

**WEB DEVELOPMENT
LABORATORY
(20ES3151)**

Week - 0

1.What is the full form of HTML?

Ans. HyperText MarkUp Language

2.What is the present version of HTML?

Ans. HTML5 with html file extension

3. Differentiate between WEB PAGE and WEB SITE

WEB PAGE	WEB SITE
A direct URL link or a website can be used to access it	A domain address is used to access it
Webpage consists of content regarding a single entity type	Website constitutes content regarding several entities
A combination of web pages is created using HTML and CSS	Information is in HTML language
An individual hypertext document linked under a website	A collection of multiple pages hosted on the server
Less time to develop	Takes more time to develop
Comparatively, less complex	More complex to develop

4. What are h1, h2, h3, h4, h5 and h6 tags ?

Ans. HTML defines six levels of headings. The heading elements are h1, h2, h3, h4, h5 and h6 with h1 being the highest level and h6 being the least. Used to create text headers for a document

5. What is the difference between LAN, WAN, MAN

Ans. LAN – A LAN connects computers within a small and specific area geographically

WAN –A WAN covers large area geographically such as continent, state or a country

MAN – A MAN is confined to a specific town, city or region

6. What is the difference between physical and logical address

Ans. 1- Logical address is generated by CPU during a program execution

2 – The physical address refers to a location in the memory unit

7. How to write a tag in HTML?

Ans. A HTML tag is special word or letter surrounded by angle brackets < and > paragraph element has <p> tag followed by paragraph text followed by a closing </p> tag

8. Brief History about HTML

Ans. HTML was created by Sir Tim Berners-Lee in late 1991 but was not released officially published in 1995 as HTML 2.0.

HTML 4.01 was published in late 1999 and was a major version of HTML.

HTML 1.0 was released in 1993 with the intention of sharing information that can be readable and accessible via web browsers

Week - 1

1. What are the different types of tags

Ans. There are two types of tags in HTML , they are:

a. Paired tags: In each paired tag set there will be an opening tag along with the closing tag, the closing tag contains a slash (/)

Ex: <h1>heading tag</h1>

<p>paragraph tag</p>

<html> html tag</html>

<head> head tag</head>

<body> body tag</body>

b. Unpaired tags: The unpaired tags does not contain the closing tag, these are also called as **singular tags**

Ex:
 , <hr> , <meta> ,<input> ,

2. AIM: To write a welcome program

PROGRAM:

```
<!DOCTYPE html>

<html>

<title>INTRODUCTION</title>

<head>

<h1>Welcome to HTML</h1>

</head>
```

OUTPUT:

Welcome to HTML

3. AIM: Performing a task using different heading tags

PROGRAM:

```
<!DOCTYPE html>

<body>

<h1>Welcome to html</h1>

<h2>Welcome to html</h2>

<h3>Welcome to html</h3>

<h4>Welcome to html</h4>

<h5>Welcome to html</h5>

<h6>Welcome to html</h6>

</body>

</html>
```

OUTPUT:

Welcome to html

WEEK – 2

1. AIM: Insertion of photo in the resume

PROGRAM:

```
<!DOCTYPE html>

<head>
<title>My Profile</title>

<h1><center>RESUME</center></h1>

</head>

<body>



<h3>ABOUT ME:</h3>

<p>My name is SPANDANA VATTIPALLI, I am 19 years old, Currently Studying second year in Information Technology at Velagapudi Ramakrishna Siddhartha Engineering College, My native place is Vijayawada. My Hobbies are reading books, playing games.</p>

</body>

</html>
```

OUTPUT:

RESUME

ABOUT ME:

My name is SPANDANA VATTIPALLI, I am 19 years old. Currently Studying second year in Information Technology at Velagapudi Ramakrishna Siddhartha Engineering College. My native place is Vijayawada. My Hobbies are reading books, playing games.



WEEK - 3

1. AIM: VIDEO & AUDIO UPLOADING

PROGRAM:

```
<!DOCTYPE html>
<head>
<title>My Profile</title>

<center><video width="420" height="315"controls>
<source src="video resume.mp4" type="video/mp4">
</video></center>

<audio controls>
<source src="bird chirp.mp3" type="audio/mpeg">

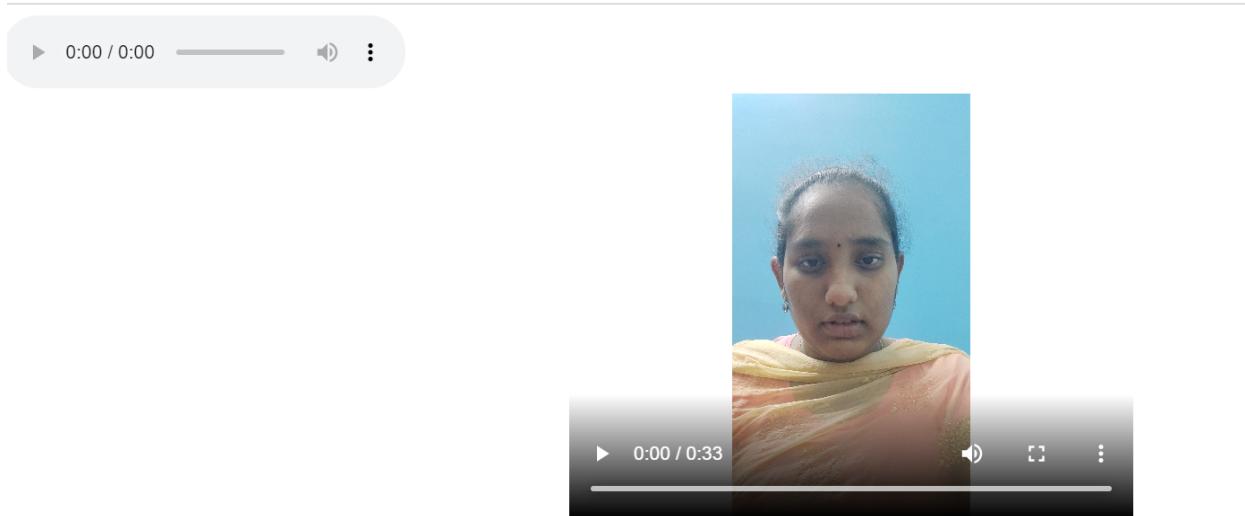
Your browser does not support the audio tag.

</audio>
```

</body>

</html>

OUTPUT:



Week - 4

1.AIM: To create different types of LISTS

PROGRAM:

```
<!DOCTYPE html>

<head>
<title>LISTS</title>
</head>

<body>
<h2>UNORDERED LIST</h2>
<h3>Fruits</h3>
<ul>
<li>Apple</li>
<li>Orange</li>
<li>Banana</li>
</ul>
```

```
<h2>ORDERED LIST</h2>
<h3>Colors</h3>
<ol>
<li>orange</li>
<li>green</li>
<li>blue</li>
```

```
</ol>
```

```
<h2>DESCRIPTION LIST</h2>
```

```
<h3>Menu</h3>
```

```
<dl>
```

```
<dt>Coffee</dt>
```

```
<dd>-espresso</dd>
```

```
<dd>-black coffee</dd>
```

```
<dd>-cold coffee</dd>
```

```
<dt>Milshake</dt>
```

```
<dd>-vanilla</dd>
```

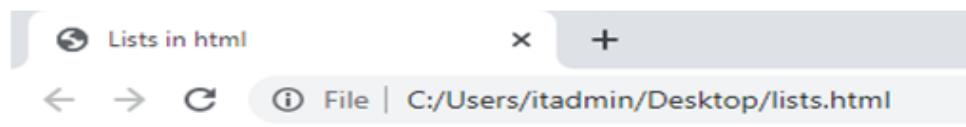
```
<dd>-chocolate</dd>
```

```
<dd>-strawberry</dd>
```

```
</body>
```

```
</html>
```

OUTPUT:



ORDERED LIST

Colors

1. orange
2. green
3. blue

DESCRIPTION LIST

Menu

Coffee

- espresso
- black coffee
- cold coffee

Milshake

- vanilla
- chocolate
- strawberry

WEEK - 5

1. AIM: To create the academic table

PROGRAM:

```
<!DOCTYPE html>

<head>
<style>
table, th, td {
    border:1px solid black;
}
</style>

<center><h2>TABLE</h2></center>

</head>

<body>
<table>
<tr>
<th>S.No</th>
<th>INSTITUTE/COLLEGE</th>
<th>BOARD</th>
<th>YEAR OF PASSING</th>
<th>CGPA</th>
</tr>
<tr>
```

```
<td>1</td>

<td>VRSEC</td>

<td>jntuk</td>

<td>2024</td>

<td>8</td>

</tr>

<tr>

<td>2</td>

<td>Sri Chaitanya Junior College</td>

<td>Board of Intermediate</td>

<td>2020</td>

<td>9</td>

</tr>

<tr>

<td>3</td>

<td>Sri Chaitanya High School</td>

<td>SSC</td>

<td>2018</td>

<td>10</td>

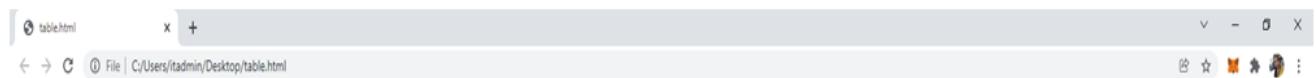
</tr>

</table>

</body>

</html>
```

OUTPUT:



TABLE

S.No	INSTITUTE COLLEGE	BOARD	YEAR OF PASSING	CGPA
1	VRSEC	Jntuk	2024	8
2	Sri Chaitanya Junior College	Board of Intermediate	2020	9
3	Sri Chaitanya High School	SSC	2018	10

WEEK - 6

1.AIM: To create a RESUME using frames

PROGRAM:

```
<!DOCTYPE html>

<html>

<frameset cols="25%,50%,25%">

<frame src="frame_1.html">

<frame src="frame_2.html">

<frame src="frame_3.html">

<frame src="frame_4.html">

</frameset>

</html>
```

OUTPUT:



v.spandana223@gmail.com

9876543210

[github](#)

[linkedin](#)

[hackerank](#)

Skill Set:

programming skills learned:

- HTML
- Python
- C
- C++
- CSS

courses completed:

- CODE RIDERS
- APSSDC

SPANDANA VATTIPALLI

About Me:

My name is SPANDANA VATTIPALLI, I am 19 years old, Currently Studying second year in Information Technology at Velagapudi Ramakrishna Siddhartha Engineering College, My native place is Vijayawada. My Hobbies are reading books, playing games.

Career Objective:

To use my skills in best possible ways to achieve company goals and to solve problems in an effective/creative manner in a challenging position

S.No	INSTITUTE/COLLEGE	BOARD	YEAR OF PASSING	CGPA
1	VRSEC	intuk	2024	8
2	Sri Chaitanya Junior College	Board of Intermediate	2020	9
3	Sri Chaitanya High School	SSC	2018	10

Strengths:

- Ability to cope up with the situation
- Planning and Accomplishing a task
- Confident and Determined

Hobbies:

- Painting
- Writing
- Playing Badminton

Personal details:

DATE OF BIRTH : 2nd February 2003

FATHER'S NAME : V.BHASKARA RAO

GENDER : Female

LANGUAGES KNOWN : English,Telugu

Week-7

AIM: Create a form using html

PROGRAM :

```
<html>
<head>
<title>Forms</title>
</head>
<body>
<form name="it" action="next.php" method="post">
Name: <input type="textbox" placeholder="Enter Name" name="Name" required>
<br>
Father Name: <input type="textbox" placeholder="Enter Name" name="Father Name" required>
<br>
Gender: Male <input type="radio" name="gender" value="Male"> Female <input type="radio" name="gender" value="Female">
<br>
Branch:
<select name="branch">
<option>--- Select ---</option>
```

```
<option> CSE </option>

<option> IT </option>

<option> ECE </option>

</select>

<br>

Languages Known: Telugu

<input type="checkbox" value="telugu" name="telugu">

English

<input type="checkbox" value="english" name="english">

Hindi

<input type="checkbox" value="hindi" name="hindi">

<pre>      <input type="submit" value="Submit"></pre>

</form>

</body>

</html>
```

OUTPUT:

The screenshot shows a Microsoft Edge browser window with the title bar "Forms". The address bar displays the file path "File | C:/Users/meetp/OneDrive/Desktop/forms.html". The main content area contains the following form fields:

- Name:
- Father Name:
- Gender: Male Female
- Branch:
- Languages Known: Telugu English Hindi

A "Submit" button is located at the bottom left of the form.

At the bottom of the screen, the Windows taskbar is visible, showing icons for Start, Search, Task View, File Explorer, Mail, File Explorer, Google Chrome, and Microsoft Edge. The system tray shows the date and time as "14-02-2022 07:27".

WEEK - 8

AIM : Create a WEBSITE using HTML

PROGRAM :

HOME PAGE:

```
<!DOCTYPE html>

<head>

<style>

table,th,td{

border:1px solid black;

}

</style>

<style>

h1 {

text-shadow: 2px 2px 5px red;

}

</style>

<title>home page</title>

<h1 style="font-size:50px;"><center>INDIAN PREMIERE LEAGUE</center></h1>

</head>
```

```
<style>

a:link {

    text-decoration: none;

}

a:visited {

    text-decoration: none;

}

a:hover {

    text-decoration: underline;

}

a:active {

    text-decoration: underline;

}

</style>

<body style="background-image: linear-gradient(to right, #ffafbd, #ffc3a0);">

<nav style="color:red;">

<a href="home.html"><span
style="font-size:25px;color:red;">HOME & & & </a>

<a href="login.html"><span
style="font-size:25px;color:red;">LOGIN & & & & </a>
```

```
<a href="registration.html"><span  
style="font-size:25px;color:red;">REGISTRATION   &ampnbsp&ampnbsp</a>  
  
<a href="editpage.html"><span  
style="font-size:25px;color:red;">EDIT   &ampnbsp&ampnbsp&ampnbsp&ampnbsp</a>  
  
<a href="more.html"><span style="font-size:25px;color:red;">MORE</a><br><br><br>  
</nav>  
  
<center></center>  
  
<div>  
  
<br><br><br>  
  
<table>  
  
<tr>  
  
<th>S.No</th>  
  
<th>TEAMS</th>  
  
<th>TOTAL MATCHES PLAYED</th>  
  
<th>WON</th>  
  
<th>LOST</th>  
  
<th>DRAW</th>  
  
<th>POINTS</th>  
  
</tr>  
  
<tr>
```

<td>1</td>

<td>MI</td>

<td>15</td>

<td>12</td>

<td>2</td>

<td>1</td>

<td>+53</td>

</tr>

<tr>

<td>2</td>

<td>CSK</td>

<td>15</td>

<td>11</td>

<td>2</td>

<td>2</td>

<td>+48</td>

</tr>

<tr>

<td>3</td>

<td>RCB</td>

<td>14</td>

<td>10</td>

<td>3</td>

<td>1</td>

<td>+40</td>

</tr>

<tr>

<td>4</td>

<td>KXIP</td>

<td>14</td>

<td>9</td>

<td>3</td>

<td>2</td>

<td>+37</td>

</tr>

<tr>

<td>5</td>

<td>SRH</td>

<td>14</td>

<td>8</td>

<td>4</td>

<td>2</td>

<td>+32</td>

</tr>

<tr>

<td>6</td>

<td>DC</td>

<td>15</td>

<td>8</td>

<td>4</td>

<td>3</td>

<td>+29</td>

</tr>

<tr>

<td>7</td>

<td>KKR</td>

<td>14</td>

```
<td>8</td>
```

```
<td>5</td>
```

```
<td>1</td>
```

```
<td>+23</td>
```

```
</tr>
```

```
<tr>
```

```
<td>8</td>
```

```
<td>RR</td>
```

```
<td>15</td>
```

```
<td>8</td>
```

```
<td>5</td>
```

```
<td>2</td>
```

```
<td>+20</td>
```

```
</tr>
```

```
</table>
```

```
</div>
```

```
</body>
```

```
</html>
```

LOGIN PAGE:

```
<!DOCTYPE html>

<head>

<title>login page</title>

<h1 style="font-size:50px;"><center>INDIAN PREMIERE  
LEAGUE</center></h1>

</head>

<style>

a:link {

    text-decoration: none;

}

a:visited {

    text-decoration: none;

}

a:hover {

    text-decoration: underline;

}

a:active {

    text-decoration: underline;

}
```

```
</style>

<body style="background-image: linear-gradient(to right, #ffafbd, #ffc3a0);>

<nav>

<a href="home.html"><span
style="font-size:25px;color:red;">HOME &nbsnbsp;&nbsnbsp;&nbsnbsp;&nbsnbsp;</a>

<a href="login.html"><span
style="font-size:25px;color:red;">LOGIN &nbsnbsp;&nbsnbsp;&nbsnbsp;&nbsnbsp;</a>

<a href="registration.html"><span
style="font-size:25px;color:red;">REGISTRATION &nbsnbsp;&nbsnbsp;&nbsnbsp;&nbsnbsp;</a>

<a href="editpage.html"><span
style="font-size:25px;color:red;">EDIT &nbsnbsp;&nbsnbsp;&nbsnbsp;&nbsnbsp;&nbsnbsp;</a>

<a href="more.html"><span
style="font-size:25px;color:red;">MORE</a><br><br><br>

</nav>

<div>

<br><br><br>

<h2><center>WELCOME TO LOGIN PAGE</center></h2>

<center>

<table>

<form>

<tr>

<td>Username</td>

<td><input type="text" name="t1"></td>

</tr>
```

```
<tr>

<td>Password</td>

<td><input type="Password" name="t2"></td>

</tr>

<tr>

<td><center><input type="submit"></td>

</tr>

</form>

</table>

</center>

</div>

</body>

</html>
```

REGISTRATION PAGE:

```
<!DOCTYPE html>

<head>

<title>registration page</title>

<style>

table{

margin-top:50px;
```

```
}

</style>

<h1 style="font-size:50px;"><center>INDIAN PREMIERE  
LEAGUE</center></h1>

</head>

<style>

a:link {

    text-decoration: none;

}

a:visited {

    text-decoration: none;

}

a:hover {

    text-decoration: underline;

}

a:active {

    text-decoration: underline;

}

</style>

<body style="background-image: linear-gradient(to right, #ffafbd, #ffc3a0);">
```

```
<nav>

<a href="home.html"><span
style="font-size:25px;color:red;">HOME &nbspp &nbspp &nbspp</a>

<a href="login.html"><span
style="font-size:25px;color:red;">LOGIN &nbspp &nbspp &nbspp</a>

<a href="registration.html"><span
style="font-size:25px;color:red;">REGISTRATION &nbspp &nbspp &nbspp</a>

<a href="editpage.html"><span
style="font-size:25px;color:red;">EDIT &nbspp &nbspp &nbspp</a>

<a href="more.html"><span
style="font-size:25px;color:red;">MORE</a><br><br><br>

</nav>

<h2><center>WELCOME TO REGISTRATION PAGE</center></h2>

<center>

<table>

<form>

<tr>

<td>Username:</td>

<td><input type="text" name="t1"></td>

</tr>

<tr>

<td>Password:</td>

<td><input type="Password" name="t2"></td>

</tr>
```

```
<tr>

<td>Surname : </td>

<td><input type="surname" name="t3"></td>

</tr>

<tr>

<td>DateofBirth: </td>

<td><input type="" name="t4"></td>

</tr>

<tr>

<td>Email: </td>

<td><input type="email" name="t5"></td>

</tr>

<tr>

<td><center><input type="submit"></td>

</tr>

</form>

</table>

</center>

</body>

</html>
```

EDIT PAGE:

```
<!DOCTYPE html>

<head>
<title>edit page</title>

<h1 style="font-size: 50px";><center>INDIAN PREMIERE
LEAGUE</center></h1>

</head>

<style>

a:link {
    text-decoration: none;
}

a:visited {
    text-decoration: none;
}

a:hover {
    text-decoration: underline;
}
```

```
}
```

```
a:active {
```

```
    text-decoration: underline;
```

```
}
```

```
</style>
```

```
<body style="background-image: linear-gradient(to right, #ffafbd, #ffc3a0);">
```

```
<nav>
```

```
<a href="home.html"><span
```

```
style="font-size:25px;color:red;">HOME &nbs
```

```
<a href="login.html"><span
```

```
style="font-size:25px;color:red;">LOGIN &nbs
```

```
<a href="registration.html"><span
```

```
style="font-size:25px;color:red;">REGISTRATION &nbs
```

```
<a href="editpage.html"><span
```

```
style="font-size:25px;color:red;">EDIT &nbs
```

```
<a href="more.html"><span
```

```
style="font-size:25px;color:red;">MORE</a><br><br><br>
```

```
</nav>
```

```
<h2><center>WELCOME TO EDIT PAGE</center></h2>
```

```
<form>
```

```
<center><input type="text" name="t1">
```

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

```
</form>
```

```
</body>
```

```
</html>
```

MORE PAGE:

```
<!DOCTYPE html>

<head>

<title>more page</title>

<center><h2>WELCOME TO MORE INFORMATION</h2></center>

<br><br>

<style>

table,th,td{

border:1px solid black;

}

</style>

</head>

<style>

a:link {

text-decoration: none;

}

a:visited {
```

```
text-decoration: none;  
}  
  
  
a:hover {  
    text-decoration: underline;  
}  
  
  
a:active {  
    text-decoration: underline;  
}  
  
</style>  
  
<style>  
* {  
    box-sizing: border-box;  
}  
  
  
body {  
    margin: 0;  
    font-family: Arial;  
}  
  
  
.header {
```

```
text-align: center;  
padding: 32px;  
}  
  
.row {  
display: -ms-flexbox; /* IE10 */  
display: flex;  
-ms-flex-wrap: wrap; /* IE10 */  
flex-wrap: wrap;  
padding: 0 4px;  
}  
  
/* Create four equal columns that sits next to each other */  
.column {  
-ms-flex: 25%; /* IE10 */  
flex: 25%;  
max-width: 25%;  
padding: 0 4px;  
}  
  
.column img {  
margin-top: 8px;
```

```
vertical-align: middle;  
width: 100%;  
}  
  
/* Responsive layout - makes a two column-layout instead of four columns */
```

```
@media screen and (max-width: 800px) {  
.column {  
-ms-flex: 50%;  
flex: 50%;  
max-width: 50%;  
}  
}
```

```
/* Responsive layout - makes the two columns stack on top of each other instead  
of next to each other */
```

```
@media screen and (max-width: 600px) {  
.column {  
-ms-flex: 100%;  
flex: 100%;  
max-width: 100%;  
}  
}  
  
</style>
```

```
<body style="background-image: linear-gradient(to right, #ffafbd, #ffc3a0);>

<nav>

<a href="home.html"><span
style="font-size:25px;color:red;">HOME &nbspp;&nbspp;&nbspp;&nbspp;</a>

<a href="login.html"><span
style="font-size:25px;color:red;">LOGIN &nbspp;&nbspp;&nbspp;&nbspp;</a>

<a href="registration.html"><span
style="font-size:25px;color:red;">REGISTRATION &nbspp;&nbspp;&nbspp;&nbspp;</a>

<a href="editpage.html"><span
style="font-size:25px;color:red;">EDIT &nbspp;&nbspp;&nbspp;&nbspp;</a>

<a href="more.html"><span
style="font-size:25px;color:red;">MORE</a><br><br><br>

</nav>

<div class="row">

<div class="column">









</div>

<div class="column">






```

```
  
</div>  
  
<div class="column">  
      
      
      
      
</div>  
  
<div class="column">  
      
      
      
      
</div>  
  
</div>  
  
<br><br><br>  
<table>  
    <tr>  
        <th>YEARS</th>  
        <th>WINNERS</th>  
    </tr>  
    <tr>
```

<td>2015</td>
<td>MUMBAI INDIANS</td>
</tr>
<tr>
<td>2016</td>
<td>SUNRISERS HYDERABAD</td>
</tr>
<tr>
<td>2017</td>
<td>MUMBAI INDIANS</td>
</tr>
<tr>
<td>2018</td>
<td>CHENNAI SUPER KINGS</td>
</tr>
<tr>
<td>2019</td>
<td>MUMBAI INDIANS</td>
</tr>
<tr>
<td>2020</td>
<td>MUMBAI INDIANS</td>

```
</tr>
<tr>
<td>2021</td>
<td>CHENNAI SUPER KINGS</td>
</tr>
</table>
</body>
</html>
```

OUTPUT:

HOME -

INDIAN PREMIERE LEAGUE

HOME LOGIN REGISTRATION EDIT MORE



S.No	TEAMS	TOTAL MATCHES PLAYED	WON	LOST	DRAW	POINTS
1	MI	15	12	2	1	+53
2	CSK	15	11	2	2	+48
3	RCB	14	10	3	1	+40
4	KXIP	14	9	3	2	+37
5	SRH	14	8	4	2	+32
6	DC	15	8	4	3	+29
7	KKR	14	8	5	1	+23
8	RR	15	8	5	2	+20

LOGIN PAGE -

INDIAN PREMIERE LEAGUE

HOME LOGIN REGISTRATION EDIT MORE

WELCOME TO LOGIN PAGE

Username:
Password:

REGISTRATION PAGE -

INDIAN PREMIERE LEAGUE

HOME LOGIN REGISTRATION EDIT MORE

WELCOME TO REGISTRATION PAGE

Username: :
Password:
Surname :
DateofBirth:
Email:

EDIT PAGE -

INDIAN PREMIERE LEAGUE

HOME LOGIN REGISTRATION EDIT MORE

WELCOME TO EDIT PAGE

MORE INFORMATION PAGE -

WELCOME TO MORE INFORMATION

HOME LOGIN REGISTRATION EDIT MORE

The collage consists of 15 images arranged in a grid. The top row shows: 1. A team group photo with fireworks in the background. 2. A player in black bowling. 3. A player in blue and yellow running with a bat. 4. A group of players in blue and yellow celebrating. The middle row shows: 1. A player in red and yellow holding a bat. 2. Two players in yellow and blue standing together. 3. A group of players in blue and yellow. 4. Two players in red and yellow talking. The bottom row shows: 1. A player in yellow and blue running. 2. A group of players in yellow and blue. 3. The IPL trophy surrounded by Vivo banners. 4. Three players in pink and blue celebrating.

YEARS	WINNERS
2015	MUMBAI INDIANS
2016	SUNRISERS HYDERABAD
2017	MUMBAI INDIANS
2018	CHENNAI SUPER KINGS
2019	MUMBAI INDIANS
2020	MUMBAI INDIANS
2021	CHENNAI SUPER KINGS

Week - 9

1. What is Django, what is the framework that is used to implement Django.

DJANGO

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

Ridiculously fast.

Django was designed to help developers take applications from concept to completion as quickly as possible.

Reassuringly secure.

Django takes security seriously and helps developers avoid many common security mistakes.

Exceedingly scalable. Some of the busiest sites on the web leverage Django's ability to quickly and flexibly scale.

FRAMEWORK

Rich ecosystem

Read Django like a system, developers say. What they mean is that there are many third-party applications that come with Django. These applications can be integrated depending on project requirements. To imagine this better, think of Legos. There are many different Lego blocks. In app development, an authorization “block” or email sending “block” is present in almost every project. Django consists of many applications — such as for authorization and sending emails — that can easily be plugged into a system.

Maturity. Django has been around for 11 years and has gone through stages of significant improvement. A lot of things have been brought to perfection and many new things have been added. Most importantly, when you’re trying to figure out how something should work in Django, you can usually find the answer. Thousands of people must have already solved any issue you’re dealing with, and you can find a solution provided by the passionate Django community.

Admin panel by default

Admin panels are designed to help you manage your application. A Django admin panel is generated automatically from Python code, whereas creating an admin panel manually would take a lot of time and be absolutely pointless.

There’s a lot of room for customization in the Django admin panel thanks to third-party applications. Additionally, Django allows you to modify the interface with third-party wrappers and add dashboards unique to your needs.

Good for SEO

Python is famous for having human-readable code, and that’s an advantage if you want your site to rank high in search results. With Django, you can generate

readable website URLs and links using the most relevant keywords and search engine optimization (SEO) best practices.

After all, a domain name is just a “human-readable” string that maps to a “computer friendly” set of numbers, known as an IP address. People fixate on getting the right domain name, but tend to neglect what the URL slug is—Django can fix that.

Pluggable

Django is pluggable by nature and can be extended with plugins. Plugins are software components that allow developers to add a specific feature to an app, leaving a lot of scope for customization. There are hundreds of packages to help you add Google Maps, create complex permissions, or connect to Stripe to process payments. And if you need to scale your project in the future, you can unplug some components and replace them with others that meet your current needs.

Libraries

Every programming language comes with its own set of libraries for solving common tasks. A software library includes prewritten code, classes, procedures, scripts, configuration data, and more. As a rule, a library is added to a program to provide more functionality or to automate a process without manually writing new code. This reduces time to market.

Django allows developers to use libraries when building any project. Some popular libraries include the Django REST framework, which is responsible for building application programming interfaces (APIs); Django CMS, which is designed to manage website content; and Django-allauth, which is an integrated set of Django applications for authentication, registration, account management, and third-party (social) account authentication.

ORM

Django is valued for its object-relational mapper that helps developers interact with databases. An object-relational mapper (ORM) is a library that automatically transfers data stored in databases such as PostgreSQL and MySQL into objects commonly used in application code.

2. What are the prerequisites to implement Django.

There are many prerequisites to learn Django:

The basic syntax of the Python

Python functions

importing external modules

understanding of Python Path concepts

conditional operators and loops

string manipulation operations

Regular Expression (RE)

object-oriented concepts

Learn REST APIs and JSON

Database Management and SQL queries

3. How to install and run Django

Install Django

Django can be installed easily using `pip`.

In the command prompt, execute the following command: `pip install django`. This will download and install Django.

After the installation has completed, you can verify your Django installation by executing `django-admin --version` in the command prompt.

Use the Django admin console

Create a superuser. You will be prompted to enter a username, email, and password.
`python manage.py createsuperuser`.

Start a local web server: `python manage.py runserver`.

Log in to the admin site using the username and password you used when you ran `createsuperuser`.

4. Write a very basic program using Django.

django_admin startproject hello_world_project

hello_world_project/

manage.py

helloworld_project/

cd hello_world_project

python manage.py migrate

python manage.py runserver__init__.py

settings.py

urls.py

asgi.py

migrating and testing project

5. What is a database.Which database is by default in Django.

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a **database management system (DBMS)**. Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

By default, the configuration uses SQLite. If you're new to databases, or you're just interested in trying Django, this is the easiest choice. SQLite is included in Python, so you won't need to install anything else to support your database.

6. What is the framework and advantages of Django.

Rich ecosystem

Read Django like a system, developers say. What they mean is that there are many third-party applications that come with Django. These applications can be integrated depending on project requirements. To imagine this better, think of Legos. There are many different Lego blocks. In app development, an authorization “block” or email sending “block” is present in almost every project. Django consists of many applications — such as for authorization and sending emails — that can easily be plugged into a system.

Maturity. Django has been around for 11 years and has gone through stages of significant improvement. A lot of things have been brought to perfection and many new things have been added. Most importantly, when you’re trying to figure out how something should work in Django, you can usually find the answer. Thousands of people must have already solved any issue you’re dealing with, and you can find a solution provided by the passionate Django community.

Admin panel by default

Admin panels are designed to help you manage your application. A Django admin panel is generated automatically from Python code, whereas creating an admin panel manually would take a lot of time and be absolutely pointless.

There’s a lot of room for customization in the Django admin panel thanks to third-party applications. Additionally, Django allows you to modify the interface with third-party wrappers and add dashboards unique to your needs.

Good for SEO

Python is famous for having human-readable code, and that’s an advantage if you want your site to rank high in search results. With Django, you can generate

readable website URLs and links using the most relevant keywords and search engine optimization (SEO) best practices.

After all, a domain name is just a “human-readable” string that maps to a “computer friendly” set of numbers, known as an IP address. People fixate on getting the right domain name, but tend to neglect what the URL slug is—Django can fix that.

Pluggable

Django is pluggable by nature and can be extended with plugins. Plugins are software components that allow developers to add a specific feature to an app, leaving a lot of scope for customization. There are hundreds of packages to help you add Google Maps, create complex permissions, or connect to Stripe to process payments. And if you need to scale your project in the future, you can unplug some components and replace them with others that meet your current needs.

Libraries

Every programming language comes with its own set of libraries for solving common tasks. A software library includes prewritten code, classes, procedures, scripts, configuration data, and more. As a rule, a library is added to a program to provide more functionality or to automate a process without manually writing new code. This reduces time to market.

Django allows developers to use libraries when building any project. Some popular libraries include the Django REST framework, which is responsible for building application programming interfaces (APIs); Django CMS, which is designed to manage website content; and Django-allauth, which is an integrated set of Django applications for authentication, registration, account management, and third-party (social) account authentication.

ORM

Django is valued for its object-relational mapper that helps developers interact with databases. An object-relational mapper (ORM) is a library that automatically transfers data stored in databases such as PostgreSQL and MySQL into objects commonly used in application code.

7. How to create an environment to start a new project in Django.

1. Install Package. First, install python3-venv package by using the following command.
2. Create a Directory. \$ mkdir djangoenv. ...
3. Create Virtual Environment. \$ python3 -m venv djangoenv. ...
4. Activate Virtual Environment.

8. What is the difference between project and app.

A *project* refers to the entire application and all its parts.

An *app* refers to a submodule of the project. It's self-sufficient and not intertwined with the other apps in the project such that, in theory, you could pick it up and plop it down into another project without any modification. An *app* typically has its own *models.py* (which might actually be empty). You might think of it as a standalone python module. A simple project might only have one app.

For your example, the *project* is the whole website. You might structure it so there is an *app* for articles, an *app* for ranking tables, and an *app* for fixtures and results. If they need to interact with each other, they do it through well-documented public classes and accessor methods.

The main thing to keep in mind is this level of interdependence between the *apps*. In practice it's all one *project*, so there's no sense in going overboard, but keep in mind how co-dependent two apps are. If you find one app is solving two problems, split them into two apps. If you find two apps are so intertwined you could never reuse one without the other, combine them into a single app.

WEEK - 10

AIM- How to implement a form in django

Step:1 - install django in command prompt

```
cmd Command Prompt
Microsoft Windows [Version 10.0.22000.434]
(c) Microsoft Corporation. All rights reserved.

C:\Users\meetp>py --version
Python 3.10.0

C:\Users\meetp>py -m venv project-name

C:\Users\meetp>project-name\Scripts\activate.bat

(project-name) C:\Users\meetp>py -m pip install Django
Requirement already satisfied: Django in c:\users\meetp\project-name\lib\site-packages (4.0.2)
Requirement already satisfied: tzdata in c:\users\meetp\project-name\lib\site-packages (from Django) (2021.5)
Requirement already satisfied: asgiref<4,>=3.4.1 in c:\users\meetp\project-name\lib\site-packages (from Django) (3.5.0)
Requirement already satisfied: sqlparse>=0.2.2 in c:\users\meetp\project-name\lib\site-packages (from Django) (0.4.2)
WARNING: You are using pip version 21.2.3; however, version 22.0.2 is available.
You should consider upgrading via the 'C:\Users\meetp\project-name\Scripts\python.exe -m pip install --upgrade pip' command.
```

Step:2 - create an environment

```
(project-name) C:\Users\meetp>cd desktop
The system cannot find the path specified.

(project-name) C:\Users\meetp>cd documents

(project-name) C:\Users\meetp\Documents>django-admin startproject five

(project-name) C:\Users\meetp\Documents>cd five

(project-name) C:\Users\meetp\Documents\five>django-admin startapp myapp

(project-name) C:\Users\meetp\Documents\five>■
```

Step:3 - open visual studio , folder five (as per me) in visual studio ,files will be uploaded

The screenshot shows the Visual Studio Code interface with the following details:

- Title Bar:** settings.py - Untitled (Workspace) - Visual Studio Code
- File Explorer (Left):** Shows the project structure under UNTITLED (WORKSPACE). It includes a folder 'five' containing __init__.py, asgi.py, settings.py, urls.py, and wsgi.py. A folder 'myapp' contains __init__.py, migrations, admin.py, apps.py, forms.py, models.py, tests.py, urls.py, views.py, and index.html.
- Code Editor (Center):** Displays the contents of settings.py. The code defines the project's configuration, including installed apps, middleware, root URLconf, and templates.
- Bottom Status Bar:** Shows the current file is settings.py, line 58, column 25. It also displays the language as Python, encoding as UTF-8, and the date and time as 06-02-2022 19:58.

```
settings.py
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'myapp'
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'five.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR / 'Templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

The screenshot shows the Visual Studio Code interface with the following details:

- Title Bar:** forms.py - Untitled (Workspace) - Visual Studio Code
- File Explorer (Left):** Shows the project structure under UNTITLED (WORKSPACE). It includes a folder 'five' containing __init__.py, asgi.py, settings.py, urls.py, and wsgi.py. A folder 'myapp' contains __init__.py, migrations, admin.py, apps.py, forms.py, models.py, tests.py, urls.py, views.py, and index.html.
- Code Editor (Center):** Displays the contents of forms.py. It defines a ModelForm named 'login' that inherits from django.forms.ModelForm and specifies the model as 'student' and the fields as '__all__'.
- Bottom Status Bar:** Shows the current file is forms.py, line 7, column 25. It also displays the language as Python, encoding as UTF-8, and the date and time as 06-02-2022 19:58.

```
forms.py
from django import forms
from myapp.models import student

class login(forms.ModelForm):
    class Meta:
        model=student
        fields='__all__'
```

views.py - Untitled (Workspace) - Visual Studio Code

```
myapp > views.py
1  from django.shortcuts import render
2
3  # Create your views here.
4  from django.shortcuts import render
5  from django.http import HttpResponseRedirect
6  from myapp.forms import Login
7
8  # Create your views here.
9  def form_view(request):
10     objects=Login()
11     if request.method == "POST" :
12         objects=Login(request.POST)
13     if objects.is_valid():
14         objects.save(commit=True)
15
16     return render(request,'index.html',{'objects':objects})
```

Ln 16, Col 1 Spaces: 4 UTF-8 CRLF Python ⚙

File Edit Selection View Go Run Terminal Help

EXPLORER

UNTITLED (WORKSPACE)

- myapp
- __pycache__
- __init__.py
- asgi.py
- settings.py
- urls.py
- wsgi.py

 myapp

- __pycache__
- migrations
- __init__.py
- admin.py
- apps.py
- forms.py
- models.py
- tests.py
- urls.py
- views.py

Templates

index.html

OUTLINE

② 0 0

ENGLISH IN 19:58 06-02-2022

urls.py - Untitled (Workspace) - Visual Studio Code

```
myapp > urls.py
1  from django.urls import path
2  from . import views
3  urlpatterns=[
4      path('form',views.form_view,name='form_view')
5  ]
```

Ln 5, Col 2 Spaces: 4 UTF-8 CRLF Python ⚙

File Edit Selection View Go Run Terminal Help

EXPLORER

UNTITLED (WORKSPACE)

- myapp
- __pycache__
- __init__.py
- asgi.py
- settings.py
- urls.py
- wsgi.py

 myapp

- __pycache__
- migrations
- __init__.py
- admin.py
- apps.py
- forms.py
- models.py
- tests.py
- urls.py
- views.py

Templates

index.html

OUTLINE

② 1 0

ENGLISH IN 20:01 06-02-2022

The screenshot shows the Visual Studio Code interface with the 'urls.py' file open in the center editor tab. The file contains Python code for defining URLs in a Django application. The code includes imports for 'django.urls' and 'myapp.views', and defines a URL pattern for the 'admin' site.

```
urls.py - Untitled (Workspace) - Visual Studio Code
1  """Five URL Configuration
2
3  The 'urlpatterns' list routes URLs to views. For more information please see:
4      https://docs.djangoproject.com/en/4.0/topics/http/urls/
5  Examples:
6      Function views
7          1. Add an import: from my_app import views
8          2. Add a URL to urlpatterns: path('', views.home, name='home')
9  Class-based views
10     1. Add an import: from other_app.views import Home
11     2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13     1. Import the include() function: from django.urls import include, path
14     2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 ...
16 from django.contrib import admin
17 from django.urls import path,include
18 from myapp import views
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('',include('myapp.urls'))
23 ]
```

The screenshot shows the Visual Studio Code interface with the 'index.html' file open in the center editor tab. The file contains HTML code for a form submission, including a POST method and CSRF token handling.

```
index.html - Untitled (Workspace) - Visual Studio Code
1  <!DOCTYPE html>
2  <html>
3      <body>
4          <form method="POST">
5              {% csrf_token %}
6              {{ objects.as_p }}
7              <input type="submit" value="Submit"/>
8          </form>
9      </body>
10 </html>
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "UNTITLED (WORKSPACE)".
- Editor:** Displays the content of the `models.py` file.
- Bottom Status Bar:** Shows "Ln 7, Col 45" and "Python".

```
myapp > models.py
1 from django.db import models
2
3 # Create your models here.
4 class student(models.Model):
5     name = models.CharField(max_length=100)
6     number = models.CharField(max_length=100)
7     branch = models.CharField(max_length=100)
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "UNTITLED (WORKSPACE)".
- Editor:** Displays the content of the `admin.py` file.
- Bottom Status Bar:** Shows "Ln 5, Col 29" and "Python".

```
myapp > admin.py
1 from django.contrib import admin
2 from myapp.models import student
3
4 # Register your models here.
5 admin.site.register(student)
```

Step:4 - For execution of the form go to command prompt

type: python manage.py makemigrations

thereafter type: python manage.py migrate

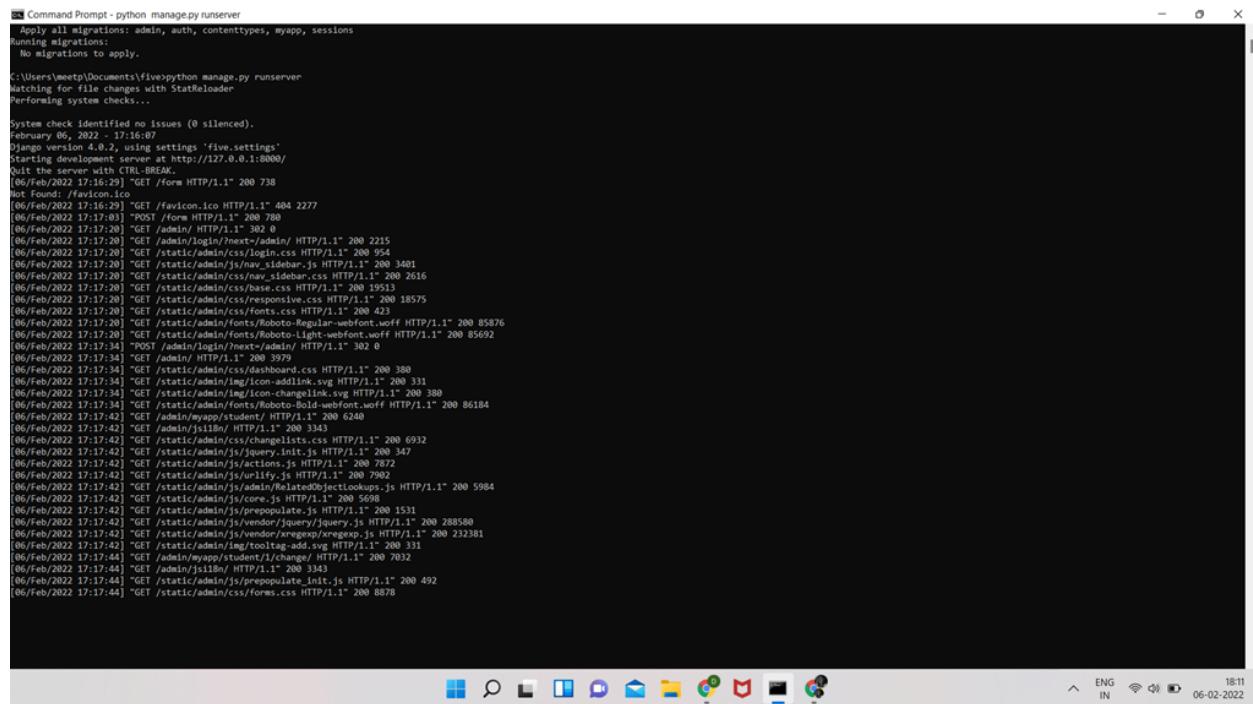
after type:python manage.py runserver

we will get an ip address

copy that and paste it in chrome

add : /form to the address

<http://127.0.0.1:8000/form>



```
C:\ Command Prompt - python manage.py runserver
  Applying all migrations: admin, auth, contenttypes, myapp, sessions
  Running migrations...
    No migrations to apply.

C:\Users\metp\Documents\five>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
February 06, 2022 - 17:16:07
Django version 4.0.2, using settings 'five.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C or -BREAK.
[06/Feb/2022 17:16:29] "GET /form HTTP/1.1" 200 738
Not Found: /favicon.ico
[06/Feb/2022 17:16:29] "GET /favicon.ico HTTP/1.1" 404 2277
[06/Feb/2022 17:17:03] "POST /form HTTP/1.1" 200 780
[06/Feb/2022 17:17:20] "GET /admin/ HTTP/1.1" 302 0
[06/Feb/2022 17:17:20] "GET /admin/login/?next=/admin/ HTTP/1.1" 200 2215
[06/Feb/2022 17:17:20] "GET /static/admin/css/base.css HTTP/1.1" 200 954
[06/Feb/2022 17:17:20] "GET /static/admin/js/nav_sidebar.js HTTP/1.1" 200 3401
[06/Feb/2022 17:17:20] "GET /static/admin/css/nav_sidebar.css HTTP/1.1" 200 2616
[06/Feb/2022 17:17:20] "GET /static/admin/css/base.css HTTP/1.1" 200 19513
[06/Feb/2022 17:17:20] "GET /static/admin/css/responsive.css HTTP/1.1" 200 18575
[06/Feb/2022 17:17:20] "GET /static/admin/fonts/fonts.css HTTP/1.1" 200 50
[06/Feb/2022 17:17:20] "GET /static/admin/fonts/fontawesome-webfont.woff HTTP/1.1" 200 85876
[06/Feb/2022 17:17:20] "GET /static/admin/fonts/Roboto-Light-webFont.woff HTTP/1.1" 200 85692
[06/Feb/2022 17:17:34] "POST /admin/login/?next=/admin/ HTTP/1.1" 302 0
[06/Feb/2022 17:17:34] "GET /admin/ HTTP/1.1" 200 3979
[06/Feb/2022 17:17:34] "GET /static/admin/css/dashboard.css HTTP/1.1" 200 380
[06/Feb/2022 17:17:34] "GET /static/admin/img/icon-oneline.svg HTTP/1.1" 200 331
[06/Feb/2022 17:17:34] "GET /static/admin/img/icon-changelist.svg HTTP/1.1" 200 380
[06/Feb/2022 17:17:34] "GET /static/admin/fonts/Roboto-Bold-webFont.woff HTTP/1.1" 200 86184
[06/Feb/2022 17:17:42] "GET /admin/myapp/student/ HTTP/1.1" 200 6240
[06/Feb/2022 17:17:42] "GET /admin/ji18n/ HTTP/1.1" 200 3343
[06/Feb/2022 17:17:42] "GET /static/admin/css/changelists.css HTTP/1.1" 200 6932
[06/Feb/2022 17:17:42] "GET /static/admin/js/vendor/jquery.js HTTP/1.1" 200 547
[06/Feb/2022 17:17:42] "GET /static/admin/js/vendor/underscore.js HTTP/1.1" 200 7872
[06/Feb/2022 17:17:42] "GET /static/admin/js/vendor/lodash.js HTTP/1.1" 200 7982
[06/Feb/2022 17:17:42] "GET /static/admin/js/admin/RelatedObjectLookups.js HTTP/1.1" 200 5984
[06/Feb/2022 17:17:42] "GET /static/admin/js/core.js HTTP/1.1" 200 5698
[06/Feb/2022 17:17:42] "GET /static/admin/js/repopulate.js HTTP/1.1" 200 1531
[06/Feb/2022 17:17:42] "GET /static/admin/img/icon-oneline.svg HTTP/1.1" 200 288588
[06/Feb/2022 17:17:42] "GET /static/admin/img/icon-changelist.svg HTTP/1.1" 200 232381
[06/Feb/2022 17:17:42] "GET /static/admin/img/tooltag-add.svg HTTP/1.1" 200 331
[06/Feb/2022 17:17:44] "GET /admin/myapp/student/1/change/ HTTP/1.1" 200 7032
[06/Feb/2022 17:17:44] "GET /admin/ji18n/ HTTP/1.1" 200 3343
[06/Feb/2022 17:17:44] "GET /static/admin/js/repopulate_init.js HTTP/1.1" 200 492
[06/Feb/2022 17:17:44] "GET /static/admin/css/forms.css HTTP/1.1" 200 8878
```

^ ENG IN 18:11 06-02-2022

student object (1) | Change student | +

← → ⌂ 127.0.0.1:8000/form

Name:

Number:

Branch:

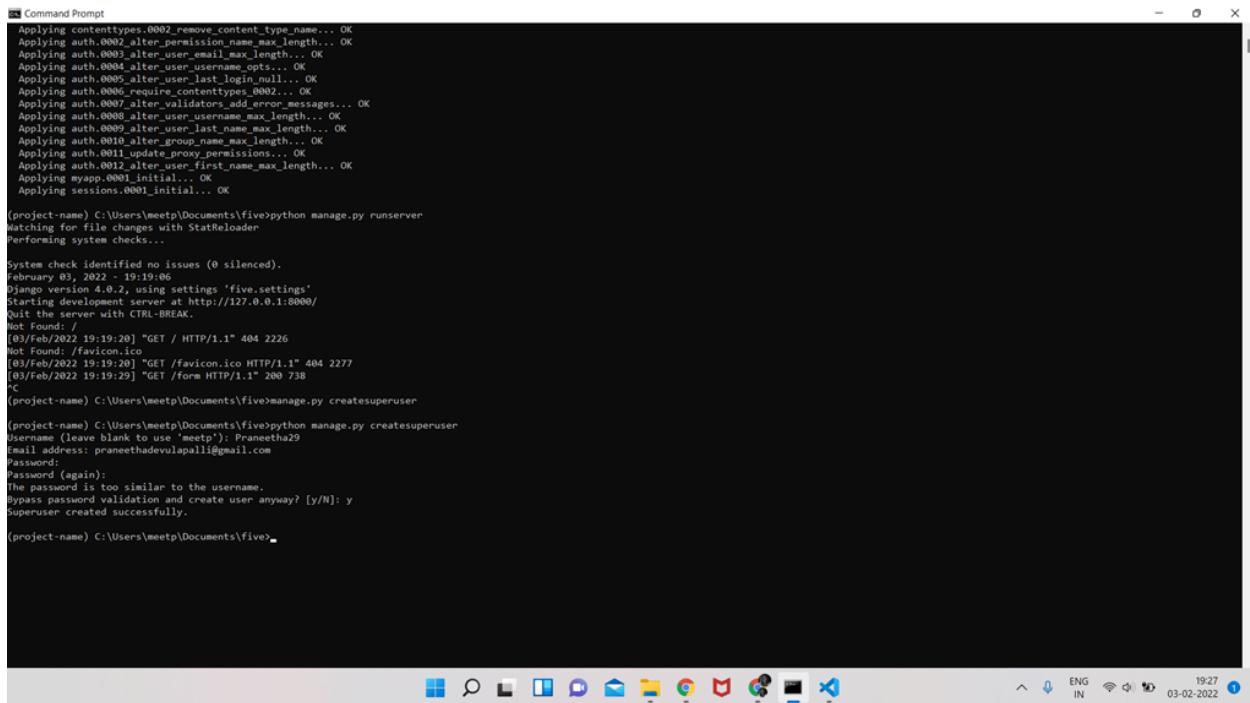
Reading list



Step: 5 - We will get a form like this

thereafter create a user id for django

by typing in cmd prompt : python manage.py createsuperuser



```
Command Prompt
Applying contenttypes.0002_remove_content_type_name... OK
Applying auth.0002_alter_permission_name_max_length... OK
Applying auth.0003_alter_user_email_max_length... OK
Applying auth.0004_alter_user_username_opts... OK
Applying auth.0005_alter_user_last_login_null... OK
Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0007_alterValidators_add_error_messages... OK
Applying auth.0008_alterUser_username_max_length... OK
Applying auth.0009_alterUser_usernamefield_max_length... OK
Applying auth.0010_alter_group_name_max_length... OK
Applying auth.0011_update_proxy_permissions... OK
Applying auth.0012_alterUser_first_name_max_length... OK
Applying myapp.0001_initial... OK
Applying sessions.0001_initial... OK

(project-name) C:\Users\meetp\Documents\five>python manage.py runserver
Watching for file changes with StatReloader...
Performing system checks...

System check identified no issues (0 silenced).
February 03, 2022 - 19:19:06
Django version 4.0.2, using settings 'five.settings'
Starting development server at http://127.0.0.1:8000
Quit the server with CTRL-BREAK.
[03/Feb/2022 19:19:20] "GET / HTTP/1.1" 404 2226
Not Found: /favicon.ico
[03/Feb/2022 19:19:20] "GET /favicon.ico HTTP/1.1" 404 2277
[03/Feb/2022 19:19:29] "GET /form HTTP/1.1" 200 738
^C
(project-name) C:\Users\meetp\Documents\five>python manage.py createsuperuser

(project-name) C:\Users\meetp\Documents\five>python manage.py createsuperuser
Username (leave blank to use 'meetp'): Prameech29
Email address: prameethadevulapalli@gmail.com
Password:
Password (again):
The password is too similar to the username.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.

(project-name) C:\Users\meetp\Documents\five>
```

Step:6 - http://127.0.0.1:8000/admin with/admin

again go to chrome and paste the address

will get a site consist of username and passowrd

will get an Django administration site

login

in student option we can cretae as many as student objects and we can delete too

The screenshot shows the Django Admin interface for a 'student object'. The URL in the browser is 127.0.0.1:8000/admin/myapp/student/1/change/. The page title is 'Django administration'.

The left sidebar shows the 'MYAPP' app with a 'Students' model selected. The main content area is titled 'Change student' and shows a single object named 'student object (1)'. The form fields are:

- Name: Praneetha
- Number: 123
- Branch: IT

At the bottom of the form are three buttons: 'Delete' (red), 'Save and add another' (blue), 'Save and continue editing' (blue), and 'SAVE' (dark blue).

The browser status bar at the bottom indicates it's Google Chrome, and the system tray shows the date and time as 06-02-2022.

WEEK - 11

1. CRUD operations on the form.

CRUD Meaning: CRUD is an acronym that comes from the world of computer programming and refers to the four functions that are considered necessary to implement a persistent storage application: create, read, update and delete.

Letter	Operation	Function
C	Create	Insert
R	Read	Select
U	Update	Edit
D	Delete	Delete

CREATE :

A screenshot of a web browser window. The address bar shows the URL `127.0.0.1:8000/form`. The title bar says "student object (1) | Change student". The page contains a form with three text input fields: "Name:" with placeholder "[]", "Number:" with placeholder "[]", and "Branch:" with placeholder "[]". Below the inputs is a "Submit" button.

Name: []
Number: []
Branch: []
Submit

READ

A screenshot of a web browser window. The address bar shows the URL `127.0.0.1:8000/form`. The title bar says "Select student to change | Django Admin". The page contains a form with three text input fields: "Name:" with value "Praneetha", "Number:" with value "11", and "Branch:" with value "IT". Below the inputs is a "Submit" button.

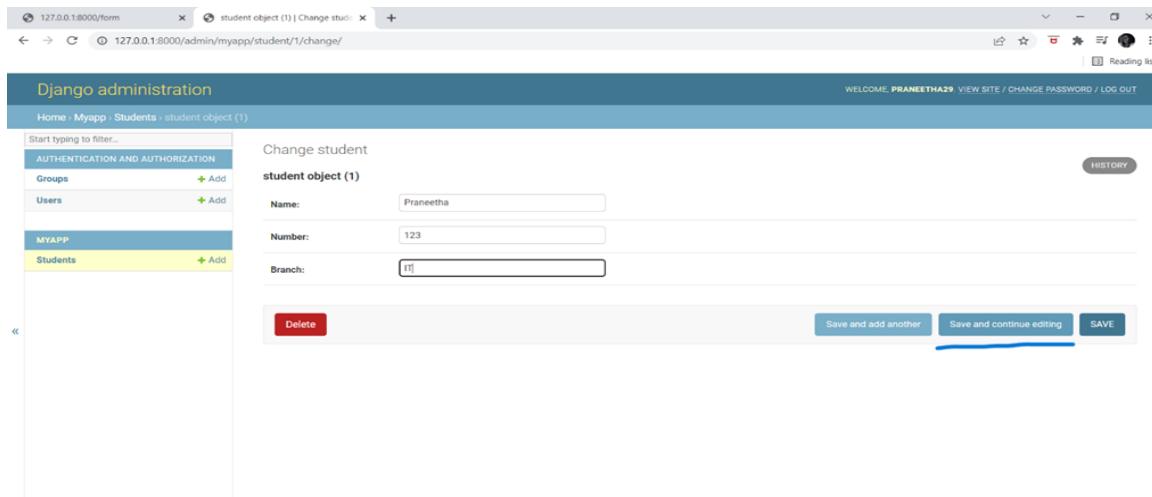
Name: Praneetha
Number: 11
Branch: IT
Submit

UPDATE

Step 1: Select and open one of the objects created.(say student object (1))

The screenshot shows the Django admin interface for a 'student object (1)' named 'Praneetha'. The interface includes a sidebar with 'MYAPP' selected, and the main area displays fields for Name (Praneetha), Number (123), and Branch (CSE). Buttons at the bottom include 'Delete', 'Save and add another', 'Save and continue editing', and 'SAVE'.

Step 2: Select the Save and continue editing button to update the object.



The screenshot shows the same Django admin interface as before, but the 'Save and continue editing' button has been highlighted with a blue underline. The student object 'Praneetha' remains the same, and the interface layout is identical to the first screenshot.

Step 3: The change is updated successfully.

The screenshot shows the Django admin interface for a 'student object' (1). The top navigation bar includes links for 'Home', 'Myapp', 'Students', and 'student object (1)'. A success message at the top right states: 'The student "student object (1)" was changed successfully. You may edit it again below.' The main content area is titled 'Change student' and shows a form for 'student object (1)'. The form fields are: Name (Praneetha), Number (123), and Branch (IT). Below the form are three buttons: 'Delete' (red), 'Save and add another' (blue), 'Save and continue editing' (blue), and 'SAVE' (blue). The left sidebar lists 'Groups' and 'Users' under 'AUTHENTICATION AND AUTHORIZATION', and 'Students' under 'MYAPP'.

DELETE :

Step 1: Select the student object to delete. (say student object(3))

The screenshot shows the Django admin interface for a 'student object' (3). The top navigation bar includes links for 'Home', 'Myapp', 'Students', and 'student object (3)'. The main content area is titled 'Change student' and shows a form for 'student object (3)'. The form fields are: Name (Praneetha Devulapalli), Number (456), and Branch (IT). Below the form are three buttons: 'Delete' (red), 'Save and add another' (blue), 'Save and continue editing' (blue), and 'SAVE' (blue). The left sidebar lists 'Groups' and 'Users' under 'AUTHENTICATION AND AUTHORIZATION', and 'Students' under 'MYAPP'.

Step 2: The student “student object(3)” was deleted successfully.

The screenshot shows the Django administration interface for a 'Students' model. On the left, there's a sidebar with 'AUTHENTICATION AND AUTHORIZATION' (Groups, Users) and 'MYAPP' (Students). The 'Students' section is highlighted. The main area has a heading 'Select student to change'. A success message at the top right says 'The student "student object (3)" was deleted successfully.' Below it, there's a list of students with checkboxes: 'STUDENT', 'student object (7)', 'student object (6)', 'student object (5)', 'student object (4)', and 'student object (1)'. At the bottom, it says '5 students'.

