Author: Kelvin (04/22)

Gym Tracker Specification

Contents

Author: Kelvin (04/22)

1. Project Summary

This will be a project to track workouts onto an application. It will also contain a csv uploader that takes a file, shows the results, and saves to the database based on user prompts. The application is intended to be a web app and currently would not need any login functionality in the first version.

The development team for this project is just one person however there will be a task breakdown so that work can be carried out efficiently.

This is really a practice project for me to demonstrate and learn the technologies, as such the database side will be just a flat table and kept extremely simple. However, in the future I could look to make it more relational and restructure it to be more scalable.

2. Technologies

We will be creating a web application using ASP.NET Framework MVC C#. For this we will be utilising the following tools:

- Visual Studios 2019 for the programming
- Microsoft SQL Server Management Studio 18 for the database side
- GitHub for version control

3. Database

As mentioned before, the application will have just one flat table. This is why most of fields are just nvarchar. In the future, there could be many tables to reference useful details such as the unit of measurement for weights, body type list, exercise list, etc. where the users would maintain and reference. But for now those do not matter.

3.1. Table Structure

3.1.1. GymTracker

Field Name	Data Type	Allow Nulls	Example Data
ID	Int	Not null (PK)	1
DateCreated	DateTime	Not null	2022-01-01 09:00
BodyPart	Nvarchar(50)	Nullable	Back
Exercise	Nvarchar(50)	Nullable	Deadlift
Sets	Nvarchar(50)	Nullable	3
Reps	Nvarchar(50)	Nullable	10 each
Weights	Nvarchar(50)	Nullable	100 > 110 > 120 (kgs)

4. Functionality and Features

Where possible, the pages should have navigation functionality to take users back to the previous page, to home etc.

Author: Kelvin (04/22)

4.1. General Features

4.1.1. Home Page

The first screen the users should see is the home page, here we should see a button that leads to the gym tracker overview. This is where we should insert new development/functionality.

4.1.2. Overview Page

As users click the Gym Tracker button, this page will display all gym tracker records from the database ordered by date created descending, i.e. most recent data first, into a table. Each row should have an edit and delete option.

This screen should also have two buttons which leads users to the following:

- A create a gym tracker record page
- A file uploader page

4.2. CRUD Functionality

CRUD functionality are the general create, read, update and delete options that are expected for the user to properly manage the records themselves.

4.2.1. Create Record

There should be a page where users can enter details into input fields on a form and click a save button to store the details in the database. After creating the users should be taken to the overview page.

4.2.2. Edit Record

From the gym tracker overview, each record should have an edit option that leads to a form where the record's details are populated on input fields. Here the user can change what they need to and click save to update an existing record.

4.2.3. Delete record

From the gym tracker overview, each record should have a delete option that leads to a form where the record's details are populated on read-only input fields with a button that leads to the records edit page or a deletion button that prompts the users confirmation and deletes the record.

4.3. File Uploader

The file uploader is where the user can upload a csv and the application should read, validate, and display the application in a dry run page.

4.3.1. File Upload Page

From the overview page, users can click a button to reach this page where a file uploader should be present to take a user's csv file. After the user clicks the upload button the application should read and validate the contents, displaying a useful error of the mistake. The validation can be anything from wrong file format, wrong data column expected, wrong data type etc. If the correct format and data are provided users would be taken to a dry run page.

The validation and data type checks may need to be hardcoded since this is the only way to read and check the data. For instance, the file will have columns which need to be in order and with the correct datatype, so something like a letter in the date created field should be validated.

4.3.2. Dry-run Page

On the dry run page users can see what they are going to upload and return if any mistakes were made. A final upload button should be present which takes the users back to the overview page on success of data being saved to the database.

4.3.3. Valid File Input

Below is an example of a valid csv input:

	A	B	C	D	E	F
1	Date	BodyPart	Exercise	Sets	Reps	Weight
2	01/04/2022	Chest	Chest Press	3	10 each	100kg
3	02/04/2022	Back	Deadlift	3	10 each	200kg

5. Task Breakdown

The development team for this project is just one person but we would still need to break down tasks for ease of development.

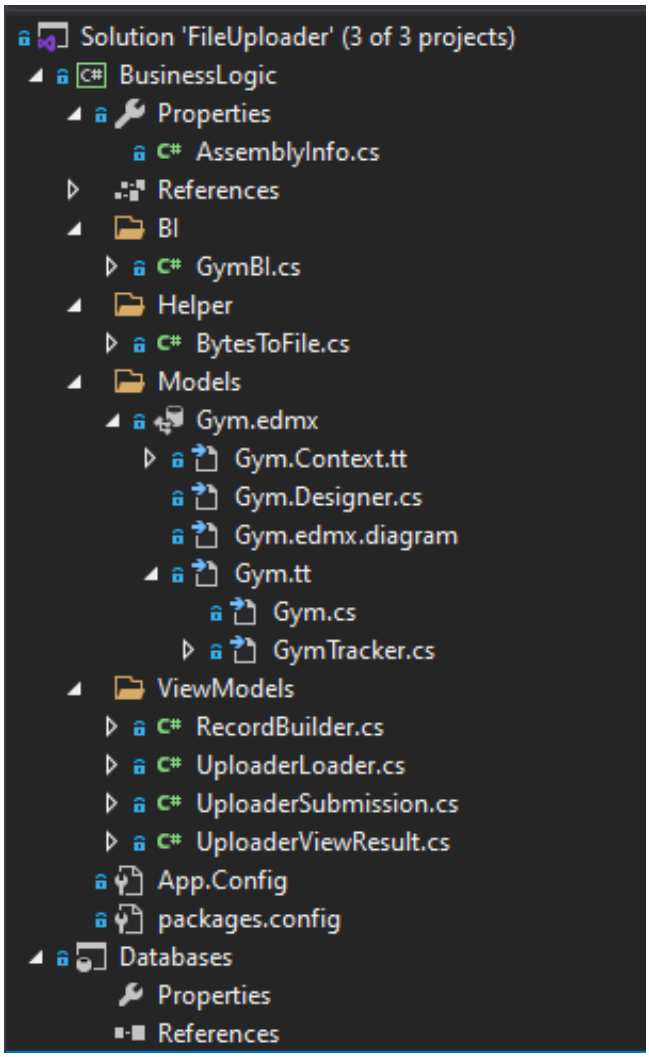
5.1. Task Breakdown Table

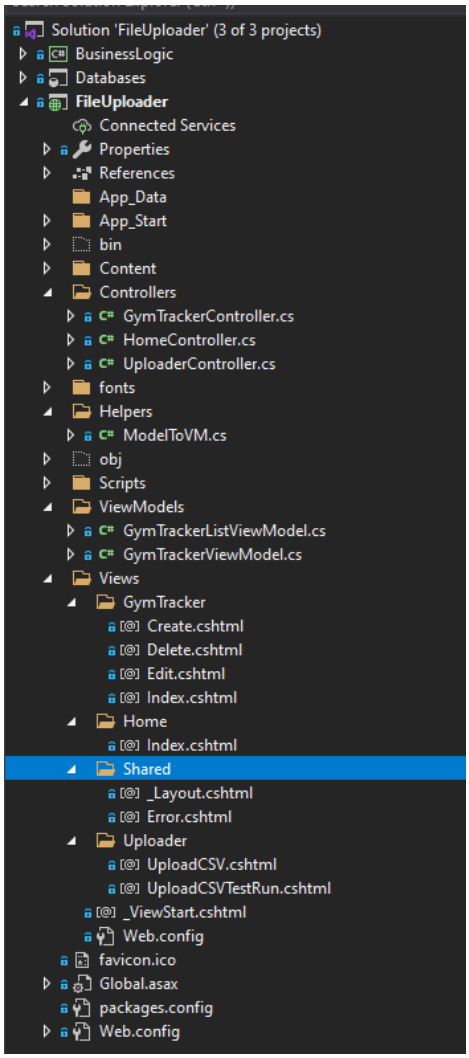
Task	Description	Estimate
Database setup	Setup of tables and some sample data via SQL scripting such that it can be ran, dropped, and reapplied with minimal effort	1 hr
Project setup	A code base project where only the main template remains, this is so that we can revert back to a blank starting checkpoint	2 hrs
Organise projects	There should be three projects: the UI website, a shared project that contains the business logic and a database project for version control. The web project will be dependent on the business logic project	2 hrs
Scripts to project	Add scripts to the database project to keep track	1 hr
Connect backend to project	Connect project to the database via the .edmx file and organise the folders between the projects	2 hrs
Add home page	Add home page UI with a button that leads to the overview page.	1 hr
Add overview page	Add overview page with table of all gym tracker records, each having an edit and delete link.	3 hrs
Add create page	Add create page that creates a gym tracker record to the database.	2 hrs
Add edit page	Add edit page that lets users edit a gym tracker record from the database based on user selection.	3 hrs
Add delete page	Add delete page that lets users delete a gym tracker record from the database based on user selection.	2 hrs
Add upload UI	Add the UI pages for the file uploader views	1 hr
Add logic to uploader	Add the business logic code for the file uploader and connect it to the UI	6 hrs
Add navigation	Add navigation throughout all the UI's	1 hr
Add styling	Add styling the whole UI	3 hrs

Author: Kelvin (04/22)

6. Actual Application

6.1. Project Layout





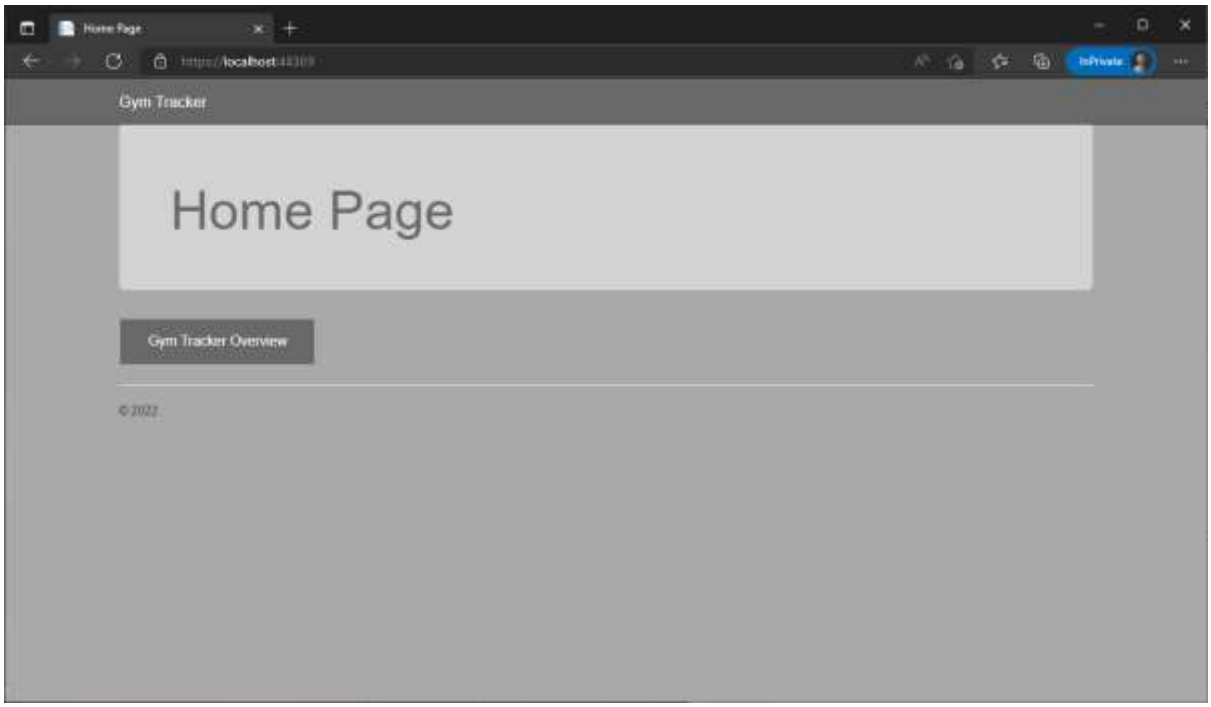
6.2. Script Used to Create Database & Table

```
IF DB_ID('Gym') IS NULL
    CREATE DATABASE [Gym]

GO

USE [Gym]

CREATE TABLE [Gym].[dbo].[GymTracker1] (
    [ID] INT NOT NULL PRIMARY KEY IDENTITY(1, 1),
    [DateCreated] DATETIME NOT NULL,
    [BodyPart] NVARCHAR(50) NULL,
    [Exercise] NVARCHAR(50) NULL,
    [Sets] NVARCHAR(50) NULL,
    [Reps] NVARCHAR(50) NULL,
    [Weights] NVARCHAR(50) NULL
)

GO

INSERT INTO [Gym].[dbo].[GymTracker1] ([DateCreated], [BodyPart], [Exercise], [Sets], [Reps], [Weights])
VALUES
('2022-04-01', 'Biceps', 'Curls', '2', '20', '8kg each'),
('2022-04-02', 'Cardio', 'Running', null, '20 mins', '8 speed')
```

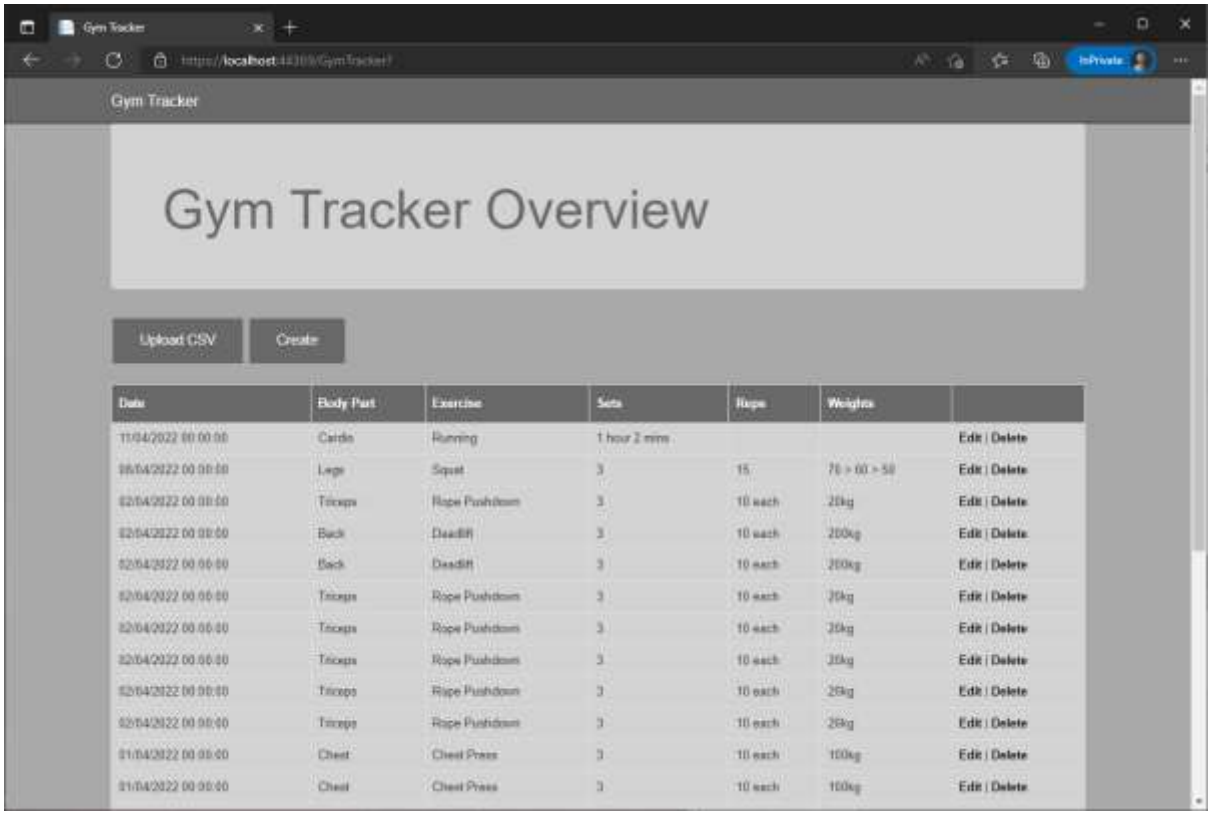
Author: Kelvin (04/22)

6.3. Screenshots of UI

6.3.1. Home Page

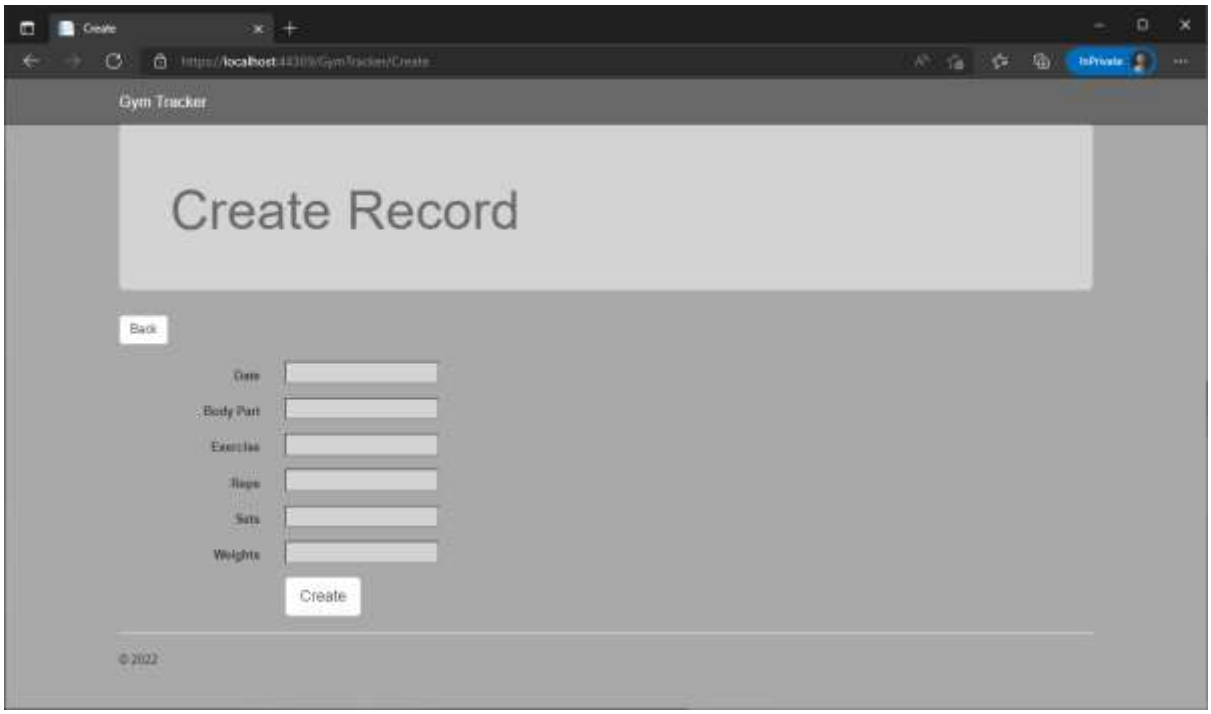


6.3.2. Gym Tracker Overview

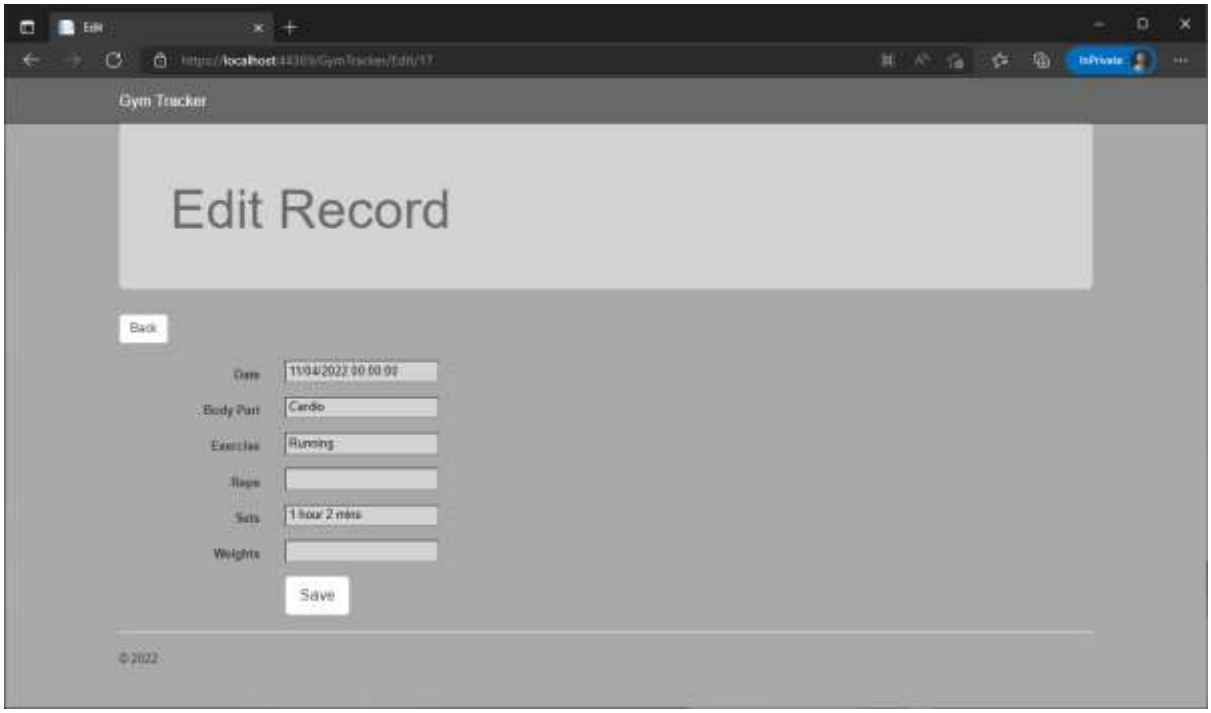


Author: Kelvin (04/22)

6.3.3. Create Page

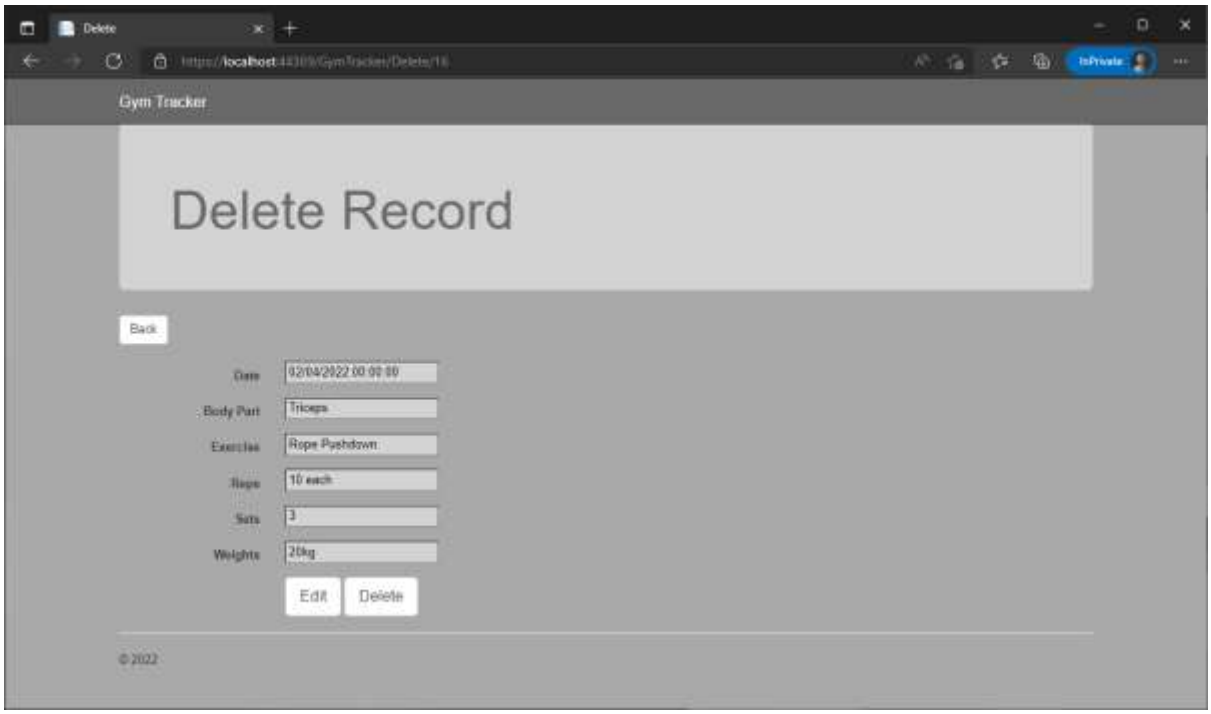


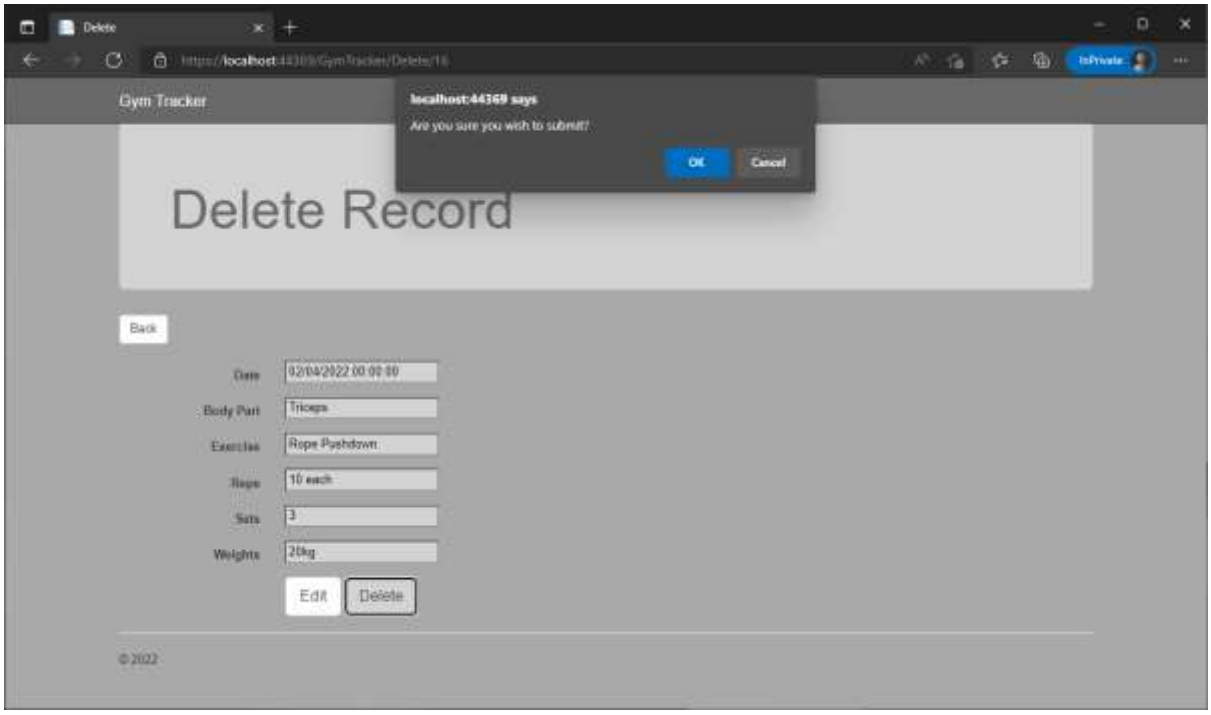
6.3.4. Edit Page



Author: Kelvin (04/22)

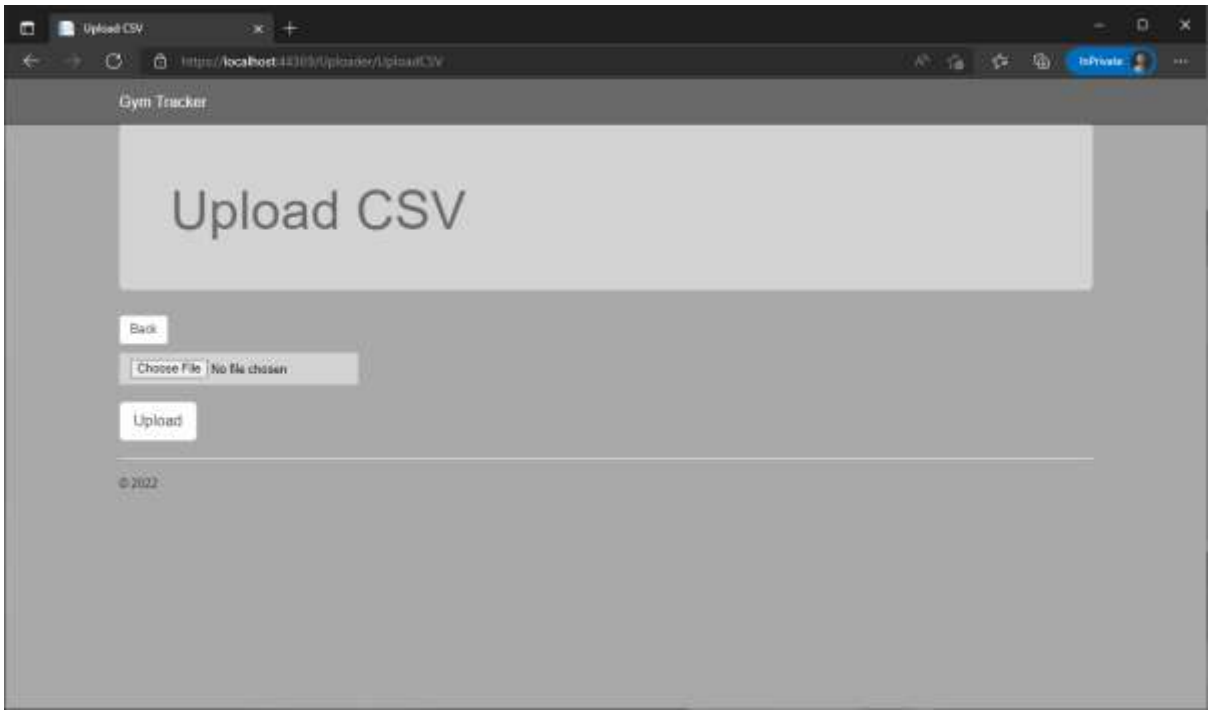
6.3.5. Delete Page



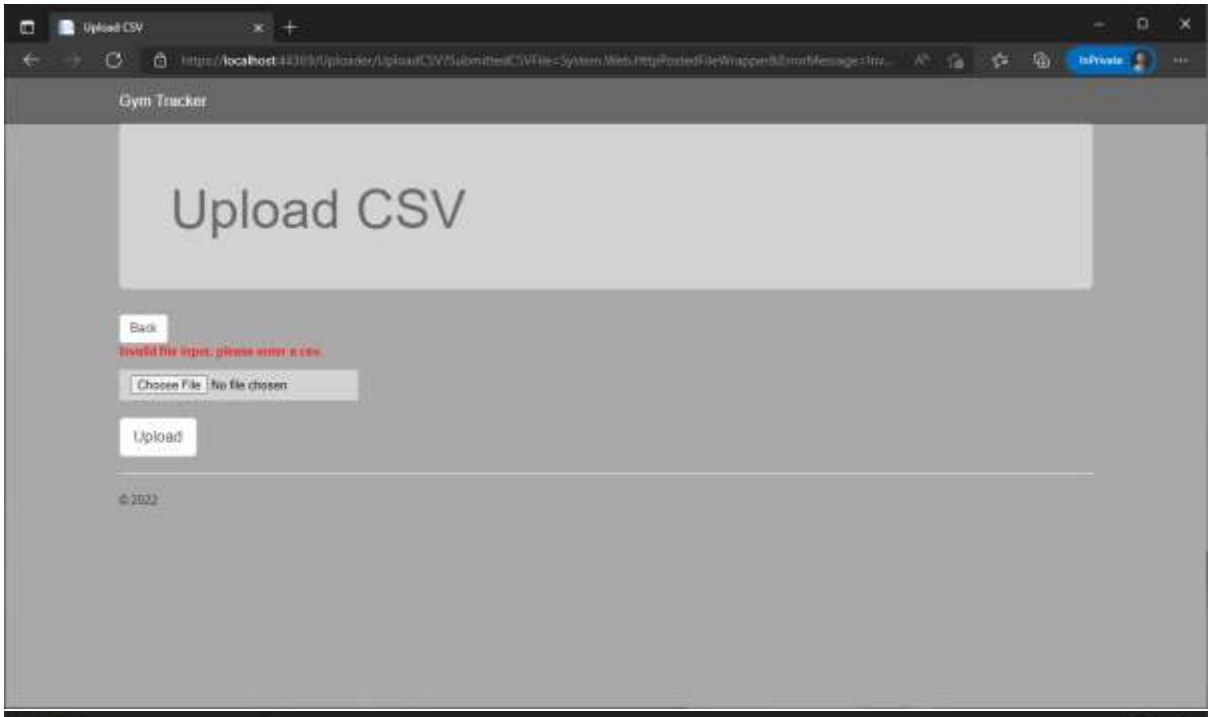


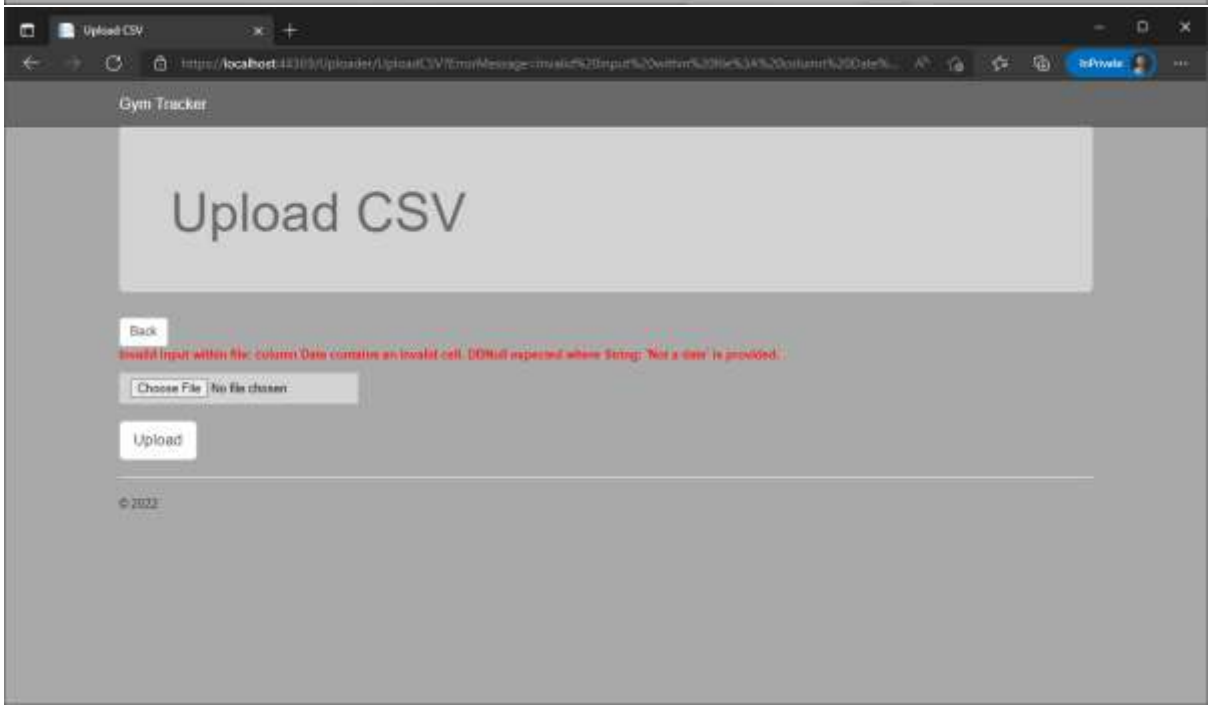
Author: Kelvin (04/22)

6.3.6. File Uploader Page

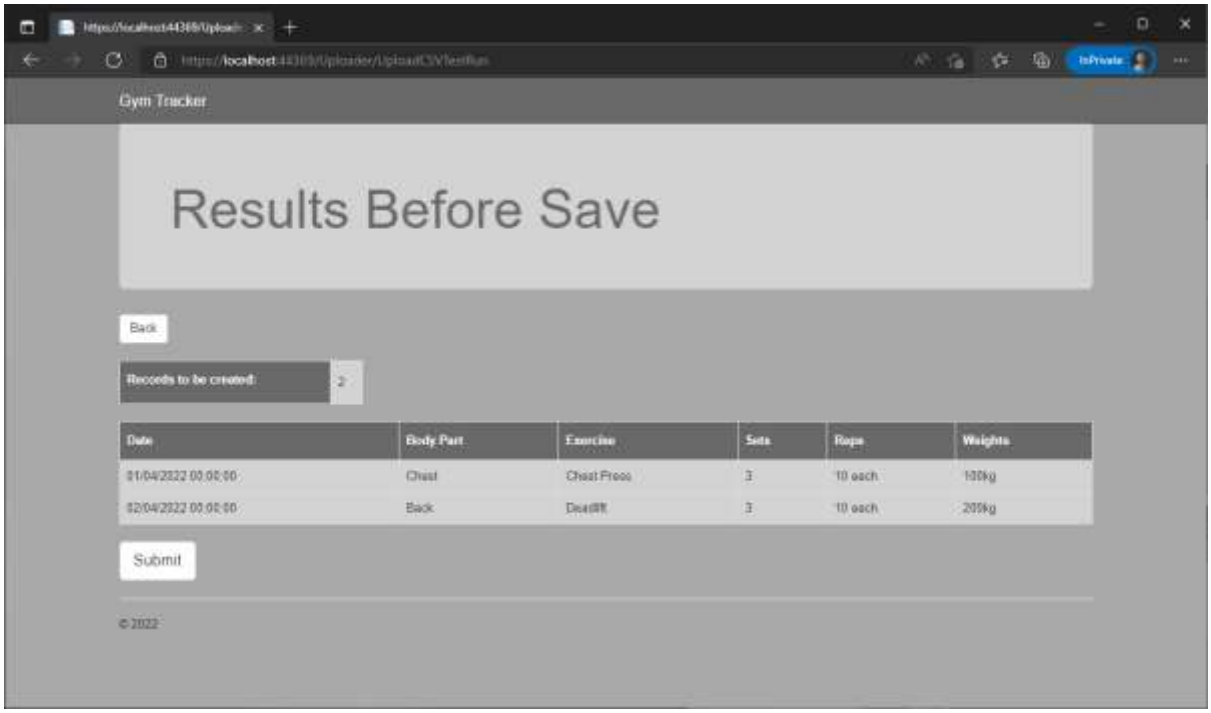


Author: Kelvin (04/22)





Author: Kelvin (04/22)



End of Document