Proceedings of Student-Faculty Research Day, CSIS, Pace University, May 4th, 2012

# Web Interfaces to Backend Databases Using PHP and MySQL

Matt Rotundo, Arun Jose, Brian Richards, Patrick Stankard, Diah Schur
Andriyan Leo, Fabio Garcia, Xu Yang, and Allen Stix
*Pace University Seidenberg School of CSIS, White Plains, NY 10606, USA*

## Abstract

*Interfaces to backend databases are examined using PHP and MySQL. A case study develops an easy-to-use interface to add, delete, and modify tables and queries for the DPS dissertation database system at Pace University. This interface will play a key role in enhancing the overall usability of editing information for DPS dissertations, increase performance, and productivity for the Pace University community.*

## 1. Introduction

Currently there are about 70 complete dissertations in the DPS dissertation database. A web interface was created, with PHP logic and SQL, to help manage the DPS dissertation database content for the existing dissertation application. In this database, the dissertations are categorized into different types and present all the associated information in each of them in one common location. The existing DPS database was therefore modified in such a way that the interface could be used to not only add but also delete or even modify the tables as well as the queries present in the database using PHP logic. The goals for this development were:

- Create an easy to use interface to edit tables and queries through a web interface.
- Create an interface that is able to add, delete, and modify queries and tables.
- Creating an interface that satisfies customers' needs.
- Enhance overall usability interface by making database modifications easier.

## 2. Database design

### 2.1. Database model specifications

Data model simply means a technique used to describe data structures as well as the various operations that can be used to validate and manipulate the same data. The repository of this database uses "My SQL" database to function as its repository. Through a discussion with the stakeholder, the information that needed to be stored in the database was identified and sieved. The original database simply had three tables. To enhance flexibility and to ensure that web-based administration is enabled, the original version of the database was altered in such a way that normalization was done using the 1st normal form, (1 NF), the 2nd normal form (2 NF) as well as the 3rd normal form (3rd NF).

The normalized version of the database is presented with 7 tables organized in 3 views. The primary keys as well as the foreign keys dominate most of the tables of the new look database. Whenever any new record is entered in the primary column, an auto increment feature executes itself.

Although character based columns can be used as primary key columns, we identified potential risks in terms of performance and data integrity with that design. So, we replaced character based primary key columns in the original logical design with numeric columns. For convenience, these new identity columns are setup to auto increment when a new record is inserted.

To enforce referential integrity, foreign keys were added wherever necessary. With the new database design, any number of committee members can be assigned to a dissertation or to an external publication. This was achieved by using junction tables connecting dissertations or external publications to their respective committee members. Currently, reports in the system assume a maximum of five committee members in dissertations and external publications.

Figure 1 below is a diagrammatic representation of the ER diagram for the dissertation database.

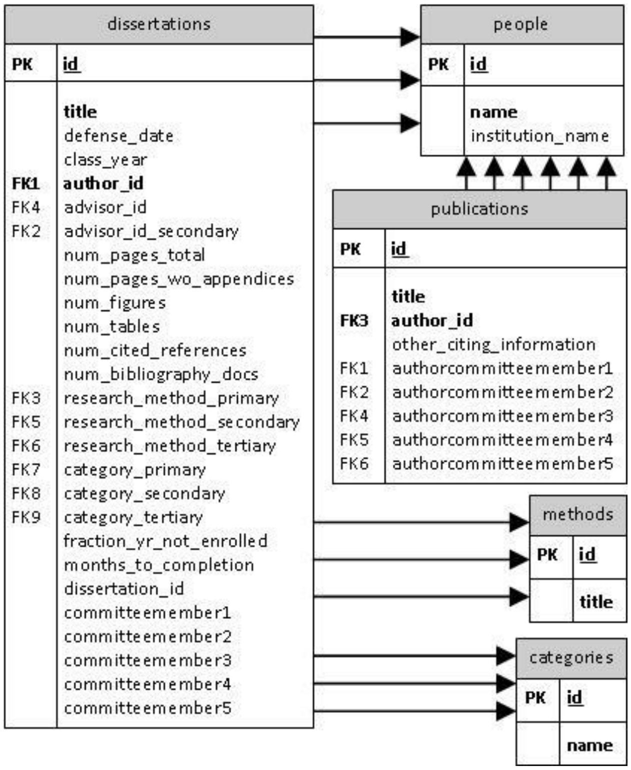


**Figure 1. ERD of DPS dissertation database**

## 2.2. Tables

A look at the schematic diagram shows seven tables in this new system. The tables are described as highlighted below:

i. category

This table is critical since it functions as a lookup table for the various dissertation categories. The information presented here is therefore crucial and can be utilized elsewhere with the use of foreign key that makes reference to the column with the “id.

| Column | Type | Null | Default | Keys |
|---|---|---|---|---|
| id | int(11) | N | | Primary Key, Auto Increment |
| name | varchar(255) | N | | |

ii. method

The “method” table also functions as a lookup table only that this time round, it is used for dissertation methods. The primary key used here is the “id” column that is referenced in the other tables marked with foreign keys.

| Column | Type | Null | Default | Key |
|---|---|---|---|---|
| id | int(11) | No | | Primary Key, Auto Increment |
| title | varchar(255) | No | | |

iii. person

This table holds the critical information on different individuals captured (stakeholders). Logically a person captured here can be of any identity ranging from being an author, advisor, or even member of a specific committee. In this case primary key column is the “id” which is preset to execute auto increment whenever a new “person” is added or alterations made

| Column | Type | N | Defa | Keys |
|---|---|---|---|---|
| id | int(11) | N | | Primary Key, Auto Increment |
| name | varchar(255) | N | | |
| institution | varchar(45) | Y | *NULL* | |

iv. dissertations

This functions as the system’s main table. It holds information on the available dissertations. “id” remains the primary key in this case, and several references exist linking other tables using foreign keys.

| Column | Type | Null | Default | Keys |
|---|---|---|---|---|
| id | int(11) | No | | Primary Key, Auto Increment |
| title | varchar(100) | No | | |
| defense_date | date | Yes | *NULL* | |
| class_year | int(11) | Yes | *NULL* | |
| author_id | int(11) | No | *NULL* | Foreign Key to person(id) |
| advisor_id | int(11) | Yes | *NULL* | Foreign Key to person(id) |
| advisor_id_secondary | int(11) | Yes | *NULL* | Foreign Key to person(id) |
| num_pages_total | int(11) | Yes | *NULL* | |
| num_pages_wo_appendices | int(11) | Yes | *NULL* | |
| num_figures | int(11) | Yes | *NULL* | |
| num_tables | int(11) | Yes | *NULL* | |
| num_cited_references | int(11) | Yes | *NULL* | |
| num_bibliography_docs | int(11) | Yes | *NULL* | |
| fraction_yr_not_enrolled | int(11) | Yes | *NULL* | |
| months_to_completion | int(11) | Yes | *NULL* | |
| research_method_primary | int(11) | Yes | *NULL* | Foreign Key to method(id) |
| research_method_secondary | int(11) | Yes | *NULL* | Foreign Key to method(id) |
| research_method_tertiary | int(11) | Yes | *NULL* | Foreign Key to method(id) |
| category_primary | int(11) | Yes | *NULL* | Foreign Key to category(id) |
| category_secondary | int(11) | Yes | *NULL* | Foreign Key to category(id) |
| category_tertiary | int(11) | Yes | *NULL* | Foreign Key to category(id) |

v. dissertation_committee

This acts as a junction table whose main function is to link the committee members to the available dissertations. It enables the establishment of a matrix of several relationships. This therefore enables easy adding of several committee members to the available dissertation. This table however does not have a primary key. This is a junction table that relates committee members to dissertations.

| Column | Type | Null | Default | Keys |
|---|---|---|---|---|
| dissertation_id | int(11) | No | | Foreign Key to dissertations(id) |
| committee_member_id | int(11) | No | | Foreign Key to person(id) |

vi. external_publication

This table has data that deals with all the external publications. It also has a primary key as "id".

| Column | Type | Null | Default | Keys |
|---|---|---|---|---|
| id | int(11) | No | | Primary Key, Auto Increment |
| title | varchar(255) | No | | |
| author_id | int(11) | No | | Foreign Key to person(id) |
| other_citing_information | text | Yes | *NULL* | |

vii. external_publication_committee

This table is also a junction table that creates a linkage platform for external publications to the committee members. Any number of existing committee members in the database can be assigned to an external publication. Flexibility is the main intention to be achieved here.  This table does not have a primary key..

| Column | Type | Null | Default | Keys |
|---|---|---|---|---|
| external_publication_id | int(11) | No | | Foreign Key to external_publication(id) |
| committee_member_id | int(11) | No | | Foreign Key to person(id) |

## 2.3. Views

Looking at this through the perspective of SQL, views mean the virtual tables that are pegged on SQL statement of result-sets. Just like a real table, it has columns and rows. In the dissertation database, there are 3 views. This makes it easy to simply the multiple tables into the desired virtual tables.

i. committeemembers

Through the front end, this view provides for a comprehensive list of committee members.

| Column | Type |
|---|---|
| Name | varchar(255) |
| Institution | varchar(45) |

ii. dissertation

This is achieved by making it possible to link several tables in the database and hence and therefore acts as a flat representation of all the available dissertations.

| Column | Type |
|---|---|
| authorid | int(11) |
| author | varchar(255) |
| classyear | int(11) |
| fractionofy | int(11) |
| dateofsucce | date |
| monthstoco | int(11) |
| dissertation | varchar(100) |
| committee | varchar(255) |
| committee | varchar(255) |
| committee | varchar(255) |
| committee | varchar(255) |
| committee | varchar(255) |
| earlieradvis | varchar(255) |
| earlieradvis | varchar(255) |
| primarysubj | varchar(255) |
| secondarys | varchar(255) |
| tertiarysubj | varchar(255) |
| primarymet | varchar(255) |
| secondarym | varchar(255) |
| tertiarymet | varchar(255) |
| numberofpa | int(11) |
| numberofpa | int(11) |
| numberoffi | int(11) |
| numberofta | int(11) |
| numberofn | int(11) |
| numberofbi | int(11) |
| titleofexter | varchar(255) |
| titleofexter | varchar(255) |

iii. externalpublication

The "external publication" provides the virtual tables with the relevant information they need. The virtual tables also get information from 'committee members' as well as from 'person tables'. In the end the external publication provides an aggregate representation of all the external publications.

| Column | Type |
|---|---|
| titleofexternalpublication | varchar(255) |
| author | varchar(255) |
| authorcommitteemember1 | varchar(255) |
| authorcommitteemember2 | varchar(255) |
| authorcommitteemember3 | varchar(255) |
| authorcommitteemember4 | varchar(255) |

| Column | Type |
|---|---|
| authorcommitteemember5 | varchar(255) |
| othercitinginformation | text |

## 2.4. Database installation

To install the database, the MySQL community version is freely available from MySQL website.

Normally, the MySQL installation package comes as a zip file and is installed by double clicking on it. Normally, inside the zip archive, there would be a file entitled ‘setup.exe’ Then, follow the rest of the instructions by choosing the defaults.

Once the installation is complete, you will be presented with a "MySQL Sign-Up" window. Click the radio button "Skip Sign-Up", and then click "Next>". For further details of MySQL installation instructions, please go to http://www.wikihow.com/Install-the-MySQL-Database-Server-on-Your-Windows-PC

## 2.5. Database and table creation steps

1. Open MySQL Workbench
2. Double click on “Local instance mysql”
3. At the top screen, there is a window section called, Query 1. Copy and paste the entire SQL command given below into Query 1. Then, click execute button ⚡

```
CREATE TABLE person(
   id int(11) NOT NULL AUTO_INCREMENT,
   name varchar(255) NOT NULL,
   institution varchar(45) DEFAULT NULL,
   PRIMARY KEY (id)
) ENGINE=InnoDB AUTO_INCREMENT=112 DEFAULT
CHARSET=latin1$$

CREATE TABLE category(
   id int(11) NOT NULL AUTO_INCREMENT,
   name varchar(255) NOT NULL,
   PRIMARY KEY (id)
) ENGINE=InnoDB AUTO_INCREMENT=35 DEFAULT
CHARSET=latin1$$

CREATE TABLE method(
   id int(11) NOT NULL AUTO_INCREMENT,
   title varchar(255) NOT NULL,
   PRIMARY KEY (id)
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT
CHARSET=latin1$$

CREATE TABLE dissertations(
   id int(11) NOT NULL AUTO_INCREMENT,
   dissertation_title varchar(100) NOT NULL,
   defense_date date DEFAULT NULL,
   class_year int(11) DEFAULT NULL,
   author_id int(11) NOT NULL,
   advisor_id int(11) DEFAULT NULL,
   advisor_id_secondary int(11) DEFAULT NULL,
   num_pages_total int(11) DEFAULT NULL,
   num_pages_wo_appendices int(11) DEFAULT NULL,
   num_figures int(11) DEFAULT NULL,
   num_tables int(11) DEFAULT NULL,
   num_cited_references int(11) DEFAULT NULL,
   num_bibliography_docs int(11) DEFAULT NULL,
   fraction_yr_not_enrolled int(11) DEFAULT NULL,
   months_to_completion int(11) DEFAULT NULL,
   research_method_primary int(11) DEFAULT NULL,
   research_method_secondary int(11) DEFAULT NULL,
   research_method_tertiary int(11) DEFAULT NULL,
   category_primary int(11) DEFAULT NULL,
   category_secondary int(11) DEFAULT NULL,
   category_tertiary int(11) DEFAULT NULL,
   PRIMARY KEY (id),
   KEY d_fk_1 (author_id),
   KEY d_fk_2 (advisor_id),
   KEY d_fk_3 (advisor_id_secondary),
   KEY d_fk_4 (research_method_primary),
   KEY d_fk_5 (research_method_secondary),
   KEY d_fk_6 (research_method_tertiary),
   KEY d_fk_7 (category_primary),
   KEY d_fk_8 (category_secondary),
   KEY d_fk_9 (category_tertiary),
   CONSTRAINT d_fk_1 FOREIGN KEY (author_id)
      REFERENCES person (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT d_fk_2 FOREIGN KEY (advisor_id)
      REFERENCES person (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT d_fk_3 FOREIGN KEY (advisor_id_secondary)
      REFERENCES person (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT d_fk_4 FOREIGN KEY (research_method_primary)
      REFERENCES method (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT d_fk_5 FOREIGN KEY
(research_method_secondary)
      REFERENCES method (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT d_fk_6 FOREIGN KEY (research_method_tertiary)
      REFERENCES method (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT d_fk_7 FOREIGN KEY (category_primary)
      REFERENCES category (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT d_fk_8 FOREIGN KEY (category_secondary)
      REFERENCES category (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT d_fk_9 FOREIGN KEY (category_tertiary)
      REFERENCES category (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=135 DEFAULT
CHARSET=latin1$$

CREATE TABLE dissertation_committee(
   dissertation_id int(11) NOT NULL,
   committee_member_id int(11) NOT NULL,
   KEY dc_fk_1 (committee_member_id),
   KEY dc_fk_2 (dissertation_id),
   CONSTRAINT dc_fk_1 FOREIGN KEY (committee_member_id)
      REFERENCES person (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT dc_fk_2 FOREIGN KEY (dissertation_id)
      REFERENCES dissertations (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$

CREATE TABLE external_publication(
   id int(11) NOT NULL AUTO_INCREMENT,
   title varchar(255) NOT NULL,
   author_id int(11) NOT NULL,
   other_citing_information text,
   PRIMARY KEY (id),
   KEY ep_fk_1 (author_id),
   CONSTRAINT ep_fk_1 FOREIGN KEY (author_id)
      REFERENCES person (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=43 DEFAULT
CHARSET=latin1$$

CREATE TABLE external_publication_committee(
   external_publication_id int(11) NOT NULL,
   committee_member_id int(11) NOT NULL,
   KEY epc_fk_1 (external_publication_id),
   KEY epc_fk_2 (committee_member_id),
   CONSTRAINT epc_fk_1 FOREIGN KEY (external_publication_id)
      REFERENCES external_publication (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT epc_fk_2 FOREIGN KEY (committee_member_id)
      REFERENCES person (id)
      ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1$$
```

## 3. Web interface design

### 3.1. PHP implementation

The application is built using a Model-View-Controller paradigm, wherein separation is maintained between logic that affects the data objects, the core application functionality and the display generation.

Application File Structure:

- _ajax
  - autocomplete.php
- _controllers
  - authentication.php
  - database.php
  - output.php
  - validation.php
- _objects
  - _object_base.php
  - category.php
  - dissertation.php
  - externalPublication.php
  - method.php
  - person.php
- _include
  - core.php
  - header.php
  - footer.php
- assets
  - admin.css
- dissertation
  - create.php
  - delete.php
  - form.php
  - manage.php
- external_publication
  - create.php
  - delete.php
  - form.php
  - manage.php
- person
  - create.php
  - delete.php
  - form.php
  - manage.php
- home.php
- index.php

#### 3.1.1. Script structuring

Footer and header scripts are necessary for every view with the main purpose of making them work properly. This should be done at the beginning and the end respectively in the file. “The scripts encapsulate the page in the HTML template along with load all of the application functionality included in (include/main.php). It is important to use this methodology in order to centralize the location of the HTML template (including it in only one header/footer file) and to ensure that the same core application logic permeates through all views”..

Core Script Structure

```
//begin session
session_start();

//global variable setup
$db_host = 'localhost';
$db_user = 'cs691-team03-2012';
$db_pass = 'xxxxxxxxxxxx';
$db_name = 'cs691-team03-2012_development';

$url_root = '/~cs691-team03-2012/admin';

$site_root = '/home/cs691-team03-2012/public_html/admin';

//include all application scripts
include $site_root.'/_controllers/authentication.php';
include $site_root.'/_controllers/database.php';
include $site_root.'/_controllers/output.php';

include $site_root.'/_objects/_object_base.php';
include $site_root.'/_objects/dissertation.php';
include $site_root.'/_objects/externalPublication.php';
include $site_root.'/_objects/person.php';
include $site_root.'/_objects/method.php';
include $site_root.'/_objects/category.php';

//load and setup core functionality
$db = new database();

//connect and select database
$db->connect($db_host,$db_user,$db_pass);
$db->setDB($db_name);

//check for authentication status
$auth = new authentication();

$auth->check();

//load data objects
$dissertation          = new dissertation();
$externalPublication    = new externalPublication();
$person                = new person();
$method                = new method();
$category              = new category();
```

### 3.1.2.Objectclasses

The abstraction level required between core application and database interaction is brought well using the object classes. The object ‘base class’ forms the foundation of functionality for all objects.

- object_base
  - var fields
  - var idfield
  - var lookupField
  - var tableName
  - function __construct()
  - function create()
  - function modify()
  - function delete()
  - function lookup()
  - function autocomplete()
  - function get()
  - function listing()
  - function getName()
- category
  - function __construct()
- dissertation
  - function __construct()
  - function retrieveCommittee()
  - function manageCommittee()
- externalPublication
  - function __construct()
  - function retrieveCommittee()
  - function manageCommittee()
- method
  - function __construct()
- person
  - function __construct()

### 3.1.3. Object constructor setup

Each object class’ constructor for each object class is used so as to provide the base object class with the data it needs to influence the correct tables

```
    function                                        __constructor()
{
     $this->fields                                               =
array('id','title','author_id','other_citing_information','disse
rtation_id');
     $this->idField              =                           'id';
     $this->lookupField           =                        'title';
     $this->tableName        =         'external_publicaton';
}
```

### 3.1.4. Object class extensions

Normally, the base ‘create’, ‘modify’, ‘delete’, ‘get’ methods require extension and hence necessitating the need for an advanced functionality.

Through the controller classes, collections needed for core functionality applications are availed. This serves to unify the code bases and also allows for the ready extension of the applications.

- authentication - All authentication functionality
  - check()
  - login()
  - logout()
- database
  - connect()
  - setDB()
  - query()
  - clean()
  - getRow()
  - getResultArray()
  - insert()
  - delete()
  - update()
  - lastID()
  - numRows()
  - whereBuilder()
- output - To be used for output buffering and meta information management.
  - **not currently implemented**

- validation - To be used for input validation.
    - **not currently implemented**

## 3.2. Front end implementation

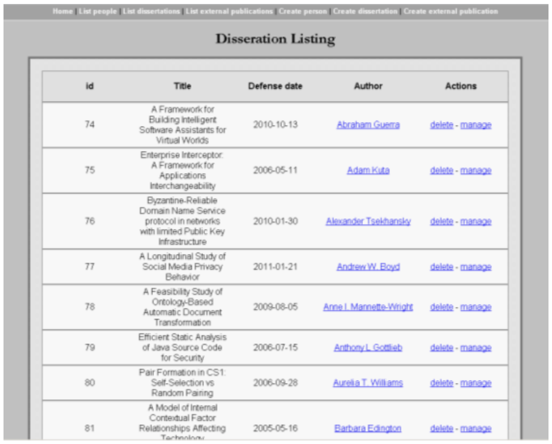
It is always expected that HTML provides an illustrative layout, it was therefore very critical that coding be done to enable administration views. A total of seven pages were created comprising of one log in page, 3 view pages with "person, dissertation, and external publication". Html was used in creating forms in regards to coding of the edit pages. PHP was used to process the data entered in the various fields available and then stored in the database in the form of My SQL.

All the information gathered from the original database were inserted in the 'easy to read table' using HTML. This application was also used not only for data entry for the languages but also for purposes of displaying the content to the users.

In creating a visually appealing page that would excite the users, the CSS code was utilized and the results were pleasing to the users. The use of CSS was chosen because it has the merit of allowing the website designer to format the web pages and organize them as he/she deems fit. In this project, CSS therefore helped in the creation of all the link areas, the forms, tables as well as the colors.

After establishing the field requirements for each web page, HTML was utilized for purposes of generating the 'edit' views. For instance, in the page "Create person", only two fields were provided for but in the page "Create external publication", seven fields were created.

Just like the 'listing views', the whole process of development was just similar. It was however necessary that the specific titles and the number of rows that each table required had to be defined prior to creating the remaining pages. Figure 2 below shows a sample "Dissertation Listing" page.

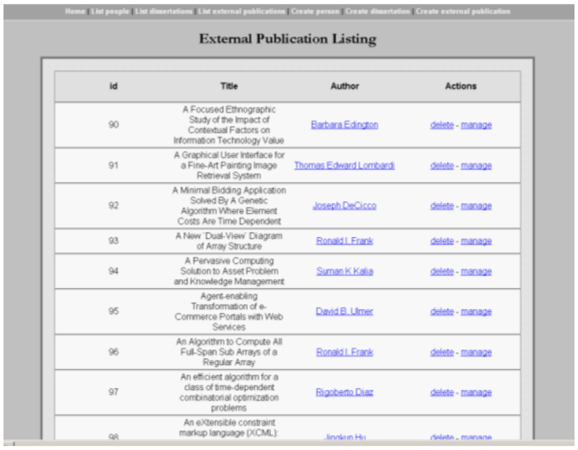

Home | List people | List dissertations | List external publications | Create person | Create dissertation | Create external publication

Disseration Listing

| id | Title | Defense date | Author | Actions |
|---|---|---|---|---|
| 74 | A Framework for Building Intelligent Software Assistants for Virtual Worlds | 2010-10-13 | Abraham Guerra | delete - manage |
| 75 | Enterprise Interceptor: A Framework for Applications Interchangeability | 2006-05-11 | Adam Kuta | delete - manage |
| 76 | Byzantine-Reliable Domain Name Service protocol in networks with limited Public Key Infrastructure | 2010-01-30 | Alexander Tsekhansky | delete - manage |
| 77 | A Longitudinal Study of Social Media Privacy Behavior | 2011-01-21 | Andrew W. Boyd | delete - manage |
| 78 | A Feasibility Study of Ontology-Based Automatic Document Transformation | 2009-08-05 | Anne I. Mannette-Wright | delete - manage |
| 79 | Efficient Static Analysis of Java Source Code for Security | 2006-07-15 | Anthony L. Gottlieb | delete - manage |
| 80 | Pair Formation in CS1: Self-Selection vs Random Pairing | 2006-09-28 | Aurelia T. Williams | delete - manage |
| 81 | A Model of Internal Contextual Factor Relationships Affecting Technology | 2005-05-16 | Barbara Edington | delete - manage |

**Figure 2. Dissertation Listing**

Implementing the auto-completion, with the titles 'author', 'committee member', 'advisor', 'category', 'method' and 'external publication' fields, required the utilization of Java Script hence this project utilized jQuery and the jQuery UI libraries. These were very instrumental in creating and managing the dissertations. Figure 3 below shows "External Publication Listing" which uses jQuery UI libraries.

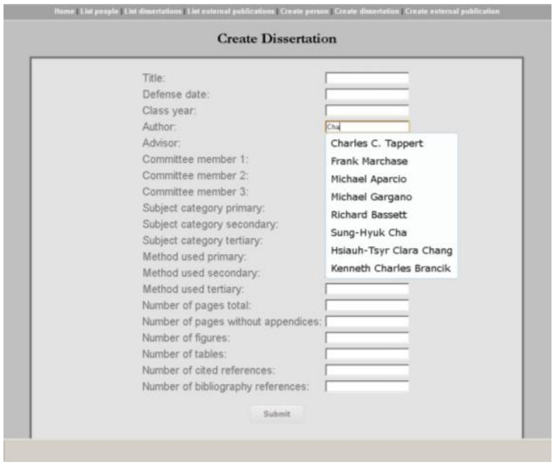

Home | List people | List dissertations | List external publications | Create person | Create dissertation | Create external publication

External Publication Listing

| id | Title | Author | Actions |
|---|---|---|---|
| 90 | A Focused Ethnographic Study of the Impact of Contextual Factors on Information Technology Value | Barbara Edington | delete - manage |
| 91 | A Graphical User Interface for a Fine-Art Painting Image Retrieval System | Thomas Edward Lombardi | delete - manage |
| 92 | A Minimal Bidding Application Solved By A Genetic Algorithm Where Element Costs Are Time Dependent | Joseph DeCicco | delete - manage |
| 93 | A New 'Dual-View' Diagram of Array Structure | Ronald I. Frank | delete - manage |
| 94 | A Pervasive Computing Solution to Asset Problem and Knowledge Management | Suman K Kalia | delete - manage |
| 95 | Agent-enabling Transformation of e-Commerce Portals with Web Services | David B. Ulmer | delete - manage |
| 96 | An Algorithm to Compute All Full-Span Sub Arrays of a Regular Array | Ronald I. Frank | delete - manage |
| 97 | An efficient algorithm for a class of time-dependent combinatorial optimization problems | Rigoberto Diaz | delete - manage |
| 98 | An eXtensible constraint markup language (XCML): | Jingkun Hu | delete - manage |

**Figure 3. External Publication Listing**

Through the use of jQuery, the event listeners find themselves bound by the conditions used to construct the fields since they can only enter characters that meet the author's established 'Unique ID'. Using jQuery UI application "Auto complete", these are well displayed at the bottom of the text box. Whenever a value is picked, the text box is filled out. Figure 4 below shows how "Auto complete" feature works during data entry.



Create Dissertation

Title:
Defense date:
Class year:
Author: Cha
Advisor:
Committee member 1:
Committee member 2:
Committee member 3:
Subject category primary:
Subject category secondary:
Subject category tertiary:
Method used primary:
Method used secondary:
Method used tertiary:
Number of pages total:
Number of pages without appendices:
Number of figures:
Number of tables:
Number of cited references:
Number of bibliography references:

**Figure 4. "Auto complete"**

The real look-up of this project is then done when you run a PHP script and it executes My SQL query that in turn searches the database. Using the field type as its parameter, the utilized PHP script runs a query that is just unique to that field and only returns values that match the author's preset ID. Through this, irrelevant values are not returned as answers to the run query.

## 4. Results

The system basically consists of two sets of pages. The first set includes all the pages that display information from the dissertation database either as a table or as a report. The second set of pages provides functionality to add and edit data. Data is validate at each data entry page to ensure that the data entered is valid and confirms to the specification.

## 5. Future Enhancements

There are a number of ways that this project can be extended to increase the breadth and utility of the data it manages. An important first step is "information portability" or more clearly the ability to organize and extract the data in various formats. This would allow for the information to be quickly imported into other systems or a generic table structure for data analysis. By creating an API (application programming interface) information could not only be readily extracted in a machine readable format (such as XML or JSON), the same API can also allow for other software to directly add and modify information. Another possible solution is to provide a simple CSV (comma separated value) or Excel formatted export. It would not allow for interaction that is nearly as robust, but would be sufficient for a basic method of extracting data.

The basic premise of this software is to manage information about documents and their relationship to various persons. Specifically we are currently concerned with dissertations, published documents, authors, advisors and committee members. This structure could be further generalized to allow it to contain relationships between numerous types of content and people, along with the associated metadata (publication date, years to completion, etc.). If this software was standardized it could be used to track information about all of the documents (papers, reports, dissertations, etc.) a student produces in the course of his graduate and undergraduate academic career. For archiving purposes, it would also not be difficult to include the actual paper itself in data storage, making it quickly available to that student, administrator or appropriate peers.

## 6. Conclusions

The web interface created as part of this project is a simple, but powerful tool to efficiently enter and maintain data in the growing dissertation database. Compared to the existing system, the new user interface with the underlying data structure sets new high standards for the data integrity. Tests around the database showed that the data is stored in the tables with 99% accuracy.

## 7. Summary

- The project involved some changes to the original database design.
- Database entities were rearranged to better represent real world relationships between dissertations, external publications and the people involved.
- The new structure is flexible enough to handle future expansions.
- Validations provided by the user interface ensure error free data entry.