Database Design for Recipe Of the Day

Prepared by Meryem Çanga, Metin Can Kiser, Mehmet Doruk Dinçtürk, Enes Oğuz, and Furkan Zayif

Kadir Has University

November 23, 2024

Tables

We have created eight tables in database according to our system requirements. These tables are users, chefs, recipes, rates, cooktype, diettype, favourites, preptype.

	Tablo	Eylem							Satır	Türü	Karşılaştırma	Boyut	Ek Yük
☐	chefs	☆	Gözat	Yapı	Ara	Ekle	Boşalt	Kaldır	2	InnoDB	utf8mb3_general_ci	32.0 KiB	-
☐	cooktype	☆	Gözat	Yapı	Ara	Ekle	Boşalt	Kaldır	4	InnoDB	utf8mb3_general_ci	16.0 KiB	-
☐	diettype	☆	Gözat	Yapı	Ara	Ekle	Boşalt	Kaldır	9	InnoDB	utf8mb3_general_ci	16.0 KiB	-
☐	favourites	☆	Gözat	Yapı	Ara	Ekle	Boşalt	Kaldır	2	InnoDB	utf8mb3_general_ci	16.0 KiB	-
☐	preptype	☆	Gözat	Yapı	Ara	Ekle	Boşalt	Kaldır	8	InnoDB	utf8mb3_general_ci	16.0 KiB	-
☐	rates	☆	Gözat	Yapı	Ara	Ekle	Boşalt	Kaldır	4	InnoDB	utf8mb3_general_ci	16.0 KiB	-
☐	recipes	☆	Gözat	Yapı	Ara	Ekle	Boşalt	Kaldır	6	InnoDB	utf8mb3_general_ci	16.0 KiB	-
☐	users	☆	Gözat	Yapı	Ara	Ekle	Boşalt	Kaldır	12	InnoDB	utf8mb3_general_ci	48.0 KiB	-
	8 tabloları	Toplam							47	InnoDB	utf8mb3_general_ci	176.0 KiB	0 B

Fig1.1: All Tables

Users Table

This table is holding all the users who are registered in the website. Columns include userID, name, surname, username, email, password, and birthdate. For this phase of the Project, we have not encrypted the users' passwords. But in future phases, the passwords will be held encrypted in the database.

TABLO YAPISI | İLİŞKİ GÖRÜNÜMÜ

	#	Adı	Türü	Karşılaştırma	Öznitelikler	Boş	Varsayılan	Açıklamalar	Ekstra	Eylem
☐	1	userID 🔑	int			Hayır	Yok		AUTO_INCREMENT	Değiştir Kaldır Daha fazla
☐	2	name	varchar(25)	utf8mb3_general_ci		Hayır	Yok			Değiştir Kaldır Daha fazla
☐	3	surname	varchar(25)	utf8mb3_general_ci		Hayır	Yok			Değiştir Kaldır Daha fazla
☐	4	username 🔑	varchar(25)	utf8mb3_general_ci		Hayır	Yok			Değiştir Kaldır Daha fazla
☐	5	email 🔑	varchar(25)	utf8mb3_general_ci		Hayır	Yok			Değiştir Kaldır Daha fazla
☐	6	password	varchar(25)	utf8mb3_general_ci		Hayır	Yok			Değiştir Kaldır Daha fazla
☐	7	birthdate	date			Hayır	Yok			Değiştir Kaldır Daha fazla

Fig1.2: Users Table Structure

Chefs Table

All chefs must be a user. Therefore chefs table holds chefID and a userID for each chef. Additionally, chefs must enter a biography (bio) which will be displayed in their profile, experience years (expYears). Additionally, some chefs are authorized to approve recipes to be displayed in the website. Authorization is being hold in the database as integer and is 0 if the chef is not authorized to review recipes.

			chefID	userID	bio	expYears	auth
☐	Düzenle	Kopyala Sil	1	1	10 yıllık profesyonel şef. Uzmanlık alanı dünya mu...	10	1
☐	Düzenle	Kopyala Sil	2	2	15 yıllık profesyonel şef. Türk mutfağı uzmanı.	15	1

Fig1.3: Chefs Table Structure

Recipes Table

This table holds the necessary information for the recipes. This tables columns are recipeID, title, instruction, createDate, chefID, serving, cookTypeID, prepTypeID, cookTime, dietID, approved. When a chef wants to insert a recipe, an authorized chef must approve before being posted on the website. In order to control this, an approved column is being hold on the table. It's type is small int as it can only have three integer values: 0 as not approved, 1 as waiting for approval, and 2 as it is approved.

	#	Adı	Türü	Karşılaştırma	Öznitelikler	Boş	Varsayılan	Açıklamalar	Ekstra	Eylem
☐	1	recipeID 🔑	int			Hayır	*Yok*		AUTO_INCREMENT	Değiştir Kaldır Daha fazla
☐	2	title	varchar(25)	*utf8mb3_general_ci*		Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	3	instruction	text	*utf8mb3_general_ci*		Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	4	createDate	datetime			Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	5	chefID	int			Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	6	serving	int			Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	7	cookTypeID	int			Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	8	prepTypeID	int			Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	9	cookTime	int			Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	10	dietID	int			Hayır	*Yok*			Değiştir Kaldır Daha fazla
☐	11	approved	smallint			Hayır	*Yok*			Değiştir Kaldır Daha fazla

Fig1.4: Recipes Table Structure

Rates Table

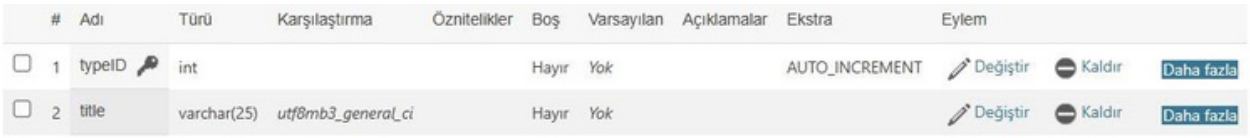
A user can rate a recipe and comment. Although the user must rate the recipe in order to comment to recipe, can only rate the recipe without leaving a comment. Therefore the table holds userID, recipeID, and content. Content can be null and rating is defined as small integer because ratings must be between 1-5.

#	Adı	Türü	Karşılaştırma	Öznitelikler	Boş	Varsayılan	Açıklamalar	Ekstra	Eylem
1	rateID	int			Hayır	*Yok*		AUTO_INCREMENT	Değiştir Kaldır Daha fazla
2	rating	smallint			Hayır	*Yok*			Değiştir Kaldır Daha fazla
3	userID	int			Hayır	*Yok*			Değiştir Kaldır Daha fazla
4	recipeID	int			Hayır	*Yok*			Değiştir Kaldır Daha fazla
5	createTime	datetime			Hayır	*Yok*			Değiştir Kaldır Daha fazla
6	content	text	*utf8mb3_general_ci*		Evet	*NULL*			Değiştir Kaldır Daha fazla

Fig1.5: Rates Table Structure

CookType, PrepType, DietType Tables

These three tables will be used for filtering and categories in the website. Each recipe must have a cooking type, preperation type, and a diet type. At first, these three types were defined as integer values. However, we decided to hold those types as tables because of the fact that new technologies and diet types can be needed in future. These three tables only hold the types' id and their title.



#	Adı	Türü	Karşılaştırma	Öznitelikler	Boş	Varsayılan	Açıklamalar	Ekstra	Eylem
1	typeID	int			Hayır	*Yok*		AUTO_INCREMENT	Değiştir Kaldır Daha fazla
2	title	varchar(25)	*utf8mb3_general_ci*		Hayır	*Yok*			Değiştir Kaldır Daha fazla

Fig1.6: CookType, PrepType, DietType tables structures

Favourites Table

Favourites table holdsthe columns favID, userID, and recipeID. In that, the user can add a recipe to their favourites and lists them in their profiles.

Queries and files

customer.php

Thiscode(customer.php) displays a table of users by generating data from the “users” table.

```
<?php require_once("connect.php");
$GLOBALS['DEBUG_MODE'] = false; $myPage = "
<table border='1'>"; $qHeader = "DESCRIBE users";
$tableStructure = myQuery($qHeader); $myPage .= "
<tr>"; while ($structureLine = $tableStructure-
>fetch_assoc()) {


    $fieldName = $structureLine["Field"];
    $myPage .= "<th>" . htmlspecialchars($fieldName) . "</th>";
}
$myPage .= "</tr>";
$qry = "SELECT * FROM users";
$resultSet = myQuery($qry);
while ($row = $resultSet->fetch_assoc()) {
    $myPage .= "<tr>";
    foreach ($row as $field) {
        $myPage .= "<td>" . htmlspecialchars($field) . "</td>";
    }
    $myPage .= "</tr>";
}
$myPage .= "</table>";
echo $myPage;
echo "<hr>Program finished...";
```

?>

userID	name	surname	username	email	password	birthdate
1	Emin Enes	Oğuz	eminoguz	eminoguz@example.com	password123	1990-01-01
2	Metin Can	Kiser	metinkiser	metinkiser@example.com	password456	1992-02-02
3	Ali	Yılmaz	aliyilmaz	aliyilmaz@example.com	password789	2000-03-03
4	Ayşe	Demir	aysedemir	aysedemir@example.com	password321	1998-04-04
5	Fatma	Çelik	fatmacelik	fatmacelik@example.com	password654	1995-05-05
6	Ahmet	Kaya	ahmetkaya	ahmetkaya@example.com	password987	2002-06-06
7	Merve	Şahin	mervesahin	mervesahin@example.com	password1234	1996-07-07
8	Hakan	Koç	hakankoc	hakankoc@example.com	password5678	1999-08-08
9	Buse	Arslan	busearslan	busearslan@example.com	password1122	1997-09-09
10	Mehmet	Ak	mehmetak	mehmetak@example.com	password3344	1993-10-10
11	Zeynep	Öztürk	zeynepozturk	zeynepozturk@example.com	password5566	1994-11-11
12	Can	Güneş	cangunes	cangunes@example.com	password7788	2001-12-12

Program finished...

Fig2.1: Select * FROM users query

SELECT name, surname, username FROM users ORDER BY username:

Retrieves and displays a list of user details (name, surname, username) with the order of username.

```
<?php
require_once("connect.php");
$GLOBALS['DEBUG_MODE'] = false;
$myPage = "<table border='1'>";
$qHeader = "DESCRIBE users";
$tableStructure = myQuery($qHeader);

$myPage .= "<tr>";
while ($structureLine = $tableStructure->fetch_assoc()) {
    $fieldName = $structureLine["Field"];
    $myPage .= "<th>" . htmlspecialchars($fieldName) . "</th>";
}
$myPage .= "</tr>";
```

```
$qry = "SELECT name, surname, username FROM users ORDER BY username ";
$resultSet = myQuery($qry);
while ($row = $resultSet->fetch_assoc()) {
    $myPage .= "<tr>";
    foreach ($row as $field) {
        $myPage .= "<td>" . htmlspecialchars($field) . "</td>";
    }
    $myPage .= "</tr>";
}
$myPage .= "</table>";
 echo $myPage;
echo "<hr>Program finished...";
?>
```

userID	name	surname
Ahmet	Kaya	ahmetkaya
Ali	Yılmaz	aliyilmaz
Ayşe	Demir	aysedemir
Buse	Arslan	busearslan
Can	Güneş	cangunes
Emin Enes	Oğuz	eminoguz
Fatma	Çelik	fatmacelik
Hakan	Koç	hakankoc
Mehmet	Ak	mehmetak
Merve	Şahin	mervesahin
Metin Can	Kiser	metinkiser
Zeynep	Öztürk	zeynepozturk

Fig2.2: SELECT name, surname, username FROM users ORDERBY username query

connect.php:

This code provides several features such as; the function which is “myQuery” to establish database connection, executing the query and returning the result.

```
<?php
function myQuery($qry)
{

    $servername = "localhost";
    $username = "db_2mdf_usr";
    $password = "1         ,";
    $dbname = "db_2mdf";
    $conn = new mysqli($servername, $username, $password, $dbname);
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }
    $result = $conn->query($qry);
    if ($GLOBALS['DEBUG_MODE']) {
        echo $qry . "<br>";
        echo $conn->error . "<hr>";
    }
    if ($result === FALSE) {
        die("Query failed: " . $conn->error);
    }
    return $result;
}
?>
```

insert.php:

The “insert.php” code responsible for insertion of new user/users into the “users” table with input data.

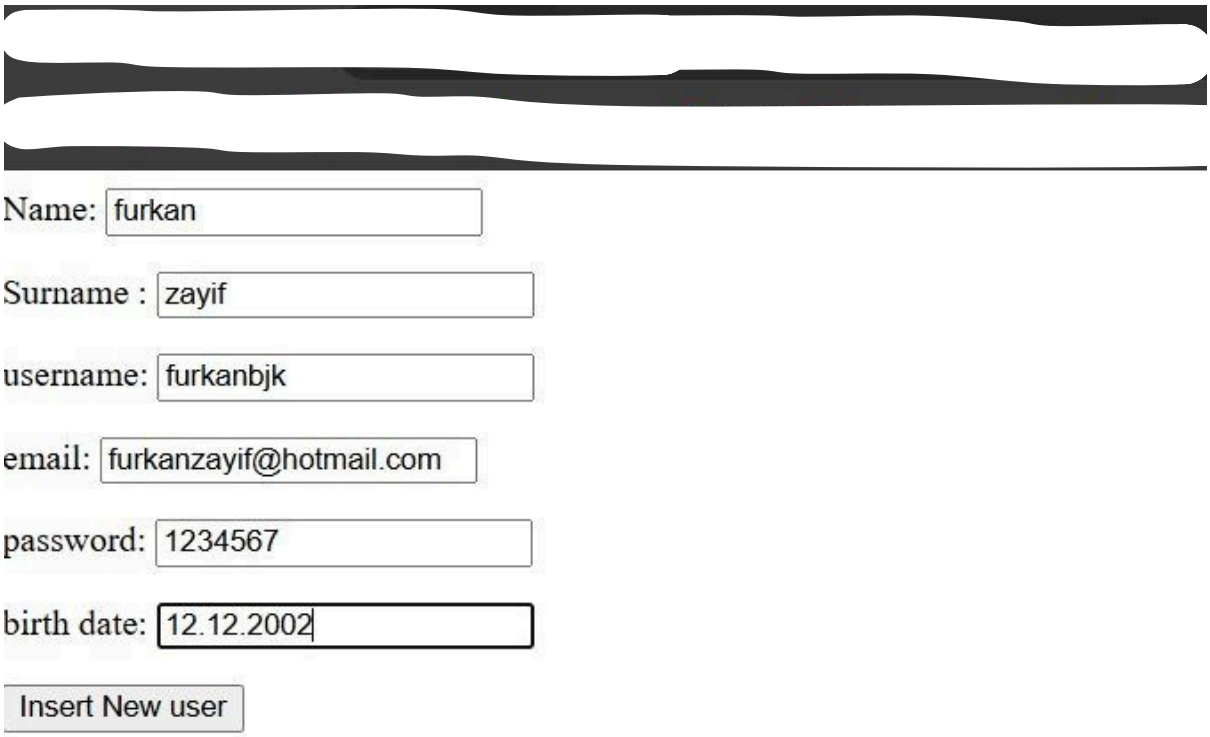


Fig2.3: insert.php page

```
<?php

$DEBUG_MODE=false;
require("connect.php");

if(isset($_POST['submit']))
{
    $name = $_POST['name'];
    $surname = $_POST['surname'];
    $username = $_POST['username'];
    $email = $_POST['email'];
    $password = $_POST['password'];
```

```
    $birthdate=$_POST['birthdate'];


     $q="INSERTINTO users (name,surname,username,email,password,birthdate) VALUES
(".
     "'$name',".
     "'$surname',".
     "'$username',".
     "'$email',".
     "'$password',".
     "'$birthdate'".
     ")";


    //echo $q;
    echo"Query:$q<br>";
    myQuery($q);


    if($DEBUG_MODE) {
        echo$q."<br>";
    }


    $result=mysqli_query($conn, $q);
    if ($result) {
        echo"Courseinserted successfully!";
    } else {
        die("Error:".mysqli_error($conn));
    }
}
else
{
```

```
echo "<form action='insert.php' method='post'>";
echo "<p>Name: <input type='text' name='name'></p>";
echo "<p>Surname : <input type='text' name='surname'></p>";
echo "<p>username: <input type='text' name='username'></p>";
echo "<p>email: <input type='text' name='email'></p>";
echo "<p>password: <input type='text' name='password'></p>";
echo "<p>birth date: <input type='text' name='birthdate'></p>";
echo "<p><input type='submit' name='submit' value='Insert New user'></p>";
echo "</form>";
}
?>
```

update.php:

The “update.php” code updates the information in “users” table and responsible about updating it with the new data which is submitted.

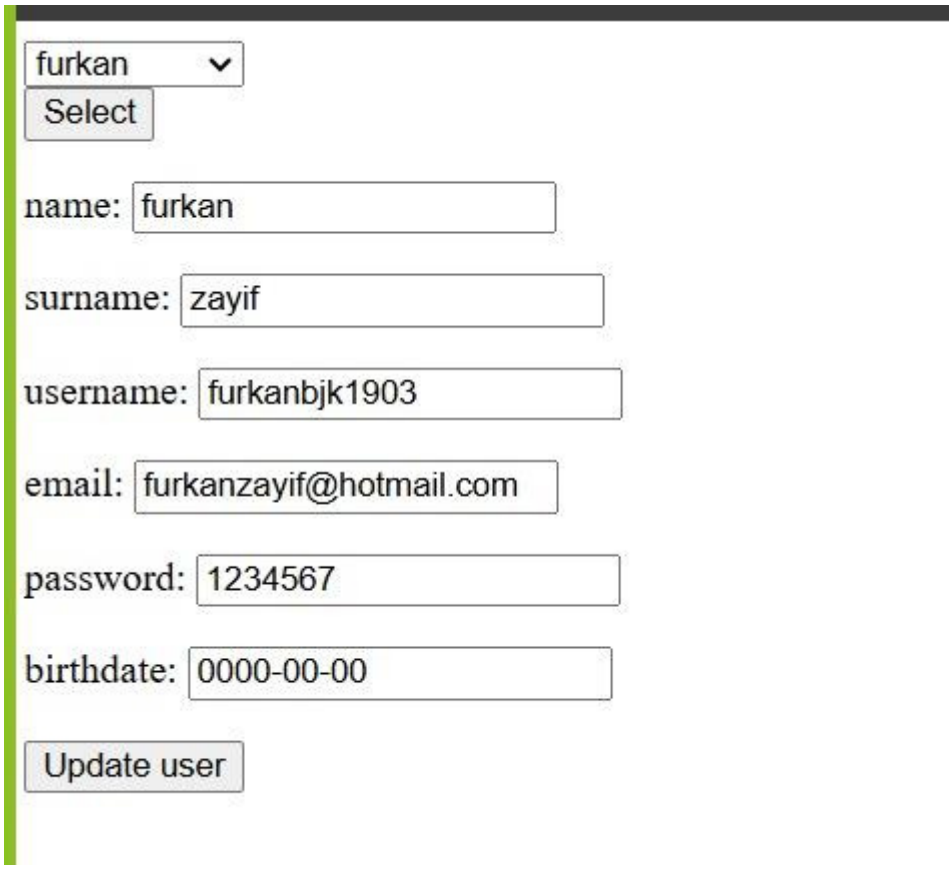


Fig2.4: update.php page

```
<?php

include("connect.php");

if(isset($_POST['submit'])) {
    $userID = $_POST['userID'];
    $name = $_POST['name'];
    $surname = $_POST['surname'];
    $username = $_POST['username'];
    $email = $_POST['email'];
    $password = $_POST['password'];
    $birthdate = $_POST['birthdate'];

    $query = "UPDATE users SET
        name = '$name',
        surname = '$surname',
        username = '$username',
        email = '$email',
        password = '$password',
        birthdate = '$birthdate'
        WHERE userID = $userID";
    if(myQuery($query)) {
        echo "User updated successfully!";
    }else {
        echo "Failed to update user.";
    }
}
```

?>

edit.php:

This code creates a feature for selecting a user, displaying their data and allowing the edit and submission.

```
<?php
$DEBUG_MODE = false;
include("connect.php");
$list = myQuery("SELECT userID, name FROM users ORDER BY name");
echo "<form action=" method='post'>";
echo "<select name='userID'>";
foreach ($list as $record) {
    echo "<option value='" . $record['userID'] . "'>" . $record['name'] . "</option>";
}
echo "</select><br>";
echo "<input type='submit' name='submit' value='Select'/>";
echo "</form>";
if (isset($_POST['userID'])) {
    $userID = $_POST['userID'];
    $result = myQuery("SELECT name, surname, username, email, password, birthdate
FROM users WHERE userID = " . $userID);
    foreach ($result as $record) {
        $name = $record['name'];
        $surname = $record['surname'];
        $username = $record['username'];
        $email = $record['email'];
        $password = $record['password'];
        $birthdate = $record['birthdate'];
    }
    echo "<form action='update.php' method='post'>";
```

```
 echo "<p><input type='hidden' value='" . $userID . "' name='userID'></p>";
    echo "<p>name: <input type='text' value='" . $name . "' name='name'></p>";
    echo "<p>surname: <input type='text' value='" . $surname . "' name='surname'></p>";
    echo "<p>username: <input type='text' value='" . $username . "' name='username'></p>";
    echo "<p>email: <input type='text' value='" . $email . "' name='email'></p>";
    echo "<p>password: <input type='text' value='" . $password . "' name='password'></p>";
    echo "<p>birthdate: <input type='text' value='" . $birthdate . "' name='birthdate'></p>";
    echo "<p><input type='submit' name='submit' value='Update user'></p>";
    echo "</form>";
}
?>
```

delete.php:

This code allows a user to delete their wanted user and additionaly allows a “Confirm Deletion” button for double check.

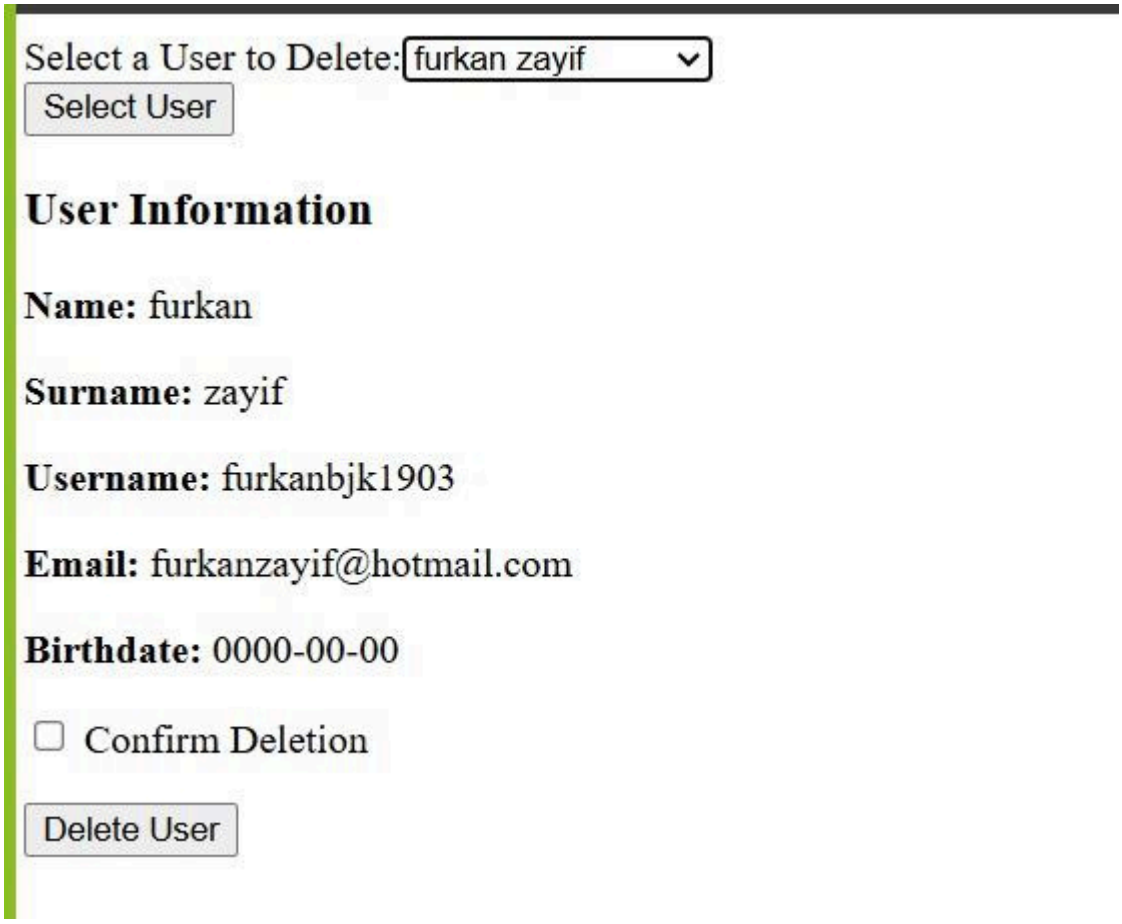


Fig2.5: delete.php page

```php
<?php
$DEBUG_MODE = false; // Enable for debugging information
include("connect.php");

$list = myQuery("SELECT userID, CONCAT(name, ' ', surname) AS fullName FROM users ORDER BY name");
echo "<form action=" method='post'>";
echo "<label for='userID'>Select a User to Delete:</label>";
echo "<select name='userID' id='userID'>";
while ($record = $list->fetch_assoc()) {
    echo "<option value='" . $record['userID'] . "'>" . $record['fullName'] . "</option>";
}
echo"</select><br>";
echo"<input type='submit' name='submit' value='Select User'/>";
echo"</form>";

if(isset($_POST['submit'])) {
    $userID = intval($_POST['userID']); // Sanitize input
    if($DEBUG_MODE) {
        echo "Selected userID: " . $userID . "<br>";
    }

    $result = myQuery("SELECT * FROM users WHERE userID = $userID");
    if($result->num_rows > 0) {
        $record = $result->fetch_assoc();
        $name = $record['name'];
        $surname = $record['surname'];
        $username = $record['username'];
        $email = $record['email'];
```

```
        $birthdate = $record['birthdate'];

        echo "<h3>User Information</h3>";
        echo "<p><strong>Name:</strong> $name</p>";
        echo "<p><strong>Surname:</strong> $surname</p>";
        echo "<p><strong>Username:</strong> $username</p>";
        echo "<p><strong>Email:</strong> $email</p>";
        echo "<p><strong>Birthdate:</strong> $birthdate</p>";
        echo "<form action='execdelete.php' method='post'>";
        echo "<input type='hidden' value='$userID' name='userID'>";
        echo "<p><label><input type='checkbox' name='confirmDelete' value='1'> Confirm
Deletion</label></p>";
        echo "<p><input type='submit' name='submit' value='Delete User'></p>";
        echo "</form>";
    } else {
        echo "<p>No user found with the given ID.</p>";
    }
}
?>
```

execdelete.php:

Deletes a user from the “users” table after the confirmation.

```
<?php
require("connect.php");

if(isset($_POST['submit'])) {
    $userID = intval($_POST['userID']); // Sanitize input

    // Check if deletion is confirmed
```

```
    if(isset($_POST['confirmDelete']) && $_POST['confirmDelete'] == '1') {
        $q="DELETE FROM users WHERE userID = $userID";


        if(myQuery($q)) {
            echo"<p>User successfully deleted.</p>";
        } else {
            echo"<p>Error deleting user. Please try again.</p>";
        }
    } else {
        echo"<p>Deletion not confirmed. User was not deleted.</p>";
    }
}
?>
```