STUDENT RECORDS MANAGEMENT SYSTEM

|  |  |
| --- | --- |
| **Author** | **Albano, Meliza Marie**  **Donato, Frances Angeli**  **Hernandez, Merwin**  **Medinaceli, Gemmalyn** |
| **Creation Date** | **January 7, 2022** |
| **Last Updated** | **January 8, 2022** |
| **Document Reference** | **Client name SINGLE LOGIN – TEST PLAN** |
| **Version** | **Version 1.0** |

**Approvals:**

|  |  |
| --- | --- |
| Approver 1 (Position) |  |
| Approver 2 (Position) |  |
| Approver 3 (Position) |  |

Table of Contents

[1. Document Control 3](#_Toc465064423)

[Change Record 3](#_Toc465064424)

[2. INTRODUCTIONS 4](#_Toc465064425)

[Overview and Purpose 4](#_Toc465064426)

[Scope 4](#_Toc465064427)

[3. Cutover Strategy 5](#_Toc465064428)

[Cutover Team 5](#_Toc465064429)

[Organization 5](#_Toc465064430)

[Roles and Responsibilities 5](#_Toc465064431)

[Cutover Approach 5](#_Toc465064432)

[Pre-cutover Migration 5](#_Toc465064433)

[Migration Proper 6](#_Toc465064434)

[A. Database 6](#_Toc465064435)

[B. Single Login Website 6](#_Toc465064436)

[C. Mini – System Configuration 6](#_Toc465064437)

[Post Migration 7](#_Toc465064438)

[Revert Procedures 7](#_Toc465064439)

[Issue, Cause and Proposed Solution 7](#_Toc465064440)

[Assumption 8](#_Toc465064441)

[Completion Criteria 8](#_Toc465064442)

[Cutover Date 8](#_Toc465064443)

[Risk Assumption 8](#_Toc465064444)

[4. Appendix A: Adding Website to IIS 9](#_Toc465064445)

[5. Appendix B: Test Plan 12](#_Toc465064446)

[6. Appendix C: Client Code Configuration 13](#_Toc465064447)

# Document Control

## Change Record

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Version | Remarks |
| January 7, 2022 | Albano, Meliza Marie  Donato, Frances Angeli  Hernandez, Merwin  Medinaceli, Gemmalyn | 1.0 | Draft |
|  |  |  |  |

# INTRODUCTIONS

## Overview and Purpose

The purpose of this document is to define how the migration of Student Records Management System. The document that will support this document should be referred to the Test Acceptance Result. Therefore, this document is not advisable to be followed unless Test Acceptance Certificate has been approved.

## Scope

The scope of this document are as follows

* Defining the approach on how this migration be performed as well as defining who would be responsible/assigned on every tasks mentioned.
* Pre – migration
  + Contains procedures that needs to be done before the implementation of actual migration proper.
* Migration Proper
  + Contains the step – by – step procedure in order to deploy the web application.
* Post Migration
  + Specifies the procedures to be done in case the migration proper fails.
  + Specifies when to decide to revert the procedure.
  + Specifies what are the possible risk or issues to be encountered and how can this be avoided.

# Cutover Strategy

## Cutover Team

The cutover team is composed of Client name (Client) and Intellismart Technology, Inc. (Vendor) representatives.

## Organization

|  |  |  |  |
| --- | --- | --- | --- |
| **Role** | **Side** | **Team Member** | **Signature** |
| Project Manager | ITI | Donato, Frances Angeli |  |
| Front – End Developer | ITI | Albano, Meliza Marie |  |
| Back – End Developer |  | Medinaceli, Gemmalyn |  |
| Back – End Developer |  | Hernandez, Merwin |  |

## Roles and Responsibilities

|  |  |
| --- | --- |
| **Roles** | **Responsibilities** |
| PM – Integration | * Develops the Cutover strategy and Cutover Plan * Provide patches/fixe and resolution defects encountered in Shallow testing |
| Project Manager | * Obtain commitment from the Cutover Team * Provide updates to cutover team * Manages cutover procedure * Work with the Client name PM for the Evaluation of Cutover Plans * Provide Required documentation |
| Client name IT Representative | * Ensure proper setup of servers, directories, database |
| Client – Functional Tester Representative | * Shallow testing of the application * Decides and gives advice to PM, IT Representative whether to proceed or rollback the application. |

## Cutover Approach

The Cutover will be conducted in accordance to this Plan. This document must be approved by all concerned parties prior to the conduction of the cutover activity.

Actual cutover shall be done within Client name premises. Representatives for each cutover team shall be present for the actual cutover.

## Pre-cutover Migration

1. Pre-Requisites
2. Upload the file containing the database on GitHub
3. Upload the file containing the system on GitHub
4. Set the GitHub repository public
5. Backup
6. Perform backup of the following:
   * + Student Records Management System – web application
     + Student Records Management System – database
7. Save the backup to safe location for revert procedure in case the migration is not successful.
8. Preparation of Server for mini system and Student Records Management System Server
   1. Ubuntu Server
      * The application runs under Oracle VM VirtualBox
      * Follow product installation guides if not yet installed
   2. Web Server
      * Tested under environment
      * Windows 10 64-bit, 8GB RAM, i5-8250u CPU for both DB and Web Server

## Migration Proper

### Database

|  |  |  |
| --- | --- | --- |
| **Database** | Start Time | End Time |
| 1. Decompressed Database |  |  |
| 1. Open MS Sql Server |  |  |
| 1. Restore the database |  |  |

### Student Records Management System Website

|  |  |  |
| --- | --- | --- |
| **Student Records Management System Website** | Start Time | End Time |
| 1. Create folder “StudentManagement” in drive C 2. Decompressed pre-requisite number 2 inside Website folder |  |  |
| 1. Open IIS and follow [Appendix A: Adding Website in IIS](#_Appendix_A:_Adding) |  |  |
| 1. Open Web.config that can be found inside the folder created in Step 1and change the following property value field   <add key="Server" value="ip-address" />  <add key="username" value="username" />  <add key="password" value="db password" /> |  |  |
| 1. Save and Restart the Server |  |  |

### Mini – System Configuration

|  |  |  |
| --- | --- | --- |
| **MINI SYSTEM** | Start Time | End Time |
| 1. Create folder JS folder in Drive C 2. Decompressed pre-requisite number 1 |  |  |
| 1. Load all the JS files in your code (normally Login page, or Home Page). See sample    1. <script src="js/jquery.min.js"></script>    2. <script src="js/jquery.signalR-2.2.0.min.js"></script>    3. <script src="js/lz-string.js"></script>    4. <script src="js/Client.js"></script> |  |  |
| **NOTE:** in case there is a conflict in jQuery since there is already jQuery in your code, disregard loading the jQuery links. |  |  |
| 1. If you want to verify that the current user in SINGLE LOGIN WEBSITE is the same user as returned by the MiniSystem when user accessed from SINGLE LOGIN WEBSITE, Open “Client.js” and Locate the Session part and compare the values returned to your database. |  |  |
| \_.Session.on('onConnect', function (**data**) {  \_.Data = data;  \_.Session.invoke('successfulConnection');  ///////////////////////////////  //Please type your code below//  ///////////////////////////////  console.log("Client onConnect: ", data);  });  Object **data**, refer to Table 1: Data |  |  |
| 1. Locate the   var server = 'http://localhost:9099/'; //Change this according to the server address  Change the IP Address and Port of the Student Records Management System website |  |  |
| 1. For DotNet (VB.Net/C#) minisystem, locate the Web.Config and Add a variable called TokenID inside the Configuration in AppSettings   <AppSettings>  <Configuration>  <add key=”TokenID" value=" aad72010-24f2-4865-9e8d-b0a8b5ab7111" />  …  </Configuration>  </AppSettings> |  |  |
| 1. For PHP, define a constant global variable and add the TokenId generated from SingleLogin   define(‘TokenID’,’aad72010-24f2-4865-9e8d-b0a8b5ab7111’); |  |  |
| 1. Follow [Appendix B](#_Appendix_C:_Client) for the login configuration |  |  |

Table 1. Data Returned Property Value Field

|  |  |  |
| --- | --- | --- |
|  | **Description ID** | **Sample Value** |
| ConnectionId | Unique key identifying the connection between the Single Login Website and the Mini-System  Date Type: GUID | aad72010-24f2-4865-9e8d-b0a8b5ab705b |
| EndDate | Specifies until when the connection ID can be used.  Data Type: Date Time | Date Time  24 hours from the time of access |
| ID\_Client | SINGLE LOGIN WEBSITE identifier of the MiniSystem | 2  Referring to let’s say Multi-Purpose Loan  Depends on the ID saved in the database |
| Username | Must be documented from client that User is the same from all minisystem |  |
| Token | Unique value from SINGLE LOGIN WEBSITE  For matching or verification purposes of mini-system and what is saved in SINGLE LOGIN WEBSITE  Data Type: GUID | aad72010-24f2-4865-9e8d-b0a8b5ab7111 |

## Post Migration

To verify that the Reservation System Website is working properly,

1. Register client to STUDENT RECORD MANAGEMENT SYSTEM WEBSITE
2. Access the STUDENT RECORD MANAGEMENT SYSTEM WEBSITE in different workstation
3. Login according to number 2
4. Client can book reservations.
5. For each mini-system opened it should be redirected to after successful login page

## Revert Procedures

1. Stop the instance of Student Records Management System in the Web Server
2. Backup all the updated mini systems (those applied with configuration) and save it to a secure directory or external drive for future reference.
3. Delete the updated files as soon as backup have been performed
4. Return the original file. This should return the original state of the mini systems.

## Issue, Cause and Proposed Solution

* All issues encountered after the completed migration (with signed – off) will be given support by ITI within\_\_\_\_ hours.
* If the issue identified is procedural, it is the function of the Client name IT Representative to provide the detail how to solve it.
* If Issue identified is Technical that needs development or deep IT knowledge, raise it to ITI.

## Assumption

* The application went through several testing between ITI and Client name.
* All documents and approvals for migrations has been secured and completed before the actual migration.

## Completion Criteria

The cutover is considered complete when the following criteria are achieved:

* All activities mentioned in this document have been completed
* Shallow testing has been completed and accepted (post migration).
* The Sign – off for this Cutover (See Last Page) have been signed by both parties.

## Cutover Date

The cutover date shall proceed on

**<<Date and Time>>**

## Risk Assumption

# Appendix A: Deploying Website on Ubuntu

**Note:** All values indicated in input fields of the images are for sample purposes only.

|  |  |
| --- | --- |
| **Step 1** | First, you need to have an Ubuntu Server. We downloaded the Oracle VM VirtualBox since we do not have a specific device for Ubuntu Server. VirtualBox can use Ubuntu Server at the same time using your windows machine. |
| **Step 2** | After downloading the Oracle VM VirtualBox, the second thing you will do is download the Ubuntu Server.iso file to have the operating system of the Oracle VM VirtualBox. After downloading and installing the Ubuntu Server, click the Start button. |
| **Step 3** | After clicking the start button, you will automatically proceed to the login screen. Once you see the login, type the username and password you created. |
| **Step 4** | Once you log in to your user account, this will be the welcome page of the Ubuntu Server. |
| **Step 5** | Fill up the Database, Password, Server, Port and Username of the server. To change the values of the application settings, double on the properties (as indicated with asterisk). A dialog box will appear where you can change the values. |
| **Step 7** | Go to IIS. Click on the website then click on Basic Settings. |
| **Step 8** | Click “Connect as…” |
| **Step 9** | Click “Set…” |
| **Step 10** | Type the Username and Password of the server pc then click “OK”. |
| **Step 11** | To verify that the credentials encoded in Step 4 are correct, click the **Test Settings** button and you should see 2 checkmarks with color green saying Authentication. |
|  | Once successfully connected, you can now access the web application through your internet explorer browser. |

# Appendix B: Client Code Configuration

For each mini – system, add the following code before your Login authentication so that when the Login page loads it checks or validates first if the event (Load event) came from the Single Login Website. If the event load did not came from the Single Login Website, then the normal behavior of the mini – system should redirect the user to the actual Login Page.

|  |
| --- |
| **C#** |

/\*\*

\* \* This is the method that will execute if the connection is successful

\* \* Use the parameter 'data' to get the connection Token and other properties from the server

\* \* The method onConnect() will not work if not called from Single Login

\*/

\_.Session.on('onConnect', function (data) {

\_.Data = data;

\_.Session.invoke('successfulConnection');

///////////////////////////////

//Please type your code below//

///////////////////////////////

**/\*EXAMPLE\*/**

setTimeout(function () {

$.ajax({

method: 'POST',

url: '/Security/Login2',

data: { userName: data.User, token: data.Token }

}).done(function (data) {

if (data.ErrorMsg) {

$(document).trigger('VerificationError', data.ErrorMsg);

return;

}

var result = JSON.parse(data.ResultSet);

var module = {};

Module.call(module, { Controller: 'Security' });

var StartSession = module.Action({

name: 'StartSession',

param: { ID: result.ID, ID\_Company: result.ID\_Company, ID\_Employee: result.ID\_Employee }

}).then(function (d) {

module.UserRow(JSON.parse(d));

});

var DailyAutomation = module.Action({

name: 'DailyAutomation',

controller: 'MenuSet'

});

$.when(StartSession, DailyAutomation).done(function () {

var session = module.Session;

InSys.Signal.Register(session('ID\_User'), session('User'), session('IPAddress'));

window.location.href = '#index';

});

});

}, 1000);

**/\*END OF EXAMPLE\*/**

});

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

|  |
| --- |
| **PHP** |

/\*\*

\* \* This is the method that will execute if the connection is successful

\* \* Use the parameter 'data' to get the connection Token and other properties from the server

\*/

\_.Session.on('onConnect', function (data) {

\_.Data = data;

\_.Session.invoke('successfulConnection');

///////////////////////////////

//Please type your code below//

///////////////////////////////

console.log("Client onConnect");

**/\*EXAMPLE\*/**

$.ajax({

method: 'POST',

url: '/pages/crc\_handler.php?method=login2',

data: { username: data.User, tokenId: data.Token }

}).done(function (data) {

var address = 'http://' + window.location.host + '/pages/crc\_main.php';

if (address != window.location.href)

window.location.href = '/pages/crc\_main.php';

});

**/\*END OF EXAMPLE\*/**

});

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/