Individual write-up (aka “take-home quiz”)

**Instructions:**

* Answer as many of the following questions as you can. The final grade of this coursework may be scaled.
* **Please include a reference list at the end for all the materials you have used to finish this write-up. Also, please be warned about the consequences of violating** [**course policies**](http://www.cs.sfu.ca/CourseCentral/470/lisat/policies.html)**.**
* Submit your answers on [CourSys](https://coursys.sfu.ca/2018sp-cmpt-470-e1/+writeup/) by April 12th 11:59pm.

**Marks will be deducted for reasons including but not limited to the following:**

* Answers are written in incomplete sentences and/or difficult to understand.
* Your submission is highly similar to those submitted by your teammates.
* Citations to copied materials are missing and/or not properly done.

**Part 1. Technical review for final project (worth 2%)**

1) Return to your team’s Git repository for the final project. On a Terminal, copy the output of command below into the first column:

**$tree -L 3**

On the copied directory tree, identify all the item(s) that **you were** **actively** involved with by highlighting them in red (example on last page) and explain the ***role*** of each item with 1-2 complete sentences.

|  |  |
| --- | --- |
| Copy of the output:  .  ├── ReadMe  ├── Vagrantfile  ├── chef  │   └── cookbooks  │   └── baseconfig  ├── logo.png  ├── mysite  │   ├── db.sqlite3  │   ├── manage.py  │   ├── mysite  │   │   ├── \_\_init\_\_.py  │   │   ├── \_\_pycache\_\_  │   │   ├── settings.py  │   │   ├── urls.py  │   │   └── wsgi.py  │   ├── polls  │   │   ├── \_\_init\_\_.py  │   │   ├── \_\_pycache\_\_  │   │   ├── admin.py  │   │   ├── apps.py  │   │   ├── forms.py  │   │   ├── migrations  │   │   ├── models.py  │   │   ├── static  │   │   ├── templates  │   │   ├── tests.py  │   │   ├── urls.py  │   │   └── views.py  │   └── static  │   ├── admin  │   ├── css  │   ├── dashboard  │   ├── img  │   ├── js  │   ├── polls  │   └── vendor  └── survey\_app  ├── Template1.html  ├── Template2.html  ├── Template3.html  ├── create\ a\ survey.html  ├── css  │   ├── customize.css  │   └── letterA.css  ├── customize.html  ├── dashboard  │   ├── css  │   ├── dashboard.html  │   ├── js  │   └── less  ├── img  │   ├── 001.jpg  │   ├── 002.jpg  │   ├── 003.jpg  │   ├── 004.jpg  │   ├── 005.jpg  │   ├── add.png  │   ├── bg-survey.jpg  │   ├── login-bg.jpg  │   ├── logo.png  │   └── title.jpg  ├── index.html  ├── js  │   └── template.js  ├── multiple.html  ├── saq.html  └── vendor  ├── bootstrap  ├── bower  ├── font-awesome  └── jquery | Your explanation of each file:    Vagrant File setup  Project implement folder  Database  Django manage.py  Project name foler  Python cache and init.py  Django project setting file  Project url (controller)  Wsgi with Django  App polls in project  Python cache and init.py  Admin.py for admin register in Django  Forms.py for from create in Django  Database migrations  Models create and datatype is in models.py  Static files Django need use and connect in framework  Templates folder in Django project  Test.py for unit test in Django  url as views in polls app will be include in my\_site folder  controller in Django, will send views to url.py  Static files |

2) From the list below, select **four** terms and **explain in your own words** how each was exemplified or implemented in your **final group project**. When you explain, use the following steps:

1. State first in 1-2 sentences what you think the term means (again, write in your words).
2. Next, provide code snippets of **two different examples** by copy-paste from your project repository.
3. Explain in 2-3 sentences *how* the example illustrates the concept.
4. For one of the two examples you provided in step 2, explain alternative implementation strategies that you may know of using at least 3 sentences (no more than 6). Provide code to supplement your explanation if you like.
5. State which of the strategies (the final implementation adopted or the alternative one you have proposed) is the best **and explain why**.

* Data binding
* Inversion of control
* Dependency injection
* Web sockets
* Template rendering
* Application-level caching
* Server-caching
* Routing
* Digest authentication or Open authentication

1. inversion of control

If Class B is under Class A, then the inversion of control means a class A object “a” kind of be foreigner key of class B object “b”. For example, Class Survey as A and Class Question as B. when the question it changed, survey’s questions will also be change

curr\_survey = Survey()

curr\_survey.survey\_text = str(request.user)

curr\_survey.save()

for data in datas:

res\_dict[pos] = data

curr\_question\_type = data['type']

curr\_question = Question()

curr\_question.survey=curr\_survey

2. Template rendering

Template rendering as back-end render data to front-end in a web application, each time we refresh the web page it will request data source from back-end and load in to front-end page.

{% for id,val in data\_dict.items %}

    <div>

        <h3>{{ id }}. {{ val.question }}</h3>

The Django template HTML page will read data from back-end request, and shows it in browser.

3. Routing

Routing means url changed by render/ redirect the original pages with request, new view and data

def try2(request):

return render(request, 'tryplates/template2.html')

def try3(request):

return render(request, 'tryplates/template3.html')

4. Open authentication

User register info is not in 3rd-party database, only in web’s server side

path('login/', auth\_views.login, {'template\_name' : 'registration/sign.html'},name='login'),

path('logout/',auth\_views.logout, {'next\_page': '/'} , name='logout'),

3) Describe **two** RESTful web service that your final project uses. Provide examples of the corresponding JSON response(s) and explain your understanding how the whole process works.

4) Comment on the scalability of your final project due to the design choices made. (Hint: refer to your TE notes)

Our Project’s scalability is pretty big, since we use python as main language, it’s easy to connect with python data analysis tools.

5) **[Up to 2 bonus marks]** Review the comments from all the [SUMMARY FORMS](https://drive.google.com/drive/folders/1-PT8bXcRgRiZ-jbnh49YUhrQT1sLz00A) your team has received and provide your response to each (defence and/or agree, and most importantly, explain why).

I agree with the point that our front-end tools is too essential for web-development. Because I know bootstrap and JQuery are basic front-end development framework. Also our survey function is too heavy and hard to implement at all. And thanks for everyone mentioned there is a good advantage of Django is it’s Python based framework, easy to connect with data analysis python framework.

**Please continue on next page.**

**Part 2. Technical review outside of project (worth 3%)**

1) Return to your submitted E3 and E4 files. How would you revise these past submissions so you could deploy the same contact form you wrote for E3 via Vagrant (keep the web stack the same as E4)? And how would you execute the database query in the form of a script to be executed within the guest machine at time of provisioning.

To answer Q1, complete your example so that the marker could do the following steps to get your web page deployed on any machine:

$ git clone git@csil-git1.cs.surrey.sfu.ca:your\_userid/writeup\_2.1.git

$ vagrant up

2) Create a new repository that illustrates how you would deploy a simple application with a LAMP web stack. Your application can be of any content and any nature but must be written entirely by you.

Like Q1, answer Q2 so that the marker could replicate the deployment on his/her own machine via:

$ git clone git@csil-git1.cs.surrey.sfu.ca:your\_userid/writeup\_2.2.git

$ vagrant up

**At submission time, please make sure you directory trees for Q1-Q2 does not contain .vagrant/; do so by removing instances like this:**

**rm -fR writeup\_2.1/.vagrant/**

3) Using Javascript and the following array variable

var countries = [

 { code: "AR", name: "Argentina", capital: "Buenos Aires" },

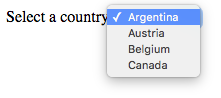
 { code: "AT", name: "Austria", capital: "Vienna" },

 { code: "BE", name: "Belgium", capital: "Brussels" },

 { code: "CA", name: "Canada", capital: "Ottawa" }

];

write an HTML page containing only JavaScript (no jQuery) that would use a loop to generate a drop-down list that would display the country names. The value attribute for each item in the list should be the country code. Add an event handler for the drop-down list’s change event. This handler should display the capital city of the currently selected country within the <div> element that is after the <script> element. Hint: Your event handler will need to use the value property of the drop-down list, the innerText or innerHTML property of the div, as well as another loop.



4) After some research, describe 5 security practices that one should always implement in any web sites.

**Please continue on next page.**

5) Read up on Search Engine Optimization (SEO) and answer the following questions.

i) Putting keywords into content is known as a “white hat” SEO technique. Explain how you see this strategy can lead to higher search ranking to your web app?

Legal and followed ethic search engine optimization, more detail in homepage and pay to google search.

ii) Explain your understanding on difference between “white hat” and “black hat” SEO techniques.

Black hat is kind of like use illegal and unethical method to optimize search engine. White hat represents legal optimize way.

iii) Below are word counts of all nouns in the content of 3 web pages:

|  |  |  |
| --- | --- | --- |
| Page1.html | Page2.html | Page3.html |
| Cat: 6  Cow: 6  Dog: 6  Home: 0  Mouse: 6  Pig: 0 | Cat: 2  Cow: 2  Dog: 2  Home: 0  Mouse: 0  Pig: 0 | Cat: 0  Cow: 1  Dog: 0  Home: 1  Mouse: 0  Pig: 1 |

       Why a typical search engine would return Page2.html for a search query of “cats and dogs”?

       Explain in your own words along with a mathematically based justification.

Cat and dog in Page1.html have 12/32 = 37.5%, in Page2.html there is 4/6 = 67%, and in Page3.html, there is only 0/3 = 0%. So Page2.html have higher rank in those 3 webpages