# VIRTUAL MUSEUM APPLICATION: DESIGN AND IMPLEMENTATION

BY

**LUMINIȚA SCRIPCARIU[*] and CRISTIAN BEJAN[1]**

"Gheorghe Asachi" Technical University of Iaşi
Faculty of Electronics,Telecommunication and Information Technology,
[1]IBM, Bucharest, Romania

**Abstract.** Virtual entities are commonly used nowadays. We designed and built an application for a virtual museum which facilitates access to art resources and easily finds authors and works. This application is built in PHP, with JavaScript functions, based on a relational database (DB) implemented in MySQL. It is an application based on a client-server model, accessible from any web browser (Internet Explorer, Mozilla Firefox, Opera, Google Chrome etc.), which can run on any computer or mobile device with Internet access.

**Key words:** database; MySQL; PHP; JavaScript; client-server model.

## 1. Introduction

Nowadays having Internet access is possible almost everywhere, at home, at the office, in public places such as parks, airports, subway stations and so on. Young people use Internet every time they need information rather than using books or other resources. Digital information has achieved a fundamental role in our knowledge society. Technology is continuously evolving. More and more devices are used to access the Internet (personal computers, laptops, notebooks, mobile phones, tablets, iPhone etc.) It is an opportunity for cultural institutions to promote their image through the Internet (Candela *et al.*, 2010).

_____

[*]Corresponding author: *e-mail*: lscripca@etti.tuiasi.ro

A classical museum can have more visitors if it has a web site with an attractive presentation of its patrimony.

Therefore the goal of our paper is to design and build an application for a virtual museum.

Distance will be no longer a problem to "visit" the museum, to view the artistic works, to read about the authors, to search a specific artistic current or historic era, representative artists, exhibits and specific artistic fields.

Many benefits can be obtained using a virtual museum application:

1° Documents and images can have a high definition.

2° The electronic zoom tool can be used to make a document more readable and to highlight specific details of a photo.

3° Fast search of a specific author and an exhibit can be made.

4° Locating the resources is facilitated running the application on a mobile device inside the real museum.

5° A lot of information about any author, exhibit, art school is offered.

6° More details can be obtained via e-mail, contacting a person from the museum.

First we analyze the specific resources of a museum and put these into categories. It is also important to establish the user categories and their rights on the application. The relational DB is designed as it is described in section **2**. The web application is presented in section **3**. Conclusions are written in section **4**.

## 2. Relational Database Design

The design process starts with the definition of entities and their attributes. The Entity-Relation Diagram (ERD) was drawn and normalized (Darwen, 2012; Elmasri & Navathe, 2007). Many-to-many (M:M) relations (between *artist* and *exhibit*, *artist* and *artistic_current*, *artist* and *art_school*, *artist* and *art_field*) were solved by intersecting the involved entities and creating new ones (called 'events') (Fig. 1).
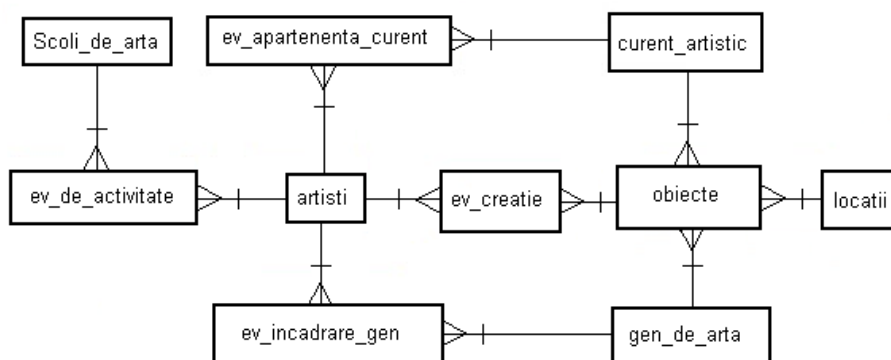


Fig. 1 – Entity-Relation Diagram.

Only one-to-one (1:1) and one-to-many (1:M) relationships are allowed.

Each entity is associated to a table and its attributes become the table columns. This mapping is made through the DB physical modeling process.

The resources of the virtual museum are stored in 10 tables named *Artists* (Fig. 2), *Artistic_Currents* (Fig. 3), *Current_Events* (Fig. 4), *Art_School* (Fig. 5), *Activity_Events* (Fig. 6), *Art_Fields* (Fig. 7), *Field_Events* (Fig. 8), *Exhibits* (Fig. 9), *Creation_Events* (Fig. 10) and *Locations* (Fig. 11).

The first attribute of each table is its primary key (PK). Relationships between the entities are mapped with foreign keys attributes (FK). Data types, dimensions and constraints are specified for each attribute. An attribute which must have unique value it is marked as UID (Unique Identifier).

| Chei | Atribut | Tip_date | Null | Default |
|------|---------|----------|------|---------|
| *PK* | *id_artist* | int(11) | No | |
| | **nume** | varchar(50) | No | |
| | **prenume** | varchar(50) | No | |
| | **prenume2** | varchar(50) | Yes | NULL |
| **UID** | **pseudonim** | varchar(50) | Yes | NULL |
| | **nationalitate** | varchar(60) | No | |
| | **tara_origine** | varchar(80) | Yes | NULL |
| | **an_nastere** | date | No | |
| | **an_deces** | date | Yes | NULL |
| | **an_debut** | date | Yes | NULL |
| **FK** | **id_maestru** | int(11) | Yes | NULL |
| | **loc_nastere** | varchar(100) | Yes | NULL |
| | **descriere** | text | Yes | NULL |

Fig. 2 – *Artists* table structure.

| Chei | Atribut | Tip_date | Null | Default |
|------|---------|----------|------|---------|
| *PK* | *id_curent* | int(11) | No | |
| | **nume** | varchar(50) | No | |
| | **data_incep** | date | Yes | NULL |
| | **data_sfarsit** | date | Yes | NULL |

Fig. 3 – *Artistic_Currents* table structure.

| Chei | Atribut | Tip_date | Null | Default |
|------|---------|----------|------|---------|
| PK | *id_ev* | int(11) | No | |
| | **fondator** | varchar(3) | No | |
| FK | **id_artist** | int(11) | No | |
| FK | **id_curent** | int(11) | No | |

Fig. 4 – *Current_Events* table structure.

An artist can simply belong to an artistic current or he/she can be its founder, so we introduce a special attribute ("fondator"), with two possible values (YES/NO) in *Current_Events* table.

| Chei | Atribut | Tip | Null | Default |
|------|---------|-----|------|---------|
| *PK* | id_scoala | int(11) | No | |
| | nume | varchar(100) | No | |
| | locatie | varchar(50) | No | |
| | data | date | Yes | NULL |

Fig. 5 – *Art_Schools* table structure.

An artist can be student at an Art School at a moment but he / she can also be a teacher at a school. So the attribute "calitate", used in *Activity_Events* table, specifies this quality of an artist at a specific date.

| Chei | Atribut | Tip | Null | Default |
|------|---------|-----|------|---------|
| *PK* | id_ev | int(11) | No | |
| | data | date | No | |
| | calitate | varchar(12) | No | |
| FK | id_scoala | int(11) | No | |
| FK | id_artist | int(11) | No | |

Fig. 6 – *Activity_Events* table structure.

Different visual art fields are included in the database (painting, sculpture, photography, graphic design, ceramics, religious art, tapestry art, video art, caligraphy and more others).

| Chei | Atribut | Tip | Null | Default |
|------|---------|-----|------|---------|
| *PK* | id_gen | int(11) | No | |
| UID | nume | varchar(12) | No | |
| UID | descriere | text | Yes | |
| | data | date | Yes | NULL |

Fig. 7 – *Art_Fields* table structure.

| Chei | Atribut | Tip | Null | Default |
|------|---------|-----|------|---------|
| *PK* | id_ev_gen | int(11) | No | |
| FK | id_artist | int(11) | No | |
| FK | id_gen | int(11) | No | |
| | data | date | Yes | NULL |

Fig. 8 – *Field_Events* table structure.

An artist can have works in different art fields, such as Leonardo da Vinci which acts as a painter, sculptor and architect, if we consider only the visual arts. So, we will record him three times, in *Type_Events* table.

A very important entity is *Exhibit* ("obiecte")*, with many attributes, which is mapped as a very large table. Four foreign keys are used to map the relationships of this table with others as it is drawn in the entity-relation diagram.

| Chei | Atribut | Tip | Null | Default |
|------|---------|-----|------|---------|
| *PK* | id_exponat | int(11) | No | |
| | nume | varchar(50) | No | |
| | tip | varchar(30) | No | |
| FK | id_gen | int(11) | No | |
| FK | id_curent | int(11) | No | |
| FK | id_ev_creatie | int(11) | No | |
| UID | imagine | BLOB | Yes | NULL |
| | descriere | text | Yes | NULL |
| | mod_prezentare | text | Yes | NULL |
| | valoare | int(11) | Yes | NULL |
| | material | varchar(50) | No | |
| | limba | varchar(50) | Yes | NULL |
| | tara | varchar(50) | Yes | NULL |
| | localitate | varchar(50) | Yes | NULL |
| | colectionar | varchar(100) | Yes | NULL |
| UID | isbn | varchar(13) | Yes | NULL |
| FK | id_locatie | int(11) | No | |

Fig. 9 – *Exhibits* table structure.

| Chei | Atribut | Type | Null | Default |
|------|---------|------|------|---------|
| *PK* | id_ev_creatie | int(11) | No | |
| FK | id_artist | int(11) | No | |
| FK | id_exponat | int(11) | No | |

Fig. 10 – *Creation_Events* Table Structure

| Chei | Atribut | Tip | Null | Default |
|------|---------|-----|------|---------|
| *PK* | id_locatie | int(11) | No | |
| | denumire | varchar(50) | Yes | |
| | adresa | varchar(50) | No | |
| | etaj | int(11) | No | |
| | camera | int(11) | No | |
| | pozitie | int(11) | No | |

Fig. 11 – *Locations* table structure.

A special table is created for security reasons, named *Users* (Fig. 12). So the database (DB) manages the resources of the virtual museum (photos, audio files, documents) and also the user accounts. A visitor account with read-only privileges is created. Passwords are recorded encrypted with a hash function in order to block stealing attempts.

| Chei | Atribut | Type | Null | Default |
|------|---------|------|------|---------|
| *PK* | id | int(11) | No | |
| | username | varchar(30) | No | |
| | hashed_password | varchar(40) | No | |

Fig. 12 – *Users* table structure.

## 3. Web Application Description

The website design is essential to increase the interest of the visitors for this application.

In this section we use PHP, MySQL and JavaScript languages to build the application for a virtual museum as a dynamic website (Nixon, 2014).

After registration and log in operation, visitors can access the Virtual Museum web application to visualize the exhibits, to find information about the artists, to locate objects and so on.

Database administrators can record data, upload files on the server, update and delete information, control the access to the application.

Data and files are organized in directories (Fig. 13): *classes* directory includes all defined classes for this website; *images* directory contains all image files used on the site; *photos* includes all photos of the exhibits presented in the virtual museum; *includes* directory contains the .php files used as templates on the site pages and some functions used by the application; *stylesheets* contains the .css files (*Cascading Style Sheet*) used for the layout of the website; *javascripts* contains some javascript functions used to check different things such as taped characters into a text box, string length, security level etc.

All these directories are included in the main directory, named *muzeu*, which contains all the .php files used for  man-to-DB interaction.



Fig. 13 – Directories Physical Structure.

The website map is given in Fig. 14. It includes several pages:

a) *index.php* – allows the login process (Fig. 15). Each user has an account and the authentication is made based on the username and a secret password. Error checking the account credentials produces an error message for the user (Fig. 16). The logout process is ended with a message (Fig. 17).

b) *administrare.php* – This is accessible for the administrator account (Fig. 18). An administrator has access to many menus which allows him to insert and update the DB content (artists, exhibits, locations, art fields, art schools, art currents), including the administration of the user accounts. This page is linked to other pages (*new_artist.php, new_obiect.php* etc.) Specific forms are created for data editing and updating.

c) *user.php* – manages the of users' access to the DB, including users' passwords.

d) *cauta.php* – creates an HTML form which allows searching based on different criteria (Fig. 19). Some search results are given in Fig. 20. Edit and delete records can be made from this page (Fig. 21).
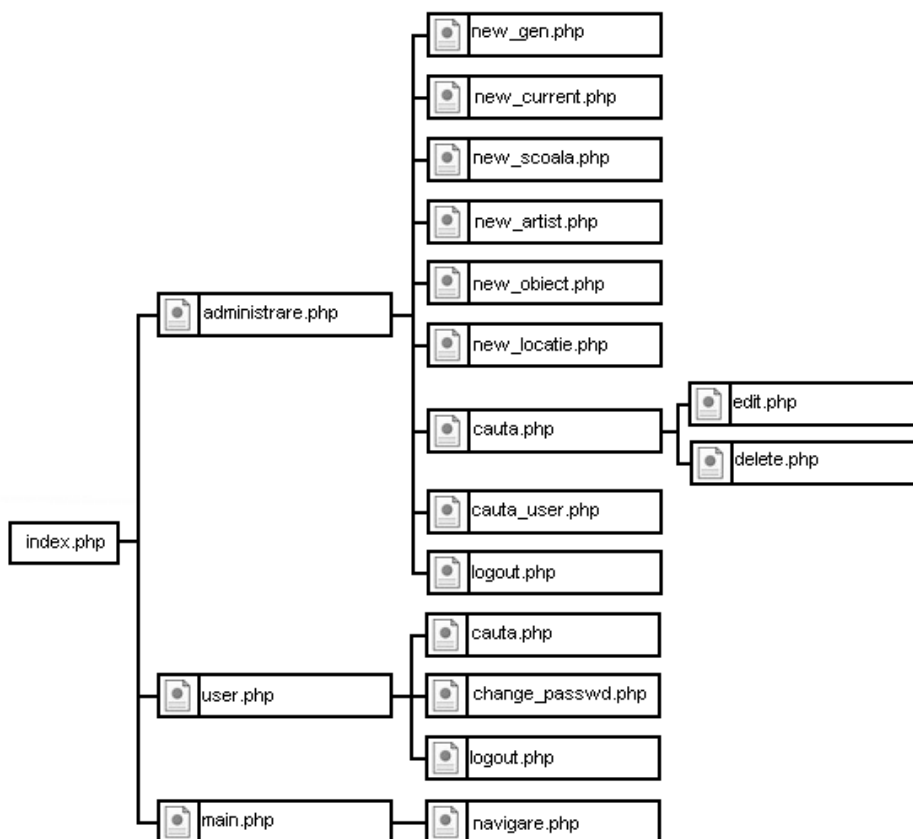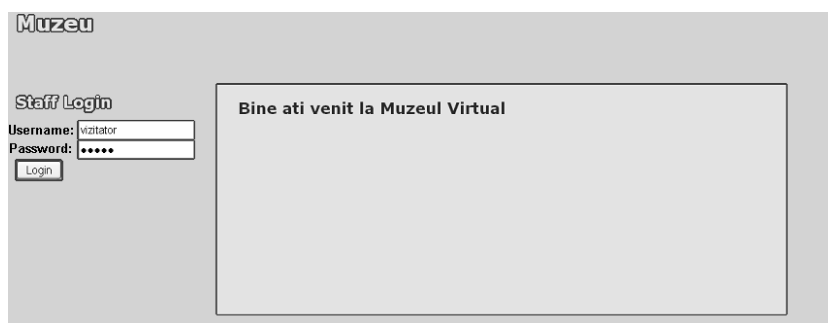


Fig. 14 – Website map.



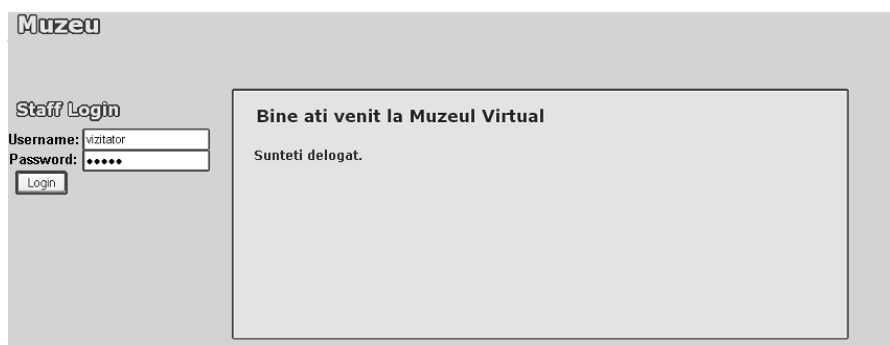Fig. 15 – Login window.

Fig. 16 – Incorect login message.


Fig. 17 – Logout window.


Fig. 18 – Administration page.
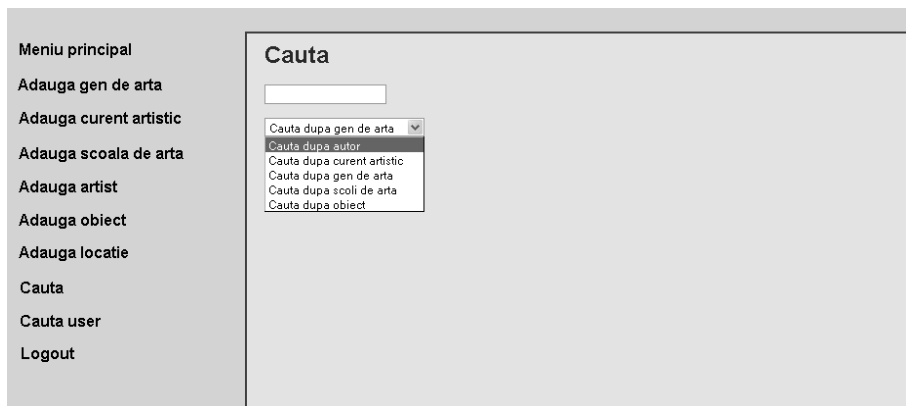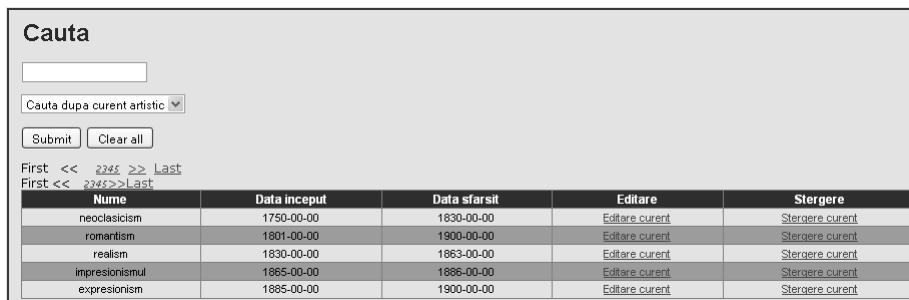
Fig. 19 – Search page.


Fig. 20 – Some search results.


Fig. 21 – Update form.

The user requests are transmitted through the web server to the DB server based on the client-server model. For a large number of simultaneous requests, the web server and the DB server must have a broadband communication channel.

## 4. Conclusion

We designed the database for a virtual museum and we built the web application which can be accessed using any web browser. MySQL, PHP and Javascript languages were used. Different kinds of queries on the DB are possible about artists, exhibits, art schools, art currents, art fields etc. The security is essential so different user accounts with different privileges are created and managed. This web application with a complete database is a useful tool for public in general and an educational tool for people wanting to access systematically the museum resources.

## REFERENCES

Candela L., Castelli D., Fox E.A., Ioannidis Y., *On Digital Library Foundations*. Internat. J. on Digital Libraries, Springer-Verlag, **11**, *1*, London, UK, 37-39 (2010).

Darwen H., *An Introduction to Relational Database Theory*. Hugh Darwen & Ventus Publ. ApS, London, UK, 2012.

Elmasri R., Navathe S.B., *Fundamentals of Database Systems*. 5th Edition, Addison-Wesley, Boston, USA, 2007.

Nixon R., *Learning PHP, MySQL, JavaScript, CSS & HTML5*. 3rd Edition, O'Reilly Media, Inc., Canada, 2014.

APLICAȚIA MUZEU VIRTUAL: PROIECTARE ȘI IMPLEMENTARE

(Rezumat)

Aplicațiile în mediul virtual sunt foarte folosite în prezent. În această lucrare, am proiectat și implementat o aplicație pentru un muzeu virtual care permite găsirea, într-un mod eficient, a informațiilor despre autori și operele lor. Aplicația este realizată în limbaj PHP cu funcții JavaScript, fiind bazată pe o bază de date relațională implementată în MySQL. Este o aplicație de tip client-server, accesibilă din orice browser de Internet (Internet Explorer, Mozilla Firefox, Opera, Google Chrome etc.), care poate fi rulată de pe orice calculator sau dispozitiv mobil cu acces la Internet.