

Bachelor's thesis
Business Information Technology
Information Systems
2013

Henri Landvik

WEB APPLICATION WITH YII FRAMEWORK



TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Business Information Technology | Information systems

2013 | 36 + 3

Anne Jumppanen

Henri Landvik

WEB APPLICATION WITH YII FRAMEWORK

This thesis is about creating a web application and a database. Database is a structured collection of data. It usually has valuable data inside. Given the importance of valuable data, it is important to keep the data secured from outside attacks. There are different kinds of databases to choose from. An object-relational database management system called PostgreSQL was used in this thesis. To make it easy for a client to access a database, it is common that a web application is created on the top of the database. An object-oriented, open source and a component-based PHP web application framework called Yii Framework were used in this thesis. It uses a model-view-controller architecture, which makes development easier for companies because different development groups can be assigned to do either model, view or controller development.

The second part of this thesis introduces the WISE project. WISE comes from the words white space test environment for broadcast frequencies. It develops and studies the efficient use of TV-band spectrum resources through cognitive radio technologies and geolocation databases.

The third part gives a view of an open source framework called Yii Framework. This thesis introduces the object-oriented aspect of Yii Framework, the model-view-controller architecture, user management, modules and the security of Yii Framework.

In the empiric part a database and a web application were developed and tested. Screenshots were used to give a clear image of how the empiric part was made step by step.

In conclusion, this thesis is summarized. The process of creating an empiric part and the creation of this thesis were pondered.

KEYWORDS:

open source, programming, database, framework

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietojenkäsittely | Tietojärjestelmät

2013 | 36 + 3

Anne Jumppanen

Henri Landvik

WEB-SOVELLUS YII-SOVELLUSKEHYKSELLÄ

Tämän opinnäytetyön tavoitteena oli kehittää tietokanta ja web-sovellus. Tavoitteena oli myös tehdä tietoturvallinen sovellus, joka pitäisi sisällään järjestelmän toimintojen testauksen. Web-sovellus kehitettiin WISE-projektin yhteydessä.

Tietokannan toteuttamiseen käytettiin PostgreSQL -nimistä tietokannan hallintajärjestelmää. Web-sovelluksen toteuttamiseen käytettiin Yii Framework -nimistä PHP-ohjelmistokehystä. Yii Framework käyttää malli-näkymä-kontrolleri-arkkitehtuuria, joka mahdollistaa joustavan ohjelmistokehityksen. Testaaminen toteutettiin PHPUnit -nimisellä testauskehysellä. Tietoturvan turvallisuuden kehittämiseen ja tarkastamiseen käytettiin internetistä ja kirjoista saatua tietoa.

Opinnäytetyön empiirisestä osiosta saatiin aikaiseksi toimiva ja tietoturallinen järjestelmäratkaisu. Järjestelmän testaaminen toi mukanaan täydentävän näkymän ohjelmistokehityksestä.

Yllämainituilla menetelmillä voidaan luoda tietoturallinen järjestelmä, mutta se edellyttää kehittäjältä tietämystä ja tutkimista. Järjestelmän testaus auttaa löytämään mahdolliset ohjelmistovirheet. Yii Framework on erinomainen sovelluskehys, joka mahdollistaa laadukkaiden nettijärjestelmien kehittämisen.

ASIASANAT:

avoin lähdekoodi, ohjelmointi, tietokanta, ohjelmointikehys

CONTENT

LIST OF ABBREVIATIONS	6
1 INTRODUCTION	7
2 WISE PROJECT	9
2.1 Definition	9
2.2 Goals	9
2.3 Members	10
3 YII FRAMEWORK	12
3.1 Features	12
3.2 Object-relational mapping	16
3.3 Model-View-Controller	18
3.4 Security	21
4 EMPIRICAL PART	23
4.1 The security of the web application	23
4.2 The database	23
4.3 The web application	25
4.4 Testing of the web application	31
5 CONCLUSIONS	33
SOURCE MATERIAL	34

APPENDICES

- Appendix 1. Database model
- Appendix 2. Source code of country restrictions class
- Appendix 3. Functional test of login and logout

PICTURES

Picture 1. Login screen.	25
Picture 2. Country restrictions.	28
Picture 3. Manage users.	29
Picture 4. User permission.	30

FIGURES

Figure 1. PHP Framework Performance Comparison (Yii Framework 2012n).	12
Figure 2. Structure of applications using ORM and relational databases (Porebski etc. 2011, 58).	17
Figure 3. Model-view-controller example (Abeyasinghe 2009, 33).	19
Figure 4. Static structure of Yii applications (Yii Framework 2012c).	20

TABLES

Table 1. Country table.	24
Table 2. Dynamic Values table.	24

LIST OF ABBREVIATIONS

AR	Active records
CRUD	Create, read, update, delete
CSRF	Cross-site Request Forgery
DAO	Data access object
DBMS	Database management system
I18N	Internationalization
L10N	Localization
MVC	Model-view-controller
OODBMS	Object-oriented database management systems
OOP	Object-oriented programming
ORM	Object-relational mapping
PDO	PHP data objects
RDBMS	Relational database management system
WISE	White space test environment for broadcast frequencies
WSD	White space device
XSS	Cross-site scripting

1 INTRODUCTION

This thesis is about creating a database and a web application. Database is a vital part of information systems because they usually store important information inside. To access important information, either a computer terminal or a web application is needed. Web applications are commonly used nowadays, because they provide an easily accessible connection to a database, and for the use high computer skills are not normally required. They give an image of what the system is like. For this thesis, however, functionality and safety are more important aspects than the look.

This thesis aims to answer the questions: What is Yii Framework? How to create a secure web application with Yii Framework?

The thesis is written as an assignment to Turku University of Applied Sciences as a part of the WISE (White space test environment for broadcast frequencies) project. In the WISE project a testbed was built for studying the use of cognitive radios using white spaces of the UHF television broadcasting band. Then the focus was turned on protection methods for shared spectrum and a new project named WISE2 was started.

The WISE project was in need of a web application that would allow an administrator to change the transmit power value according to the country. It was assigned to the author to build the system, which would have a database and a web application. An administrator will have privileges to change only a couple of restrictions from the database. Privileges are given only to an administrator and no other users will be created.

The application will provide country specific information for a white space device. The white space means an area, where some wireless device has been given a frequency range that is unexploited geographically or in timely manner. The white space device is a device designed to detect the presence of existing areas of unused airwaves.

The database is implemented with an object-relational database management system called PostgreSQL and it was chosen by Turku University of Applied Sciences (TUAS) in a meeting with a company called Fairspectrum. PostgreSQL was chosen because of its good geographical features, which are vital for the project. The web application is implemented with Yii Framework, which is an object-oriented model-view-controller (MVC) framework. This framework was chosen by TUAS, because of its security and object-oriented features.

This thesis will be of assistance to a person interested in creating a safe object-oriented model-view-controller framework with PHP programming language. The thesis will also be beneficial for the WISE2 project.

2 WISE PROJECT

2.1 Definition

WISE (White Space Test Environment for Broadcast Frequencies) is a project (2011-2012) in which a database testbed was built for studying the use of cognitive radios that can use the white spaces of the UHF television broadcasting band (470-790 MHz). In WISE, an open cognitive radio geolocation database testbed was established, which allowed concrete studies of the algorithms, usability and interfaces of cognitive radio systems operating in television broadcast bands. (WISE 2013a.)

In 2013 started a new project called WISE2, which focuses on incumbent protection methods for shared spectrum. For TV white spaces digital video broadcasting and wireless microphones are considered. The ASA/LSA spectrum sharing is investigated in the 2,3 – 2,4 GHz frequency band to coexist with PMSE equipment such as wireless cameras. The WISE2 project will be funded at least till the end of 2014 in Finland by Tekes. (WISE 2013a.)

Similar projects are going forward in countries like the United States of America, Great Britain, China and Russia.

2.2 Goals

The goal of the WISE project was to develop a testbed for studying the use of cognitive radios using the white space frequencies of the UHF broadcasting band and a geolocation database. (WISE 2013a.)

The goal of the WISE2 project is to use the testbed and TV white spaces for numerous pilot projects such as follows:

- Intelligent transport systems.
- Wireless surveillance systems.

- Machine-to-machine communications.
- Rural broadband.
- ASA/LSA.
- Smart grid. (WISE 2013a.)

2.3 Members

WISE2 has the following members:

- Turku University of Applied Sciences (TUAS)
- University of Turku (UTU)
- Aalto University
- Digita
- Fairspectrum
- Nokia
- Elektrobit
- Helsingin Seudun Liikenne (HSL)
- NSN
- Satel
- Teleste
- Jyväskylän kaupunki
- Qem Software Oy
- Viola Systems
- Finnish Communications Regulatory Authority (FICORA) (WISE 2013b.)

Aalto University develops the communications and networking. The university also develops signal processing and acoustics.

University of Turku looks after business affairs and takes care of innovation development.

Engineers from Turku University of Applied Sciences are involved in the project working with radio interference measurements.

IT Bachelor students from Turku University of Applied Sciences have created a database testbed, which enables versatile practical studies of the operation of the cognitive radios using TV white spaces. The practical studies include the studies of the algorithms, usability and the interfaces. (WISE 2013a.)

Applications that were developed and are still developed by IT Bachelor students from Turku University of Applied Sciences are such as follows:

- PMSE Manager – a wireless microphone manager
- Windows Phone WISE Client
- Java Client
- Front End
- Device Registry
- ASA/LSA (Authorized Shared Access / Licensed Shared Access) manager
- TPM (Trusted Platform Module) Authentication
- Protocol to Access White Space Database (PAWS)

3 YII FRAMEWORK

3.1 Features

Yii Framework is an object-oriented, high performance, model-view-controller framework. It is created to be a productive, adjustable and easy to maintain as a final produce. Yii does not cost anything and it is created with PHP5. It is an open source application, which is created for quick development. Yii Framework has instruments for testing and for debugging web applications. Yii can be extended with different kinds of extensions. Modules, widgets, controllers, actions, filters, validators, behaviors, application components, 3rd-party libraries and more can be added to Yii Framework. In Yii, all code can be personalized for specific requirements. Yii Framework is also tested to be the most efficient framework in autumn 2012, as seen in Figure 1. This proves that Yii is a very powerful framework. (Yii Framework 2012a; 2012b; 2012j; 2012n.)

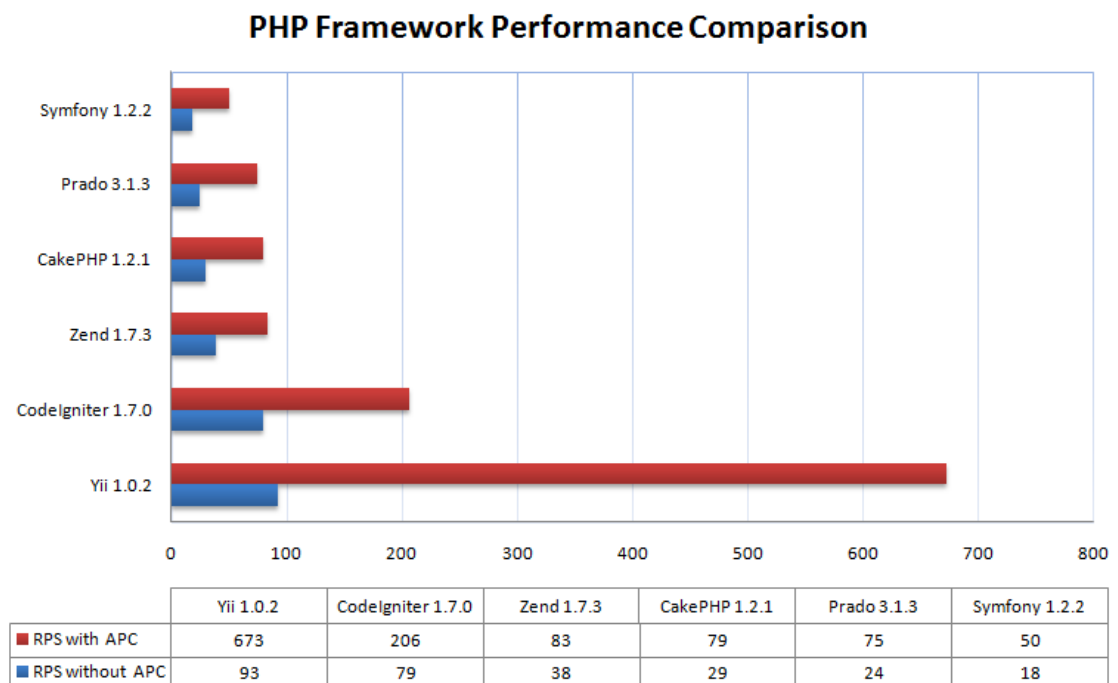


Figure 1. PHP Framework Performance Comparison (Yii Framework 2012n).

Theming

Even though there are many extensions available in Yii Framework, only few themes can be found from the Internet, which makes it recommendable to hire a graphical designer for a project. For changing the look of a theme, it is possible to implement a skin to Yii Framework. It allows the user/developer to change e.g. the color of a theme rapidly from one to another. Using skins, on the other hand, might lower the performance of an application, which makes it up to the developer to choose whether to implement skins or not. (Yii Framework 2012m.)

Internationalization and localization

When planning a new Internet software, different time zones and languages need to be taken into account since there can be many users from all around the world. In Yii, regions and languages can be changed without any manufacturing alterations. This is possible as Yii offers support for Internationalization (I18n) and Localization (L10N). (Yii Framework 2012j.) Internationalization is an action that can disseminate time zone information and location information from servers to servers and from clients to servers (Kovari etc. 2004, 539). Localization occurs when the specific location modules are added to the system and the text is translated. This is accomplished when the information is processed from an internationalized application. (Yuan etc. 2010, 98.)

Authentication and authorization

In Yii Framework, authentication is typically done with a username and a password, which is the normal procedure in web applications. For people's identification, it is also possible to use different kinds of methodologies. Fingerprints, smart cards or other procedures could be used for identifying a user in Yii. Authentication means that the person, who is trying to access the system, will be

confirmed as the one, who the person affirms to be. When the person is authenticated, it needs to be checked that the person is allowed to control particular functions in the system. Yii Framework, which provides an integrated framework for authentication and to authorize a person, can be modified for different system requirements. (Yii Framework 2012h.)

Testing

For testing purposes Yii offers functional and unit testing (Yii Framework 2012f). Tests that need a certain kind of operator input, return a certain kind of outcome, and test a particular operation of a web-application, are called functional tests (SeleniumHQ 2012). Tests that are written for the operation of a programming method or a programming class are called unit tests (Dirk 2010, 294).

Automatic code generation

Yii is implemented with a module called gii. It is an automatic code generator that makes possible to create models, modules, forms, CRUD's and controllers without any coding. (Yii Framework 2012g.) CRUD comes from the words create, read, update, delete. It is a programming class that is implemented with methods that create, read, update and delete data from a database. (Winesett 2010, 66 – 68.)

Caching

Yii provides different caching options. Caching is a way to increase the efficiency of internet software. In Yii, it is possible to e.g. save cached data to a database, into a particular file, into memory, add an extension that enables non-caching, or use caching in some other way. (Yii Framework 2012k.)

Model-View-Controller

Yii has a commonly used planning model called model-view-controller (MVC). The intention of the planning model is to divide the model from the view. The link between the model, the view and other different components is a controller. The controller ties information and transports information between all components. (Yii Framework 2012c; 2012d.)

Database overview

Yii offers an object called data access object (DAO). It enables connecting to a dissimilar database management system (DBMS). The data access object is constructed atop of the PHP data objects (PDO) add-on. PDO is an add-on that states a steady interface with light features to get accession to different kinds of databases with PHP. To decrease the number of threats of SQL injection assaults, Yii Framework is implemented with an object-oriented technique. This technique goes by the name of Yii query builder, which is designed to help construct SQL queries in a safe manner. (Yii Framework 2012e.) (PHP 2012.)

Yii has also added an object called active record (AR) object. Active record is an object, which captures database accession, binds a row from a view or a table in the database, and inserts domain rationality to that data (Fowler etc. 2011, 160). Active record is executed with an extensively used approach called object-relational mapping (ORM). Object-relational mapping is a technique that makes a layer between a model and relational database systems (Porebski etc. 2011, 58). (Yii Framework 2012e.)

Security

Yii is defended against Cross-site scripting (XSS), Cross-site Request Forgery (CSRF) and against cookie attacks (Yii Framework 2012l), which makes it well

protected against ordinary attacks. The security of Yii Framework is discussed more in detail in Chapter 3.4.

Other features

In Yii, errors can also be handled. Models, actions and forms can be created manually in Yii Framework. The performance of an application can be tuned for a greater speed. Web-services, message logging functions and more can also be implemented with Yii Framework. (Yii Framework 2012o; 2012p; 2012q; 2012r; 2012s.)

The features of Yii show that Yii can be customized in many ways, and it is a powerful framework for creating internet applications.

3.2 Object-relational mapping

One of the main things in PHP frameworks is object-relational mapping (ORM). It is a technique, which makes an abstraction layer between a model and relational database systems. The ORM functionality is included in Yii Framework (PHP Frameworks 2012). When looking at Figure 2, we can see the construction that is built on top of an ORM instrument. The ORM mapper creates equivalent database tables from the data model as the data model is included in the software. For running specific queries in the software application, PHP Data Object (PDO) is commonly used by the ORM instruments. PDO is an add-on, which states a steady interface with light features to get accession to different kinds of databases with PHP (PHP 2012). (Porebski etc. 2011, 58.)

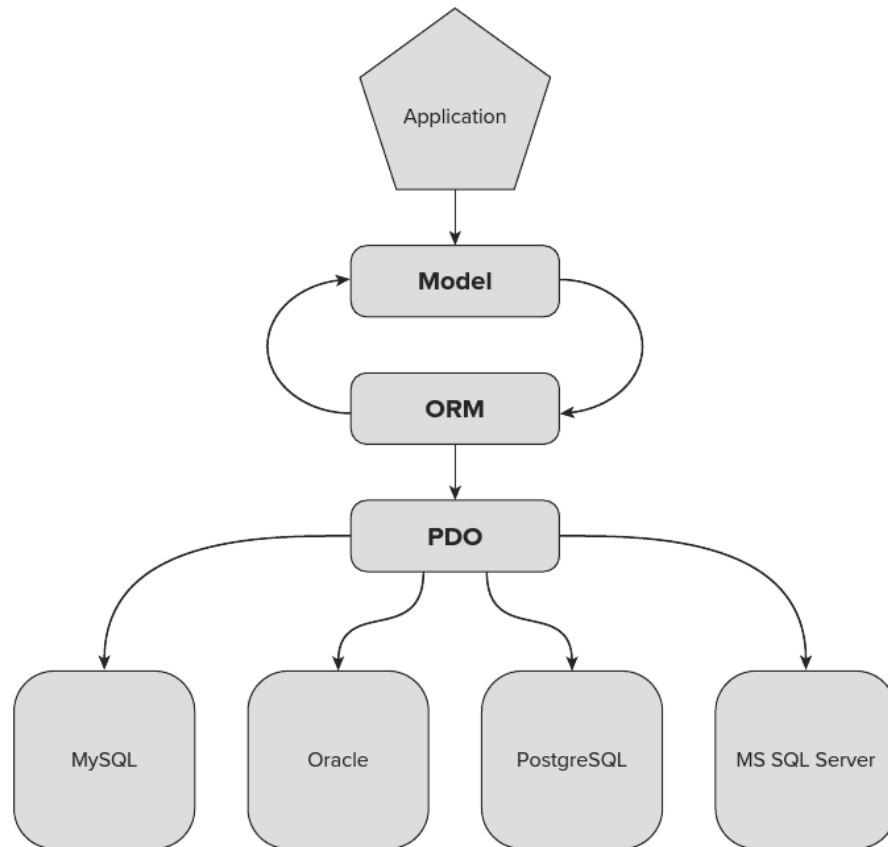


Figure 2. Structure of applications using ORM and relational databases (Porebski etc. 2011, 58).

Some of the most common problems with ORM

In relational databases, numerous mechanical, conceptual or cultural problems might arrive when application is developed using object-oriented data. Some of the most typical problems are:

- Conceptual differences – People think in a different way about object-oriented programming (OOP) and have different ways to advance towards it. These differences can take programmers and database administrators to profound misconceptions.
- Constraints – Object-oriented programming languages do not offer declarative constraints, which are the inner part of the data model.

- Data structures – Relational database management systems (RDBMS) use table data models. Object-oriented programming languages on the other hand use nested data structures with object lists.
- Data types – RDBMSs have more specified rules than OOP. For example, OOP string types have limitless length, but RDBMSs have fixed maximum length.
- Encapsulation – Relational database management systems do not know that object-oriented programming highlights hidden objects.
- Inheritance – Object-oriented programming uses inheritance, but this practice is not supported by relational databases.
- Maintenance – Software development usually requires changes in the database. Requests from software developers as being redundant may corrupt a database. Although no damage might occur, disproportion highly raises the cost of maintenance. (Porebski etc. 2011, 59-60.)

Different ways to resolve impedance mismatch is either to use Object-oriented database management systems (OODBMSs) or abandon OOP completely. (Porebski etc. 2011, 60.)

3.3 Model-View-Controller

Yii Framework has been created by using model-view-controller (MVC) architecture. It is a planning model that divides software development to three different sections. (Yii Framework 2012c.) Those sections are model, view, and controller. The data of the software is stored in the model. The functions of the model are to control how the software domain behaves and how the data behaves. It gets requests from the controller and the view, and it changes, and responds information to them. The view shows a limited amount of data from the model. The view is designed to show only the needed information to the user. This way the data can be sorted, and structured to different users according to their needs. The controller ties and delivers the information between all components. It reacts to actions and might appeal to modifications on the view

or the model. When taking the dependence of components into account, the model is independent. This means, that the model is not dependable on the controller or the view. Nonetheless, the controller and the view are not independent as they are depending on the model. With the structure of the model-view-controller, the components can be developed and tested separately. When we look at Figure 3, we can see an example where the teens are listed into the system. The model has the data of the teens. The controller loads the teens and sends them to the user as a view. The view shows the results to the user as a list of two teens with their age and name. (Abeysinghe 2009, 32-33.)

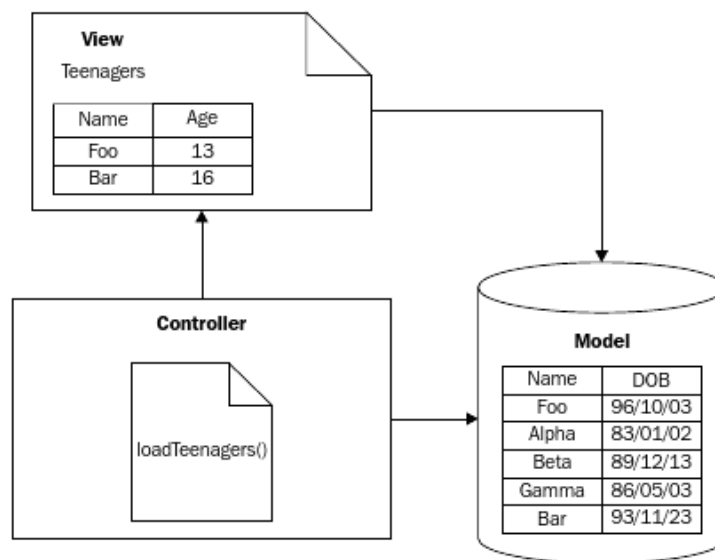


Figure 3. Model-view-controller example (Abeysinghe 2009, 33).

Yii front-controller

When looking at Yii Framework more closely, it has also a front-controller which gathers particular info about a user request. The front-controller summarizes the execution context so that the request could be processed. The software then sends the user request forward into a suitable controller for further management. The name of the front-controller is application. When looking at Figure 4, we can see Yii Framework's usual workflow.

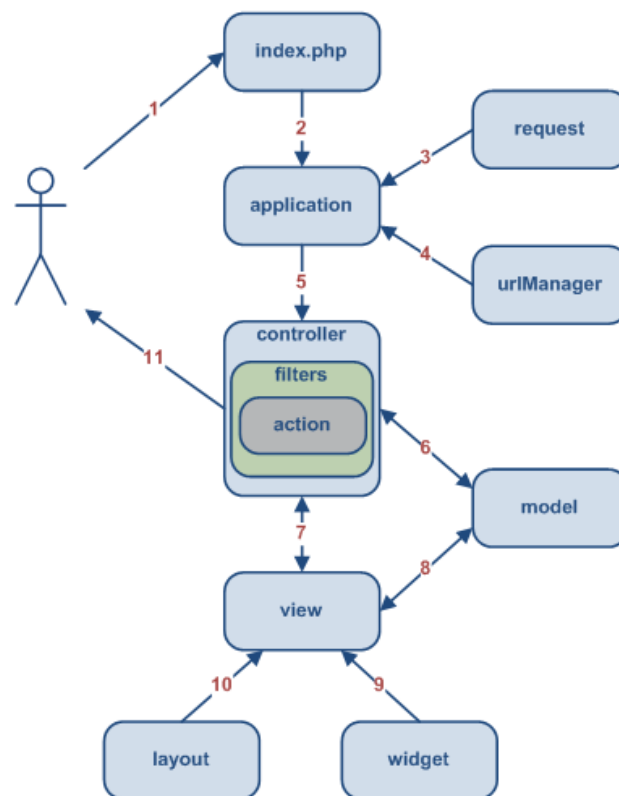


Figure 4. Static structure of Yii applications (Yii Framework 2012c).

The workflow of a request goes in the following way:

1. First a request is created by the user. Then the internet server manages the request.
2. Application instance is created by a script. Then the script is executed.
3. Front-controller called application gets elaborate user request info. This info is received from an application component called request.
4. Controller and application definitions are received to the front-controller application from an application component called urlManager.
5. Instance is created by the front-controller called application of the requested controller. Then the user request is processed for further handling. Operation called show is referring to a method called actionShow in the controller class. This operation is specified by the controller. After that the filters are made and run.
6. Model is read from the database.

7. View is rendered with the model.
8. Attributes are read and displayed by the view from the model.
9. Few widgets are run by the view.
10. Results are immersed in a layout.
11. Actions are completed, and the outcomes are displayed to the user. (Yii Framework 2012c.)

3.4 Security

Yii Framework prevents Cross-site scripting which is also known as XSS. XSS happens when VBScript, JavaScript, ActiveX, HTML or Flash is injected by the invaders into fragile software. It takes place when Internet software gathers malicious information from a user and it tries to trick other software users and collect information from them.

Yii Framework prevents Cross-Site Request Forgery also known as CSRF. It appears when a malicious Internet page induces a user's Internet browser to execute a non-desired operation on a site that is being trusted.

Yii Framework also prevents Cookie Attacks. It guards cookies from being under attack. This is highly important, because session IDs are usually saved in cookies. If an invader gets a session ID in his hands, the invader intrinsically possesses essential information from the session.

Countermeasures to stop cookie invasions

There are numerous countermeasures to stop cookies being invaded. The software can use SSL to make a secure communication channel and transfer an authentication cookie over an HTTPS connection. Verify cookie information and discover if it is changed.

Closing the sessions properly is important to reduce danger of being invaded. This includes closing session tokens and cookies. It is also important to stop

cross-site scripting, discover that cookie data is being altered and checked and make sure that cookie validation is activated in Yii Framework.

Security conclusion

Other security measurements are in the developer(s) or in the developer team's hands. For creating secure software with Yii Framework the developer either needs to use an extension or create secure software by himself using PHP. Making secure software in Yii Framework strongly depends on the developer or developing team. Yii does not offer in itself big security protections like authentication in default and therefore extensions are in need for authentication. Extensions might also be needed to be modified for making a web application secure. (Yii Framework 2012l.)

4 EMPIRICAL PART

4.1 The security of the web application

Security in Yii Framework depends a lot on the developer's choices. Authentication and User management need to be created with extensions or by creating everything from scratch. Password encrypting is done by phpass, by a portable PHP password hashing framework, which is being used e.g. in Wordpress. There are many extensions for Yii and they are useful for more secure applications.

The extension called "yii-user" helps that a database can be used for authentication. (Yii Framework Extension 2013a.)

The extension called "rights" gives permissions to users. (Yii Framework Extension 2013b.)

Phpass is a hashing framework that hashes passwords to make them secure from invaders. (Phpass 2013.)

The extensions "yii-user" and "rights" are used in this thesis. The hashing framework called phpass is used for password encryption.

4.2 The database

The first test version uses PostgreSQL as a database implementation. In this thesis, two tables were designed and implemented, Country table and Dynamic Values table. These two tables are explained below more specifically. All the other tables, which have been obtained from the extensions used for authentication and user management, will not be considered in this thesis.

Table 1. Country table.

Column	Type	Not Null	Default	Constraints
country_id	integer	NOT NULL	nextval	Primary Key
country_acronym_id	varchar	NOT NULL		
country	varchar	NOT NULL		
config_server_name	varchar	NOT NULL		
config_area_id	varchar	NOT NULL		

Table 1 contains information about the countries which use the system. The data includes information about the “country_id” which has the script “nextval” for auto increment. The “country_id” starts from an integer value 1 and is the table’s primary key. The “country_acronym_id” is an abbreviation of the country e.g. “FI”. The “country” tells the full name e.g. Finland. The “config_server_name” is the name according to the country. The “config_area_id” is the area where the configuration is.

Table 2. Dynamic Values table.

Column	Type	Not Null	Default	Constraints
dynamic_id	integer	NOT NULL	nextval	Primary Key
max_frequency	numeric	NOT NULL		
min_frequency	numeric	NOT NULL		
transmit_power	numeric	NOT NULL		
country_id	integer	NOT NULL		Foreign Key

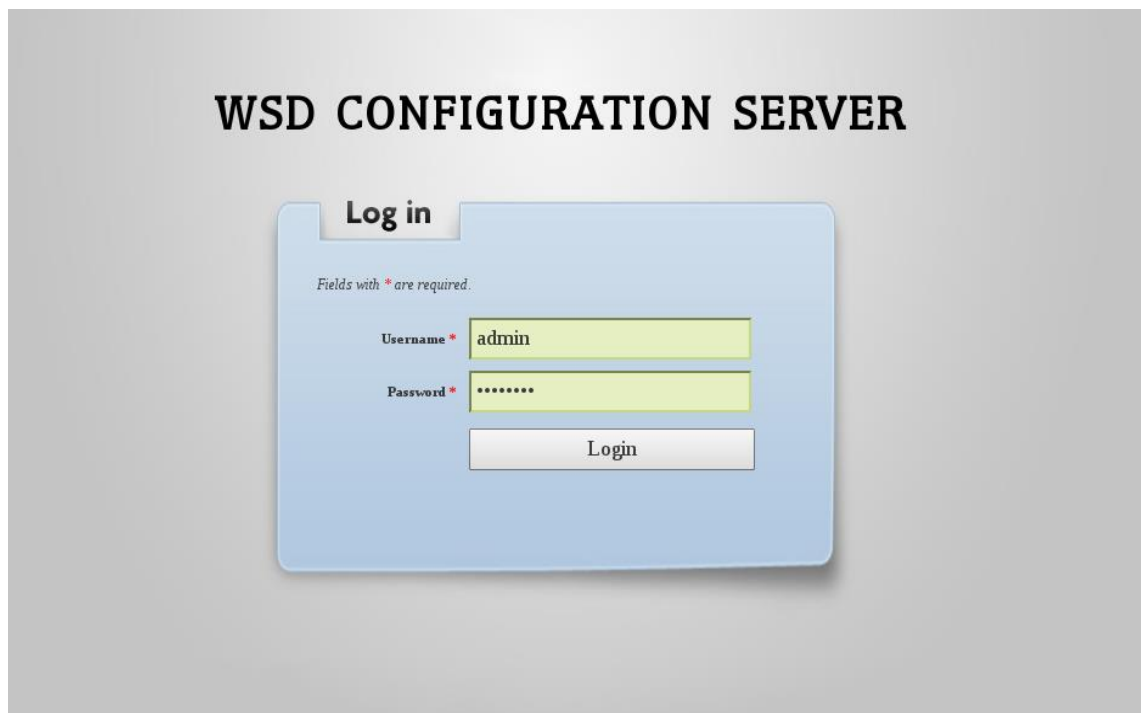
Table 2 has information about the dynamic values. The “dynamic_id” has the script “nextval” for auto increment. The “dynamic_id” starts from an integer value 1 and is the table’s primary key. The “max_frequency” has information about the maximum allowed frequency to be used and the data type is numeric. The “min_frequency” tells the minimum allowed frequency to be used and the data

type is numeric. The “transmit_power” has a transmit power value which is displayed in a numeric form. The “country_id” tells which country is being used. The “country_id” is a foreign key and there is a binding to the “Country” table. The database model is found in Appendix 1.

Other tables are named “authassignment”, “authitem”, “authitemchild”, “profiles_fields”, “profiles_table”, “rights” and “users” table. They are needed for “yii-user” and “rights” extensions which make user management and controlling user rights possible.

4.3 The web application

The first test version of the web application has a login interface to access the server. The application has been designed to crypt the password. The web application saves the password to the database and requires an administrator to log in for accessing the application.



WSD CONFIGURATION SERVER

Log in

*Fields with * are required.*

Username * admin

Password *

Login

Picture 1. Login screen.

As seen in Picture 1 to access this system an administrator username and a password are required. All of the sites are redirected to the login screen so that the login is required before anyone can access the application. Phpass is used for encrypting passwords to the database.

```
// Portable PHP password hashing framework (phpass) configuration
'hasher'=>array (
    'class'=>'Phpass',
    'hashPortable'=>false,
    'hashCostLog2'=>18,
),
```

Phpass is a hashing framework and needs to be configured in the project. It is possible to enable or disable portability and increase the iteration of an encrypting password. Disabling portability makes phpass to use bcrypt encryption algorithm which is a secure password encryption algorithm. Enabling portability makes use of a secure encryption algorithm but without bcrypt.

```
/**
 * Declares the validation rules.
 * The rules state that username and password are required,
 * and password needs to be authenticated.
 */
public function rules()
{
    return array(
        // username and password are required
        array('username, password', 'required'),
        // password needs to be authenticated
        array('password', 'authenticate'),
    );
}
```

The login has validation rules configured, which requires a username and an authenticated password. Otherwise the login is not allowed. For an authenticated login the web application is using the Yii extensions called “yii-user” and “rights”. Extensions are needed, because Yii Framework does not have a database authentication built-in.

```

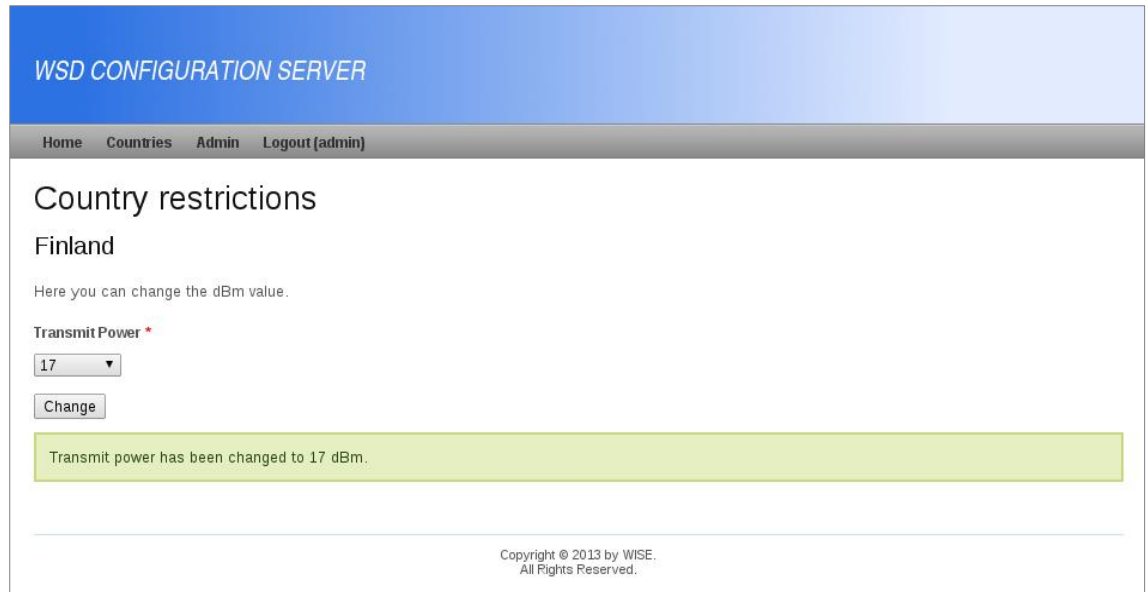
/**
 * Redirects the login page
 */
public function actionLogin()
{
    // Defines this methods layout
    $this->layout = 'loginlayout';
    $model=new LoginForm;

    // if it is ajax validation request
    if(isset($_POST['ajax']) && $_POST['ajax']==='login-form')
    {
        echo CActiveForm::validate($model);
        Yii::app()->end();
    }

    // collect user input data
    if(isset($_POST['LoginForm']))
    {
        $model->attributes=$_POST['LoginForm'];
        // validate user input and redirect to the previous page if valid
        if($model->validate() && $model->login())
            // Redirects to home page after login
            $this->redirect(array('site/index'));
    }
    // display the login form
    $this->render('login',array('model'=>$model));
}

```

The controller function “actionLogin” redirects the user after the login to the index page. The login layout is defined in the beginning of this function, because the login and the other pages have different layouts.



Picture 2. Country restrictions.

In the “Country Restrictions” section dBm values can be changed from -100 to 100. In the first phase a country needs to be selected. After the country has been selected the dBm values can be changed as seen in Picture 2. After the transmit power has been changed a note will be printed out to the website. Codes are found in Appendix 2.

WSD CONFIGURATION SERVER

[Home](#) [Countries](#) [Admin](#) [Logout \[admin\]](#)

Manage Users

- [Create User](#)
- [List User](#)
- [Manage User](#)
- [Manage Profile Field](#)

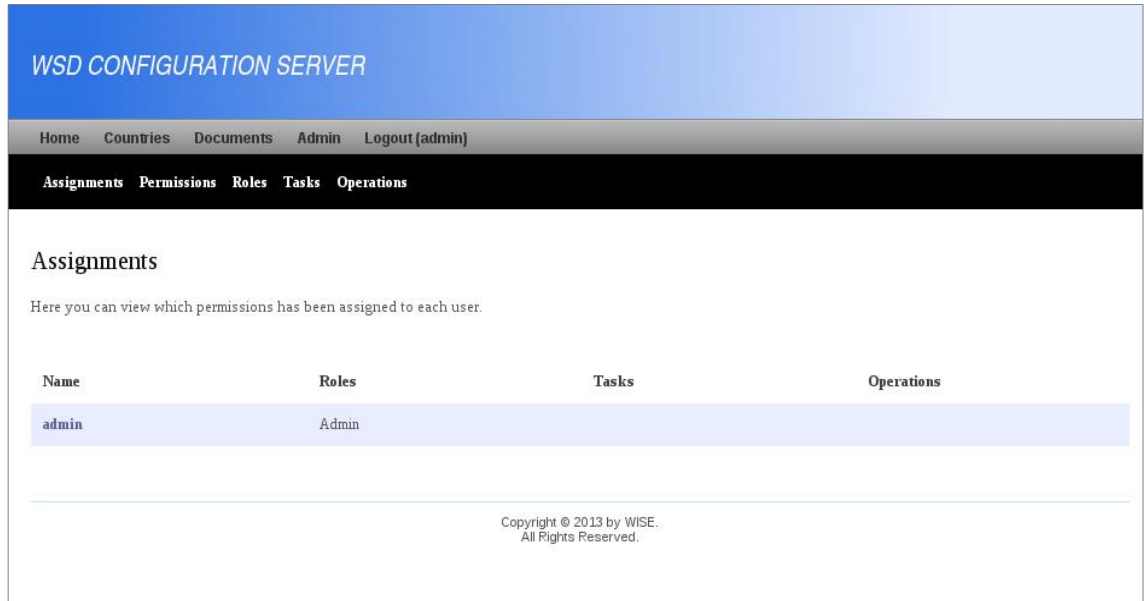
Displaying 1-1 of 1 result.

Id	username	E-mail	Registration date	Last visit	Status	Superuser	
1	admin	admin@gmail.com	18.01.2013 13:47:16	18.01.2013 13:47:16	Active	Yes	

Copyright © 2013 by WISE.
All Rights Reserved.

Picture 3. Manage users.

New users can be created, listed and managed as seen in Picture 3. All functions come from the Yii extension called “yii-user”. Managing users is made quite easy with this extension and it has the basic functions for managing a user. The user can log in either by using a user name or an email. Users can register, activate accounts through email, recover passwords and users have a profile page. Users and profile fields can be managed. For viewing, editing and saving data there is a profile field widget. There is also a date widget, a file upload widget and a profile relation widget.



Picture 4. User permission.

Assignments, permissions, roles, tasks and operations can be changed in the system as seen in Picture 4.

The Rights extension has the following features:

- Internationalization (I18N)
- Runtime caching to raise performance
- Sort authorization items by dragging and dropping
- Assigning authorization items to users
- View demonstrating each role's assigned tasks and operations
- User interface optimized for usability, role, tasks and operation management
- Installer for easy and quick set up
- Authorization item generation
- Controller filter for checking access
- Support for business rules, cross-browser and cross-database compatibility

4.4 Testing of the web application

The web application test is done by using functional testing. Testing covers login and logout testing. Doing functional testing in Yii requires that PHPUnit and Selenium Remote Control testing frameworks are installed because Yii is tightly integrated with these two frameworks. PHP extension called Pear also needs to be installed to create functional tests.

```
<phpunit bootstrap="bootstrap.php"
    colors="false"
    convertErrorsToExceptions="true"
    convertNoticesToExceptions="true"
    convertWarningsToExceptions="true"
    stopOnFailure="false">

    <selenium>
        <browser name="Internet Explorer" browser="*iexplore" />
        <browser name="Firefox" browser="*firefox" />
    </selenium>

</phpunit>
```

A couple of modifications are also needed to be done before a developer can start running tests. First the “phpunit.xml” file is to be configured and a default browser needs to be chosen. Most operating systems do not have Internet Explorer installed beforehand and that is why it is important to delete the line where Internet Explorer is written.

Secondly the test URI needs to be defined before testing.

```
define('TEST_BASE_URL', 'http://localhost/PROJECT_NAME/index-test.php/');
```

After that the test can be launched by writing “phpunit” and the name of the file on to the terminal.

```
protected/tests$ phpunit functional/SiteTest.php
```

Functional test from this thesis tries to log in to the system using admin values.

```
// test admin login process, including validation
$this->assertElementPresent('name=LoginForm[username]');
$this->type('name=LoginForm[username]', 'admin');
$this->click("//input[@value='Login']");
$this->waitForTextPresent('Password cannot be blank. ');
$this->type('name=LoginForm[password]', 'admin');
$this->clickAndWait("//input[@value='Login']");
$this->assertTextNotPresent('Password cannot be blank. ');
$this->assertTextPresent('Logout');
```

After the test has successfully logged in it tries to log out.

```
// test logout process
$this->clickAndWait('link=Logout (admin)');
```

The functional test is found in Appendix 3.

5 CONCLUSIONS

This thesis started fascinatingly when the author was commissioned by TUAS to do a web application belonging to the WISE project. The focus of the WISE project was to develop and study the efficient use of TV-band spectrum resources through cognitive radio technologies and geolocation databases.

The purpose of this web application was to give access for administrator(s) to change the dBm values of transmit power from the database. The aim was also to make a secure web application. In a project meeting of TUAS the choice of the framework was discussed and Yii Framework was chosen to be used.

Yii Framework was completely new for the author, but it looked promising. After much reading and many tutorials the author decided to try to test Yii more specifically. After that the learning of Yii became quite fast and it was possible to do useful software development and testing.

The author had some experience about Java and very little about PHP. The MVC architecture in general including Yii Framework was quite unfamiliar to the author. It took quite a long time to understand the logic behind the MVC architecture and the designing of the database was not easy at all either. Hard work produced results at last.

The composing of the thesis with its empirical part has been a successful learning process. The web application of the thesis was carried out as a part of the WISE project, which has been a good and important step forward, and this thesis is part of that crucial step. Now the future of the WISE project will be in the future of the WISE2 project.

SOURCE MATERIAL

Abeyasinghe, S. 2009. PHP Team Development : Easy and Effective Team Work Using MVC, Agile Development, Source Control, Testing, Bug Tracking, and More. First Edition. Birmingham:Packt Publishing.

Dirk, M. 2010. Expert PHP 5 Tools. Birmingham:Packt Publishing.

Fowler, M.; Rice, D.; Foemmel, M.; Hieatt, E.; Mee, R. & Stafford, R. 2011. Patterns of Enterprise Application Architecture. Seventeenth printing. Boston:Addison-Wesley.

Kovari, P.; Boardman, L.; Dring, G.; Johnson, R.; Kitabayashi, H.; Moghal, S.; Mueller, R.; Nagarajan, S. & Zeng, Y. 2004. WebSphere Business Integration Server Foundation V5.1 Handbook. First Edition. Durham:IBM.

Phpass 2013. Portable PHP password hashing framework. Referred 21.6.2013

<http://www.openwall.com/phpass/>

PHP 2012. Introduction. Referred 13.11.2012

<http://www.php.net/manual/en/intro.pdo.php>.

PHP Frameworks 2012. PHP Frameworks. Referred 23.11.2012

<http://www.phpframeworks.com/>.

Porebski, B.; Przystalski, K. & Nowak, L. 2011. Building PHP Applications with Symfony, CakePHP, and Zend Framework. Hoboken:Wrox.

SeleniumHQ 2012. Test Design Considerations. Referred 14.11.2012

http://seleniumhq.org/docs/06_test_design_considerations.html#types-of-tests.

Winesett, J. 2010. Agile Web Application Development with Yii 1.1 and PHP5. Birmingham:Packt Publishing.

WISE 2013a. WISE Home. Referred 18.10.2013

<http://wise.turkuamk.fi/?page=home>

WISE 2013b. WISE Consortium. Referred 18.10.2013

<http://wise.turkuamk.fi/?page=consortium>

Yii Framework Extension 2013a. Yii-user. Referred 21.6.2013

<http://www.yiiframework.com/extension/yii-user/>

Yii Framework Extension 2013b. Rights. Referred 21.6.2013

<http://www.yiiframework.com/extension/rights>

Yii Framework 2012a. About Yii. Referred 12.11.2012

<http://www.yiiframework.com/about/>.

Yii Framework 2012b. Testing. Referred 12.11.2012

<http://www.yiiframework.com/doc/guide/1.1/en/test.overview>.
Yii Framework 2012c. Model-View-Controller (MVC). Referred 13.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/basics.mvc>.
Yii Framework 2012d. Best MVC Practices. Referred 13.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/basics.best-practices>.
Yii Framework 2012e. Working with Database. Referred 13.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/database.overview>.
Yii Framework 2012f. Testing. Referred 14.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/test.overview>.
Yii Framework 2012g. Automatic Code Generation. Referred 14.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/topics.gii>.
Yii Framework 2012h. Authentication and Authorization. Referred 15.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/topics.auth>.
Yii Framework 2012i. Extending Yii. Referred 15.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/extension.overview>.
Yii Framework 2012j. Internationalization. Referred 15.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/topics.i18n>.
Yii Framework 2012k. Caching. Referred 16.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/caching.overview>.
Yii Framework 2012l. Security. Referred 20.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/topics.security>.
Yii Framework 2012m. Theming. Referred 20.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/topics.theming#skin>.
Yii Framework 2012n. Performance of Yii. Referred 20.11.2012
<http://www.yiiframework.com/performance/>.
Yii Framework 2012o. Error Handling. Referred 20.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/topics.error>.
Yii Framework 2012p. Working with Form. Referred 20.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/form.overview>.
Yii Framework 2012q. Performance tuning. Referred 20.11.2012
<http://www.yiiframework.com/doc/guide/1.1/en/topics.performance>.

Yii Framework 2012r. Web Service. Referred 20.11.2012

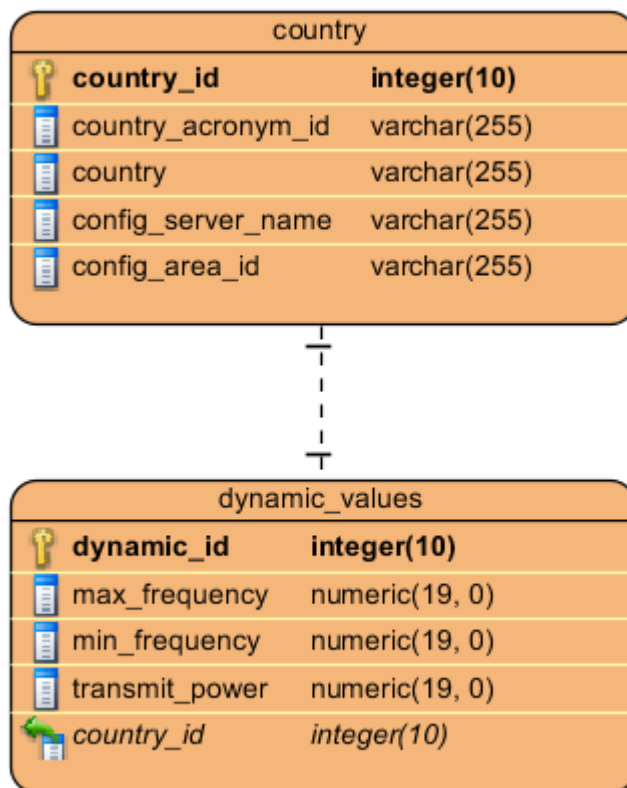
<http://www.yiiframework.com/doc/guide/1.1/en/topics.webservice>.

Yii Framework 2012s. Logging. Referred 20.11.2012

<http://www.yiiframework.com/doc/guide/1.1/en/topics.logging>.

Yuan, J. X.; Chen, X. & Yu, F. 2010. Liferay User Interface Development. Birmingham:Packt Publishing.

Database model



Source code of country restrictions class

```

<?php
/* @var $this DynamicValuesController */
/* @var $model DynamicValues */
/* @var $form CActiveForm */

?>

<!-- Prints out text "Country restrictions" -->
<h1>Country restrictions</h1>

<!-- Prints out the country which user has selected -->
<h2> <?php
    // Creates a SQL session. Gets Country name by country_id
    // from PostgreSQL database.
    $country_id = Yii::app()->session['country_id'];
    $country = CHtml::listData(Country::model()->findAll(array(
        'select'=>'country',
        'condition'=>'country_id=:country_id',
        'params'=>array(':country_id'=>$country_id)),
        'country_id', 'country');
    echo $country[''];

?>
</h2>

<!-- Prints out text "Here you can change the dBm value." -->
<p>Here you can change the dBm value.</p>

<!-- form starts -->
<div class="Form">

<!-- Widget begins -->
<?php $form=$this->beginWidget('CActiveForm', array(
    'id'=>'dynamic-values-txpower-form',
    'enableAjaxValidation'=>false,
)); ?>
<!-- Echoes error summary -->
<?php echo $form->errorSummary($model); ?>
<div class="row">
    <!-- Gets country_id and transmit_power from database and saves it to an array -->
    <?php $array = CHtml::listData(DynamicValues::model()->findAll(), 'country_id', 'transmit_power'); ?>
    <!-- Creates a session using country_id -->
    <?php $country_id = Yii::app()->session['country_id']; ?>
    <!-- Sorts keys to find right country_id -->
    <?php
        $theKey = null;
        $keys = array_keys( $array );
        foreach ( $keys as $key => $keyString) {
            if($keyString == $country_id) {
                $theKey = $key;
                break;
            }
        }
    ?>
    <!-- Prints transmit power text -->
    <?php echo $form->labelEx($model,'transmit_power'); ?>
    <!-- Gets transmit power values from table model and prints out drop down list -->
    <?php echo $form->dropDownList($model,'transmit_power', $model->getValueOptions(), array(
        'options' => array($array[$keys[$theKey]] => array('selected' => true))); ?>
    <!-- Prints errors -->
    <?php echo $form->error($model,'transmit_power'); ?>
</div>

<!-- Submit button named "Change" -->
<div class="row buttons">
    <?php echo CHtml::submitButton('Change', array('id' => 'change')); ?>
</div>

<!-- If transmit power has succesfully been changed -->
<?php if(Yii::app()->user->hasFlash('success')): ?>

    <!-- Echoes a message note that transmit power has been succesfully changed -->
    <div class="flash-success">
        <?php echo Yii::app()->user->getFlash('success'); ?>
    </div>

<?php endif; ?>

<?php $this->endWidget(); ?><!-- Widget ends -->

```

Functional test of login and logout

```
class SiteTest extends WebTestCase
{
    public function testLoginLogout()
    {
        $this->open('');

        // test login process, including validation
        $this->assertElementPresent('name=LoginForm[username]');
        $this->type('name=LoginForm[username]', 'demo');
        $this->click("//input[@value='Login']");
        $this->waitForTextPresent('Password cannot be blank. ');
        $this->type('name=LoginForm[password]', 'demo');
        $this->clickAndWait("//input[@value='Login']");
        $this->assertTextNotPresent('Password cannot be blank. ');
        $this->assertTextPresent('Logout');

        // test logout process
        $this->clickAndWait('link=Logout (demo)');

        // test admin login process, including validation
        $this->assertElementPresent('name=LoginForm[username]');
        $this->type('name=LoginForm[username]', 'admin');
        $this->click("//input[@value='Login']");
        $this->waitForTextPresent('Password cannot be blank. ');
        $this->type('name=LoginForm[password]', 'admin');
        $this->clickAndWait("//input[@value='Login']");
        $this->assertTextNotPresent('Password cannot be blank. ');
        $this->assertTextPresent('Logout');

        // test logout process
        $this->clickAndWait('link=Logout (admin)');
    }
}
```