

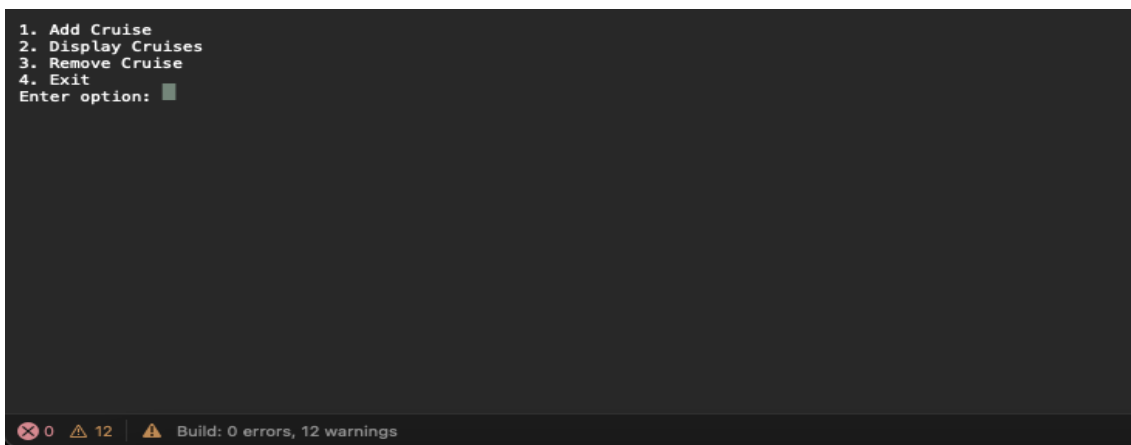
CRUISE SHIP DESIGN

The code provided is a design of a cruise ship menu that allows a user to add , display , remove cruises and exit the application. This a simple based program as it is not complicated by straight forward. The program is made out of 3 classes which are program, cruise and cruise manager.

The cruise manager will be dealing with functions like adding ,removing ,displaying of the cruise to the user. The program will be used to run the entire program and the cruise class will just be to help with the name and the destination port of the cruise ship

The first to appear is a 4 option menu which makes the user to either chose 1 which will be adding cruises , option 2 which is displaying cruise , option 3 removing cruise ,option 4 exit.

This is shown below.



```
1. Add Cruise
2. Display Cruises
3. Remove Cruise
4. Exit
Enter option: █
```

The screenshot shows a terminal window with a dark background. The menu options are listed in white text. Below the options is a prompt 'Enter option:' followed by a small white square cursor. At the bottom of the terminal, there is a status bar with icons for errors and warnings, and text indicating 'Build: 0 errors, 12 warnings'.

Figure 1

When the user clicks option one there are given options to add a name and the port of where the cruise is going .This is in figure 2. Also in figure 2 also shows when the port details are saved it is shown in cyan as a good colour to view and make the text to be different to everything else. Then it shows back to the menu again.

```
Terminal - Cruise Ship
1. Add Cruise
2. Display Cruises
3. Remove Cruise
4. Exit
Enter option: 1
Cruise Name: G21
Cruise port: London
Port saved successfully.
1. Add Cruise
2. Display Cruises
3. Remove Cruise
4. Exit
Enter option: 2
```

Figure 2

You can continue adding more cruise ships on the system as it is added to an xml file which is named cruises.xml. This is shown on the code and how it stored.

```
///
/// </summary>
private string xmlFilePath = "cruises.xml";

public CruiseManager()
{
    cruises = new List<Cruise>(); /// schema
    LoadCruisesFromXml(); /// this is where we load from the xml file thatb we have saved ouer cruise f
}

public void AddCruise(Cruise cruise)
{
    cruises.Add(cruise);
    SaveCruisesToXml(); /// this is saving the cruise to xml
    Console.ForegroundColor = ConsoleColor.Cyan;
    Console.WriteLine("Port saved successfully.");
    Console.ResetColor();
}

public void DisplayCruises()
{
    Console.ForegroundColor = ConsoleColor.DarkCyan;
    Console.WriteLine("Cruise:");
```

Figure 3

When clicked on option 2 this will show all of the stored cruises that are on the system that has been saved. This sis shown in dark cyan. It will show both the name and the destination of the cruise.

```
Cruise Name: G21
Cruise port: London
Port saved successfully.
1. Add Cruise
2. Display Cruises
3. Remove Cruise
4. Exit
Enter option: 2
Cruise:
Name: Gim, Port: maini
Name: San , Port: LondonPORT
Name: G21, Port: London
1. Add Cruise
2. Display Cruises
3. Remove Cruise
4. Exit
Enter option: █
```

0 12 Build: 0 errors, 12 warnings

Figure 4

Here as you can see there are 3 ports and to remove a port click on remove cruise and when removed it will show a bright yellow colour saying it has been removed as shown in figure 5. If a wrong port is entered it will not work and will make the user to try again and this will be a loop with a condition for the user to enter then it will proceed with the program.

```
Cruise:
Name: Gim, Port: maini
Name: San , Port: LondonPORT
Name: G21, Port: London
1. Add Cruise
2. Display Cruises
3. Remove Cruise
4. Exit
Enter option: 3
Cruise Name to Remove: G21
Cruise "G21" has been successfully removed.
1. Add Cruise
2. Display Cruises
3. Remove Cruise
4. Exit
Enter option: █
```

0 12 Build: 0 errors, 12 warnings

Figure 5

The user can exit the program when done. The user will click option 4 and the program will terminate.

Logical description of Object classes and relations

CruiseManager.cs

This manages the Cruise objects in the program. This provides functions like adding, displaying and removing ports as mentioned previously. This also saves data loading from the xml file and makes the program to run. The program is layered like this :

Fields.

List <Cruise>, All of the data of the cruises are stored here and this is a private list that contains all cruise objects. Another field is the use of cruises.xml. This file path is used for saving and loading the cruise data in the program.

Methods

In this program there are methods used , here we have the addCruise , this adds a new cruise to the list .This will then save the cruise in the save cruises To cml , then display a confirmation that the cruise has been saved.

Another method used is DisplayCruises , this makes the cruises to be displayed on the screen. This allows the user to be able to see what cruises are there., In this also another method like RemoveCruises is there and this makes the user to be able to see the displayed cruises and remove cruises if they want to. Other methods include SaveCruisesToXML which makes the sure that the cruises are saved to the system file and LoadCruisesFromXML which makes sure that the saved cruises are loaded and available to use.

Cruise.cs

This class defines the property inside the project . This maintains stuff like the cruise name and which port the cruise is going to.

Program .cs

This is a class that is simple and used for managing the data in this program. This is interacts with the cruiseManager class allowing the user to add a, display and remove cruises . This is where the menu is displayed. The menu involved here are :

Add cruise , which allows the user to create a cruise name and port. In this section the cruise will be added to a list and saved .

Display Cruises , This displays all the cruises

Remove Cruises , This allows the user to view and remove the cruises that the user wants to remove

Exit , This just exits the application.

This class ensures a smooth operation of the program.

Relations

Here provides the working relation between the classes . The cruise class represents cruise entities and constraints attributes like storing and cruise name and ports .This cruise class connects to the cruise manager , of which every cruise that is created, read or deleted are within the cruise manager. Hence making a relation between the two.

The CruiseManager class manages the collect of cruise objects and handles them. As mentioned above thus interacts with the cruise objects and with the Program .cs it will work in cohesion to make the user to interact with the program

The program class is the entry point and user interface of the application. This is the glue of the program . The program allows the user to interact with all of the commands added. The cruise Manager cs will rely on the user input and this will connect well with the cruise .cs as the user will be creating a cruise and how it will work.

This creates a good system that works smoothly and provides a user friendly system.

User Interaction.

In this program the importance of user interaction is important .The program works as intended. When the user is given a menu that has 4 options. When saving and removing a port this output a different colour. This gives the user to be able to experience a different feel in each decision they have. The program allows the user to be able to have full control and know everything that the user wants to do. Having an xml file helps keep and load information for the user so reach cruise that the user adds this is saved. It ensures that the user has all control of the program. This ensures the user to be able to do anything that they would require to make the program function.

Business logic description.

In the business side of it , the program functions as planned . The user can add a new cruise and destination about it. The main flow is that the cruise can travel one place to another. This code implements the idea as mentioned with the above classes and stated before.

Testing report

Test	Expected output	Output	Report
Add ports and remove ports	yes	yes	The function worked as intended
Port is saved	Change colour	Change of colour	The text confirming that the port has been saved has changed colours to ensure the user that it works.
Menu works	YES	YES	The menu works as intended.

Translation and storage

The data here is stored with an xml file. For example the cruise data is stored in an xml file , that can be accessed at any time. This helps make sure the loading of the file is easier as it can be extracted from it. Using xml files helps the user to be able to remove data and add. This makes the user interaction easier as seen on the from class with the menu provided.

In this pr gram saveCrusiesTo Xml. Is used to save the list of cruises ,and load cruises is used to load up the list of cruises that has been saved from SaveCrusisesToXml.

Evaluation of implementation

I have implemented the use of creating cruises and the destination they land. The program allows the user to be able to remove and delete which cruises. All of the files are saved in a xml file and this made sure that the data can be stored and accessed easily and can be deleted. This makes the program when they click on display menu they can see the amount of cruises that are there, and can be deleted if wanted. One of the functions I failed to implement was making sure that the user can have a passport number and the conditions of it. However other implementations are added like each passenger name cannot have a number in it .

Conclusion

In conclusion this program uses methods such as encapsulation and stores the data to an xml file. The user can extract data from it as well as remove data from it. The program works fully for the intended purpose.