AngSalitaNgDiyos.com

Liturgical Site

Software Quality Assurance Plan

In Partial Fulfillment of the Requirements
in Software Quality Assurance

By:
Dimapilis, Joshua C.
Elizondo, Kimberly Mae B.
Urquiza, Trixia Marie A.

IT - 111

February 2015

I. Software Project Management Plan

This Software Quality Assurance Plan document, was developed specifically for the project *AngSalitaNgDiyos.com Liturgical Site.* During the project's *Planning stage,* the proponents arrived with the following description of the ventures at hand. The purpose of this representation is to *accurately reflect* the project's *scope* and *structure.* The following will assess the planned activities to be fulfilled throughout the cycles of the project.

A. Project Structure

The following sections of the *Project Management Plan* will be discussed in length in the succeeding segments:

- Vision and Scope
- Feasibility and Risk Analysis
- Management Approach
- Technical Approach

In the event of *end user reviews*, these first two chapters will initially be recommended for the reviewer's benefit.

B. Project Content

a. Vision and Scope

The succeeding sections describes the conditions driving the development of the calendar tool. This section introduces the application as intended and describes the scope and limitations of the development effort.

i. Vision

In the advent of the technology age, more and more industries are beginning to grasp the benefits of having an *online presence.* This is why even the *religious* sector of our country aims to enhance their capability of *catering* the needs of their intended spiritual audience, online.

AngSalitaNgDiyos.com, is an online Catholic Liturgical site that accommodates the online audience and their needs, spiritually and information - wise.

In line with this, the proponents of this project have tasked the IT – 111 students to create a calendar structure that can display the *Lectionary Cycles* and *Liturgical Feasts* within a certain year. It is known that every year, the *liturgical* calendar changes in line with certain technicalities defined by Lectionary Cycles (3 cycles for Sunday Lectures, 2 cycles for Weekday Lectures). The proposed calendar structure will identify specifically when these dates will be and arrange them accordingly.

The primary goal of the *Liturgical Site*'s calendar structure is to provide an informative, detailed and accurate representation of all the lectionary cycles and religious feasts within a specific year. This will designate the said dates and represent them in an online version of the Calendar.

ii. Scope

The project aims to provide a tool that helps online users to identify the designated dates of *religious lections and feasts* within a specified year.

- Create a database that includes all the *religious events* (e.g. lections, feasts and celebrations) within the year
- Create a tool that utilizes the created database to insert the specified *events* in line with specific requirements that are specified by the Catholic church
- Provide a calendar (that uses the database and the tool's synergy) to display links to the necessary *pages* and/or *audio* files, representing lections and mass readings for each date
- Encourage the online community to visit the *AngSalitaNgDiyos.com* site for informative and spiritual purposes

The functions of the system include the following:

- The main target market of this tool will mainly be defined as *users*. These users are those who access the calendar via the site, *AngSalitaNgDiyos.com*
- The calendar tool will automatically display the dates within the month that it is being accessed
- The calendar will display 6 types of *events*: (1) **Sunday readings**, (2) **Weekday readings**, (3) **Moveable feasts**, (4) **Solemnities**, (5) **Memorials** and (6) **Special Feasts**
- Each type of item has its own definite *business requirement*
    - **Sunday readings** have 3 yearly cycles: Year A, Year B and Year C. Each cycle has a designated set of Sunday readings which are variably different from each cycle. Year determinant is the remainder of the sum of all the

digits within the year, divided by 3. The succeeding table illustrates the determinant.

Remainder	Year
0	Year A
1	Year B
2	Year C

Table 1.1 **Determinant of year cycle for Sunday readings**

Example -For the Year 2013:
Sum of all digits in the year
= 2+0+1+3= 6
Remainder of sum divided by 3
= 6 / 3 = 2 remainder 0
Remainder is 0. Therefore 2013 is considered within the Year A cycle.

- For **Weekday readings,** there are two sets of readings. Year 1 &Year 2. The succeeding table describes the determinant.

Year Type	Year
Odd	Year 1
Even	Year 2

Table 1.2 **Determinant of year cycle for Weekday readings**

Example:
Year 2013 is an odd year. Therefore, it is within the Year 1 cycle.

- In the case of **Movable feasts,** the dates are set based on other movable feasts / solemnities within the year. (Some examples include *Easter Sunday, Palm Sunday* and *Pentecost Sunday*)
- For **Solemnities** and **Memorials,** they have their own set of readings, which replace Sunday and/or weekday readings excepts for specific Sundays (i.e. Sundays in Advent, Lent and weekdays of Holy Week / Easter Octave).Solemnities and memorials are big – time feasts. (e.g. *Presentation of the Lord,* and *Annunciation of the Lord*)
- For **Special Feasts,** these dates are static and are not day – sensitive.

- Event items such as *Movable feasts, Solemnities* and *Memorials* will be displayed as text, which will serve as markers for the specific events they represent

- These items that will be displayed in the calendar, will be in the form of *links* that will redirect to specific *pages* / *audio files*

The tool is envisioned to possess the following versions of the application framework:

- Web – version
- Mobile – version

The timeframe for the system development process is ten weeks; for the finished product, the scheduled system evaluation is March 31, 2015.

b. Feasibility and Risk Analysis

i. Feasibility Standards

This section addresses the issues of *application complexity* as well as the anticipated *risks* in *schedule* and *operation* procedures. The following factors have been defined in order to verify the project's feasibility. The table below illustrates these points and the team's proposition for the development and quality assurance stages of the project.

Standard	Risks / Issues	Proposition
Application Complexity	• The various requirements included in the calendar warrants well – defined conditions and proper synergy between the database and the tool. • The integration of the tool to the site itself should also be considered of vital importance	• Proper standards in code development must first be defined, upon establishment of these defined standards, following it is key • The usage of the tools provided by our *adviser*, and a proper understanding of the MVC Framework is essential • Development procedures must be in line with the professional opinions of our *mentors* and *adviser* • For System Integration, it must be done in line with the system
Schedule Constraints	• The allotted timeframe for the project development and quality assurance	• Development – wise, the tool's functions must be assessed if the 10–week development process can accommodate the necessary

	phase is a maximum of 10 weeks.	changes and coding efforts by the team • In the Quality Assurance phase, the tool should be assessed by certain metrics and success factors to be defined in the latter part of this section • A schedule for both *SOFTDEV* and *QUALITY* has been given, and for the success of the development and testing phases, the team's processes must be in line with this defined schedule
Client, User and Organizational Risks	• The risk of obtaining an error in logic in one year will mean an inheritance of the succeeding years' errors • The risk of having to engage in maintenance and support for the application, after development	• Proper development of the product must be implemented. The client must verify the correctness of the data in the database of the proposed system, before the implementation phase • Plans for maintenance and support must be in place, even after implementation. This plan must be proposed to the client, along with the system
Operational Feasibility	• Risk of web - hosting and storing massive amounts of data in the database • Risk of user apathy and unresponsiveness from the target market	• Hosting the site, with large amounts of data may be mitigated by specifying early on with the client, the expected and the actual size of the data to be stored and used for the project • Through proper dissemination in local churches, and on the web – users may be properly oriented with the goal of the tool and the site itself

Table 1.3 **Feasibility Standards**

ii. Quality Metrics

The project may be deemed as successful if the following short – term metrics are satisfied:

- Continuity of development phase – consistent delivery of individually required features during each iteration / cycles, by meeting the scheduled evaluation of March 31, 2015
- Explicit affirmation from client – confirmation of met requirements / expectations by the users
- Convenience, Speed and Reliability – if the proposed tool was able to affect the users positively; if the users find the tool informative, and if the users feedback about the tool's response includes it being *rapid*, *reliable* and *accurate*

iii. Success Factors

The tool's implementation can be considered successful if it meets the following criteria:

- Assurance that the system conforms to the mentioned business requirements and client standards; receiving a passing rate in the Quality Assurance Testing Phase
- Successful integration to the implemented site, which must yield improvement in the user experience
- A positive response by the spiritual audience online; either by their valued response or by their patronization of the system / high utilization

c. Management Approach

i. Development

The development process to be used is the *Agile Methodology.* There will be intensive development, and succeeding iterations (0,1, and 2). Each iteration involves functional integration and detailed change requests, adjustments and tracking – in accordance to client decisions. Bugs and issues may be found during each iteration, when quality assurance testing is done, fixing these bugs will be prioritized after each iteration.

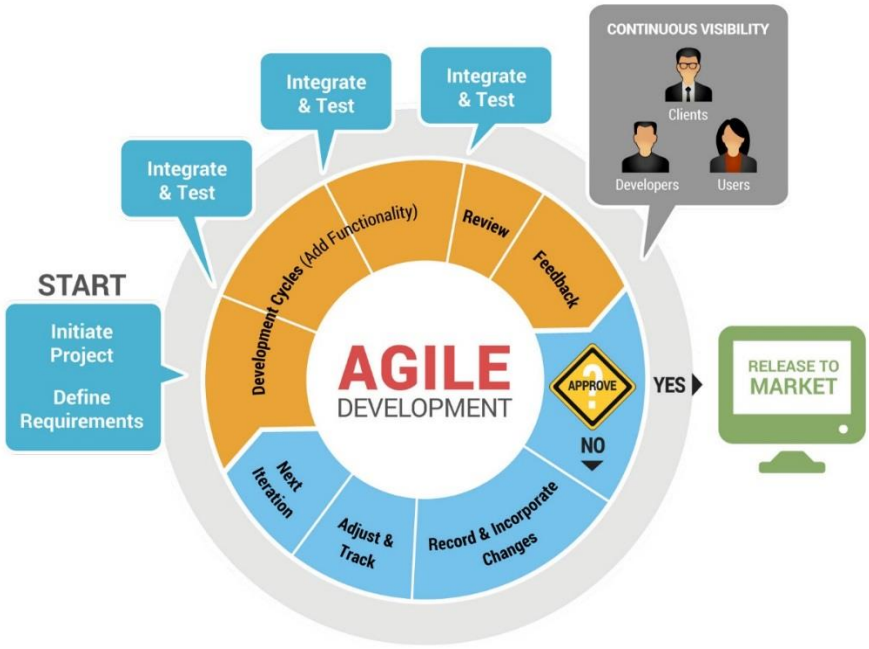


Image 1.1 **Agile Methodology, source: [Code2u.net]**

ii. Communication

Meetings will be done at least thrice a week. Updates, for daily scrum and weekly status reports are located in the repository. For each iteration, the team will consider each sets of 3 weeks as a sprint. There will be 3 sprints in this term. The sprints will be aligned with each iteration that has been scheduled by the team's *SOFTDEV* adviser.

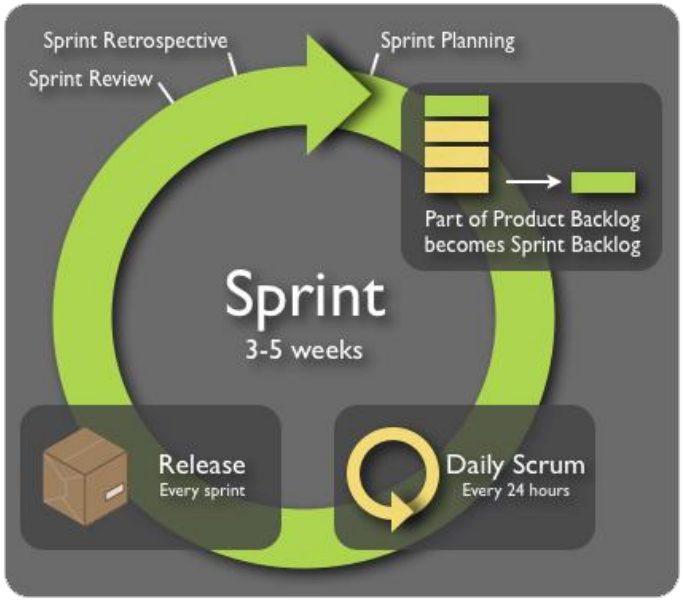


Image 1.2 **Sprint, source: [Kaeru.se]**

iii. Quality Assurance

From the above-mentioned methodology, the Quality Assurance Testing Phase will be following the same process of *Developing, Base-lining, and Testing*.

The processes of which were mentioned above happens each iteration / sprint. The steps inside each phase, involve the following:

- Research and Analysis
- Prototyping and Designing
- Testing and Planning

Various static tests and dynamic tests have been duly scheduled in line with the iterations. Naturally, the static testing schedules have been given priority over the succeeding dynamic tests.

The following image illustrates the proponents' intended QAT Phase of the Project (which involves Testing, Quality Control and Quality Assurance):

Image 1.3 **Testing, Quality Control and Quality Assurance, source: SystemsAppsControls.com**

iv. Roles and Responsibilities

The following responsibilities have been designated for each team member to ensure the project's success. The table below illustrates the individual roles of each member.

Name	Roles	Responsibilities
Joshua C. Dimapilis	Project Manager / Developer	• Overseeing of the Project Status and Progression • Management and leadership of the Project team • Planning and Evaluation of Development and Quality Assurance
Kimberly Mae B. Elizondo	Quality Assurance Tester / Developer	• Quality Assurance Consulting • Monitoring of schedule, iterations and sprints • Business Requirements Analysis
Trixia Marie A. Urquiza	Quality Assurance Tester / Developer	• Database Design and Management Consulting • Approval of Change Requests and adjustments • Management of documentation and scrum

Table 1.4 **Teams' Roles and Responsibilities**

d. Technical Approach

i. Technologies for Development

The following tools are to be used for development:

Generic Tools	Specific Tools
Programming Languages	PHP, HTML5, CSS3
Database Server	MySQL
Web Server	Apache Server
Framework, Extensions	Yii 2.0, fullcalendar.io

Coding Tools	Yii PHP Framework, Sublime Text, fullcalendar.io
Mobile Phone Testing	Devices running JellyBean or newer versions of Android
Documentation Tools	Microsoft Office, Microsoft PowerPoint, MySQL Workbench
Repository	*code.google*

Table 1.5 **Tools Used for the Project**

ii. Use Cases

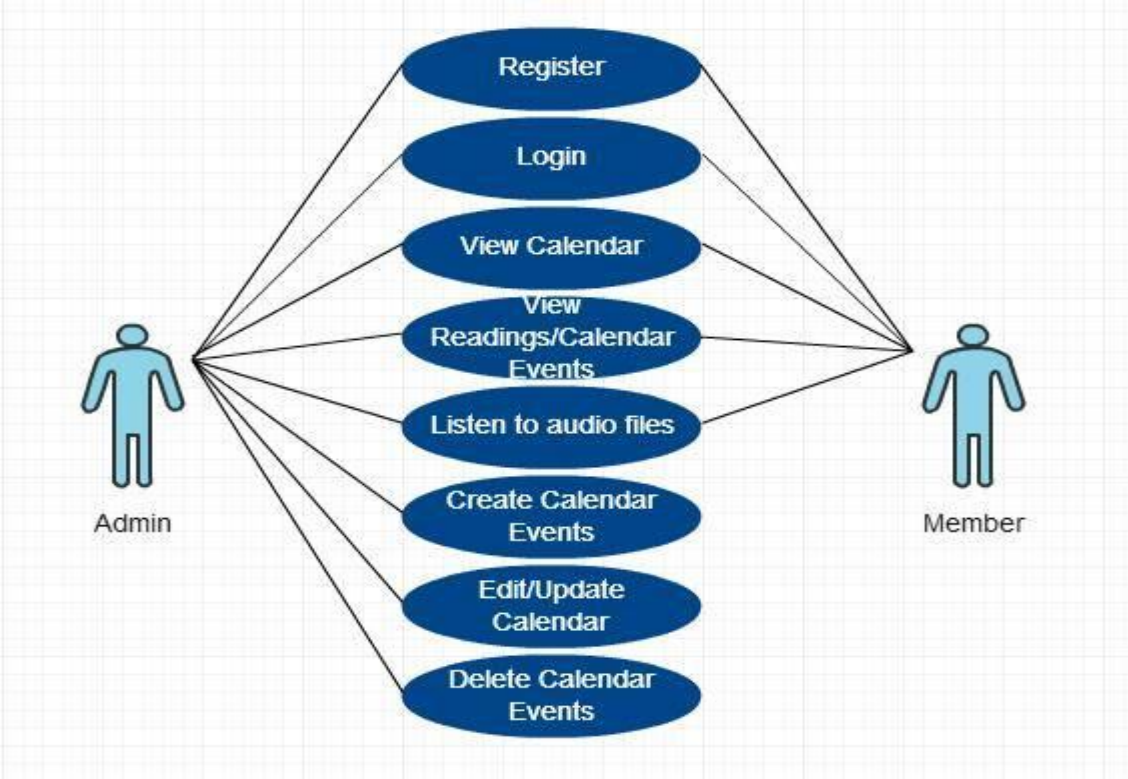


Image 1.4 **Use Case Diagram**

e. Glossary of Terms

Below is a glossary of project – specific terms:

- Calendar
- Events
- Lectionary Cycles
- Memorials

- Movable Liturgical Feasts
- Reading Set
- Religious Events
- Solemnities
- Special Feasts
- Sunday readings
- Weekday readings

II. Requirements Document

The Requirements class of deliverables are produced during the Requirements stage and updated if necessary during the Design, Development, and Integration & Test stages. The purpose of the Requirements class is to accurately define the scope, structure, and high-level functionality of the database application under design.

A. Requirements Structure

The Requirements class of deliverables is composed of three related documents:

• The Logical Database Description

• The Requirements Document

• The Requirements Traceability Matrix

B. Requirements Content

a. Logical Database Description (LDD)

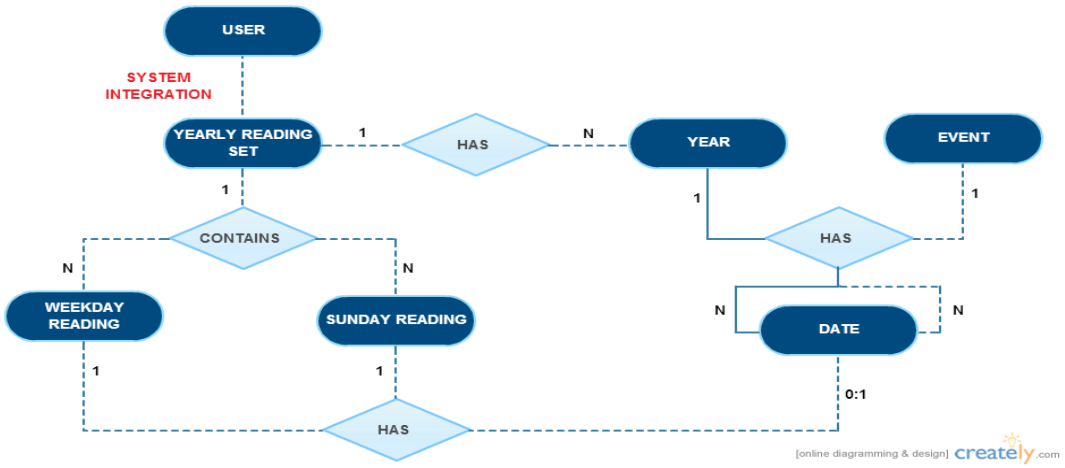


Image 1.5 **Logical Database Description**

Entity: USER
Description:

A **user** is a person who uses the system and access the calendar.

Relationship:

A user with an admin role can create, read, update and delete calendar events.
A user with a member role can view calendar events.

Actor Interaction:

ROLE	ACTOR
Create	Admin
Read	Admin, Member
Update	Admin
Delete	Admin

Entity: YEAR
Description:

A **year** is an entity that can hold zero or multiple events.

Relationship:

A year contains multiple dates, events, a Sunday reading cycle and a weekday reading cycle.

Actor Interaction:

ROLE	ACTOR
Create	Admin
Read	Admin, Member
Update	Admin
Delete	Admin

Entity: DATE
Description:

A **date** is defined by a month and a numerical date, and is an entity that can hold zero or multiple events.

Relationship:

A day can have zero or multiple events.

Actor Interaction:

ROLE	ACTOR
Create	Admin
Read	Admin, Member
Update	Admin
Delete	Admin

Entity: EVENT

Description:

An **event** is an occurrence that warrants a presence in the calendar. Events are relatively of great importance, and are based on the business requirements provided by the client. The event table has four *lookup* tables: Movable Feasts, Special Feasts, Memorials and Solemnities.

Relationship:

An event can either be classified as a movable feast, special feast, memorial, or solemnity.

Several or zero events can occur in a single day.

Actor Interaction:

ROLE	ACTOR
Create	Admin
Read	Admin, Member
Update	Admin
Delete	Admin

Entity: YEARLY READING SET

Description:

A **yearly reading set** is a compilation of readings that are separated by cycle types and numbers. (i.e. For *Sunday Readings*: Cycle A, B and C, for *Weekday Readings*: Cycle 1 and 2).

Relationship:

A yearly reading set contains many weekly readings.

A yearly reading set contains many Sunday readings.

Actor Interaction:

ROLE	ACTOR
Create	Admin
Read	Admin, Member
Update	Admin
Delete	Admin

Entity: WEEKDAY READINGS
Description:

A **weekday reading** refers to a reading that is used by lecturers and priests during weekdays.

Relationship:

Each weekday should have one weekday reading.

Actor Interaction:

ROLE	ACTOR
Create	Admin
Read	Admin, Member
Update	Admin
Delete	Admin

Entity: SUNDAY READINGS
Description:

A **Sunday reading** refers to a reading that is only used during Sundays.

Relationship:

Each Sunday should have one Sunday reading.

Actor Interaction:

ROLE	ACTOR
Create	Admin
Read	Admin, Member
Update	Admin
Delete	Admin

b. Software Requirements Document (SRD)

The following information provides an overview of the system requirements gathered through a collaboration with the project's client.

- **System Requirements**

The system must:

1. Provide a user-friendly interface for easier navigation
2. Provide a login page so that ***registered users*** can access the site
3. Provide a sign up page to allow unregistered users to register and access the site
4. Contain a calendar filled with generated readings and audio files
5. Allow ***registered users*** (***admin or member role***) to view the calendar and calendar events
6. Not allow ***unregistered users*** to view the calendar and calendar events

7. Filter the calendar if specific year is chosen by the user (***admin or member roles***)
8. Filter the calendar if specific month is chosen by the user (***admin or member roles***)
9. Allow ***registered users*** (***admin or member roles***) to listen to audio files included in the calendar
10. Allow ***users with admin role*** to add calendar events
11. Allow ***users with admin role*** to update or modify calendar events
12. Allow ***users with admin role*** to delete calendar events
13. Update the calendar display if ***CRUD function*** is performed by the user
14. Update the system database if ***CRUD function*** is performed by the user

- **User Requirements**

  A registered user (***admin or member role***):

  1. Must have a username and a password to access the website
  2. Must access the website using the login page provided by the system
  3. Can view the calendar in a ***monthly basis***
  4. Can navigate through the different months/years of the calendar
  5. Can choose a specific year to navigate
  6. Can choose a specific month to navigate
  7. Can perform ***create*** function in the calendar (***users with admin role***)
  8. Can read the contents of the calendar
  9. Can perform ***update*** function in the calendar (***users with admin role***)
  10. Can perform ***delete*** function in the calendar (***users with admin role***)
  11. Can click audio files provided in the calendar
  12. Can click links
  13. Can click buttons or tabs within the page
  14. Can play audio files provided in the calendar
  15. Can listen to audio files provided in the calendar

c. Requirements Traceability Matrix (RTM)

The following Requirements Traceability Matrix is derived from the software requirements document indicated above.

Requirements Traceability Matrix	
Project Name:	***AngSalitaNgDiyos.com Liturgical Site***
Project Type:	***Website***
Project Start Date:	***12-Jan-15***
Project End Date:	***12-Apr-15***
Project Adviser/s:	***Mr. Joe Gene Quesada (Software Development)***
	Mr. Allan B. Cotecson (Quality Assurance)
	Mr. Ernesto Boydon (Software Engineering)
	Ms. Ma. Theresa Montemayor (Database Design and Management)
Project Manager:	***Joshua C. Dimapilis***

Requirement ID	*Requirement Description*	*Requirement Type*	*Status*	*Priority*	*Software Module*	*Test Case Number*	*Tested In*	*Implemented In*	*Verification*
A. SYSTEM REQUIREMENTS									
001	Provides a user-friendly interface for easier navigation	Highly Needed	In Progress	High	System Interface				
002	Provides a login page so that ***registered users*** can access the site	Highly Needed	In Progress	High	Login				
003	Provides a sign up page to allow unregistered users to register and access the site	Highly Needed	In Progress	High	Login				
004	Contains a calendar filled with generated readings and audio files	Highly Needed	In Progress	High	Calendar				
005	Allows ***registered users*** (***admin or member roles***) to view the calendar and calendar events	Highly Needed	In Progress	High	Calendar				

Image 1.6.1 **Requirements Traceability Matrix Part 1**

006	Does not allow ***unregistered users*** to view the calendar and calendar events	Highly Needed	In Progress	High	Calendar				
007	Filters the calendar if specific year is chosen by the user (***admin or member roles***)	Highly Needed	In Progress	High	Calendar				
008	Filters the calendar if specific month is chosen by the user (***admin or member roles***)	Highly Needed	In Progress	High	Calendar				
009	Allows ***registered users*** (***admin or member roles***) to listen to audio files included in the calendar	Highly Needed	In Progress	High	Calendar				
010	Allows ***users with admin role*** to add calendar events	Needed		Medium	Event				
011	Allows ***users with admin role*** to update or modify calendar events	Needed		Medium	Event				
012	Allows ***users with admin role*** to delete calendar events	Needed		Medium	Event				
013	Updates the calendar display if ***CRUD function*** is performed by the user	Needed		Medium	Event				
014	Updates the system database if ***CRUD function*** is performed by the user	Needed		Medium	Calendar				

Image 1.6.2 **Requirements Traceability Matrix Part 2**

B. USER REQUIREMENTS									
015	Registered user must have a username and a password to access the website	Highly Needed	In Progress	HIgh	Login				
016	Registered user must access the website using the login page	Highly Needed	In Progress	High	Login				
017	Registered user can view the calendar in a ***monthly basis***	Highly Needed	In Progress	High	Calendar				
018	Registered user can navigate through the different months/years of the calendar	Highly Needed	In Progress	High	Calendar				
019	Registered user can choose a specific year to navigate	Highly Needed	In Progress	High	Calendar				
020	Registered user can choose a specific month to navigate	Highly Needed	In Progress	High	Calendar				
021	Registered user can view Weekday Readings	Highly Needed	In Progress	High	Calendar				
022	Registered user can view Sunday Readings	Highly Needed	In Progress	High	Calendar				
023	For ***users with admin role***, he/she can perform ***create*** function in the calendar	Needed		Medium	Event				
024	For ***users with admin/member role***, he/she can ***read*** the contents of the calendar	Highly Needed	In Progress	High	Calendar				
025	For ***users with admin role***, he/she can perform ***update*** function in the calendar	Needed		Medium	Event				
026	For ***users with admin role***, he/she can perform ***delete*** function in the calendar	Needed		Medium	Calendar				

Image 1.6.3 **Requirements Traceability Matrix Part 3**

027	Registered user can click audio files provided in the calendar	Highly Needed	In Progress	High	Calendar				
028	Registered user can click links	Highly Needed	In Progress	High	Calendar				
029	Registered user can click buttons or tabs within the page	Highly Needed	In Progress	High	Calendar				
030	Registered user can play the audio files provided in the calendar	Highly Needed	In Progress	High	Calendar				
031	Registered user can listen to audio files provided in the calendar	Highly Needed	In Progress	High	Calendar				

Image 1.6.4 **Requirements Traceability Matrix Part 4**

III. Design Document

The Design class of deliverables are produced during the Design stage and updated if necessary during the Development and Integration & Test stages. The purpose of the Design class is to accurately define the scope, structure, and high-level functionality of the database application under design.

A. Design Structure

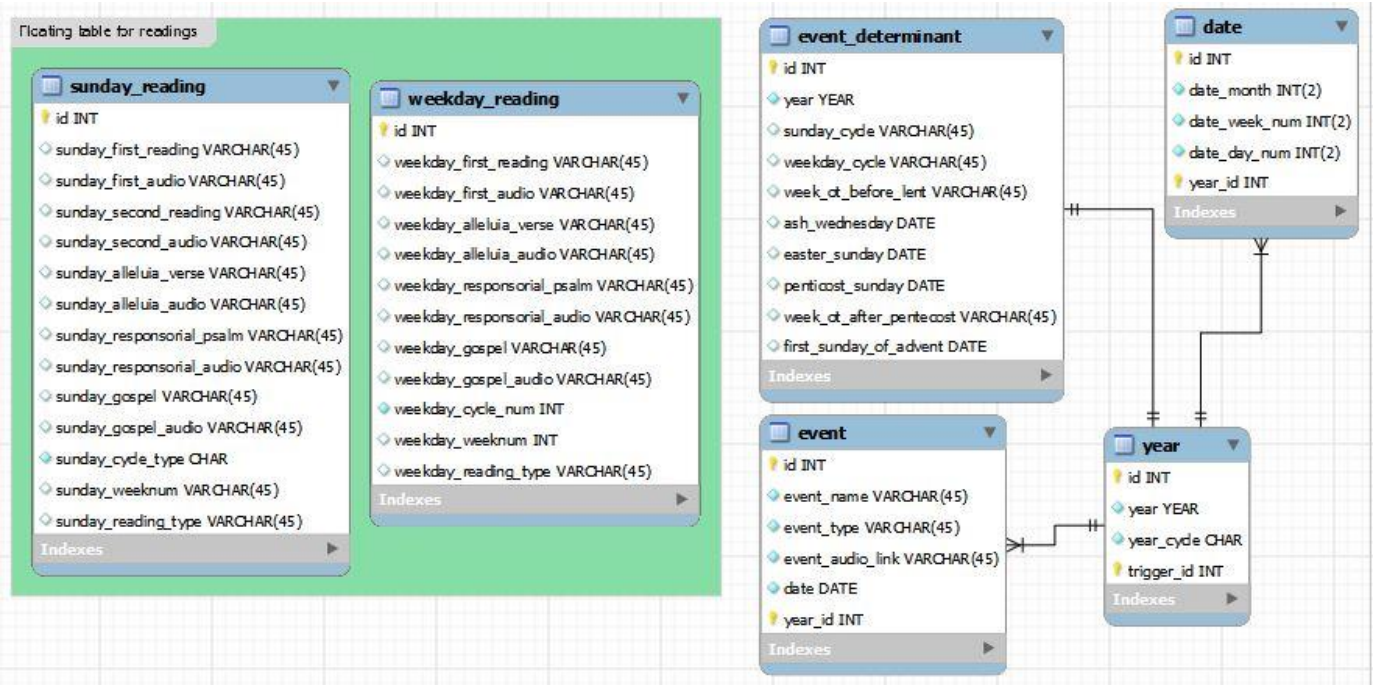
The Design class of deliverables is composed of three related documents:

- The Physical Database Description
- The Software Design Document
- The Requirements Traceability Report

B. Design Content

a. Physical Database Description (PDD)

Image 1.5 **Entity Relationship Diagram**

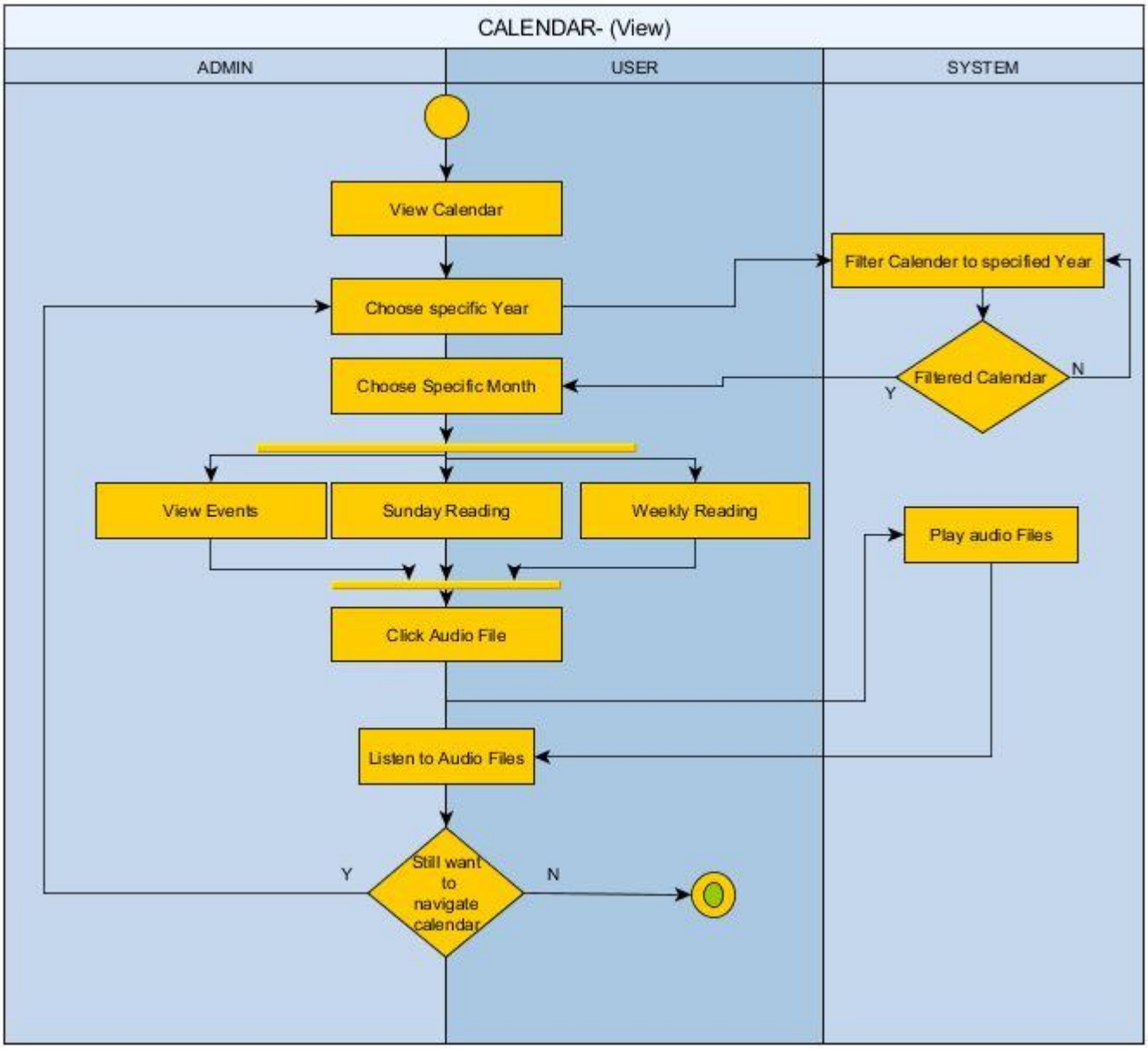


The physical database description defines the basic structure of the application at a conceptual level. The PDD focuses on providing a detailed description of the database structure to be implemented for the application.

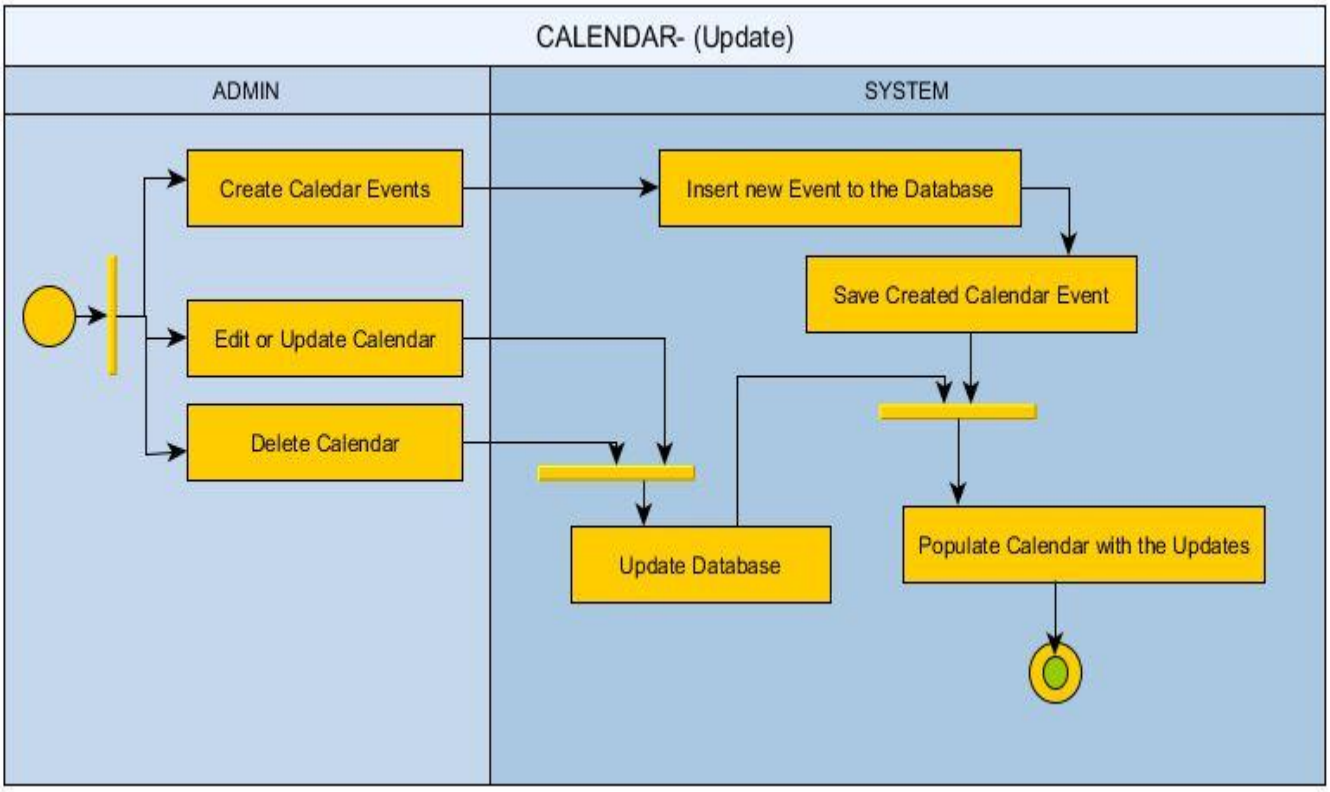
The PDD consists of an introduction, an Entity Relationship Diagram (ERD) and a series of table and field descriptions that define the relationships between the entities, field characteristics, and business rules.

The PDD is included by reference in the Design Document.

b. Software Design Document (SDD)



Image(s) 1.6 **Swim Lane Diagram**



c. Requirements Traceability Matrix (RTM)

The RTM makes use of the analysis listings in the SDD and its parent SRD. The purpose of the RTM is to show that each design element is related to a specific requirement in the SRD, that all goals in the project plan have at least one associated requirement, and that no requirements in the SRD are related to non-existent goals.

IV. Online Help

A. Help Structure

The structure of the Online Help section are composed of the following documents:

- Data Areas
- Optional Scenarios
- Frequently Asked Questions
- Data Dictionary

B. Data Areas

The following areas are to be considered as vital information in deriving business requirements for the application. Therefore, data from the succeeding sections are to be mapped in accordance to the following information:

a. **Lectionary Cycles**
    i. **Sunday Readings**
    ii. **Weekday Readings**
b. **Special Events**
    i. **Moveable Feasts**
    ii. **Memorials**
    iii. **Solemnities**
    iv. **Special Feasts**

C. Optional Scenarios and Frequently Asked Questions

The following are optional scenarios segregated according to the users involved in the Software Quality Assurance Plan. These scenarios are done in the form of *Frequently Asked Questions* for the benefit of comprehensibility. Responses to the said questions are immediately to be sufficed after the complete processing of the system:

a. **Tester**
    i. “How can I verify displayed items from the database?”
    ii. “How can I verify event details from an online resource?”
    iii. “How can I test the CRUD?”
b. **User**
    i. “How can I be able to view the calendar without an account?”

D. Data Dictionary

date

event_determinant

Column name	DataType	PK	NN	UQ	BIN	UN	ZF	AI	Default	Comment
id	INT	✓	✓					✓		
year	YEAR		✓							
sunday_cycle	VARCHAR(45)									
weekday_cycle	VARCHAR(45)									
week_ot_before_lent	VARCHAR(45)									
ash_wednesday	DATE									
easter_sunday	DATE									
penticost_sunday	DATE									
week_ot_after_pentecost	VARCHAR(45)									
first_sunday_of_advent	DATE									
event_audio_link	VARCHAR(45)		✓							
date	DATE		✓							
year_id	INT	✓	✓							

sunday_reading

Column name	DataType	PK	NN	UQ	BIN	UN	ZF	AI	Default	Comment
id	INT	✓	✓					✓		
sunday_first_reading	VARCHAR(45)									
sunday_first_audio	VARCHAR(45)									
sunday_second_reading	VARCHAR(45)									
sunday_second_audio	VARCHAR(45)									
sunday_alleluia_verse	VARCHAR(45)									
sunday_alleluia_audio	VARCHAR(45)									
sunday_responsorial_psalm	VARCHAR(45)									
sunday_responsorial_audio	VARCHAR(45)									
sunday_gospel	VARCHAR(45)									
sunday_gospel_audio	VARCHAR(45)									
sunday_cycle_type	CHAR		✓							
sunday_weeknum	VARCHAR(45)									
sunday_reading_type	VARCHAR(45)									

weekday_reading

Column name	DataType	PK	NN	UQ	BIN	UN	ZF	AI	Default	Comment
id	INT	✓	✓					✓		
weekday_first_reading	VARCHAR(45)									
weekday_first_audio	VARCHAR(45)									
weekday_alleluia_verse	VARCHAR(45)									
weekday_alleluia_audio	VARCHAR(45)									
weekday_responsorial_psalm	VARCHAR(45)									
weekday_responsorial_audio	VARCHAR(45)									
weekday_gospel	VARCHAR(45)									
weekday_gospel_audio	VARCHAR(45)									
weekday_cycle_num	INT		✓							
weekday_weeknum	INT									
weekday_reading_type	VARCHAR(45)									

year

Column name	DataType	PK	NN	UQ	BIN	UN	ZF	AI	Default	Comment
id	INT	✓	✓					✓		
year	YEAR		✓							
year_cycle	CHAR		✓							
trigger_id	INT	✓	✓							

V. Implementation Map

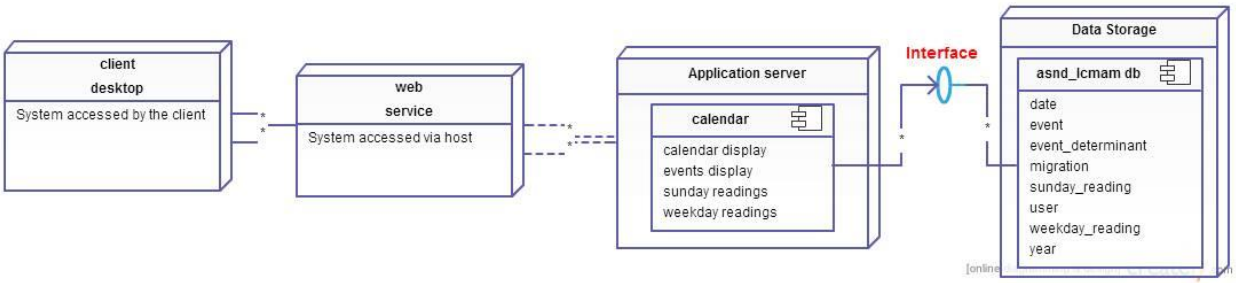


Image 1.7 **Implementation Map**

For the ***Requirements Traceability Matrix*** part of this section, please refer to ***Software Requirements Document*** section of

VI. Test Plan

This Software Test Plan was defined for the purpose of documenting the test procedures, test cases, and test steps required to validate the development effort.

A. Background & Introduction

Projects done using the Agile Method of development, are necessarily given a requirement of passing a certain level of confidence that the system is working at par with the industry's requirements. To be able to satisfy this requirement, testing must be done. Testing chosen specifically for this project involves various stages (i.e. Static, Dynamic Quality Assurance Testing, etc.) It is included in this project's success factors that the project's passing rate should be in line with the standards defined in this test plan.

Scheduled *Static* and *Dynamic Tests* have been given to the proponents of this project. Therefore, during the said intended schedules of testing, the team's prototype for the current iteration will be assessed by an external Quality Assurance Team using generically independent standards to rate the deemed project.

In conclusion of the each test phase, evaluation by the external Quality Assurance Team will be duly noted and assessed. For static test results, correction to the specified documentation and files will be implemented as soon as the reception of errors / comments. For dynamic test results, a record of issues and a log of bugs must be

specified to identify the specified change requests and fixes to be implemented by the team for the project. Succeeding Quality Assurance tests will be subject to the results of the previous static / dynamic test, in line with previous updates and corrections done by the team.

B. Assumptions

Before the basis of any assumption, the following dependencies are core concepts that may serve as the scaffolding of the project's test phase:

- The system must have sufficed the following basic deliverables, some of which, have been declared in the project scope:
    - Basic document deliverables (i.e. SQAP, Test Cases, etc.) to be evaluated **(Static testing)**
    - A database that includes all the *religious events* (e.g. lections, feasts and celebrations) within the year **(Static testing)**
    - Create a tool for value-inclusion in the database **(Dynamic testing)**
    - Provide a calendar that represents lections and mass readings for each date **(Dynamic testing)**
- Developer and designer involvement in the project have been clarified
    - The roles and responsibilities of each member of the team have been defined and clarified
- The Gathering Requirements phase has been accomplished successfully, and the current phase includes designing and development

In testing the system, the following assumptions are considered and are expected:

- System requirements have been set, and conditions dependent on the system have been provided by the proponents
    - The testing plan, test cases and other necessary requirements for testing to proceed have been provided by the proponents
- An external Quality Assurance Team has been selected to review the selected deliverables
- A certain standard has been set to properly evaluate the said deliverables; preferably these standards are set in a scale that can be quantified through values, and duly evaluated through description

Assuming that all the dependencies and assumptions are satisfied, specified and scheduled testing phases may proceed. It is important to note that these testing phases are specifically contingent to the schedules set for the project's evaluation.

C. Test Items / Programs

The proponents have prepared the following to properly establish the items that must be tested during the phases of Quality Assurance Testing:

a. **Documentation for Static Testing**
    i. Project Requirements Definition
    ii. Project Logical Design
    iii. Project Physical Design
    iv. Database Design
    v. User Interface
    vi. Software Program Logic / Code
    vii. Software Error Handling
    viii. Test Plan, Test Cases
    ix. Overall Quality of Documentation
    x. Documentation Completeness
b. **Calendar display**
    i. Calendar Interface
    ii. Features
c. **CRUD for data storage**
    i. Create capability
    ii. Read capability
    iii. Update capability
    iv. Delete capability

D. Features To Be Tested / Features Not To Be Tested

The features not to be tested are to be duly evaluated in line with the present prototype during dynamic testing. Therefore, the following features of the system must be *initally* tested and duly assessed:

a. **Security of User Registration and Login (Registration and Login)**
    i. User Site Access

ii. User Site Registration
iii. User Site Login

b. Correctness and legitimacy of event display (Calendar view)

i. Calendar Display
ii. Calendar Display Inspection
iii. Calendar Display Refresh
iv. Calendar Display Verification

c. Completeness of events being displayed (Reading view)

i. Events Display
ii. Calendar and Events Display Inspection
iii. Events Display Refresh
iv. Events Display Verification

d. Readability of calendar events (Event view)

i. Calendar Display
ii. Event Display
iii. Event Readability

e. Usability of calendar links (Link Redirection)

i. Link Display
ii. Link Redirection
iii. Verification of Link Display and Redirection Item

f. Dynamism of Administrator and User Privileges (CRUD)

i. Available Administrator Features
ii. Available User Features
iii. Create Items
iv. Read Items
v. Update Items
vi. Delete Items

The mentioned features are given explanations and certain margins of validity and fault in the Pass/Fail Criteria.

E. Approach

The agile approach for development warrants a brief definition of the Quality Assurance Testing Phases of this project. For the currently defined phases, the following table illustrates the necessary information for the approach in each QAT phase:

Name	Approach	Methodologies Involved
Static Testing	• Documentation – oriented	• Identification of requirements, targets and methods

	• Specification and Requirements Scrutiny	• Analysis of completeness of information necessary to proceed to development phase • Evaluation of documentation and conformity to standards in the industry
Dynamic Testing	• Project and prototype – oriented • Testing for Cases / Scenarios	• Identification of features involved in project • Verification of the project's achievement of the required functionality through evaluation *via* the *selected standards* and quality assurance *metrics*

Table 1.6 **Approaches for the Project's Quality Assurance Testing Phases**

F. Pass / Fail Criteria

The succeeding table summarizes the specific criteria that may serve as the establishment of the necessary standards to properly evaluate and analyze the application's verification of the required functionalities:

Feature	Margin of Validity	Margin of Fault
Security of User Registration and Login	• User is **registered with a viable username and password** of his/her choice • Username and password **works effectively**	• User **cannot be registered with a viable** username and password • Username and password **does not work effectively**
Correctness and legitimacy of event display	• The events being displayed are **in line** with the client's requirements • The events being displayed are **identical** to the data presently stored in the database	• The events being displayed are **not in line** with the client's requirements, and are variably different • The events being displayed are **not identical** to the data presently stored in the database

Completeness of events being displayed	• The data present in the calendar are **complete** and **conforms** to the populated data	• The data present in the calendar are **incomplete** and **does not conform** to the populated data
Readability of calendar events	• The events displayed in the calendar are **readable** and are **recognizable** • The data **brings clarity** to the prospect users	• The events displayed in the calendar are **not readable** and are **unrecognizable** • The data causes the prospect users to be **misled**
Usability of calendar links	• The links **properly redirect** the user to the necessary audio files / text files / websites	• The links are **intertwined** and **does not follow** the intended file/page to be redirected to.
Dynamism of Administrator and User Privileges	• Administrator privileges are **well-defined** • CRUD **performs** required functionality	• Administrator privileges and user privileges are **not defined** • CRUD **does not perform** required functionality

Table 1.7 **Pass / Fail Criteria**

The following table summarizes the Itemized list of expected and excusable outputs and the reason for providing space for leniency:

Feature	Expected Outputs	Excusable Outputs	Reason for Tolerances and Leniency
Registration	• User must be properly registered	• User may have difficulty and may yield faults typing certain fields	• User error is not engaged by the system, mitigation factors maybe included
Login	• User must be able to login as user / as an admin according to their respective roles	• User may have difficulty and may yield faults typing certain fields	• User error is not engaged by the system, mitigation factors maybe included
Calendar	• Calendar will display output	• A lag in the rendering of	• Somehow, client side queries will take its toll in

	but with inherent latency	events in the calendar	the event renderer, which means that *inherent latency* is an application limitation
Events	• Events will be rendered in accordance to the displayed month	• Some events will load slower than others	• Multiple select statements will be assessed by the system, therefore some events will yield various differentiated effects
Link Redirection	• Links will be assigned for events that have audio / text files for them	• Audio files and text files may vary	• This depends on the physical attributes and specifications of the tester's devices, which may vary
CRUD	• CRUD must be able to perform the mentioned functions synchronized with the database	• CRUD must exercise similarities with app	•

Table 1.8 **Itemized Features and Excusable Outputs**

G. Test Deliverables

a. Deliverables Matrix

The table below shows the list of deliverables that should be produced during the testing phase of the project. Specific deliverables are preferred to be delivered as a part of test validation. These deliverables should opt to support the project's overall objectives and maintain the quality.

DELIVERABLES	
Documents	Test Approach Document
	Test Schedule
	Test Specifications
	Requirements Traceability Matrix
Test Case / Bug Write - Ups	Test Cases / Results
Reports	Test Status Reports
	Test Final Report – Sign Off

Table 1.9 **Deliverables Matrix**

b. Documents

The section of the test plan describes some of the documents needed in performing the testing phase of the project.

i. Test Approach Document

This document describes the overall test approach to be taken in the testing phase of the project. When this document is completed, the Test Lead will distribute it to the development lead, user representative, project manager and others as required for review and sign-off.

Since the project uses the agile approach for development, the following table shows the necessary information needed in proceeding to the testing phase.

Name	Approach	Methodologies Involved
Static Testing	• Documentation – oriented • Specification and Requirements Scrutiny	• Identification of requirements, targets and methods • Analysis of completeness of information necessary to proceed to development phase • Evaluation of documentation and conformity to standards in the industry
Dynamic Testing	• Project and prototype – oriented • Testing for Cases / Scenarios	• Identification of features involved in project • Verification of the project's achievement of the required functionality through evaluation *via* the *selected standards* and quality assurance *metrics*

Table 2.0 **Approach Matrix**

ii. Test Schedule

The test schedule document of this project is thoroughly described and illustrated in another section of this document. Please refer to the ***Schedule section*** of this test plan document.

iii. Test Specifications

The test specification document describes the system requirements, functional and design specifications primarily derived in collaboration with the client. This document provides details with regard to the construction of test cases and the basis of test scenarios indicated in the test cases.

Basically, the software requirements document of this project indicated in the Software Quality Assurance Plan became one of the main reference of the test items listed in the test cases aside from the information gathered by personally collaborating with the project adviser and the client. For further details, please refer to the Software Quality Assurance Plan particularly the Software Requirements document, the test case document and the test cases provided by the developer team.

iv. Requirements Traceability Matrix

001	Provides a user-friendly interface for easier navigation	Highly Needed	In Progress	High	System Interface				
002	Provides a login page so that ***registered users*** can access the site	Highly Needed	In Progress	High	Login	**3.1.3**			
003	Provides a sign up page to allow unregistered users to register and access the site	Highly Needed	In Progress	High	Login	**3.1.2**			
004	Contains a calendar filled with generated readings and audio files	Highly Needed	In Progress	High	Calendar	**3.2.1**			
005	Allows ***registered users*** (***admin or member roles***) to view the calendar and calendar events	Highly Needed	In Progress	High	Calendar	**3.2.1** **3.3.3**			
006	Does not allow ***unregistered users*** to view the calendar and calendar events	Highly Needed	In Progress	High	Calendar				
007	Filters the calendar if specific year is chosen by the user (***admin or member roles***)	Highly Needed	In Progress	High	Calendar	**3.3.4**			
008	Filters the calendar if specific month is chosen by the user (***admin or member roles***)	Highly Needed	In Progress	High	Calendar	**3.3.4.1**			
009	Allows ***registered users*** (***admin or member roles***) to listen to audio files included in the calendar	Highly Needed	In Progress	High	Calendar				

Image 1.8 **Requirements Traceability Matrix Part 1**

016	Registered user must access the website using the login page	Highly Needed	In Progress	High	Login	**3.1.3**			
017	Registered user can view the calendar in a ***monthly basis***	Highly Needed	In Progress	High	Calendar	**3.2.1**			
018	Registered user can navigate through the different months/years of the calendar	Highly Needed	In Progress	High	Calendar	**3.3.4**			
019	Registered user can choose a specific year to navigate	Highly Needed	In Progress	High	Calendar				
020	Registered user can choose a specific month to navigate	Highly Needed	In Progress	High	Calendar				
021	Registered user can view Weekday Readings	Highly Needed	In Progress	High	Calendar				
022	Registered user can view Sunday Readings	Highly Needed	In Progress	High	Calendar				
023	For ***users with admin role***, he/she can perform ***create*** function in the calendar	Needed		Medium	Event				
024	For ***users with admin/member role***, he/she can ***read*** the contents of the calendar	Highly Needed	In Progress	High	Calendar	**3.3.2**			
025	For ***users with admin role***, he/she can perform ***update*** function in the calendar	Needed		Medium	Event				
026	For ***users with admin role***, he/she can perform ***delete*** function in the calendar	Needed		Medium	Calendar				
027	Registered user can click audio files provided in the calendar	Highly Needed	In Progress	High	Calendar				

Image 1.9 **Requirements Traceability Matrix Part 2**

c. Defect Tracking and Debugging

i. Testing Workflow

The following diagram illustrates the testing process for developers and adopters for static and dynamic testing.

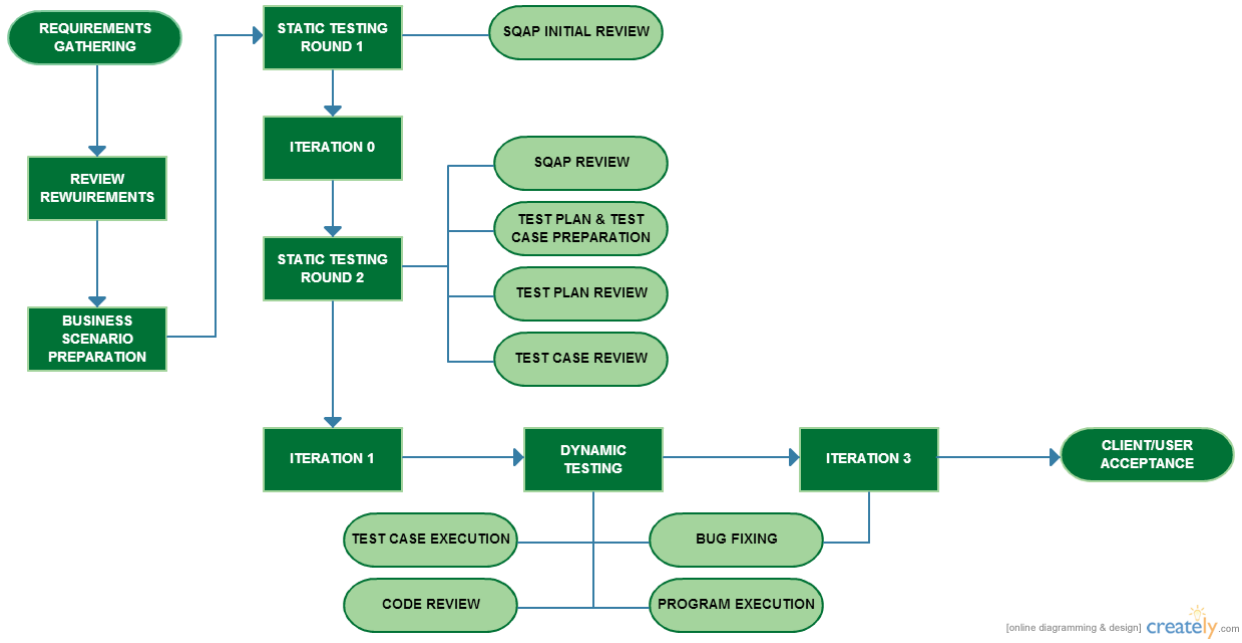


Image 2.0 **Testing Workflow**

ii. Defect Reporting

All bugs or defects should be logged using the Bug Report file created by the developer team. Adopters are also recommended to send bug reports upon the usage of the system. The developers will update the bug report file and notify the adopters after the defect has been reported. The Bug Report file is available in the later part of the Test Plan. Please refer to ***Bug Reports*** section under ***Resource and Environmental Needs*** of the Test Plan to see the actual Bug Report file.

d. Reports

i. Testing Status Reports

The testing status report is created by the developer team to monitor testing activities, its status and the comments in each testing activity. It is provided to the testing team to track

progress in the testing phase. The following images shows the sample table included in the testing status report. Please refer to the link provided below for the actual copy of the file.

TESTING STATUS REPORT							STATUS LEGEND	
Project Name:	AngSalitaNgDiyos.com Liturgical Site						Completed	
Project Type:	Website						Pending	
Project Start Date:	12-Jan-15						Yet to Start	
Project End Date:	12-Apr-15							
Project Adviser/s:	Mr. Joe Gene Quesada (Software Development)							
	Mr. Allan B. Cotecson (Quality Assurance)							
	Mr. Ernesto Boydon (Software Engineering)							
	Ms. Ma. Theresa Montemayor (Database Design and Management)							
Project Manager:	Joshua C. Dimapilis						TESTING	
Project Members:	Kimberly Mae B. Elizondo							
	Trixia Marie A. Urquiza						1. Enter **Tasks Planned** and identify if task is executed in **Task Executed** column	
Tester/s Name:								
							2. Identify **Task Execution Status**	
Tasks Planned	Task Executed (Yes/No)	Task Execution Status	Execution Effort (Hrs)	Complete (%)	Remaining (%)	Comments		
		Blocked			100		3. Input **execution effort by number of hours**	
		Blocked			100			
		Blocked			100			
		Blocked			100			
		Blocked			100		4. Enter percentage of task **completeness**	
		Blocked			100			
		Blocked			100			
		Blocked			100		5. Leave **comments**	

Image 2.1 **Testing Status Report**

File link: https://drive.google.com/file/d/0B-yh-IYX9YZvcW9JTG1jNXRHZW8/view?usp=sharing

ii. Phase Completion Reports

After completing the testing phase, the lead tester must submit the documents provided by the developer team to the testing team. The documents must be accomplished already for review and sign-off. The list of expected documents are as follow:

1. Test Cases
2. Testing Status Report
3. Bug Report

A discussion between the developers and the testing team regarding the content of these documents must also be conducted for further issues and queries.

iii. Test Final Report – Sign Off

The final test report document will be provided by the test lead to the developer team. This document is necessary to verify which testing actually happened, the coverage of the testing phase, the results and the assessment of the system's readiness for deployment.

H. Testing Tasks / Setup

TASKS	DESCRIPTION
Create a Test plan	Detailed documentation of the test plan
Create Test Cases	Detailed documentation of the test cases to be used
Create a Development Database	A development environment type of database to be configured and used during the testing
Execute testing	Execute test scripts and other necessary tasks included in the test case
Detect and evaluate bugs and errors	During the testing there are bugs and errors that the testers will encounter
Report defects encountered during test execution	A detailed report on the defects encountered during test execution
Summary Report of Test	A Summary of the entire test including defects detected and the result of the test.

Table 2.1 **Tasks Setup**

I. Resources and Environmental needs

a. Testing Tools

i. Tracking Tools

In accomplishing the testing phase of this project, the testing team will perform a manual testing strategy to identify system bugs and defects. They will track system performance by executing system processes one by one. If there's any recommendation, the testing team can use any tracking tool compatible in checking the quality of the system.

b. Test Environment

The testing phase of the project will require specific hardware and software requirements that must be compatible in the system testing procedures. The required specifications are as follows:

i. Hardware

The minimum hardware requirements that will be used to test the application are:

- A completely working desktop/laptop
- Stable Internet connection

ii. Software

In addition to the hardware specifications, the following list of software should be considered / must be present in the workstation that will be used for testing:

- The hosted system

c. Bug Severity and Priority Definition

Identifying bug severity and priority level are both very important in categorizing bugs and prioritizing whether the bug must be fixed immediately or not. The testing phase will definitely need the definition of each bug severity and priority level. The tester will be the one responsible in assigning the severity and priority level in each of the test scenario he/she tested. The lead tester must see to it that a correct severity level is assigned to each system bug.

i. Severity List

Severity Level	Severity Description
Level 1 (Critical)	• Impaired functionality • Critically impacting client's processes • The module/product crashes or the bug causes non-recoverable conditions • System crashes or very unstable • Database/file corruption • Potential data loss • Program hangs requiring reboot • Security violation
Level 2 (High)	• Major system feature is missing but not critical • Major system component unusable due to failure or incorrect functionality • Bugs cause problems such as lack of functionality, or insufficient or unclear error messages that can have a major impact to the user • Prevents other areas of the system from being tested • Bugs can have a work around but the work around is inconvenient or difficult
Level 3 (Medium)	• Incorrect functionality of component or process • There is a simple work around the bug • Some system feature or functionality is missing or not working

Level 4 (Low)	• A cosmetic error, spelling mistake, typographical errors. • Documentations errors • Signed-off severity 3 bugs

Table 2.2 **Severity List**

ii. Priority List

Priority	Priority Level	Priority Description
5	**Very High**	This bug must be fixed immediately. The system cannot proceed with further processes with this bug.
4	**High**	Important bugs that should be fixed as soon as possible. (Loss of function)
3	**Moderate**	Bugs that should be fixed within the time available. (Gradual performance)
2	**Low**	Bugs that are not important (at this time). These bugs can be fixed after all other high priority bugs are fixed.
1	**Very Low**	Enhancements/Good to have features incorporated – just are out of the current scope. (Minor nuisance)

Table 2.3 **Priority List**

d. Bug Reporting

In recording reports on existing bugs, the testers can use the table in an excel file created by the project team. The table is consist of three columns wherein they will log all the

BUG REPORT								STATUS LEGEND
Project Name:	*AngSalitaNgDiyos.com Liturgical Site*							*Active*
Project Type:	*Website*							*Resolved*
Project Start Date:	*12-Jan-15*							*Fixed*
Project End Date:	*12-Apr-15*							*Blocked*
Project Adviser/s:	*Mr. Joe Gene Quesada (Software Development)*							
	Mr. Allan B. Cotecson (Quality Assurance)							
	Mr. Ernesto Boydon (Software Engineering)							
	Ms. Ma. Theresa Montemayor (Database Design and Management)							
Project Manager:	*Joshua C. Dimapilis*							**TESTING INSTRUCTIONS:**
Project Members:	*Kimberly Mae B. Elizondo*							1. Enter **Test Description** and explain results in **Expected and Actual Behavior**
	Trixia Marie A. Urquiza							2. Identify **which Test Case** bug / issue was found
Test No.	*Test Description*	*Expected Behavior*	*Actual Behavior*	*Test Case*	*Tester and Proof of Test*	*Developer's Resolution*	*Status*	3. Input name of Tester and **format** of proof of test (script, printscreen)
1							Blocked	4. Leave Developer's Resolution **Blank**
2							Blocked	5. Change Status to '**Active**'
3							Blocked	
4							Blocked	
5							Blocked	
6							Blocked	
7							Blocked	
8							Blocked	
9							Blocked	
10							Blocked	
11							Blocked	

tests performed, describe the procedures of the test and record its results. The link to the actual bug report excel file is shown below.

Image 2.2 **Bug Report**

File link: https://drive.google.com/file/d/0B-yh-IYX9YZvR3pQMnV0SXI4TlE/view?usp=sharing

J. Responsibilities

a. Developer

The developer team of this project, mainly consists of ***Joshua C. Dimapilis, Kimberly Mae B. Elizondo*** *and* ***Trixia Marie A. Urquiza*** from **Asia Pacific College**, take this project to perform software development activities that will suffice the client's current needs. The responsibilities of the developer team are as follows:

1. Develop the system application, in line with the client's requirements
2. Develop use cases and finalize requirements in collaboration with the clients/adopters
3. Develop Software Quality Assurance Plan of the system
4. Development and coding processes involved in building up the system
5. Conduct different kind of tests for different group (static and dynamic testing)
6. Support user acceptance testing
7. Develop test plan and test cases primarily for testing procedures
8. Maintenance and support on possible system bugs upon initial deployment of the system

b. Adopter

The adopter or mainly the ***client*** of this project is the ***Archdiocesan Liturgical Commission*** located in Manila. They will undertake formal adoption, testing, validation and application of the system developed upon accomplishing all the requirements needed in the system. The adopter's main responsibilities in this project are as follows:

1. Contribute to the development of use cases and requirements through intense review
2. Involvement in the iterations conducted upon developing the project
3. Contribute to the development and execution of the development test scripts through intense review
4. Conduct full User Acceptance, regression, and end-to-end testing; the said tests will include identifying testing scenarios, creating of test scripts, executing scripts and reporting of results
5. Feedbacks and recommendations upon initial testing of the system for further enhancements

c. Testing Process Management Team

The testing process management team is responsible in managing the entire testing process, workflow and quality management which is led by the testing team, the people who will test the system and record the test results. The developers and the adopter are also included in the testing process management team but with minimal involvement. This team's responsibilities are as follows:

1. Monitor and manage honesty and integrity in the testing process
2. Conduct different kind of tests (static and dynamic testing)
3. Support testing activities
4. Provide an accurate testing result for further enhancements in the system
5. Develop bug and test reports
6. Feedbacks and recommendations after conducting the testing procedures

K. Staffing and Training

The team decided to use the simple method for staffing people for the project's testing phase. The testing team is consists of the people who will test the system components and functionalities and is mainly composed of 3 testers. They are assumed to be knowledgeable of the system they are going to test. Test cases will serve as testing tools in evaluating if the components of the system pass the set of criteria. These test cases are created by project team and will be handed over to the testing team before proceeding to the testing phase. The lead tester may designate the test cases to the other testers for organization and specific evaluation. Testers will test each system component and will take note of the result using a test result table or document. Once the system is completely developed, all members of the project team will test the system thoroughly.

L. Schedule

DELIVERABLES	DATE	STATUS
1st round of Static Testing • First Round of Static Testing • Project Requirements Definition • Project Logical Design • Project Physical Design	**2/12/2015**	**COMPLETE**

2nd Round of Static Testing • Database Design • User Interface • Software Program Logic / Code • Software Error Handling • Test Plan • Test Cases • Overall Quality of Documentation • Documentation Completeness	**3/12/2015**	
Dynamic Testing • Test Plan Procedure • Test Cases • Program Flow • User Interface • Error Handling • Over all User Experience	**03/30/2015**	

Table 2.4 **Schedule**

M. Risks and Contingencies

The following are the overall risks on the testing process:

Areas	Risks	Propositions
Tester	• Lack of personnel resources when testing is to begin. • Lack of Knowledge about the system	• Beforehand testers will be advised about the testing schedule. • Testers will be given resources such as wiki

		page and test plan to provide background about the system and as well as on how the testing goes.
System	• Delayed response • Encountered bugs / errors • Delivered wrong output	• It is indicated in the test plan all the possible exceptions for defects / errors, testers must be aware of the list of allowable defects. • For the defects / errors detected that are not part of the exempted error findings. Testers must note the reports about the errors.
Environment	• Lack of internet connection • Lack of software and / or hardware • Risk of web - hosting and storing massive amounts of data in the database	• Before testing takes place, all the necessary resources must be checked if properly working. • Hosting the site, with large amounts of data may be mitigated by specifying early on with the client, the expected and the actual size of the data to be stored and used for the project
Change Management	• Changes to the original scope of the system	• If changes on the scope have been made, testers will be advised on the possible effects of the changes and will state to the documentation so that they are aware.

Table 2.5 **Risks and Contingencies**

N. Approvals

The names listed below are the authorized people who can approve the process as complete and allow the project to proceed to the next level.

NAME	POSITION	DATE	SIGNATURE

VII. Deployment Plan

A. Purpose

a. Introduction

Project Identification	
Project Name	AngSalitaNgDiyos.com Liturgical Site
Project Deployment Team	Joshua C. Dimapilis
	Kimberly Mae Elizondo
	Trixia Marie Urquiza
Date Created	03/08/2015

Table 2.6 **Project Deployment Identification**

B. Deployment Strategy

The deployment of the project involves the integration of the existing system created by the group adviser, and the system being created by the team. Proper allocations to the deployment strategy are to be discussed in line with the client's requirements. Cross-side scripting, server-hosting and testing are all part of the *Deployment Strategy* meeting to be conducted with the group adviser and the client.

C. Deployment Overview

Deployment Summary	
Number of Sites or Recipients	
Target Deployments	
Target Group	
Scheduled Dates	
Deployment Approach	
Benefits	
Assumptions	
Risks	

Table 2.7 **Deployment Summary**

D. Deployment Schedule and Resources

- The required schedule and resources are still to be announced. ***(03/08/2015)***
- The tentative integration schedule can be within the range of the following dates: March 23 – March 28, 2015 ***(03/11/2015)***

E. Engagement Strategy

a. Description

Deployment strategy involves the following criteria:

- Functionality – Functions of developed application must be inherited by the integrated calendar
- Scalability – The calendar allows room for growth, and is in line with the existing system's possible opportunities for extension
- Performance – The calendar performs in line with what is expected of it based on the discussed quality assurance assessments in this plan

b. Technical Migration / Deployment Methods

For technical migration, the necessary deliverables are:

- Functional application
- Deployment schedule
- Administrator rights to existing team adviser's site
- Wrapper / iFrame instructions
- "Go – signal" from the client

The deployment steps include:

- Usage of administrator rights to be able to access page to which it will be deployed
- Creation of site wrappers that may include the functional application
- Deployment approval from the client
- Deployment metrics to ensure alignment with deployment schedule

F. Technology, Infrastructure and Support Considerations

a. Testing Methods and Customer Acceptance

Testing methods include the testing specifications described in the Test Plan section of this document. The quality assurance metrics are still to be considered in circumstances of testing.

Testing methods that must be exercised on the client side:

- System Integration Testing *(By proponents)*
- User – Acceptance Testing
- Regression Testing

b. Training Requirements

Requirements for training the idealized users of the application are naturally in place. The steps involved in this training requirements may be assessed using the system's quality metrics discussed in the Software Project Management Plan of this document. Intersecting the metrics to the features to be tested in the Test Plan, we may arrive with the following training manual sections:

- **Registration and Login**
- **Calendar view**
- **Reading view**
- **Event view**
- **Link Redirection**
- **CRUD**

c. Possible Issues and Conflicts

After Static and Dynamic Testing has been accomplished, this area will be populated with possible issues, bugs, defects and other conflicts that may arise during application deployment and usage for the client.

VIII. Acceptance Plan

IX. Installation & Acceptance

X. Appendices

The following section summarizes the documents referenced in this document.

A. References

Document Name and Version	Description	Location
Deployment Strategy and Plan	Deployment and Strategy Plan from Ohio Board of Regents	• University System of Ohio • Download Link from Ohio Regents
CDC UP Test Case Template	Centers for Disease Control and Prevention UP Test Case Template	• CDC UP Test Case Template

Table X.X **List of References**