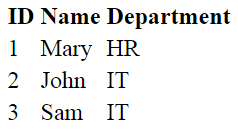
**List view in asp.net core mvc**

**Example**   
  
Let's understand this with an example. We want to retrieve all the employees and display them on a web page as shown below. At the moment, from styling perspective the page does not look that good. In our upcoming sessions we will install Bootstrap and style the page to make it look good.   
  
  
   
  
**IEmployeeRepository - Code Change**  
  
Modify IEmployeeRepository interface to include GetAllEmployees() method. As you can see this method returns the list of all employees.

public interface IEmployeeRepository  
{  
    Employee GetEmployee(int Id);  
    IEnumerable<Employee> GetAllEmployees();  
}

**MockEmployeeRepository - Code Change**  
  
At the moment, in our application we have only one class (MockEmployeeRepository) that implements IEmployeeRepository interface. So modify, MockEmployeeRepository class to provide implementation for GetAllEmployees() method as shown below. Notice, GetAllEmployees() returns the hard-coded list of employees we have in the private field \_employeeList.  

we will provide another implementation for IEmployeeRepository interface. This new implementation will retrieve employees from a SQL server database.

public class MockEmployeeRepository : IEmployeeRepository

{

    private List<Employee> \_employeeList;

    public MockEmployeeRepository()

    {

        \_employeeList = new List<Employee>()

    {

        new Employee() { Id = 1, Name = "Mary", Department = "HR", Email = "mary@pragimtech.com" },

        new Employee() { Id = 2, Name = "John", Department = "IT", Email = "john@pragimtech.com" },

        new Employee() { Id = 3, Name = "Sam", Department = "IT", Email = "sam@pragimtech.com" },

    };

    }

    public IEnumerable<Employee> GetAllEmployees()

    {

        return \_employeeList;

    }

    public Employee GetEmployee(int Id)

    {

        return this.\_employeeList.FirstOrDefault(e => e.Id == Id);

    }

}

**HomeController - Code Change**   
  
Modify, Index() action method in the HomeController as shown below. Notice, we retrieve the list of employees by calling GetAllEmployees() method and pass that list to the View.

public class HomeController : Controller

{

    private IEmployeeRepository \_employeeRepository;

    public HomeController(IEmployeeRepository employeeRepository)

    {

        \_employeeRepository = employeeRepository;

    }

    public ViewResult Index()

    {

        // retrieve all the employees

        var model = \_employeeRepository.GetAllEmployees();

        // Pass the list of employees to the view

        return View(model);

    }

    public ViewResult Details()

    {

        HomeDetailsViewModel homeDetailsViewModel = new HomeDetailsViewModel()

        {

            Employee = \_employeeRepository.GetEmployee(1),

            PageTitle = "Employee Details"

        };

        return View(homeDetailsViewModel);

    }

}

**Index.cshtml - Code Change**

* Set IEnumerable<EmployeeManagement.Models.Employee> as the model for the view using @model directive
* Use the foreach loop to loop over each employee in the list of employees and dynamically generate a table row and table cells to display ID, Name and Department property values.

@model IEnumerable<EmployeeManagement.Models.Employee>

<html>

<head>

    <title></title>

</head>

<body>

    <table>

        <thead>

            <tr>

                <th>ID</th>

                <th>Name</th>

                <th>Department</th>

            </tr>

        </thead>

        <tbody>

            @foreach (var employee in Model)

            {

                <tr>

                    <td>

                        @employee.Id

                    </td>

                    <td>

                        @employee.Name

                    </td>

                    <td>

                        @employee.Department

                    </td>

                </tr>

            }

        </tbody>

    </table>

</body>

</html>