Elektroniczna przychodnia lekarska (PHP, MYSQL)

Paweł Wilczek Łukasz Szkaradek Łukasz Mamak

Zespołowe przedsięwzięcie inżynierskie
Informatyka
Rok. akad. 2017/2018, sem. I
Prowadzący: dr hab. Marcin Mazur

Spis treści

1 Opis projektu

1.1 Członkowie zespołu

1. Łukasz Szkaradek (kierownik projektu).
2. Paweł Wilczek.
3. Łukasz Mamak.

1.2 Cel projektu (produkt)

Celem projektu jest stworzenie strony internetowej z zastosowaniem php i mysql, która ma na celu obsługę wirtualnej przychodni lekarskiej oraz zwiększenie jej wydajności.

1.3 Potencjalny odbiorca produktu (klient)

Klientem może być przychodnia lekarska chcąca usprawnić wydajność niskim kosztem.

1.4 Metodyka

Projekt będzie realizowany przy użyciu (zaadaptowanej do istniejących warunków) metodyki *Scrum*.

2 Wymagania użytkownika

2.1 User story 1

Jako pacjent, chciałbym mieć możliwość zapisu do danego specjalisty za pośrednictwem strony internetowej przychodni, dzięki czemu nie musiałbym udać się tam osobiście i mógłbym przez to lepiej zarządzać czasem jaki mam do dyspozycji danego dnia.

2.2 User story 2

Jako pacjent chciałbym mieć wgląd do mojej karty pacjenta poprzez stronę przychodni tak abym mógł łatwo sprawdzić aktualną listę moich wizyt z lekarzem.

2.3 User story 3

Jako dyrektor placówki chciałbym posiadać konto admina by zarządzać elektroniczną przychodnią tak, aby w pełni czuwać nad moimi pracownikami.

2.4 User story 4

Jako lekarz chce żeby oprogramowanie przychodni miało łatwy i szybki dostęp do wszystkich moich aktualnych wizyt.

2.5 User story 5

Jako recepcjonista w przychodni chciałbym mieć dostęp do elektronicznej bazy przychodni co w przeciwieństwie do tradycyjnej formy skróciło by mój czas reakcji i zmniejszyło ilość mojej pracy.

2.6 User story 6

Jako pacjent chciałbym mieć dostęp do aktualności w przychodni takie jak opóźnienia w przyjmowaniu przez lekarzy, tak aby w takim przypadku inaczej zagospodarować czas.

2.7 User story 7

Jako dyrektor chciałbym, aby platforma internetowa zwiększyła wydajność naszej placówki poprzez łatwość w rejestracji i katalogowaniu danych.

2.8 User story 8

Jako lekarz chciałbym mieć dostęp do elektronicznej wersji karty pacjenta co umożliwiało by mi poznanie historii choroby pacjenta i jej edycje w razie konieczności.

3 Harmonogram

3.1 Rejestr zadań (Product Backlog)

- Data rozpoczęcia: 11.10.2017.
- Data zakończenia: 15.11.2017.

3.2 Sprint 1

- Data rozpoczęcia: 15.11.2017.
- Data zakończenia: 29.11.2017.
- Scrum Master: Łukasz Szkaradek.
- Product Owner: Łukasz Mamak.
- Development Team: Łukasz Mamak, Paweł Wilczek, Łukasz Szkaradek.

3.3 Sprint 2

- Data rozpoczęcia: 29.11.2017.
- Data zakończenia: 27.12.2017.
- Scrum Master: Łukasz Mamak.
- Product Owner: Paweł Wilczek.
- Development Team: Paweł Wilczek, Łukasz Szkaradek, Łukasz Mamak.

3.4 Sprint 3

- Data rozpoczęcia: 27.12.2017.
- Data zakończenia: 24.01.2018.
- Scrum Master: Paweł Wilczek.
- Product Owner: Łukasz Szkaradek.
- Development Team: Łukasz Szkaradek, Łukasz Mamak, Paweł Wilczek.

4 Product Backlog

4.1 Backlog Item 1

Tytuł zadania. Stworzenie bazy danych.

Opis zadania. Stworzenie bazy danych, która służyć będzie do testowania początkowych zadań.

Priorytet. Średni.

Definition of Done. Stworzenie w MYSQL bazy danych i tablic:

SPECJALIZACJE:

- id specjalizacji (automatycznie przypisywany numer),
- nazwa specjalizacji (nazwa specjalizacji),

AKTUALNOŚCI:

- id aktualnosci (automatycznie przypisywany numer),
- data (data napisania aktualności) ,
- opis (treść aktualności)

PACJENCI:

- id pacjenta (automatycznie przypisywany numer),
- imie (imię pacjenta),
- nazwisko (nazwisko pacjenta),
- pesel (pesel pacjenta),
- karta pacjenta (treść karty pacjenta),
- nr telefonu
- kod (automatycznie generowany kod, który wraz z peselem będzie umożliwiał logowanie się pacjenta)

SPOTKANIA:

- id spotkania (automatycznie przypisywany numer),
- id specjalizacji (numer odnoszący się do konkretnego wiersza w tablicy specjalizacje),
- id lekarza (numer odnoszący się do konkretnego wiersza w tablicy lekarze),
- id osoby (numer odnoszący się do konkretnego wiersza w tablicy pacjenci),
- data odbycia (data wpisywana przez recepcjonistę, która określa datę spotkania z lekarzem),
- data zapisu (automatycznie przypisywana data podczas tworzenia wiersza),
- stan (równa się 0 lub 1, w zależności czy spotkanie zostało zaakceptowane przez recepcjonistę)

LEKARZE:

- id lekarza (automatycznie przypisywany numer),
- login (login używany przy logowaniu na stronie recepcji),
- haslo (hasło używane przy logowaniu na stronie recepcji),
- imie (imię lekarza),
- nazwisko (nazwisko lekarza),
- id specjalizacji (numer odnoszący się do konkretnego wiersza w tablicy specjalizacje, określa jaką specjalizację posiada dany lekarz),
- nr pokoju (numer pokoju w którym dany lekarz przyjmuje),

4.2 Backlog Item 2

Tytuł zadania. Strona tytułowa oraz formularz logowania pacjenta.

Opis zadania. Stworzenie strony głównej wraz z interfejsem i logowaniem dla pacjenta.

Priorytet. Średni.

Definition of Done. Napisanie w języku PHP strony głównej w której będą zawarte:

1. Formularz logowania pacjenta (poprzez pesel i kod).

2. Odnośniki do reszty podstron takich jak:

- formularz rejestracji pacjenta

- strony recepcji.

Strona zawierać będzie skrypt, który po wykryciu zmiennych wysłanych przez formularz logowania pacjenta wyświetli:

- imię pacjenta,
- nazwisko pacjenta,
- pesel pacjenta,
- listę umówionych spotkań z lekarzami,
- formularz przez, który tworzyć będzie spotkania wraz z skryptem obsługującym daną operację (dodanie do tablicy "spotkania" danych: id specjalizacji, id osoby).

4.3 Backlog Item 3

Tytuł zadania. Aktualności.

Opis zadania. Dodanie do strony głównej listę aktualności.

Priorytet. Niski.

Definition of Done. Edycja strony głównej tak aby w przejrzysty sposób wyświetlała dane z tablicy "aktualności": - Data aktualności (data), - Treść aktualności (opis).

4.4 Backlog Item 4

Tytuł zadania. Formularz zapisu dla pacjentów.

Opis zadania. Stworzenie strony dzięki której będzie można zapisać się do listy pacjentów.

Priorytet. Niski.

Definition of Done. Napisanie strony w języku PHP zawierającej formularz z danymi do wprowadzenia (Imię, Nazwisko, Pesel, Telefon kontaktowy). Po otrzymaniu danych zostaną one wprowadzone do tablicy "pacjenci". Ponadto strona po wypełnieniu formularza generować będzie 5-cyfrowy kod, który również będzie wprowadzany do tablicy.

4.5 Backlog Item 5

Tytuł zadania. Aktualizacja bazy danych.

Opis zadania. Dodanie do bazy danych potrzebnych tablic i ewentualna edycja starych tablic w razie potrzeby.

Priorytet. Wysoki.

Definition of Done. Dodanie do bazy danych, tablic:

RECEPCJA:

- id recepcjonisty (automatycznie przypisywany numer),
- imie (imię recepcjonisty),
- nazwisko (nazwisko recepcjonisty),
- login (login używany przy logowaniu na stronie recepcji),
- haslo (hasło używane przy logowaniu na stronie recepcji)

DYREKTOR:

- id dyrektora (automatycznie przypisywany numer),
- imie (imię dyrektora),
- nazwisko (nazwisko dyrektora),
- login (login używany przy logowaniu na stronie recepcji),
- haslo (hasło używane przy logowaniu na stronie recepcji).

Stworzenie 2 widoków, ułatwiających obsługę strony:

LEKARZE:

- imie (imię lekarza z tablicy "lekarze"),
- nazwisko (nazwisko lekarza z tablicy "lekarze"),
- specjalizacja (nazwa specjalizacji z tablicy "specjalizacje", odpowiednia dla konkretnego lekarza),
- nr pokoju (nr pokoju z tablicy "lekarze")

LISTA:

- id spotkania (id spotkania z tablicy "spotkania"),
- id osoby (id pacjenta z tablicy "pacjenci" odpowiednie dla konkretnego spotkania),
- id lekarza (id lekarza z tablicy "lekarze" odpowiednie dla konkretnego spotkania),
- pesel (pesel z tablicy "pacjenci"),
- imie osoby (imię osoby z tablicy "pacjenci"),
- nazwisko osoby (nazwisko osoby z tablicy "pacjenci"),
- specjalizacja (nazwa specjalizacji z tablicy "specjalizacje"),

- imie lekarza (imię lekarza z tablicy "lekarze"),
- nazwisko lekarza (nazwisko lekarza z tablicy "lekarze"),
- nr pokoju (numer pokoju z tablicy "lekarze"),
- data odbycia (data odbycia z tablicy "spotkania"),
- data zapisu (data zapisu z tablicy "spotkania"),
- stan (stan z tablicy "spotkania")

4.6 Backlog Item 6

Tytuł zadania. Strona logowania pracowników.

Opis zadania. Napisanie strony, która będzie dawać dostęp do logowania i wyświetlania listy opcji dla: dyrektora, recepcjonisty i lekarza.

Priorytet. Średni.

Definition of Done. Stworzenie strony na, której będzie formularz z danymi do wprowadzenia: login, hasło. Wraz z skryptem który po ich otrzymaniu przeszuka tablice: lekarze, dyrektor, recepcja w poszukiwaniu prawidłowości. Jeżeli znajdzie wykorzysta funkcję "session" i wpiszę do niej dane takie jak imię, nazwisko, id oraz tablica z informacją z której tablicy dana osoba pochodzi. W przypadku wykrycia zmiennych w "session", wypisuje imię, nazwisko oraz id danej osoby, oraz w zależności od zmiennej tablica listę odnośników do strony "zarzadzaj"

DLA LEKARZE:

- odnośnik wraz z informacją w formie GET proszącą o listę spotkań.
- odnośnik wraz z informacją w formie GET proszącą o kartę pacjenta.

DLA RECEPCJA:

- odnośnik wraz z informacją w formie GET proszącą o lista osób zapisanych.
- odnośnik wraz z informacją w formie GET proszącą o lista lekarzy .
- odnośnik wraz z informacją w formie GET proszącą o lista dostępnych specjalizacji.

DLA DYREKTOR:

- odnośniki wraz z informacjami w formie GET proszącą o możliwość edycji, usuwania, dodawania do tablic:
- a) lekarze
- b) specjalizacje

- c) recepcja
- d) spotkania
- e) pacjenci
- f) aktualnosci

4.7 Backlog Item 7

Tytuł zadania. Wyświetlanie, edycja, usuwanie.

Opis zadania. Stworzenie strony obsługujących edycję, wyświetlanie i usuwanie danych z tablic.

Priorytet. Wysoki.

Definition of Done. Stworzenie strony "zarzadzanie", która po otrzymaniu informacji formą GET i zweryfikowaniu uprawnień z funkcji "session", będzie udostępniać formie formularza wraz skryptem je obsługującym opcje dla osób z tablicy:

LEKARZE:

- listę spotkań (wyświetlanie),
- kartę pacjenta (wyświetlanie i edycja)

RECEPCJA:

- listę osób zapisanych (wyświetlanie i edycja),
- listę lekarzy (wyświetlanie),
- listę dostępnych specjalizacji (wyświetlanie)

DYREKTOR:

- listę lekarze (wyświetlanie, edycja i usuwanie),
- listę specjalizacje (wyświetlanie, edycja i usuwanie),
- listę recepcja (wyświetlanie, edycja i usuwanie),
- listę spotkania (wyświetlanie, edycja i usuwanie),
- listę pacjenci (wyświetlanie, edycja i usuwanie),
- listę aktualnosci (wyświetlanie, edycja i usuwanie)

5 Sprint 1

5.1 Cel

Stworzenie bazy danych dzięki której możliwe będzie przeglądanie strony tytułowej i formularza zapisu pacjentów.

5.2 Sprint Planning/Backlog

Tytuł zadania. Stworzenie początkowej bazy danych.

- Estymata: M.

Tytuł zadania. Strona tytułowa oraz formularz logowania pacjenta.

- Estymata: L.

Tytuł zadania. Aktualności.

- Estymata: S.

5.3 Realizacja

Tytuł zadania. Stworzenie początkowej bazy danych.

Wykonawca. Łukasz Mamak

Realizacja. Zadaniem do wykonania było stworzenie niezbędnej do dalszych kroków bazy danych. Realizacja tegoż zadania przebiegła bez większych komplikacji. W skrypcie zostało utworzonych kilka tabel, począwszy od aktualności, poprzez tabele pacjenci, lekarze, na tabeli spotkania kończąc. W każdej tabeli zdefiniowany został klucz główny dla wartości "niepowtarzalnych" w zależności od tabeli. Zdefiniowane zostały także klucze obce, które zostały wykorzystane do utworzenia relacji między parą tabel. Podsumowując, zadanie zostało zrealizowane bez istotnych do odnotowania problemów, nie wychodzących poza ramy błędnego wpisania składni danego polecenia tudzież tzw. "literówek."

Kod programu (środowisko `verbatim`):

```
 create database przychodnia character set utf8 collate utf8_unicode_ci;
use przychodnia;
Set time_zone='+1:00';

CREATE TABLE aktualnosci (
id_aktualnosci INT NOT NULL auto_increment,
data TIMESTAMP NOT NULL,
opis VARCHAR(50) NOT NULL,

CONSTRAINT c_pk0 PRIMARY KEY(id_aktualnosci)
) ENGINE = InnoDB;

CREATE TABLE pacjenci (
id_pacjenta INT NOT NULL auto_increment,
imie VARCHAR(50) NOT NULL,
nazwisko VARCHAR(50) NOT NULL,
karta_pacjenta VARCHAR(255) NOT NULL,
pesel BIGINT NOT NULL,
nr_telefonu BIGINT NOT NULL,
```

```
kod VARCHAR(50) NOT NULL,

CONSTRAINT c_pk PRIMARY KEY(id_osoby)
) ENGINE = InnoDB;

CREATE TABLE lekarze (
id_lekarza INT NOT NULL auto_increment,
login VARCHAR(50) NOT NULL,
haslo VARCHAR(50) NOT NULL,
imie VARCHAR(50) NOT NULL,
nazwisko VARCHAR(50) NOT NULL,
id_specjalizacji INT NOT NULL,
nr_pokoju VARCHAR(50) NOT NULL,

CONSTRAINT c_pk2 PRIMARY KEY(id_lekarza)
) ENGINE = InnoDB;


CREATE TABLE spotkania (
id_spotkania INT NOT NULL auto_increment,
id_specjalizacji INT NOT NULL,
id_lekarza INT NOT NULL,
id_osoby INT NOT NULL,
data_odbycia VARCHAR(50),
data_zapisu TIMESTAMP NOT NULL,
stan enum('0','1') NOT NULL DEFAULT '0',

CONSTRAINT c_pk3 PRIMARY KEY(id_zabiegu),
CONSTRAINT c_fk FOREIGN KEY(id_specjalizacji) REFERENCES lekarze (id_lekarza)
ON UPDATE CASCADE ON DELETE CASCADE,
CONSTRAINT c_fk2 FOREIGN KEY(id_lekarza) REFERENCES lekarze (id_lekarza)
ON UPDATE CASCADE ON DELETE CASCADE,
CONSTRAINT c_fk3 FOREIGN KEY(id_osoby) REFERENCES pacjenci (id_pacjenta)
ON UPDATE CASCADE ON DELETE CASCADE
) ENGINE = InnoDB;



CREATE PROCEDURE zatw ()
BEGIN
UPDATE spotkania SET stan='1' WHERE FROM_UNIXTIME(data_odbycia) <= NOW() AND stan='0';
END;

.
```

Tytuł zadania. Strona tytułowa oraz formularz logowania pacjenta.

Wykonawca. Łukasz Szkaradek

Realizacja. Celem zadania było wykonanie strony internetowej w której zawarte będą logowanie pacjenta, wyświetlanie spotkań danego pacjenta oraz ich dodawanie poprzez formularz. Wykonanie zadania odbyło się bez większych trudności, choć wykonanie niektórych algorytmów takich jak wyświetlanie wyboru godzin z pominięciem już zarezerwowanych oraz przeszłych godzin czy wyświetlanie odpowiednich dat dla wybranych dni wymagało większego wysiłku. Przydatna przy wykonaniu zadania była funkcja session, która przechowuje dane aż do ich usunięcia lub wyłączenia przeglądarki. Przy przesyłaniu danych z formularza została użyta forma POST. Zadanie zostało wykonane w całości, a wszystkie jego funkcje działają prawidłowo.

Kod programu (środowisko PHP):

```
<?php
ob_start();
date_default_timezone_set('Europe/Warsaw');
session_start();
$host="<<ip serwera>>";
$db_user="<<nazwa użytkownika>>";
$db_password="<<hasło użytkowinka>>";
$database="<<nazwa bazy danych>>";
$link= mysqli_connect($host,$db_user,$db_password,$database);
$zat="call zatw()";
$zat_w=mysqli_query($link,$zat);
if (isset($_GET[powrot])) {
session_destroy();
header("Location: index.php");
}
if (isset($_POST[stan])){
$zapytanie="SELECT * FROM pacjenci WHERE pesel='$_POST[pesel]' and kod='$_POST[kod]'";
$wykonaj=mysqli_query($link,$zapytanie);
if(@mysqli_num_rows($wykonaj)){
while($wiersz=mysqli_fetch_assoc($wykonaj)) {
$_SESSION[zalogowany] = "pacjent";
$_SESSION[id_p] = $wiersz['id_pacjenta'];
$_SESSION['baza'] = 'pacjent';
}
} else {
$brak='1';
}
}
?>
<html>
<head>
<title>Przychodnia lekarska</title>
<META http-equiv=Content-Language content=pl>
<META http-equiv=Content-Type content="text/html; charset=windows-1250">
<style>
a {
    color: black;
```

```
    text-decoration: none;
}
</style>
</head>
<body style="background-size: 100% 200%;" background="tlo.jpg" bgproperties="fixed">
<center>
<?php if($_SESSION['baza']=="pacjent") {
echo '<a href="?powrot">Powrót</a>';
} else {
echo '<a href="formularz.php">Formularz</a>';
}
?> | <a href="recepcja.php"><?php if($_SESSION['zalogowany']=="ok") {
echo $_SESSION['login'];
} else {
echo 'Zaloguj się';
}
?></a><br><br>
<img style="margin:-7px;width: 101%; text-align:center;" src="logo.png" alt="Logo" />
<?php
if (isset($brak)) {echo "Błąd! Brak osoby w bazie!";}
if ($_SESSION[baza] == "pacjent") {
$zapytanie="SELECT * FROM pacjenci WHERE id_pacjenta='$_SESSION[id_p]'";
$wykonaj=mysqli_query($link,$zapytanie);
if(@mysqli_num_rows($wykonaj)){
echo '<table border="1">';
while($wiersz=mysqli_fetch_assoc($wykonaj)) {
echo "<tr><td>Imię: " . $wiersz['imie'] . "</td><td>Nazwisko: " . $wiersz['nazwisko'] . "<
}
echo '</table><br><br>';
}
if (isset($_POST[w_godziny])){
if ($_POST[w_godziny]=="") {header("Location: index.php");}
$odb = strtotime($_SESSION['w_dnia'] . '-' . date('Y') . ' ' . $_POST[godzina] . '.00');
$zapytanie_x="SELECT * FROM spotkania WHERE id_osoby='$_SESSION[id_p]' and id_specjalizacj
$wykonaj_x=mysqli_query($link,$zapytanie_x);
if(!@mysqli_num_rows($wykonaj_x)) {
$zapytanie2="INSERT into spotkania (id_specjalizacji, id_lekarza, id_osoby, data_odbycia,
$wykonaj2=mysqli_query($link,$zapytanie2);
}
unset($_SESSION['w_dnia']);
unset($_SESSION[w_specjalizacji]);
unset($_SESSION[w_lekarza]);
header("Location: index.php");
}
if (isset($_POST[spotkanie]) or isset($_POST[lekarz])) {
if (isset($_POST[spotkanie])) {
echo "Wybierz lekarza:<br>";
$_SESSION['w_specjalizacji'] = $_POST[id_specjalizacji];
$_SESSION['w_dnia'] = $_POST[dzien];
echo '<form action="" method="POST"><select name="w_lekarza">';
```

```
$zapytanie1="SELECT * FROM lekarze where id_specjalizacji='".$_SESSION['w_specjalizacji'].
$wykonaj1=mysqli_query($link,$zapytanie1);
while($wiersz1=mysqli_fetch_assoc($wykonaj1)) {
echo '<option value="' . $wiersz1['id_lekarza'] . '">' . $wiersz1['imie'] . " " . $wiersz1
}
echo '</select><br> <input type="submit" name="lekarz" value="Dalej"></form><br><br><a hre
}
if (isset($_POST[lekarz])) {
$_SESSION[w_lekarza] = $_POST[w_lekarza];
$zapytanie="SELECT * FROM spotkania WHERE id_lekarza='".$_POST[w_lekarza]."'";
$wykonaj=mysqli_query($link,$zapytanie);
if(@mysqli_num_rows($wykonaj)){
echo "Wybierz godzinę spotkania:<br>";
while($wiersz=mysqli_fetch_assoc($wykonaj)) {
$zaj[] = date ('j-m G.i' , $wiersz['data_odbycia']);
}
}
$arr = array('9.00', '9.30', '10.00', '10.30', '11.00', '11.30', '12.00', '12.30', '13.00'
echo '<form action="" method="POST"><select name="godzina">';
foreach ($arr as &$value) {
if ($_SESSION['w_dnia'] == date ('j-m')) {
if ($value > date('G.i') and !is_numeric(array_search($_SESSION['w_dnia'].' '.$value, $zaj
echo '<option value="' . $value . '">' . $value . "</option>";
}
} else {
echo $_SESSION['w_dnia'].' '.$value;
if (!is_numeric(array_search($_SESSION['w_dnia'].' '.$value, $zaj))) {
echo '<option value="' . $value . '">' . $value . "</option>";
}
}
}
echo '</select></td><br> <input type="submit" name="w_godziny" value="Zapisz"></form><br><
}
} else {
echo '<button onclick="dodaj()">Dodaj spotkanie</button><br><div  style="height:50px">';
$zapytanie_s="SELECT * FROM specjalizacje";
$wykonaj_s=mysqli_query($link,$zapytanie_s);
echo '<form action="" method="POST"><input type="text" hidden=hidden name="pesel" value="'
echo '<table style="display: none; text-align:center;" id="dodaj" border="1">';
echo '<tr><td>Wybierz specjalizację:</td><td><select name="id_specjalizacji">';
while($wiersz=mysqli_fetch_assoc($wykonaj_s)) {
echo '<option value="' . $wiersz['id_specjalizacji'] . '">' . $wiersz['nazwa_specjalizacji
}
echo '</select></td></tr>';
echo '<tr><td>Wybierz dzień spotkania:</td><td><select name="dzien">';
$dod = 0;
$dzien = date("N");
$dzien_m = date("j");
$miesiac = date("n");
if ($dzien == 6) {
```

```
$dzien = 1;
$dzien_m = $dzien_m + 1;
if ($dzien_m > date("t")) {
$dzien_m = 1;
if ($miesiac == 12) {
$miesiac = 1;
} else {
$miesiac = $miesiac + 1;
}
}
$dzien_m = $dzien_m + 1;
if ($dzien_m > date("t")) {
$dzien_m = 1;
if ($miesiac == 12) {
$miesiac = 1;
} else {
$miesiac = $miesiac + 1;
}
}
}
if ($dzien == 7) {
$dzien_m = $dzien_m + 1;
if ($dzien_m > date("t")) {
$dzien_m = 1;
if ($miesiac == 12) {
$miesiac = 1;
} else {
$miesiac = $miesiac + 1;
}
}
}
while($dzien+$dod <= 5) {
if ($dzien_temp == date("t")) {
$dzien_m = -1 * $dod + 1 ;
if ($miesiac == 12) {
$miesiac = 1;
} else {
$miesiac = $miesiac + 1;
}
}
$dzien_temp = $dzien_m + $dod;
echo '<option value="' . $dzien_temp . '-' . $miesiac . '">' . $dzien_temp . '.' . $miesia
switch($dzien+$dod) {
    case "1":
        echo "Poniedziałek";
    break;
    case "2":
        echo "Wtorek";
    break;
    case "3":
```

```
        echo "Środa";
    break;
    case "4":
        echo "Czwartek";
    break;
    case "5":
        echo "Piątek";
    break;
}

echo "</option>";
$dod = $dod + 1;
}
echo '</select></td></tr></table><input style="display: none" id="p_dodaj" type="submit" n
echo '<br><br>Lista spotkań z lekarzami:<table width="1000" style="text-align:center;" bor
echo "<td>Specjalizacja</td><td>Imie lekarza</td><td>Nazwisko lekarza</td><td>Numer pokoju
$zapytanie_z="SELECT * FROM spotkania WHERE id_osoby='".$_SESSION[id_p]."'";
$wykonaj_z=mysqli_query($link,$zapytanie_z);
while($wiersz=mysqli_fetch_assoc($wykonaj_z)) {
$zapytanie_s="SELECT * FROM specjalizacje WHERE id_specjalizacji='".$wiersz['id_specjaliza
$wykonaj_s=mysqli_query($link,$zapytanie_s);
while($wiersz_s=mysqli_fetch_assoc($wykonaj_s)) {
$specjalizacja = $wiersz_s['nazwa_specjalizacji'];
}
$zapytanie_l="SELECT * FROM lekarze WHERE id_lekarza='".$wiersz['id_lekarza']."'";
$wykonaj_l=mysqli_query($link,$zapytanie_l);
while($wiersz_l=mysqli_fetch_assoc($wykonaj_l)) {
$imie_lekarza = $wiersz_l['imie'];
$nazwisko_lekarza = $wiersz_l['nazwisko'];
$nr_pokoju = $wiersz_l['nr_pokoju'];
}
echo '<tr><td>'.$specjalizacja.'</td><td>'.$imie_lekarza.'</td><td>'.$nazwisko_lekarza.'</
if ($wiersz['stan']=='0') {
echo 'Nie';
} else {
echo 'Tak';
}
echo'</td>';
}
echo '</table>';
}
} else {
echo'<br><br><br>
<form action="" method="POST">
<table width="300">
<tr>
<td align="right">Pesel:</td>
<td align="let"><input type="text" name="pesel"></td>
<td align="right">Kod:</td>
<td align="let"><input type="text" name="kod"></td>
```

```
</tr>
<tr>
<td align="center" colspan="4"><input type="submit" name="stan" value="Sprawdź"></td>
</tr>
</table>
</form>
<br>
<table border="0" width="300">
<tr>
<td align="left">
Witaj na naszej przychodni lekarskiej. Proszę podać swój pesel oraz kod.
</td>
</tr>
<tr>
<td align="right"><i></i></td>
</tr>
</table>';
echo '<br><br><br>
<table border="1px">
<tr><td style="text-align: center;" colspan="2">Aktualności</td></tr>';
$aktualnosci="SELECT * FROM aktualnosci";
$wykonaj=mysqli_query($link,$aktualnosci);
if(@mysqli_num_rows($wykonaj)){
while($wiersz=mysqli_fetch_assoc($wykonaj)) {
echo '<tr><td width="150">'.$wiersz['data'].'</td><td width="950">'.$wiersz['opis'].'</tr>
}
}
}
?>
</center>

<script>

var d = document.getElementById('dodaj');
var pd = document.getElementById('p_dodaj');

function dodaj() {
    if (d.style.display === 'none') {
        d.style.display = '';
        pd.style.display = '';
    } else {
        d.style.display = 'none';
        pd.style.display = 'none';
    }
}
</script>
</body>
</html>
```

.

Tytuł zadania. Aktualności.

Wykonawca. Paweł Wilczek

Realizacja. Zadanie polegało na utworzeniu aktualności na stronie głównej przychodni lekarskiej dzięki wyłuskaniu odpowiednich informacji z tabeli "aktualności" i umieszczeniu ich na stronie głównej dzięki skryptowi PHP.

Tabela aktualności:

```
CREATE TABLE aktualnosci (
id_aktualnosci INT NOT NULL auto_increment,
data TIMESTAMP NOT NULL,
opis VARCHAR(50) NOT NULL,

CONSTRAINT c_pk0 PRIMARY KEY(id_aktualnosci)
) ENGINE = InnoDB;
```

Skrypt PHP:

```
echo '<br><br><br>
<table border="1px">
<tr><td style="text-align: center;" colspan="2">Aktualności</td></tr>';
$aktualnosci="SELECT * FROM aktualnosci";
$wykonaj=mysqli_query($link,$aktualnosci);
if(@mysqli_num_rows($wykonaj)){
while($wiersz=mysqli_fetch_assoc($wykonaj)) {
echo '<tr><td width="150">'.$wiersz['data'].'</td><td width="950">'.$wiersz['opis'].'</tr>
}
}
}
```

5.4 Sprint Review/Demo

«Sprawozdanie z przeglądu Sprint'u – czy założony cel (przyrost) został osiągnięty oraz czy wszystkie zaplanowane Backlog Item'y zostały zrealizowane? Demostracja przyrostu produktu».

6 Sprint 2

6.1 Cel

«Określić, w jakim celu tworzony jest przyrost produktu».

6.2 Sprint Planning/Backlog

Tytuł zadania. Formularz zapisu dla pacjentów.

- Estymata: M.

Tytuł zadania. Aktualizacja bazy danych.

- Estymata: XL.

«Tutaj dodawać kolejne zadania»

6.3 Realizacja

Tytuł zadania. «Tytuł».

Wykonawca. ¡¿.

Realizacja. «Sprawozdanie z realizacji zadania (w tym ocena zgodności z estymatą). Kod programu (środowisko `verbatim`):

```
for (i=1; i<10; i++) ...
```

».

Tytuł zadania. «Tytuł».

Wykonawca. ¡¿.

Realizacja. «Sprawozdanie z realizacji zadania (w tym ocena zgodności z estymatą). Kod programu (środowisko `verbatim`):

```
for (i=1; i<10; i++) ...
```

».

6.4 Sprint Review/Demo

«Sprawozdanie z przeglądu Sprint'u – czy założony cel (przyrost) został osiągnięty oraz czy wszystkie zaplanowane Backlog Item'y zostały zrealizowane? Demostracja przyrostu produktu».

7 Sprint 3

7.1 Cel

«Określić, w jakim celu tworzony jest przyrost produktu».

7.2 Sprint Planning/Backlog

Tytuł zadania. Strona logowania pracowników.

- Estymata: XL.

Tytuł zadania. Wyświetlanie, edycja, usuwanie.

- Estymata: XXL.

«Tutaj dodawać kolejne zadania»

7.3 Realizacja

Tytuł zadania. «Tytuł».

Wykonawca. ¡¿.

Realizacja. «Sprawozdanie z realizacji zadania (w tym ocena zgodności z estymatą). Kod programu (środowisko `verbatim`):

```
for (i=1; i<10; i++) ...
```

».

Tytuł zadania. «Tytuł».

Wykonawca. ¡¿.

Realizacja. «Sprawozdanie z realizacji zadania (w tym ocena zgodności z estymatą). Kod programu (środowisko `verbatim`):

```
for (i=1; i<10; i++) ...
```

».

7.4 Sprint Review/Demo

«Sprawozdanie z przeglądu Sprint'u – czy założony cel (przyrost) został osiągnięty oraz czy wszystkie zaplanowane Backlog Item'y zostały zrealizowane? Demostracja przyrostu produktu».

«Tutaj dodawać kolejne Sprint'y»

Literatura

[1] S. R. Covey, *7 nawyków skutecznego działania*, Rebis, Poznań, 2007.

[2] Tobias Oetiker i wsp., Nie za krótkie wprowadzenie do systemu LaTeX 2_ε, `ftp://ftp.gust.org.pl/TeX/info/lshort/polish/lshort2e.pdf`

[3] K. Schwaber, J. Sutherland, *Scrum Guide*, `http://www.scrumguides.org/`, 2016.

[4] `https://agilepainrelief.com/notesfromatooluser/tag/scrum-by-example`

[5] `https://www.tutorialspoint.com/scrum/scrum_user_stories.htm`