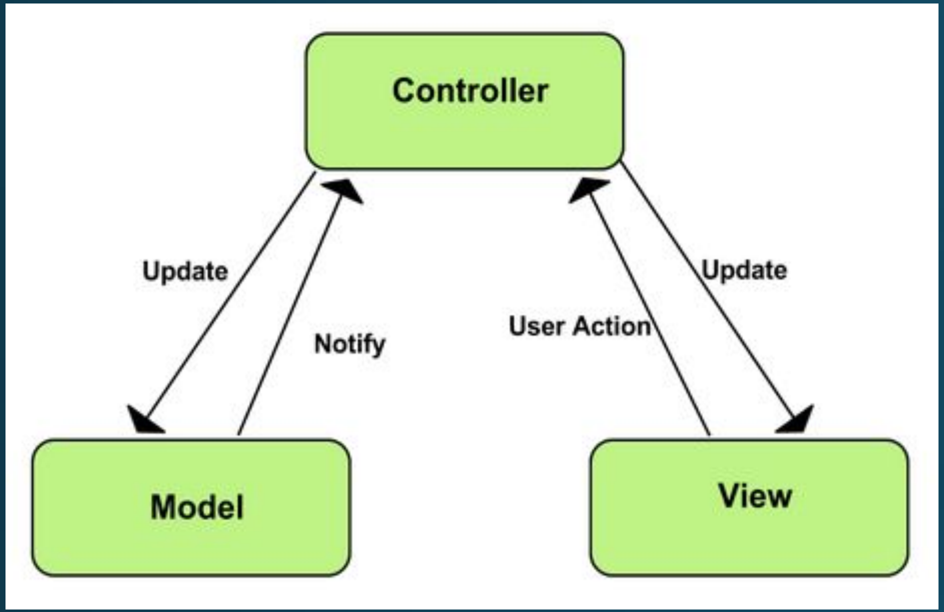


MVC Architecture

What is it?

- MVC Framework is an organizational structure within programming focused on compartmentalizing aspects of an application into three main components:
  - M – Model
  - V – View
  - C – Controller
- Each component is built to handle specific development aspect of an application.

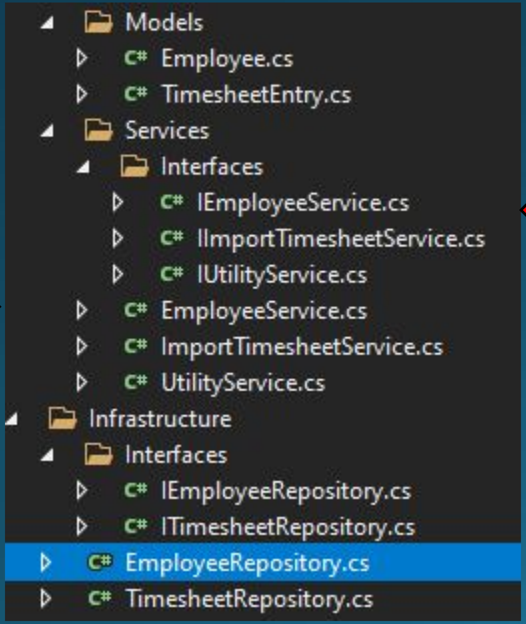
Model (M)

- The Model component handles the data and data structures of an application. This includes the objects that the code will act upon along with their related logic. Model objects are generally supported by an interface and are updated by the controller.

Ex:

Model Object Class:
Employee.cs

Supporting Interface:
EmployeeService.cs



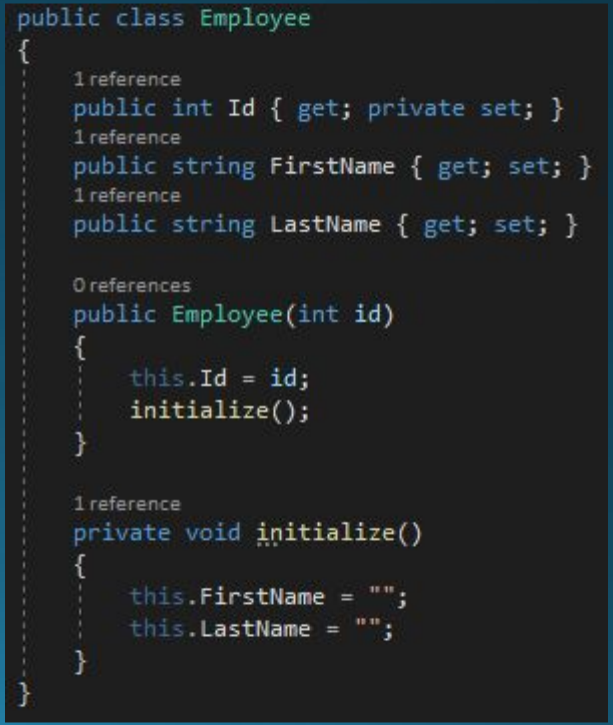
```
◢ Models
   ▷ C# Employee.cs
   ▷ C# TimesheetEntry.cs
◢ Services
   ◢ Interfaces
      ▷ C# IEmployeeService.cs
      ▷ C# IImportTimesheetService.cs
      ▷ C# IUtilityService.cs
   ▷ C# EmployeeService.cs
   ▷ C# ImportTimesheetService.cs
   ▷ C# UtilityService.cs
◢ Infrastructure
   ◢ Interfaces
      ▷ C# IEmployeeRepository.cs
      ▷ C# ITimesheetRepository.cs
   ▷ C# EmployeeRepository.cs
   ▷ C# TimesheetRepository.cs
```

Supporting Interface Prototype:
IEmployeeService.cs

Model Object Class:
Employee.cs

Object Members:

1. Class Variables
2. Constructor
3. Object Methods

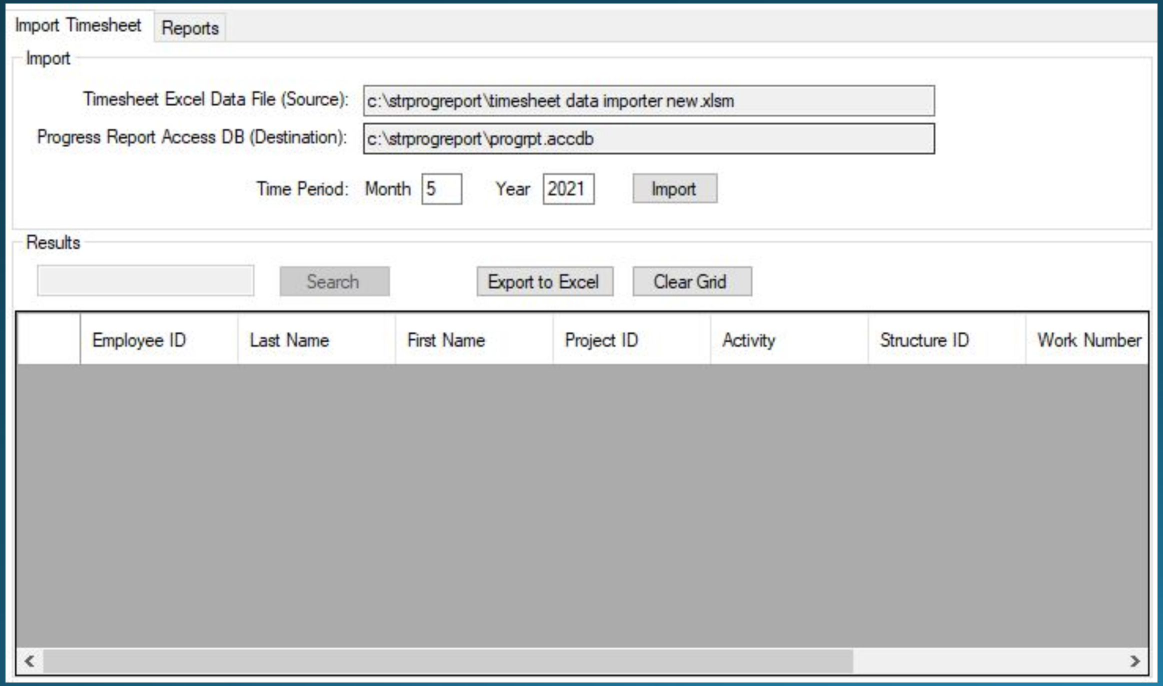


```
public class Employee
{
    1 reference
    public int Id { get; private set; }
    1 reference
    public string FirstName { get; set; }
    1 reference
    public string LastName { get; set; }

    0 references
    public Employee(int id)
    {
        this.Id = id;
        initialize();
    }

    1 reference
    private void initialize()
    {
        this.FirstName = "";
        this.LastName = "";
    }
}
```

View (V)

- The View component refers to the frontend or UI component of the application. It is the way the program is presented to the user along with its supporting code. It takes input from the user and passes it to the controller.

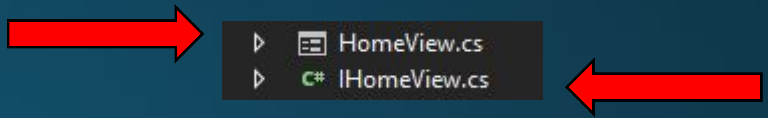
Ex:



Timesheet Excel Data File (Source): c:\strprogreport\timesheet data importer new.xlsm

Progress Report Access DB (Destination): c:\strprogreport\progrpt.accdb

Time Period: Month 5 Year 2021 Import



Application View: Visible UI containing all interactable elements

IHomeView: Supporting code for UI. Contains the behavior code for button and field elements along with any supporting methods

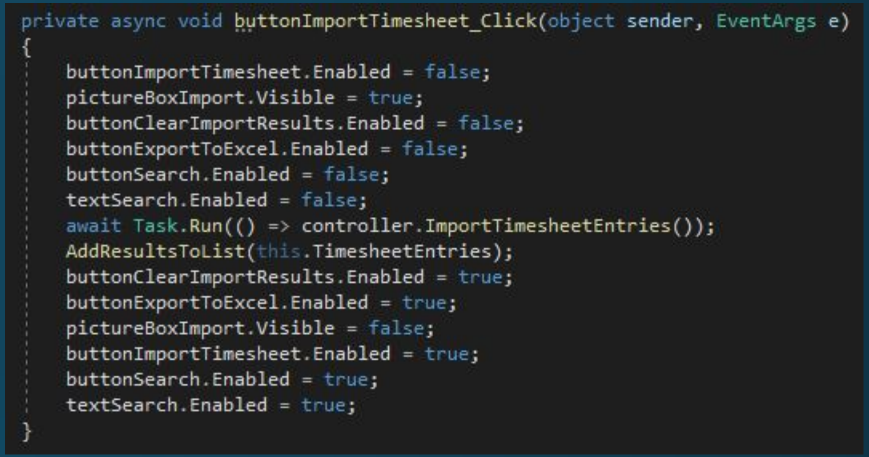
Controller (C)

- The Controller component acts as an intermediary between the Model and View components. It handles user input and updates the other components accordingly. Overall, it handles the general control flow of the application.

Controller Flow Example

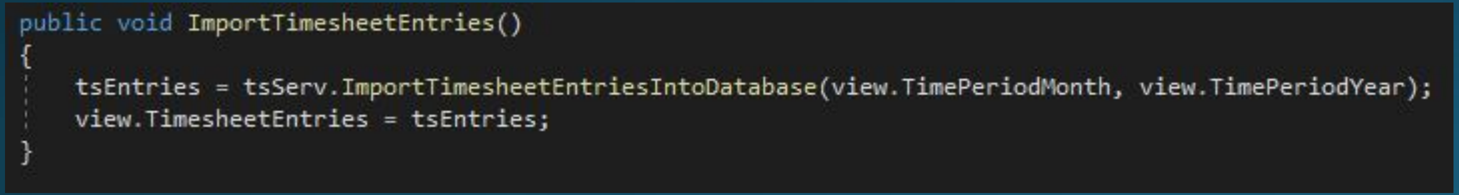
1. Import button pressed by user in the View UI

2. Import button method called in the View support code



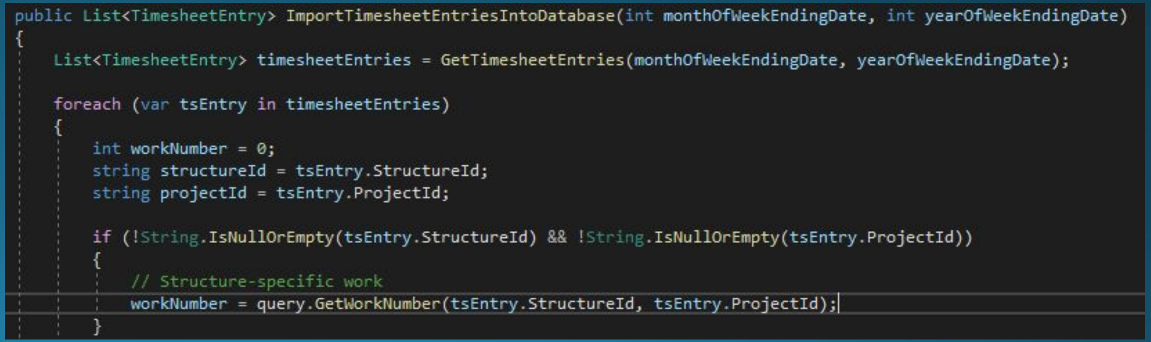
```
private async void buttonImportTimesheet_Click(object sender, EventArgs e)
{
    buttonImportTimesheet.Enabled = false;
    pictureBoxImport.Visible = true;
    buttonClearImportResults.Enabled = false;
    buttonExportToExcel.Enabled = false;
    buttonSearch.Enabled = false;
    textSearch.Enabled = false;
    await Task.Run(() => controller.ImportTimesheetEntries());
    AddResultsToList(this.TimesheetEntries);
    buttonClearImportResults.Enabled = true;
    buttonExportToExcel.Enabled = true;
    pictureBoxImport.Visible = false;
    buttonImportTimesheet.Enabled = true;
    buttonSearch.Enabled = true;
    textSearch.Enabled = true;
}
```

3. View calls the ImportTimesheetEntries() method from the controller



```
public void ImportTimesheetEntries()
{
    tsEntries = tsServ.ImportTimesheetEntriesIntoDatabase(view.TimePeriodMonth, view.TimePeriodYear);
    view.TimesheetEntries = tsEntries;
}
```

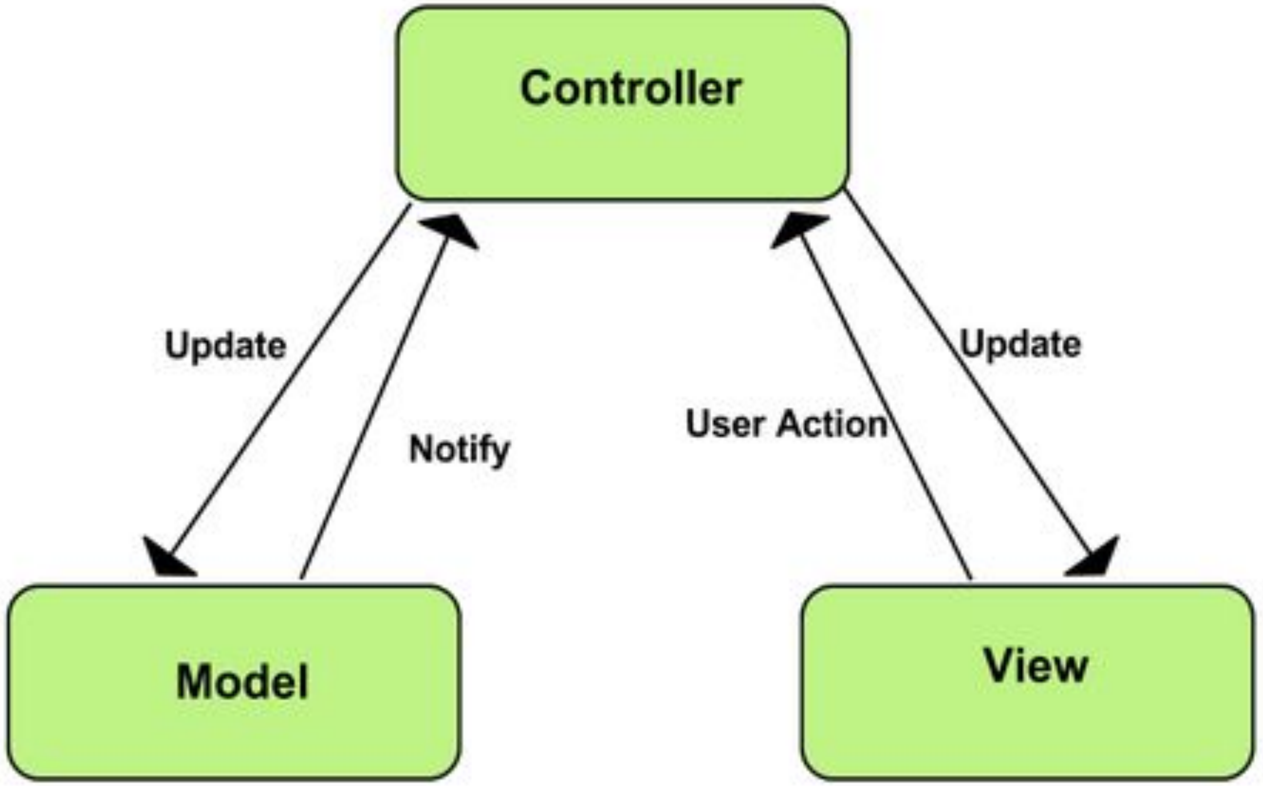
4. The Controller calls the Model interface method to execute the final button behavior



```
public List<TimesheetEntry> ImportTimesheetEntriesIntoDatabase(int monthOfWeekEndingDate, int yearOfWeekEndingDate)
{
    List<TimesheetEntry> timesheetEntries = GetTimesheetEntries(monthOfWeekEndingDate, yearOfWeekEndingDate);

    foreach (var tsEntry in timesheetEntries)
    {
        int workNumber = 0;
        string structureId = tsEntry.StructureId;
        string projectId = tsEntry.ProjectId;

        if (!String.IsNullOrEmpty(tsEntry.StructureId) && !String.IsNullOrEmpty(tsEntry.ProjectId))
        {
            // Structure-specific work
            workNumber = query.GetWorkNumber(tsEntry.StructureId, tsEntry.ProjectId);
        }
```

MVC Overview

1. The View takes in input from the user via the UI or any manner of presentation.
2. The View passes the input to the Controller.
3. The Controller interprets the input and updates the Models and/or View through the application control flow.



MVC Example

MVC Example

View: User Interface

Controller: Engine

Model: Fuel

Pros and Cons

- Pros:
  - Simplifies code maintenance
  - Decreases complexity of individual components
  - Allows for simpler testing of individual objects
  - Improves overall code compartmentalization and efficiency
- Cons:
  - Compartmentalization increases complexity of code tracing
  - Increased overall testing complexity

Short Readings

- MVC Architecture – Chrome Developers
  - https://developer.chrome.com/docs/apps/app_frameworks/
- MVC Tutorial for Beginners: What is, Architecture & Example – Guru99
  - https://www.guru99.com/mvc-tutorial.html#:~:text=The%20Model%2DView%2DController%20(,development%20aspect%20of%20an%20application.
- MVC Framework – Introduction – tutorialspoint
  - https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm

Short Videos

- MVC Explained in 4 Minutes – Web Dev Simplified
  - https://youtu.be/DUg2SWWK18I
- What is MVC architecture? - Abhay Redkar
  - https://youtu.be/mtZdybMV4Bw
- What Is MVC? Simple Explanation – Traversy Media
  - https://youtu.be/pCvZtjoRq1I

Long Form Tutorial Video

- Step-by-step ASP.NET MVC Tutorial for Beginners | Mosh
  - https://youtu.be/E7Voso411Vs