

**JOBIFY**

**Software Design Document**

– FPT University, Oct 2021 –

**Table of Contents**

[I. Overview 3](#_heading=h.gjdgxs)

[1. Code Packages 3](#_heading=h.30j0zll)

[2. Coding Conventions 3](#_heading=h.1fob9te)

[II. Code Designs 4](#_heading=h.3znysh7)

[1. Account management 4](#_heading=h.2et92p0)

[a. Class Diagram 4](#_heading=h.tyjcwt)

[b. Class Specifications](#_heading=h.a1j5epfe1rmo) 4

[c. Sequence Diagram(s) 4](#_heading=h.1t3h5sf)

[d. Database queries 5](#_heading=h.4d34og8)

[2. Banner management 5](#_heading=h.2s8eyo1)

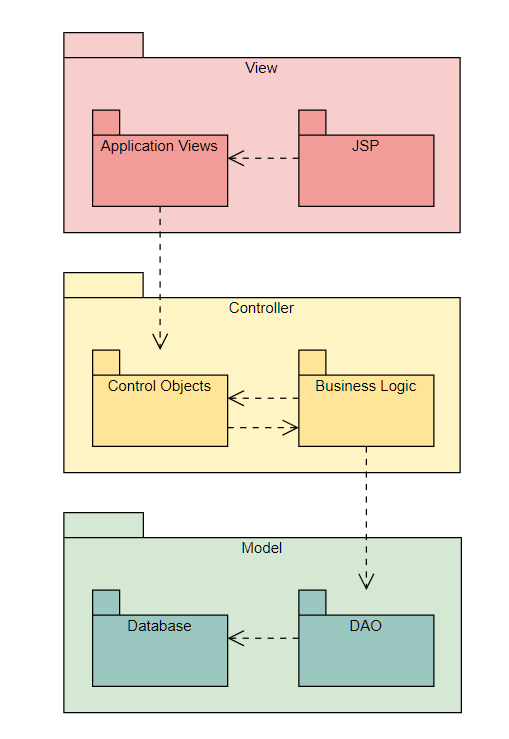
[III. Database Tables 5](#_heading=h.17dp8vu)

[1. <Table name 1> 5](#_heading=h.3rdcrjn)

[2. <Table name 2…> 5](#_heading=h.26in1rg)

# I. Overview

## 1. Code Packages/Namespaces



***Package descriptions & package class naming conventions***

| **No** | **Package** | **Description** |
| --- | --- | --- |
| *01* | *Controller* | *4 sub-package: admin, employee, employer, common* |
| *02* | *DAO* | *Connect to database* |
| *03* | *View* |  |
| *04* | *Model* |  |

## 2. Coding Conventions

* Follow camelCase convention
* Class names should be nouns, in mixed cases with the first letter of each internal word capitalized. Interface names should also be capitalized just like class names.
* Methods should be verbs, in mixed case with the first letter lowercase and with the first letter of each internal word capitalized.
* Variable names should be short yet meaningful. One-character variable names should be avoided except for temporary variables.

# II. Code Designs

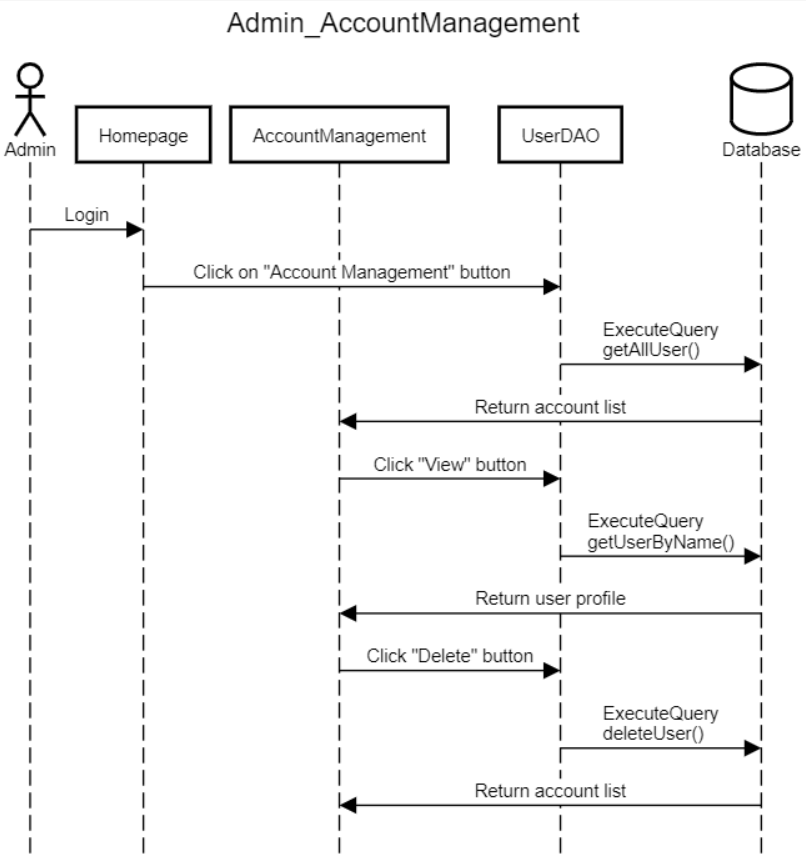
## Account management

### 

### Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *AccountManagement* | *When admin wants to see list of accounts on system, it will call thí method to select all account on database* |

### Sequence Diagram(s)



### Database queries

* View user list: select \* from [user]
* Delete user: DELETE FROM [user] WHERE username=?
* View user profile: select u.username,u.password,u.email,u.fullname,u.dob,u.gender, u.address,u.phone,u.image,u.roleid,u.locationid,u.majorid,r.name,l.name, m.name from [user] u join role r on r.id= u.roleid join location l on l.id= u.locationid join major m on m.id= u.majorid where username=? and password=?

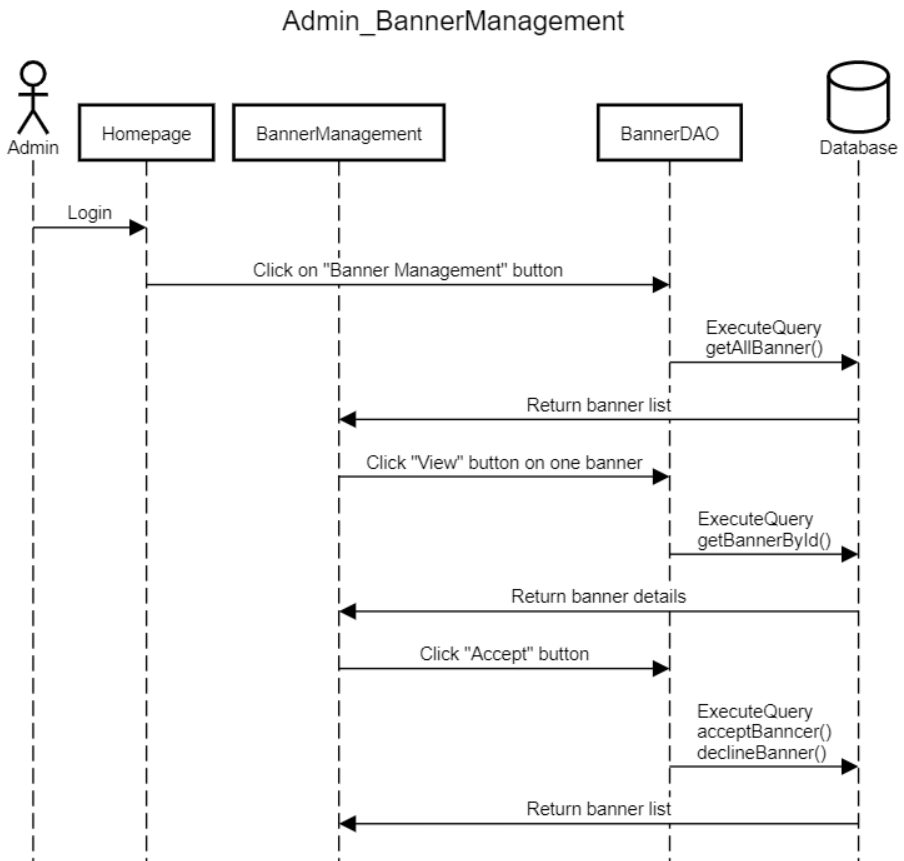
## 2. Banner management

### 

### Class specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *Banner Management* | *When admin wants to approve or decline one banner posted by an employers, it will call this method to view all list banners on database* |

### Sequence diagram



### Database queries

01.Banner Management

select \* from banner join job on jobid=job.id join status on statusid= status.id join Location on job.locationid= Location.id join Major on job.majorid= Major.id join Jobtype on job.jobtypeid= Jobtype.id where statusid =1 order by banner.id offset ? rows fetch next ? rows only

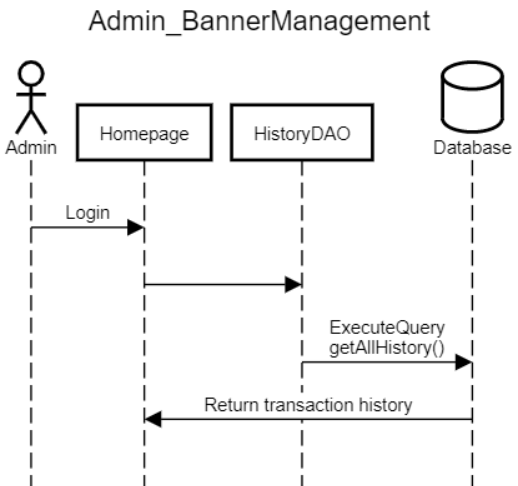
## 3. View transaction history

### 

### Class specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *Banner Management* | *When admin wants to approve or decline one banner posted by an employers, it will call this method to view all list banners on database* |

### Sequence diagram



### Database queries

## 4. CV Management Employees

### 

### Class specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *CreateCV* | *When the user inputs all fields of a CV and clicks “Create CV”, it will call this method to add a CV to the database* |
| *02* | *EditCV* | *When employee inputs all fields of a CV and clicks “Edit CV”, it will call this method to update information of the CV in the database* |
| *03* | *ViewCV* | *When employee wants to see his CV and clicks on “View CV” button, it will call this method to view his CV* |

### Sequence diagram

### 

### Database queries

01. CreateCV

insert into cv(fullname,dob, gender,locationid,phone,contact,educationid,school,experience,username, statusid, image,time)values(?,?,?,?,?,?,?,?,?,?,?,?,?)

02. EditCV

update cv set fullname=?, dob=?, gender=?, locationid=?, phone=?, contact=?, educationid=?, school=?, image=?,experience=? where id=?

0.3 ViewCV

select \* from cv join education on educationid= education.id join location on locationid=location.id where username=? and jobid is null order by cv.id offset ? rows fetch next ? rows only

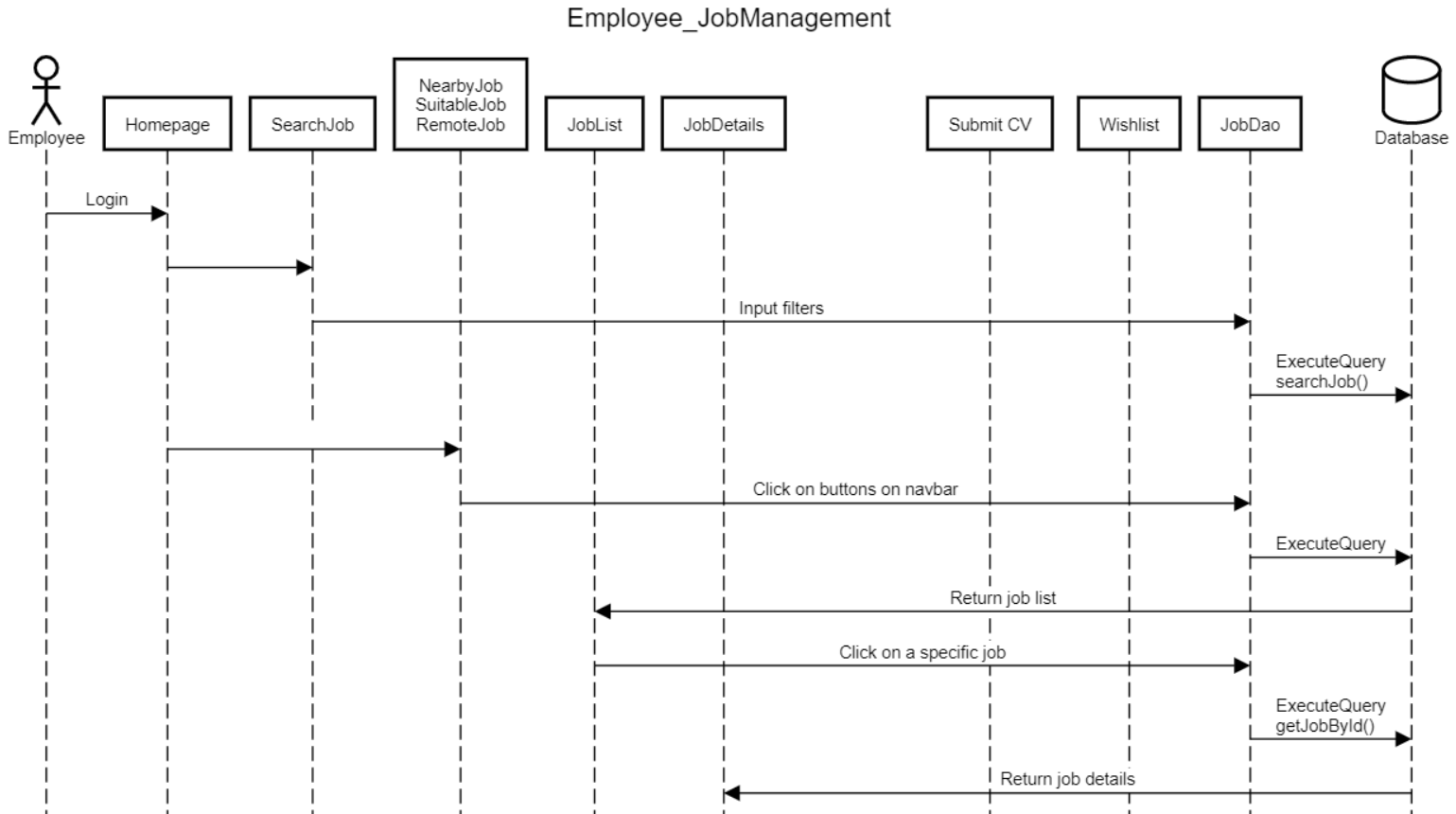
### 5. Job Management Employees

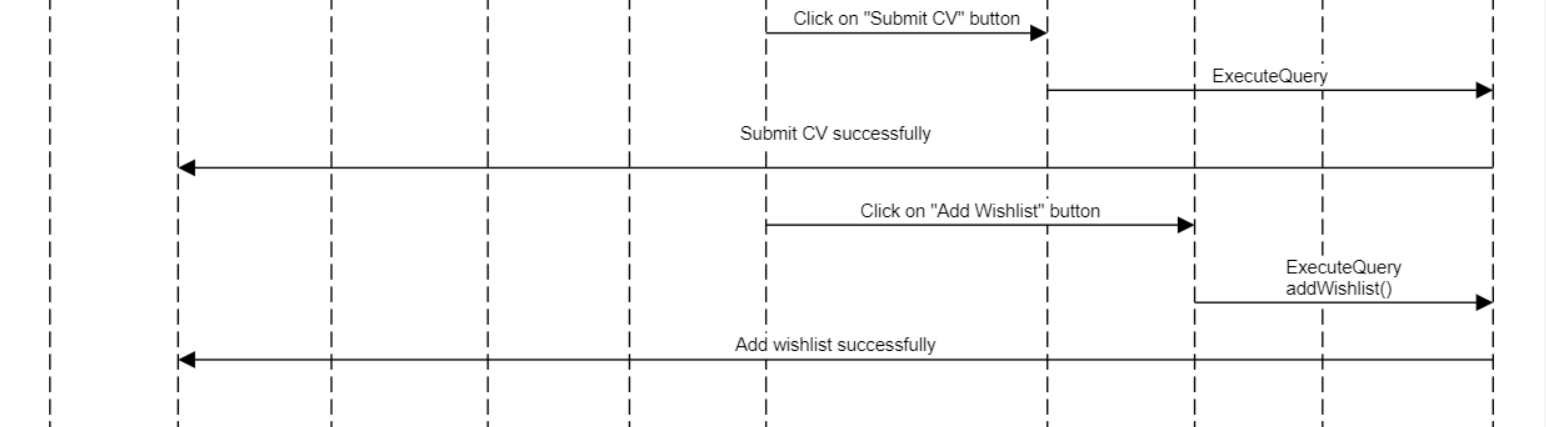
### 

### Class specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *SearchJob* | *When the user inputs fields on the search bar and dropdown list on the homepage, they can filter the job list.* |
| *02* | *FulltimeJob* | *Filter job by major of user and return Full-time job list* |
| *03* | *RemoteJob* | *Filter job by major of user and return remote job list* |
| *04* | *ParttimeJob* | *Filter job by major of user and return Part-time job list* |
| *05* | *JobDetails* | *View details of one job* |
| *06* | *SubmitCV* | *When viewing an appropriate job, employees can submit their CV to employers* |
| *07* | *AddToWishlist* | *User can save this job into wishlist to view later* |

### Sequence diagram





### Database queries

01.Search job

select \* from job join location on location.id= job.locationid join major on major.id=job.majorid join jobtype on jobtype.id= jobtypeid where (job.name like '%" + input + "%' or company like '%" + input + "%') order by job.id offset ? rows fetch next ? rows only

02.FulltimeJob

select \* from job join location on location.id= job.locationid join major on major.id=job.majorid join jobtype on jobtype.id= jobtypeid where jobtypeid= 2 and majorid=? order by job.id offset ? rows fetch next ? rows only

03.RemoteJob

select \* from job join location on location.id= job.locationid join major on major.id=job.majorid join jobtype on jobtype.id= jobtypeid where jobtypeid= 3 and majorid=? order by job.id offset ? rows fetch next ? rows only

04.ParttimeJob

select \* from job join location on location.id= job.locationid join major on major.id=job.majorid join jobtype on jobtype.id= jobtypeid where jobtypeid= 1 and majorid=? order by job.id offset ? rows fetch next ? rows only

05.JobDetails

select \* from job join location on location.id= job.locationid join major on major.id=job.majorid join jobtype on jobtype.id= jobtypeid where job.id=?

06.SubmitCV

insert into cv(fullname, dob, gender, locationid, phone, contact, educationid, school, experience, username, statusid, jobid)values(?,?,?,?,?,?,?,?,?,?,?,?)

07.AddToWishlist

insert into wishlist(username, jobid) values (?,?)

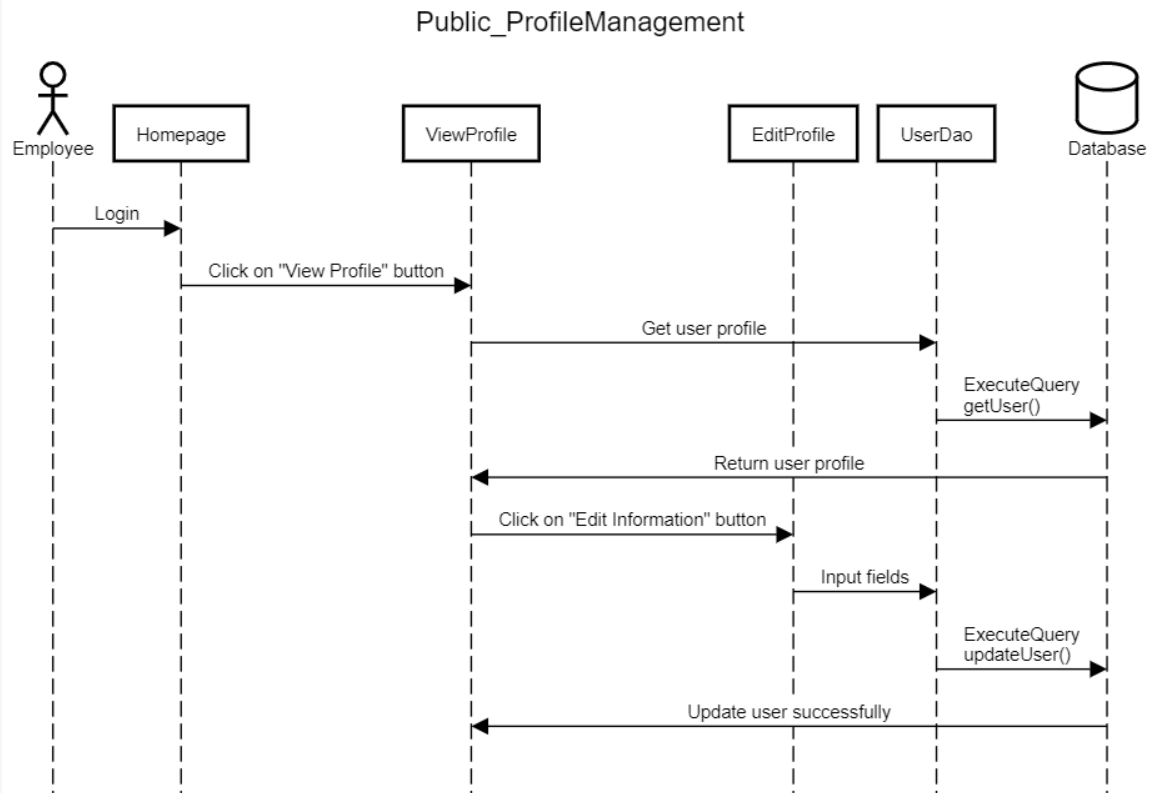
## 6. Profile Management

### 

### Class specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *ViewProfile* | *When a user wants to view their profile, they can see their personal information such as: name, location, age, major,...* |
| *02* | *EditProfile* | *User can edit their personal information if they are incorrect* |
| *03* | *ChangePassword* | *User can change password* |

### Sequence diagram



### Database queries

01.ViewProfile

-select u.username,u.password,u.email,u.fullname,u.dob,u.gender, u.address,u.phone,u.image,u.roleid,u.locationid,u.majorid,r.name,l.name, m.name from [user] join role r on r.id= u.roleid join location l on l.id= u.locationid join major m on m.id= u.majorid where username=? and password=?

02.EditProfile

-UPDATE [user] SET fullname = ?, dob=?, gender=?, address=?, phone=?, image=?, locationid=?, majorid=? WHERE username = ?

03.ChangePassword

-update [user] set [password] = ? where username=?

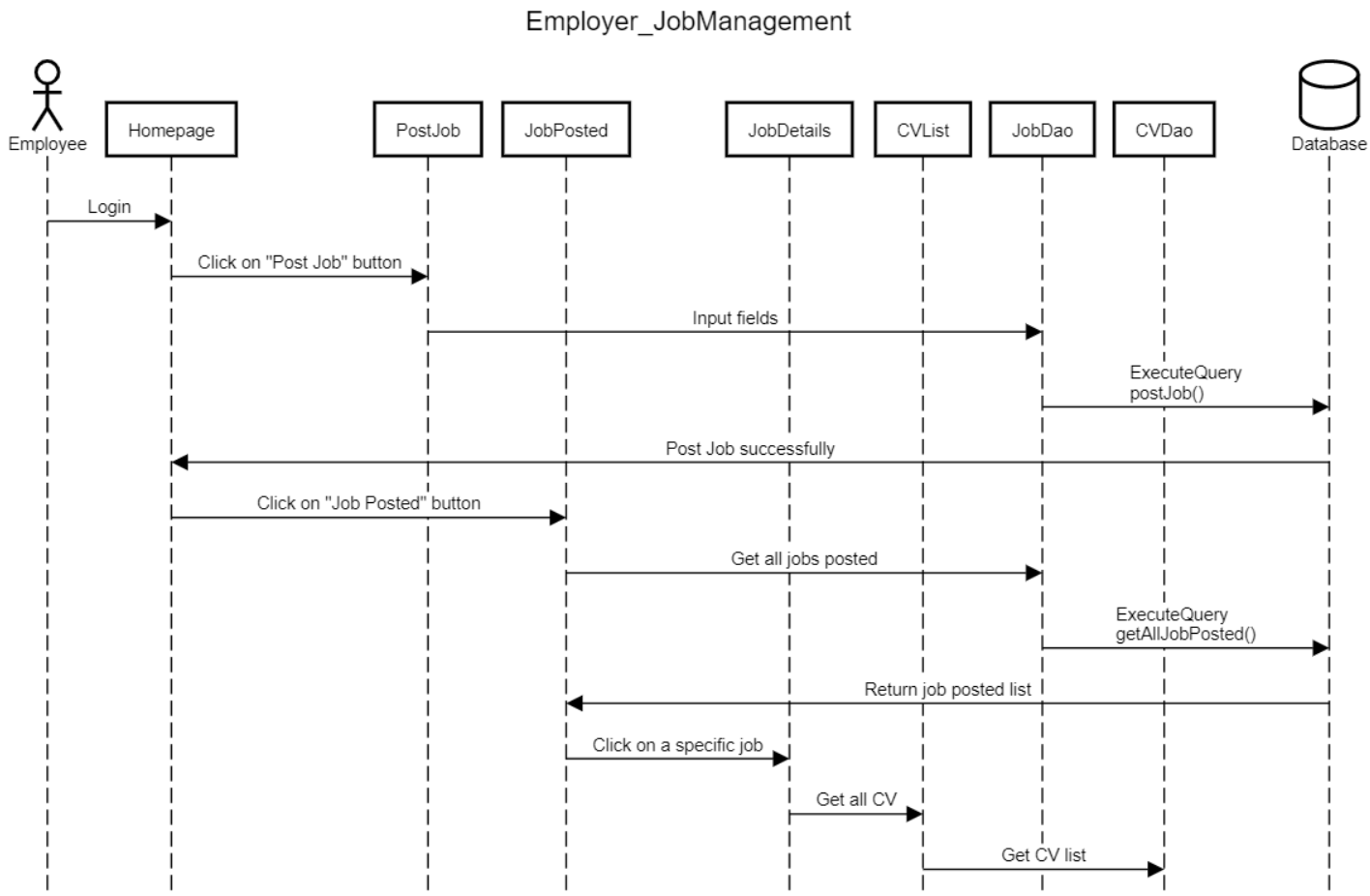
## 7. Job Management Employers

### 

### Class specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *PostJob* | *Employers can post job to find appropriate candidates for a position* |
| *02* | *JobPosted* | *Employers can see list job their posted and can view list CV applying for this job* |
| *03* | *CVList* | *Employers can view CVs which applying for a specific job* |

### Sequence diagram



### Database queries

01.JobPosted

-select \* from job join location on location.id= job.locationid join major on major.id=job.majorid join jobtype on jobtype.id= jobtypeid where username= ? order by job.id offset ? rows fetch next ? rows only"

02.PostJob

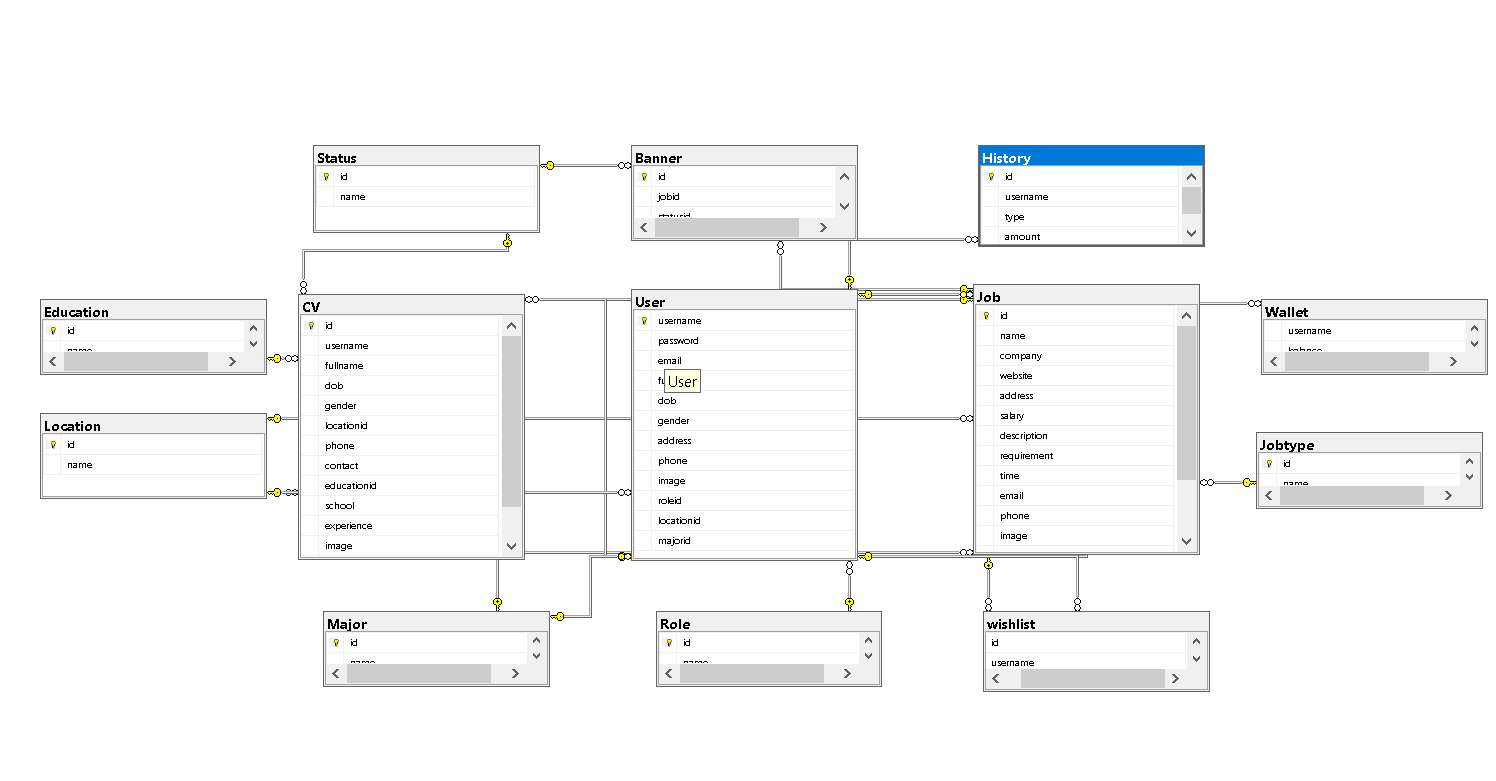
-insert into job(name,company, website,address,salary,description,requirement,[time],email,phone,image,username, locationid, majorid, jobtypeid)values(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)"

03.ListCV

-select \* from cv join status on status.id=statusid join education on educationid= education.id join job on job.id=jobid where jobid=? and statusid=1 order by cv.id offset ? rows fetch next ? rows only

## III. Database Design

## 1. Database Schema



## 2. Table Description

| **No** | **Table** | **Description** |
| --- | --- | --- |
| 01 | Banner | Store job as banner information  - Primary keys:   * id: int   - Foreign keys:   * jobID: int * statusID: int |
| 02 | CV | Store CV information  - Primary keys:   * id: int   - Foreign keys:   * locationID: int * educationID: int * statusID: int * jobID: int   - Columns:   * username: nvarchar(50) * fullname: nvarchar(50) * dob: date * gender: boolean * phone: nvarchar(50) * contact: nvarchar(50) * school: nvarchar(50) * experience: nvarchar(50) * image: nvarchar(50) |
| 03 | Education | Store education information  - Primary keys:   * id: int   - Columns:   * name: nvarchar(50) |
| 04 | History | Store history transaction information  - Primary keys:   * id   - Columns:   * username * type * amount |
| 05 | Job | Store job information  - Primary keys:   * id   - Foreign keys:   * locationID * majorID * jobtypeID   - Columns:   * name * company * website * address * salary * description * requirement * time * email * phone * image * username |
| 06 | Jobtype | Store job type information  - Primary keys:   * id: int   - Columns:   * name: nvarchar(50) |
| 07 | Location | Store location information  - Primary keys:   * id: int   - Columns:   * name: nvarchar(50) |
| 08 | Major | Store major information  - Primary keys:   * id: int   - Columns:   * name: nvarchar(50) |
| 09 | Role | Store role information: admin, employee, emoloyer  - Primary keys:   * id: int   - Columns:   * name: nvarchar(50) |
| 10 | Status | Store job type information  - Primary keys:   * id: int   - Columns:   * name: nvarchar(50) |
| 11 | User | Store user information  - Primary keys:   * username: nvarchar(50)   - Foreign keys:   * roleID: int * locationID: int * majorID: int   - Columns:   * password: nvarchar(50) * email: nvarchar(50) * fullname: nvarchar(50) * dob: date * gender: boolean * address: nvarchar(50) * phone: nvarchar(50) * image: nvarchar(50) |
| 12 | Wallet | Store job as banner information  - Columns:   * username: nvarchar(50) * balance: nvarchar(50) |
| 13 | Wishlist | Store job as banner information  - Primary keys:   * id: int   - Foreign keys:   * jobID: int   - Columns:   * username: nvarchar(50) |