# System Requirements Specification

## Index

**For**

# Attendance Management Application

**Version 1.0**

# ATTENDANCE MANAGEMENT APPLICATION

## System Requirements Specification

## 1 PROJECT ABSTRACT

The **Attendance Management Application** is a Entity Framework 4.8 with MS SQL Server database connectivity. It enables users to manage various aspects of Attendance management and organization.

**Following is the requirement specifications**:

| | Attendance Management Application |
|---|---|
| | |
| Modules | |
| 1 | Attendance |
| | |
| Attendance Module Functionalities | |
| | |
| 1 | Create an Attendance |
| 2 | Update the existing Attendance details |
| 3 | Get the Attendance by Id |
| 4 | Get all Attendances |
| 5 | Delete an Attendance |

# 2 Assumptions, Dependencies, Risks / Constraints

## 2.1 Attendance Constraints

- When fetching an Attendance by ID, if the Attendance ID does not exist, the operation should throw a custom exception.
- When updating an Attendance, if the Attendance ID does not exist, the operation should throw a custom exception.
- When removing an Attendance, if the Attendance ID does not exist, the operation should throw a custom exception.

## Common Constraints

- For all rest endpoints receiving @RequestBody, validation check must be done and must throw custom exception if data is invalid
- All the database operations must be implemented on entity object only
- Do not change, add, remove any existing methods in service layer
- In Repository interfaces, custom methods can be added as per requirements.
- All RestEndpoint methods and Exception Handlers must return data wrapped in **ResponseEntity**

# 3 Business Validations

- Id (Int) Key, Not Null
- Player Name are marked as [Required] to ensure they are not null,
- Duration (Attendance) is not null.
- Attendance int not null
- Game Type string not null
- Notes not null
- Date (DateAttendance) of the Attendance not null.

# 4 REST ENDPOINTS

Rest End-points to be exposed in the controller along with method details for the same to be created

## 4.1 AttendanceCONTROLLER

<table>
<tr><th>URL<br>Exposed</th><th>Purpose</th></tr>
<tr><td>1. /api/Attendance/GetAllAttendances<table><tr><td>Http Method</td><td>GET</td></tr><tr><td>Parameter</td><td>-</td></tr><tr><td>Return</td><td><IEnumerable<Attendance>></td></tr></table></td><td>Fetches all the Attendances</td></tr>
<tr><td>2. api/Attendance/CreateAttendance<table><tr><td>Http Method</td><td>POST</td></tr><tr><td>Parameter 1</td><td>Attendance</td></tr><tr><td>Return</td><td>Attendance</td></tr></table></td><td>Add a new Attendance</td></tr>
<tr><td>3. /api/Attendance/DeleteAttendance<table><tr><td>Http Method</td><td>DELETE</td></tr><tr><td>Parameter 1</td><td>Int (id)</td></tr><tr><td>Return</td><td>-</td></tr></table></td><td>Delete Attendance with given Attendance id</td></tr>
<tr><td>4./ api/Attendance/GetAttendanceById<table><tr><td>Http Method</td><td>GET</td></tr><tr><td>Parameter 1</td><td>Int (id)</td></tr><tr><td>Return</td><td>Attendance</td></tr></table></td><td>Fetches the Attendance with the given id</td></tr>
<tr><td>5. /api/Attendance/UpdateAttendance<table><tr><td>Http Method</td><td>PUT</td></tr><tr><td>Parameter 1</td><td>Int (id)</td></tr><tr><td>Parameter 2</td><td>Attendance</td></tr><tr><td>Return</td><td>Attendance</td></tr></table></td><td>Updates existing Attendance</td></tr>
</table>

# 5. TEMPLATE CODE STRUCTURE

## 5.1 Package: AttendanceManagementApp

**Resources**

| Names | Resource | Remarks | Status |
|---|---|---|---|
| Package Structure | | | |
| controller | Attendance Controller | Controller class to expose all rest-endpoints for auction related activities. | Partially implemented |
| Web.Config | Web.Config file | Contain all Services settings and SQL server Configuration. | Already Implemented |
| Interface | IAttendanceService, interface | Inside all these interface files contains all business validation logic functions. | Already Implemented |
| Service | AttendanceService CS file<br>file | Using this all class we are calling the Repository method and use it in the program and on the controller. | Partially Implemented |
| Repository | IAttendanceRepository<br><br>AttendanceRepository CS file and interface. | All these interfaces and class files contain all CRUD operation code for the database.<br>Need to provide implementation for service related functionalities | Partially Implemented |
| Models | Attendance cs file | All Entities/Domain attribute are used for pass the data in controller. | Already Implementation |

## 5.2 Package: AttendanceManagementApp.Tests

**Resources**

**The AttendanceManagementApp.Tests project contains all test case classes and functions for code evaluation. Don't edit or change anything inside this project.**

## 6. EXECUTION STEPS TO FOLLOW

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**

2. **To open the command terminal the test takers need to go to the Application menu (Three horizontal lines at left top) Terminal → New Terminal.**

3. **On command prompt, cd into your project folder (cd <Your-Project-folder>).**

4. **To connect SQL server from terminal:**
   **(AttendanceManagementApp /sqlcmd -S localhost -U sa -P pass@word1)**
   - **To create database from terminal -**

     **1> Create Database AttendanceDb**
     **2> Go**

5. **Steps to Apply Migration(Code first approach):**
   - **Press Ctrl+C to get back to command prompt**
   - **Run following command to apply migration-**
     **(AttendanceManagementApp /dotnet-ef database update)**

6. **To check whether migrations are applied from terminal:**
   **(AttendanceManagementApp /sqlcmd -S localhost -U sa -P pass@word1)**

   **1> Use AttendanceDb**
   **2> Go**
   **1> Select * From __EFMigrationsHistory**
   **2> Go**

7. **To build your project use command:**
   **(AttendanceManagementApp /dotnet build)**

8. **To launch your application, Run the following command to run the application:**
   **(AttendanceManagementApp /dotnet run)**

9. **This editor Auto Saves the code.**

10. **To test any Restful application, the last option on the left panel of IDE, you can find ThunderClient, which is the lightweight equivalent of POSTMAN.**

11. **To test web-based applications on a browser, use the internal browser in the workspace. Click**

**on the second last option on the left panel of IDE, you can find Browser Preview, where you can launch the application.**

**Note: The application will not run in the local browser**

12. **To run the test cases in CMD, Run the following command to test the application: (AttendanceManagementApp.Tests/dotnet test --logger "console;verbosity=detailed") (You can run this command multiple Attendances to identify the test case status,and refactor code to make maximum test cases passed before final submission)**

13. **If you want to exit(logout) and continue the coding later anyAttendance (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B - command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**

14. **These are Attendance bound assessments the Attendancer would stop if you logout and while logging in back using the same credentials the Attendancer would resume from the same Attendance it was stopped from the previous logout.**

15. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**