

SW Engineering CSC648/848 Spring 2022

LinkedSF - Section 02 Team3

Milestone 2 (1 March 2022)

Christian Mcglothen (Team Lead) (cmcglothen@mail.sfsu.edu)

Jeff Lo (Back End Leader)

Briget Soriano (Front End Leader)

Dominique Dutton (Front End)

Richard Li (Github Specialist)

Justin Yee (Back End)

<i>Date Submitted</i>	<i>Date Revised</i>
<i>8 March 2022</i>	

1. Functional Requirements:

- **Student (Priority 1)**

- GUI:

- Create Registration Page for STUDENTS
 - Should Ask for Basic Info Such As (Username, Fname, Lname, Email, Password)
 - Once STUDENT is Logged In, GUI should Display a few job Posts.
 - Should have a search engine (Filter/Sort Job Positions)

- Functionality:

- Once a student has completed registration, they should be redirected to the student login page.
 - The completion of the registration form will populate the database with the following info (Username, Fname, Lname, Email, Password, Etc)
 - Students will be able to search/filter jobs based on tech area, job position, and skills (Parametric Search)
 - Jobseeker can easily view company information
 - Jobseeker can Update his/her profile information
 - Jobseeker can Apply for suitable job plan
 - Jobseeker can easily Cancel member registration
 - Jobseeker can View his/her Inbox
 - Jobseeker can easily Post comment on our site

- **Employer/Company (Priority 1)**

- GUI

- Will allow the Company to Sign-up/Register
 - Will contain fields such as (company name, email, username, etc)
 - Once the company has signed up, they will be redirected to the login page
 - Once they have completed login, they will be redirected to their homepage in which they will have the capability to

create job posts and possibly see the amount of traffic their post received

- Functionality:

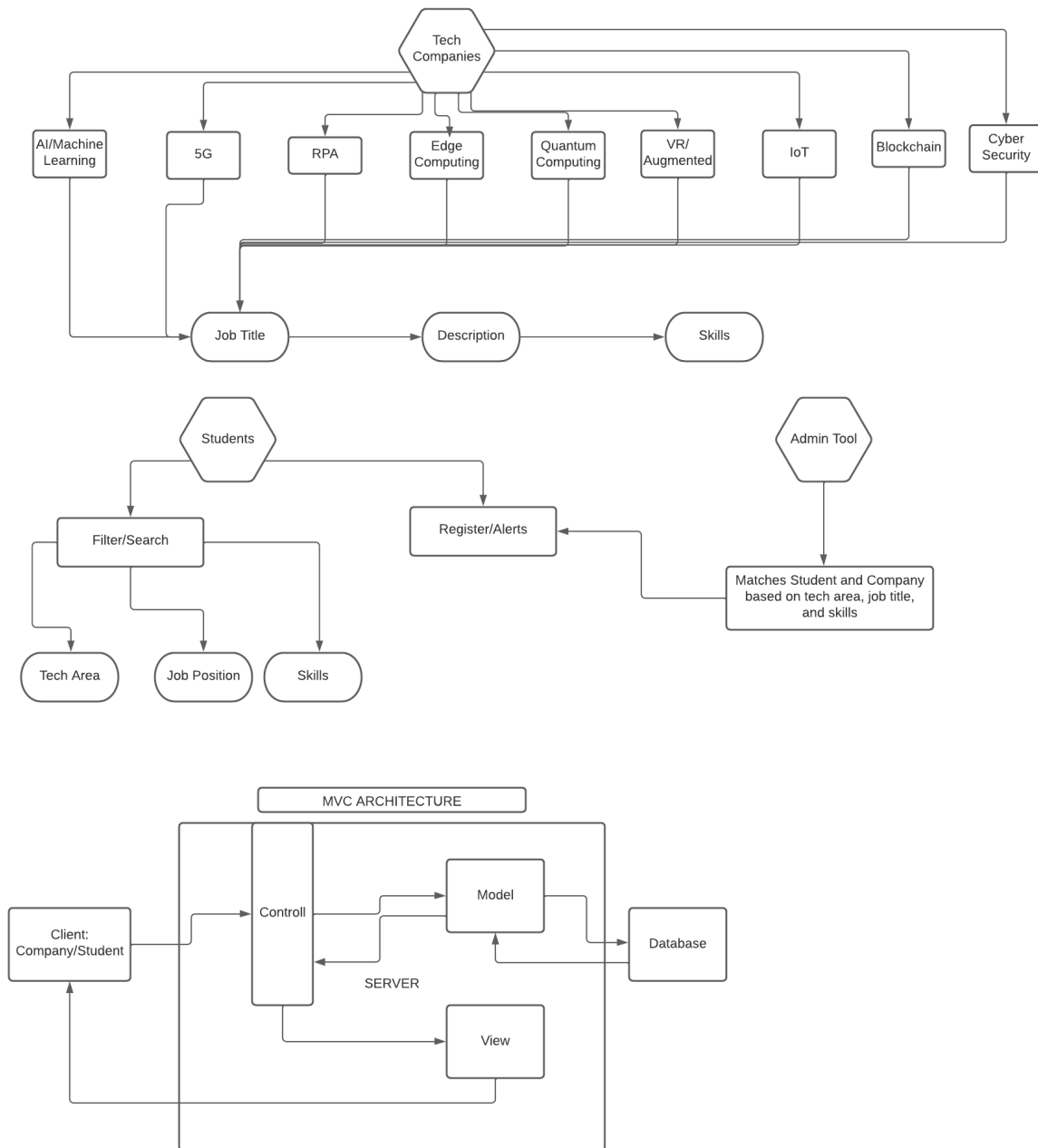
- Each company will be allowed to post jobs in 9 areas (AI/Machine Learning, 5G, RPA, Edge Computing, Quantum Computing, VR/Augmented, IoT, Blockchain, Cyber Security)
- Each field area will consist of 3 specifications (Job Title, Description, and Skills)
- Companies may also have the ability to filter their own posts
- Company can view Compose message by Jobseeker and Admin
- Company can View Inbox

- **Admin Tool (Priority 1/ Some Specs Have Priority 2 and 3)**

- Functionality:

- When a student and company have similar interests, both the company and the student will be alerted/sent a notification
- Manage student searches and companies ability to post jobs
- Admin can Manage a Package or plan for Jobseeker
- Admin can view Register member and Manage Register Member in our site
- Admin Can View Jobseeker Details
- Admin Manage Company Details
- Admin can View Job alert
- Admin can Manage Compose message Details
- Admin can View Inbox
- Admin can view post comment
- Admin can View Feedback

Rough Draft Of Functional Requirements:



2. UI Mockups and Storyboards:

Student Persona GUI:

Student Register Page

Back button

Student Register Page

Logo

Username
First Name + Last Name
Email
Confirm Password/
Password
Submit

Link for either employer/
job seeker login page



- Student must enter the following information: username, first name, last name, email, and password
 - Both username and email must be unique: if student enters an associated email or username, a flash message will appear alerting them the username/email is already taken

- Email should contain an “@” symbol
- Password should be minimum 8 characters long
- The student will be asked to confirm password before added to the database
- Back button will take student back to the landing page
- Once the student has successfully completed registration and has selected the submit button, all of the asked fields will be properly stored within the student table
- Student will be redirected to student login page

Student Login Page

Back button

Employer or Student Login Page

Logo

Company Name/
Username

Password

Submit

Link for either employer/
job seeker signup page



- Mockup for either employer login page or student login page
- There will be separate login pages for either student and employer to properly match information to either student or employer table in database
 - If student has been redirected from student registration page, the login page will be for student login
- Bottom link will redirect to either employer or student sign up page
- The entered information will be sent to the backend to check for the specified user
 - If the password does not match to the entered account, a flash message will appear alerting the student the password is incorrect
 - If the username does not match an entry in the student table, a flash message will appear alerting the student no user exists
- Once student has successfully logged in, they will be redirected to the student home page

Student Home Page

Title/Logo

Student Page

Logout/Inbox/
Profile

Job Search

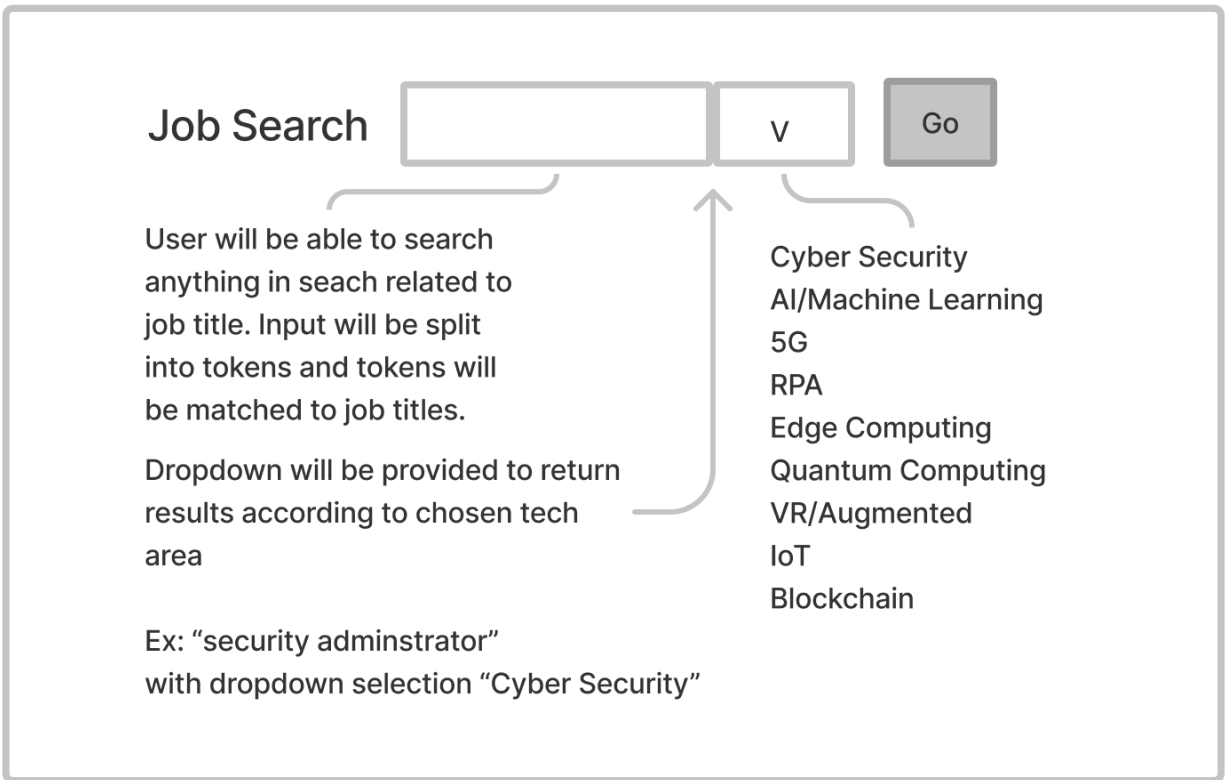
Filter Dropdown

Job
posting

Job
posting

Job
posting

Footer



- Once student has been authenticated and authorized to access student page, job search bar and job posting will appear in the main content area
 - Job postings will appear based on newly posted
 - At max 8 postings will appear only showing 3/4 at a time
 - Arrow options on the sides of the job postings content will appear allowing the user to traverse back and forth between the 8 postings
- Navbar will display redirects to logout, view student profile, or view inbox
- When student has completed job search, they will be redirected to the job postings page
 - Job search will allow student to search filter by tech area

Job Search Results

Title/Logo

Job Posting Page

Logout/Inbox/
Profile or
Login/Signup

Job Posting

Job Posting

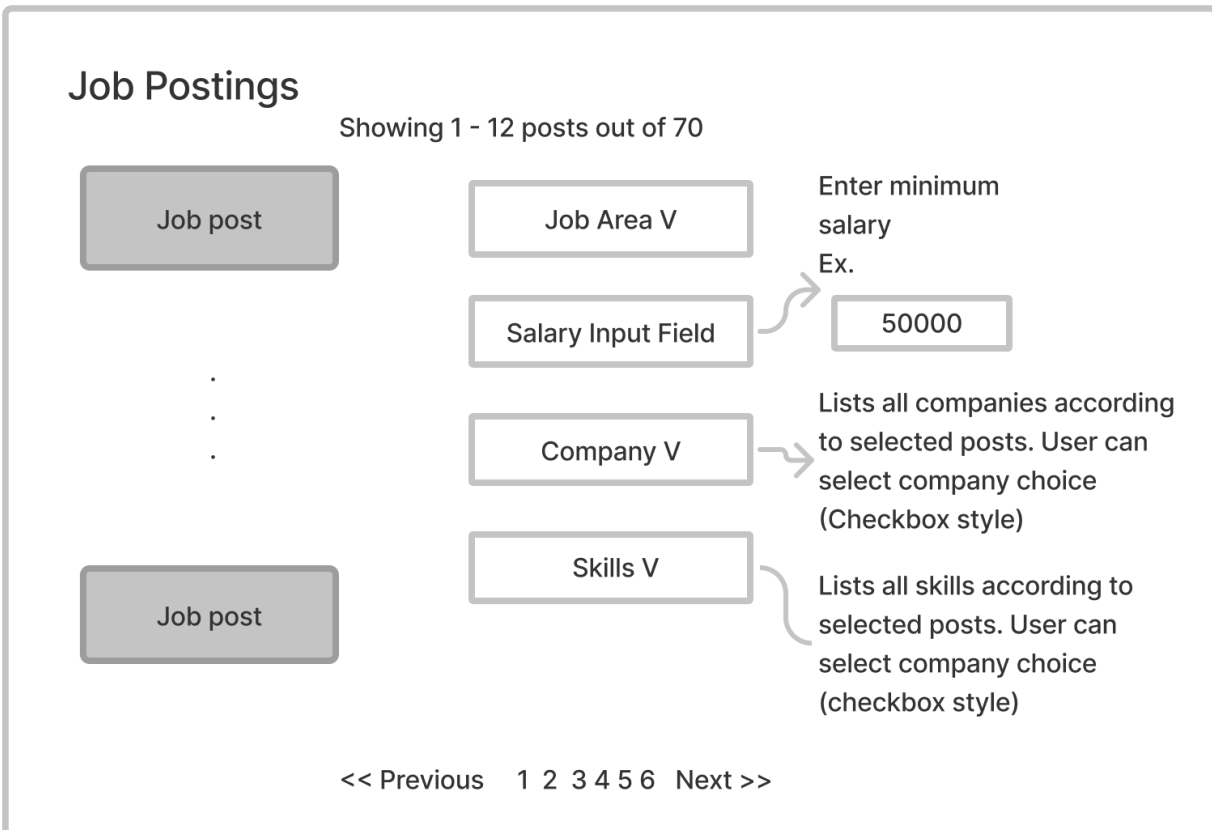
Job Posting

Filter options

Dropdowns

Checkboxes

Footer



- The main content area will display the job postings according the the search
 - To the right, a filter area will appear where the student can filter according to dropdown and checkbox content
 - At most 12 posting will appear in a horizontal rows spanning to $\frac{2}{3}$'s of the page
 - Each time the student selects/deselects a filter option, the page will automatically reload reflecting the updated filter change
 - The filter section will remain sticky to the page so as the user scrolls through job postings, the user will always have the option to select/deselect filter choices
- When the user has scrolled through all 12 listings, user will reach end of the page to give the user th choice to go to the next page
- Filters will allow students to filter according to job area, job skill, job salary, and company
 - Salary will be an input field where the student can enter the minimum salary

- Both company and skills can have more than one selection for their dropdown options

Employer Persona GUI:

Employer Register Page

Back button

Employer Register Page

Logo

Company Name

Username

Email

Confirm Password/
Password

Submit

Link for either employer/
job seeker login page



- Employer must enter the following information: username, company, email, and password
 - Company, username and email must be unique: if employer enters an associated company, email, or username, a flash

message will appear alerting them the
username/email/company is already taken

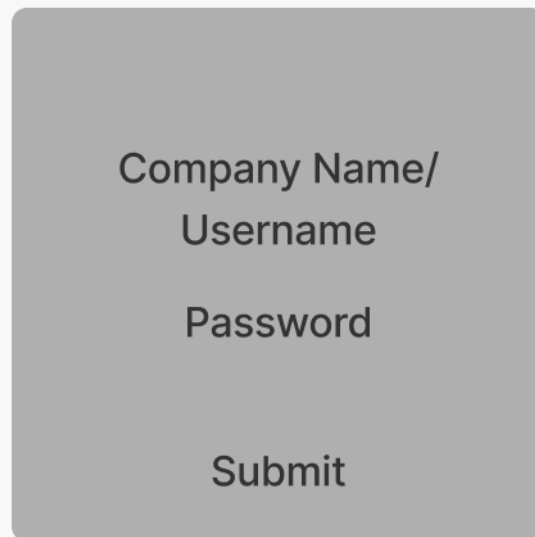
- Email should contain an "@" symbol
- Password should be minimum 8 characters long
- The employer will be asked to confirm password before added to the database
- Back button will take employer back to the landing page
- Once the employer has successfully completed registration and has selected the submit button, all of the asked fields will be properly stored within the company table
- Employer will be redirected to company login page

Employer Login Page

Back button

Employer or Student Login Page

Logo

A gray rectangular box representing a login form. It contains the following text elements from top to bottom: 'Company Name/' followed by 'Username' on the next line, 'Password' on the next line, and 'Submit' at the bottom.

Company Name/
Username
Password
Submit

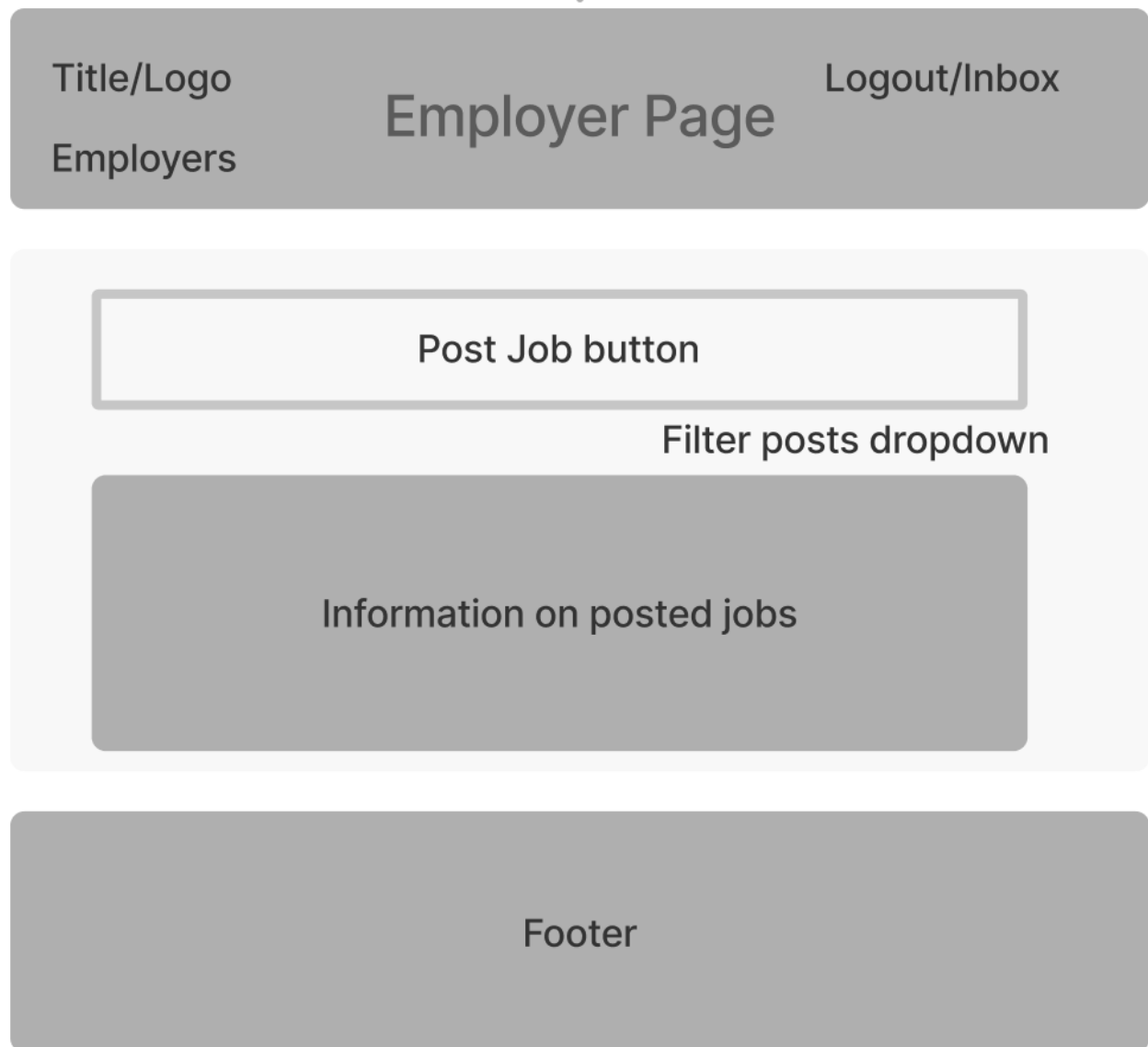
Link for either employer/
job seeker signup page

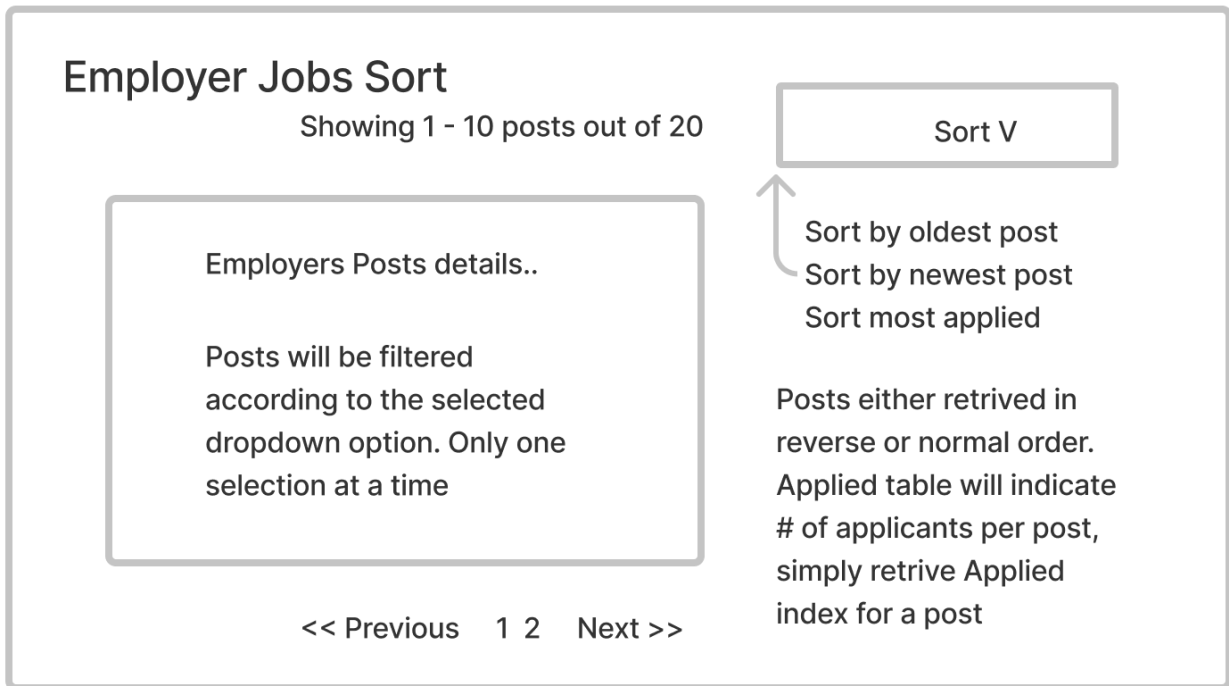


- Mockup for either employer login page or student login page
- There will be separate login pages for either student and employer to properly match information to either student or employer table in database
 - If employer has been redirected from employer registration page, the login page will be for employer login
- Bottom link will redirect to either employer or student sign up page
- The entered information will be sent to the backend to check for the specified company user

- If the password does not match to the entered account, a flash message will appear alerting the employer the password is incorrect
- If the username does not match an entry in the student table, a flash message will appear alerting the employer no user exists
- Once student has successfully logged in, they will be redirected to the employer home page

Employer Home Page





- Once employer has been authenticated and authorized to access employer page, job post button and username affiliated job postings will appear in the main content area
 - Posted job posting will give a brief description of what is posted
 - Job title will be displayed
 - Company name will be displayed
 - Posted date will be displayed
 - Number of applicants will be displayed
 - Edit button will be displayed allowing the employer to edit the job post
 - Remove button will be displayed allowing the employer to remove listing
 - “Post Job” button will redirect employer to post job page where the employer will need to fill out the required information:
 - Company name, job title, description, wages, city, state, skills
 - A dropdown will appear requiring employer to select one of the 9 tech areas
 - Hashtags will be implemented to narrow down skills
- Employer will be able to sort the posts in the following manner:

- Sort by descending/ascending date posted
- Sort by most applied
- At most 10 posted jobs will display, in the area before job listing is displayed, the amount of posted jobs will be displayed
 - Ex: "20 total posted jobs"
- If the amount of posts exceeds 12, when the employer reaches the twelve post, page numbers will be displayed in accordance to how many total posts, allowing the employer to traverse through job postings through page numbers
- Navbar will display redirects to logout, view employer profile, or view inbox

Student Use Case #1:

- John is in his final semester at SFSU. He wants to find a job that will utilize his knowledge on Artificial Intelligence. He registers an account with LinkedSF and browses through the job posting page. He filters the search results to limit to Artificial Intelligence roles.

Employer Use Case #1:

- Jane's company in Cyber Security is getting overwhelmed and is looking to hire newly graduated students. Jane registers with LinkedSF and waits to be authenticated. Afterwards, she is able to post job listings with specific information for students to see.

3. High level Architecture, Database Organization:

Database Organization:

- Student/Applicant/Jobseeker Table:
 - Primary Key UID/SID should be unique to each student/jobseeker with no redundancy. UID/SID should be of type Integer
 - Username should be unique to each student/jobseeker with no redundancy. Username should be of type VarChar (45)
 - First Name should be of type VARCHAR(45) and Non-Null to ensure the field has a value
 - Middle Name (Optional)
 - Last Name Should be of type VARCHAR(45) and Non-Null to ensure the field has a value
 - Email should be unique to each student/jobseeker with no redundancy. Email Should be of type VarChar(128) and Non-Null to ensure the field has value
 - Password should be VarChar(128) and Non-Null to ensure the field has value
 - Created-Date should be of type DATETIME and Non-Null to insure the field has value
 - Resume Should be of type BLOB which holds a PDF file, this field should also be Non-Null to ensure that a resume is uploaded.
- Company Table:
 - Primary Key CID should be unique to each Company with no redundancy. CID should be of type Integer
 - Username should be unique to each Company with no redundancy. Username should be of type VarChar (45) and Non-Null to ensure the field has value

- Email should be unique to each Company with no redundancy. Email Should be of type VarChar(128) and Non-Null to ensure the field has value
 - Password should be VarChar(128) and Non-Null to ensure that the field has value
 - Created-Date should be of type DATETIME
- Post Table:
 - Primary Key PID should be unique to each Post with no redundancy. PID should be of type Integer
 - Foreign Key CID should be unique to each Company with no redundancy. CID should be of type Integer derived from Company Table
 - Company name should be of type VARCHAR(45)
 - JobTitle should be of type VarChar(128) and Non-Null
 - Description should be of type VARCHAR(4096)
 - Skills Should be of type VARCHAR(4096)
 - Wages Should be of type VARCHAR(128)
 - City Should be of type VARCHAR(128)
 - State Should be of type Text VARCHAR(128)
 - Job_Field should be of type VARCHAR(128)

MySQL Database and Commands:

```
CREATE DATABASE IF NOT EXISTS LinkedSF;
```

```
USE LinkedSF;
```

```
CREATE TABLE IF NOT EXISTS `LinkedSF`.`Company` (
  `idCompany` INT NOT NULL,
  `Company_Name` VARCHAR(45) NOT NULL,
```

```

`Company_Username` VARCHAR(45) NOT NULL,
`Company_Email` VARCHAR(128) NOT NULL,
`Password` VARCHAR(128) NOT NULL,
`Created` DATETIME NOT NULL,
PRIMARY KEY (`idCompany`),
UNIQUE INDEX `idCompany_UNIQUE` (`idCompany` ASC) VISIBLE,
UNIQUE INDEX `Company_Name_UNIQUE` (`Company_Name` ASC)
VISIBLE,
UNIQUE INDEX `Company_Username_UNIQUE` (`Company_Username`
ASC) VISIBLE,
UNIQUE INDEX `Company_Email_UNIQUE` (`Company_Email` ASC)
VISIBLE)
ENGINE = InnoDB;

```

```

CREATE TABLE IF NOT EXISTS `LinkedSF`.`JobPost` (
  `idJobPost` INT NOT NULL,
  `Job_Title` VARCHAR(128) NOT NULL,
  `Job_Description` VARCHAR(4096) NOT NULL,
  `Job_Skills` VARCHAR(4096) NOT NULL,
  `Job_Pay` VARCHAR(128) NOT NULL,
  `Job_Street_Address` VARCHAR(128) NOT NULL,
  `Job_City` VARCHAR(128) NOT NULL,
  `Job_State` VARCHAR(128) NOT NULL,
  `FK_Companyid` INT NOT NULL,
  `Job_Field` VARCHAR(128) NOT NULL,
  PRIMARY KEY (`idJobPost`),
  UNIQUE INDEX `idJobPost_UNIQUE` (`idJobPost` ASC) VISIBLE,
  INDEX `JobPost to Company_idx` (`FK_Companyid` ASC) VISIBLE,
  CONSTRAINT `JobPost to Company`
    FOREIGN KEY (`FK_Companyid`)
    REFERENCES `LinkedSF`.`Company` (`idCompany`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;

```

```

CREATE TABLE IF NOT EXISTS `LinkedSF`.`JobSeeker` (
  `idJobSeeker` INT NOT NULL,
  `JS_Username` VARCHAR(45) NOT NULL,
  `First_Name` VARCHAR(45) NOT NULL,
  `Last_Name` VARCHAR(45) NOT NULL,
  `Email` VARCHAR(128) NOT NULL,
  `Password` VARCHAR(128) NOT NULL,
  `Resume` BLOB NOT NULL,
  `Created` DATETIME NOT NULL,
  PRIMARY KEY (`idJobSeeker`),
  UNIQUE INDEX `JS_Username_UNIQUE` (`JS_Username` ASC)
  VISIBLE,
  UNIQUE INDEX `idJobSeeker_UNIQUE` (`idJobSeeker` ASC) VISIBLE,
  UNIQUE INDEX `Email_UNIQUE` (`Email` ASC) VISIBLE)
ENGINE = InnoDB;

```

```

CREATE TABLE IF NOT EXISTS `LinkedSF`.`Applied` (
  `idApplied` INT NOT NULL,
  `Applied` DATETIME NOT NULL,
  `Fk_Postid` INT NOT NULL,
  `Fk_JobSeekerid` INT NOT NULL,
  UNIQUE INDEX `idApplied_UNIQUE` (`idApplied` ASC) VISIBLE,
  PRIMARY KEY (`idApplied`),
  INDEX `PostLink_idx` (`Fk_Postid` ASC) VISIBLE,
  INDEX `JobseekerLink_idx` (`Fk_JobSeekerid` ASC) VISIBLE,
  CONSTRAINT `PostLink`
    FOREIGN KEY (`Fk_Postid`)
      REFERENCES `LinkedSF`.`JobPost` (`idJobPost`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  CONSTRAINT `JobseekerLink`
    FOREIGN KEY (`Fk_JobSeekerid`)
      REFERENCES `LinkedSF`.`JobSeeker` (`idJobSeeker`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)

```

ENGINE = InnoDB;

Media Storage:

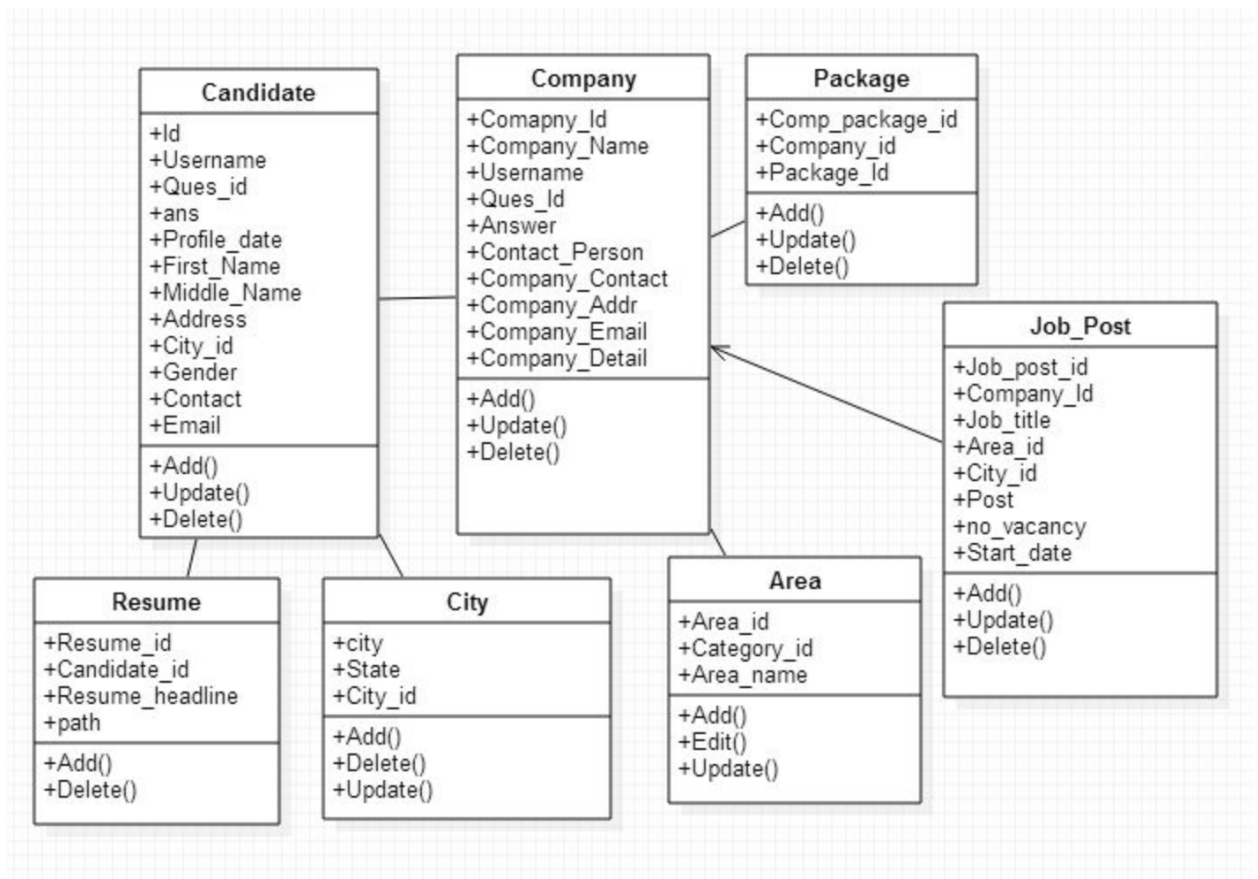
- BLOB:
 - The only BLOB that will be used is located in the Jobseekers table to store each Job Seekers resume.

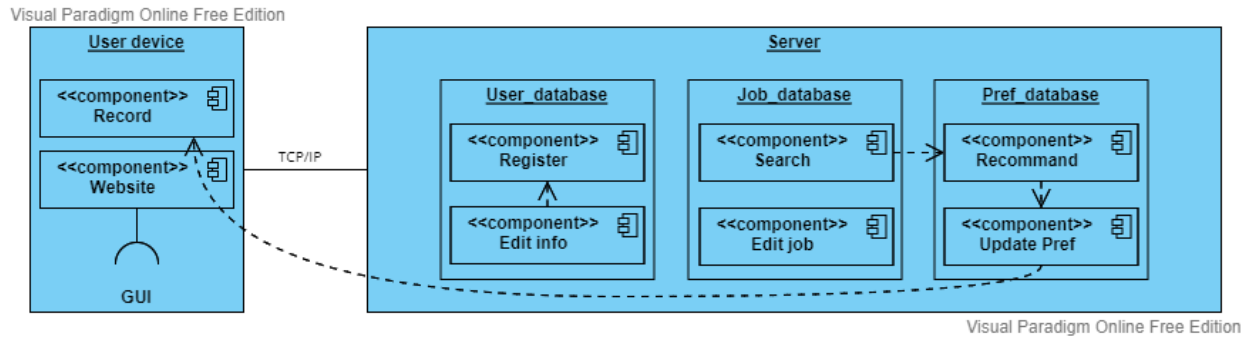
Search/filter architecture and implementation:

- Parametric Search:
 - Jobseeker/Student will be given a Checklist for 9 job areas (AI/Machine Learning, 5G, RPA, Edge Computing, Quantum Computing, VR/Augmented, IoT, Blockchain, Cyber Security)
 - They will be able to select from a drop down state in which they would like to work (i.e when a student select california, only jobs in california will appear in GUI)
 - In the search bar, they will be allowed to type anything related to job title and their input will be split into tokens. If any of these tokens match a part of the job title, the job post will be displayed in the GUI (Using %Like)

4. High level Architecture, Database Organization:

High Level UML Diagrams:





5. Identifying key risks for our project:

Skills Risks: We have a team that consists of different skill sets and familiarity. Additional learning will be needed to ensure that tasks are completed accurately.

Schedule Risks: Prioritizing time to meet every week to ensure we are able to stay on schedule to complete the project. Everyone has other classes, jobs, and personal life that can affect our ability to all meet together at the same time.

Technical Risks: One key risk is trying to determine how to view the resume which is of type BLOB and converting it to be viewed by the Company and resume holder.

Teamwork Risks: Communication between team members and ensuring equal distribution of assigned tasks.

Legal/Content Risks: Ensuring that job postings comply with local and state law, which can be mitigated through admin approval for posts.

6. Project Management:

Starting from milestone 0, we have divided the work between frontend and backend developers. As we continue into milestone 2, we take into account the number of steps in each milestone and categorize them based on frontend or backend development. As a team, we have established meeting times (Friday-Sunday 2:00pm) and make sure that all work is divided evenly between groups. If one group is having more difficulty than the other, we all come together to find a solution and return to respective tasks. We manage these tasks in discord by creating new channels for each milestone and setting goals for completion along with regular zoom sessions. As a group we have realized that no particular person is an expert in any one area and we all have to do research in order to complete tasks.