

## Documentation Cashregister Sales version 2.0

### Screenshot Cashregister:

Artikelnummer	Description	Number	Item_price	Subtotal	VAT
2800000037826	Bell pepper Red	2	1.22	2.44	0.22
2800000037833	Bell pepper Green	4	1.30	5.20	0.47
2800000037840	Cucumber	1	0.91	0.91	0.08
2800000037888	Strawberries	1	3.50	3.50	0.32
2800000037925	Pineapple	1	2.79	2.79	0.25
2800000037857	Kaiser Bun	4	0.46	1.84	0.17
2800000037864	Italian Bol	2	0.70	1.40	0.13
2800000037871	Baguette	1	1.43	1.43	0.13
Total incl. VAT				26.19	2.36

Errormessage: Article not in assortment!

Barcode scan: 8711000008416

Number: 1

Exit

Information

Printing

Next Customer

26.19

30.00

REFUND

3.80

TRANSFER DATA

PAYING

© 2020 all rights reserved dj.jansen@casema.nl

### CashRegister system.

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 below the display is showed.

By valid login the notification bar shows a green sign.

The logon is invisible while scanning.

With the logon from barcode accesslevel 3 a 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 2 startdigits 24 (in free range EAN) , 5 random generated digits by the program and 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 barcodelabel is saved in folder . /Barcodes/Accounts/

For testing purposes a sheet of barcodes is included. (print in high resoltulion for scanning!)

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.

With paying print the receipt if desired, with the "Printing" button, by cash paying use the modest calculator on the screen to fill in the cash paid 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. By typing a wrong number correct with “DEL” button.

After payment is done press the “Next Client” button to make the necessary bookings and free the register for the next customer

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

A INFO button is available for information of the program.

When the account switch from level 1 to level 2 or level 3 or back the orderlines remains, so it's possible to book return goods with accepted account. If the account switches back the spinbox is reset to it's original state. When the login 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.

Accesslevel 0: Operation blocked for the user.

Accesslevel 1. Normal operation. (No administration and no plusminbutton visible)

Accesslevel 2 . Expose an checkable button  $\pm$  for return products. The spinrange from the spinbox changes from 1, 99 to  $-1, -99$  with button checked.

Accesslevel 3. Expose the Administration button. Pressing this button reveals a combobox with 11 menulines. The items are:

Accounts submenu	<i>New account</i> <i>View / change account</i>
Articles submenu	<i>Insert new articles</i> <i>View / change articles</i> <i>Booking loss articles</i> <i>View loss articles</i>
Purchase submenu	<i>Collecting purchases</i> <i>Unknown supplier ordering</i> <i>Receiving / processing deliveries</i> <i>Ordering suppliersnumber suppliersname</i> <i>Ordering suppl.....</i> <i>.....</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>
Payments view / pay	
Parameters view / change	

Sales – view

Turnover submenu – view

*Daily gross turnover*

*Monthly gross turnover*

*Yearly gross turnover*

Reprint purchase orders

Reprint sales receipts

### **Accounts submenu:**

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 barcodelabel is saved in folder . /Barcodes/Accounts/ after inserting a new employee, a barcode can be printed for logon in the menuchoice View / change Account 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 Account” click the first field of the list with employees to select.

### **Articles**

The table articles holds a column barcode (String 13 positions).

By inserting a new article a ean 13 barcode is generated. The first 2 digits are in the free EAN range 28. The following 10 digits is the generated serial number and the last number is a validity check number.

It's also possible with inserting a new barcode to overrule the generated number with a scanned commercial product barcode. So both can coexist to each other.

When inserting by scanning fill in the form completely before scanning and place the cursor in the scanning field.

In this module a image is saved of the barcode in the folder . /Barcodes/Articles

This image can be printed for labeling the product or storage bin in the warehouse, so it's enabled for scanning.

With the program Sales, the sales can be established by scanning barcodes.

The amount can be set with the little spinbox. The amount can be filled with the little arrows, or turning the mousewheel on the field. The amount can be set from 1 to 99. The amount is default set to 1.

The module scans the barcode, looks up in the database the articlenumber, description, price and counts the subtotals and VAT. When scanning is completed, a receipt can be printed.

If a item cannot be provided with a barcode, it is possible to assign the product to a button.

Press the button, after setting the amount if more then one piece required.

See explanation under heading ‘Programmable buttons’.

If scanning is not possible, in case of a damaged barcode, the barcode can be filled manually.

In this case the <Enter> must be pressed on the keyboard.

The program checks if 13 numbers are filled and checks the validity check number.

The module provides a display with heading and 8 product lines.

The lines are scrolling, if more then 8 lines are added.

Below the display screen the totals including VAT and total VAT is displayed.

If the third scan is established the close button is blocked, until next client button is pressed.

Printing of the order is possible until the button next client is pressed.

In the print module the total price and total VAT is counted and printed in the tail heading.

After the next client button is pressed the print button and the next client button is blocked, until the first scan for the next client is established. By scanning the table sales is filled with the receipt number, article number, description, amount, price, subtotal and subVAT, also in the table articles the stock data is updated and the annual\_consumption is increased.

With pressing the next client button the table afdrachten (payments) is filled with the totals of VAT.

By scanning is checked on 5 error conditions:

1. The checksum of the barcode is wrong. (wrong barcode or damaged)
2. The product is not in the range of the company.
3. Too little stock for the orderline.
4. Error message if not logged on.
5. Operation blocked for this user.

The errors are showed below the product display in the color red.

With choice 3, also the amount of current stock is showed.

For return goods a  $\pm$  button is added.

This button is visible if the logon barcode is valid as code 2 or code 3 . See **Accounts**.

If the button (checkable) is set on – the spinbox range change from -1 to -99.

The logon can be switched from access 1 to access 2 or 3 and back with remaining of the orderlines, so it's possible to book return goods by a competent person.

## Articles Request/change

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 and surely in the start-up phase the order-size and minimum stock. When the system builds a history of usage, the system calculates 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 available for stock control.

We will explain the purpose of these columns.

The minimum\_stock is used for purchasing, if  $\text{item\_stock} + \text{order\_balance} < \text{minimum\_stock}$  a orderline is generated. These orderlines 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 decreased and item\_stock is increased with the delivered number.

The column category determines 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 handled manual.

Minimum stock is calculated on base of annual\_consumption and category (delivery time).

The calculation of the 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.

With every issue of products the annual\_consumption is increased.

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 the column, so always the last year is preserved for calculation of stock.

You can influence the order size by changing the paramID 5 and 6.

(Ordercosts and surcharge for interest and storage)

A thumbnail picture can be showed in the articles base productlines. Place a picture as a png file 80 x 80 pixels in folder thumbs (use Paint for scaling and save as png) and set the path in the field thumbnail as ./thumbs/<name of picture> It is possible to make subfolders with picturegroups in the folder thumbs, for organisational reasons.

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.

Clicking on the first field of the tableview opens a form loss.

With this form the warehouse differences, damaged products, obsolete products or shelf life products can be written off. Fill in number of products and choose category of loss. The products are booked in the table loss and reduced from stock.

## **Purchase submenu**

### *Collecting orderlines.*

The menuchoice iterates the whole table articles and generate based 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.

### *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 the procedure for the whole list.

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

The suppliernumber is also stored in the articles table.

### *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.

A orderlist per supplier is printed.

Follow the procedure for all lines in the range.

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

#### *Receiving / processing deliveries*

The 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 item\_stock is increased, 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.

View lists for all orders, still to be ordered, ordered and delivered orders are present.

### **Suppliers**

#### *Suppliers new*

The zip field is not coupled with check tables, because so many variations over the world exists.

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 are booked in table invoices after delivery.

The form includes a checkbox to check if invoice is received and approved to pay.

By checking this box the payment is booked with the paydata. After this booking is done the checkbox is disabled for checking and the text from the box is changed in payed.

### **Imports**

Imports are done by csv files. (most used import standard)

The program determines if headings are present in the importfiles 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.

To import files must be stored in the folders ./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 sales\_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 sales\_import.log in your home directory (Windows: /users/<username>,

Linux: /home/<username>

#### *Import expired articles*

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

The deleted products are stored in table loss as Obsolete.

If errors occur – the errors are saved in a file sales\_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.

Purchase\_orderlines fields delivery and delivery\_date are updated.

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

### **Remark by imports:**

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

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

### **Payments Request/Paying**

With this menuchoice a tableview from the payments towards instances sorted on not paid and receiptnumber is shown.

With this menuchoice it is also possible to book a payment towards instances by clicking on the third field from the payments request tableview. This action opens a payform with a checkbox pay. By checking this box the payment is booked with the paydata. After this booking is done the checkbox is disabled for checking and the text from the box is changed in paid.

### **Programmable Buttons.**

Accessible with the admin button (Level 3)

There are up to 8 buttongroups each with 23 programmable buttons with the third button (colored differently) as a group button to switch between the buttongroups.

The groupbuttons can be provided or changed with your proprietary grouptext in the table params paramID 8 – 15, for groups 1 to and included 8.

Buttontext until 10 positions (plus one linefeed) per line over maximum 3 lines.

Besides the buttontext also the buttoncolor foreground and background 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 #F8F7EE.

It is possible to change the number of buttonpages from 1 to 8.

Change in params the value in 1 for the buttonpage, following on the stop page. For instance if you change for buttongroup 4 the field value in 1, you get 3 pages of buttons. To undo this action change value back to 4 and you will have 8 pages of buttons. The changes become visible after a restart of the program. In this way you will be able to use from 1 to 8 buttonpages.

**Productbuttons:**

If logged on as administrator and after refreshing the display, the buttons shows the buttonnumber and the linked barcode if provided. When clicking the button, the articles table is showed. Browse to the wanted product to link and click on the first field, the product is chosen for linked product and a buttonform for the clicked button opens. Besides the buttontext also the buttoncolor foreground and background 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 #F8F7EE.

The buttontext is until 10 positions per line (plus one linefeed) over maximum 3 lines.

If a button with a existing text is choosen the text is replaced and the button is linked towards the new barcodenumber.

The new buttontext becomes visible after refreshing the buttonpages.

**Parameters.**

With this menuchoice a tableview from the params table sorted on paramID is shown and with clicking on the thirst field you can change item and value and for the groupbuttons buttontext and colors and number of buttonpages. (See "Programmable Buttons")

**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

EVEN/ODD YEAR leave alone, for it is handled by the system.

RECEIPTNUMBER leave alone, for it is handled by the system

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.

The parameters 8 - 15 are respectively for buttongroup 1 - 8, buttoncolor and adjusting number of buttonpages. For further explanation see "Programmable buttons".

**Sales – view**

The viewing list of the realized sale orders

**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 the folder ./Barcodes/employees/

A barcode sheet for products is stored in the folder ./Barcodes/Articles

Two databases are provided in the installation folder.

One filled database with approximately 38000 items for testing purposes, but not assignable to sales

One empty database to fill from scratch with your own items, only the tables for program functions are filled.

As a database system is used posgreSQL, which make it possible to use this application as a client-server application with several point of sales. I have not tested this because i don't have a network available. But with the right IP configuration this should be a piece of cake with a little knowledge of networking. The server can easily host on Linux. The application sales.py is ported and tested for Windows and Ubuntu (might work with other Linux operation systems too).

## Database cashregister:

```
cashregister=# \d
```

List of relations			
Schema	Name	Type	Owner
public	accounts	table	postgres
public	articles	table	postgres
public	buttons	table	postgres
public	invoices	table	postgres
public	loss	table	postgres
public	params	table	postgres
public	payments	table	postgres
public	purchase_orderlines	table	postgres
public	sales	table	postgres
public	suppliers	table	postgres

(10 rows)

```
cashregister=# \d accounts
```

Table "public.accounts"				
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")
```

cashregister=# \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(6)			''::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(20)			''::character varying
selling_price	double precision			0
supplierID	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)

cashregister=# \d buttons

Table "public.buttons"				
Column	Type	Collation	Nullable	Default
buttonID	integer		not null	
buttontext	character varying(30)			''::character varying
barcode	character varying(13)		not null	''::character varying
fg_color	character varying(7)			'#000000'::character varying
bg_color	character varying(7)			'#F8F7EE'::character varying

Indexes:

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

cashregister=# \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
item_unit	character varying(16)			''::character varying
supplierID	integer			0
orderlineID	integer			0
paydate	character varying(10)			''::character varying
bookdate	character varying(10)			''::character varying

Indexes:

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

```
cashregister=# \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)
```

```
cashregister=# \d params
```

Table "public.params"				
Column	Type	Collation	Nullable	Default
paramID	integer		not null	
item	character varying(20)			''::character varying
value	double precision			0
buttongroup	character varying(30)			''::character varying
bg_color	character varying(7)			'#FBF7EE'::character varying
fg_color	character varying(7)			'#000000'::character varying

```
Indexes:
```

```
    "paramID_pkey" PRIMARY KEY, btree ("paramID")
```

```
cashregister=# \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

```
Indexes:
```

```
    "payID_pkey" PRIMARY KEY, btree ("payID")
```



cashregister=# \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
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")

cashregister=# \d sales

Table "public.sales"				
Column	Type	Collation	Nullable	Default
ID	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

Indexes:

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

cashregister=# \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
houzenumber	character varying(14)			''::character varying
zipcode	character varying(7)			''::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

Indexes:

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

### General remarks by the used database items.

The items are not specific for retail, nor amounts and prices are realistic and for becoming real values the data should be applied to a working company and the data should be dynamic.