems online.

## *3.1 Analysis*

The first thing for any design team to get right is the location of the booking form. If this isn't obvious, users can become frustrated before they even begin and potential sales may be lost. There is no 'correct' place for the booking form. The main thing is that it is obvious to users and easily accessible. In our case only EasyJet have this advantage compare with the Buzz which the user has to make few clicks to go the booking form.

### 3.1.1 Step 1: Ticket Search

The first step in online flight bookings is the ticket search. Here the user's goal is to find suitable flights. This involves selecting trip types, destinations, travel dates and times, and fare options. So how best can users are supported in this task?

**Provide a progress indicator**

For first time or infrequent users, it is important to spell out what the process involves. This is best done as early as possible and should include what steps the user will be asked to perform and, if possible, how long the entire process will take. Both of the sites provide useful graphics at the top of each page, indicating the current and other remaining steps left in the purchasing process.

**Provide 'one-way' option and mute return fields if chosen**

To support users booking one-way flights, we recommend providing a one-way option and muting return fields if this is chosen. This simple feature reduces on-screen complexity and enables users to focus on only those fields relevant to their journey. Buzzway provides this useful facility.

**Use drop-down menus rather than text entry boxes for selection of departure and destination points**

Selecting departure and destination points can be a difficult task for many users. Users have problems spelling city or airport names, remembering airport codes and figuring out which cities or airports are served by a particular carrier.

The sites have a drop-down menu rather than text entry boxes for selection of departure and destination points. Drop-downs can also be 'dynamic', providing destination lists tailored to an already entered departure point. This feature can speed up the selection process by only presenting 'live' options to users.

**Allow users to specify a time band**

Allow users to specify a time band as broad or narrow as they wish. This may be important, particularly on 'shuttle' routes with regular flights. Again, EasyJet and Buzz didn't offer this availability to the users.

**Allow ticket type restrictions**

Buzz offer different ticket type options. For example, children, adult, infants, etc. With the use of drop-down list the users have that availability to choose the numbers and the type of tickets that they want.

### 3.1.2 Step 2: Select Flights

Having searched for flights, the next step is to select suitable flights. Here the user' s goal is to peruse all flight options and chose those which best fit their needs in terms of price, departure/arrival time, or whatever other criteria are appropriate.

**Search Results Can Be Displayed By a Variety of Criteria**

Users should be able to view search results in the way that is most important to them. There are many ways in which search results can be displayed, for example by price, number of stopovers / time in the air, airline, time of departure etc. Easyjet allows the user to see flights by seating class, date and time of day, or by price.

**Provide on one screen all the information a user needs to be able to compare the alternatives, including route, time and price**

It is essential to be able to compare possible flights, not only on departure and arrival times but also on price information. Failing to provide price information when presenting alternative itineraries means asking the user to 'pogo' back and forth through multiple options until they find a flight that suits their pocket. Presenting price and schedule information means that users can quickly *and* easily compare the options. *Buzzaway and EasyJet* show the price for each leg of outgoing and return flights. This facilitates maximum flexibility, as the user may want to trade an earlier, cheaper flight with a more expensive later one.

**Provide currency conversion**

In order for users to immediately grasp the cost of their ticket, it is helpful to provide a currency conversion service online. None of the two airlines have this availability, which is very important because of the new euro currency which is now in place.

### 3.1.3 Step 3: Remind the user about the itinerary

Before committing to purchasing, it is a good idea to remind users of their itinerary. This can be presented at the top of the screen, before ticketing details. Both of the sites don't give to the users this availability

Having selected their flight, the next step is for the user to purchase the flight tickets. This involves reviewing fare rules (or airline terms and conditions) and supplying

ticketing information such as passenger details, billing address, delivery address and credit card details.

### 3.1.4 Step 4: Purchase Tickets

Users should be able to enter their details in a clear, logical fashion. Also, users should be able to discriminate between billing and delivery addresses. This is also a good point at which to present other options such as meal type.
Both of the sites, in contrast, provide a simple form and clearly separate personal and billing details. Users are also given advice on the correct entry of phone numbers.

### 3.1.5 Step 5: Confirm page

Having entered passenger and billing information, the user has one last step - to either confirm or cancel their booking.

**Customer reviews flight details and prices and confirms or cancels their booking**

The stakes get higher once money comes into the equation. Here users need to feel in control of their purchase. After entering credit card details, users should be able to make a final check on flight information before making a final commitment. Although *Buzzaway*, for example, do provide a confirmation page, this information is not available at the exact moment when a final decision is made.

### 3.1.6 Other Issues

Occasionally users make errors. Users can forget to enter some essential information, or make input errors such as incorrectly spelling an airport. In these instances, error messages are required. Whether these are provided by means of pop-up windows, new pages, or additional information presented on the same page depends on the context. What is most important is that previously entered information is not lost, the error message is clear and concise, and that the user is made aware of how to rectify the problem as easily as possible. Both of the sites they offers to the users this availability.

## *3.2 Web – Database Architectures*

### 3.2.1 Client Server (C/S)

Client/Server is simply an architectural method of providing information to an end user; but that's where the simplicity ends. Client/Server is a general description of a networked system where a client program initiates contact with a separate server program (usually on a different machine) for a specific function or purpose. The client exists in the position of the requester for the service provided by the server.

As large scale, complex information systems have evolved over the past two decades, the Client/Server model of computing has come to be generally accepted as the preferred architecture for application design and deployment.

C/S computing architecture is currently the heart and soul of enabling technologies like groupware and workflow systems.

The effects of future C/S technologies on our industry are going to be just as profound as the transformation we just went through, when network computing applied a giant chainsaw to monolithic (mainframe-based) applications and separated them into Client and Server (C/S) components.

The term Client/Server has traditionally been associated with a desktop PC connected over a network to some sort of SQL-database server. In fact, the term Client/Server formally refers to a logical model that provides for a division of tasks into 'client' and 'server' layers or 'tiers'.

### 3.2.2 Two-Tier Client/Server Architectures

In a two-tier client/server architecture, the client communicates directly with the database server. The application or business logic either resides on the client or on the database server in the form of stored procedures.

**Fat Clients**

Much of the processing occurs on the fat clients, but datasets of information are delivered to the client using Structured Query Language (*SQL*) techniques to perform requests from a database server, which simply reported the results of queries.
The more complex the application, the fatter the client becomes and the more powerful the client hardware must be to support it. In addition, the network 'footprint' using fat clients, is very large, so that the effective bandwidth of the network, and thus the corresponding number of users who can effectively use the network, is reduced.

**Fat Servers**

An alternative 'thin' Client <-> 'fat' Server configuration, where the user invokes procedures stored at the database server, is another approach that is used in the 2-tiered architecture. The 'fat' Server model is more effective in gaining performance, because the network footprint, although still heavy, is lighter than the fat Client approach.

### 3.2.3 Three-Tier Client/Server Architectures

A newer generation of Client/Server implementations takes this segmented model a step further and adds a middle tier to achieve a '3-tier' architecture.
In a three-tier or multi-tier environment, the client implements the presentation logic (thin client). The business logic is implemented on an application server(s) and the data resides on database server(s).

A Multi-tier architecture is thus defined by the following three component layers:

1. A front-end component, which is responsible for providing portable presentation logic;
2. A back-end component, which provides access to dedicated services, such as a database server.

3. A middle-tier component, which allows users to share and control business logic by isolating it from the actual application;

Other advantages of Multi-Tier Client/Server architectures include:

1. Changes to the user interface or to the application logic are largely independent from one another, allowing the application to evolve easily to meet new requirements.
2. Network bottlenecks are minimized because the application layer does not transmit extra data to the client, only what is needed to handle a task.
3. When business logic changes are required, only the server has to be updated. In two-tier architectures, each client must be modified when logic changes.
4. The client is insulated from database and network operations. The client can access data easily and quickly without having to know where data is or how many servers are on the system.
5. Database connections can be 'pooled' and thus shared by several users, which greatly reduces the cost associated with per-user licensing.
6. The organization has database independence because the data layer is written using standard SQL which is platform independent. The enterprise is not tied to vendor-specific stored procedures.
7. The application layer can be written in standard third or fourth generation languages, such as Java, C or COBOL, with which the organization's in-house programmers are experienced.

### 3.2.4 N-Tier Architectures

As the Client/Server model continued to evolve, more sophisticated multi-tier solutions appeared where client-side computers began to operate as both clients and servers.

This latest refinement of the Client/Server model came when software developers recognized that the smaller, specialized processes were easier to design, faster to implement and cheaper to maintain. These same principles were in turn, applied to the server side of the equation, resulting in smaller, specialized server processes.

## *3.3 Approaches to Integrating the Web and DBMS*

### 3.3.1 CGI (Common Gateway Interface)

In general, a Web server is only able to send documents and to tell a browser what kinds of documents it is sending. By using CGI, the server can also launch external programs (i.e., CGI programs). When the server recognizes that a URL points to a file, it returns the contents of that file. When the URL points to a CGI program, the server will execute it and then sends back the output of the program's execution to the browser as if it were a file.

Before the server launches a CGI program, it prepares a number of environment variables representing the current state of the server, who is requesting the action, and so

on. The program collects this information and reads STDIN. It then carries out the necessary processing and writes its output to STDOUT (the standard output stream). In particular, the program must send the MIME header information prior to the main body of the output. This header information specifies the type of the output.

### 3.3.2 SSI (Server-Side Includes)

*SSI* is the process that gets the server to parse the document before sending it to the browser. As well as directly including a named file into a document, SSI provides special commands to include the current data and time, or report the last-modification date of a file or its size. In particular, it provides a command to allow a program to be executed in the manner of CGI and to incorporate its output into the document. All SSI commands are embedded within regular HTML comments. A server that does not support SSI passes the commands on to the browser, which ignores them because they are formatted as comments. A server that does understand SSI, parses the HTML from the top down, executing each comment-embedded command and replacing the comment with the output of the command.

Many of the security risks of SSI are similar to those of CGI.

### 3.3.3 HTTP Cookies

An HTTP cookie is a technique that helps maintain state in Web applications. A cookie is in fact a small text file containing:

1. Name of the cookie.
2. Domains for which the cookie is valid.
3. Expiration time in GMT.
4. Application-specific data such as user information etc.

Cookies are sent by the server to the browser, and saved to the client's disk. Whenever necessary, the server can request a desired cookie from the client.

### 3.3.4 HTTP server (APIs)

HTTP server (Web server) Application Programming Interface (API) adds functionality to the server or even changes server behavior and customizes it. Such additions are called *non-CGI gateways*.

The central theme of Web database sites created with HTTP server APIs is that the database access programs coexist with the server. They share the address space and run-time process of the server.

This approach is in direct contrast with the architecture of CGI, in which CGI programs run as separate processes and in separate memory spaces from the HTTP server. At the present, there are two main APIs: the Netscape Server API (NSAPI) and Microsoft Information Server API (ISAPI).

### 3.3.5 JDBC

JDBC package defines a database access API that supports basic SQL functionality and enables access to a wide range of relational DBMS products. On top of JDBC, higher-level APIs can be built: an *embedded SQL for Java* and *a direct mapping of relational database tables to Java classes*. It is based on dynamic SQL. Although often thought to stand for Java Database Connectivity, JDBC is a trademark name, not an acronym.

### 3.3.6 JSQL

Another JDBC-based approach uses Java with embedded SQL, which is a specification for Java with static embedded SQL, proposed by a consortium of organizations (Oracle, IBM and Tandem). JSQL comprises a set of clauses that extend Java to include SQL constructs as statements and expressions. A JSQL translator transforms the JSQL clauses into standard Java code that accesses the database through a call-level interface. It is based on static embedded SQL. Thus, it facilitates static analysis for syntax checking, type checking, and schema checking, which may help produce more reliable programs at the loss of some functionality/flexibility.

### 3.3.7 Scripting Languages

Scripting languages allow the creation of functions embedded within HTML code. This allows various processes to be automated and objects to be accessed and manipulated. There are mainly two scripting languages associated with HTML: *JavaScript* and *VBScript*.

### 3.3.8 The Microsoft Active Platform

The Microsoft Active Platform is open, standard-based software architecture for delivering applications over the Internet and intranets'. There are various tools, services and technologies in Active Platform such as HTML, Scripting languages and components (Java or ActiveX). When these technologies are combined, the result can be put on the client machine, giving an *Active Desktop*, or it can be put on the Web server, giving an *Active Server*. Active Server Pages (ASP) is a programming model that allows dynamic, interactive Web pages to be created on the Web server. ASP is build around files with the extension '.asp'.

# 4 Security

The strong need for information security is attributed to several factors, including: the availability of sensitive information stored in corporations and governments databases to the outside world; the ease with which malicious code can be distributed by ill-intentioned people via automation (for example, reading a victim's address book and propagating viruses automatically to all addresses in the book); the ease with which

computer crimes are perpetrated anonymously from across geographic boundaries; and the general lack of forensic evidence in computer crimes, which makes the detection and prosecution of criminals difficult.

## *4.1 Basic Threats and Countermeasures*

Full countermeasures against communication security threats can only be achieved with both physical and data security. In particular it is often possible to ensure that attempts to infiltrate or tap communications are either detectable, unprofitable (no information is gained), or both.

### 4.1.1 Confidentiality

Confidentiality ensures that the contents of communications are not available to unauthorized parties. This is often provided by encryption in which the data is transformed so that it is unreadable by an eavesdropper. The receiving party is provided with the means of transforming the received data back to its original form.

### 4.1.2 Integrity

Integrity services provide a high level of confidence that information is received in the same condition as sent, thus protecting against undetected tampering with data in transit. To achieve this, an Integrity Check Value (ICV) is typically appended to the message near the message source. The receiver has a secret key that allows him to recalculate the integrity check value, and so can check that the received ICV is correct.

### 4.1.3 Source and Destination of authentication

Authentication comprises mutual verification of identities between sender and receiver. The appropriate exercise of confidentiality and integrity mechanisms or other cryptographic means may achieve this.

Two particular kinds of threat emerge against users of switched public networks, which are of particular concern to users with stringent security requirements such as defense.

### 4.1.4 Traffic Analysis

Traffic analysis consists of an attack whereby the communication channel is tapped and statistical information about the data traffic is accumulated. Specifically, the volume, timing, source and destination addresses of communication data are collected by the attacker.

The following are possible countermeasures against attackers gaining information via traffic analysis.

1. **Broadcasting (destination protection)**

Consider a number of sites forming a closed user group, connected across a public network. If all data from each site were broadcast to every member of the group then a tapper would not know the "real" destination site from the "dummy" ones.

2. **Dummy data (timing protection)**

Each user produces a constant amount of traffic by adding dummy traffic to meaningful traffic then the volume and timing of the latter will be effectively hidden, as meaningful data and dummy data will be indistinguishable to an eavesdropper after encryption.

## *4.2 Security Issues*

Security risks exist in many areas of a Web database application. Without special software, all Internet traffic travels in the open and anyone with a little bit skill can intercept data transmission on the Internet.
In general, security issues in Web database applications include the following:

- Data transmission (communication) between the client and the server is not accessible to anyone else except the sender and intended receiver (*privacy*).
- Data cannot be changed during transmission (*integrity*).
- The receiver can be sure that the data is from the sender (*authenticity*).
- The sender can be sure the receiver is genuine (*non-fabrication*).
- The sender cannot deny he/she sent it (*non-repudiation*).
- The request from the client should not ask the server to perform illegal or unauthorized actions.
- The data transmitted to the client machine from the server must not be allowed to contain executables that will perform malicious actions.

### 4.2.1 Proxy Servers

A proxy server is a system that resides between a Web browser and a Web server. It intercepts all requests to the Web server to determine if it can fulfill the requests itself. If not, it forwards the requests to the Web server. Due to the fact that the proxy server is between browsers and the Web server, it can be utilized to filter requests, i.e. be a defense for the server.

### 4.2.2 Firewalls

A firewall is a system designed to prevent unauthorized access to or from a private network (Intranet). It can be implemented in either hardware, software, or both.

All data entering or leaving the Intranet (connected to the Internet) must pass through the firewall. They are checked by the firewall system and anything that does not meet the specified security criteria is blocked.

### 4.2.3 Kerberos

Kerberos is a server of secured usernames and passwords. It provides one centralized security server for all data and resources on the network. Database access, login, authorization control, and other security measures are centralized on trusted Kerberos servers. The main function is to identify and validate a user.

### 4.2.4 Secure Sockets Layers (SSL) and Secure HTTP (S-HTTP)

SSL is an encryption protocol developed by Netscape for transmitting private documents over the Internet. It works by using a private key to encrypt data that is to be transferred over the SSL connection. Both Netscape and Microsoft IE support SSL.

Another protocol for transmitting data securely over the Internet is called Secure HTTP, a modified version of the standard HTTP protocol. Whereas SSL creates a secure connection between a client and a server, over which any amount of data can be sent securely, S-HTTP is designed to transmit individual messages securely.

In general, the SSL and S-HTTP protocols allow a browser and a server to establish a secure link to transmit information. However, the authenticity of the client (the browser) and the server must be verified. Thus, a key component in the establishment of secure Web sessions using SSL or S-HTTP protocols is the digital certificate. Without authentic and trustworthy certificates, the protocols offer no security at all.

# 5 Database Location

Organizations move into distributed databases for one of these two reasons. First reason is that they have outgrown the capacity, like storage capacity or processing power capacity, of their current database server. This situation requires separating a logically centralized database into a distributed database that use the potential of two or more computing processes. Typically, these two computers are sitting side by side, but it may include geographically separated sites, running compatible DBMS products.

In today's business environment, scalability is a prime characteristic of both distributed and centralised databases which is driving business to begin exploring this type of technology. (Noel, 1996). Many of the recent available commercial database servers can support horizontal scaling, which lets database systems include additional servers or processors. The database system transparently distributes the data and the database processing load. Most top-down operations in the business world today work with a comparably sized or scaled-down version of the original centralized database. This approach ensures transparency while maintaining data integrity and applications compatibility.

Second reason is that organisations with physically separate sites, each maintaining its own data, are candidates for a distributed database. A possible solution

for this type of organization might be to combine all the dispersed databases into one central database. However, this approach has several disadvantages. First, a central computer can cause phenomenal processing demands during peak times for itself. Second one, a central computer would become a single point of failure. The other one is communications costs for remote access to a central computer can be very costly. Instead of using a centralized database, a bottom-up integration of a distributed database can cope with these problems. The database is combining existing databases running on mixed systems into a single, virtual distributed database. This approach preserves an organization's investment in database software and applications, as well as allowing the data to be stored where it's used most. Also, this will accomplished with a comparably sized budget and staff.

## *5.1 Centralized Vs Distributed Technology*

In the centralized systems data stored in tables in a central DBMS. Users will access the tables through either dumb terminals or their PCs via a LAN. There is only one server running all applications and centralized database software. Typically, these servers use more powerful machines (for example, a Unix-based minicomputer or a mainframe). The service computer receives the requested pages and transmits them to the user terminals via a communication network. Even if run on a PC LAN, however, this configuration still can solves many of the problems with maintaining data integrity, eliminating redundancy of data, and processing changes, deletions, and updates concurrently. Centralized databases also simplify routine services, like security, back up, and maintenance. However, centralized system has many problems which are very costly.

Those issues can be coped by using a decentralized configuration that has both data and applications located in independent sites. The main key of decentralized configuration is the concept that the data is not a shared resource. Each site maintains its data locally and updates the central database at regular time based. In situations where the data needs to be a shared resource, a distributed multi user database can be the solution. As with the decentralized model, each site has an independent CPU and DBMS, and also its own data and applications.

# 6 Graphical User Interface GUI

## *6.1 Easy Jet*

## *6.2 Buzzway*

## 6.3 Proposed Navigation

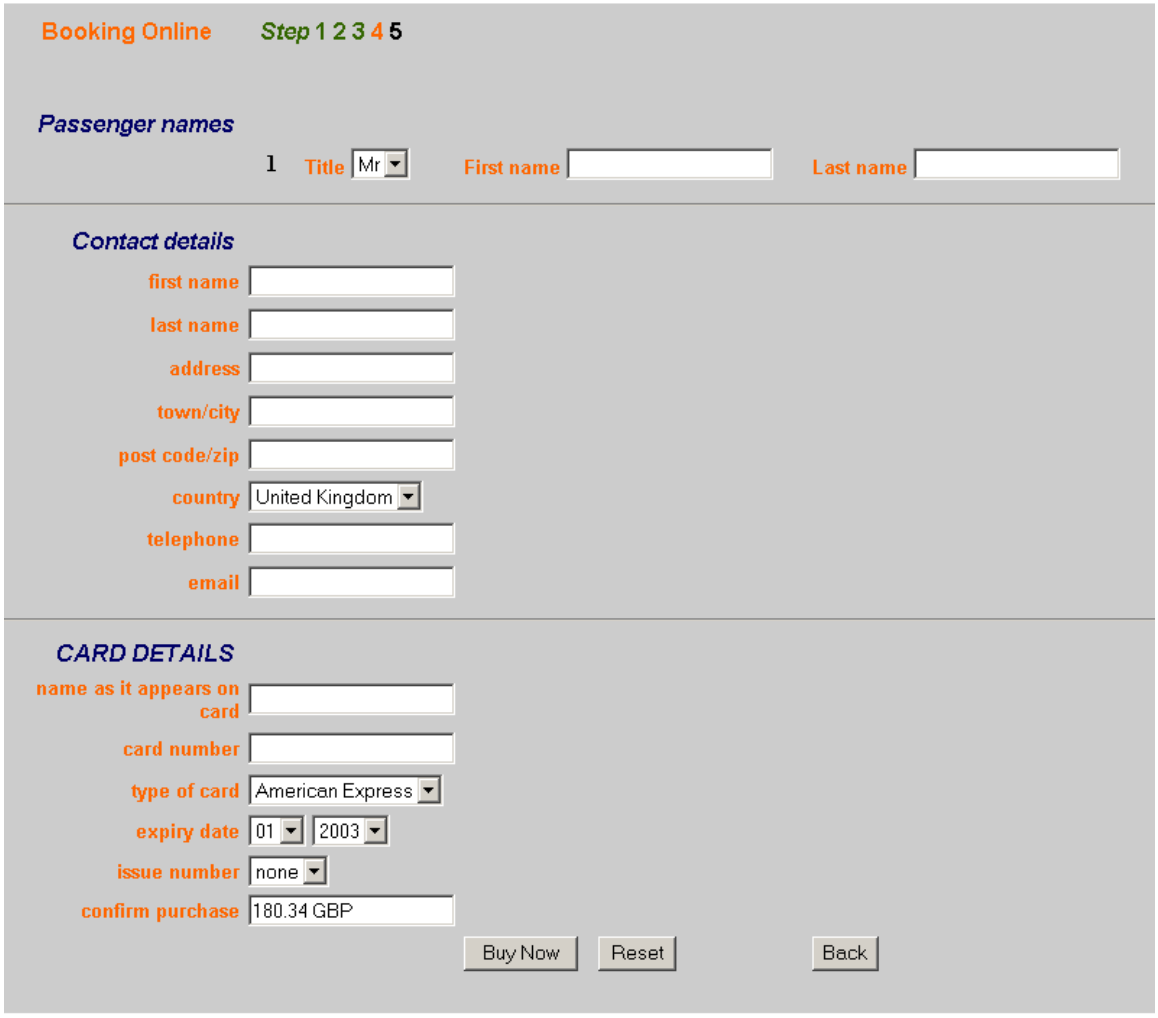
Booking Online Step 1 2 3 4 5

| OUTBOUND | Date | Flight | Departs Time | Arrival Time | Price(GBP) |
|---|---|---|---|---|---|
| ◉ | Sunday 13 January 2002 | 451 | London Lutonat 13:40 | Athens at 19:30 | 112.00 GBP |
| ○ | Sunday 13 January 2002 | 453 | London Luton at 22:55 | Athens at 04:45 | 62.00 GBP |
| ○ | Monday 14 January 2002 | 451 | London Luton at 13:40 | Athens at 19:30 | 82.00 GBP |
| ○ | Monday 14 January 2002 | 453 | London Luton at 22:55 | Athens at 04:45 | 42.00 GBP |
| ○ | Tuesday 15 January 2002 | 451 | London Luton at 13:40 | Athens at 19:30 | 32.00 GBP |
| ○ | Tuesday 15 January 2002 | 453 | London Luton at 22:55 | Athens at 04:45 | 27.00 GBP |

| RETURN | Date | Flight | Departs Time | Arrival Time | Price(GBP) |
|---|---|---|---|---|---|
| ○ | Sunday 13 June 2002 | 451 | Athens at 06:50 | London Luton at 08:40 | 32.50 GBP |
| ◉ | Sunday 13 June 2002 | 453 | Athens at 20:00 | London Luton at 21:50 | 47.50 GBP |
| ○ | Sunday 14 June 2002 | 451 | Athens at 06:50 | London Luton at 08:40 | 37.50 GBP |
| ○ | Sunday 14 June 2002 | 453 | Athens at 20:00 | London Luton at 21:50 | 57.50 GBP |
| ○ | Sunday 15 June 2002 | 451 | Athens at 06:50 | London Luton at 08:40 | 47.50 GBP |
| ○ | Sunday 15 June2002 | 453 | Athens at 20:00 | London Luton at 21:50 | 47.50 GBP |

Proceed  New Search  Currency Calculator


Booking Online Step 1 2 3 4 5
OUTBOUND
Date
Flight
Departs Time
Arrival Time
Price(GBP)
Sunday 13 January 2002
451
London Lutonat 13:40
Athens at 19:30
112.00 GBP
RETURN
Date
Flight
Departs Time
Arrival Time
Price(GBP)
Sunday 13 June 2002
453
Athens at 20:00
London Luton at 21:50
47.50 GBP
Total cost of your selection
OUTBOUND
airfare per person
112.00 GBP
plus TAX per person
5.00 GBP
plus Barclays fat cat charge per person
5.50 GBP
total per person
122.50 GBP
RETURN
airfare per person
47.50 GBP
plus TAX per person
7.34 GBP
total per person
54.84 GBP
GRAND TOTAL
total for 1 passenger
177.34 GBP
Currency Calculator
Book Secure
Back


Booking Online
Step 1 2 3 4 5
Passenger names
1
Title
Mr
First name
Last name
Contact details
first name
last name
address
town/city
post code/zip
country
United Kingdom
telephone
email
CARD DETAILS
name as it appears on card
card number
type of card
American Express
expiry date
01
2003
issue number
none
confirm purchase
180.34 GBP
Buy Now
Reset
Back

Booking Online *Step* 1 2 3 4 5

## *CONFIRMATION*

| *OUTBOUND* | Date | Flight | Departs Time | Arrival Time |
|---|---|---|---|---|
| | Sunday 13 January 2002 | 451 | London Lutonat 13:40 | Athens at 19:30 |
| *RETURN* | Date | Flight | Departs Time | Arrival Time |
| | Sunday 13 June 2002 | 453 | Athens at 20:00 | London Luton at 21:50 |

*Passenger names*

1 Title *Mr* First name *Dionysis* Last name *Diamantopoulos*

Class *First Class*

Confirm purchase 180.34 GBP

New Booking

# 7 References


http://www.w3.org

http://www.microsoft.com

http://www.oracle.com

http://ibm.com

http://java.sun.com

http://apache.org

http://mysql.com

http://www.perl.com

http://www.psinet.ch/html/e-biz/easyjet.htm#solution

http://www.ontoknowledge.org/oil/TR/introduction.html

http://www.aipa.it/english%5B4/unifiednetwork%5B2/feasibilitystudy%5B1/doc6.asp

http://softlab.icsd.aegean.gr/~dspin/pubs/conf/1997-CMS-WebSec/html/w3sec.html

http://www.caad.ed.ac.uk/publications/ear/salih.html

http://www2.bc.edu/~gallaugh/research/ism95/cccsa.html

http://echoman.com/knowledgesource/Web_v_clientserver.htm#_Section_3._More

http://www.tint.ltd.uk/services/system_integration.shtml

http://www.webtechniques.com/archives/1999/05/imho/

Thomas Connolly Database Systems Third Edition.

Alan Dix. Human Computer Interaction

Noel, R. 1996. Scale Up in Distributed Databases.

http://www.cs.rpi.edu/~noel/distr_scaleup/distributed.html