# 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.

## Definitions & Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
|  |  |
|  |  |
|  |  |
|  |  |

## Project Overview

* Building a Model using ADO.NET , WEB API

## FUNCTIONALITIES:

1. XYZ COMPANY ‘S StudentS LIST TO BE DISPLAYED WITH ADO.NET ,WEB API , MS SQL SERVER
2. To Create a simple application for XYZ Student House.
   * Allows to add 2 Students Information
   * Displays existing Student Information
   * Display a Student details by movie ID
   * Delete a Student By ID
   * Update Existing Student name by Student ID

## Scope

* C#.NET,WEB API
* ADO.NET
* SQL SERVER.

## Target Audience

Beginners

## Hardware and Software Requirement

### Hardware Requirements

|  |  |  |
| --- | --- | --- |
| # | Item | Specification/Version |
|  | Processor | Intel® coreTM i5 or higher |
|  | RAM | 8 GB |
|  | HARD DISK | OS partitation 200GB |

### Software Requirements

|  |  |  |
| --- | --- | --- |
| # | Item | Specification/Version |
|  | Operating System | Windows 10 |
|  | Application Software | Visual Studio2019 or higher |
|  | Database Server | MS Sql Server 2014 Expression Edition |

# Functional Requirements

## Functional Requirements

*<List the high level use cases or behaviors and actors of the system. This requirement # and name must be unique and must be referred elsewhere in the document. >*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Req. # | Req. Name | Req. Description | Actors / Users | Comments |
|  |  |  |  |  |

System Architecture Diagram

* Understanding Requirements and Database Design.
* Creating Solution and Adding projects to It.
* Creating The business Objects.
* Creating Data Access Layer in
* Creating Business Logic Layer in C#.Net.
* Creating Presentation Logic Layer in PostMan
* Designing Models,BUSINESS LOGIC
* Business Rules Validations.
* Securing Your App.

# Design Specification

## Pre-requisite Steps

A Web API APPLICATION ALLOWS A USER TO MANAGE STudents I OR DISPLAY STUDENTS DETAILS,CRUD Operations

## POSTMAN --UI

STEPS AS FOLLOWS:

* Create New Web Api  : MoviesRepoController.cs
* Add 2 classes(Student.cs,StudentRepository.cs) to Models folder in Solution Explorer
* Add 1 interface(IStudent.cs)
* Configure Ado.NET Tech in StudentRepository.cs by implementing methods of IStudent.cs
* StudentsRepoController is connected to StudentRepository.cs
* POSTMAN (UI)

Web.CONFIG:

</entityFramework>

  <connectionStrings>

    <add name="SampleDB" connectionString="data source=DESKTOP-A8KSJON;initial catalog=COllegeDB;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework" providerName="System.Data.SqlClient" />

  </connectionStrings>

</configuration>

PROJECT-STRUCTURE

SCENARIO-1

|  |  |
| --- | --- |
| Type | Name |
| Class | CollegeDB |
| Class | Student |
| Class | StudentRepository |
| interface | IStudent |
| ApiController | STudentsRepoController |
|  |  |
|  |  |

CLASS METHOD DESCRIPTION:

SCENARIO-2

|  |  |  |
| --- | --- | --- |
| Class | Method-name/Properites | Description |
| Student.cs | StudentId | Int |
|  | StudentName | String |
|  | DateOfBirth | datetime |
|  | Height | decimal |
|  | weight | Float |

|  |  |  |
| --- | --- | --- |
| Class/interface | Method/properties | Description |
| IMovie.cs | List<Movie>  GetStudentList(); | Returns all existing movie details |
|  | void newStudent(Student); | a)Takes input for all properies defined in Movie class of  B)to add  2 movies information. |
|  | STudent GetStudentById(int id); | Takes movie id as parameter returns movie record. |
|  | Void UpdateStudentById(int id); | To update movie details |
|  | Void DeleteStudentById(int id) | To delete movie |

**Project Overview**

LearnToday is an online training application which allows students to view all the courses and allows them to enroll for the courses. It also features trainer registration and signin actions.

**Table Specification**

**The sample Database and Table will be given for your reference:**

Database Name: **CollegeDB**

 Table: **Student**

**ER Diagram**

https://c1h-word-edit-15.cdn.office.net:443/we/s/hA3596C17DAD9A003_resources/1033/progress.gif

**Design Considerations**

* Create the Database and the set of tables as per the table specifications.
* Create the model classes as per the requirement.
* Use ADO.NET tech
* Business logic should be implemented in the Service classes.
* Use A DO.NET layer to interact with the Database.

**IStudent.cs**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Method Type** | **Method Name** | **Parameter** | **Return Type** | **Description** |
| Method: GET | getAllStudents() | No Parameter | IEnumerable<Student> | This method will return all the students. |
| Method: GET | getStudentById | int id | Student | Here the student will be searched by studentid. If searched student is found it will return HttpStatus.OK and the searched student data. Otherwise it will return HttpStatus.NOT\_FOUND with a message "Searched Data not Found". |

 Create a new StudentRepositoty with the below 3 action methods:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Method Type** | **Method Name** | **Parameter** | **Return Type** | **Description** |
| Method: GET | Method/Propertiess | No Parameter | IEnumerable<student> | This method will return all the students in ascending order . |
| Method: Post | IEnumerable<Student>  GetStudentList(){  // Ado.NET code here  } | Student model | void | This method is used to method to enroll student for the course. Input student information thru the request body. Use try-catch block. Return HttpStatus.CREATED with the model object if enrollment is successful; otherwise return HttpStatus.BAD\_REQUEST with exception object in catch block. |
| Method:@DELETE | void AddStudent(Student)  {  // Ado.NET code here  } | Student | void | This method is used to remove the student information by the enrollment id Use a try catch block. If deletion successful then return HttpStatus.OK. If enrollement information is not found by searched id then return HttpStatus.NOT\_FOUND with a message "No enrollement information found". Under catch block return HttpStatus.BAD\_REQUEST with exception object if some exceptions occur in the program. |
|  | Student GetMovieById(int id)  {  //ADO.NET code here;  } | Int id | Student |  |
|  | Void UpdateMovieById(int id); | Int id | Void |  |
|  | Void DeleteMovieById(int id) | Int id | void |  |

MovieRepository .cs -------Component names:

|  |  |
| --- | --- |
| Type | Name |
| Class | StudentRepository.cs |
| Class | Student.cs |
| Configuration file | Web.config |
| interface | IStudent.cs |
| ApiController | StudentsRepoController |

* **FOLLOWING STEPS WILL EXPLAIN WHAT EACH CLASS IS WILL DO FOR PROJECT**

|  |  |
| --- | --- |
| *Component Name* | StudentRepository .cs |
| *Component Definition* | *ADO.NET Code* |
| *Type* | *Class :ISTudent { }* |
| *MethodName* | public List<Student> GetStudentList()  { |
| *Access modifers* | *public* |
| *Return type* | *List<Student>* |
| *Interactions* | *IStudent.cs* |
| *Resources* | *Web.config* |

|  |  |
| --- | --- |
| *Component Name* | *Student.cs* |
| *Component Definition* |  |
| *Type* | *Class* |
| *String names* |  |
| *Access modifers* | *public partial* |
| *Properties* | *int StudentId*  *string STudentName*  *datetime DateOfBirth*  *decimal Height*  *float Weight* |
| *Interactions* |  |
| *Resources* | *---* |

STEP4:

|  |  |
| --- | --- |
| *Component Name* | STudentsRepoController *.cs* |
| *Component Definition* | *Execution* |
| *Type* | *ApiController* |
| *MethodName* | *Get(),getById(int),post(),put(),delete()* |
| *Access modifers* | *public* |
| *Return type* | HttpResponseMessage (Post,GetById,put(),Delete() |
| *Interactions* | *STudentRepository.cs* |
| *Resources* | *----------------------* |

## General Design Constraints

## Requirements to Design Traceability

# Submission

## Code submission instructions

* **\*Note:  
  Don't change the given code Skeleton.  
  Do not include new namespaces.**

## Data Design

Database Scenario:

**Below is POSTMAN screenshot for getAllStudent action:**

https://c1h-word-edit-15.cdn.office.net:443/we/s/hA3596C17DAD9A003_resources/1033/progress.gif

**Below is POSTMAN screenshot for postStudent action:**

**Below is POSTMAN screenshot for deleteStudent action:**

**Below is POSTMAN screenshot for deleteStudent action if searched    enrolment id is not found:**