**CS673F13 Software Engineering** 

**Group Project 2 - MySurvey**

**Software Design Document**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Fliden | Project Leader | *FD* | 11/12/2013 |
| Gloria | Design Lead | *GJ* | 11/14/2013 |
| Imaad | Implementation lead (joint) | *MIK* | 11/15/2013 |
| Chao-Te | Environment and  Integration Leader | *CY* | 11/22/2013 |
| Chih-Chi | Implementation lead (joint) |  |  |
| Fyodor Wolf | Backup Project Leader / Configuration Leader |  |  |
| Samuel Tan | Testing Lead | *ST* | 11/21/2013 |
| Beth Haines | Requirements Lead | *BH* | 11/12/2013 |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
| **1.1** | **Beth Haines** | **11//8/2013** | **Added Introduction** |
| **1.2** | **Gloria Jensen** | **11/11/2013** | **Added Design Patterns & Software Architecture** |
| **1.3** | **Fyodor** | **11/11/2013** | **Added Unique Hash Algorithm** |
| **1.4** | **Fliden** | **11/12/2013** | **Added minimum features (from SPMP)** |
| **1.5** | **Beth Haines** | **11/12/2013** | **Added additional features** |
| **1.6** | **Fliden** | **11/12/2013** | **Added classes and methods section** |
| **1.7** | **Beth Haines** | **11/12/2013** | **Added database design descriptions** |
| **1.8** | **Beth Haines** | **11/18/2013** | **Added more to the introduction** |
| **1.9** | **Beth Haines** | **11/20/2013** | **Added software architecture diagram** |

[Introduction](#h.87t9hln2vjz0)

[Software Architecture](#h.buttcq9i221r)

[Design Patterns](#h.x18fj36s1121)

[Key Algorithms](#h.mtfbusfb0eq3)

[Classes and Methods](#h.7ucksmkf6rzx)

[References](#h.15tmymhipvdv)

[Glossary](#h.8n34lvocupub)

# Introduction

This SDD will outline the project requirements for mySurvey, both the core set of features and additional features, explain the software architecture and design patterns that we intend to use, and the data base design, including an explanation of the fields and relationships. The document will also explain the algorithms we are using to hash certain fields, and the key classes and methods.

## Project Overview

MySurvey is an internet application that will provide the ability to create, take and analyze surveys. We anticipate that our survey creators will be data analysts from corporations and from the education sector, that will be looking for data from their designated target group of survey-takers. The users will be both survey-creators, our customers, who create the surveys and survey-takers who answer the questions on the survey and provide the data to the survey-creators.

The mySurvey application has two sets of features, a core set, which is the basic functionality we expect to deliver to the customer, and a set of additional features, which are “extras” that will be scheduled as time permits.

The core set of features is comprised of:

1. User Login

A survey-creator will be able to login and gain access to his surveys by entering his username and password.

2. User Registration

A new user will be able to create an account with username, password, first and last names.

3. Create Survey

A survey creator will be able to create a new survey with a unique URL and will be able to add basic questions to the survey that could have different question types, such as single choice (choose the one correct answer), multiple answer (choose all of the correct answers for that question) and a question for entering comments.

4. View Created Surveys

A survey creator will be able to view all the surveys and survey questions that he has entered onto the site.

5. Take Survey

A survey-taker will be able to access the survey and to enter his answers.

The additional features include:

1. Templates for question types   
 These templates will include sample questions like “What is your age?”, and

ideally questions for different subject areas like “Which products have you used?

2. Branching from one question to a different question depending on the answer

This will allow the survey-taker to proceed from one question to the next logical question, depending on his answer. For instance, if the question was something like “Are you a student?”, a “yes” answer might lead to the question “What courses are you taking?”, while a “no” answer might lead to the question “What is your profession?”

3. Creating an “About Us” page

This page will give some basic information about our company and the kinds of surveys they’ll be able to create with our product.

4. Saving the progress of the survey

This functionality will allow the survey creator to partially complete a survey, save it, and return to it at a later time

6. Additional reports

Additional reports will show statistics for how many people took the survey and the numbers of different responses to the questions. This may include some data analysis, as well.

5. Scheduling a start time for the survey

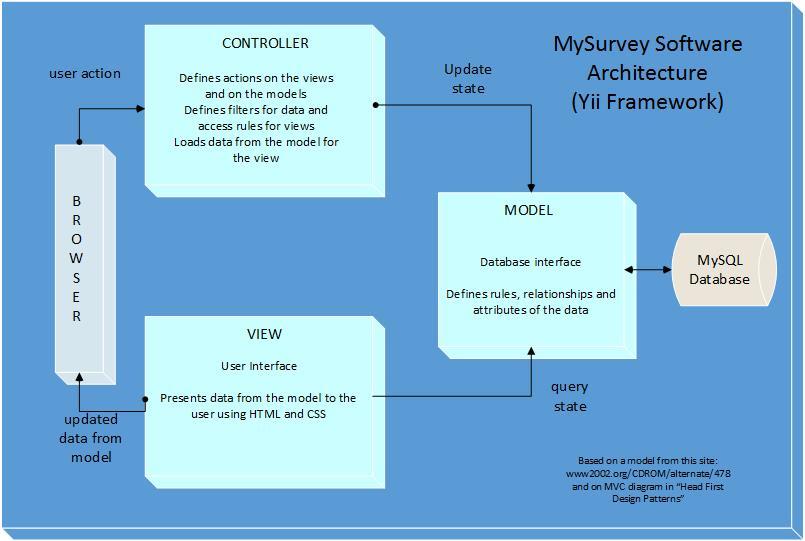
The survey creator will be able to set a time when the survey will become available for the survey takers.

6. Creating different functionality for different levels of users

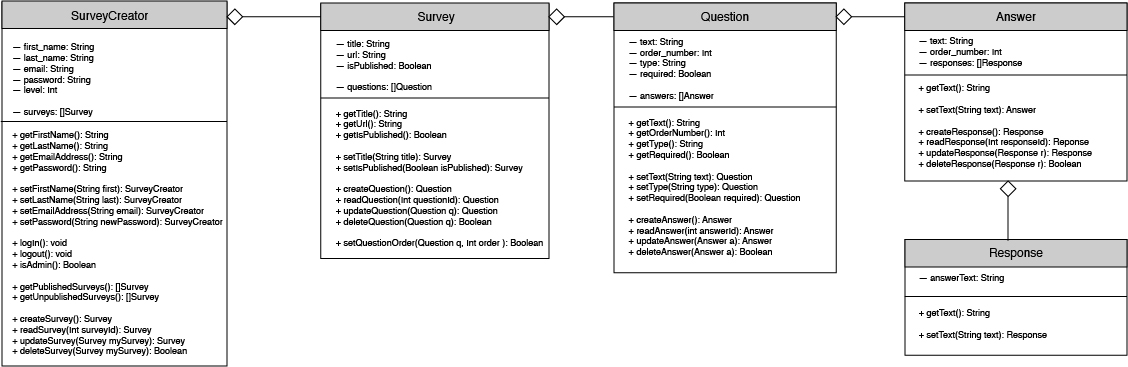
Currently, the Register function automatically assigns a level of 0 for Administrator. More functionality will be added to distinguish between different kinds of users and permit different access depending on the level. For instance, a level 0 user will be able to delete any survey. A level 1 user will only be able to delete his own surveys, etc.

# Software Architecture

The MySurvey application uses a repository architecture that performs transactions against our SQL database to create, delete, edit, publish, unpublish and view surveys. In addition, it securely maintains the Survey Creator account information.



MySurvey is comprised of the following components: SurveyCreator, Survey, Question, Answer and Response. The relationship and dependencies between components is illustrated in the class diagram and summarized below.



**MySurvey Component Dependencies:**

* A Survey Creator can create an array of Surveys.
* Each Survey is comprised of an array of Questions.
* There are different types of Questions. Depending on the type of Question, each can have at least 1 Answer and up to many Answers.
* Each Answer can have a potential Response that the Survey Creator will want to record for reporting purposes. *We are handling reports in Iteration 3. More information to come as we complete that feature.*

**Database Design:**

The database consists of five main tables, survey\_creator, survey, survey\_question, survey\_answers and survey\_response. (Note: tbl\_migration is shown on the diagram, and is only used by the yii framework for version control.)

***survey\_creator*** table

Description:

***survey\_creator*** contains data about each survey creator.

Attributes:

id int(11) required

Primary key

Auto-generated sequential number

email varchar(45) required

Email address of the survey creator

Entered by the survey creator in the Register function

Must be unique

password varchar(45) required

Password of the survey creator

Entered by the survey creator in the Register function

and then hashed with the hashing algorithm described below

under “Key Algorithms”

Must be at least 8 characters and match when re-entered

first\_name varchar(45)

First name of the survey creator

Entered by the survey creator in the Register function

last\_name varchar(45)

Last name of the survey creator

Entered by the survey creator in the Register function

level int(11) required

Level for the user (0 = “Administrator”, etc.)

Generated in the Register function

***survey*** table

Description:

***survey*** contains data about each survey created by the survey creators.

Attributes:

id int(11) required

Primary key

Auto-generated sequential number

url varchar(80) required

Unique URL where survey takers will be able to access the survey  
 Generated by the system based on a common url plus the unique survey ID

created datetime required

Time stamp when the survey is created

Generated by the system

survey\_creator\_id int(11) required

ID of the survey creator creating the survey

Foreign key to survey\_creator table

is\_published varchar(1)  
 Flag set when the survey is published and available to the survey takers

Set to “0” when the survey is created,

can be changed to 1 through the “Publish/Unpublish” functionality title varchar(100) required

Title or name of the survey

Entered by the survey creator when the survey is created

Must be at least 6 characters

***survey\_question*** table

Description:

***survey\_question*** contains data about each survey question created by the survey creators for a particular survey.  
Attributes:

id int(11) required

Primary key

Auto-generated sequential number

survey\_id int(11) required

ID of the parent survey

Foreign key to survey table

order\_number int(11) required

Order number of the questions in the survey

Set by the application when the question is created or resequenced

type int(11) required

Type of the survey question

Set by the application when the question is created

text varchar(1000)

Text of the question, example “Are you a student?”

Entered by the survey creator when the survey question is created

***survey\_answer*** table

Description:

***survey\_answer*** contains data about each survey answer for a particular question on the survey created by the survey creators. (Note: If the question is a short-answer type, a “blank” answer record will be created for that question without text. This will keep the structure between question ->answer ->responses.)

Attributes:

id int(11) required

Primary key

Auto-generated sequential number

survey\_question\_id int(11) required

ID of the parent survey question

Foreign key to the survey question table

survey\_answer\_response\_time time

Amount of time (maximum) to answer the question

Entered by the survey creator when the survey answer is created

survey\_answer\_next\_link varchar(80)

Link to next survey question

Entered by the survey creator when the survey answer is created

(future functionality)

text varchar(1000)

Pre-supplied answer to a survey question

Entered by the survey creator when the survey answer is created

order\_number int(11) required

Order of the answer in that question (used for multiple choice)

Set by the application when the survey is created or when the answers

are resequenced

***survey\_response*** table

Description:

***survey\_response*** cotnains data about each survey response created by the survey takers.

Attributes:

id int(11) required

Primary key

Auto-generated sequential number

survey\_answer\_id int(11) required

ID of the parent survey answer

Foreign key to the survey answer table

survey\_answer\_choice\_letter varchar(5)   
 Letter for the answer choice (used for multiple choice, multiple answer)

Entered by the survey creator when the survey answer is created

survey\_response\_time time

Amount of time it takes the survey taker to answer the question

Calculated by the application (future functionality)

survey\_response\_responder varchar(45) required

Hashed value to determine the ID of the survey taker

Set by the application when the survey taker starts taking the survey

survey\_response\_text blob

Text of the response

Entered by the survey taker when responding to the survey question

The business rules are as follows:

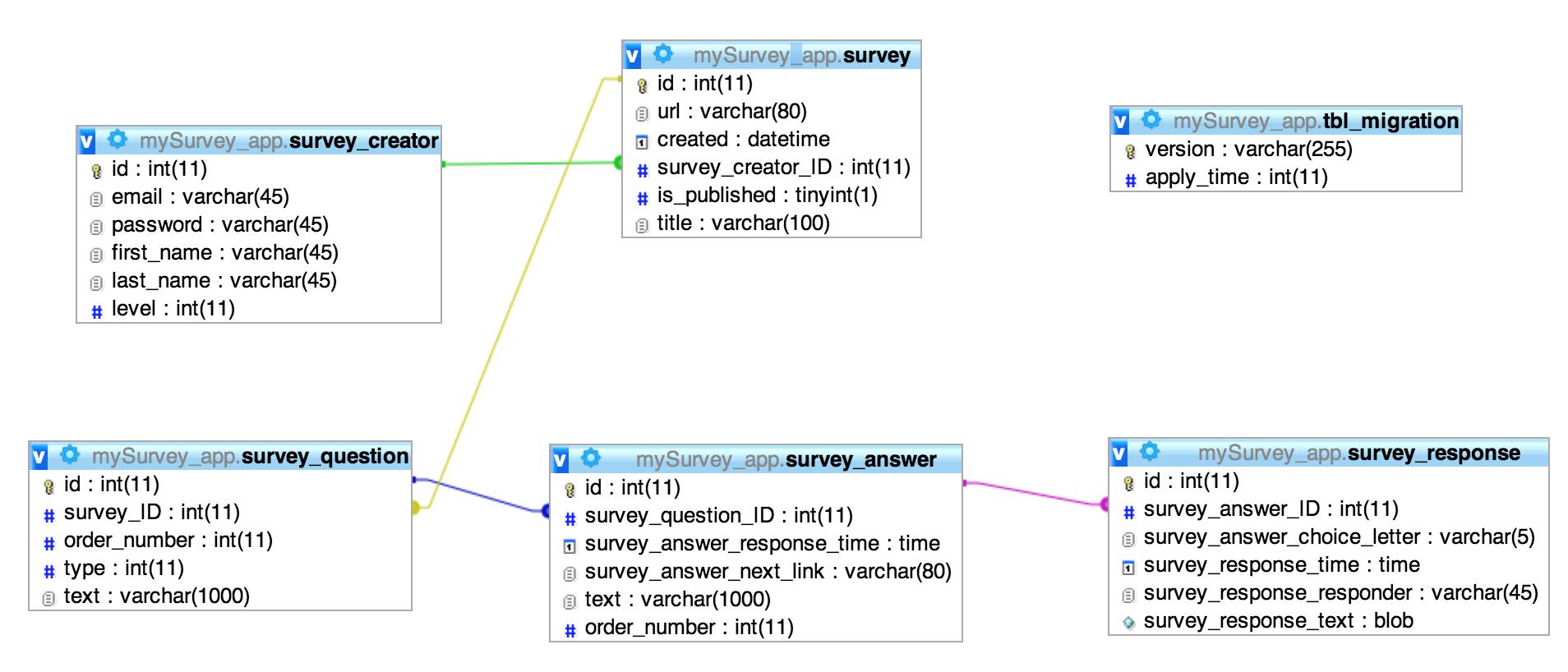
1. There will be one account created for each survey\_creator.

2. Each survey\_creator can have many surveys.

3. Each survey can have many questions.

4. Each question instance of survey\_question can have many answers.

5. Each answer instance of survey\_answer can have many responses.



# Design Patterns

We are using the Model-View-Controller (MVC) design pattern which separates our business logic from the user interface. The design of the MySurvey website is as follows:

**Models:** These files represent the underlying data structure of the MySurvey application. They include the validation rules for logging in and maintaining the Survey Creator accounts. In addition, they include all of the logic needed to create complex surveys.

*MySurvey Models: LoginForm.php, Survey.php, SurveyAnswer.php, SurveyCreator.php, SurveyQuestion.php, SurveyResponse.php*

**View:** These files are responsible for presenting the MySurvey models in a format that our end users can interact with. This includes the homepage with the login and register screen and all of the required web pages to create, update and view surveys.

*MySurvey Views: Homepage, Surveys (landing page), Create a Survey, Survey Editor, Reports (landing page), Account Settings (landing page)*

**Controller:** These files bind our models and views together into the runnable MySurvey application. Our controllers are responsible for dealing directly with end user requests such as displaying the login page, processing functions that access the SQL database, and publishing / unpublishing surveys.

*MySurvey Controllers: SiteController.php, SurveyController.php, QuestionController.php, AnswerController.php, ResponseController.php*

# Key Algorithms

Unique Hash Generator:

When generating a unique hash for a survey, the application creates a random string of a supplied length (default 6 if not supplied) out of lower case, upper case and alpha-numberic characters. It makes a recursive call to regenerate another string if a matching hash already been stored in the databases.

|  |
| --- |
| private function generate\_unique\_url($length = 6){  $result = “”  for(0->$length) $result .= get\_random\_alpha\_num\_char();  if(find\_model\_with\_hash($result)){  $result = generate\_unique\_url($length);  }  return $result;  } |

Survey Edit fix\_sortable\_input\_names:

Whenever changes are made on the survey update form, that affect other questions or answers, the fix\_sortable\_input\_names function is run in order to re-construct the form’s input’s name attributes. This is done in order to have a well structured $\_POST on the survey side that reflect model relationships and order.

fwolf TODO: ELABORATE ON HOW THIS WORKS.

# Classes and Methods

<https://github.com/fydo23/mySurvey/blob/master/mySurvey_app/phpdoc/>

or zipped file:

<https://github.com/fydo23/mySurvey/blob/master/mySurvey_app/phpdoc/phpdoc.zip>

# References

Aasen,Rick; Carlisle,Ed; Carson, Nick; Cerauskis, Leland; Jeffers, Eric; Green, Travis;

Matson, Blake ; Ritchey, Josh; Vo, Phuong, “Software Design Document”, [www.cise.ufl.edu/class/cen3031fall/**documents**/examples/SDD\_Example\_1\_2011.pdf](http://www.cise.ufl.edu/class/cen3031fa11/documents/examples/SDD_Example_1_2011.pdf)

Yii Framework, “Best MVC Practices”, <http://www.yiiframework.com/doc/guide/1.1/en/basics.best-practices>

# Glossary