

A Drupal Beginner's Exercise Book - 1

- Foodstuff purchase calculation of cafeterias -

2021.6.15 Japon Frog Kero

1. Introduction

"Drupal" is one of the easiest way to create a homepage or blog on the Internet.

This book is an introduction to drupal's rupal rupal ruding app development examples.

Specifically, we will develop "X Foodstuff Purchase Calculation App for Making Y Dishes ". This document only deals with some of the basic uses of Drupal.

In this book, not only Drupal but also tools such as HTML, JavaScript, CSS, etc. appear, but each time it comes out, it is fine to be an introduction level, so please check it on the web or something and study it.

This book is also very practical in not appearing in the Japanese version of Drupal Introductory (asof May 28, 2021), so I think it will be helpful for those who have suffered a setback with Drupal along the way.

This app will be developed on a PC(MacBook Air early 2015) instead of a smartphone, but I think it will work on Windows PCs.

2. Install Drupal

First, you'll want to make the latest version of Drupal "Drupal 9" or the previous version of "Drupal 8" available.

If you search for "Drupal Install" in your browser, you will see countless search results.

Once you've found a specific and easy-to-understand description that fits your PC(Mac or Windows), install it accordingly.

There are also two places to install Drupal: "local host"(your PC) and "server" (an overseas Drupal support company), depending on the description. When you install Drupal on a server and develop an app, you can run it not only on your PC, but also on your own smartphone or other people's PCs and smartphones.

Once you've installed it on your PC, you'll be dealing with your Drupal site on that PC alone.

If you search the web, overseas service company versions such as Pantheon and Aquia, as well as domestic companies such as Studio-umi and ANNAI, are also doing installation. There are also a variety of installation explanations, so try your hand at doing what you like.

In this book, you will frequently click menu tags, phrases, buttons, etc. on the screen.

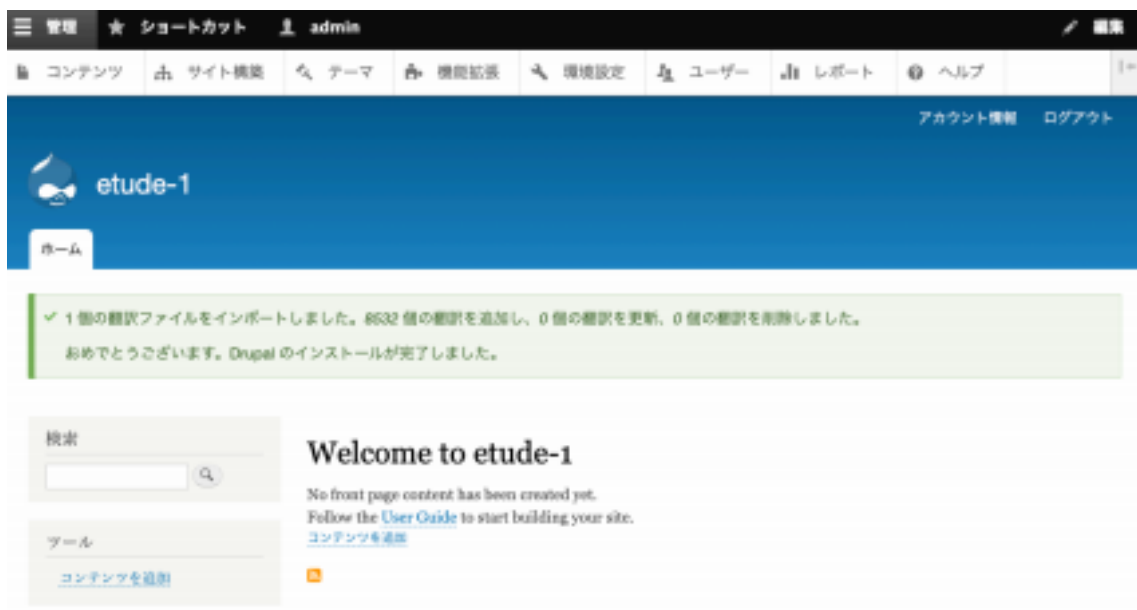
button that says "→" will move to the next image surface.

Note that "→" takes some time .

Drupal9 was introduced in this document, but the following special operations were required: After Drupal installation is complete, go to Settings → Maintenance Mode in Development → Put Sites in Maintenance Mode , and save Configuration → Extend → Uninstall → □ Update Manager ✓ Remove → " → Online " → " ✓ Put Site in Maintenance Mode " ✓ Save Configuration → [Enhancements] → □ Update Manager) and [Install]. You should now be available to install new themes and modules. In any case, if the installation is successful, you will see a screen similar to Figure 1. ✓

Congratulations. Drupal installation is complete. It will be displayed. The site name of this application was set to "etude-1" in the installation procedure.

[Figure 1: Initial screen]



3. Application creation plan

Outline the steps for creating an app.

(1) Change of theme

Drupal has basic design "headers" (consisting of top blue-color bands, logo marks, menu tags, etc.) and "footers" (displaying acknowledgments and destinations), which are called "themes." On the initial screen, "Baltic 8.9.13" ("Baltic")

theme, but you can change it according to your preferences. This document uses a common theme called Bootstrap.

2

(2) Creating a front page

Under the "Welcome to etude-1" display on the initial screen in Figure 1, you can see the message: "The front page content has not yet been created. The sentence is attached small. First, create a front page (= formal initial screen).

(3) Create vendor notes

Add an identification symbol to familiar vendors who are always buying ingredients, and register the store name, TEL, fax, and person in charge.

(4) Creation of food notes

Individual identification symbols are attached to ingredients, and vendors and unit prices are registered.

(5) Creation of cooking menus and recipes

Add a separate identification symbol to the dish and register the name, image, price, ingredients used, and the recipe.

(6) Create order notes

When an order is placed, it records its contents and displays the progress of cooking and serving to the owner and employees in an easy-to-understand way.

(7) Creation of a formula for calculating the amount of food

If you have a catering reservation, you need to calculate the ingredients you need and their quantity based on the recipe, rather than "intuition and experience", and calculate which vendors to order them from, in order to cook them without waste.

(8) Creation of purchase orders for purchasing ingredients

This is what this book is about.

Print out the results calculated in (7) and create a "purchase order" screen that can be ordered by faxing to the supplier.

This book is designed to be read through trial and error by actually moving your hand toward the PC.

Drupal It took me about two years to reach the level of this book, but I still know only a small part of Drupal. In this book, only what has been done well so far is culminated, so even beginners will be able to complete the "food purchase order" in less than a week. I think that it is good to study Drupal further by using the success experience as a spring.

4. App creation

(1) Change of theme

Change to the theme "Bootstrap 3" (Bootstrap Three). Click Themes in the Site

Administration tool bar ("Content, Site Building, Themes, Enhancements, Environments, Users, Reports, and Help" at the top of the header in Figure 1) and you're presented with a blue button called "+ Install New Theme"). "Set the default theme for your website. Another theme is available. When you click the letter [Theme], the following screen (Figure 2) will appear. English, but don't be afraid.

[Figure 2: Drupal theme selection screen]



Scroll down and select Bootstrap. It changes to a similar screen for Bootstrap, with the section Downloads at the bottom (Figure 3).

[Figure3: Download Bootstrap]



In the latest version of "8.x-3.23 released 14 June 2020", right-click tar.gz (249.63KB), and then left-click Copy Link (address).

In the browser, click ← (or "<") Drupal Go back to the initial screen of the site management tool Select Themes in the bar and click + Install New Theme→"Next URL Inn from

4

Paste the address of the link above in the "Stall" column, and click [Install] at the bottom.

When the installation is successful, you will see a screen called "Update Manager". In the Next Steps section, click Install a newly added theme.

Return to the Themes screen.

"Uninstalled theme" has the "Bootstrap 3" of the earlier. Install this and set it to default. This will then move to the beginning of the Installed Themes. Click Settings to the right of the theme chart. Bootstrap will appear.

Press the Save Configuration button at the bottom.

At the top, click "[✓ Settings option saved.] is displayed. If you click "Back to site" in the top menu, you will see a screen that is much easier thanBaltic. This isthe initial screen for Bootstrap 3(Figure 4).

[Figure4: Bootstrap Theme Initial Screen]



A black cabbage-like logo is the symbol of Bootstrap3 for Drupal. Let's change this to your own logo.

On the toolbar, select Themes, and then click Settings inBootstrap 8.x 3.23, which is an installed theme.

Thebootstrapsettings screen will appear.

Click [Logo Image] at the bottom.

If you uncheck "Use logo provided by theme", the column "Upload logo image" will appear, so under [Select file], put your own logo image

5

You can. Jpeg and png files are available. The size of the logo changes depending on the file size. If it is jpeg format, I think that it is good in size of about 1 to 2KB. Click Save Configuration at the bottom→(after→)and click Back toSite. Next, change the "site name".

Click Preferences on the toolbar → basic site settings in the System section. Change the"etude-1"in the "Site Name" column to the appropriate name (e.g., "Marumaru Shokudo"), click [Save Configuration] at the bottom, and [Return to site]. If you do, the background color of the header is white, and it is difficult to distinguish it from the area below it, so I will change the background color of the header.


(1)-1 Change theme background color (use CSS)

Add a module calledAsset CollectortoDrupal andwriteCSS with atool

called CSSCollector in Asset Collector. CSS is the Language for specifying the style of a web page. Please study a little on the net.

(1)-1-1 Introduction of the module "AssetCollector"

If you add a new tab in your browser and search for the word "Drupal assetinjector module download", you will find "Asset Collector" in the search | Drupal.org Nov 24, 2015 -- ... I think that there is a site called .

Click it and scroll all the way down the screen where it appears, followed by  in the yellow green square in the Downloads column, followed by a 青 color of "tar.gz (24.67KB)". Right-click on it to "copy the address of the link" (or "copy link").

Return to drupal's administration screen, click "→" in the Site Administration toolbar, and then click Install New Modul.

Paste the address above in the "Install from next URL" column on the "Install new module" screen, and click [Install].

✓ Installation completed successfully. When you see "Next steps" section, [click Enable newly added modules](#) . To return to the "Extend" screen, check the "☐ Asset Collector" added at the bottom, and click "Install" at the bottom.

When the installation is complete, click "Environment" on the site management toolbar. The item "Asset Collector" has been added to the "Development" column. When you click on it, you'll see that you can use two functions: CSSInjector and JS Injector. "JS" is "JavaScript".

(1)-1-2 Use "CSSCollector" in "AssetCollector"

Click +Add Css → in the CSS Injector section.

In the Label field, for example, type myheader.

In the "Code" column, the following code (Figure 5) is written.

6

Note that "/* ... */" is a comment, so you don't have to write it.

Figure 5: Decorating headers with CSS. (create.css myheader)]

```
#navbar { /* header("navigation bar") */ background: #b1f9D0; /*
Background color to #b1f9D0 (light green)*/
}
About .navbar-brand { /* header brand name (Marumaru Shokudo) */ font-size:
230%; /* Font size doubled by 2.3 */
font-style: italic; /* Italic fonts */
}
.menu--main li { /* About menu tags */
margin: 3px; /* Outer frame thickness to 3px */
border: 1px solid #000000; /* Outline thickness 1px, solid line, color in black */
background: #ffffcc; /* Background color to #ffffcc (light beige) */ }
```

And [Save].

Back to Site and the header is shown in Figure 6.

[Figure 6: Header change result]



(2) Creating a front page

However, the initial screen in Figure 6 still says, "The front page content has not yet been created. It is displayed.

So we create a front page (initial screen).

The CreateBasic page → the BasicPage → the → Page from the Content toolbar.

The "Title" field will change the phrase displayed as "Welcome" on the initial screen. For example, "SanzanKai taste! Sake! Groping in the dark! Prayer! I put it.

7

In the Body field, you can write words and sentences, or paste image files for pictures and photos.

When you click the "Mountain and Sun" icon on the line that says "Source", the [File Selection] button will be displayed, so select it from the PC. You can select the layout of the page, left, right, center, etc., so please try it.

Click "URL alias" on the right side of the screen, and the "URL alias" field will be displayed. Enter /node/here and click Save.

Now the initial screen (front page) is now available, but the "Search Field" and "Tools" on the right side of the screen are unnecessary this time, so erase them.

(2)-1 Clear "Search field" and "Tools" on the right side of the screen

Site Build → Block Layout.

For Search and Tools in the second block, click ▼ in Settings ▼ to the right of the line and select Disable. → Page Save Block → Back to Site. The initial screen is now complete (Figure 7).

[Figure 7: Completed version of the initial screen (example)]



(3) Create vendor notes

From now on, I will make some "notes (ledgers)".

As a Drupal, all of them are basically formatted by "Content Type", entered individual data in "Add Content", and listed in "View". The goal of this book was "X Y Food purchase calculation app for making dishes in total".

8

Imagine that.

You are the chef of Marumaru Shokudo.

From one banquet hall," On the specified date, bring this dish to 20 people by 12:00 p.m. “.

Now you need to stock ingredients. "What to buy from where and how much?" It is a question.

Write a list of your usual vendors in your notes in advance. Let's call this a vendor note.

Are you required to have "vendor name", "TEL", "fax", "person in charge"? You'd like to display them at the beginning with an appropriate identifier ("Store ID"). Format is shown in the header portion of Table 1, and "individual data" (example) is shown in the second line and later.

[Table 1: Vendor Notes]

Store ID (identification number)	Vendor name (simple string)	TEL (simple string)	FAX (simple string)	person in charge (simple string)
s01	Greenman O7	03-3502-8111	03-3502-8112	Kichisaburo

s02	TakoHachishaku Fish Shop	04-4613-9000	04-4613-9001	Samejima
s03	Yamaneko Meat Shop	05-6789-0123	05-6789-0124	Inokuma

(3)-1 Creating "content type" for vendor notes

First, create a "content type" (formatting).

Toolbar Site Building→Content Type→+ Add Content Type. If you enter "Vendor" in the "Name" field, "System internal name: syruxian [Edit]" is displayed on the right. Since syruxianiserrebeable, click Edit to rename the "SystemInner Name" column to supplier.

Also, enter "Store ID" in the "Label intitle field" field. In [Save and manage fields], it will be on the "Manage fields" screen. ★the "Add+ field→select "Text (plane)" in the "Add new file" column. (★ is a recurring point.)

If you enter "Vendor name" in the "Label" field, "System internal name: field_shiruxian ming [Edit]" will be displayed on the right, so click [Edit] to change the name of the "System internal name" column to "field_supplier_name". Please enter the underbar properly.

→ Savenext screen→"Set fields" screen, but →"Save field settings"→SaveSettings.

Similarly, enter TEL, fax, and contactperson(see★above). The system internal names should be field_supplier_tel field_supplier_fax" and field_supplier_person"".

This is the end of creating the vendor notes content type.

9

(3)-2 Enter "individual data" in vendor notes

If you select→+ Add Content→The CreateDestination screen.

Enter the data as shown in Table 1.

For example, inthe"Store ID" column, "s01", "Supplier Name"column is "Green shop7", "TEL"column is "03-3502-8111", "Fax"column is "03-3502-8112", "Person in charge" columnis "Yoshizaburo", and → [Save]→ Individual data entered earlier is displayed.

Similarly, enter individual data for s02 and s03.

(3)-3 View vendor notes

Now that you've entered the vendor information according to the content type Vendor, let's list them.

Use a feature called "View".

In the Toolbar Site→,→ View,add+view→the AddView screen. In the "Name of view" field, if you enter "Vendor note", the system internal name will be displayed. Change to a phrase (in English) that you can understand in Edit. For example,supplier_note. In

the "View Settings" field, the display remains "Content", the type specification is "Vendor", and the order is "Title (=ID)".

In the "Page Settings" ☐ , put ☒ in "Create a page".

The Page title field remains in vendor notes.

Leave the Path field as well as the name of supplier_note "Path". In the "Page display settings" column, the display format is set to "Table". Leave the "Number of displays" field blank and uncheck ☒ "Use pager". ☐ ☒ in Create Menu Link.

Leave the Menu and Link Text fields intact, and go to the Save → view edit screen.

Scroll down to see how it's done in Preview. Currently, the title column is only displayed vertically as s01, s02, and s03. Since these are "store IDs", change the "title" to "storeId".

In the "Field" section in the "Display" column above, blue is written in the letter "Content: Title (Tie-to-Use)". Click title in this parentheses. A small screen appears and says "Title" in the "Label" column, so change this to "Store ID" and [Apply]. When you return to the edit screen of the view and look at the preview, the item name is "StoreID".

Let's also display other information entered in the content type "supplier". If you click "Add" in the "Field" section earlier, a small screen of "Add Field" will appear.

If you enter supplier in the Search field, only the line related to supplier is displayed. If you look at the title, you have entered everything except the "body", so check ☐ on the left side of these items and click [Add field and set].

10

The settings screen for each field appears, but without doing everything, click Apply and continue. The last button is the Apply button. → Return to the edit screen. If you look at the preview, you will certainly see all the data you have entered about the vendor. However, it seems better to change the order.

In the Field|, under → ▼ ▼ ▼ , a small screen "Sort Fields" appears.

A "cross arrow" appears on the left side of each line. You can sort them by dragging them. The order is "Store ID", "VendorName", "TEL", "Fax", "Person in charge", and then returns to the edit screen under Apply.

If you look at the preview, you can see the list you want.

[Save].

Try [Back to site].

The header has a menu tag called Vendor Notes.

Click this to see the list of vendors as seen in the preview earlier (Figure 8).

[Figure 8: "Vendor Notes" screen]

管理

★ ショートカット

admin

コンテンツ

サイト構築

テーマ

機能拡張

環境設定

ユーザー

レポート

ヘルプ

中丸丸食堂

ホーム

仕入先ノート

[アカウント情報](#)
[ログアウト](#)

仕入先ノート

ID	仕入先名	TEL	FAX	担当者
s01	八五郎お七	03-3802-8111	03-3802-8112	吉三郎
s02	たこ八野魚店	04-4813-9000	04-4813-9001	鈴木
s03	山本コシ肉店	05-6789-0123	05-6789-0124	田中

Powered by Drupal
[コンタクト](#)

(4) Creation of food notes

Next, make a "food note".

The point is almost the same as the "vendor note", but it also includes new ones.

The required items are "food name", "unit price of 100 g of food (yen)" ("100 g unit price"),and "supplier".

These will be displayed at the beginning with an appropriate identifier ("Food ID"). The unit price of 100 g will be a constant price without fluctuation for the time being.

Format is shown in the header portion of Table 2, and "Individual data" (example) is shown in the second line and later.

11

[Table 2: Ingredients Notes]

Ingredient ID (Title)	Ingredient name (Plain Tequis))	100g unit price (number (integer))	Vendor (entity reference)
f001	onion	36	s01
f002	potato	42	s01
f003	carrot	48	s01
f004	Ikageso	128	s02
f005	Octopus Feet	256	s02
f006	amberjack	512	s02
f007	Chicken Momo	77	s03
f008	Pork belly	222	s03

f009	Beef Loin	444	s03
------	-----------	-----	-----

Table 2 includes a store ID as data for Vendor, but if you enter it using the entity reference type of input method, you can follow this store ID and access the data that you entered in vendor notes, such as the vendor name.

(4)-1 Creation of "content type" of food notes

First, create a "content type" (formatting).

Toolbar Site Building→Content Type→+ Add Content Type. If you enter "Ingredients" in the "Name" field, "System internal name: shikai[Edit]" will be displayed on the right. Shikai is errable, so click Edit to rename the System InternalName field to foodstuff.

Also, enter "Ingredient ID" in the "Label in title field" field. In [Save and manage fields], it will be on the "Manage fields" screen. Select "Text (→)" in the "Add New Field" field in the Add+ Field.

If you enter "Food name" in the "Label" field, the right side will say "System internal name: field_?????? Edit" is displayed, so click Edit to rename the System Internal Name field to field_foodstuff_name.

→ Save next screen→"Set fields" screen, but →"Save field settings"→Save Settings. Next is the setting of the field "100g unit price".

Add + Field→ Under Select Field Type, click ▼ →, click Number (integer).

Labels are "100 g unit price" and the internal name of the field is field_foodstuff_unitprice"100%". →Save and Next→Save Field Settings→Save Settings. Next, set up the field Vendor.

12

(4)-1-1 Setting of "Entity Reference Field"

An entity reference field is a field that can reference (access) information of other content types by following the data that you put into that field.

This works in views. Figure 9 explains the principle. Now I'm trying to create a "content type A view". If content type A has field a1 of entity reference type, and the data in it is the same as the data in title in Content Type B, and "Content referenced by field a1" is added as a relation to the view, you can see all the field data of content type B. In addition, this feature can be layered so that content type C can be referenced in Figure 9.

[Figure 9: View of Content Type A with EntityReference Field]

コンテンツタイプ「A」

フィールド名	タイトル	フィールド a1	フィールド a2	...
フィールドタイプ	プレーンテキスト	エンティティ参照	その他	...
データ	:	:	:	:
	a101	b34	:	:
	a102	b98	:	:
	:	:	:	:

(「『A』のビュー」のリレーションに「フィールド a1 から参照されているコンテンツ」を追加)

コンテンツタイプ「B」

フィールド名	タイトル	フィールド b1	フィールド b2	...
フィールドタイプ	プレーンテキスト	その他	エンティティ参照	...
データ	:	:	:	:
	b98	...	c008	:
	b99	:
	:	:	:	:
	:	:	:	:

(「『A』のビュー」のリレーションに「フィールド b2 から参照されているコンテンツ」を追加)

コンテンツタイプ「C」

フィールド名	タイトル	フィールド c1	フィールド c2	...
フィールドタイプ	プレーンテキスト	エンティティ参照	プレーンテキスト	...
データ	:	:	:	:
	c008	...	お宝	:
	c009	:
	:	:	:	:

This feature eliminates the need to keep vendor names, faxes, and personnel in the content type "Ingredients" in duplicate with the content type "Vendor".

13

Now, it is set to "Vendor", but in → "Select fieldtype", click → [Content].

The "Label" field is entered as "Vendor". The internal name of field_foodstuff_supplier is .

If you click Save and →, save field settings → the Vendor settings for ingredients screen. "Content type" in the "▼ Reference type" column is checked "□ Vendor". → settings settings.

You now have a field of entity reference type. It becomes the "Manage fields" screen, and the fields that you have set so far are displayed. The order of the wheels may not be in the order you typed them, but don't worry.

(4)-2 Entering "individual data" for food notes

If you select → Content, add+ →, and select Ingredients, you will be on the Create Food screen.

Enter the data as shown in Table 2.

For example, in the "IngredientsID" column, "f001", "Ingredient name" column is "Onion", "10 g unit price" column is "36", "Supplier" column is "s01", and →[Save]→ Individual data entered earlier is displayed. Similarly, refer to Table 2 to enter individual data for f002 through f009.

(4)-3 Display of "view" of ingredient notes

Now that you've entered ingredient information according to the content type Ingredients, use views to list them. The operation is almost the same as for vendor notes. In the toolbar[site →] and from →[Add + view], →[Add view] screen. In the "Name of view" field, type "Ingredient Notes" to display the internal name of the system. Change to a phrase (in English) that you can understand in Edit. For example, foodstuff_note. In the "View Settings" field, the display remains "Content", the type specification is "Ingredients", and the order is "Title (=Ingredient ID)".

In the "Page Settings" ☐ , put ☒ in "Create a page".

The Page title column remains in the Ingredients Note.

Leave the Path field as "Path" as foodstuff_note the view. In the "Page display settings" column, the display format is set to "Table". Leave the "Number of displays" field blank and uncheck ☒ "Use pager". ☐ ☒ in Create Menu Link.

Leave the Menu and Link Text fields intact, and go to the Save → view edit screen.


In Preview, the title column displays f001, f009, and so on. Since these are "food IDs", change the [Title] of the field to "IngredientID". It also displays other information entered in the content type "foodstuff". If you click "Add" in the "Field" section earlier, a small screen of "Add Field" will appear.

14

If you type "food(partial string of foodstuff)" in the Search field, only the line related to "foodstuff" is displayed.

If you look at the title, you have entered everything except the "body", so check ☐ to the left of these items and click [Add field and set].

The settings screen for each field appears, but without doing everything, click Apply and continue. The last button is the Apply button. → Return to the edit screen. When you look at the preview, the name of the ingredient is displayed at the end, so sort the columns so that they look like table 2.

If you look at the preview, you can see a list of the data you entered. Next, you want vendors to display vendor names, not IDs. On the right side of the view edit  , → to Add in relationships → ☐

field_foodstuff_supplier check "Content" referenced from →" check "Add relation" → Set to Apply.

Return to the edit view, so change the "Do not use relation" in the "Add" → "Vendor name" in the "Field" section → [Add field to set] → "Do not use relations" in the "Relationships" field to "field_foodstuff_supplier: Content" → [Apply].

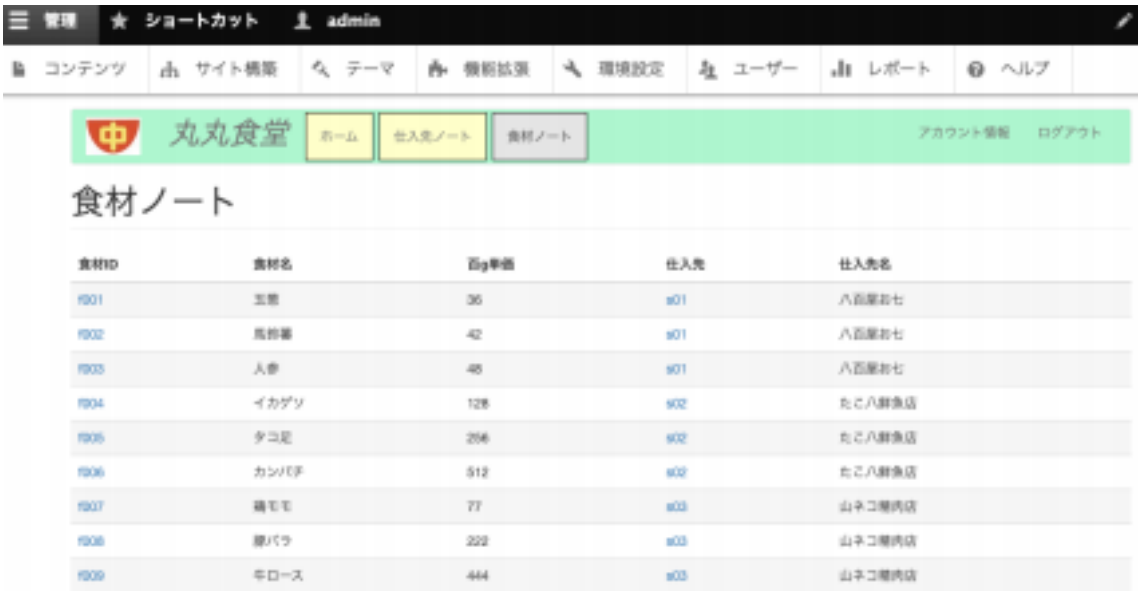
Return to the view edit screen, so check the preview. The vendor ID and vendor name are displayed correctly.

[Save].

Try [Back to site].

You can create a menu tag called "Ingredient Notes" in the header, and click it to display the list of ingredients as seen in the previous review (Figure 9).

[Figure 9: "Ingredients Note" screen]



食材ID	食材名	価格単位	仕入先	仕入先名
1901	玉葱	36	s01	八百屋お七
1902	長ねぎ	42	s01	八百屋お七
1903	人参	48	s01	八百屋お七
1904	イカダシ	128	s02	たこ八軒魚店
1905	タコ足	256	s02	たこ八軒魚店
1906	カンパチ	512	s02	たこ八軒魚店
1907	鶏もも	77	s03	山手二階肉店
1908	豚バラ	222	s03	山手二階肉店
1909	牛ロース	444	s03	山手二階肉店

15

(5) Creation of cooking menus and recipes

"Cooking menu & recipes" is also content, but this time I will try another way. Enter data in bulk using an external file. Image files are also bulk from the outside Put.

(5)-1 Creation of "content type" of cooking menus and recipes

Just like previous "suppliers" and "ingredients", first create a content type. The name of the content type is "Dish" and the internal name of the system is "dish". The label in the title field is "Cooking ID", for example: d01, d02,... And number. The additional fields are the items in the table below. For each item, indicate the field type, label, system internal name, and points to note.

order	Field type	label	System internal name	Points to keep in mind
1	Plain text	Name of dish	field_dish_name	
2	image	Cooking image	field_dish_image	File type:jpeg, png, gif The file directory is"myimage"
3	Number (integer)	Unit price (Yen)	field_dish_unitprice	

4	Plain text	a blurb	field_dish_salestalk	
5	content	Ingredient ID	field_dish_foodstuff	Number of values to allow → unlimited, Reference type is "Ingredients", sorting criteria are "ID",direction "Ascending"
6	Number (integer)	Amount of ingredients	field_dish_foodamount	Number of values allowed → unlimited
7	file	Recipes	field_dish_recipe	File type:txt, pdf, doc, docx file directory is "myimage"

(5)-2 Enter "Individual Data" for cooking menus & recipe notes

Just in case, try entering individual data by hand input. If bulk input in the file is over, erase it later.

If you select →+ Content in the →, select Cooking to the Create Reason screen.


In the ID field, enter "test", "dish name", "susi", "cooking image", appropriate jpeg file, "unit price (yen)" and "blurb" appropriately.

In the Ingredients field, enter one ingredient ID in multiple boxes. If you don't have enough boxes, add one with Add Another Item. The pop-up menu displays information that can be entered 青 input in a small number of characters, so you can choose from it.

For "Recipes", try selecting the appropriate txt, pdf, doc, docx file. Make sure that the content is properly done in Save. In order to enter data in bulk using an external file, you need to add a module to Drupal. It is a module for uploading image files, etc. "IMCE" and a module for CSV file injection "CSV importer".

16

(5)-2-1 Introduction of "IMCE" and "CSV importer" modules

If you add a new tab in your browser and search for the phrase "Drupal IMCE module download", you will see "IMCE" in | Drupal.org think there is a site called "I'm not going to do it". Click it and scroll all the way down the screen, followed by  in the 黄 green square in the Downloads column, followed by a 青 color 青 tar.gz (144.49KB)." Right-click this to copy the address of the link (or copy the link). Return to drupal's administration screen, click "→" on the Site Administration toolbar, and then click Install New Modules.

Paste the address above in the "Install from next URL" column on the "Install new module" screen, and click [Install].

✓ Installation completed successfully. When you see "Next steps" section, [click Enable newly added modules](#). To return to the "Extend" screen, check the ☐ Imce File Manager added to the bottom, and click "Install" at the bottom.

Similarly, on a separate tab in your browser, search for the phrase Drupal CSV importer module download, and then click CSV | Drupal.org. In the Downloads

column, see 8.x-1.11 ... Right-click 黄 "tar.gz (24.69KB青)" in the green square to copy link address (or copy link). Go back to drupal's administration screen and install and activate this module.

When the installation is complete, click "Environment" on the site management toolbar. "Imce File Manager" has been added in the "Media" column, and "CSV importer" has been added in the "Development" column.

(5)-2-2 Upload files with "IMCE"

You can also enter image files individually when you add content, but if you have a lot of content, you can support it by loading multiple image files into the server at once and linking the content with an identifier.

First, prepare the necessary image files and store them in a folder somewhere. Next, "Put out the default body field of any content type and click the "chain" icon (link function) in the edit menu bar" A small screen called "Add Link" opens, so click [Open File Browser](#). The left column of the public:// displays the folder configuration from the bottom to the bottom. If you don't see public:// click "Click" You should see several folders with "myimage" init.

Click the myimage folder to open it, and then select the image file you want → a0> Add + File → because it displays the folder file configuration on your PC. → file is uploaded in the file open (or select for upload).

17

Do the same for the Recipe pdf file, select the file you want, and then click Open (or Select for Upload).

Close the small screen of File Manager (or File Browser) and close Add Link.

(5)-2-3 Check the ID of the imported file or entity reference field value First, it is the file ID (identifier). On the Administration menu → Site Build, under Views, edit a view named File. On the view editing screen, click "File: File ID (FID)" at the beginning of the "Field" column, and uncheck "Excluded display", the FID (ID) will be displayed to the left of the filename uploaded in the "Preview" column. Include this value as the ID of the file in the CSV data.

Then, if you also want to include entity reference fields in CSV data, do the same, and in the administrative menu site building → View, edit the view named Content. [Add] in the "Fields" column of the view editing screen. Check the ☐ to the left of the line title ID, category → And Set fields →

In Apply, a field called Content: ID (ID) is added. If you look at the preview, there is a column called "ID", and the numbers are arranged at the bottom. This is the internal identity of the context system. Include this value as data in the entity reference field in the CSV data.

Please note that the order of IDs and files and data may not be available.

(5)-2-4 Create a csv data file

Create a data file with the appropriate text editor or Microsoft Excel.

CSV stands for Comma Separated Value and is a Separated Value in Comm(a comma).

Csv data should be imaged as a table.

The first line is "header", which side-by-side with the item name separated by a comma. In Drupal, this item name must match the system internal name of the field when the content type was created. There is also information to add. Order of fields is free. It seems that you may make up the half-width space after the comma to make it look better. For example: The first two items(title and langcode)are required. title, langcode, body, field_dish_name, field_dish_image|target_id, field_dish_image|alt, field_dish_unitprice, field_dish_salestalk, field_dish_foodstuff, field_dish_foodstuff, field_dish_foodstuff, field_dish_foodamount, field_dish_foodamount, field_dish_foodamount, field_dish_recipe (LF)

Here

Title is the cooking ID.

"langcode" is called "language code", and in Japanese, set "ja". "body" is the "body" in the content. For the content type "Cooking", the body is

18

Because it is empty (kara), please do not put anything.

field_dish_name is the name of the dish.

field_dish_image|target_id contains the ID (FID) of the image file.

field_dish_image|alt contains the appropriate English alternate name for the image file. "field_dish_unitprice" is the unit price of the dish. Half-width integer.

field_dish_salestalk is a blurb. No more than 256 characters.

field_dish_foodstuff and field_dish_foodamount are arranged three at a time however, when creating content types, multiple (unlimited) data was included. This time, I will be able to put up to three for the time being. If there are fewer than three, leave them blank (required as fields).

Since field_dish_foodstuff is an entity reference field, it is not a "food ID", but "system internal ID of the content" examined in the previous section.

Note that foodstuff is listed in the order of youth, and foodcount is in the order corresponding to foodstuff. field_dish_recipe contains the ID (FID) of the file you examined earlier. LF is a line feed. Microsoft Excel files into CSV files. Depending on the conversion or editor settings, line breaks may be CR+LF, but in CSV files, line breaks are only LF. Please be careful. Along with the header, we're entering the data as shown in Table 3.

[Table 3: Cooking Menus & Recipes] CSV File]

title, langcode, body, field_dish_name, field_dish_image|target_id,
field_dish_image|alt,
field_dish_unitprice, field_dish_salestalk, field_dish_foodstuff, field_dish_foodstuff,
field_dish_foodstuff, field_dish_foodamount, field_dish_foodamount,
field_dish_foodamount, field_dish_recipe (header so far. Line breaks are LF only,

Save this .csv file name "menurecipe".

Use the CSV importer module to import menu recipe .csv in Drupal.

19

Displays other information imported into the content type dish. If you click "Add" in

the "Field" section earlier, a small screen of "Add Field" will appear.

If you type dish in search, only the line related to dish is displayed. On the left side ☐, check the box, and then click Add Field and Set.

The settings screen for each field appears, but do almost nothing and click Apply and Continue. However, in the "Cooking Image", the "Image Style" column is "Thumbnail (100×100)". The last button is the Apply button. → Return to the edit screen.

If you look at the preview, there is nothing in the "BODY" column, so I'll delete it. In addition, the terms FIELD_DISH_FOODSTUFF and FIELD_DISH_FOODAMOUNT: Delta have insanity, so let's delete these fields as well.

In the "Fields" section of the [edit screen](#), click [Content:Body](#), [Content:Ingredients: Delta](#) ([food material \(field_dish_foodstuff: delta\)](#)), [[Content](#)] [Food](#) : [Delta](#) ([Amount of food](#))

20

([field_dish_foodmount: Delta](#))) for each of them, and then click Delete [at the bottom](#).

If you look at the preview, it is displayed as "one line of one content", but the order of items is messy, so let's order by ID, dish name, cooking image, unit price, blurb, food, ingredient quantity, and cooking method. Sort under ▼ in the field term.

In the →page Back to site, make sure that you have a menu tag called "Cooking Notes" and that clicking on it will list "Cooking Notes". The menu tag "Cooking Notes" does not necessarily appear to the right of "Vendor Notes" and "Ingredient Notes". To re-order the menu tags, you can click Edit Menu operation in site building → Menu → Main navigation, and then drag the cross mark on the menu link column with →Save.

Once this is done, let's remove the content test (Figure 10).

[Figure 10: Cooking Notes screen]

料理ID	料理名	料理イメージ	単価円	説明文句	食材ID	食材量	調理法
001	餃子		200	美味一品	FO01, FO03, FO05	2, 4, 8	gyoza.pdf 30.20 円
002	ラーメン		500	美味二品	FO03, FO05, FO07	3, 4, 8	ramen_01.pdf 60.50 円
003	チャーハン		400	美味二品	FO07, FO08	7, 8	chahan.pdf 33.34 円
004	味噌豆腐		600	美味四品	FO02, FO03, FO06	2, 5, 8	miso_tofu_01.pdf 60.70 円
005	野菜炒め		300	美味五品	FO04, FO06, FO09	1, 4, 7	vegetable.pdf 24.75 円

(6) Create order notes

(6)-1 Create "content type" of order notes

Create the following content types:

Content type name: Order.

System internal name:order.

Title field label: Changed toOrder ID.

The following is a display of the fields that you want to add:

21

order	Field type	label	System internal name	Points to keep in mind
1	Plain text	Seat No.	field_orderer	
2	date	Order date and time	field_order_time	Default date selects Current Date and Time
3	content	Order dishes	field_order_dish	Check content type "Cooking"
4	Number (integer)	Number of orders	field_order_quantity	

(6)-2 Enter "individual data" in order notes

The "suppliers", "ingredients", and "dishes" created so far are fixed and rarely increase or decrease or change (in possible, the price of ingredients varies, but here it shall be stable for a while.) 。

However, "orders" occur from time to time and increase. Moreover, the input is an unfamiliar customer service (or more) may be in charge of the customer. So, create a screen that anyone can easily enter orders without making mistakes. Here, the figure 11 Think of a "Menu & Order" screen like this. Enter your seat number first and the menu of the dish.As you look, enter the number of dishes and the number of dishes to automatically add the name of the dish, the unit price, and the and update the total amount. Click the [Order button] to confirm the order.

[Figure] 11: "Menu & Order" screen outline]

ヘッダー

座席No.

品番	料理名	単価	数量	小計
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

料理メニュー


注文ボタン 合計

: 入力欄 : 自動表示

(6)-2-1 Introduction of module "webform"

First, add a module called "webform" that creates an input screen. Similar to the method of introducing "IMCE" and "CSV importer", open a new screen in the browser, search for the phrase "Drupal webform download" and find the suitable version of the webform.

Right-click "tar.gz" in the Download field to return to the Drupal extras → Drupal extras, paste the address into the → [Enhancements → + Install new module] → "Install from next URL" field, and then → Install → [Enable newly added modules] → return to the "Extend" screen. In the module name displayed by typing "webform" in the thinly displayed "Filter by name and description" field, check all items that can be checked in the left checkbox, and click [Install] at the bottom. → after a few minutes, you will be returned to the "Expand" screen. In my case, I did not return easily, so when I tried to press the [Install] button again or unchecked items other than "webform", I suddenly returned to the "Extension" screen and checked all the webform related items that I had checked earlier.

I want to see the unchecked Webform Custom Form Example in the item, so click  right side. Since webform Devel (disabled) and devel(missing) are displayed, you can install additional devel and enable Webform Devel.

Find, install, enable, and install the module "devel" as you did with the previous module addition method. Then enable Webform Devel and install it.

Then check Webform Custom Form Example and make it effective in Install.

Please note that if you do not put ✓ in the checkbox one by one and click the "Install" button at the bottom, it will not be "enabled".

(6)-2-2 Introduction of module "Token"

Add a new module called Token.

I think of this as a tool like "Omazina", which is used to access data a little bit with a small hand, y/sheds, which is different from Drupal's legitimate data access method. Look for and deploy as in previous modules.

(6)-2-3 Creation of "Order Entry" screen

First, create the "Order Entry" screen on the right side of the "Menu & Order" screen. Build Site → [Webforms] → [+ Add webform] .

In the "Title" field, "Order entry", the internal name of the system is "dish_order", → [Save]. On the Order Entry screen, click + Add element.

First, set No. Create an input field. Type is add element by selecting "textfield" to allow arbitrary strings. The title is "Seat No. The corresponding "key" is "sheetno" → Save.

Next, set the set of "part number, dish name, unit price, quantity, and small scale" one line at a time. To create a set, use the type flexbox.

[+ Add element] → The Add element key for flexbox layout → is
numbered flexbox 1 (numbered "1") → Save.

Return to the "Order Entry" screen and add flexbox 1 to the title. To the right of the line is a button called +Add element. Click this to add the item as an element in the flexbox 1 set.

Click this +Add element to create a cooking ID input field first. The type is Entity select, the title is Dish ID, and the key is dishid1 (number 1). The same is below. The content type is ☐ → "Save + Add element" with "Title display" in "FORM DISPLAY" as "→" and "→" in "FORM". Immediately, the next element type input.

Write the following token for the type "Dish Name", title "Dish Name", key "dishname1", and Computed value/markup.

[webform_submission:values:dishid1:entity:field_dish_name] This means that "the title value of dishid1 originating on the webform, access the specified type ("dish"), and display it by quoting the data of field_dish_name (dish name) in it. It is like.

In ☐, check "Automatic update the computed value using Ajax" and set title display to "invisible" [Save + Add elements].

Similarly, write the following token for the type Computed token, the title "Unit price (yen)," "dishprice1", and computed value/markup.

[webform_submission:values:dishid1:entity:field_dish_unitprice] Also, check ☐ Automatic update the computed value using Ajax, set title display to invisible, and save + Add element. Quantity is the type Number, Title Quantity, key dishquantity1, and Title display is → Invisible. Save + Add element.

"Small scale" is the type "Computed Twig", the title "Small meter", and the key "dishsum1". Write to Computed value/markup for the following spell (this was taught by a master, so keep in mind: "Spell" because it is a little long compared to token.) in. { % if data.dishprice1|length and

```
data.dishquantity1|length %} { data.dishprice1 *  
data.dishquantity1 } { % else % } { % endif % }
```

Also, check ☐ Automatic update the computed value using Ajax → Title display is → [Save].

First of all, set this in the "Build" screen of "Order input", and try to check the browse screen in [Save elements] and [Browse].

The image above in Figure 11 is displayed, and the box field (seat No., part number, quantity) and the name of the dish, unit price, and small meter should be displayed automatically as shown in the figure below.

In the same way, create a style in which you can order up to five items.

Change the previous number "1" to "2", "3", "4", "5" in order and flexbox2 ... Try making 5.

Finally, set the Total element in +Add element at the top of the Build screen in OrderEntry, not in flexbox5.

The types are Computed Twig, Total title, and dish total key. The Twig to write to Computed value/markup is as follows:

```
{% if data.dishquantity1|length %} {{ data.dishprice1 * data.dishquantity1 +
data.dishprice2 * data.dishquantity2 + data.dishprice3 * data.dishquantity3 +
data.dishprice4 * data.dishquantity4 + data.dishprice5 *
data.dishquantity5 }} {% else %} {% endif %}
```

"data.dishprice1 * data.dishquantity1" looks good in "data.dishsum1", but it does not display well.

Note that [Submit button] is easy to understand if you change the name of "Send button label" in [Edit] (or [Customization]) to "Order button". Changes are not reflected on the Build screen, but they are properly "order button" on the view screen. If you look at the screen in [View], it looks like Figure 12.

In this case, already seat No. and enter two dishes and their number.

In Build, click Save elements at the bottom to confirm the configuration.

店舗No.	メニューID	メニュー名	単価	数量	小計
123	002	ラーメン	600	3	1800
	005	野菜炒め	300	4	1200
	- なし -				
	- なし -				
	- なし -				
合計					2700

(6)-2-4 Create "Menu & Order" screen

The "Menu & Order" screen first creates a view "(Cooking) Menu" screen, and then block" (automatically placed) in the center of the screen, and then "order entry" On the right side of the screen (seeSecondary block) is to make it.

(6)-2-4-1 Create "(Cooking) Menu" view

Toolbar SiteBuilding→View→Add + View.

"Name of view" and "Menu & Order", system internal name "menu_order" in the view In the settings, the type specification is "Cooking".

Check ☐ Create a page, →Page title,and leave pathmenu_orderpage.

In "Page display settings", change "Display format" to "Table". The default for "Number of displays" is "Number of displays"10] and then erase the☐ Use pager" check and check "Create menu link". The "Menu" column that came out 「Main navigationAnd leave the "Link Text" column as it is. →Save and Edit → It will be the view "Menu & Order" edit screen.

Edit and add fields.

Change the display of "Title" from "Title" to "CookingID".

Add "Dish Name", "Cooking Image", "Unit Price (Yen)" and "Blurb" in Figure 10. For instructions,see (5)-3 View cooking menus and recipe notes. The item names seem to be menus, and "product name", blank, "yen (tax included)", "recommended", respectively

I will. In addition, "Sorting criteria" is added by "Sort titles in ascending order", click "Post date (descending order)" and [Delete].

[Save] and [Return to site] and click on the menu [Menu & Order]. I'm not here. illustration 13 If it is like, it is ok.

[Figure] 13: "Menu & Order" screen only for "Menu"]



The screenshot shows a web application header with a logo '中 丸丸食堂' and navigation links: 'ホーム', '仕入れノート', '食料ノート', '料理ノート', 'メニュー&注文' (highlighted), and 'アカウント情報'. Below the header is the title 'メニュー&注文'. The main content is a table with 5 columns: '料理ID', '品名', an image, '円 (税込)', and 'おすすめ'.

料理ID	品名		円 (税込)	おすすめ
d01	ギョーザ		200	美味一番
d02	ラーメン		500	美味二番
d03	チャーハン		400	美味三番
d04	麻婆豆腐		600	美味四番
d05	野菜炒め		300	美味五番

(6)-2-4-2 Create "Menu & Order" screen

Toolbar SiteBuilding→BlockLayout →Second → Place Block inWebform→Block Settings.

Change the"Webform"written inthe "Title" field to "Order Entry". The screen changes, but there is a "System internal name" column at the bottom, so change thisto"order_input", for example.

Inthe Webformfield, type"Order entry" as dish_orderinstead of the system internal name "Dish_order". (I don't use the internal name of the system, so my head is confused.) In the "Restrict browsing" field, type/menu_orderPage" in the page. →"Save Block"→ "Order Force"isin"Second" on the "Block Layout" screen.

[Back to site] and the "Order entry" screen on the right side of the menu [Menu & Order] screen But it is very cramped and does not display the input value.

So we use CSS to change the width of the left and right screens.

The Toolbar [→] [AssetInjector]→[CSSInjector] →[+ Add Css Injector]→ isthesettings screen of Add Css Injector.

If you entermenuordercss in the Labelfield, the system internal name will be the same. In the Code column, write the following CSS:

27

```
.col-sm-9 {
width: 650px;
}
.col-sm-3 {
width: 500px;
}
```

This means that the width of the screen(menu screen) of the class name col-sm-9 should be 650 pixels, and the width of the screen (order entry screen) of"col-sm 3" should be 500 pixels. It means. (Pixels The number is appropriate.)

Click the Page section of the Conditions column to CSS Specifies the path of the

screen to use . It is the path of the "Menu & Order" screen `"/menu_orderIt` is.

In the→,under Back to Site, make sure the screen looks like Figure 14.

[Figure] 14: "Menu & Order" screen]

料理ID	品名	円 (税込)	おすすめ
d01	ギョーザ	200	美味一番
d02	ラーメン	500	美味二番
d03	チャーハン	400	美味三番
d04	麻婆豆腐	600	美味四番
d05	野菜炒め	300	美味五番

注文入力			
座席No.			
124			
d01	ギョーザ	200	2 400
d02	チャーハン	400	4 1600
d05	野菜炒め	300	3 900
-なし			
-なし			
合計 2900			
注文ボタン			

For now, you can think of what you thought in Figure 11 as you're done. There are also challenges to make it look better, but study CSS separately to improve it.

(6)-3 Webform Content Creator, which transfers data fromwebform

"dish_order" to the content type"order", is a model that transfers webform data to content.

Order Entry allows up to five inputs for a single order. Within a webform, these input data are "lumps", making it difficult to work finely with individual dishes. So, when we move this data into content, we break it apart and put it in "information about hitachi's order dishes is one content".

For example, the first dish of webform "dish_order" is to the content type"order", the same dish is to the next content type"order", the third dish is to the next content type"order",... , and move. Each content of the content type order is numbered with a unique ID number(nid), so that the transferred content can be treated separately for each dish.

28

Just remember the "very important issues" here. That means that even if a seater orders only one or four dishes, there will always be five order content. If you only order one dish, the remaining four content is unnecessary "sky (kara) content". Because empty content increases with each order, administrators must either delete them on their own or build the ability to delete them automatically at any time. This time, it is too much, so I will keep it as a "challenge" in the future.

(6)-3-1 Introduction of the module "Webform ContentCreator"


Deploy in the same way as previous modules.

Upon review, search for the word "Drupal webformcontent creator download" in the browser search, find the latest version of the module that fits, copy the link (address) of "tar.gz" and return to Drupal, and then return to the toolbar [Enhancements] → [+ New module]

Install → Paste from the following URL → Install from next URL → Install
 → Enable → Check ☐ Webform Content Creator, and then follow the steps in
 → (bottom row) Install.

(6)-3-2 Use of the module "Webform Content Creator"

This module is used differently than other modules.

Toolbar [Extensions] → If you click " " in the description of "Webform Content Creator" and change it to "▼" "▼" "the machine name, version, "... required, followed by Help, Rights Limit, and Configuration. Click this Configuration → webformContent Creator list. → the add webform content creator entity → the + Add configurations screen.

The Title field is, for example, dishorder2order1 (webform system name + 2 + content type system name + 1). "2" is "to". Content Creator has the same system internal name. The last "1" in the name is a number that indicates the number in one order entry. Create Content Creator from 1 to 5.

(Repeat from + Add configuration above to Save below.) Select "Order entry" in the "webform" field, "Content" column in "Content", and "Order" in the "Badr" field → "Save", and the screen returns to "Content Creator List". Click Manage Fields in the dishorder2order1 line to the Manage Fields screen.

- ☐ the Field_order_dish Order Dish (field_order_dish), click - Select - ▼ and select Dish ID (dishid1) - Entity select.

- ☐ number of orders (field_order_quantity), click - Select - ▼ and select Dish quantity1 - Number.

Check ☐ Seat No. field_orderer, click - Select - ▼ and select Seat No. (sheetno) - Text Field.

"☐ Order ID (title)" ([Note! If you check "ID(nid)" and click ☐ (left in the "Select" column) in the "Custom" column, the gray box on the right will turn white.

29

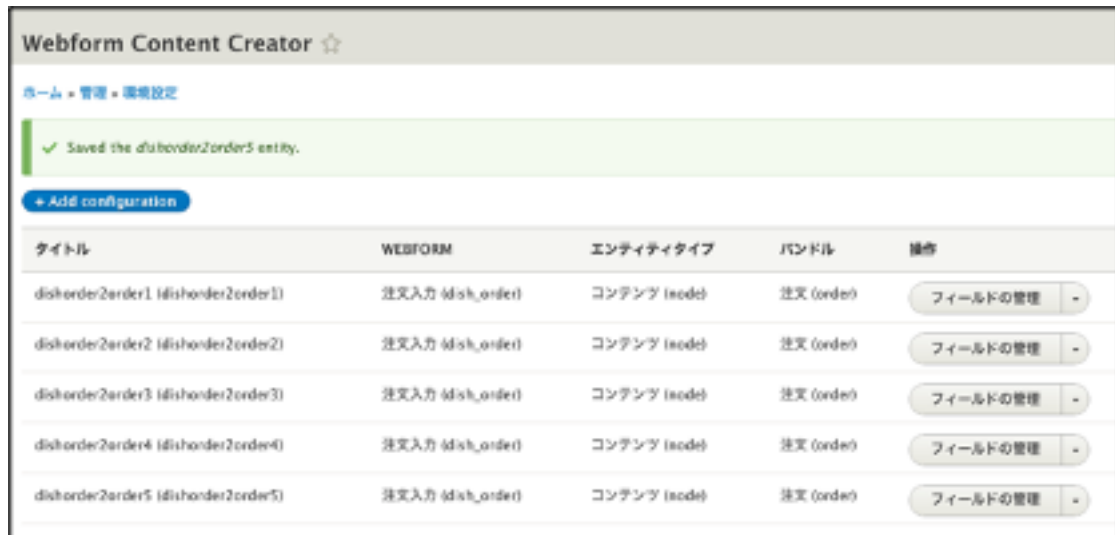
it is not a good place to go Enter webform_submission:sid1 here. In front of "1", add a half-width space. "sid" is the "Submission ID", which is a unique integer in the webform "Order Entry". "webform_submission:sid" means "display sid made by webform" in Token. This is the first of one order, so add (half-width space)+1. The second to fifth numbers each. [Please be careful about this!]

- ☐ the "User", click - Select - ▼ and select Submit by (uid) - entity_reference.

→ Save.

Finally, the Webform Content Creator list looks like Figure 15.

Figure 15: Webform Content Creator List



(6)-4 Display of "View" of order notes

Create a "view" of the order entry.

The operation is almost the same as for vendor notes.

The name of the view is order note, the systeminternal name is order_note, and in the view settings, the type specification is Order.

Check ☐ Create a page, →Page title is still, and Path order_notepage.



In "Page display settings", change "Display format" to "Table". Clear the default of "10" for "Number of displays", uncheck "Use ☐ pager" and check "Create menu link". Change the "Menu" column that came out to "Main navigation", and the "LinkText" column will be left as it is →[Save and Edit] → View "Order Notes" edit screen.

Edit and add fields.

Change the display of "Title" from "Title" to "OrderID".

30

The additional field "Order Date and Time" is set to "Time Zone Override" to "Tokyo", "Date format" to "Default short date", and "Apply" (or [Apply]). the same as below).

Sort "OrderID" and "Order Date and Time" and bring "Order Date and Time" first. The following field should be "Dish Name", but it may be first. "Dish name" has already been set as "field_dish_name", but this is not in the content type "Order", it is in the content (= "Dish") referenced from "Order Dish (field_order_dish)", so set "Relationship" to refer to it. Click the  section on  right of the edit screen to display the "Relationships" section. → Add → check ☐ field_order_dish → add relationships → Apply. Then, "Change: Content" will be entered in the "field_order_dish" section of the edit screen. Then select "Dish Name" in the add field. Set Relationship to field_order_dish: Content → Apply.

Additional fields "Orders" and "Seats No." does not use relationships, but is set up obediently.

Add Filter Criteria.

□ tofield_order_dish →→ The Operatorfieldis changed toNOT NULL →Apply .
Do□ same forfield_order_quantityorders (order count).

These filters allow the order note view to prevent orders from appearing if the order dish is blank or the number of orders is blank.

Add Filters for Search.

Specific order ID,dish name, seat No. to be searchable in .

For "Specific Order ID",check "Add" → "□ Title" in "Filter Conditions" → [Setup filter criteria]→ Check"□ Show this filter to visitors so that they can change it" → "Label"field selects "Order ID", "Operator" column selects "Normal table → Apply] .

"Dish name" ischecked in "Add" →"□ Cooking Name (field_dish_name)"in "FilterConditions" in "→ Field_dish_name"in "Set by adding filterconditions→"□ Display this filter to visitors and change it" → "Label"column is "Cooking name", "Operator" column is "Included", "Include", "Include", " The Relationshipfield isfield_order_dish: Content" and →Application.

"Seat No. Check "Add filter conditions"→"□ Seat No.field_orderer)" → in "Setby adding filterconditions"→ "□Display this filter to visitors and change it further" → "Label"column is "Seat No. 」 → [Apply] .

Sort Criteria, but set Content: Post Date and Time (descending) and Content: Ordered Dishes (ascending order).

This is the end of the setting of the view "Order Note".

The data entered in "Menu & Order" in advance is, for example, a figure in the preview 16 Like Make sure it becomes.

31

[Figure 16:Preview screen of view "Order Notes"]

タイトル

注文ノート

外部設置フィルター

注文ID

料理名

座席No.

適用

コンテンツ

注文日時	注文ID	料理名	注文数	座席NO.
2021/05/06 - 21:30	94 1	ギョーザ	3	95
2021/05/06 - 21:30	94 2	ラーメン	4	95
2021/05/06 - 19:05	90 1	ギョーザ	5	91
2021/05/06 - 19:05	90 2	チャーハン	2	91
2021/05/06 - 19:05	90 3	野菜炒め	3	91

(6)-4-1 "Sumi" (cooked or served) display in order notes

A little challenging greed came out.

Add a feature to show "sumi" for dishes that have been cooked (or served). Drupal

seems to be unable to edit the content once it is created outside of the edit screen. For example, the plain text field "Cooked? I was not nough to find a way to enter "Sumi" or "Sumi" at the end of cooking except in the extras. (There may be a module somewhere.) So I used the "comment" input.

In the Management menu[site construction] →[Comment type]→[+ Add comment type]→ in the "Sym? " " in the system internal name column,"sumiqu" in the target entity type column is "content"→[Save].

Comment type "Syss? Under ManageFields→ on the Comment label, click ▼ on the Edit▼ and select Delete → Delete.

Administration menu [Site construction]→[Content type]→ [Manage fields] →[+ Add field]→ In "Order", select "Comment" in the "Add new field" column, and "Cooked?"), the internal name of the system field_(sumi9) (9 is instead of qu). →[Save and Next] →"CommentType" column is "Sym? Select →Keep Field Settings→ Uncheck Thread Format, and set TheNumber of Comments Per Page to 1.

Uncheck "Display a reply form to a comment in the same page as the comment" →Save Settings.

You can now add a comment field.

In addition to the view, I will display it.

Administration menu [Site Construction]→ [View]→If "Order Notes" is "Disabled", click "Enabled" on the right edge of the line → Click "Edit" in "OrderNotes" →"Feel"

32

If there is a "broken or lost handler", delete it (click its wording to[Delete]) →Add Field→"Cooked? Check →[Add and set]→[Rewrite results]→ ☐ Check "Overwrite the output of this field with the specified string" → Enter "Sumi" in the "Text" field → [Apply]→ Save.

Back to Site →OrderNotes displays Order Notes, and on the right edge of the table, click Cooked? The column is now ready.

Click the applicable order ID to display "Sumi" here. → the information for that order ID, 青 in a letter that says Add Comment. Click → Click the green [✓ Save] button at the bottom of the "Add Comment" →

"Cooked?" " and "(no subject)", but do nothing, and click on the menu [Order Notes].

You should now see an order note, such as Figure 17.

(Order ID: 90 2 and 90 3 are "sumi".))

(Cooking) Sumi can now be displayed with mouse clicks only (4 pochi).

Figure 17: "Cooked? "Order note" with column added]

丸丸食堂

ホーム 注文先ノート 食材ノート 料理ノート メニュー&注文 注文ノート

アカウント情報 ログアウト

注文ノート

注文ID 料理名 座席No. 適用

注文日時	注文ID	料理名	注文数	座席No.	調理済?
2021/05/06 - 21:30	94 1	ギョーザ	3	95	
2021/05/06 - 21:30	94 2	ラーメン	4	95	
2021/05/06 - 19:05	99 1	ギョーザ	5	91	
2021/05/06 - 19:05	99 2	チャーハン	2	91	スミ
2021/05/06 - 19:05	99 3	野菜炒め	3	91	スミ

(7) Data processing for food purchase

From now on, when there is an order such as seat No.95 in Figure 17 (order ID: 94 1 and 94 2), "What ingredients are needed in total for these cooking and what vendor should I order them from?" Also, what is the cost? Consider the calculation process.

Please consider that, like "Order → Immediate Cooking", "Order → Ingredients Purchase → Cooking", so if the caterer receives a cooking reservation for a banquet a few days away.

Order ID: 94 1 is 3 gyoza with dish ID: d01, and order ID: 94 2 is 4 ramen with dish ID: d02.

The normal flow of thought may be as follows:

- (1) Look at the "Order Notes" to know that the order IDs are 94 1 and 94 2.
- (2) Order ID: 94 1 is cooking ID: d01 is 3 servings.

33

- (3) In addition to looking at the "Cooking Notes" and making the dish ID: d01 for one serving, ingredients f001 need 2, f003 is 4, and f005 is 6.
- (4) When they are in front of three people, ingredients f001 need $2 \times 3 = 6$, f003 is 12, and f005 is 18.
- (5) Similarly, order ID: 94 2 is cooking ID: d02 is 4 servings.
- (6) Cooking ID: To make d02 for one serving, ingredients f003 need 3, f005 is 6, and f007 is 9.
- (7) If they are in front of four people, food f003 needs 12, f005 is 24, and f007 is 36.
- (8) The total amount of ingredients from (4) and (7) is 6 for f001, $12 + 12 = 24$ for f003, 18 for f005, and 36 for f007.
- (9) Looking at the "Ingredients Note", each food cost = food unit price \times Because it is the total amount of ingredients, f001 is $36 \times 6 = 216$ yen, f003 is 1152 yen, f005 is 4608 yen, f007 is 2772 yen.
- (10) The supplier of ingredients f001 is "Green shop 7" with vendor ID: s01.
- (11) Look at the "Supplier Notes" and fax the following order form to "Greenman O7". "Sell 600g carrots (food ID: f001 ingredient name) because the unit quantity is 100g). The cost is 216 yen."
- (12) For other ingredients, (8) Create a purchase order by calculating the same calculation below and fax it.

In this way, the usual way is to refer to each note sequentially. But the drupal

way is a little different.

Drupal's View feature lists all the content and its reference information on a single line, so there's no way to take advantage of it.

In short, you can pull out all the necessary information in a view and modify it in languages such as JavaScript and CSS.

Name the view "Purchase Notes" with all the information you need. In data processing, "purchase notes" are displayed, then "purchase calculation" is performed by a script built in JavaScript of Asset Inquirer, and "purchase order issuance" is performed by JavaScript and CSS.

(7)-1 Create "Purchase Notes" in the view

Table 3 shows the required information for orders with seat No.95 in Per Ingredient ID for each order ID.

The cooking ID is entered because it is required to refer to "Cooking Notes", the Ingredient ID to refer to "Ingredient Notes", and the Vendor ID to Refer to "Vendor Notes". Here, the "quantity" is the amount of ingredients per meal (×100 g), and the "unit price" is the food cost per 100 g (yen).

[Table 3: Information required for purchasing ingredients for orders for seat No. 95 in Figure 17]

Order ID	Cooking ID	Number of orders	Ingredient ID	Ingredient name	quantity	unit price	Vendor ID	Vendor name	FAX
94 1	d01	3	f001	onion	2	36	s01	Greenman O7	03-3502-8111
94 1	d01	3	f003	carrot	4	48	s01	Greenman O7	03-3502-8111
94 1	d01	3	f005	Octopus Feet	6	256	s02	Octopus Yasshaku Fish Shop	04-4613-9000
94 2	d02	4	f003	carrot	3	48	s01	Greenman O7	03-3502-8111
94 2	d02	4	f005	Octopus Feet	6	256	s02	Octopus Yasshaku Fish Shop	04-4613-9000
94 2	d02	4	f007	Chicken Momo	9	77	s03	Yamanko Meat Shop	05-6789-0123

First, create a view that includes all the information you need.

The base is "Order Note", but it is remodeled with a different view name.

Click "▼" in the Administration menu [Site Construction] → [View] → "Edit ▼" in "Order Notes" and select "Duplicate" → "Name of view" is "Purchase Note". The system internal name is purchase_note and → [Duplicate].

In the view management screen, first change the "Title" from "Order Note"

to "Purchase Note", "Pas" from "/order_note" to "/purchase_note", and the "Menu" name from "Order Note" to "Purchase Note".

In the Saveview →, under Back to Site, see what happened.

If the menu "Purchase Notes" is to the left of "Order Notes", in [SiteConstruction] → [Menu] → [Edit Menu] in "Main navigation", change the order so that the menu "Purchase Notes" in "Main Navigation" comes to the right.

On the "Purchase Notes" screen, if you slowly take the cursor to the right edge of the line in the search field, the [Pen to circle] mark will appear like a ghost. When you click this, the word "Edit View" appears, and when you click it, it changes to the view edit screen. Modify the file to display the information in Table 3.

Field "Content: Cooked? To delete the .

Add a Cooking ID. This was titled "Order Dish" (label) for the content type "Order", but change the label here to "Cooking ID". Sort the field order and bring it before the "Dish Name".

"Seats" No. The label name is changed to "Reservation", and the line is "Order" ID bring before I will. The label name of "Filter Conditions" was also changed to "Reservation", and "Operator" was changed to "Regular Expression" And then [Apply] and bring the line before the title. For the time being, here [Keep] and then return to the site to open the Menu & Order screen.

"Seat No. 1" in "Order Entry" In the column, for example, "6/10 12:00 Imperial Hotel / Feng-no-Ying", order the dishes accordingly.

In addition, this time, I will order dishes at "Seat No." such as "6/12 14:00 Sabo Kaikan / Room 987". When you open Purchase Notes, you'll see a list similar to Figure 18.

[Figure 18: Case study of "purchase note" with reservation]

仕入れノート					
予約	注文ID	料理名	適用		
注文日時	予約	注文ID	料理ID	料理名	注文数
2021/05/09 - 21:58	6/12 14:00 砂の会館987号室	98 1	d02	ラーメン	16
2021/05/09 - 21:58	6/12 14:00 砂の会館987号室	98 2	d05	野菜炒め	16
2021/05/09 - 21:46	6/10 12:00 帝国ホテル 函の側	97 1	d01	ギョーザ	50
2021/05/09 - 21:46	6/10 12:00 帝国ホテル 函の側	97 2	d02	ラーメン	30
2021/05/09 - 21:46	6/10 12:00 帝国ホテル 函の側	97 3	d03	チャーハン	20
2021/05/09 - 21:46	6/10 12:00 帝国ホテル 函の側	97 4	d05	野菜炒め	40
2021/05/06 - 21:30	95	94 1	d01	ギョーザ	3
2021/05/06 - 21:30	95	94 2	d02	ラーメン	4
2021/05/06 - 19:05	91	90 1	d01	ギョーザ	5
2021/05/06 - 19:05	91	90 2	d03	チャーハン	2
2021/05/06 - 19:05	91	90 3	d05	野菜炒め	3

By the way, you want to stock ingredients for reservations on 6/10 and 6/12 together.


To display only those reservations, enter "6/1[0-2]" in the "Reservation" search field (filter) and [Apply]. This is the string "6/1" followed by one character from 0 to 2.

If you're representing a four-character string in between, and it matches it, it will show you what you want to see. In addition, if you make a reservation only for "Sabo Kaikan", you will be hooked with a partial string such as "sand".

Next, add the "Ingredient ID" and "Quantity" field, but before that, erase the contents of the order IDs 97~ (for imperial hotels) and 98~ (for Sabo Kaikan) to avoid clutter.


Check the ☐ to the left of the line with administrative menu [Content] → Title 97.. and 98.. and check →[delete]in Action is "Delete Content".

Return to the edit screen of the view "Purchase Notes" and add a field. Since "Ingredient ID (field_dish_foodstuff)" in Table 3 is in the content referenced from "Order Dish (field_order_dish)" (= "Dish"), relationship is "field_order_dish: Content".

In  Settings for Multiple Fields, uncheck ☒ Show all references in one line. (This is the ingredients) Appears on a separate line for each ID → App. Just in case, add Ingredient ID in ascending order to Sorting Criteria (relational field_order_dish: Content).

Return to adding fields, but since "Amount of ingredients (for one dish)" is also in the same content as "Ingredient ID", apply relationship as "field_order_dish: Content". In the view, it appears on the same line, so I'll modify it later in JavaScript to "Amount of ingredients per line".

Next, add the fields "Food name" and "100 g unit price (unit price per 100 g of ingredients (yen))".

"Food name" is in the content type "foodstuff", and it is referenced from the field "field_dish_foodstuff (food ID)" of the content type "dish", so add "relationship" first.  Altitude → Add relationships → Check content referenced by ☐ field_dish_foodstuff → Add relationship to set → Relationship field_order_dish: → Apply. The added relationship is (field_order_dish: content) field_dish_foodstuff: content.

To add a field for "Ingredient Name", check the "Add" field → "☐ Food name", → [Add field and set] → Relationship selects "field_dish_foodstuff: Content", and the label is "Ingredient Name" → [Apply].

"Per hundred g unit price" is also "food name", also select "field_dish_foodstuff: Content" in the relationship → then name [Apply].

Next, you'll add fields for Store ID, Vendor Name, and Fax.

36

Store ID is located in the content type supplier and is referenced from the field "foodstuff_supplier (store ID)" of the content type "foodstuff", so first put a check in "Content referenced from ☐ field_foodstuff_supplier" as "Relationship → Add and set relationships → Relationship is set field_dish_foodstuff: → to Apply. The added relationship is (field_dish_foodstuff: content) field_foodstuff_supplier: content. To add a field for store ID, check the Field field → ☐ Vendor → and set up → Relationship selects field_dish_foodstuff: Content, label is Store ID. → is Apply.

Vendor Name selects Vendor Name, → Add Field → Relationship is field_foodstuff_supplier: content → Apply.

Fax is similarly selected, → Add fields → relationship - ship field_foodstuff_supplier: → Apply.

This is the end of the information gathering, but to save display space, stop the time display for "order date and time" and only the date. In field editing, click Content:

Order Date and Time →Select Date WithoutHTML Year in Date FormatandApply.
Add Filter Criteria.

□ Food ID →Apply→ The relationshipis field_order_dish: Not Null, and the operator isnot →Applied. Increase the order of this to "Contests: Number of orders (not empty)".

So far, you can see a list of Figure 19. [Save] and return to the menu "Purchase Note" Try it.

This table serves as a materialfor creatinga note by searching by
"Reservation"and/or"Order ID"and/or "Cooking Name" to narrow down the required information.

[Figure 19: View of "Purchase Notes" (partial)]

仕入れノート												
予約		注文ID		料理名		適用						
注文日時	予約	注文ID	料理ID	料理名	注文数	食材ID	食材名	食材量	料理価	店ID	仕入先名	FAX
05-06	95	94 1	d01	ギョーザ	3	f001	豆腐	2.4.6	36	s01	八百屋お七	03-3502-8112
05-06	96	94 1	d01	ギョーザ	3	f003	人参	2.4.6	48	s01	八百屋お七	03-3502-8112
05-06	96	94 1	d01	ギョーザ	3	f005	タコ足	2.4.6	256	s02	タコ八鮮魚店	04-4613-9001
05-06	96	94 2	d02	ラーメン	4	f003	人参	3.6.9	48	s01	八百屋お七	03-3502-8112
05-06	96	94 2	d02	ラーメン	4	f005	タコ足	3.6.9	256	s02	タコ八鮮魚店	04-4613-9001
05-06	96	94 2	d02	ラーメン	4	f007	鶏モモ	3.6.9	77	s03	白ネコ鶏肉店	05-6789-0124
05-06	91	90 1	d01	ギョーザ	5	f001	豆腐	2.4.6	36	s01	八百屋お七	03-3502-8112
05-06	91	90 1	d01	ギョーザ	5	f003	人参	2.4.6	48	s01	八百屋お七	03-3502-8112

(7)-2 Remodeling the view "Purchase Notes" with JavaScript

All the information necessary for ordering ingredients was pulled out in bulk in the view.
NowI'm going to modifyit inJavaScript. Learn javascript accordingly.

37

But how do I touch the "Purchase Notes" view? Open the "Source Code" on the view display screen and examine it.

(7)-2-1 Before remodeling with JavaScript, examine the "Source Code (HTML)" of the view Display the "Purchase Notes"from the main menu, and in the browser menu[Develop→[View The Source of The Page],or [View] →[View] → [View Source] (or [Verify Source]), you will see the browser backstage (source code). If you feel sick,click [X] and it willdisappear, so rest assured.

If you are patient andlookaround line 160to200, you will see familiar words such as "order date and time" and "reservation" (Figure 20). If your browser is safari,🔍 search for your browser by clicking on it as soon as it's available.

Inthis, a seriesof strings that start with "<th"and end with the corresponding "</th>"is to be called "elements". <thcan be<abcor <efg. The corresponding"</abc>"and"</efg>"are oneelement. "Corresponding" means <abc...

<abc...
 <abc... ... </abc> ...
 </abc>
 ... </abc>
 It means.

Response
 Correspondence
 Correspondence

[Figure 20: Part of the source code of "Purchase Notes" (near "Order Date and Time")]

```

135 <div class="view-content">
136 <div class="table-responsive">
137 <table class="table table-hover table-striped">
138 <thead>
139 <tr>
140 <th id="view-field-order-time-table-column" class="views-field views-field-field-order-time" scope="col">注文日時</th>
141 <th id="view-field-orderer-table-column" class="views-field views-field-field-orderer" scope="col">予約</th>
142 <th id="view-field-title-table-column" class="views-field views-field-title" scope="col">注文ID</th>
143 <th id="view-field-order-dish-table-column" class="views-field views-field-field-order-dish" scope="col">料理ID</th>
144 <th id="view-field-dish-name-table-column" class="views-field views-field-field-dish-name" scope="col">料理名</th>
145 <th id="view-field-order-quantity-table-column" class="views-field views-field-field-order-quantity" scope="col">注文数</th>
146 <th id="view-field-dish-foodstuff-table-column" class="views-field views-field-field-dish-foodstuff" scope="col">食材ID</th>
147 <th id="view-field-foodstuff-name-table-column" class="views-field views-field-field-foodstuff-name" scope="col">食材名</th>
148 <th id="view-field-dish-foodamount-table-column" class="views-field views-field-field-dish-foodamount" scope="col">食料量</th>
149 <th id="view-field-foodstuff-unitprice-table-column" class="views-field views-field-field-foodstuff-unitprice" scope="col">材料価</th>
150 <th id="view-field-foodstuff-supplier-table-column" class="views-field views-field-field-foodstuff-supplier" scope="col">店ID</th>
151 <th id="view-field-supplier-name-table-column" class="views-field views-field-field-supplier-name" scope="col">仕入業者</th>
152 <th id="view-field-supplier-fax-table-column" class="views-field views-field-field-supplier-fax" scope="col">FAX</th>
153 </tr>
154 </thead>

```

In Figure 20, see class="..."in... For example,"views-fieldfield-order-time"for "order date and time" or "views-field-field-orderer"for "reservation" is a name specific to the "class" to which each news report belongs. This will locate you on the HTML to do what you've modified in JavaScript and CSS. Note thatviews-field is not uniqueand canbe ignored.

module. Asset Collector is located in the Settings section of the management menu.
 → AssetCollector→JSCollector→+AddJs →The Label field is "customersnotejs".
 (The system internal name remains the same.))

After programming JavaScript in the "Code" column, select [Page] for "Conditions"→, add "/"to the system internal name of the view "Purchase Notes" in the "Page" column, write "/purchase_note", and →[Save].

You may put"abc"or an appropriate character in the"Code" column for the time being and [Save], and later write correctly in the code column in "Edit".

(7)-2-3 Get view data in JavaScript

Here's what to write in the Code column of EditJsCollector:

【Please note】 If you cope the following script part into the Js Collector code field, the quote mark (also known as single, double) may cause Syntax Error due to font reasons, so rewrite all each quote mark in the code field after the copipe. It's ok just to "rewrite". The symmetrical half-width quote mark works correctly.

In JavaScript, you can get all the elements belonging to that class with the following command that specifies the class name in theHTML sentence.

```
document.getElementsByClassName(' class name ')
```

This information enters an "array" in the JavaScript grammar as a string.

For example, write a script like this: "const"means "constant". const orderTime =

```
document.getElementsByClassName('views-field-field-order-time') ThisallowsorderTime[0] to
```

```
<th id="view-field-order-time-table-column" class="views-field views-field-field-order-time" scope="col"> Order date and time </th>, butorderTime[1] includes:
```

```
<td headers="view-field-order-time-table-column" class="views-field views-field-field-order-time"><time datetime="2021- 05-06T12:30:39Z">05-06</time></td>
```

The string "Is entered". (黄 color marker is the class name.) 青 color is the information you really want. How many elements are in is obtained from the number of elements in orderTime in the following description. const orderLength = orderTime.length

To bring all the elements of the purchase notes list to JavaScript, write: (At the end of the line, click // ... " is "in-line comment".)

Extract information from the view "Purchase Notes"

```
const orderTime = document.getElementsByClassName('views-field-field-order-time'); // Order date and time const
orderer = document.getElementsByClassName('views-field-field-orderer'); // Reserved const orderID =
document.getElementsByClassName('views-field-title'); // Order ID
const dishID = document.getElementsByClassName('views-field-field-order-dish'); // Cooking ID const dishName =
document.getElementsByClassName('views-field-field-dish-name'); // Dish name const dishQuant =
document.getElementsByClassName('views-field-field-order-quantity');// Orders
```

39

```
const foodstuffID = document.getElementsByClassName('views-field-field-dish-foodstuff'); // Ingredients ID const
foodstuffName = document.getElementsByClassName('views-field-field-foodstuff-name'); // Ingredient name const foodamount
= document.getElementsByClassName('views-field-dish-foodamount');// Quantity of ingredients const foodstuffPrice =
document.getElementsByClassName('views-field-field-foodstuff-unitprice');// Material Unit Price const supplierID =
```

```
document.getElementsByClassName('views-field-field-foodstuff-supplier'); // Store ID const supplierName =
document.getElementsByClassName('views-field-field-supplier-name'); // Vendor Name const FAX = doc
ument.getElementsByClassName('views-field-field-supplier-fax'); FAX const orderLength = orderTime.length; // The number
of "order dates and times" is the number of elements
```

Then, from each element, only the item name such as "Order date and time" and data (value) such as "05-06" are taken and put into a separate array variable.

To cut only the data out of the element, you just need to add the following underlying lines to the variable containing the element:

Variable `textContent.trim()`

For example, `orderTime[1]`. Write `textContent.trim()` and you get "05-06". First, declare an array variable that contains only data. As a variable name, for example, the name of the variable containing the element is all named with the addition of "A" of "array".

The repeating syntax then throws the data into the array variable.

The above is "... `GetElementsByClass` ... After writing, add: "const" is a constant that cannot be changed once it is in, but "let" is not a "constant" but a declaration of a vessel with a value that changes.

Declare an array variable that contains only the data to be cut out of the **element** (A means array)

```
const orderTimeA = []; // Order date and time
const ordererA = []; // Reservations
const orderIDA = []; // Order ID
const dishIDA = []; // Cooking ID
const dishNameA = []; // Dish Name
const dishQuantA = []; // Orders
const foodstuffIDA = []; // Ingredient ID
const foodstuffNameA = []; // Ingredient Name
const foodamountA = []; // Amount of ingredients
const foodstuffPriceA = []; // Cost per material
const supplierIDA = []; // Store ID
const supplierNameA = []; // Vendor name
const FAXA = []; FAX
```

Take data from an element and put it in a new array variable

```
for(let index = 0; index < orderLength; index++) { // Repeat in {} from number 0 to number -1=
orderTime[index].textContent.trim(); // Order date and time orderA[index] = order[index].textContent.trim(); //
Reserve orderIDA[index] = orderID[index].textContent.trim(); // Order ID dishIDA[index] = dishID[index].textContent.trim();
// Dish ID dishNameA[index] = dishName[index].textContent.trim(); // DishQuantA[index] = dishQuant[index].
textContent.trim(); // Orders foodstuffIDA[index] = foodstuffID[index].textContent.trim(); // Ingredients ID
foodstuffNameA[index] = foodstuffName[index].textContent.trim(); // Ingredient name foodamountA[index] =
foodamount[index].textContent.trim(); // FoodstuffPriceA[index] = foodstuffPrice[index].textContent.trim(); // Material
unit price supplierIDA[index] = supplierID[index].textContent.trim(); // Store ID supplierNameA[index] =
supplierName[index].textContent.trim(); // Vendor Name FAXA[index] = FAX[index]. end of textContent.trim(); // fax } // {}
```

(7)-2-4 Display the "amount of ingredients" in the view "Purchase Notes" correctly
In Figure 19, the Food Quantity column displays multiple comma-separated data.
This is because it is a field with multiple values.

The array variable `foodamountA[index]` also contains multiple data in a single variable. For example, the values of the first six variables are:

```
foodamountA[0] = amount of ingredients
foodamountA[1] = 2,4,6
foodamountA[2] = 2,4,6
foodamountA[3] = 2,4,6
foodamountA[4] = 3,6,9
foodamountA[5] = 3,6,9
::
```

These

```
foodamountA[0] = amount of ingredients
foodamountA[1] = 2
foodamountA[2] = 4
foodamountA[3] = 6
foodamountA[4] = 3
foodamountA[5] = 6
::
```

At the same time as you piece the data as shown, you also want the display to be one at a time.

```
let kokoValue = foodamountA[1].split(",");
```

Then, the data in `foodamountA[1]` is separated by ",", and "valued array variables" `ofkokoValue[0]=2`, `kokoValue[1]=4`, `kokoValue[2]=6` are automatically generated. `KokoValue` has a zero start and three of them from 0 to 2. Note that `kokoValue.length` tells you how many arrays occur.

Also, see. As a very thankful feature of `textContent`, if you set `let foodamount[i].textContent = data`, the data will be reflected (displayed on the screen) in the source code (HTML) from `foodamount [i]`! (The following is an example.) Figure 20)

```
<td headers="view-field-dish-foodamount-table-column" class="views-field views-field-field-dish-foodamount"> data</td>
```

After all, the script for "displaying one amount of ingredients" is added as follows.

Display comma-separated ingredients one by one

```
for(let index = 1; index < orderLength; index++) { // From number 1 to number of elements -1 { ... Repeat } let
  kokoValue = foodamountA[index].split(","); // Cut out comma-delimited values and put them in
  array variables for(let ko = 0; ko < kokoValue.length; ko++) { // Repeat in the cut number order
    foodamountA[index] = kokoValue[ko]; // foodamountA[.. ] with only one value foodamount[index].
    textContent = kokoValue[ko]; // Add index by 1 because you have finished one line of the view with
    only one value in the corresponding source code index++; //
  } // Repeat inner for syntax index--; // Reduce index value by 1 too much } // Repeat outer for syntax
```

Once you've entered the code so far, let's [Save] Js Collector, go back to the site, and look at the "Purchase Not" screen. If it were shown in Figure 21, it would be a success.

仕入れノート												
予約		注文ID		料理名		適用						
注文日時	予約	注文ID	料理ID	料理名	注文数	食材ID	食材名	食材量	材料価	店ID	仕入先名	FAX
06-06	95	94 1	d01	ギョーザ	8	f001	玉葱	2	36	s01	八百屋お七	03-3532-8112
05-06	95	94 1	d01	ギョーザ	5	f003	人参	4	48	s01	八百屋お七	03-3532-8112
05-06	95	94 1	d01	ギョーザ	3	f005	タコ足	6	250	s02	タコ八野鳥店	04-4813-9001
06-06	95	94 2	d02	ラーメン	4	f003	人参	3	48	s01	八百屋お七	03-3532-8112
05-06	95	94 2	d02	ラーメン	4	f005	タコ足	6	250	s02	タコ八野鳥店	04-4813-9001
05-06	95	94 2	d02	ラーメン	4	f007	鶏モモ	9	77	s03	山子コ博肉店	05-6789-0124
05-06	91	90 1	d01	ギョーザ	5	f001	玉葱	2	36	s01	八百屋お七	03-3532-8112
05-06	91	90 1	d01	ギョーザ	5	f003	人参	4	48	s01	八百屋お七	03-3532-8112

(7)-2-5 What to do if JavaScript doesn't work

The person who can move JavaScript in one shot is "Talent Ant". I was a talent pear. If Figure 21above doesn't work, there are two possible things. One is when the process is terminated and "stopped". The other is that there is no change in the screen.

However, if the process does not end and something is infinitely "moving" (the crooked arrow in the center of the browser is spinning around), or the browser screen disappears, and only one line appears at the top, such as "It's under repair..."

In the latter case, kill the browser first, launch drupal's management screen again, and then clear all caches → → Settings and Performance. Then under The Ring Boundary → Asset Collector → →, click Edit in the "pchasenotejs" section to check the code field.

For the former ("stopped"), select the browser menu [Development] (for safari) or → [Development / Management] (for chrome) and [View JavaScript Console]. If javascript is the cause, an error message appears in red in the console. Read it and deal with it.

「Uncaught TypeError:... If it comes out, it is a typo somewhere. For other methods of dealing with errors, please study "on the spot" separately on the net or something. If you deal with a single error, multiple error messages often disappear. In Google Chrome, unchecked runtime.lastError: The message port closed before a response was received. You may get an error message, but there is no problem without looking at it.

(7)-2-6 Aggregate the amount of ingredients by ingredients

In JavaScript code, orderLength was the number of line number +1 (item name) in the view table.

The aggregation method is to initially set the "aggregation column" to zero, process all data line in order from the first line (except for the item name (zero line) in the view list, and when it is finished, the value of the aggregation column is issued.

First, declare the array variable stuffSum (meaning Sum for foodstuff). let
stuffSum = {};

Since we do not know what ingredients come out, we use "associative arrays" that do not have to be a sequence number. If you declare an associative array, you must declare it with={}. Then, in the for syntax, repeat the following, from line 1 of the data in the list to line "orderLength - 1":

Read "StoreID" (=m) to determine whether it is first or not.

If it's your first appearance, create a new summary field for that store ID. It is such a condition.

StuffSum[m] = {}; // "m" store with new storage

read "IngredientID" (=s) to determine whether it is first or out.

If it is the first time, the aggregation column of the ingredients will be newly established in stuffSum[m]. The ingredients are decided by the shop, so it will be like this.

StuffSum[m][s] = {}; // "m" store ingredient "s" is newly created storage and zero is put in it as the initial value.

stuffSum[m][s] = 0;

If in parentheses of the statement "!" indicates "nega" and "if(! stuffSum[supID])" means "stuffSum [supID] does not exist" (= if it is the first time), and if it is, it will run in the {} that follows.

In the end, the script looks like this: (Kopipe a little later.))

Aggregate the amount of ingredients by ingredients

let stuffSum = {}; // Declaration of associative arrays

for (let index = 1; index < orderLength; index++) { // Repeat data line let supID =

supplierIDA[index]; // Assign vendor ID to supID let stuffID = foodstuffIDA[index]; // Ingredient ID To

stuffID if(! stuffSum [supID]) { // If the store ID is the first issue, stuffSum[supID] = {}; //

New aggregation field for that store ID } // Otherwise, the previous line will not be

performed if(! stuffSum [supID][stuffID] { // If the ingredient ID is first issued

stuffSum[supID][stuffID] = {}; // New stuffSum[supID][stuffID] = 0; Set the initial value of the summary

field for that ingredient ID } // Otherwise, the previous two lines will not be

performed stuffSum[supID][stuffID] += dishQuantA[index] * foodamountA[index];

} // Add the number of orders in the summary column in the previous line × The amount of ingredients.

You can now calculate the total amount of each ingredient for all the ingredients you use.

However, when ordering ingredients from the right store, it is necessary to display not only the total amount of ingredients, but also the store name, fax number, food name, food unit price, and price for each ingredient.

Make it easy to get this information out as well.

Store-related information should be named "supplier", and food-related information should be named "foodstuff".

```
let foodstuff = {};
```

These variables should be treated like stuffSum's "covanzame" and include information. These processes are included in the script block "Aggregate the amount of ingredients by 青" in the following characters.

Aggregate the amount of ingredients by ingredients

```
let stuffSum = {}; // Declaration of associative arrays for food quantity
aggregation Let supplier = {}; // Declaration of associative arrays for vendor
information let foodstuff = {}; // Declaration of associative arrays for food information for (let
index = 1; index < orderLength; index++) { // Repeat data line only let supID =
supplierIDA[index]; // Assign vendor ID to supID let stuffID = foodstuffIDA[index]; // If(!
stuffSum [supID]){ // If the store ID is first issued stuffSum[supID] = {}; // New
aggregation field for that store ID

supplier [supID] = {}; // New vendor information field supplier [supID] ['name'] =
supplierNameA[index]; // Assign vendor name

supplier [supID] ['Fax'] = faxA[index]; // Assign fax number

foodstuff [supID] = {}; // New Information field by Vendor } // Otherwise, the previous line will not be
carried out and if(! stuffSum [supID][stuffID]){ // If the ingredient ID is first issued
stuffSum[supID][stuffID] = {}; // New aggregation field for the foodID stuffSum [supID][ stuffID] =
0; // Set the initial value of the summary field for the food ID

foodstuff [supID] [stuffID] = {}; // Add information field for the ingredients foodstuff [supID] [ stuffID]
['name'] = foodstuffNameA[index]; // Add ingredient name to the ingredient information field
foodstuff[supID][stuffID] ['price'] = foodstuffPriceA[index]; // Substitute unit price in ingredient information field
} // Otherwise, the previous two linees would not be performed stuffSum[supID] [stuffID]
+= dishQuantA[index] * foodamountA[index];

} // Add the number of orders in the summary column in the previous line × The amount of ingredients.
```

Finally, to see the progress in the JavaScript console,

```
console.log(stuffSum);
console.log(supplier);
console.log(foodstuff);
```

I'll add.

Now you should save Js Collector and see something like Figure 22 in The Back to Site→Purchase Notes→ Browser Development→ShowJavaScript Console.

[Figure 22:JavaScript console display / amount of ingredients, store name and fax,food nameandunit price per supplier] 44



(7)-2-7 Writing from JavaScript to HTML

Now that you have the information, you can issue a purchase order to each vendor.

I'm sorry to say that it is low-tech, but we will pay attention to the ICT technology of the other party, print out the purchase order this time, and separately fax it by people.

Vendor Notes already displays a list (view) of the required information.

Follow the list and write "Purchase Order" to the screen.

Write to View in its follow-on range (see "<div ..." to the corresponding</div>)script called Write Afterwards.

To get the range of views, use the following statement:

```
const view = document.querySelector('.view-content'); ... (1) The period of  
'view-content'means'class'.
```

What you write is an HTML element. For example, set the class "myAddContent". Put the following elements in the variable:

```
let temp = '<div class = "myAddContent"></div>'; ..... (2) To put this  
element in an HTML sentence that displays the screen,
```

```
view.insertAdjacentHTML('afterend',temp); ..... (3)
```

And so on. This means to insert temp content immediately after the end of the view.

The source code of the display screen is 青 as shown in Figure 23.

[Figure 23: Sourcecode creation in the middle]

```
Screen setting description ...  
<div class = "view-content">  
A list of views, etc.  
</div> // End of "view-content"  
<div class = "myAddContent"></div>  
:
```

If ' is 'beforeend' instead of ' afterend', it will be inserted "just before the end (corresponding </div>)", and you will be able to write html elements one after the other in JavaScript.

For example, let's put the declaration of "purchase order issue" in the class "myAddContent". First, get the class name = " myAddContent" range.

```
const myAdd = document.querySelector('. myAddContent');
```

Write the text you want to display html and substitute it for temp.

```
temp = '<div id = "decration" >===== purchase order issue =====</div>';
```

The id setting is to decorate the character with CSS later.

If you write something like this, the temp content will be displayed on the screen.

```
myAdd.insertAdjacentHTML('beforeend',temp);
```

In summary, it looks like this:

```
From here, it is "purchase order creation"
const view = document.querySelector('.view-content');
let temp = '<div class="myAddContent"></div>';
view.insertAdjacentHTML('afterend',temp);
const myAdd = document.querySelector('. myAddContent');
temp = '<div id="decoration">===== purchase order issue =====</div>';
myAdd.insertAdjacentHTML('beforeend',temp);
```

After [Save], →[Back to site] and looking at the menu[Purchase Notes], you will see figure 24 in the lower left of the screen.

You can make modifications such as moving characters larger or center later in the CSS injector.

[Figure 24:Display "Purchase order issuance" in the lower left of "Purchase Notes"]



(7)-2-8 Calculation of the order "table" part to each supplier

Calculate the most important "table" part.

Since it is repeated foreach vendor, the for syntax is used, but since the necessary information is put in the associativearray, it is not possible to iterat while increasing index one by one. Use the method "for -in syntax". For example, in the following expression:

```
for( let A in B ) { ... }
```

Then, iteration is repeated for all A in associative array B (or "object" general).

Let's actually use it.

For all suppliers and all ingredients in it, the script that calculates the price with the food source by the total amount of ingredients multiplied by the unit price is as follows.

Create purchase order sheets

```
for (let sID in supplier) { // Repeat per vendor
  for ( let fID in stuffSum[sID]) { //
    Repeat by food by vendor
    Let fPrice = foodstuff [sID][fID]['price']; // Material unit
    price
    let fAmount = stuffSum[sID][fID]; // Total ingredients
    let subtotal = fAmount * fPrice; // Subtotal calculation
  }
}
```

To add the vendorname, fax number, and food name to this, 青 as follows: In addition, we calculated the "small amount" of the ingredients, but all the ingredients to the supplier were combined.

46

"We're going to calculate the total amount (just "total"). I'll show you that in the red. In addition, the console () is followed ingreen .log verify that these operations are correct.

Create purchase order sheets

```
for (let sID in supplier) { // Repeat per vendor
let supName = supplier[sID]['name']; // Vendor Name
let supFAX = supplier[sID]['FAX']; // FAX
let groundTotal = 0; // Total amount initial setup by vendor for ( let fID in
stuffSum[sID]) { // Repeat for each ingredient by vendor let fName =
foodstuff[sID][fID]['name']; // Ingredient name
let fPrice = foodstuff[sID][fID]['price']; // Cost per material
let fAmount = stuffSum[sID][fID]; // Total ingredients
let subtotal = fAmount * fPrice; // Subtotal calculation
console.log(supName, supFAX, fName, fPrice, fAmount, subtotal);
groundTotal += subtotal; // Add subtotal to total amount
}
console.log(supName, groundTotal);
}
```

If this works, go back to the site, open Purchase Notes (click the menu again if it's already open), and try displaying the JavaScript Console. If there is a display like Figure 25, it is successful.

[Figure 25: JavaScript console display / information required for ordering sheet

八百屋お七	"03-3502-8112"	"三喜"	"38"	16	578
八百屋お七	"03-3502-8112"	"入替"	"48"	04	2112
八百屋お七					2685
タコ八鮮肉店	"04-4613-9001"	"タコ足"	"256"	72	18432
タコ八鮮肉店	"04-4613-9001"	"イカダシ"	"128"	3	384
タコ八鮮肉店	"04-4613-9001"	"カンパチ"	"512"	12	6144
タコ八鮮肉店					24860
山本コ増肉店	"05-6788-0124"	"鶏丁"	"77"	50	3850
山本コ増肉店	"05-6788-0124"	"豚バラ"	"222"	16	3552
山本コ増肉店	"05-6788-0124"	"牛白肉"	"444"	21	9324
山本コ増肉店					16726

If you don't see these and you get an error message, consider checking javascript descriptions, fixing quotes, and so on.

If it works, go back to JS Collector and .log the console statement.

(8) Creation of purchase orders for purchasing ingredients

(8)-1 Display the information necessary for the purchase order on the "Anyway" screen

Now that you have the information you need to write a purchase order, you can display this information on the screen anyway. Once you're onscreen, qualify it with CSS to complete the Purchase Order. The general form image of the purchase order is to page down each vendor, write the "purchase order" in capital letters in the center

of the first line, the vendor name and fax number in the lower left, and the company name, address, and contact information on the right side with a little lower than that, put a table of the food name, unit price, quantity, and small sum under it, and finally display the total amount.

You already have all this information.

47

Use functions to make Js Collector's scripts easier to read. "Purchase order" will be written in function "sheetHeader", vendor name and fax will be written in function "supOnchu", company name and others will be written in function "myCompany", and "Table" will be written in function "tableStart", "tableBody", and "tableEnd".

Each function is inserted in the following deficit locations in the script Create Order Table: // Create order sheets

```
for(let sID in supplier) { // Repeat sheetHeader(); // [function] atama write on
purchase order let supName = supplier[sID]['name']; // Vendor name
let supFAX = supplier[sID]['FAX']; // FAX
supOnchu(supName, supFAX); // [Function] Supplier and fax number
myCompany(); // [Function] Company information
tableStart(); // [function] food order table header let groundTotal = 0; // Total
amount initial setup by vendor for ( let fID in stuffSum[sID]) { // Repeat by food
by vendor let fName = foodstuff[sID][fID]['name']; Ingredient name
let fPrice = foodstuff[sID][fID]['price']; // Cost per material
let fAmount = stuffSum[sID][fID]; // Total ingredients
let subtotal = fAmount * fPrice; // Subtotal calculation
tableBody(fName, fPrice, fAmount, subtotal); // [Function] The body of the food order table
groundTotal += subtotal; // Add subtotal to total amount }
tableEnd(groundTotal); // [Function] Food Order Table Footy }
```

In JavaScript, how to create a function, but if you say only what you need this time, it is as follows. For example, create a function with the name "anyterm".

Declaration ... function anyterm (argument){ what you want to do ...}

How to use ... anyterm (argument); Throw it into somewhere appropriate where the argument can be used. For example, in sheetHeader, since it is a function that just writes "purchase order", there is no "argument" (parentheses are required), and it looks like this. "+=" is an expression that indicates "seam".

```
function sheetHeader() {
temp = '<div class="eachSheet">'; Finally, don't forget </div> at the end!! temp += '<div
id="sheetName"> purchase order </div>'; Write "Purchase Order"
myAdd.insertAdjacentHTML('beforeend', temp); // write temp just before the end of myAdd }
```

</div>, which corresponds to "<div class="eachSheet", is added when the purchase order creation to one vendor is complete, so it is not attached here, but immediately before moving on to the next vendor process. supOnchu is the following script:

```
function supOnchu(shop, fax) { // Put vendor name/fax variable as argument temp = '<div
class="supplier">';
temp += '<div id="supName">' + shop + '&nbsp;&nbsp;  Inside <br></div>'; temp += '<div id
="supFAX">FAX:&nbsp;&nbsp; ' + fax + '<br></div></div>';
myAdd.insertAdjacentHTML('beforeend', temp);
}
```

Here, the string '<div id="supName">' and the shop and string ' '; Medium '</div>' is connected and added to temp. The same is for faxes. ' ' means "space" in one way. Html grammars ignore spaces without meaning, so you need these special expressions to display them in a browser.

48

MyCompany just writes the following complaints:

```
function myCompany() {
  temp = '<div class="myCompany">';
  temp += '<div id="myCompName"> My Company Inc. </div>'; temp += '<div id =
"myPcode">〒160-0021</div>';
  temp += '<div id="myAddress",1-> 1 - 1</div>'; temp += '<div id = "myTEL">TEL: 03-3204-
7166</div>'; temp += '<div id = "myFAX">FAX: 03-3204-7167</div></div>';
  myAdd.insertAdjacentHTML('beforeend',temp);
}
```

Next is the display of the "Food Order Table" part.

HTML uses< table <./table>fromtable.. <table border="1" class="stuffTable"> ...
</table>

automatically set the border to1and the table with the class namestuffTable. The table includes"Header (from <thead.. to </thead>)", "Body (from <tbody.. to </tbody>)" and "Footer part (<tfoot.. to </tfoot>) to the end. These< listed in </table>from"table.. Since you can write many headers, seteach line (even just one line) from <tr(table row).. to </tr>.

This time, the header part simply displays the item name side by line.

Each square in theheader <th.. </th>.

Therefore, the function"tableStart"that displays theheader part is

described as follows. function tableStart() {
temp = '<table border="1" class="stuffTable">';
temp += '<thead>'; // Header start
temp += '<tr class="myKoumoku">'; Set class name"myKoumoku" to item name line temp +=
'<th> food name </th>'; // Item 1 is "ingredient name" (same below) temp += '<th>000 g
unit price </th>'; Item 2 "100g unit price"
temp += '<th> quantity (× 100 g) </th>'; // item 3 "Quantity (× 100 g)"temp
+= '<th>(yen) </th>'; // Item 4 "Small Scale"
temp += '</tr>'; Item name line end
temp += '</thead>'; // Header End
temp += '<tbody>'; // The next body part starts. put out of repetition

The following is the main part of the food order table. It is described as follows.

```
function tableBody(name, price, amount, sub) {
  temp += '<tr>'; declare to write one line temp += '<td>' + name + '</td>'; // Ingredient name temp +=
'<td>' + price.toLocaleString() + '</td>'; // Three-digit display of unit price temp += '<td>' +
amount.toLocaleString() + '</td>'; // Quantity displayed in three digits temp += '<td>' +
sub.toLocaleString() + '</td>'; // Three-digit display temp += '</tr>'; // End of line 1 }
```

Finally, it is the end part of the food order table. We also included a view of the total totals and the end of the "purchase order" process by vendor, so it was a bit of a bit of a go-to. Seethecomments below: function tableEnd(total) {
temp += '</tbody>'; // Body End


```
temp += '<tfoot>'; // Footer start
temp += '<tr><th></th><th></th><td id="goukeina"> total </td>';
```

49

```
temp += '<th id="goukeiti">' + total.toLocaleString() + '</th></tr>'; let addTux = Math.floor(total * 1.08);
// total 1.08 times, discarded number temp += '<tr><th></th><th></th><td="zeikomina"> Total
(including consumption tax) </td>'; temp += '<th id="zeikomiti">' + addTax.toLocaleString() +
'</th></tr>';
temp += '</tfoot>'; // Footer End
temp += '</table>'; End of the entire food order table
temp += '</div>'; End of class name "eachSheet"
myAdd.insertAdjacentHTML('before',temp); // Write "Purchase Order" by Vendor }
```

In the Js Collector above, if you go back to the site and open the purchase notes menu, and you see something like Figure 26 below the list by view, you're successful. In addition, we will create a function "kaipage()" that "papage" foreach purchase order. function kaipage() {
temp = '<div id = "pagebreak"></div>';
myAdd.insertAdjacentHTML('beforeend',temp);
}

are. In fact, CSS processing breaks pages.

Throw this function immediately after displaying ====purchaseorder issue ====and at the end of the function "tableEnd". (Corresponding This function won't work until you create a CSS script.)

[Figure 26:Displaying the information required for "Purchase Order" / Part of the lower part of "Purchase Note"]

05-06	91	903	d05	野菜炒め	3	1009	牛ロース	7	444	s03	山本コ精肉店
-------	----	-----	-----	------	---	------	------	---	-----	-----	--------

===== 発注書発行 =====			
発注書			
八百屋お七 御中			
FAX: 03-3502-8112			
マイカンパニー株式会社			
〒160-0021			
東京都新宿区歌舞伎町1丁目1-1			
TEL: 03-3204-7166			
FAX: 03-3204-7167			
食材名	百g単価	数量(x百g)	小計 (円)
玉葱	36	16	576
人参	48	44	2,112
		合計	2,688
		合計(消費税込)	2,903

発注書			
タコ八鮮魚店 御中			
FAX: 04-4613-9001			
マイカンパニー株式会社			

(8)-2 Stylepurchase orders with CSS

Style purchase orders in CSS.

In the Administration→,under asset →,go to→CSS Injector and +Add CssInjector.

The labelfield is "ppcchasenotecss".

In the code column, I will write as follows for the time being.

```
#declaration { /* id="declaration" range characters */ font-size: 48px; /* Character size to 48
pixels */ display: flex; /* Three lines are required to display in the center */
```

50

```
justify-content: center;
align-items: center;
}
```

```
#pagebreak {
page-break-before : always; /* Page page */
}
```

The "#declaration" in the first line is the id name of the "purchase order issue" set in JavaScript. "#represents" id. By the way, ". is a class. The following "font-size: 48px; It's obvious, isn't it? The font size is 48 pixels. "px" is a unit of composition of the screen called "pixels". By the way, a 13-inch MacBook is 1280×800 pixels. Below, there are various parameter settings, but please change the number somewhat by yourself and look at the actual display and play. It will be a study.

The next three line means "center" anyway.

#pagebreak is a "page breaks" execution script. In JavaScript, click "kaipage(); The page breaks will be made at the part where you wrote.

Comments are noted by /* and */.

Now, it is [page] in "condition", but → in the page column of "/purchase_note" is described. This is the path name of the view "Purchase Notes". → Save. When you open [Purchase Notes] back to the site, it is written in the center just below the list as "=== purchase order issue =====", but I think that it is no different from Figure 26 after that. Try "print". If you look at the preview, the contents of each paged purchase order by vendor should be displayed in small size in the upper left corner of each page. If this is not the case, try the following: →→Performance→Clear All Caches→When you see "✓ Cache cleared", go back to →→ See Purchase Notes again. This is a good way to get rid of it. If that doesn't work, check the CSS injector. Now let's qualify "contents" with CSS. Open the CSS "ppcchase_note.css" again.

First of all, it is described as "purchase order" on each page.

```
In the JavaScript "purchase_note.js", the function "sheetHeader" has the id
"sheetName". For example, the following description is used for this. #sheetName { /*
id="sheetName" range characters are */ height: 200px; /* Display characters below 100px from
the top of the /* box */ padding-top: 100px; /* box to 200px the height of the virtual box
containing the characters */ font-size: 36px; /* Character size is 36px */
display: flex; /* Center with 3 following line */
justify-content: center;
align-items: center;
}
```

51

Vendors and Faxes are grouped together by the class name `supplier`. This is because it is displayed on the left side of the purchase order page. Do the following:

```
.supplier { /* class="supplier" range matters */ position: absolute; /* in the
"absolute placement" method */
left: 15%; /* Align from left to left of the screen at 15% */ }
```

Vendor qualifies as follows:

```
#supName the range { /* id=" supName" is */ font-size: 24px; /* Character size 24 pixels */
text-decoration-line: underline; /* underline */
}
```

Let's display fax as it is without modification.

"Company information" is summarized by the class name `"myCompany"`. This is displayed on the right side of the page. Write as follows:

```
. myCompany { /* About my company */
position: absolute; /* With the "Absolute Placement" method */
left: 65%; /* Align from left to 65% */ }
```

Now that you display [Purchase Note], "My Company Co., Ltd." was slightly higher than "Supplier Inu", so I decided to lower it by about 2 line. Learning CSS further can be tedious, so I'll fix JavaScript. Write a new line mark "< br>" just before the string "MyCompany" in the function "myCompany". Note that this seems to be the deprecated method. The next is the "Food Order Table". JavaScript writes the class name "stuffTable" in the function "tableStart", so I use this to be involved in the entire table. First of all, it is a vertical position, but it must be below the fax number of the company. Also, the left and right margins should be equal, and the width of the table itself should be 800 pixels. Make the text a little larger than the fax number. See below.

```
. stuffTable {
margin-top: 150px; /* Top open 150px */
margin-left: auto; /* Auto-adjust left and right margins evenly */ margin-right:
auto; /* Same -/
width: 800px; /* Table width to 800px */
font-size: larger; /* One step larger character size */
}
```

The table item names are centered on the cell, and the numbers are right-centered.

```
. The "th" item in the myKoumoku th { /* class="myKoumoku" is */ text-align: center; /*
characters in the center of the cell */
}
```

```
#suji item with a specific { /* id of "suji" is */
text-align: right; /* Right-side */
}
```

Oops, you didn't set "Numbers right-front" in JS Collector. Write "id="suji" in "<>" just before the number (number) is described. The locations are in the

functions"tbody" and"tbodyEnd". For example, temp+= '<td>' + price.toLocaleString() ... But,

「temp += '<td id="suji">' + price.toLocaleString() ... It will be.

Set the cell width of the table.

```
. stuffTable td:first-child { /* Thefirst "td" item in the table (ingredientname) is */ width: 400px; /* Cell width to 400px */ text-align: center; /* Characters in the center of the cell */ }
```

```
. stuffTable td:nth-child(2) { /* The second"td"item in the table(unit price) is */ width: 100px; /* Cell width to 100px */ }
```

Now [Save] CSS.

I'm a little greedy again.

Let's say that the order date (purchase order issue date and time) is displayed in the purchase order and is the "ID(identification mark)" of the purchase order.

This is the last time.

(8)-3 Snake foot (display of "order date")

Represents the Settings (Environment) field.→ [Asset Injector] → [JS Injector]

→ 「purchasenotejsEdit. At the end of the "Code" column, add the following

functions:

```
function hachubi() { // order date
let date = new Date(); Get current date and time let year = date.getFullYear(); // Cut year let
month = date.getMonth() + 1; // Cut month (month starts zero) let day = date.getDate(); //
Cut day let time = date.toLocaleTimeString(); // Time acquisition
let dfmt = 'YYYY MM Month DD Day'; // Date Display Formatting dfmt =
dfmt.replace(/YYYY/g,year);// Replace YYYY in display format with real year dfmt =
dfmt.replace(/MM/ g,month); For months, dfmt = dfmt.replace(/DD/g,day);// For days, return
dfmt to add time to date return dfmt; // return dfmt value for day }
```

In addition, write the following line just before the line "My Company Co., Ltd." written in the function"myCompany".

```
temp += '<div id ="hachubi">, order date: ' + hachubi() + '</div>';
```

Now, after the purchase notes list, if you have a "purchase order" in Figure 27, it's complete by each vendor.

発注書

八百屋お七 御中

FAX: 03-3502-8112

発注日: 2021 年 5 月 27 日 22:48:57

マイカンパニー株式会社

〒160-0021

東京都新宿区歌舞伎町1丁目1-1

TEL: 03-3204-7166

FAX: 03-3204-7167

食材名	百g単価	数量(×百g)	小計 (円)
玉葱	36	16	576
人参	48	44	2,112
		合計	2,688
		合計(消費税込)	2,903

This has finally reached the goal of this book.

Thank you very long.

The result of this document is available at:

<https://dev-maru-1.pantheonsite.io/>

user name: anyone

pass word: any1

You can order dishes and search for purchase notes without logging in, but if you log in, you can "add comments" of "Cooked". Give it a try. All logos and illustrations that appear are handwritten by me, so there is no copyright problem.

I don't think Drupal can only create a homepage or blog site. I think that it can be a means to build a huge core system that integrates not only online advertising and e-commerce, but also business management and production management within companies such as manufacturing.

Even so, I am incompetent, unsophisticated, curious but bored, and there is not much time in my life anymore.

I'm expecting someone to take the cultural baton about Drupal. Above

[Appendix1: JsCollector"ppcasenotejs"code] (Please use it with a copy.)

Extract information from the view "Purchase Notes"

```
const orderTime = document.getElementsByClassName('views-field-field-order-time');// Order date and time const
orderer = document.getElementsByClassName('views-field-field-order');// Reserved const orderID =
document.getElementsByClassName('views-field-title');// Order ID const dishID =
document.getElementsByClassName('views-field-field-order-dish');// Culinary ID const dishName =
document.getElementsByClassName('views-field-field-dish-name');// Dish name const dishQuant =
document.getElementsByClassName('views-field-field-order-quantity');// Number of orders const foodstuffID =
document.getElementsByClassName('views-field-field-dish-foodstuff');// const foodstuffName =
document.getElementsByClassName('views-field-field-foodstuff-name');// const foodamount =
document.getElementsByClassName('views-field-field-dish-foodamount');// const foodstuffPrice =
document.getElementsByClassName('views-field-field-foodstuff-unitprice');// const supplierID =
document.getElementsByClassName('views-field-field-foodstuff-supplier');// const supplierName =
document.getElementsByClassName('views-field-field-supplier-name');// const FAX =
document.getElementsByClassName('views-field-field-supplier-fax');// const orderLength = orderTime.length;// The
number of "order dates and times" is the number of elements
```

Declare an array variable that contains only the data to be cut out of the element (A means array)

```
const orderTimeA = []; // Order date and time
const ordererA = []; // Reservations
const orderIDA = []; // Order ID
const dishIDA = []; // Cooking ID
const dishNameA = []; // Dish Name
const dishQuantA = []; // Orders
const foodstuffIDA = []; // Ingredient ID
const foodstuffNameA = []; // Ingredient Name
const foodamountA = []; // Amount of ingredients
const foodstuffPriceA = []; // Cost per material
const supplierIDA = []; // Store ID
const supplierNameA = []; // Vendor name
const FAXA = []; // FAX
```

Take data from an element and put it in a new array variable

```
for(let index = 0; index < orderLength; index++) { // Repeat in {} from number 0 to number -1=
orderTime[index].textContent.trim();// Order date and time orderA[index] = order[index].textContent.trim();//
Reserve orderIDA[index] = orderID[index].textContent.trim();// Order ID dishIDA[index] = dishID[index].textContent.trim();//
Dish ID dishNameA[index] = dishName[index].textContent.trim();// DishQuantA[index] = dishQuant[index].
textContent.trim();// Orders foodstuffIDA[index] = foodstuffID[index].textContent.trim();// Ingredients ID
foodstuffNameA[index] = foodstuffName[index].textContent.trim();// Ingredient name foodamountA[index] =
foodamount[index].textContent.trim();// FoodstuffPriceA[index] = foodstuffPrice[index].textContent.trim();// Material
unit price supplierIDA[index] = supplyirID[index].textContent.trim();// Store ID supplierNameA[index] =
supplierName[index].textContent.trim();// Vendor Name FAXA[index] = FAX[index].textContent.trim();// FAX
} // End of {}
Display comma-separated ingredients one by one
for(let index = 1; index < orderLength; index++) { // 1 to number "number of elements -1" {...} Repeat let
kokoValue = foodamountA[index].split(",");// Cut out comma-delimited values and put them in array variables
for(let ko = 0; ko < kokoValue.length; ko++) { // Repeat as many times as you cut out foodamountA[index]
= kokoValue[ko]; // FoudamountA[...] with only one value foodamount[index].textContent = kokoValue[ko]; //
Index++; // Increase index by 1 because you have finished one line of the view with only
one value in the corresponding source code } // Repeat the inner for syntax
index--; // Increase index by too much, so reduce it by 1 } // Repeat outer for syntax
```

Aggregate the amount of ingredients by ingredients

```
let stuffSum = {}; // Declaration of associative arrays for food quantity
aggregation Let supplier = {}; // Declaration of associative arrays for
vendor information Let foodstuff = {}; // Declaration of associative arrays for food
information
```

```
for (let index = 1; index < orderLength; index++) { // Repeat data line let supID = supplierIDA[index]; // Assign vendor ID to supID let stuffID = foodstuffIDA[index]; // Ingredient ID To stuffID if(! stuffSum [supID]){ If the store ID is first issued, stuffSum[supID] ={}; // Establish a summary field for the store ID supplier [supID] ={}; // New supplier [supID ] }['name']= supplierNameA[index]; // Assign vendor name supplier[supID]['Fax'] = faxA[index]; // Assign fax number foodstuff [supID] = {}; // New ingredient information field by vendor } // Otherwise, the previous line will not be implemented and if(! stuffSum [supID][staffID]){ If the ingredient ID is first issued stuffSum[supID][staffID] = {} ; // New aggregation field for the foodID stuffSum [supID][ staffID] = 0; Initial value setting of the summary column of the food ID Foodstuff [supID] [staffID] = {} ; // New information column for the ingredientfoodstuff [supID ] [stuffID] ['name']= foodstuffNameA[index]; // Add the name of the ingredient to the ingredient information field foodstuff [ supID] [staffID] ['price'= foodstuffPriceA[index]; // We will not do the previous two line and stuffSum[supID][stuffID]+= dishQuantA[index] * foodamountA[index];// Number of orders in the summary field × Add ingredients }
```

=====

```
const view = document.querySelector('.view-content');// Get the range of class name view-content on HTML let temp = '<div class ="myAddContent"></div>'; Provides elements for the new class name "myAddContent" view.insertAdjacentHTML('after',temp); // immediately after view-content, put the element of"myAddContent" const myAdd = document.querySelector('. myAddContent'); Get class name myAddContent range temp += '<div id = "decalation">===== purchase order issue =====</div>';// Prepare elements to display "Purchase order issue" myAdd.insertAdjacentHTML('before',temp); // Write the above element just before the end of the range of the class name myAddContent kaipage();
```

Create purchase order sheets

```
for (let sID in supplier) { // Repeat per vendor sheetHeader();// Function atama writes purchase orders let supName = supplier[sID] ['name']; // Vendor Name let supFAX = supplier[sID]['FAX']; // FAX supOnchu(supName,supFAX);// [Function]Supplier and fax number myCompany();// [Function]In-house Information tableStart();// [Function] Food Order Table Header let groundTotal = 0; // Total amount initial setup by vendor for ( let fID in stuffSum[sID]) { // Repeat by ingredient by vendor let fName = foodstuff [sID] [fID] ['name']; // Ingredient Name let fPrice = foodstuff [sID] [fID] ['price']; // Cost per material let fAmount = stuffSum[sID][fID];// Total ingredients let subtotal = fAmount * fPrice; // Subtotal calculation tableBody(fName,fPrice,fAmount,subtotal);// [Function]The body of the food order table groundTotal += subtotal; // Add subtotal to the total amount }
```

```
} tableEnd(groundTotal);// [Function]Food Order Table Futta }
```

```
function sheetHeader() { // [function] atama write of purchase order temp = '<div class="eachSheet">'; Finally,don't forget </div>"at the end!! temp += '<div id="sheetName"> purchase order </div>'; Write "Purchase order" write myAdd.insertAdjacentHTML('before',temp); // temp just before the end of myAdd }
```

```
function supOnchu (shop,fax) { // [function] vendor and fax number temp = '<div class="supplier">; Set classnameupplier temp += '<div id ="supName">' + shop + '&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;& Oin <br/></div>'; // Vendor Name + "Goka" + New Line temp += '<div id ="supFAX">FAX:&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;& + fax + '<br/></div></div>'; Add fax+ line breaks myAdd.insertAdjacentHTML('beforeend',temp); // write temp just before the end of myAdd }
```

```
id="myCompName"><br> My Company Inc. </div>'; Company name (fixed description)
```

56

```
temp += '<div id = "myPcode">〒 160-0021</div>'; Company zip code (dite) temp += '<div id = "myAddress"> 1-1-1  
</div>'; Company address (dit) temp += '<div id = "myTEL">TEL:03-3204-7166</div>'; Company phone number (dite)  
temp += '<div id = "myFAX">FAX: 03-3204-7167</div></div>'; MyAdd.insertAdjacentHTML  
('before',temp); // write temp just before the end of myAdd }
```

```
function tableStart () { // [function] ingredient order table header  
temp = '<table border="1" class="stuffTable">'; The beginning of the "table"  
temp += '<thead>'; // Header start  
temp += '<tr class="myKoumoku">'; Set class name "myKoumoku" to item name line temp += '<th> Food  
name </th>'; // Item1 is "Food name" (similar to below) temp += '<th> million g unit  
price </th>'; // Item 2 "Hundred g Unit price"  
temp += '<th> quantity ( × 100 g) </th>'; // item 3 ,Quantity (× 100 g)  
temp += '<th> scale (yen)</th>'; // Item4 "Small Scale"  
temp += '</tr>'; // Item name line end  
temp += '</thead>'; // Header End  
temp += '<tbody>'; // The next body part starts. Put out of repetition }
```

```
function tableBody(name,price,amount,subt) { // [function]the body of the food order  
table temp += '<tr>'; // Declare to write one line  
temp += '<td>' + name + '</td>'; It's the name of the ingredient.  
temp += '<td id="suji">' + price.toLocaleString() + '</td>'; 3-digit temp += '<td id="suji">' +  
amount.toLocaleString() + '</td>'; Display quantity in 3 digits temp += '<td id="suji">' +  
subt.toLocaleString() + '</td>'; 3-digit display temp += '</tr>'; // End of one line  
}
```

```
function tableEnd(total) { // [function] it is the end of the entire food order table temp +=  
'</tbody>'; // Body part end  
temp += '</tfoot>'; // Footer start  
temp += '<tr><th></th><th></th><td id="goukeina"> total </td>'; "Total" description temp += '<th id=" suji">' +  
total.toLocaleString() + '</th></tr>'; Total value let addTux = Math.floor(total * 1.08); // total 1.08 times, discarded by  
a few parts temp += '<tr><th></th><th></th><td id=" zeikomina"> total (including consumption tax) </td>'; "8% Total  
including consumption tax" temp += '<th id=" suji">' + addTux.toLocaleString() + '</th></tr>'; Total value including tax temp +=  
'</tfoot>'; // Footer end  
temp += '</table>'; End of the entire food order table  
temp += '</div>'; End of class name="eachSheet" myAdd.insertAdjacentHTML('before',temp); //  
Write "Purchase Order" by Vendor kaipage();  
}
```

```
function kaipage () { // [function] page breaks  
temp = '<div id = "pagebreak"></div>';  
myAdd.insertAdjacentHTML('beforeend',temp);  
}
```

```
function hachubi() { // [function] order date calculation  
let date = new Date(); Get current date and time  
let year = date.getFullYear(); // Cut Year  
let month = date.getMonth() + 1; // Cut month (month starts at zero)  
let day = date.getDate(); // Cut Day  
let time = date.toLocaleTimeString(); // Time Acquisition  
let dfmt = 'YYYY MM Month DD Day'; // Date Display Formatting  
dfmt = dfmt.replace(/YYYY/g,year); // Replace YYYY in display format with real year  
dfmt = dfmt.replace(/MM/g,month); // for months  
dfmt = dfmt.replace(/DD/g,day); // for days  
dfmt += time; // Add time to date  
Returns dfmt; // returns the value of dfmt  
}
```


[Appendix 2:CSSCollector"pchasenotecss"code] (Please use it with a copy.)

```
#declaration characters in the {{/* id="declaration" range */ font-size: 48px; /*
Character size to 48 pixels */
display: flex; /* 3 line required to display in the center */
justify-content: center;
align-items: center;
}

#pagebreak {
page-break-before : always; /* Page page */
}

#sheetName { /* id="sheetName" range characters are */
height: 200px; /* The height of the virtual box containing the characters to 200px */
padding-top: 100px; /* Show characters 100px below the top of box */
font-size: 36px; /* Character size is 36px */
display: flex; /* Center with 3 following line */
justify-content: center;
align-items: center;
}

.supplier { /* class="supplier" range includes */
position: absolute; /* With the "Absolute Placement" method */
left: 15%; /* Align from left to 15% of the screen */
}

#supName { /* id="supName" range characters are */
font-size: 24px; /* Character size to 24 pixels */
text-decoration-line: underline; /* Underline */
}

.myCompany { /* About my company */
position: absolute; /* With the "Absolute Placement" method */
left: 65%; /* Align from left to 65% */
}

.stuffTable {
margin-top: 150px; /* Top open 150px */
margin-left: auto; /* Auto-adjust left and right margins evenly */
margin-right: auto; /* Dooe */
width: 800px; /* Table width to 800px */
font-size: larger; /* One step larger character size */
}

.MyKoumoku th { /* The"th"item in the class"myKoumoku"is */
text-align: center; /* Characters in the center of the cell */
}

#suji item with a specific { /* id of "suji" is */
text-align: right; /* Right-side */
}

.stuffTable td:first-child { /* The first "td"item in the table (ingredient name) is */
width: 400px; /* Cell width to 400px */
text-align: center; /* Characters in the center of the cell */
}

.stuffTable td:nth-child(2) { /* The second "td"item in the table (unit price) is */
width: 100px; /* Cell width to 100px */
}
```

End

