9/15/2016

Members

Dean narendra

edgars kruze

nikola chobanov

robbert van den berg

User Requirement Specification

Restaurant Simulation URS

Version

Version 0.3

# Table of Contents

[Table of Contents 2](#_Toc463195557)

[Application Introduction 3](#_Toc463195558)

[Overview 3](#_Toc463195559)

[Document Introduction 4](#_Toc463195560)

[Purpose 4](#_Toc463195561)

[Scope 4](#_Toc463195562)

[Overview 4](#_Toc463195563)

[User Characteristics 4](#_Toc463195564)

[Term List 5](#_Toc463195565)

[Requirements 6](#_Toc463195566)

[Functional Requirements 6](#_Toc463195567)

[Non-Functional Requirements 6](#_Toc463195568)

[Requirement Description 7](#_Toc463195569)

[Non-Functional Requirement Description 8](#_Toc463195570)

[Use Case Diagram 9](#_Toc463195571)

[Use Cases 10](#_Toc463195572)

[User Interface 17](#_Toc463195573)

# Application Introduction

This section introduces the restaurant simulation application. It provides the overview of the application.

## Overview

This application, Restaurant Simulation Application, is used to test an arrangement of a restaurant. This application allows to have control on the restaurant. The user can put components on the grid, determine the number of staff, the number of customers and the peak hour options.

After arranging the restaurant, the user can run the simulation. When the simulation is started, customers will start coming to the restaurant. Customers will have the option to eat or just have a drink. If the customers choose to eat they will be placed on the table. Whilst, if they choose to drink only they will be placed at the bar. When the restaurant is full, no table is available, they need to wait at the waiting area or they have the option to leave.

When a big customer group comes in, more than 4 peoples, they will be placed at the group area which is where tables will be merged. In order to merge the table it requires user interaction on the application. When there is a smoking customer, they will be placed at the smoking area.

Once the simulation has started, the application will record how many people are coming, average number of customers per hour, the number of customers decide to eat or just drink and the number of customers that decide to leave without waiting.

When the simulation is finished, the user can save the arrangement of the tables including all the data that is important for the user which was gathered during the simulation. The user also have an option to load the saved arrangement and simulation

# Document Introduction

This section introduces the requirements specification for the Restaurant Simulator. It provides the purpose and scope of the application.

## Purpose

This document describes the functions and requirements specified for Restaurant Simulator Application. The application is used to arrange a restaurant scheme. Arranging a scheme of a restaurant in the simulator offers major benefits. Because you don’t have to do it in the real world. Besides that, it will also simulate real life situations so that you know if the arrangement is effective or not.

## Scope

Our client wants an effective method to graphically represent the arrangement of the restaurant and simulate it in a real life condition. This application eliminates the traditional method of arranging the table which will save time.

## Overview

This document describes general software constraints of the application. The majority of this document focuses on the specific requirements list. A complete list of requirements is given followed by each requirement explained in detail in the next section.

## User Characteristics

The main group of users for this application is everyone who owns a restaurant or has plans to open a restaurant.

# Term List

**Application** – Computer program, which we use for simulating the restaurant.

**Bar** – Place where people can have drinks.

**Component** – Component contains object such as table, smoking area and waiting area.

**Reservation system** – System that gives possibility to book a table or a few tables for visitors in the restaurant.

**Restaurant Plan** – Restaurant plan is the arrangement of components that the user has made.

**Smoking area** – Special place for smokers.

**Terrace** - Additional area outside of a restaurant where visitors can enjoy fine food and fresh air.

**User** –Person who runs the application.

**Waiting area** – Place which visitors can use when waiting for a free table.

# Requirements

In this part, we will specify the functional requirements and the non-functional requirements that this Restaurant Simulator Application has.

## Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement Number | Name of Requirement | Relevant Use Cases | Moscow |
| REQ 1 | Add Component | 1 | Must |
| REQ 2 | Create Waiting Area | 14 | Must |
| REQ 3 | Creating Smoking Area | 13 | Should |
| REQ 4 | Merging Table | 10 | Must |
| REQ 5 | Un-Merging Table | 11 | Must |
| REQ 6 | Deleting Component | 3 | Must |
| REQ 7 | Save File | 4 | Must |
| REQ 8 | Save as File | 5 | Must |
| REQ 9 | Load File | 6 | Must |
| REQ 10 | Start Simulation | 7 | Must |
| REQ 11 | Pause Simulation | 8 | Should |
| REQ 12 | Stop Simulation | 9 | Must |
| REQ 13 | Creating Group Area | 12 | Must |
| REQ 14 | Customer Flow Number | 15 | Must |
| REQ 15 | The Number of Staff | 16 | Could |
| REQ 16 | Peak Hour Option |  | Should |
| REQ 17 | Customer Flow Counter |  | Should |
| REQ 18 | Changing Smoking Area Size |  | Could |
| REQ 19 | Changing Waiting Area Size |  | Could |
| REQ 20 | Changing Group Area Size |  | Could |
| REQ 21 | Changing Table Size | 2 | Could |
| REQ 22 | Moving Table |  | Could |
| REQ 23 | The Number of Minimum Table |  | Must |

## Non-Functional Requirements

|  |  |  |
| --- | --- | --- |
| Requirement Number | Name of Requirement | Moscow |
| NREQ 1 | Portable Application | Must |
| NREQ 2 | Doesn’t take a lot of computer resource | Must |
| NREQ 3 | Easy to Maintain | Must |

## Requirement Description

**Add Component**

This functionality is used to add one of the components on restaurant grid.

**Change Waiting Area Size**

This functionality is used to add the waiting area on the restaurant grid.

**Creating Smoking Area**

This functionality is used to add smoking area on the restaurant grid.

**Merging Table**

This functionality is used to merge the tables if there is a group of more than 4 people. It can only be done inside the group area of the restaurant.

**Un-Merging Table**

This functionality is used to un-merge the merged tables after a group of more than 4 people finish eating. It can only be done inside the group are of the restaurant.

**Deleting Component**

This functionality is used to delete a component that has been placed on the restaurant grid.

**Save File**

This functionality is used to save the restaurant plan to the current save file.

**Save as File**

This functionality is used to save the restaurant plan to a new save file.

**Load File**

This functionality is used to load a restaurant arrangement that you have made before

**Start Simulation**

This functionality is used to start the restaurant simulation. The visitor will come to the restaurant with your current arrangement.

**Pause Simulation**

This functionality is used to pause the restaurant simulation. Everything will be paused and you can continue the simulation from the current condition.

**Stop Simulation**

This functionality is used to stop the restaurant simulation. The visitor visualization will be gone but the restaurant arrangement and the data can be stored.

**Creating Merging Table Area**

This functionality is used to create the area where a table can be merged if a big customer group comes.

**Customer Flow**

This functionality is used to determine how much customer that will come in an hour.

**The Number of Staff**

This functionality is used to determine the number of staff that is working in current situation.

**Peak Hour Option**

This functionality is used to simulate the restaurant if it is in peak hour time. It will increase the number of customers that come.

**Customer Flow Counter**

This functionality is used to count the number of customers that come and eat in the restaurant. Leaving customers won’t be counted.

**Changing Smoking Area Size**

This functionality is used to change the smoking area size if the current smoking area is not large enough for smoking customers.

**Changing Waiting Area Size**

This functionality is used to change the size of the waiting area if the current waiting area is not large enough for customers who are waiting for a table.

**Changing Group Area Size**

This functionality is used to change the size of the group area if there are a lot of customer groups with more than 6 people.

**Changing Table Size**

This functionality is used to change the size of a table. So that you don’t need to do delete and replace the current table for another table with a different size.

**Moving Table**

This functionality is used to move the table from its current position. So that you don’t need to delete and replace the current table position to another position.

**The Number of Minimum Table**

This functionality is used to determine how many tables are needed to start the application. In default the minimum number of tables is 1.

## Non-Functional Requirement Description

**Portable Application**

This non-functional requirement makes it so that the application can be run on several windows versions such as Windows 7, 8 and 10. The application itself doesn’t need to be installed and can be run straightaway.

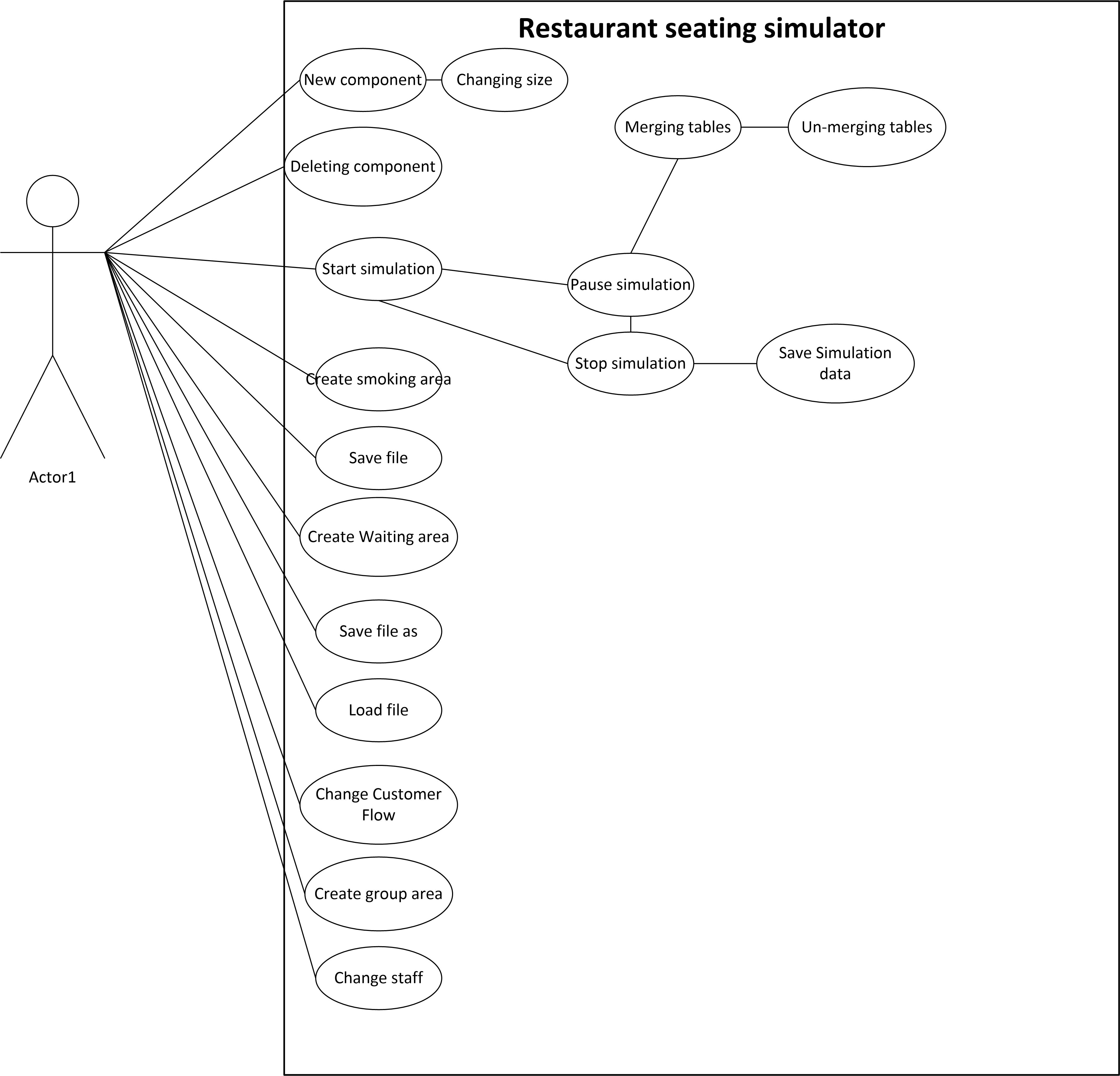
**Doesn’t Take a Lot of Resource**

This non-functional requirement makes the application light weighted. It won’t be taking more than 30% of the computer process with Intel i5 Processor with 4GB RAM.

**Easy to Maintain**

This non-functional requirement makes the application easy to maintain and develop. Because we are going to code it in a good structure such as giving a comment on each method.

# Use Case Diagram



# Use Cases

Controls mentioned in use cases, refers to controls in GUI from next section.

|  |  |
| --- | --- |
| **Use Case 1** | Adding New Component on Restaurant Grid |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation not running |

**Main Success Scenario:**

1. User selects component from Component menu
2. System opens Component property menu
3. User selects properties for component in Component property menu
4. User clicks in the area, where user wants component to be placed
5. System updates the changes on the screen

**Extensions:**

3.a User can’t change any properties in Component property menu

1. Selected component doesn’t have any adjustable properties

2. Use case continues with step 4

3. End use case

4.a System can't place component in selected area

1. System informs end-user that component can't be placed in selected area, because the area is already occupied.
2. End of use case.

|  |  |
| --- | --- |
| **Use Case 2** | Changing properties for component on Restaurant Grid |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation not running |

**Main Success Scenario:**

1. User selects component from Restaurant Grid
2. System opens Component property menu
3. User adjusts properties provided in Component property menu
4. User clicks "Confirm" button in Component property menu
5. System updates table values
6. System updates the changes on the screen

**Extensions:**

3.a User can’t select any properties in Component property menu

1. Selected component doesn’t have any adjustable properties

1. End of use case.

5.a System can't update component

1. System informs end-user that component can't be updated, because another component is too close.
2. System resets table values, to its original values.
3. End of use case.

|  |  |
| --- | --- |
| **Use Case 3** | Deleting component from Restaurant Grid |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation not running |

**Main Success Scenario:**

1. User selects component on Restaurant Grid
2. System opens Component property menu
3. User clicks "Delete" button in Component property menu
4. System removes selected component from Restaurant Grid
5. System updates internal state
6. System updates the changes on the screen

**Extensions:**

2.a System doesn’t open Component property menu

1. User didn’t select component on Restaurant Grid

2. End use case

|  |  |
| --- | --- |
| **Use Case 4** | Save File |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Open project |

**Main Success Scenario:**

1. User clicks “Save” button.
2. System saves the file.

**Extensions:**

1.a Project file has not been saved on the device yet.

1. Go to Use Case 5

|  |  |
| --- | --- |
| **Use Case 5** | Save file as a new file |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Open project |

**Main Success Scenario:**

1. User clicks ‘’Save’’ button
2. System shows a file dialog-box.
3. User chooses the directory where the file will be stored, and name the file.
4. System saves the file.

**Extensions:**

1.a User didn’t intend to save file

1. User clicks "Cancel"
2. End of use case

|  |  |
| --- | --- |
| **Use Case 6** | Load File |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | None |

**Main Success Scenario:**

1. User clicks “Open file” button
2. System shows a file dialog-box.
3. User selects file to open.
4. System closes previous project.
5. System loads the file.

**Extensions:**

3.a Another project is open

1. 1. System asks if user wants, to save the project, if the user chooses cancel that will end the use case.

3.a File can’t be loaded.

1. System informs user that file can’t be loaded and given the choice to stop or choose another file.
2. End of use case.

|  |  |
| --- | --- |
| **Use Case 7** | Start simulation |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | None |

**Main Success Scenario:**

1. User clicks on “Start simulation” button
2. System starts simulation with default values see section 0.0 (to be updated)

**Extensions:**

1.a Simulation can't start, because there are no tables on Restaurant Grid.

1. System informs user, that there should be at least 1 table on the Restaurant Grid, for simulation to start.
2. End of use case

|  |  |
| --- | --- |
| **Use Case 8** | Pause Simulation |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation running |

**Main Success Scenario:**

1. User click on “Pause simulation” button
2. System pauses simulation

|  |  |
| --- | --- |
| **Use Case 9** | Stop Simulation |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation running |

**Main Success Scenario:**

1. User click on "Stop simulation" button
2. System pauses simulation
3. System stops simulation.
4. System asks user, if user wants to save statistics from this simulation.
5. System updates changes on the screen.
6. System updates internal state.

**Extensions:**

4.a If user clicks "Yes"

1. Go to Use Case 17

4.b If user clicks "No"

1. Use case continues with step 5.

|  |  |
| --- | --- |
| **Use Case 10** | Merge table components |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation is in paused state. |

**Main Success Scenario:**

1. User selects table in group area.

2. System opens side menu.

3. User clicks "Merge".

4. User selects different table in the group area.

5. System updates internal state.

6. System updates changes on the screen.

**Extensions:**

4.a User selects same table again

1. System informs user, that you can't merge table with itself.
2. End of use case.

4.b User selects table outside designated area for merging

1. System informs user, that these tables are not available for merging.
2. End of use case.

|  |  |
| --- | --- |
| **Use Case 11** | Un-Merging Tables |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation is in paused state. |

**Main Success Scenario:**

1. User selects table in group area
2. System opens side bar
3. User clicks "Un-merge"
4. User places tables back on Restaurant Grid
5. System updates internal state
6. System updates changes on the screen

**Extensions:**

4.a System can't place component in selected area

1. System informs end-user that component cannot be placed in selected area, because the area is already occupied
2. End of use case

|  |  |
| --- | --- |
| **Use Case 12** | Creating group area |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | User selected tool for creating group area in component bar |

**Main Success Scenario:**

1. User selects starting point for area on Restaurant Grid
2. User moves mouse, to draw rectangle area for group area
3. System draws area on Restaurant Grid
4. System updates internal state

**Extensions:**

3.a System can’t draw area on Restaurant Grid

1. There is already existing group area on Restaurant Grid
2. Drowned area is overlapping with another area.
3. End of use case

|  |  |
| --- | --- |
| **Use Case 13** | Creating Smoking Area |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | User selects tool for creating smoking area from component bar |

**Main Success Scenario:**

1. User selects starting point for area on Restaurant Grid
2. User moves mouse, to draw rectangle area for smoking area
3. System draws area on Restaurant Grid
4. System updates internal state

**Extensions:**

3.a System can’t draw area on Restaurant Grid

1. There is already existing smoking area on Restaurant Grid
2. Drowned area is overlapping with another area.
3. End of use case

|  |  |
| --- | --- |
| **Use Case 14** | Creating Waiting Area |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | User selects tool for creating waiting area from component bar |

**Main Success Scenario:**

1. User selects starting point for area on Restaurant Grid
2. User moves mouse, to draw rectangle area for Waiting Area
3. System draws area on Restaurant Grid
4. System updates internal state

**Extensions:**

3.a System can’t draw area on Restaurant Grid

1. There is already existing waiting area on Restaurant Grid
2. Drowned area is overlapping with another area.
3. End of use case

|  |  |
| --- | --- |
| **Use Case 15** | Changing Customer Flow Number |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation not running |

**Main Success Scenario:**

1. User types integer in textbox, next to field “Customer flow”
2. User clicks “Confirm” button
3. System updates internal state

**Extensions:**

1.a User gives value larger than maximum allowed

1. Customer flow set to maximum value

1.b User gives negative or non-numerical value

1.Customer flow is set to minimum.

|  |  |
| --- | --- |
| **Use Case 16** | Changing Staff amount in restaurant |
| **Level:** | User-Goal |
| **Primary Actor:** | End-User |
| **Preconditions:** | Simulation not running |

**Main Success Scenario:**

1. User types integer in textbox, next to field “Restaurant staff”
2. User clicks “Confirm” button
3. System updates internal state

**Extensions:**

1.a User gives value larger than maximum allowed

1. Restaurant staff set to maximum value

1.b User gives negative or non-numerical value

1. Restaurant staff is set to minimum.

|  |  |
| --- | --- |
| **Use Case 17** | Save simulation data |
| **Level:** | User-Goal |
| **Primary Actor:** | System |
| **Preconditions:** | End of simulation |

**Main Success Scenario:**

1. System saves statistic data to file
2. System shows message box, that save has been successful.
3. Return to use case 9, step 5.

# User Interface

As we are creating a Windows application for PC’s and laptops we’ve decided to use all benefits of a big screen. As a result, we have decided to make a single window app where all components will be displayed when the program starts. Sketches of the main app’s window are represented below.

