**1. Ruby on Rails**

Prerequisites: Ruby on Rails installed, MySQL server running locally.

Steps:

1. Install the mysql2 gem:

bundle add mysql2

1. Configure config/database.yml:

default: &default

adapter: mysql2

encoding: utf8mb4

pool: 5

username: root

password: pass

host: localhost

port: 3306

development:

<<: \*default

database: company

1. Create an Employee model:

rails generate model Employee

1. Run the query in a controller or Rails console:

employees = Employee.all

employees.each { |emp| puts emp.inspect }

**2. Node.js**

Prerequisites: Node.js installed, MySQL server running locally.

Steps:

1. Install the mysql2 package:

npm install mysql2

1. Create a connection and execute the query:

const mysql = require('mysql2/promise');

async function queryEmployees() {

const connection = await mysql.createConnection({

host: 'localhost',

user: 'root',

password: 'pass',

database: 'company',

port: 3306

});

const [rows] = await connection.execute('SELECT \* FROM employee');

console.log(rows);

await connection.end();

}

queryEmployees();

**3. Blazor (C#)**

Prerequisites: NET SDK installed, MySQL server running locally, Pomelo.EntityFrameworkCore.MySql NuGet package.

Steps:

1. Configure the database context in Program.cs:

builder.Services.AddDbContext<CompanyContext>(options =>

options.UseMySql(

"server=localhost;port=3306;user=root;password=pass;database=company",

ServerVersion.AutoDetect("server=localhost;port=3306;user=root;password=pass;database=company")

));

1. Define the Employee model and CompanyContext:

public class Employee

{

public int Id { get; set; }

public string Name { get; set; }

}

public class CompanyContext : DbContext

{

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

}

1. Run the query in a Razor component or service:

@inject CompanyContext \_context

@code {

private List<Employee> employees = new();

protected override async Task OnInitializedAsync()

{

employees = await \_context.Employees.ToListAsync();

}

}

**4. ASP.NET**

Prerequisites: .NET SDK installed, MySQL server running locally, Pomelo.EntityFrameworkCore.MySql NuGet package.

Steps:

1. Configure the connection in appsettings.json:

{

"ConnectionStrings": {

"CompanyDB": "server=localhost;port=3306;user=root;password=pass;database=company"

}

}

1. Define the Employee model and CompanyContext:

public class Employee

{

public int Id { get; set; }

public string Name { get; set; }

}

public class CompanyContext : DbContext

{

public CompanyContext(DbContextOptions<CompanyContext> options) : base(options) { }

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

}

1. Run the query in a controller:

[ApiController]

[Route("[controller]")]

public class EmployeeController : ControllerBase

{

private readonly CompanyContext \_context;

public EmployeeController(CompanyContext context)

{

\_context = context;

}

[HttpGet]

public IEnumerable<Employee> Get()

{

return \_context.Employees.ToList();

}

}

**5. Python with Flask**

Prerequisites: Python installed, MySQL server running locally.

Steps

1. Install packages:

pip install Flask mysql-connector-python

1. Create a Flask app:

from flask import Flask

import mysql.connector

app = Flask(\_\_name\_\_)

@app.route('/employees')

def get\_employees():

connection = mysql.connector.connect(

host='localhost',

user='root',

password='pass',

database='company',

port=3306

)

cursor = connection.cursor(dictionary=True)

cursor.execute('SELECT \* FROM employee')

employees = cursor.fetchall()

cursor.close()

connection.close()

return {'employees': employees}

if \_\_name\_\_ == '\_\_main\_\_':

app.run()