

Employee Management System

An Industrial Project Report Submitted in Partial Fulfillment of the Requirements for the

Award of Degree

Of

Bachelor of Technology

In

Computer Science and Engineering

By

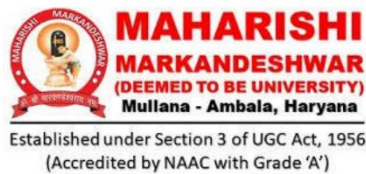
Pawan Kumar (11171109)

Under the supervision of

Arun Yadav

Product Engineering Software Architect Tech (Manager)

Arun.Yadav@crowe.com



M.M. Engineering College, Mullana, Ambala

Maharishi Markandeshwar (Deemed to be University), Mullana, Ambala, Haryana, India

May 2021

Candidate's Declaration

I hereby certify that the work which is being presented in the industrial project entitled “**Employee Management System**” in fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering of M.M Engineering College, Mullana, Ambala, Haryana, India is an authentic record of my work carried out during a period from Jan 2021 to May 2021, under the supervision of **Arun Yadav (Product Engineering Software Architect Tech (Manager))**. The matter presented in this project report has not been submitted by me for the award of any other degree of this or any other Institute/University.

Pawan Kumar (11171109)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date

Arun Yadav
Product Engineering Software Architect Tech (Manager)
Arun.Yadav@crowe.com

Acknowledgement

I wish to express my deep sense of indebtedness and sincerest gratitude to my guide **Arun Yadav (Product Engineering Software Architect Tech (Manager))**. For his invaluable guidance and constructive criticism throughout this work. He has displayed unique tolerance and understanding at every step of progress and encourages me. I deem it my privilege to have carried out my project work under his able guidance.

I would especially like to thank **Dr. Sandip Kumar Goel (Professor and Head, CSE Department, MMEC)** without whom, this work would not have been as it is now.

As a Final Personal Note, I am grateful to my parents, who are inspirational to me in their understanding, patience and constant encouragement.

Pawan Kumar (11171109)

		Supervisor's Performa	
Maharishi Markandeshwar Engineering College			
Computer Science & Engineering			
Progress Report of Project Work For B. Tech Students			
Semester: 8th		Section: A3	
1	Name of Student	Pawan Kumar	
2	Roll Number	11171109	
3	Project Title	Employee Management System	
4	Prior of the Report	1 st Assessment Date	10/04/2021
		2 nd Assessment Date	23/04/2021
		3 rd Assessment Date	10/05/2021
5	Name of Supervisor(s)		
6	Progress in Percentage(%)	30%	(1 st Assessment)
		50%	(2 nd Assessment)
		100%	(3 rd Assessment)
Supervisor Signature with Date (1 st Assessment)		Arun Yadan Product Engineering Software Architect Tech (Manager)	
Supervisor Signature with Date (2 nd Assessment)		Arun Yadan Product Engineering Software Architect Tech (Manager)	
Supervisor Signature with Date (3 rd Assessment)		Arun Yadan Product Engineering Software Architect Tech (Manager)	

Abstract

The Employee Management System Project is created using ASP.Net Web API. In this project we used RESTfull services to operate the data on the database. The Database we used in this project is Microsoft SQL Server. And Inorder to connect the database to our web application we used Entity Framework.

So, This Project contains many awesome technologies like ASP.Net Web API, Entity Framework, REST Services, JavaScript, JQuery, Ajax, HTML, CSS and BOOTSTRAP.

Web API as the name suggests, is an API over the web which can be accessed using HTTP protocol. It is a concept and not a technology. We can build Web API using different technologies such as Java, .NET etc. For example, Twitter's REST APIs provide programmatic access to read and write data using which we can integrate twitter's capabilities into our own application.

RESTful Web Services are basically REST Architecture based Web Services. In REST Architecture everything is a resource. RESTful web services are light weight, highly scalable and maintainable and are very commonly used to create APIs for web-based applications. This tutorial will teach you the basics of RESTful Web Services and contains chapters discussing all the basic components of RESTful Web Services with suitable examples.

TABLE OF CONTENTS

Candidate's Declaration.....	ii
Acknowledgement.....	iii
Supervisor Performa.....	iv
Abstract.....	v
List of Tables.....	vi
List of Figures.....	vii
Chapter 1. Introduction about Company.....	1-6
1.1 Company Profile.....	1
1.2 Technologies Used.....	1
1.3 Company Products.....	4
1.4 Company Clients.....	5
Chapter 2. Overview of Internship Training.....	7-16
2.1 Technology Learnt.....	7
2.2 Selection of Project work based upon technology learnt.....	14
2.3 Platform requirements.....	16
Chapter 3. Introduction & Objectives of Project Work.....	17
Chapter 4. Methodology Used.....	18-21
4.1 Proposed Algorithm or technique.....	18
4.2 Data flow Diagrams Used.....	20
Chapter 5. Results & Discussion.....	22-32
5.1 Screen shots obtained by running project with Description.....	22
Chapter 6. Future Scope.....	33
REFERENCES.....	34
APPENDIX A: Source Code.....	35-51

List of Tables

Table No.	Table Caption	Page No.
Table-1	List of Crowe Technologies	3

List of Figures

Figure No.	Figure Caption	Page No.
Fig-1	ASP.Net MVC & ASP.Net Web forms	3
Fig-2	Git Logo	9
Fig-3	Git Lexicon	9
Fig-4	ASP.Net Web API	12
Fig-5	Waterfall Model Diagram	18

1. INTRODUCTION

Crowe Global, commonly referred to as Crowe, previously **Crowe Horwath International**, is a multinational professional services network. It is the 9th largest such global accounting network in the world. The network consists of more than 220 firms with over 42,000 employees in 130 countries.

David Mellor, the current CEO of Crowe Global, succeeded former CEO Kevin McGrath in April 2018. In June 2018, the global network rebranded and changed its name to Crowe. Over 220 member firms across the world are now known under the new single name.

For 100 years, Crowe Global has been making smart decisions for multinational clients working across borders. Ranked the eighth largest accounting network in the world, the network has a total global workforce of more than 36,500 professionals and global revenues in 2019 reached US\$4.4 billion.

Member firms are recognized as leaders in their markets with awards and accolades including: Fortune 100 Best Companies to Work For 2018; top five leader for Strategic Risk Consulting by ALM Intelligence; Le Fonti award for 2018 International Tax Firm of the Year; Consultancy Advisory Firm of the Year 2017 at the Finance Monthly Global Awards; Best Audit Firm by The Accounting and Corporate Regulatory Authority of the Singapore Government.

1.1 Company Profile

Crowe Horwath is a public accounting, consulting and technology firm with offices around the world. ... The firm and its subsidiaries also help clients make smart decisions that lead to lasting value with its tax, advisory and consulting services.

“**Crowe**” is the brand name under which the member firms of **Crowe Global** operate and provide professional services, and those firms together form the Crowe Global network of independent audit, tax, and consulting firms. Crowe may be used to refer to individual firms, to several such firms, or to all firms within the Crowe Global network. The Crowe Horwath Global Risk Consulting entities, Crowe Healthcare Risk Consulting LLC, and our affiliate in Grand Cayman are subsidiaries of **Crowe LLP**. Crowe LLP is an Indiana limited liability partnership and the U.S member firm of Crowe Global. Services to clients are provided by the individual member firms of Crowe Global, but Crowe Global itself is a Swiss entity that does not provide services to clients. Each member firm is a separate legal entity responsible only for its own acts and omissions and not those of any other Crowe Global network firm or other party. Visit www.crowe.com/disclosure for more information about **Crowe LLP**, its subsidiaries, and Crowe Global.

1.2 Technologies Used

Crowe Horwath LLP is a public accounting, consulting, and technology firm with offices across the globe. Connecting deep industry and specialized knowledge with innovative Microsoft technology, our dedicated professionals create value for our clients with integrity and objectivity. Crowe’s technology expertise includes certifications for **ERP and CRM (Dynamics 365), Azure, PowerBI, Microsoft IoT**, and more. Crowe brings decades of industry experience to its manufacturing, financial services, and

healthcare clients. Crowe has maintained a focus on the metals and automotive supplier industries for decades and has built industry specialized accelerators for **Microsoft Dynamics 365** that enhance the platform for the industries unique requirements.

Our in-depth industry knowledge and decades of consulting and accounting experience helps you use insights from Microsoft technology to make better, more informed business decisions – and, ultimately, add value to the enterprise. Metals companies choose Crowe for their **Dynamics 365 ERP** implementation because of:

- A proven track record of successful Dynamics implementations for many of the world's leading metals companies
- An in-depth understanding of the metals industry and a long tradition of providing thought leadership across a broad range of business and management issues
- A proven, proprietary intellectual property that turns the Dynamics 365 platform into a powerful, metals-specific **ERP platform**
- Faster implementation, greater return, a lower total cost of ownership, and outstanding service from a dedicated team of Crowe professionals

Delivering services with innovative technology

Audit services: Crowe audit professionals maintain a rigorous quality control system and use proprietary and outsourced technology to improve process efficiency and effectiveness.

External audits

- Financial statement audits
- Benefit plan audits
- Specialized audits
- Fund and portfolio audits

Other services

- Financial reporting
- Accounting advisory

Tax services: Crowe tax professionals provide services using innovative tax technology solutions to help ease compliance and improve efficiency and tax cash flow.

- State and local tax
- Federal tax
- International tax
- Private client services

Crowe has generated numerous innovative technologies and tools to help clients effectively manage complex processes and regulatory requirements.

Here are just a few examples:

Crowe Caliber helps financial institutions handle all aspects of model risk management, providing consistent model risk workflow management and documentation standards and a documented audit trail to meet examiners' expectations	Crowe R&D Navigator is a Webbased solution that helps automate the process of claiming federal and state research and development (R&D) tax credits.	C-TRAC® solution helps taxexempt organizations meet tax compliance requirements in less time, accurately completing the data gathering and reporting for required forms and schedules without straining internal resources
CiRT® solution enhances communication and efficiency in the audit and tax compliance process and provides features to assign and delegate requests, track and monitor project progress, and address issues quickly.	Crowe Outpatient Charge Capture Analytics is a clinical documentation management program for outpatient services that can help healthcare providers streamline their processes and quickly increase net revenue.	Crowe Use Tax Simplifier solution leverages expense information already available in ERP systems to simplify use tax determinations on purchases. The solution uses guided questionnaires, custom tax matrices, and tax rate tables to accurately determine varying state requirements and provide standardized reporting formats for audit support.
Crowe Credit Income Recovery solution helps healthcare organizations sort through the resolution of credit balances, streamlining notoriously complex healthcare accounting and producing dramatic improvements in efficiency and productivity.	Crowe Navigator for Lenders™ solution helps credit providers perform a number of essential monitoring, analysis, and reporting functions, improving their ability to monitor and manage risk in their portfolios in a cost-effective manner.	Crowe Use Tax Simplifier solution leverages expense information already available in ERP systems to simplify use tax determinations on purchases. The solution uses guided questionnaires, custom tax matrices, and tax rate tables to accurately determine varying state requirements and provide standardized reporting formats for audit support.
Crowe Metals Accelerator is an industry-specific ERP solution that directly addresses the business, financial, and technology requirements of metals manufacturers.	Crowe Revenue Cycle Analytics helps healthcare organizations automate and accelerate the timeconsuming process of monthly financial reporting and analysis	Crowe Conflict Minerals Tracker® solution provides an efficient and thorough method for administering the complex data collection, analysis, and reporting requirements associated.

Table 1 List of Crowe Technologies

1.3 Company Products

Our greatest strength is our ability to understand the strategic needs of our clients combined with the business experience of the professionals we bring to create customized solutions and meet those clients' needs.

Our service teams are committed to delivering value to multinational clients doing business across borders. We are part of a truly international network of business experts with whom we share a commitment to delivering technical excellence and the highest standards of client service.

We invite you to find out more about our services in Audit, Tax, Advisory and Risk.

Audit: Audit services are essential to establish credibility and build reputation - critically important assets in a global economy undermined by business scandals on multiple continents. Companies that successfully address audit issues improve the quality of their financial reporting and stand to gain credibility internationally with leading sources of capital.

Given the importance of audit services, many senior executives and audit committees look to Crowe Horwath International member firms for efficient procedures, the value of their work, and personal service. Your organization stands to gain in multiple areas - reputation, insight, service, and cost - by finding experienced auditors who truly value your business.

Tax: Every day – somewhere in the world – tax regulations, rules, and treaties change. Strategies that reduced taxes yesterday may not work today. New opportunities to save money could be missed – especially as you enter new markets. The talented tax staff in your organization might not have time to keep up with each nuance. Your current tax advisor may think your company is too small for personal attention – or your advisor may be too small to serve you.

Successful companies consider tax implications before they make business decisions, so they do not pay more than their legal obligation. This makes international tax compliance, consulting, and structuring a crucial element in your global strategy.

Advisory: Many acquisitions fail to live up to expectations. The reasons range from poor deal structure, poor strategic fit, failure to identify problems with the quality of earnings, overly optimistic estimates of synergies, to lack of an integration plan.

Evaluating a company in another country compounds these risks. You are dealing with a different language and cultural barriers; different business ethics, legal systems, filing regulations, and accounting principles; transfer pricing that affects taxation – and often government involvement.

But international deals often provide the best growth opportunities. They can offer improved returns from economies of scale, new target markets for existing products/services, access to commodity materials, and a hedge against seasonality.

Even savvy managements and private equity investors cannot know everything they should to make a deal successful, so they need an experienced international advisor.

Risk: As an international business, you manage a challenging array of risks on multiple fronts: strategic, operations, compliance, and reporting. Today's stakeholders – including shareholders, customers, and employees – expect high standards.

While cultures and customs vary, the language of identifying and monitoring risks is consistent in leading boardrooms and management suites throughout the world. Effective and consistent risk management gives you and your employees the confidence to focus on achieving your organization's key business objectives.

Along with every risk comes opportunity. Therefore, leading organizations carefully assess and evaluate their portfolio of risks and allocate appropriate resources to identify and manage their key risks more effectively than competitors do. For such organizations, having an effective risk management function is now an essential management discipline.

1.4 Company Client

International Clients: In their day-to-day operations and activities, multinationals and transnational companies with global operations are faced with a mixture of financial, legal, IT and operational challenges, both in their existing markets and in the markets they seek to develop.

Successful multinationals cannot make progress without an Assurance and Advisory Firm, which can rely on a worldwide global network of Member Firms, whose professionals share their language, culture, concerns, vision and business objectives.

Crowe | Callens, Pirene, Theunissen & C^o is affiliated to Crowe Global, which ranks among the 10 largest Audit and Accounting networks worldwide and is in a position to support international clients with their state of the art service offerings fully aligned to the needs of international businesses.

With more than 750 offices in more than 130 countries and more than 33.000 employees, our network has a real global presence.

What makes our Mid Tier Network different?

Our strong market position in the key economic centers (such as China, the United States, Asia Pacific, the Middle East and Central America) as well as our neighbor countries (Germany, France, UK, the Netherlands).

Audit

IT Services

Tax & Legal

Corporate Finance

Accounting & Reporting

Risk Management

Crowe Horwath HTL

Since 1915 Horwath Hotel, Tourism and Leisure consulting has maintained its status as the world's number one hospitality consulting organization. Horwath HTL is the industry's choice for expert

professional service throughout the world. Horwath HTL is a distinct and integral part of our global international network, offering complete solutions also for hotel and tourism operators in Belgium.

Through involvement in thousands of projects the years, Horwath HTL has amassed extensive, in-depth knowledge and expertise regarding the needs of hotels, real estate companies and financial institutions.

Today, Horwath HTL is the world's largest consulting organization specializing in the hospitality industry, with 50 offices in 39 countries. Horwath HTL is recognized for its comprehensive, analytic and innovative approach to serving clients, whether they are local, regional or global.

Horwath HTL's areas of specialty are:

Hotel Asset Management

Hotel Planning & development

Tourism & Leisure Services

Hotel Valuation

Strategic Advice

With the help of these subject-matter-experts we believe Crowe | Callens, Pirenne, Theunissen & C° can make the difference in delivering external audit services and risk consulting services to the Hotel and Tourism sector in Belgium.

CHAN Healthcare

Crowe Global and CHAN Healthcare have combined resources to create one of the largest and most experienced providers of internal audit, risk management, and financial advisory services in the healthcare industry worldwide.

CHAN Healthcare provides all internal audit services a nonprofit healthcare organization might need to improve its operations — from implementing our risk-based Internal Audit Model to providing services related to coding and compliance, IT, and governance education.

Together with our Firm's position in the Crowe Global's Healthcare Specialty Group, this combination strengthens the ability of Crowe | Callens, Pirenne, Theunissen & C° to help healthcare industry service providers in Belgium (hospitals, national health insurance providers, local public welfare institutions, rest homes, elder facility providers,...) to mitigate emerging risk and financial issues.

2 Definition of Internship:

According to the National Association of Colleges and Employers (NACE), an internship is a form of experiential learning that integrates knowledge and theory learned in the classroom with practical application and skills development in a professional setting. Internships give students the opportunity to gain valuable applied experience and make connections in professional fields they are considering for career paths and give employers the opportunity to guide and evaluate talent.

What Constitutes an Internship?

An internship is a short-term, hands-on, supervised work experience with a professional organization that is designed to increase a student's knowledge of a professional career field. More than a part-time job or volunteer experience, an internship includes intentional learning objectives related to increasing student knowledge, training to develop additional skills, and quality supervision to guide and mentor the intern.

The internship is for 6 Months(till August 2021) at **Crowe Horwath**. During this internship till now I have learnt basics of many technologies and a good hand on practice on each of the technologies.

2.1 Technologies Learnt:

- **C# (C Sharp)**

C# is a general-purpose, multi-paradigm programming language encompassing static typing, strong typing, lexically scoped, imperative, declarative, functional, generic, object-oriented, and component-oriented programming disciplines.

C# (C-Sharp) is a programming language developed by Microsoft that runs on the .NET Framework.

C# is used to develop web apps, desktop apps, mobile apps, games and much more.

- **ASP.Net MVC**

ASP.NET MVC is basically a web development framework from Microsoft, which combines the features of MVC (Model-View-Controller) architecture, the most up-to-date ideas and techniques from **Agile development**, and the best parts of the existing ASP.NET platform.

ASP.NET MVC is not something, which is built from ground zero. It is a complete alternative to traditional ASP.NET Web Forms. It is built on the top of ASP.NET, so developers enjoy almost all the ASP.NET features while building the MVC application.

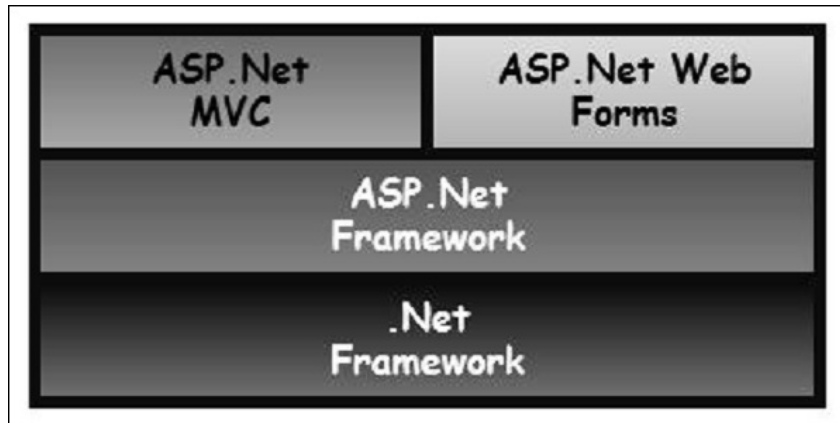


Fig-1 ASP.Net MVC and ASP.Net Web forms

Why ASP.NET MVC?

Microsoft decided to create their own MVC framework for building web applications. The MVC framework simply builds on top of ASP.NET. When you are building a web application with ASP.NET MVC, there will be no illusions of state, there will not be such a thing as a page load and no page life cycle at all, etc.

Another design goal for ASP.NET MVC was to be extensible throughout all aspects of the framework. So when we talk about views, views have to be rendered by a particular type of view engine. The default view engine is still something that can take an ASPX file. But if you don't like using ASPX files, you can use something else and plug in your own view engine.

There is a component inside the MVC framework that will instantiate your controllers. You might not like the way that the MVC framework instantiates your controller, you might want to handle that job yourself. So, there are lots of places in MVC where you can inject your own custom logic to handle tasks.

The whole idea behind using the Model View Controller design pattern is that you maintain a separation of concerns. Your controller is no longer encumbered with a lot of ties to the ASP.NET runtime or ties to the ASPX page, which is very hard to test. You now just have a class with regular methods on it that you can invoke in unit tests to find out if that controller is going to behave correctly.

Benefits of ASP.NET MVC

Following are the benefits of using ASP.NET MVC –

- Makes it easier to manage complexity by dividing an application into the model, the view, and the controller.
- Enables full control over the rendered HTML and provides a clean separation of concerns.
- Direct control over HTML also means better accessibility for implementing compliance with evolving Web standards.
- Facilitates adding more interactivity and responsiveness to existing apps.
- Provides better support for test-driven development (TDD).

- Works well for Web applications that are supported by large teams of developers and for Web designers who need a high degree of control over the application behavior.



Fig-2 Git Logo

- **Git Hub**

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

Git Hub Logo

What is GitHub?

GitHub is like Wikipedia for programmers. You can edit files, see who changed what, vie old versions of files, and access it from anywhere in the world.

What is Git?

Git is an open source code management system. The basic idea of git is to keep track of different versions of code or text, so you can easily compare what has changed.

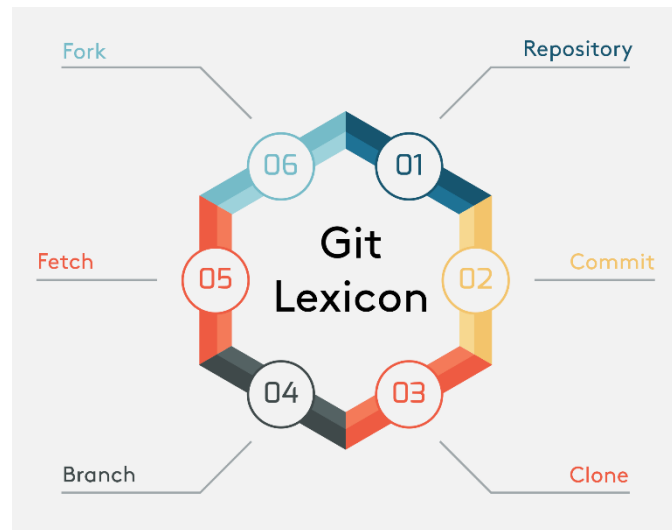


Fig-3 Git Lexicon

1. **Repository** : The most fundamental element of GitHub, a repository is essentially a project's folder. Repositories store every single project file, its documentation and its revision history of every document.
2. **Commit**: Commits are easily one of the most frequented activities by a developer using GitHub. A Commit is like 'saving' an updated file to its original folder.
3. **Clone**: Clones are literally clones (Copies) of a repository that sit on the developer's computer instead of a server. Clones are great since you can download a file to tinker around with offline.
4. **Branch** : A branch is a parallel version of repository. It is contained with in the repository, but does not affect the primary or master branch allowing to work freely without disrupting the "live" version.
5. **Fetch** : Fetching refers to getting changes from an online repository (like GitHub.com) without merging them in. Once these changes are fetched you can compare them to your local branches.

6. **Fork** : A ‘fork’ is a personal copy of another user’s repository that lives in your GitHub account. Forks allow you to freely make changes to a project without affecting the original.

Why GitHub is special?

At the heart of GitHub is Git, an open source project started by Linux creator Linus Torvalds. Matthew McCullough, a trainer at GitHub, explains that Git, like other version control systems, manages and stores revisions of projects. Although it’s mostly used for code, McCullough says Git could be used to manage any other type of file, such as Word documents or Final Cut projects. Think of it as a filing system for every draft of a document.

Some of Git’s predecessors, such as CVS and Subversion, have a central “repository” of all the files associated with a project. McCullough explains that when a developer makes changes, those changes are made directly to the central repository. With distributed version control systems like Git, if you want to make a change to a project you copy the whole repository to your own system. You make your changes on your local copy, then you “check in” the changes to the central server. McCullough says this encourages the sharing of more granular changes since you don’t have to connect to the server every time you make a change.

GitHub is a Git repository hosting service, but it adds many of its own features. While Git is a command line tool, GitHub provides a Web-based graphical interface. It also provides access control and several collaboration features, such as a wikis and basic task management tools for every project.

The flagship functionality of GitHub is “forking” – copying a repository from one user’s account to another. This enables you to take a project that you don’t have write access to and modify it under your own account. If you make changes you’d like to share, you can send a notification called a “pull request” to the original owner. That user can then, with a click of a button, merge the changes found in your repo with the original repo.

These three features – fork, pull request and merge – are what make GitHub so powerful. Gregg Pollack of Code School (which just launched a class called TryGit) explains that before GitHub, if you wanted to contribute to an open source project you had to manually download the project’s source code, make your changes locally, create a list of changes called a “patch” and then e-mail the patch to the project’s maintainer. The maintainer would then have to evaluate this patch, possibly sent by a total stranger, and decide whether to merge the changes.

This is where the network effect starts to play a role in GitHub, Pollack explains. When you submit a pull request, the project’s maintainer can see your profile, which includes all of your contributions on GitHub. If your patch is accepted, you get credit on the original site, and it shows up in your profile. It’s like a resume that helps the maintainer determine your reputation. The more people and projects on GitHub, the better idea picture a project maintainer can get of potential contributors. Patches can also be publicly discussed.

Even for maintainers who don't end up using the GitHub interface, GitHub can make contribution management easier. "I end up just downloading the patch anyway, or merging from the command line instead of from the merge button," says Isaac Schlueter, the maintainer of the open source development platform Node.js. "But GitHub provides a centralized place where people can discuss the patch."

Lowering the barrier to entry democratizes open source development, and helps young projects grow. "Node.js wouldn't be what it is today without GitHub," Schlueter says.

Besides its public facing open source repositories, GitHub also sells private repositories and on-premise instances of its software for enterprises. These solutions obviously can't take full advantage of GitHub's network effect, but they can take advantage of the collaboration features. That's how GitHub makes money, but it's not alone in this market.

Atlassian acquired a competitor called BitBucket in 2010. And earlier this year Atlassian launched Stash, a product that enables you to host private, on-premise Git repositories with BitBucket/GitHub-style collaboration features. The company also sells developer collaboration tools like the bug tracker Jira and the wiki Confluence. Competition from Atlassian, which took \$60 million in funding from Accel Partners in 2010, could help explain why GitHub took this round of funding, and hint at some possible future directions for the company. For example, Schlueter says GitHub's issue tracking feature could eventually compete with JIRA for some projects.

The money may be in private and on-premise hosting, but the love is in the public repositories. Perhaps most importantly, GitHub has become the Library of Alexandria for code examples. Since Git encourages granular recording of changes, programmers, be they absolute beginners or experts, can trace the steps of some of the greatest developers in the world and find out how they solved thorny problems. But if GitHub were ever to meet the same fate as the Library of Alexandria, it could be reconstructed from all those local forks distributed on so many developers laptops all over the world. Regardless of how this investment works out, that's a hell of a legacy for the GitHub team to leave behind.

- **ASP.Net Web API**

ASP.NET Web API is a framework that makes it easy to build HTTP services that reach a broad range of clients, including browsers and mobile devices. ASP.NET Web API is an ideal platform for building RESTful applications on the .NET Framework.

When you're building APIs on the Web, there are several ways you can build APIs on the Web. These include HTTP/RPC, and what this means is using HTTP in Remote Procedure Call to call into things, like Methods, across the Web.

The verbs themselves are included in the APIs, like Get Customers, Insert Invoice, Delete Customer, and that each of these endpoints end up being a separate URI.

Introduction to Web API

Web API is a programming interface/application type that provides communication or interaction between software applications. Web API is often used to provide an interface for web sites and client applications to have data access. Web APIs can be used to access data from a database and save data back to the database.

ASP.NET Web API is a framework that make it easy to build HTTP web service that reaches a broad range of clients, including browser, mobile applications, Desktop application and IOTs.

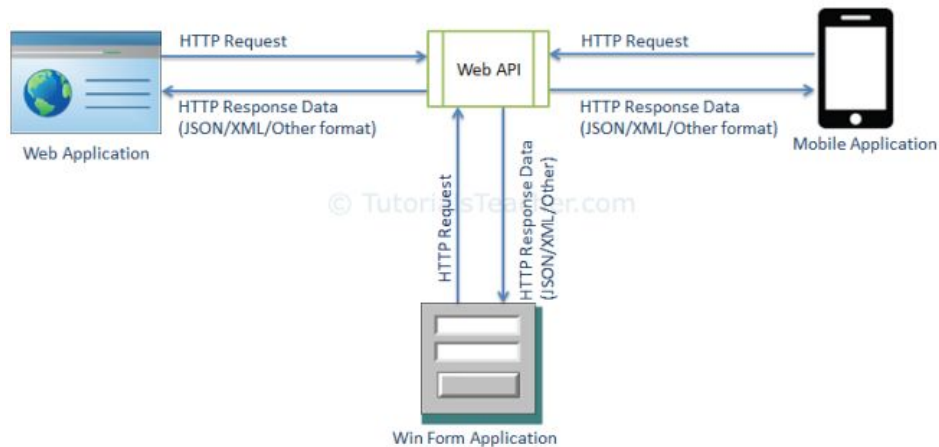


Fig-4 ASP.Net Web API

- **REST Features**

Rest Stands for Representational state transfer .Its introduce in 2000 by Roy Fielding. In REST architecture, a REST Server simply provides access to resources and the REST client accesses and presents the resources. Here each resource is identified by URIs/ Global IDs. REST uses various representations to represent a resource like Text, JSON and XML. JSON is now the most popular format being used in Web Services.

HTTP Methods

The following HTTP methods are most commonly used in a REST based architecture.

1. GET – Provides a read only access to a resource.
2. PUT – Used to create a new resource.
3. DELETE – Used to remove a resource.
4. POST – Used to update an existing resource or create a new resource.

REST Constraints

REST constraints are design rules that are applied to establish the distinct characteristics of the REST architectural style.

The formal REST constraints are,

1. Client-Server
2. Stateless
3. Cache
4. Interface / Uniform Contract
5. Layered System
6. Code-On-Demand

- **HTML**

1. HTML stands for Hyper Text Markup Language
 2. HTML is the standard markup language for creating Web pages
 3. HTML describes the structure of a Web page
 4. HTML consists of a series of elements
 5. HTML elements tell the browser how to display the content
- **CSS**
 1. CSS stands for Cascading Style Sheets
 2. CSS describes how HTML elements are to be displayed on screen, paper, or in other media
 3. CSS saves a lot of work. It can control the layout of multiple web pages all at once
 4. External stylesheets are stored in CSS files
 - **BOOTSTRAP**

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.
 - **JavaScript**

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Advantages of JavaScript

Less server interaction – You can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.

Immediate feedback to the visitors – They don't have to wait for a page reload to see if they have forgotten to enter something.

Increased interactivity – You can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.

Richer interfaces – You can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

- **Ajax**

Ajax is a set of web development techniques using many web technologies on the client-side to create asynchronous web applications. With Ajax, web applications can send and retrieve data from a server asynchronously without interfering with the display and behaviour of the existing page.

AJAX is a developer's dream, because you can:

Update a web page without reloading the page
Request data from a server - after the page has loaded
Receive data from a server - after the page has loaded
Send data to a server - in the background

- **JQuery**

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

2.2 Selection of Project work based upon technology learnt

The Project **Employee Management System** is selected based upon the technologies that I have learnt.

The front-end part of this project is developed by using HTML, CSS, BOOTSTRAP and JavaScript. And the back-end part is developed using some Ajax, JQuery and JavaScript. The big role to creating its back-end is maintained by ASP.Net Web API and REST Features.

Rest features plays a very important role in this project. Because in order to fetch or modify the data in Database the REST features are very helpful.

REST stands for REpresentational State Transfer. It relies on a stateless, client-server, cacheable communications protocol -- and in virtually all cases, the HTTP protocol is used.

REST is an architecture style for designing networked applications. The idea is that, rather than using complex mechanisms such as CORBA, RPC or SOAP to connect between machines, simple HTTP is used to make calls between machines.

In many ways, the World Wide Web itself, based on HTTP, can be viewed as a REST-based architecture. RESTful applications use HTTP requests to post data (create and/or update), read data (e.g., make queries), and delete data. Thus, REST uses HTTP for all four CRUD (Create/Read/Update/Delete) operations.

REST is a lightweight alternative to mechanisms like RPC (Remote Procedure Calls) and Web Services (SOAP, WSDL, et al.).

Despite being simple, REST is fully-featured; there's basically nothing you can do in Web Services that can't be done with a RESTful architecture. REST is not a "standard". There will never be a W3C recommendataion for REST, for example. And while there are REST programming frameworks, working with REST is so simple that you can often "roll your own" with standard library features in languages like Perl, Java, or C#.

What is API?

API stands for Application Programming Interface. API can be explained as a set of functions or subroutines or set of procedures or set of communication protocols, which is used to build software and applications.

Example 1:

Suppose you want to create a weather app or an app that can easily convert the currencies to a desired currency .Now you will think that this weather and currency both are changeable quantities and you alone can not keep track of those things. Then how can you build that app?

In this case, if you get an access to the server of weather department or get an access into the database of stock exchange department then your work can be more easy .But question is that how can you fetch data from those servers, as your application is totally different from those applications. You can do this with the help of API. It's an

Example 2:

Suppose you have a confectionery shop in the middle of the city and you want to spread the business. So, decided to create an online website to sell those stuffs .But here is a twist, you want to link your tally based billing software with the website .So whenever there is an order appears into your website that order data will go to the software database. So how you can do this thing?

Here also the answer is API.API will help you to integrate these two platforms.

Example 3:

In modern application you have often seen that ,whenever you try to log in to these applications they are giving you the option to login using Google account or Facebook account .So here you can ask yourself, how these can be possible , as these applications are totally different from that of Google and Facebook. And these are not the child applications either of these two companies.

What is Web API?

A Web API is just an API that uses HTTP as its communication method – even the basic loading of an HTML page is a form of API use. Generally though, the term ‘Web API’ means requesting data in the form of JSON or XML, since those are more purely data-focused – things like map data, lists of tweets matching a search term, additional user profile data, that sort of thing.

Web APIs are used by an application usually in one of two ways. First, the application server (running Rails, PHP, Node etc.) can make the API request itself, and build the resulting data into its HTML that it sends to the browser. Alternatively, the JavaScript code running on the browser can make an HTTP request directly using XHR and using DOM manipulation functions to show the data without needing to reload the page.

2.3 Platform Requirements

1. Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that expand the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Azure DevOps client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML, and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) were supported in the past.

The most basic edition of Visual Studio, the Community edition, is available free of charge. The slogan for Visual Studio Community edition is "Free, fully-featured IDE for students, open-source and individual developers".

Microsoft SQL Server

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications—which may run either on the same computer or on another computer across a network (including the Internet). Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users.

Chrome Browser

Chrome is a free Internet browser officially released by Google on December 11, 2008. Its features include synchronization with Google services and accounts, tabbed browsing, and automatic translation and spell check of web pages. It also features an integrated address bar/search bar, called the omnibox.

INTRODUCTION & OBJECTIVES OF PROJECT WORK

INTRODUCTION

An employee management system is a platform where all work-related as well as important personal details of an employee is stored and managed in a secure way. By using this system, you can manage admin activities in an easier and quicker way.

Employees are the pillar of any organization and an ideal employee management tool makes a big difference to an organization.

An employee management system consists of crucial work-related and important personal information about an employee. In a nutshell, it is an online inventory of all employees of an organization.

Employees are the strength of any organization, and it is more so in case of a growing business. It is crucial to handle this aspect of your business well. A good employee management system can actually make a world of difference to an organization, especially true in case of startups and small businesses, where the focus should be on growing the business more than anything else.

OBJECTIVES OF PROJECT WORK**1. It's a ready source of information:**

Employee Management System performs as a readily available source of information between the organization and the employee. Contact information, salary information, posts, work schedule, education information etc. is what most database systems consist of.

2. Highly efficient system:

Employee management systems are highly efficient. A member of the organization can easily retrieve information about his/her colleague whenever required, and that too on short notice. One can avoid making calls to the employee out on vacation just to retrieve an address to send an important letter.

3. Reliable accuracy:

Since the information is mostly fed in by the employees themselves you can be sure the information is accurate since it's straight from the source. Moreover, an employee can access their information at any time. Therefore, he/she can keep it updated and correct mistakes, if any.

4. Updated data:

The information added to the employee management system can be available for as long as an employee is working in a firm or if needed, longer than that. Also, if at any point in time, the employee data changes the employee themselves can make the alterations. As a result, obsolete data is a rare find on such systems.

5. Employee management system allows confidentiality:

Specific information about the employee can not only be set to be kept private from public viewing but can also be set to be kept private from anyone other than the admin of the software or the head of the organization. Therefore, it is safer to have an employee management system in an organization, big or small, than have your bank account information lying in some drawer of a dingy desk.

4 INTRODUCTION

A project management methodology is essentially a set of guiding principles and processes for managing a project. Your choice of methodology defines how you work and communicate.

So how do you choose a project management methodology?

What methodology you choose will depend on your team, project-type, and project-scope. Choosing project management methodologies (PMM) is one of the first decisions you'll have to make as a project manager.

What methodology you pick will have a profound and ongoing impact on how you and your team works.

Different project management methodologies have their own pros and cons for different project types. Some are geared for speed, some for comprehensiveness.

4.1 Proposed Algorithm or Technique

1. Waterfall

The Waterfall methodology is the oldest methodology on this list. It was first outlined by Dr. Winston Royce in 1970 as a response to managing the increasingly complex nature of software development. Since then, it has become widely adopted, most prominently in the software industry.

The Waterfall methodology is sequential. It is also heavily requirements-focused. You need to have a crystal clear idea of what the project demands before proceeding further. There is no scope for correction once the project is underway.

The Waterfall method is divided into discrete stages. You start by collecting and analyzing requirements, designing the solution (and your approach), implementing the solution and fixing issues, if any. Each stage in this process is self-contained; you wrap up one stage before moving onto another. Graphically, you can represent it as follows:

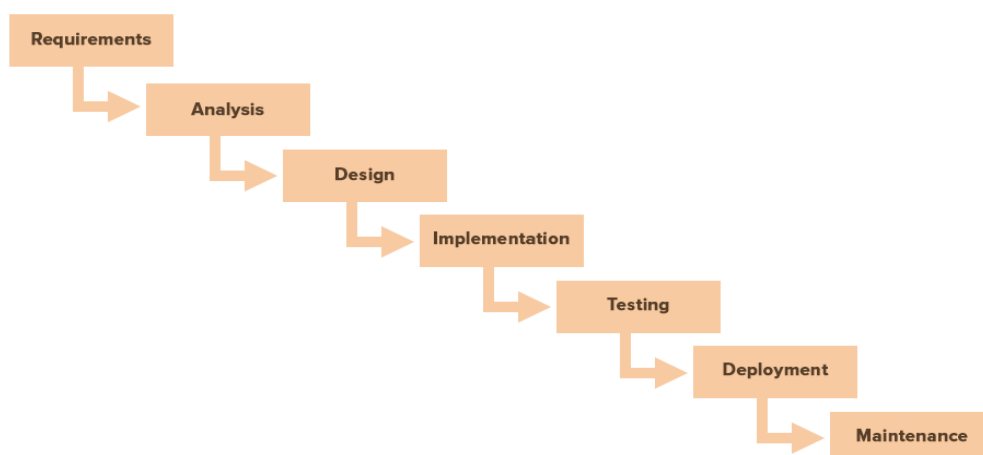


Fig-5 Waterfall Model diagram

Advantages

Ease of use: This model is easy to understand and use. The division between stages is intuitive and easy to grasp regardless of prior experience.

Structure: The rigidity of the Waterfall method is a liability, but can also be a strength. The clear demarcation between stages helps organize and divide work. Since you can't go back, you have to be "perfect" in each stage, which often produces better results.

Documentation: The sharp focus on gathering and understanding requirements makes the Waterfall model heavily reliant on documentation. This makes it easy for new resources to move in and work on the project when needed.

Disadvantages

Higher risk: The rigidity of this methodology means that if you find an error or need to change something, you have to essentially start the project from the beginning. This substantially increases the risk of project failure.

Front-heavy: The entire Waterfall approach depends heavily on your understanding and analyzing requirements correctly. Should you fail to do that - or should the requirements change - you have to start over. This lack of flexibility makes it a poor choice for long and complex projects.

Best for:

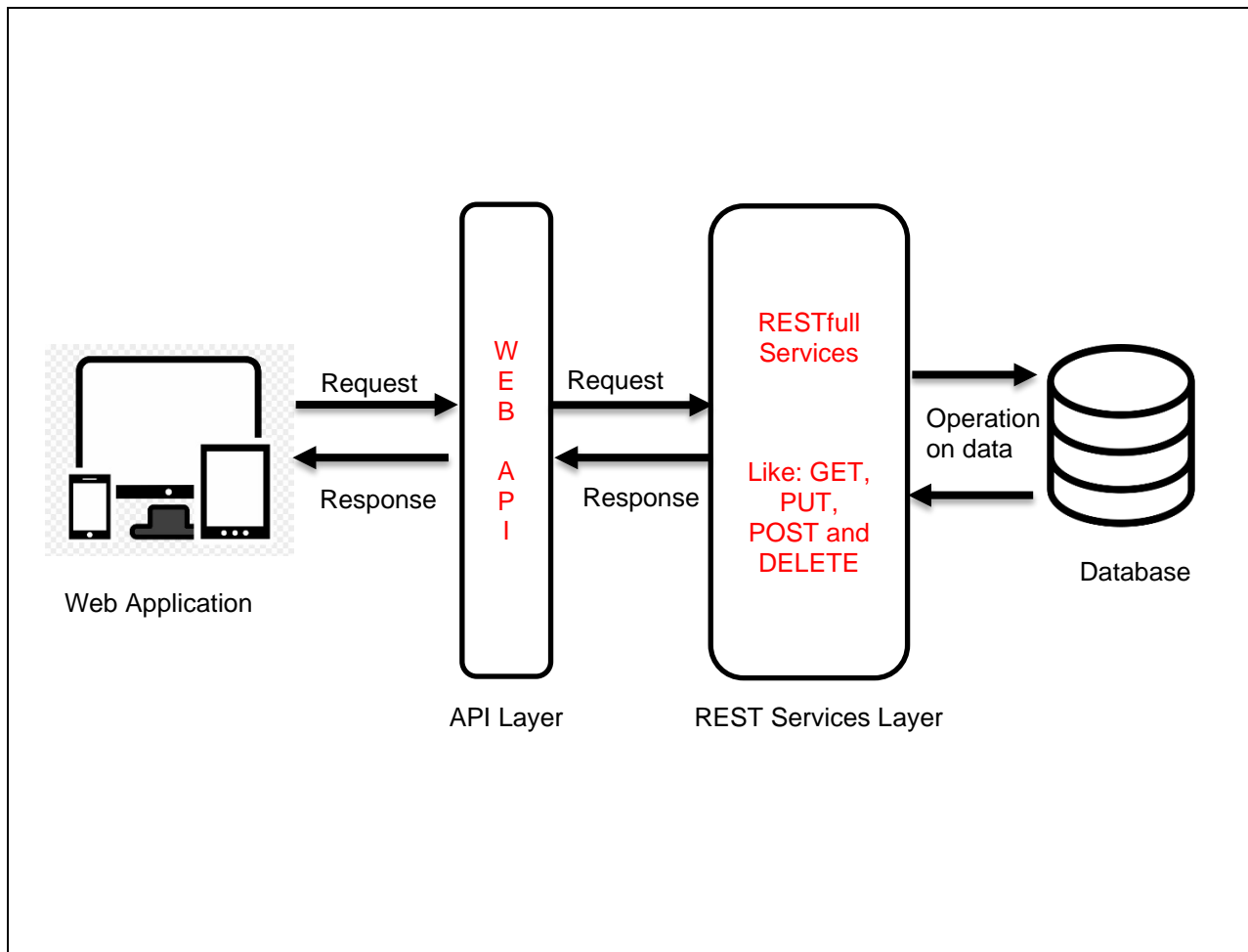
The Waterfall methodology is most commonly used in software development. It works best for the following project types:

Short, simple projects

Projects with clear and fixed requirements

Projects with changing resources that depend on in-depth documentation

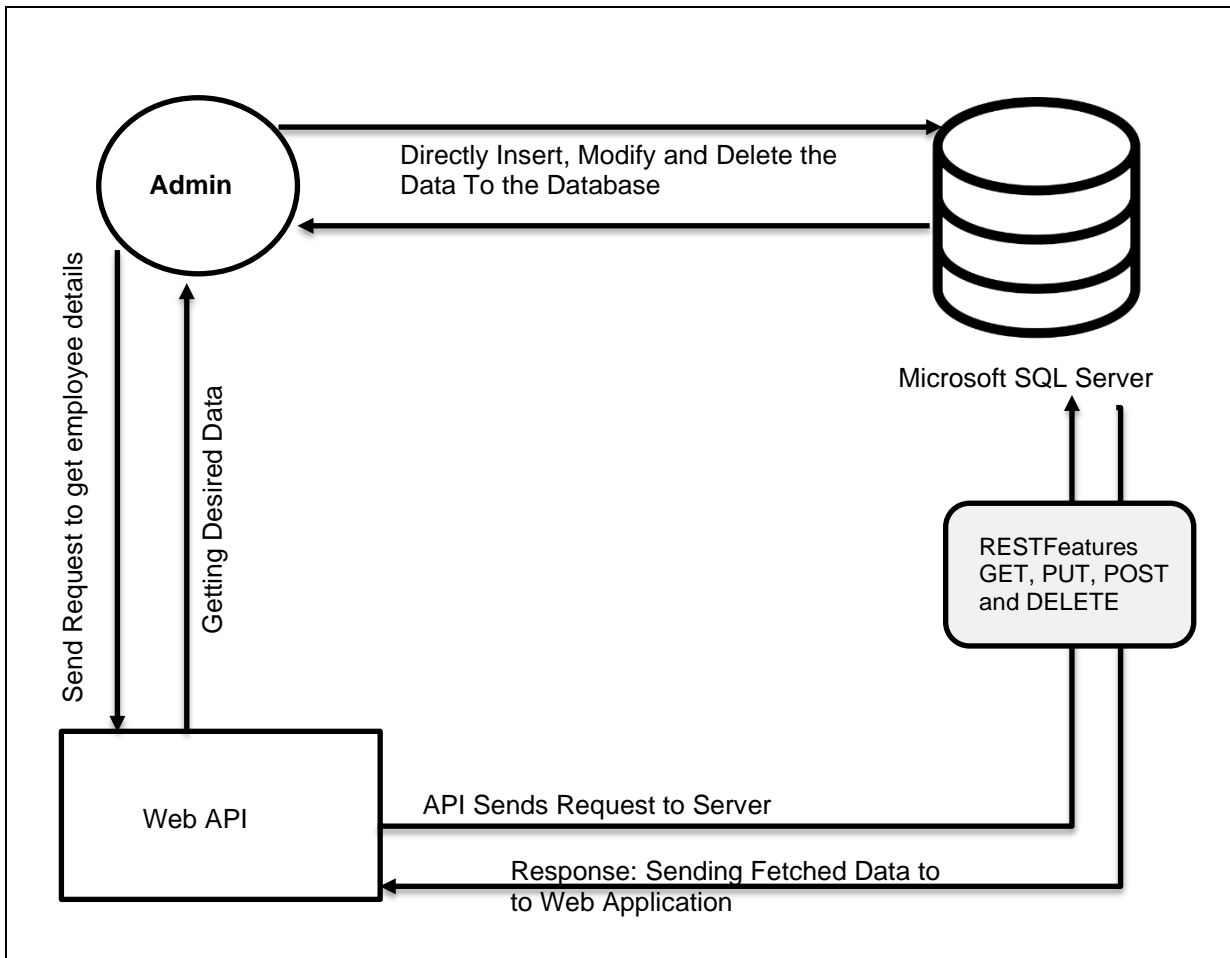
4.2 Data flow diagrams used:



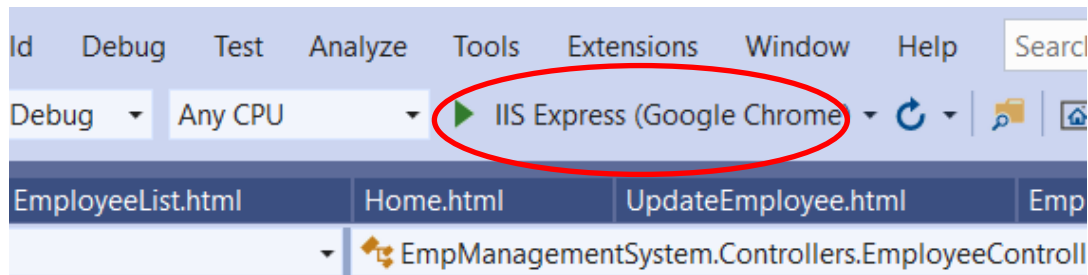
The Web API EmployeeService that we are going to build will retrieve the data from the Employees database table. We will be using Entity Framework to retrieve data from the SQL server database.

Layered System:

The REST allows us to use a layered system architecture where we deploy the APIs in server A, and store data on server B and authenticate requests in server C., For example, a client cannot simply tell whether he is connected directly to the server or to an intermediary server on the way.



5 Screenshots obtained by running project with description.



loyee

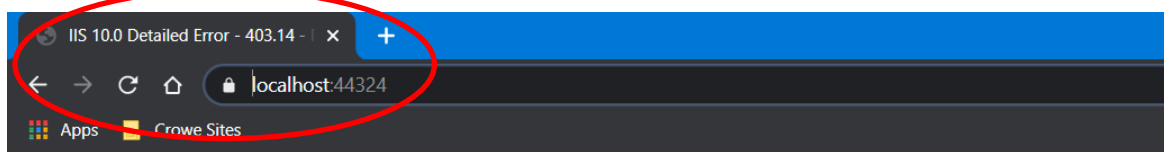
nar, 31 days ago | 1 author, 1 change

`pActionResult GetEmployees()`

```
able<Employee> employees = null;
```

```
es = this._employeeRepository.GetEmployees().ToList();
```

When we click on that green icon (IIS Express), Our project will run on local host.



HTTP Error 403.14 - Forbidden

The Web server is configured to not list the contents of this directory.

Most likely causes:

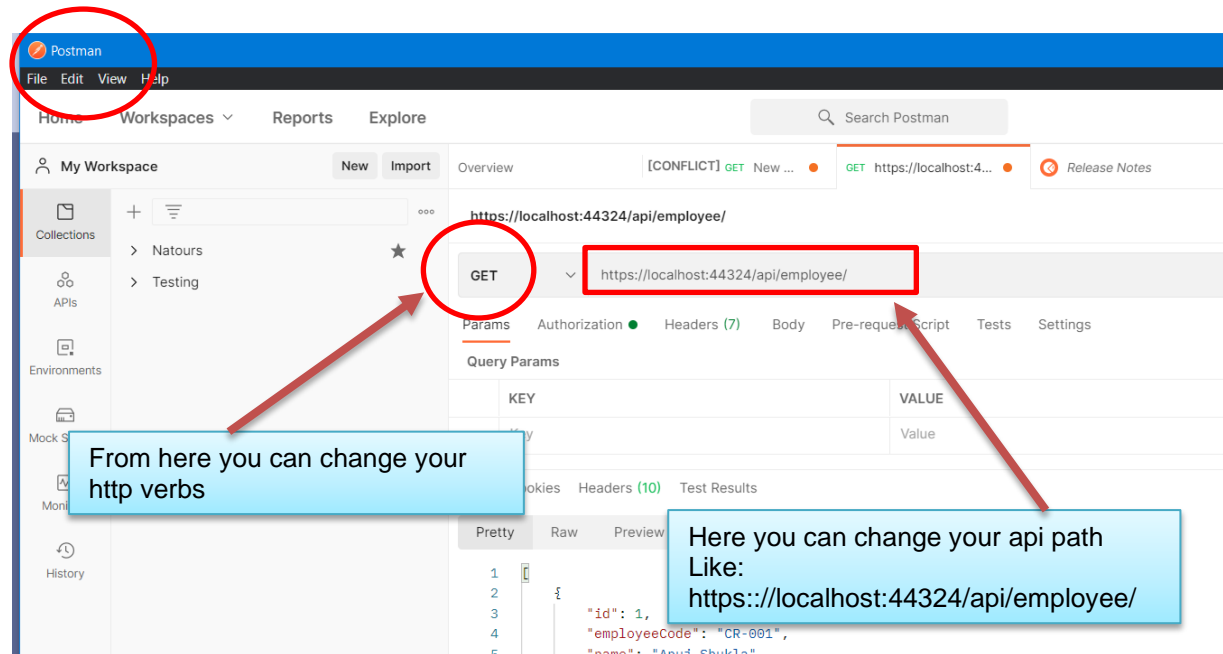
- A default document is not configured for the requested URL, and directory browsing is not enabled on the server.

Things you can try:

- If you do not want to enable directory browsing, ensure that a default document is configured and that the file exists.
- Enable directory browsing.
 1. Go to the IIS Express install directory.

So, When we encounter this screen our project is ready to go. This error is shows that you are ready to go to your project link.

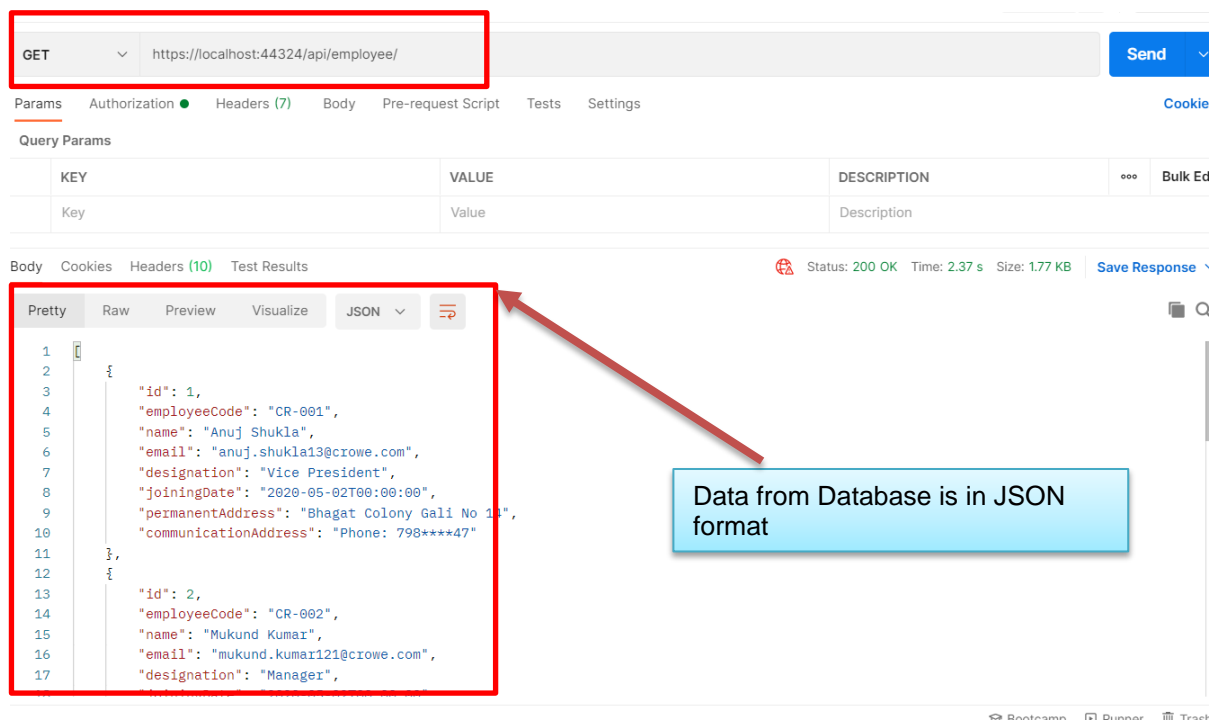
Now Open up the Postman to test your API that you have been created to operate data on the server.



So, after it lets assume we want all the records that are stored in database.

We will make that link as: <https://localhost:44324/api/employee/>

This link will fetch all the records from database in a JSON format.



Now lets say, we want a single employee record. And for that we will fetch that record by its id. Using the the link <https://localhost:44324/api/employee?id=2>

The “?” question mark is here to say that this is optional.

The screenshot shows a REST client interface. The request is a GET to `https://localhost:44324/api/employee?id=2`. The response is a JSON object with the following details:

```
{
  "id": 2,
  "employeeCode": "CR-002",
  "name": "Mukund Kumar",
  "email": "mukund.kumar121@crowe.com",
  "designation": "Manager",
  "joiningDate": "2020-05-02T00:00:00",
  "permanentAddress": "Bhagat Colony Gali No 14",
  "communicationAddress": "Phone: 798****47"
}
```

Just cross check to the database, whether we get a valid record or not.

The screenshot shows Microsoft SQL Server Management Studio with a query executed on the `EmployeeManagementSystem` database. The query is:

```
SELECT TOP (1000) [Id]
, [EmployeeCode]
, [Name]
, [Email]
, [Designation]
, [JoiningDate]
, [PermanentAddress]
, [CommunicationAddress]
FROM [EmployeeManagementSystem].[dbo].[Employees]
```

The results table shows the following data:

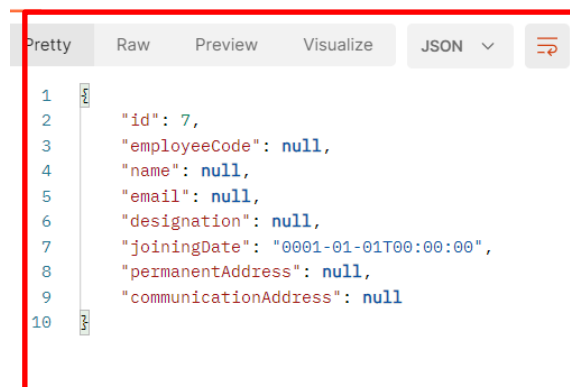
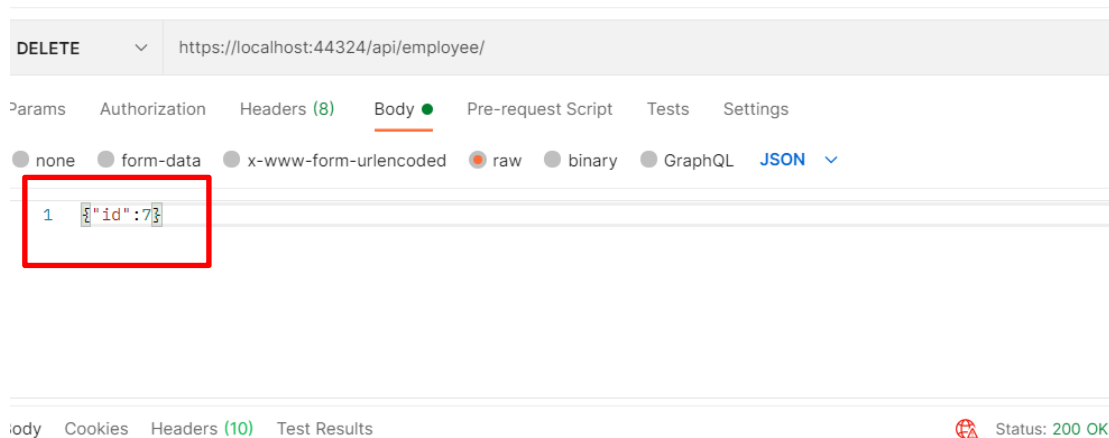
	Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	3	CR-003	Arun Yadav	arun.yadav@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
4	4	CR-004	Sachin Tekchandani	sachin.tekchandani@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
5	5	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	
6	7	CR-006	Dharmendra Chaurasia	dharmendra.chaurasia@crowe.com	Project Manager	2021-04-26		Phone: 798****47

Now Lets Delete a record whose id is 7.

In order to do that we just set the HTTP verb to DELETE and link will be:

<https://localhost:44324/api/employee/>

And under body section just type in JSON format { "id":7 }



Database before deletion.

The screenshot shows a database table with 8 columns: Id, EmployeeCode, Name, Email, Designation, JoiningDate, PermanentAddress, and CommunicationAddress. The table contains 7 records. The first record (Id: 1) is highlighted with a red box.

Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	CR-003	Arun Yadav	arun.yadav@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
4	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	
5	CR-007	Pawan Kumar	pawan.kumar121@crowe.com	Project Manager	2020-05-02	Bareilly Colony Gali No 14	Phone: 798****47

Database after deletion

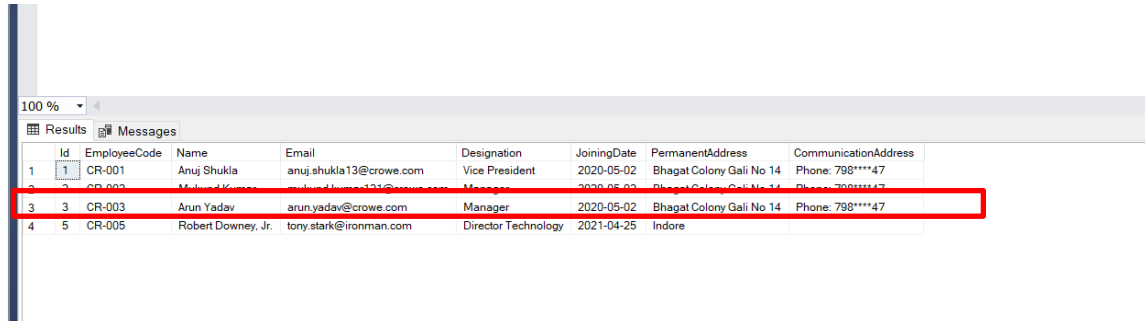
The screenshot shows the same database table as before, but with only 4 records remaining. The first record (Id: 1) is highlighted with a red box.

Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	CR-003	Arun Yadav	arun.yadav@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
4	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	

Now, we will see the UPDATE feature.

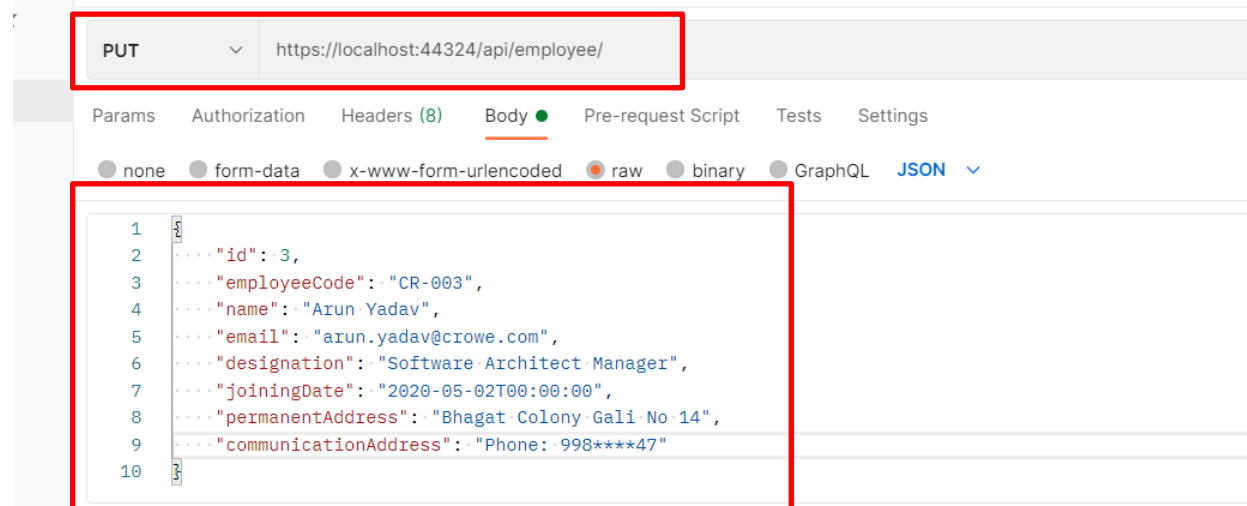
Lets say we want to update the record whose id is 3.

Before Updation

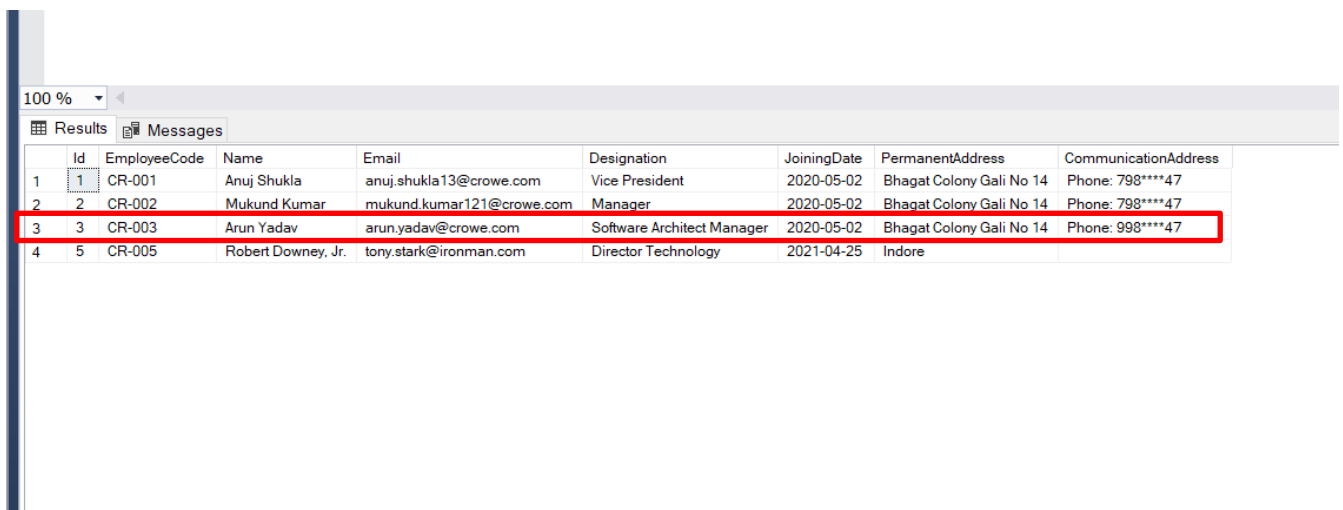


	Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	3	CR-003	Arun Yadav	arun.yadav@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
4	5	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	

Change the HTTP verb to PUT and your link will be <https://localhost:44324/api/employee/>



After Updation in database



	Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 998****47
4	5	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	

Now, we will see how to insert the record in database.

Change the HTTP verb to POST and make the link <https://localhost:44324/api/employee/> Database before insertion.

The screenshot shows a REST client interface. At the top, there is a table with employee data. Below it, the 'POST' method is selected for the URL `https://localhost:44324/api/employee/`. The 'Body' tab is active, showing a JSON payload for a new employee record. The status bar at the bottom indicates a successful response with status 200 OK.

	Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 998****47
4	5	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	

POST `https://localhost:44324/api/employee/`

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
{
  "id": 6,
  "employeeCode": "CR-006",
  "name": "Keanu Reeves",
  "email": "reeves.keanu@crowe.com",
  "designation": "Project Manager",
  "joiningDate": "2020-05-02T00:00:00",
  "permanentAddress": "Chicago",
  "communicationAddress": "Phone: 9678****47"
}
```

Body Cookies Headers (10) Test Results Status: 200 OK

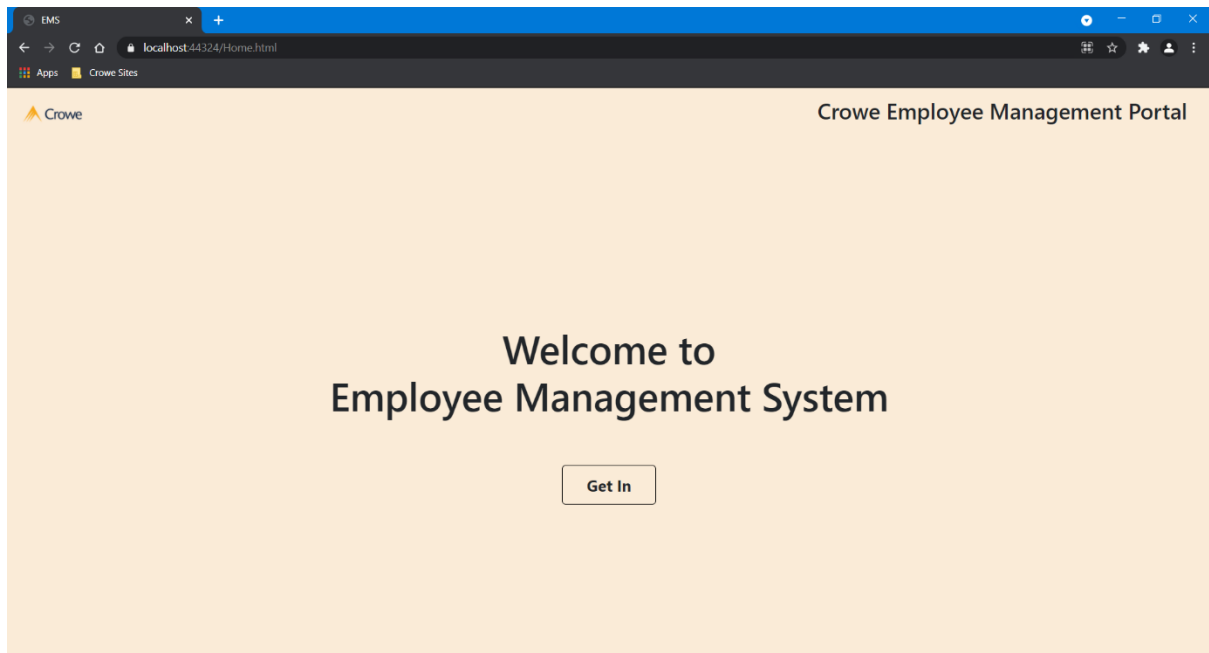
Database After Inserting the record.

The screenshot shows the same database table as before, but with an additional record at the bottom. The new record has an ID of 11, which is highlighted with a red box. The other records remain the same.

	Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 998****47
4	5	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	
5	11	CR-006	Keanu Reeves	reeves.keanu@crowe.com	Project Manager	2020-05-02	Chicago	Phone: 9678****47

Here in this record we will see the id is 11 and we send id as 6 so, this is because of autoincrement of record id, as the previous record is 5 and we are sending new record id as 6 so it sums up it and the autoincrement functionality is normally working by adding 1 to its previous record id.

Screen shots of our web application UI



The landing page of our web application with descriptions.

The screenshot shows the 'EmployeeList' page of the Crowe Employee Management Portal. The browser address bar indicates the URL is localhost:44324/EmployeeList.html. The page has a header 'Crowe Employee Management Portal' and a sub-header 'Manage Employees'. A green box highlights the 'Add Employee Button' (a blue square with a white plus icon) and a text box says 'Add Employee Button to add the record'. A yellow box highlights the 'Search for names...' input field, with a blue box labeled 'Search Field' pointing to it. A red box highlights the main table of employee records, with a blue box labeled 'Dynamically generated table with all the records.' pointing to it. The table has columns: S.No, Employee Code, Name, Email, Designation, Joining Date, and Actions. The Actions column contains 'Edit' (pencil icon) and 'Delete' (trash icon) buttons for each record. A red box highlights the pagination controls at the bottom of the table, with a red box labeled 'Table Pagination which will contain 5 records at a time' pointing to it. A red box labeled 'Dynamically Created Edit and Delete button with each record' points to the Actions column. A detailed view of the table and pagination is shown below the main screenshot.

S.No	Employee Code	Name	Email	Designation	Joining Date	Actions
1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02T00:00:00	Edit Delete
2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02T00:00:00	Edit Delete
3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02T00:00:00	Edit Delete
4	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25T00:00:00	Edit Delete
5	CR-006	Keanu Reeves	reeves.keanu@crowe.com	Project Manager	2020-05-02T00:00:00	Edit Delete

Previous 1 2 Next

Joining Date	Actions
05-02T00:00:00	Edit Delete
05-02T00:00:00	Edit Delete
05-02T00:00:00	Edit Delete
04-25T00:00:00	Edit Delete
05-02T00:00:00	Edit Delete

5 CR-006 Keanu Reeves reeves.keanu@crowe.com

Previous 1 2 Next

Landing Page without Description.

The screenshot shows a web browser window with the URL `localhost:44324/EmployeeList.html`. The page has a header with the Crowe logo and the title "Crowe Employee Management Portal". Below the header is a green bar with the text "Manage Employees". Underneath is a search bar with the placeholder "Search for names..." and a blue button with a plus sign. Below the search bar is a table with the following data:

S.No	Employee Code	Name	Email	Designation	Joining Date	Actions
1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02T00:00:00	Edit Delete
2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02T00:00:00	Edit Delete
3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02T00:00:00	Edit Delete
4	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25T00:00:00	Edit Delete
5	CR-006	Keanu Reeves	reeves.keanu@crowe.com	Project Manager	2020-05-02T00:00:00	Edit Delete

Below the table is a pagination bar with the text "Previous", "1", "2", and "Next".

Add Employee Page.

The screenshot shows a web browser window with the URL `localhost:44324/AddEmployee.html`. The page has a header with the Crowe logo and the title "Crowe Employee Management Portal". Below the header is a green bar with the text "Register Employee". Underneath is a blue button labeled "Manage Employees". Below the button is a form with the following fields:

- Employee Code: CR-012
- Name:
- Email:
- Designation:
- Joining Date: mm/dd/yyyy --:-- --
- Permanent Address: ☐
- Communication Address: ☐

At the bottom of the form is a blue button labeled "Add Employee".

The screenshot shows the same Add Employee page as above, but with the following data filled in:

- Employee Code: CR-012
- Name: Rahul Kumar
- Email: rahul.dravin@gmail.com
- Designation: Business Analyst
- Joining Date: 05/02/2021 03:15 PM
- Permanent Address: ☒ Indore
- Communication Address: ☐

The "Add Employee" button is still at the bottom.

Register Employee

Manage Employees

CR-012

Rahul Kumar

rahul.dravin@gmail.com

Business Analyst


05/02/2021 03:15 PM

Permanent Address ☒

Indore

Communication Address ☐

Add Employee



Employee Added Successfully!

OK

- Programability
- Service Broker
- Storage
- Security
- EnrollmentSystem
- test
- WebApiTrainingDb
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability

	Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 998****47
4	5	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	
5	11	CR-006	Keanu Reeves	reeves.keanu@crowe.com	Project Manager	2020-05-02	Chicago	Phone: 8678****47
6	12	CR-012	Rahul Kumar	rahul.dravin@gmail.com	Business Analyst	2021-05-02	Indore	

Now we will see update functionality.

Update Employee

Manage Employees

11

CR-006

Keanu Reeves

reeves.keanu@crowe.com

Manager

2020-05-02T00:00:00

Chicago

Phone: 88678****47

Update Employee

Manage Employees

11

CR-006

Keanu Reeves

reeves.keanu@crowe.com


Manager

2020-05-02T00:00:00

Chicago

Phone: 88678****47

Update Employee



Record Updated Successfully!

OK

Check in Database.

	Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
3	3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 998****47
4	5	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	
5	11	CR-006	Keanu Reeves	reeves.keanu@crowe.com	Manager	2020-05-02	Chicago	Phone: 8678****47
6	12	CR-012	Rahul Kumar	rahul.dravin@gmail.com	Business Analyst	2021-05-02	Indore	

Now, we will see Delete functionality.

Manage Employees						
Search for names..						
S.No	Employee Code	Name	Email	Designation	Joining Date	Actions
1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02T00:00:00	Edit Delete
2	CR-002	Mukund Kumar	mukund.kumar121@crowe.com	Manager	2020-05-02T00:00:00	Edit Delete
3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02T00:00:00	Edit Delete
4	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25T00:00:00	Edit Delete
5	CR-006	Keanu Reeves	reeves.keanu@crowe.com	Manager	2020-05-02T00:00:00	Edit Delete
Previous 1 2 Next						

Suppose We want to delete this record

Delete Employee

Manage Employees

2

CR-002

Mukund Kumar

mukund.kumar121@crowe.com

Manager

2020-05-02T00:00:00

Bhagat Colony Gali No 14

Phone: 798****47

Delete Employee

Delete Employee

Manage Employees

2

CR-002

Mukund Kumar

mukund.kumar121@crowe.com


Manager

2020-05-02T00:00:00

Bhagat Colony Gali No 14

Phone: 798****47

Delete Employee



Are you sure?

Once deleted, you will not be able to recover this record!

Cancel OK

Delete Employee

Manage Employees

2

CR-002

Mukund Kumar

mukund.kumar121@crowe.com


Manager

2020-05-02T00:00:00

Bhagat Colony Gali No 14

Phone: 798****47

Delete Employee



Your Data is Safe.

OK

Delete Employee

Manage Employees

2

CR-002

Mukund Kumar

mukund.kumar121@crowe.com

Manager

2020-05-02T00:00:00

Bhagat Colony Gali No 14

Phone: 798****47

Delete Employee

✓

Record Deleted Successfully!

Ok

Manage Employees

+

S.No	Employee Code	Name	Email	Designation	Joining Date	Actions
1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02T00:00:00	Edit Delete
2	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02T00:00:00	Edit Delete
3	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25T00:00:00	Edit Delete
4	CR-006	Keanu Reeves	reeves.keanu@crowe.com	Manager	2020-05-02T00:00:00	Edit Delete
5	CR-012	Rahul Kumar	rahul.dravin@gmail.com	Business Analyst	2021-05-02T00:00:00	Edit Delete

Previous
1 2 Next

- 📁 Programmability
- 📁 Service Broker
- 📁 Storage
- 📁 Security
- 📁 EnrollmentSystem
- 📁 test
- 📁 WebApiTrainingDb
- 📁 Security
- 📁 Server Objects
- 📁 Replication
- 📁 OnlyData

100 %
Results Messages

	Id	EmployeeCode	Name	Email	Designation	JoiningDate	PermanentAddress	CommunicationAddress
1	1	CR-001	Anuj Shukla	anuj.shukla13@crowe.com	Vice President	2020-05-02	Bhagat Colony Gali No 14	Phone: 798****47
2	3	CR-003	Arun Yadav	arun.yadav@crowe.com	Software Architect Manager	2020-05-02	Bhagat Colony Gali No 14	Phone: 998****47
3	5	CR-005	Robert Downey, Jr.	tony.stark@ironman.com	Director Technology	2021-04-25	Indore	
4	11	CR-006	Keanu Reeves	reeves.keanu@crowe.com	Manager	2020-05-02	Chicago	Phone: 88678****47
5	12	CR-012	Rahul Kumar	rahul.dravin@gmail.com	Business Analyst	2021-05-02	Indore	

CHNA8RP8222\MSSQLSERVER2019.EmployeeManagementSystem - dbo.Employees - Microsoft SQL Server Management Studio

File Edit View Project Table Designer Tools Window Help

EmployeeManagementSystem Execute

Object Explorer

- Connect
- CHNA8RP8222\MSSQLSERVER2019 (SQL Server 15.0.2000.5)
 - Databases
 - System Databases
 - Database Snapshots
 - EmployeeDB
 - EmployeeManagementSystem
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.Employees
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Service Broker
 - Storage
 - Security
 - EnrollmentSystem
 - test

CHNA8RP8222\MSSQL...- dbo.Employees SQLQuery1.sql - C:\IZEK\KumarP2 (69)

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
EmployeeCode	varchar(100)	<input type="checkbox"/>
Name	varchar(100)	<input type="checkbox"/>
Email	varchar(50)	<input type="checkbox"/>
Designation	varchar(50)	<input type="checkbox"/>
JoiningDate	date	<input type="checkbox"/>
PermanentAddress	varchar(100)	<input checked="" type="checkbox"/>
CommunicationAddress	varchar(100)	<input checked="" type="checkbox"/>

Future Scope of application:

This system is very flexible so that the maintenance and further amendments based on the changing environment and requirements can be made easily. Any changes that may lead to system failures are prevented with security measures. The project will support a multi-user environment, which is more than one user can access simultaneously.

It can be further developed to include more operations and analysis, as changes are required in the system to adapt to the external developments. Further enhancements can be made to the system at any later point in time. Coding procedures can be modified according to the needs of the user. The system code is also well designed that it will form the basis for further enhancement and also new operations can be included in the system. The reports can be represented in all necessary protection. Added options can be designed in reports.

<https://www.google.com/>

<https://dotnet.microsoft.com/apps/aspnet>

<https://www.entityframeworktutorial.net/what-is-entityframework.aspx>

<https://www.w3schools.com/>

<https://stackoverflow.com/>

<https://www.tutorialsteacher.com/webapi/web-api-tutorials>

Home.html code:

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>EMS</title>
  <link rel="stylesheet" href="./Resources/bootstrap-5.0.0-beta3-
dist/css/bootstrap.min.css">
  <style>
    #getInBtn:hover {
      background-color:aquamarine;
    }
  </style>
</head>
<body style="background-color: antiquewhite;">
  <nav class="navbar navbar-expand-lg navbar-light" style="background-
color:antiquewhite">
    <a class="navbar-brand" href="./Home.html"></a>
    <div class="d-flex" style="margin-left: 60%">
      <h3>Crowe Employee Management Portal</h3>
    </div>
  </nav>

  <center>
    <h1 style="position: absolute;top: 50%; left: 50%; transform: translate(-50%, -
50%); font-size: 50px;">Welcome to<br>Employee Management System</h1>

  </center>
  <button id="getInBtn" class="btn btn-lg"
onclick="window.location.href='./EmployeeList.html'" style="border-color:
black;height: 50px; width: 120px ; font-weight:700 ;position: absolute;top: 70%; left:
50%; transform: translate(-50%,-50%)">Get In</button>
</body>
</html>

```

EmployeeList.html Code:

```
<!DOCTYPE html>
<html lang="en">
<head>

  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>EmployeeList</title>

  <!--BOOTSTRAP-->
  <link rel="stylesheet" href="./Resources/bootstrap-5.0.0-beta3-
dist/css/bootstrap.min.css">

  <!--Bootstrap CDN for popper-->
  <link rel="stylesheet"
href="https://pro.fontawesome.com/releases/v5.10.0/css/all.css" integrity="sha384-
AYmEC3Yw5cVb3ZcuHtOA93w35dYTsvhLPVnYs9eStHfGJvOvKxVfELGroGkvsg+p"
crossorigin="anonymous" />
  <style>
    #myInput {
      background-position: 10px 10px;
      background-repeat: no-repeat;
      width: 30%;
      font-size: 16px;
      padding: 12px 20px 12px 40px;
      border: 1px solid #ddd;
      margin-bottom: 12px;
    }

    a,
    a:link,
    a:visited,
    a:hover,
    a:active {
      text-decoration: none;
    }
  </style>
  <!--jQuery-->
  <script src="./Resources/jquery-3.6.0.js"></script>

  <!--Table Sorter jQuery-->
  <script type="text/javascript" src="./Resources/Mottie-tablesorter-
08bf513/js/jquery.tablesorter.js"></script>
  <script type="text/javascript" src="./Resources/Mottie-tablesorter-
08bf513/js/jquery.tablesorter.widgets.js"></script>

  <!--JavaScript for EmployeeList-->
  <script src="Resources/js/emp.js"></script>

</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-light" style="background-
color:antiquewhite">
    <a class="navbar-brand" href="./Home.html"></a>
    <div class="d-flex" style="margin-left: 60%">
      <h3>Crowe Employee Management Portal</h3>
    </div>

  </nav>
```

```

    <div class="alert alert-success" role="alert" style="margin-top: 20px; margin-
bottom: 20px;">
        <center><h4 class="alert-heading">Manage Employees</h4></center>
    </div>
    <div class="d-flex justify-content-center">
        <div class="spinner-border" role="status" id="loading">
            <span class="visually-hidden">Loading...</span>
        </div>
    </div>
    <div style="width: 70%; margin-left: auto; margin-right: auto;">
        <!--Search Input box-->
        <input type="text" id="myInput" onkeyup="searchFunction()" placeholder="Search
for names.." title="Type in a name">
        <button id="addEmpBtn" onclick="window.location.href='./AddEmployee.html'"
class="btn btn-primary" style="height: 52px" title="Add New Employee"><i class="fas
fa-plus-circle fa-lg"></i></button>
        <!--Employee List-->
        <table id="empList" class="table tablesorter">
            <thead class="thead-dark">
                <tr class="table-active">
                    <th scope="col">Employee Code</th>
                    <th scope="col">Name</th>
                    <th scope="col">Email</th>
                    <th scope="col">Designation</th>
                    <th scope="col">Joining Date</th>
                    <th scope="col">Actions</th>
                </tr>
            </thead>
            <tbody></tbody>
        </table>
        <div class="pagination-container">
            <ul class="pagination"></ul>
        </div>
    </div>
</div>
</body>
</html>

```

Emp.js Code:

```
const api_url = "https://localhost:44324/api/employee";

async function getapi(url) {
  const response = await fetch(url);
  var data = await response.json();
  console.log(data);
  if (response) {
    hideloader();
  }
  show(data);

  $("#emplist").tablesorter();

  $(function () {
    getpagination(5);
  });
  //Pagination Script will run when the table data is loaded
  var table = '#emplist';
  $('#maxRows').on('change', function () {
    var maxrow = parseInt($(this).val());
    if (parseInt($(this).val()) == 5000) {
      maxrow = 10;
    }
    getpagination(maxrow);
  });

  function activePage() {
    var $el = $('[data-page].active');

    if ($el.length) {
      return $el.data('page');
    }

    return 0; // just needs to not exist so finder fails
  }

  $(function () {
    $('.prev-btn, .next-btn').on('click', function (e) {
      e.preventDefault();
      e.stopPropagation(); // prevent the parent event from firing

      var page = activePage() + ($(event.target).hasClass('prev-btn') ? -1 : 1);

      $('[data-page="' + page + '"]').trigger('click');
    });
  })

  function getpagination(maxrows) {
    $('table tr:eq(0)').prepend('<th>S.No</th>');
    var id = 0;
    $('table tr:gt(0)').each(function () {
      id++;
      $(this).prepend('<td> ' + id + '</td>');
    })
    $('.pagination').html('')
    var trnum = 0;
    var maxrows = maxrows;
    var totalrows = $(table)[0].rows.length;
    $(table + ' tr:gt(0)').each(function () {
      trnum++;
      if (trnum > maxrows) {
        $(this).hide()
      }
      if (trnum <= maxrows) {

```

```

        $(this).show();
    }
});
//if(totalrows > maxrows){
var pagenum = Math.ceil(totalrows / maxrows);
$('.pagination').append('<li class="page-item"><a class="page-link prev-btn"
href="#">Previous</a></li>').show();
for (var i = 1; i <= pagenum; i++) {
    $('.pagination').append('<li class="page-item" data-page="' + i + '"><a
class="page-link" href="#">' + i + '</a></li>').show();
}
$('.pagination').append('<li class="page-item"><a class="page-link next-btn"
href="#">Next</a></li>').show();
//}
$('.pagination li:first-child').addClass('active');
$('.pagination li').on('click', function () {
    var pagenum = $(this).attr('data-page');
    var trIndex = 0;
    $('.pagination li').removeClass('active');
    $(this).addClass('active');
    $('table tr:gt(0)').each(function () {
        trIndex++;
        if (trIndex > (maxrows * pagenum) || trIndex <= ((maxrows * pagenum) -
maxrows)) {
            $(this).hide();
        } else {
            $(this).show();
        }
    });
});
}; //end of pagination
}

getapi(api_url);

function hideloader() {
    document.getElementById('loading').style.display = 'none';
}

function show(data) {
    let lastEmployeeId = 0;
    for (let r of data) {
        lastEmployeeId = r;
        var tbodyRef =
document.getElementById('empList').getElementsByTagName('tbody')[0];

        // Insert a row at the end of table
        var newRow = tbodyRef.insertRow();

        // Insert a cell at the end of the row
        var newCell1 = newRow.insertCell();
        var newCell2 = newRow.insertCell();
        var newCell3 = newRow.insertCell();
        var newCell4 = newRow.insertCell();
        var newCell5 = newRow.insertCell();
        var newCell6 = newRow.insertCell();

        // Append a text node to the cell
        var newText1 = document.createTextNode(r.employeeCode);
        var newText2 = document.createTextNode(r.name);
        var newText3 = document.createTextNode(r.email);
        var newText4 = document.createTextNode(r.designation);
        var newText5 = document.createTextNode(r.joiningDate);

        var editAnchor = document.createElement('a');

```



```

editAnchor.href = "../UpdateEmployee.html";

var editIcon = document.createElement('i');
editIcon.className += "fas fa-pencil fa-sm";
editAnchor.appendChild(editIcon);

//<i class="fas fa-pencil fa-lg">
editAnchor.addEventListener('click', function () {
    var empId = r.id;
    localStorage.setItem('empId', empId);

    var empCode = r.employeeCode;
    localStorage.setItem('empCode', empCode);

    var empName = r.name;
    localStorage.setItem('empName', empName);

    var empEmail = r.email;
    localStorage.setItem('empEmail', empEmail);

    var empDesignation = r.designation;
    localStorage.setItem('empDesignation', empDesignation);

    var joinDate = r.joiningDate;
    localStorage.setItem('joinDate', joinDate);

    var permAddr = r.permanentAddress;
    localStorage.setItem('permAddr', permAddr);

    var commAddr = r.communicationAddress;
    localStorage.setItem('commAddr', commAddr);
});
editAnchor.innerHTML += " Edit &nbsp;&nbsp;&nbsp;";

var delAnchor = document.createElement('a');
delAnchor.href = "../DeleteEmployee.html";
delAnchor.style.color = 'red';
var delIcon = document.createElement('i');
delIcon.className += "fas fa-trash fa-sm";
delAnchor.appendChild(delIcon);

delAnchor.addEventListener('click', function () {
    var empId = r.id;
    localStorage.setItem('empId', empId);

    var empCode = r.employeeCode;
    localStorage.setItem('empCode', empCode);

    var empName = r.name;
    localStorage.setItem('empName', empName);

    var empEmail = r.email;
    localStorage.setItem('empEmail', empEmail);

    var empDesignation = r.designation;
    localStorage.setItem('empDesignation', empDesignation);

    var joinDate = r.joiningDate;
    localStorage.setItem('joinDate', joinDate);

    var permAddr = r.permanentAddress;
    localStorage.setItem('permAddr', permAddr);

    var commAddr = r.communicationAddress;

```

```

        localStorage.setItem('commAddr', commAddr);
    });
    delAnchor.innerHTML += " Delete"
    /*var delAnchor = document.createElement('a');
    var linkDel = document.createTextNode(" Delete");
    delAnchor.appendChild(linkDel);*/

    newCell1.appendChild(newText1);
    newCell2.appendChild(newText2);
    newCell3.appendChild(newText3);
    newCell4.appendChild(newText4);
    newCell5.appendChild(newText5);
    newCell6.appendChild(editAnchor);
    newCell6.appendChild(delAnchor);
}
localStorage.setItem('lastEmpId', lastEmployeeId.id);
}

//Script for search
function searchFunction() {
    var input, filter, table, tr, td, i, txtValue;
    input = document.getElementById("myInput");
    filter = input.value.toUpperCase();
    table = document.getElementById("empList");
    tr = table.getElementsByTagName("tr");
    for (i = 0; i < tr.length; i++) {
        td = tr[i].getElementsByTagName("td")[2];
        if (td) {
            txtValue = td.textContent || td.innerText;
            if (txtValue.toUpperCase().indexOf(filter) > -1) {
                tr[i].style.display = "";
            } else {
                tr[i].style.display = "none";
            }
        }
    }
}
}
}

```

AddEmployee.html Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Add Employee</title>
  <link rel="stylesheet" href="./Resources/bootstrap-5.0.0-beta3-
dist/css/bootstrap.min.css">
  <script src="./Resources/jquery-3.6.0.js"></script>
  <script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>
  <script type="text/javascript">

    function SubmitEmpdata() {
      var url = "https://localhost:44324/api/employee/addemployee";
      var reqdata = {
        employeeCode: document.getElementById("empCode").value,
        name: document.getElementById("empName").value,
        email: document.getElementById("empEmail").value,
        designation: document.getElementById("empDesignation").value,
        joiningDate: document.getElementById("joinDate").value,
        permanentAddress: document.getElementById("permAddr").value,
        communicationAddress: document.getElementById("commAddr").value
      }
      var stringReqdata = JSON.stringify(reqdata);

      $.ajax({
        async: false,
        type: "POST",
        url: url,
        data: stringReqdata,
        dataType: "json",
        context: document.body,
        contentType: 'application/json; charset=utf-8',
        success: function (data, textStatus, jqXHR) {
          swal({
            title: "Employee Added Successfully!",
            icon: "success",
            button: "Ok",
          }).then(function (isConfirm) {
            if (isConfirm) {
              location.reload();
              location.href = './EmployeeList.html';
            }
          });
        },
        statusCode: {

          400: function (responseObject, textStatus, jqXHR) {
            swal({
              title: "Check Email Again!",
              icon: "warning",
              button: "Ok",
            }).then(function (isConfirm) {
              if (isConfirm) {
                location.reload();
              }
            });
          },

          409: function (responseObject, textStatus, jqXHR) {
            swal({
              title: "A record with the same Email Id or EmployeeCode is
already present!",
```

```

        icon: "error",
        button: "Ok",
    }).then(function (isConfirm) {
        if (isConfirm) {
            location.reload();
        }
    });
    },
    404: function (responseObject, textStatus, jqXHR) {
        swal({
            title: "Page Not Found!",
            icon: "error",
            button: "Ok",
        }).then(function (isConfirm) {
            if (isConfirm) {
                location.reload();
            }
        });
    });
    }
    });
}

</script>
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-light" style="background-color:antiquewhite">

        <a class="navbar-brand" href="./Home.html"></a>
        <div class="d-flex" style="margin-left: 60%">
            <h3>Crowe Employee Management Portal</h3>
        </div>

    </nav>
    <div class="alert alert-success" role="alert" style="margin-top: 20px; margin-bottom: 20px;">
        <center><h4 class="alert-heading">Register Employee</h4></center>
    </div>
    <div style="width: 70%; margin-left: auto; margin-right: auto; padding: 20px 0px;">
        <button id="addEmpBtn" onclick="window.location.href='./EmployeeList.html'"
class="btn btn-primary" style="height: 52px" title="Employee List">Manage
Employees</button>
        <form>
            <div class="form-group" style="margin-top: 10px">
                <input class="form-control" id="empCode" aria-describedby="ID"
disabled>
            </div>
            <div class="form-group" style="margin-top: 10px">
                <input class="form-control" id="empName" aria-describedby="empName"
placeholder="Name" required>
            </div>
            <div class="form-group" style="margin-top: 10px">
                <input type="email" class="form-control" id="empEmail"
placeholder="Email" required>
            </div>
            <div class="form-group" style="margin-top: 10px">
                <input class="form-control" id="empDesignation" list="designationList"
placeholder="Designation" required>

```

```

        <datalist id="designationList">
            <option>Trainee Engineer</option>
            <option>Software Engineer</option>
            <option>System Analyst</option>
            <option>Programmer Analyst</option>
            <option>Senior Software Engineer</option>
            <option>Project Lead</option>
            <option>Project Manager</option>
            <option>Program Manager</option>
            <option>Architect</option>
            <option>Technical Specialist</option>
            <option>Deliver Manager</option>
            <option>Delivery Head</option>
            <option>Business Analyst</option>
            <option>Director Technology</option>
            <option>Director</option>
            <option>Vice President</option>
            <option>President</option>
            <option>CEO</option>
        </datalist>
    </div>
    <div class="form-group" style="margin-top: 10px">
        <input type="datetime-local" class="form-control" id="joinDate"
placeholder="JoiningDate Date: 0001-01-01T00:00:00" required>
    </div>
    <div class="form-group" style="margin-top: 10px">
        <label for="permAddress">Permanent Address</label>
        <input type="checkbox" id="myCheckPermAddr" onclick="showPermAddr()">
        <input class="form-control" id="permAddr" placeholder="Permanent
Address" style="display: none;" required>
    </div>
    <div class="form-group" style="margin-top: 10px">
        <label for="commAddress">Communication Address</label>
        <input type="checkbox" id="myCheckCommAddr" onclick="showCommAddr()">
        <input class="form-control" id="commAddr" placeholder="Communication
Address" style="display: none;" required>
    </div>

    </form>
    <button onclick="SubmitEmpdata();" id="submitBtn" type="submit" value="Submit"
class="btn btn-primary" style="margin-top: 10px">Add Employee</button>

</div>
<script>
    function showPermAddr() {
        var checkBox = document.getElementById('myCheckPermAddr');
        var permAddrInpt = document.getElementById('permAddr');
        if (checkBox.checked == true) {
            permAddrInpt.style.display = "block";
        } else {
            permAddrInpt.style.display = "none";
            permAddrInpt.value = '';
        }
    }

    function showCommAddr() {
        var checkBox = document.getElementById('myCheckCommAddr');
        var commAddrInpt = document.getElementById('commAddr');
        if (checkBox.checked == true) {
            commAddrInpt.style.display = "block";
        } else {
            commAddrInpt.style.display = "none";
            commAddrInpt.value = '';
        }
    }
}

```

```

var empIdStr = localStorage.getItem('lastEmpId');
localStorage.removeItem('lastEmpId');
var lastEmpId = parseInt(empIdStr)+1;
var a = ('00' + lastEmpId).slice(-3)

var getLastEmpIdInc = "CR-" + a;
document.getElementById('empCode').value = getLastEmpIdInc;

</script>
</body>
</html>

```

UpdateEmployee.html Code:

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Update Employee</title>
  <link rel="stylesheet" href="./Resources/bootstrap-5.0.0-beta3-
dist/css/bootstrap.min.css">
  <!-- <link rel="stylesheet" href="Resources/bootstrap-5.0.0-beta3-
dist/js/bootstrap.bundle.min.js"/>-->
  <!-- JavaScript Bundle with Popper -->
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
JEW9xMcG8R+phH31jmWH6WNP0WintQrMb4s7Z0dauHnUtxwoG2vI5DkLtS3qm9Ekf"
crossorigin="anonymous"></script>
  <script src="Resources/jquery-3.6.0.js"></script>
  <script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>
  <script src="Resources/js/emp.js"></script>
  <script type="text/javascript">
    function updateChange() {
      var url = "https://localhost:44324/api/employee/modifyemployee";

      var reqdata = {
        id: document.getElementById("empId").value,
        employeeCode: document.getElementById("empCode").value,
        name: document.getElementById("empName").value,
        email: document.getElementById("empEmail").value,
        designation: document.getElementById("empDesignation").value,
        joiningDate: document.getElementById("joinDate").value,
        permanentAddress: document.getElementById("permAddr").value,
        communicationAddress: document.getElementById("commAddr").value
      }
      var stringReqdata = JSON.stringify(reqdata);
      jQuery.ajax({
        async: false,
        type: "PUT",
        url: url,
        data: stringReqdata,
        dataType: "json",
        context: document.body,
        contentType: 'application/json; charset=utf-8',
        success: function (data, textStatus, jqxhr) {
          swal({
            title: "Record Updated Successfully!",
            icon: "success",
            button: "Ok",
          }).then(function (isConfirm) {
            if (isConfirm) {
              location.reload();
            }
          });
        }
      });
    }
  </script>

```

```

        location.href = './EmployeeList.html';
    }
    });
},
error: function (jqXHR, textStatus, errorThrown) {
    console.log(errorThrown);
}
});
}
</script>
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-light" style="background-
color:antiquewhite">
        <a class="navbar-brand" href="./Home.html"></a>
        <div class="d-flex" style="margin-left: 60%">
            <h3>Crowe Employee Management Portal</h3>
        </div>
    </nav>

    <div class="alert alert-success" role="alert" style="margin-top: 20px; margin-
bottom: 20px;">
        <center><h4 class="alert-heading">Update Employee</h4></center>
    </div>

    <div style="width: 70%; margin-left: auto; margin-right: auto; padding: 20px
0px;">
        <div class="form-group">
            <button id="addEmpBtn"
onclick="window.location.href='./EmployeeList.html'" class="btn btn-group btn-primary"
style="margin-bottom:10px" title="Employee List">Manage Employees</button>
        </div>
        <form id="updateForm">
            <div class="form-group">
                <input class="form-control" id="empId" aria-describedby="ID" disabled>
            </div>
            <div class="form-group" style="margin-top: 10px">
                <input class="form-control" id="empCode" aria-describedby="ID"
placeholder="Employee Code: CR-00n" disabled>
            </div>
            <div class="form-group" style="margin-top: 10px">
                <input class="form-control" id="empName" aria-describedby="empName"
placeholder="Name" disabled>
            </div>
            <div class="form-group" style="margin-top: 10px">
                <input type="email" class="form-control" id="empEmail"
placeholder="Email" required>
            </div>
            <div class="form-group" style="margin-top: 10px">
                <input class="form-control" id="empDesignation" list="designationList"
placeholder="Designation" value="Manager" required>
                <datalist id="designationList">
                    <option>Trainee Engineer</option>
                    <option>Software Engineer</option>
                    <option>System Analyst</option>
                    <option>Programmer Analyst</option>
                    <option>Senior Software Engineer</option>
                    <option>Project Lead</option>
                    <option>Project Manager</option>
                    <option>Program Manager</option>
                    <option>Architect</option>
                    <option>Technical Specialist</option>
                </datalist>
            </div>
        </form>
    </div>

```

```

        <option>Deliver Manager</option>
        <option>Delivery Head</option>
        <option>Business Analyst</option>
        <option>Director Technology</option>
        <option>Director</option>
        <option>Vice President</option>
        <option>President</option>
        <option>CEO</option>
    </datalist>
</div>
<div class="form-group" style="margin-top: 10px">
    <input class="form-control" id="joinDate" placeholder="JoiningDate
Date: 0001-01-01T00:00:00" disabled>
</div>
<div class="form-group" style="margin-top: 10px">
    <input class="form-control" id="permAddr" placeholder="Permanent
Address" disabled>
</div>
<div class="form-group" style="margin-top: 10px">
    <input class="form-control" id="commAddr" placeholder="Communication
Address" required>
</div>
</form>
<button onclick="updateChange();" id="submitBtn" type="submit" value="Submit"
class="btn btn-primary" style="margin-top: 10px">Update Employee</button>
</div>
<script>
    document.getElementById('empId').value = localStorage.getItem('empId');

    document.getElementById('empCode').value =
localStorage.getItem('empCode');

    document.getElementById('empName').value =
localStorage.getItem('empName');

    document.getElementById('empEmail').value =
localStorage.getItem('empEmail');

    document.getElementById('empDesignation').value =
localStorage.getItem('empDesignation');

    document.getElementById('joinDate').value =
localStorage.getItem('joinDate');

    document.getElementById('permAddr').value =
localStorage.getItem('permAddr');

    document.getElementById('commAddr').value =
localStorage.getItem('commAddr');
</script>
</body>
</html>

```


EmployeeRepository.cs Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using EmpManagementSystem.Data;
using EmpManagementSystem.Models;

namespace EmpManagementSystem.Repositories
{
    public class EmployeeRepository : IEmployeeRepository
    {
        private readonly EmpManagementSystemDbContext _context;

        public EmployeeRepository(EmpManagementSystemDbContext context)
        {
            this._context = context;
        }

        public IEnumerable<Employee> GetEmployees()
        {
            return this._context.Employees.ToList();
        }

        public Employee GetEmployee(int employeeId)
        {
            return this._context.Employees.FirstOrDefault(e => e.Id ==
employeeId);
        }

        public Employee AddEmployee(Employee employee)
        {
            var checkEmail = this._context.Employees.FirstOrDefault(e =>
e.Email == employee.Email);
            var checkEmpCode = this._context.Employees.FirstOrDefault(e =>
e.EmployeeCode == employee.EmployeeCode);
            if(checkEmail != null || checkEmpCode != null)
            {
                return null;
            }

            var result = this._context.Employees.Add(employee);

            _context.SaveChanges();
            return result;
        }

        public Employee UpdateEmployee(Employee employee)
        {
            //var checkEmail = this._context.Employees.FirstOrDefault(e =>
e.Email == employee.Email);
            //if (checkEmail != null)
            //{
            //    return null;
            //}
            var result = _context.Employees.FirstOrDefault(e => e.Id ==
employee.Id);
            if (result != null)
            {
                result.EmployeeCode = employee.EmployeeCode;
                result.Name = employee.Name;
                result.Email = employee.Email;
                result.Designation = employee.Designation;
                result.JoiningDate = employee.JoiningDate;
                result.PermanentAddress = employee.PermanentAddress;
                result.CommunicationAddress =
employee.CommunicationAddress;
            }
        }
    }
}
```

```

        _context.SaveChanges();

        return result;
    }

    return null;
}

public bool DeleteEmployee(int employeeId)
{
    var result = _context.Employees.FirstOrDefault(e => e.Id ==
employeeId);
    if (result != null)
    {
        _context.Employees.Remove(result);
        _context.SaveChanges();
        return true;
    }

    return false;
}

}

public interface IEmployeeRepository
{
    IEnumerable<Employee> GetEmployees();

    Employee GetEmployee(int employeeId);

    Employee AddEmployee(Employee employee);

    Employee UpdateEmployee(Employee employee);

    bool DeleteEmployee(int employeeId);
}
}

```

EmployeeManagementDbContext.cs Code:

```

using System.Data.Entity;
using EmpManagementSystem.Models;

namespace EmpManagementSystem.Data
{
    public class EmpManagementSystemDbContext : DbContext
    {
        public EmpManagementSystemDbContext()
            : base("name=EmployeeManagementSystemDb")
        {
        }

        public DbSet<Employee> Employees { get; set; }
    }
}

```

EmployeeController.cs Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
using System.Web.Services.Description;
using EmpManagementSystem.Data;
using EmpManagementSystem.Models;
using EmpManagementSystem.Repositories;

namespace EmpManagementSystem.Controllers
{
    public class EmployeeController : ApiController
    {
        private readonly IEmployeeRepository _employeeRepository;

        public EmployeeController()
        {
            this._employeeRepository = new EmployeeRepository(new
EmpManagementSystemDbContext());
        }

        // GET: Employee
        public IHttpActionResult GetEmployees()
        {
            IEnumerable<Employee> employees = null;

            employees = this._employeeRepository.GetEmployees().ToList();

            if (!employees.Any())
            {
                return NotFound();
            }

            return Ok(employees);
        }

        // GET: Employee
        public IHttpActionResult GetEmployee(int id)
        {
            Employee employee = null;
            employee = this._employeeRepository.GetEmployee(id);

            if (employee == null)
            {
                return NotFound();
            }

            return Ok(employee);
        }

        [HttpPost]
        public IHttpActionResult AddEmployee([FromBody]Employee employee)
        {
            if (!ModelState.IsValid)
                return BadRequest(ModelState);

            employee = this._employeeRepository.AddEmployee(employee);
            Console.WriteLine(employee);
            if (employee != null)
            {
                return Ok(employee);
            }
        }
    }
}
```

```

        //BadRequest("There is an existing result with this Email or
Employee Code")
        return Conflict();
    }

    [HttpPut]
    public IHttpActionResult ModifyEmployee(Employee employee)
    {
        if (!ModelState.IsValid)
            return BadRequest("Invalid data.");

        employee = this._employeeRepository.UpdateEmployee(employee);

        return Ok(employee);
    }

    [HttpDelete]
    public IHttpActionResult DeleteEmployee(Employee employee)
    {
        if (employee.Id <= 0)
            return BadRequest("Not a valid student id");

        var isDeleted =
this._employeeRepository.DeleteEmployee(employee.Id);
        if (isDeleted)
        {
            return Ok(employee);
        }
        else
        {
            return BadRequest("Invalid data");
        }
    }
}
}

```

WebApiConfig.cs Code:

```

using System.Net.Http.Formatting;
using System.Web.Http;
using Newtonsoft.Json.Serialization;

namespace EmpManagementSystem
{
    public static class WebApiConfig
    {
        public static void Register(HttpConfiguration config)
        {
            // Web API configuration and services

            config.Routes.MapHttpRoute(
                name: "DefaultApi",
                routeTemplate: "api/{controller}/{id}",
                defaults: new { id = RouteParameter.Optional }
            );

            JsonMediaTypeFormatter jsonMediaTypeFormatter =
config.Formatters.JsonFormatter;
            jsonMediaTypeFormatter.SerializerSettings.ContractResolver = new
CamelCasePropertyNameContractResolver();
        }
    }
}

```