Ccube

CODE CONTEST CALENDAR

R Ruhi Taj (15CO239)

Y M Greeshma (15CO253)

B Vasudha (15CO211)

CODE CONTEST CALENDAR

INTRODUCTION

AIM:

In today's growing coding cultures, coding sites have really taken over gaming sites. Each coding competition organized by each website has its own unique mark. Considering the huge number of websites; it has become really hard for the coders to keep track of the various contests and the contests which are apt to their level of skills in coding. In order to overcome the current problem, the idea of code contest calendar is strong.

In this project, the data of the coding competitions is collected from all the websites and stored in SQL database for proper management and retrieval. Also, the competitions will be sorted according to their dates like a calendar and the user can view present, past and future contests with ease. Each coding website will be categorized and stored in the SQL database. Contests along with their dates will be stored and organized. Students can easily see the available contests. Web crawler techniques like beautiful soup will also be implemented.

RELEVEANCE:

The project is expected to have a calendar with all the dates with navigable functionalities. The selection of a particular would be able to open a new page, which has all the competitions on that particular date. Any user would be able to access the data. The website has the list of all the important list of competitions, list of famous websites and the list of important data required to learn about all the coding competitions.

All the functions inside the website have so far been successful. All the requirements are achieved. The look and feel of the project has also come out to be good. The server has a very fast loading speed in the internet. The idea of rating each website is not implemented as the frequency in which the competitions are organised does not rate the quality of the website and rather by the users of the websites, which requires permission from the administration of the website.

DESCRIPTION:

The project is used to navigate all the competitions going across the world, on a particular date as selected in the calendar. The user only has to select the particular date on the calendar. The website has its speciality in teaching its user with all the best information available across the internet. Web crawling and query are the major implementations of the project. The special feature of the website is entering the date in the web link and obtaining the competitive coding links and competitions on that date.

CODE CONTEST CALENDAR

REQUIREMENT ANALYSIS

The data base and websites

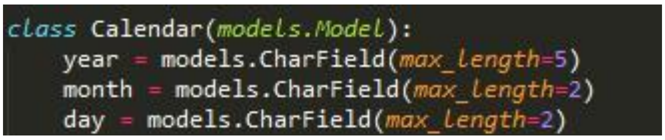
Technical requirements:

Languages:	Python, Java Script, HTML, CSS, MySQL,AJAX
Frame Work:	Django, Git
Techniques:	Beautiful soup of python for Web Crawling

Data Base Requirements:

The Calendar:

The data required for the calendar are the dates in a calendar which are entered into the database, by adding rows into the table **Calendar** by writing the code starting from 1995 to 2020, by a python code. Hence Python, HTML, MySQL are the languages that were used to link the python code (used to write the code for calendar) to MySQL data base and then transferring the data into HTML pages. **Calendar** is a table with the following attributes:



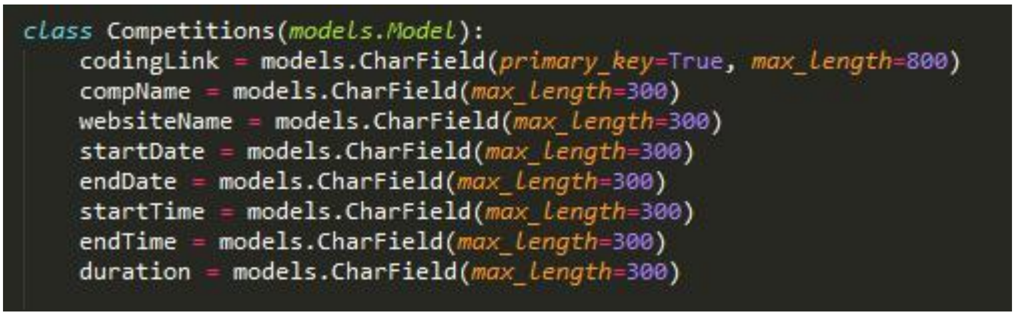
```
class Calendar(models.Model):
    year = models.CharField(max_length=5)
    month = models.CharField(max_length=2)
    day = models.CharField(max_length=2)
```

DEFINITION:

The primary key combines all the three attributes-year, month and day which produce a new attribute called Date.

Competitions:

The data required for the competitions are different coding competitions around the world which were collected from around the world from the websites- Code chef, Spoj, Leet Code, Hacker Rank, Top Coder, and Code Forces. Python and MySQL are the languages that were used for the competitions. **Competitions** is a table with the following attributes:



```
class Competitions(models.Model):
    codingLink = models.CharField(primary_key=True, max_length=800)
    compName = models.CharField(max_length=300)
    websiteName = models.CharField(max_length=300)
    startDate = models.CharField(max_length=300)
    endDate = models.CharField(max_length=300)
    startTime = models.CharField(max_length=300)
    endTime = models.CharField(max_length=300)
    duration = models.CharField(max_length=300)
```

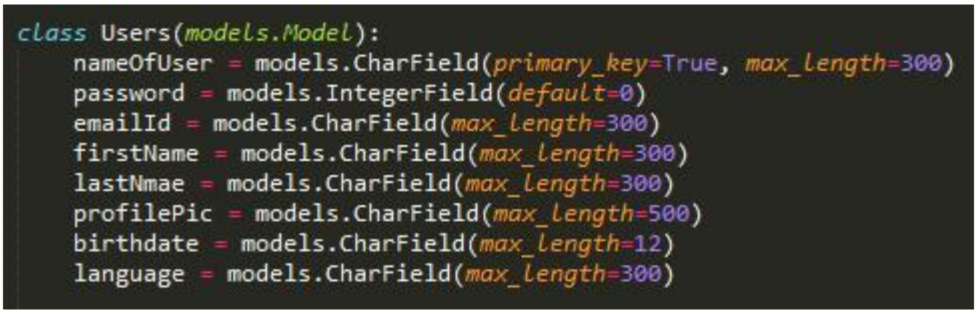
DEFINITION:

The primary key is the coding link, which is unique by default.

CODE CONTEST CALENDAR

Users:

In order to login into the website, the user data is collected. At present, the page is open to all the users, but it is planned to allow customised pages for the users. **Users** is the table with the following attributes:



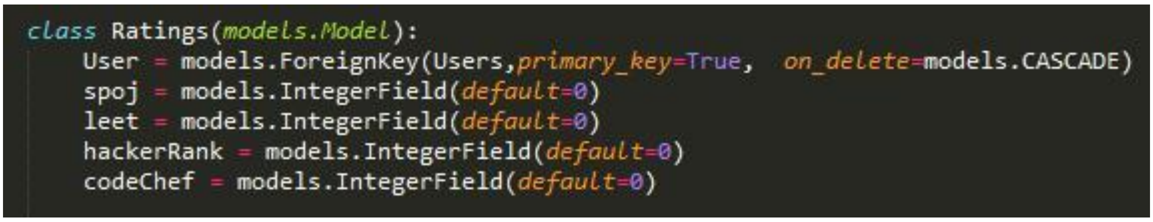
```
class Users(models.Model):
    nameOfUser = models.CharField(primary_key=True, max_length=300)
    password = models.IntegerField(default=0)
    emailId = models.CharField(max_length=300)
    firstName = models.CharField(max_length=300)
    lastNmae = models.CharField(max_length=300)
    profilePic = models.CharField(max_length=500)
    birthdate = models.CharField(max_length=12)
    language = models.CharField(max_length=300)
```

DEFINTION:

The name of the user is the primary key of Users, and hence each user has a unique User Name.

Ratings:

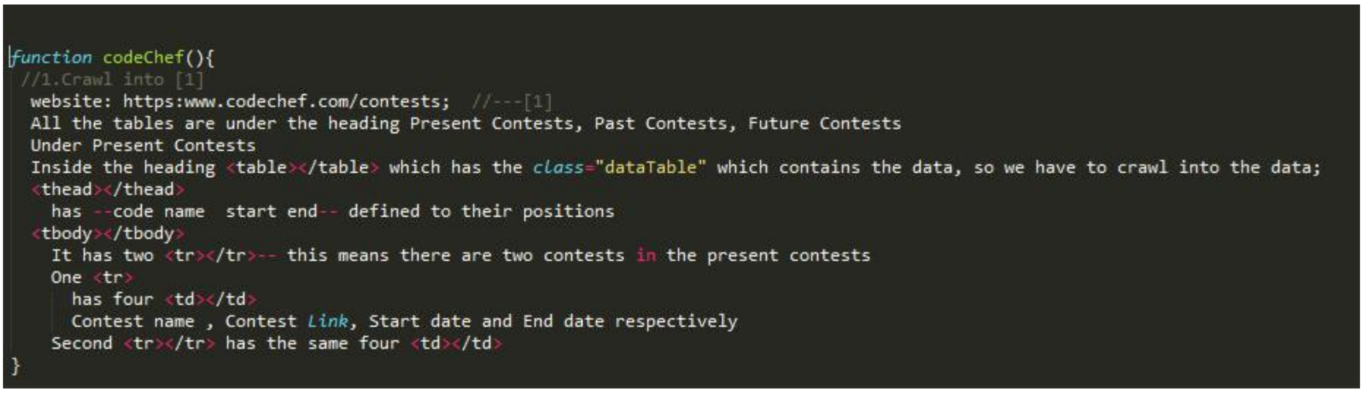
Ratings of each user given to the website are taken from each user of the website in order to customise the page to the user. The ratings table is dependent on the users table. The **users** table has the following attributes:



```
class Ratings(models.Model):
    User = models.ForeignKey(Users,primary_key=True,  on_delete=models.CASCADE)
    spoj = models.IntegerField(default=0)
    leet = models.IntegerField(default=0)
    hackerRank = models.IntegerField(default=0)
    codeChef = models.IntegerField(default=0)
```

DEFINTION:
The user is the primary key of the ratings.

Data Base for Competitions Table:



```
function codeChef(){
 //1.Crawl into [1]
  website: https:www.codechef.com/contests;  //---[1]
  All the tables are under the heading Present Contests, Past Contests, Future Contests
  Under Present Contests
  Inside the heading <table></table> which has the class="dataTable" which contains the data, so we have to crawl into the data;
  <thead></thead>
    has --code name  start end-- defined to their positions
  <tbody></tbody>
    It has two <tr></tr>-- this means there are two contests in the present contests
    One <tr>
      has four <td></td>
      Contest name , Contest Link, Start date and End date respectively
    Second <tr></tr> has the same four <td></td>
}
```

The above described method was used to crawl the website-CodeChef by using beautiful soup in python and similar method was repeated for other websites to obtain data base.

Literature Survey:

Although the idea is a completely new one, the inspiration to the idea comes from a company that collects the data of a particular product from all the other online shopping companies and makes the process of online shopping easy and cheap.

As we have used beautiful soup for the crawling i.e extra ting the data from the HTML and the XML webpages ,we have been guided from the different documentations about the beautiful soup , one among them which is most useful to us is from the link given below: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Django which is a webframe work is studied from the documentation https://media.readthedocs.org/pdf/django/latest/django.pdf

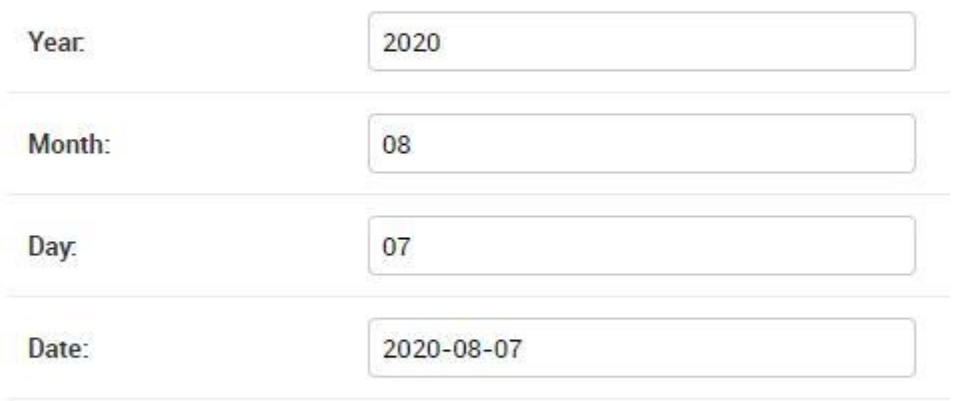
Design*:*

The idea in itself is an entirely new one. So, we build the crawler, front end, back end and website structures from scratch. The website would contain a calendar decorated by other links and methods supporting competitive coding environment.

A toggle button allows the user to change a setting between two states. i.e displaying the previous month’s date , next month’s date from the present months date. s

CODE CONTEST CALENDAR

CALENDAR's row in the database:



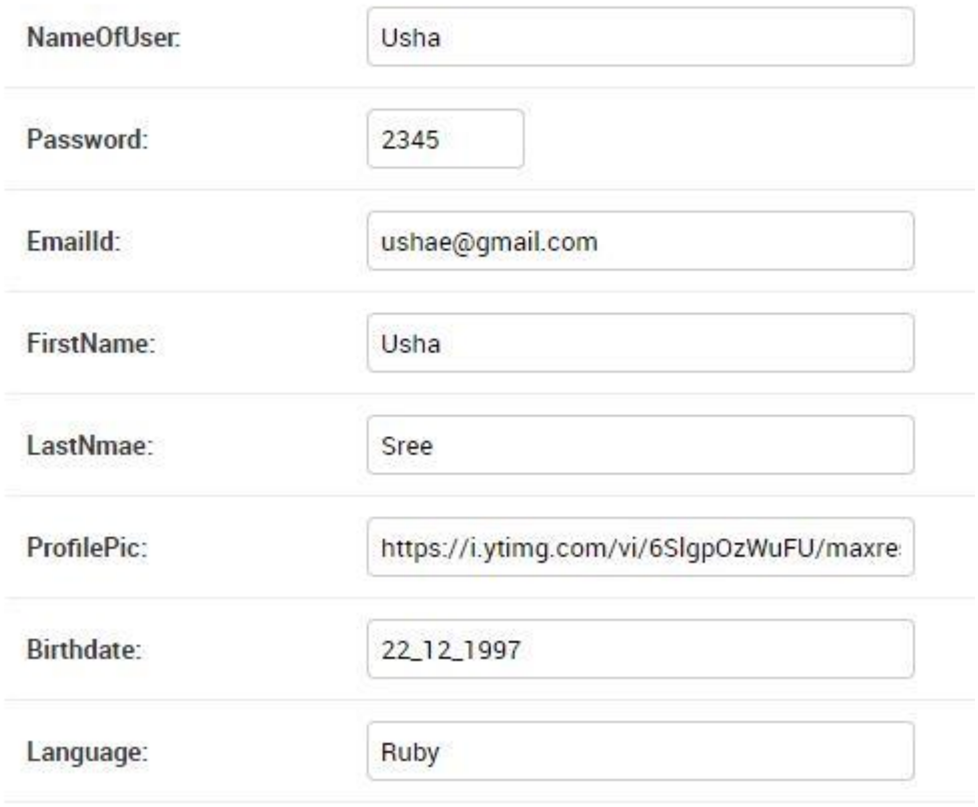
Year:	2020
Month:	08
Day:	07
Date:	2020-08-07

STORAGE:

This is the snippet of the way the code is stored in the database.

The **Date** is the primary key of the Calendar.

USER's row in the database:



NameOfUser:	Usha
Password:	2345
EmailId:	ushae@gmail.com
FirstName:	Usha
LastNmae:	Sree
ProfilePic:	https://i.ytimg.com/vi/6SlgpOzWuFU/maxre
Birthdate:	22_12_1997
Language:	Ruby

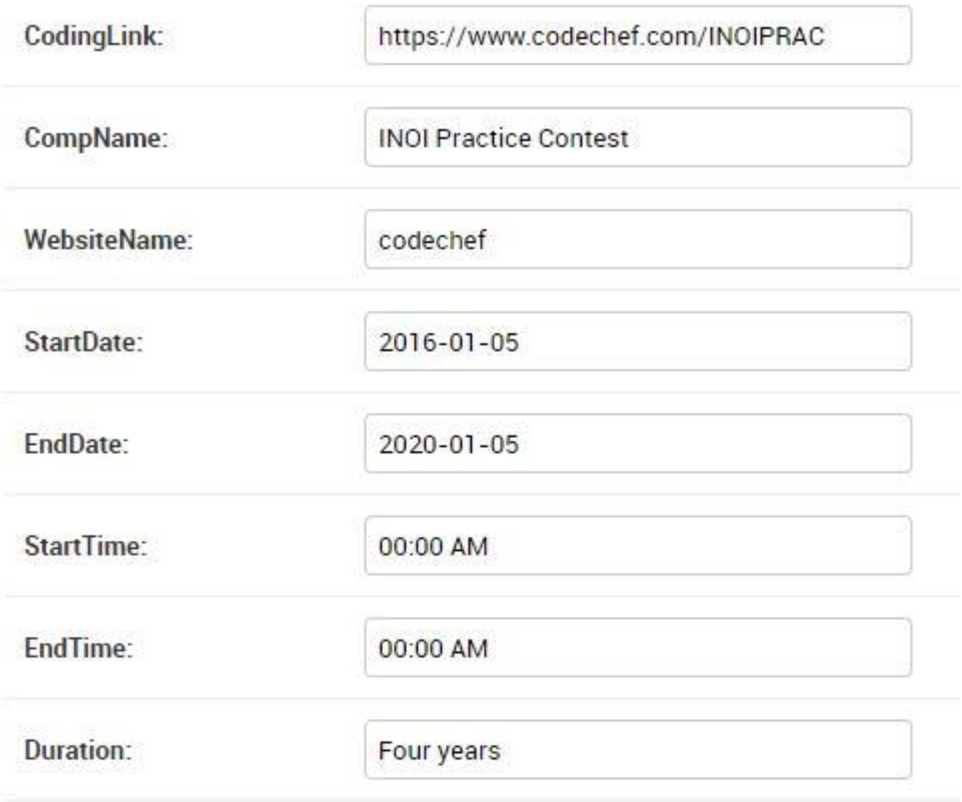
STORAGE:

This is the snippet of the way the user data is stored in the database.

The **NameofUser** is the primary key of the Calendar.

CODE CONTEST CALENDAR

COMPETITION’s row in the database:



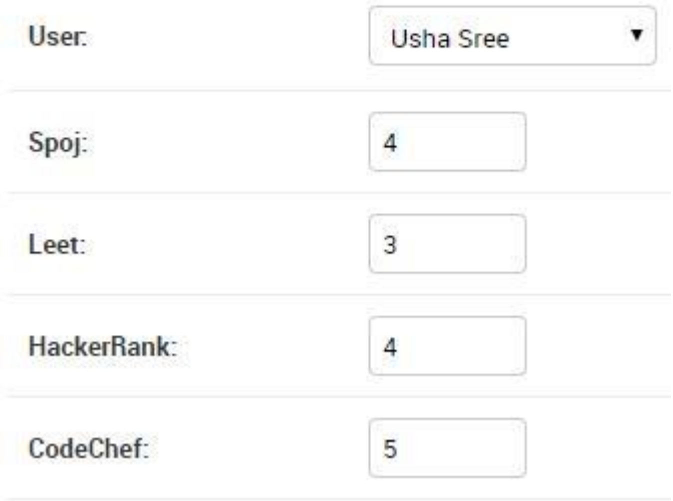
CodingLink:	https://www.codechef.com/INOIPRAC
CompName:	INOI Practice Contest
WebsiteName:	codechef
StartDate:	2016-01-05
EndDate:	2020-01-05
StartTime:	00:00 AM
EndTime:	00:00 AM
Duration:	Four years

STORAGE:

This is the snippet of the way the competitions’ data is stored in the database.

The **CodingLink** is the primary key of the Calendar.

RATING’s row in the database:



User:	Usha Sree ▼
Spoj:	4
Leet:	3
HackerRank:	4
CodeChef:	5

STORAGE:

This is the snippet of the way the competitions’ data is stored in the database.

The **User** is the primary key of the Calendar.

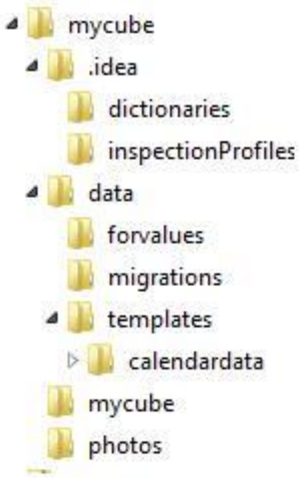
CODE CONTEST CALENDAR

IMPLEMENTATION

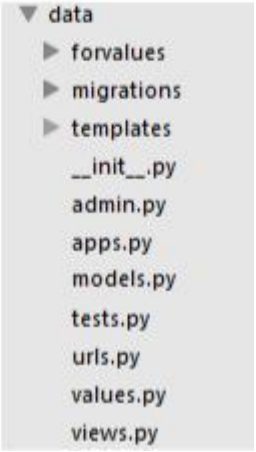
Django is the framework in the entire project was code.

File Structure:

This is the complete file structure of the project-Ccube.



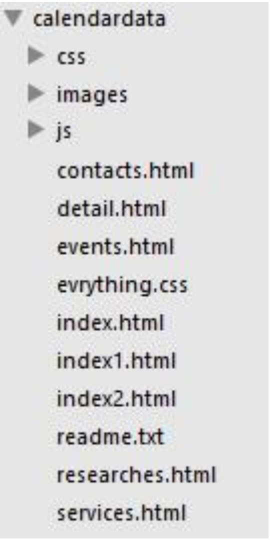
```
mycube
    .idea
        dictionaries
        inspectionProfiles
    data
        forvalues
        migrations
        templates
            calendardata
    mycube
    photos
```



```
data
    forvalues
    migrations
    templates
    __init__.py
    admin.py
    apps.py
    models.py
    tests.py
    urls.py
    values.py
    views.py
```

These are the .py folders which manages the website, since the beginning of the website.

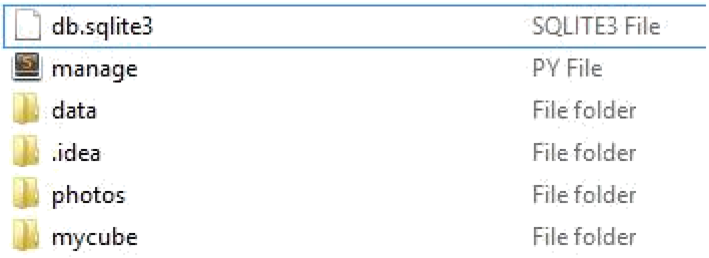
Now inside the directory here: mycube\data\templates\calendardata



```
calendardata
    css
    images
    js
    contacts.html
    detail.html
    events.html
    evrything.css
    index.html
    index1.html
    index2.html
    readme.txt
    researches.html
    services.html
```

The calendardata directory has these sub-directories as shown. The ***index.html*** file has the page which gets called, each time the website is called, index.html is run.

db.sqlite3 is the SQLITE3 File which has all the data and hence is the database for the website.



db.sqlite3	SQLITE3 File
manage	PY File
data	File folder
.idea	File folder
photos	File folder
mycube	File folder

CODE CONTEST CALENDAR

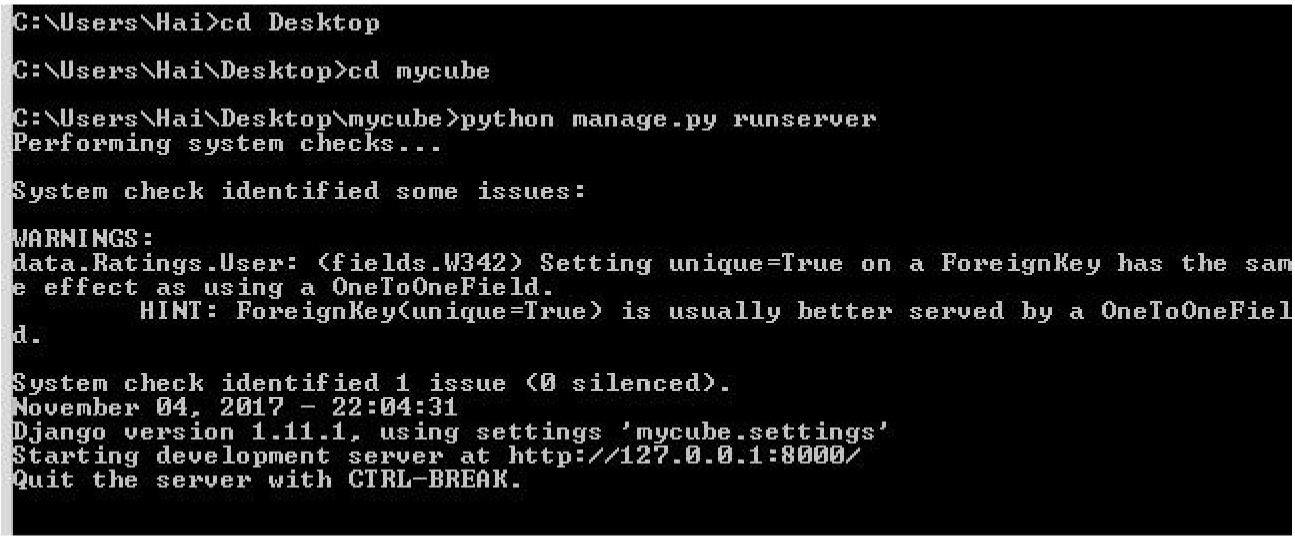
Important Snippets of Code:

→ Running the server:

After entering the folder containing all the folders, more precisely the folder containing ***manage.py*** , using the command prompt, the following code is run and the local server is set on the computer and it has the following code:

```
import os
import sys

if __name__ == "__main__":
    os.environ.setdefault("DJANGO_SETTINGS_MODULE", "mycube.settings")
```



```
C:\Users\Hai>cd Desktop

C:\Users\Hai\Desktop>cd mycube

C:\Users\Hai\Desktop\mycube>python manage.py runserver
Performing system checks...

System check identified some issues:

WARNINGS:
data.Ratings.User: (fields.W342) Setting unique=True on a ForeignKey has the sam
e effect as using a OneToOneField.
        HINT: ForeignKey(unique=True) is usually better served by a OneToOneFiel
d.

System check identified 1 issue (0 silenced).
November 04, 2017 - 22:04:31
Django version 1.11.1, using settings 'mycube.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

→ After the server is run , settings.py is called as mentioned above:

Django settings for mycube project.

Generated by 'django-admin startproject' using Django 1.11.1.

For more information on this file, see
https://docs.djangoproject.com/en/1.11/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/1.11/ref/settings/

Quick-start development settings - unsuitable for production

See https://docs.djangoproject.com/en/1.11/howto/deployment/checklist/

CODE CONTEST CALENDAR

→ When the home page is called, this is the snippet which get called and the functions gets directed to index.html, which as explained earlier has

```
def index(request) :
    all_dates = Calendar.objects.all()
    all_users = Users.objects.all()
    all_ratings = Ratings.objects.all()
    all_competitions = Competitions.objects.all()
    template = loader.get_template('calendardata/index.html')
    context_data = {
        'all_dates': all_dates,
        'all_users': all_users,
        'all_ratings': all_ratings,
        'all_competitions': all_competitions
    }
    return HttpResponse(template.render(context_data, request))
```

→ When the date is called with the url of the home page, this snippet of pyton code code gets called:

```
def detail(request, date):
    allcompetitions = Competitions.objects.all()
    time = date[0:4]+'-'+date[4:6]+'-'+date[6:8]
    return render(request, 'calendardata/detail.html' , { 'competitions' : allcompetitions, 'timing'
: time} )
```

→ Code for calendar and its display in **JavaScript,CSS and HTML:**

```
<div id="demo" class="yui3-skin-sam yui3-g"> <!-- You need this skin class -->

  <div id="leftcolumn" class="yui3-u">
     <!-- Container for the calendar -->
     <div id="mycalendar"></div>
  </div>
  <div id="rightcolumn" class="yui3-u">
   <div id="links" style="padding-left:20px;">
      <!-- The buttons are created simply by assigning the correct CSS class -->
      <button id="togglePrevMonth" class="yui3-button">Toggle Previous Month's
Dates</button><br>
      <button id="toggleNextMonth" class="yui3-button">Toggle Next Month's Dates</button><br> <p
      style="color:black;"><br><br> LINK TO Competitions: <a id="myid" href="#"
style="color:#5b0957;" target="_blank"><span id="selecteddate"></span></a></p>  </div>
```

```
  </div>
</div>
<script type="text/javascript">
YUI().use('calendar', 'datatype-date', 'cssbutton', function(Y) {
    var calendar = new Y.Calendar({
      contentBox: "#mycalendar",
      width:'340px',
      showPrevMonth: true,
      showNextMonth: true,
      date: new Date()}).render();

    // Get a reference to Y.DataType.Date
    var dtdate = Y.DataType.Date;

    // Listen to calendar's selectionChange event.
    calendar.on("selectionChange", function (ev)
    { var newDate = ev.newSelection[0];

          var cool = dtdate.format(newDate);
     // window.alert(cool);
     var snig = "http://127.0.0.1:8000/data/";
     for(var i=0;i<cool.length; i++ ){
        if(cool[i]!='-'){
            snig+= cool[i];
        }
     }
      var element=document.getElementById('myid');
    element.setAttribute("href",snig);

      Y.one("#selecteddate").setHTML(dtdate.format(newDate)); });

    Y.one("#togglePrevMonth").on('click', function (ev)
      { ev.preventDefault();
      calendar.set('showPrevMonth',
    !(calendar.get("showPrevMonth"))); });

    Y.one("#toggleNextMonth").on('click', function (ev)
      { ev.preventDefault();
      calendar.set('showNextMonth', !(calendar.get("showNextMonth")));
    });
});
</script>
```

→ Linking database to python to HTML to crawl through the website and get access to the compettitions on that date:

From the above code of ***detail,*** we have seen that timings and competition is passed into the detail.html folder, in the ***detail.html*** folder, this is the code that is important and gets executed to print all the coding dates on that 'timing' passed into the folder.

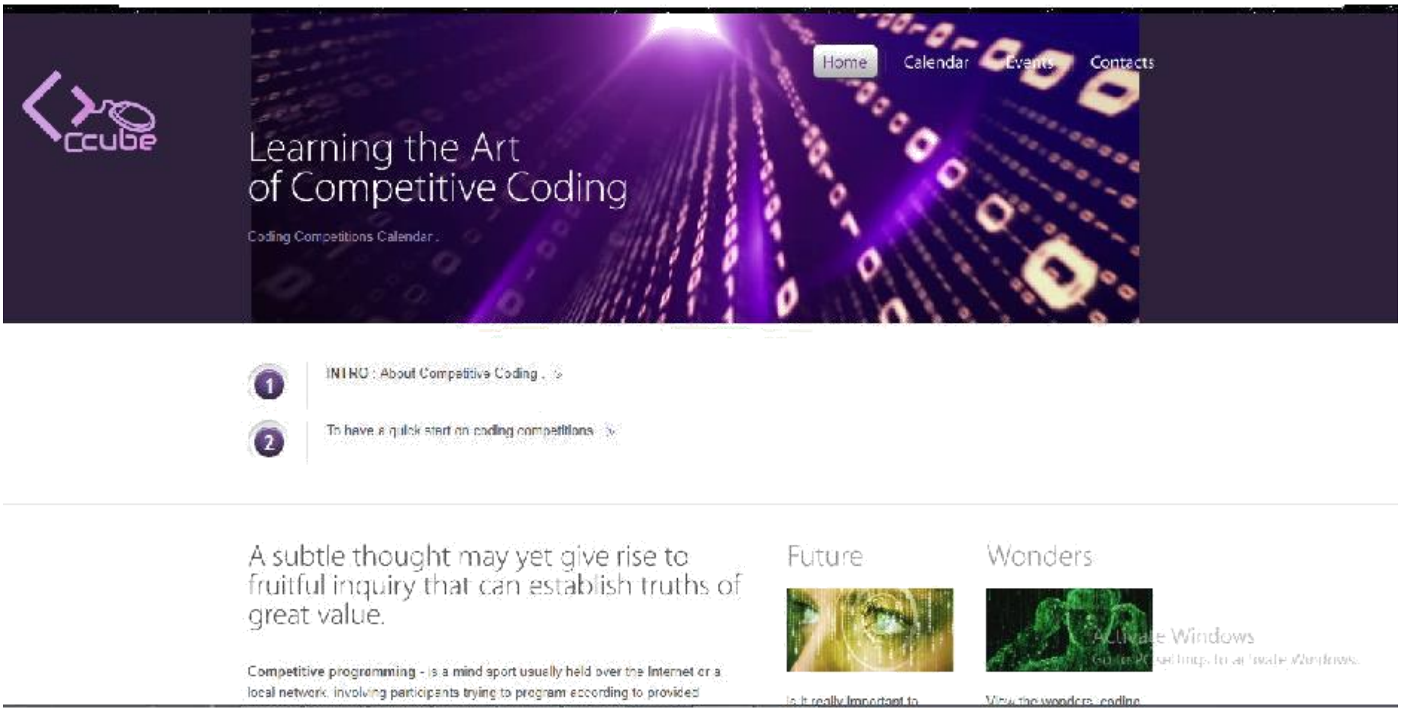
```
<div class="w3-container w3-black w3-padding-64 w3-xxlarge"
 id="menu"> <div class="w3-content">

   <h1 class="w3-center w3-jumbo" style="margin-bottom:64px">COMPETITIONS</h1>
   <div class="w3-row w3-center w3-border w3-border-dark-grey">
    <a href="javascript:void(0)" onclick="openMenu(event, 'Competitions');" id="myLink">
    <div class="w3-col s4 tablink w3-padding-large w3-hover-red">List of Competitions </div>
    </a>
   </div>
   <div id="Competitions" class="w3-container menu w3-padding-32 w3-white">
     {% for competition in competitions %}
       {% if timing >= competition.startDate %}
        {% if timing <= competition.endDate %}
        <h1>
        <b><a href="{{ competition.codingLink }}/" target="_blank">{{ competition.compName
        }}</a></b>
        <span class="w3-right w3-tag w3-dark-grey w3-round">{{ competition.startDate }}{{ " to "}}
        {{ competition.endDate
        }} </span>
        </h1>
    <p class="w3-text-violet">{{ competition.websiteName
    }}</p> <hr>
        {% endif %}
      {%endif %}
     {% endfor %}
   </div>
</div>
```

CODE CONTEST CALENDAR

Home page of the website:

This page contains information on the database of the important methods of learning the competitive coding. By navigating from this page to Calendar tab will have the calendar, which is the main aim of the project.



Learning the Art
of Competitive Coding

Coding Competitions Calendar .

1 INTRO : About Competitive Coding . >

2 To have a quick start on coding competitions >

A subtle thought may yet give rise to fruitful inquiry that can establish truths of great value.

Competitive programming - is a mind sport usually held over the Internet or a local network, involving participants trying to program according to provided

Future

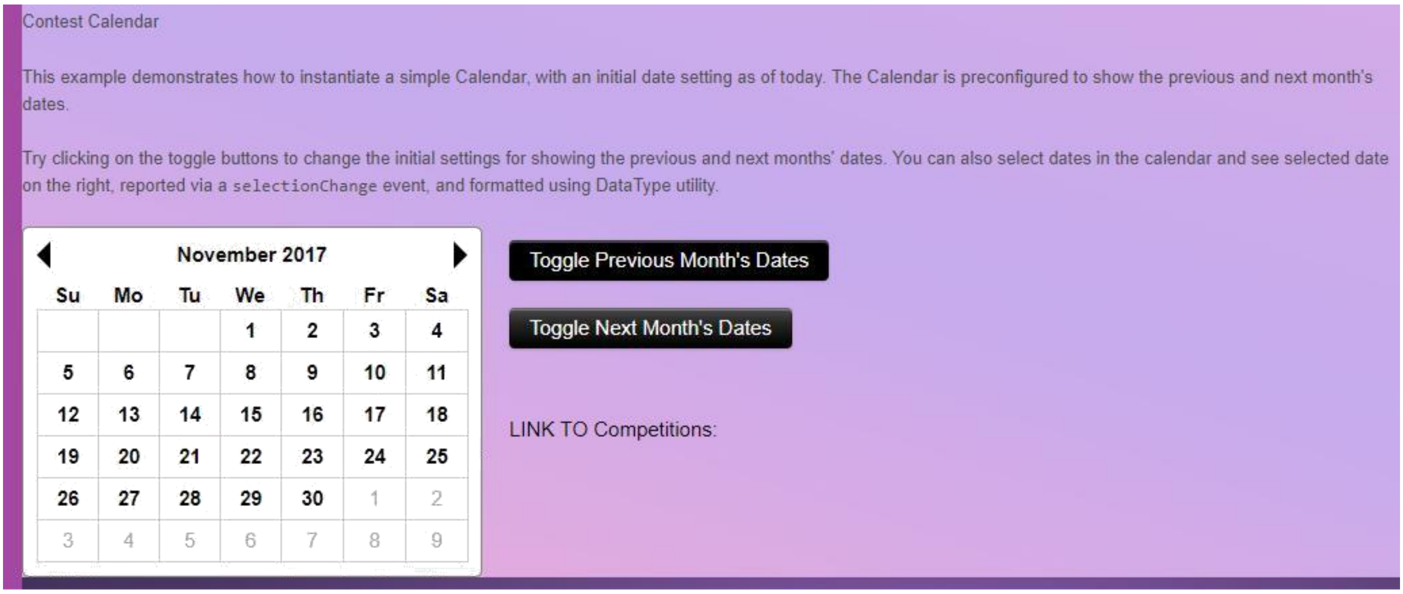
Is it really Important to

Wonders

View the wonders coding

Calendar part of the website:

This page contains all the following features that will be displayed. Toggle buttons fetches data from the database of the website and displays it on the calendar with faded font as seen.



Contest Calendar

This example demonstrates how to instantiate a simple Calendar, with an initial date setting as of today. The Calendar is preconfigured to show the previous and next month's dates.

Try clicking on the toggle buttons to change the initial settings for showing the previous and next months' dates. You can also select dates in the calendar and see selected date on the right, reported via a `selectionChange` event, and formatted using DataType utility.

◀ November 2017 ▶

Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2
3	4	5	6	7	8	9

LINK TO Competitions:

CODE CONTEST CALENDAR

The Calendar:

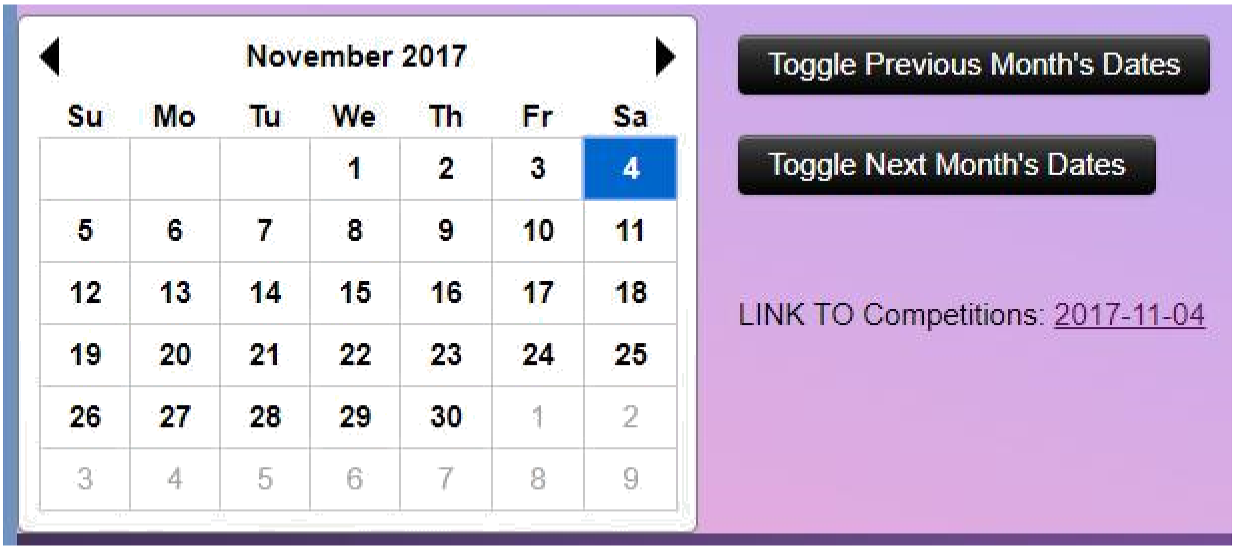
As soon as any date is selected from the calendar, that particular date gets displayed on the right of the text- LINK TO Competitions :

OUTPUT:

This is the calendar that gets displayed in the web page. JavaScript is the scripting language that was used to navigate the selections and got linked with the database of calendar dates.

The selected date is passed in to the HTML pages, where the date is selected. When it is queried, the data queries inside the data base of the Calendar.

Ex: The date November 4, 2017 is selected and hence a clickable link is displayed as :

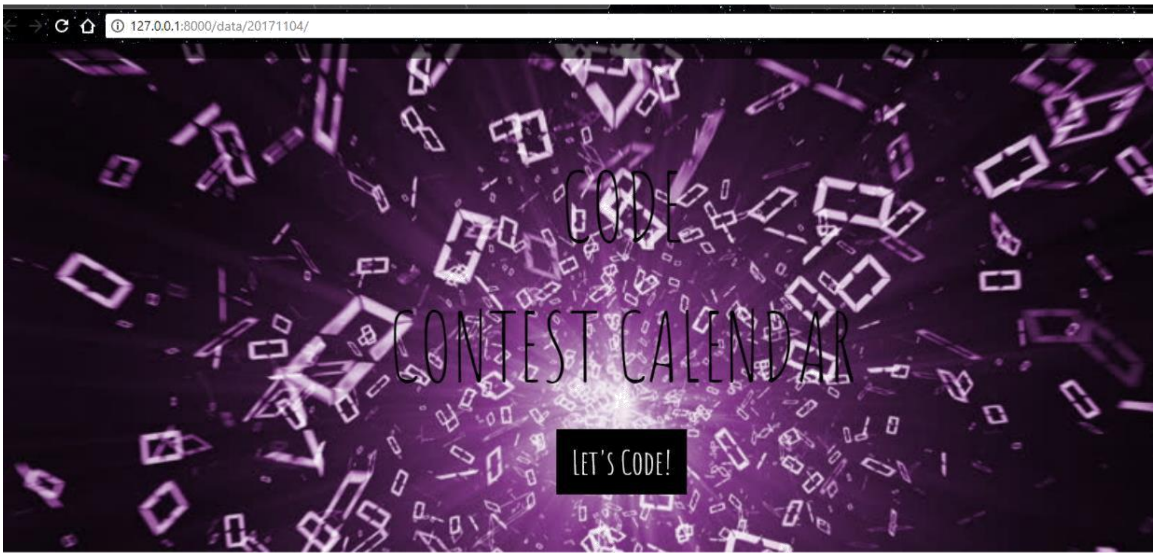


November 2017

Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2
3	4	5	6	7	8	9

LINK TO Competitions: 2017-11-04

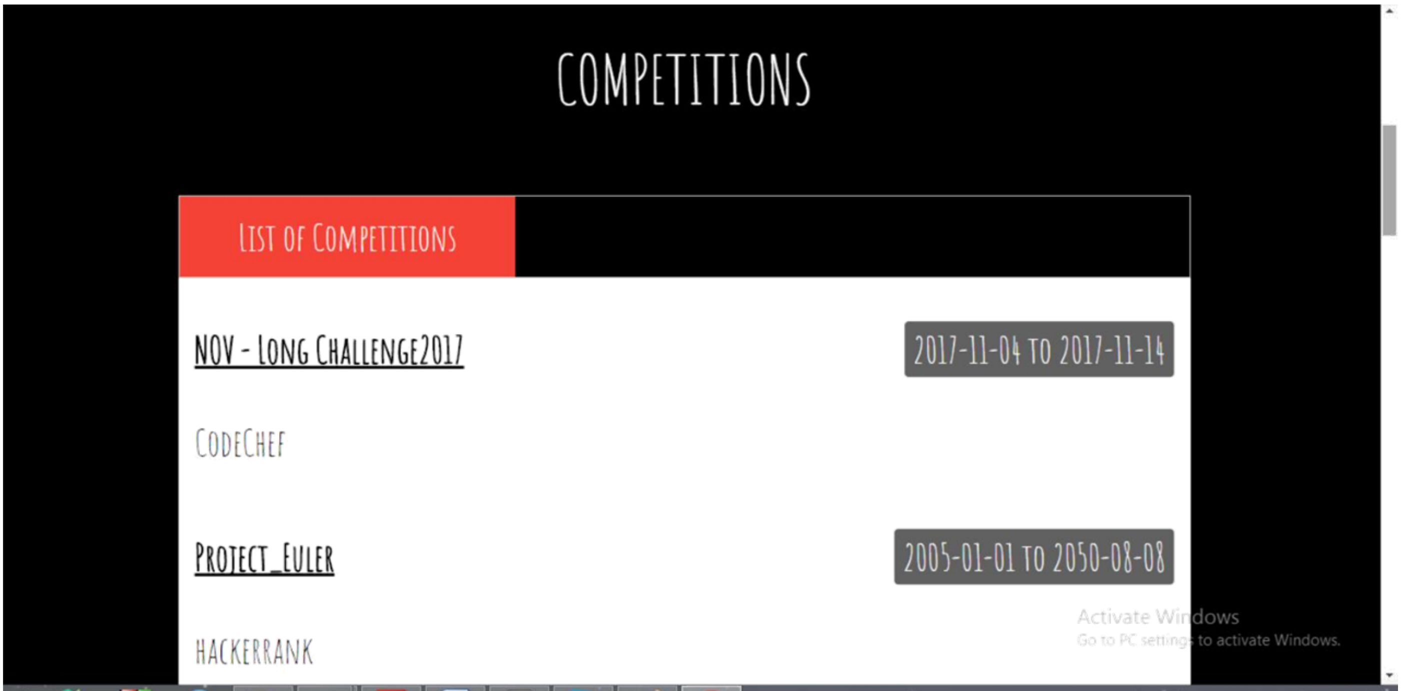
On clicking the link, you are automatically directed to the following page:



127.0.0.1:8000/data/20171104/

CODE CONTEST CALENDAR

By web crawling, all the competitions on this date are displayed here:



COMPETITIONS

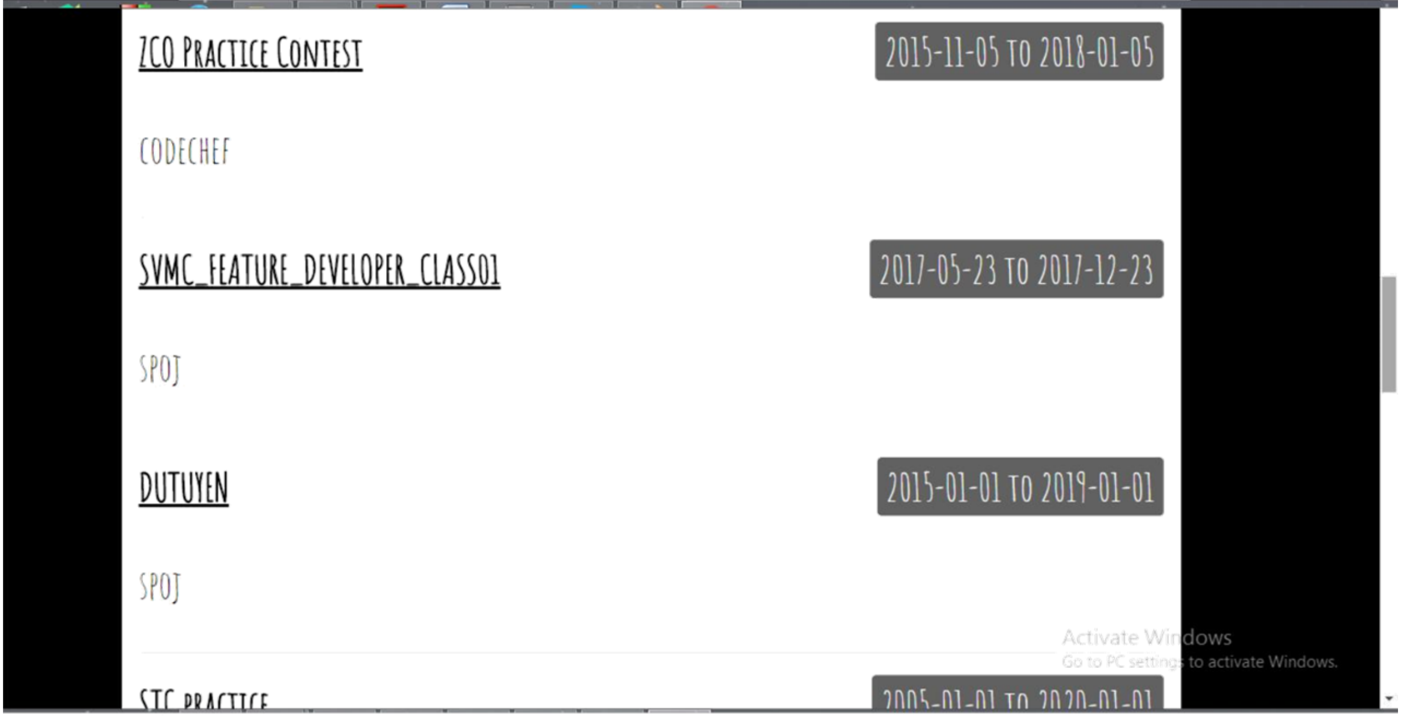
LIST OF COMPETITIONS

NOV - LONG CHALLENGE2017 — 2017-11-04 TO 2017-11-14

CODECHEF

PROJECT_EULER — 2005-01-01 TO 2050-08-08

HACKERRANK



ZCO PRACTICE CONTEST — 2015-11-05 TO 2018-01-05

CODECHEF

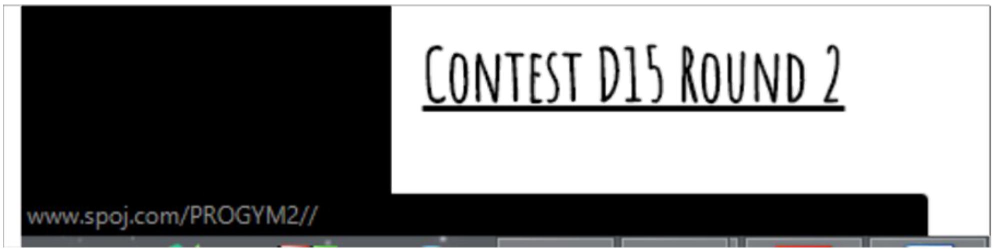
SVMC_FEATURE_DEVELOPER_CLASS01 — 2017-05-23 TO 2017-12-23

SPOJ

DUTUYEN — 2015-01-01 TO 2019-01-01

SPOJ

STC PRACTICE — 2005-01-01 TO 2020-01-01



CONTEST D15 ROUND 2

www.spoj.com/PROGYM2//

On hovering to the text, we can see that the respective link is to displayed, that is http://www.spoj.com/PROGYM2//

CONCLUSION AND FUTURE WORK

Applications:

A website is created where a user can login or register. A week calendar is displayed where a particular day is selected and the competitions present on that day from every site is displayed, sorted according to the preference of the user from past participation in competitions which is accomplished by machine learning. The home page shows the ongoing competitions by default. When a particular competition is selected it goes to the competition link. The default page displayed is the competition of the present date, but the user can always change the date to view the competitions of a particular date.

Data in the Database:

The competitions present in different websites are already categorized into present, future and past. This data is collected from the competitive coding sites like CodeChef, HackerRank, HackerEarth, Euler, SPOJ, Codewars, CodeForces, Coderbyte, CodeEval, LeetCode and Codingame. One entity that deals with the competitions ,it consists of the competition name, Coding site, the link to the competition, the start date and time, the end date and time, the duration of the competition and the difficulty rating (rated by the users).

The other entity which deals with the user data consists of the Username, Password, Email-id and other primary details.The database of a individual user also contains the metadata obtained from the maching learning techniques of user’s activity, which includes the favorite coding site of the user and the preferred duration of the user. The plan is to extend this to favorite type of test(Algorithm, Maths, Code challenge etc.,).

USERS OF THE SYSTEM:

The users of the system are the coders in short. This is the best and the only online coding comparison website, as it involves intelligent queries merged along with Web mining. The users range can vary from students to professionals, authors to teachers, employed to unemployed. The highly user friendly interface due to the intelligent database planned will not only help in easy usage but also in up-gradation of coding skills. The users can use the website, without getting logged in. All the functions are same to a logged in user, except the unique sorted interface of the logged in user and the ability to rate the website or the coding competition.

TECHNIQUES USED:

SQL is the database query language used. HTML, CSS, Python(BeautifulSoup), JavaScript and AJAX are thescripting language used for website development and crawling.

FUTURE SCOPE AND FURTHER IMPLEMENTATIONS:

In the front-end the user operations can be browsing the different competitions and the websites according to various dates. The user can also rate the websites which is star based where a 5 star rating indicates the competitions in this site are highly difficult compared to other sites' competitions which has a lower star rating. By intelligent machine learning techniques every user gets a unique preference display according to his/her ratings.

We presently four three entities, one for the user and the other for the competitions. Both the entities will be linked to each other. The user data base will have primary details of each user, and the meta data created from his previous activities which are obtained from the machine learning techniques. The user data base will have the websites sorted according to the activity of the each individual user. This data will be used, when the user is logged into the website, to display the competitions. The database of the competitions will have the the competitions of each website put under a table with fields- Competition Link, Competition name, Coding site of the competition, start date, end date, start time, end time and the duration of the contest. Each individual user is entitled to have interest in a competition of a particular time length. Machine learning techniques could be used, in order to give suggestions to the users. Competition link is used as the primary key in the database of the competitions and email ID as the primary key to the database of the users.

The highly user friendly interface due to the intelligent database planned will not only help in easy usage but also in up-gradation of coding skills. The users can use the website, without getting logged in. All the functions are same to a logged in user, except the unique sorted interface of the logged in user and the ability to rate the website or the coding competition.