High Level Design

For Red Turf

Version 1.3

Tanvi IT Solutions Inc

4211 Pleasant Valley Rd,

Suite 230

Chantilly,

VA- 20151

**Revision History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version (x.y) | Date of Revision | Description of Change | Reason for Change | Affected Sections | Approved By |
| 1.1 | 1/10/18 | Baseline Version |  |  | Nagoor Inaganti |
| 1.2 | 8/10/18 | Addition of modules | Requirement Development | Module Architecture | Nagoor Inaganti |
| 1.3 | 15/10/18 | Change in the design approach | Integration of modules | Design Approach | Nagoor Inaganti |

**Approval History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version (x.y) | Prepared By | Reviewed By/Date | Approved By/Date |
| 1.1 | Vijaya Somarepetta | Monalisha Mishra  1/10/18 | Nagoor Inaganti  1/10/18 |
| 1.2 | Vijaya Somarepetta | Monalisha Mishra  8/10/18 | Nagoor Inaganti  8/10/18 |
| 1.3 | Vijaya Somarepetta | Monalisha Mishra  15/10/18 | Nagoor Inaganti  15/10/18 |

Table of Contents

[1 Overview 4](#_Toc382398296)

[1.1 Scope of this Document 4](#_Toc382398297)

[1.3 Platform and Tools 4](#_Toc382398299)

[1.4 References 4](#_Toc382398300)

[1.5 Assumptions 4](#_Toc382398301)

[1.6 Design Constraints 5](#_Toc382398302)

[2 Design Approach/ System Overview 5](#_Toc382398303)

[3 Alternative Design Approaches Considered 5](#_Toc382398304)

[4 System Architecture 5](#_Toc382398306)

[5 Use Cases 6](#_Toc382398307)

6 Class Diagram………………………………………………………………………………………………………………7

[7 External Interfaces 8](#_Toc382398310)

[8 Initialization and Termination 8](#_Toc382398311)

[9 Error Handling Strategy 8](#_Toc382398312)

[10 Design for Testability and Maintainability 8](#_Toc382398313)

[11 Traceability Matrix 9](#_Toc382398314)

# Overview

Red Turf project is an Enterprise Business Intelligence product. The project is targeted to be used as a part of Campaign and Constituent Management services. Red Turf is a cloud based integrated online system which can be accessed from any location from devices like Laptops, tablets and/ or on smart phones. The project is focused on alleviating issues of data integrity and technology to streamline the process. The Red Turf system performs elastic search along with various social media integration without any security breach.

## Scope of this Document

The scope of the document is to perform the high-level design by identifying the different modules and their interfaces. The modules in the Red Turf project include Search, Dashboard, Resident, Task, List, Mapping and Electronic Check-in.

## Platform and Tools

The following are the tools that are being used and the platform on which the system is being designed for

|  |  |
| --- | --- |
| Tool Name | Purpose for which the Tool is proposed to be used |
| MVC framework .NET | Tool is used for coding and development of the project. |
| SQL | Tool used for communicating with the traditional database. |
| SOLR- LUCENE | Tool used for Big Data storage |
| Azure | Tool used for hosting the project in the cloud |
| ArcGIS | Tool used for Geospatial information |

## References

The documents referred to in the preparation of this document or those on which this document is based, are as follows:

Software Requirement Specification

Project Management Plan

Build Plan

## Assumptions

The following assumptions are made during this design:

The coding is performed using .NET programming language and in the MVC framework.

The code is compatible with the hosting environment.

## Design Constraints

The following are the constraints considered while preparing this document:

External integration requirement (for open architecture)

Customer specific architecture

# Design Approach/ System Overview

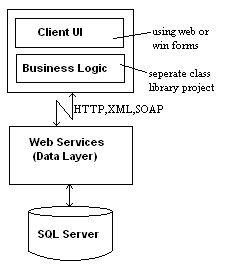
The design approach includes overview of the system that is being architected. This section provides the information needed for a system development team to build and integrate the code and integrate the software modules into a functional product. The design approach chosen for this project is object-oriented approach. Hence, the features such as Search, Dashboard, Resident, Task, List, Mapping and Electronic Check-in are integrated along with the Geographic Information System.

# Alternative Design Approaches Considered

The different alternatives considered in the design approach includes functional-oriented approach. The following are the advantages of choosing object-oriented programming. It provides a great benefit in designing large programs, which can be easily divided into smaller parts and helps in distinguishing the components or phases that need to be executed or planned in a certain way. Object-oriented programming based on the main features that are: 1. *Abstraction*: It helps in letting the useful information or relevant data to a user, which increases the efficiency of the program and make the things simple. 2. *Inheritance*. It helps in inheriting the methods, functions, properties, and fields of a base class in derived class. 3. *Polymorphism*: It helps in doing one task in many ways with help of overloading and overriding which is also known as compile time and run time polymorphism respectively. 4. *Encapsulation:*It helps in hiding the irrelevant data from a user and prevents the user from unauthorized access. Since data integrity is the important feature of the application, encapsulation feature of the approach assists in maintaining high level security of the data.

# System Architecture

The system architecture includes different modules and processes in the software. UML diagram illustrates the different components and their interaction with each other. An XML Web service is used to access the data layer. The typed dataset is returned across the HTTP layer to the business rule layer. The application then consumes the dataset for the user interface presentation of the data.



# Use Cases

Use case is a list of actions or event steps typically defining the interactions between a role (known in the [Unified Modeling Language](https://en.wikipedia.org/wiki/Unified_Modeling_Language) (UML) ) and a system to achieve a goal.

**Use Case 1:**

When the user creates a username and password during the login session, the web application opens.

**Use Case 2:**

Once the user logs in to the application, the user can search the database using the Search module. The features of the Search module include search by last name, search using first name & last name, search using a string, search by name with special characters.

**Use Case 3:**

The user can also view the election analysis, charts, notifications and import data from the Dashboard module.

**Use Case 4:**

The user has access to the database containing the call lists, mail lists, walk lists. In this module the user can create, edit or delete the lists if they have the permission to perform the activities.

**Use Case 5:**

The mapping module in the application using the geo spatial information helps in displaying the data about the voters and the election campaigns. User can select four points in the map and view the data within the contoured area.

**Use Case 6:**

User can electronically check-in using the application before the election and hence help reduce the waiting time before voting in the election.

User

# Class Diagram:

The modules are accompanied by a UML class diagram. The class diagram contains the important classes and the classes used in interacting with the other modules.

 Login

Userid : string

Name: string

Address: string

Contact: string

Location: string

Email: string

viewAnswers()

manageData()

viewReports()

 Login

User name: string

Password: string

authenticate()

 Administrator

User name: string

Password: string

Create ()

Update ()

Delete ()

View ()

Data

DataId : Number

View ()

Campaign

Manager

User name: string

Password: string

Name: string

Address: string

Contact: string

Subject : string

Email: string

Attendance()

View ()

Volunteer

VolunteerId : String VolunteerName: string

VolunteerLocation: string

warehouseAddress : String

warehouseManagerId : String

 Reports

View ()

Charts ()

Close ()

# External Interfaces

The interfaces of the software being developed to the other software components in the system, which are external to the software being described in this document. These may be other components like the user interface or third-party software components, or customer developed software components that are being used in the system. The software being developed is a building block for some other piece of software, the external interface that can be used by the user software are to be described here. Here take into consideration, extensibility in terms of the APIs. The geo spatial codes and addresses are validated using the data from the database of US postal services.

# Initialization and Termination

**Initialization:** When the admin logs on to the cloud and configures the web application, the project is available for use. Admin with the access to the database server invokes this software. The sequence in which the processes are spawned in the system are described in the system architecture section.

**Termination:** The termination of the application is initiated by the admin. When the admin terminates the service through the hosting environment the application is halted.

# Error Handling Strategy

The error handling strategy is followed in the design of the individual modules. For .NET framework, the most popular logging libraries are probably NLog.

NLog typically creates a static logger instance in every class that needs to write anything to the log:

|  |
| --- |
| private static Logger logger = LogManager.GetLogger("LoggerName"); |

The language independency of error messages is helpful in the error handling strategy. With the attribute, additional metadata can be used to table the log. From the table log the designers and coders handle the errors.

# Design for Testability and Maintainability

**Testability:** Each module is tested based on the logging of the state in log files or other mechanisms as specified in the Use Cases.

**Maintainability:** Debug levels and messages for each level which can be optionally turned on and off at run time. The code size of each module along with increase and adverse effects that may be caused because of maintainability is also considered.

# Traceability Matrix

Refer the updated Traceability matrix. Traceability matrix is updated with High Level Design references.