ECF – NGO

Software Design Document

Team: Bazzinga!!

Table of Contents

**1. INTRODUCTION**

1.1 Purpose

1.2 Scope

1.3 Overview

1.4 Reference Material

**2. SYSTEM OVERVIEW**

**3. SYSTEM ARCHITECTURE**

3.1 Architectural Design

**4. DATA DESIGN**

4.1 Data Design

**5. HUMAN INTERFACE DESIGN**

5.1 Screen Images

**6. REQUIREMENTS MATRIX**

**7. APPENDICES**

Introduction :

This document provides a high level description of the planned design and implementation of Bazzinga.

Purpose :

This document provides a high level description of the design and implementation o Bazzinga, a Equal Community Foundation ECF - NGO real time problem solution which includes automation system for donor management system. The low level components are designed, coded and tested.

Scope :

Bazzinga provides a Web platform where Donor can donate desired amount/money to the NGO through web based application.

Intended audience:

This document has been written for consumption by software engineers, software architects, user interface designers.

System Overview:

Minimum Requirement for the system – Apache Server,.net Framework 4.5, IIS 6.0 and above ,Web Browser,Php 5.4 and above.

Section 1: Introduction

A brief explanation of the purposes, goals, and format of this System Design Document.

Section 2: System Overview

An overview of the goals and objectives for Donor Management System project. This section also provides a short explanation about each component and process to be developed/implemented under the Donor Management System project.

Section 3: System Architecture

This Section gives a high level overview of how responsibilities of the system are partitioned and then assigned to subsystems. It provides decomposition of the subsystems in the architectural design. The system will be separated into various software modules. The main packages provides a user interface for test script management.

Section 4: Data Design

This Section is subdivided into data description and data dictionary; the former explains how the information domain of the system is transformed into data structures and how the major data is stored , processed and organized. The latter lists the system entities along with the types and descriptions.

Section 5: Component Design

In this section we take a closer look at each component in a more systematic way.

Section 6: Interface Design

Here we provide an overview of the user interface of the Donor Management System.

Section 7: Requirement Matrix

In this part of the document we discuss requirements matrix which is used to check if the current project requirements are being met, and to help in the creation of  software requirement specification various deliverable documents, and project plan tasks.

1.4 Reference Material

www.ecf.org.in

1.5 Definitions and Acronyms

GUI:Graphical User Interface

API:Application Programming Interface

**2. SYSTEM OVERVIEW**

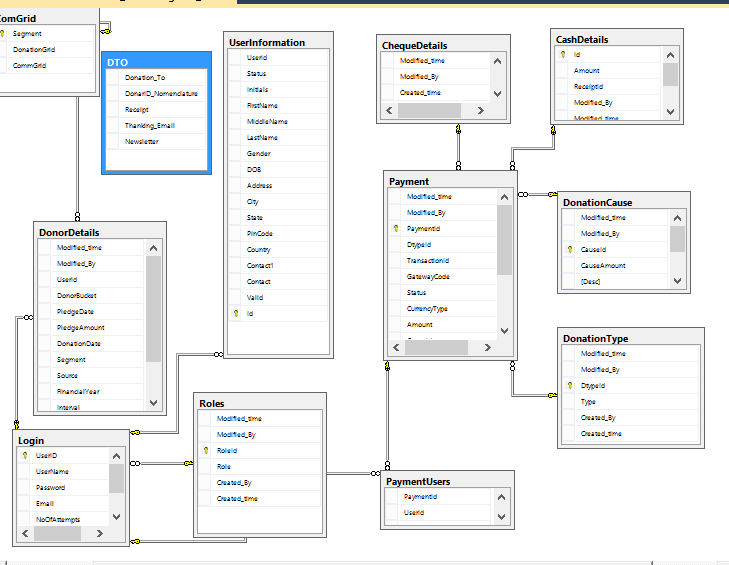
Functionality – The donor management system provides automated system for donation through a web based application.Donor can register,login and make desired donations as per his/her interest.

Context – Donor management system is a web-app encompassing many components and features within itself. Donor management system is a system for complete management of donors .

Design- The software has been designed keeping in mind the user's perspective of a Problem faced by NGO. The various software components – WCF, PHP,HTML,CSS,Javascript,Sql Server etc have been chosen in such a way that interaction between them is not cumbersome. The design has been simple yet effective. The components have been thoroughly explained in the component design section.

Background information- The user of the system must have an account in Donor Management system. He/she should have basic knowledge of logging and usage of internet.It has its own user-authentication model, we have just plugged it in.

Database Design:



**3. SYSTEM ARCHITECTURE**

Donor management system basically comprises of components- C# .net,WCF and Database SQL SERVER with front end validation in Php, HTML and JS with all new js frameworks like jquery.

Php is chosen to integrate with existing system of the ECF.

C#.net is a high-level language framework that encourages rapid development and clean, pragmatic design. It lets us build high-performing, elegant Web applications quickly.

The advantages of choosing WCF over other available frameworks are:

- Single Box

- Easier Development of SOA applications.

Windows Communication Foundation (WCF) is a framework for building service-oriented applications. Using WCF, you can send data as asynchronous messages from one service endpoint to another. A service endpoint can be part of a continuously available service hosted by IIS, or it can be a service hosted in an application. An endpoint can be a client of a service that requests data from a service endpoint. The messages can be as simple as a single character or word sent as XML, or as complex as a stream of binary data. A few sample scenarios include:

* A secure service to process business transactions.
* A service that supplies current data to others, such as a traffic report or other monitoring service.
* A chat service that allows two people to communicate or exchange data in real time.
* A dashboard application that polls one or more services for data and presents it in a logical presentation.
* Exposing a workflow implemented using Windows Workflow Foundation as a WCF service.
* A Silverlight application to poll a service for the latest data feeds.

Third-party packages:

- Password strength checking MD5.

To develop the User-Interface we have used HTML,CSS.

**4. DATA DESIGN**

4.1 Data Description

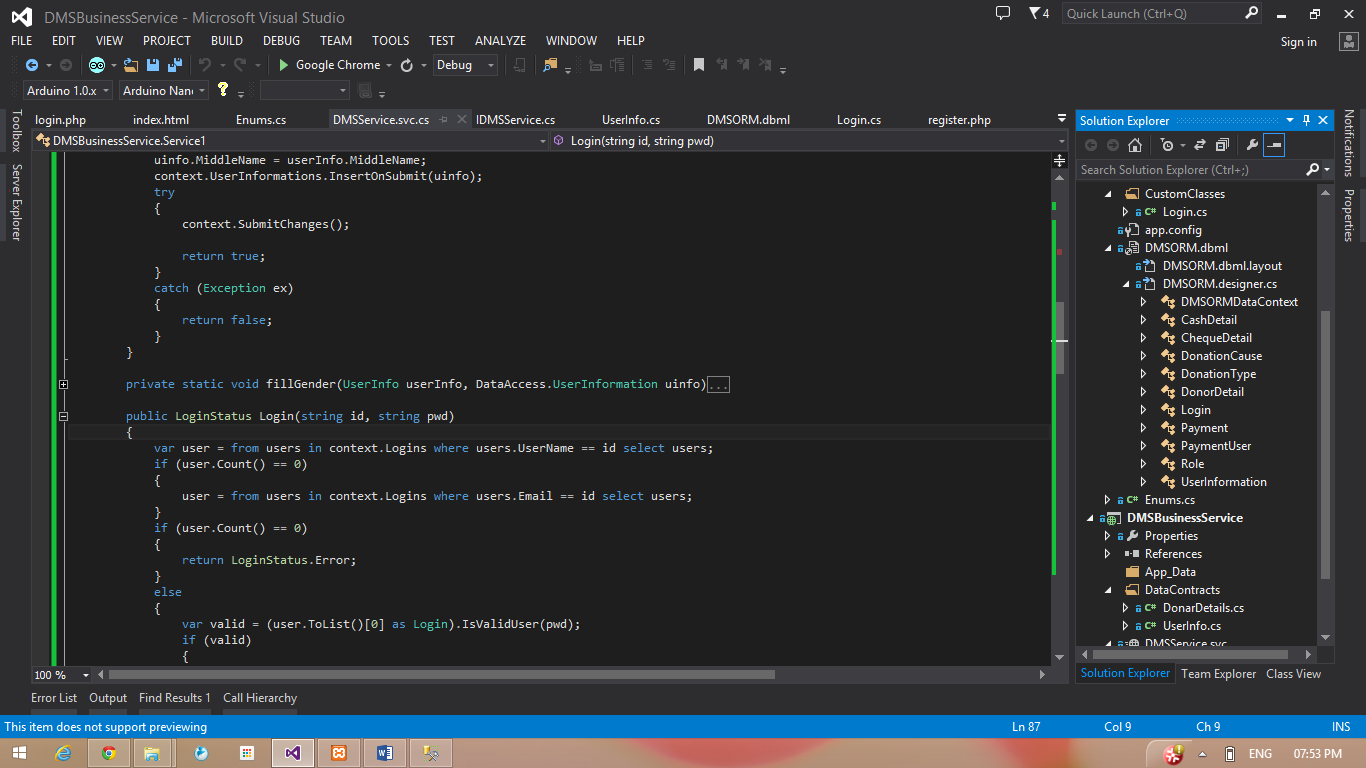
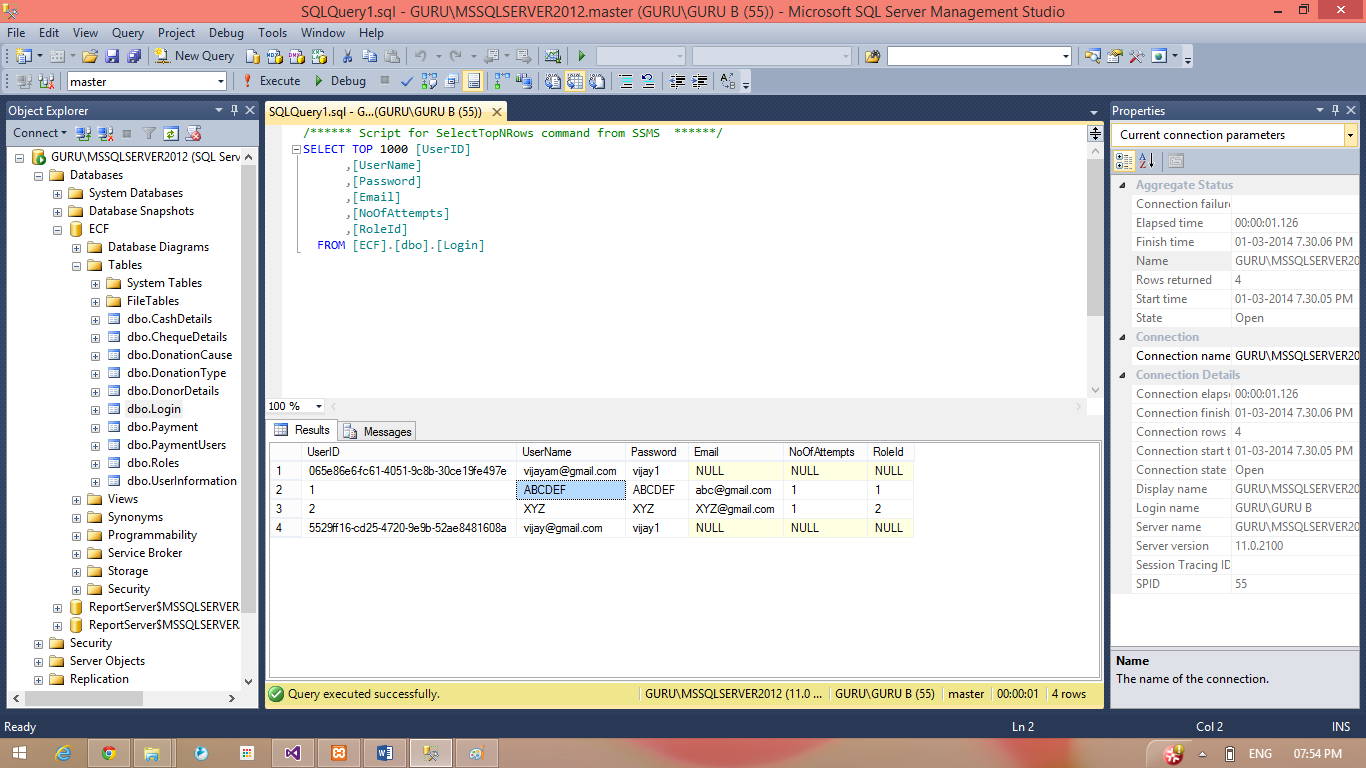
Donor management system uses the database called SQL Server .

**5. Human Interface Design**

5.1 User Interface Design







All users of Donor management system (Donor, Admin, Employee) will access the DMS user interface over a web browser. DMS will provide several different web page views for users to authenticate, create accounts,Donate.

**6. Requirement Matrix**

|  |  |
| --- | --- |
| Functions | FID |
| Login | 1 |
| Register | 2 |

|  |  |
| --- | --- |
| Detail | Description |
| FID | Function ID |
| Functional Requirements Table | Table No 1 |
| Non-Funtional Requirements | Table No 2 |

TABLE NUMBER 1:

|  |  |
| --- | --- |
| FID | Components |
| 1 | PHP |
|  | CSS |
| 2 | WCF |
|  | SQL SERVER |
|  | CSS/Web api |

**7. APPENDICES**

|  |  |
| --- | --- |
| NAME | DESCRIPTION |
| API | It is a protocol intended to be used as an interface by software components to communicate with each other. |
| GUI | It is a type of user interface that allows users to interact with electronic devices using images rather than text commands. |
| Database engine | It is the underlying software component that a database management system uses to create, read, update and delete data from a database. |
| HTTP | It is designed to enable communications between clients and servers. |
| Github | It is a web-based hosting service for software development projects. |