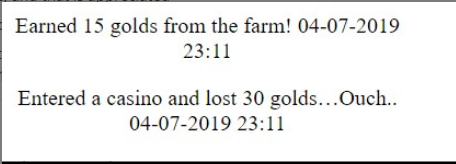
Ninja Gold Notes

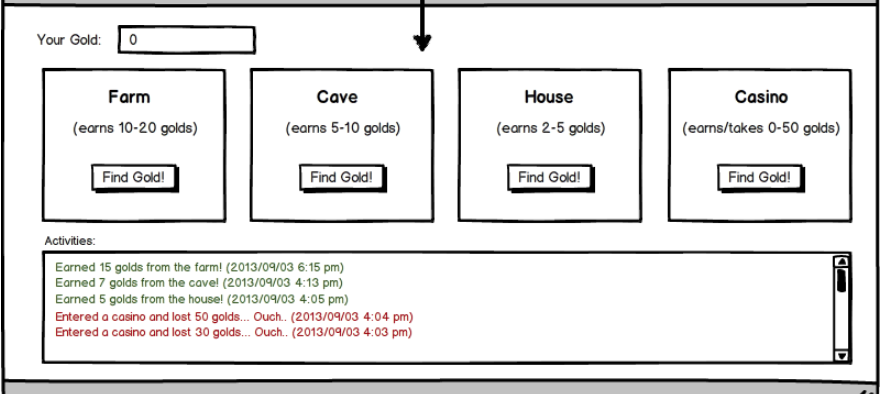
Need to solve:

* How to make the activities list
  + First try starts to work
    - In .py
      * activity\_string="<p>you won lots of gold</p><p>You won a little gold</p>"
      * return render\_template("ninjagold.html", activity\_results=activity\_string)
    - In html
      * <div><!--This div will hold the activities-->
      * {{ activity\_results|safe}}
      * </div>
  + As a test, I used dictionary with static content and it worked Sunday night
    - **kent\_h**
    - [8:49 PM](https://dojo.news/coding-dojo/pl/hi66t4gwej8mpxgcg8q8e41rsa)
    - yes, and that is appreciated
    - Look...I got it to work
    - activity\_string={1:'Earned 15 golds from the farm! 04-07-2019 23:11', 2:'Entered a casino and lost 30 golds…Ouch.. 04-07-2019 23:11'}
    - {% for activity in activity\_results %} <p>{{activity\_results[activity] |safe}}</p> {% endfor %}
    - **Image Pasted at 2019-4-7 20-50.png**
    - 
    - **shannon.steen**
    - [8:50 PM](https://dojo.news/coding-dojo/pl/mn6bzzmiptyrteiykw4m7kr7dy)
    - awesome
  + Late Sunday realized I did not have any keys to control things except for numbers and that it is too easy to write over dictionaries, so I switched to lists. It works except the system variables show up as NONE and that won’t iterate
* It looks like there is only one html page probably call it display.html
* How to redirect full CD explanation at bottom of this file:
  + This may explain the second route that CD called for
  + Add redirect as an output as in:
    - from flask import Flask, render\_template, request, redirect
  + make this the last line of the working @app.route where xxx is the other route:
    - * return redirect("/xxxx")
  + Make new route like below whose only purpose is to re-display the page:
    - @app.route("/xxxx")
    - *def* show\_user():
    - print("Showing the User Info From the Form")
    - print(request.form)
    - return render\_template("pagename.html")
      * And this:  
         if 'total\_gold' not in session:
      * session['total\_gold'] = 0
      * if 'activity\_list' not in session:
      * session['activity\_list'] = []
* Key value pairs for forms
  + Looks like there is only one name/value for each location
  + The submit button is exactly the same for all
* Key value pairs for session cookies
  + On his list
    - Key total\_gold
      * Value starts at zero when cookie created
    - Activities are also stored as key--value
* How to generate random numbers from -50 to +50, and other ranges
  + x = random.randint(-50,50)
* How to generate activities such as
  + - Earned 15 golds from the farm (mmyydd hh:mm)
    - Entered a casino and lost 50 golds…Ouch (mmyydd hh mm)
  + What is needed:
    - A way to color such as If loss, then div class=loss\_text….else class = won\_text
    - date\_time= current data and time converted to string
    - Generate the values if won, and values if lost
      * If form key=farm, then
        + did\_win= True
        + new\_gold=random 10-20
      * if form key=cave, then
        + did\_win= True
        + new\_gold=random 5-10
      * if form key=house, then
        + did\_win= True
        + new\_gold=random 2-5
      * if form key = Casino
        + new\_gold=random -50 to +50
        + if new\_gold >0

did\_win=True

else did\_win=False

* + - * NOTE>>>>May not need the did\_win variable



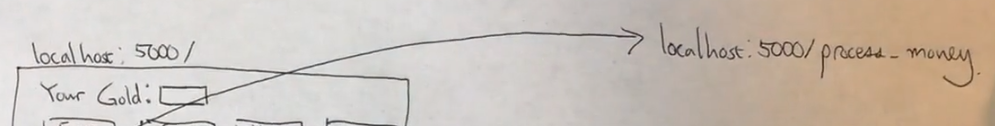
Earned 15 golds from the farm! 04-07-2019 23:11

Entered a casino and lost 30 golds…Ouch.. 04-07-2019 23:11

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

>>>Below is much from video

Per below, the page that does some work is called route process\_money



We learned that when form is submitted the browser will package will package that as a dictionary and make that available to another url

displaypage.html for forms, but of course submitting one submits them all

In every case below there is a hidden field because user is not entering any data

Farm

Cave

House

Casinao

**import** random  
x = random.randint(-50,50)  
print(x)

Dictionary that come over is from form is

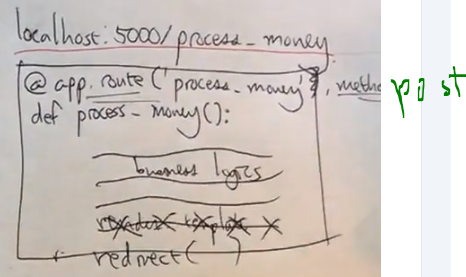
Key=Name from html…value=value from form

========-

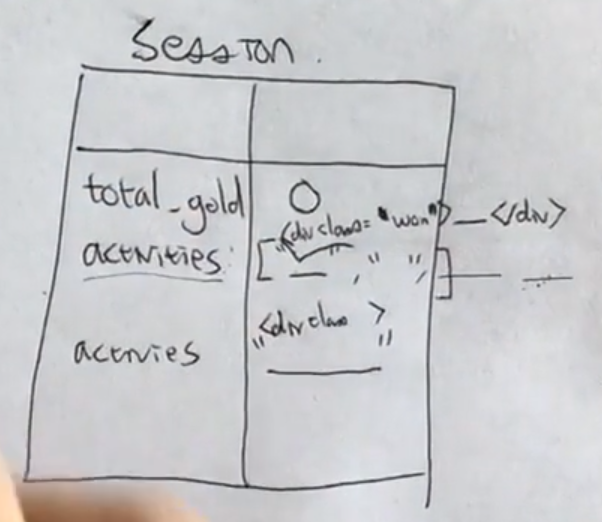
First route

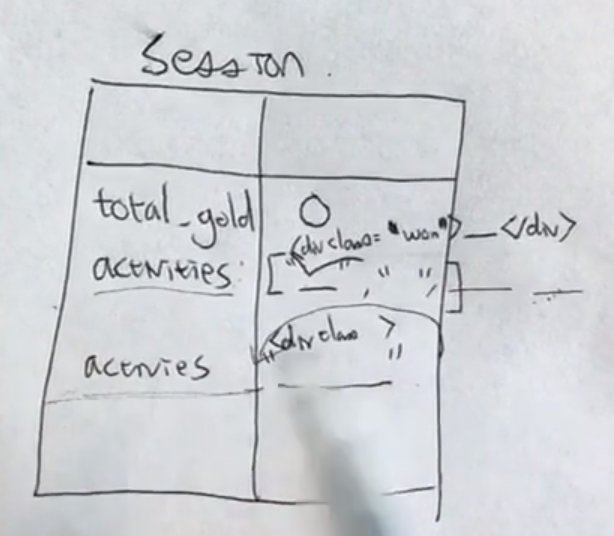
He said the below is a post, not a get

Don’t render to another page, but redirect back to the same page (look up redirect from previous CD info), but maybe one way to redirect is to use same html name…or maybe not



He said that we want to remember gold and total activities and the way to do that is with session cookies as key value pairs as below:





|  |  |  |
| --- | --- | --- |
| Key----v | value | Notes |
| total\_gold |  | If not there, then create and set to zero |
| Activities |  | Could be list…loop and print…….  Have to show in colors based on gain or lost  One way is to store as list…but then he seemed to say store as string that increases each time…..html including a div with class indicating won or lost…with string technique put new activities in front |
|  |  |  |
|  |  |  |

>>>End video

Below is text from page:

For this assignment, you're going to create a mini game that helps a ninja make some money! When you start the game, your ninja should have 0 gold. The ninja can go to different places (farm, cave, house, casino) and earn different amounts of gold. In the case of a casino, your ninja can earn or lose up to 50 gold. Your job is to create a web app that allows this ninja to earn gold and to display their past activities.

Consider the following code:

**my\_proj/server.py**

*def* index():

message = "<ul><li>Hello</li></ul>"

return render\_template("index.html", message=message)

my\_proj/templates/index.html

{{ message }} {{ message|safe }}

The point is without the safe pipe, the output is <ul><li>Hello</li></ul>"

But with the safe pipe, the output is:

* Hello

By default, Jinja will convert any [html entities with character entities](https://www.w3schools.com/html/html_entities.asp). To prevent this from happening, we used the safe pipe, which you can read about [in the Flask documentation](http://jinja.pocoo.org/docs/2.10/templates/#working-with-automatic-escaping) and [on StackOverflow](https://stackoverflow.com/questions/12341496/jinja-2-safe-keyword).

### Working with Automatic Escaping

When automatic escaping is enabled, everything is escaped by default except for values explicitly marked as safe. Variables and expressions can be marked as safe either in:

1. the context dictionary by the application with MarkupSafe.Markup, or
2. the template, with the |safe filter

The main problem with this approach is that Python itself doesn’t have the concept of tainted values; so whether a value is safe or unsafe can get lost.

If a value is not marked safe, auto-escaping will take place; which means that you could end up with double-escaped contents. Double-escaping is easy to avoid, however: just rely on the tools Jinja2 provides and don’t use builtin Python constructs such as str.format or the string modulo operator (%).

Jinja2 functions (macros, super, self.BLOCKNAME) always return template data that is marked as safe.

String literals in templates with automatic escaping are considered unsafe because native Python strings (str, unicode, basestring) are not MarkupSafe.Markup strings with an \_\_html\_\_ attribute.

List of tasks:  
 Create a new Flask project called ninja\_gold



Create the template as shown in the wireframe above, with 4 separate forms



Have the root route render this page



Have the "/process\_money" POST route increase/decrease the user's gold by an appropriate amount and redirect to the root route



NINJA BONUS: Display all the activities performed by the user in a log on the HTML, as shown in the wireframe



NINJA BONUS: Have the activities be color-coded as shown above (+ money is green, - money is red)



NINJA BONUS: Add a reset button to restart the game



SENSEI BONUS: Have the activities display in descending order, with the most recent activity first



SENSEI BONUS: Provide winning parameters to the game--for example, a user must obtain 500 gold in less than 15 moves. Only display the reset button once the user has won or lost.



SENSEI BONUS: Complete the "/process\_money" route without 4 conditional statements (i.e. without doing if farm...elif cave...etc.)

------------CD on redirect:

When we have finished processing the POST data, we can perform a GET request on behalf of the client, which will now be the request that is completed should the client refresh the page. This is called ***redirecting***. **Always redirect after handling POST data to avoid data being handled more than once!**

Let's see how it works by going back to our form\_test project. Our create\_user method is the one processing the POST request. Run the project and submit the form. Hit the refresh button in your browser--you should see a pop up like the one on the right. The warning explains that we are sending the same form data to be processed again. While it's not a big deal yet, imagine our method was inserting a user into the database. Clicking *continue* would add that user to the database again. That would be no good!

Let's add a separate method that will be solely responsible for rendering the show page, and then change the last line of our method handling the POST data from render\_template to redirect to the route that will render the page:

from flask import Flask, render\_template, request, redirect # don't forget to import redirect!

@app.route('/users', methods=['POST'])

*def* create\_user():

print("Got Post Info")

print(request.form)

name = request.form['name']

email = request.form['email']

return redirect("/show") # changed this line!

# adding this method

@app.route("/show")

*def* show\_user():

print("Showing the User Info From the Form")

print(request.form)

return render\_template("show.html")

=========Shannon convo on .py laout:

**kent\_h**

[6:48 PM](https://dojo.news/coding-dojo/pl/k4maq5anm7rkuy1yjobu9emt4a)

Instructions say:

Note: You should only have 2 routes for this assignment -- '/' and '/process\_money'

however, they also say to use redirect which uses two routes as in: <http://learn.codingdojo.com/m/101/5818/39727> saying: return redirect("/show")

And return render\_template("show.html")

Also, see image from MChoi's video

So I am planning to use '/' for the first actions when user enters page And the other route mentioned, '/process\_money' , for the redirect page

Does that sound right?

**Image Pasted at 2019-4-7 18-48.png**

But, no, maybe because here he says process\_money does the work of calculating things

**Image Pasted at 2019-4-7 18-50.png**

Oh, he has a clue at the bottom of process\_money....it shows "redirect"

It helps for me to talk this stuff out

I think maybe even though there are two routes...there is only one html page......

New Messages

**shannon.steen**

[6:54 PM](https://dojo.news/coding-dojo/pl/d8ra9ncfwib59p56x1zcsfo7oo)

you got it. you can do this with only one html page

**kent\_h**

[6:54 PM](https://dojo.news/coding-dojo/pl/buxht3nt37dfmx7c6topp17c8a)

The redirect instructions say: When we have finished processing the POST data, we can perform a GET request

I wonder how that works with my earlier problem of get versus post

Thanks.....

**shannon.steen**

[6:54 PM](https://dojo.news/coding-dojo/pl/1mnw8rh3yi8pug4xsatdka773e)

so you can have both GET and POST in the same route

**kent\_h**

[6:54 PM](https://dojo.news/coding-dojo/pl/6khy5x55nbyh3xkhdiouxmfk9o)

I am going to start on my html page. I think my css willl work better if I get my div and other tags right

[6:55 PM](https://dojo.news/coding-dojo/pl/bfsq98zndfbwffsiqu8wzsbpye)

Oh.....I have seen something like that...do I need an if elif statement to check for which is coming in?

**shannon.steen**

[6:55 PM](https://dojo.news/coding-dojo/pl/pz75cn313jrnm8ips8akuwwfnh)

yep!

you send that code earlier today. hidden inputs work really well for this kind of thing too

**kent\_h**

[6:56 PM](https://dojo.news/coding-dojo/pl/hbgptonzwjbkdg7xipbty3397o)

I will see, but when that route is entered, it will not have been sent by the html form submit button, but by the other route...I think

**shannon.steen**

[6:57 PM](https://dojo.news/coding-dojo/pl/q39dem6ukjfgtkym3d76qpd9rr)

ideally, I think in this assignment specifically, you just use the '/' to display the html page and '/process\_money' to perform yoru logic

after logic done, you can redirect

And yes, CSS works great when your elements are encapsulated in the proper divs

=====end shanon on .py layout