# rRestaurant Management System

## Restaurant Part

### Basic Information

Give the following information of the restaurant: name, company contact information (address, telephone, wechat, qq, e-mail), manager, comments, icon, photos

### Asset Management

(This functionality will be ignored, since it is not significant for a small or middle level restaurant)

Manage the asset owned by this restaurant, like tables, chairs, pots... Each entity generally contain the following information: serial number, name, icon, pictures, price, quantity. We also need to record the event of each asset, like asset buying, asset destruction. Each asset event might contain the following information: asset entity, event quantity (quantity before event, delta quantity, quantity after event), event time, event type (add event), comment.

### Employee Management

Manage employees of the restaurant. For each employee we store the following information: employee number, name, person contact information(address, telephone, wechat, qq, email, id, birthday, age), emergency person contact info, on board date, leave date, leave reason, salary rules, start working time, end working time. Since this part also manage the salary of employee. So here we need to define salary rules as entity. It should contain the following information: salary type(base, deduce, award, workload, hours), amount, employee. To calculate the salary of each employee, we have to log the salary event. The salary event can contain the following fields: employee, salary rule, event start time, event end time, comment.

## Sales Part

### Dish Management

Restaurant needs to maintain all dishes they sell. The following operations should be supported by the dish management system:

* Add new dishes for selling
* Modify information for some dishes, like price, ingredients, descriptions, pictures…
* Delete dishes no longer being sold

#### Dish Ingredient

To integrate with the inventory management, dishes need to main the ingredient. The ingredient table of one kind of dish should at least contain the following information:

* Ingredient information
* Ingredient quantity
* Ingredient UoM(unit of measure)
* Ingredient unitPrice
* Ingredient amount

Since the ingredient unit price was cached in the dishes, we need also to add a flag on the dish header to indicate that some ingredient has changed its price. So that manager could refresh the ingredient unit price and amount manually. Here we don't support automatically updating dish object if one of the ingredient has changed its price. Otherwise, the price changing operation would be with low performance.

#### Dish table

The dish itself is an atomic selling unit. The dish can contain the following information (id ignored here):

* Dish name
* Dish barcode
* Dish UoM
* Dish picture uris
* Dish description
* Dish salesPrice
* Dish ingredient cost
* Dish available status
* Dish profit
* Dish profit percentage
* Dish ingredient table

#### Relationship between Dish and Dish ingredient

Dish owns many DishIngredients. If Dish was removed, then DishIngredient identity must be removed too. And if the dish doesn’t need the ingredient, then the ingredient must also be removed from the database.

All Dishes need to implement an interface which contains the following method:

* CalculateIngredientCost

### Customer Management

### Sales Order Management

### Invoice Management

## Purchase Part

### Inventory Management

### Purchase Order Management

### Supplier Management

## Finance Part

TBD…