Airport Traffic Simulator

[Test report]

Teacher:

Chung Kuah

ProCP

19/02/2018

Developers: Yoanna Borisova, Teodor Genov, Vladimir Katrandzhiev,

Monika Kerulyte, Ignas Kybransas, Rostislav Tinchev

Contents

[Preface 3](#_Toc510158509)

[Important note 3](#_Toc510158510)

[Testing: saving/loading data 4](#_Toc510158511)

[Testing: weather manipulation 7](#_Toc510158512)

[Testing: checkpoint manipulation 8](#_Toc510158513)

[Testing: adding airplanes 10](#_Toc510158514)

[Testing: login 11](#_Toc510158515)

[Conclusions 12](#_Toc510158516)

# Preface

In this document, we will add test cases. Based on these cases, we will make test and debug report from a programmer’s side and user’s, which is unfamiliar with the simulation, side.

# Important note

We have stated in most of the test cases that the user authentication is a precondition for the test. This is true, but it will be implemented at a later stage of the development of the application. We believe that the authentication doesn’t need a proof of concept, as it is something that we have all done multiple times and it is clear, that it is doable. Thus, we haven’t implemented it as a part of this iteration’s deliverable.

We are also aware that some of the tests would give different results than what we have specified in this document, but this will also be cleared out in future iterations, since this is only a proof of concept that we are submitting right now.

# Testing: saving/loading data

**Test case ID**: 1.

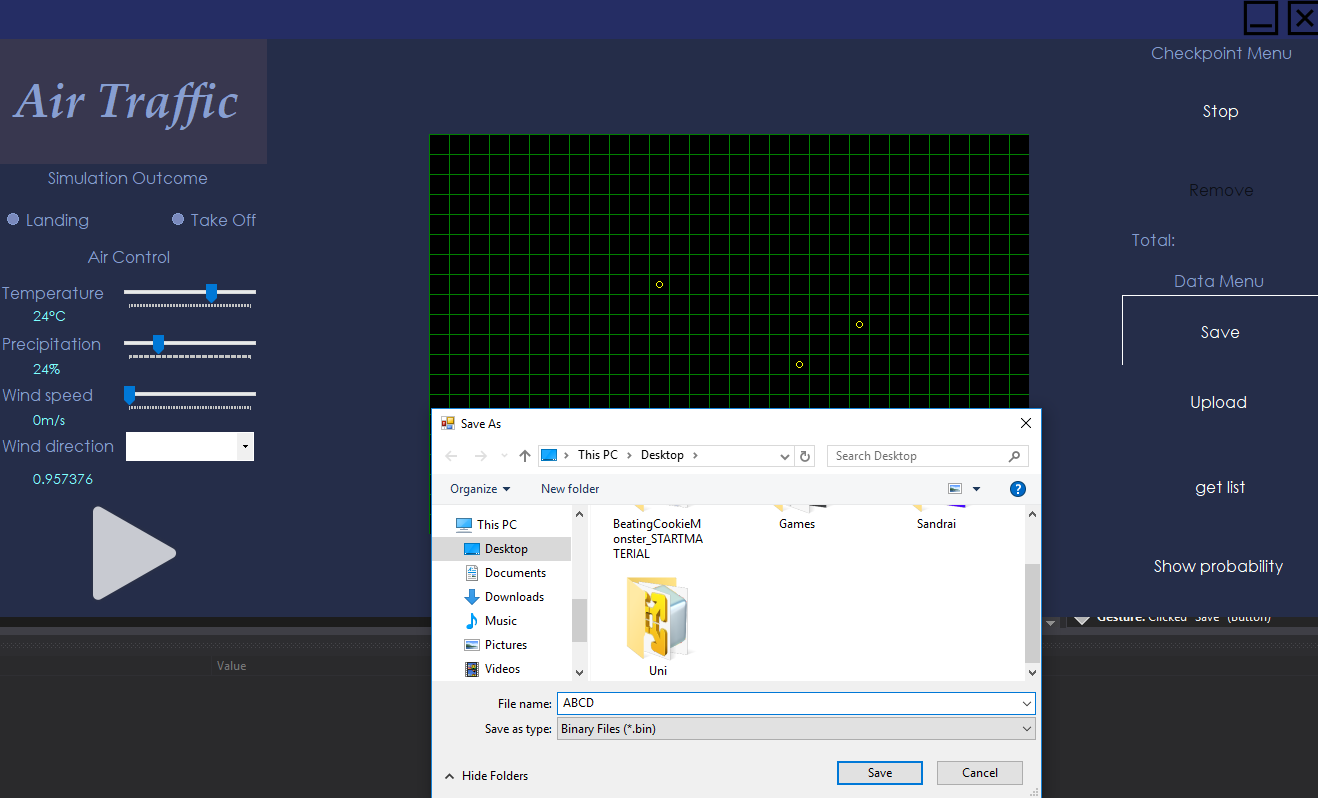
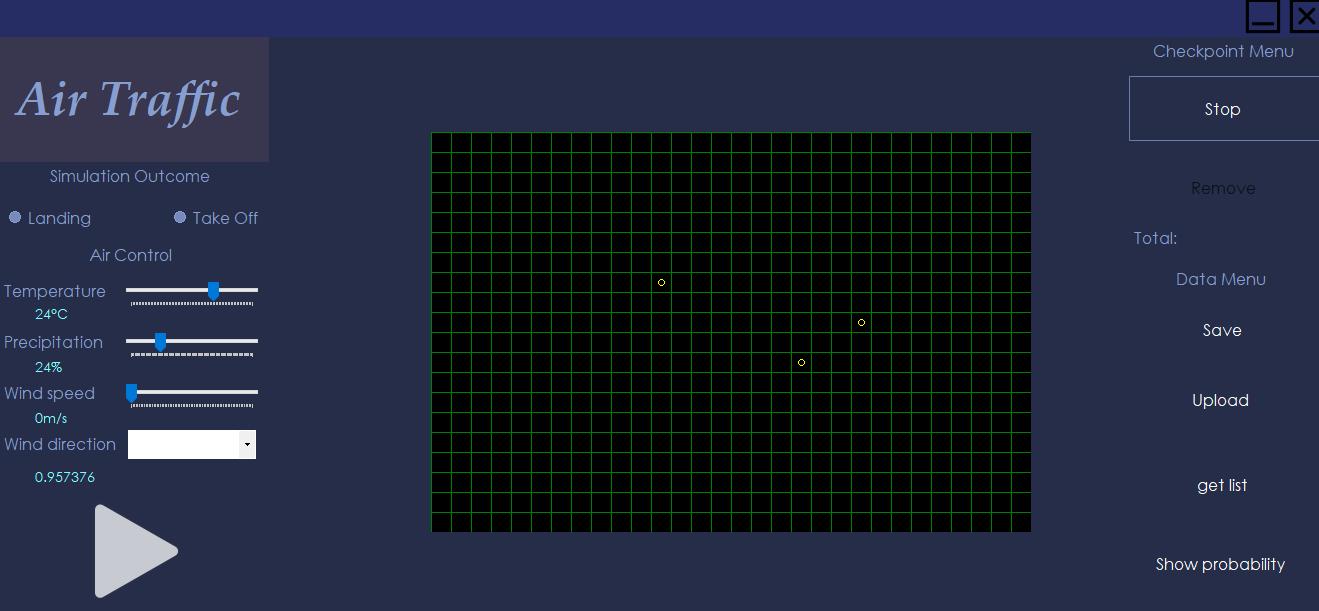
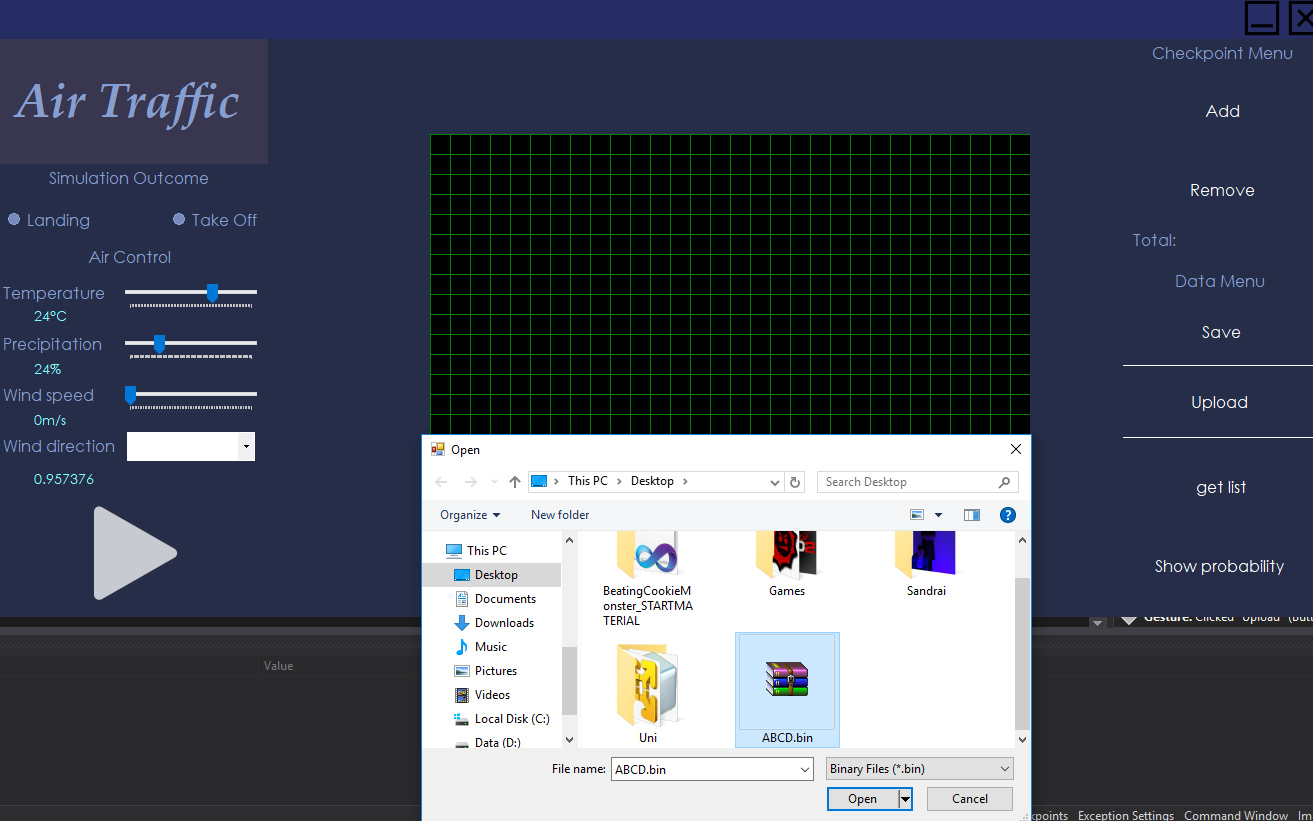
**Test case**: check if simulation parameters data is saved into file document and then uploaded to the simulation with exactly parameters.

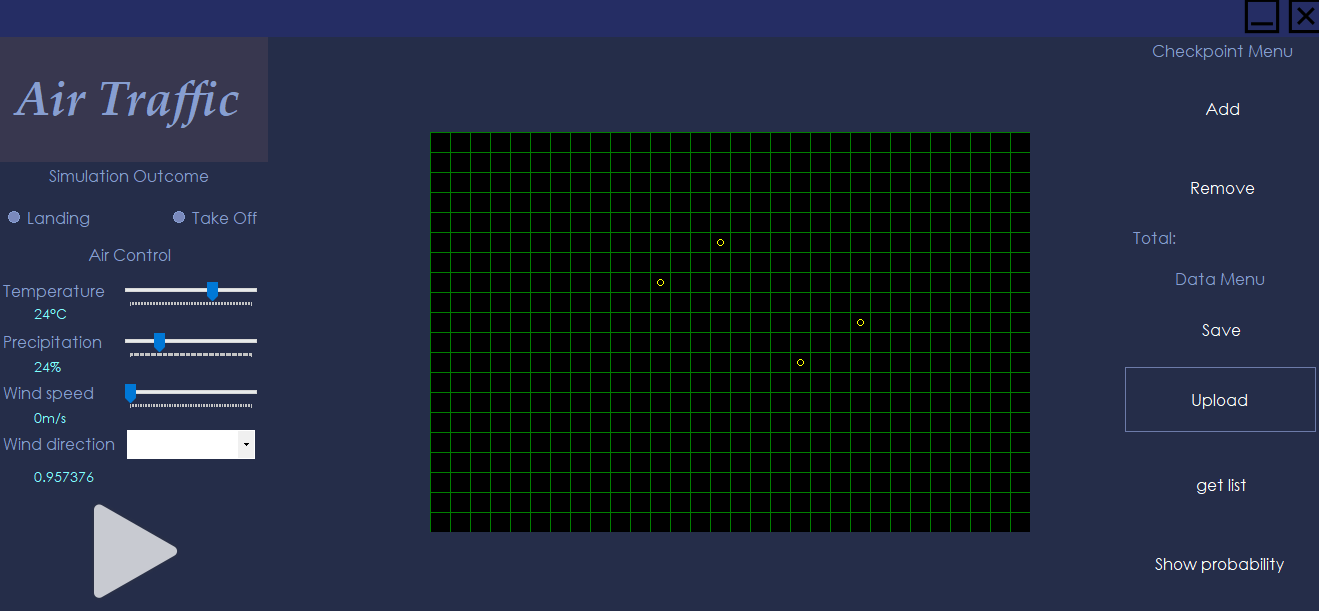
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Severity | Test scenario | Preconditions | Test steps | Test data | Expected result |
| 1.a | High | Check if data is saved to the file based on data from simulation. | Program is launched.  User is logged in.  Simulation is running. | 1. Click Save button. 2. Enter file name. 3. Select location where to save file. 4. Click Save button. | Data from running simulation. | File with data from running simulation. |

**Test case:** check if simulation parameters data is loaded from a text file.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Severity | Test scenario | Preconditions | Test steps | Test data | Expected result |
| 1.b | High | Check if simulation is running based on data from selected file. | Program is launched.  User is logged in.  Simulation is running. | 1. Click Load button. 2. Select desired file with data from a previous simulation. 3. Click Open. | Data from selected file. | Simulation with data from selected file. |

**Test report**: Test was partly successful, The file can be saved everywhere with “bin” format, you can also load only “bin” format files to get the checkpoints loaded to the simulation.



**Debug report**:

Weather conditions are not implemented in saving file so it’s only loading the saved checkpoints. It is not possible to serialize the Form class so to make an ability to save the whole simulation we are going to implement all objects as serializable and make them in one file.

While pressing “Tab” anywhere on the application every checkpoint disappear from the graphics. It will be repaired.

**UX report**:

User 1: User easily completed task by following given steps.

User 2: User took a little bit more in step 2, because couldn’t find where to enter file name, but the rest completed easily.

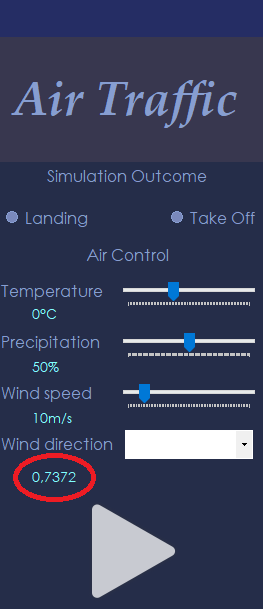
# Testing: weather manipulation

**Test case ID:** 3.

**Test case:** check weather manipulation.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Severity | Test scenario | Preconditions | Test steps | Test data | Expected result |
| 3.a | High | Check if weather manipulation influences simulation. | Program is launched.  User is logged in.  Simulation is running. | 1. Slide temperature bar in Air Control panel. 2. Slide precipitation bar in Air Control panel. 3. Slide wind speed bar in Air Control panel. | Temperature: 0°C.  Precipitation: 50%.  Wind speed: 10m/s. | 0,7372 |

**Test report**: Test was successful. The outcome is exactly as expected.



**Debug report**:

**UX report**:

User 1: User easily completed task by following given steps.

User 2: User easily completed task by following given steps.

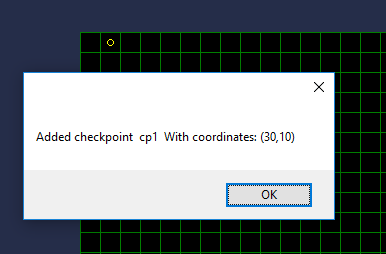
# Testing: checkpoint manipulation

**Test case ID:** 4.

**Test case:** check checkpoint manipulation.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Severity | Test scenario | Precondition | Test steps | Test data | Expected result |
| 4.a | Medium | Check if new checkpoint is added. | Program is launched.  User is logged in.  Simulation is running. | 1. Click Add button in Checkpoint Menu panel. 2. Click on the intersection point of the grid overlaying the map. | Intersection point:  Row – 1 Column - 2. | A circle is drawn into intersection point (square).  Checkpoint is added with location of X;Y = (30,10) |

**Test report**: Test was successful. The outcome is exactly as expected.



**Debug report**: ///

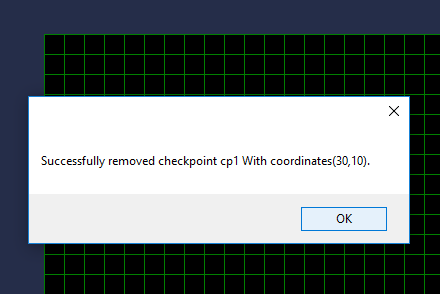
**UX report**:

User 1: User easily completed task by following given steps.

User 2: User took a little bit more time in step 1, because couldn’t find Add button, but the rest completed easily.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Severity | Test scenario | Precondition | Test steps | Test data | Expected result |
| 4.b | Medium | Check if selected checkpoint is deleted. | Program is launched.  User is logged in.  Simulation is running.  A checkpoint exists at intersection point A1. | 1. Click Remove button in Checkpoint Menu panel. 2. Click on the already existing checkpoint (circle is drawn on intersection point (square)). | Intersection point:  Row – 1 Column – 2. | A drawn circle on given intersection point Disappears. Checkpoint is deleted. |

**Test report**: Test was successful. The outcome is exactly as expected.



**Debug report**: ///

**UX report**:

User 1: User easily completed task by following given steps.

User 2: User easily completed task by following given steps.

# Testing: adding airplanes

**Test case ID:** 5.

**Test case:** check add airplanes to the air space.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Severity | Test scenario | Precondition | Test steps | Test data | Expected result |
| 5.a | Medium | Check if airplanes are added to the airspace. | Program is launched.  User is logged in.  Simulation is running. | 1. Select add new airplanes. 2. Set airplane’s attributes. 3. Click Release. | Incoming direction: 160.  Bearing: 340.  Altitude: x (this will be specified when we find out what altitude will be appropriate for the early stages of the airport approach).  Speed: y (same as with the altitude). | An airplane enters the airspace from direction 160. It is flying at altitude x and moving with speed y. It’s current bearing is 340. |

**Test report**: ///

**Debug report**: ///

**UX report**: ///

# Testing: login

**Test case ID:** 6.

**Test case:** check login functionality.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Severity | Test scenario | Precondition | Test steps | Test data | Expected result |
| 6.a | Critical | Check response on entering valid credentials. | Program is launched. | 1. Enter credentials. 2. Click Login. | Username: usrnm.  Password: pswrd. | Login is successful, and user is redirected to the simulation screen. |

**Test report**: ///

**Debug report**: ///

**UX report**: ///

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Severity | Test scenario | Precondition | Test steps | Test data | Expected result |
| 6.b | Critical | Check response on entering invalid credentials. | Program is launched. | 1. Enter wrong credentials. 2. Click Login. | Invalid credentials | Login is unsuccessful, and error message is displayed. |

**Test report**: ///

**Debug report**: ///

**UX report**: ///

# Conclusions

**Test report**:

///

**Debug report**

///

**UX report**

User 1 - user interacted easily with the application. He followed given steps in test cases and managed to complete tasks with no struggle. He concluded that GUI is well design, everything is clear, and it is easy to use.

User 2 – user followed steps in use cases She struggled at some steps but managed to complete all tasks.