Project report on XML and DATABASE, course 7.5 points with in Computer Engineering

Title:

Flight booking Website

Givenname Surname:

Shixian Wen

Flightbook Website

Shixian Wen 2015-03-11

Abstract:

This Project mainly focus on using the combination of XML technologies and a My SQL relational database and the HTML(PHP) web page design. XML is composed of markup language for structuring data and has become a standard way to describe data on the Web.

Flightbooking is a flight booking website which uses these technologies to provide a platform for people searching suitable flights, booking tickets and revoking tickets. Also, you can enjoy the latest CNN news on our websites by using RSS technology.

This website can get user's input data from the client-side, using 3 methods to validating it HTML form, JavaScript and XML Schema Definition(XSD). Then the application will store these data in the XML files and Database. Third, data stored in the MY SQL database are retrieved, transformed and presented using XSLT, XPATH, XQUERY and CSS.

Keywords: XML, XSLT, XPATH, MYSQL, XQUERY, XSD, HTML, PHP, RSS,CSS, Flightbooking

content

Introduction

1.1 Background

Everybody has need to book the tickets for travel and get the information of what's happened around us. In this application, XML technologies tools such as XML, XSLT, XPATH, MYSQL, XQUERY, XSD,RSS are used to describe, transform, present and transport data. The database system helps us to store data more persistent and well-structured. Flightbooking Website has explored XML technologies ,MySQL database technologies, HTML,PHP techonologies fully in fulfilling its function.

1.2 Overall aim

The overall aim of this project is to demonstrate a practical use of XML technology and a relational database in order to achieve the designing goal: a website that can be used to search book, revoke flights and leave comments to the airline company.

2 Theory

2.1 XML

Quoted from Wikipedia " Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format which is both human-readable and machine-readable. It is defined by the W3C's XML 1.0 Specification[2] and by several other related specifications,[3] all of which are free open standards.[4]

The design goals of XML emphasize simplicity, generality and usability across the Internet.[5] It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures[6] such as those used in web services.

Several schema systems exist to aid in the definition of XML-based languages, while many application programming interfaces (APIs) have been developed to aid the processing of XML data."

2.2 XSD

Quoted from Wikipedia "XSD (XML Schema Definition), a recommendation of the World Wide Web Consortium (W3C), specifies how to formally describe the elements in an Extensible Markup Language (XML) document. It can be used by programmers to verify

each piece of item content in a document. They can check if it adheres to the description of the element it is placed in.[1]

Like all XML schema languages, XSD can be used to express a set of rules to which an XML document must conform in order to be considered "valid" according to that schema. However, unlike most other schema languages, XSD was also designed with the intent that determination of a document's validity would produce a collection of information adhering to specific data types. Such a post-validation infoset can be useful in the development of XML document processing software."

2.3 XPATH

Quoted from Wikipedia" XPath, the XML Path Language, is a query language for selecting nodes from an XML document. In addition, XPath may be used to compute values (e.g., strings, numbers, or Boolean values) from the content of an XML document. XPath was defined by the World Wide Web Consortium (W3C)"

2.4 XSL

Quoted from Wikipedia" XSLT (Extensible Stylesheet Language Transformations) is a language for transforming XML documents into other XML documents or other formats such as HTML for web pages, plain text or into XSL Formatting Objects, which may subsequently be converted to other formats, such as PDF PostScript and PNG."

2.5 Database

Quoted from Wikipedia" Formally, a "database" refers to a set of related data and the way it is structured or organized. Access to this data is usually provided by a "database management system" (DBMS) consisting of an integrated set of computer software that allows users to interact with one or more databases and provides access to all of the data contained in the database (although restrictions may exist that limit access to particular data). The DBMS provides various functions that allow entry, storage and retrieval of large quantities of information as well as provide ways to manage how that information is organized."

2.6 PHP

Quoted from Wikipedia" PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language."

2.7 XML DOM

Quoted from W3school" The XML DOM is

A standard object model for XML

A standard programming interface for XML

Platform- and language-independent

The XML DOM defines the objects and properties of all XML elements, and the methods (interface) to access them.

In other words: The XML DOM is a standard for how to get, change, add, or delete XML elements."

2.8 HTML

Quoted from Wikipedia" HyperText Markup Language, commonly referred to as HTML, is the standard markup language used to create web pages.[1] It is written in the form of HTML elements consisting of tags enclosed in angle brackets (like <html>). HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent empty elements and so are unpaired, for example <img>. The first tag in a pair is the start tag, and the second tag is the end tag (they are also called opening tags and closing tags).

Web browsers can read HTML files and compose them into visible or audible web pages. Browsers do not display the HTML tags and scripts, but use them to interpret the content of the page. HTML describes the structure of a website semantically along with cues for presentation, making it a markup language, rather than a programming language."

3 Method and code implementation

3.1 overall system design

From figure 1, you can see that our web application are consisted of three tier that include: Client Tier, Logic Tier(Application Server) and Database Tier. Client Tier plays a role of interface between our clients and application server. This Tier collects user's input using HTML forms and using post method to send it to the Application Sever. Logic Tier plays a role of manipulating users input, processing it using PHP, JavaScript and XML technologies. Then, it creates a connection to MYSQL database for data storage and sends a response to the user such as telling them your tickets have been booked successfully. The Database saves the data that have been processed in the Application Server and also provides methods to retrieving and manipulating the data, ensuring the data persistence across the whole system.

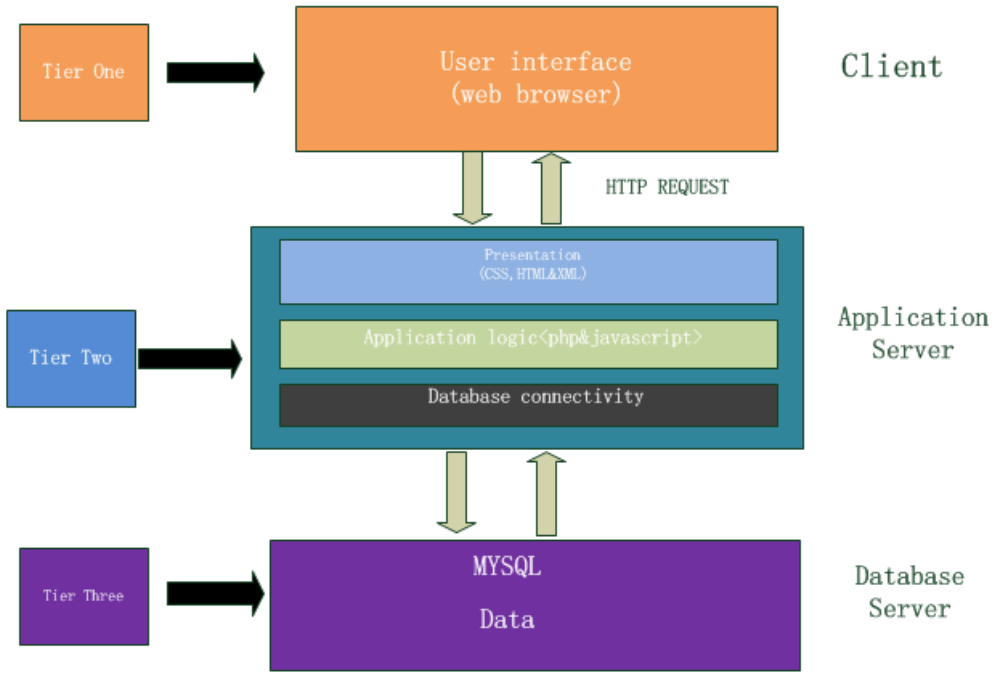

Figure 1 **overall system design**

3.2 Data flow chart:

In order to achieve data security and data integrity, we use the following data flow chart shown in figure 2. When user types their tickets information in the HTML form, our system collects and converts the input data into an XML file. By using XSD and javaScript to validate the input data, we can tell if these input data are correct or not. If the result is incorrect, we will return alert window to the user, reminding them to type it again. If the result is correct, we save these data in the database and local xml files.

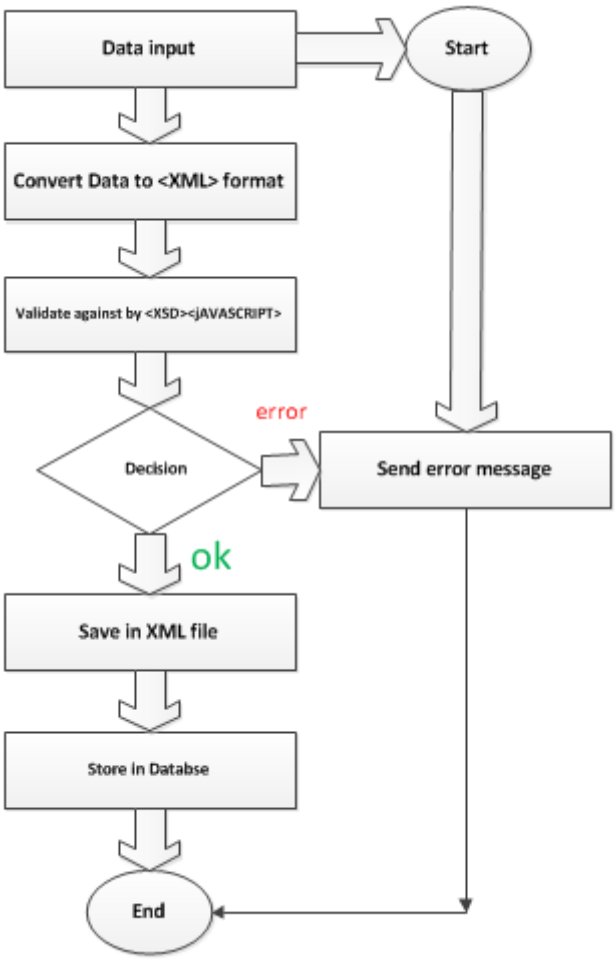

Figure 2 data flow diagram

3.3 Data Restriction Model

In this section, we will look the Data Restriction Model in details. As shown in Data flow chart, we use Both JavaScript and XSD to ensure that all the data that going to our system are in well format and correct. Figure 3 and Figure 4 show the result of the combination of JavaScript and XSD to ensure our system safety. Figure 5 gives us a sample of how we write XSD. If you want to look the code implementation in details, please refer to books.xsd and validate.min.js in the file.

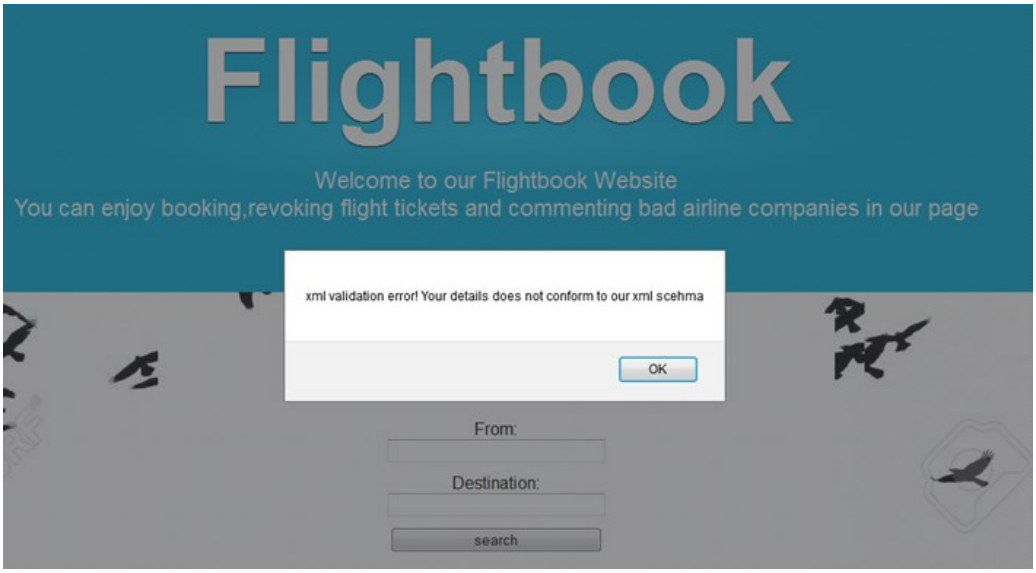

Figure 3 XSD validation error

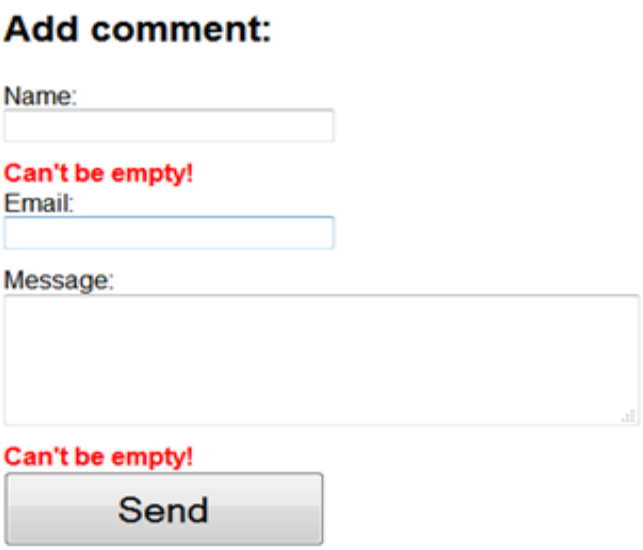

Figure4 JavaScript validation

```
<?xml version="1.0"?>
<xsd:schema version="1.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema" >

    <xsd:element name="Books" type="BooksType" />
    <xsd:complexType name="BooksType">
        <xsd:sequence>
            <xsd:element maxOccurs="unbounded" minOccurs="0" name="Profile" type="BookType" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="BookType">
        <xsd:sequence>
            <xsd:element name="firstname" >
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:maxLength  value="20"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="surname">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:maxLength  value="20"/>
                    </xsd:restriction>
                </xsd:simpleType>
```

Figure 5 books.xsd

3.4 User Data Structure

In this Section, we will discuss the user Data Structure. When a user successfully searched the flight they want in our website, they can fill the “book your tickets form” in our webpage which will successfully generate the XML data storing the user’s personal information and put them into database. We use their username as a key. If users want to revoke their tickets latter, they can fill another form to revoke the tickets they have booked. Our website will check the username and password, comparing them with the one that we store in our database to verify the validity. If they are not satisfying the service of the airline company, they can leave some comments in our website. In figure 6, figure 7 and figure 8, you can see our data structure of flight information, booking information and comments information separately.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<airlines xmlns:xsd="http://www.w3.org/2001/XMLSchema-instance" xsd:noNamespaceSchemaLocation="airlines.xsd">
    <airline id="1">
        <company>British Airways</company>
        <name>BA272</name>
        <price>255$</price>
        <from>Stockholm Arlanda Airport</from>
        <des>London Heathrow Airport</des>
        <flydate>2015/4/7</flydate>
        <flytime>17:21</flytime>
    </airline>
    <airline id="2">
        <company>British Airways</company>
        <name>BA274</name>
        <price>100$</price>
        <from>Stockholm Arlanda Airport</from>
        <des>London Heathrow Airport</des>
        <flydate>2015/4/7</flydate>
        <flytime>14:21</flytime>
    </airline>
    <airline id="3">
        <company>China Airline</company>
        <name>CA911</name>
        <price>225$</price>
        <from>Stockholm Arlanda Airport</from>
        <des>Beijing</des>
        <flydate>2015/4/6</flydate>
        <flytime>08:21</flytime>
    </airline>
</airlines>
```

figure 6 data structure of flights

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Books xmlns:xsd="http://www.w3.org/2001/XMLSchema-instance" xsd:noNamespaceSchemaLocation="books.xsd">
    <Profile username="mike">
        <firstname>Philip</firstname>
        <surname>Opuogen</surname>
        <password>look</password>
        <author_id>mike</author_id>
        <flight>BA272</flight>
    </Profile>
    <Profile username="nickname">
        <firstname>Shixian</firstname>
        <surname>wen</surname>
        <password/>
        <author_id>nickname</author_id>
        <flight>CA911</flight>
    </Profile>
    <Profile username="chengxuyuan">
        <firstname>gu</firstname>
        <surname>tianyu</surname>
        <password>asdf</password>
        <author_id>chengxuyuan</author_id>
        <flight>CA911</flight>
    </Profile>
```

Figure 7 data structure of books

```
<?xml version="1.0"?>
<xml>
    <comment time="1336315201">
        <email>shixianwen1993@gmail.com</email>
        <name>shixianwen</name>
        <message>China airline is awesome</message>
    </comment>
    <comment time="1336315956">
        <email></email>
        <name>weiliu;</name>
        <message>CA911 Iakes me to home </message>
```

Figure 9 data structure of comments

3.5 User Data Processing

When our application get the data from the users, we process the data via XPATH, XSD, PHP and SQUERY technologies.

For example when users successfully submit their booking data in our webpage, we use book.php to process the user data. First we use XPATH to see whether their username has already stored in our xml files, if they have already been registered, we return an alert for the users to change a username. If it is not, we will check if the data is validated or not by using XSD. If is being validated, we will update the XML files and database by using DOMXML and SQUERY. In figure 10 there are sample program of how we process the data.

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        session_start();
        // Process the registration form
        if (isset($_POST['submitregistration'])) {
            $firstname = $_POST['firstname'];
            $surname = $_POST['surname'];
            $username = $_POST['username'];
            $password = $_POST['password'];
            $flight = $_POST['flight'];



            //check if the username already exist in the file using xpath
            $doc = new DOMDocument;
            $doc->load('xml/books.xml');
            $xpath = new DOMXPath($doc);
            $finusername = $xpath->query("/Books/Profile[@username='$username']");

            //loop through to find values matching the username
            foreach ($finusername as $theusername) {
                echo '<h1>fail to pass the schemaValidate</h1>';
                $dom = new DOMDocument;
                $dom->load('xml/books.xml');
                $xpaths = new DOMXPath($dom);
                $query = sprintf("/Books/Profile[@username='$username']");
                foreach ($xpaths->query($query) as $record) {
                    $record->parentNode->removeChild($record);
                }
                $dom->save("xml/books.xml");
                header("location:index.php?bad=jjjjjjjj");
            } else {
            echo '<h1>successful to pass the schemaValidate</h1>';
                //add record to database
require("dbConnection.php"); // connection to the database

//insert into the Profile table
$query="INSERT INTO books (username, firstname, surname, password, flight)
"VALUES ('$username', '$firstname', '$surname', '$password', '$flight')";
        echo $query;
        //$query="SELECT firstname FROM `books`";
$result = mysql_query($query);

        if(!$result){
                echo "Failed!";
                $message = 'Error: ' . mysql_error() . "\n";
                echo $message;
                die($message);
        }else {echo "Success!";}

foreach ($finusername as $theusername) {

    print($theusername->nodeValue);
}
//if theusername return null
if (isset($theusername)) {

    header("location:index.php?category=jjjjjjjj");
} else {
    echo '<h1>firstname = ' . $firstname . '</h1>';

    //This will add a record into the register.xml file
    $xml = simplexml_load_file('xml/books.xml');
    $item = $xml->addChild('Profile');
    $item->addAttribute('username', $username);
    $item->addChild('firstname', $firstname);
    $item->addChild('surname', $surname);
    $item->addChild('password', $password);
    $item->addChild('author_id', $username);
    $item->addChild('flight', $flight);
    file_put_contents('xml/books.xml', $xml->asXML());



    $docs = new DOMDocument;
    $docs->load('xml/books.xml');
    if (!$docs->schemaValidate('xml/books.xsd')) {
    $query2="INSERT INTO login (username, password) ".
    "VALUES ('$username', '$password')";
            echo $query2;
     $result2= mysql_query($query2);
            if(!$result2){
                    echo "login Failed!";
                    $message = 'Error: ' . mysql_error() . "\n";
                    echo $message;
                    die($message);
            }else {echo "login Success!";}

    //$_SESSION['USERNAME'] = $username;
// open the inventory page page to allow user log on to the system
                header("location:index.php?book=jjjjjjjj");
            }
        }
    }
    ?>
    </body>
</html>
```

Figure 10 process the books information and insert data into xml and database

When User wants to revokes their tickets, they can use revoke form shown in our websites which is processed by revoke.php. In revoke.php, we retrieve password and username from the database, comparing them with the user's typing. If the result is matched, we delete the booking data in the xml and database via Squery and PHP. The sample program shown in figure 11

```
    </head>
    <body>
        <?php
        session_start();
        if (isset($_POST['submitLogin'])) {
            //create a connection to the database
            require("dbConnection.php");

            $sql = "SELECT * FROM login WHERE username = '" . $_POST['username'] .
                "' AND password = '" . $_POST['password'] . "';";
            $result = mysql_query($sql);
            $numrows = mysql_num_rows($result);
            $username2 = $_POST['username'];

            if ($numrows == null) {
                header("location:index.php?login=jjjjjjjj");
            } else {

                $_SESSION['USERNAME'] = $username2;
// open the inventory page page to allow user onto the system
                $sql = "DELETE FROM books WHERE username = '" . $_POST['username'] .
                    "' AND password = '" . $_POST['password'] . "' AND flight = '" . $_POST['flight'] . "';"
                //echo '**********'.$sql;
                $result = mysql_query($sql);
               // echo '**********'.$result;
                header("location:index.php?revoke=jjjjjjjj");
            }
        }
        ?>
    </body>
```

figure 11 match and delete data in database

3.6 XML Transformation

We use XSLT technology to display all the comments. The XSL document(comments.xsl) specifies how the XML(comments.xml) document should be displayed. The description of XSL document is a static HTML. XSL treat the XML as a tree to access and search the data. Figure 11 shows the comments.xsl.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
    <h2>comments</h2>
    <table border="1">
    <tr bgcolor="#9acd32">
      <th align="left">email</th>
      <th align="left">name</th>
      <th align="left">message</th>
    </tr>
    <xsl:for-each select="xml/comment">
    <tr>
      <td><xsl:value-of select="email"/></td>
      <td><xsl:value-of select="name"/></td>
      <td><xsl:value-of select="message"/></td>
    </tr>
    </xsl:for-each>
    </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Figure 11 XSLT(comments.xsl) file

3.7 Load remote XML by using RSS

In our webpage, by getting feeds from the RSS document which stores in XML format, the latest news of CNN is updated automatically and represented in our homepage. You can pick one which you want to read. The code implementation are shown in Figure 12

```
function getFeed($url){
    $x = simplexml_load_file($url);
    echo"<ul>";
    foreach ($x->channel->item as $entry){
       echo "<li><a href = '$entry->link' title = '$entry->title'>".$entry->title."</a></li>";
    }
    echo"</ul>";
}
```

Figure 12 automatically upload latest news by using rss providing by CNN

3.8 Database design and management

In Web Application, we have provided back-end data storage in MYSQL. We use this relational database management system to manage data. PHP script help us to create a

connection to the database and use SQL statement to create, retrieve, query, delete, update and define the records. In figure 13, it shows the database structure of our system.

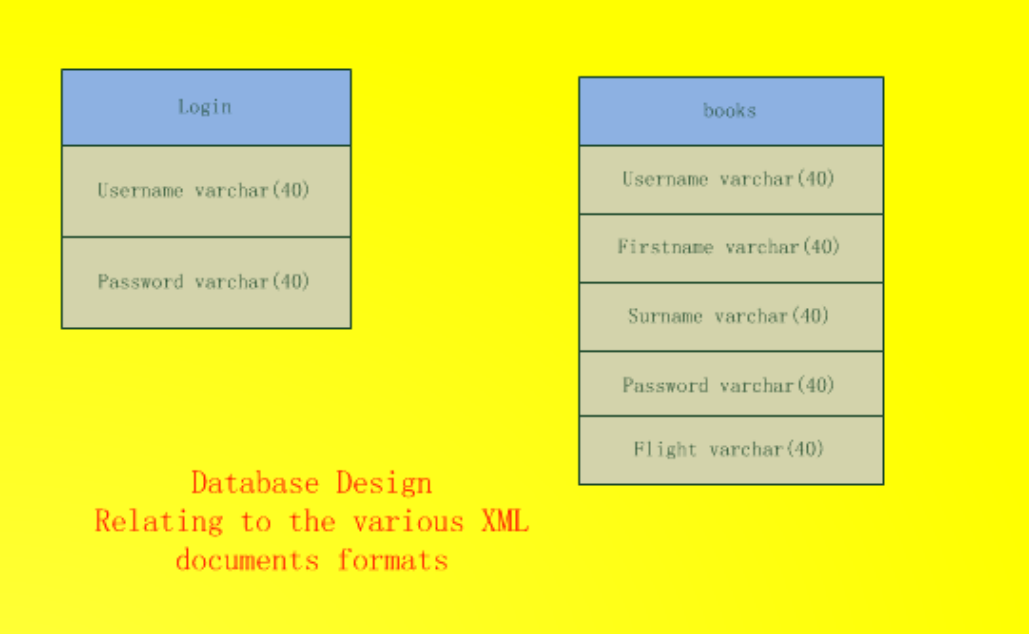

figure 13 Database Structure

4 Results

In this Chapter, we will present a captured screen of our Flight booking Website. It includes screen shot of different webpage and also the alert message from the misbehavior of users.

4.1 The main Page

In the main page, you can search Flights according to your destination. Also you can book and revoke tickets and leave some comments to improve our service. The main page is shown in Figure 14:

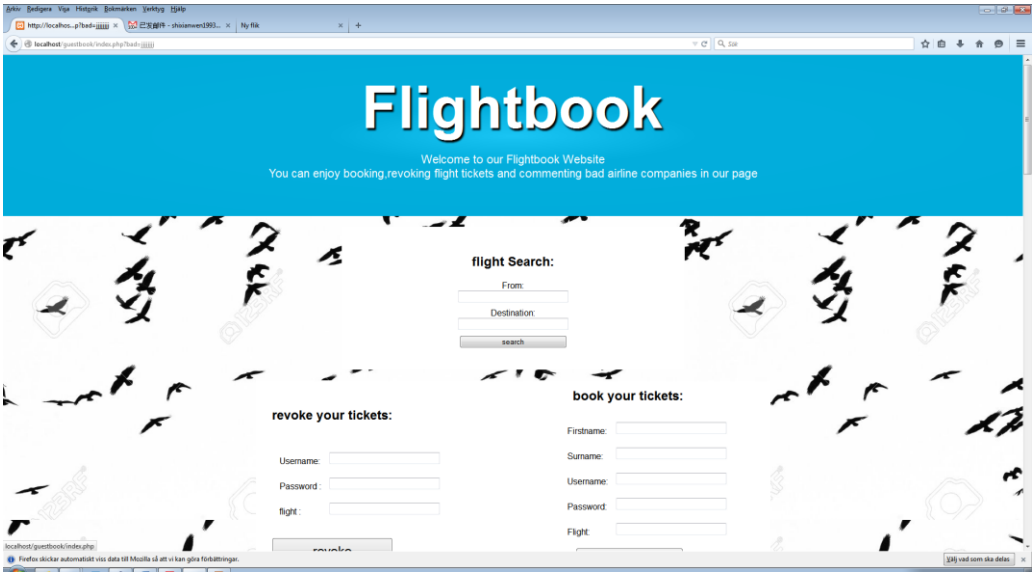

figure 14 main page of the flighting page

4.2 latest news in CNN

We use RSS provided by CNN to get the latest news. When users surf our webpage, they can also enjoy the latest news of the whole world(see in figure 15).

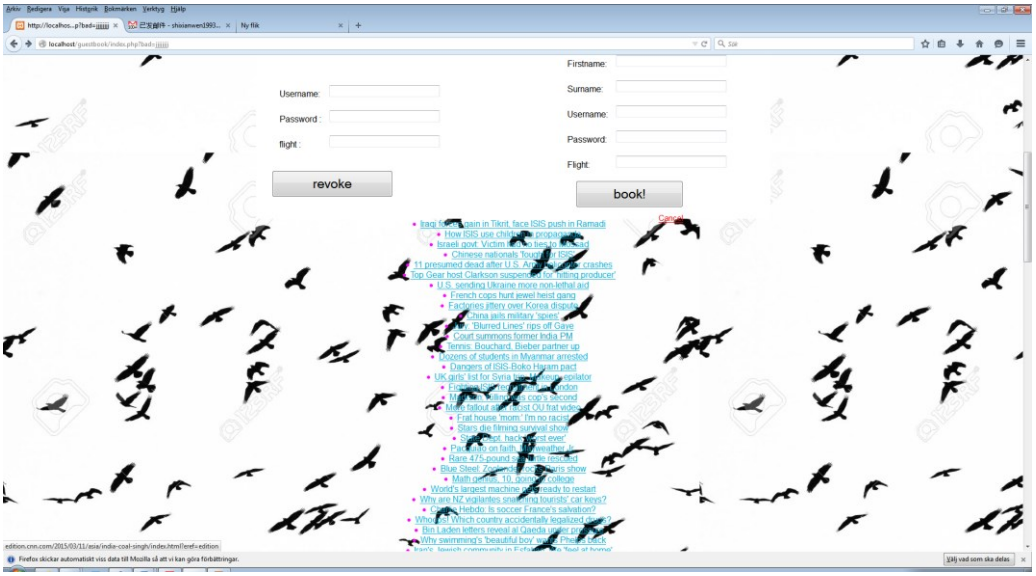

figure 15 lastest news of CNN

4.3 comments page

In our webpage, you can view and leave to improve our service(see in figure 16)

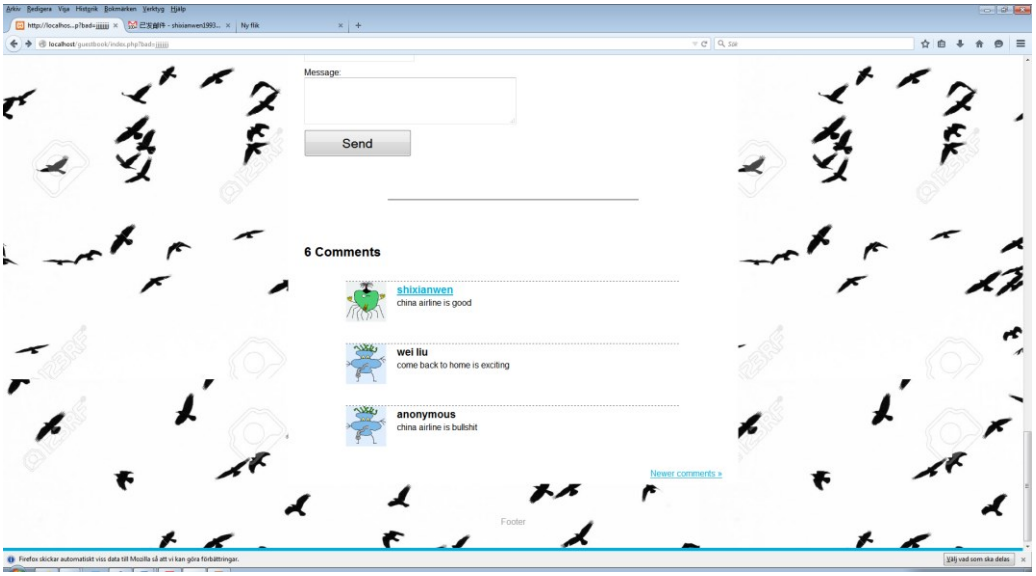

figure 16 comments page

4.4 comments xslt

By using comments xslt, we transform our comments.xml into xhtml(see figure 17).

comments

email	name	message
shixianwen1993@gmail.com	shixianwen	china airline is good
	wei liu	come back to home is exciting
	anonymous	china airline is bullshit
huanghuan@gmail.com	huanghuan	+ Pagination!
wen	shixian	231230
rwer	123	fdsfqe

Figure 17 comments.xlst

4.5 Error Page messages

Our application will generates some alerts because of the user's misbehavior such as do not type the required elements in the form or the username are already existed(see figure 3 and figure 4).

4.6 Flight search results

You can search flights from our page. For example if you can type the start point as Arlanda airport and the destination as London Healthrow airport. You can get a results shown in figure 18

company	name	price	from	des	flydate	flytime
British Airways	BA272	255$	Stockholm Arlanda Airport	London Heathrow Airport	2015/4/7	17:21
British Airways	BA274	100$	Stockholm Arlanda Airport	London Heathrow Airport	2015/4/7	14:21

go back to main page

Figure 18

5 Conclusions and further work

The project goal in developing Flightbook web application and relation database has been fully implemented. You can easily book your tickets and absorb the world news in our website. This project uses the XML technologies (XML,NAMESPACE,XPATH,XSLT,XSD,RSS, HTML,PHP,CSS, JavaScript,SQL, and MYSQL relational database. The overall aims purposed in the proposal were all implemented. However, Because of limited time, we save our xml in our local file without protection. Further goal is to use XML native database to save the XML files. The Application can easily access into the XML files by using the XML native database.

6 Gaps and similarities between xml technology and relational database models

Quoted from [3]

"The aim of RDBS is to store large amounts of data enabling efficient access and ensuring their consistency. XML is intended to serve as a format for structuring and exchanging hypertext documents."

Structure:

Similarities: "1:Both XML documents and relational schemata are element types and attributes for xml as well as relations and attributes for RDBS. 2: The name of an XML attribute defined within a DTD or an XML Schema has to be unique within its element type, again similar to an RDBS attribute's name which has to be unique within its relation."

Difference: "1:For each XML document, it is required that all component element types are rooted in a single element type. This is in contrast to RDBS, where part-of hierarchies cannot be realized by means of nesting since relations consist of atomic-valued attributes only.2: schema information of an explicit schema specification is replicated within XML documents in that each element and each attribute value is annotated with the corresponding element type name and attribute

name, respectively. The instance level of an RDBS is quite simpler, since values exclusively belong to attributes, which are in turn composed to tuples 3: Similar to RDBS, XML allows expressing null values as well as defaulting values."

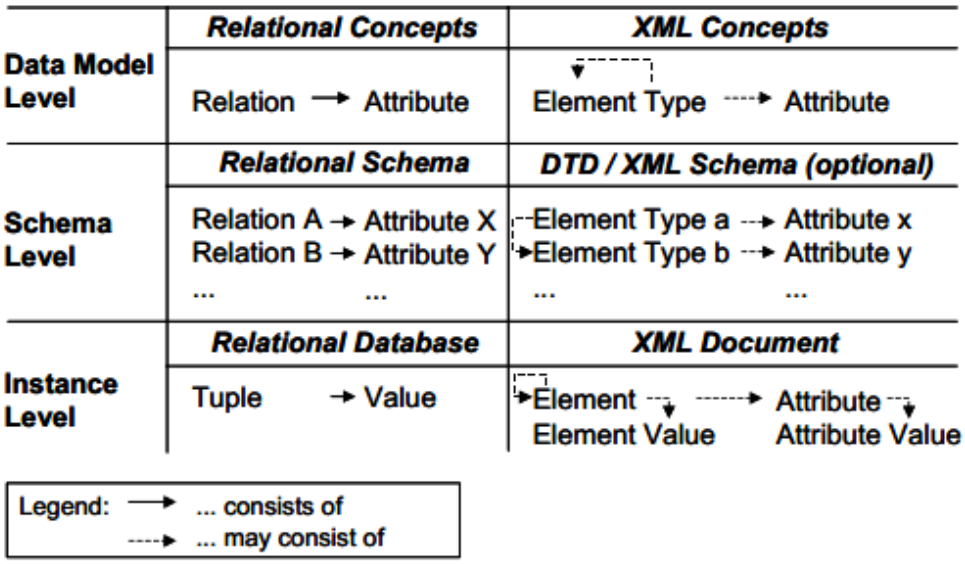

Fig. 1. Concepts at Different Levels of Abstraction

Storing and retrieving data by mapping:

Straightforward mapping: neither always possible nor desirable due to data model heterogeneity and schema heterogeneity.

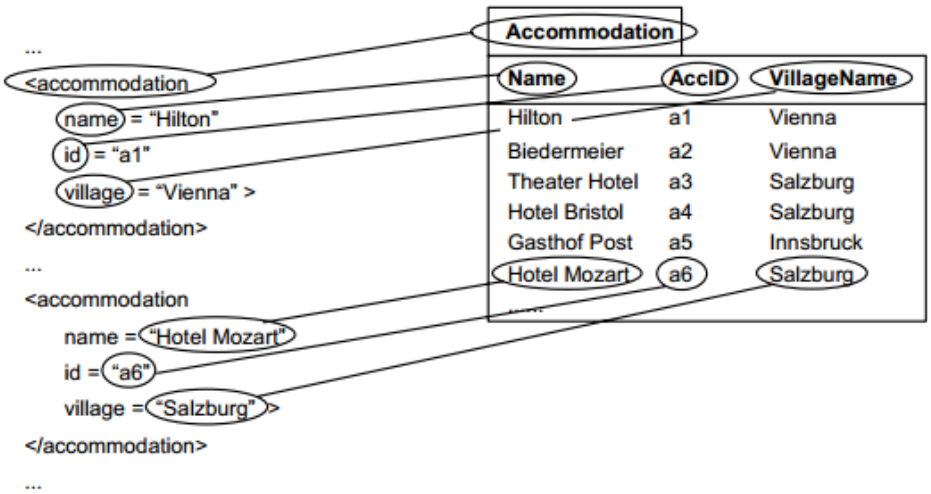

Fig. 7. Straightforward Mapping of XML Concepts to Relational Concepts

Basic kinds of mapping: have a base relation

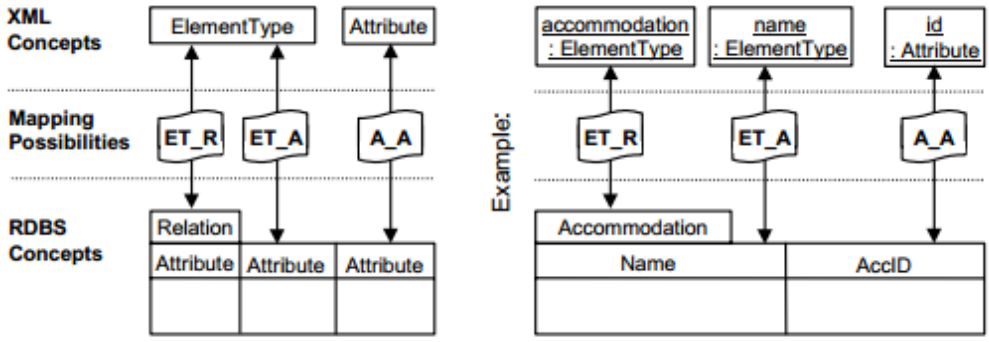

Fig. 8. Basic Kinds of Mappings

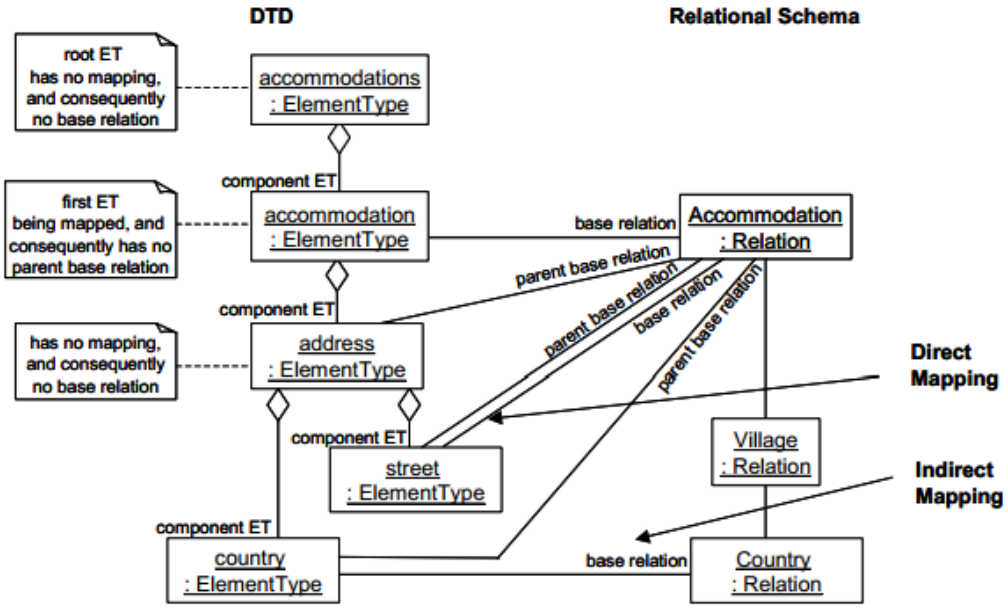

Fig. 9. Exemplary Mappings

Reasonable mapping:

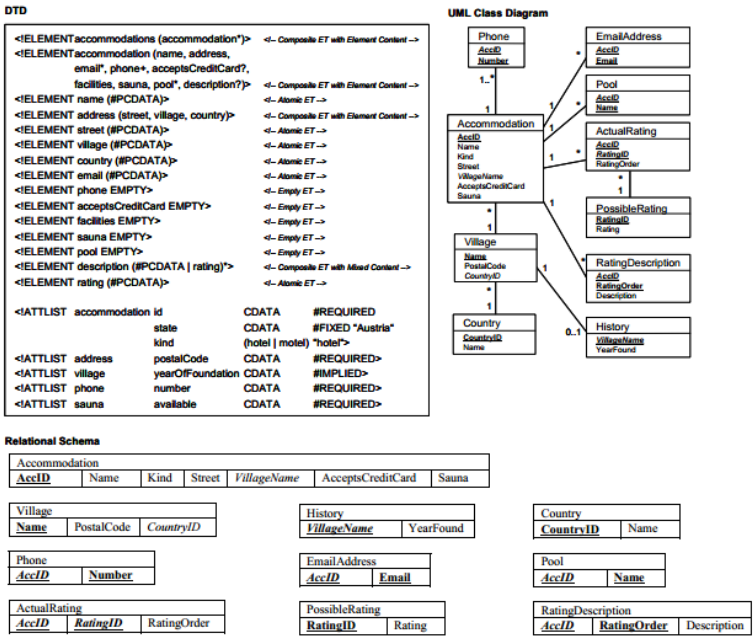

Fig. 10. Exemplary DTD, Relational Schema, and UML Class Diagram

7 References

[1] Wikipedia. http://en.wikipedia.org/wiki/Wiki

[2] W3School. http://www.w3schools.com/

[3] Kappel, Gerti, Elisabeth Kapsammer, and Werner Retschitzegger. "Integrating XML and relational database systems." *World Wide Web* 7.4 (2004): 343-384.