

ASSEMBLE - DBMS SEM4 PROJECT - CEG

CS6106 DATABASE MANAGEMENT SYSTEMS

ASSEMBLE

Platform to Assemble your Team
Skill-based Team Management System.



Creators of ASSEMBLE

Students of CEG, Computer Science Dept.
RED-tags (2023)



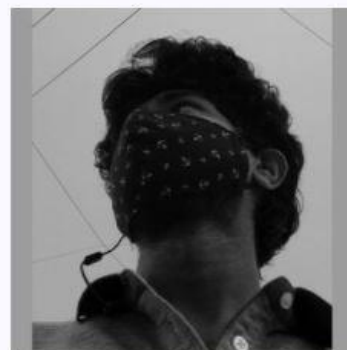
Gokulnath M

2019103522



Thiruchelvan T

2019103591



Jyotir Aditya Giri A

2019103531

ABSTRACT

Our idea is to create a platform where people are recognized by the projects that they build and the skills that they possess. And to enable brilliant like-minded people to collaborate easily. People or groups can find the necessary skill they require with our platform. We allow talented people to be viewed by other users by posting about their projects and their skills and help them find collaborators for their side-projects and works.

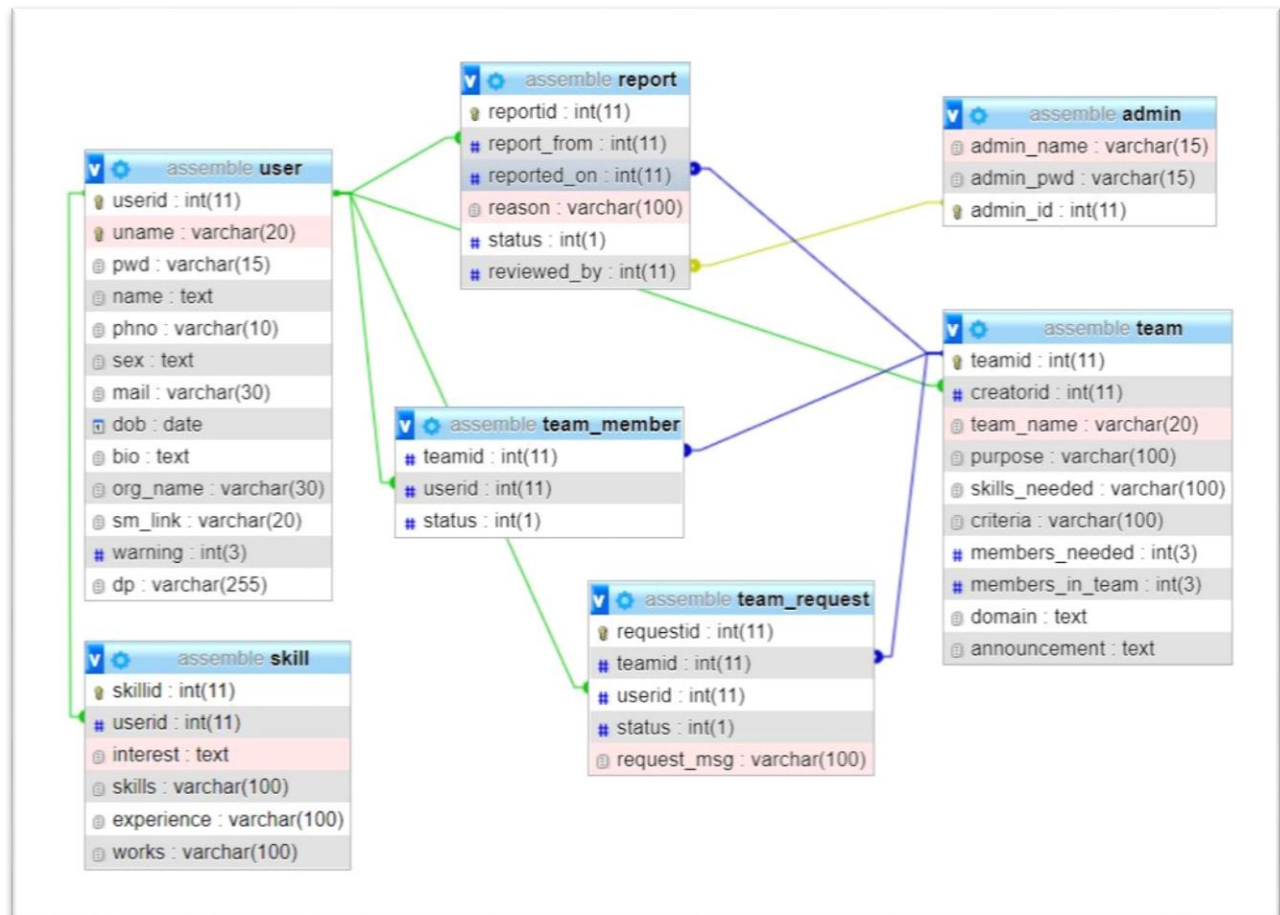
One of the sole barriers to talent identification is not knowing where to search. Whenever small-scale projects happen, people often form teams with known faces and not the right skilled people. Most of the time forming teams with familiar people will end up in a situation where a few members are not interested in the project. The current pandemic showed us that it is possible for humans to stay apart and still innovate through online mode. To adapt to the online mode of networking, ASSEMBLE helps!

ASSEMBLE is the name of the platform we created. It allows users to provide their personal and their skillset information to create their account which can be used by other users to search for the skillset they are looking for. We provide provisions for users to create their own team which can be viewed by others and interested users can send request to join which will be reviewed by the team creator based on their user profile.

Users can also report other users by providing necessary reasons which will be reviewed by the reviewer and the user will be flagged if found guilty. Flagging will appear on the user profile and thus makes the platform to be a safe place to find the right person for your projects.

The platform is built such that team creators can instruct their team members and thus allows communication between users. It primarily focuses on providing services for users to pursue their passion and to do side projects in a more informal way based on pure talent and the spirit of collaboration.

RELATIONAL SCHEMA



SOFTWARE REQUIREMENTS

FRONTEND - HTML, CSS, JQUERY, BOOTSTRAP

BACKEND - PHP

DATABASE - MYSQL

HOSTING - XAMPP

TABLE - USER

STRUCTURE

The screenshot shows the phpMyAdmin interface for the 'assemble' database. The 'Table structure' view for the 'user' table is displayed. The table has 13 columns: userid, uname, pwd, name, phno, sex, mail, dob, bio, org_name, sm_link, warning, and dp. The 'userid' column is the primary key and is an auto-incrementing integer. The 'dp' column is a varchar(255) with a default value of 'avatar.png'.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	userid	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	uname	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
3	pwd	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More
4	name	text	utf8mb4_general_ci		No	None			Change Drop More
5	phno	varchar(10)	utf8mb4_general_ci		No	None			Change Drop More
6	sex	text	utf8mb4_general_ci		No	None			Change Drop More
7	mail	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
8	dob	date			No	None			Change Drop More
9	bio	text	utf8mb4_general_ci		Yes	NULL			Change Drop More
10	org_name	varchar(30)	utf8mb4_general_ci		Yes	NULL			Change Drop More
11	sm_link	varchar(20)	utf8mb4_general_ci		Yes	NULL			Change Drop More
12	warning	int(3)			Yes	0			Change Drop More
13	dp	varchar(255)	utf8mb4_general_ci		No	avatar.png			Change Drop More

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	userid	12	A	No	
Edit Drop	uname	BTREE	Yes	No	uname	12	A	No	

TABLE - SKILL

STRUCTURE

The screenshot shows the phpMyAdmin interface for the 'assemble' database. The 'Table structure' view for the 'skill' table is displayed. The table has 6 columns: skillid, userid, interest, skills, experience, and works. The 'skillid' column is the primary key and is an auto-incrementing integer. The 'userid' column is a foreign key to the 'userid' column in the 'user' table.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	skillid	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	userid	int(11)			No	None			Change Drop More
3	interest	text	utf8mb4_general_ci		No	None			Change Drop More
4	skills	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
5	experience	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
6	works	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	skillid	8	A	No	
Edit Drop	skill_fk_userid	BTREE	No	No	userid	8	A	No	

TABLE - TEAM

STRUCTURE

The screenshot shows the phpMyAdmin interface for the 'assemble' database. The 'team' table structure is displayed in 'Table structure' view. The table has 10 columns: teamid, creatorid, team_name, purpose, skills_needed, criteria, members_needed, members_in_team, domain, and announcement. The primary key is teamid. The table is indexed with a primary index on teamid and a secondary index on creatorid.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	teamid	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	creatorid	int(11)			No	None			Change Drop More
3	team_name	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
4	purpose	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
5	skills_needed	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
6	criteria	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
7	members_needed	int(3)			No	None			Change Drop More
8	members_in_team	int(3)			No	0			Change Drop More
9	domain	text	utf8mb4_general_ci		No	None			Change Drop More
10	announcement	text	utf8mb4_general_ci		Yes	'Further details will be shared soon.'			Change Drop More

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	teamid	15	A	No	
Edit Drop	team_fk_creator	BTREE	No	No	creatorid	15	A	No	

TABLE - TEAM_MEMBER

STRUCTURE

The screenshot shows the phpMyAdmin interface for the 'assemble' database. The 'team_member' table structure is displayed in 'Table structure' view. The table has 3 columns: teamid, userid, and status. The primary key is teamid. The table is indexed with a primary index on teamid and a secondary index on userid.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	teamid	int(11)			No	None			Change Drop More
2	userid	int(11)			No	None			Change Drop More
3	status	int(1)			No	None			Change Drop More

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	member_fk_user	BTREE	No	No	userid	14	A	No	
Edit Drop	member_fk_team	BTREE	No	No	teamid	14	A	No	

Partitions: No partitioning defined

STATUS: 1 - CURRENT MEMBER

STATUS: 3 - REMOVED FROM TEAM

TABLE - TEAM_REQUEST

STRUCTURE

The screenshot shows the phpMyAdmin interface for the 'assemble' database. The 'Table structure' tab is selected for the 'team_request' table. The table has 5 columns:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	requestid	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	teamid	int(11)			No	None			Change Drop More
3	userid	int(11)			No	None			Change Drop More
4	status	int(1)			Yes	0			Change Drop More
5	request_msg	varchar(100)	utf8mb4_general_ci		No	Hey! I'd like to join your team.			Change Drop More

Below the columns, there are sections for 'Indexes' and 'Partitions'. The 'Indexes' section shows three indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	requestid	24	A	No	
Edit Drop	request_fk_user	BTREE	No	No	userid	24	A	No	
Edit Drop	request_fk_team	BTREE	No	No	teamid	24	A	No	

STATUS: 0 - REQUEST PENDING | 1 - REQUEST REJECTED
STATUS: 2 - REQUEST ACCEPTED | 3 - REMOVED FROM TEAM

TABLE - REPORT

STRUCTURE

The screenshot shows the phpMyAdmin interface for the 'assemble' database. The 'Table structure' tab is selected for the 'report' table. The table has 6 columns:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	reportid	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	report_from	int(11)			No	None			Change Drop More
3	reported_on	int(11)			No	None			Change Drop More
4	reason	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
5	status	int(1)			No	None			Change Drop More
6	reviewed_by	int(11)			No	None			Change Drop More

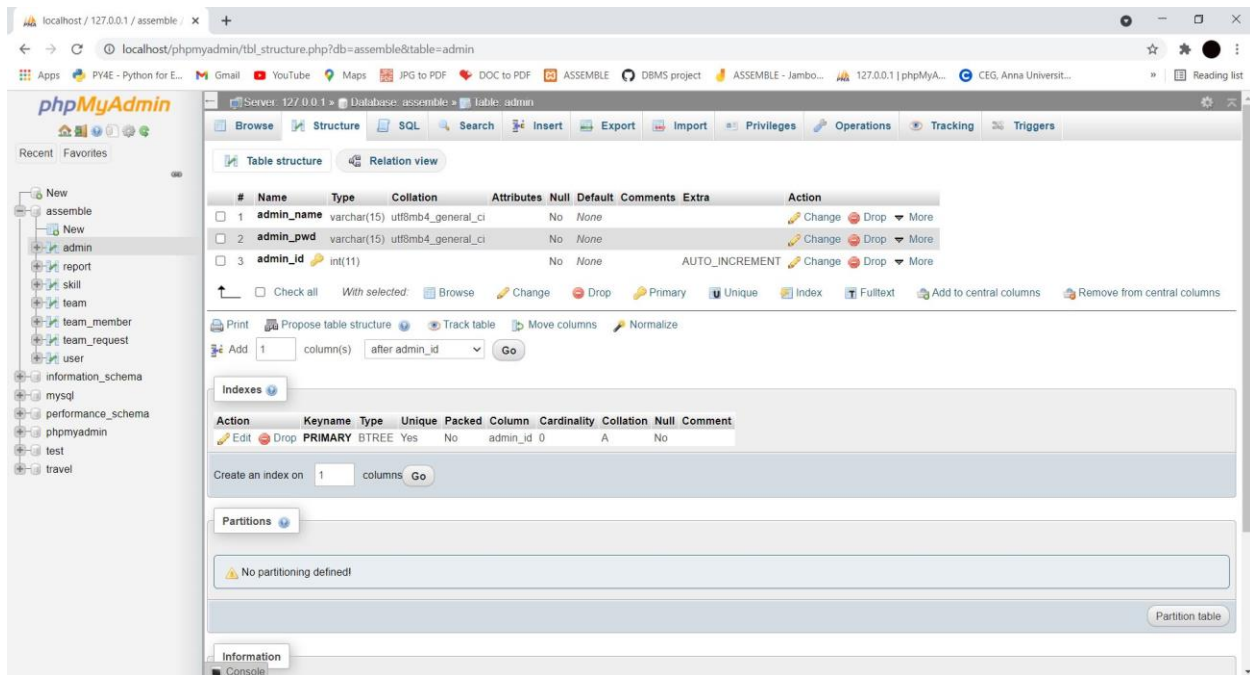
Below the columns, there are sections for 'Indexes' and 'Partitions'. The 'Indexes' section shows four indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	reportid	1	A	No	
Edit Drop	report_fk_from	BTREE	No	No	report_from	1	A	No	
Edit Drop	report_fk_on	BTREE	No	No	reported_on	1	A	No	
Edit Drop	report_fk_by	BTREE	No	No	reviewed_by	1	A	No	

STATUS: 0 - REVIEW PENDING | 1 - REPORT DISMISSED
STATUS: 2 - WARNING GIVEN FOR TEAM CREATOR

TABLE - ADMIN

STRUCTURE

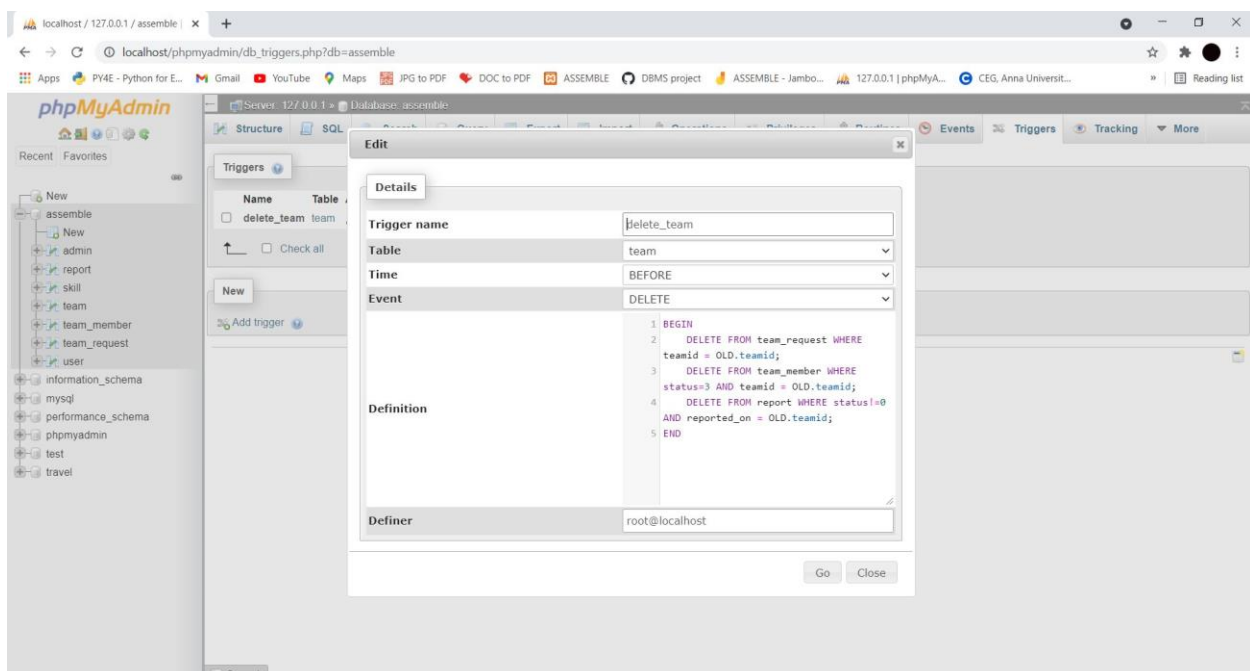


The screenshot shows the phpMyAdmin interface for the 'assemble' database. The 'admin' table structure is displayed in 'Table structure' view. The table has three columns: 'admin_name' (varchar(15)), 'admin_pwd' (varchar(15)), and 'admin_id' (int(11) with AUTO_INCREMENT). The 'admin_id' column is the primary key. The interface also shows options for indexes, partitions, and information.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	admin_name	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More
2	admin_pwd	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More
3	admin_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More

TRIGGER - DELETE_TEAM

When team is deleted, Reviewed reports, Requests to that team And team_member entries who left the team(status=3) will be deleted.



The screenshot shows the phpMyAdmin interface for the 'assemble' database. The 'Triggers' tab is selected, and the 'delete_team' trigger is being edited. The trigger is defined for the 'team' table, triggered BEFORE the DELETE event. The trigger definition is as follows:

```
1 BEGIN
2   DELETE FROM team_request WHERE
   teamid = OLD.teamid;
3   DELETE FROM team_member WHERE
   status=3 AND teamid = OLD.teamid;
4   DELETE FROM report WHERE status!=0
   AND reported_on = OLD.teamid;
5 END
```

The trigger is defined by 'root@localhost'.

TRIGGER - REPORT & LEAVE TEAM (BACKEND)

When a user reports & leaves a team, team_member status updated as 3,
Request to that team is deleted from team_request table,
And, in table team, members_in_team is decremented by 1.

```
$qry = $con->prepare("INSERT INTO report (report_from, reported_on, reason, status, reviewed_by)
VALUES (?, ?, ?, ?, ?);");
$qry->bind_param("iisii", $userid, $teamid, $report, $i, $adminid);

if($qry->execute()){
    $query = $con->prepare("UPDATE team_member SET status = 3 /*Left the team*/
WHERE userid = ? AND teamid = ?;");
    $query->bind_param("ii", $userid, $teamid);

    $query2 = $con->prepare("DELETE FROM team_request WHERE userid = ? AND teamid = ?;");
    $query2->bind_param("ii", $userid, $teamid);

    $query3 = $con->prepare("UPDATE team SET members_in_team = members_in_team - 1
WHERE teamid = ?;");
    $query3->bind_param("i", $teamid);

    if(($query->execute())&&($query2->execute())&&($query3->execute())){
        echo "Reported and Left the team successfully."
    }
}
```

TRIGGER - ACCEPT REQUEST (BACKEND)

Before accepting a request, checks if the user has already been in that team & left
it(status=3 in team_member), if so, updates the status to 1. Else, creates new entry.
Updates the status of request in team_request table as 2.

```
if($row['members_in_team']==$row['members_needed']){
    echo "Team is ASSEMBLED! <br>Increase \"Members Needed\" via \"Edit Details\" <br>to add new members.";
}
else{
    $res1 = mysqli_query($con,"SELECT * FROM user U WHERE uname = '".$uname."'");
    $row1 = $res1->fetch_assoc();
    $uid = $row1["userid"];

    $res = $con->query("SELECT userid FROM team_member WHERE status = 3 /*Left the team*/
AND userid = '".$uid."' AND teamid = '".$teamid."'");
    $count = mysqli_num_rows($res);

    if($count>0){
        $query = "UPDATE team_member SET status = 1 /*Current member*/
WHERE userid = '".$uid."' AND teamid = '".$teamid."'";
    }
    else{
        $query = "INSERT INTO team_member (teamid, userid, status)
VALUES ('".$teamid."','".$uid."',1)";
    }

    $query2 = $con->prepare("UPDATE team_request SET status = 2 /*Accept request*/
WHERE userid = ? AND teamid = ?;");
    $query2->bind_param("ii", $uid, $teamid);

    $query3 = $con->prepare("UPDATE team SET members_in_team = members_in_team + 1
WHERE teamid = ?;");
    $query3->bind_param("i", $teamid);
}
```


TRIGGER - REMOVE MEMBER (BACKEND)

When team creator removes a member, Entry is deleted from team_member table, Status of request to that team in team_request table is updated to 3, And, in table team, members_in_team is decremented by 1.

```
$uid = $row1["userid"];

$query = $con->prepare("UPDATE team_request SET status = 3 /*Removed from team*/
                        WHERE userid = ? AND teamid = ?;");
$query->bind_param("ii", $uid, $teamid);

$query2 = $con->prepare("DELETE FROM team_member WHERE userid = ? AND teamid = ?;");
$query2->bind_param("ii", $uid, $teamid);

$query3 = $con->prepare("UPDATE team SET members_in_team = members_in_team - 1
                        WHERE teamid = ?;");
$query3->bind_param("i", $teamid);

if(($query->execute())&&($query2->execute())&&($query3->execute())){
    echo "Member Removed.";
}
```

TRIGGER - WARN TEAM CREATOR (BACKEND)

When warning is issued, status of report in report table is updated as 2, And, warning for team creator in user table is incremented by 1.

```
$res = $con->query('SELECT * FROM report WHERE reportid = "'.$reportid.'" ');
$row = $res->fetch_assoc();

$res1 = $con->query('SELECT * FROM team T
                    INNER JOIN user U
                    ON T.creatorid = U.userid
                    INNER JOIN skill S
                    ON U.userid = S.userid
                    WHERE teamid = "'.$row["reported_on"].'" ');
$row1 = $res1->fetch_assoc();

$qry2 = $con->query("UPDATE report SET status = 2 WHERE reportid = "'.$reportid.'"");
$qry3 = $con->query("UPDATE user SET warning = warning + 1 WHERE userid = '".$row1['creatorid']."'");

if(($qry2)&&($qry3)){
    echo "Gave Warning.";
}
```

TRIGGER - UNDO REVIEW (BACKEND)

When report review is undone,

If the current report status is 1(report dismissed), status of report in report table is updated as 0,

If the current report status is 2(warning issued), status of report in report table is updated as 0,

And, warning for team creator in user table is decremented by 1.

```
$res = $con->query('SELECT * FROM report WHERE reportid = "'.$reportid.'" ');
$row = $res->fetch_assoc();

$res1 = $con->query('SELECT * FROM team T
                    INNER JOIN user U
                    ON T.creatorid = U.userid
                    INNER JOIN skill S
                    ON U.userid = S.userid
                    WHERE teamid = "'.$row["reported_on"]." ');
$row1 = $res1->fetch_assoc();

if($row['status']==1){
    $qry2 = $con->query("UPDATE report SET status = 0 WHERE reportid = '".$reportid."'");

    if($qry2){echo 'Review Pending';}else{echo "Couldn't take action.".mysql_error($con);}
}
else{
    $qry3 = $con->query("UPDATE report SET status = 0 WHERE reportid = '".$reportid."'");
    $qry4 = $con->query("UPDATE user SET warning = warning - 1 WHERE userid = '".$row1['creatorid']."' ");

    if(($qry3)&&($qry4)){echo 'Review Pending';}else{echo "Couldn't take action.".mysql_error($con);}
}
```