icon.jpg СОФИЙСКИ УНИВЕРСИТЕТ “СВ. КЛИМЕНТ ОХРИДСКИ”

ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

КУРСОВ ПРОЕКТ

ПО ИЗКУСТВЕН ИНТЕЛЕКТ

Тема:

Реализирайте експертна система, която съхранява информация за софтуерни компании. Системата трябва да дава възможност за търсене на подходяща компания от страна на кандидати за работа, които въвеждат опита и предпочитанията си. В допълнение системата трябва да дава възможност на експерти да въвеждат данни за нови компании и да променят данните за вече известни компании.

Студент:

Моника Мариова Спасова , група 3, 44950

София, юни 2016 г.

**1.Формулировка на задачата**

Реализирайте експертна система, която съхранява информация за софтуерни компании. Системата трябва да дава възможност за търсене на подходяща компания от страна на кандидати за работа, които въвеждат опита и предпочитанията си. В допълнение системата трябва да дава възможност на експерти да въвеждат данни за нови компании и да променят данните за вече известни компании.

**2.Използвани алгоритми**

Използваният метод за намиране на най-подходящата позиция за даден потребител спрямо въведените данни, работи по следния начин:

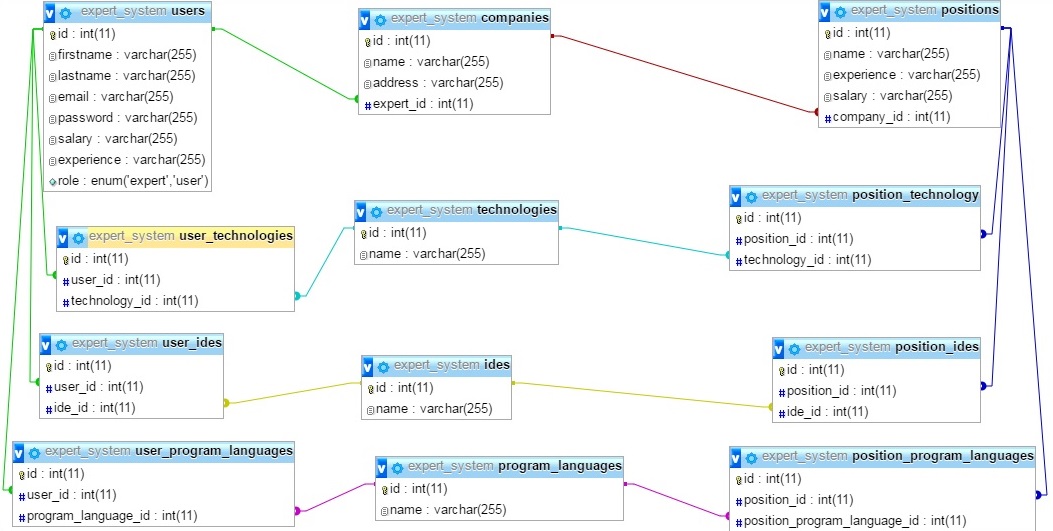
- всяко едно от изискванията в позицията (опит, парично възнаграждение, познати технологии, програмни езици и среди) е един критерий, по който се търси подходяща позиция за дадения кандидат . Следователно има 5 критерия.

- всеки един от критериите има дадено тегло – за критерия „опит”, теглото има стойност „2” , за останалите 4 критерия теглото е със стойност „1”

- дадена позиция се смята за най-подходяща, когато покрива поне 4 от критериите, като задължително един от критериите трябва да бъде този за „опит”. При неудовлетворяване на критерия за „опит” позицията не се разглежда като подходяща.

**3.Описание на програмната реализация**

Базата от данни “expert\_system” се състои от 12 таблици.

**

Таблицата “users” отговаря за съхранението на данните на потребителите и експертите, като съответната роля се съхранява в полето “role”. Тези данни се записват при регистрацията на нов потребител/експерт. Таблицата “users” e свързана с “user\_technologies”, “user\_ides”, “user\_program\_languages” чрез така нар. „foreign” ключове. Аналогично таблицата „positions” e свързана с таблиците “position\_technology”, “position\_ides”,”position\_program\_languages”.

В таблиците “technologies”,”ides”,”program\_languages” се съдържат съответно технологии, среди за програмиране и програмни езици.

Наличието на свързващи таблици е небходимо, за да бъде удовлетворено отношението “много към много” . Например една позиция може да изисква познаването на няколко езици, а същевременно един и същ език да бъде необходим за повече позиции.

За връзката с базата данни се използва „PDO( PHP Data Objects)” разширението. За целта се използва функцията „connect()”.

*public function connect()*

*{*

*return new PDO('mysql:host=localhost;dbname=expert\_system','root','',*

*array(PDO::MYSQL\_ATTR\_INIT\_COMMAND => "SET NAMES utf8"));*

*}*

Чрез „MYSQL\_ATTR\_INIT\_COMMAND” и неговата стойност се задава данните, които биват взимани, въвеждани и обновявани да имат „UTF8” кодиране.

За да се използва системата е необходима регистрация. Въвежданите данни в регистрационната форма и формата за влизане се подлагат на валидация. Това се осъществява чрез класа „ValidatorAbstract”, който се наследява и методът „isValid()”. Този метод приема данните, които са попълнени и ги валидира спрямо условията , които трябва да се изпълняват. Тези условия се задават за всяка една от формите, като едно от най-често използваните е „required” (условие, което изисква съответното поле да не е празно). За обработването на верните данни при регистрация и тяхното съхраняване в базата се грижи функцията „register()”.

*function register($data)*

*{*

*$hashed\_password = password\_hash(trim($data['password']), PASSWORD\_DEFAULT);*

*$stmt = connection()->prepare("INSERT INTO users(firstname, lastname, email, password, role)*

*VALUES(:firstname, :lastname, :email, :hashedPassword, :role);");*

*$stmt->bindParam(':firstname', trim($data['firstname']), PDO::PARAM\_STR);*

*$stmt->bindParam(':lastname', trim($data['lastname']), PDO::PARAM\_STR);*

*$stmt->bindParam(':email', trim($data['email']), PDO::PARAM\_STR);*

*$stmt->bindParam(':hashedPassword', $hashed\_password, PDO::PARAM\_STR);*

*$stmt->bindParam(':role', $data['role'], PDO::PARAM\_STR);*

*$stmt->execute();*

*}*

Преди да се извика тази функция , данните биват валидирани и от функцията „exists()”.

*function exists($email)*

*{*

*$exists = false;*

*$stmt = connection()->prepare("SELECT email FROM users WHERE email = :email");*

*$stmt->bindParam(':email', trim($email), PDO::PARAM\_STR);*

*$stmt->execute();*

*if ($stmt->fetch()) {*

*$exists=true;*

*}*

*return $exists;*

*}*

Тази функция има за цел да проверява дали в базата е регистриран потребител/експерт със същия имейл адрес. Ако функцията върне „истина” , то тогава се изписва съответно съобщение за грешка , а именно , че подобен потребител е вече регистриран. В противен случай се извиква гореспоменатата функция „register()” и с нейна помощ данните се записват в базата (таблицата „users”). При успешна регистрация потребителят/експертът може да използва линка за влизане в системата. За влизане в системата се използват въведените при регистрация имейл и парола. Влизането става през формата „login”. Въведените данни отново се проверяват от „ValidatorAbstract” класа и при вярност на данните се изпращат към функцията „select()”, която проверява дали въведените данни съвпадат с някои от данните в таблицата „users”.

*function select($email, $password)*

*{*

*$login = array('loginPassword' => false);*

*$stmt = connection()->prepare("SELECT id, password, role FROM users WHERE email = :email");*

*$stmt->bindParam(':email', trim($email) , PDO::PARAM\_STR);*

*$stmt->execute();*

*$result = $stmt->fetch(PDO::FETCH\_OBJ);*

*if (password\_verify(trim($password), $result->password) == true) {*

*$login = array(*

*'loginPassword' => true,*

*'role' => $result->role,*

*'user\_id' => $result->id*

*);*

*}*

*return $login;*

*}*

При успех функцията връща идентификационния номер и ролята на влизащия в системата потребител. Тези два атрибута, заедно с булевата променлива „is\_logged”, на която и се дава стойност „истина”, се запазват в бисквитки. Влизащият се пренасочва към съответната страница спрямо това каква е ролята му. Ако е потребител се пренасочва към страницата „index\_user.php”, в противен случай бива насочен към „index\_expert.php”. При първоначалното си влизане , ако влизащият е експерт , то той вижда празна таблица и меню с три линка – „Home”, „Add Company”, „Logout”. Първият линк е препратка към „index\_expert.php”, вторият се използва за въвеждане на нова софтуерна компания, а третият за излизане от системата. При влизане във втория линк експертът вижда форма , състояща се от две полета, които се отнасят за името и адреса на въвежданата софтуерна компания. При въвеждане на данни за нова компания , отново те биват валидирани , на база на това дали са попълнени всички полета и дали в базата не съществува софтуерна компания със същото име и същия адрес. Проверката за това дали нововъведената компания е уникална се прави чрез функцията „exists()”.

*function exists($data)*

*{*

*$exists = false;*

*$stmt = connection()->prepare("SELECT id FROM companies WHERE name = :name AND address = :address LIMIT 1");*

*$stmt->bindParam(':name', trim($data['name']), PDO::PARAM\_STR);*

*$stmt->bindParam(':address', trim($data['address']), PDO::PARAM\_STR);*

*$stmt->execute();*

*if ($stmt->fetch()) {*

*$exists = true;*

*} else {*

*insert($data);*

*}*

*return $exists;*

*}*

При истина се връща съобщение за грешка, а в противен случай се използва функцията „insert()” за въвеждане на данните в базата (таблица „companies”).

*function insert($data)*

*{*

*$expert\_id = $\_COOKIE['id'];*

*$stmt = connection()->prepare("INSERT INTO companies(name, address, expert\_id)*

*VALUES(:name, :address, :expert\_id);");*

*$stmt->bindParam(':name', trim($data['name']), PDO::PARAM\_STR);*

*$stmt->bindParam(':address', trim($data['address']), PDO::PARAM\_STR);*

*$stmt->bindParam(':expert\_id', $expert\_id, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

На всяка въведена компания в базата съответства поле „expert\_id”, което е идентификационният номер на експерта, който е въвел данните. След като е въведена поне една софтуерна компания и при влизане в линка „Home”, експертът вижда таблица, в която се съдържат данните на всички компании, които е въвел. В съответната таблица се съдържа и линк с надпис „Edit”, с което се дава възможност за промяна на данните на дадена компания. Името на компанията представлява линк , който води експерта към друга страница, която при първоначално влизане отново има празна таблица и линк с надпис „Add New Position”. Този линк пренасочва експерта към форма за добавяне на нова позиция в съответната софтуерна компания. Тази форма съдържа шест полета за попълване – име на позицията, изискван опит в месеци, съответното парично възнаграждение в левове, програмните езици, програмните технологии и програмните среди, които кандидатът трябва да знае. При успешното преминаване на валидацията, данните биват записвани в базата с помощта на функциите „insertPosition()”, „insertTechnology()”, „insertLanguage()”, „insertIdes()”, които съответно записват данните в таблиците – „positions”, „position\_technology”, „position\_program\_languages”, „position\_ides”, като във втората, третата и четвъртата таблица се записват идентификационните номера на въведената позиция и идентификационните номера на технологиите, програмните езици и програмните среди.

*function insertPosition($data)*

*{*

*$db = new PDO('mysql:host=localhost;dbname=expert\_system','root','',*

*array(PDO::MYSQL\_ATTR\_INIT\_COMMAND => "SET NAMES utf8"));*

*$stmt = $db->prepare("INSERT INTO positions(name, experience, salary, company\_id)*

*VALUES(:name, :experience, :salary, :company\_id);");*

*$stmt->bindParam(':name', trim($data['name']), PDO::PARAM\_STR);*

*$stmt->bindParam(':experience', trim($data['experience']), PDO::PARAM\_STR);*

*$stmt->bindParam(':salary', trim($data['salary']), PDO::PARAM\_STR);*

*$stmt->bindParam(':company\_id', $data['id'], PDO::PARAM\_INT);*

*$stmt->execute();*

*return $db->lastInsertId();*

*}*

*function insertTechnology($id, $data)*

*{*

*foreach ($data as $value) {*

*$stmt = connection()->prepare("INSERT INTO position\_technology(position\_id, technology\_id)*

*VALUES(:position\_id, :technology\_id);");*

*$stmt->bindParam(':position\_id', $id, PDO::PARAM\_INT);*

*$stmt->bindParam(':technology\_id', $value, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*}*

*function insertLanguage($id , $data)*

*{*

*foreach ($data as $value) {*

*$stmt = connection()->prepare("INSERT INTO position\_program\_languages(position\_id, position\_program\_language\_id)*

*VALUES(:position\_id, :program\_id);");*

*$stmt->bindParam(':position\_id', $id, PDO::PARAM\_INT);*

*$stmt->bindParam(':program\_id', $value, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*}*

*function insertIdes($id, $data)*

*{*

*foreach ($data as $value) {*

*$stmt = connection()->prepare("INSERT INTO position\_ides(position\_id, ide\_id)*

*VALUES(:position\_id, :ide\_id);");*

*$stmt->bindParam(':position\_id', $id, PDO::PARAM\_INT);*

*$stmt->bindParam(':ide\_id', $value, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*}*

Сега, когато е въведена поне една позиция за дадена компания при влизане в линка на компанията, се вижда таблица със съответните свободни позиции. Името на позицията представлява линк , на който се съдържа детайлна информация за нея. В таблицата с позициите е дадена възможност за обновяване на информацията и изтриване на позицията.

В случай , че влизащият в системата е с роля „user”, той бива пренасочван към страницата „index\_user.php”. При първоначалното си влизане потребителят вижда меню с четири линка – „Home”, „Edit Your Data”, „Proposed Position”, „Logout” и своите данни, като повечето от тях са празни . Първият линк води към „index\_user.php”, вторият линк препраща потребителя към форма, в която може да промени своите данни . Тази форма съдържа седем полета – за промяна на името и фамилията, опита в месеци, желаната заплата, научените програмни езици и технологии , използваните програмни среди. При въвеждане на данните в тази форма отново се прави валидация и при успешно валидиране данните се подават на функциите – „update()”, „deleteUserIdes()”, „deleteUserLanguages()”, „deleteUserTechnologies()”, „insertUserTechnology()”, „insertUserLanguage()”, „insertUserIdes()”, които обновяват информацията за потребителя.

*function update($id, $data)*

*{*

*$stmt = connection()->prepare("UPDATE users SET firstname = :firstname, lastname = :lastname, salary = :salary, experience = :experience WHERE id = :id");*

*$stmt->bindParam(':id', $id, PDO::PARAM\_INT);*

*$stmt->bindParam(':firstname', trim($data['firstname']), PDO::PARAM\_STR);*

*$stmt->bindParam(':lastname', trim($data['lastname']), PDO::PARAM\_STR);*

*$stmt->bindParam(':salary', trim($data['salary']), PDO::PARAM\_STR);*

*$stmt->bindParam(':experience', trim($data['experience']), PDO::PARAM\_STR);*

*$stmt->execute();*

*}*

*function deleteUserIdes($id)*

*{*

*$stmt = connection()->prepare("DELETE FROM user\_ides WHERE user\_id = :user\_id");*

*$stmt->bindParam(':user\_id', $id, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*function deleteUserLanguages($id)*

*{*

*$stmt = connection()->prepare("DELETE FROM user\_program\_languages WHERE user\_id = :user\_id");*

*$stmt->bindParam(':user\_id', $id, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*function deleteUserTechnologies($id)*

*{*

*$stmt = connection()->prepare("DELETE FROM user\_technologies WHERE user\_id = :user\_id");*

*$stmt->bindParam(':user\_id', $id, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*function insertUserTechnology($id, $data)*

*{*

*foreach ($data as $value) {*

*$stmt = connection()->prepare("INSERT INTO user\_technologies(user\_id, technology\_id)*

*VALUES(:user\_id, :technology\_id);");*

*$stmt->bindParam(':user\_id', $id, PDO::PARAM\_INT);*

*$stmt->bindParam(':technology\_id', $value, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*}*

*function insertUserLanguage($id, $data)*

*{*

*foreach ($data as $value) {*

*$stmt = connection()->prepare("INSERT INTO user\_program\_languages(user\_id, program\_language\_id)*

*VALUES(:user\_id, :program\_id);");*

*$stmt->bindParam(':user\_id', $id, PDO::PARAM\_INT);*

*$stmt->bindParam(':program\_id', $value, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*}*

*function insertUserIdes($id, $data)*

*{*

*foreach ($data as $value) {*

*$stmt = connection()->prepare("INSERT INTO user\_ides(user\_id, ide\_id)*

*VALUES(:user\_id, :ide\_id);");*

*$stmt->bindParam(':user\_id', $id, PDO::PARAM\_INT);*

*$stmt->bindParam(':ide\_id', $value, PDO::PARAM\_INT);*

*$stmt->execute();*

*}*

*}*

При обновяване на данните и влизане в линка „Home”, потребителят вече може да види всички свои данни попълнени.

При влизане в третият линк „Proposed Positions”, потребителят вижда таблица с най-подходящите за него позиции спрямо неговите въведени данни. Ако не са намерени такива позиции е показан падащ списък , от който потребителят може да избере, за да види позиции, които са подходящи поотделно спрямо неговия опит, исканото заплащане, познатите технологии, програмни среди и езици. За определянето на позициите за даден потребител се използват следните функции – „selectByIdes()”, „selectBySalary()”, „selectByExperience()”, „selectByTechnologies()”, „selectByLanguages()”, „getMostAppropriatePositionsId()”.

*function selectByIdes($ides)*

*{*

*$stmt = connection()->prepare("SELECT DISTINCT pi.position\_id, c.name AS company\_name, p.name, p.company\_id*

*FROM position\_ides pi*

*JOIN positions p ON p.id = pi.position\_id*

*JOIN companies c ON p.company\_id = c.id*

*WHERE pi.ide\_id IN ($ides)");*

*$stmt->execute();*

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

*}*

*function selectBySalary($salary)*

*{*

*$db = new PDO('mysql:host=localhost;dbname=expert\_system','root','',*

*array(PDO::MYSQL\_ATTR\_INIT\_COMMAND => "SET NAMES utf8"));*

*$stmt = $db->prepare("SELECT p.id AS position\_id, p.name, p.salary, p.experience, p.company\_id,c.name AS company\_name*

*FROM positions p*

*JOIN companies c ON p.company\_id = c.id*

*WHERE p.salary >= $salary");*

*$stmt->execute();*

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

*}*

*function selectByExperience($experience)*

*{*

*$db = new PDO('mysql:host=localhost;dbname=expert\_system','root','',*

*array(PDO::MYSQL\_ATTR\_INIT\_COMMAND => "SET NAMES utf8"));*

*$stmt = $db->prepare("SELECT p.id AS position\_id, p.name, p.salary, p.experience, p.company\_id,c.name AS company\_name*

*FROM positions p*

*JOIN companies c ON p.company\_id = c.id*

*WHERE p.experience <= $experience");*

*$stmt->execute();*

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

*}*

*function selectByTechnologies($technologies)*

*{*

*$stmt = connection()->prepare("SELECT DISTINCT pt.position\_id, c.name AS company\_name, p.name, p.company\_id*

*FROM position\_technology pt*

*JOIN positions p ON p.id = pt.position\_id*

*JOIN companies c ON p.company\_id = c.id*

*WHERE pt.technology\_id IN ($technologies)");*

*$stmt->execute();*

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

*}*

*function selectByLanguages($languages)*

*{*

*$stmt = connection()->prepare("SELECT DISTINCT ppl.position\_id, c.name AS company\_name, p.name, p.company\_id*

*FROM position\_program\_languages ppl*

*JOIN positions p ON p.id = ppl.position\_id*

*JOIN companies c ON p.company\_id = c.id*

*WHERE ppl.position\_program\_language\_id IN ($languages)");*

*$stmt->execute();*

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

*}*

*function getMostAppropriatePositionsId($positions)*

*{*

*$position\_ids = array();*

*$positions\_count = array\_count\_values($positions);*

*foreach ($positions\_count as $key => $value) {*

*if ($value >= 4) {*

*$position\_ids[] = $key;*

*}*

*}*

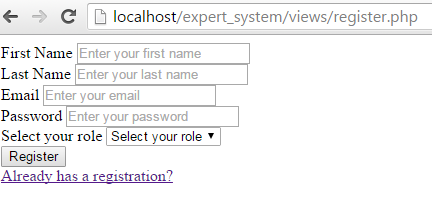
*return $position\_ids;*

*}*

Четвъртият линк „Logout” служи за излизане от системата.

**4.Примери, илюстриращи работата на програмната система**

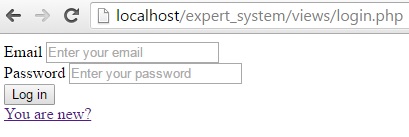
Всичко започва с регистрацията, която изглежда по следния начин:



*Фиг. 1 Регистрационна форма*

При нея всички полета са задължителни за попълване. При непопълване на дадено поле под съответното се изписва съобщение за грешка. Ако въведеният имейл адрес съвпада с вече регистриран потребител отново се извежда съобщение за грешка. При успешно завършване на регистрацията формата се връща в първоначалното си състояние и бива изписано потвърждителното съобщение за успешното регистриране в системата. Ако потребителят вече се е регистрирал може да използва линка „Already has a registration ?”, за да влезе в системата.

Формата за влизане в системата е следната:



*Фиг.2 Форма за влизане*

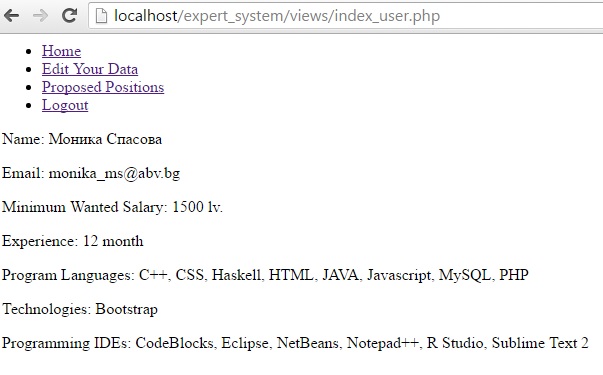
Както при регистрационната форма, така и тук са задължителни всички полета. При непопълване на някое поле излиза съобщение за грешка. При попълване на неверни данни отново се извежда предупреждаващ надпис за неправилни данни. В зависимост от ролята, която е избрал потребителят при регистрацията си, той бива препратен, при успешно влизане в системата, към две различни страници. Ако е избрана роля „expert”:



*Фиг.3-а Начален изглед на потребител „expert”*

При първоначалното си влизане експертът вижда празна таблица без данни.

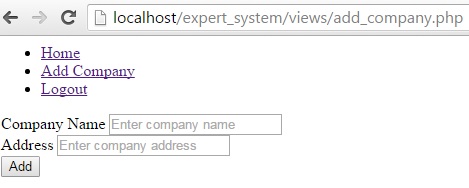
Ако избраната роля е „user”, началната страница изглежда по следния начин:



*Фиг.3-б Начална страница на потребител с роля „user”*

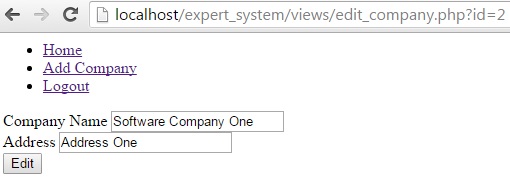
При първоначалното влизане на потребител с роля „user”, единствено са попълнени данните – име и имейл адрес.

За въвеждане на данни за компания се ползва линкът „Add Company”, който изпраща експерта към следната форма:



*Фиг. 4 Форма за добавяне на софтуерна компания*

Тук отново и двете полета са задължителни за попълване. При въвеждане на данни за компания, която вече е била въведена, се изписва съобщение за грешка. При успешно записване на компанията в системата се извежда потвърждаващо съобщение. Експертът може да прегледа добавените от него компании при посещаване на линка „Home” – Фиг.3-а . Данните за дадена компания могат да бъдат променени при влизане в линка „Edit” на съответната компания. При промяната на данни за компания, експертът вижда следната форма:



*Фиг.5 Промяна на данни за дадена компания*

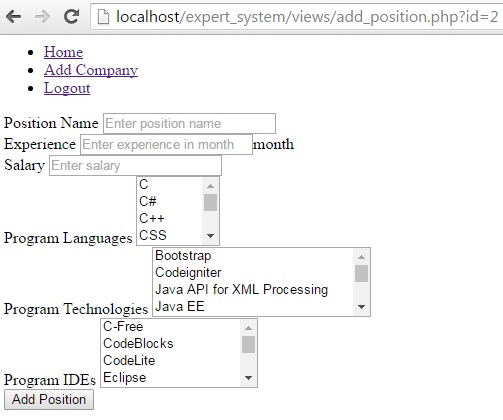
На Фиг.3-а се вижда, че името на дадена компания представлява линк. Този линк пренасочва експерта към следната страница:



*Фиг.6 Детайлна страница на дадена компания*

Тук са показани всички свободни позиции за дадена компания. При невъведени данни таблицата е празна.

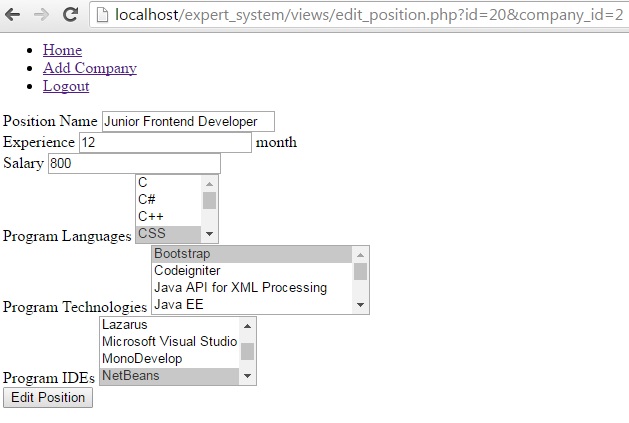
Нова позиция се добавя при влизане в линка „Add New Position”. Той пренасочва експерта към следната страница:



*Фиг.7 Въвеждане на данни за нова позиция*

В тази форма и шестте полета за задължителни, като при последните три може да се избере повече от една опция. При непопълване на дадено поле, под него се изписва съобщение за грешка. При верни данни позицията се записва в системата и се извежда потвърждаващо съобщение.

На Фиг.6 се вижда, че данните за дадена позиция могат да бъдат променени чрез влизане в линка „Edit”. Формата за промяна на дадена позиция изглежда така:



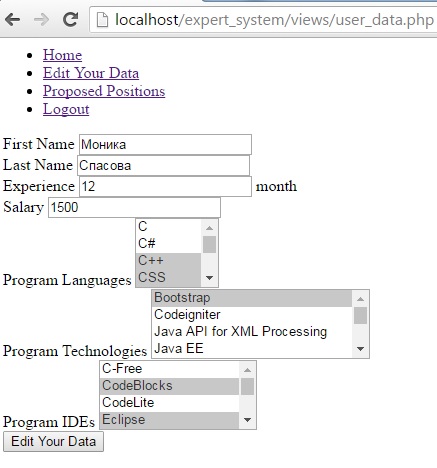
*Фиг.8 Промяна на данни за дадена позиция*

На Фиг.6 също и името на дадена позиция представлява линк към детайлна информация за позицията. Страницата изглежда по следния начин:



*Фиг.9 Детайлна информация на дадена позиция*

В случай, че влезлият в системата потребител е с роля „user”, той може да си добави/промени своите данни, влизайки в линка „Edit Your Data”(Фиг.3-б). При добавяне/промяна на данните е показана следната форма:



*Фиг.10 Промяна на потребителска информация*

И тук всички полета са задължителни, като при първоначално влизане са попълнени единствено първите две полета. При непопълнено поле се изписва съобщение за грешка под него. При успешно запазени данни се извежда потвърждаващо съобщение. За преглед на подходящите позиции спрямо въведените данни на потребителя, трябва да се влезе в линка „Proposed Positions”. Страницата има следния изглед:



*Фиг.11 Предложени позиции спрямо потребителските данни*

На потребителя се дава възможност за селектиране на позиции спрямо зададен критерий. Това става при избор на опция от падащото меню „Find Positions”.

За преглед на детайлна информация на позиция се използва линкът „View More”, който пренасочва потребителя към следната страница:



*Фиг.12 Детайлна информация за позиция*

За излизане от системата се използва линкът „Logout”, който пренасочва потребителя към формата за влизане.

**5.Литература**

1. <http://php.net/manual/en/book.pdo.php> - посетен на 23.05.2016

2. <http://www.w3schools.com/jquery/default.asp> - посетен на 22.05.2016

3. <http://php.net/manual/en/ref.array.php> - посетен на 23.05.2016