



Bilkent University

Department of Computer Engineering

# CareerBridge

*Project short-name: CB (Group Number: 11)*

## Final Report

Berk Çakar - 22003021

Atak Talay Yücel - 21901636

Kutay Tire - 22001787

Yiğit Yalın - 22002178

Instructor: Özgür Ulusoy

Teaching Assistant: Zülal Bingöl

Final Report

June 5, 2023

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Database Systems course CS353.

# Contents

<b>CareerBridge</b>	<b>1</b>
<b>1 System Description</b>	<b>4</b>
<b>2 Contributions</b>	<b>5</b>
<b>3 Final E/R Diagram</b>	<b>8</b>
<b>4 List of Tables</b>	<b>9</b>
User	9
AppUser	9
Admin	9
SystemReport	9
Application	9
ApplicationReview	9
Notification	9
Post	9
Comment	10
JobAdvertisement	10
JobAdvertisementResponse	10
Profile	10
LanguageProficiency	10
Award	10
Project	10
Certification	10
Publication	11
TestScore	11
Skill	11
SkillInJobAdvertisement	11
Experience	11
EducationalExperience	11
Degree	11
DegreeInJobAdvertisement	11
WorkExperience	12
VoluntaryExperience	12
SkillAssessor	12
SkillAssessment	12
<b>5 Implementation Details</b>	<b>12</b>
5.1 Environment, Frameworks, and Languages	12
5.2 Encountered Problems	13
<b>6 Advanced Database Components</b>	<b>13</b>
6.1 Views	13

6.1.1 User Login View	13
6.1.2 Get User Role View	13
6.2 Triggers	14
6.2.1 Skill Assessment Notification Trigger	14
6.2.2 Comment Notification Trigger	14
6.2.3 Application Notification Trigger	15
6.2.4 Add User to AppUser Trigger	15
6.3 Assertions	15
6.3.1 Check if Each User Has at Most One Master Skill	15
6.4 Procedures	16
6.4.1 Procedure for Increasing the Application Count of an Advertisement by One	16
6.4.2 Procedure for Increasing the View Count of an Advertisement by One	16
6.5 System Report Queries	16
<b>7 User Manual</b>	<b>20</b>
7.1 Login Page	20
7.2 Register Page	21
7.3 Browse Ads Page	22
7.4 Browse Ads Page with Filtering Options	23
7.5 Ad Description Page	24
7.6 Applied Ads Page	25
7.7 Published Ads Page	26
7.8 Candidates (Applicants) of an Ad Page	27
7.9 Publish New Ad Page	28
7.10 Profile Page	29
7.10.1 Information Page	29
7.10.2 Biography Page	29
7.10.3 Work Experience Page	29
7.10.4 Education Experience Page	29
7.10.5 Voluntary Experience Page	30
7.10.6 Projects Page	30
7.10.7 Certifications Page	30
7.10.8 Awards Page	30
7.10.9 Test Scores Page	30
7.10.10 Publications Page	30
7.10.11 Languages Page	30
7.10.12 Skills Page	31
7.10.13 Resume Page	31
7.11 User Feed	32
7.12 Admin Page	33

# **1 System Description**

CareerBridge is a professional social networking platform that not only connects individuals, businesses, and organizations but also hugely benefits both employees and recruiters with various options prepared for them. It enables the recruiters to create job advertisements with plenty of specifications such as the working type, location, pay range of the job, required degrees or skills, and a detailed description. Career experts or recruiters can also create informative posts which are seen and can be commented on by the users in the "Feed" page of CareerBridge. On the other hand, employees can explore job opportunities, build their professional profiles, and showcase their skills and qualifications. Most importantly, they can filter and list these job advertisements based on their likings.

With CareerBridge, making job applications is no longer a problem. Once an employee finds a job s/he wants to apply for, the necessary information is retrieved from his/her profile, and uploading a cv becomes the only thing left for the employee. After a successful application, the recruiter who gave the advertisement can go through the applications and respond to them in a positive or a negative manner.

Distinctly, CareerBridge has a third type of user which is the skill assessor. Skill assessors are people who have mastered a skill and earned the role to assess the individuals' skills in CareerBridge. Skill assessors can then rate the skills of individuals from the profile of users.

Finally, the admins also have the authority to create system reports that list useful information. These reports include the number of users from each user type, total job applications, average ratings of skills rated by the skill assessors or the average number of advertisement views for each company.

## 2 Contributions

### **Kutay Tire:**

#### Proposal Report:

- Wrote the introduction part
- Drew the E/R diagram

#### Design Report:

- Prepared the table schemas of the entities
- Wrote the functional dependencies of each relation
- Wrote the necessary SQL queries used in Login and Register pages.

#### Final Report:

- Wrote the system description of CareerBridge
- Wrote the advanced database components
- Drew the final E/R diagram
- Wrote the list of tables
- Wrote the User's Manual
- Wrote work allocation for himself

#### Implementation:

- Implemented the "Feed" page and necessary functions to create posts or comments
- Implemented "Job Ads" page and necessary functions to make a job application or evaluate one for the case of recruiters
- Implemented the "System Report" page for the admin and necessary functions to create the system report with desired information.

### **Berk Çakar:**

#### Proposal Report:

- Wrote the Project Description, Necessity of the Database, Utilization of the Database, and Requirements sections
- Drew the E/R diagram

#### Design Report:

- Drew the UI mockups
- Fixed the E/R diagram based on the given feedback
- Wrote the SQL Queries for forgot password, browse ads, browse ads with filtering, and ad description pages
- Wrote views, assertions, and procedures as advanced database components

#### Final Report:

- Drew the E/R diagram
- Wrote work allocation for himself
- Took screenshots of different views of the application

### Implementation:

- Initialized the project structure (Python FastAPI backend + Angular frontend), and determined the required dependencies (i.e., external packages) for the implementation.
- Implemented the authentication system in the backend
- Initialized the user, user profile, and skill assessment related endpoints.
- Overall refactoring and bug fixing of the backend code
- Worked on connecting frontend to backend via API calls
- Implemented the routing and access guard mechanisms in the frontend
- Implemented a “toaster” service, so that success and error messages can be delivered to the end user graphically
- Containerized the project using Docker for automating the deployment process

### **Atak Talay Yücel:**

#### Proposal Report:

- Wrote the Necessity of the Database with Yiğit
- Helped drawing E/R diagram

#### Design Report:

- Wrote SQL queries

#### Final Report:

- Wrote work allocation for himself

### Implementation:

- Implemented “Feed”, “Job Ad”, “System Report”, and “Candidates” pages on frontend
- Implemented models and services on frontend
- Worked on the connection between frontend and backend

### **Yiğit Yalın:**

#### Proposal Report:

- Wrote the Project Description, Necessity of the Database, Utilization of the Database, and Limitation sections

#### Design Report:

- Wrote SQL queries

#### Final Report:

- Wrote work allocation for himself

### Implementation:

- Implemented views & dialogs on frontend
- Implemented models and services on frontend
- Worked on the connection between frontend and backend

### 3 Final E/R Diagram

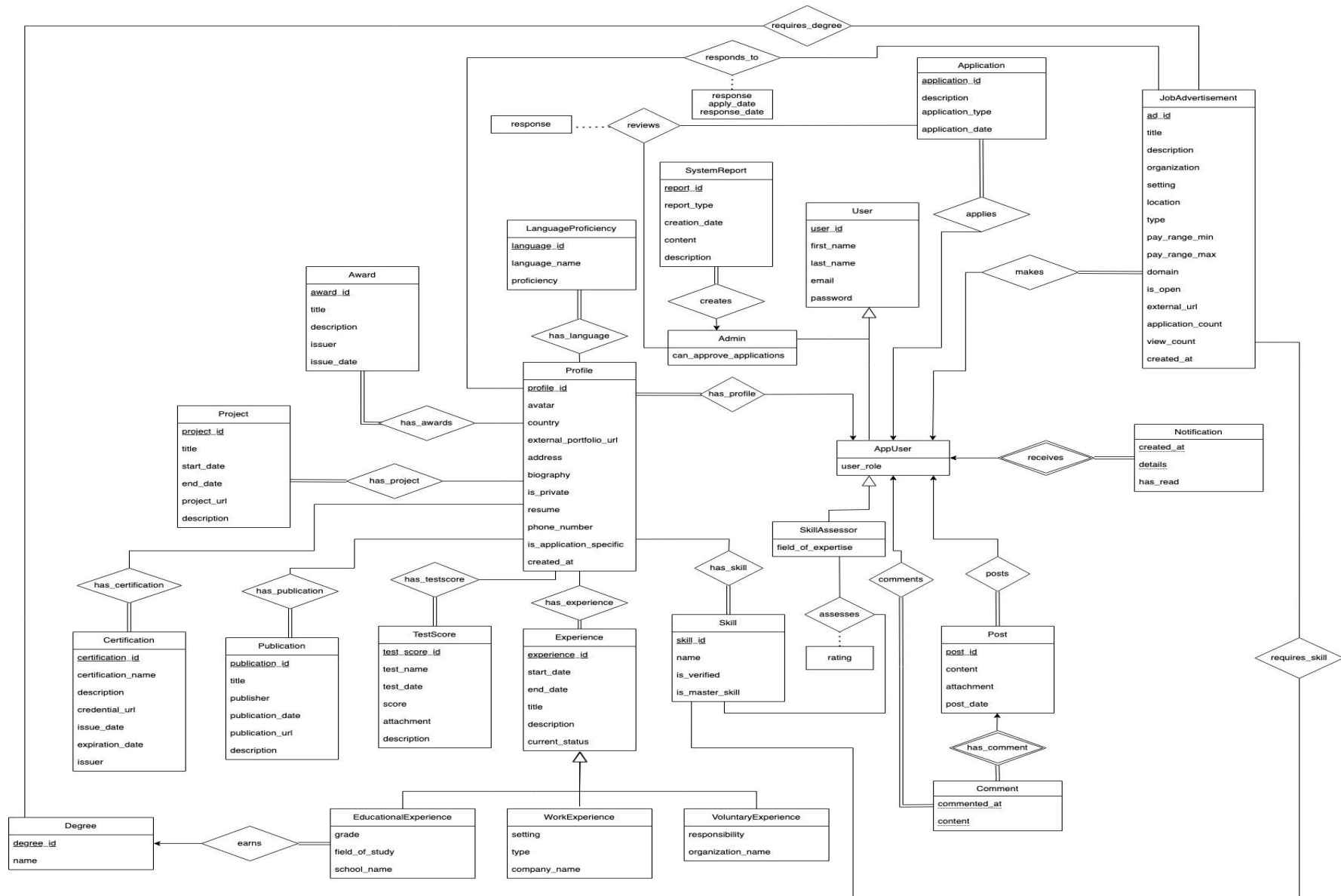


Figure 1: Final Entity-Relationship Model of the CareerBridge Application

## 4 List of Tables

### User

**Relational Model:** User(user\_id, first\_name, last\_name, email, password)

**Primary Key:** {(user\_id)}

**Foreign Keys:** None

### AppUser

**Relational Model:** AppUser(user\_id, user\_role)

**Primary Key:** {(user\_id)}

**Foreign Keys:** user\_id references User(user\_id)

### Admin

**Relational Model:** Admin(user\_id, can\_approve\_applications)

**Primary Key:** {(user\_id)}

**Foreign Keys:** user\_id references User(user\_id)

### SystemReport

**Relational Model:** SystemReport(report\_id, report\_type, creation\_date, content, description, admin\_id)

**Primary Key:** {(report\_id)}

**Foreign Keys:** admin\_id references Admin(user\_id)

### Application

**Relational Model:** Application(application\_id, user\_id, description, application\_type, application\_date)

**Primary Key:** {(application\_id)}

**Foreign Keys:** user\_id references AppUser(user\_id)

### ApplicationReview

**Relational Model:** ApplicationReview(application\_id, admin\_id, response)

**Primary Key:** {(application\_id, admin\_id)}

**Foreign Keys:** application\_id references Application(application\_id),  
admin\_id references Admin(user\_id)

### Notification

**Relational Model:** Notification(user\_id, created\_at, details)

**Primary Key:** {(user\_id, created\_at, details)}

**Foreign Keys:** user\_id references AppUser(user\_id)

### Post

**Relational Model:** Post(post\_id, user\_id, content, attachment, post\_date)

**Primary Key:** {(post\_id)}

**Foreign Keys:** user\_id references AppUser(user\_id)



### **Comment**

**Relational Model:** Comment(post\_id, content, commented\_at, user\_id)

**Primary Key:** {(post\_id, content, commented\_at)}

**Foreign Keys:** post\_id references Post(post\_id), user\_id references AppUser(user\_id)

### **JobAdvertisement**

**Relational Model:** JobAdvertisement(ad\_id, creator\_id, title, description, organization, setting, location, type, pay\_range\_min, pay\_range\_max, domain, is\_open, external\_url, application\_count, view\_count, created\_at)

**Primary Key:** {(ad\_id)}

**Foreign Keys:** creator\_id references AppUser(user\_id)

### **JobAdvertisementResponse**

**Relational Model:** JobAdvertisementResponse(profile\_id, ad\_id, response)

**Primary Key:** {(profile\_id, ad\_id)}

**Foreign Keys:** user\_id references Profile(profile\_id), ad\_id references JobAdvertisement(ad\_id)

### **Profile**

**Relational Model:** Profile(profile\_id, user\_id, avatar, country, external\_portfolio\_url, address, biography, is\_private, resume, phone\_number, is\_application\_specific, created\_at)

**Primary Key:** {(profile\_id)}

**Foreign Keys:** user\_id references AppUser(user\_id)

### **LanguageProficiency**

**Relational Model:** LanguageProficiency(language\_id, profile\_id, language\_name, proficiency)

**Primary Key:** {(language\_id)}

**Foreign Keys:** profile\_id references Profile(profile\_id)

### **Award**

**Relational Model:** Award(award\_id, profile\_id, title, description, issue\_date, issuer)

**Primary Key:** {(award\_id)}

**Foreign Keys:** profile\_id references Profile(profile\_id)

### **Project**

**Relational Model:** Project(project\_id, profile\_id, title, description, start\_date, end\_date, project\_url)

**Primary Key:** {(project\_id)}

**Foreign Keys:** profile\_id references Profile(profile\_id)

### **Certification**

**Relational Model:** Certification(certification\_id, profile\_id, certification\_name, description, credential\_url, issue\_date, issuer, expiration\_date)

**Primary Key:** {(certification\_id)}

**Foreign Keys:** profile\_id references Profile(profile\_id)

## **Publication**

**Relational Model:** Publication(publication\_id, profile\_id, title, description, publication\_date, publisher, publication\_url)

**Primary Key:** {(publication\_id)}

**Foreign Keys:** profile\_id references Profile(profile\_id)

## **TestScore**

**Relational Model:** TestScore(test\_score\_id, profile\_id, test\_name, description, test\_date, score, attachment)

**Primary Key:** {(test\_score\_id)}

**Foreign Keys:** profile\_id references Profile(profile\_id)

## **Skill**

**Relational Model:** Skill(skill\_id, profile\_id, name, is\_verified, is\_master\_skill)

**Primary Key:** {(skill\_id)}

**Foreign Keys:** profile\_id references Profile(profile\_id)

## **SkillInJobAdvertisement**

**Relational Model:** SkillInJobAdvertisement(skill\_id, ad\_id)

**Primary Key:** {(skill\_id, ad\_id)}

**Foreign Keys:** skill\_id references Skill(skill\_id), ad\_id references JobAdvertisement(ad\_id)

## **Experience**

**Relational Model:** Experience(experience\_id, profile\_id, title, start\_date, end\_date, description, current\_status)

**Primary Key:** {(experience\_id)}

**Foreign Keys:** profile\_id references Profile(profile\_id)

## **EducationalExperience**

**Relational Model:** EducationalExperience(experience\_id, grade, field\_of\_study, school\_name, degree\_id)

**Primary Key:** {(experience\_id)}

**Foreign Keys:** experience\_id references Experience(experience\_id), school\_name references School(school\_name), degree\_id references Degree(degree\_id)

## **Degree**

**Relational Model:** Degree(degree\_id, name)

**Primary Key:** {(degree\_id)}

**Foreign Keys:** None

## **DegreeInJobAdvertisement**

**Relational Model:** DegreeInJobAdvertisement(degree\_id, ad\_id)

**Primary Key:** {(degree\_id, ad\_id)}

**Foreign Keys:** degree\_id references Degree(degree\_id), ad\_id references JobAdvertisement(ad\_id)

**WorkExperience**

**Relational Model:** WorkExperience(experience\_id, setting, type, company\_name)

**Primary Key:** {(experience\_id)}

**Foreign Keys:** experience\_id references Experience(experience\_id), company\_name references Company(company\_name)

**VoluntaryExperience**

**Relational Model:** VoluntaryExperience(experience\_id, responsibility, organization\_name)

**Primary Key:** {(experience\_id)}

**Foreign Keys:** experience\_id references Experience(experience\_id), organization\_name references NonProfitOrganization(organization\_name)

**SkillAssessor**

**Relational Model:** SkillAssessor(user\_id, field\_of\_expertise)

**Primary Key:** {(user\_id)}

**SkillAssessment**

**Relational Model:** SkillAssessment(skill\_id, assessor\_id, rating)

**Primary Key:** {(skill\_id, assessor\_id)}

**Foreign Keys:** skill\_id references Skill(skill\_id), assessor\_id references SkillAssessor(user\_id)

## 5 Implementation Details

### 5.1 Environment, Frameworks, and Languages

Python FastAPI was chosen as the primary backend framework to handle interactions between the database and the website. It provided a fast and efficient way to build web APIs. Angular, a popular frontend framework, was utilized to handle the processing of the user interface. Material UI, a CSS framework, was also employed to enhance the UI of the pages of CareerBridge. Given the website's requirement for high user interaction and dynamic content generation based on user input, JavaScript was employed to dynamically create content. In that sense, JavaScript, HTML and CSS were the main languages used for frontend development.

Regarding the database management system (DBMS), MySQL version 8.0 was used as the database system to store and manage data. Additionally, Docker was utilized to containerize the application, enabling easy deployment and management of the entire system. This allowed for consistent and reproducible environments, facilitating the setup and maintenance of the application.

## 5.2 Encountered Problems

### 5.2.1 Queries Taking Too Long But The Pages Are Quick To Load

When we first connected the backend to the frontend, we noticed that the queries were taking too long to get a response. However, Angular was able to render the pages more quickly. To prevent this, we blocked the UI until the queries were completed. Once the queries are completed, we unblock the UI and render the page.

### 5.2.2 Data Is Not Populated When Testing

As can be guessed, in the development environment we are using a local database. However, when we run the tests, the database is not populated. Therefore, we need to populate the database before running the tests. For this purpose, we created a Docker volume with populated data and shared with the team members. Then, we mount this volume to the database container when running the tests.

### 5.2.3 Syntax Errors When Including Advanced SQL Features Such As Views, Triggers, Functions, and Procedures

In order to apply these concepts, we followed the notation and syntax as given in the textbook. However, we encountered syntax errors when running the queries. Apparently, the syntax of these concepts are not standardized. Therefore, we had to find the correct syntax for MySQL version 8.0. After checking the corresponding documentation, we have slightly modified the queries and they worked.

## 6 Advanced Database Components

### 6.1 Views

#### 6.1.1 User Login View

```
CREATE VIEW UserLogin AS
SELECT u.user_id, u.first_name, u.last_name, u.email,
       u.password
FROM User u
LEFT OUTER JOIN AppUser au ON u.user_id = au.user_id
LEFT OUTER JOIN Admin ad ON u.user_id = ad.user_id;
```

#### 6.1.2 Get User Role View

```
CREATE VIEW GetUserRole AS
SELECT u.user_id, au.user_role,
       CASE WHEN ad.can_approve_applications IS NOT NULL THEN
1 ELSE 0 END AS is_admin
FROM User u
LEFT OUTER JOIN AppUser au ON u.user_id = au.user_id
```

```
LEFT OUTER JOIN Admin ad ON u.user_id = ad.user_id;
```

## 6.2 Triggers

### 6.2.1 Skill Assessment Notification Trigger

```
CREATE Trigger skillAssessmentNotificationTrigger
AFTER INSERT ON SkillAssesment
REFERENCING NEW ROW AS new_row
FOR EACH ROW
BEGIN
WITH user_skill AS (
    SELECT user_id, name
    FROM Skill
    WHERE skill_id = new_row.skill_id
)
INSERT INTO Notification (user_id, notification_date,
notification_content)
SELECT user_id, NOW(),
    CONCAT('You have been assessed a ', new_row.rating, '
for ', name)
FROM user_skill;
END;
```

### 6.2.2 Comment Notification Trigger

```
CREATE Trigger commentNotificationTrigger
AFTER INSERT ON Comment
REFERENCING NEW ROW AS new_row
FOR EACH ROW
BEGIN
WITH post_temp AS (
    SELECT
        CONCAT(U.first_name, " ", U.last_name) AS
commentor_name,
        P.user_id as post_user_id
    FROM User as U, Post as P
    WHERE U.user_id = new_row.user_id
        AND P.post_id = new_row.post_id
)
INSERT INTO Notification (user_id, notification_date,
notification_content)
SELECT post_user_id, NOW(),
    CONCAT(commentor_name, ' commented on your post')
FROM post_temp
END;
```

### 6.2.3 Application Notification Trigger

```

CREATE TRIGGER applicationNotificationTrigger
AFTER INSERT ON ApplicationReview
REFERENCING NEW ROW AS new_row
FOR EACH ROW
BEGIN
WITH application_temp AS (
    SELECT A.user_id, A.application_type,
           CONCAT(U.first_name, " ", U.last_name) AS
admin_name
    FROM Application as A, User as U
    WHERE application_id = new_row.application_id
          AND U.user_id = new_row.admin_id
)
INSERT INTO Notification (user_id, notification_date,
notification_content)
SELECT user_id, NOW(),
       CASE WHEN new_row.response IS TRUE THEN CONCAT(
           'Your ', application_type, ' application has
been approved by ',
           admin_name
       )
       ELSE CONCAT(
           'Your ', application_type, ' application has
been rejected by ',
           admin_name
       )
       END
FROM application_temp;
END;

```

#### **6.2.4 Add User to AppUser Trigger**

```

CREATE TRIGGER add_user_to_appuser
AFTER INSERT ON User
FOR EACH ROW
BEGIN
    INSERT INTO AppUser (user_id, user_role)
    VALUES (NEW.user_id, 'Professional');
END;

```

### **6.3 Assertions**

#### **6.3.1 Check if Each User Has at Most One Master Skill**

```

CREATE ASSERTION num_of_master_skill
CHECK (1 >= all(SELECT COUNT(*)
                FROM Skill S, Profile P
                WHERE S.profile_id = P.profile_id AND
                S.is_master_skill = 1);

```

## 6.4 Procedures

### 6.4.1 Procedure for Increasing the Application Count of an Advertisement by One

```
CREATE PROCEDURE increase_application_count(IN ad_id_param
INT)
BEGIN
    UPDATE JobAdvertisement
    SET application_count = application_count + 1
    WHERE ad_id = ad_id_param;
END;
```

### 6.4.2 Procedure for Increasing the View Count of an Advertisement by One

```
CREATE PROCEDURE increase_view_count(IN ad_id_param INT)
BEGIN
    UPDATE JobAdvertisement
    SET view_count = view_count + 1
    WHERE ad_id = ad_id_param;
END;
```

## 6.5 System Report Queries

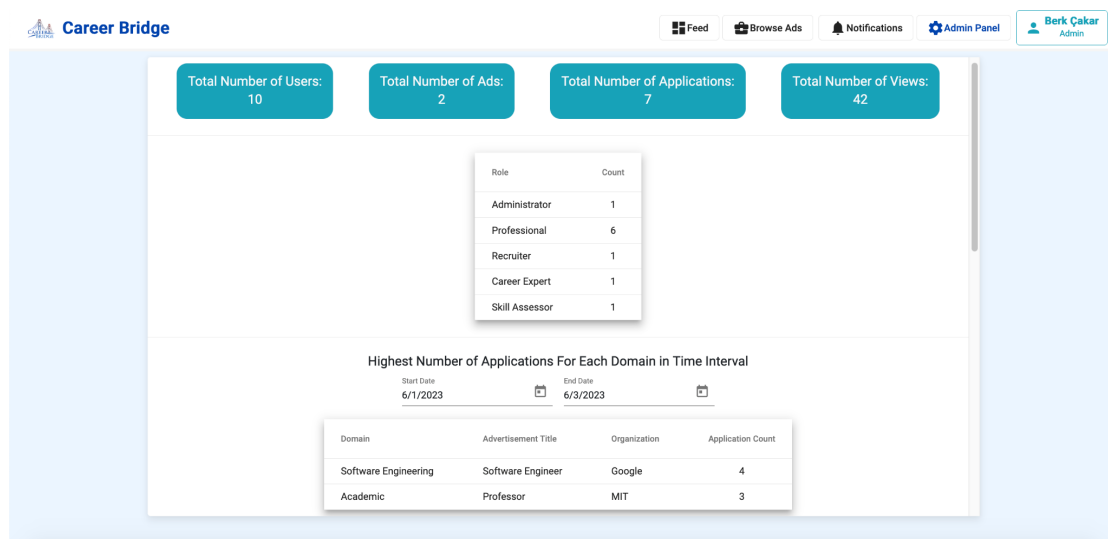


Figure 2

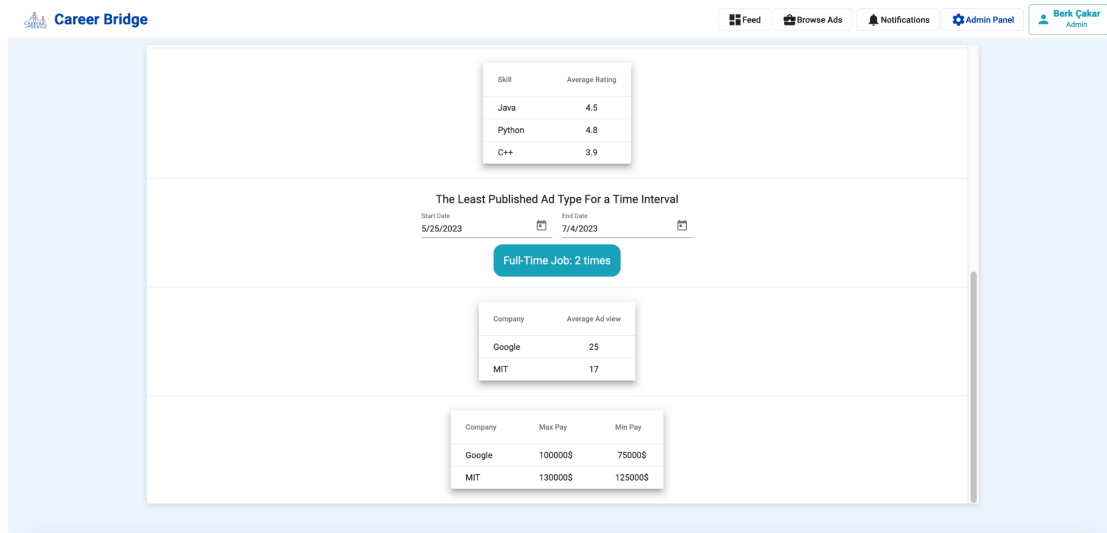


Figure 3

### 6.5.1 Query to retrieve total number of users, total number of ads, total number of ad applications, and total number of ad views in CareerBridge:

```
SELECT
    COUNT(*) AS user_count
FROM
    User;
SELECT
    COUNT(*) AS ad_count,
    SUM(view_count) as ad_view_count,
    SUM(application_count) as ad_application_count
FROM
    JobAdvertisement;
```

### 6.5.2 Query to retrieve the total number of users for each role in CareerBridge (sorting according to role name or number of users is possible):

```
SELECT
    user_role,
    COUNT(*) AS user_count
FROM
    (
        SELECT
            user_role
        FROM
            AppUser
        UNION ALL
        SELECT
            'Admin' AS user_role
        FROM
```



```

        Admin
    ) AS combined_table
GROUP BY
    user_role
ORDER BY
    user_count DESC;

```

**6.5.3 Query to retrieve the most applied ad for each domain in a given time interval (sorting according to domain, advertisement title, company name, or application count is possible):**

```

WITH domain_counts AS (
    SELECT
        JA.domain AS domain_name,
        JA.ad_id AS ad_id,
        COUNT(*) as application_count
    FROM
        JobAdvertisement JA,
        JobAdvertisementResponse JAR
    WHERE
        JA.ad_id = JAR.ad_id
        AND JAR.apply_date BETWEEN @start_date
        AND @end_date
    GROUP BY
        JA.domain,
        JA.ad_id
)
SELECT
    DC.domain_name,
    DC.ad_id,
    JA.title,
    JA.organization,
    DC.application_count
FROM
    domain_counts DC,
    JobAdvertisement JA
WHERE
    JA.ad_id = DC.ad_id
    AND DC.application_count = (
        SELECT
            MAX(application_count)
        FROM
            domain_counts
    )
ORDER BY
    DC.application_count DESC;

```

**6.5.4 Query to retrieve the average rating of available skills in CareerBridge (sorting according to skill name or average rating is possible):**

```

SELECT
    S.skill_name AS skill_name,
    AVG(SA.rating) AS average_rating
FROM
    Skill S,
    SkillAssessment SA
WHERE
    S.skill_id = SA.skill_id
GROUP BY
    S.skill_name
ORDER BY
    S.skill_name ASC;

```

#### **6.5.5 Query to retrieve the least applied type of ad in a given time interval(@start\_date and @end\_date are coming from user input):**

```

WITH ad_counts AS (
    SELECT
        JA.type AS ad_type,
        COUNT(*) as application_count
    FROM
        JobAdvertisement JA,
        JobAdvertisementResponse JAR
    WHERE
        JA.ad_id = JAR.ad_id
        AND JAR.apply_date BETWEEN @start_date
        AND @end_date
    GROUP BY
        JA.type
)
SELECT
    AC.ad_type,
    AC.application_count
FROM
    ad_counts AC
WHERE
    AC.application_count = (
        SELECT
            MIN(application_count)
        FROM
            ad_counts);

```

#### **6.5.6 Query to retrieve the average number of job advertisement view for each organization :**

```

SELECT
    JA.organization,
    AVG(JA.view_count) AS average_view_count
FROM
    JobAdvertisement JA

```

```
GROUP BY
    JA.organization
ORDER BY
    JA.organization ASC;
```

### 6.5.7 Query to retrieve the average pay range for job advertisements for each organization:

```
SELECT
    JA.organization,
    AVG(JA.pay_range_min) AS average_min_pay,
    AVG(JA.pay_range_max) AS average_max_pay
FROM
    JobAdvertisement JA
GROUP BY
    JA.organization
ORDER BY
    JA.organization ASC;
```

## 7 User Manual

### 7.1 Login Page

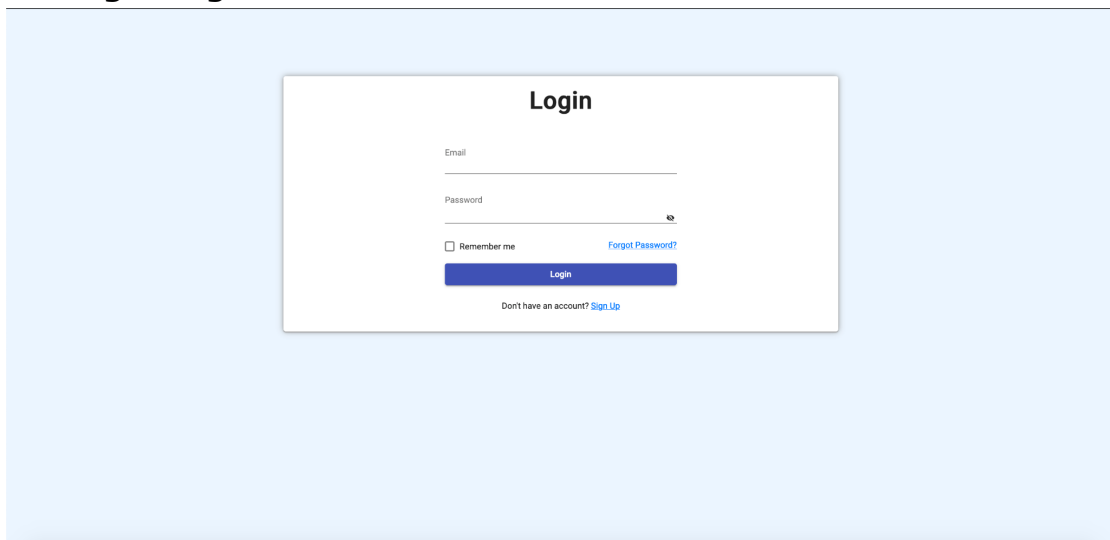


Figure 4

The first thing the user encounters is the Login page. If the user already has an account, s/he can login the app by entering his/her email address and password. All user types (admin, recruiter, employee, skill assessor) login the app through this page. However, if the user does not have an account yet, s/he can click the **Sign Up** button that directs him/her to the Register page.

## 7.2 Register Page

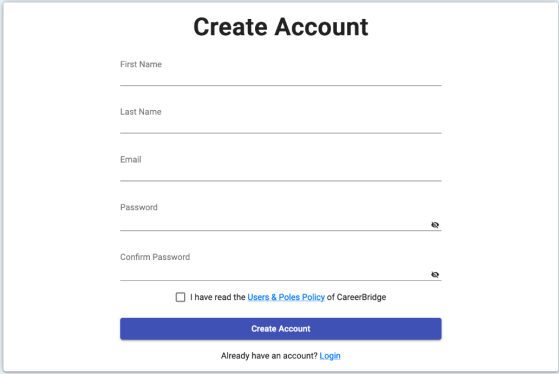
The image shows a 'Create Account' registration form centered on a light blue background. The form is a white rectangle with a thin grey border. At the top, the title 'Create Account' is centered in a bold, black font. Below the title are five input fields, each with a label to its left: 'First Name', 'Last Name', 'Email', 'Password', and 'Confirm Password'. The 'Password' and 'Confirm Password' fields have a small eye icon to their right, indicating a toggle for password visibility. Below these fields is a checkbox with the text 'I have read the Users & Poles Policy of CareerBridge'. At the bottom of the form is a solid blue button with the text 'Create Account' in white. Below the button, centered, is the text 'Already have an account? Login' with 'Login' as a blue hyperlink.

Figure 5

After the user is directed to the Register page, s/he can create an account by specifying an email address and a password. The user also needs to confirm this password by entering it for a second time. Final stage for creating the account is to check the box for User's Policy. After all this, clicking the Create Account button successfully creates the account and the user can login the app through the Login page.

## 7.3 Browse Ads Page

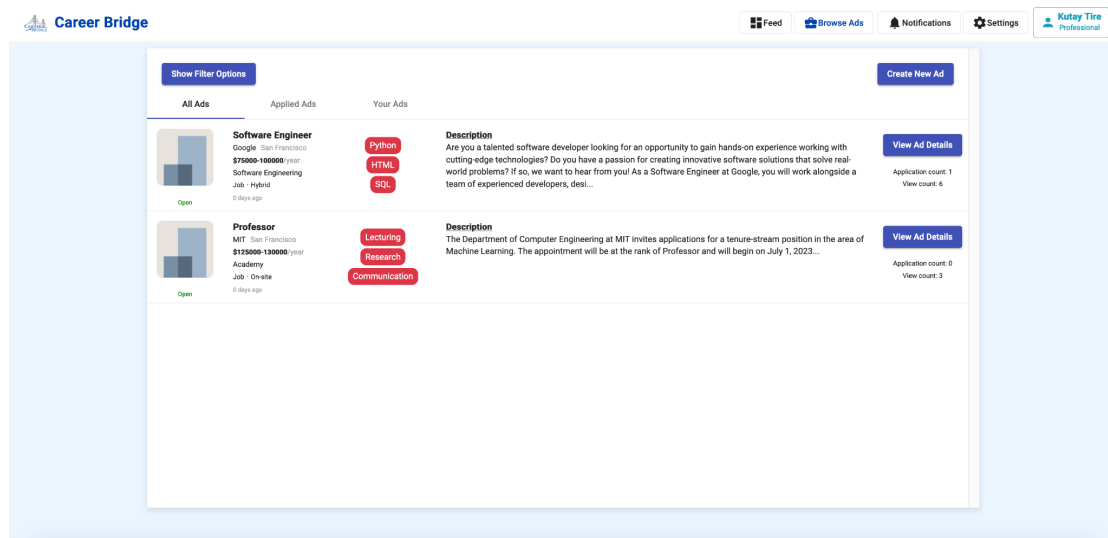


Figure 6

One of the main pages of CareerBridge is the "Browse Ads Page" that is available to all app users. In this page, all given advertisements that are being stored in the JobAdvertisement table are listed. The user can see the organization that published the ad, job title, the location, type and domain of the job, required skills, the pay range, and a brief description about the job. In addition to these, the view and application counts are also demonstrated. When the user presses the View Ad Details button, the view count is automatically updated. These ads can be filtered which is explained in the next page.

## 7.4 Browse Ads Page with Filtering Options

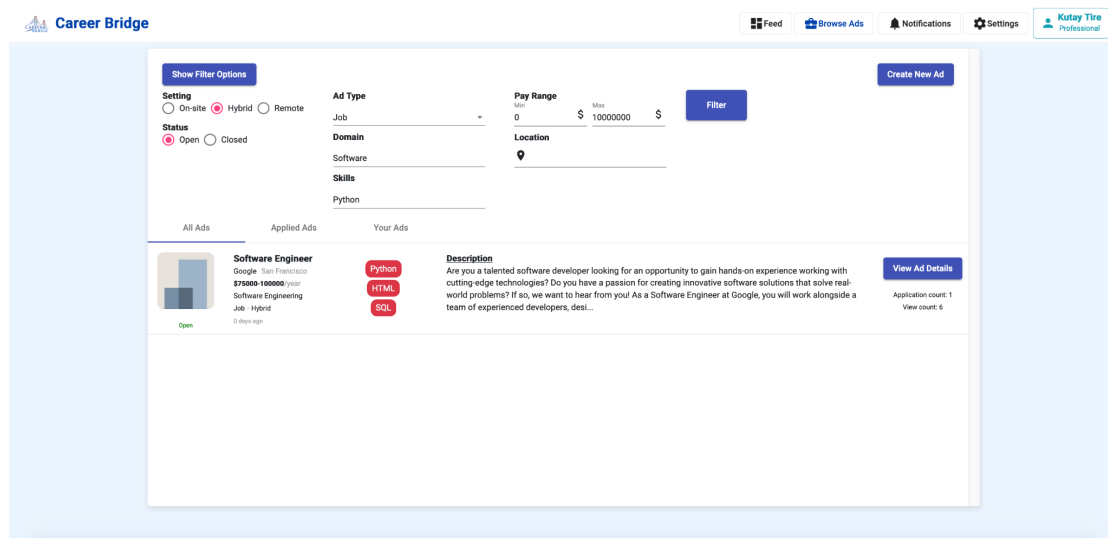


Figure 7

This page shows the possible filtering options that appear when the Show Filtering Options button is clicked. As it can be seen, the ads can be filtered by their types, pay range, location, setting, status, domain, skills and required degrees. This scenario illustrates a possible filtering combination and the result after clicking the Apply Filter button. From this page, the users can make the necessary adjustments and search for the ads that suit them. Naturally, some of the filtering options can be left empty and then the resultant query ignores them.

## 7.5 Ad Description Page

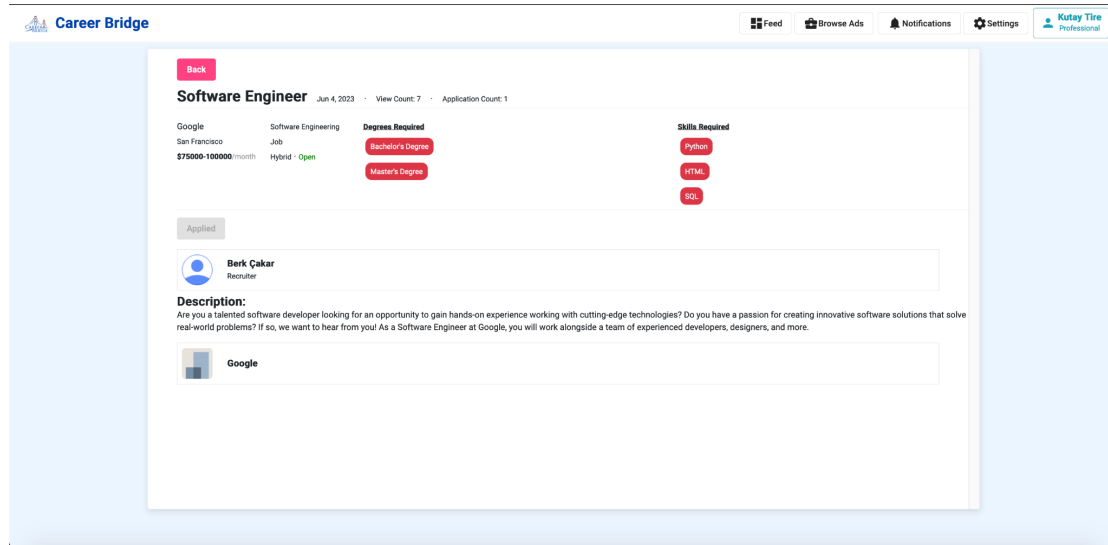


Figure 8

This page opens when the user clicks the View Ad Details button from the Browse Ads page. As it can be seen, the page shows the title of the ad, required skills and degrees, the post date of the ad, pay range, type, and the location of the job at the very top. The user can also see the application and view counts and brief information about the recruiter who posted the ad. The organization of the job and its description are also displayed. The user can apply for the job by clicking the Apply button. As all of the required information as well as the CV of the user is present on the profile, this application is made by retrieving the data from the profile of the user. With this feature, CareerBridge tries to facilitate the jobs of its users and minimize the effort required for applying for a job. After a successful application, the application count is incremented by one and the respective recruiter can now see the application.

## 7.6 Applied Ads Page

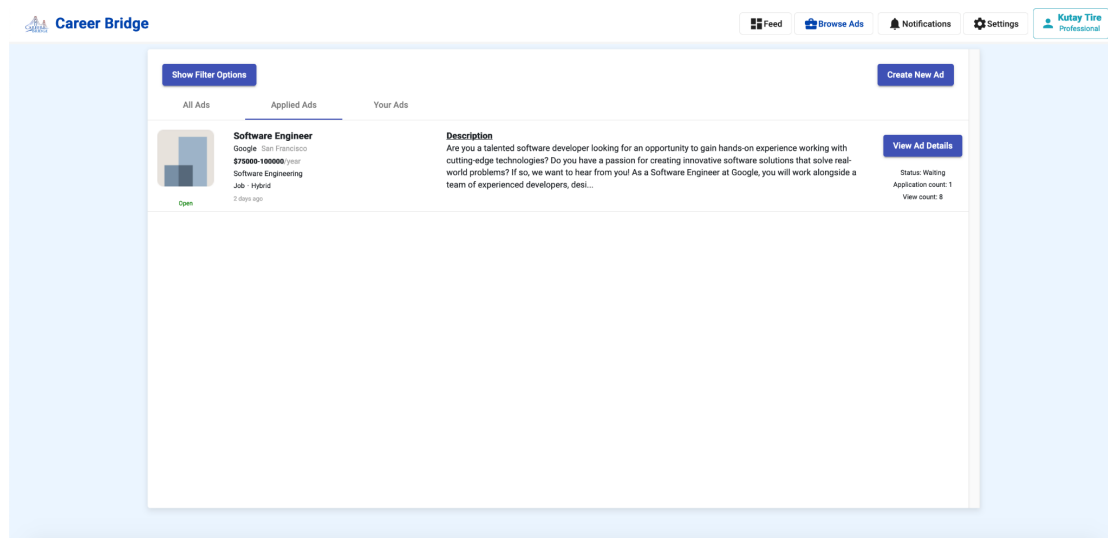


Figure 9

After the user applies for a job, s/he can see the application and track the status of it through the Applied Ads page by clicking the Applied Ads button. The application date together with the mentioned attributes of the advertisements can be seen from here. Also, the status of the applications (Waiting, Accepted, Rejected) can be seen from this page.



## 7.7 Published Ads Page

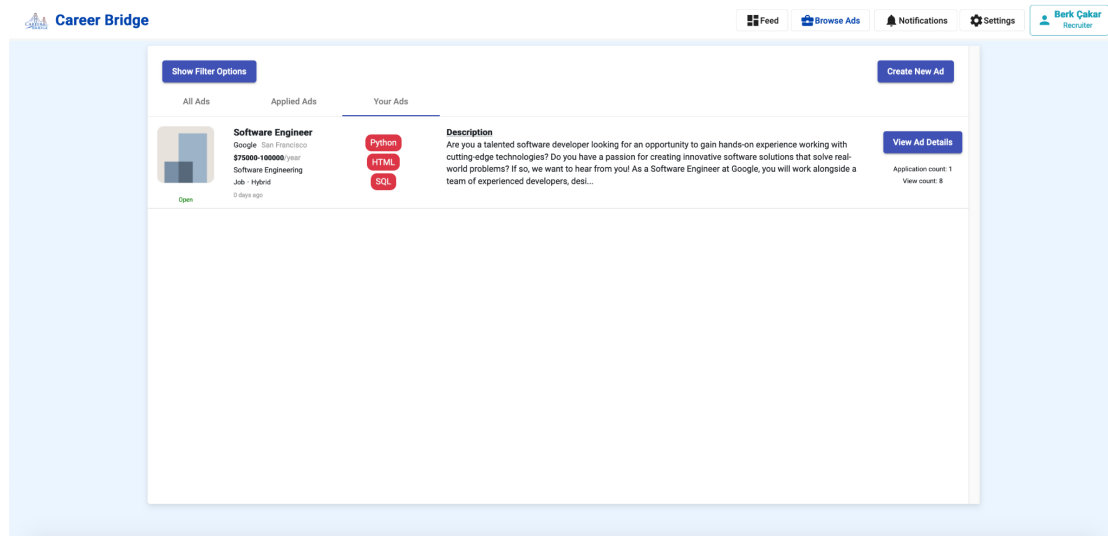


Figure 10

This page is available for the recruiters in CareerBridge and can be accessed by clicking the **Your Ads** button. From this page, the recruiters can see all the ads they published, the view and application counts of the ads and their details. They can also delete or modify an ad if they please. They can view the candidates that applied for a job ad by clicking the **View Candidates** button. The opened page after clicking the button is displayed below.

## 7.8 Candidates (Applicants) of an Ad Page

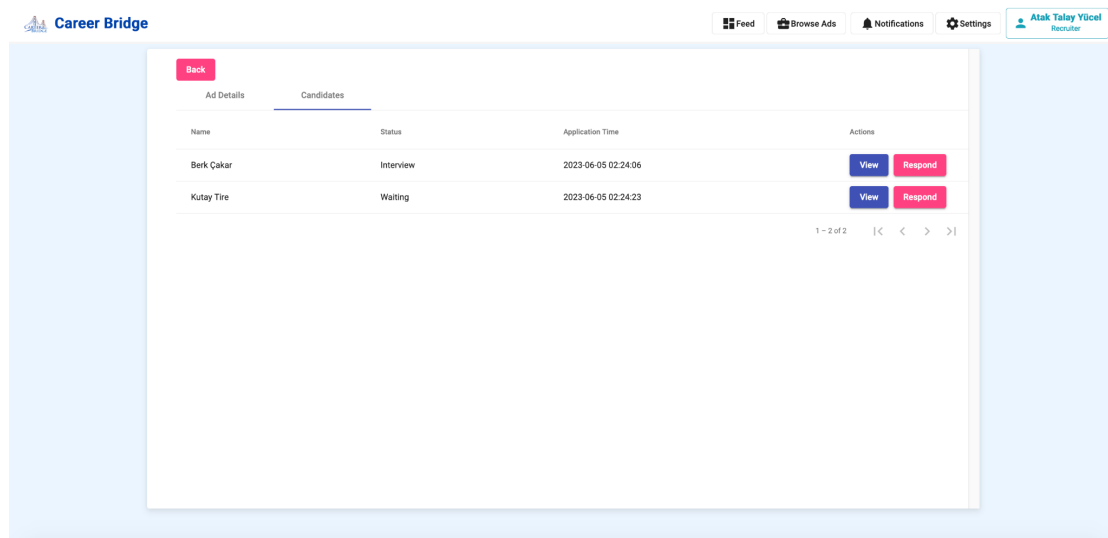


Figure 11

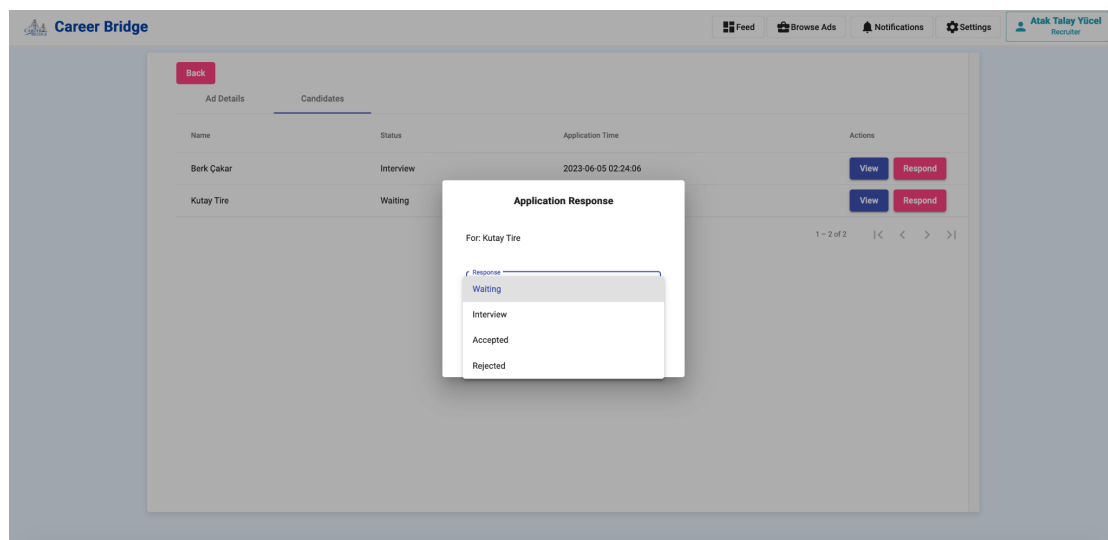


Figure 12

The recruiters can also see the possible candidates that applied for their ads. From this page, they can access some basic information about the applicants such as their email addresses or names. The status of the application is also demonstrated here which can be updated by clicking the **Respond** button for the respective advertisement. The recruiters will make their decisions by clicking the **Profile** button and looking at the profile information of the applicant. The recruiters will be able to either accept or reject the application. After the necessary response is given, the user who made the application will also be able to see it.

## 7.9 Publish New Ad Page

The screenshot shows the 'Create New Ad' form in the Career Bridge application. The form is a modal window with a white background and a pink 'Post' button at the bottom. It contains the following fields and data:

- Title \***: Database Admin
- Organization Name \***: BCC
- Location \***: Ankara, Turkey
- Domain \***: Software
- Setting**: On-site (dropdown menu)
- Type**: Job (dropdown menu)
- Pay Range**: 12000 \$ - 15000 \$
- Please fill below fields with comma separated elements**:
  - Engines Required \***: Bachelors of Science, Bachelors of Engineering
  - Skills Required \***: MySQL, Bash, Python, Linux
- Description**: We are looking for a full-time Database Admin!

The background shows the Career Bridge interface with a sidebar on the left containing job listings like 'Software Engineer' and 'Professor', and a main area on the right with 'View Ad Details' buttons. The top navigation bar includes links for Feed, Browse Ads, Notifications, Settings, and a user profile for Berk Çakar, Recruiter.

Figure 13

From this page, the recruiter can publish a new job advertisement. This page is accessed by clicking the **Create New Ad** from the Browse Ads page. The recruiters have to specify the title, location, pay range, type, domain, status, setting and description of the job in addition to required degrees and skills. After that, the advertisement is displayed to the users.

## 7.10 Profile Page

Profile page is accessed by clicking the avatar and username at the top right corner of the app. The opened segments after entering the profile of the user are explained below. The profile page itself contains the avatar of the user, his/her username, and verified skill with its rating if it exists.

### 7.10.1 Information Page

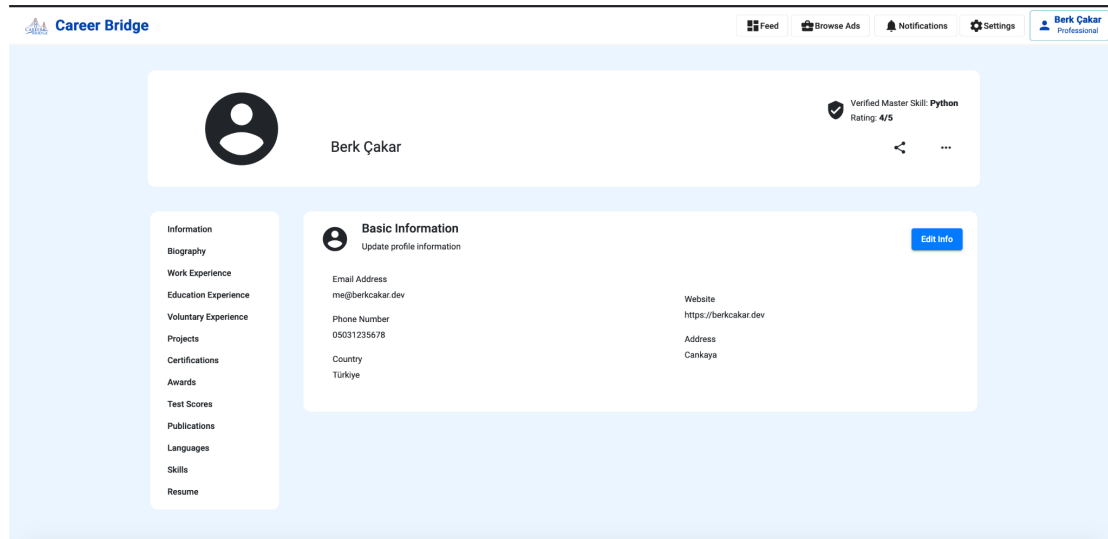


Figure 14

The information page includes the basic information of the user such as the email address or phone number of the user. It can be modified by clicking the pencil icon at the right.

### 7.10.2 Biography Page

This page is accessed by clicking the **Biography** button from the Profile page. It includes biography information of the user and can be modified by clicking the pencil icon at the right.

### 7.10.3 Work Experience Page

This page is accessed by clicking the **Work Experience** button from the Profile page. It includes work experience information of the user such as the job title, company name and durations. By clicking the **Add Experience** button, a new work experience can be added by filling the necessary entries.

### 7.10.4 Education Experience Page

This page is accessed by clicking the **Education Experience** button from the Profile page. It includes education experience information of the user such as the attended school, earned degree and durations. By clicking the **Add Experience** button, a new education experience can be added by filling the necessary entries.

### 7.10.5 Voluntary Experience Page

This page is accessed by clicking the **Voluntary Experience** button from the Profile page. It includes voluntary experience information of the user such as the attended non-profit organization, description of work done and durations. By clicking the **Add Experience** button, a new voluntary experience can be added by filling the necessary entries.

### 7.10.6 Projects Page

This page is accessed by clicking the **Projects** button from the Profile page. It includes project information of the user such as the detailed explanation and title of the project and its durations. By clicking the **Add Project** button, a new project can be added by filling the necessary entries.

### 7.10.7 Certifications Page

This page is accessed by clicking the **Certifications** button from the Profile page. It includes certifications information of the user such as the title of the certification, its description and issue and expiration dates. By clicking the **Add Certification** button, a new certification can be added by filling the necessary entries.

### 7.10.8 Awards Page

This page is accessed by clicking the **Awards** button from the Profile page. It includes award information of the user such as the issuer of the award, its title and issue date, and description of the award. By clicking the **Add Award** button, a new award can be added by filling the necessary entries.

### 7.10.9 Test Scores Page

This page is accessed by clicking the **Test Scores** button from the Profile page. It includes test score information of the user such as the title, score and date of the test. By clicking the **Add Test Score** button, a new test score can be added by filling the necessary entries.

### 7.10.10 Publications Page

This page is accessed by clicking the **Publications** button from the Profile page. It includes publication information of the user such as the title and date of the publication or its publisher. By clicking the **Add Publication** button, a new publication can be added by filling the necessary entries.

### 7.10.11 Languages Page

This page is accessed by clicking the **Languages** button from the Profile page. It includes language information of the user such as the language name and the proficiency of the user. By clicking the **Add Language** button, a new language can be added by filling the necessary entries.

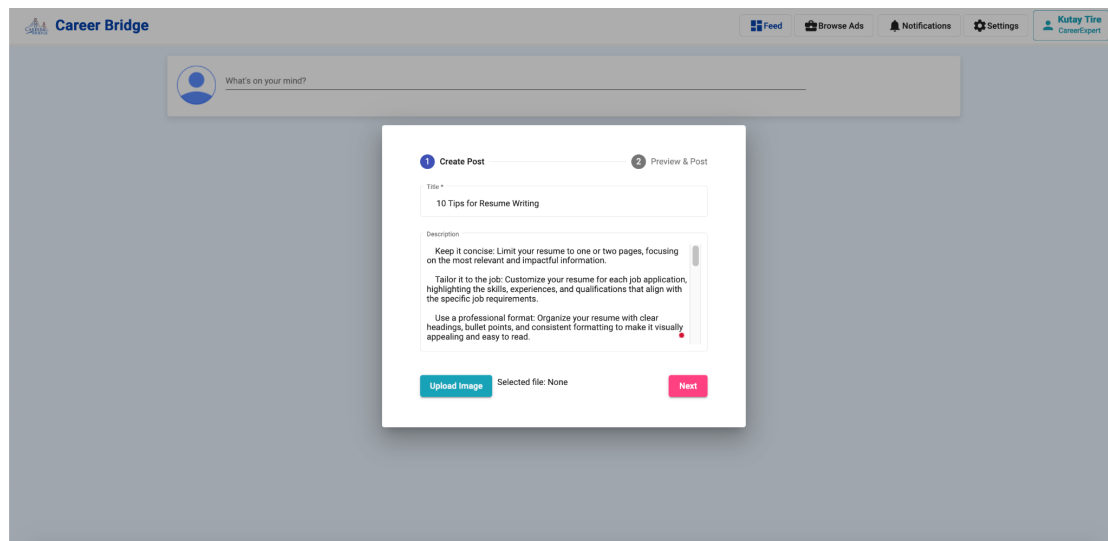
#### **7.10.12 Skills Page**

This page is accessed by clicking the **Skills** button from the Profile page. It includes skill information of the user such as name of the skill, whether it is a master skill or whether it is verified by a skill assessor. By clicking the **Add Skill** button, a new skill can be added by filling the necessary entries.

#### **7.10.13 Resume Page**

This page is accessed by clicking the **Resume** button from the Profile page. From this page, a user can upload his/her resume and it is later reviewed by the recruiters when the user makes a job application. The resume can be uploaded by clicking the **Upload Resume** button and selecting the file to upload.

## 7.11 User Feed



The screenshot shows the 'Create Post' dialog box in the Career Bridge application. The dialog has two steps: '1 Create Post' and '2 Preview & Post'. The 'Create Post' step is active, showing a form with a 'Title' field containing '10 Tips for Resume Writing' and a 'Description' field containing several paragraphs of text. Below the description field, there is an 'Upload Image' button and a 'Selected file: None' label. A 'Next' button is located at the bottom right of the dialog. The background shows the Career Bridge interface with a navigation bar at the top and a user profile section on the left.

Figure 15

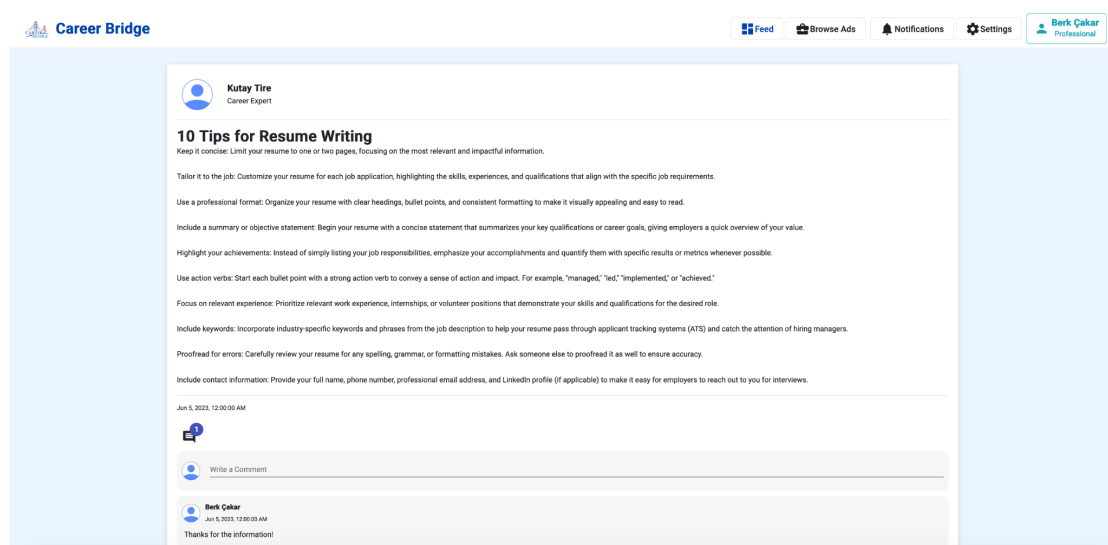


Figure 16

The Feed page is available to all users and can be accessed by clicking the **Feed** button. From the Feed page, career experts can create informative posts which are then displayed to all users. All users have the ability to comment on posts. Creating a post is done by clicking the "What's on your mind?" part at the top of the page. This page opens a dialogue from where the recruiters can write the title and description of their posts. By clicking the "Write a comment" part of the post, a user can write his/her comment.

## 7.12 Admin Page

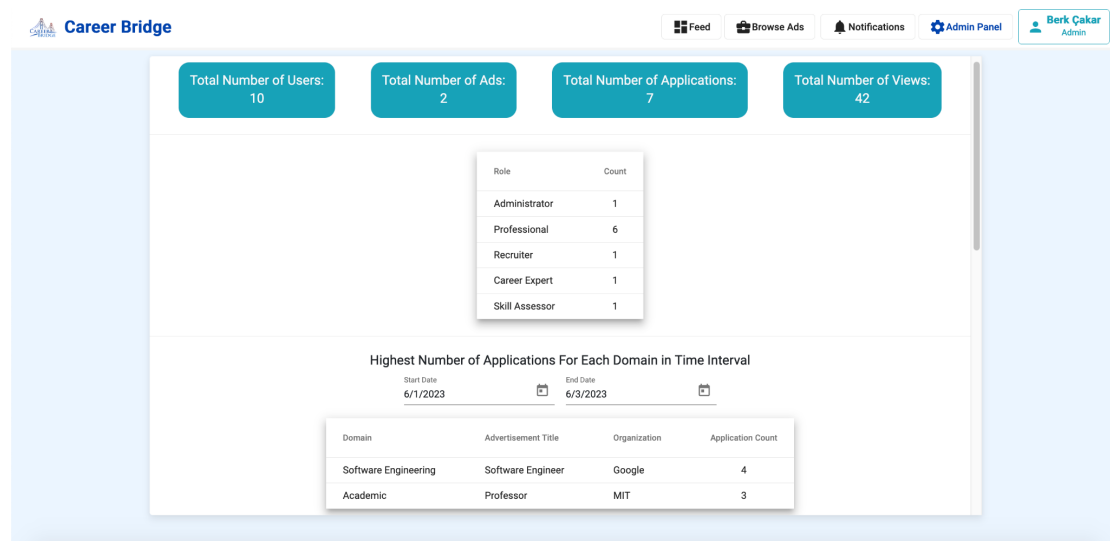


Figure 17

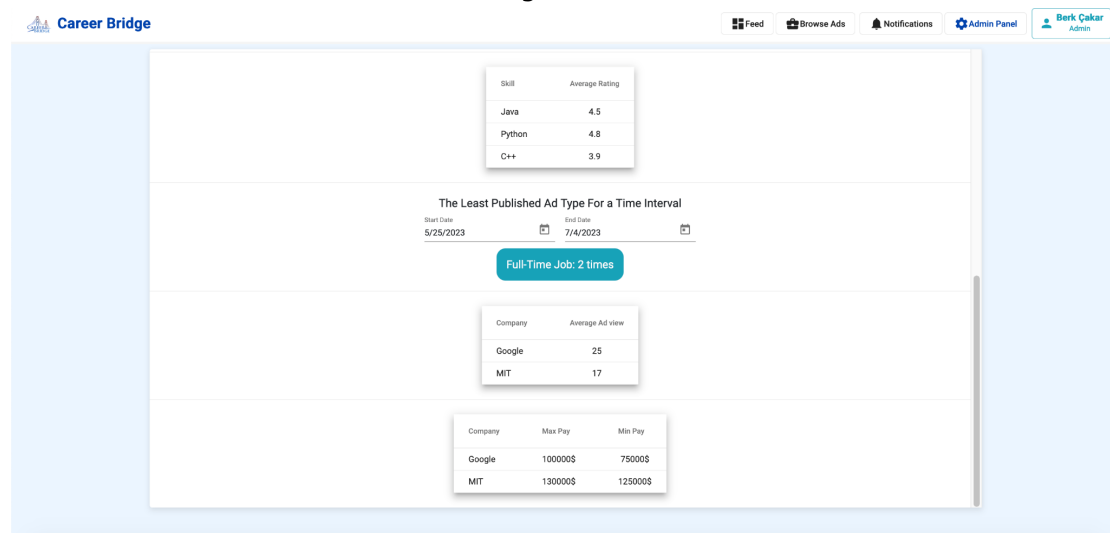


Figure 18

This is the specialized page for the admins of CareerExpert. When the **Admin Panel** button is pressed, the system shows some important data about CareerBridge. As this button is only available to admins, admins are the only ones who can see these reports. The page consists of information about the total number of users, ads, applications and ad views. Some following information is about the total number of users in each role, highest number of applications in each role, average skill rating for each skill, least published ad type for a time interval and average number of ad views for each company. Final information is about average minimum and maximum pay for each company. Admins can use this information to track CareerBridge overall.