

eat (one's) own dog food (redirected from *Eat His Own Dog Food*)

Also found in: [Financial](#).

EATING YOUR OWN DOG FOOD

1. To use the product(s) one's company produces or develops as a means of demonstrating or validating its quality, capabilities, or superiority to other products. Used primarily in reference to software industries, the phrase is thought to have originated with advertisements for Alpo dog food in the 1980s, in which actor Lorne Green promoted the product by pointing out that he fed it to his own dogs.

The company sent out a memo to all of its employees telling them to eat their own dog food to demonstrate their new operating system's speed and ease of use.

2. By extension, to use software one's company is developing—usually in its beta form—so as to test it for flaws and ensure its ease of use by end users before it is released.

We didn't have time to eat our own dog food before the new operating system's release, so I'm worried it may still have a lot of glitches that haven't been accounted for yet.

See also: [dog](#), [eat](#), [food](#), [own](#)

BALISAGE 2019

Ari Nordström | ari.nordstrom@gmail.com

eat (one's) own dog food (redirected from *Eat His Own Dog Food*)

Also found in: [Financial](#).

eat (one's) own dog food

1. To use the product (or one's company) produces or develops as a means of demonstrating or validating its quality, capabilities, or superiority to other brands. Used primarily in reference to software industries, the phrase is thought to have originated with advertisements for Alcot dog food in the 1990s, in which actor Luke Green promoted the product by pointing out that he fed it to his own dogs.

The company sent out a memo to all of its employees telling them to eat their own dog food to demonstrate their new operating system's speed and ease of use.

2. By extension, to use software one's company is developing—usually in its beta form—so as to test it for flaws and ensure its ease of use by end users before it is released.

We didn't have time to eat our own dog food before the new operating system's release, so I'm worried it may still have a lot of glitches that haven't been accounted for yet.

See also: [dog](#), [eat](#), [food](#), [own](#)

PROGRAMMING



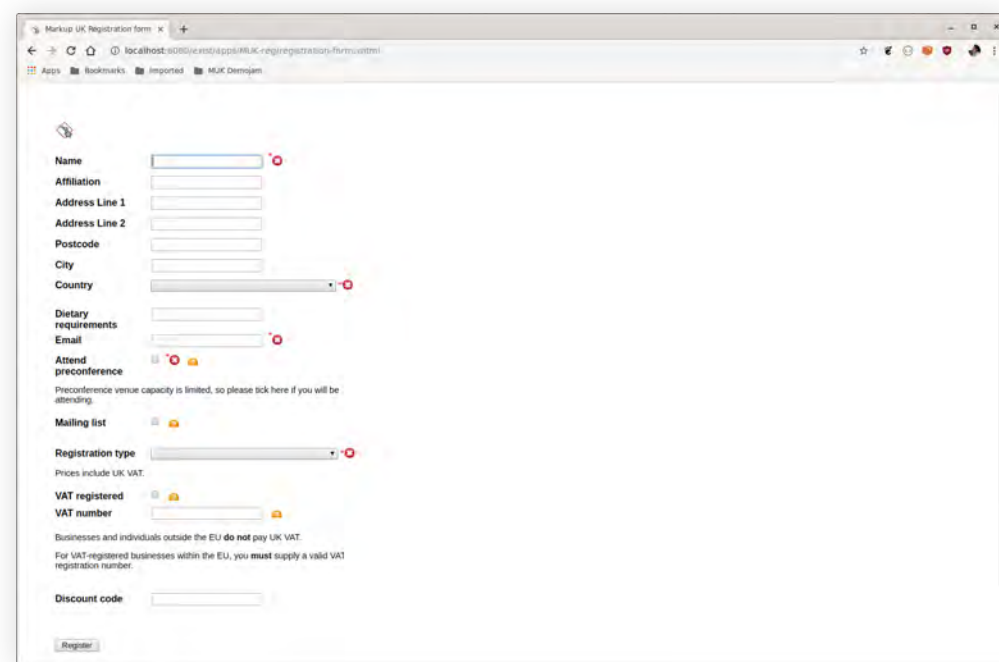


A CONFERENCE IS BORN

PANIC



2018: PARTIAL SOLUTION



The screenshot shows a web browser window with the title 'Markup UK Registration form'. The address bar shows a local file path. The form contains the following fields and sections:

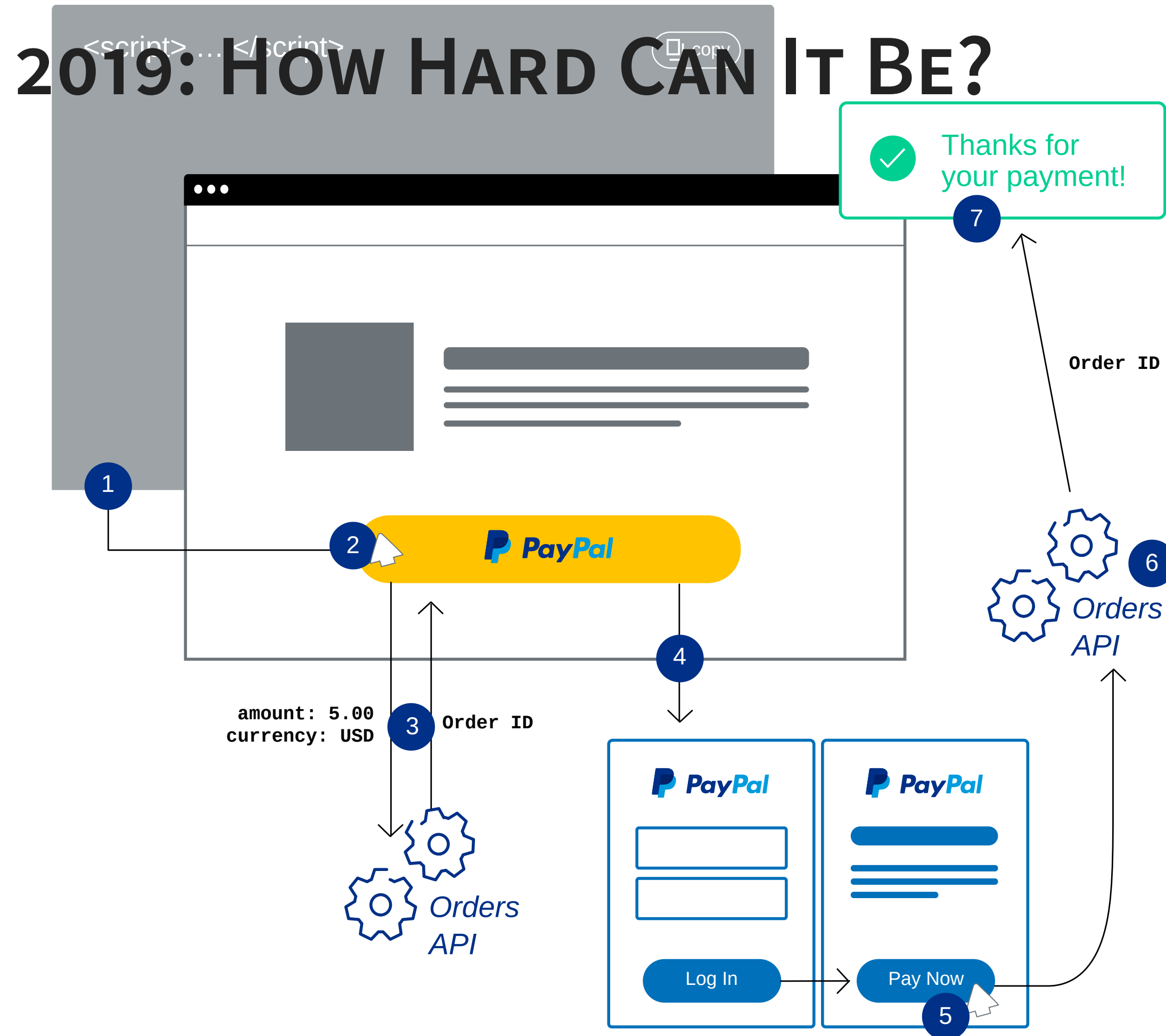
- Name**: Text input field.
- Affiliation**: Text input field.
- Address Line 1**: Text input field.
- Address Line 2**: Text input field.
- Postcode**: Text input field.
- City**: Text input field.
- Country**: Dropdown menu.
- Dietary requirements**: Text input field.
- Email**: Text input field.
- Attend preconference**: Checkboxes for 'Yes' and 'No'.
- Mailing list**: Checkboxes for 'Yes' and 'No'.
- Registration type**: Dropdown menu.
- VAT registered**: Checkboxes for 'Yes' and 'No'.
- VAT number**: Text input field.
- Discount code**: Text input field.

At the bottom of the form is a 'Register' button.





2019: HOW HARD CAN IT BE?



OFF THE PAYPAL WEBSITE...

```
<script src="https://www.paypal.com/sdk/js?client-id=sb"/>
<script>paypal.Buttons().render('body');</script>
...
<form
  action="https://www.paypal.com/cgi-bin/webscr"
  method="post">
  <input
    type="hidden"
    name="cmd"
    value="_xclick"/>
</form>
```

```
<INPUT TYPE="hidden" NAME="currency_code" value="GBP"/>
```




No CUSTOM AMOUNTS

EARLY-BIRD, ACADEMIC, FULL, DISCOUNTS...



GENERATING BUTTON CODE

My Saved Buttons

[Back to My Profile](#)

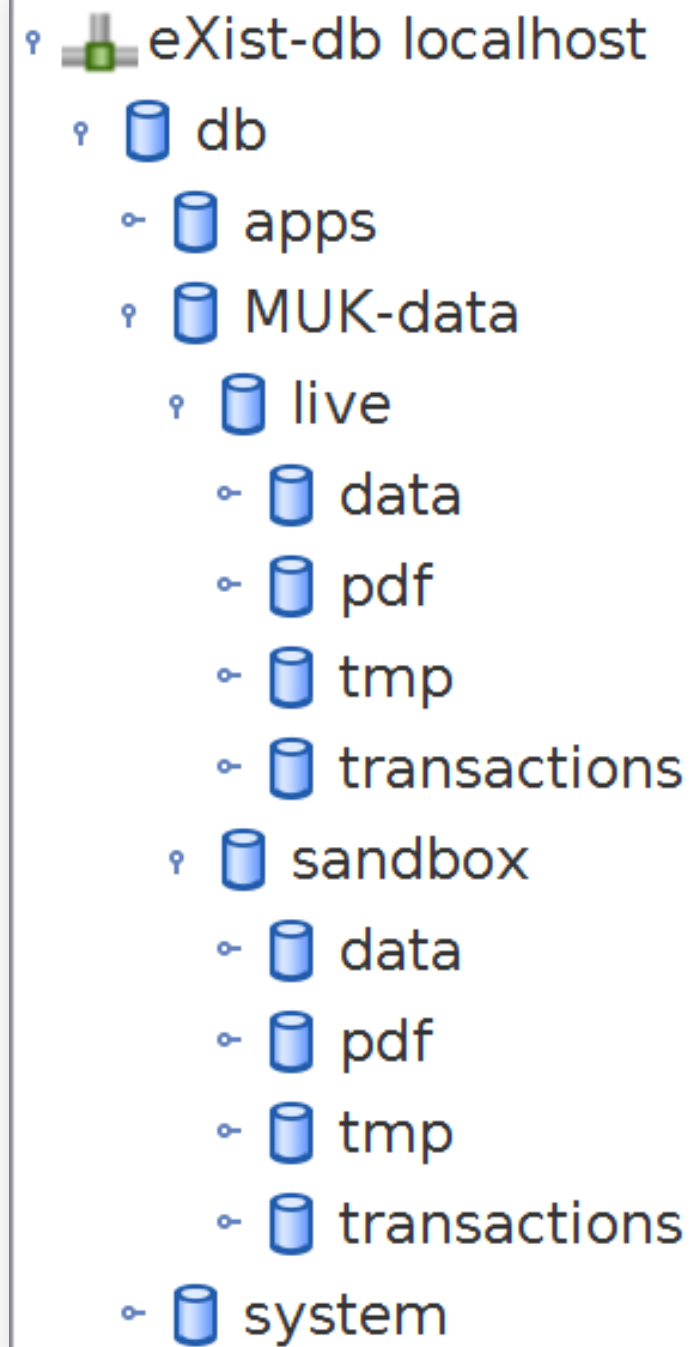
We automatically save buttons you create when you're logged in. Here, you can change most button characteristics, view the HTML code or create new buttons that are similar to existing ones. However, you may not be able to change fields if you've created them using an API.

Click **Action** or one of the links in **Related Items** to get started.

Item name	Qty. available	Price		Related Items
▶ Notify-test			Action ▼	Create new button
▶ Registration			Action ▼	Reports
▶ Full			Action ▼	Manage checkout page styles
▶ PC			Action ▼	
▶ Academic			Action ▼	
▶ Early-bird			Action ▼	
▶ Sample Buy Now Button			Action ▼	
▶ Sample Add to Cart Button			Action ▼	
▶ Sample Subscription Button			Action ▼	


```
<button id="early-bird-sandbox">
  <form
    action="https://www.sandbox.paypal.com/cgi-bin/webscr"
    method="post"
    target="_top">
    <input
      type="hidden"
      name="cmd"
      value="_s-xclick"/>
    <input
      type="hidden"
      name="hosted_button_id"
      value="PM4DRL6BX3226"/>
    <input
      type="image"
      src="https://www.sandbox.paypal.com/en_US/GB/i/btn/btn_bu
      border="0"
      name="submit"
      alt="PayPal - The safer, easier way to pay online!"/>
    <img
      alt=""
      border="0"
```

SANDBOX/LIVE



A screenshot of a database explorer interface showing the eXist-db localhost hierarchy. The root node is 'eXist-db localhost', which contains a 'db' node. The 'db' node contains several sub-nodes: 'apps', 'MUK-data', 'live', 'sandbox', and 'system'. The 'live' and 'sandbox' nodes each contain sub-nodes for 'data', 'pdf', 'tmp', and 'transactions'.

- 🔗 eXist-db localhost
 - 🔗 db
 - 🔗 apps
 - 🔗 MUK-data
 - 🔗 live
 - 🔗 data
 - 🔗 pdf
 - 🔗 tmp
 - 🔗 transactions
 - 🔗 sandbox
 - 🔗 data
 - 🔗 pdf
 - 🔗 tmp
 - 🔗 transactions
 - 🔗 system

LOTS OF BUTTONS

```
<buttons>
  <button id="early-bird-sandbox">
    ...
  </button>

  <button id="academic-sandbox">
    ...
  </button>

  <button id="pc-sandbox">
    ...
  </button>

  <button id="full-sandbox">
    ...
  </button>

  ...
</buttons>
```

A dog is swimming in the water, wearing a red life preserver. The dog's head is above water, and it appears to be looking towards the camera. The water is dark and rippled. The text "THANKFULLY I FOUND A THIRD OPTION" is overlaid in white, bold, sans-serif font across the middle of the image.

THANKFULLY I FOUND A THIRD OPTION

OVERVIEW

BASIC INTEGRATION

1. Set Up Your
Development Environment

2. Add Script

3. Render the Buttons

4. Set Up Transaction

5. Capture Transaction

6. Verify Transaction

7. Test it

8. Go Live

INTEGRATION FEATURES

BEST PRACTICES

TROUBLESHOOT

REFERENCE

5. Capture the transaction

Next, implement the `onApprove` function, which is called after the buyer approves the transaction on paypal.com. This function:

- Optionally calls PayPal using `actions.order.get()` to get the transaction details and display them to the buyer.
- Calls PayPal using `actions.order.capture()` to capture the funds from the transaction.
- Shows a message to the buyer to let them know the transaction is successful.

PAYPAL DEVELOPER DOCUMENTATION

Note: For the basic integration you'll do this step on the client. For advanced use cases, you can also [call from your server](#) to capture a transaction.

```
<script>
paypal.Buttons({
  createOrder: function(data, actions) {
    return actions.order.create({
      purchase_units: [{
        amount: {
          value: '0.01'
        }
      }]
    });
  },
  onApprove: function(data, actions) {
    // Capture the funds from the transaction
    return actions.order.capture().then(function(details) {
      // Show a success message to your buyer
      alert('Transaction completed by ' + details.payer.name.given_name);
    });
  }
});
```

HTML

I FOUND THIS EXAMPLE

```
<script>
  paypal.Buttons({
    createOrder: function(data, actions) {
      // Set up the transaction
      return actions.order.create({
        purchase_units: [{
          amount: {
            value: '0.01'
          }
        }]
      });
    }
  }).render('#paypal-button-container');
</script>
```



```
amount: {  
  value: '{$amount}'  
}
```

A BETTER BUTTON

```
<script>
  paypal.Buttons({{
    createOrder: function(data, actions) {{...;}},

    onApprove: function(data, actions) {{...}},

    onCancel: function(data, actions) {{...}}

  }}).render('#paypal-button-container');
</script>
```

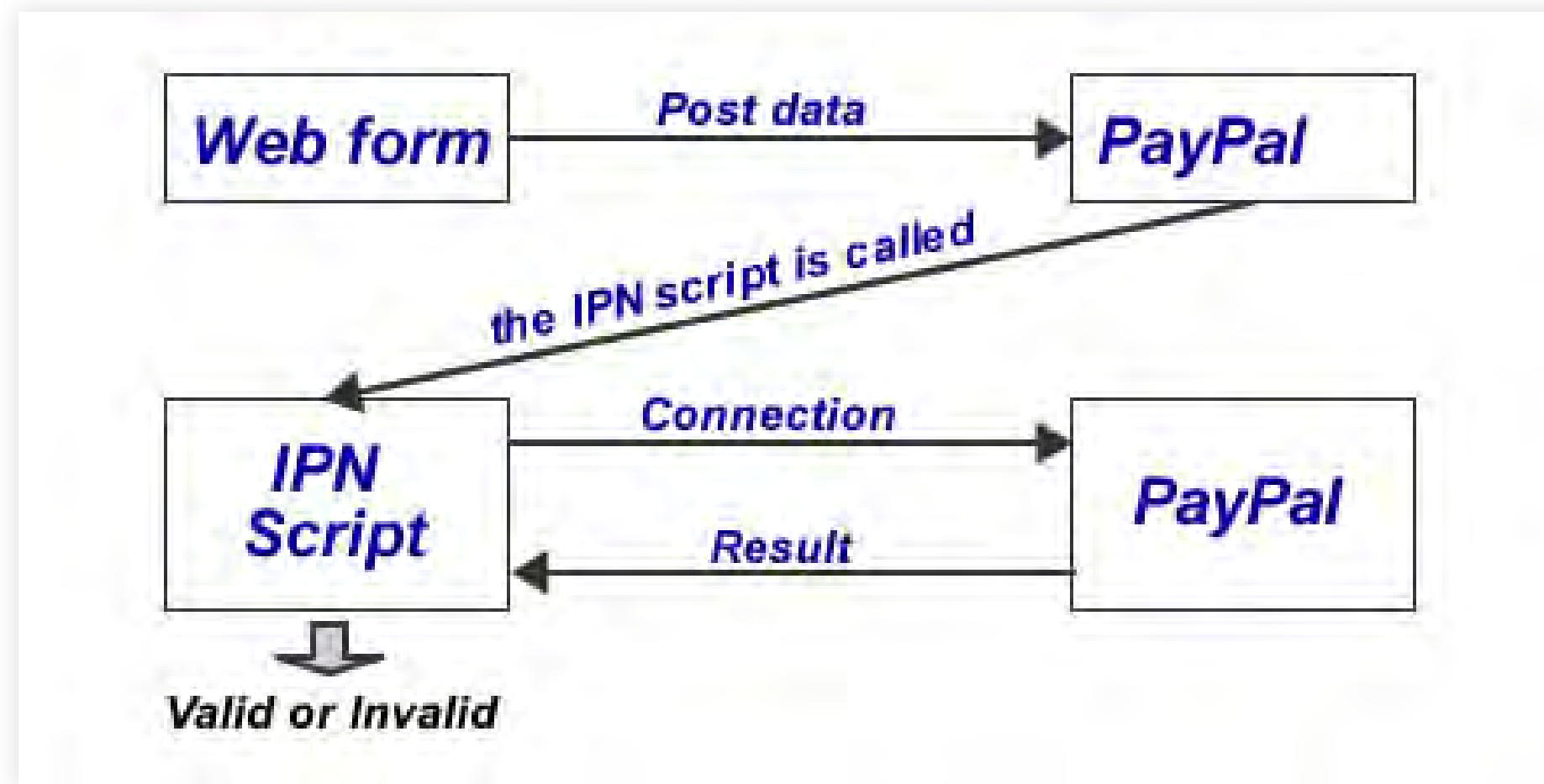


```
onApprove: function(data, actions) {{
  return actions.order.capture().then(function(details) {{

    actions.redirect(
      '{$muk:config//.../thanks.xquery?tmp-file={$registered}}'
    );
    // Call your server to save the transaction
    return fetch(...);
  }});
}},

onCancel: function(data, actions) {{
  actions.redirect('{$muk:config//.../cancel.xquery}');
}}
}}).render('#paypal-button-container');
```

BUT FIRST, YOU NEED TO VERIFY



IPN MESSAGE

```
verify_sign=Atk0fCXbDm2hu0ZELryHFjY-Vb7PAUvS6nMXgysbElEn9v-1XcmSoGtf&  
payer_email=gpmac_1231902590_per%40paypal.com&  
txn_id=61E67681CH3238416&  
payment_type=instant&  
last_name=User&  
address_state=CA&  
receiver_email=gpmac_1231902686_biz%40paypal.com&  
payment_fee=0.88&  
receiver_id=S8XGHLYDW9T3S&  
txn_type=express_checkout&  
item_name=&  
mc_currency=USD&  
item_number=&  
residence_country=US&  
test_ipn=1&  
handling_amount=0.00&  
transaction_subject=&  
payment_gross=19.95&  
shipping=0.00
```



Um, what's a "listener?"

I CREATED A TEST FORM

```
<form
  target="_new"
  method="post"
  action="http://localhost:8080/exist/rest/db/test/test.xquery"
  enctype="multipart/form-data">

  <input type="hidden" name="payer_id" value="LPLWNMTBWMFAY"/>
  <input type="hidden" name="payment_status" value="Completed"/>

  <!-- code for other variables to be tested ... -->
  <input type="hidden" name="custom" value="my_info"/>

  <input type="submit"/>
</form>
```


POINTING OUT THIS XQUERY

```
xquery version "3.1";
declare namespace exist = "http://exist.sourceforge.net/NS/exist";
declare namespace xmldb="http://exist-db.org/xquery/xmldb";
declare namespace request="http://exist-db.org/xquery/request";
declare option exist:serialize "method=xml media-type=text/xml indent=2";

let $post-data := request:get-data()

return
<post-data>
  {$post-data}
</post-data>
```

HEAR, HEAR!

A Guide to Training a Deaf Dog









By Barry Eaton

HELP ME (YOU'RE MY ONLY HOPE)




```

1 xquery version "3.1";
2
3 module namespace ipnl = "http://markupuk.org/registration/ipn-listener";
4
5 declare namespace err = "http://www.w3.org/2005/xqt-errors";
6 declare namespace rest = "http://exquery.org/ns/restxq";
7 declare namespace output = "http://www.w3.org/2010/xslt-xquery-serialization";
8 declare namespace http = "http://expath.org/ns/http-client";
9
10 (: Indicates if the sandbox endpoint is used :)
11 declare variable $ipnl:use-sandbox := true();
12
13 (: Production Postback URL :)
14 declare variable $ipnl:verify-uri := "https://ipnpb.paypal.com/cgi-bin/webscr";
15
16 (: Sandbox Postback URL :)
17 declare variable $ipnl:sandbox_verify_uri := "https://ipnpb.sandbox.paypal.com/cgi-bin/webscr";
18
19 (: Response from PayPal indicating validation was successful :)
20 declare variable $ipnl:valid := "VERIFIED";
21
22 (: Response from PayPal indicating validation failed :)
23 declare variable $ipnl:invalid := "INVALID";
24
25 declare
26     %rest:POST("${body}")
27     %rest:path("/registration/ipnl")
28 > function ipnl:listener($body) {};
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54 declare
55     %private
56 > function ipnl:verify-ipn($body) as xs:boolean {};
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112 (:~
113 : Parses an IPN from a HTTP Request Body into an XDM array
114 :
115 : Each entry in the array is a sequence of two values, the first is a key, the second is the value
116 :
117 :)
118 declare
119     %private
120 > function ipnl:parse-ipn-from-body($body) as array(*) {};
121
122
123
124
125
126
127
128 declare
129     %private
130 > function ipnl:serialize-ipn-for-body($my-post as array(*)) as xs:string? {};
131
132
133
134
135
136
137
138 declare
139     %private
140 > function ipnl:get-paypal-uri() {};
141
142
143
144
145
146
147
148 declare
149     %private
150 > function ipnl:substr-count($str as xs:string, $pattern as xs:string) as xs:integer {};
151
152
153
154

```

MY SIMPLE "LISTENER"

```
declare
  %rest:POST("${body}")
  %rest:path("/paypal-transaction-complete")
function pay:listener($body) {
  let $json := fn:parse-json(util:base64-decode($body))

  return
  xmldb:store($muk:transactions-collection-uri,
    concat('uid-', $json?custom, '.xml'),
    <transaction tstamp='{current-dateTime()}'
      orderId='{ $json?orderId }'
      userID='{ $json?custom }'/>),
  pay:all-good()
};

declare function pay:all-good(){
  (: Reply with an empty 200 response to indicate to paypal the IPN was received :)
  (
    <rest:response>
      <http:response status="200" reason="OK"/>
    </rest:response>,
    ""
  )
}
```

ADDING THE ENDPOINT

```
onApprove: function(data, actions)
  ...

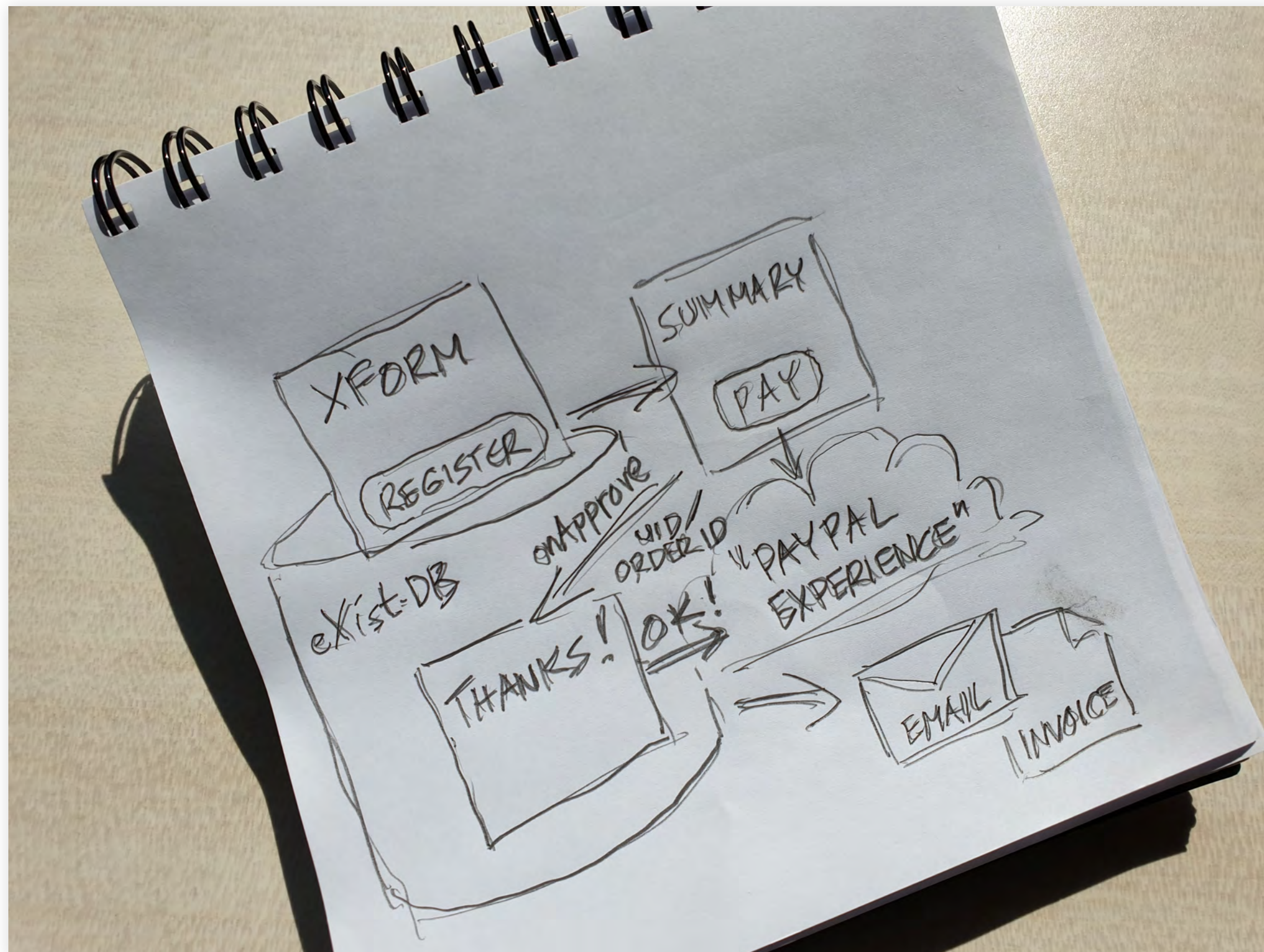
  // Call your server to save the transaction
  return fetch(
    '{$muk:config//domain}/exist/restxq/paypal-transaction-complete', {{
  method: 'post',
  body: JSON.stringify({{
    orderID: data.orderID,
    custom: {string($uid)}
  }})
  }});

  ...

onCancel: function(data, actions) {{
  ...
```


AND IT ALL STARTED WORKING

Recent activity			More >
Ready to send	Payments received	Payments sent	Activity (including balance & fees)
7 Apr 2019	Payment from test buyer Completed		£280.00 GBP
7 Apr 2019	Payment from Mark Up Completed		£215.00 GBP
5 Apr 2019	Payment from Mark Up Completed		£215.00 GBP
10 Mar 2019	Payment from Mark Up Completed		£215.00 GBP



NOW THAT I'M A PROGRAMMER...



ADMIN PAGE



eXist Domain



MUK Data Root



**MUK Data
Collection**

Sandbox

Live

Temp

Transactions

PDF

Environment

SET AMOUNTS AND DISCOUNTS

Pricing in GBP Inc VAT

Full

336

Early-bird

258

Academic

204

PC Member

204

Speaker

0

Current Discounts

PRAGUE2019

243

TEST

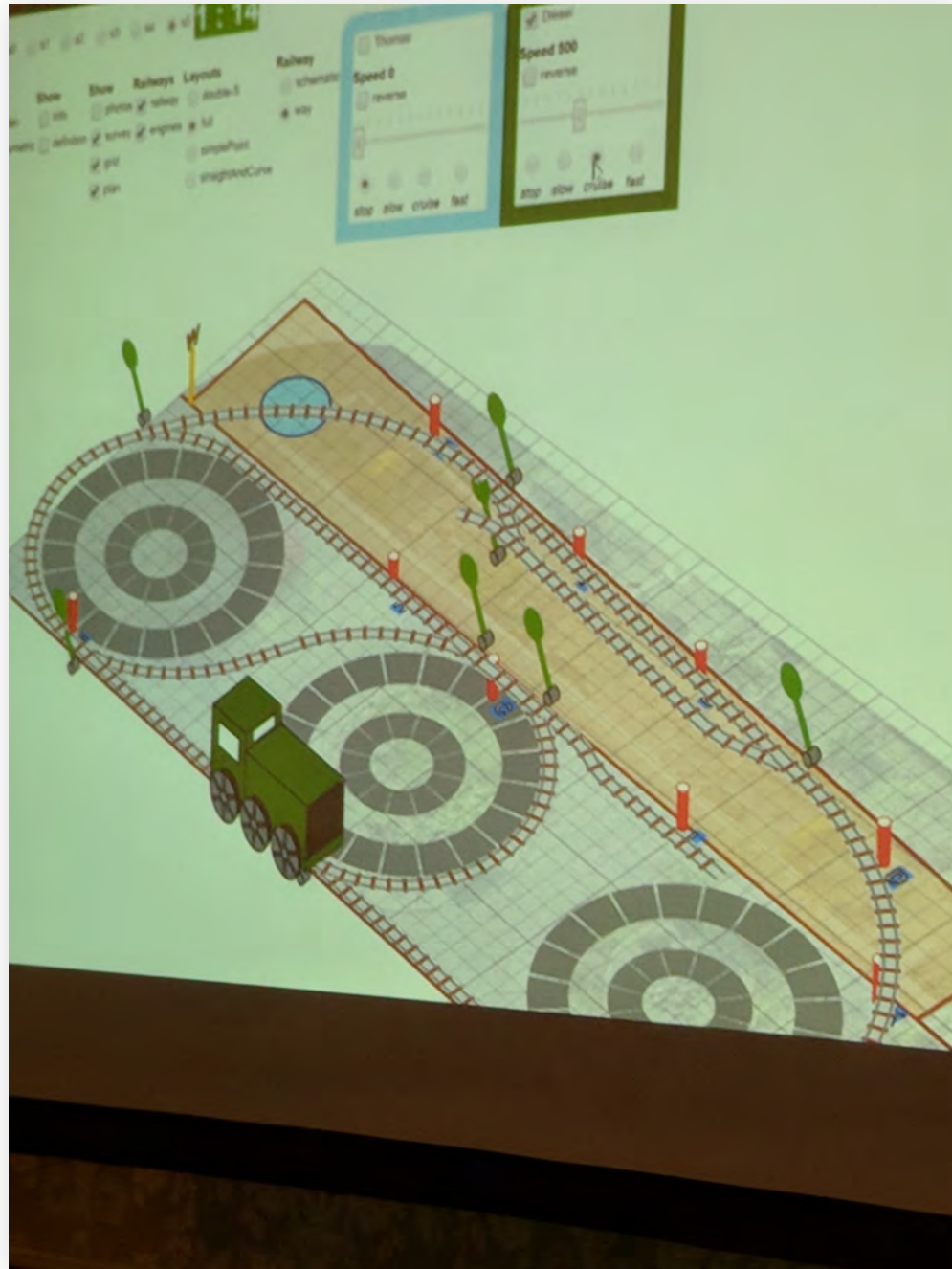
1.5

Add Row

Delete Selected Row

Update Pricing

DEMOJAM VOTING APP



Presentations

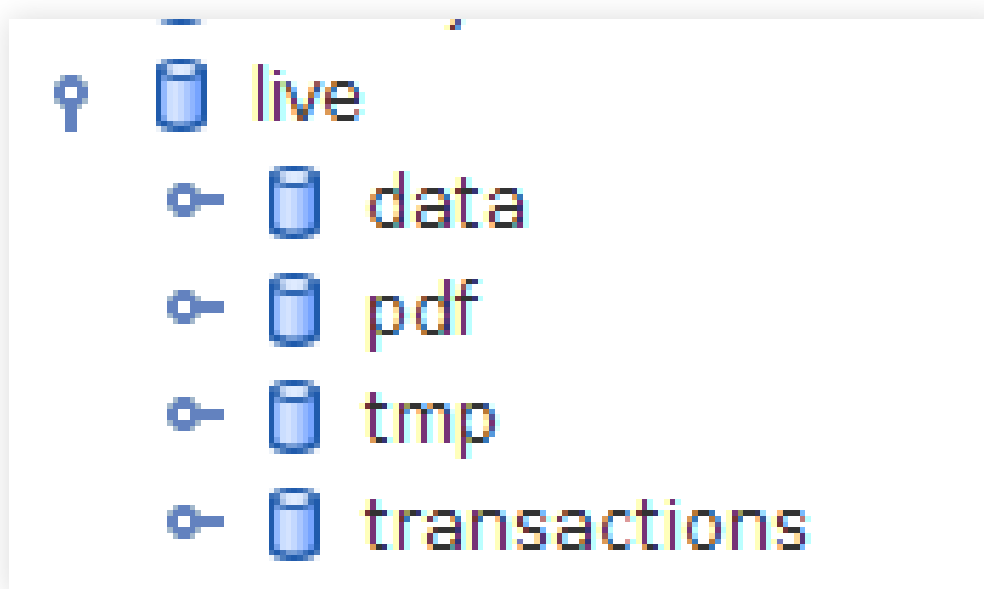
- ☐ Edgar Wallace - Beer Is My Life
- ☐ John Smythe - Why I Am
- ☐ Adrian Hubert Rollerblade - The Joy of Money
- ☐ Charlotte Always - Thinking Fish
- ☐ Mycroft James - Study in Pink
- ☐ William Superman - Why My Life Is Ordinary
- ☐ Antonia Bucket - My Life
- ☐ Winona Kiefer - Interesting Newspaper

Clippings

- ☐ Steven Seagal - Romantic Notions
- ☐ JFK - How I Bit the Bullet
- ☐ Lyndon B Johnson - Good Times
- ☐ Mia War - Final Call
- ☐ A B Letter - Books
- ☐ Test Again - For Show
- ☐ test - test
- ☐ Final - Call

Vote

REPORTS



UID 46 Sheila T	full		2019-05-19T12:06:14.429-04:00
UID 47 Stephen C	full		2019-05-20T16:14:55.34-04:00
UID 48 M	full		2019-05-22T10:14:02.226-04:00
UID 49 Nic G	full		2019-05-22T17:18:38.062-04:00
UID 50 Nic G	full		2019-05-23T05:18:57.337-04:00
UID 51 Andor D	academic		2019-05-23T10:28:53.594-04:00
UID 52 Jenny Shee	speaker		2019-05-23T11:14:45.796-04:00

2020: ENTERPRISE-READY?



THANK YOU!

