CHAPTER III
RESEARCH METHODS

3.1 Research Design

Research design that is used to developed backend of a web based job seeking using Laravel framework with SAW method for employee ranking in Japan is applied research where the research can be used directly. Applied research strives to apply result to the specific problem that a company is facing.

The backend development process of job seeking using SAW in Japan will be explained by the flowchart shown in Fig. 3.1.

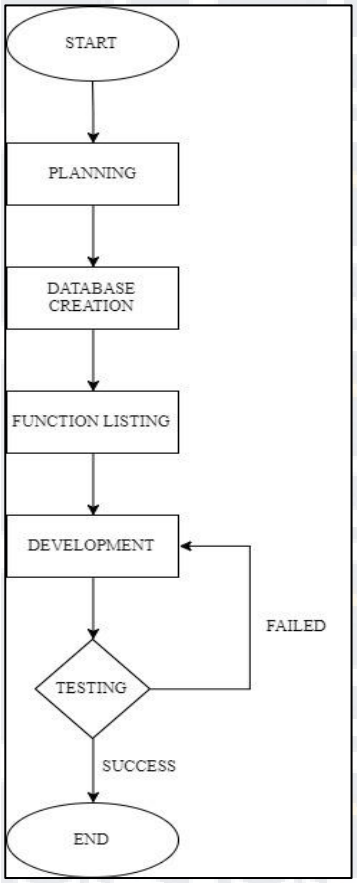

Figure 3.1 *Flowchart of Backend Development Process*

The development process started off with planning, which we planned to make a job seeking website to help candidates and companies to find jobs and offer jobs. During the planning phase, we think of many things that are needed for either candidates or companies so that they will have a convenient and flexible website to use to search and offer jobs. After all the planning process, we proceed to database creation, which we create tables that are needed for the system to store data and we also create the relation between those tables that are created.

Then we proceed to function listing phase where we list down all the functions that are needed for the system to work such as login, create, update, read, delete, SAW. After that we proceed to the development phase, where the codes and the function that are listed will be developed in this phase. We are using sublime as the IDE to create the project and we also use Virtual Machine for our database so that the database will always be synchronize between developers. After developing the function we will test it, if the function failed to work properly it will be redeveloped and will be tested again until it will work properly.

3.2 Unified Modeling Language (UML)

3.2.1 Use Case Diagram

The use case diagram of the online job seeking backend system is explained in Fig. 3.2 as shown below:

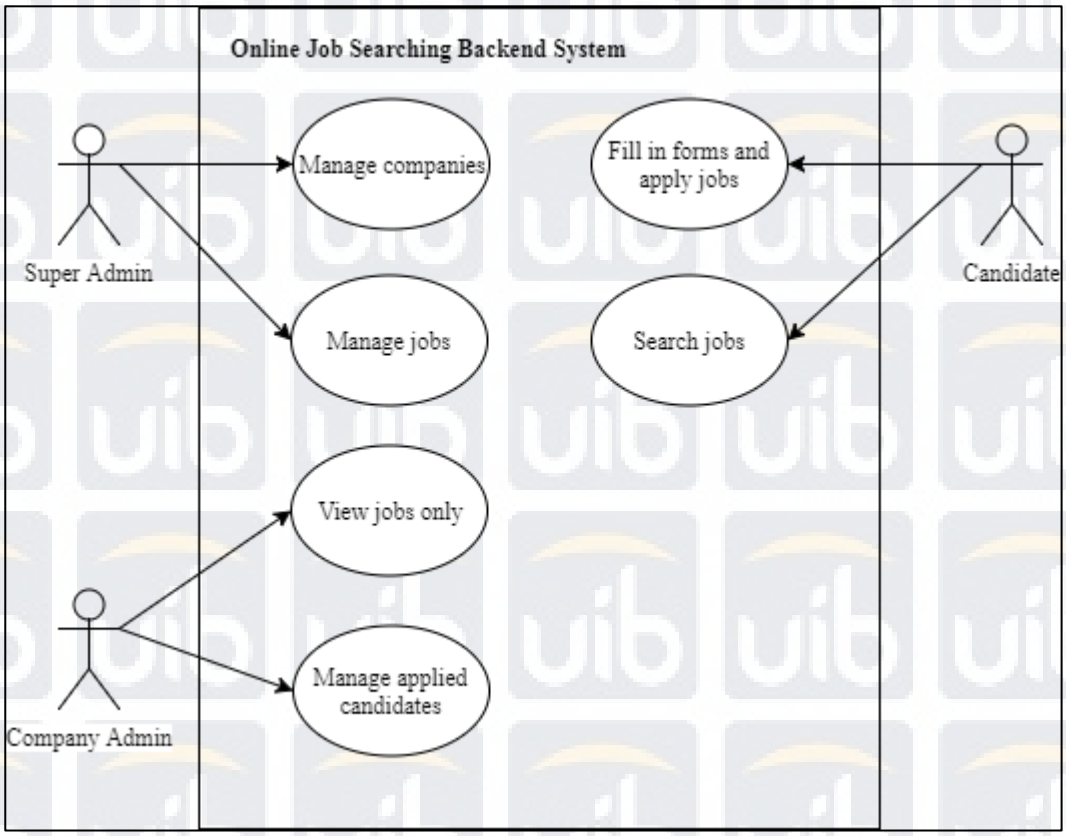

Figure 3.2 *Use Case Diagram of Backend Job Seeking System*

There are three roles that will be used in this system which are super admin, candidate and company admin. Super admin is has the highest privilege in this system which it can manage the companies which means create, read, update and delete companies data and also manage jobs which also means create, read, update and delete the jobs which it receive from the companies.

For the company admin role, it has the privilege to view jobs that are created by super admin and manage applied candidates which means it has the privilege to see who are those candidates that apply for the jobs given and it also

has the privilege to mark as seen which means if the candidates curriculum vitae has been reviewed by the companies, they can mark it as seen.

Lastly is the candidate role, which it has privilege fill in the forms to apply for jobs and will be reviewed by the companies and the privilege to search for jobs that are posted by super admin.

3.2.2 Sequence Diagram

1. Login

Login sequence diagram can be seen in Fig. 3.3. There will be action that is invoked from the frontend, and it will enter to user controller, then login() function will be called and json will be returned back to the frontend.

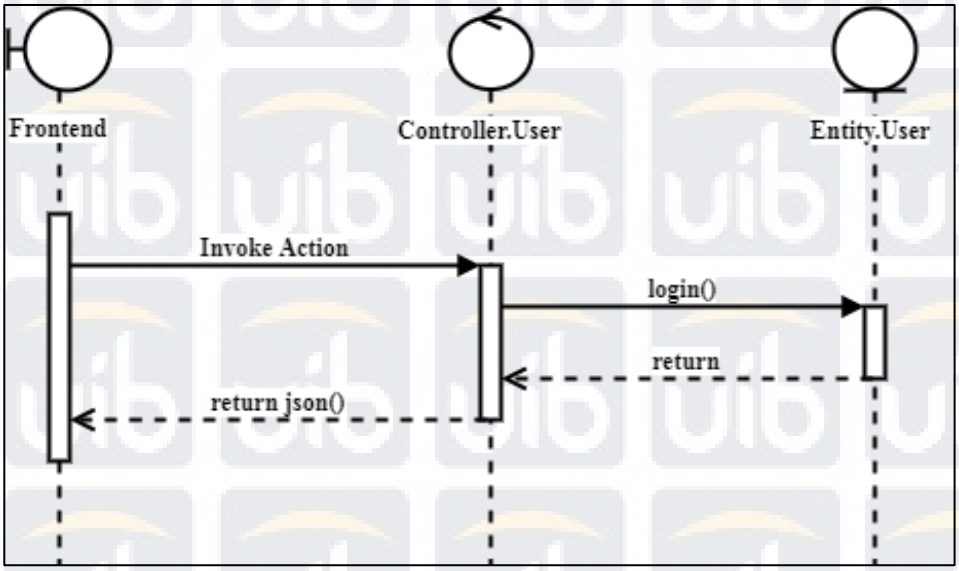

Figure 3.3 *Login Sequence Diagram*

2. Create Company

Create company sequence diagram can be seen in Fig. 3.4. There will be action that is invoked from the frontend, and it will enter to company controller, then saveCompany() function will be called and json will be returned back to the frontend.

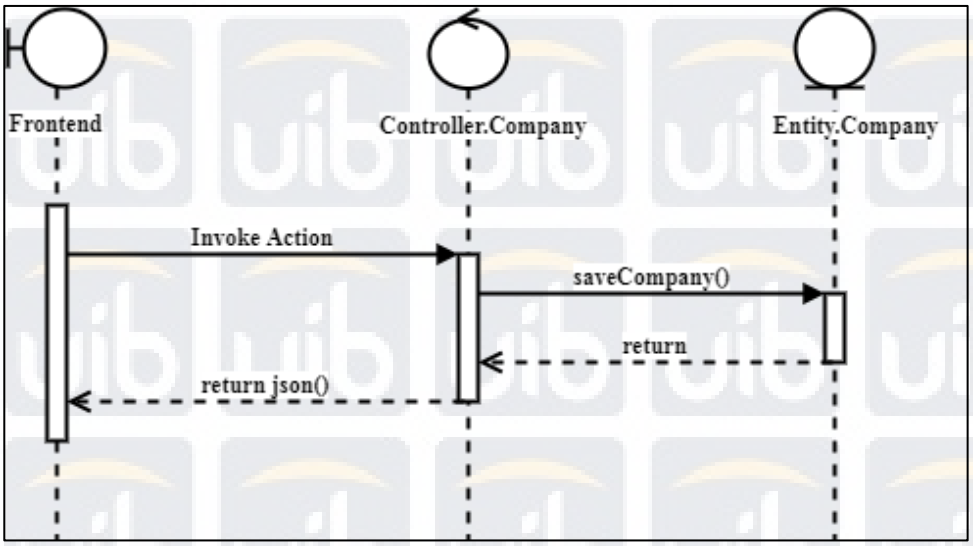

Figure 3.4 *Create Company Sequence Diagram*

3. Update Company

Update company sequence diagram can be seen in Fig. 3.5. There will be action that is invoked from the frontend, and it will enter to company controller, then updateCompanyDetail() function will be called and json will be returned back to the frontend.

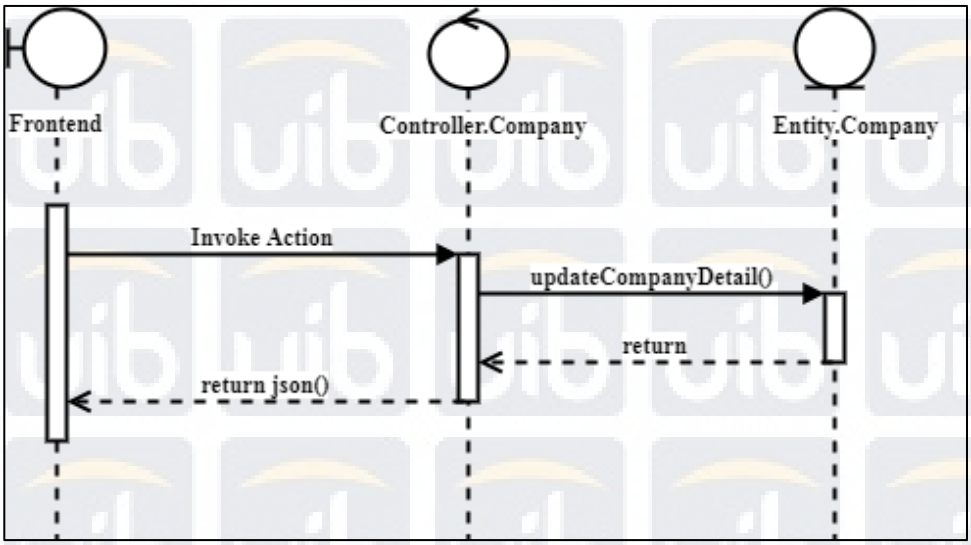

Figure 3.5 *Update Company Sequence Diagram*

4. Read Company

Read company sequence diagram can be seen in Fig. 3.6. There will be action that is invoked from the frontend, and it will enter to company controller, then getCompanyDetail() function will be called and json will be returned back to the frontend.

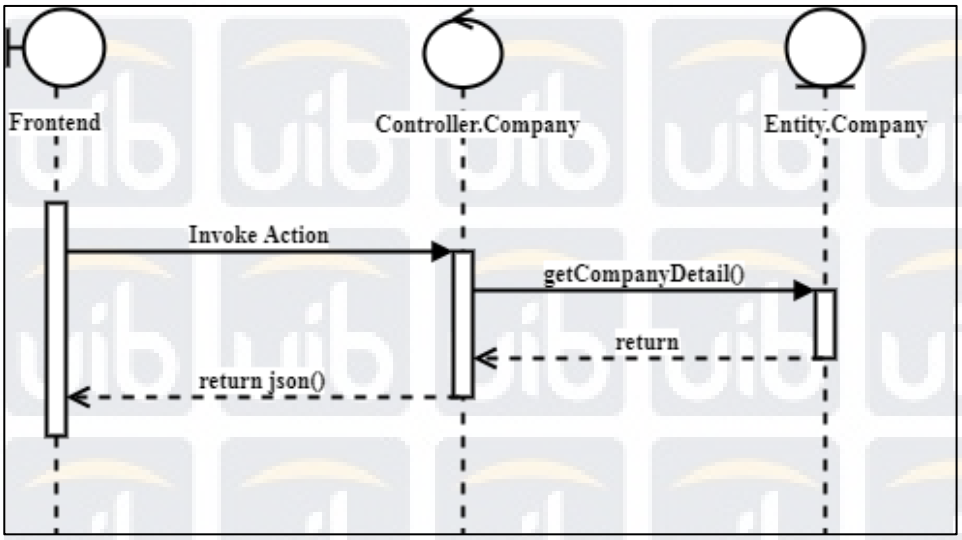

Figure 3.6 *Read Company Sequence Diagram*

5. Get Company List

Get company list sequence diagram can be seen in Fig. 3.7. There will be action that is invoked from the frontend, and it will enter to company controller, then getCompanyList() function will be called and json will be returned back to the frontend.

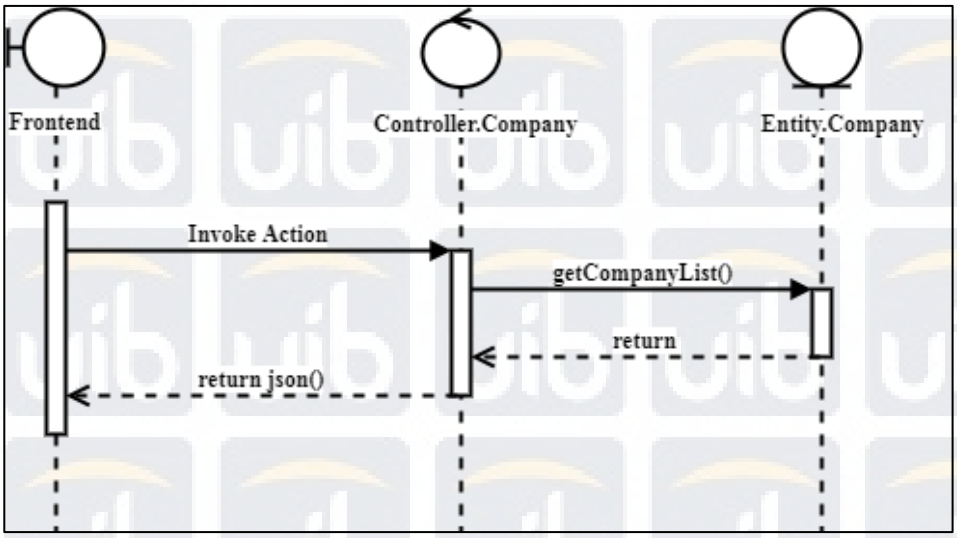

Figure 3.7 *Get Company List Sequence Diagram*

6. Create Job

Create job sequence diagram can be seen in Fig. 3.8. There will be action that is invoked from the frontend, and it will enter to job controller, then saveJob() function will be called and json will be returned back to the frontend.

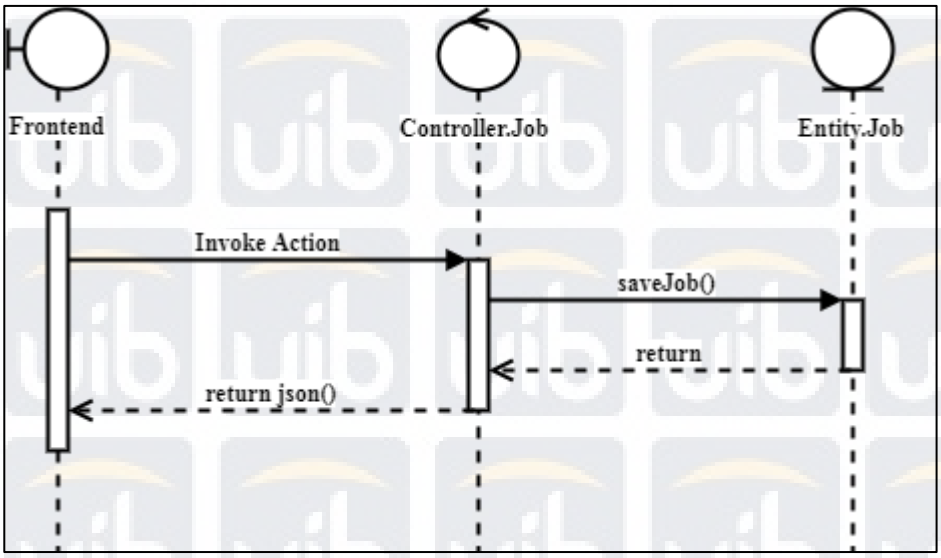

Figure 3.8 *Create Job Sequence Diagram*

7. Read Job

Read job sequence diagram can be seen in Fig. 3.9. There will be action that is invoked from the frontend, and it will enter to job controller, then getJobDetail() function will be called and json will be returned back to the frontend.

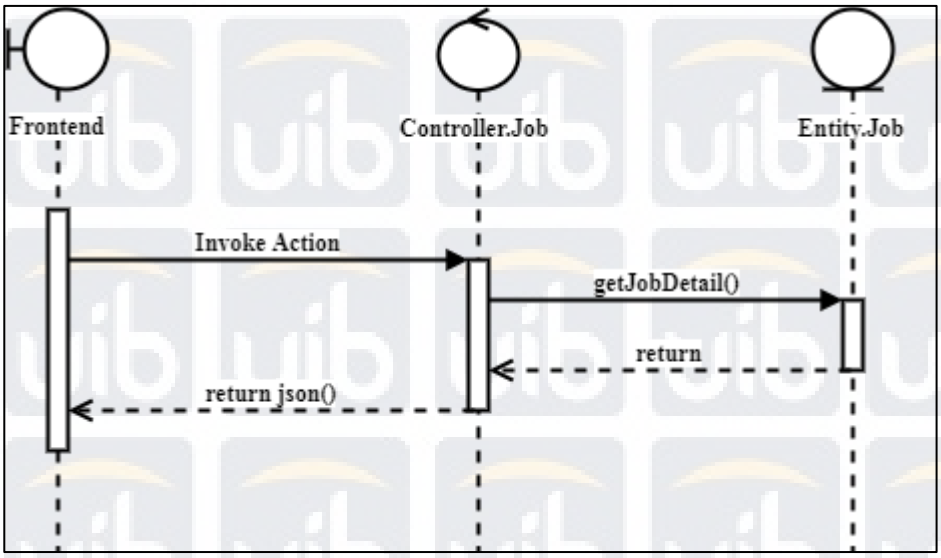

Figure 3.9 *Read Job Sequence Diagram*

8. Get Job List

Get job list sequence diagram can be seen in Fig. 3.10. There will be action that is invoked from the frontend, and it will enter to job controller, then getJob() function will be called and json will be returned back to the frontend.

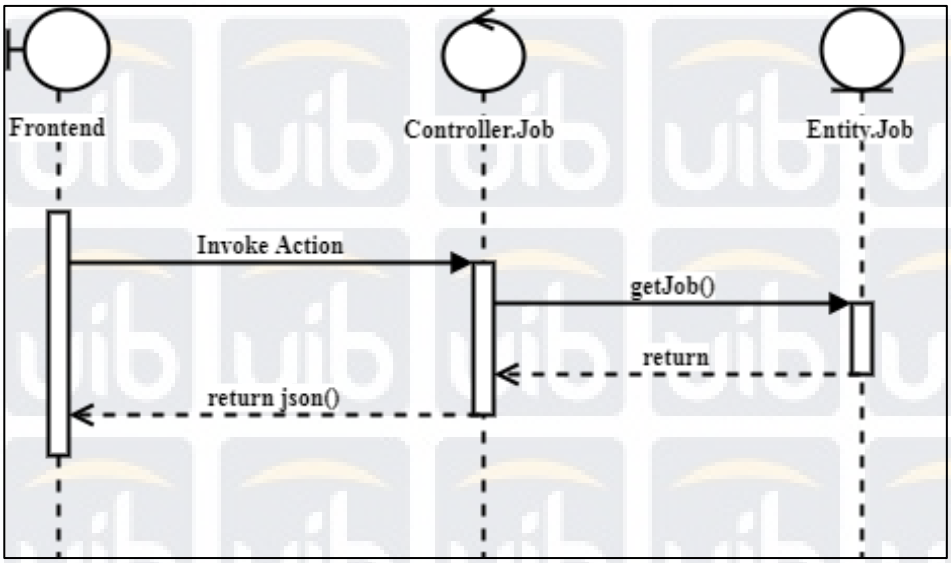

Figure 3.10 *Get Job List Sequence Diagran*

9. Get Prefecture List

Get prefecture list sequence diagram can be seen in Fig. 3.11. There will be action that is invoked from the frontend, and it will enter to prefecture controller, then getPrefectureList() function will be called and json will be returned back to the frontend.

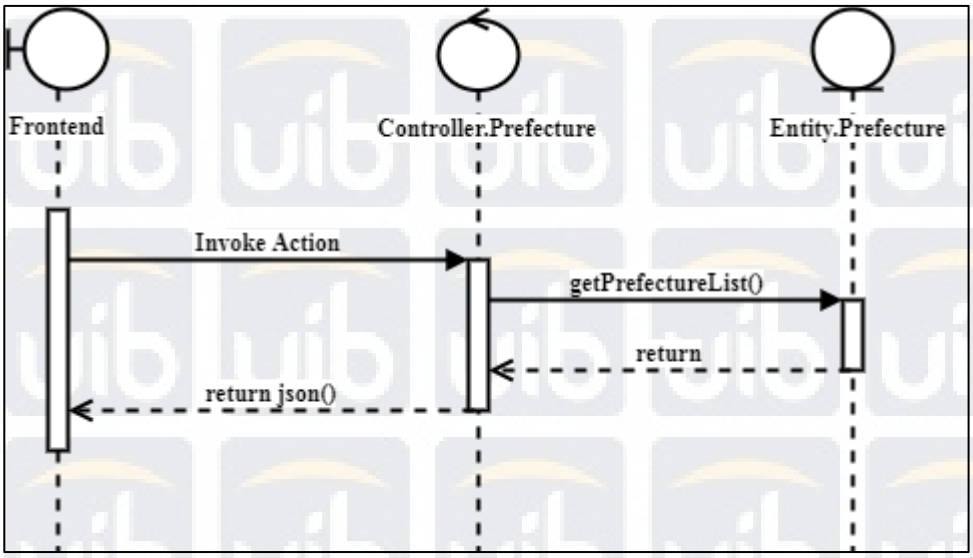

Figure 3.11 *Get Prefecture List Sequence Diagram*

10. Get City List

Get city list sequence diagram can be seen in Fig.3.12. There will be action that is invoked from the frontend, and it will enter to prefecture controller, then getPrefectureList() function will be called and json will be returned back to the frontend.

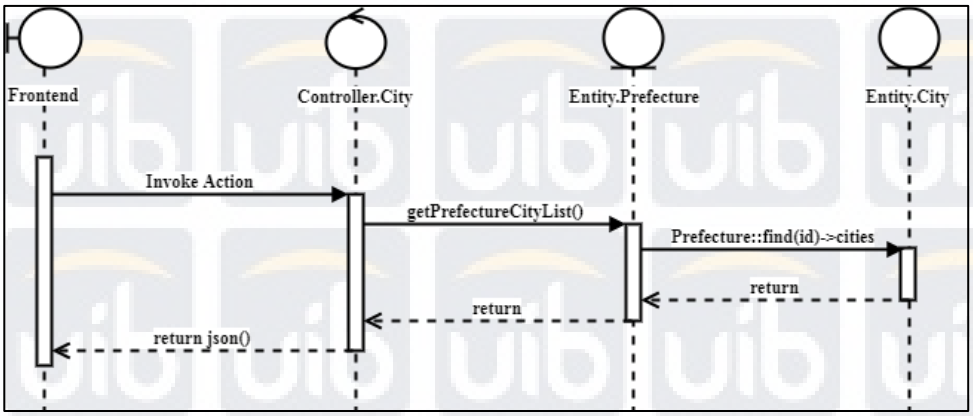

Figure 3.12 *Get City List Sequence Diagram*

11. Get SAW Application List

Get SAW application list sequence diagram can be seen in Fig.3.13. There will be action that is invoked from the frontend, and it will enter to applicant controller, then getSAWApplicantList() function will be called and json will be returned back to the frontend.

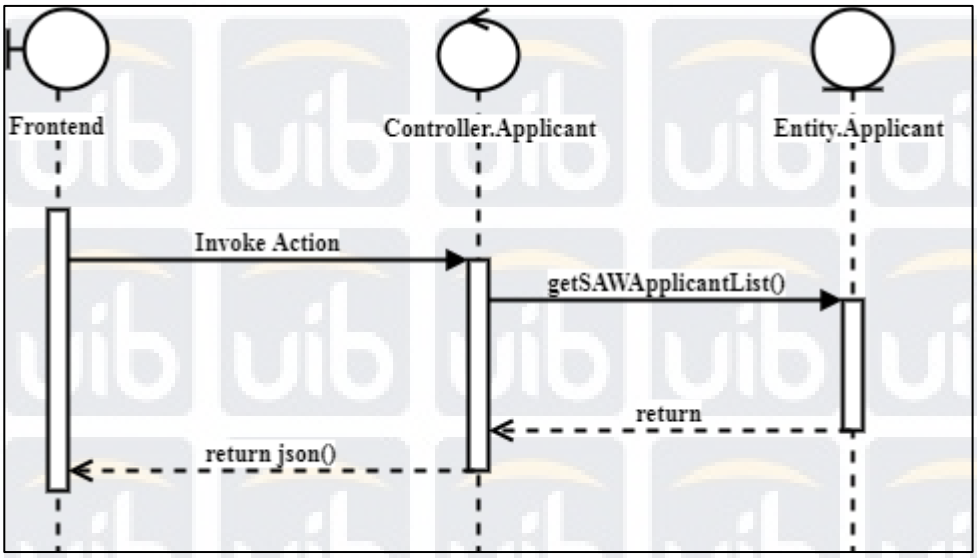

Figure 3.13 *Get SAW Application List Sequence Diagram*

3.3 Entity Relationship Diagram

The entity relationship diagram of the online job seeking backend system is explained in Fig. 3.14 as shown below:

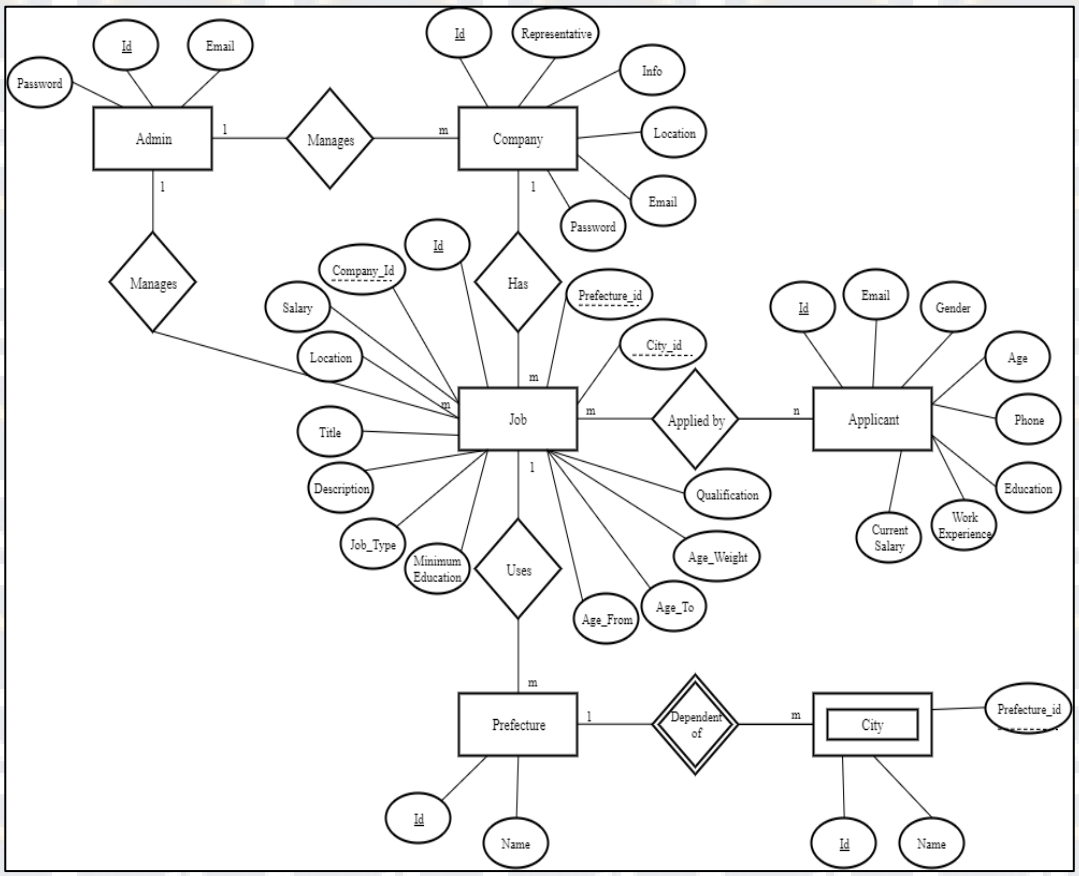

Figure 3.14 *Entity Relationship Diagram of Backend Job Seeking System*

There are 5 strong entities and 1 weak entity that are existed in the entity relationship diagram. The entity of admin manages company and job which it creates the relation between company and job, and therefore as what is shown in the diagram above, company has job. And job is using the entity of prefecture to determine the position of the job offered, city is a weak entity because it will depends on the prefecture, if there is no prefecture, then city will also not exist. Lastly, job is also related to applicant, where job can be applied by applicant and it will be reviewed by company.

3.4 Simple Additive Weighting (SAW)

In this online job seeking backend, we are using Simple Additive Weighting to rank the candidates who apply for a particular job. We limit the criteria in this research to two, which are age and education. So candidate that apply for that particular job will be rank according to age and education of the job requirement and weight of the age and education that are set by the company for that particular job which will be explained using a sample case. In Tbl. 3.1 is the weight that is set by the company.

Table 3.1

Weight Sample

Age from	25
Age to	35
Minimum education	Bachelor degree
Age weight	50% / 0.5
Education weight	50% / 0.5

And in Tbl. 3.2 is the candidates that apply for this particular job

Table 3.2

Candidate Sample

Candidate	Age	Education
C1	30	Bachelor
C2	25	Diploma
C3	28	Bachelor

So we will first count the range between the age from and age to and add 1, which is 11. And determine the range of each candidate age by subtracting the age to and their age and subtract by 1 because age is consider a cost. Lastly, the range of each candidate age over the range of the age requirement is being multiplied by the weight which has been set before. And for education, in the system itself the

weight of every education is being set from 0 to 6. Those with no education their weight is 0, primary school is 1, junior high school is 2, senior high school is 3, undergraduate is 4, master degree is 5 and doctoral is 6. The way to count the education is the same with the way to count the age but the range between the minimum education and the candidate education will be added by 1 instead of subtract by 1 because education is consider a benefit. The result of each candidate age can be seen in Tbl 3.3 and candidate education can be seen in Tbl 3.4.

Table 3.3

Age Result

Candidate	Result
C1	$\frac{4}{11}$ x 0.5 = 0.18
C2	$\frac{9}{11}$ x 0.5 = 0.4
C3	$\frac{6}{11}$ x 0.5 = 0.54

Table 3.4

Education Result

Candidate	Result
C1	$\frac{1}{3}$ x 0.5 = 0.167
C2	$\frac{0}{3}$ x 0.5 = 0
C3	$\frac{1}{3}$ x 0.5 = 0.167

The result of the rank can be seen in Tbl. 3.5

Table 3.5

Rank Result

Candidate	Result
C1	$0.18 + 0.167 = 0.347 = 34.7\%$
C2	$0.4 + 0 = 0.4 = 40\%$
C3	$0.54 + 0.167 = 0.707 = 70.7\%$

As shown in Tbl 3.5 we can conclude that C3 is the best candidate for the company to choose and next is followed by C2 and the last is C1.