A

**Software Requirement Specification**

ON

**“Online Job Portal”**

In Partial Fulfillment of

***Master’s Degree in Computer Application***

University of Pune

**Submitted By:**

*Amit Prakash Sundriyal*

**Under the Guidance of**

*Dr. Darshana Desai*

****

INDIRA COLLEGE OF ENGINEERING AND MANAGEMENT (ICEM)

PARANDWARI, PUNE-410506

2019-2020

**Content**

|  |  |  |
| --- | --- | --- |
| S No. | Project Titles | Page No. |
|  | Introduction   1. Purpose………………………………………………………………… 2. Document Conventions………………………………………………... 3. Product Scope…………………………………………………………. | 4 |
|  | Overall Description   1. Product Perspective……………………………………………………. 2. Product Functions……………………………………………………... 3. User Classes and Characteristics……………………………………… 4. Operating Environment……………………………………………….. 5. Assumptions and Dependencies………………………………………. | 5-7 |
|  | System Features   1. Functional Requirements……………………………………………… | 8 |
|  | External Interface Requirements   1. User Interfaces………………………………………………………… 2. Hardware Interfaces…………………………………………………… 3. Software Interfaces……………………………………………………. 4. Communications Interfaces…………………………………………… | 9 |
|  | Non-Functional Requirements   1. Performance Requirements……………………………………………. 2. Safety Requirements…………………………………………………... 3. Security Requirements………………………………………………… 4. Software Requirements………………………………………………... | 10-12 |
|  | Other Requirements   1. Analysis Model………………………………………………………... 2. Activity Diagram…………………………………………… 3. Class Diagram……………………………………………… 4. Component Diagram……………………………………….. 5. Use Case Diagram…………………………………………. 6. Deployment Diagram………………………………………. 7. Module Hierarchy Diagram………………………………. | 13-23 |
|  | Database Schema   1. Master Tables……………………………………………………… 2. Login………………………………………………………............. 3. Company………………………………………………………....... | 24-30 |
|  | Screens   1. Login………………………………………………………............. 2. Home Page……………………………………………………….... 3. Companies………………………………………………………..... 4. Student Side……………………………………………………….. | 31-43 |

1. **Introduction**
2. **Purpose**

It provides facility to the Job Seeker to search for various jobs as per his qualification. Here Job Seeker can registered himself on the web portal and create his profile along with his educational information. Job Seeker can search various jobs and apply for the Job.

This Portal is also designed for the various employer who required to recruit employees in their organization. Employer can registered himself on the web portal and then he can upload information of various job vacancies in their organization. Employeer can view the applications of Job Seeker and send call latter to the job seekers.

1. **Document Conventions**

This document uses the following conventions: -

Heading Font: 24pt

Subheading Font: 18pt

Content Font: 14pt

Font Style: Times New Roman

1. **Product Scope**

Work can be done in a computerized way, no need of paperwork, to increase the accuracy and efficiency of the admin manages the Employee data and Jobseekers data, Analysis of the overall process.

1. **Overall Description**
2. **Product Perspective**

A distributed Online Job Portal Database system stores the following information:

* **Admin Details**

It includes admin username and password, from which admin can log in.

* **Employee Details**

It includes Company name, Roles, Contact Person’s Number and Email. This information may be used for keeping the records of the Employee or for any other kind of information related to Employee and Company.

* **Jobseekers Details**

It includes Jobseekers Presonal details and their Academics details.

1. **Product Functions**

The major features of online job poral as shown in below: -

* Product and Component Base.
* User Accounts to control the access and maintain Security
* It Contain Better Storage Capacity
* Accuracy of work
* Easy and fastest retrieval information
* Well design report
* Access of any information individually
* Work becomes very speed
* Easy to update information

1. **User Classes and Characteristics**

The system will support three types of user privileges Admin, Jobseekers and Employee. Student will have access to student functions, and the Staff and TPO will have access to both student and placement management functions and Admin have a full access of the whole system.

* The Admin should be able to do the following functions:
  + - * Approve Registration request (Jobseekers and Employee).
      * View all the Jobseekers and Employees details.
      * View Feedback given by Jobseekers.
* The Jobseekers should have following management functionalities:
  + - * Search Jobs.
      * Apply for Jobs.
      * View walkins details.
      * Give Feedback..
* The Employee should be able to do the following functions:
  + - * Post Jobs details.
      * Post Walkins details.
      * View all the Jobseeker’s application.
      * Accept Jobseeker’s application.
      * Delete Jobs.
      * Delete Walkins.

1. **Operating Environment**

Operating environment for the placement management system is as listed below.

* Distributed database
* Client/Server System
* Web Browser
* Database: MySQL
* Platform: Php

1. **Assumptions and Dependencies**

* Each User must have a User ID and Password.
* Server must always run under Windows System.
* Internet connection is a must.
* Text readers should be installed to view the help files.

1. **System Features**
2. **Functional Requirements**
3. **Manage Data**

Using this system, the administrators can maintain the information of each Jobseeker and Employee. For future references, the administrators can store the Employee information along with Jobseeker information.

1. **Upload Information**

Jobseekers will upload all the required information once and their CV, so that there will be no problem of uploading again and again. On clicking the Apply button, they can apply for job. This is where a smart system comes into play, using which you can easily register to only the companies you are interested in and get real-time notifications.

1. **Tracking job application**

Using these Jobseeker are up to date with their job application status at every stage. This also simplifies the management effort by just posting a status and all interested Jobseekers are notified accordingly.

1. **External Interface Requirements**
2. **User Interface**

* Front-end software: Bootstrap 4
* Back-end software: Php, MySQL Database

1. **Hardware Requirements**

* Desktop.
* Internet Connection.

1. **Software Requirements**

Following are the software used for the placement management online application.

|  |  |
| --- | --- |
| **Software used** | **Description** |
| Operating system | We have chosen window operating system for its support and user-friendliness. |
| Database | To save the records, students records we have chosen  MySQL database. |

1. **Communication Interface**

This project supports all types of browsers. We are using simple electronic forms for the Login forms, profile update, and applying etc.

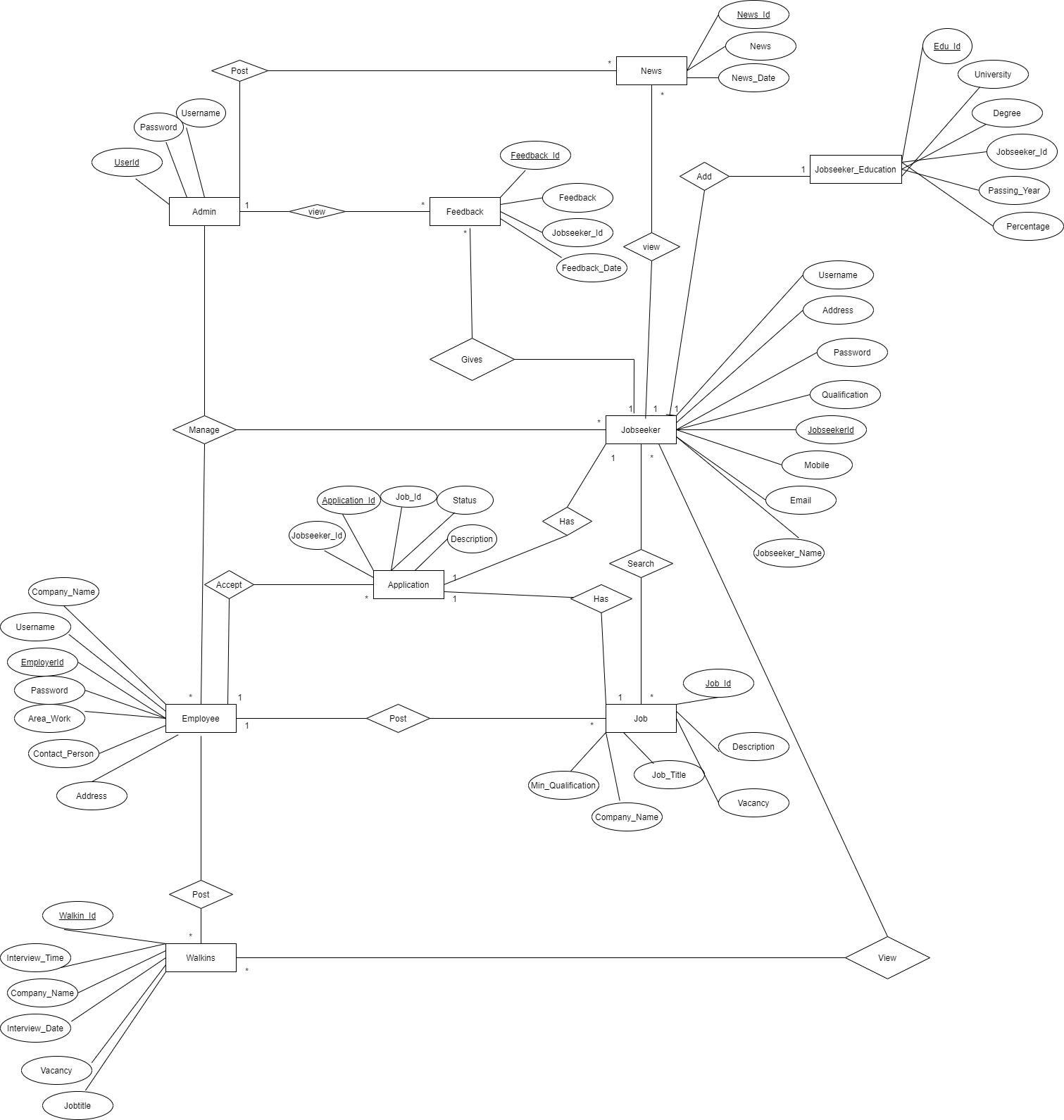
1. **Non-Functional Requirements**
2. **Performance Requirements**

The steps involved to perform the implementation of Placement Cell database are as listed below: -

**E-R Diagram**

The E-R Diagram constitutes a technique for representing the logical structure of a database in a pictorial manner. This analysis is then used to organize data as a relation, normalizing relation and finally obtaining a relation database.

* **Entities**: Which specify distinct real-world items in an application.
* **Properties**/**Attributes**: Which specify properties of an entity and relationships.
* **Relationships**: Which connect entities and represent meaningful dependencies between them.

**

*The diagram shows the ER diagram of Online Job Portal*

1. **Security Requirements**

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed up log, up to the time of failure. Security systems need database storage just like many other applications. However, the special requirements of the security market mean that vendors must choose their database partner carefully.

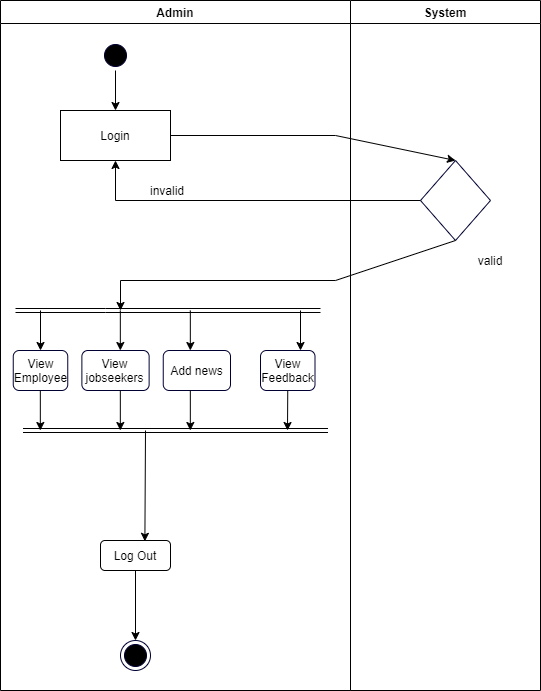
1. **Software Quality Attributes**

* **Availability**: The Jobseekers should be available on the specified date and specified time in the Company.
* **Correctness**: The Jobseekers should reach correct time on Drive location and All data manage in a correct format because of Distributed Database.
* **Maintainability**: Employee should maintain correct schedules of Drive and Locations.
* **Usability**: Post Job notice and Drive notice by using the company data no need to fill all the information about company each and every notice and Jobseeker not need to fill their information for all the companies by single click all the information is stored.
* **Adaptability:** This system is no adaptability requirement.
* **Flexibility:** It includes consideration of all the possible way to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be done based on the future upcoming requirements.
* **Robustness:** Robust data Back-end.

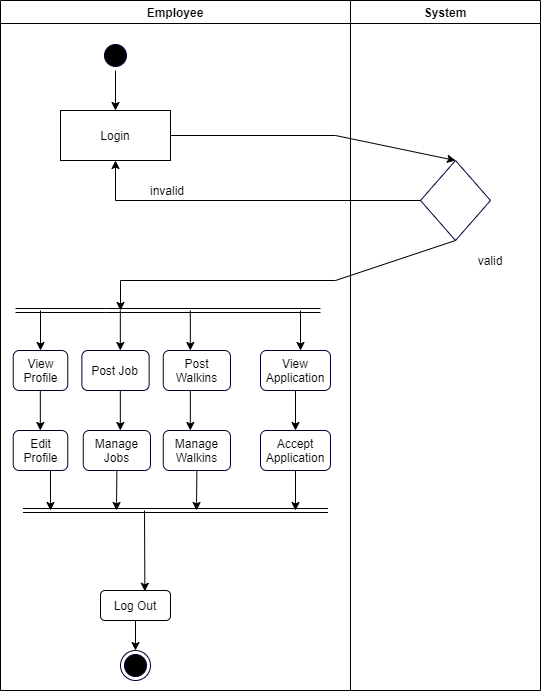
1. **Other Requirements**
2. **Analysis Model**
3. **Activity Diagram**

This is the Activity UML diagram of Online Job Portal System which shows the flows between the activity of Jobseeker, Employee and Admin. The main activity involved in this UML Activity Diagram of Online Job Portal System are as follows:

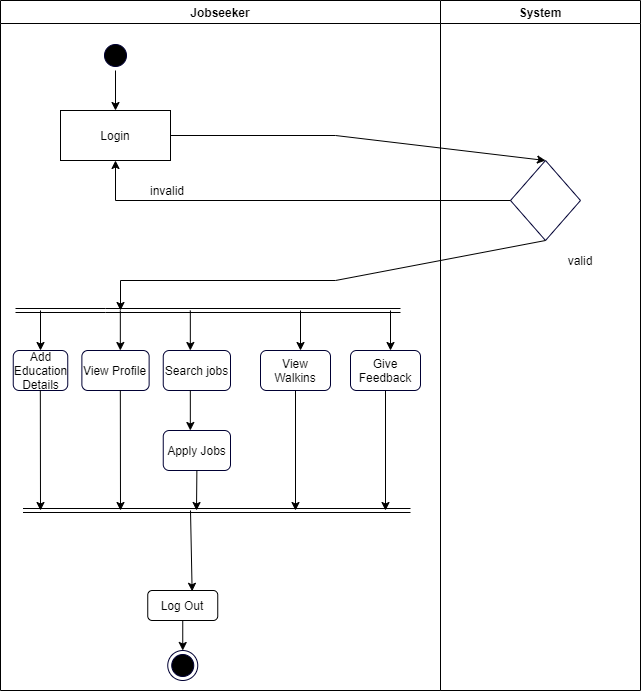
* Admin Activity
* Jobseeker Activity
* Employee Activity

**

*Activity Diagram of Admin*

**

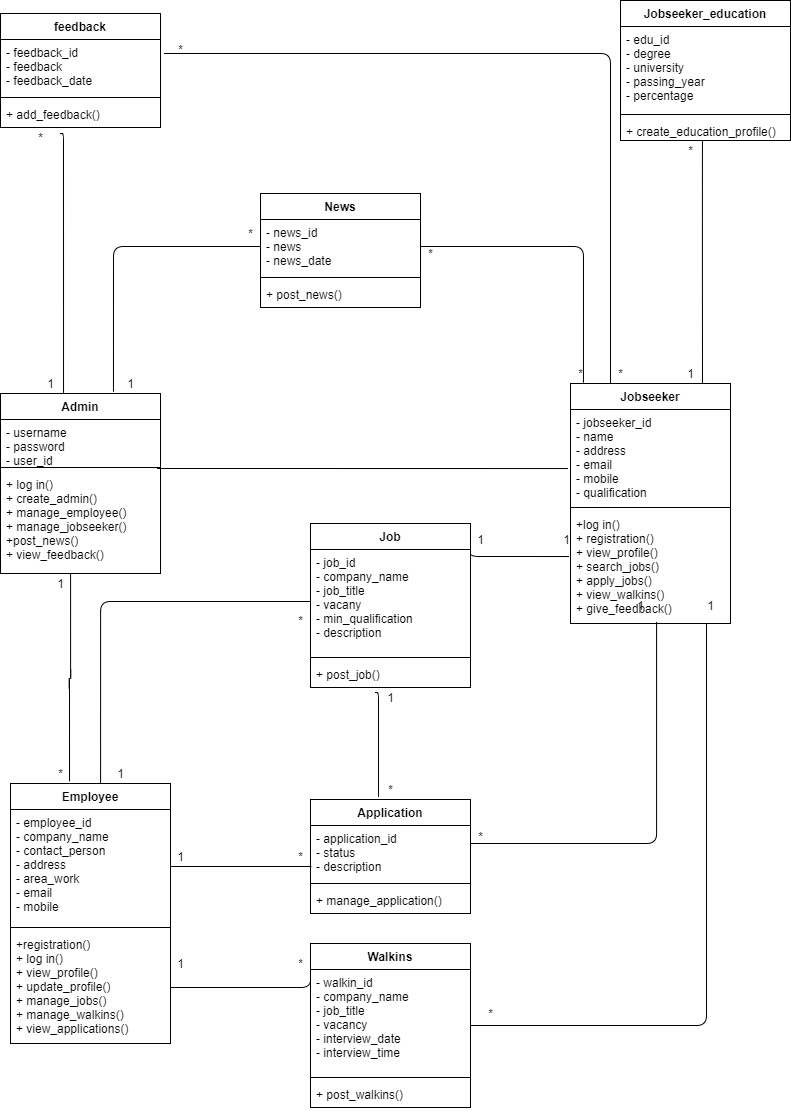
*Activity Diagram of Employee*

**

*Activity Diagram of Jobseeker*

1. **Class Diagram**

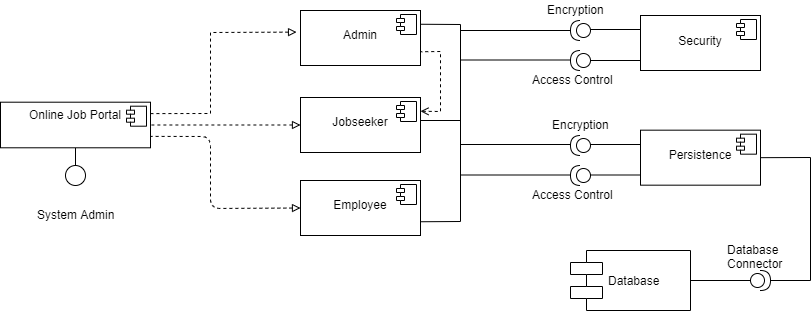
Class Diagram describes the structure of a Online Job Portal System classes, their attributes, operations (or methods), and the relationships among objects. The main classes of the Online Job Portal System are Employee, Jobseeker, Admin, Job, and Application.

**

*Class Diagram for Online Job Portal System*

1. **Component Diagram**

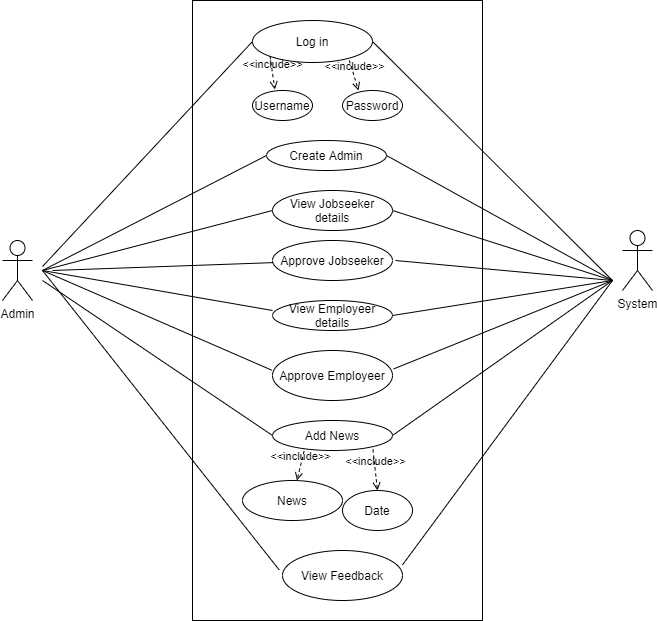
This is a Component diagram of Online Job Portal System which shows components, provided and required interfaces, ports, and relationships between the Admin, Jobseeker, Employee. This type of diagrams is used in Component-Based Development (CBD) to describe systems with Service-Oriented Architecture (SOA). Placement Cell Information System UML component diagram, describes the organization and wiring of the physical components in a system.



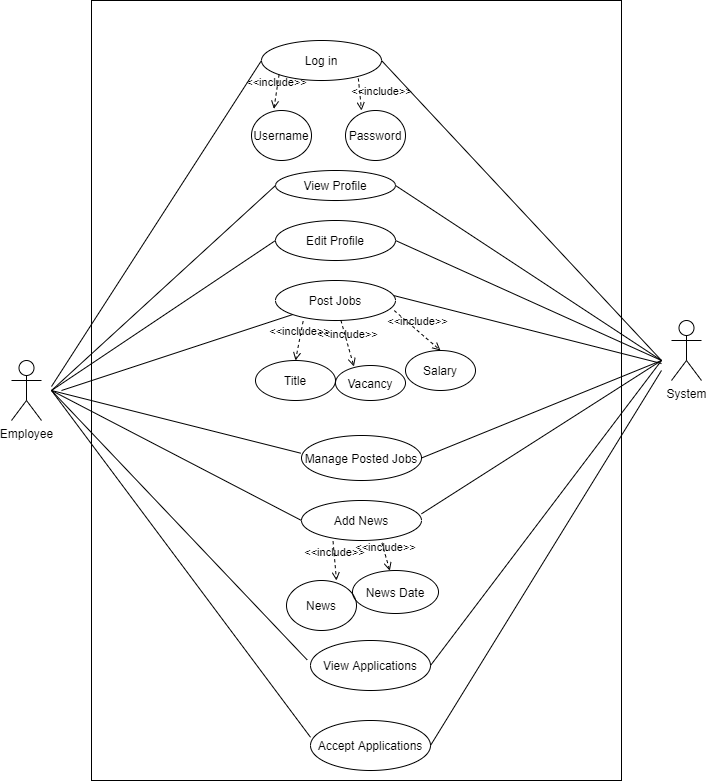
*Component Diagram for Online Job Portal System*

1. **Use Case Diagram**

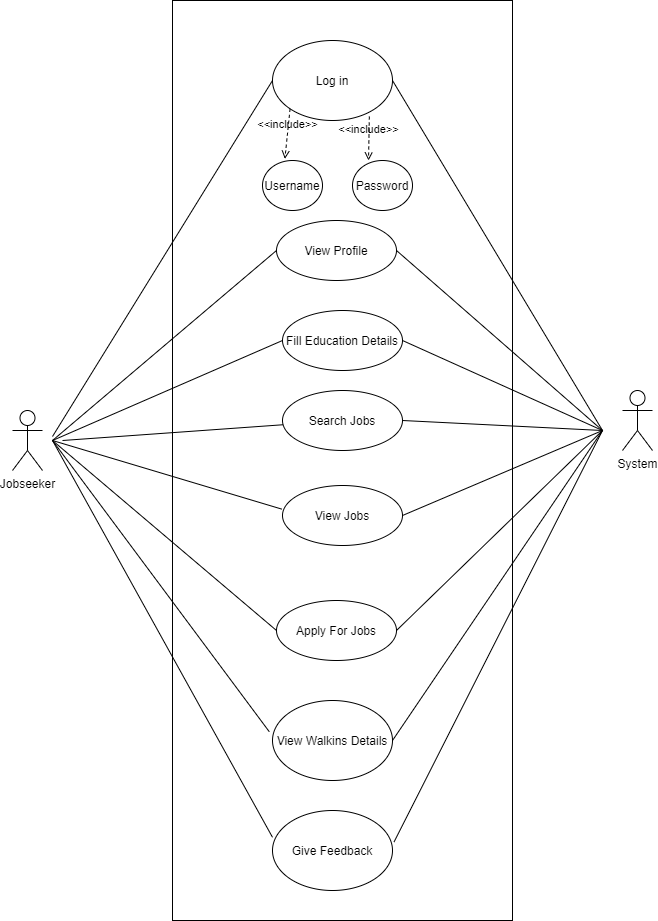
This Use Case Diagram is a graphic depiction of the interactions among the elements of Online Job Portal System. It represents the methodology used in system analysis to identify, clarify, and organize system requirements of Online Job Portal System. The main actors of Online Job Portal System in this Use Case Diagram are: Admin, Employee, and Jobseeker. Major elements of the UML use case diagram of Online Job Portal System are shown on the picture below.



*Use case Diagram for Admin*

**

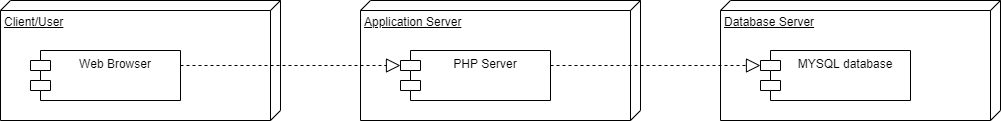
*Use case Diagram for Employee*

**

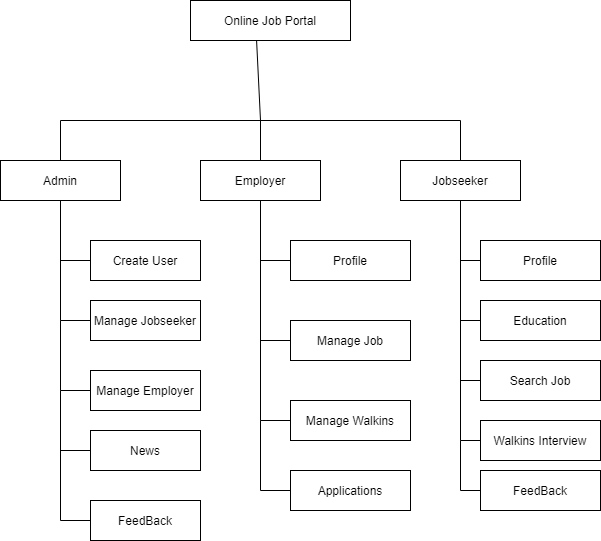
*Use case Diagram for Jobseeker*

1. **Deployment Diagram**

A deployment diagram consists of nodes which describe the physical devices used inside the system. In Online Job Portal System their have a three nodes client/user, application server and Database Server. The database server uses to store all the information about the system and it provide a security of Data. Application server is use to host a web application to the server, and client and user is a platform where it can access that application.



1. **Module Hierarchy Diagram**

The hierarchy chart shows the relationship between various modules. Online Job Portal System have 3 modules Admin, Employee, Jobseeker. These modules have their own modules with their functionality shows blow: -

1. **Database Schema**

**Admin**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Constraint** | **Description** |
| **1** | UserId | int(11) | Not Null | Primary Key | Store Admin Id |
| **2** | UserName | varchar(20) | Not Null | None | Admin’s Username |
| **3** | Password | varchar(20) | Not Null | None | Admin’s Password |

**Jobseeker**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Constraint** | **Description** |
| **1** | JobSeekId | int(11) | Not Null | Primary Key | Store Jobseeker Id |
| **2** | JobSeekerName | varchar(20) | Not Null | None | Jobseeker’s Name |
| **3** | Address | varchar(100) | Not Null | None | Jobseeker’s Address |
| **4** | City | varchar(20) | Not Null | None | Store City Name |
| **5** | Email | varchar(40) | Not Null | None | Jobseeker’s Email |
| **6** | Mobile | bigint(20) | Not Null | None | Jobseeker’s Number |
| **7** | Qualification | varchar(20) | Not Null | None | Jobseeker’s Qualification |
| **8** | Gender | varchar(10) | Not Null | None | Jobseeker’s Gender |
| **9** | BirthDate | date | Not Null | None | Jobseeker’s Birthday |
| **10** | Resume | varchar(200) | Not Null | None | Jobseeker’s Resume |
| **11** | Status | varchar(10) | Not Null | None | Jobseeker’s Registration Status |
| **12** | UserName | varchar(20) | Not Null | None | Jobseeker’s Username |
| **13** | Password | varchar(20) | Not Null | None | Store Password |
| **14** | Question | varchar(100) | Not Null | None | Security Question |
| **15** | Answer | varchar(50) | Not Null | None | Security Answer |

Top of Form

Bottom of Form

**Employee**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Constraint** | **Description** |
| **1** | EmployerId | int(11) | Not Null | Primary Key | Strore Employee id |
| **2** | CompanyName | varchar(20) | Not Null | None | Store Company Name |
| **3** | ContactPerson | varchar(20) | Not Null | None | Contact person of the Company |
| **4** | Address | varchar(100) | Not Null | None | Address of the Company |
| **5** | City | varchar(20) | Not Null | None | Store City Name |
| **6** | Email | varchar(40) | Not Null | None | Contact Person;s Email |
| **7** | Mobile | bigint(20) | Not Null | None | Contact Person;s Number |
| **8** | Area\_Work | varchar(40) | Not Null | None | Company’s Domain |
| **9** | Status | varchar(10) | Not Null | None | Store Status |
| **10** | UserName | varchar(20) | Not Null | None | Employee Username |
| **11** | Password | varchar(20) | Not Null | None | Employee Password |
| **12** | Question | varchar(100) | Not Null | None | Security Question |
| **13** | Answer | varchar(50) | Not Null | None | Security Answer |

**Application**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Constraint** | **Description** |
| **1** | ApplicationId | int(11) | Not Null | Primary Key | Application id |
| **2** | JobSeekId | int(11) | No Null | Foreign Key | Jobseeker id |
| **3** | JobId | int(11) | No Null | Foregin Key | Job id |
| **4** | Status | varchar(30) | No Null | None |  |
| **5** | Description | varchar(200) | No Null | None |  |

**Feedback**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Constraint** | **Description** |
| **1** | FeedbackId | int(11) | Not Null | Primary Key | Store Feedback Id |
| **2** | JobSeekId | int(11) | Not Null | Foreign Key | Jobseeker Id |
| **3** | Feedback | varchar(200) | Not Null | None | Store Feedback |
| **4** | FeedbakDate | date | Not Null | None | Store Feedback Date |

**Jobseeker’s Education**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Constraint** | **Description** |
| **1** | EduId | int(11) | Not Null | Primary Key | Jobseeker’s Education Id |
| **2** | JobSeekId | int(11) | Not Null | Foreign Key | Jobseeker Id |
| **3** | Degree | varchar(20) | Not Null | None | Jobseeker;s Degree |
| **4** | University | varchar(100) | Not Null | None | Store University |
| **5** | PassingYear | mediumint(9) | Not Null | None | Jobseeker passing year |
| **6** | Percentage | float | Not Null | None | Jobseeker’s Percentage |

**Job**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Default** | **Comments** |
| **1** | JobId | int(11) | Not Null | Primary Key | Store Job Id |
| **2** | CompanyName | varchar(20) | Not Null | None | Store Company Name |
| **3** | JobTitle | varchar(50) | Not Null | None | Store Job Title |
| **4** | Vacancy | int(11) | Not Null | None | Total Number Of Vacancies |
| **5** | MinQualification | varchar(50) | Not Null | None | Minimum Qualification Requried |
| **6** | Description | varchar(200) | Not Null | None | Extra Information related to the job |

**News**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Constraint** | **Description** |
| **1** | NewsId | int(11) | Not Null | Primary Key | Store NewsId |
| **2** | News | varchar(200) | Not Null | None | Store News |
| **3** | NewsDate | date | Not Null | None | Store News Date |

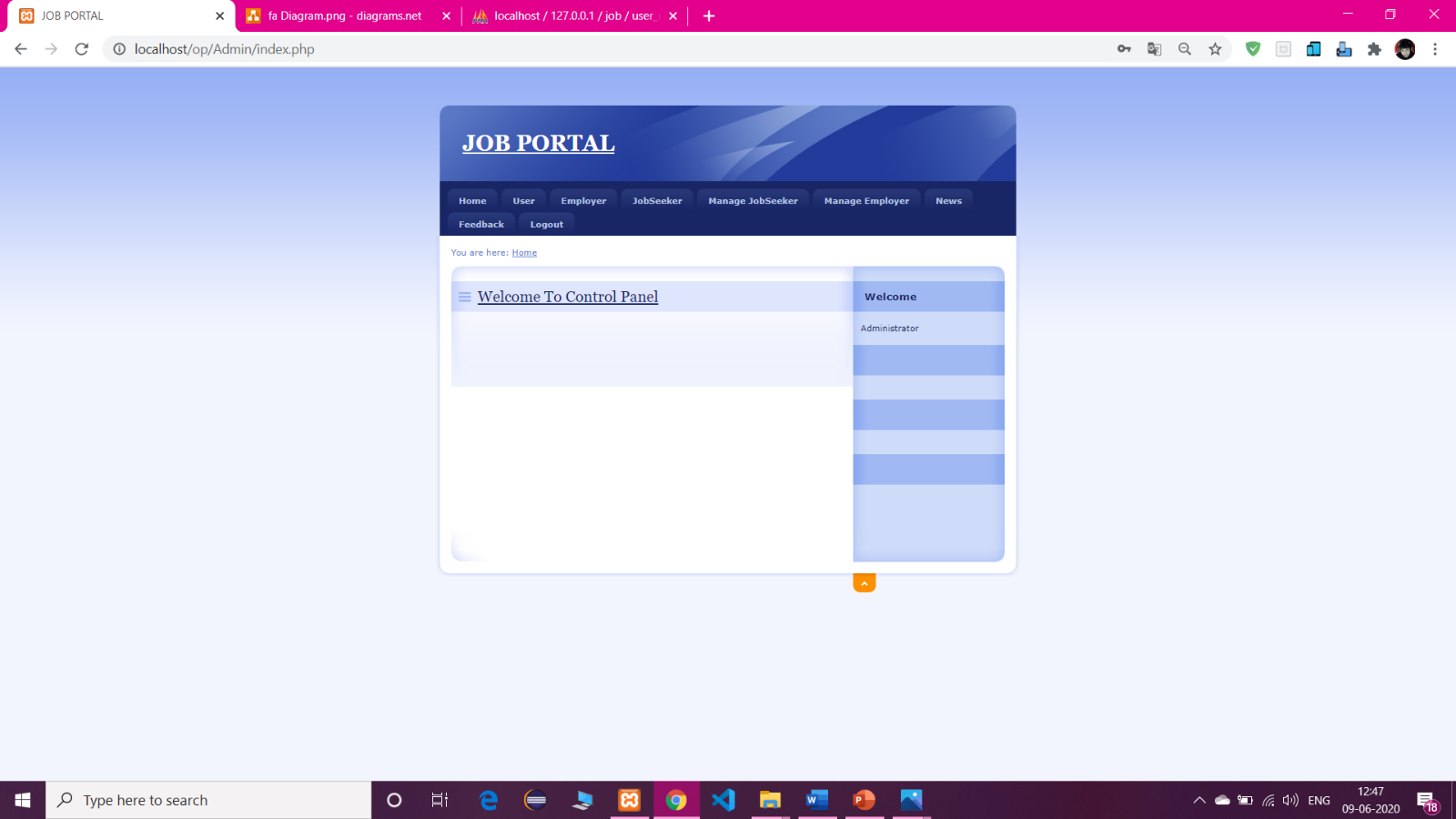
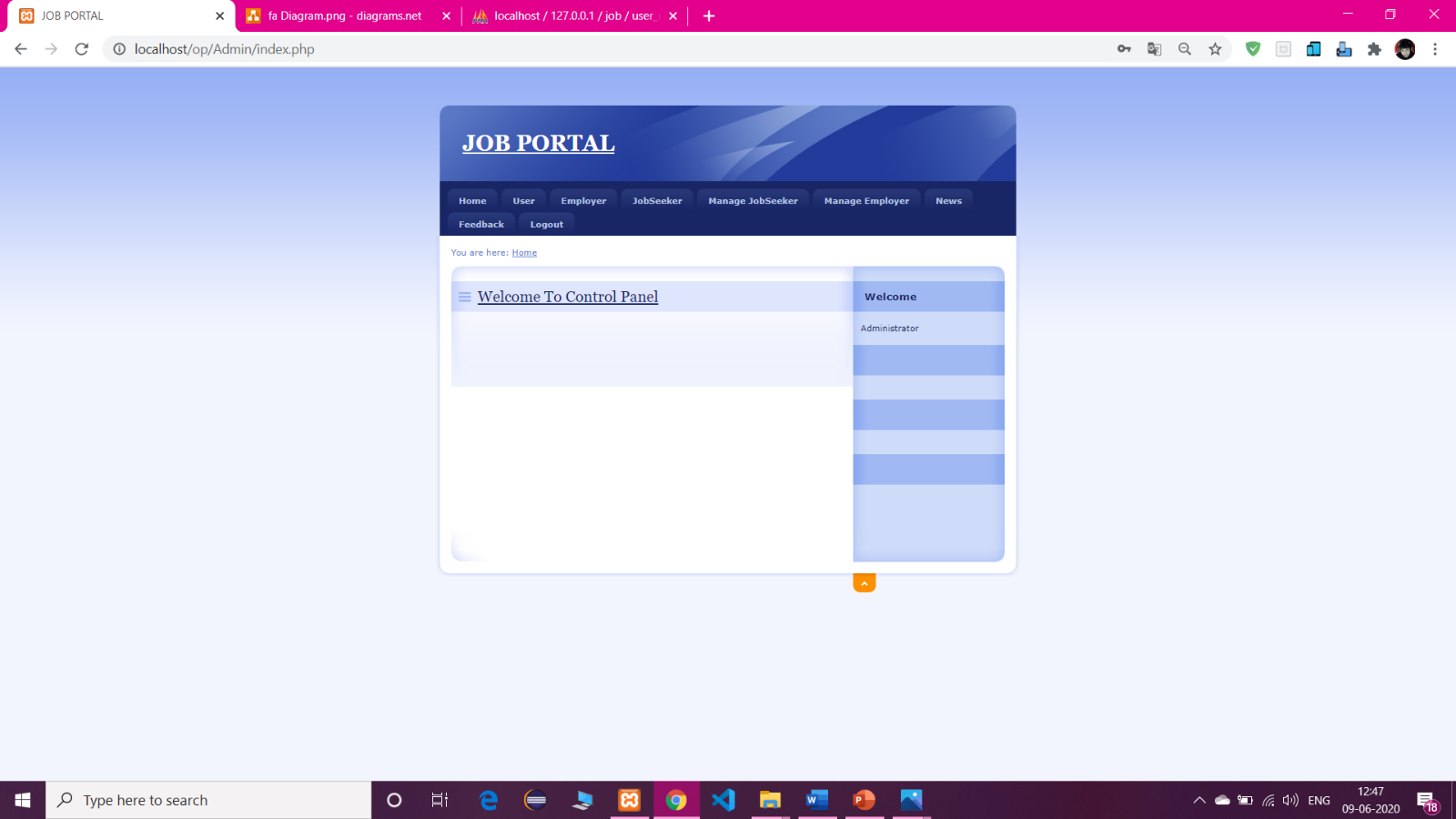
**Walkins**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Type** | **Null** | **Constraint** | **Description** |
| **1** | WalkInId | int(11) | Not Null | Primary Key | Store Walkin Id |
| **2** | CompanyName | varchar(20) | Not Null | None | Store Company Name |
| **3** | JobTitle | varchar(50) | Not Null | None | Store Job title |
| **4** | Vacancy | int(11) | Not Null | None | Total Vacancies |
| **5** | MinQualification | varchar(50) | Not Null | None | Minimum Qualification Required |
| **6** | Description | varchar(100) | Not Null | None | Extra Job Requriment |
| **7** | InterviewDate | date | Not Null | None | Walkin Date |
| **8** | InterviewTime | time | Not Null | None | Walkin Time |

1. **Screens**
2. **Login**

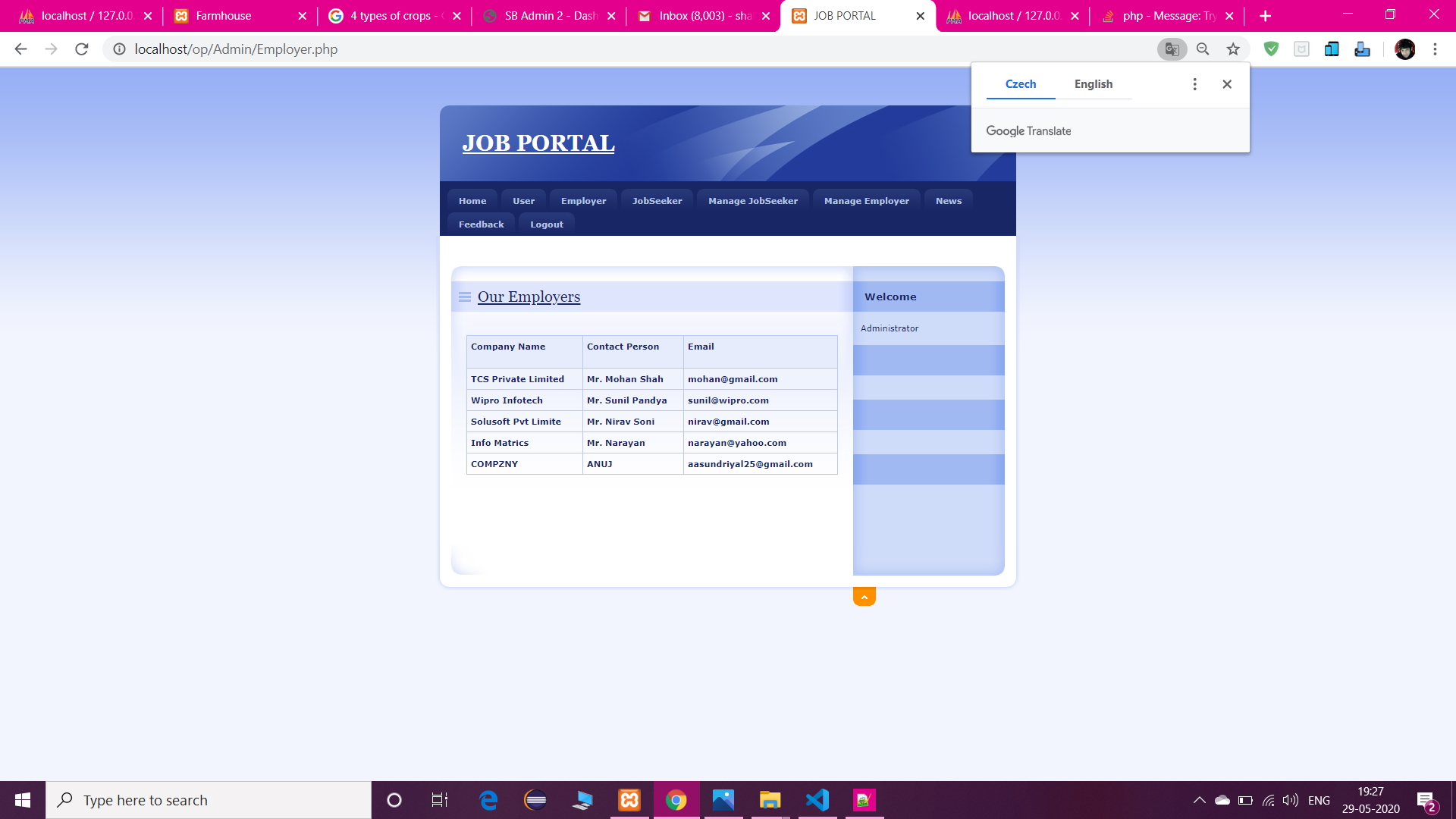
****

* 1. **Home Page**

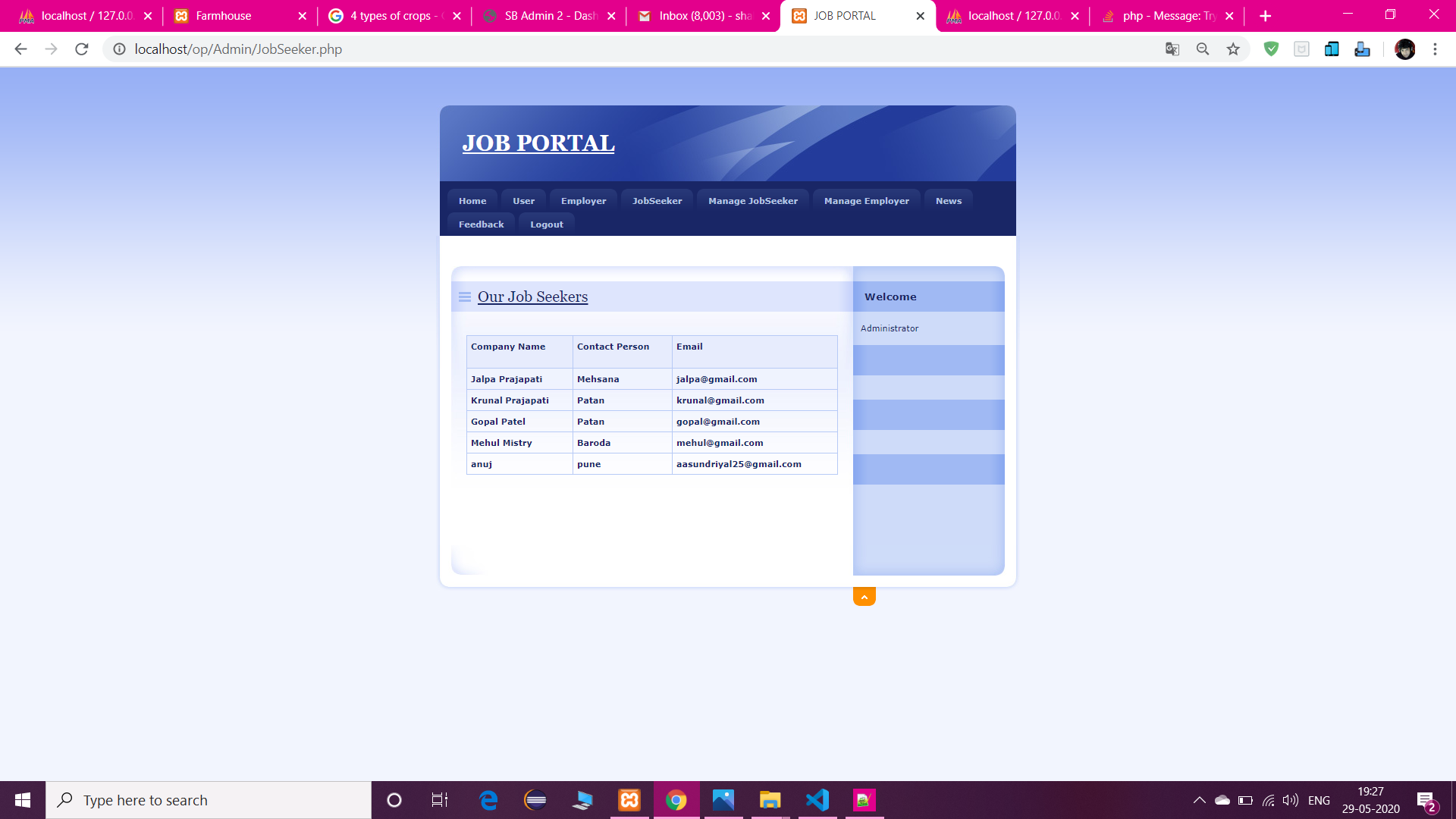
****Top of Form

Bottom of Form

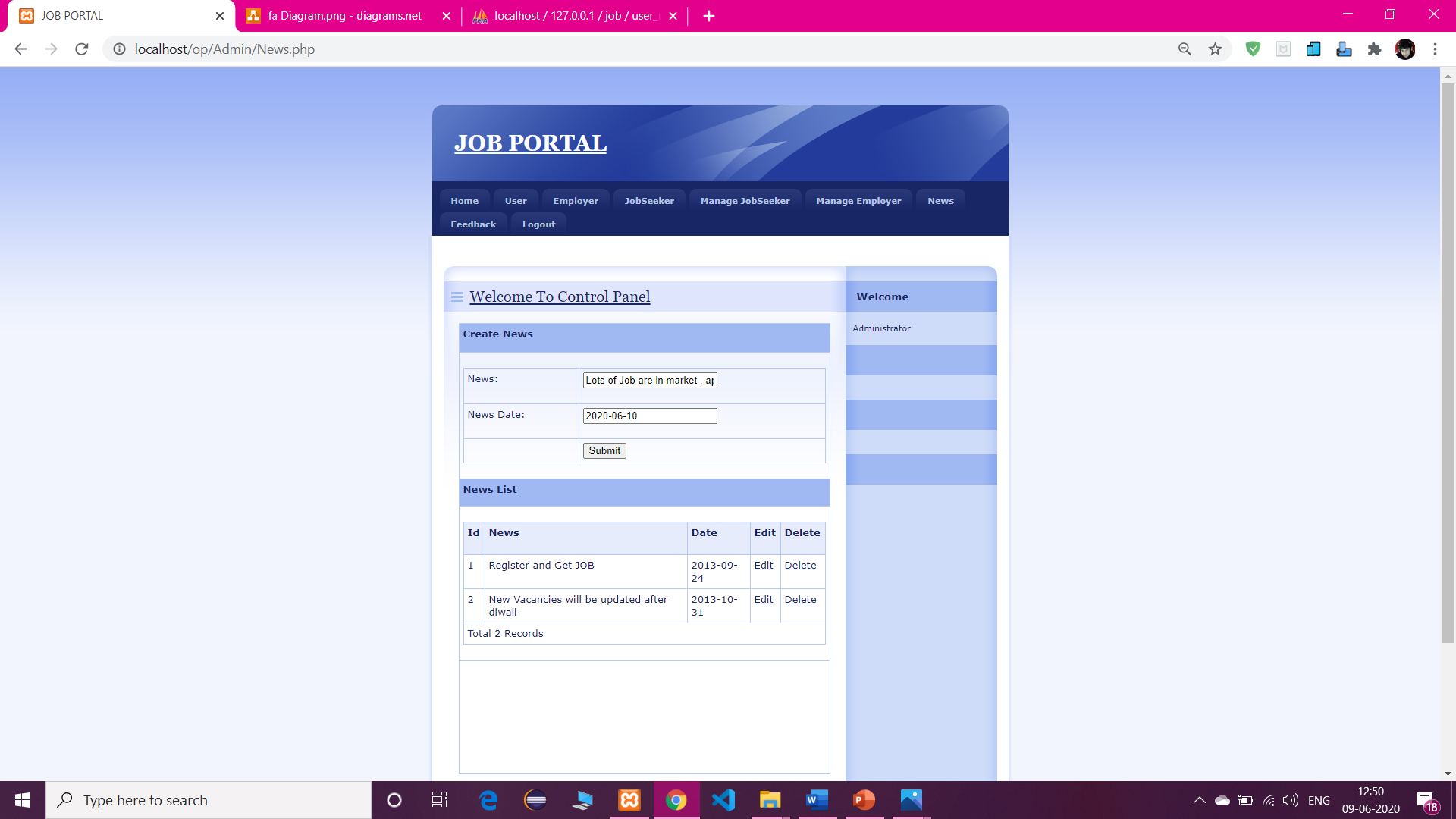
1. **View Employee**

****

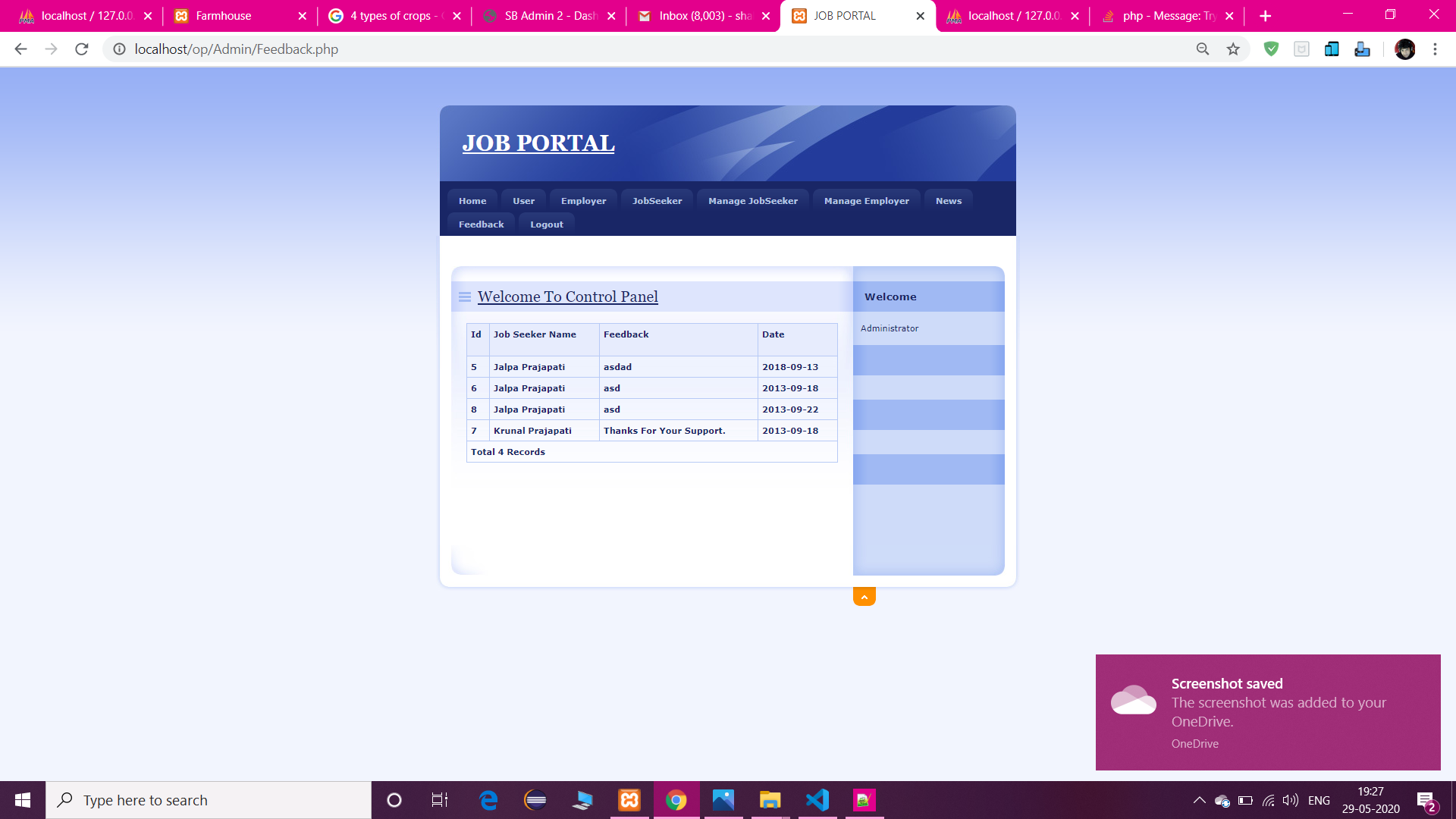
1. **View Jobseeker**

****

1. **Post News**

****

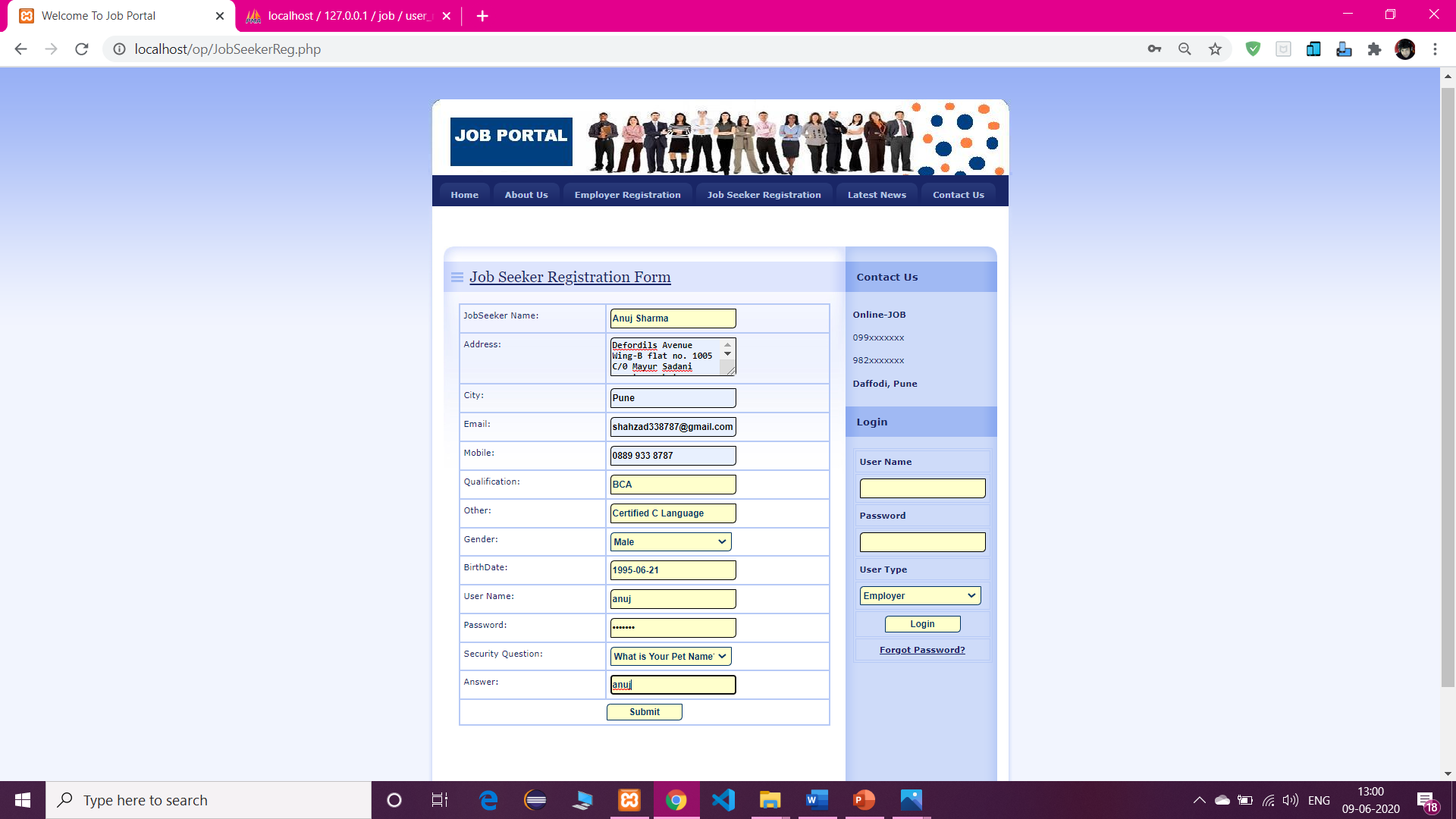
1. **View Feedback**

****

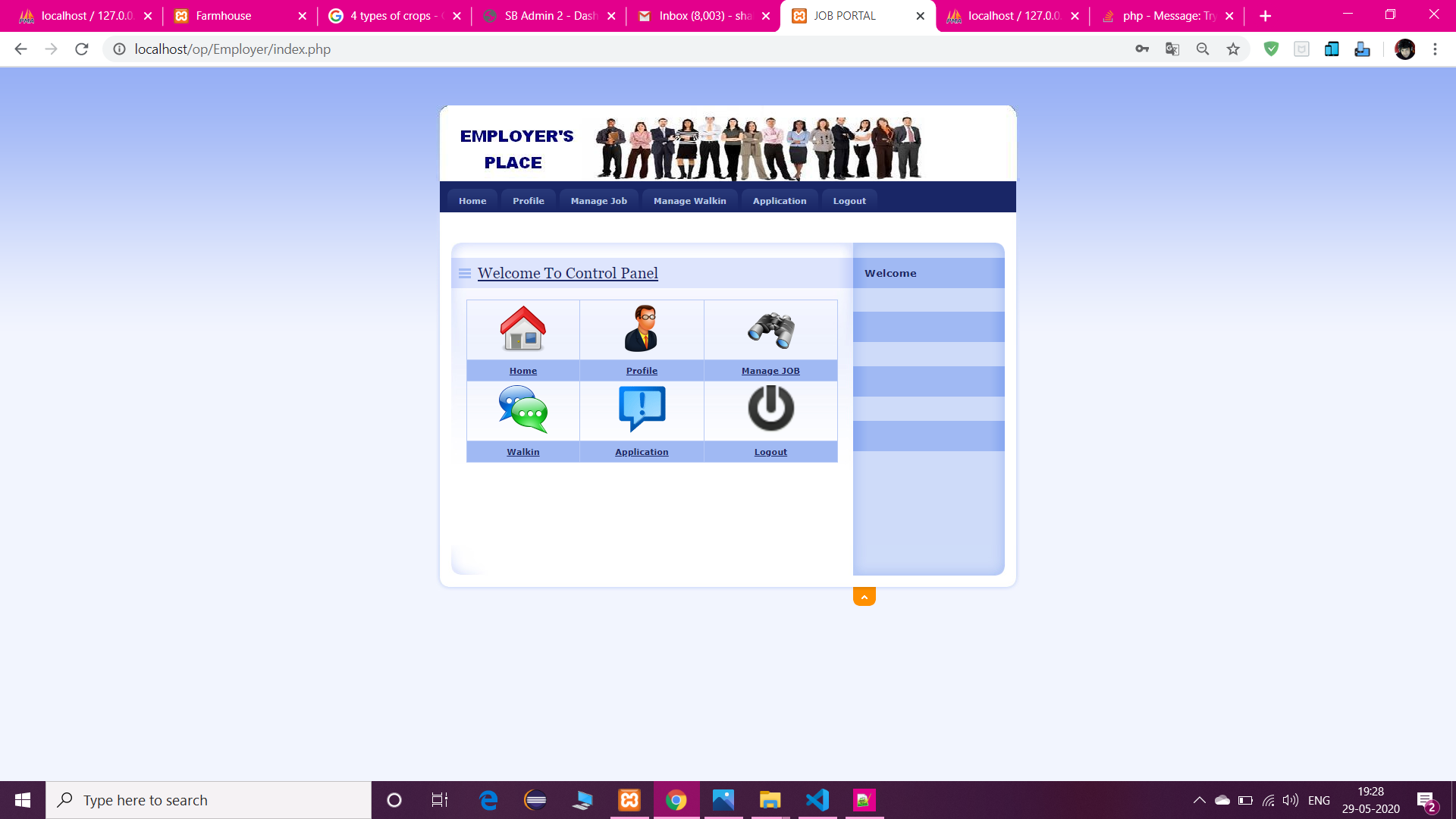
1. **Employee Registration**

****

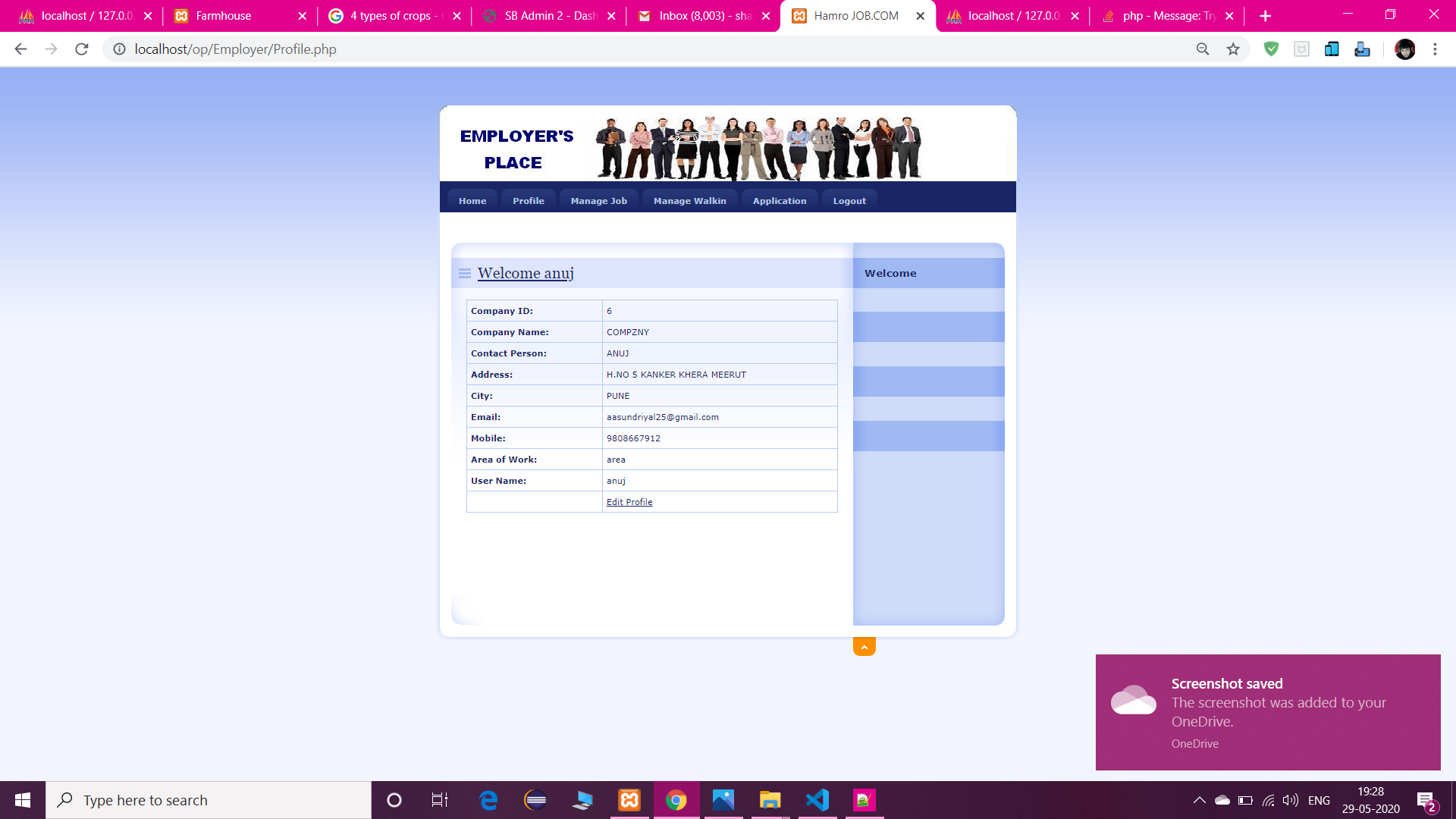
1. **Jobseeker Registration**

****

1. **Employee Dashboard**

****

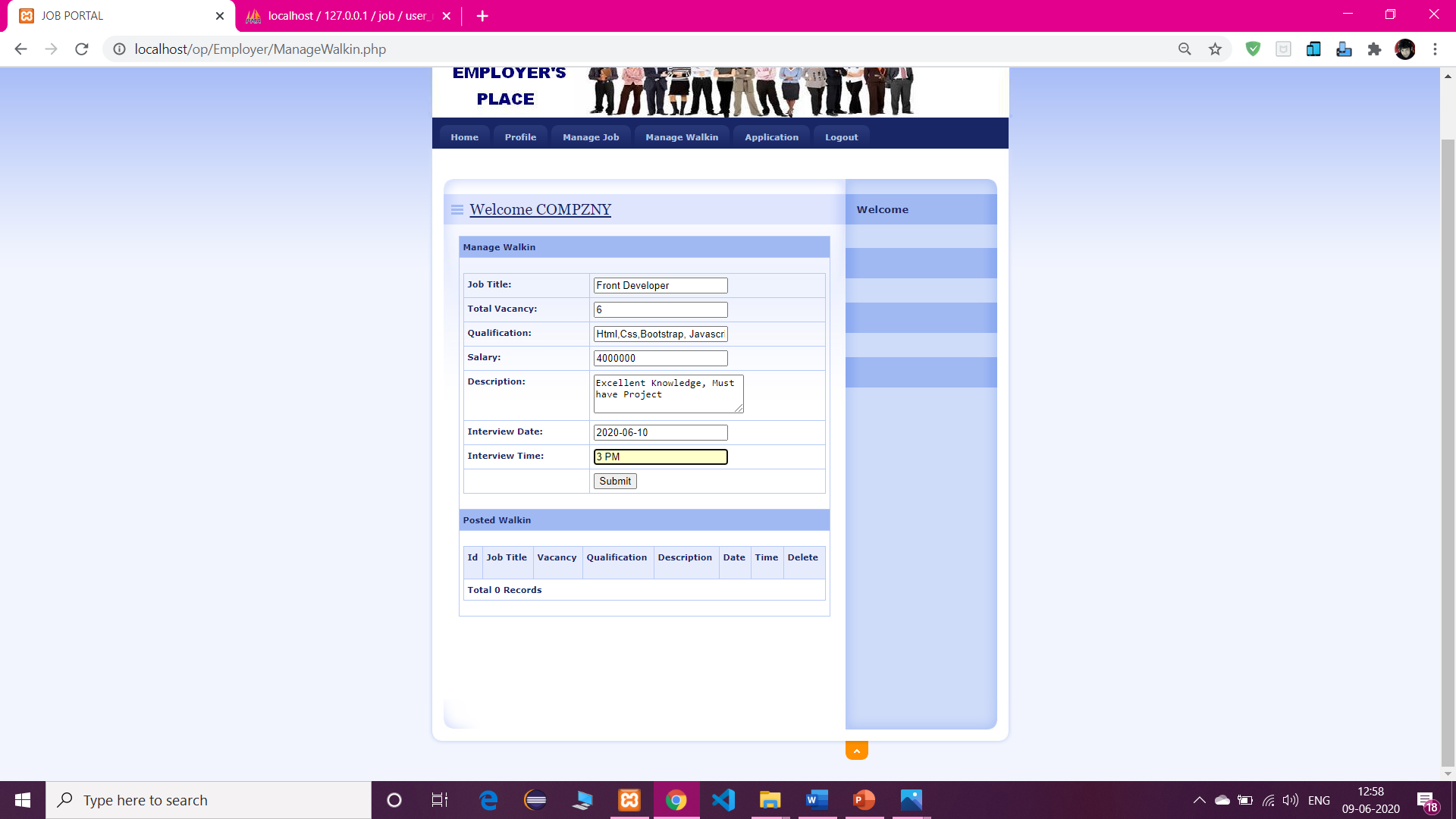
1. **Employee’s Profile**

****

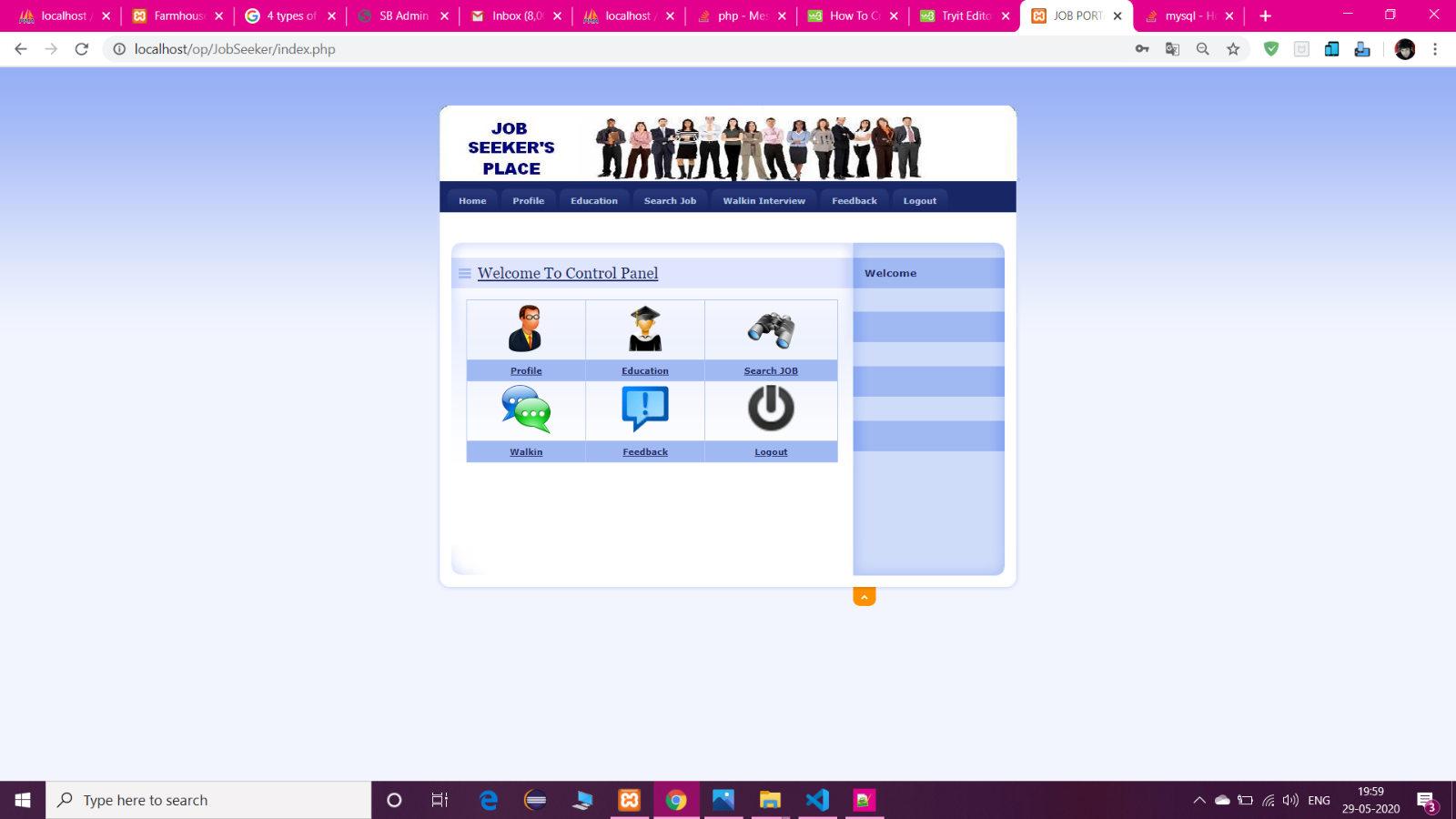
1. **Manage Job**

****

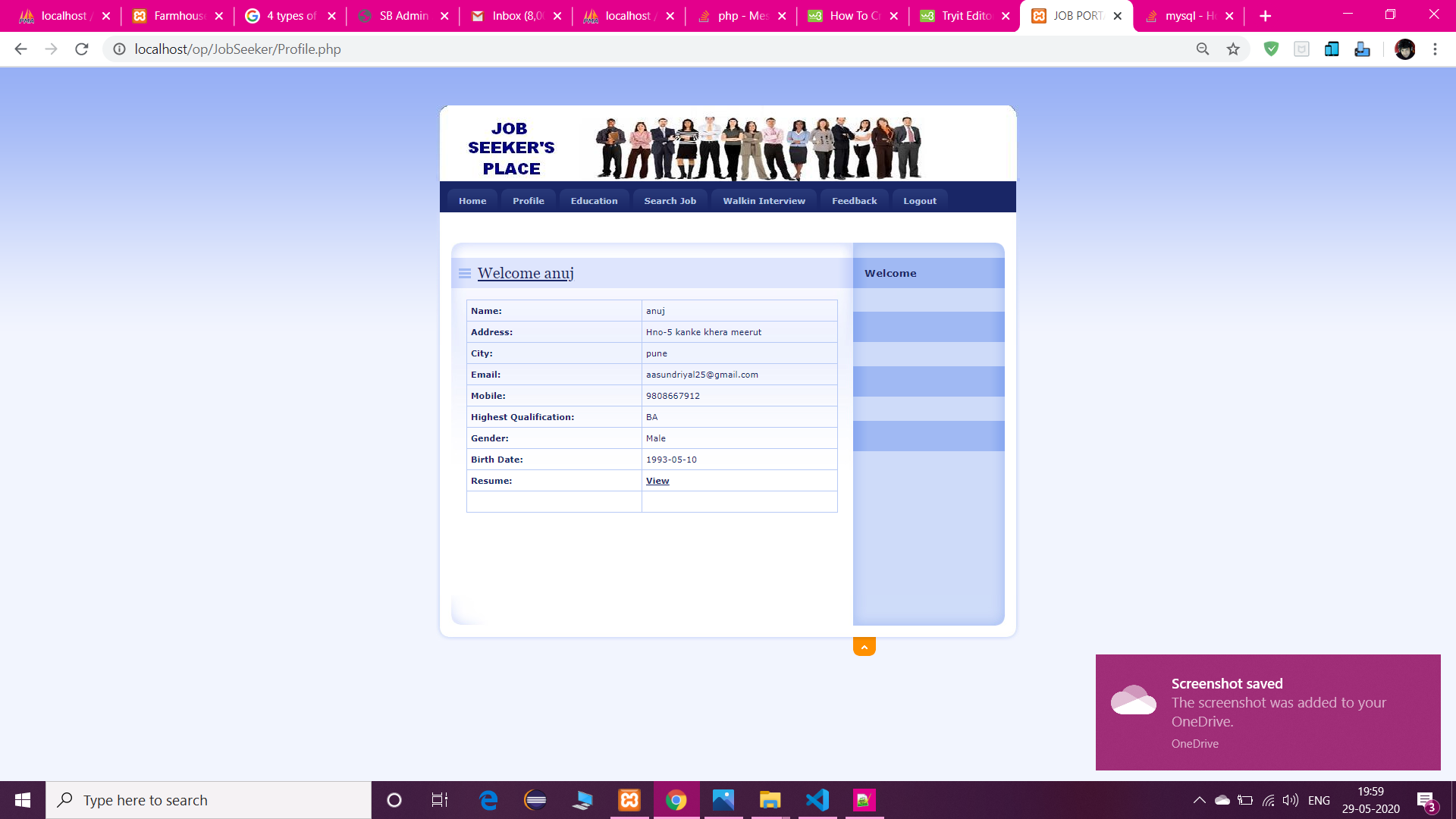
1. **Manage Walkins**

****

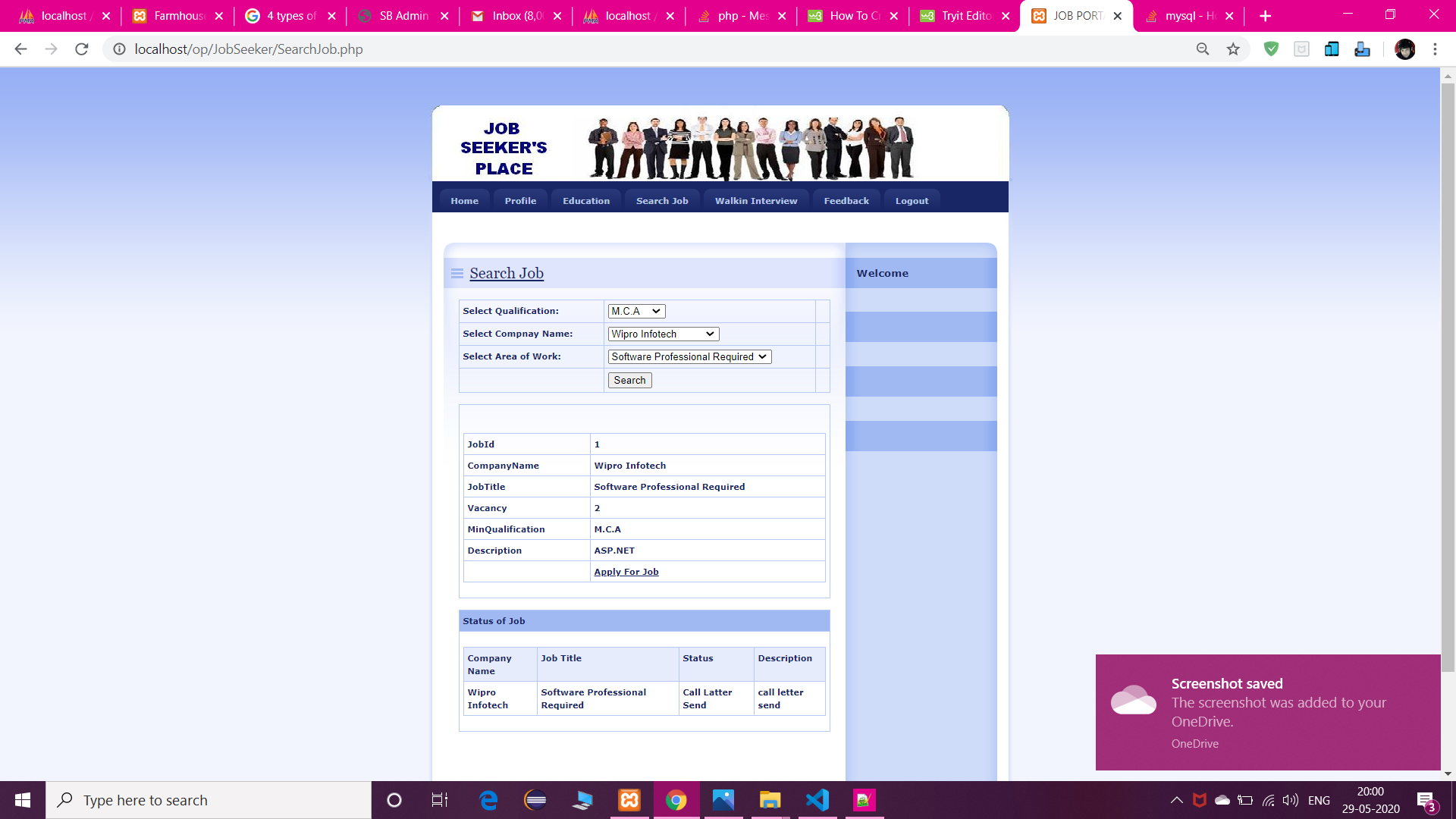
1. **Jobseeker’s Dashboard**

****

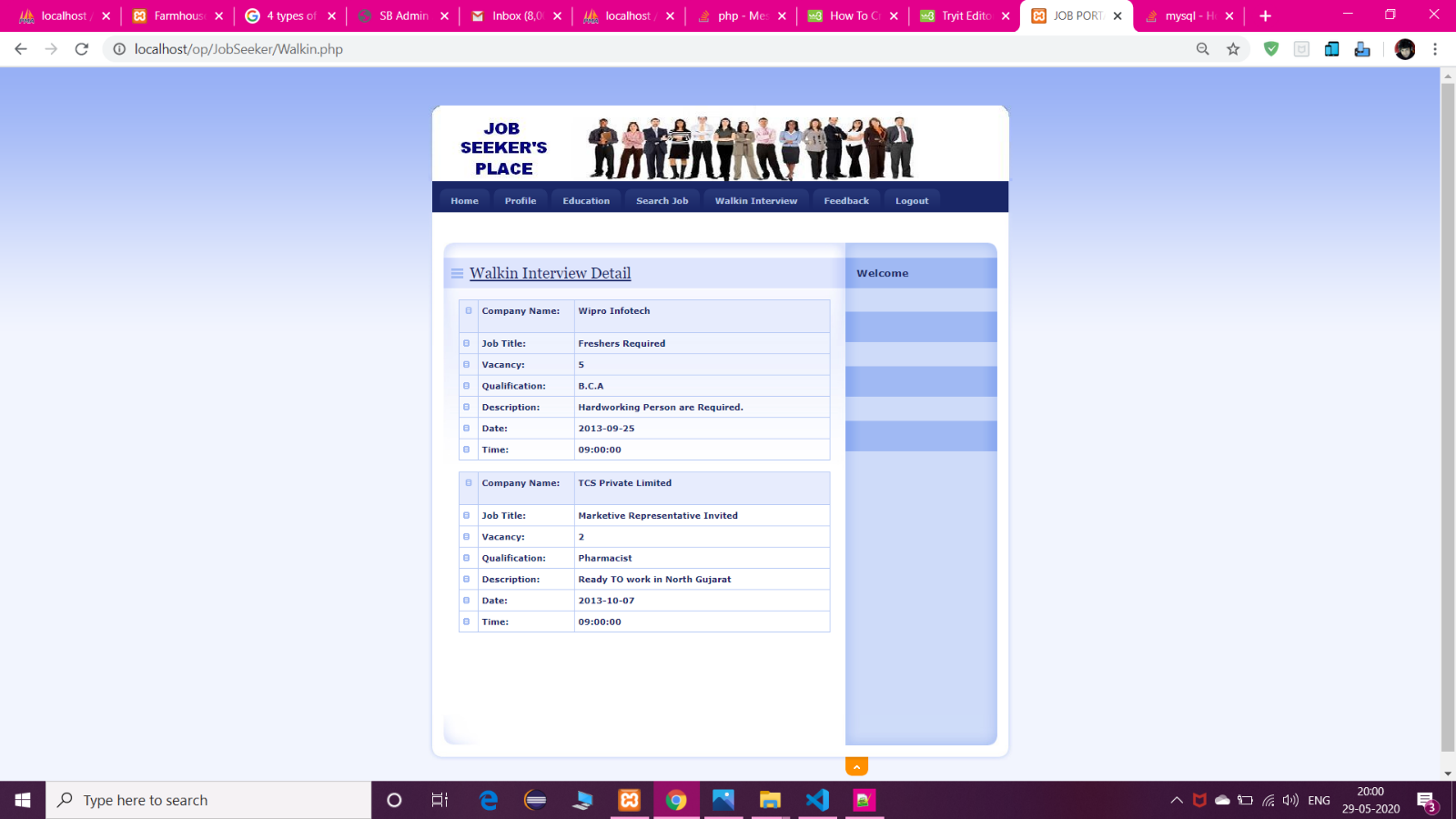
1. **View Profile**

****

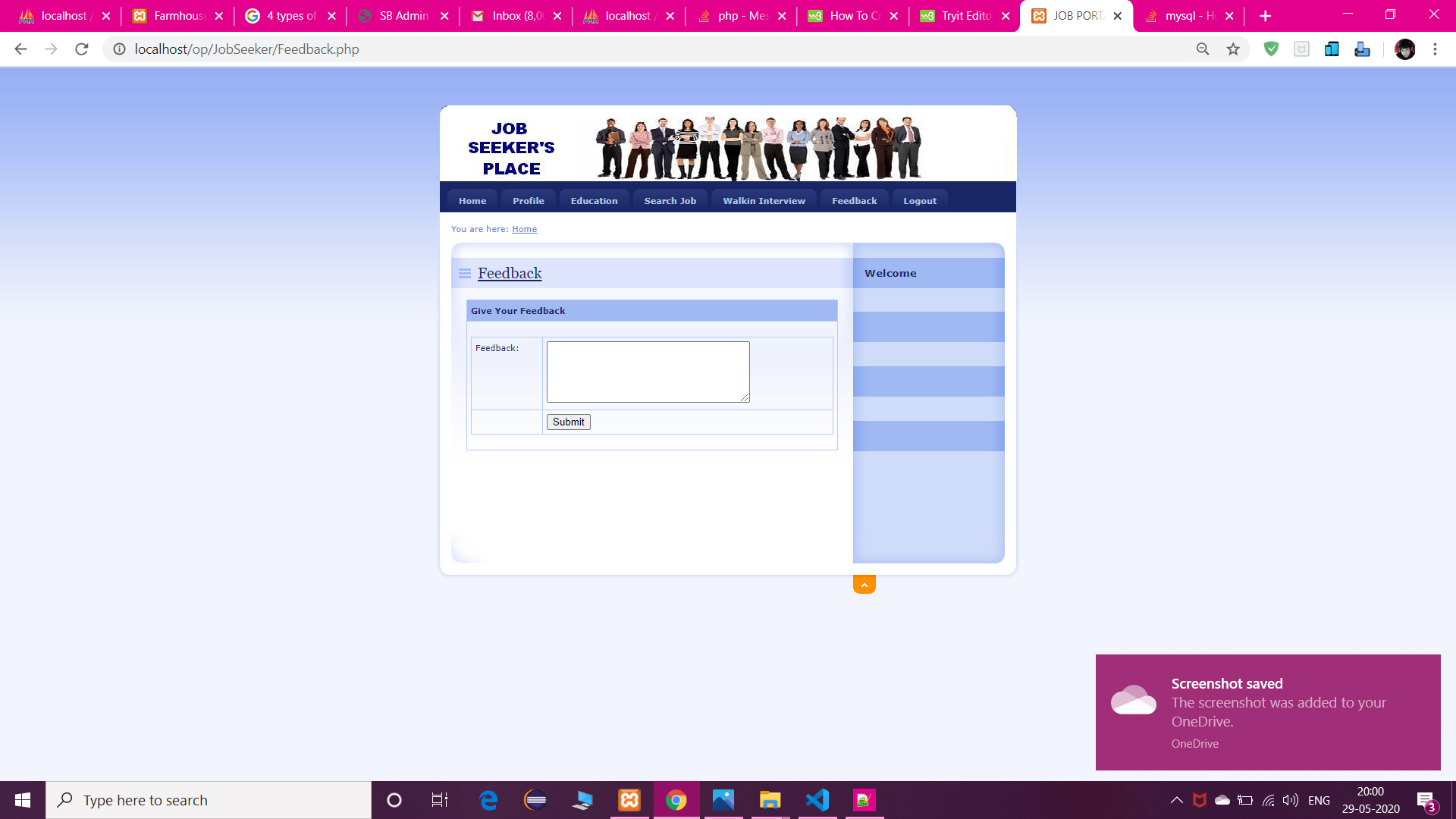
* 1. **Search Job**

****

* 1. **View Walkins Details**

****

* 1. **Feedback**

****

# **Test Procedure and Implementation**

Once source has been generated, software must be tested to uncover and correct as many errors as possible before delivery to the customer.

The test plan is an important product of software design. A test can pre describe various kinds of activities that will be performed to demonstrate that the software product is working as required.

Any engineered test can be tested on one of the following two ways: -

* + 1. WHITE BOX TESTING
    2. BLACK BOX TESTING

**White box Testing**

White box testing is also called glass box testing. It is a test case design that uses control structure of the procedure design to derive test cases. Using white box testing aids, the software engineer can derive test cases that:

1. Guarantee all independent paths within a module have been exercised at least once using mechanism flow graph notation.
2. Exercise all logical decision on their trace and false sides. This is called conditional testing.
3. Execute all loops at their boundaries and within their operational bounds. This is called loop testing.
4. Exercise all internal data structure to ensure their validity. This is called “Data flow testing”.

## **Basis Path Testing: -**

Basis path testing is a white-box testing technique first proposed by *Tom McCabe*. The basis path method enables the test case to derive a logical complexity measure of a procedural design and this measure as a guide for defining a basis set of execution paths. Test cases derived to exercise the basis set are guaranteed to execute every statement in the program at least one time during testing.

## **Condition Testing: -**

Condition testing is a test case design method that exercise the logical conditions contained in a program module. The condition testing method focuses on testing each condition in the program. Condition testing strategies have two advantages, first, measurement of test coverage of a condition is simple, second, and the test coverage of conditions in a program provides guidance for the generation of additional tests for the program.

Purpose of condition testing is to detect not only errors in the conditions of a program but also other errors in the program. *Branch testing* is simplest condition testing strategy.

**Black box Testing**

Black box testing is also called behavioral testing, which focuses on the functional requirements of the software. Black box testing enables the software engineer to derive sets of input condition that will fully exercise the functional requirement for the program.

Black box testing attempts to find error in the following categories:

1. Incorrect or missing function.
2. Interface error.
3. Error in data structure or external database access.
4. Behavior or performance error.
5. Initialization or termination error.

Unlike white-box testing, which is performed early in the testing process, Black box testing tends to be applied during later stages of testing. By applying Black-box techniques, we derive a set of test cases that satisfy the following criteria:

1. Test cases that reduce, by a count that is greater than one, the number of additional test cases that must be designed to achieve reasonable testing and
2. Test cases that tell us something about the presence or absence of classes of errors, rather than an error associated only with the specific test at hand.

**Test Case**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Test Step Description** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 1 | Email Id Validation | While logging Email Id should exist in the database | Same as expected | Pass |
| 2 | Password verification | Password must be more than 6 character | Same as expected | Pass |
| 3 | User should not leave any field blank | User should not leave any field blank accept optional fields | Same as expected | Pass |
| 4 | New User entry | Should be transferred to | Same as  expected | Pass |
| 5 | Verified user Entry point | The verified user should be transferred to directly users own page | Same as expected | Pass |
| 6 | Correct message Delivery | While using the application if there is any error or notification user should be displayed a proper message | Same as expected | Pass |

# **BIBLIOGRAPHY**

**Reference Books**

* Begineerbook.com
* Getbootstrap.com
* MySql: The Complete database
* Udemy