



**2018-2019 Spring**

**CS 353-Database Systems**

**Company Interview and Employment Review Platform**

**Project Proposal**

**Group 4**

**Group Members**

**Mehmet Ege Acıcan - 21602186 Section03**

**Koray Gürses - 21401254 Section01**

**Gülnihal Muslu - 21502606 Section01**

**Can Ozan Kaş - 21501228 Section02**

**Teaching Asistant**

**Mustafa Can Çavdar**

**Due Date: 05.04.2019**

# Table of Contents

<b>1. REVISED E/R DIAGRAM.....</b>	<b>3</b>
<b>2.RELATION SCHEMAS.....</b>	<b>5</b>
2.1. User.....	5
2.2. RegularUser.....	6
2.3. Admin.....	6
2.4. Job.....	7
2.5. Applies.....	8
2.6. Posts.....	9
2.7. Review.....	10
2.8. Interview.....	10
2.9. Employee Review.....	11
2.10. Company.....	12
2.11. Follows.....	13
2.12. Offers.....	13
2.13. Works.....	14
2.14. Request.....	15
2.15. Chat.....	16
2.16. Ban.....	17
2.17. Hires.....	17
<b>3. FUNCTIONAL DEPENDENCIES AND NORMALIZATION OF TABLES.....</b>	<b>18</b>
<b>4. FUNCTIONAL COMPONENTS.....</b>	<b>18</b>
4.1. Use Case and Scenarios.....	18
4.1.1.....	19
4.1.2.....	20
4.1.3.....	21
4.2. Algorithms.....	22
4.2.1. Deletion of a Review Algorithm.....	22
4.2.2. Job Application Algorithm.....	22
4.2.3. Ranking Algorithm.....	22
4.2.4. Ban System.....	22
4.3. Data Structures.....	23
<b>5. USER INTERFACE AND CORRESPONDING SQL STATEMENTS.....</b>	<b>23</b>
5.1. Signup Page.....	23
5.2. Login Page.....	24
5.3. Reset Page.....	25
5.4. Followed Companies Page.....	26

<b>5.5. Applied Jobs Page.....</b>	<b>27</b>
<b>5.6. Home Screen Page.....</b>	<b>28</b>
<b>5.7. User's Profile Settings Page.....</b>	<b>30</b>
<b>5.8. View Company's Interview Review Page.....</b>	<b>33</b>
<b>5.9. Create Job Page.....</b>	<b>35</b>
<b>5.10. Chat Page.....</b>	<b>37</b>
<b>5.11. Shared UI Designs.....</b>	<b>38</b>
5.11.1. Home Screen.....	38
5.11.2. Sign Up Page.....	39
5.11.3. Login Page.....	40
5.11.4. Reset Password Page.....	40
<b>5.12. UI Design For Users.....</b>	<b>41</b>
5.12.1. Edit Profile(Settings).....	41
5.12.2. Post Review.....	42
5.12.3. Chat.....	42
5.12.4. Applied Jobs.....	43
5.12.5. Followed Companies.....	43
5.12.6. View Employee Review For Company.....	44
5.12.7. View Interview Review For Company.....	44
5.12.8. View Jobs For Company.....	45
<b>5.13. UI Design For Company.....</b>	<b>46</b>
5.13.1. Company's Edit Profile Page.....	46
5.13.2. Check Candidate Page.....	46
5.13.3. Create Job Page.....	47
5.13.4. View Interview Review For Company.....	47
5.13.5. View Jobs For Company.....	48
<b>5.14. UI Design For Admins.....</b>	<b>48</b>
5.14.1. Check Request Page.....	48
<b>6. ADVANCED DATABASE COMPONENTS.....</b>	<b>49</b>
<b>6.1. Top10 Users.....</b>	<b>49</b>
<b>6.2. Sorting Reviews For a Particular Company.....</b>	<b>49</b>
<b>6.3. Follow Rank.....</b>	<b>49</b>
<b>6.4. Triggers.....</b>	<b>50</b>
<b>7.IMPLEMENTATION PLAN.....</b>	<b>50</b>
<b>8.WEBSITE.....</b>	<b>50</b>

# 1. REVISED E/R DIAGRAM

Considering the feedback we have got from our teaching assistant, to have a more accurate database system structure, we made changes below:

- User entity is the parent of RegularUser entity and Admin entity. RegularUser entity has extra attribute banTime and now, there is a relation named ban between Admin and RegularUser.
- Review entity is now the parent of Interview and EmployeeReview entities. It also has the attributes LikeCount and ViewCount. Interview entity has only one attribute which is InterviewProcess and EmployeeReview entity has the attributes Pros, Cons, Position, SalaryInfo and WorkExp.
- Admin, Review and Company entities are now related with a ternary relation named Request.
- RegularUser has two relation with Company entity. The entity Works has the attributes Position, Salary and WorkSince.
- We aggregated Company, Admin, Review, Interview and EmployeeReview entities. RegularUser has a relation named Posts with this aggregation. Aggregation makes total participation to the relation Posts.
- We removed all foreign key representations from the diagram.
- Job entity is now a weak entity and makes total participation to the relation Offers.
- We added a new feature that allow users to chat with each other. Also a ranking system with the the attributes of entity Review, LikeCount and ViewCount.
- CompanyPassword and CompanyEmail for Company entity added to allow them sign up.
- Position attribute is added to the Job entity.



## 2.RELATION SCHEMAS

### 2.1.User

#### Relational Model:

User(UserID, username, password, email, UserResume, address, age, phonenumber)

#### Functional Dependencies:

UserID -> username, password, email, UserResume, address, age, phonenumber

username -> UserID, password, email, UserResume, address, age, phonenumber

email -> UserID, username, password, UserResume, address, age, phonenumber

#### Candidate Keys:

{{(UserID), (username), (email)}}

#### Normal Form: BCNF

#### Table Definition:

```
Create table User(  
    UserID int Primary Key Not NULL Auto Increment,  
    Username varchar(32) Not NULL Unique,  
    UserResume varchar(255),  
    Password varchar(32) Not NULL,  
    Address varchar(100),  
    Email varchar(75) Not NULL Unique,  
    Age int Check (Age >= 18),  
    Phonenumber varchar(10)  
);
```

## 2.2.Regular User

### Relational Model:

RegularUser(UserID, Bantime)

### Functional Dependencies:

UserID -> Bantime

### Candidate Keys:

{(UserID)}

### Normal Form: BCNF

### Table Definition:

```
Create table RegularUser(  
    UserID int,  
    Bantime int,  
    Foreign Key (UserID) References User(UserID)  
)ENGINE = InnoDB;
```

## 2.3.Admin

### Relational Model:

Admin(UserID)

### Functional Dependencies:

None

**Candidate Keys:**

{{UserID}}

**Normal Form:** BCNF

**Table Definition:**

```
Create table Admin(  
    UserID int,  
    Foreign Key (UserID) References User(UserID)  
)ENGINE = InnoDB;
```

## 2.4.Job

**Relational Model:**

Job(JobID, JobName, Salary, JobDescription, Position)

**Functional Dependencies:**

jobID, jobName -> salary, JobDescription, Position

**Candidate Keys:**

{{JobID, jobName}}

**Normal Form:** BCNF

**Table Definition:**

```
Create Table Job(
```



```
JobID int,  
JobName varchar(32) Not NULL,  
Salary int Not NULL,  
JobDescription varchar(100) Not NULL,  
Position varchar(32) Not NULL,  
Foreign Key (JobID) References Company(CompanyID),  
Primary Key (JobID, JobName)  
)ENGINE = InnoDB;
```

## 2.5.Applies

### Relational Model:

Applies(JobID, UserID, JobName, CompanyID, State)

### Functional Dependencies:

JobID, UserID, JobName, CompanyID -> State

### Candidate Key:

{(JobID, UserID, JobName, CompanyID)}

### Normal Form: BCNF

### Table Definition:

```
Create Table Applies(  
    State int,  
    CompanyID int,  
    JobID int,  
    UserID int,  
    JobName varchar(32) Not NULL,
```

Foreign Key (JobID) References Job(JobID),  
Foreign Key (UserID) References User(RegularUser),  
Foreign Key (CompanyID) References Company(CompanyID),  
Primary Key (JobID, CompanyID, UserID, JobName)  
)ENGINE = InnoDB;

## 2.6.Posts

### Relational Model:

Posts(UserID, ReviewID, Anonymous)

### Functional Dependencies:

UserID, ReviewID -> Anonymous

### Candidate Key:

{(JobID, ReviewID)}

### Normal Form: BCNF

### Table Definition:

Create Table Posts(  
    UserID int,  
    ReviewID int,  
    Anonymous Boolean Not NULL,  
    Foreign Key (ReviewID) References (Review),  
    Foreign Key (UserID) References (RegularUser),  
    Primary Key (UserID, ReviewID)  
)ENGINE = InnoDB;

## 2.7.Review

### Relational Model:

Review(ReviewID, LikeCount, ViewCount)

### Functional Dependencies:

ReviewID -> LikeCount, ViewCount

### Candidate Key:

{(ReviewID)}

### Normal Form: BCNF

### Table Definition:

```
Create Table Review(  
    ReviewID int Primary Key Auto Increment Not NULL,  
    LikeCount int,  
    ViewCount int  
);
```

## 2.8.Interview

### Relational Model:

Interview(ReviewID, InterviewProcess)

### Functional Dependencies:

ReviewID -> InterviewProcess

**Candidate Key:**

{(ReviewID)}

**Normal Form:** BCNF

**Table Definition:**

```
Create Table Interview(  
    ReviewID int,  
    InterviewProcess varchar(255),  
    Foreign Key (ReviewID) References Review  
)ENGINE = InnoDB;
```

## 2.9.Employee Review

**Relational Model:**

EmployeeReview(ReviewID, CompanyName, Pros, Cons, Position, SalaryInfo, workExp)

**Functional Dependencies:**

ReviewID -> CompanyName, Pros, Cons, Position, SalaryInfo, workExp

**Candidate Key:**

{(ReviewID)}

**Normal Form:** BCNF

**Table Definition:**

```
Create Table EmployeeReview(
```

```
ReviewID int,  
CompanyName varchar(50),  
Pros varchar(255),  
Cons varchar(255),  
Position varchar(32),  
SalaryInfo int,  
WorkExp varchar(255),  
Foreign Key (ReviewID) References Review(ReviewID)  
)ENGINE = InnoDB;
```

## 2.10.Company

### Relational Model:

Company(CompanyID, CompanyName, CompanyAddress, CompanyEmail)

### Functional Dependencies:

CompanyID -> CompanyName, CompanyAddress, CompanyEmail

CompanyName -> CompanyID, CompanyAddress, CompanyEmail

### Candidate Key:

{{CompanyID}, {CompanyName}}

### Normal Form: BCNF

### Table Definition:

```
Create Table Company(  
    CompanyID int Primary Key Auto Increment Not NULL,  
    CompanyName varchar(70) Not NULL Unique,  
    CompanyAddress varchar(255) Not NULL,
```

CompanyEmail varchar(70) Not NULL  
);

## 2.11.Follows

### Relational Model:

Follows(CompanyID, UserID)

### Functional Dependencies:

None

### Candidate Key:

{{CompanyID}, UserID}}

**Normal Form:** BCNF

### Table Definition:

```
Create Table Follows(  
    CompanyID int,  
    UserID int,  
    Foreign Key (CompanyID) References Company(CompanyID),  
    Foreign Key (UserID) References User(UserID),  
    Primary Key (CompanyID, UserID)  
)ENGINE = InnoDB;
```

## 2.12.Offers

### Relational Model:

Offers(JobID, CompanyID)

**Functional Dependencies:**

None

**Candidate Key:**

{(JobID, CompanyID)}

**Normal Form:** BCNF

**Table Definition:**

```
Create Table Offers(  
    JobID int,  
    CompanyID int,  
    Foreign Key (JobID) References Job(JobID),  
    Foreign Key (CompanyID) References Company(CompanyID),  
    Primary Key (JobID, CompanyID)  
)ENGINE = InnoDB;
```

## 2.13.Works

**Relational Model:**

Works(UserID, CompanyID, Position, Salary, WorkSince)

**Functional Dependencies:**

UserID, CompanyID -> Position, Salary, WorkSince

**Candidate Key:**

{{UserID, CompanyID}}

**Normal Form:** BCNF

**Table Definition:**

```
Create Table Works(  
    UserID int,  
    Position varchar(50),  
    Salary int,  
    WorkSince Date,  
    CompanyID int,  
    Foreign Key (CompanyID) References Company(CompanyID),  
    Foreign Key (UserID) References User(UserID),  
    Primary Key (UserID,CompanyID)  
)ENGINE = InnoDB;
```

## 2.14.Request

**Relational Model:**

Request(CompanyID, ReviewID)

**Functional Dependencies:**

None

**Candidate Key:**

{{UserID, CompanyID}}

**Normal Form:** BCNF



**Table Definition:**

```
Create Table Request(  
    CompanyID int,  
    ReviewID int,  
    Foreign Key (CompanyID) References Company(CompanyID),  
    Foreign Key (ReviewID) References Review(ReviewID),  
    Primary Key (CompanyID,ReviewID)  
)ENGINE = InnoDB;
```

**2.15.Chat****Relational Model:**

Chat(ReceiverID, SenderID)

**Functional Dependencies:**

None

**Candidate Key:**

{(ReceiverID, SenderID)}

**Normal Form:** BCNF

**Table Definition:**

```
Create Table Chat(  
    ReceiverID int Unique,  
    SenderID int Unique,  
    Foreign Key (ReceiverID) References User(UserID),
```

Foreign Key (SenderID) References User(UserID)  
)ENGINE = InnoDB;

## 2.16.Ban

### Relational Model:

Ban(ReceiverID, SenderID)

### Functional Dependencies:

None

### Candidate Key:

{(ReceiverID, SenderID)}

### Normal Form: BCNF

### Table Definition:

Create Table Ban(  
    ReceiverID int Unique,  
    SenderID int Unique,  
    Foreign Key (ReceiverID) References User(UserID),  
    Foreign Key (SenderID) References User(UserID)  
)ENGINE = InnoDB;

## 2.17.Hires

### Relational Model:

Hires(CompanyID, userID)

### Functional Dependencies:

None

**Candidate Key:**

{(CompanyID, userID)}

**Normal Form: BCNF****Table Definition:**

```
Create Table Hires(  
    CompanyID int,  
    UserID int,  
    Foreign Key (CompanyID) References Company(CompanyID),  
    Foreign Key (UserID) References User(UserID),  
    Primary Key (CompanyID, UserID)  
)ENGINE = InnoDB;
```

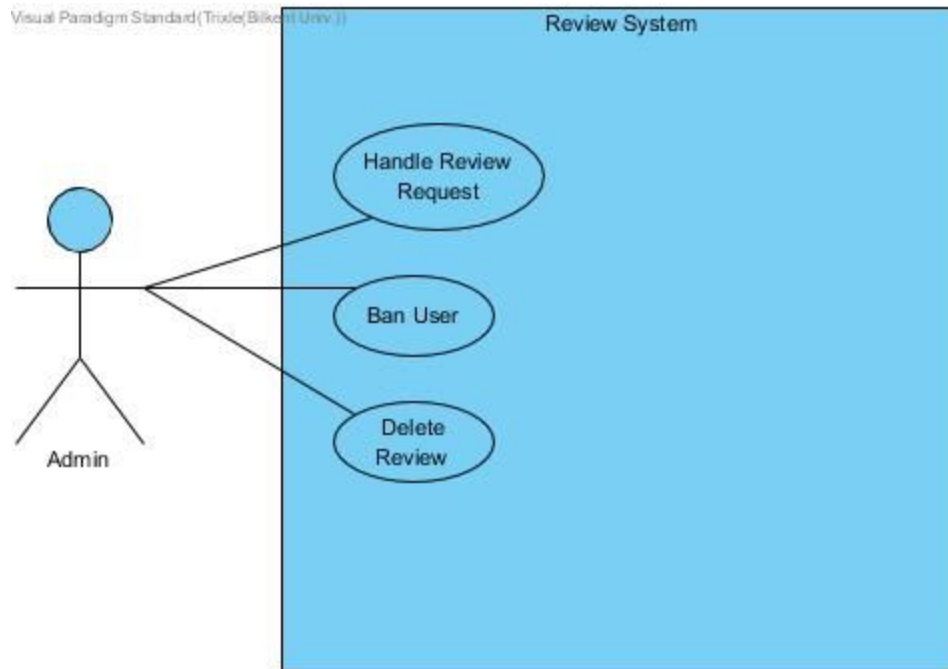
### **3.FUNCTIONAL DEPENDENCIES AND NORMALIZATION OF TABLES**

The relational schemas part has the functional dependencies and normalization is not necessary since BCNF is used.

### **4.FUNCTIONAL COMPONENTS**

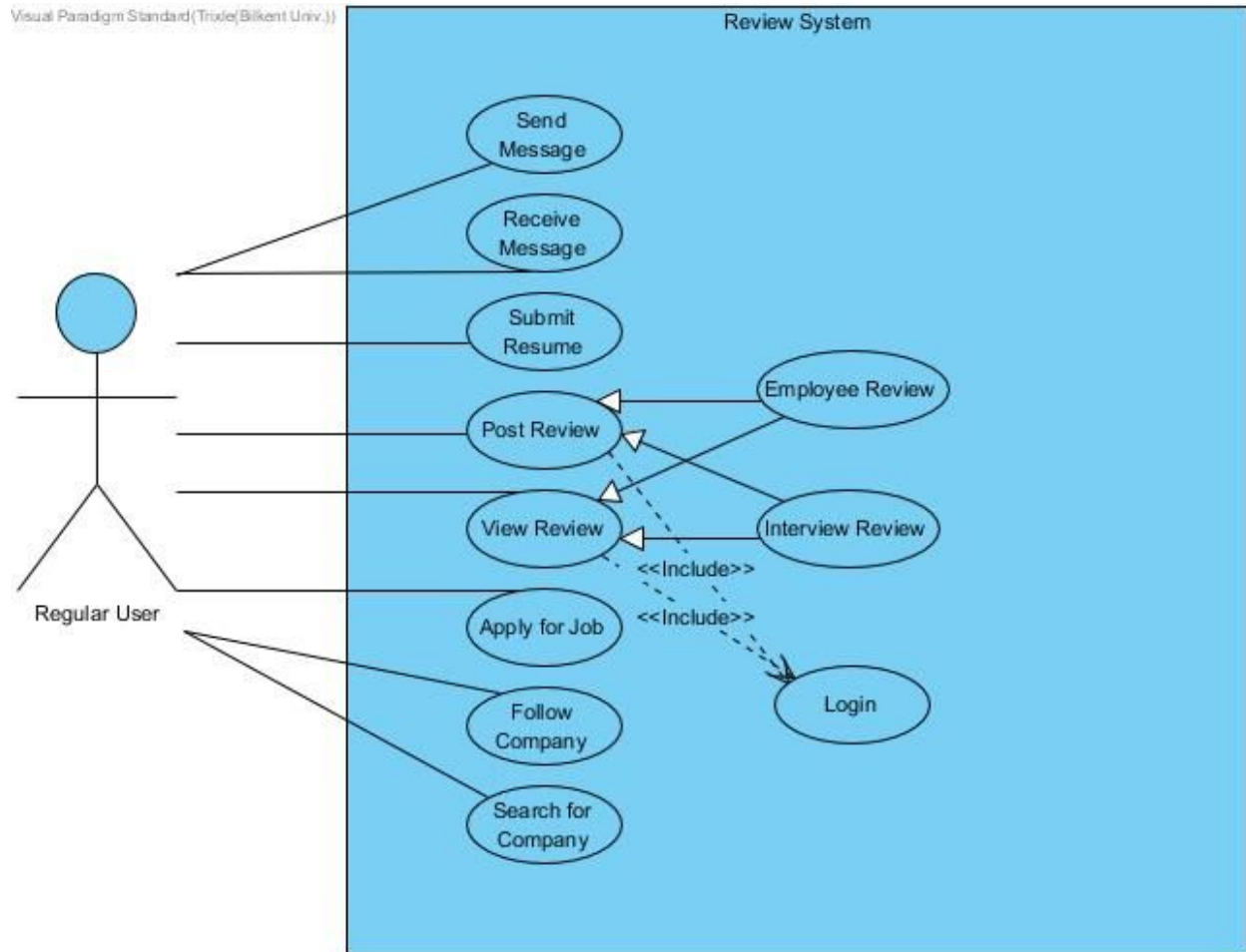
#### **4.1.Use Cases and Scenarios**

#### 4.1.1.Admin:



- **Handle Review Request:** Admin can delete the reviews that are required from the companies.
- **Delete Review:** When admin accepts the user's deletion request or when the admin declines the review to be posted, he/she can delete the review.
- **Ban User:** When the users write inappropriate reviews or give false information about the company and its interview processes, admin will ban those users.

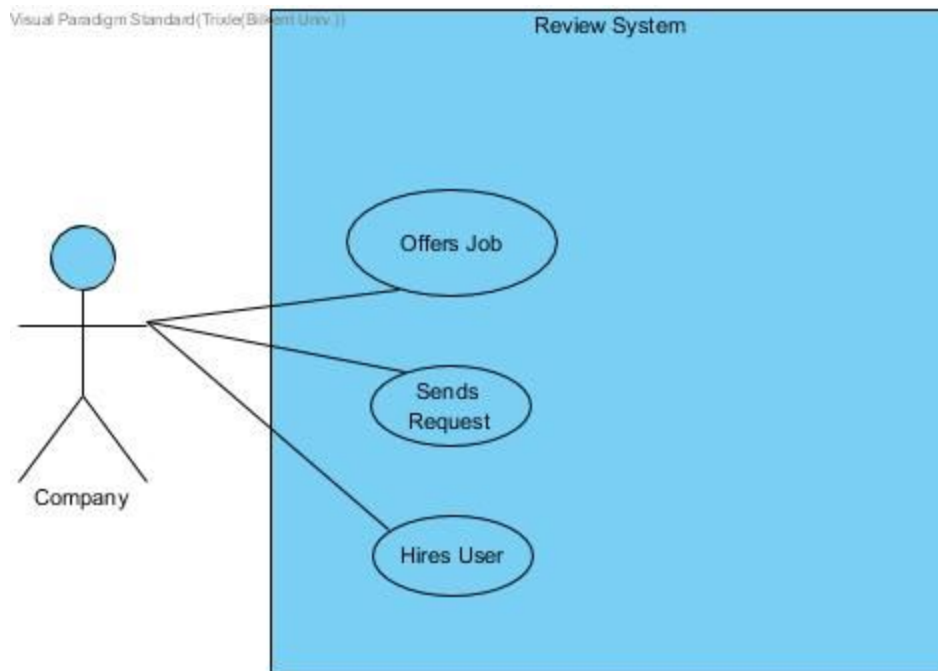
#### 4.1.2.Regular User:



- **Send Message:** User can send message to the other users.
- **Receive Message:** User can receive message from the other users.
- **Submit Resume:** Users can upload their resume in their profile from their “edit profile” page.
- **Apply for the Job:** User can apply to a job which is posted by a particular company.
- **Follow Company:** Users can follow companies to get notified,

- **Search for the Company:** Users can search for the companies through a search box to learn the reviews of the company and its overall score.
- **Post Review:** Users can post two types of review on company's review page; employee review and interview review. To post employee review, user must have a work experience with that company.
- **View Review:** User can see the reviews from a company's review page.

#### 4.1.3.Company:



- **Offers job:** Companies can create an application form for candidate users to apply.
- **Sends Request:** Companies can send requests to admins for deletion inappropriate reviews.
- **Hires User:** Companies can accept an application of a user to a job offer that they create.

## **4.2.Algorithms**

### **4.2.1.Deletion of a Review Algorithm**

When a company wants to send a deletion request for a review which has an inappropriate content, it does that by submitting a request for deletion for a particular review. These requests are sent to a page that can be seen for only by admins. If an admin also finds the review inappropriate or irrelevant, review will be deleted by the admin.

### **4.2.2.Job Application Algorithm**

A job application can be done in two ways. In the first way, the user searches for a company in search box. After clicking the search button, the user is directed to the page that contains reviews of that company and the offered jobs of that company. The application can be done by the buttons next to the offered jobs text. In the second way, the user can reach that page from the profile page. In the profile page, the button that directs the user to the followed companies has the links for reviews of that companies. This page is the same page that was mentioned in the first way.

### **4.2.3.Ranking Algorithm**

The reviews have the attributes named “LikeCount” and “ViewCount” are used for that algorithm. “LikeCount” is incremented for a review when other users click the like button. “ViewCount” is incremented when a user search for a company’s reviews and read it. Rank is obtained by dividing “ViewCount” to “LikeCount”. It basically means that if a review will be more popular when it was seen and liked at the same time.

### **4.2.4 Ban System**

Bans will be based on the requests. If a request contains inappropriate language, admin will be able to ban that user based on the “ReviewID”.

### 4.3.Data Structures

Numeric, String and Date types are used for schemas.

Numeric Types are used for id's of entities, numerical values and some counting attributes.

String Types are used for name attributes of the entities.

Date Type is only used for bantimes of the users.

## 5.User Interface Design and Corresponding SQL Statements

### 5.1.Sign up Page:

Google Chrome

http://www.onixia.com/signup

WELCOME TO ONIXIA

**Individual Account**

Username\*

Password\*

Email\*

Address

Age

Phone Number

Create Account

**Company Account**

Company name\*

Password\*

Email\*

Address

Create Account

**Inputs:** @userID, @password, @email, @address, @age, @phone\_number

**Process:** The sign up pop-up page link is located on top right of the log-in or home screen. This pop-up link will only be visible when the user have not logged in yet. User



must enter a unique username, password and email. Address, age and phone number information will be optional for user to fill in. There are also word count limitations for username(32), password(32), email(75) and address(100).

### SQL Statements:

INSERT INTO user

VALUES(@userID, @password, @email, @address, @age, @phone\_number)

## 5.2.Login Page:

Google Chrome

http://www.onixia.com/login

WELCOME TO ONIXIA

Individual Account

Username

Password

Submit

Company Account

Company name

Password

Submit

[Forgot your password?](#)

**Inputs:** @userID, @password

**Process:** This pop-up page can be reached by clicking a link which is located in the top right side of the page near the sign up pop-up link. User should enter his/her username

and password correctly in order to log-in or sign-in to the website. If the user forgot its password, they can click “forgot your password” link to open the reset password page.

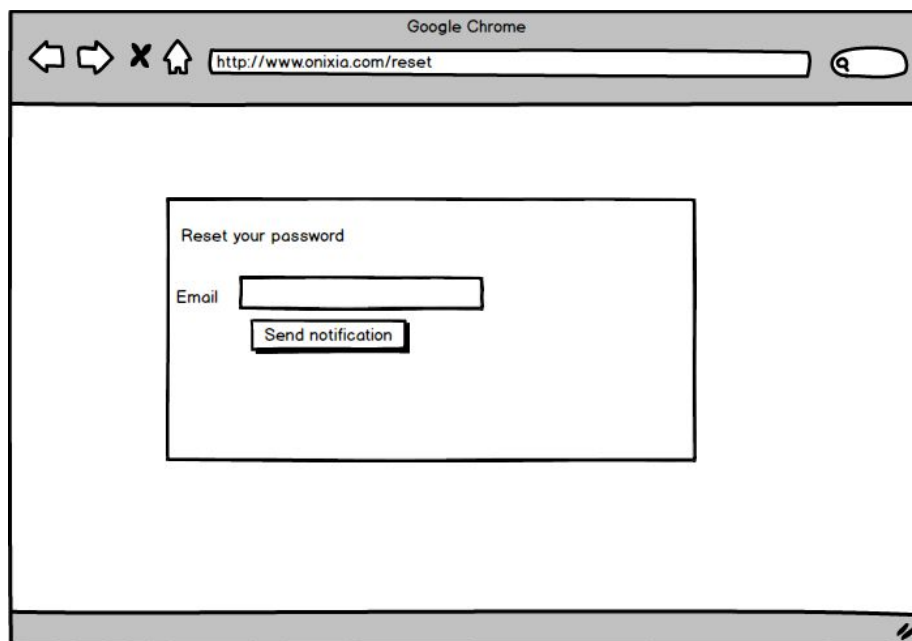
#### SQL Statements:

```
SELECT *
```

```
FROM user
```

```
WHERE userID = @userID AND password = @password
```

### 5.3.Reset Page:



The image shows a web browser window titled "Google Chrome". The address bar contains the URL "http://www.onixia.com/reset". The main content area displays a form titled "Reset your password". Inside the form, there is a label "Email" followed by a text input field. Below the input field is a button labeled "Send notification".

**Inputs:** @email

**Process:** When the user forgot his/her password during sign in session, they can open this window by clicking “forgot your password” from sign in page. User enters their

Email and click send notification. Few minutes later, new password will assigned to their account and also new password will be sent to the user via email by us.

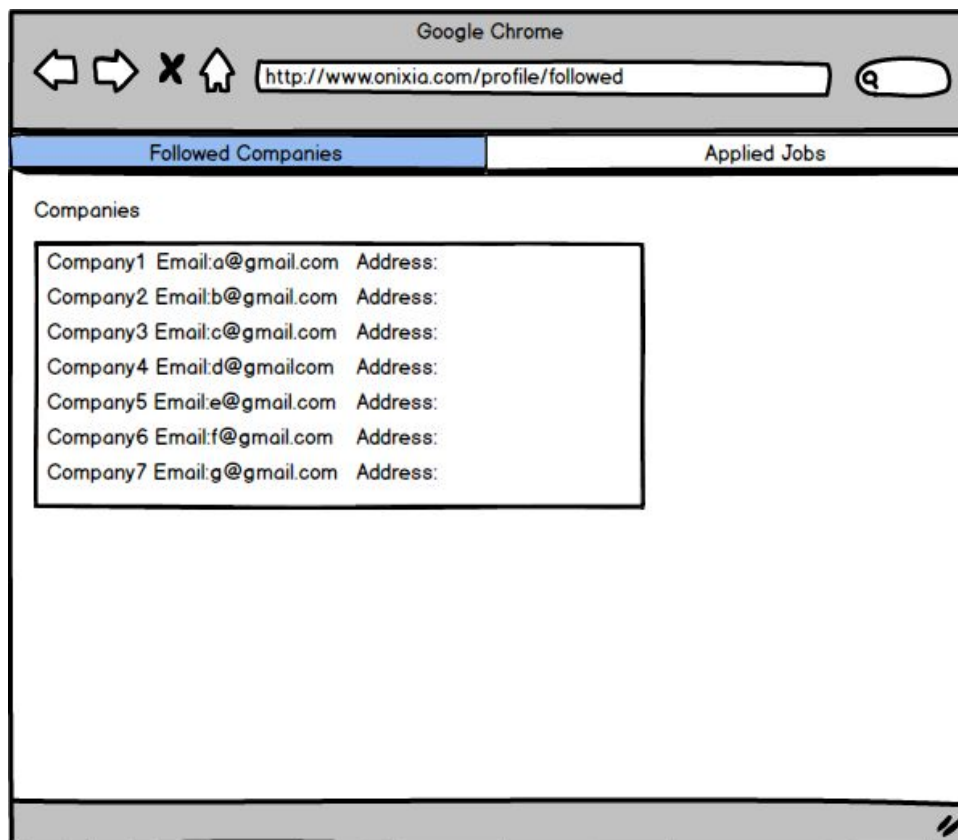
### SQL Statements:

```
SELECT *
```

```
FROM user
```

```
WHERE email = @email
```

## 5.4.Followed Companies Page

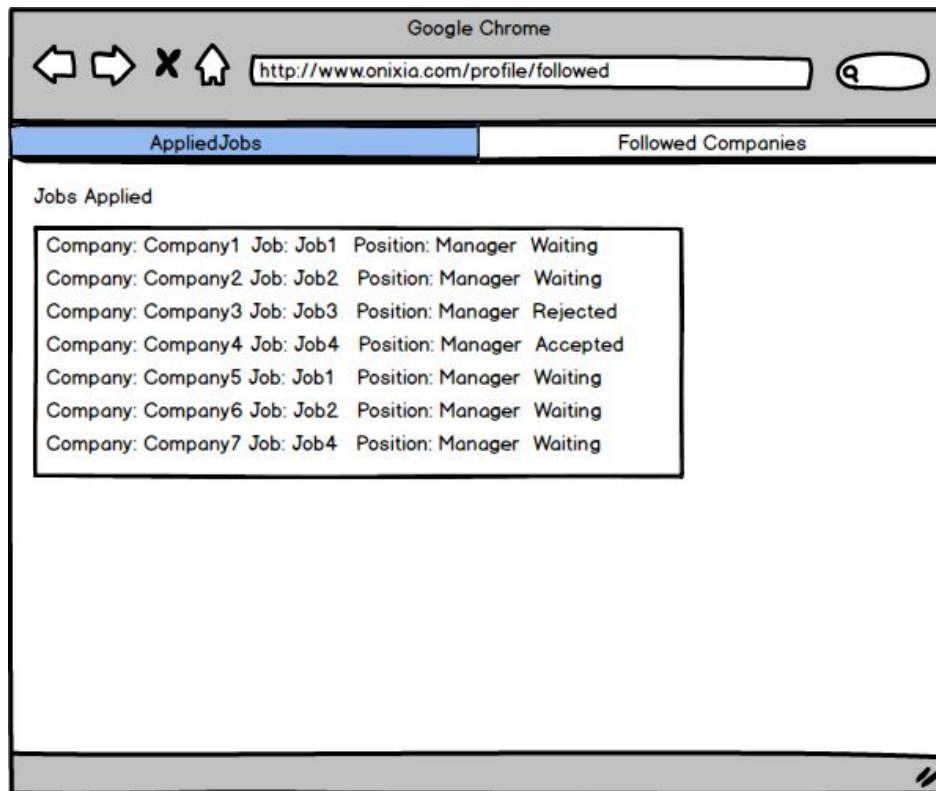


**Process:** When the user wants to list his followed companies, s/he can press the “Followed Companies” button and the companies will be listed with their information.

**SQL Statement:**

```
SELECT CompanyName, CompanyEmail, CompanyAddress  
FROM Company Natural Join Follows Natural Join RegularUser  
WHERE userID = @userID
```

## 5.5.Applied Jobs Page

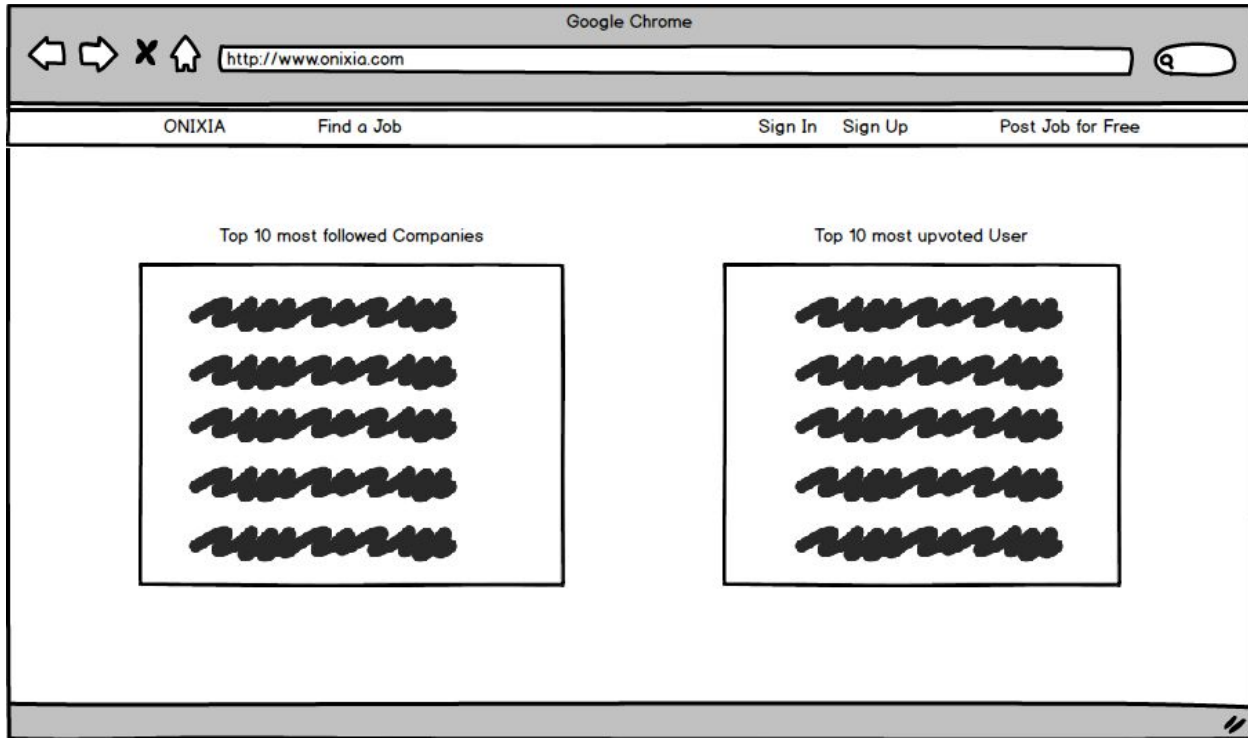


**Process:** When the user wants to list his applied job list, s/he can press the “Followed Companies” button and the applied job will be listed with their information.

**SQL Statement:**

```
SELECT CompanyName, JobName, Position, Status  
FROM Company NATURAL JOIN Job NATURAL JOIN Applies  
WHERE userID = @userID
```

## 5.6.Home Screen Page



**Process:** Non-logged in users can see a menu bar and two tables in the home screen of the Onixia. Menu bar includes find a job, post job for free page and also sign in and sign up page. Furthermore, it includes two tables that show top 10 mos followed companies and top 10 most upvoted user.

### SQL Statement:

#### Top10 upvoted users

```
CREATE VIEW users_top10(name,rank)
SELECT username, MAX(likeCount/viewCount) AS m
FROM Post NATURAL JOIN Review NATURAL JOIN User
GROUP BY username
ORDER BY m DESC
LIMIT 10
```

**Top 10 followed companies.**

```
CREATE VIEW (CompanyName, follower)
SELECT CompanyName, count(CompanyID) AS c
FROM Company NATURAL JOIN Follows
GROUP BY CompanyName
ORDER BY c DESC
LIMIT 10
```

## 5.7. User's Profile Setting Page

Google Chrome

http://www.onixia.com/settings

Search For Job Search Company Submit Review Following Applied Job List Settings Logout

### Koray's Profile

Write Your Resume:

<b>Reset Password</b>	<b>Reset E-mail</b>
Current Password: <input type="password"/>	Current E-mail: <input type="text"/>
New Password: <input type="password"/>	New E-mail: <input type="text"/>
Confirm New Password: <input type="password"/>	Confirm New E-mail: <input type="text"/>
Address: <input type="text"/>	Workplace: <input type="text"/>
Age: <input type="text"/>	Salary: <input type="text"/>
Phone Number: <input type="text"/>	Position: <input type="text"/>

**Inputs:** @resume, @password, @newpassword, @address, @age, @phonenumber, @email, @newemail, @workplace, @salary, @position

**Process:** Users can change their passwords and email along with their personal information through this page.

### SQL STATEMENTS:

#### Change Password

UPDATE user

```
SET password = @newpassword  
WHERE userID = @userID
```

### **Change Email**

```
UPDATE User  
SET email = @email  
WHERE userID = @userID
```

### **Add Resume**

```
INSERT INTO user.resume  
VALUES(@resume)  
WHERE userID = @userID
```

### **Add Address**

```
INSERT INTO user.address  
VALUES(@address)  
WHERE userID = @userID
```

### **Add Age**

```
INSERT INTO user.age  
VALUES(@age)  
WHERE userID = @userID
```

### **Add Phone Number**

```
INSERT INTO user.phonenumber  
VALUES(@phonenumber)  
WHERE userID = @userID
```



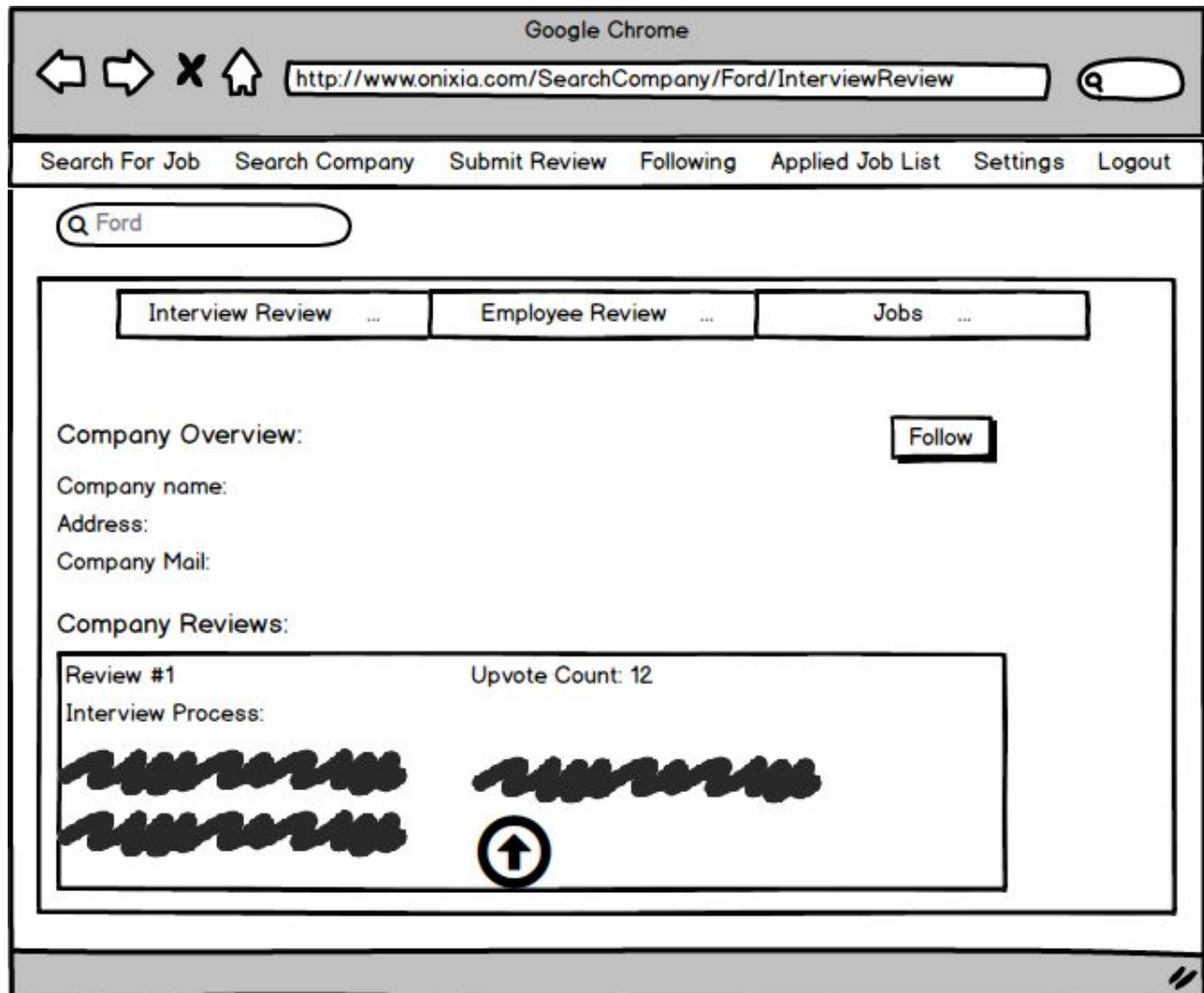
### **Add Phone Number**

```
INSERT INTO job,workplace, job.salary, job.position  
VALUES(@workplace, @salary, @position)  
WHERE userID = @userID
```

### **View Profile Details**

```
SELECT userID, email, address, age, phonenumber, workplace, salary, position, resume  
FROM user, job  
Where userID = @userID
```

## 5.8.View Company's Interview Review Page



**Inputs:** @searchquery

**Process:** User can search the company that he/she desires by typing the name of the company to the search box. In this page, user can choose between interview and employee reviews and jobs. User can also upvote another user's post by clicking upvote button.

## **SQL STATEMENT:**

### **Company Search**

```
SELECT CompanyName
      CASE WHEN @searchquery = CompanyName
      END
FROM Company
```

### **Sorting reviews for a particular company**

```
CREATE VIEW users_top10c(name,rank)
SELECT username, MAX(likeCount/viewCount) AS m
FROM Post NATURAL JOIN Review NATURAL JOIN User
WHERE CompanyName = @CompanyName
GROUP BY username
ORDER BY m DESC
```

## 5.9.Create a Job Page

Google Chrome

http://www.onixia.com/CreateaJob

Create a Job Check Candidates Search Company Settings Logout

Job Description

Job Name

Position

Salary

**Inputs:** @JobName, @position, @salary, @jobDescription

**Process:** Companies can create a job through this page by filling following attributes; salary, position, job name and job description. Then they have to click the Create a Job button in order to create a job.

**SQL STATEMENTS:****Add Job Name**

```
INSERT INTO job.JobName  
VALUES(@JobName)
```

**Add position**

```
INSERT INTO job.position  
VALUES(@position)
```

**Add salary**

```
INSERT INTO job.salary  
VALUES(@salary)
```

**Add Job Description**

```
INSERT INTO job.jobDescription  
VALUES(@jobDescription)
```

**Check Candidates**

**Process:** Companies can check the candidates for their created jobs. Company can accept or decline the candidates for the particular job through this menu.

**SQL STATEMENTS:**

There is no particular SQL statement for this page.

## 5.10.Chat

Google Chrome

http://www.onixia.com/chat

username:

Message Box:

Received Messages

**Input:** @userID, @message

**Process:** Users can chat through this menu.

### SQL STATEMENTS:

There is no particular SQL statement for this page.

### Submit Review

**Input:** @pros, @cons, @position, @salary, @workexp, @interviewProcess

**Process:** User selects the type of the review and after fill the relevant chapters, user will upload the post to the particular company.

## SQL STATEMENTS:

### If employee review

```
INSERT INTO EmployeeReview.pros, EmployeeReview.cons,  
EmployeeReview.position, EmployeeReview.salary, EmployeeReview.workexp  
VALUES(@pros, @cons, @posiiton, @salary, @workexp)
```

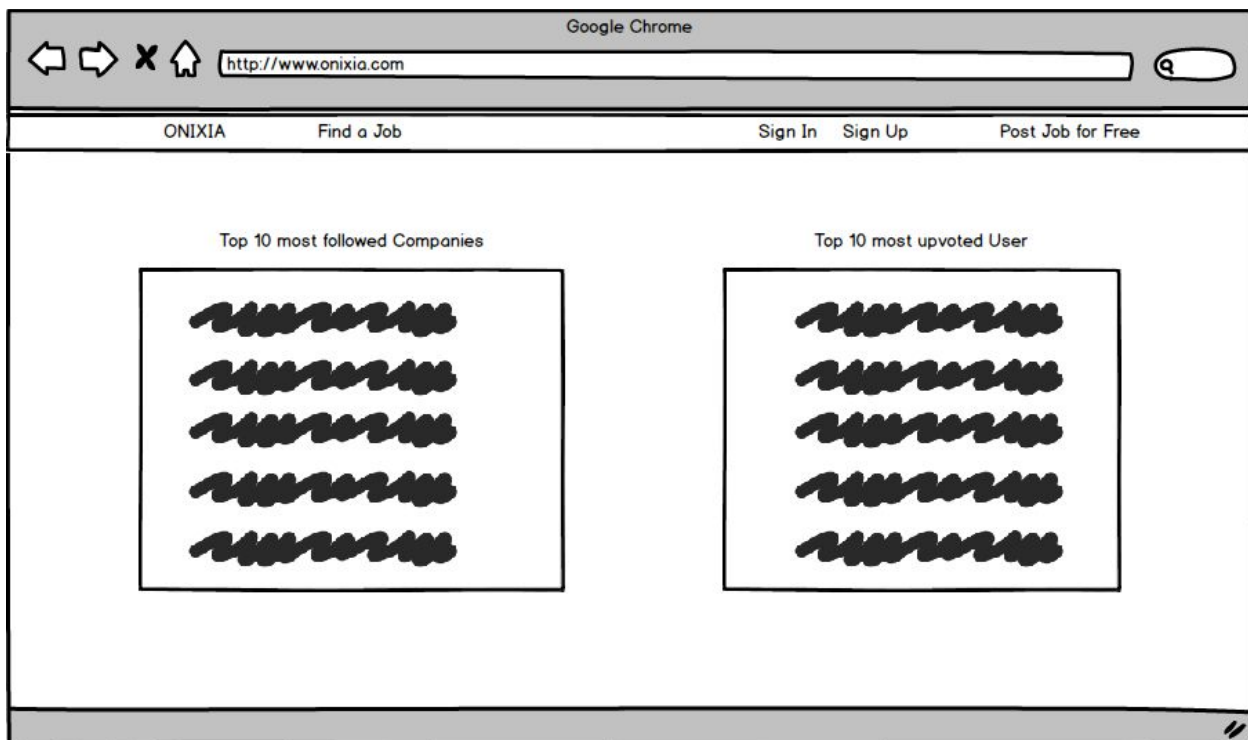
### If interview review

```
INSERT INTO InterviewReview.interviewProcess  
VALUES(@interviewProcess)
```

## 5.11 Shared UI Designs

This UI Designs shared for all types of users.

### 5.11.1 Home Screen



### 5.11.2 Sign Up Page

Google Chrome

http://www.onixia.com/signup

WELCOME TO ONIXIA

**Individual Account**

Username\*

Password\*

Email\*

Address

Age

Phone Number

Create Account

**Company Account**

Company name\*

Password\*

Email\*

Address

Create Account



### 5.11.3 Login Page

The screenshot shows a web browser window titled "Google Chrome" with the address bar displaying "http://www.onixia.com/login". The page content is titled "WELCOME TO ONIXIA" and features two login forms side-by-side. The left form is for an "Individual Account" and includes fields for "Username" and "Password", followed by a "Submit" button. The right form is for a "Company Account" and includes fields for "Company name" and "Password", followed by a "Submit" button. Below the forms is a link that says "Forgot your password?".

Google Chrome

http://www.onixia.com/login

WELCOME TO ONIXIA

Individual Account

Username

Password

Company Account

Company name

Password

[Forgot your password?](#)

### 5.11.4 Reset Password Page

The screenshot shows a web browser window titled "Google Chrome" with the address bar displaying "http://www.onixia.com/reset". The page content is titled "Reset your password" and includes a single form with an "Email" field and a "Send notification" button.

Google Chrome

http://www.onixia.com/reset

Reset your password

Email

## 5.12 UI Design for Users

This UI design is explicit for regular users.

### 5.12.1 Edit Profile(Setting)

Google Chrome

http://www.onixia.com/settings

Search For Job Search Company Submit Review Following Applied Job List Settings Logout

## Koray's Profile

Write Your Resume:

<b>Reset Password</b>	<b>Reset E-mail</b>
Current Password: <input type="password"/>	Current E-mail: <input type="text"/>
New Password: <input type="password"/>	New E-mail: <input type="text"/>
Confirm New Password: <input type="password"/>	Confirm New E-mail: <input type="text"/>
Address: <input type="text"/>	Workplace: <input type="text"/>
Age: <input type="text"/>	Salary: <input type="text"/>
Phone Number: <input type="text"/>	Position: <input type="text"/>

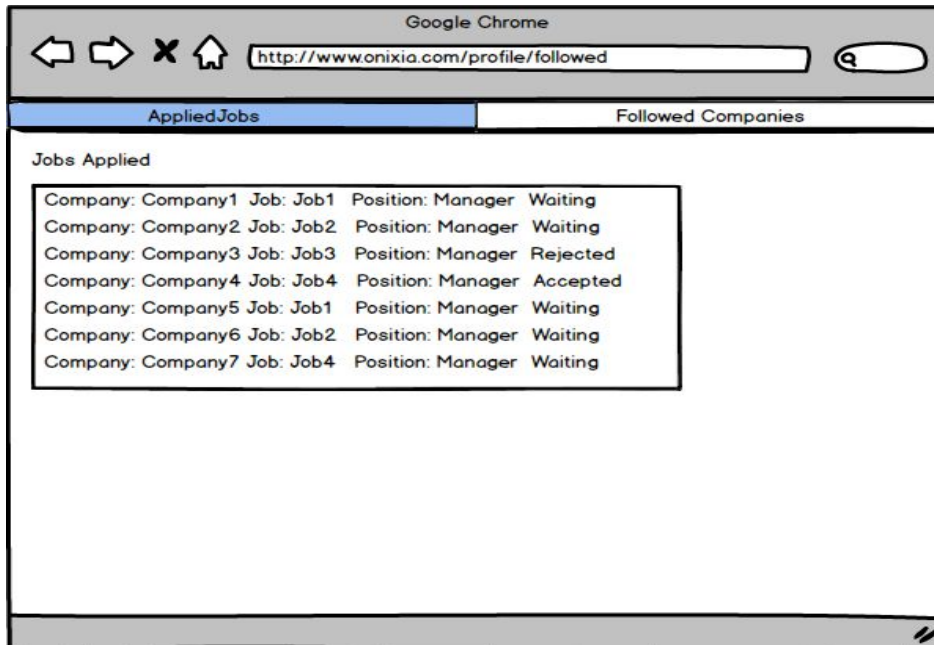
## 5.12.2 Post Review

A screenshot of a web browser window titled "Google Chrome" showing the URL "http://www.onixia.com/SubmitReview". The browser's address bar and navigation icons are visible. Below the address bar is a navigation menu with links: "Search For Job", "Search Company", "Submit Review", "Following", "Applied Job List", "Settings", and "Logout". The main content area of the page contains two radio buttons for selection: "Interview Review" and "Employee Review". Below these is a label "Review" followed by a large, empty rectangular box for text input.

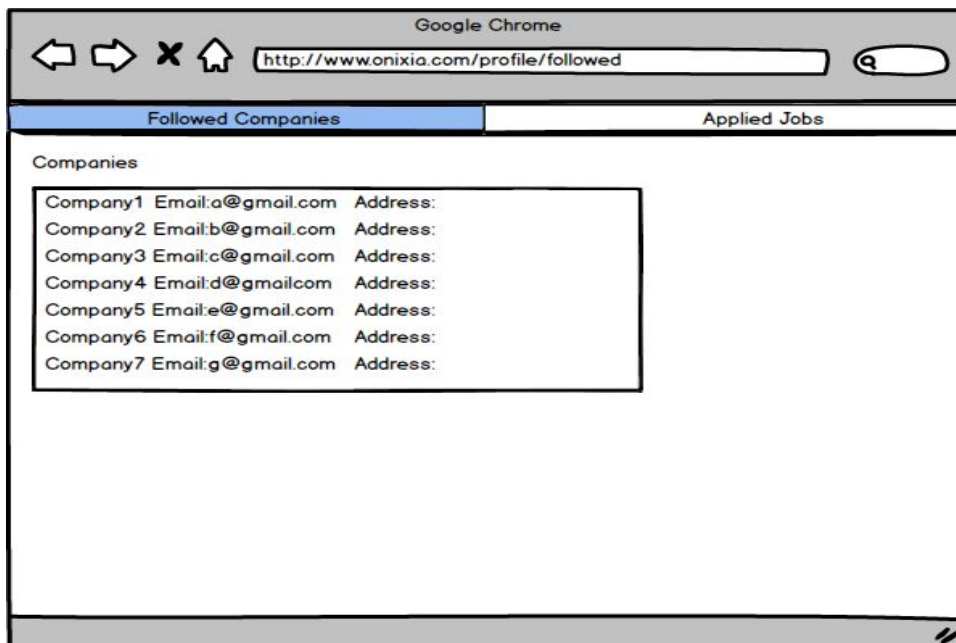
## 5.12.3 Chat

A screenshot of a web browser window titled "Google Chrome" showing the URL "http://www.onixia.com/chat". The browser's address bar and navigation icons are visible. The main content area is divided into two sections. On the left, there is a "username:" label next to a text input field, and below it, a "Message Box:" label next to a larger text input area. On the right, there is a label "Received Messages" above a large, empty rectangular box for displaying chat messages.

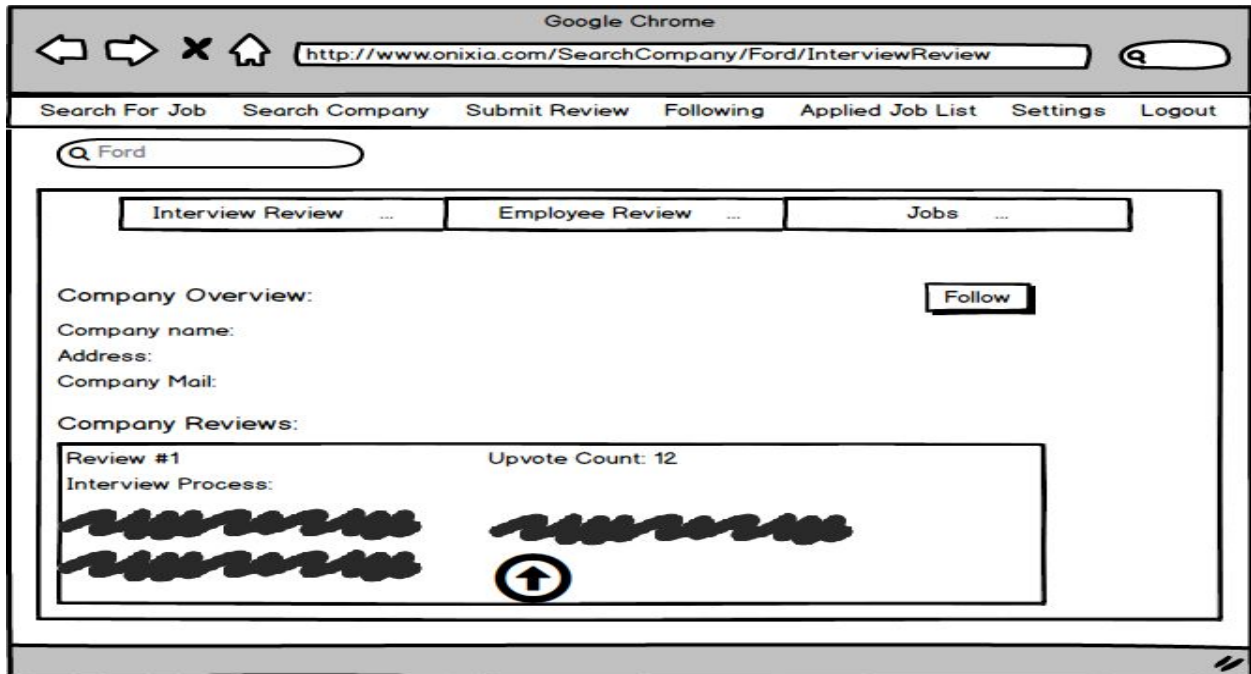
## 5.12.4 Applied Jobs



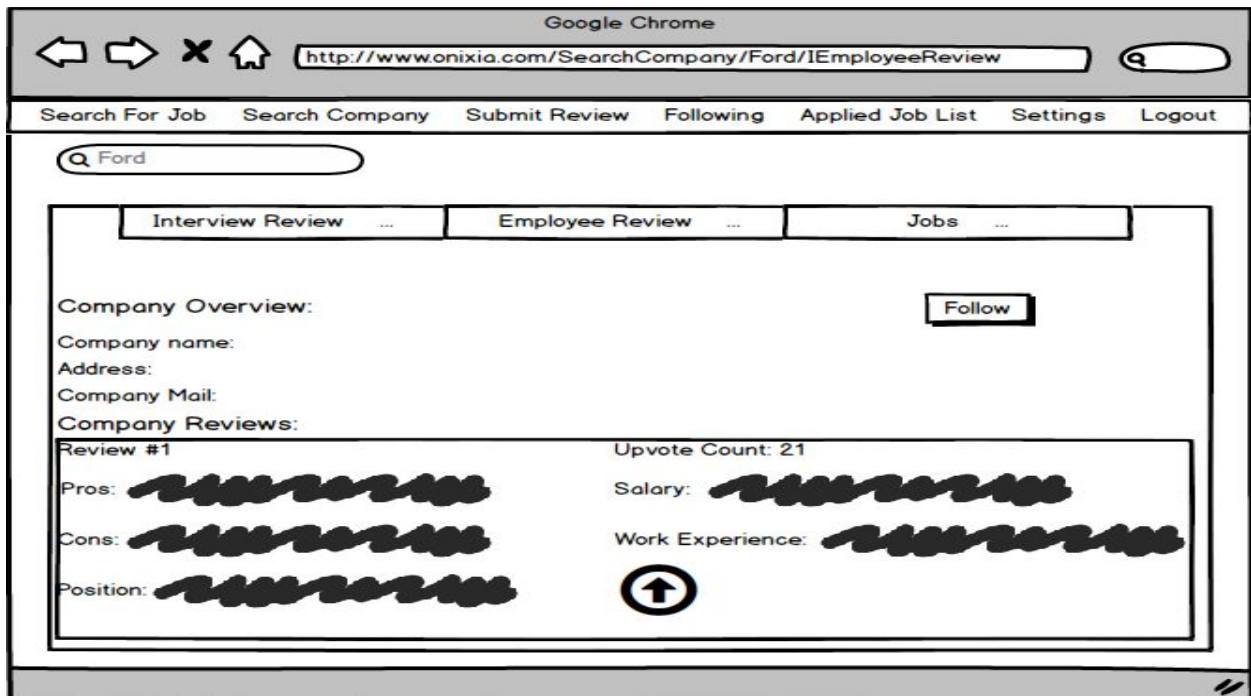
## 5.12.5 Followed Companies



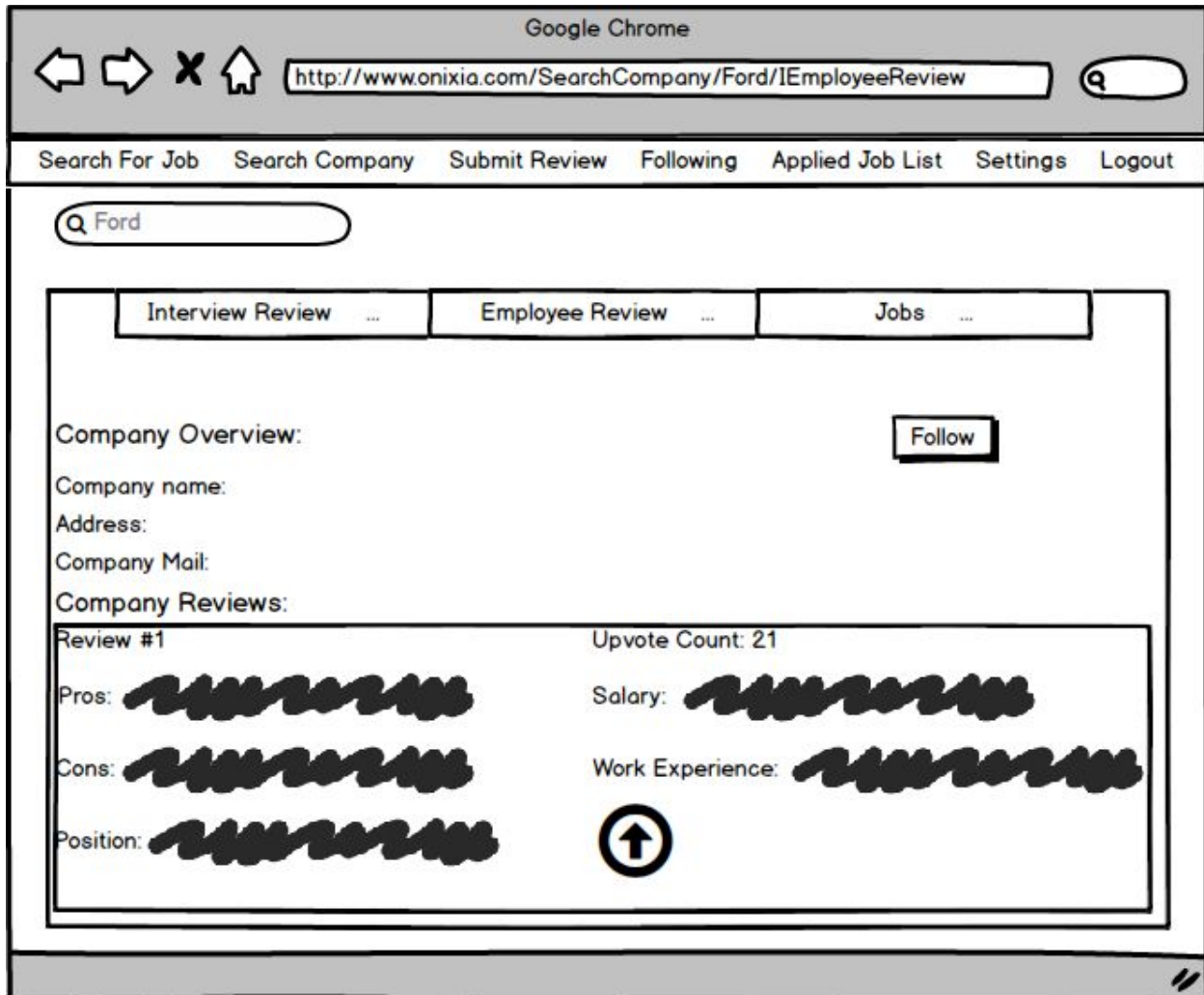
### 5.12.6 View Employee Review For a Company



### 5.12.7 View Interview Review For a Company



## 5.12.8 View Jobs For a Company



## 5.13 UI Design for Company

This UI design is for the company.

### 5.13.1 Company's Edit Profile Page(Settings)

The screenshot shows a web browser window titled "Google Chrome" with the address bar displaying "http://www.onixia.com/settings". The page has a navigation bar with links: "Create a Job", "Search Company", "Settings" (active), and "Logout". The main heading is "Company Name". Below this, there are two columns of form fields. The left column is titled "Reset Password" and contains fields for "Current Password:", "New Password:", "Confirm New Password:", and "Address:". The right column is titled "Reset E-mail" and contains fields for "Current E-mail:", "New E-mail:", "Confirm New E-mail:", and "Change Company Name:". All password and email fields are masked with asterisks or empty text boxes. The browser window has a standard toolbar with back, forward, and search icons.

### 5.13.2 Check Candidate Page

The screenshot shows a web browser window titled "Google Chrome" with the address bar displaying "http://www.onixia.com/CheckCandidates". The page has a navigation bar with links: "Create a Job", "Check Candidates" (active), "Search Company", "Settings", and "Logout". The main content area is a table with three columns: "Job Name", "Candidate Name", and "Status". There are three rows of data. Each row shows a job name, a candidate name, and two buttons: "Accept" and "Decline". The job names and candidate names are represented by wavy lines, indicating they are placeholders or redacted. The browser window has a standard toolbar with back, forward, and search icons.



### 5.13.3 Create Job Page

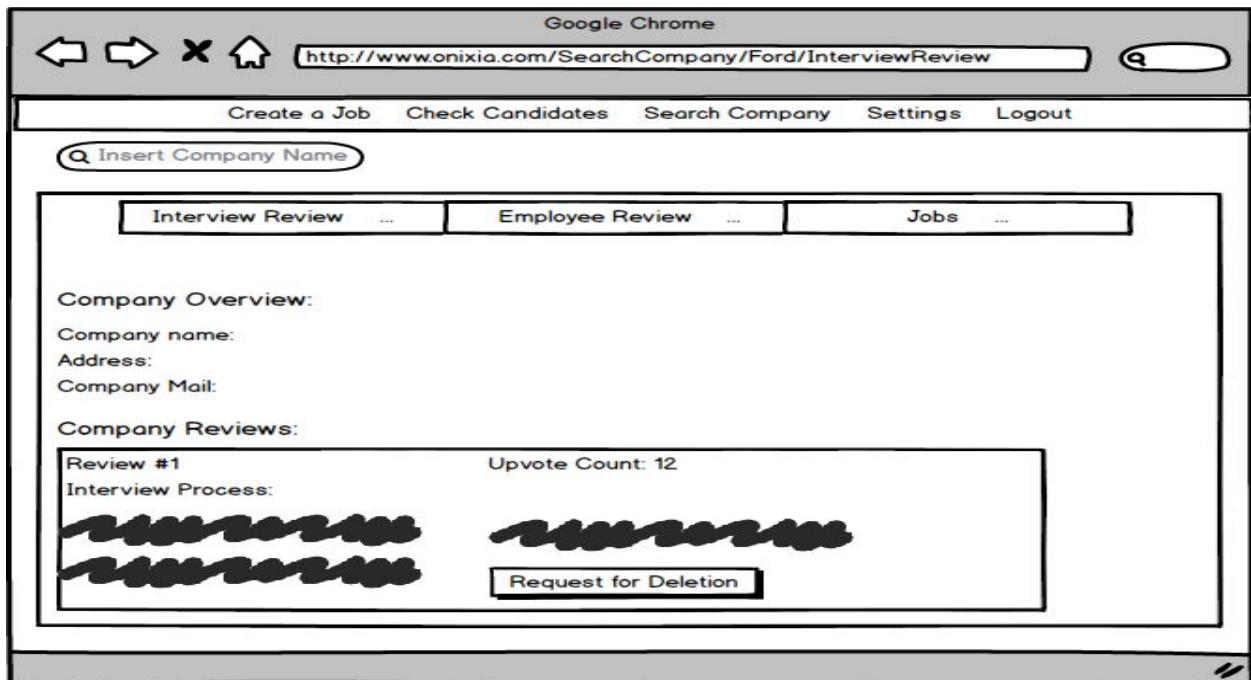
The screenshot shows a web browser window titled "Google Chrome" with the address bar displaying "http://www.onixia.com/CreateaJob". The browser's navigation bar includes back, forward, and home buttons. The website's navigation menu contains "Create a Job", "Check Candidates", "Search Company", "Settings", and "Logout". The main content area is titled "Job Description" and features three input fields on the left: "Job Name", "Position", and "Salary". To the right of these fields is a large, empty rectangular box for the job description. At the bottom center of the form is a "Create a Job" button.

### 5.13.4 View Employee Review For a Company

The screenshot shows a web browser window titled "Google Chrome" with the address bar displaying "http://www.onixia.com/SearchCompany/Ford/EmployeeReview". The navigation menu is the same as in the previous page. Below the menu is a search bar with the placeholder text "Insert Company Name". A horizontal tab bar contains three tabs: "Interview Review ...", "Employee Review ...", and "Jobs ...". The "Employee Review" tab is currently selected. The main content area is titled "Company Overview:" and lists "Company name:", "Address:", and "Company Mail:". Below this is a section titled "Company Reviews:" which displays three rows of data: "Pros:", "Cons:", and "Position:". Each row is followed by a series of black, scribbled-out text. To the right of this section, there are two more rows: "Salary:" and "Work Experience:", also followed by blacked-out text. At the bottom right of the content area is a "Request for Deletion" button.



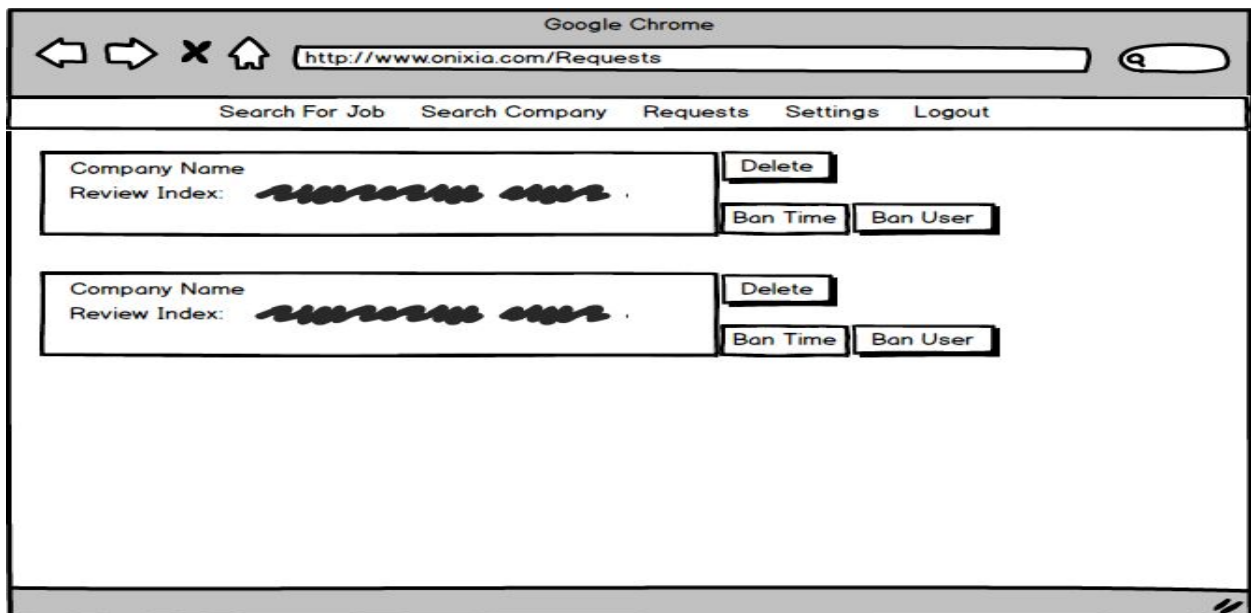
### 5.13.5 View Interview Review For a Company



## 5.14 UI Design for Admins

This UI design is for Admins only.

### 5.14.1 Check Request Page



UI designs for review pages for a particular company is same with the user's design. The only difference is instead of upvote button, it has delete review and ban the user button.

## 6.ADVANCED DATABASE COMPONENTS

### 6.1.Top10 users

A view to see top 10 users with most liked reviews is below.

```
CREATE VIEW users_top10(name,rank)
SELECT username, MAX(likeCount/viewCount) AS m
FROM Post NATURAL JOIN Review NATURAL JOIN User
GROUP BY username
ORDER BY m DESC
LIMIT 10
```

### 6.2. Sorting reviews for a particular company

```
CREATE VIEW users_top10c(name,rank)
SELECT username, MAX(likeCount/viewCount) AS m
FROM Post NATURAL JOIN Review NATURAL JOIN User
WHERE CompanyName = @CompanyName
GROUP BY username
ORDER BY m DESC
```

### 6.3. Follow Rank

A view to see top 10 followed companies.

```
CREATE VIEW (CompanyName, follower)
SELECT CompanyName, count(CompanyID) AS c
```

FROM Company NATURAL JOIN Follows  
GROUP BY CompanyName  
ORDER BY c DESC  
LIMIT 10

## 6.4 Triggers

- When a new review is added to the database, the related top10 list can change and this feature requires trigger.
- When an user is banned, s/he cannot send any reviews and this procedure starts as soon as after the ban by adding this user to the ban table.
- When a company approve an application, the user will immediately informed by his/her profile.

## 7.IMPLEMENTATION PLAN

To implement our project, we planned to use PHP, JSP, JQuery, HTML and CSS and for the data management, we will use MySQL.

## 8.WEBSITE

<https://github.com/gmuslu/CS353/tree/master/doc>