

## Documentation Catering Point Of Sale. (POS)



Screenshot mainscreen.

Database structure catering:

List of relations			
Schema	Name	Type	Owner
public	additional	table	postgres
public	article_grouplines	table	postgres
public	articles	table	postgres
public	buttons	table	postgres
public	clients	table	postgres
public	employees	table	postgres
public	groupbuttons	table	postgres
public	invoices	table	postgres
public	loss	table	postgres
public	order_lines	table	postgres
public	params	table	postgres
public	payments	table	postgres
public	purchase_orderlines	table	postgres
public	sales	table	postgres
public	suppliers	table	postgres
public	tables_layout	table	postgres

catering=# \d additional

Table "public.additional"				
Column	Type	Collation	Nullable	Default
addID	integer		not null	
barcode	character varying(13)			''::character varying
description	character varying(13)			''::character varying
item_price	double precision			0
number	double precision			0
item_unit	character varying(16)			''::character varying
article_group	character varying(40)			''::character varying
location_warehouse	character varying(8)			''::character varying

Indexes:

"additional\_pkey" PRIMARY KEY, btree ("addID")

catering=# \d article\_grouplines

Table "public.article_grouplines"				
Column	Type	Collation	Nullable	Default
lineID	integer		not null	0
grouplinertext	character varying(40)			''::character varying

Indexes:

"article\_grouplines\_pkey" PRIMARY KEY, btree ("lineID")

catering=# \d articles

Table "public.articles"				
Column	Type	Collation	Nullable	Default
barcode	character varying(13)		not null	
description	character varying(50)			''::character varying
item_price	double precision			0
item_stock	double precision			0
item_unit	character varying(16)			''::character varying
minimum_stock	double precision			0
order_size	double precision			0
location_warehouse	character varying(8)			''::character varying
article_group	character varying(40)			''::character varying
thumbnail	character varying(50)			''::character varying
category	integer			0
order_balance	double precision			0
order_status	boolean			true
mutation_date	character varying(10)			''::character varying
annual_consumption_1	double precision			0
annual_consumption_2	double precision			0
VAT	character varying(4)			'high'::character varying
short_descr	character varying(15)			''::character varying
selling_price	double precision			0
selling_contents	double precision			0
additional	integer			0
supplierID	integer			0
ordering_manual	integer			0

Indexes:

"barcode\_pkey" PRIMARY KEY, btree (barcode)

"barcode\_idx" btree (barcode)

Referenced by:

TABLE "loss" CONSTRAINT "barcode\_fkey" FOREIGN KEY (barcode) REFERENCES articles(barcode)



catering=# \d buttons

Table "public.buttons"				
Column	Type	Collation	Nullable	Default
buttonID	integer		not null	
buttontext	character varying(60)			''::character varying
barcode	character varying(13)		not null	''::character varying
reference	character varying(5)			''::character varying
accent	integer			0
bg_color	character varying(7)			'#FFFFF0'::character varying

Indexes:

"buttonID\_pkey" PRIMARY KEY, btree ("buttonID")

catering=# \d clients

Table "public.clients"				
Column	Type	Collation	Nullable	Default
clientID	integer		not null	
employee	character varying(20)			''::character varying
barcode	character varying(8)			''::character varying

Indexes:

"clients\_pkey" PRIMARY KEY, btree ("clientID")

Referenced by:

TABLE "order\_lines" CONSTRAINT "clientID\_fkey" FOREIGN KEY ("clientID") REFERENCES clients("clientID")

catering=# \d employees

Table "public.employees"				
Column	Type	Collation	Nullable	Default
barcodeID	character varying(8)		not null	
firstname	character varying(20)			''::character varying
lastname	character varying(30)			''::character varying
access	integer			1
callname	character varying(20)			''::character varying

Indexes:

"barcodeID\_pkey" PRIMARY KEY, btree ("barcodeID")

catering=# \d groupbuttons

Table "public.groupbuttons"				
Column	Type	Collation	Nullable	Default
groupID	integer		not null	0
reference	character varying(2)			''::character varying
buttongrouptext	character varying(60)			''::character varying
bg_color	character varying(7)			'#FFFFF0'::character varying

Indexes:

"groupbuttons\_pkey" PRIMARY KEY, btree ("groupID")

catering=# \d invoices

Table "public.invoices"				
Column	Type	Collation	Nullable	Default
invoiceID	integer		not null	
barcode	character varying(13)			''::character varying
description	character varying(50)			''::character varying
delivery	double precision			0
item_price	double precision			0
supplierID	integer			0
orderlineID	integer			0
paydate	character varying(10)			''::character varying
bookdate	character varying(10)			''::character varying
item_unit	character varying(16)			''::character varying

Indexes:

"invoices\_pkey" PRIMARY KEY, btree ("invoiceID")

catering=# \d loss

Table "public.loss"				
Column	Type	Collation	Nullable	Default
lossID	integer		not null	
number	double precision			0
category	character varying(22)			''::character varying
bookdate	character varying(10)			
barcode	character varying(13)			

Indexes:

"lossID\_pkey" PRIMARY KEY, btree ("lossID")

"fki\_barcode\_fkey" btree (barcode)

Foreign-key constraints:

"barcode\_fkey" FOREIGN KEY (barcode) REFERENCES articles(barcode)

catering=# \d order\_lines

Table "public.order_lines"				
Column	Type	Collation	Nullable	Default
ID	integer		not null	
barcode	character varying(13)			''::character varying
description	character varying(40)			''::character varying
short_descr	character varying(15)			''::character varying
number	double precision			0
item_price	double precision			0
sub_total	double precision			0
sub_vat	double precision			0
mutation_date	character varying(10)			''::character varying
callname	character varying(20)			''::character varying
clientID	integer			0

Indexes:

"tables\_accounts\_pkey" PRIMARY KEY, btree ("ID")

"clientID\_fkey" btree ("clientID")



catering=# \d params

Table "public.params"				
Column	Type	Collation	Nullable	Default
paramID	integer		not null	
item	character varying(20)			''::character varying
value	double precision			0

Indexes:

"paramID\_pkey" PRIMARY KEY, btree ("paramID")

catering=# \d payments

Table "public.payments"				
Column	Type	Collation	Nullable	Default
payID	integer		not null	
kind	character varying(25)			''::character varying
amount	double precision			0
bookdate	character varying(10)			''::character varying
paydate	character varying(10)			''::character varying
instance	character varying(25)			''::character varying
accountnumber	character varying(25)			''::character varying
ovorderID	integer			0
basis	double precision			0

Indexes:

"payID\_pkey" PRIMARY KEY, btree ("payID")

catering=# \d purchase\_orderlines

Table "public.purchase_orderlines"				
Column	Type	Collation	Nullable	Default
orderlineID	integer		not null	
barcode	character varying(13)			''::character varying
description	character varying(50)			''::character varying
item_price	double precision			0
item_unit	character varying(16)			''::character varying
item_stock	double precision			0
minimum_stock	double precision			0
order_size	double precision			0
ordering_manual	integer			0
supplierID	integer			0
bookdate	character varying(10)			''::character varying
ordered	double precision			0
order_date	character varying(10)			''::character varying
delivery	double precision			0
delivery_date	character varying(10)			''::character varying

Indexes:

"purchase\_orderlines\_pkey" PRIMARY KEY, btree ("orderlineID")

catering=# \d sales

Table "public.sales"				
Column	Type	Collation	Nullable	Default
salesID	integer		not null	
receiptnumber	integer			0
barcode	character varying(13)			''::character varying
description	character varying(40)			''::character varying
number	double precision			0
item_price	double precision			0
sub_total	double precision			0
sub_vat	double precision			0
callname	character varying(20)			''::character varying
mutation_date	character varying(10)			''::character varying
short_descr	character varying(15)			''::character varying
clientID	integer			0

Indexes:

"ID\_pkey" PRIMARY KEY, btree ("salesID")

catering=# \d suppliers

Table "public.suppliers"				
Column	Type	Collation	Nullable	Default
supplierID	integer		not null	0
company_name	character varying(40)			''::character varying
street	character varying(40)			''::character varying
houenumber	character varying(14)			''::character varying
residence	character varying(40)			''::character varying
telephone	character varying(13)			''::character varying
email	character varying(200)			''::character varying
addition	character varying(1000)			''::character varying
zipcode	character varying(7)			''::character varying

Indexes:

"suppliers\_pkey" PRIMARY KEY, btree ("supplierID")

catering=# \d tables\_layout

Table "public.tables_layout"				
Column	Type	Collation	Nullable	Default
ID	integer		not null	
occupied	integer			0
table_seat	character varying(4)			''::character varying
clientID	integer			0
callname	character varying(20)			''::character varying

Indexes:

"tables\_layout\_pkey" PRIMARY KEY, btree ("ID")

"fki\_clientID\_fkey" btree ("clientID")

## Point of Sale Catering

### Custom daily operations

The system detects if a logon barcode or a product barcode is scanned.  
When no logon is established processing is blocked.



The red sign in the notification bar above the display is showed.

By valid logon the notification bar shows a green sign.

The logon is invisible with scanning.

When the logged employee logs the barcode a second time, the employee is logged out.

When another employee logs his barcode, the logon is switched towards this employee.

The normal operation starts with logon by barcode.

When customers arrive, the employee push the button “Open Table/Change Tables/Seats”

The table management screen is displayed:



Each button on the screen represents a seat. The green buttons represents tables paired with two seats. The brown buttons represents tables with 4 seats. In total 100 seats are available.

By clicking the buttons the seats are checked as occupied. For groups every combination of tables and seats is possible. For conformation chose “Open new table” (default choice) and push button “Apply Seats”. With this “Open new table” more arrangements scan be made after each other. Chose the occupied seats and push the button “Apply seats” and repeat this for the next client or group of clients.

With applying a popup is showed, asking to print a client barcode. This serves the purpose to switch fast between clients with the printed barcode.

During the stay, it’s possible to add or to free up tables or seats by leaving or arriving members of the group. When no costs are made it it’s possible to remove all seats.

This is established with menu choice “Change table client <number>”, with the number of the client who booked the arrangement. Only customers linked by the concerned employee are showed in the choice fields. For every client combination from the concerning employee is a line added in the menu choices.

When costs are made the last seat cannot be removed in this screen. It will be removed if the payment is done in the mainscreen.

On the buttons for the seats the table and seatnumber, clientnumber and serving employee is showed.

The buttons with there belongings and states are visible for every employee, but only operable for the employee who established the linking.

With the “Switch Employee” button it is possible to transfer the service to another employee by end of working time or interim absence.

If al buttons for the seats are pushed, push the “Refresh” button to make the changes visible, and push “Close” button to leave the tablemanagement screen. Next push the “Select Client” button to select the customers to serve, or for fast selecting or switching between clients use the barcode for the concerning client. Here only the linked clients for the concerning employee can be chosen.

After this steps has been accomplished, it it possible to choose products with the productbuttons, or scanning products. Default a number of 1 is chosen. If more of the same products is required, use the little spinbox to change the number (1 towards 99) before scanning of pushing a button. After scanning of each product or pushing a product button the number of the spinbox is reset to 1.

For return bookings a little button before the number is provided. Push this checkable button before operating and the numbers will be operable from -1 towards -99. After each operation the spinbox will been reset to it's orinignal state, to avoid wrong bookings.

The product buttons are organized as follows:

The 10 maingroups with the double arrows are accessable by pushing the arrowbuttons up or down. Here a picture of the maingroup is presented between the arrowbuttons. With the button above the logo 5 subgroups of each maingroup can be chosen. Next a productbutton can be pushed, possibly preceded with the right number if more then 1 product required. The consumptions and their prices are showed in the little display. Also the total amount of the consumptions is showed. For a clearer total overview push the button “Big Display”

After serving the client, the next client can be chosen, with the barcode, or by the button.

If the client belongs to another employee, first switch the employee with the barcode and next fulfill the client code scan or choose with button “Select client”

Booking additional products:

With selected client it is possible to book additional products. These products (for instance sugar, milkcups, potatoes,vegetables, meat and so on) are not direct attributable to a customer but booked on groupitems as additional costs. The transactions must be established if the products are picked from the storage.

If a client will pay the bill, then chose the concerning client, prints the receipt if desired, with the “Printing” button, by cash paying use the modest calculator on the screen to fill in the cash payed by the customer in the CASH field and push the “REFUND” button. The change will be calculated and showed in the CHANGE field, taking into account the rounding, which is set in advance. The customer also can pay by a pin transaction.

After payment is done press the “TRANSFER PAYED” button to make the necessary bookings and free the tables / seats for this customer. If no costs are made, it's also possible to make a “null” booking to free tables and seats on the main screen.

A notification bar is visible for displaying several notifications during operation.



The following documentation concerns description of the backend for administration reasons and is accessible with level 2 by pushing the “Administration” button.

The structure of the adminmenu:

Employees submenu	<i>New employee</i> <i>View / change employee</i>
Articles submenu	<i>Insert new articles</i> <i>View / change articles</i> <i>Booking loss articles</i> <i>View loss articles</i>
Article-groups lines – view / change	
Purchase submenu	<i>Collecting purchases</i> <i>Manual ordering</i> <i>Unknown supplier ordering</i> <i>Receiving / processing deliveries</i> <i>Ordering suppliernumber suppliersname</i> <i>Ordering suppl...</i> <i>To order yet - view</i> <i>Ordered – view</i> <i>All orders – view</i> <i>Delivered orders - view</i>
Suppliers submenu	<i>Suppliers new</i> <i>Suppliers view / change</i> <i>Invoices suppliers /paying</i>
Imports submenu	<i>Import new articles</i> <i>Import price changes</i> <i>Import expired articles</i> <i>Import deliveries articles</i>
Groupbuttons grouptext /color	
Payments view / pay	
Parameters view / change	
Sales – view	
Additional view	
Turnover submenu – view	<i>Daily gross turnover</i> <i>Monthly gross turnover</i> <i>Yearly gross turnover</i>
Reprint purchase orders	
Reprint sales receipts	

We will describe the menuitems with their purpose and possibilities.

### **Employee submenu:**

With the logon from barcode accesslevel 2 the “Administration” button is activated. With this button and submenu accounts it is possible to generate other barcodes for logon purposes.

The barcodefield will be generated as 5 random digits by the program. (The first 2 digits starts with 24 the 8th digit is a check number). The program will check and correct for duplicates. Access is default set on level 1, change if desired. If accesslevel is set to 0 processing is blocked for the employee.

The first name, last name and callname of the employee must be inserted.

The callname field will be printed on the saleslip, it's also saved in the sales table.

The barodelabel is saved in folder . /Barcodes/Employees/ after inserting a new employee, a barcode can be printed for logon in the menuchoice View / change employee with the print button. Before printing remove the number under barcode for security reasons and replace by logon name.

(For instance use application Paint)

By the menuchoice “View / change employee” click the first field of the list with employees to select.

### **Articles submenu:**

General remarks:

It is important to pay particular attention to the completion of the fields, for reasons of clarity and stock control.

Always fill in description (for purchase) and short description (for receipts and display text),

And surely in the start-up phase the order-size and minimum stock. When the system builds a history of usage, the system calculate the items minimum stock and order-size

In the table articles, the columns minimum stock, category, order\_size, order\_status, order\_balance, annual\_consumption1 and annual-consumption2 are present.

We will explain the purpose of these columns.

The minimum\_stock column is used for purchasing, if  $\text{item\_stock} + \text{order\_balance} < \text{minimum\_stock}$  a orderline is generated. These lines are generated in the “Purchase submenu” Collecting purchases.

If the purchase orderlines are collected the field order\_status is set to False, so no further orders are generated until the products are delivered. The order-balance is increased with the ordered number.

After delivery the field order\_status is reset to True, so the product is released for calculations of stock, order\_balance is reduced with the delivered number.

The column category determine the delivery time. The category runs from 3 days to 1 year in 8 steps and 1 category daily fresh products. The latter is outside the supply system and must be manual handled.

The calculation of this items is established by the Camp formula:

Camp formula:  $Q = \sqrt{2DF/HP}$

Stands for:

Q = Quantity

- D = Demand / year (table articles)
- F = Fixed Costs (order costs conversion costs) (from table params)
- H = Stock costs as a percentage of the price (from table params)
- P = Price of the product (from table articles)

The annual\_consumption is counted by the article usage in a year.

Even year annual\_consumption\_1

Odd year annual\_consumption\_2

By starting of a new year the oldest year of annual consumption is set to 0, so the counting restarts, with this column, so always the last year is preserved for calculation of stock.

It is possible to influence the minimum stock and order size by changing the paramID 5 and 6 (Ordercosts and surcharge for interest and storage)

Columns description, short description, selling\_price and selling\_contents and their usage:

The column description is used for the product as purchased. For instance a bottle Chardonnay white wine 2007 of 0.75 liter. The short description can be in this case Glass white wine.

The item\_price Euro 6.95. The selling\_price 5.00 Euro (1 glass). The selling\_price for the bottle 21.95. The selling\_contents is in this case 0.15 for the glass, and for the bottle 1. The selling by glass can be established with a button coupled to barcodenumber and with 0.15 as selling\_content. By selling a bottle the item can be scanned or sold by button or both.

Articles:

By inserting articles 3 items are important. The checkbox manual, the checkbox additional and inserting articles inserting generated or scanned.

The checkbox manual will be set, if the manager wants to control the ordersize. This is surely in the startup phase important. The orderlines will be generated, but the order-size can be adjusted.

The checkbox additional must be set, when the product are not directly assignable and to calculate to the customer. For instance cleaning products, sugar, milkcups, cookies, potatoes, vegetables, meat and so on. These products will be booked on the several article\_groups for further calculation.

By inserting articles it is possible to use a generated article by the system, or to use a scanned article product in the commercial range. The generated article is an EAN13 number starting with 28 (free, so no conflicts arise with commercial numbers), then 10 digits serial number and last a checksum number as a validity check. (usage for instance own products like maindish, desert, cup of coffee)

When a commercial number is wanted the generated number can be overruled in the last field.

Fill in all items, position the cursor in the last field and scan the product. The product is inserted without pushing the button, with the scanned code.

The other menu choices are for changing articles, booking loss articles, and view loss articles.

Loss articles are divided in obsolete, warehouse differences, damaged or shelf life.

### **Article-groups lines – view / change**

The article-group lines are group classification of products.

The group-lines are to determine by the manager, because it integrated in the database.

Up to 19 lines filled lines will be used by the program.

Click on the first field of the line from the list and change the contents if desired.

### **Purchase submenu**

*Collecting orderlines.*

This menuchoice iterates the whole table articles and generate on contents of the fields item\_stock, minimum\_stock, order\_balance, order\_status and category the orderlines.

The list is stored in the table purchase\_orderlines, with the clientnumber if known and also indicated whether the orderline should be ordered manual.



*Manual ordering*

Represents the products marked as manual\_ordering in the database.

The menuitem opens a list of productlines.

Click on the 1st field of listitem to order, fill in the number of products to order and print the order.

Follow this procedure for the whole list.

The ordered number and date of ordering are stored in the table purchase\_orderlines

*Unknown supplier ordering*

Represents the products with no supplier known in the articlebase.

The menuitem opens a list of productlines.

Click on the field of listitem to order, fill in the suppliernumber and print the order.

Follow this procedure for the whole list.

The ordered number and date of ordering and supplier are stored in the table purchase\_orderlines

*Ordering suppliernumber suppliername*

These lines are represented by suppliernumber + suppliername

The ordering is realised as a collection per supplier.

To order push the Print /Order button on the list per supplier.

An orderlist per supplier is printed.

Follow this procedure for all lines in this range.

The ordered number and date of ordering and supplier are stored in the table purchase\_orderlines

*Receiving / processing deliveries*

This menuline calls all orderlines.

By clicking on the first field of the listed items a form is showed, where the delivered number can be processed. The field order\_balance is reduced and the field order-status is reset in table articles

The delivered number is added and date of delivery is stored in the table purchase\_orderlines

Thurdermore choices for viewing list are available for information.

**Suppliers***Suppliers new*

For inserts new suppliers

Common fields for this purpose.

The zip field is not coupled with check tables, because so many variations over the world exists, that it cannot be realized in this context.

A textform is included for country like requirements like VAT numbers, chamber of commerce numbers and so on (1000 positions possible)

*Suppliers view / change*

For changing items above.

*Invoices suppliers /paying*

These items booked in table invoices after delivery.

The form includes a checkbox to check if invoice is received and approved to pay. The paydate is stored, the checkbox is disabled, the caption is changed in paid.

**Imports**

Imports are done by csv files.

The separate item of the file field is default ',' and is stored as ascii code 44 in table params. Can be changed if the csv field gets another separator character.

The program provides, if headlines with columnnames in the importfiles exists.

The program determines if these headings are present and skips the heading, if provided.

All the present files in the concerning folder will be treated in one go.

If the files are processed, they are moved to same subdirectory under ImportsDone

The importfiles must be stored in folder ./forms/Imports under New, Prices, Expired and Deliveries for processing.

#### *Import new articles*

Articles that have to be added to the table articles

If errors occur – the errors are saved in a file catering\_import.log in your home directory (Windows: /users/<username> ,

Linux: /home/<username> )

#### *Import price changes*

Articles with changed prices. The item\_price is changed in table articles.

If errors occur – the errors are saved in a file catering\_import.log in your home directory (Windows: /users/<username> ,

Linux: /home/<username> )

#### *Import expired articles*

These articles must be removed. The article records in this file are deleted from table articles.

If errors occur – the errors are saved in a file catering\_import.log in your home directory (Windows: /users/<username> ,

Linux: /home/<username> )

#### *Import Deliveries articles*

Number of delivered products is increased by item\_stock in table articles, the order\_status is reset to True and the order\_balance is deducted by the delivered number.

#### **Remark by imports:**

It will be clear that imports are far more efficient then manual processing.

So it is worth the effort to ask your suppliers to operate with this importlists. Mind the layout of the files matches.

#### **Groupbuttons and productbuttons**

For groupbuttons a menuitem is set.

A list of buttons is presented. Clicking on the first fiels of this list opens a groupbuttonform, Insert or change the buttontext maximum 5 lines of 14 characters per line.

As logged on administrator the buttonnumber and reference is showed on the button.

Besides the buttontext also the buttoncolor can be chosen. <https://htmlcolorcodes.com/> is a site where it's very easy to chose the right color with a slider and a cross movable with your mouse. You see the hex color right copy and paste in the color field of the buttonform, don't forget to add # before the colorcode.

If no colorcode or wrong code is chosen the default color is applied #FFFFFFF0.

#### **Productbuttons (900!)**

When logged administrator 1<sup>st</sup> the text on the buttons changed in Buttonnumber, 2<sup>nd</sup> reference and 3<sup>thrd</sup> line barcodenumber if applied. To change, push the button.

Insert or change the buttontext maximum 4 lines of 14 characters per line.

Here it's possible to place a bullet on the 5<sup>th</sup> line for favorite products, by checking the checkbox.

A list of products is showed, browse to your desired product for the button, click on the first field and a buttonform popup.

Besides the buttontext also the buttoncolor can be chosen. <https://htmlcolorcodes.com/> is a site where it's very easy to choose the right color with a slider and a cross movable with your mouse.

You see the hex color right copy and paste in the color field of the buttonform, don't forget to add # before the colorcode.

If no colorcode or wrong code is chosen the default color is applied #FFFFFF0.

For the productbuttons it may be recommended to preserve the first page of buttons for most sold items.

### **Payments – view / pay**

This are the payments for instances, for instance VAT payments.

In the list click first field and then in the opened form check the checkbox to book payment.

The paydate is saved in the table payments.

The list is also for viewing purposes.

### **Parameters view / change**

These are the parameters for the system. To change click the first field in the list.

Parameters:

VAT low: as a factor presented percentage change if desired

VAT high: as a factor presented percentage change if desired

VAT zero is hard coded in the program as it's always 0

CSV separator ASCII separator as ascii code for import files change if csv file separator changed.

EVEN/ODD YEAR leave alone, for the system change this.

ORDERCOSTS PURCHASE costs of administration and handling of 1 order – change if desired

STORAGE SURCHARGE as a factor presented percentage for surcharge stock (costs for storage and loss products)

The last two items influence also the minimum-stock and order-sizes.

CHANGE rounding. The rounding of cash payments by checkout sell.

### **Sales – view**

The viewing list off the realized sale orders.

### **Additional view**

The viewing list of the additional products, products not directly attributable to a customer.

### **Turnover submenu – view**

#### *Daily gross turnover*

Viewing daily total gross turnover + daily total VAT

Insert yyyy-mm-dd in the searchfield,

#### *Monthly gross turnover*

Viewing monthly total gross turnover + monthly total VAT

Insert yyyy-mm in the searchfield

#### *Yearly gross turnover*

Viewing yearly total gross turnover + yearly total VAT

Insert yyyy in the searchfield.

### **Reprint purchase orders**

Give a list of all produced purchase orders. Pick one and print.

### **Reprint sales receipts**

Give a list of all sales. Pick one and print.



For testing is a barcode sheet with logon codes provided in folder ./Barcodes/employees/  
The barcodes for clients starts with 26 and are stored in folder ./Barcodes/clients/  
Some Barcodes for products are stored in folder ./Barcodes/Articles

Three databases are provided in the installation folder.

One partly filled database as default,

One empty database to fill from scratch with your own propriety items.

One filled database with approximily 38000 items for testing purposes, but not assignable to catering.