|  |  |
| --- | --- |
|  | **Cognizant Academy**  **Employee Management System**  **ASP.Net MVC, Entity Framework, SQL Server**    **Integrated Capability Test**  **Version 1.0** |
| |  |  |  |  | | --- | --- | --- | --- | |  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** | | **Name** | T.Narasimha Rao |  |  | | **Role** |  |  |  | | **Signature** |  |  |  | | **Date** |  |  |  | |
|  |

Table of Contents

[1.0 Introduction 3](#_Toc23440118)

[1.1 Purpose of this document 3](#_Toc23440119)

[1.2 Definitions & Acronyms 3](#_Toc23440120)

[1.3 Project Overview 3](#_Toc23440121)

[1.4 Scope 3](#_Toc23440122)

[1.5 Target Audience 3](#_Toc23440123)

[1.6 Hardware and Software Requirement 3](#_Toc23440124)

[1.6.1 Hardware Requirements 3](#_Toc23440125)

[1.6.2 Software Requirements 4](#_Toc23440126)

[2.0 Functional Requirements 4](#_Toc23440127)

[2.1 Functional Requirements 4](#_Toc23440128)

[2.2 Use case Diagram 4](#_Toc23440129)

[2.3 System Architecture Diagram 4](#_Toc23440130)

[3.0 Design Specification 5](#_Toc23440131)

[3.1 Pre-requisite Steps 5](#_Toc23440132)

[3.2 UI Design 5](#_Toc23440133)

[3.3 General Design Constraints 6](#_Toc23440136)

[4.0 Submission 6](#_Toc23440138)

[4.1 Code submission instructions 6](#_Toc23440139)

[5.0](#_Toc23440140) Application Screens and UI Guide lines 8

# Introduction

## Purpose of this document

The purpose of the Document is to elaborate the functional requirements and technical design specification of the problem statement to be solved.

Employee Management System is to automate or computerize all employee management operations.

## Definitions & Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
| EMS | Employee Management System |
|  |  |
|  |  |
|  |  |

## Project Overview

Employee Management system is an application that enables users to create and store Employee details. This application is helpful to department of the organization which maintains data of employees related to an organization.

## Scope

Admin have all privileges to access all the information of EMS. So that admins will insert, list and remove the employees. This case study is targeted to Web based applications using ASP.NET MVC and Entity Framework. It includes limited number of operations due to targeted candidates (learners) and time constraint.

## Target Audience

Learner Level

## Hardware and Software Requirement

### Hardware Requirements

|  |  |  |
| --- | --- | --- |
| # | Item | Specification/Version |
| 1 | Operating System | Window 7/10 |
| 2 | RAM | 4 GB RAM |
| 3 | HDD | 500 GB HDD |
| 4 | Processor | Dual Core Intel |

### Software Requirements

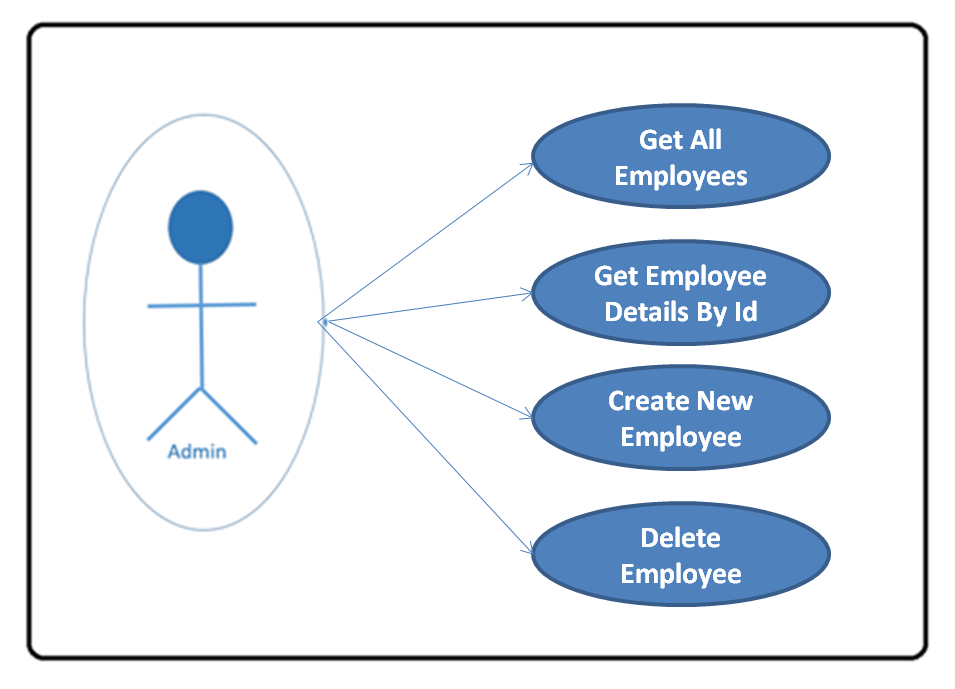
|  |  |  |
| --- | --- | --- |
| # | Item | Specification/Version |
| 1 | .NET Framework | 4.5 or above version |
| 2 | C# | 4.5 or above version |
| 3 | Visual Studio | 2017/2019 |
| 4 | SQL Server | 2014 or above version |
| 5 | Browser | Chrome or IE |

# Functional Requirements

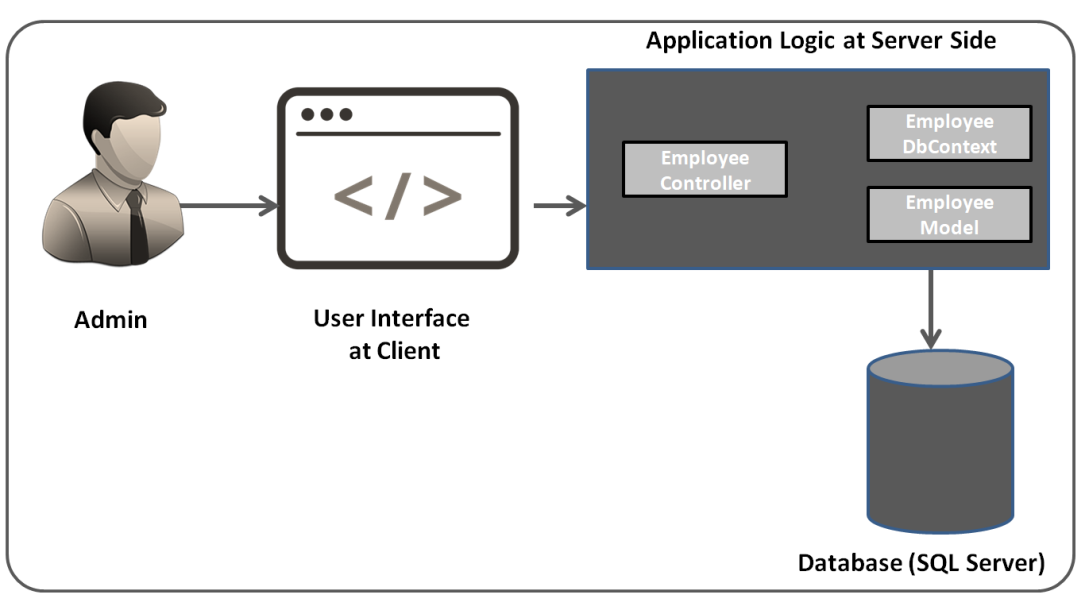
## Functional Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Req. # | Req. Name | Req. Description | Actors / Users | Comments |
| EMS\_1 | Add Emp | Admin can add new employee details | Admin | Store employee in database using EF |
| EMS\_2 | Delete Emp | Admin can delete the employee details based on the Empno | Admin | Delete employee details from database |
| EMS\_3 | List out All Emps | Display all employee details | Admin | Get All Employee data from database |
| EMS\_4 | Get Emp details by Empno | Get employee details based on the given Empno | Admin | Get required employee details from database |

## Use case Diagram



## System Architecture Diagram



# Design Specification

## Pre-requisite Steps

* C#.NET
* ASP.NET MVC
* Entity Framework
* HTML,CSS, JS

## Data Design

Structure of employee data (Model class)

|  |  |  |
| --- | --- | --- |
| S. No | Data Field | Description |
| 1. | Empno | Unique Id for every employee |
| 2. | Ename | Full name |
| 3. | Job | Designation of the employee |
| 4. | Salary | Salary of the employee |
| 5. | Deptno | Represent Department Number |

## 

## General Design Constraints

* 1. Prepare required controllers and action method as per the details mentioned in the following table.
  2. Design the view based on the output screens mention in the document.

|  |  |  |
| --- | --- | --- |
| Controller class | EmployeeControler | * Provides action methods to perform operations on Employee data * Create object for EmployeeManagementDbContext class with object name should be “context” |
| Action Methods | * Index() * Create() * Details() * Delete() | Provide required logic in the action methods |
| Model class | Employee | Define Employee class with require properties |
| DataContext Class | EmployeeManagementDbContext | * Data context should define under Models * Connection name should be “EmployeeConnection” define web.config. * DbSet<T> property name should “Employees” |
| View Pages | * Index.cshtml * Create.cshtml * Details.cshtml * Delete.cshtml |  |

# Submission

## Code submission instructions

Do not change the code skeleton given as your code will be auto evaluated.

You can validate your solution against sample test cases during the assessment duration.

Your last submitted solution will be considered for detailed evaluation.

Make sure to submit the solution before the specified time limit. You will not be allowed to submit the solution once the mentioned time for the assessment is over.

Action methods should have a meaningful name

Remove unused Action methods

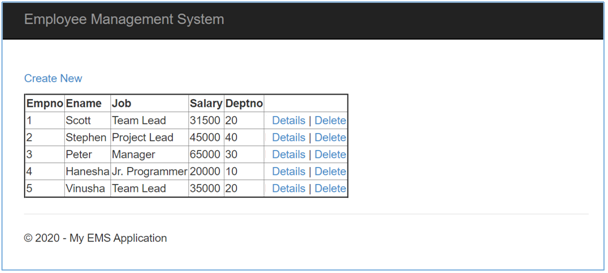
There should not be any hard coded values in code. It has to be referenced from Web.config file

Database connection string should be set in the ConnectionStrings section of Web.config and NOT in the AppSettings

Meaningful names should be given to the controls created in View

# Application Screens and UI Design Instructions

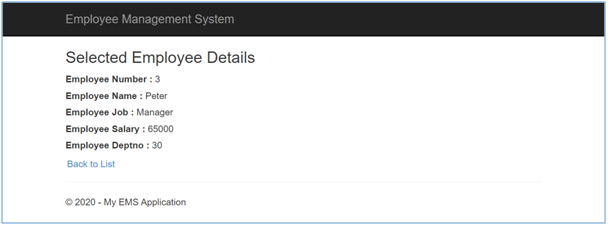
1. Display all employee details in table format with required hyperlinks to perform CRUD operations



**Instructions**

* Hyperlink Text should be “Employee Management System” which is in the Menu bar. It is common to all views. Prepare in Layout page.
* Title should be “Index - Employee Management Application”
* “Create New” Hyperlink required to navigate to create page
* Display data in table format as show in the above screen

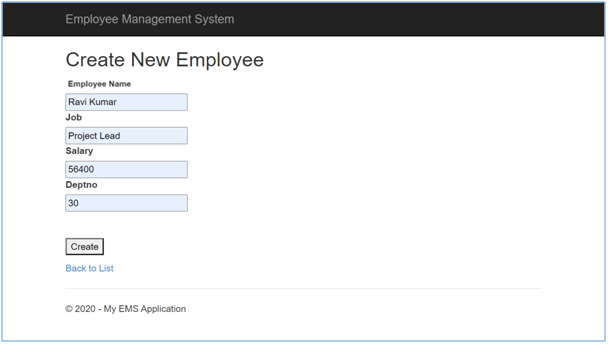
1. Display selected employee details based on the selected empno



**Instructions**

* Title should be “Details - Employee Management Application”
* “Back to List” Hyperlink required to navigate to home page
* Display data in proper format as show in the above screen

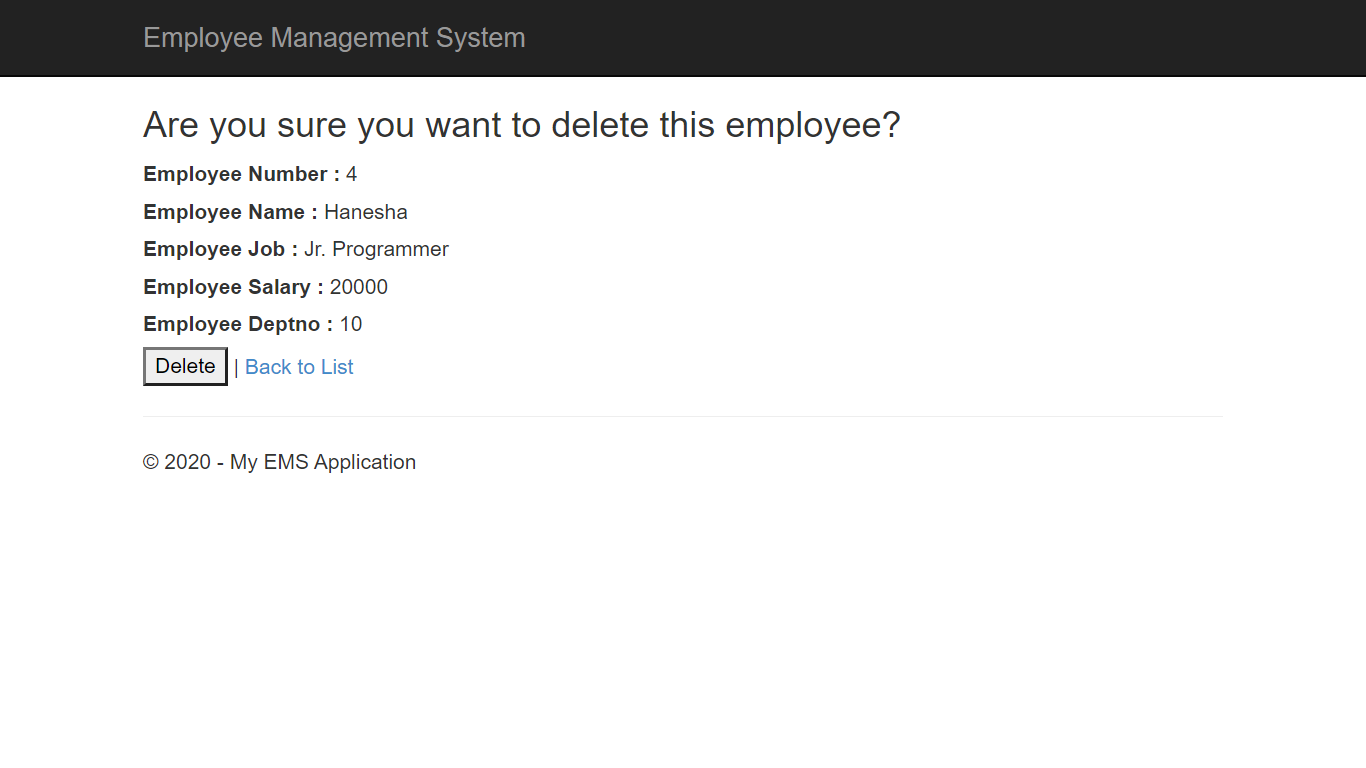
1. Create new employee details screen



**Instructions**

* Add submit button with value “Create” and id should be “createEmployee”
* All files is mandatory and proper validations.
* For Employee name filed, validation error message should be “Please Provide Employee Name”
* Use data annotations on Model class to provide validations
* Apply “text-danger” class to apply styles to error messages

1. Delete the selected employee details screen



**Instructions**

* Title should be “Delete - Employee Management Application”
* Create h3 tag and provide confirmation message “Are you sure you want to delete this employee?”
* Create “Back to List” hyperlink
* Create submit button with the value “Delete”