**Problem statement :**

Make a program to draw random numbers from strip containing unique integers ,

conditions

first drawn must be whole number

scond drawn must be a factor of first drawn

third drawn must be product of first two

4th one any

5th one must be smallest among all

6th one any number

using above number

make a 2 d 6\*3(6 col and 3 rows) grid from strips of all alphabets

1st strip English upper case

2nd strip English lower case

3rd Greek alphabets (24)

4th any random combination

5th roman any (number or letter)

6th roman any (number or letter)

note

each first symbol will pay any 6 of a kind as 2 and followed up symbols pays twice the previous one  
Also store each spin data in the database and show the grid set on the UI by using any tool , library.   
Make full documentation for the code and for the game.

**6x3 grid game**

**Back-end**

A. First we made function **draw\_random\_number** on the basis of the follow conditons:

1. First drawn number: whole number (between 0 to 25)
2. Second drawn number:factor of the first drawn number
3. Third drawn number:multiple of first and second drawn numbers
4. Forth drawn number:any number
5. Fifth drawn number :smallest among all the numbers that are in the random\_list
6. Sixth number :any number

B. To generate the strip (vertical to horizontal): We made **generate\_strip** function on conditions:

We do this for three cells

1. First Strip for only uppercase english alphabets : choose on the basis of the random\_list’s first number for each cell in this strip
2. Second Strip for only lowercase english alphabets : choose on the basis of the random\_list’s second number for each cell in this strip
3. Thrid Strip for only Greek letters : choose on the basis of the random\_list’s thrid number for each cell in this strip between 24 only.
4. Forth Strip combination of english alphabets : here we will not choose on the basis of the random\_list’s number as the combination has no specific combination , so we choose randomly by random.sample(it is defined in libraries section)
5. Fifth Strip for roman number : choose on the basis of the random\_list’s fifth number for each cell in this strip
6. Sixth Strip for roman number : choose on the basis of the random\_list’s sixth number for each cell in this strip

C. Function generate\_strip\_combination to take grid list and generate final\_grid single list containing all the symbols , alphabets in it that we got from the **generate\_strip** function

D. Get desired grid: **create\_grid\_from\_combination** function to select the randomly three elements from the generate\_strip\_combination function for each strip having 3 symbols and then grid\_conversion\_to\_specific\_format : to get the grid in desired format of 6 cross 3 (6 columns and 3 rows)

E. Calculate winnings : We made function **calculate\_winnings(grid)** , grid : parameter got from **create\_grid\_from\_combination** function (3x6)  
Gives winning on the set of rules

1. Same symbol to be come minimum 6 times horizontally in all 6 columns (if same symbol come in row we will not count )  
   a. If same symbol come 6 times simultaneously and adjacently then there will be one win   
   b. If above condition is true and if the same symbol is coming more than one time in the same column then we will multiply the above winning with number of time the symbol come. We used *multiplier* to store the number of times the same symbol come.

Example 1 :  
No win condition

1 2 3 4 5 6 : strips  
['O', 'R', 'I', 'u', 'O', 'XV']

['ι', 'O', 'T', 'k', 'O', 'XII']

['G', 'Z', 'O', 'I', 'b', 'N']  
The above grid will not generate the win as the symbol O is not coming 6 times total it is coming 5 times. As O is missing in the sixth strip

Example 2:  
1 2 3 4 5 6 : strips  
['O', 'R', 'I', 'u', 'O', 'XV']

['ι', 'O', 'T', 'k', 'O', 'XII']

['O', 'Z', 'O', 'I', 'b', 'N']  
Here the symbol O is coming 6times but it’s not coming simultaneously and also not in all strips (missing O in 6th strip)

Example 3:  
1 2 3 4 5 6 : strips  
['O', 'R', 'I', 'u', 'O', 'XV']

['ι', 'O', 'T', 'k', 'O', 'XII']

['O', 'Z', 'O', 'I', 'b', 'O']  
Here the symbol O is coming 7 times and simultaneously and also in all strips so base win will be 2 and the symbol O is coming two times in strip 1 , so we will multiply the base win by 2 so final win will be 2 \*2 =4

1. Win for first symbol is 2 and for follow up symbols it will be twice the prevoius one  
   For this grid

1 2 3 4 5 6 : strips  
['O', 'R', 'I', 'u', 'O', 'XV']

['t', 'O', 'T', 'k', 'O', 'XII']  
['p', 'Z', 'O', 'I', 'b', 'N']  
If symbol O gives winning then the win will be 2 and multiple of 2 as it is first symbol  
If symbol t gives winning then the win will be 2 times of previous one that is 2\*2=4 and multiple of 4 as it is second symbol  
If symbol p gives winning then the win will be 2 times of the previous one that is 4\*2 =8 and multiple of 8 as it is thrid symbol

1. If there is win for more than one symbol then all wins will be added together and then generate the final win.

F. **start\_game** : to finally get execute the whole code   
here we will pass **final** to the front-end where final =[final\_grid,winnings]

After that we store all numbers in the list randomList   
Here we store in list to easily access and list is mutable(can be edited)

This contains the 6 numbers based on the condition:

1st number: It should be whole number

2nd number: must be a factor of first drawn

3rd number: must be product of first two

**Front-end & Database**

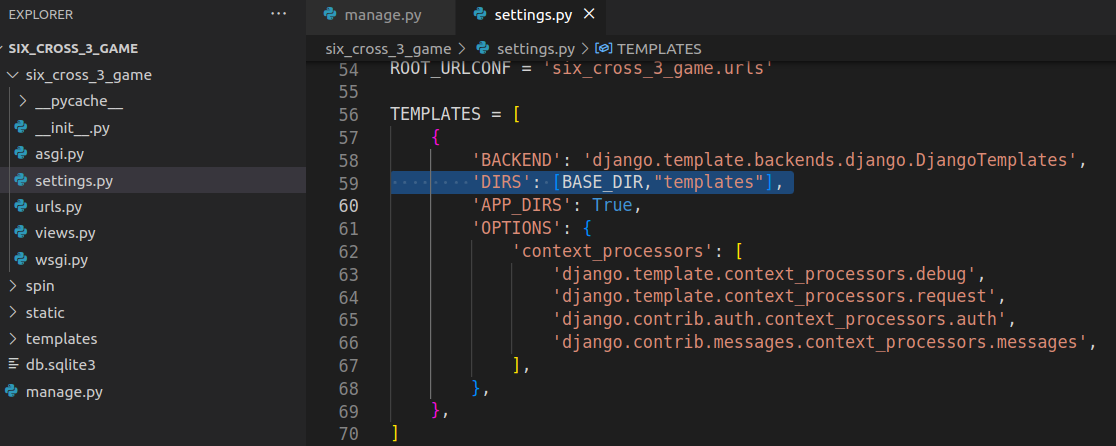
For showing our grid in the front end we used the Django as it is the full stack web framework which includes database in it by default. Because we will need database to store the spins data (more infomation is exaplined in database section).  
Django is a free and open-source, Python-based web framework that follows the model–template–views architectural pattern.

Steps:

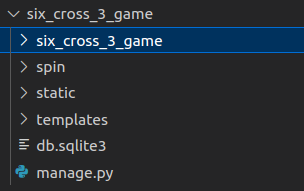
1.Install django: pip install django

2.Make django project: django-admin startproject six\_cross\_3\_game

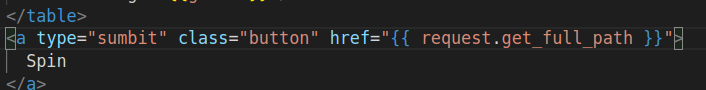
3.Make **index.html** in the **templates** folder: all html files if there is not template folder make one and add [BASE\_DIR ,”templates”] to DIRS , DIRS includes directories whre django will search for templates(html files ) , BASE\_DIR contains project path and then we include templates path in it.



4.Make static folder in the game directory. Static folder contains our game logic , css files , js files, images.

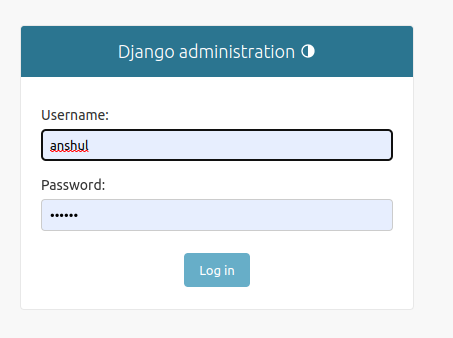
5. In index.html make front end code to be shown in UI:

* Here we use style tag for styling css , we can also use inline css but it will be quite cumbersome and not a good practice.
* We get data from backend and show it in table format dynamically , with the help of jinja template we can execute python in the index.html like Tags are surrounded by {% and %} like this:
* We will render each table row <tr> and one table row has six table data tags <td> and in each <td> we show each data element getting from backend  
  Data coming from backend python code is :   
  [   
   [  
   ['V', 'χ', 'I', 'F', 'j', 'CMIII'],   
   ['g', 'CMIII', 'K', 'i', 'n', 'K'],   
   ['n', 'θ', 'ζ', 'I', 'CMLXI', 'D']  
   ],  
  0]
* For showing **win** in the UI getting data between <h2> tag <h2>Winnings: {{grid.1}}</h2>

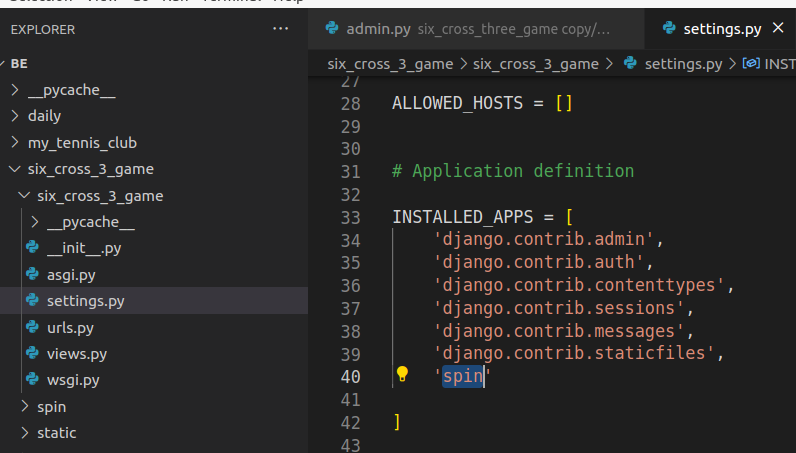
* Button to spin (to show next grid on the frontend) , when user click on the Spin page will refresh and we get new response from the backend (so to get more random grid). href: getting path to the current page and when user click page will go to the href url which is same to the current page for this project.

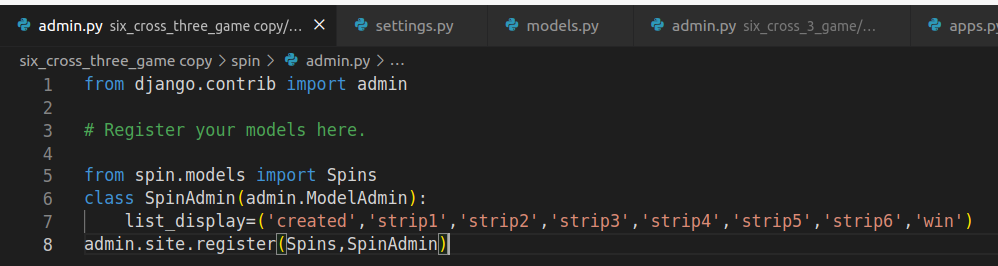
6. We need to store our spins data in the database , we have sqlite3 database by default in django.  
A standalone command-line shell program called sqlite3 is provided in SQLite's distribution. It can be used to create a database, define tables, insert and change rows, run queries and manage an SQLite database file

7. Super User : For creating superuser, first reach the same directory as that of manage.py and run the following command: python manage.py createsuperuser

* Then enter the Username of your choice and press enter.  
  Username: anshul (or any other)
* Then enter the Email address and press enter.(It can be left blank)  
  Email address: example@gmail.com
* Next, enter the Password in-front of the Password field and press enter.Enter a strong password so as to keep it secure.
* Password: \*\*\*\*\*\*
* Then again enter the same Password for confirmation.
* Password(again): \*\*\*\*\*\*
* Superuser created successfully if above fields are entered correctly.
* Now we can login into our Django Admin page by running the command python manage.py runserver . Then, open a Web browser and go to “/admin/” on your local domain – e.g., http://127.0.0.1:8000/admin/ and then enter the same Username and Password.

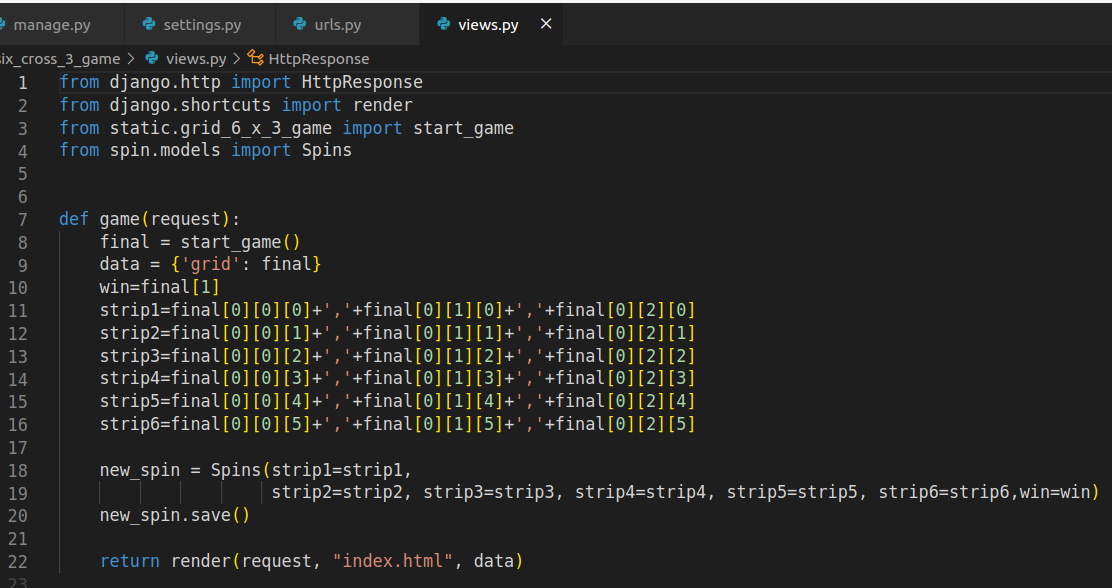
8. Make new app **spin** in the six\_cross\_3\_game in six\_cross\_3\_game directory containing manage.py and from there enter the command : python manage.py startapp **spin** and add spin to the installed apps (as shown in below image)

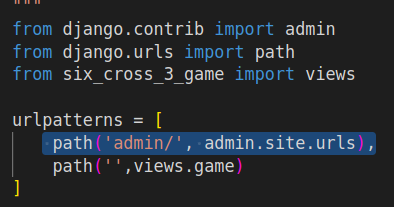
* We store date and time , strip1 , strip2 , strip3 , strip4 , strip5 , strip6, win data in the database.
* First make model in models.py to make our new table in the database
* Under models.py models is imported default  
   we create class Spins taking models as argument and creating table columns as  
  created: current date and time of the spin  
  strip1 to strip6 : each strip contains 3 symbols and it is in alphabet so we use CharField and we need to specify the max\_length in it to 100, because the length of the alphabet we get will not exceeds 100.  
  win: contains win in numbers so we used IntegerField here.
* under admin.py / register out model here

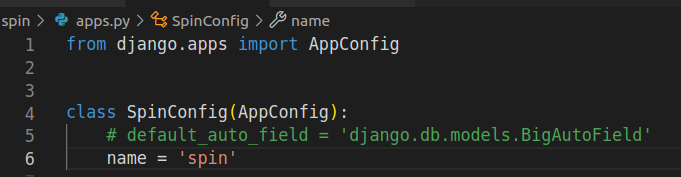


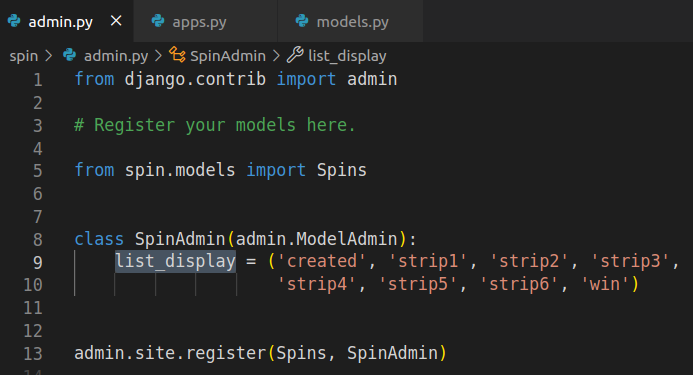
9. In views.py we make our new function **game** in this we get data from python code (under static/grid\_6\_x\_3\_game.py here our main code logic is written)

* from django.http import HttpResponse: Importing httpresponse :HttpResponse (source code) provides an inbound HTTP request to a Django web application with a text response. This class is most frequently used as a return object from a Django view.
* from django.shortcuts import render :In Django, render() is one of the most used functions that combines a template with a context dictionary and returns an HttpResponse object with the rendered text.
* from static.grid\_6\_x\_3\_game import start\_game : importing our **start\_game** function to start the game.
* Call our start\_game function and pass it to the data dictionary as value of ‘grid’
* Render a HTML Template (index.html )as Response: In return we render out request and index.html file with data in which we are getting the grid data.
* And every time we are getting the data from logic we store it in the database by creating the strip1 , strip2 , strip3 , strip4 , strip5 , strip6, win
* We called Spin from the models and pas the new value getting from the final and then save by using save() in line numnber 20.



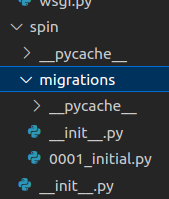
10. Urls.py : path to **urls.py** /six\_cross\_3\_game/six\_cross\_3\_game/urls.py under urls.py we want our game to come in front end screen at the homepage of the app <http://127.0.0.1:8000/> so we add path(‘’,views.game) by importing views 

11. Under /six\_cross\_3\_game/spin/apps.py we add our app by **apps.py** convention merely allows Django to load spin app automatically when INSTALLED\_APPS contains the path to an application module rather than the path to a configuration class.

12. To register our model to show in the database at <http://127.0.0.1:8000/admin/> under **admin.py** we made new class by importing Spins from models.py and in list\_display to control which fields are displayed on the change list page of the admin when our spins data is change.

13. Now after making any change in any of app’s models file we need to run following command python3 manage.py makemigrations

After this command run following command to finally implement database changes accordingly  
python3 manage.py migrate

After running makemigrations and migrate a new table would have been created in database. We can check it from spin -> migrations -> 0001\_initial.py.

**Libraries and Tools**

For Back-end

1. Random module: Random module is an in-built module of Python that is used to generate random numbers in Python. This module can be used to perform random actions such as generating random numbers, printing random a value for a list or string, etc.

* Random.randint() : Returns a random integer within the range
* Random.choice() :Returns a random item from a list, tuple, or string
* Random.sample() :Returns a particular length list of items chosen from the sequence

2. String module : Python String module contains some constants, utility function, and classes for string manipulation. It’s a built-in module and we have to import it before using any of its constants and classes.

* ascii\_uppercase is a pre-initialized string used as string constant. In Python, string ascii\_uppercase will give the uppercase letters ‘ABCDEFGHIJKLMNOPQRSTUVWXYZ’.
* ascii\_lowercase is a pre-initialized string used as string constant. In Python, string ascii\_lowercase will give the lowercase letters ‘abcdefghijklmnopqrstuvwxyz’.
* ascii\_letters is a pre-initialized string used as string constant. ascii\_letters is basically concatenation of ascii\_lowercase and ascii\_uppercase string constants.