CS-M14 Industrial Project
Design Document

Matthew Lewis
523015@swansea.ac.uk

25th May 2012

Contents

1 Introduction

This document contains a semi-formal design specification for the architecture used in both the Customer Web Application and the Administration Web Application.

2 System Architecture

The architecture used for both the Customer and Administration Web applications is loosely based on the Model-View-Controller (MVC) architecture. The MVC in architecture, in its simplest form, is where the user interface or *view* of the application is developed separately from the actual application logic.

Model The part of the application is where the main functions of the application are contained. For example, if interaction with a database is involved, it is where the various transaction functions are defined.

Controller The Controller is involved with processing the interaction with the users of the system. After receiving an input from a user it uses the relevant model to perform certain actions.

View The part of the architecture is the user interface of the application. The data displayed on the view is 'pushed' from the model to the view.

However, the applications utilise a modified MVC architecture by only using controllers and views. Each web page of the system that is accessible on the web can be seen as a *controller* that performs the program logic and the retrieval/updating of data. Each controller interacts with a number of Data Access Objects that provide an abstract layer on top of the database by providing the functions that will be performed on the database. The controller then displays the appropriate view to the user.

3 Data Access Objects

For each the four basic groups of functionality I have produced a data access object that includes all the low- level queries that interact with the database. These queries include reading from the database, inserting new records, and deleting records. By using the data access objects I have avoided the possibility of having to replicate the data access code in each of the controller files. They provide an abstract interface to the controllers so that data can be retrieved, updated and deleted without any attention to the specific queries being performed.

3.1 Samples

getUnsubmittedSampleByID Returns the information about an un-submitted sample using the the specified sample ID.

getSampleByCode Searches in the 'Samples In' table for a sample with the specified sample code and returns the information stored about it.

SampleDAO
+ getUnsubmittedSampleByID(sampleID: int)
+ getSampleByCode(sampleCode: String)
+ getSampleByRef(sampleRef: String)
+ getSubmittedSampleByID(sampleID: int)
+ getSampleCounts(swanCode: String, yearCode: String)
+ updateAnalysisData(sample: Array, data: Array)
+ countSamples(stage: int, swanCode: String, yearCode: String, fromDate: int, toDate: int)
+ deleteSample(sampleID: int)
+ saveSample(sample: Array)
+ checkIfUniqueSampleRef(sampleRef: String, swanCode: String)
+ getUnsubmittedSamples(swanCode: String, limit: int, offset: int, from: int, to: int)
+ getSubmittedSamples(swanCode: String, limit: int, offset: int, from: int, to: int)
+ getReceivedSamples(swanCode: String, limit: int, offset: int, from: int, to: int)
+ getInProgressSamples(swanCode: String, yearCode: String, limit: int, offset: int, from: int, to: int)
+ getCompletedSamples(swanCode: String, yearCode: String, limit: int, offset: int, from: int, to: int)
+ submitSample(sample: Array)
+ unsubmitSample(sample: Array)
+ bookInSample(sample: Array)
+ completeSample(sample: Array)
+ canModifySample(swanCode: String, sampleID: int)
+ getNextSampleCode(swanCode: String)

getSampleByRef Searches both the 'Samples In' and 'Samples Pending' tables for a sample with the specified sample reference and returns the information stored about it.

getSubmittedSampleByID Returns the information about an already submitted sample using the the specified sample ID.

updateAnalysisData Used to update the data about what machines or techniques have been performed on a sample.

countSamples Counts how many samples have been submitted under a specific Swan code.

deleteSample Removes a sample record from the system.

saveSample Inserts a new sample into the database or updates an existing one.

checkIfUniqueSampleRef Used to check that a user has not already specified an existing sample with the sample reference they are attempting to use.

getUnsubmittedSamples Returns a list of un-submitted samples which can be limited and offset for paging purposes. It can also be filtered using from and to dates. It is also possible to specify a Swan code so that the results are limited to that Swan code.

getSubmittedSamples Returns a list of submitted samples which can be limited and offset for paging purposes. It can also be filtered using from and to dates. It is also possible to specify a Swan code so that the results are limited to that Swan code.

getReceivedSamples Returns a list of received samples which can be limited and offset for paging purposes. It can also be filtered using from and to dates. It is also possible to specify a Swan code so that the results are limited to that Swan code.

getInProgressSamples Returns a list of 'In Progress' samples which can be limited and offset for paging purposes. It can also be filtered using from and to dates. It is also possible to specify a Swan code so that the results are limited to that Swan code.

getCompletedSamples Returns a list of 'In Progress' samples which can be limited and offset for paging purposes. It can also be filtered using from and to dates. It is also possible to specify a Swan code so that the results are limited to that Swan code.

submitSample Moves an existing sample from the 'Samples Pending' table to the 'Samples In' table and updates the current stage the sample is at.

unsubmitSample Moves an existing sample from the 'Samples In' table back to the 'Samples Pending' table and updates the current stage the sample is at.

bookInSample Used to mark that the sample has been assigned a sample code by the lab and update the stage the sample is at.

completeSample Used when the sample has been through all stages and needs to be marked as completed.

canModifySample Checks the Swan code of the sample to ensure that the current user can make modifications or view the sample.

getNextSampleCode When booking in samples they are given a unique sample code which is assigned by the lab. The code is calculated sequentially. This function checks what the previous sample code was and increments the value.

3.2 Submitters

SubmittersDAO
+ getSubmitterByCode(submitterCode: String)
+ getSubmitters(swanCode: String)
+ removeSubmitter(submitterCode : String)
+ updateSubmitter(submitterCode: String, name: String, phone: String, email: String)
+ createNewSubmitter(swanCode: String, submitterCode: String)
+ canModifySubmitter(swanCode: String, submitterCode: String)

getSubmitterByCode Retrieves the details about a specific submitter.

getSubmitters Retrieves a list of all submitters that have been created under a specific Swan code.

removeSubmitter Removes a submitter from the system.

updateSubmitter Saves the changes to the submitters details into the database.

createNewSubmitter Inserts a new submitter with the specified Submitter code into the database.

canModifySubmitter Used to check whether the current user is able to modify or delete a specific submitter by ensuring that the Swan code of the submitter matches the Swan code of the current user.

3.3 Users

UsersDAO
+ getUsers()
+ getUserBySwanCode(swanCode: String)
+ authenticateUser(userID: String, password: String)

getUsers Returns a list of all the users currently in the database.

getUserBySwanCode Retrieves the details of a specific user by using their Swan code.

authenticateUser Checks a users Swan code and password against those stored in the database.

3.4 Settings

SettingsDAO
+ getSettings()
+ updateSetting(name: String, value: String)

getSettings Retrieves all the current settings from the database such as the current year code and administrator password.

updateSetting Used to change the value of a specific setting. Used, for example, when changing the current year code.

4 Controllers

Samples

The samples controller handles all the logic for retrieving and displaying the five different lists of samples, one for each stage of processing. It performs functions such as the pagination of results, controlling which list to display based on what tab is currently selected and filtering results by a specified date range.

4.1 Submitters

The submitters controller is used to manage the list of submitters that are associated with a Swan code. The controller handles operations such as the addition, editing and deletion of submitters.

4.2 Submit

The submit controller handles the submission of a new sample to the system. It checks that all required fields are completed. It also allows users to save a sample and resume the submission process at a later point in time.

4.3 Confirm

Before a sample is submitted the user must confirm the details of the sample. The confirm controller is used to display the sample in a confirmation page from which the user can submit their sample or return to make changes to it.

4.4 Draw

The draw controller is simply used to provide the user with the chemical structure drawing tool so that they can draw their own structure for a specified sample. If a sample file already exists for the sample, the controller will read its contents and output to the sketcher interface allowing for modifications to be made to the structure.

4.5 Sample

The sample controller handles the display of sample data to the user. The controller first checks that the user is allowed to view the sample. It also reads the contents of the structure file, if one is provided, so that the structure can be drawn graphically using a JavaScript & HTML5 canvas library.

With the administration application, the sample controller also handles the movement of samples through each stage such as booking in, marking as 'in progress' and completing a sample.

4.6 Users

The users controller is used to display information about a specific user such as their details and statistics about how many samples they have submitted.

4.7 Settings

The settings controller handles the changes to the settings of the system such as the administrator password and the year code. p

4.8 Results

The results controller for the customer application is used to download the result file for a given sample. The controller checks that the current user is allowed to download the result file first. If they are, the controller looks for the file on the server and offers it for download to the user.

With the administration application the results controller is used to attach the result files to a sample. The controller takes a list of files, creates a zip archive of them and moves the zip file into the results directory from where it can be downloaded.

5 Database Design

5.1 Table Design

The design of the tables is restricted by the fact that the existing system is being used simultaneously and interacting with the same data. Due to this, the structure of the tables had to remain the same. However, it was possible to add additional fields to the tables so that the extra functionality could be added in the new system.

Previously the system used the date fields, such as DateCreated or DateInLab, to determine which stage is sample is currently at. To simplify this process a new field SampleStage was added to the samples table which is used to track the current stage of a sample.

5.1.1 Samples In/Pending Tables

Field	Type	Null	Comments
Sample_id	int(10)	No	Unique ID of the sample
Swan_code	char(8)	No	Swan code of the submitter
Submitter_code	char(10)	No	Submitter code of the submitter
Year_code	char(4)	No	
Sample_code	varchar(2)	Yes	Lab's sample code
Sample_ref	varchar(20)	No	Customers sample reference
DateCreated	datetime	Yes	
DateCompleted	datetime	Yes	
MolFormula	varchar(50)	Yes	Molecular Formula
MolWeight	varchar(20)	Yes	Molecular Weight
MeltPoint	varchar(20)	Yes	Melting point of sample
Solvents	varchar(50)	Yes	
Supervisor_email	varchar(50)	No	
Submitter_name	varchar(50)	No	
Submitter_telephone	varchar(50)	No	
Submitter_email	varchar(50)	No	
NMSSC	int(11)	Yes	Identifies what services are required
EICI	int(11)	Yes	Identifies what services are required
ESI	int(11)	Yes	Identifies what services are required
FAB	int(11)	Yes	Identifies what services are required
MALDI	int(11)	Yes	Identifies what services are required
AccMass	int(11)	Yes	Identifies what services are required
GCMS	int(11)	Yes	Identifies what services are required
LCMS	int(11)	Yes	Identifies what services are required
COSHH	char(10)	Yes	COSHH rating of sample
COSHHtext	varchar(250)	Yes	
SpecialRequirements	text	Yes	
DateSubmitted	datetime	Yes	
DateReceived	datetime	Yes	
DateInLab	datetime	Yes	
struct_file_recd	int(11)	Yes	Structure file provided?

struct_filepath	varchar(200)	Yes	Location of file
struct_filesize	int(11)	Yes	Size of structure file
SampleStage	int(11)	No	Current stage sample is at
LRy	double	Yes	Machine/Technique data
HRy	double	Yes	Machine/Technique data
LFy	double	Yes	Machine/Technique data
HFy	double	Yes	Machine/Technique data
LESy	double	Yes	Machine/Technique data
HESy	double	Yes	Machine/Technique data
LMy	double	Yes	Machine/Technique data
HMy	double	Yes	Machine/Technique data
GCy	double	Yes	Machine/Technique data
HGCy	double	Yes	Machine/Technique data
LCy	double	Yes	Machine/Technique data
HLCy	double	Yes	Machine/Technique data
NIL	double	Yes	Machine/Technique data
OA	double	Yes	Machine/Technique data
results	int(1)	Yes	Results been attached to sample?
DateOut	datetime	Yes	

5.1.2 Submitters Table

Field	Type	Null	Comments
Submitter_code	char(10)	No	Unique code to identify the submitter
Swan_code	char(8)	No	Swan code of the submitter
Supervisor_email	varchar(50)	Yes	
Submitter_name	varchar(50)	Yes	
Submitter_telephone	varchar(50)	Yes	
Submitter_email	varchar(50)	Yes	

5.1.3 Users Table

Field	Type	Null	Comments
Swan_code	varchar(8)	No	
Surname	varchar(50)	No	
Title	varchar(10)	Yes	
Initials	varchar(7)	Yes	
First_name	varchar(30)	No	
Institution	varchar(36)	No	
department	varchar(45)	No	
Address_line_1	varchar(40)	Yes	
Address_line_2	varchar(40)	Yes	
TownCity	varchar(20)	Yes	
Postcode	varchar(20)	Yes	
Telephone	varchar(20)	No	

Fax	varchar(20)	Yes
email_address	varchar(50)	No

5.1.4 Passwords Table

Field	Type	Null	Comments
s_code	varchar(8)	No	Swan code
p_sample	varchar(40)	No	Password for the sample system
p_date	varchar(30)	No	Date of last login

5.1.5 Settings Table

Field	Type	Null	Comments
Name	varchar(20)	No	Name of the setting
Value	varchar(20)	No	Actual value of setting