**Flight Control Project – Daniel Krigel 1018**

Project Summary

Console Simulator creates a new flight each x seconds (as determined in the program) and  
sends the flight to the **FlightsAPI** which goes to **FlightService** and updates the Database with the new flight.  
**FlightService** then passes the flight to **TerminalService** which adds the Flight to a Queue waiting to enter Terminal 1.

When Terminal 1 is free it will take a flight from the queue (the first waiting) and the process of the terminal will start, it proceeds in each terminal (each with its own rules) until it will come back to Terminal 4 and depart.

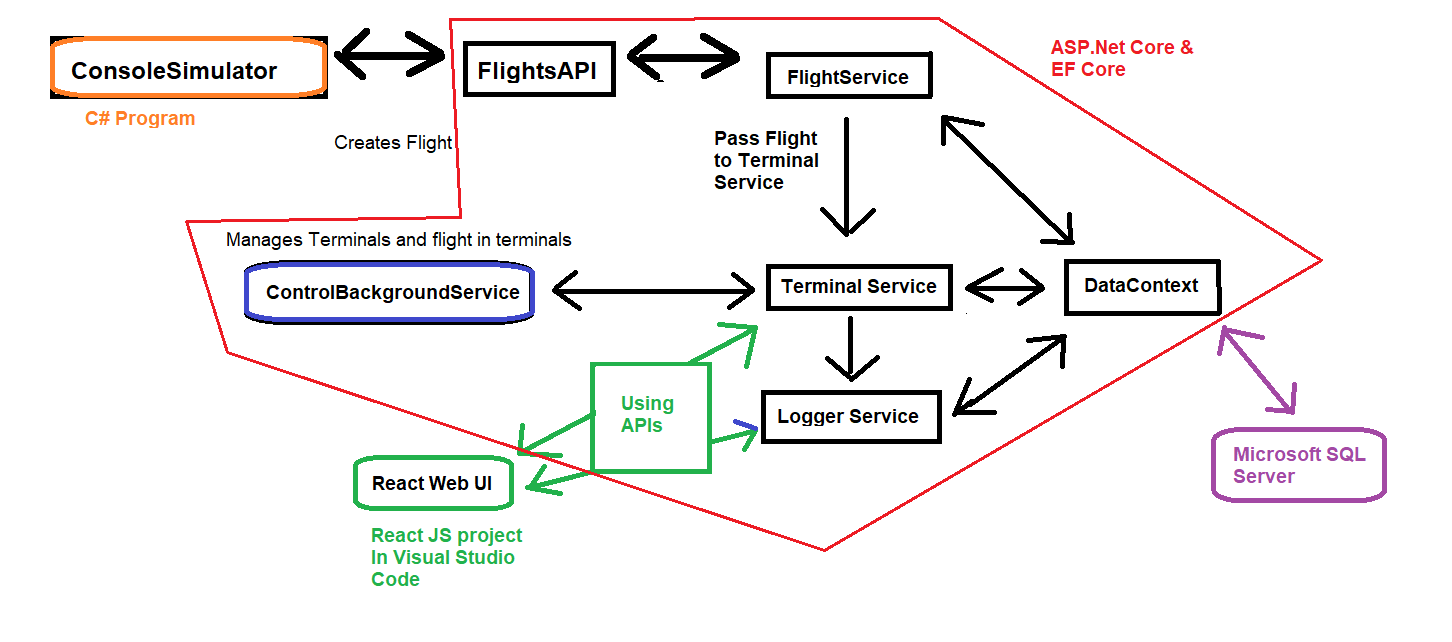
Each Terminal manages and checks it’s Flight property using **ControlBackgroundService**.

**ControlBackgroundService** is the brain that manages the terminals and flights inside the terminals and passes a flight from one terminal to the next it supposed to go to.

The program saves the logs in the Database and presents them in the UI as also shows each terminal status in the UI.

The UI is written using React with JavaScript, using Visual Studio Code.

The rest was written using EF Core, ASP.NET Core and C# in Visual Studio Community 2022.

****

Start

DB before Code, Create Db (Microsoft SQL Server)

Add-Migration and Update-Database (C# and EF) and manually set 8 terminals in DataBase.

Set wait time in Terminals in Db and manually set NextTerminalId

(Wait time for each terminal [3, 3.5, 4, 9, 7, 12, 13, 6])

(Next terminal Id for each terminal [2, 3, 4, 5, 6, 8, 8 ,4])

ControlBackgroundService

It consumes the TerminalService scoped service and uses it to work with it.  
It resets on startup all the Flight fields in the Terminals in the Database.

It initializes the terminals list it has to work with in the memory and sets the Next Terminal, it holds the <Terminal> object which is taken by the NextTerminalId set in the database.

A “while” loop runs infinitely and checks the Airport queue (Flights that are waiting to enter Terminal 1) and also required so that the TerminalService instance in the ControlService won’t be disposed of.

It sends each terminal to a function create in a new thread so every terminal can check itself parallelly.

TerminalCheck(terminal) function

It start with a small delay to prevent it running too quickly and causing issues,

It checks if a flight is inside the current terminal, if there is, it will proceed according to the conditions written and will wait before moving to the MoveNextTerminal function.

MoveNextTerminal(terminal) function

Makes sure that in case it is terminal number 3, to not enter 4 too quickly as it might cause blockade in the airport and will stall everything.

Checks that the NextTerminal is empty and proceeds to move the flight to the next terminal and update the database in Terminals and in Logs.

In case current terminal is 5, and 6 is busy, it will attempt to enter terminal number 7.

ExitAirport (terminal) function

Almost the same as previous function only it doesn’t proceed to a next terminal.

Used for Terminal 4 when flight is flagged for departure.

CheckAirportQueue() function

Checks if Terminal 1 is empty and takes a flight from the terminalService queue.