Parking Enterprise Management System

Architecture Overview





Unified Development, Inc. & The Net Impact

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH DUNCAN SOLUTIONS, INC. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN DUNCAN’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, DUNCAN ASSUMES NO LIABILITY WHATSOEVER, AND DUNCAN DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Duncan products are not intended for use in medical, lifesaving, or life sustaining applications.

Duncan may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Duncan reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Duncan sales office or your distributor to obtain the latest specifications and before placing your product order.

RxV, MX and the Duncan logo are trademarks or registered trademarks of Duncan Solutions or its subsidiaries in the United States and other countries.

\*Other names and brands may be claimed as the property of others.

Copyright © 2008, Duncan Solutions, Inc. All rights reserved.

Table of Contents

1 General 9

1.1 Project general description 9

1.2 Solution general description 9

1.3 Guidelines from the functional design (Optional) 9

1.4 Development guidelines (Optional) 9

1.4.1 Best Practices. 9

1.4.2 Standards. 9

1.5 Risks & Assumptions 9

2 Architecture 9

2.1 Physical architecture 9

2.1.1 Diagram 9

2.2 Logical architecture 9

2.2.1 Application server architecture 9

2.2.2 Services server architecture 9

2.2.3 Web architecture 9

3 3. Database (Duncan + UniDev) 9

3.1 General description 9

3.2 Tables list 9

3.3 Database diagram 9

3.4 Views 9

3.5 Stored procedures 9

3.6 Triggers 9

3.7 Jobs 9

4 Infrastructure (Duncan + UniDev) 9

4.1 General 9

4.2 Logging handling (Log4Dot Net) 9

4.3 Auditing and tracing handling 9

4.4 Exception handling 10

4.4.1 General 10

4.4.2 Custom Exceptions 10

4.5 Monitoring 10

4.5.1 Database monitoring for performance 10

4.6 Data Access 10

4.6.1 General 10

4.6.2 Typed Datasets 10

4.7 Common 10

4.7.1 Common Enums 10

4.7.2 Common Objects 10

4.7.3 Common functions 10

4.8 Multi-Language Support 10

4.9 Printing Mechanism 10

4.10 Cache management 10

4.10.1 Web site cache 10

4.10.2 Application cache 10

4.10.3 Database cache 10

4.11 Properties files based Configuration 10

4.11.1 Global configuration 10

4.11.2 Web site configuration 10

4.12 Unit Testing 10

4.12.1 Method 10

4.12.2 Tested Components 10

4.13 Transactions 10

4.14 Validations 10

4.14.1 Application validations 10

4.14.2 Web site validations 10

5 Business entities 10

5.1 General 10

5.2 Entities diagram 10

5.3 Entities description 11

6 Security 11

6.1 General 11

6.1.1 Web site security 11

6.1.2 Server security 11

6.2 Authentication 11

6.3 Authorization 11

6.4 Cross-site scripting (XSS) 11

6.5 SQL injection 11

6.6 Script injection 11

6.7 User input validation 11

6.8 Cryptography 11

7 Performance and Load Testing 11

7.1 General 11

7.2 Performance 11

7.3 Load Testing 11

8 Interfaces 11

8.1 Interfaces diagram 11

9 Data conversions 11

9.1 General 11

9.2 Mappings 11

10 External Services 11

10.1 General 11

10.2 Windows services 11

10.3 Queue Management 11

11 Class Design 11

11.1 General Structure 11

11.2 State management 11

11.2.1 Cookies 12

11.2.2 Application State 12

11.2.3 Session State 12

11.3 Master pages and themes 12

11.4 Pages 12

11.5 User controls 12

11.6 Custom controls 12

11.7 Resources 12

11.8 Common functions 12

11.9 Error Pages 12

11.10 UI graphic guidelines 12

11.11 Class diagrams 12

11.12 Description 12

11.12.1 Features 12

11.12.2 Web parts 12

11.12.3 Templates 12

11.12.4 Event handlers 12

11.12.5 Http handlers. 12

11.12.6 Lists 12

12 Application Server 12

12.1 General 12

12.2 Server gateway 12

12.2.1 General 12

12.3 Business Logic Layer 12

12.3.1 General 12

12.3.2 <BL 1> 12

12.4 Data Access Layer 12

12.4.1 General 12

12.4.2 <DAL 1> 12

13 3rd party dependences 12

13.1 General 12

13.2 Dependences diagram 13

****Revision**** History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Author | Description | Revision Date |
| 1.0 |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

****Document Conventions and Notes****

1. URL used in examples in this document will use host of **dev.pemsportal.com** unless otherwise indicated.
2. References to classes, methods and properties are given with a fully-qualified namespace and shown with a different font and color. This should ensure that the reader of this document will be able to find the referenced code in the project source. For example, Duncan.PEMS.Security.AuthorizationManager, references the class ‘AuthorizationManage’ in the namespace ‘Duncan.PEMS.Security’.

Introduction

# General

## Project general description

Parking Enterprise Management System (PEMS) is a system to manage client parking resources, including meters, sensors, gateways, cashboxes and spaces. The system is web-based.

## Solution general description

## Guidelines from the functional design (Optional)

## Development guidelines

### Best Practices.

Convention-over-Configuration (CoC) plays a pivotal role in the PEMS system development and operation. It is where coding conventions such as naming drive development and system functionality and process operates upon a pre-defined convention verses configuration. PEMS takes advantage of this methodology to build the system. Throughout this document CoC concepts are applied to PEMS, from authorization to routing to multi-tenancy.

See the Wikipedia article [Convention over configuration](http://en.wikipedia.org/wiki/Convention_over_configuration). There are many resources that discus CoC as applied to MVC 4. For instance [this site](http://www.simple-talk.com/dotnet/asp.net/asp.net-mvc-controllers-and-conventions/) provides insight into how CoC is a central tenant of MVC 4.

Recommend using NuGet (<http://visualstudiogallery.msdn.microsoft.com/27077b70-9dad-4c64-adcf-c7cf6bc9970c>) to manage third-party packages in Visual Studio.

### Standards and Software

The following operating system, DBMS, development environments, and software are used in the development and deployment of PEMS.

Servers

Microsoft Windows Server 2008 R2

Microsoft IIS 7.5

DBMS

Microsoft Sql Server 2008 R2 or better

Development Environment and Languages

Microsoft Visual Studio 2012 (<http://msdn.microsoft.com/en-us/library/dd831853%28v=vs.110%29.aspx>)

Razor Syntax ([tutorial](http://www.asp.net/web-pages/tutorials/basics/2-introduction-to-asp-net-web-programming-using-the-razor-syntax))

C#

Transact-SQL (<http://msdn.microsoft.com/en-us/library/bb510741%28SQL.100%29.aspx>)

Software, frameworks, and packages

Microsoft .NET 4.5 (<http://www.microsoft.com/en-us/download/details.aspx?id=30653>)

ASP.NET MVC 4.0 (<http://www.asp.net/mvc/mvc4>)

Entity Framework 5.0 (<http://www.nuget.org/packages/EntityFramework/5.0.0>)

NetSqlAzMan 3.6.0.15 (<http://netsqlazman.codeplex.com/>)

Other 3rd-Party packages - For up-to-date packages and versions, see the ‘Assemblies’ directory in TFS PEMS Project repository.

## Risks & Assumptions

# Architecture

The architecture model selected is based upon best practices of MVC 4 to implement a multi-tenancy system that is scalable, maintainable and customizable while following the tenants of MVC 4. A key factor in the selection of the architecture is a development model that utilizes the MVC 4 tools available in Visual Studio 2012 to reduce the work-load on the developers and simplify the development model. This will aide in the reduction of code complexity and improve maintainability.

The following constraints and goals drive the architecture.

* The system implementation may be distributed for scalability.
* The architectural model stresses strong interface contracts with separation/isolation of functionality into components that encapsulate operational capabilities and provide a high degree of re-use.
* System component model shall allow for any component to be replaced by like functionality or be deconstructed to sub-components as development unfolds.
* Authentication and authorization will be implemented with a common framework pattern that is transparent to the page implementation pattern.
* NetSqlAzMan provide authorization model and implementation for page and menu access.
* Authentication will be based upon .NET/MVC best-practices.
* Authorization control is at page level.
* Multi-tenancy will be implemented via the MVC Routing pattern. The URL model shall be flexible within the constraints of MVC routing model.
* System components to be independently maintainable and unit-testable.
* Capability to distribute functionality among multiple hosts.

## Physical architecture

### Diagram

## Logical architecture

### Application server architecture

### Services server architecture

### Web architecture

# 3. Database (Duncan + UniDev)

## RBAC (Roles-Based Access Control) Database

### General description

### Tables list

### Database diagram

### Views

### Stored procedures

### Triggers

### Jobs

## PEMS (RipNet) Database

### General description

### Tables list

### Database diagram

### Views

### Stored procedures

### Triggers

### Jobs

## Reporting Database

### General description

### Tables list

### Database diagram

### Views

### Stored procedures

### Triggers

### Jobs

## Maintenance Database

### General description

### Tables list

### Database diagram

### Views

### Stored procedures

### Triggers

### Jobs

# Infrastructure (Duncan + UniDev)

## General

## Logging handling (Log4Dot Net)

## Auditing and tracing handling

## Exception handling

### General

### Custom Exceptions

## Monitoring

### Database monitoring for performance

## Data Access

### General

### Typed Datasets

## Common

### Common Enums

### Common Objects

### Common functions

## Multi-Language Support

## Printing Mechanism

## Cache management

### Web site cache

### Application cache

### Database cache

## Properties files based Configuration

### Global configuration

### Web site configuration

## Unit Testing

### Method

### Tested Components

## Transactions

## Validations

### Application validations

### Web site validations

# Business entities

## General

## Entities diagram

## Entities description

# Security

## General

### Web site security

### Server security

## Authentication

## Authorization

## Cross-site scripting (XSS)

## SQL injection

## Script injection

## User input validation

## Cryptography

# Performance and Load Testing

## General

## Performance

## Load Testing

# Interfaces

## Interfaces diagram

# Data conversions

## General

## Mappings

# External Services

## General

## Windows services

## Queue Management

# Class Design

## General Structure

## State management

### Cookies

### Application State

### Session State

## Master pages and themes

## Pages

## User controls

## Custom controls

## Resources

## Common functions

## Error Pages

## UI graphic guidelines

## Class diagrams

## Description

### Features

### Web parts

### Templates

### Event handlers

### Http handlers.

### Lists

# Application Server

## General

## Server gateway

### General

## Business Logic Layer

### General

### <BL 1>

## Data Access Layer

### General

### <DAL 1>

# 3rd party dependences

## General

## Dependences diagram

