**Dokumentation Webanwendung Urlaubsverwaltung von Herrn Adam, Stümpel u. Wedemeyer**

In dem uns zugeteilten Projekt haben wir für die IT-Solution & Design GmbH ein Programm

zur Urlaubsverwaltung entwickelt.

**Die Musskriterien:**

- Verwaltung der Urlaubstage und –Anträge

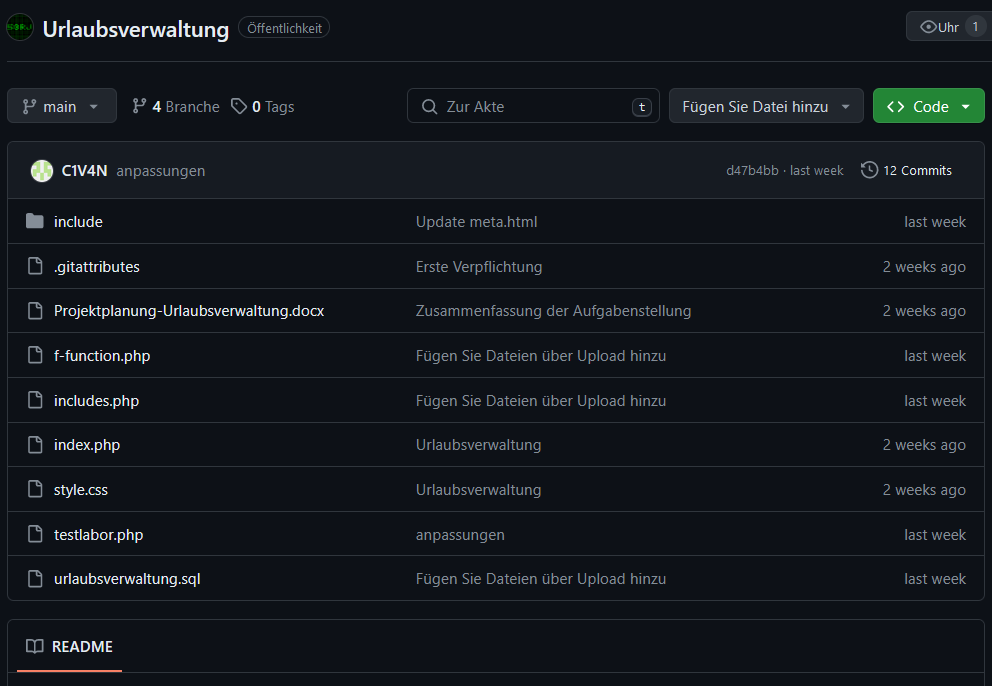
- Zentrale Festlegung von vom Unternehmen festgelegten Urlaubstagen

- Verschiedene Berechtigungsstufen (urlaubswillig, genehmigungs- und verwaltungsberechtigt)

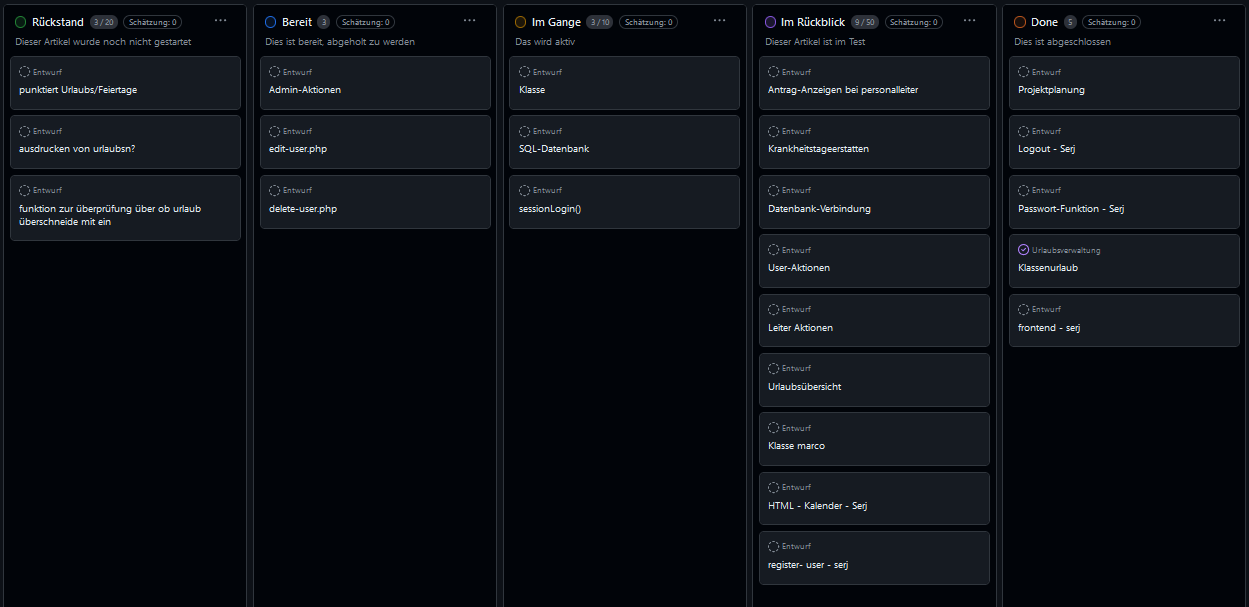
- Anzeige von Resturlaub für Einzelne und alle

- Anzeige von festgelegten Urlaubs- und Feiertagen

Nach Prüfen der Anforderungen, haben wir in einem Brainstorming, die Verteilung der Aufgaben und das Format für die Datenbank besprochen. Für effektives Arbeiten erstellten wir uns einen gemeinsamen Mainbranch. Von diesem hat sich dann jeder ein Repository geklont, um eigenständig arbeiten zu können.



Nach Einrichtung des Arbeitsumfeldes nutzten wir zusätzlich, für die bessere Organisation, ein Kanban board.

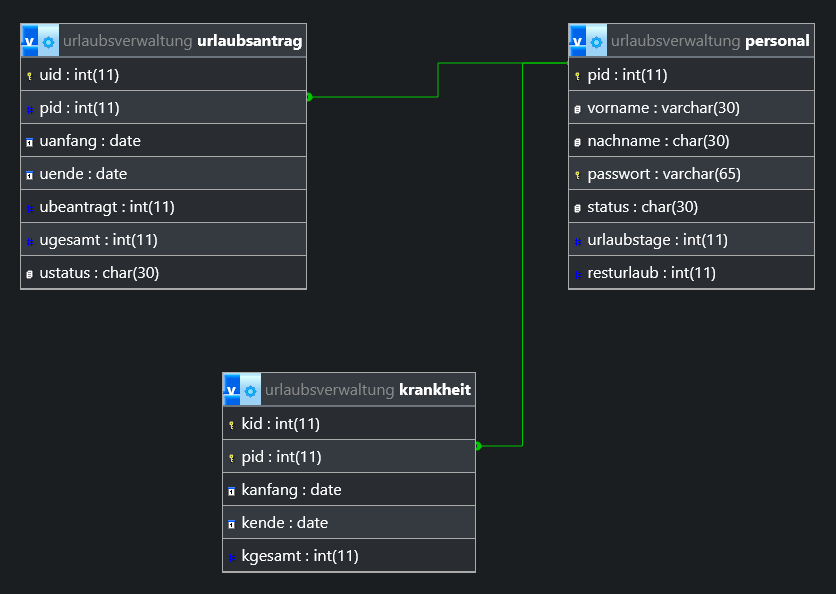


Damit wir die Funktionen und die Webseite von Anfang an korrekt und ohne viele Anpassungen Programmieren können, haben wir mit einem [Entity-Relationship-Modell begonnen.](https://de.wikipedia.org/wiki/Entity-Relationship-Modell) Im Anschluss wurde die Datenbank strukturiert und die Kardinalitäten zugeordnet.

Das Verhältnis von der Tabellen personal zu urlaubsantrag und zu krankheit ist 1 zu N. Aus diesem Grund ist die PID der Personaltabelle als Primary Key (UNIQUE), die PID der urlaubsantragstabelle und der krankheitstabelle als Foreign Key (INDEX) zugeordnet worden.

Die IDs der Tabellen urlaubsantrag (UID) und krankheit (KID) wurden als Primary Key (UNIQUE) gesetzt.

Damit es hier keine Redundanzen gibt sind die UID, KID und PID in der SQL Datenbank auf AUTO INCREMENT eingestellt.



Auf Basis dieser Datenbank sind im Anschluss unsere Funktionen Programmiert worden. Unsere Programmlogik

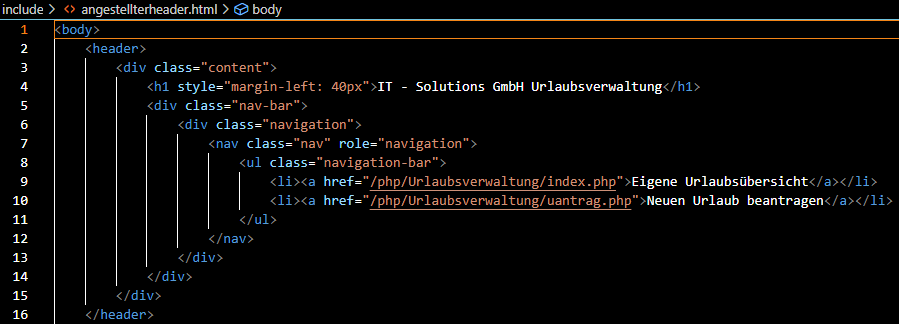
sieht vor, dass je nach Status des Users (Admin, Abteilungsleiter, Angestellter) andere Header inkludiert werden.

Im Beispielbild sehen Sie den Admin und den Angestellten mit den links zu den dazugehörigen Headerdatein.

**Admin:**



**Angestellter:**



**Für unser Programm wurden die folgenden Funktionen verwendet:**

Die Funktion addUser kann ausschließlich vom Admin für die Erstellung eines neuen Users benutzt werden und gibt je nach Fall eine Rückmeldung ob der Vorgang erfolgreich war.

function addUser($pdo, $vorname, $nachname, $status, $passwort){

    $statement = $pdo->prepare("INSERT INTO personal (vorname, nachname, status, passwort, urlaubstage, resturlaub) VALUES (?, ?, ?, ?, 30, 30)");

    $statement->execute([$vorname, $nachname, $status, $passwort]);

    $pid = $pdo->lastInsertId();

    if ($pid) {

        echo "Registrierung des Users mit der ID: $pid erfolgreich ausgeführt.";

    } else {

        echo "Es gab ein Problem bei der Registrierung des Users.";

    }

}

Die Funktion connServer stellt die Verbindung zum Server her, durch try and catch wird bei einem Fehlschlag eine Rückmeldung erzeugt.

function connServer(){

    try {

        $pdo = new PDO('mysql:host=localhost;dbname=urlaubsverwaltung', 'root', '');

        $pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

        return $pdo;

    } catch (PDOException $e) {

        error\_log("Verbindung fehlgeschlagen: " . $e->getMessage());

        return null;

    }

}

Die Funktion schowAllData differenziert über die tableMap welche Tabelle angesprochen werden soll und liefert eine Liste der enthaltenen Datensätze zurück.

function showAllData($pdo, $zahl){

    try {

        $tableMap = [

            1 => "personal",

            2 => "urlaubsantrag",

            3 => "krankheit"];

        if (array\_key\_exists($zahl, $tableMap)) {

            $sql = "SELECT \* FROM " . $tableMap[$zahl];

            $stmt = $pdo->prepare($sql);

            $stmt->execute();

            return $stmt->fetchAll(PDO::FETCH\_ASSOC);

        } return [];

    } catch (PDOException $e) {

        error\_log("Datenbankfehler: " . $e->getMessage());

        return [];

    }

}

Die Funktion login startet eine Session und nimmt die PID und das Passwort vom User entgegen. Im Anschluss wird in der Datenbank geprüft, welchen Status (Admin, Abteilungsleiter oder Angestellter) der User hat, und demensprechend der zugehörige header inkludiert.

function login(){

    session\_start();

    $pid = $\_POST["pid"];

    $passwort = $\_POST["passwort"];

    $conn = connServer();

    $\_SESSION["status"] = "SELECT personal.status FROM personal WHERE pid = $pid AND passwort = $passwort";

    if ($\_SESSION["status"] == "Angestellter") {

        $\_SESSION['logged\_in'] = true;

        include "include/angestellterheader.html";

    }

    if ($\_SESSION["status"] == "Abteilungsleiter") {

        $\_SESSION['logged\_in'] = true;

        include "include/abteilungsleiterheader.html";

    }

    if ($\_SESSION["status"] == "Admin") {

        $\_SESSION['logged\_in'] = true;

        include "include/adminheader.html";

    }

}

Die Funktion nichtVerplanteUrlaubstage prüft die Session ID, stellt eine verbindung zur Datenbank her und gibt die verbleibenden Urlaubstage für den User aus.

function nichtVerplanteUrlaubstage()

{

    $pid = $\_SESSION['personal\_id'];

    $conn = connServer();

    $sql = "SELECT p.resturlaub

            FROM personal p

            WHERE p.pid = $pid";

    $stmt = $conn->query($sql);

    $result = $stmt->fetch(PDO::FETCH\_ASSOC); // Fetch als assoziatives Array

    $resturlaub = $result['resturlaub']; // Zugriff auf das spezifische Feld

    echo "<p>" . $resturlaub . "</p>";

}

Die Funktion checkStatus prüft ob die Session Variablen (Status) gesetzt wurde und inkludiert den Richtigen header. Sollte dies nicht der Fall sein, erfolgt eine Weiterleitung auf die login.php Seite.

function checkStatus(){

    if (isset($\_SESSION['status'])) {

        switch ($\_SESSION['status']) {

            case "Angestellter":

                include "include/angestellterheader.html";

                break;

            case "Abteilungsleiter":

                include "include/abteilungsleiterheader.html";

                break;

            case "Admin":

                include "include/adminheader.html";

                break;}

    } else {

        if (!isset($\_SESSION['attempted\_redirect'])) {

            $\_SESSION['attempted\_redirect'] = true;

            header("Location: login.php");

            exit;

        } else {

            session\_unset();

            session\_destroy();

            header("Location: login.php");

            exit;}

    }

}

Die Funktion freigabenUrlaub bekommt eine Variable übergeben, die den Status (genehmigt, beantragt, abgelehnt) enthält. Im Anschluss werden die Datensätze in einer jeweiligen Tabelle spezifisch für den User aufgelistet.

function freigabenUrlaub($wert)

{

$pid = $\_SESSION['personal\_id'];

$conn = connServer();

$sql = "SELECT u.uanfang, u.uende, u.ubeantragt

FROM personal p, urlaubsantrag u

WHERE p.pid = $pid

AND u.pid = p.pid

AND u.ustatus = '$wert'";

$genehmigteUrlaube = $conn->query($sql);

$anzeige = array();

while ($row = $genehmigteUrlaube->fetch(PDO::FETCH\_ASSOC)) {

$anzeige = $row;

echo "<tr>";

foreach ($anzeige as $kommende\_urlaube) {

echo "<td>" . $kommende\_urlaube . "</td>";

}

echo "</tr>";

}

}

Die Funktion urlaubsantrag nimmt die User PID, den urlaubsanfang und das urlaubsende entgegen. Im Anschluss wird eine temporäre Tabelle erzeugt, die die Differenz von uanfang / uende berechnet und die Wochenenden ausschließt. Am ende wird der vollständige Datensatz in die Haupttabelle der SQL Datenbank geschrieben und die Temporäre Tabelle gelöscht.

function urlaubsantrag($pid, $urlaubsanfang, $urlaubsende)

{

if ($urlaubsanfang < $urlaubsende) {

$pdo = connServer();

$stmt = $pdo->prepare("CREATE TEMPORARY TABLE IF NOT EXISTS TempUrlaub (uanfang DATE, uende DATE, tage INT);

-- Berechnung und Einfügen der Urlaubstage unter Ausschluss der Wochenenden

INSERT INTO TempUrlaub (uanfang, uende, tage)

VALUES (:uanfang, :uende,

(DATEDIFF(:uende, :uanfang) + 1

- ((DATEDIFF(:uende, :uanfang) + 1) / 7 \* 2)

+ (CASE WHEN WEEKDAY(:uanfang ) = 6 THEN 1 ELSE 0 END)

+ (CASE WHEN WEEKDAY(:uende) = 5 THEN 1 ELSE 0 END)));

-- Daten von der temporären Tabelle in die Haupttabelle übertragen

INSERT INTO urlaubsantrag (pid, uanfang, uende, ubeantragt, ugesamt, ustatus)

SELECT :pid, uanfang, uende, tage, 30, 'beantragt'

FROM TempUrlaub;

-- Lösche die temporäre Tabelle nach Gebrauch

DROP TEMPORARY TABLE IF EXISTS TempUrlaub;");

$stmt->bindParam(":pid", $pid);

$stmt->bindParam(":uanfang", $urlaubsanfang);

$stmt->bindParam(":uende", $urlaubsende);

$stmt->execute();

echo "Urlaubsantrag erfolgreich eingetragen!";

} else {

echo "Der Urlaubsantrag konnte nicht eingetragen werden.";

}

}

Die Funktion krankheitseintrag nimmt die User PID, den krankheitsanfang und das krankheitsende entgegen. Im Anschluss wird eine temporäre Tabelle erzeugt, die die Differenz von kanfang / kende berechnet und die Wochenenden ausschließt. Der Datensatz wir dann in die Haupttabelle krankheit eingetragen. Des weiteren wird in einer SELECT Abfrage geprüft, ob es Überschneidungen von genehmigtem Urlaub und Krankheitstagen gibt, welche dann rückerstattet werden. Zum Schluss wird die temporäre Tabelle wieder gelöscht.

function krankheitseintrag($pid, $krankheitsanfang, $krankheitsende)

{

if ($krankheitsanfang < $krankheitsende) {

$pdo = connServer();

$stmt = $pdo->prepare("CREATE TEMPORARY TABLE IF NOT EXISTS TempAU (kanfang DATE, kende DATE, tage INT);

-- Berechnung und Einfügen der Krankheitstage unter Ausschluss der Wochenenden

INSERT INTO TempAU (kanfang, kende, tage)

VALUES (:kanfang, :kende,

(DATEDIFF(:kende, :kanfang) + 1

- ((DATEDIFF(:kende, :kanfang) + 1) / 7 \* 2)

+ (CASE WHEN WEEKDAY(:kanfang ) = 6 THEN 1 ELSE 0 END)

+ (CASE WHEN WEEKDAY(:kende) = 5 THEN 1 ELSE 0 END)));

-- Daten von der temporären Tabelle in die Haupttabelle übertragen

INSERT INTO krankheit (pid, kanfang, kende, kgesamt)

SELECT :pid, kanfang, kende, tage

FROM TempAU;

-- Erstattet die Krankheitstage dem Resturlaub wieder gut

UPDATE personal

SET resturlaub = resturlaub + (

SELECT SUM(DATEDIFF(LEAST(k.kende, u.uende), GREATEST(k.kanfang, u.uanfang)) + 1)

FROM krankheit k

JOIN urlaubsantrag u ON k.pid = u.pid

WHERE k.kanfang <= u.uende

AND k.kende >= u.uanfang

AND k.pid = :pid)

WHERE pid = :pid;

-- Lösche die temporäre Tabelle nach Gebrauch

DROP TEMPORARY TABLE IF EXISTS TempAU;");

$stmt->bindParam(":pid", $pid);

$stmt->bindParam(":kanfang", $krankheitsanfang);

$stmt->bindParam(":kende", $krankheitsende);

$stmt->execute();

echo "Krankheitsantrag erfolgreich eingetragen!";

} else {

echo "Der Krankheitseintrag konnte nicht eingetragen werden.";

}

}

Die Funktion urlaubsgenehmigung nimmt die PID des Users, die Urlaub ID (UID) entgegen und setzt den ustatus auf genehmigt. Diese Funktion wird nur vom Abteilungsleiter benutzt.

function urlaubsgenehmigung($pid, $uid)

{

$pdo = connServer();

$stmt1 = $pdo->prepare(

"UPDATE urlaubsantrag SET ustatus = 'genehmigt' WHERE uid = :uid;

UPDATE personal SET resturlaub = resturlaub - (SELECT ubeantragt FROM urlaubsantrag WHERE pid = :pid AND uid = :uid)

WHERE pid = :pid"

);

$stmt1->bindParam(":pid", $pid);

$stmt1->bindParam(":uid", $uid);

$stmt1->execute();

}

Die Funktion urlaubsablehnung nimmt die Urlaub ID (UID) entgegen und setzt den ustatus auf abgelehnt. Diese Funktion wird nur vom Abteilungsleiter benutzt.

function urlaubsablehnung($uid)

{

$pdo = connServer();

$stmt1 = $pdo->prepare("UPDATE urlaubsantrag SET ustatus = 'abgelehnt' WHERE uid = :uid");

$stmt1->bindParam(":uid", $uid);

$stmt1->execute();

}

Die Funktion kommende\_urlaube\_anzeigen prüft nach Verbindung mit der SQL Datenbank

welche Urlaube auf den Status beantragt gesetzt sind.

Diese werden dann von einer Schleife Datensatzweise in einer Tabelle ausgegeben.

function kommende\_urlaube\_anzeigen()

{

$conn = connServer();

$sql = "SELECT u.uid, p.vorname, p.nachname, u.uanfang, u.uende

FROM personal p

JOIN urlaubsantrag u ON u.pid = p.pid

WHERE u.ustatus = 'beantragt' AND u.uende > CURRENT\_DATE()";

$urlaube = $conn->prepare($sql);

$urlaube->execute();

while ($row = $urlaube->fetch(PDO::FETCH\_ASSOC)) {

echo "<tr>";

foreach ($row as $key => $value) {

if ($key != 'uid') {

echo "<td>" . $value . "</td>";

}

}

echo '<td>

<form method="POST">

<input type="hidden" name="uid" value="' . ($row['uid']) . '">

<button type="submit" class="btn-allow" name="uannehmen">Annehmen</button>

<button type="submit" class="btn-deny" name="uablehnen">Ablehnen</button>

</form>

</td>';

echo "</tr>";

}

}