# Master Team Project Fall 2022: Risto

**Team 3 of Global Distributed Software Development (Hochschule Fulda):**

- **Luis Miguel García Marín (Team Lead)**
  luis-miguel.garcia-marin@informatik.hs-fulda.de
- **Jesús Moreno Durán (Back End Lead)**
- **Noman Ali (Front End Lead)**
- **Paras (GitHub Master)**
- **Vichitar Dagar (Back End Team Member)**
- **Alhassane Dondo Toure (Front End Team Member)**

**Milestone 1**
**Date: 07.12.2022**

| **History Table** | |
|---|---|
| **Date submitted** | **Date revised** |
| 07.12.2022 | 07.12.2022 |
| | |

# 1. Executive Summary

This application focuses on the reservation of a place in a restaurant in a comfortable and simple way, no personal data is requested, no credit cards are required, etc. In addition, it is also possible to evaluate the personal experience of each place, which would allow us to discover new places quickly.

This is Risto, a website where you have everything at hand and in a very simple way, for all audiences.

Risto makes life easier for both restaurants and customers. It is true that there are applications that include similar functionalities to this app, but Risto is more complete and simpler.

A person can make a reservation in a Restaurant and from the time of booking up to 2 hours before they could cancel the reservation with the option of specifying a reason for the cancellation.

The restaurant will have different sections in which it will see the reservations made for each day and select if the people of the reservation have already appeared or not and if there has been any problem (more people than specified, late time...). This would be seen by an administrator and he/she could take the necessary measures.

In order to avoid users with the aim of doing wrong who may book and then not go, new users will be in a trial period and if they miss the reservation the account will be suspended without prior notice. To appeal this suspension, they will have to contact an administrator.

To reduce the user's waiting time and increase the capacity of customers a restaurant can serve in a day, the application also offers the possibility to select the food that will be wanted. This will improve the experience for all our users.

Risto will also offer the possibility of reserving a place in the restaurant's car park if it exists in the same way as the tables.

# 2. Personae and main Use Cases

## Personas

### 1. Customer

**Characteristics** – People who come to the restaurant and reserve tables, they pay money in exchange for services and food, they are not part of the restaurant but critical in its functioning.

**Goals** – They book a table at the restaurant at a specific time. They also order food with the application before or after their arrival. They review the food and the services.

**Skills** – They know how to use a web browser and are supposed to be able to navigate through the interface on our web page.

**Pain** – When they are not able to book a reservation or order food, possible reasons being not able understand the interface, another one could be the system failing to work properly.

## 2. Waiter

**Characteristics** – People who take the food from the chef to the customer's table, and take orders from the customer to give them to the chef.

**Goals** – Their job is to take orders from the chef and to deliver the order to a specified table. They also can check order requests and give them to chefs to prepare food.

**Skills** – They should know how to use a web browser and are supposed to be able to navigate through the interface on our web page. They must also have knowledge of the physical table location and address.

**Pain** – When they can not check the order request, do not receive any reply from the chef or are not able to give orders to the chef.

## 3. Manager/ Host/Hostess

**Characteristics** – They manage and overlook the working of the restaurant.

**Goals** – They see and update the status of table occupancy after the customers leave and greet the new ones.

**Skills** – They should know how to use a web browser and are supposed to be able to navigate through the interface on our web page.

**Pain** – When the system (portal) fails to work and they are not able to see and update the details of the table.

## 4. Website Admin

**Characteristics** – They overlook the website.

**Goals** – They check content, reviews, or images that are supposed to be posted on restaurants' pages before they are visible on the live website. They can then authorize the content and allow it to be posted/visible on the website.

**Skills** – They should know how to use a web browser and are supposed to be able to navigate through the interface on our web page.

**Pain** – When the system (portal) fails to work and they are not able to check and authorize the content to be posted on the web site.

## Use-Cases

1. **Booking Table** – Customers can reserve a table of a restaurant through our website. They can select the table and the time slot. The website will then generate a reservation number, through which they can get entry into the restaurant at the pre-agreed time. They can also search for different restaurants and choose from various options available.

2. **Ordering Food** – Customers can also order food before (pre-order) or after their arrival at the restaurant. All they have to do is use their reservation number and select what they want to eat. Food ordered through the website will be sent to the waiter who will inform the Chef to start preparing the meal shortly before (in case of pre-order) customer's arrival or if the food has not been pre-ordered, the customer can instantly place an order through the website, when they have already arrived at the restaurant at the correct time.

3. **Getting and giving back prepared orders** – Waiters can check the website to get customer orders. They then will give orders (from the customer verbally and/or through the website) to chefs, who will cook the orders. Once the order is prepared the chef informs the waiter verbally and the waiter has to fetch the order/food at the table.

4. **Updating the status of the table occupancy** – As soon as a table is free the manager will update the availability of the table. This will be reflected on the live website, so that if anybody wants to come to the restaurant without booking and see that a particular table is empty until the next reservation, then they can come directly, order food, and have a fine dining experience.

5. **Customer Review and website content update** – Customers can review the food and services only by entering their reservation number, and only after they have had the food. Then, a website admin can then check the review for any foul words and allow them to post their review. The website admin can also check the updates or new content that is to be posted on the website, to see if the content is appropriate to be posted online. He/she can then allow the content to be posted on the live website.

6. **Restaurants can add their details on the website** – Restaurants can register on the website to be searched and selected by the customers. They must provide complete details (name, contact information, address…). Most importantly, the features that could be provided by them to the customers. For instance if they don't have a parking space, they should mention it on the website.

## 3. List of main data items and entities

**1. Restaurant:** This term refers to any listed restaurant in the software offering reservation services. Restaurants serve as the primary source of information to the application.

**2. Resource:** A term used to refer to any storage entity of the software.

**3. User:** Any client of the software interacting with the interface to use the services offered.

**4. Customer:** A known client of the software making use of the features offered. A client may manage their own data, posts or reservations. A customer can further read only the data public on the website.

**5. Guest:** An anonymous client of the software making use of the features offered. Guests can only browse website.

**6. Administrator:** A term used to represent a user responsible for approving customer posts or requests. Administrators can also manage user registrations and other maintenance related information on the software.

**7. Manager:** A user responsible for regulating the information related to the restaurant they belong to. They can manage vacancy of tables, and the daily calendar.

**8. Reservation:** An entity used to represent the process involving a customer reserving a seat(s) at a Restaurant.

**9. Restaurant Lookup:** An entity used to represent the process of searching for specific restaurants by the users.

**10. Staff:** An entity that represents a set of users that represent a particular restaurant.

**11. Food:** An entity to represent one specific food in regard to a particular food category and a particular restaurant.

**12. Table:** An entity used to represent a seat belonging to a particular restaurant.

**13. Table Map:** An entity used to represent a layout map of tables in a particular restaurant.

**14. Posts:** An entity that represents a request for reservation or a query, or a feedback uploaded by a particular user.

**15. Waiter:** An entity used to represent a staff member particularly responsible to serve customers.

**16. Food Category:** A logical entity used to characterize a particular set of similar food entities.

**17. Order:** An entity used to represent a customer's food request to a restaurant.

**18. Notification:** A logical entity used to represent information delivered to the customer from a restaurant or the software end.

**19. Promotion:** used to refer to the logic of a restaurant being highlighted on the landing page of the software for a particular amount of time.

**20. Parking:** An entity used to represent the logical concept of parking of a customer vehicle in the restaurant parking lot.

## 4. Initial list of functional requirements

1. Restaurants can register in the system.
2. Restaurants can upload information about themselves to subscribe to the service.
3. Site administrators can approve or decline restaurant info for posting.
4. Site administrators can deal with typical admin duties.
    - 4.1. Site administrators can manage user registrations.
    - 4.2. Site administrators can accept or decline review posts.
    - 4.3. Site administrators can delete posts.
    - 4.4. Site administrators can ban users with bad behavior.
5. Customers can search for restaurants.
6. Customers can check the table status of a restaurant (live view).
7. Customers can manage their reservations.
    - 7.1. Customers can make a reservation.
    - 7.2. Customers can change their reservation.
    - 7.3. Customers can cancel their reservation.
8. Customers can make orders of food before and after their arrival to a restaurant in which they have made a reservation.
9. Customers can manage posts of reviews.
    - 9.1. Customers can post reviews.
    - 9.2. Customers can edit their reviews.
    - 9.3. Customers can delete their reviews.
10. Managers (Host/hostess) can review daily calendar.

11. Managers (Host/hostess) can check incoming guests to greet them.
12. Managers (Host/hostess) can check the table status of their restaurant.
13. Managers (Host/hostess) can set tables for free when they are available.
14. Waiters can check the food order list.
15. Waiters can fetch orders and their corresponding tables.
16. Waiters can set food orders which they have given to a table as delivered.
17. Customers can check the food processing time.
18. Customers can check the list of food with their price.
19. Customers can reserve a parking spot near the restaurant parking point.

## 5. List of non-functional requirements

1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in Milestone 0. Application delivery shall be from chosen cloud server.
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers.
3. All or selected application functions must render well on mobile devices.
4. Data shall be stored in the database on the team's deployment cloud server.
5. Full resolution free media shall be downloadable directly, and full resolution media for selling shall be obtained after contacting the seller/owner.
6. No more than 50 concurrent users shall be accessing the application at any time
7. Privacy of users shall be protected, and all privacy policies will be appropriately communicated to the users.
8. The language used shall be English (no localization needed).
9. Application shall be very easy to use and intuitive.
10. Application should follow established architecture patterns.
11. Application code and its repository shall be easy to inspect and maintain.
12. Google analytics shall be used (optional for Fulda teams).
13. No e-mail clients shall be allowed.
14. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.
15. Site security: basic best practices shall be applied (as covered in the class) for main data items.
16. Application shall be media rich (images, video etc.). Media formats shall be standard as used in the market today.
17. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development.
18. For code development and management, as well as documentation like formal milestones required in the class, each team shall use their own GitHub to be set-up by class instructors and started by each team during Milestone 0.
19. The application UI (WWW and mobile) shall prominently display the following exact text on all pages "Fulda University of Applied Sciences Software Engineering Project, Fall 2022 For Demonstration Only" at the top of the WWW page (Important to not confuse this with a real application).

## 6. Competitive analysis

| *Features* | Our System | Opentable.com | Quandoo.de | Thefork.com |
|---|---|---|---|---|
| **Simple interface** | ✔ | ✓ | ✓ | ✓ |
| **Display available time slots** | ✔ | ✓ | ✓ | ✓ |
| **View images and Menu** | ✔ | ✓ | ✓ | ✓ |
| **Review Food and Place** | ✔ | ✓ | ✓ | ✓ |
| **Live view of reserved and available table** | ✔ | ✖ | ✖ | ✖ |
| **Pre-order food with table reservation** | ✔ | ✖ | ✖ | ✖ |
| **Waiter-less food ordering** | ✔ | ✖ | ✖ | ✖ |
| **Reserve Parking with while booking Table** | ✔ | ✖ | ✖ | ✖ |

### Live View
With our system the customer can have a live view (top view of a graphical map of the restaurant with tables represented by diagrams) of the tables present in the restaurant and status of their occupancy. So customers can see exactly what tables are available and where those tables are located, for example if customers want a table near a window or at the very end, something more discrete, they can see that on the web page and book accordingly. *The status of table occupancy will be updated by the manager of the restaurant when the customer has paid and left the table.*

### Pre-order Food
With us, customers can even pre-order food during your reservation. They will have full access to the Menu on the web page and can pre-order food before even reaching the restaurant. When they arrive at the time of reservation, the food will be there in no time. The chef will be notified of the upcoming pre-orders shortly before the time of arrival of the customer.

### Waiter-less food ordering
If customers want to order but can not find any waiter around, then they can use the web page to order food, by selecting the dish and the quantity of food, they must provide their

reservation number to place an order. The order will be sent to the chef to prepare and the system will automatically, by checking the reservation number, tell the waiter on which table he/she is supposed to deliver the food.

**Reserve Parking while booking Table**
Customers can even reserve a Parking spot while booking a table. So when they arrive at the restaurant they don’t have to look for a parking spot, this results in a much more smooth and stress-free dining experience.

## 7. High-level system architecture and technologies used

Below is a list of the technologies used in Team 3's software stack and technologies used:

- Server Host: Amazon Web Services 1vCPU 1 GB RAM
- Operating System: Ubuntu 22.04 Server
- Database: MySQL v8
- Web Server: NGINX 1.18.0
- Server-Side Language: Node.js
- Additional Technologies:
  - Frontend Framework: React
  - Backend Framework: Express
  - IDE: Visual Studio Code
  - Web Analytics: Google Analytics
  - SSL Cert: Lets Encrypt (Cert Bot)
  - SASS: 3.6.5
- Supported Browsers: Chrome and Microsoft Edge

## 8. Team and roles

Our team is made up of:

| Name | Role |
|---|---|
| Luis Miguel García Marín | Team Leader and Document Master |
| Jesús Moreno Durán | Back End Lead |
| Noman Ali | Front End Lead |
| Paras | GitHub Master |
| Vichitar Dagar | Team member back end |
| Alhassane Dondo Toure | Team member front end |

## 9. Checklist

- Team found a time slot to meet (online) outside of the class: **DONE**
- GitHub master chosen: **DONE**
- Team decided and agreed together on using the listed SW tools and deployment server: **DONE**
- Team ready and able to use the chosen back and front end frameworks and those who need to learn are working on learning and practicing: **DONE**
- Team lead ensured that all team members read the final M1 and agree/understand it before submission: **DONE**
- GitHub organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.): **DONE**