**Airport Challenge - Task**

We have a request from a client to write the software to control the flow of planes at an airport. The planes can land and take off provided that the weather is sunny. Occasionally it may be stormy, in which case no planes can land or take off. Here are the user stories that we worked out in collaboration with the client:

As an air traffic controller

So I can get passengers to a destination

I want to instruct a plane to land at an airport

|  |  |
| --- | --- |
| **Object** | **Message** |
| * air traffic controller |  |
| * plane |  |
| * airport | * **land-plane** |

airport 🡨 **land-plane** 🡪 plane landed

Looking at the message above, I now know that we will need to have an **airport** to **land** a plane. Plus a plane to land at the airport.

Tests required:

* Do I have an airport class?
* Does my airport class *respond\_to* a method to land a plane?
* Do I have a plane?
* Does land\_plane method return a plane?

1. Create a test file ‘*airport\_spec.rb*’ to test if we have a class airport.
2. **Run rspec:** got (NameError: uninitialized constant Airport)

(**RED**)

The error above occurred because there isn’t yet a file ‘*airport.rb*’ with the class ‘Airport’.

To resolve this error, a file ‘*airport.rb*’ needs to be created in the lib folder.

The ‘*airport\_spec.rb*’ file also needs to have (required ‘filename’) added to it.

1. **Run rspec:** got no errors,

(**GREEN**)

* When running Feature test in pry, Ruby was now able to find the Airport class and created a new instance of it.

A screenshot of a cell phone

Description automatically generated

1. **Time to refactor**
2. Feature test:

Feature test tried to ‘.land\_plane’ in our new ‘lisbon\_airport’

(got NoMethodError: undefined method ‘land\_plane’)

This happened because there’s no method in ‘*airport.rb*’ to land a plane yet.

Added a new test to ‘*airport\_spec.rb*’ to test if a new instance of the

class Airport would respond to the method ‘land\_plane’

1. **Run rspec:** got (NameError: undefined local variable or method)

(**RED**)

From my understanding this error occurred because there’s no method ‘.land\_plane’ in ‘*airport.rb*’.

Next step is to add method ‘.land\_plane’ to ‘*airport.rb*’

Got ( SintaxError: ) Looking at the error below:

* Failure/Error: require ‘airport’ 🡪 we know from this line what file to look
* SintaxError: ‘.rb:3:’ 🡪 this is telling which line to look for
* Last piece of info that lead to the error

A screenshot of a cell phone

Description automatically generated

Managed to resolve the issue above but my test syntax was incorrect, as I was not being able to get through it I checked the Boris-bikes walkthrough.

1. **Run rspec:** got no errors,

(**GREEN**)

1. **Time to refactor**
2. Feature test:

Feature tested the land\_plane method, and the result I got was nil. I want land\_plane to give me a plane.

Created the ‘*plane\_spec.rb*’ file to test if class Plane exists.

1. **Run rspec:** got (NameError: uninitialized constant Plane)

(**RED**)

The error above occurred because there isn’t yet a file ‘*plane.rb*’ with the class ‘Plane’.

To resolve this error, a file ‘*plane.rb*’ needs to be created in the lib folder.

The ‘*plane\_spec.rb*’ file also needs to have (required ‘filename’) added to it.

1. **Run rspec:** got no errors,

(**GREEN**)

1. **Time to refactor**
2. Added a new test to ‘*plane\_spec.rb*’, looking for land\_plane to return a plane.
3. **Run rspec:** got error (expected a **plane**, got **nil**)

(**RED**)

The error above occurred because there’s no instructions in ‘*plane.rb*’ to return new instances of the class Plane.

To resolve this error, I added (plane = Plane.new) to the class Plane.

The above did not work, I realised that I’m testing method land\_plane to return plane. So just added (“plane”) inside method land\_plane.

Neither solution above worked, so I searched for help in the Boris-bikes Walkthrough to get through this one. I made a few mistakes, but the test is now passing both rspec and in Feature testing.

1. **Run rspec:** got no errors,

(**GREEN**)

1. **Time to refactor**
2. This is the end of testing for this story. All test are working.

(**GREEN**)

As an air traffic controller

So I can get passengers on the way to their destination

I want to instruct a plane to take off from an airport and confirm that it is no longer in the airport

|  |  |
| --- | --- |
| **Object** | **Message** |
|  |  |
|  |  |
|  |  |

As an air traffic controller

To ensure safety

I want to prevent landing when the airport is full

|  |  |
| --- | --- |
| **Object** | **Message** |
|  |  |
|  |  |
|  |  |

As the system designer

So that the software can be used for many different airports

I would like a default airport capacity that can be overridden as appropriate

|  |  |
| --- | --- |
| **Object** | **Message** |
|  |  |
|  |  |
|  |  |

As an air traffic controller

To ensure safety

I want to prevent takeoff when weather is stormy

|  |  |
| --- | --- |
| **Object** | **Message** |
|  |  |
|  |  |
|  |  |

As an air traffic controller

To ensure safety

I want to prevent landing when weather is stormy

|  |  |
| --- | --- |
| **Object** | **Message** |
|  |  |
|  |  |
|  |  |

Your task is to test drive the creation of a set of classes/modules to satisfy all the above user stories. You will need to use a random number generator to set the weather (it is normally sunny but on rare occasions it may be stormy). In your tests, you'll need to use a stub to override random weather to ensure consistent test behaviour.