Testing Specification

for

Campus Ministry Software

Version 1.0 approved

Prepared by Zahara Kazmi

Loyola University Maryland

April 5, 2017

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

The Software Design document is produced to describe the design of the software system we are developing. This document will provide details for how the software will be built and aid in the development. The document will be a graphical and narrative description of the software design including architectural style, use case model, sequence diagram and class diagrams

1.1 Goals and objectives

This project will include a Web interface with will be connected to a database as well as a Machine Learning Database for Event Planning. The software will include use for Campus Ministry Staff, Interns and Guest users. The database will store information for Staff, interns, hours worked, projects working on and program planning. The interns and Staff will interact with database via Web interface. The Machine Learning Algorithm will be used to query for program planning for specified amount of attendance.

1.2 Statement of scope

This Software Design Document is to explain in detail an overall architecture of the software as well detailed description of the software processes. This document will work as a proof of concept for the building of software and base level functionality understanding for the client. This document will include diagrams and architecture which include a high-level design as well as modules and components design. This document will be used in conjunction with the SRS document and expanding on the classes created and classes to be developed.

1.3 Software context

This Software will enable Campus Ministry to store and transform data into information to support work flow. This software will replace previous work flow of using a file system. This software will allow staff and interns to have concurrent access to the database as well to guest in order to enter information. The database structure will improve integrity of the data as well providing a single location access point for all staff and guest members from any location. Data redundancy and inconsistency would be removed and data will come directly from source instead being manually entered by interns. It will increase the security and reliability of the office in collected information. Improvement in the various qualities will improve data quality as well thus resulting in improved and faster work flow.

Another aspect of this software is the Machine Learning Algorithm which will be providing suggestions for planning future events based on information from previous event. This will allow interns to have targeted audience to improve quality of the events.

1.4 Major constraints

Machine Learning Algorithm implementation of this software does not have data to be trained on and tested on. The office has already combining ed their collected data into one file and lost data for over the years. Thus, the algorithm needs to be confirmed via pseudo-data or use unsupervised learning algorithm. The Algorithm will be confirmed based on accuracy for the specified algorithm and its functionality to work with the different features.

1.5 References

In order to develop this document several different online resources are being implemented for illustrations and graphical representation. Draw.io is used to create and implement the graphical representation of software. A Table of Figures is included to reference all the various figures within this document.

2. Test Plan

This section describes the overall testing strategy to be implemented for Campus Ministry Software. It describes in detail the project management issues that required to execute the tests effectively.

2.1 Testing Strategy

The Campus Ministry Software heavily depends user-interface and how well users are able to use the system. As a result, this software will be tested using a mixture of white and black box testing. Black-box testing will be conducted by sample user who will fill out the forms and run different queries to get the results. The sample user

2.1.1 Unit Testing

This software contains three different components and of those components use different types of implementation thus each will require different unit testing mechanism. Unit testing for each of the components will be done by several different tools. The software is a test-driven design and implementation thus it will be tested as it is being made. For each of the different components of the software the testing strategy is described below along with the component.

2.1.1.1 Web Interface/Forms Unit Testing Strategy

This component of the software is the primary interaction component for all the users of the software. This component encompasses both the Guest forms – to be used by program attendees for - and user log-in pages to be used by campus ministry staff. These forms will be testing tools as well as by sample users. The testing tools to be implemented for the Web forms include using Selenium, a portable software-testing framework for web applications. Another tool that will be used for web interface is Parasoft which will allow for automated peer code reviews preparation, notification and tracking. Both of these

2.1.1.2 Database Unit Testing Strategy

One of the major components of this software and the main back end for this software is the MySQL database. The data entered via web forms will be stored in the data. The two components we need to test for the database is storage of data in each of the tables and running specific queries. The database will be tested using TAP which allow us to write MySQL testing scripts. These test scripts will be written for each of the table and for the queries as well.

2.1.1.3 Machine Learning Algorithm Unit Testing Strategy

The last component of this software is the Machine Learning Algorithm and this will be programmed on python. In order to evaluate and test each of our function, Python testing scripts will be created. The Python scripts will allow us to do coverage testing for each of the method and we can check using sample data how well our algorithm works.

2.1.2 Integration Testing

The Campus Ministry Software involves three different components and integration testing will be a crucial part in its implementation. For integration testing, we would still have to test each of our components but instead of how well each of the function does within these components we have to look at the component as a whole.

2.1.2.1 Web Interface/Forms Integration Testing Strategy

The data input forms will be tested using a sample user pool to test how well users are able to actual use the web interface. Users will be asked to fill out the forms without any guidance and the amount of time it takes for them to correctly fill out form without any errors will be our measuring tool. An automated tool for integration testing which is to be implemented “TestingWhiz”. This will allow us to measure several different aspects of testing and automate forms filling as well.

2.1.2.2 Database Integration Testing Strategy

The database will be tested by running queries on the whole database. We will request information which combines more than one table. MySQL and Tap scripts will be used to write the queries and run them to test how well and how quick the results are returned. An automated tool which will be implemented for database testing is “Mydebugger” will allow us to run testing on the database and will provide even more coverage.

2.1.2.3 Machine Learning Algorithm Integration Testing Strategy

The AI component of this project will have Integration testing built into since our algorithm will provide us an overall accuracy percentage. This accuracy percentage will tell us how well our algorithm is doing based on our sample data. Another testing tool which we can add for testing the algorithm is writing scripts which test the algorithm to get higher accuracy. The Machine Learning will be tested via scripts and the accuracy within the algorithm will be used as measurement tool for how well our code is written.

2.1.3 High-Order Testing

2.1.3.1 Whole System Testing

This software will have three different components combined and working efficiently once it is completed thus we would need to have sample users use this system to evaluate it. Due to the complexity of the software, it is difficult to test the program with real people using this software. As a result, we will have two different groups who will be testing this software. One of the sample group will be the white box tester who would be people who familiar with coding and aware of how this project is developed. The other group will be our black box testers who will have no idea about how the back-end is processed and how it is being developed. Using two different groups will give us an idea about how well people accustomed to programming can use and for those who are not accustomed to programming. Using sample users will also allow us to get input about what component does not work well with the others.

2.1.3.2 Usability Testing

Usability testing will be done with real users and it will be measured how long it takes for then to learn the system. As a sample user uses the system, it will be measured how long it takes them to really understand and how well they understand the software. This will be done in two different ways, remote and on location. For remote testing, users will be sent the forms and they will be asked to fill them out followed by a survey about how well the software worked for them. On Location testing will be done at Campus Ministry staff meeting and users will be asked to use this software without any explanation. The systems usability will be tested how seeing how long it takes for them to start using system without pauses and unsure about outcomes.

2.1.3.3 Maintainability Testing

The maintenance of this software is solely dependent on the administrator thus to test the maintainability we would have to test how well he is able to use the back-end of the software. This will be done by providing the admin with a user manual and they will be asked to fix issues and update the software to check how well they are able to complete those tasks. The user will not be provided with any background except for the user manual and they how quickly they are able to learn to use this system.

2.1.3.4 Performance Testing

The performance of the software is crucial since we are working with several different components which combine and their mobility as a unit is very important. One way to measure the performance of our software will be to execute load testing. This will allow us to check how well our software does when it is being requested and used by multiple people. We will use a Query Surge, a tool which will allow us to run load and stress testing on the software to measure its performance.

2.2 Testing tools and Environment

As we have mentioned for each of the components above, we will be using several different tools to test the program. Below we have listed all the different tools and what component they will be used for. These tools are linked in the Appendices section.

Component	Testing Tool	Programming Language of Component
Database	TAP	MySQL
Database	MySQL scripts	MySQL
Database	Mydebugger.com	MySQL
Database	STK/Unit	MySQL
Web Interface	PHPunit.de	PHP
Web Interface	phpt	PHP
Web Interface	Simpletest.org	PHP
Web Interface	Star Tutorial	PHP
Web Interface	Selenium	HTML
Web Interface	Parasoft	HTML
Web Interface	Query Surge	HTML
Machine Learning	Python Scripts	Python

Figure 1 Software Component and Testing Tool Table

2.3 Test Schedule

Due to the complexity of this software and its integration of different components, this software must be a test-driven development. The unit testing for the software is completed as it is being developed and the integration testing is done as each section is completed. Since most of the software is user input and user involved, this needs to be tested by users to better measure how well our software does. Below we have included a tentative schedule for when each of the testing should be done.

Testing Type	Starting Date	End Date
Unit Testing	03/10/2017	04/15/2017
Integration	03/29/2017	04/17/2017
System Testing	04/15/2017	04/18/2017
Performance	04/19/2017	04/25/2017
Maintenance	04/24/2017	05/01/2017

Figure 2 Testing Schedule for Software

3. Test Procedure

3.1 Unit Test Cases

3.1.1 Unit test Case Input and Expected Result

The web forms need to be tested for each of their field for acceptable input and result. Each of the forms have been explained in their own section below.

3.1.1.1 Attendee Form Test Case and Expected Result

This is the form to be used for guest users who attend campus ministry events and to get their information for future use. In the table below we have listed all the information being requested, type of input acceptable, input given and expected result for wrong input. For the attendee form only the information below needs to be tested because users will be checking boxes for the other information.

Test Case #	Field Name	Valid Input	Given Input	Result
1	Name – first and last	Zahara Kazmi	1234456	Error – name must be a string
2	Name – first and last	Zahara Kazmi	23zahara23	Error – must only have letters
3	Name – first and last	Zahara Kazmi	Bobby Jones	No Error
4	Name – first and last	Zahara Kazmi	@xyzml	Error – must contain letters only
5	Phone Number	410-000-0000	410-216-2222	Acceptable input for field
6	Phone Number	410-000-0000	Xyz-xyz-xx23	Error- must be numeric
7	Phone Number	410-000-0000	@@-##-####	Error – must be numeric
8	Email	ex@gmail.com	zqk@loyola.edu	Acceptable input for field
9	Email	ex@gmail.com	123@gmail.com	Acceptable input for field
10	Email	ex@gmail.com	noyes@gmail.com	Acceptable input for field
11	Email	ex@gmail.com	Zwe123.com	Error – must be valid email
12	Email	ex@gmail.com	123@zahara.com	Error – must be valid email

Figure 3 Attendee Form Test Cases

3.1.1.2 Retreat Form Test Case and Expected Result

The retreat form will be used by students and staff members attending Loyola retreats to provide their information. This information is used by campus ministry to have for each member on the retreat to keep track of their dietary restriction, medical information and emergency contact. Below we have listed all the information being requested and the test cases to be carried for all the fields. We are only running test cases for inputs for fields where user will input information, if user will pick from drop down or check boxes from already existing option, it is not being tested since no wrong input can be given.

Test Case	Field/Information	Input Accepted	Input Given	Result
13	Name – first and last	Zahara Kazmi	1234456	Error – name must be a string
14	Name – first and last	Zahara Kazmi	23zahara23	Error – must only have letters
15	Name – first and last	Zahara Kazmi	Bobby Jones	No Error
16	Name – first and last	Zahara Kazmi	@xyzml	Error – must contain letters only
17	Class Year	####- 2019	2017	Acceptable input for field
18	Class Year	####- 2019	17	Error – must be year format
19	Class Year	####- 2019	Seven	Error – must be numeric
20	Class Year	####- 2019	@17	Error – must not contain symbols
21	Student ID	####### - 1657238	1658238	Acceptable input for field
22	Student ID	####### - 1657238	Xyzsyz	Error – must be numeric
23	Student ID	####### - 1657238	XYZ-143	Error – must be numeric
24	Student ID	####### - 1657238	16582389870	Error – must only be 7 digits
25	Phone Number	410-000-0000	410-216-2222	Accepted
26	Phone Number	410-000-0000	Xyz-xyz-xx23	Error- must be numeric
27	Phone Number	410-000-0000	@@-##-####	Error – must be numeric
28	Email	ex@gmail.com	zqk@loyola.edu	Accepted
29	Email	ex@gmail.com	123@gmail.com	Accepted
30	Email	ex@gmail.com	noyes@gmail.com	Accepted
31	Email	ex@gmail.com	Zwe123.com	Error – must be valid email
32	Email	ex@gmail.com	123@zahara.com	Error – must be valid email
33	Birth date	08/21/1995	08/21/1998	Accepted
34	Birth date	08/21/1995	04/11/2017	Error – must be at least 18
35	Birth date	08/21/1995	04/05/2017	Error – must be at least 18
36	Address Line 1	# Street	2	Error – must be street name
37	Address line 1	# street	2 avenue	Acceptable input for field
38	Address line 2	Letters/#	Apt 1	Acceptable input for field

39	Address line 2	Letters/#	Campus House	Acceptable input for field
40	City	Letters only	Baltimore	Acceptable input for field
41	City	Letters only	B12	Error – must be letters only
42	City	Letters only	@btm	Error – must be letters only
43	Zip code	5 numbers - 21234	2e34	Error- must be numbers only
44	Zip code	5 numbers	21210	Accepted
45	Zip code	5 numbers	Sefhkjsfh	Error – must be numbers only

Figure 4 Retreat Form Test Cases

3.1.1.3 Wedding Form Test Case and Expected Result

The wedding form will be used by perspective bride and groom to wedding and personal information to campus ministry for wedding preparation. The information which is being asked via selection is not being tested because the user will be given a selection to pick from and any answer can be chosen. Below we have listed all the test cases for Wedding Reservation form.

Test Case	Field/Information	Input Accepted	Input Given	Result
1	Wedding Date	Month-Day-year	Today's Date	Error- Wedding Date must be in future
2	Wedding Date	Month-Day-year	Past Date	Error- Wedding Date must be in future
3	Wedding Date	Month-Day-year	06/15/2017	Acceptable input for field
4	Rehearsal Date	Month-Day-year	Today's Date	Error- Wedding Date must be in future
5	Rehearsal Date	Month-Day-year	Past Date	Error- Wedding Date must be in future
6	Rehearsal Date	Month-Day-year	06/15/2017	Acceptable input for field
7	Name – first and last	Zahara Kazmi	1234456	Error – name must be a string
8	Name – first and last	Zahara Kazmi	23zahara23	Error – must only have letters
9	Name – first and last	Zahara Kazmi	Bobby Jones	Acceptable input for field
10	Name – first and last	Zahara Kazmi	@xyzml	Error – must contain letters only
11	Religious Affiliation	Words Only	Islam1234	Error – Must be letters only
12	Religious Affiliation	Words only	Islam	Accepted
13	Phone Number	410-000-0000	410-216-2222	Accepted

14	Phone Number	410-000-0000	Xyz-xyz-xx23	Error- must be numeric
15	Phone Number	410-000-0000	@@-##-####	Error – must be numeric
16	Email	ex@gmail.com	zqk@loyola.edu	Acceptable input for field
17	Email	ex@gmail.com	123@gmail.com	Acceptable input for field
18	Email	ex@gmail.com	noyes@gmail.com	Acceptable input for field
19	Email	ex@gmail.com	Zwe123.com	Error – must be valid email
20	Email	ex@gmail.com	123@zahara.com	Error – must be valid email
21	Class Year	####- 2019	2017	Accepted
22	Class Year	####- 2019	17	Error – must be year format
23	Class Year	####- 2019	Seven	Error – must be numeric
24	Class Year	####- 2019	@17	Error – must not contain symbols
25	Address Line 1	# Street	2	Error – must be street name
26	Address line 1	# street	2 avenue	Acceptable input for field
27	Address line 2	Letters/#	Apt 1	Acceptable input for field
28	Address line 2	Letters/#	Campus House	Acceptable input for field
29	City	Letters only	Baltimore	Acceptable input for field
30	City	Letters only	B12	Error – must be letters only
31	City	Letters only	@btm	Error – must be letters only
32	Zip code	5 numbers - 21234	2e34	Error- must be numbers only
33	Zip code	5 numbers	21210	Acceptable input for field
34	Zip code	5 numbers	Sefhkjsfh	Error – must be numbers only
35	Church Parish	Words Only	Chruch1	Error – Must be letters only
36	Church Parish	Words only	Cathedral	Acceptable input for field

Figure 5 Wedding Form Test Cases

3.1.1.4 Log- in Form Test Case and Expected Result

The log in form will be used by Campus Ministry Staff and interns to gain access to the web forms to run queries and interact with the database. The login form restricts access of other users and only campus ministry staff will be able to access this page. Below we have listed the test cases for this form.

1	Username	Alpha-numeric	Zqkazmi12	Acceptable input for field
2	Username	Alpha-numeric	12345	Error – must be alpha-numeric
3	Username	Alpha-numeric	Zqkazmi	Error – must be alpha-numeric
4	Password	Alpha-numeric (lovely23)	12345	Error – password cannot be all numbers
5	Password	Alpha-numeric	Lovely23	Accepted
6	Password	Alpha-numeric	Lovely	Error – must be alpha-numeric

Figure 6 Login Form Test Cases

One of the underlying assumption for the above test cases is that the accepted username and password must match and be allowed by the administrator.

3.1.1.5 Chapel Choir Test Case and Expected Result

The chapel choir form will be used by the choir director to keep track of all members of choir and to have members provide their information via the form. Below we have included all the fields in this form which will require testing and other fields are drop down selections.

Test Case	Field/Information	Input Accepted	Input Given	Result
1	Name – first and last	Zahara Kazmi	1234456	Error – name must be a string
2	Name – first and last	Zahara Kazmi	23zahara23	Error – must only have letters
3	Name – first and last	Zahara Kazmi	Bobby Jones	Acceptable input for field
4	Name – first and last	Zahara Kazmi	@xyzml	Error – must contain letters only
5	Class Year	####- 2019	2017	Acceptable input for field
6	Class Year	####- 2019	17	Error – must be year format
7	Class Year	####- 2019	Seven	Error – must be numeric
8	Class Year	####- 2019	@17	Error – must not contain symbols
9	Phone Number	410-000-0000	410-216-2222	Acceptable input for field
10	Phone Number	410-000-0000	Xyz-xyz-xx23	Error- must be numeric
11	Phone Number	410-000-0000	@@-##-####	Error – must be numeric
12	Email	ex@gmail.com	zqk@loyola.edu	Acceptable input for field
13	Email	ex@gmail.com	123@gmail.com	Acceptable input for field
14	Email	ex@gmail.com	noyes@gmail.com	Acceptable input for field

15	Email	ex@gmail.com	Zwe123.com	Error – must be valid email
16	Email	ex@gmail.com	123@zahara.com	Error – must be valid email
17	Birth date	08/21/1995	08/21/1998	Acceptable input for field `
18	Birth date	08/21/1995	04/11/2017	Error – must be at least 18
19	Birth date	08/21/1995	04/05/2017	Error – must be at least 18
20	Address Line 1	# Street	2	Error – must be street name
21	Address line 1	# street	2 avenue	Acceptable input for field
22	Address line 2	Letters/#	Apt 1	Acceptable input for field
23	Address line 2	Letters/#	Campus House	Acceptable input for field
24	City	Letters only	Baltimore	Acceptable input for field
25	City	Letters only	B12	Error – must be letters only
26	City	Letters only	@btm	Error – must be letters only
27	Zip code	5 numbers - 21234	2e34	Error- must be numbers only
28	Zip code	5 numbers	21210	Acceptable input for field
29	Zip code	5 numbers	Sefhkjsfh	Error – must be numbers only
30	Folder no	2 digits	456	Error – must be two digit number
31	Folder no	2 digits	Lik	Error – must be two digit number
32	Folder no	2 digit	02	Acceptable input for field

Figure 7 Chapel Choir Test Cases

3.1.1.6 Program Form Test Case and Expected Result

The program form will be used by intern and staff to initiate a new event to get list of attendees. This data will be used in machine learning algorithm such as date, time and location. Since this information is carried over for the AI as well it is very important to properly test all the data in this form. The test cases are only for fields where users will be inputting information and no test cases are being done for selection options.

Test Case	Field Name	Input Accepted	Given	Result
1	Event Name	Speed Faithing	12345	Error – name must be a string
2	Event Name	Speed Faithing	Speed2	Error – must only have letters

3	Event Name	Speed Faithing	Mass of Holy Spirit	Acceptable input for field
4	Event Name	Speed Faithing	@mAss	Error – must contain letters only
5	Semester	Letter and two-digit year – FA17	Fall	Error – Must semester year format
6	Semester	Letter and two-digit year – FA17	2017	Error – Must semester year format
7	Semester	Letter and two-digit year – FA17	SP17	Acceptable input for field

Figure 8 Program Form Test Cases

3.1.1.7 Interest Form Test Case and Expected Result

The interest form will work as sort of contact form for campus ministry. This form will be used when anyone wants to get in contact with campus ministry. They would have to pick reason they want to contact campus ministry and send their information. Below we have listed all the test cases for the requested information for interest form.

1	Name – first and last	Zahara Kazmi	1234456	Error – name must be a string
2	Name – first and last	Zahara Kazmi	23zahara23	Error – must only have letters
3	Name – first and last	Zahara Kazmi	Bobby Jones	No Error
4	Name – first and last	Zahara Kazmi	@xyzml	Error – must contain letters only
5	Phone Number	410-000-0000	410-216-2222	Accepted
6	Phone Number	410-000-0000	Xyz-xyz-xx23	Error- must be numeric
7	Phone Number	410-000-0000	@@-##-####	Error – must be numeric
8	Email	ex@gmail.com	zqk@loyola.edu	Acceptable input for field
9	Email	ex@gmail.com	123@gmail.com	Acceptable input for field
10	Email	ex@gmail.com	noyes@gmail.com	Acceptable input for field
11	Email	ex@gmail.com	Zwe123.com	Error – must be valid email
12	Email	ex@gmail.com	123@zahara.com	Error – must be valid email

Figure 9 interest Form Test Cases

3.1.1.8 Search Query Test Case and Expected Result

This search form will be used for Campus Ministry staff to run queries on the database and although users will input simple data this needs to be converted into SQL format. The way this form has been formatted is user will get a chance to select from which table they are looking and then

they would need to enter specific data. An example of something on this form might is also explained is as follows:
Search (Drop Down Selection from table) where (field selection) equals (user input)
Below the test cases for this form have been listed.

Test Case #	Field Name	Input Accepted	Given	Result
1	Name – first or last	Zahara or Kazmi	1234456	Error – name must be a string
2	Name – first or last	Zahara or Kazmi	23zahara23	Error – must only have letters
3	Name – first or last	Zahara or Kazmi	Bobby Jones	Acceptable input for field
4	Name – first or last	Zahara or Kazmi	@xyzml	Error – must contain letters only
5	Event Name	Speed Faithing	12345	Error – name must be a string
6	Event Name	Speed Faithing	Speed2	Error – must only have letters
7	Event Name	Speed Faithing	Mass of Holy Spirit	Acceptable input for field
8	Event Name	Speed Faithing	@mAss	Error – must contain letters only
9	Class Year	####- 2019	2017	Acceptable input for field
10	Class Year	####- 2019	17	Error – must be year format
11	Class Year	####- 2019	Seven	Error – must be numeric
12	Class Year	####- 2019	@17	Error – must not contain symbols
13	Folder no	2 digits	456	Error – must be two digit number
14	Folder no	2 digits	Lik	Error – must be two digit number
15	Folder no	2 digit	02	Acceptable input for field

Figure 10 Search Query Test Cases

Sample Query: Select (Choir) where Name equals "Brandon".

3.1.2 Stubs and Drivers

3.1.2.1 Web Form Stubs and Drivers

The forms described above are guest form and this forms input is read from the webpage and stored in the database via PHP scripts. During testing, we cannot save the information in the database since it is not valid information. Thus, we need to make stubs and driver for our form. The stubs for these forms will be read in the data, make sure it is correct format and simply inform user that information has been accepted. These stubs will not save information to database but they will check for proper format for all the different fields. Since these guest forms are simply requesting information, they would not require drivers but only stubs.

3.1.2.2 PHP Script Drivers Stubs and Drivers

The PHP scripts read in the data from webpages and process the input to MySQL insertion statements. Since these scripts are reading in data from webpages we would need to create stubs and drivers for these scripts. The driver for these scripts will provide sample data to be processed for insertion into MySQL scripts. The driver would simply provide sample data to scripts and we would need to create drivers for each of our MySQL table. The stubs for PHP scripts will accept MySQL statement created by script to make sure it is in acceptable format. If it is accepted MySQL format for insertion statement, the stub would return data entered and if not accepted it will return an error for why it cannot be accepted. We would need to create stubs and drivers for all the different tables within the database because each table accepts different information thus their insertion statements will be different as well.

3.2 Integration testing

3.2.1 Test cases for Integration Testing and Expected Result

3.2.1.1 Web Form Test Case and Expected Result

Integration testing for the web forms must be very different from unit testing for this component. Each of the form was requesting different type of information from the user but the result for all the forms was submission. All the forms can be tested by checking if the submission goes through or not. Each form has a submit button at the end and if someone required information is not given or wrong format is given then the form will not submit. Since the integration test will be the same for all the forms, instead of showing it for each form, we have listed all the test cases below for the component altogether. Below we have listed test cases for Web from integration testing.

Test Case #	Field	Input Accepted	Given	Result
1	Submit button	All required information in form	Just one field – name	Error- Please enter required information before submitting
2	Submit button	All required information	All required information	Data saved in database
3	Submit button	All required information	All required +additional information	Data saved in database

Figure 11 Web Form Integration Test Case

3.2.1.2 Search Query Script Test Case and Expected Result

Integration testing for the Query form has to be to make sure the database returns something to the user. In unit testing, it is already being checked to make sure our input is in MySQL query format thus with integration we must check if it database is returning results to main screen. Below we have listed all the possible inputs and results for integration testing for query form.

Test Case #	Field	Input Accepted	Given	Error
1	Result Button	MySQL statement	MySQL Statement	Error – Too Specific – no information found
2	Result Button	MySQL statement	MySQL Statement	Result table printed

Figure 12 Search Query Integration Test Cases

3.3 High-Order Testing

3.3.1 Performance and Load/Stress Testing

To conduct performance testing for this software we will be using sample users as well as testing tools. In performance testing, the testing tools to be implemented will be Apache JMeter and Open STA. Both of these tools are used for performance testing and they will test different aspects of the software's performance. Apache JMeter will combine the HTML with database, allow us to run test on the whole test and how well it performs. Using Apache JMeter we can write sample test plans and then in GUI we can pick which plan to run on what type of load. JMeter will let us run heavy load on server, group of sever and specific form. This will test the strength of the forms and server and analyze the overall performance. Similarly, Open STA will also run HTTP stress testing along with Performance testing. Open STA will allow us to record in the browser and produce simple scripts which we can edit. The scripts then can be executed to simulate many users to generate high performance load on the server. Another tool which can be implemented is Radview which will allow us to easily write scripts and analyze our software. Radview will provide correlation, parameterization, response validation and load generation to test the strength of our software. Using these tools individually and a combination of them we should be able to test how well our software is performing and we would also have the subjective experience of the users to along with numeric performance.

3.3.2 Security Testing

Security is a major priority and component of our software thus conducting security testing is significantly important to the software's performance. In designing this software's security testing, we have considered several different principals. One of them is confidentiality and that not only means protecting information provided by users but also who has authorization to see it. Data integrity means that during data transfer nothing is being altered and we are keeping data as it is being provided. The focus is the authentication – users need to have access to be able to look at data and access it. There are several different ways the security can be compromised such as injection, broken authentication and session management, cross-site scripting and missing function level access control. Testing the security is very important and some of the tools we will use to test the security are FxCop and FindBugs. Both tools allow us to create rules for our software and they will try everything to get into our system and see if they are successful. The tools will give feedback on what works and what does not work, giving developer a chance to make changes to better protect the software.

4. Usability Testing

4.1 Testing Plan

The Usability testing of this software will be done by using a sample of users and the users learning to use the software. Some of measurements for usability testing to be considered includes skill level of user, time required to get used to software, measure of increase in user productivity and user's attitude. Due to the time constraint of this project, the only parameter we could have an issue testing is productivity since that must be a measure over certain period.

4.2 Sample Users

Usability testing can only be effective if we have a range of novice and expert users. Since this software will be used authenticated users who have access to the data and by users who will simply be entering information – we have to test the usability for both of those users. The authenticated user sample will be campus ministry staff and interns. The users who will be using this software will get to test and see if it works for them. The other set of users will be picked at random from campus ministry events and will be asked to fill out the form.

4.3 Testing

In order to conduct the testing, users will fill out a survey which will ask the about their skill level of computers, the time it took them to learn the software and how well they liked the software. The survey results will allow us to measure how well our software performs in terms of usability.

4.4 Analysis

Using the feedback from the testing and survey, we can evaluate the usability of the software. The feedback will also allow us to measure what components in software are complex and are not easy to understand. Knowing about the flaws in our design will provide us with an opportunity to make future changes. The analysis of usability testing should meet the criteria below to be declared usable:

1. A sample user who is novice should be able to get used to the software in about an hour and they should find the software to be easy to enter information into.
2. A sample user who is an expert should be able to get used to the software in less than 30 minutes and they should find the software easy to find and enter information.

If the software usability testing passes the above criteria, it is decla

5. Performance Testing

5.1 Testing Plan

The performance testing for this software will include running load and stress testing as well as web load. Much of the performance of this software will depend on the type of server we use and its connectivity. For performance testing we will not be using sample user but instead we will implement automated tools which will test the software via provided scripts.

5.2 Tools

The tools we will be using for testing the performance of our software include JMeter, Open STA and Radview. These tools will allow us to test the speed, reliability and accuracy of our software. These tools will test the strength of our software by running different types of load and using three different tools gives a good comparison to get an average of how well our software works.

5.3 Testing

The testing will be automated and scripts will be written as specified by the tools the test the software. The different loads we need to test include heavy load, medium and low load on server. We will also run multi-threading testing using these tools and recorded tests will provide us with the analysis for our software. Since the testing is automated, we can run as many test as necessary and analyze the results to evaluate the limits for the software.

5.4 Analysis

The testing results must adhere to criteria listed below to acceptable as a well performing software.

- The server response time does not degrade to a noticeable point and at all times it has a relatively quick response time.
- The system degradation does not impact the security of the system.
- The reliability and accuracy of the system should not decrease as load increases.

6. Testing Specific to Web Application

6.1 Content Testing

The content testing for this software will be done using Markup validation service. This validation service will be executed for all the HTML pages. The markup will provide all different types of error with in the page and what is expected and what is returned. The service does not provide any sort of score but the we can measure how well formed the webpages are by counting the number of errors. The webpages will be considered valid and complete if the number of error is less than 10. If the amount of error is greater than 10, the webpage should be changed to fix the error to achieve completeness.

6.2 Database Testing

The database queries and database itself is being tested separately, explained under unit testing and integration testing. In order to test the connectivity of the software, the server will continuously check the database server to make sure connectivity is possible.

6.3 Compatibility Testing

This software is developed as web application available on any browser. This software is being designed to be used and accessible on any device regardless of OS, Browser or any other constraints as long the device is connected to the internet. The webpages might appear differently on different bowsers and device depending on screen size but their overall design is to fit any screen. This software will be considered to have achieved compatible testing if it works the same on different platforms. The users who will be tested the software will conduct the compatibility testing as well since they will be running it on their own devices.

6.4 Navigation Testing

The system will be hosted on an online commercial server but due to the separation in user type we will have different navigation for the users.

6.4.1.1 Data Entry Form Navigation Testing

The data entry forms or “guest” forms will be separate individual forms which will be provided to users. These forms will be shared with users via links in emails and provided as redirection through the Loyola website. The user should not need to search these forms anywhere but should be provided to users.

6.4.1.2 Authenticated Site Navigation Testing

The authenticated site will be private link only accessible by staff and users. The website will be linked on the current campus ministry website so it can be accessed by all staff members. The staff should have previously been registered on the system by the administrator.

6.5 Security Testing

The security testing technique for the authenticated site as well the administration side will be done by using the techniques mentioned in section 3.3 under security testing.

7. Appendices

7.1 Tools Link/References

This section includes the names and the links for all the variance tools mentioned in this document which will be implemented to test this software.
Apache JMeter - http://jmeter.apache.org/
Open STA - http://opensta.org/
Radview - http://www.radview.com/
FxCop - https://www.owasp.org/index.php/FxCop
FindBugs - http://findbugs.sourceforge.net/
Selenium - http://www.seleniumhq.org/
Query Surge - http://www.querysurge.com/
Tap - https://theory.github.io/mytap/
Debugger - http://mydebugger.com/
SimpleTest - http://www.simpletest.org/