

# Work Distribution Tree

## Web application

**Summer Project'11  
Programming Club  
Science and Technology Council  
IIT Kanpur**

**By: Deepak Sharma  
Mentored by: Ankesh Kumar Singh**

## 1. Web application

**Definition:** A web application is an application that is accessed over a network such as the Internet or an intranet. The term may also mean a computer software application that is hosted in a browser-controlled environment (e.g. a Java applet) or coded in a browser-supported language (such as JavaScript, combined with a browser-rendered markup language like HTML) and reliant on a common web browser to render the application executable.

**Structure:** Though many variations are possible, the most common structure is the three-tiered application. In its most common form, the three tiers are called presentation, application and storage, in this order. A web browser is the first tier (presentation), an engine using some dynamic Web content technology (such as ASP, ASP.NET, CGI, ColdFusion, JSP/Java, PHP, Perl, Python, Ruby on Rails or Struts2) is the middle tier (application logic), and a database is the third tier (storage). The web browser sends requests to the middle tier, which services them by making queries and updates against the database and generates a user interface.

## 2. Work Distribution Tree

**Introduction:** It is a web application that lets allocation and management of work in a hierarchical organization. Each node can allocate works to his child node depending upon its position in the hierarchy tree. The powers of work allocation is a function of its position in the hierarchy and the position of node to which the work is going to be assigned.

**Structure:** This web application uses a three-tiered structure.

The first tier is the **web browser** which can be any of the default browsers installed on the systems.

The second tier is the logic of the application. It uses general scripting languages like **php, javascript, ajax, css** to produce dynamic web pages.

The third tier is the **database** which is used for storing all the information of nodes and the works. MySQL has been chosen as the Relational Database Management System for the project. The frontend used is phpmyadmin.

### 3. Implementing Logic

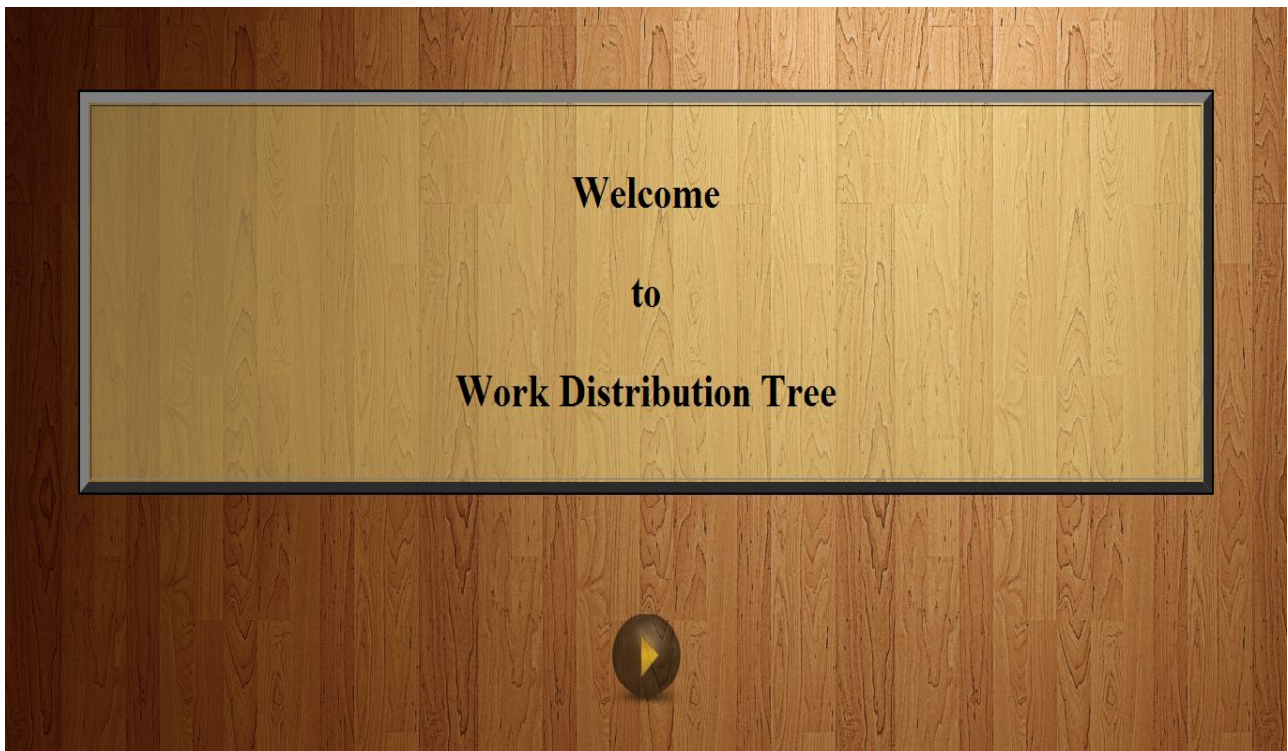
#### Logic at database level:

**Table 'Users'**- It stores the details of all nodes of the tree. It also includes the details of administrator node. A column, called 'Flag', differentiates admin node from other nodes and has value '1' for admin and '0' for others. Each node has a unique id which distinguishes it from other nodes.

**Table 'Work'**- Stores the details of all the works assigned to different nodes. Each record has a serial number, its description, id of the node to whom it is assigned, id of the node by whom it is assigned, deadline, status and remark about the work.

### 4. Developing the GUI

**Welcome Page:** The welcome page is provided with a link to login page.



**Login Page:** User is prompted to enter his login id and password which is submitted to the page itself. A MySQL query is made to the 'Users' table and checked if there is any record present there with that login id and password. If present, the user is directed to its homepage. If not found or in case of wrong password, the user is directed to the page stating 'Record not found' and a link to go to login page again.

**Homepage:** The id of the currently logged in user is stored in a session variable `$_SESSION['UserID']`. If 'Flag' value of this record is zero, the user is interpreted as a general user and directed to its homepage. In case 'Flag' value is 1, the user is interpreted to be administrator and a different homepage with different options, is displayed.



General User Homepage

## Designing of homepage

### 1. General User Homepage: Consists of 5 div elements.

1. Tree div: Cover the right half of the screen and displays the nodes in a hierarchy. The size of div is fixed and the top most node size is equal to the size of tree div. A recursive function is called to display each node (which is displayed as a div with its width and colour as calculated by the function). The function receives an id number and length value as its parameters. It then searches the total number of child nodes of the node( whose id it has received as its argument). The width of a child div is calculated as the length of parent div divided by the number of child divs.

When the div is going to be displayed, it is assigned a colour based on whether it is the div of the node which has logged in, or its child node, or sub-child nodes or other nodes.

2. Header div: It is present at the header of the page and has links for homepage and logout page.

3. Main div: Here the profile of logged in node is displayed with links to the options, 'Edit Profile', 'Add new child', 'Delete child node'.

4. Work div: Displays a table of works assigned to the node who has logged in.

5. Bottom div: Shows its contents when a div of the tree structure is clicked. The profile of the clicked div with a list of options is displayed based on the colour of the div.

Color code '#FFDAB9': It is the div of the currently logged in node. When clicked, an option "View Works" is displayed in the bottom div. Through this link, a node can manage the works assigned to it.

Color code '#F0E68C': It is the child div of the logged in node. When clicked, four options, i.e. "View Works", "Assign New Work", "Delete Work" and "Edit Work" is displayed in the bottom div.

Color code '#FFFF66': It is the node that comes under the child nodes of logged node. When clicked, an option "View Works" is displayed in bottom div.

Color code '#B0B0B0': These are the divs of inaccessible nodes. When clicked, only the profile of that nodes is displayed.

### **How this is implemented?**

Each colour is assigned a numeric code ranging from 1-4. When a div is clicked, the code of its colour is passed to a page 'showbot.php' which then returns the text to be displayed for each type of colour.

## **2. Administrator Homepage:** Consists of 4 div elements.

1. Header div: It is present at the header of the page and has links for homepage and logout page.

2. Tree div: Same procedure is used to display the tree div, except one thing. Now, each of the div is assigned the same colour which is the colour of the child node in the general user homepage. The trick lies here. Now the administrator can view, add, edit and delete the works of all the nodes at any level of tree.

3. Bottom div: Shows its contents when a div of the tree structure is clicked. The profile of the clicked div with a list of options is displayed based on the colour of the div.

4. Main div: It is the div which displays the profile of administrator and provide links for editing his own profile, Creating new node at any level of tree, Deleting nodes and editing nodes profile.



Administrator Homepage

**Edit Profile page:** This page can be accessed from admin as well as general user account. The profile of logged in node is displayed with an option to edit it. When clicked, a javascript code is executed, and the profile is displayed in a form, where it can be edited and saved again. Saving of new details is implemented by making sql queries to the 'Users' table.

**Adding new node:** This page checks whether the user is administrator or general user and displays a different div for each case.

**1. Admin case:** A form is displayed with fields of selecting parent and profile details of the new node. When the form is submitted, a form validation is done and the form is passed to a page which adds that node into the 'Users' table as a new record and automatically assigns a unique UserID to it.

**2. General user case:** A form is displayed with fields to enter the details of the new creating node. On submitting, a form validation is done and the details are passed to a page via POST method. And the record is saved into the 'Users' table.

**Deleting node:** Again a check is made for the admin or general case and a different div is displayed for each case.

**1. Admin case:** A form with a list of all the nodes to be deleted, is displayed. The form is submitted to a php page which uses a recursive function to delete the selected nodes and the tree underneath each of them.

**2. General user case:** A list of child nodes is displayed with the help of HTML form. The filled form is passed to a php code written in a different page, which

delete the selected child nodes and tree underneath each of them.

**Editing profile of any node:** This page can only be accessed from administrator account. A security check is executed (by checking the flag value of the logged user) to make sure that only administrator can access it. It prompts administrator to choose the node whose profile is to be edited. The UserID of the chosen node is passed via URL to next page which displays a form with the profile details of the chosen node. Options of editing profile and changing password are provided. The bottom div displays a form depending on which option is chosen. The details are modified and then saved again in 'Users' table.

**Managing Works:** It gets displayed when the options available on the bottom div of the homepage are clicked. Depending on which colour div is being viewed, different options are displayed.

**1. Logged in div:** A table of works allocated to the node is displayed with links to each work. When a work description is clicked, an option of updating work status is shown. Through this, the node can update the status of the works allocated to it.

**2. Child div:** A table of works allocated to the child is displayed with links to each work, through which the work can be edited or deleted.

A button to add new work is provided which, when clicked, calls a javascript code to display a form in the bottom div. The work details are filled and the work is saved to the 'Work' table.

If an already assigned work is clicked, it shows options for editing and deleting that particular work.

**2. Sub-child div:** A table of works assigned to that particular node is displayed which can only be viewed. There are no options available to edit or delete any work there.

**Logging out:** This page destroys all the session variables, displays a "Successfully logged out" message and a link to go to welcome screen.

**Styling:** A css page prjstyle.php is used to style all the divs and other elements of application. Beside this, individual stylings are also provided to some elements using in pages itself.

## 5. List of pages used

**1. wel.php** /\* Displays the welcome screen \*/

**2. base.php** /\* Makes connection to the database \*/

3. **index.php** /\* Displays the login page. Recieves the login id and password and make query to database \*/
4. **test.php** /\* Displays the homepage \*/
5. **showbot.php** /\* Returns the div contents to test.php \*/
6. **edprof.php** /\* Displays page to edit profile of logged in node \*/
7. **edprofile.php** /\* Saves edited profile to table \*/
8. **add.php** /\* Displays the page to add new node \*/
9. **addacc.php** /\* Add new node to the table \*/
10. **del.php** /\* Displays the page to delete a node \*/
11. **delnode.php** /\* Delete the node from table \*/
12. **worksection.php** /\* Displays the page with list of works \*/
13. **workbot.php** /\* Returns the div contents to worksection.php \*/
14. **ed\_del.php** /\* Displays the page to edit, delete or update a particular work \*/
15. **ed\_del-wrk.php** /\* Apply the changes to selectes work at table level \*/
16. **upwrk.php** /\* Updates work status. Changes the value of status field in Work table \*/
17. **delwrk.php** /\* Delete the selected work from Work table \*/
18. **editwrk.php** /\*Saves the edited work in Work table \*/
19. **edprofadmin.php** /\*Recieves the id of the node to be edited and pass it to editprofileadmin.php \*/
20. **editprofileadmin.php** /\*Receieve the id of node to be edited, displays its profile, and pass edited profile to phplast.php \*/
21. **phplast.php** /\*Recieves edited profile and save it in table 'Users' \*/
22. **logout.php** /\* Destroys session variables and log out the user \*/
23. **prjstyle.css** /\* Applies styling properties to various divs and text, etc. \*/

## 6. Performance

The overall performance of the completed project is satisfactory. The pages are logically linked and provide a smooth interaction with the users. A good design of tree has been implemented which has further options to style it. Form validation codes have been successfully implemented so that proper data is saved into the tables. In short, a good basic design of the application has been built which has lot of options for further modifications and adding of new links.

## **7. Further improvements**

A lot of options are available for further improvements. The external CSS styling file can be modified to style up the whole application in a completely different way. Redundancy of records can be implemented to make the project more efficient and realistic. A mailing system and chat system can also be modulated along with it.

## **8. References**

- Devshed forums
- Google
- W3schools

## **9. Acknowledgements**

- Coordinators, Programming Club
- SnT Council, IITK, and other concerned authorities
- Tobias Ratschiller, creator of the phpmyadmin project