



MIDDLE EAST TECHNICAL UNIVERSITY



DEPARTMENT OF COMPUTER ENGINEERING

CENG 491 - SENIOR PROJECT

FINAL DESIGN REPORT

18.01.2008

PREPARED BY



Aycan Tekerek

1347988

aycantekerek@gmail.com

Bariş Yanar

1348093

barisyanar@gmail.com

Şeniz Yıldırım

1502814

senizyildirim@gmail.com

Wai Phyoe Maung

1368687

waiphyoe2003@gmail.com

TABLE OF CONTENTS

1. INTRODUCTION.....	4
1.1. Background	4
1.2. Project Redefinition	5
1.3. Project Goals and Scope.....	8
1.4. Design Goals	9
1.4.1 Ease of Use.....	9
1.4.2 Consistency.....	9
1.4.3 Interoperability and Portability.....	10
1.4.4 Security.....	10
1.5 Design Constraints.....	11
1.5.1 Language Constraints.....	11
1.5.2 Experience & Skills of Members Constraints.....	11
1.5.3 Time Constraints.....	11
1.5.4 Resource Constraints.....	11
2. ARCHITECTURAL DESIGN.....	12
2.1. System Modules and Overall Architecture.....	13
2.2. Functional Design.....	14
2.2.1 Data Objects and Modeling.....	14
2.2.2 Data Flow Diagrams.....	21
2.2.3 Data Dictionary.....	26
2.2.4 Database Design.....	30
2.3. Behavioral Design.....	46
2.3.1. State Transition Diagram	46
3. SYSTEM DESIGN.....	47
3.1. Use Case Diagrams and Description.....	47
3.2. Class Diagram.....	55
3.3. Sequence Diagrams	64
3.4. Activity Diagrams	66
4. USER INTERFACE DESIGN.....	74
4.1 Main Page.....	75
4.2 Sign Up Pages.....	76
4.3 User Profile Main Pages.....	77

4.4 CV Creation Page.....	78
4.5 Job Search Results Page.....	79
4.6 Market Analysis Page.....	79
5. SYNTAX SPECIFICATION.....	80
6. TESTING ISSUES.....	81
7. PROJECT SCHEDULE.....	82
7.1. Project Task Set and Work Packages.....	82
7.2. Gantt Chart.....	89
8. APPENDIX.....	90
9. REFERENCES.....	90

1. INTRODUCTION

This document is the final design report of our Project named as İSTEİS. In the first part of the document, you can find our Project definition, the names and descriptions of its modules. Then we have discussed our general Project goals and scope, later we stated our design goals and constraints. The second part explains our modules and overall architectural design. We have drawn detailed Data Flow Diagrams, describe our data objects, data dictionary and our database design, the behavioral design of our Project is also explained by state transition diagram on this section. On the System Design level, you can see our updated Use Case diagrams and scenarios. We have also put class, sequence and activity diagrams of our system on this part. The upcoming part of the report is where we simply discussed and showed the graphical user interfaces of our modules. Later, we explained our syntax specification and testing issues. At the last part, you can see our work packages and explanation of what we have done so far and what we have next on our Schedule. The complete Project Schedule can be seen in appendix as a Gantt Chart.

1.1 Background

Today, internet is in every part of our lives. Every one of us without distinction of age, sex etc. uses internet extensively. Due to this technological improvement, traditional ways of doing many things have changed and are still changing. These days, we do shopping, pay our bills on the internet, make reservations, take tickets, and even get educational degrees online. Web services and applications make most of these possible.

Wikipedia defines Web services as a software system designed to support interoperable Machine to Machine interaction over a network. Web services are frequently just Web APIs that can be accessed over a network, such as the Internet, and executed on a remote system hosting the requested services.[1] In recent years, web technology has made a big progress and web services has started to been used in many different styles. Most popular one of these styles today is Service Oriented Architecture (SOA).

“SOA is an architectural style for building software applications that use services available in a network such as the web. SOA enables businesses to leverage existing investments by allowing them to reuse existing applications, and provides interoperability

between different applications and technologies.”[2] Service-oriented architecture is a hot topic in enterprise computing. Because especially a web services – based SOA has the potential of dramatically speeding up the application development process. It is also a way to build applications and systems that are more adaptable then systems become more agile in responding to changing business needs. SOA is clearly the wave of the future.

We are building an online portal using web services – based Service Oriented Architecture. SoftwareAG which is one of the leader companies in SOA solutions is our supporting company in this Project.

1.2 Project Redefinition

Our Project is a gateway portal where we will be able to get together both job seekers and job providers in cooperation with career web pages. Essentially, our Project will act as traffic driver to those career web sites. Simply, a job seeker can become a member of our portal and leave his/her CV and do job searches within all offers from our source career web pages and apply any of them. While a job provider company can publish their vacant positions on any of our cooperating career web sites by having a membership of both our portal and the subjected career web page. And also they can search employees from the CV pools of the source web sites on which they also have an account. Because we have two types of external users and our administrators as internal users, we have to consider the services that we provide for our users on our portal in three different categories. There will be services for job seekers and similar or different services for job providers, employer companies and services for system administrators to manage the portal. To give these services to our portal’s users, we need services from our cooperating career web pages. These needed services are basically data insertion, selection and update services on their databases, the statistical data and their interpretation.

Our portal will run on a membership system for real users. At first, both job seekers and providers have to have an account to use our portal. TC Kimlik No validation for all users and also Vergi Kimlik No validation for companies will be made before any account is activated. And we are not planning to charge any fees from job seeker users but job providers have to pay a little membership fee to have an account on our portal.

The services that we will offer for job providers can be listed as:

Payment Module:

We need this module for both taking the membership fees from job provider users to have an account on our portal and transferring the membership fees of chosen career web pages by employer companies to work with and publish their job offers. At this point, we are simply thinking of informing employer companies about what information we need from them on this payment action and how much they need to pay, and then we provide a bank account number that they can transfer total membership fees for both our portal and any career web page that they want to become a member of. After this transfer, we will be transferring the amount that is needed for a membership on the corresponding career web page to this web sites bank account that we would already arrange with this career web page. The administrator users of our system will be in control of all these actions.

Publish Site Selection Module:

This module allows our job provider users to select on which cooperating career web sites they publish their job offers. To help employer companies on their choice we will have a section where we compare these career pages on different categories such as; membership fees, extra services they provide, the statistics of their preference by job seekers, numbers of positioned job seekers to published offers on their web sites. And then we will grade each career web site on these categories. We think this service helps job providers and decreases the time spent on this decision.

Employee Search Module:

By this module job providers can search employee having the properties that they want on different categories through the job seeker information pools of the cooperating career web pages on which they have a membership.

Information Module:

Our member employer companies will be informed of any change or improvement made on our portal or career web sites services. Also, they can see if there is any application to their published job offers and who did these applications. Moreover, this module will inform job provider users about their membership statuses of both our portal and cooperating career web page.

The services that we will provide for job seekers can be categorized as:

Job Search Module:

By this module job seekers can search job offers on our portal, and see all resulting offers with the related search tags gathered from all career web sites that we are working with. They can see the published offer as it is given to the career web site with the reference of the subjected career web page.

Job Application Module:

This module allows Job seekers leave their CVs in the forms word or PDF documents, videos or the standard form that we provide for them. We will be sending all these information to each career web site that we will be working with. Applications to the jobs that they choose will be made according to related career web sites' ways which is preceded as it is on our portal.

Information Module:

We will inform our job seeking members with offers made to them and new job offers published on any career site that is in cooperation with our portal. We will also planning to show job seekers different job opportunities from those that they are looking for according to their skills, experience. Moreover, most popular and preferable jobs, positions and companies will be listed on our portal. Their application status will also be sent by e-mail to them.

Feedback Module:

This module will get into action after positioning of any job seeker occurs. We will send polls to job seekers to get the feedback of the process. These polls will contain questions about our portal considering such as the ease of use, any lacking properties or services. The referring career web site will also be questioned about their services, etc.

For management and control of our system, we provide services for our system administrator users. These are:

User Management Module:

System administrators of our system can add or delete users, the addition of users is limited to the new system administrators and the job provider companies that paid the

membership fee of our portal. However, deletion can be applied to all kinds of users such as job seekers who does not use their account for a long time period or job provider companies which do not renew their membership on our portal and want the deletion of their accounts. Moreover any problem regarding user ids or passwords of our members will be solved by our system administrators. The informing of each user about their status or announcements is also controlled by administrators.

Poll Management Module:

Poll management process consists of adding/updating/deleting polls, sending the polls to the related users, gathering the poll results and making comments out of them and updating the related parts of the system according to those comments.

Commercial Adds Management Module:

We will be taking and publishing commercial adds on our portal. These ads' addition to system, arrangement, publishment, deletion will be made by system administrators. The seen adds will be changed according to the user type on our portal. For example; we will be showing different ads to job seekers and job providers cause they may generally be interested in different products.

The basic architecture of the project is SOA and all the communications with the cooperating career web pages will be made through the web services that these web sites provide us.

1.3 Project Goals and Scope

Our goal in this Project is to develop a portal which makes job and employee finding processes much easier and less time consuming. Our portal will prevent job seekers to search all career web pages and do the same create an account, upload a CV, search job offers, and apply to job procedure. Also it will simplify finding a career we site to publish offers or searching employee processes for job providers.

Besides, on doing this Project we have goals like;

- building a web based service oriented architecture
- completing web portal at the end of 7 months.

- . creating a portal which is interactive, easy to use, efficient and visually attractive.

The scope of this Project is providing a methodology and guideline to build a gateway portal integrating many services from different sides. Because this integration idea can be used in applications in many different sectors. This Project will help developers in;

- Proposal of a web – based Project,
- Analysis and documentation of current gateway portal applications,
- Specification of a detailed user, system and developer requirements,
- . Design of an appropriate system having necessary properties,
- Implementation of Service Oriented Architecture and Web Services on Project.

1.4 Design Goals

Besides general Project goals, we have design goals for our portal. Our Project is web – based SOA application so it has to have this specific architecture's design principles. Interoperability, portability, modularity and reusability are the main properties of a SOA based applications. Moreover, our platform has Web Services layer, this component makes security is an important issue for our Project. Ease of use and consistency are other important design issues of a system.

1.4.1 Ease of Use:

Our portal will be user-friendly; clear, sightly, and not subsuming useless structures. If a visitor encounters pages, links, animations and popups much more than adequate it will create a bad impression, which is undesirable. Also the pages and menus will be designed well. Priority functions like login, sign up, and search should be more striking to obtain the user-friendliness and clearness.

1.4.2 Consistency:

Our portal should be able to manage invalid user inputs or inconsistent conditions. It provides error checking to ensure the right input format and returns errors and warnings to the user.

1.4.3 Interoperability and Portability:

Interoperability is the most important principle of SOA. This principle allows different distributed web services to run on a variety of software platforms and hardware architectures. Our portal will be operating with different career web pages and all these web pages can have different platforms, so it has to be interoperable. Moreover, our system is going to be software and hardware independent. The internet connection will be sufficient to access and use our portal.

1.4.4 Security:

Web services provide significant new benefits for SOA-based applications, but they also expose significant new security risks. Creating and managing a secure Web services environment involves dealing with various Internet, XML, and Web services security mechanisms. So, we are building a web service based SOA application and we will be storing many important personal information on our portal, we will be taking into account below security issues:

- Transport-level security such as firewalls, virtual private networks, basic authentication, non-repudiation, and encryption.
- Message-level security such as using authentication tokens to validate requester identity and authorization assertions to control access to provider services.
- Data-level security such as encryption and digital signature to protect against altering stored and/or transmitted data.
- Environment-level security such as management, logging, and auditing to identify problems that need to be fixed and establishing trusted relationships and communication patterns.

For securely transferring data, we are planning to use https protocol instead of http. Https is an updated version of http which is combined with SSL (Secure Socket Layer). SSL provides cryptographic system with two encrypted keys, one public key and a private key which is only known to recipient. In consideration of environment-level security the user management will be done by our administrator users. Every user on the portal will be allowed to see only his/her information.

1.5 Design Constraints

1.5.1 Language Constraints

Our Project will be built based on Web Services and AJAX Technologies and we are going to do the development in java on the tools provided for us by our supporting company SoftwareAG. WebMethods Integration Server is used to design and integrate our services.

1.5.2 Experience & Skills of Members Constraints

Although, we have some experience on programming and design, our capabilities are restricted. Web Services and Service Oriented Architecture concepts are new to all of our Project members. Also, we are going to work on development tools that we have not used before, so we will have unexpected problems considering both the architecture and development tools throughout the Project. However, we have a supporting company on this Project, in any problem we have experienced people to consult and get help from.

1.5.3 Time Constraints

The most important constraint on our Project is the time constraint. We have to finish our portal by June and also we should provide a prototype at the end of this semester. We have strict Schedule consisting of determined milestones. We have to follow this program and at the end finish our Project in time. Any time loss can cause us to be left behind Schedule and it will be so hard to compensate this loss.

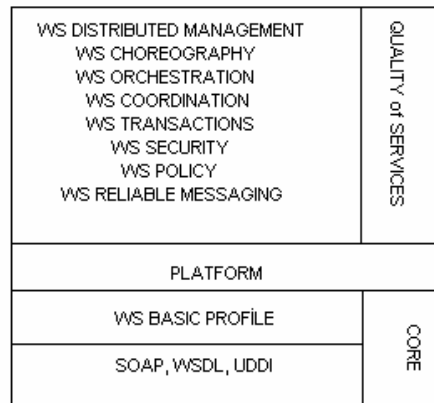
1.5.4 Resource Constraints

We explained that we were going to use WebMethods Integration Server and Application Designer tools of SoftwareAG to develop our Project. Both these tools have their own hardware and software requirements. We will be working with WebMethods Integration tool on a virtual machine set up by Vmware software. Application Designer can be embedded into Eclipse development environment. We have to had installed and made necessary adjustments on these software before starting to develop our Project. For hardware we need PCs has properties: 1Gz or higher CPU, 512 MB main memory, TCP/IP connection.

2. ARCHITECTURAL DESIGN

2.1 System Modules and Overall Architecture

Our project, isteis, provides and demands web services, and requires **Enterprise Service Bus (ESB)** for integration of data and applications with our trading partners. Briefly, it is an implementation of Service Oriented Architecture (SOA). SOA is a specific architectural style that is concerned with loose coupling and dynamic binding between services. Below is the infrastructure of SOA.

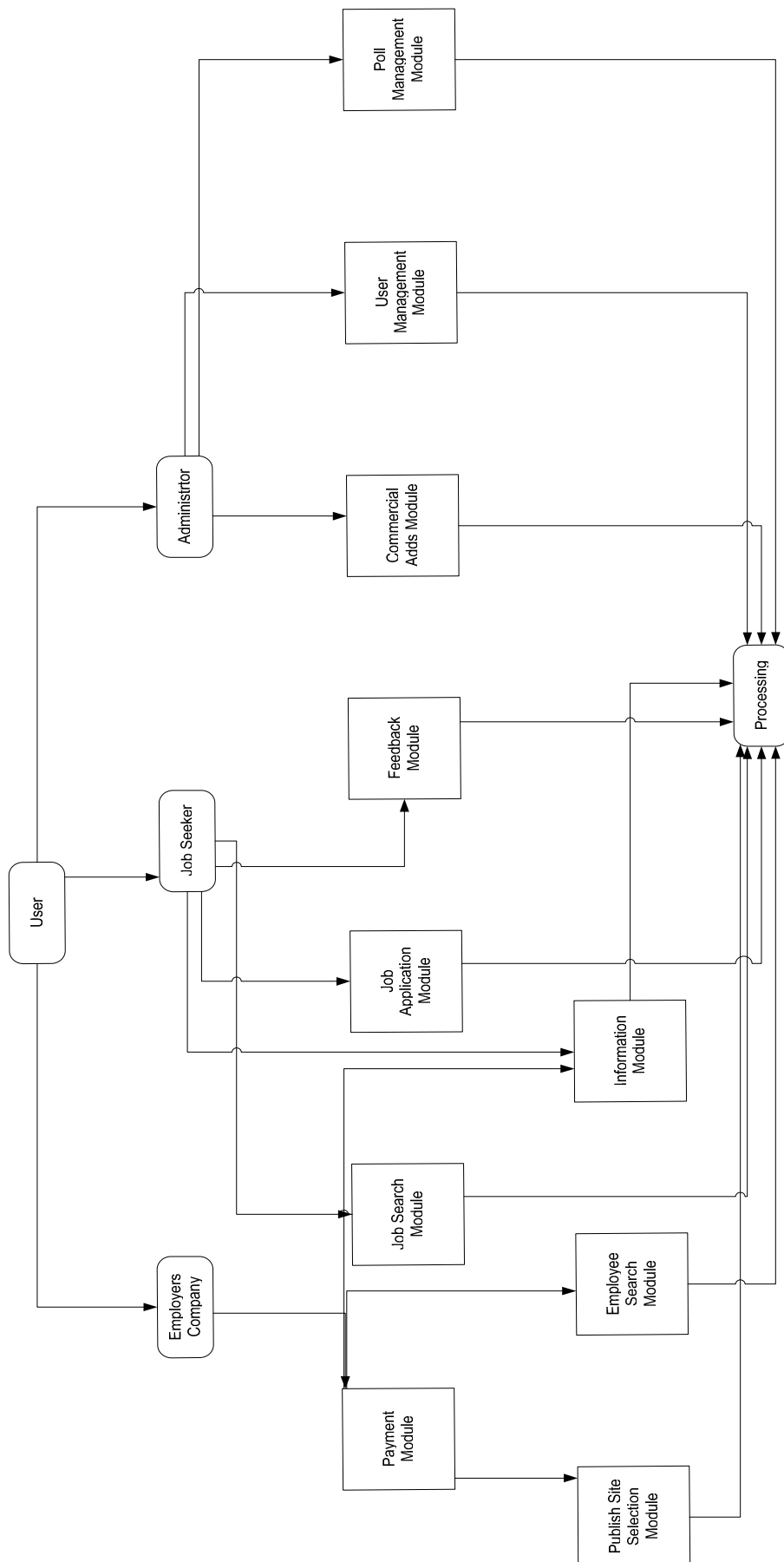


Here, WSDL, UDDI, and SOAP are the fundamental pieces of the SOA infrastructure. WSDL is used to describe the service; UDDI, to register and look up the service; and SOAP, as a transport layer to send messages between service consumer and service provider.

We have discussed our modules in problem definition section. Those are namely:

- . Payment Module
- Publish Site Selection Module
- Employee Search Module
- Information Module for our member employer companies and,
- . Job Search Module
- . Job Application Module
- . Information Module
- . Feedback Module for our portal's job seeker users.
- . User Management Module
- Poll Management Module
- Commercial Adds for our system administrator users.

2.2 Structure Chart of the System



2.3 Functional Design

2.3.1 Data Objects and Modelling

Entity Sets

Career Site

ID#	Site Name
-----	-----------

Job Seeker

ID#	Name	Last Name	Username	Password	e-Mail 1	e-Mail 2	Father's Name	Birth date	Secret Question	Secret Answer
-----	------	-----------	----------	----------	----------	----------	---------------	------------	-----------------	---------------

Employer ID#

Job Provider

ID#	Career Site ID#	Company Name	Applicant's Full Name	Applicant's Position	Phone#	Country	Province	City	District
-----	-----------------	--------------	-----------------------	----------------------	--------	---------	----------	------	----------

Address	Sector	Employee#	Username	Password	Secret Question	Secret Answer
---------	--------	-----------	----------	----------	-----------------	---------------

Polls

ID	Poll_ID	Vote
----	---------	------

CV

Personal Information

Seeker ID#	Gender	Phone#	Mobile Phone#	TC Identity #	Military Service Status	Marital Status	Country	Province	District
------------	--------	--------	---------------	---------------	-------------------------	----------------	---------	----------	----------

Educational Information

Seeker ID#	Education Level	Status	Graduation Date	Department	School
------------	-----------------	--------	-----------------	------------	--------

Work Experience Information

Seeker ID#	Company Name	Business Sector	Work Area	City/Country	Start Date	End Date
------------	--------------	-----------------	-----------	--------------	------------	----------

Language Information

Seeker ID#	Language Name	Reading Level	Writing Level	Speaking Level	Place Learned
------------	---------------	---------------	---------------	----------------	---------------

Computer Programs

Seeker ID#	Program Name	Knowledge Level
------------	--------------	-----------------

Programming Languages

Seeker ID#	Language Name	Knowledge Level
------------	---------------	-----------------

Operating / Network Systems

Seeker ID#	System Name	Knowledge Level
------------	-------------	-----------------

Databases

Seeker ID#	DBMS Name	Knowledge Level
------------	-----------	-----------------

Institutional Software

Seeker ID#	Software Name	Knowledge Level
------------	---------------	-----------------

Office Utilities

Seeker ID#	Utility Name
------------	--------------

Certificate Information

Seeker ID#	Certificate Name	Date	Institution
------------	------------------	------	-------------

Seminar Information

Seeker ID#	Seminar Name	Institution	Duration	Date
------------	--------------	-------------	----------	------

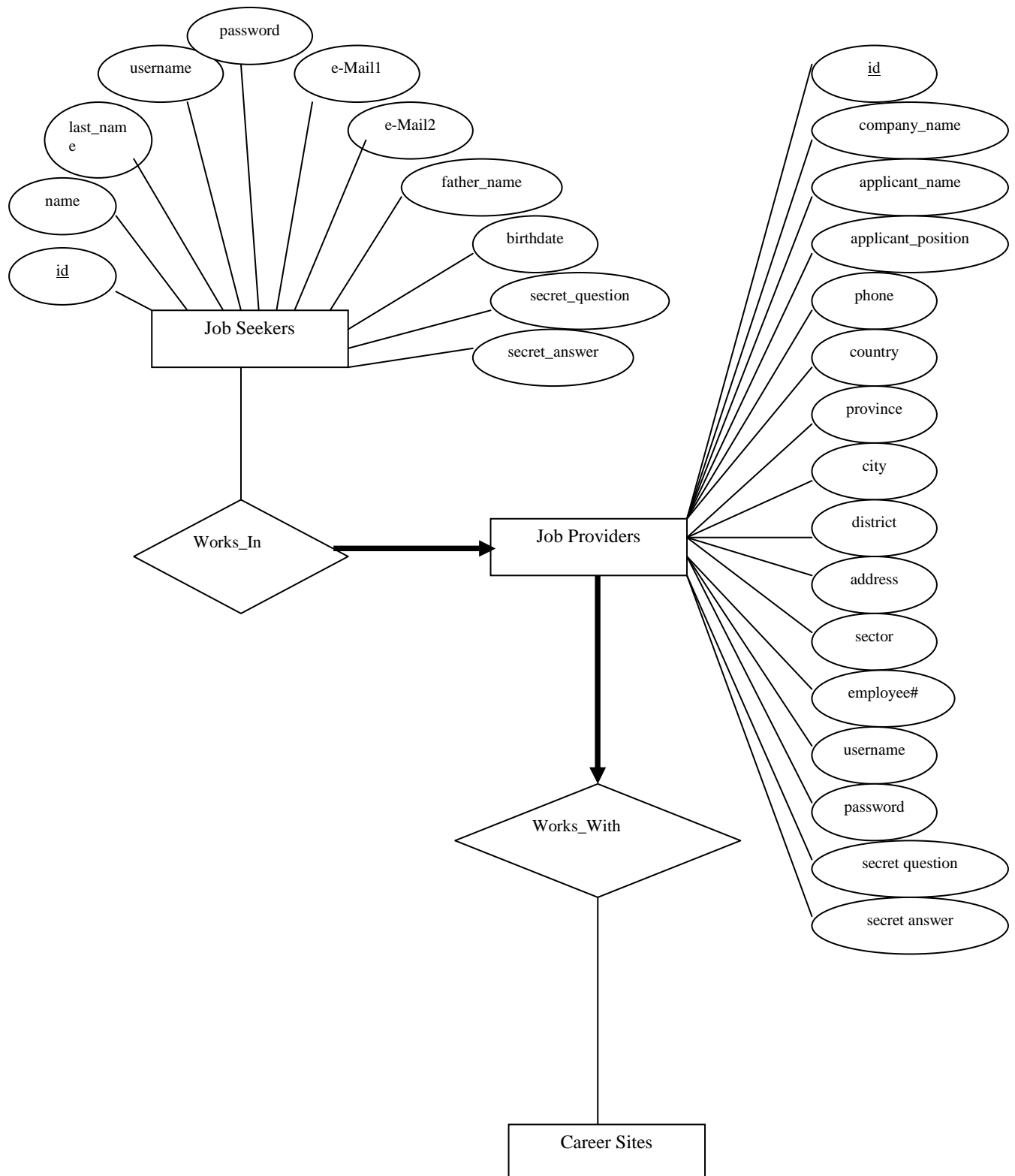
Examinations

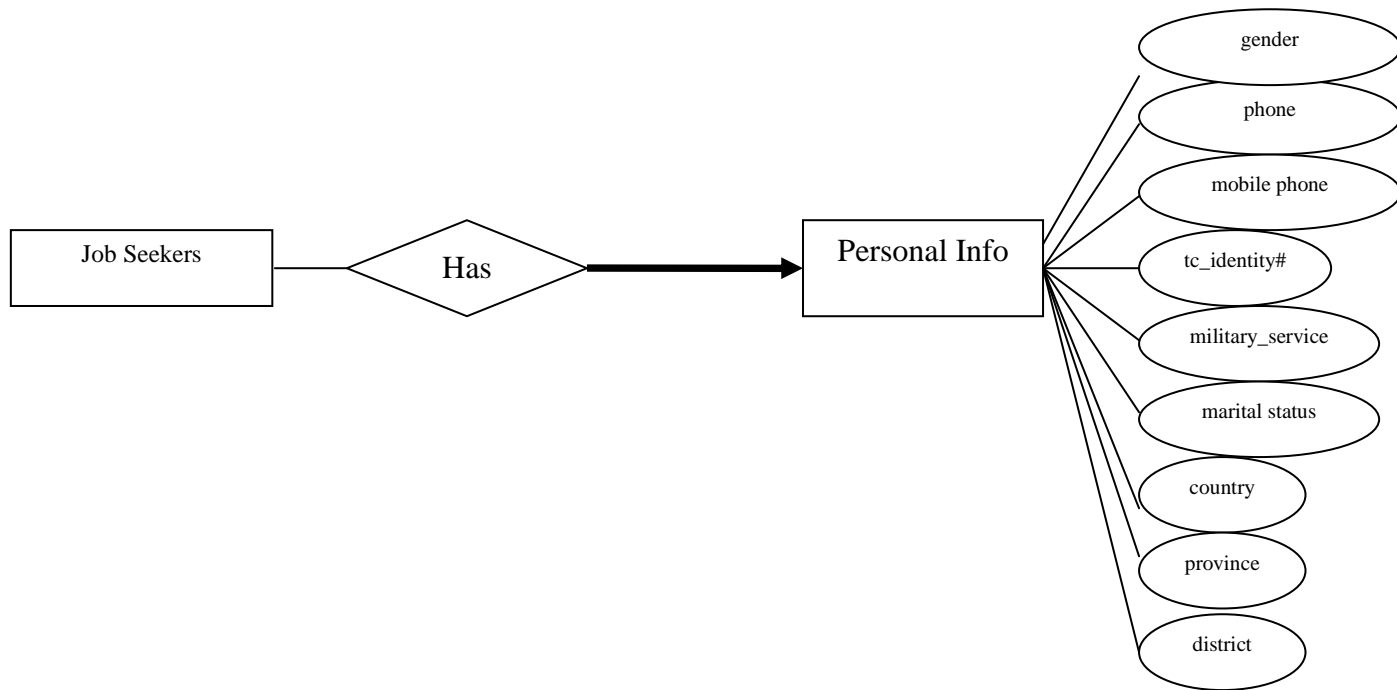
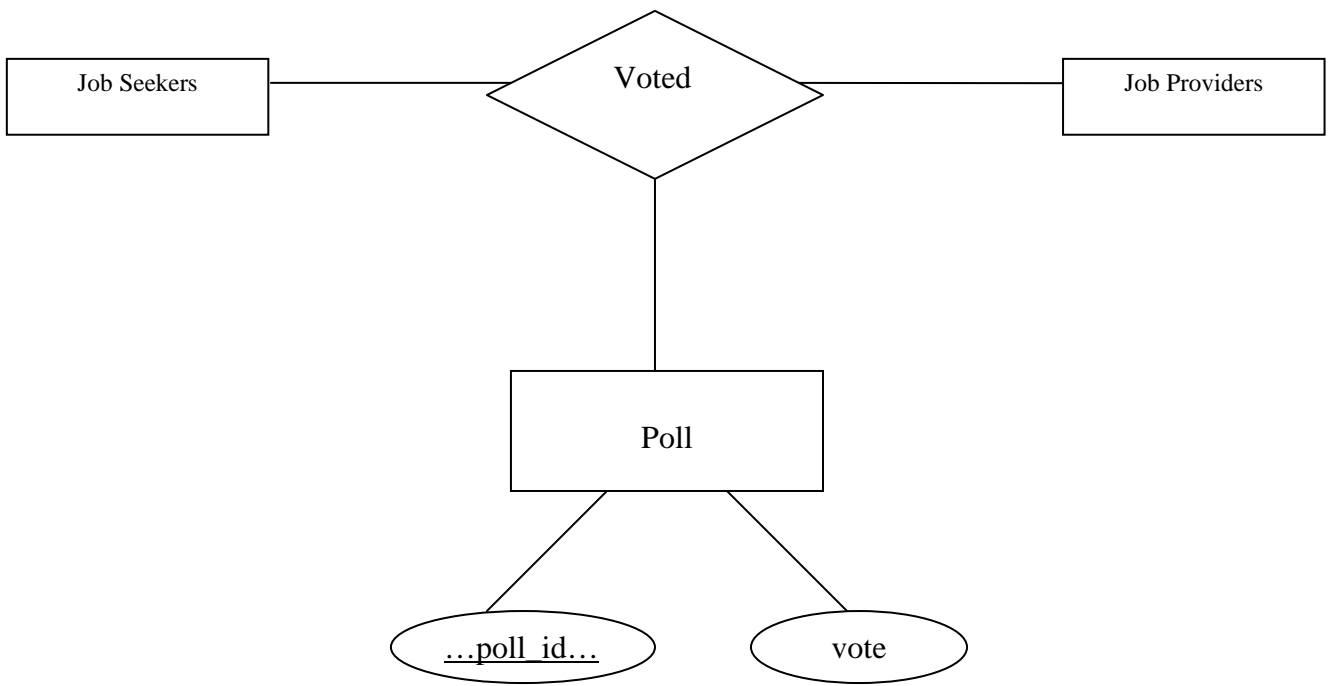
Seeker ID#	Examination Name	Examination Grade	Institution	Date
------------	------------------	-------------------	-------------	------

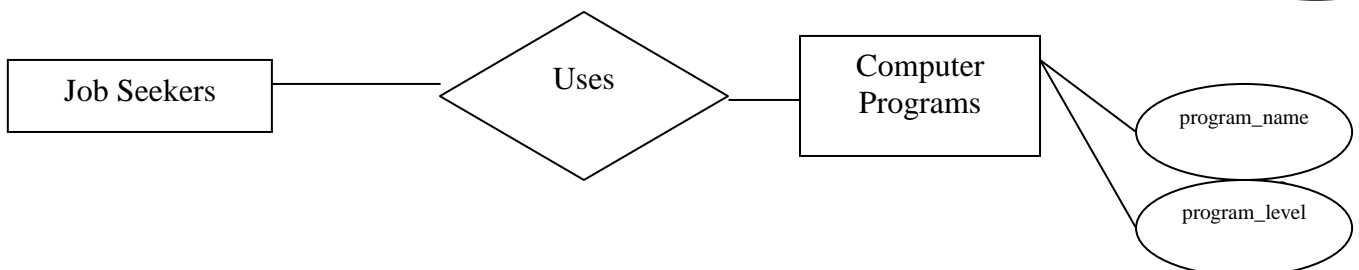
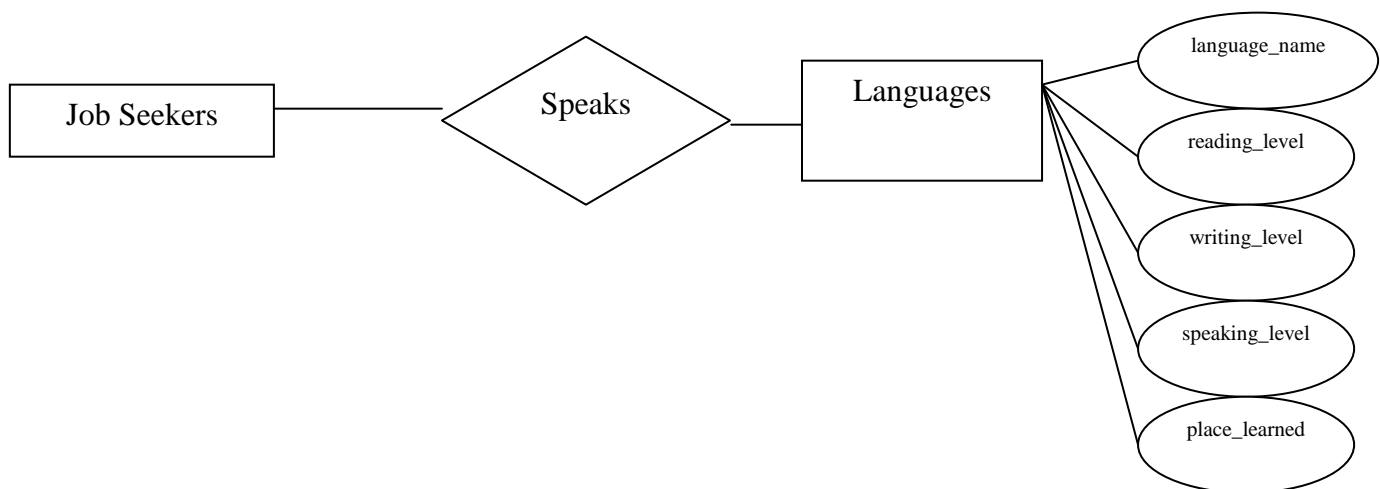
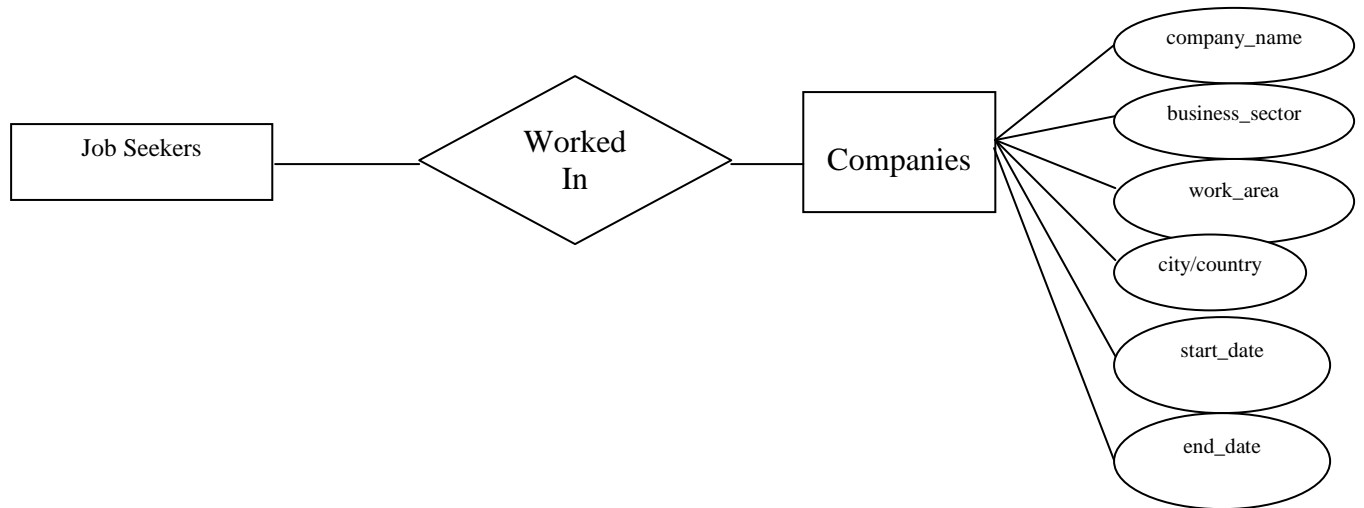
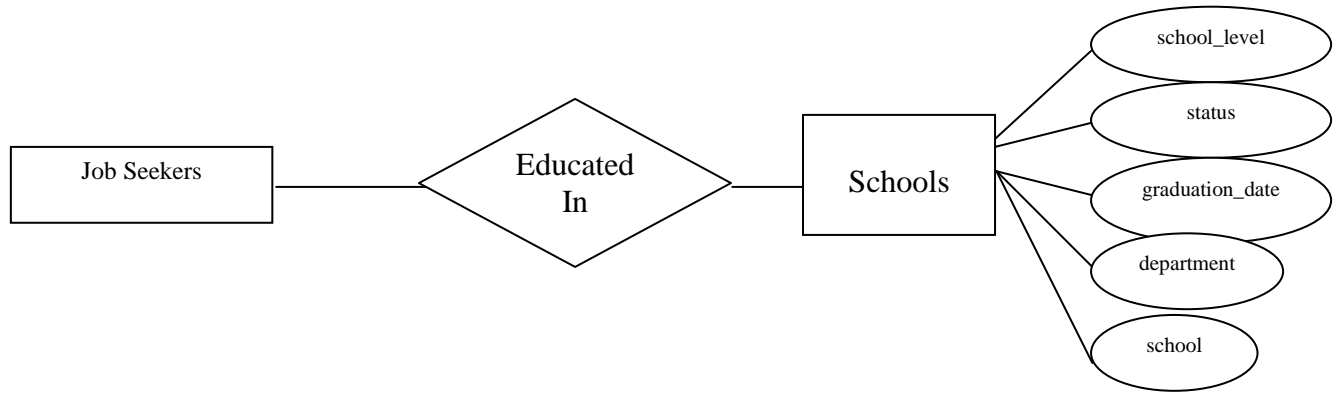
References

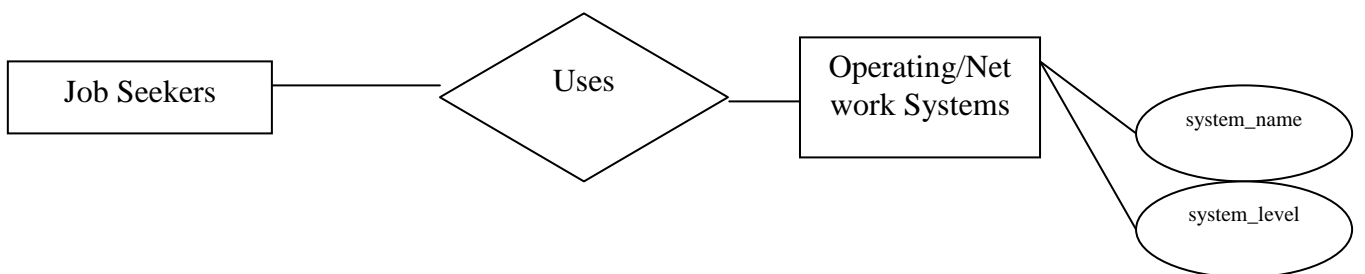
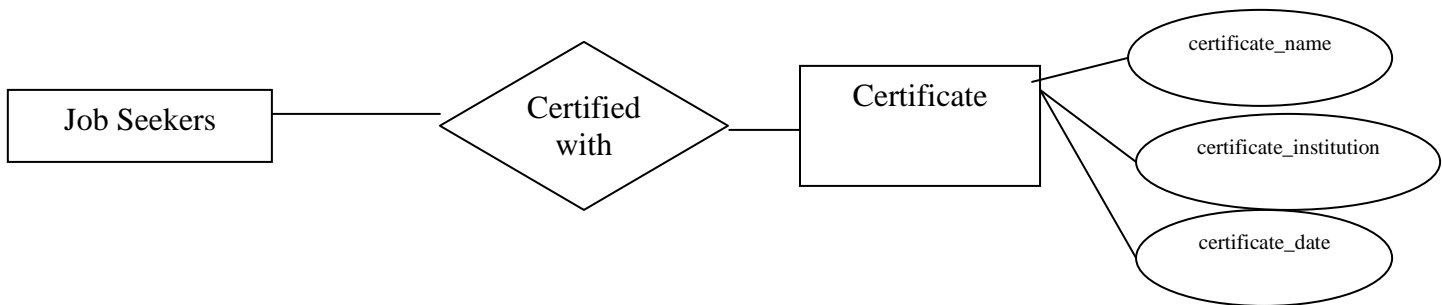
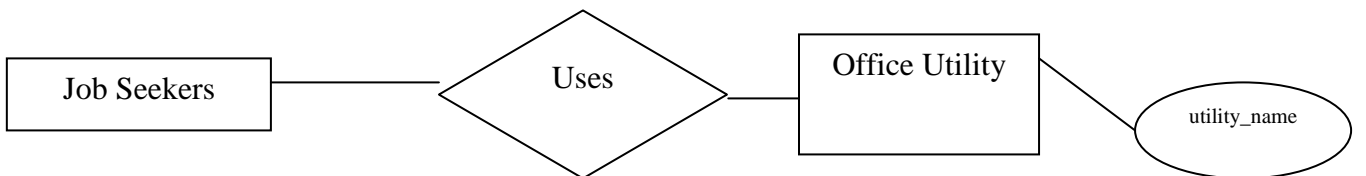
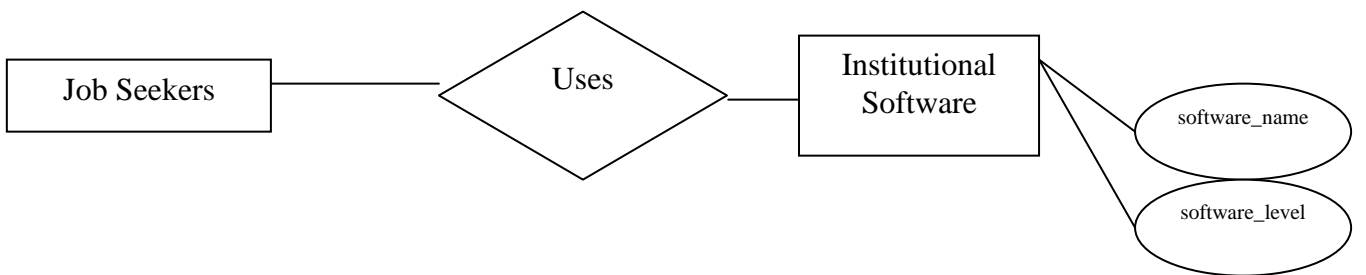
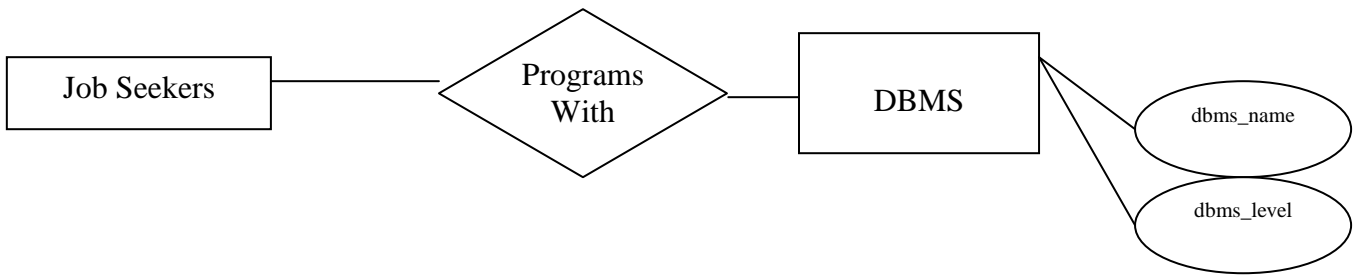
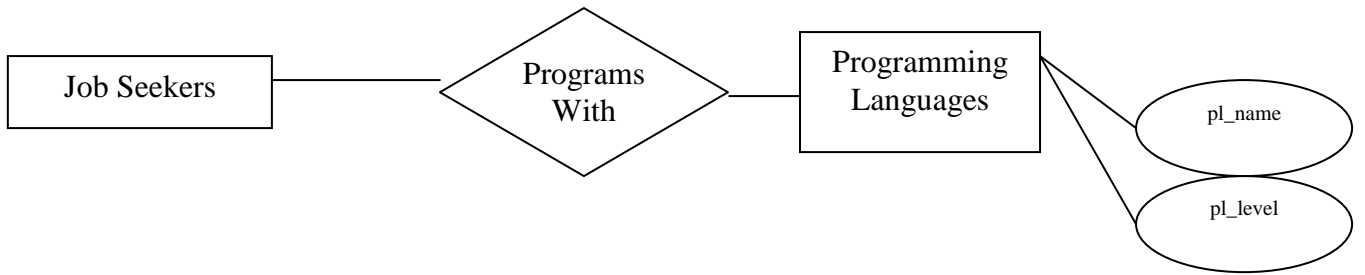
Seeker ID#	Reference's Name	Reference's Last Name	Company	Position	Phone	e-Mail	Reference Type
---------------	---------------------	--------------------------	---------	----------	-------	--------	-------------------

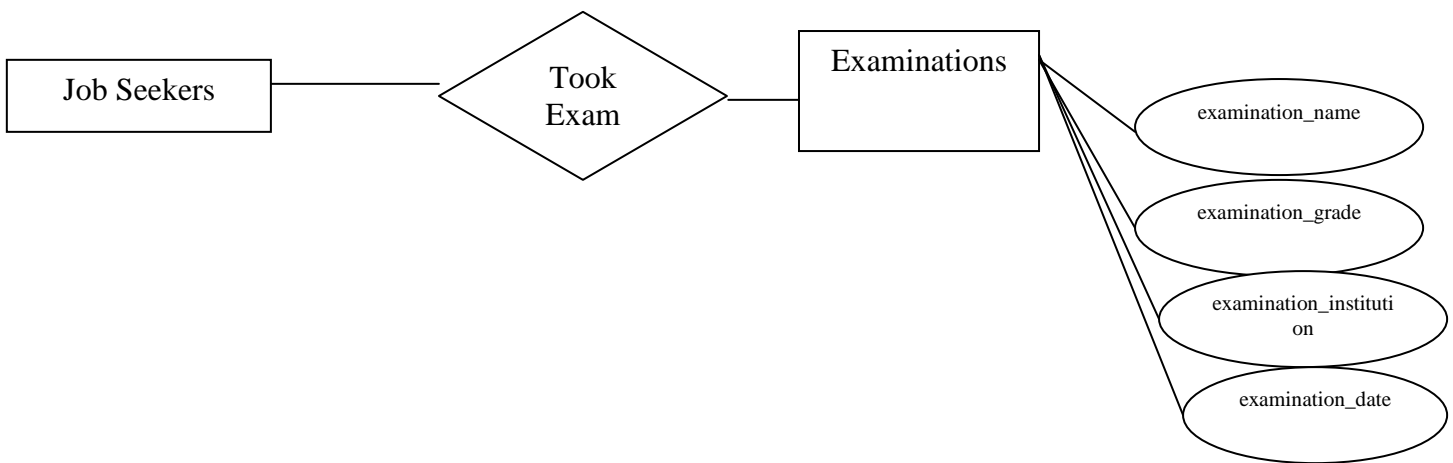
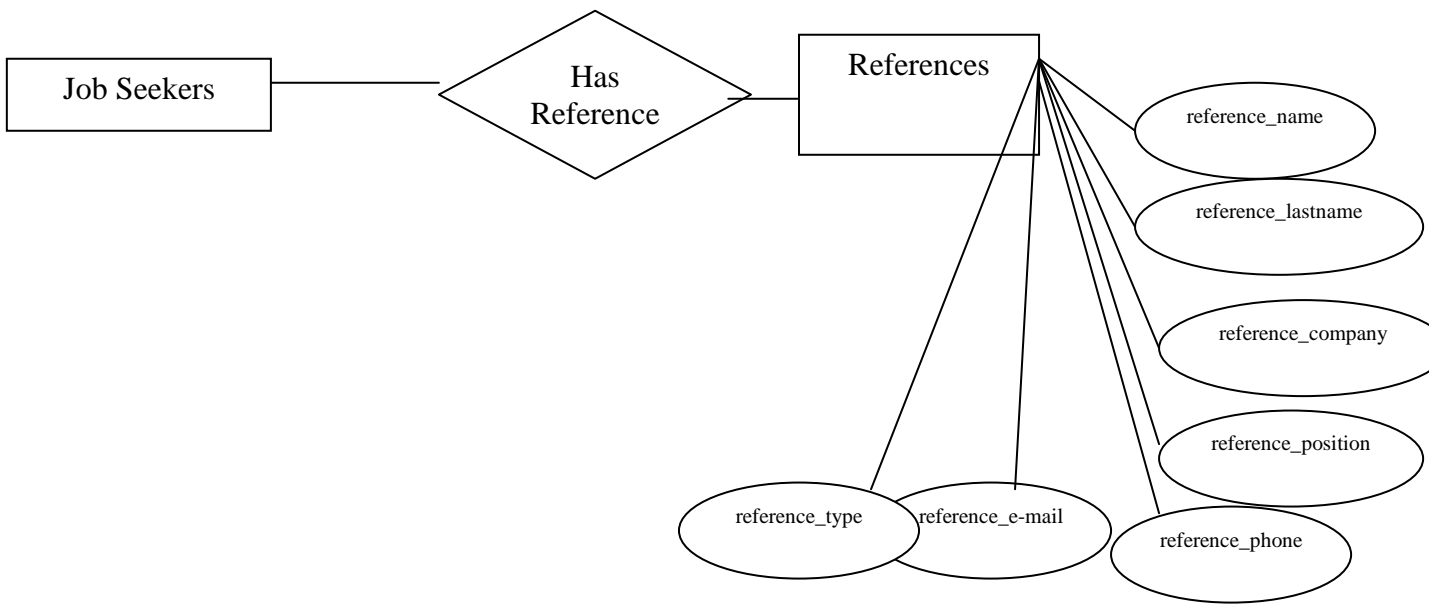
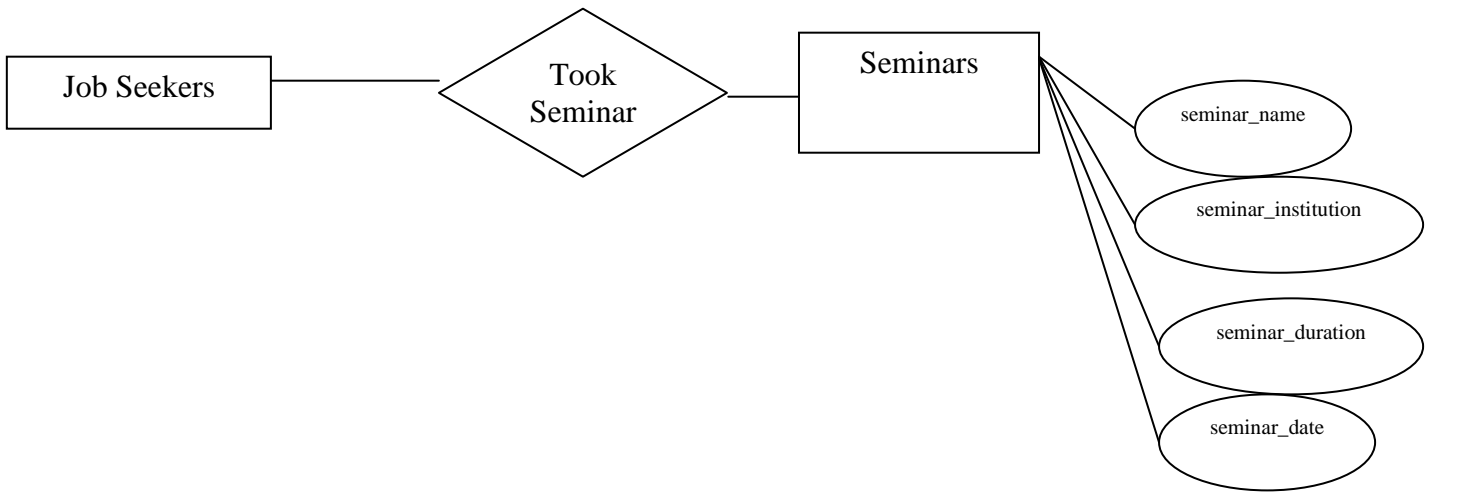
ER Diagrams





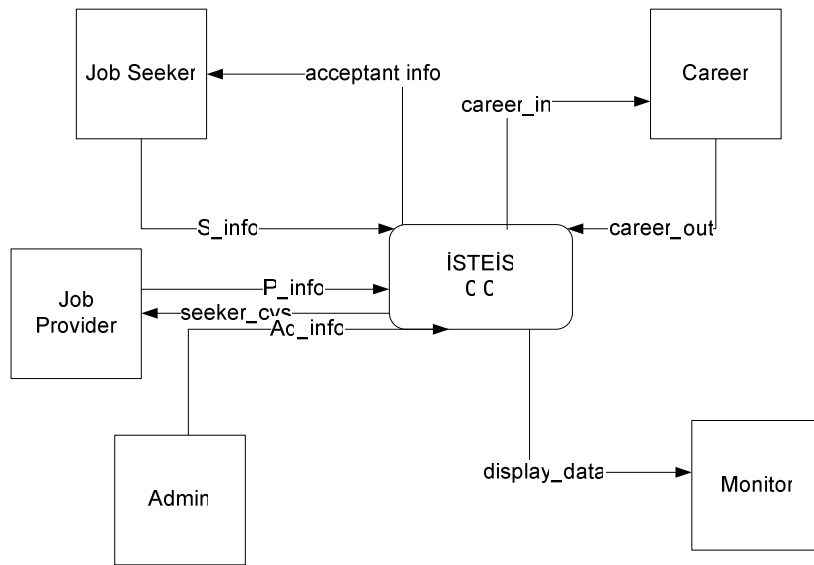




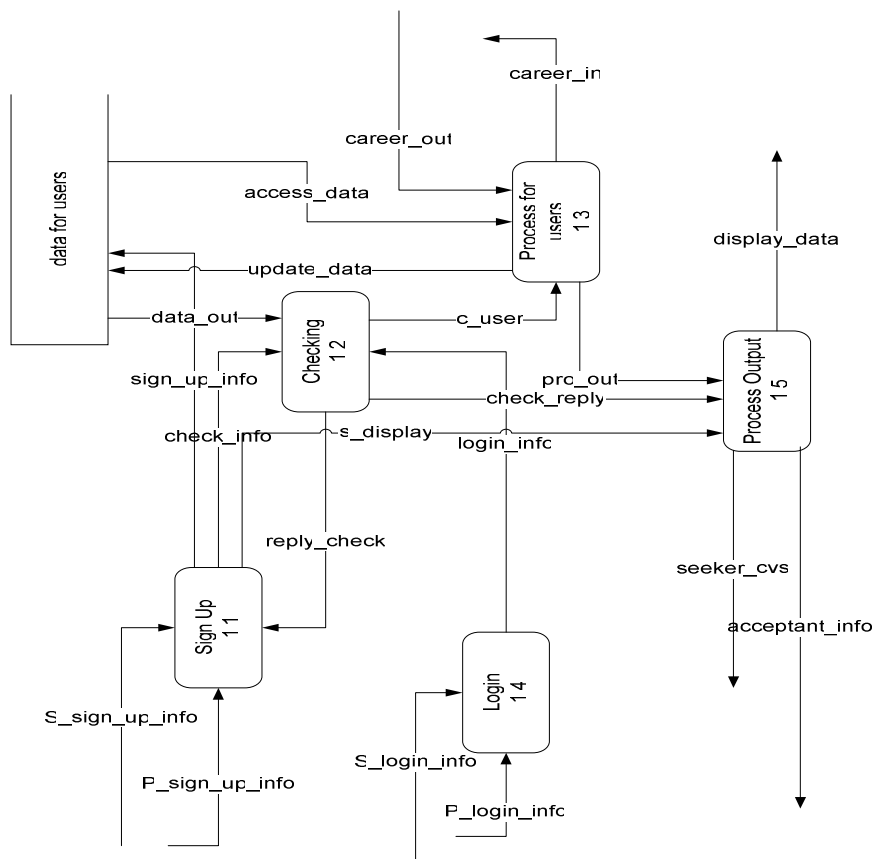


2.3.2 Data Flow Diagrams

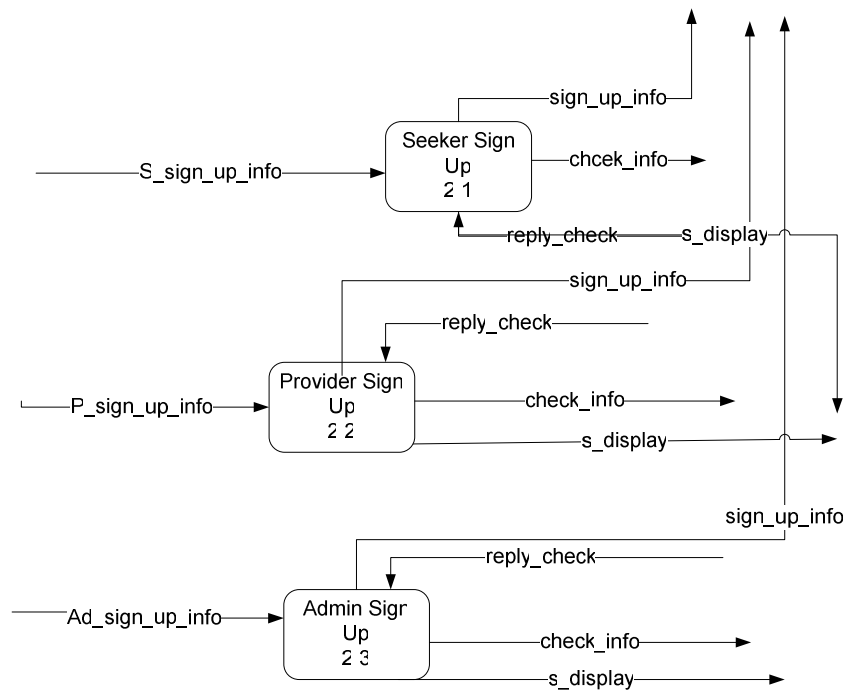
a. DFD : Level 0



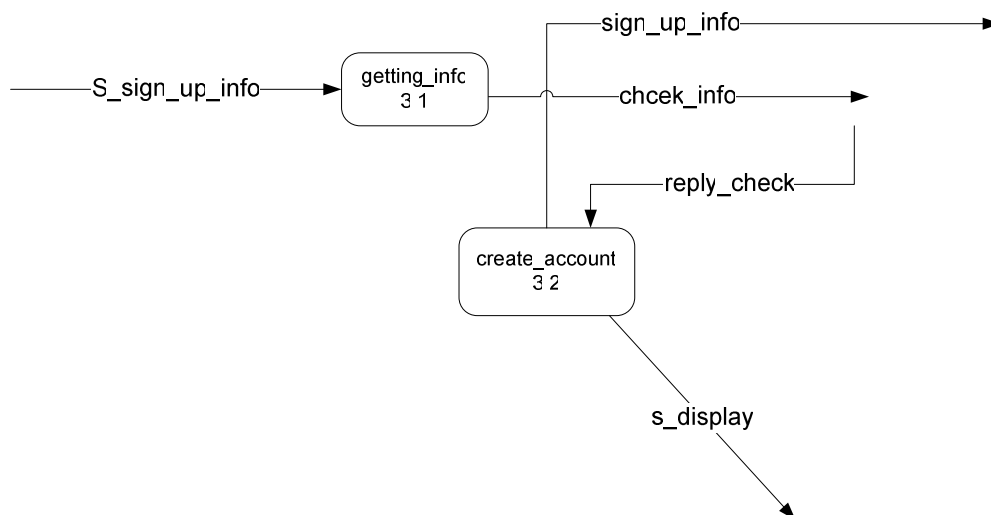
b. DFD : Level 1 for “iSTEIS”



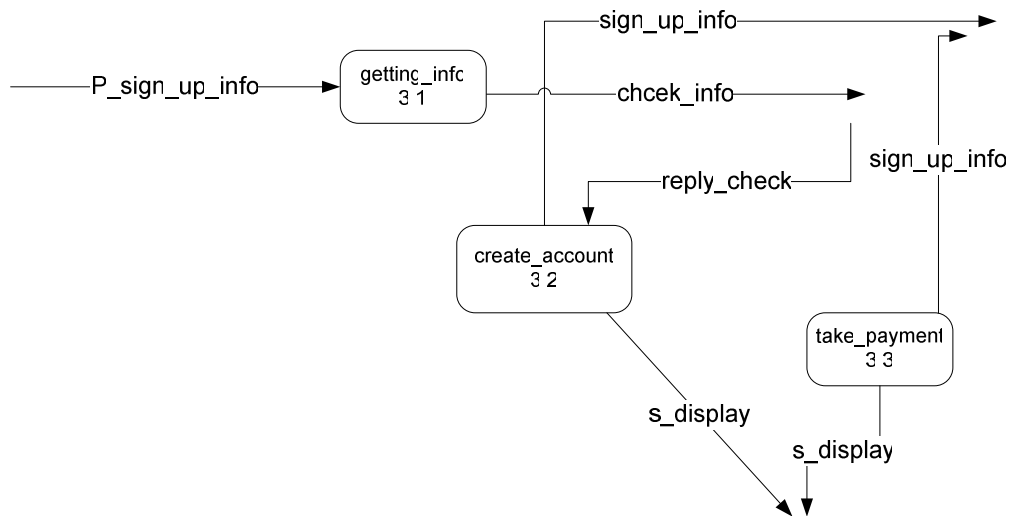
c. DFD : Level 2 Sign Up 1.1



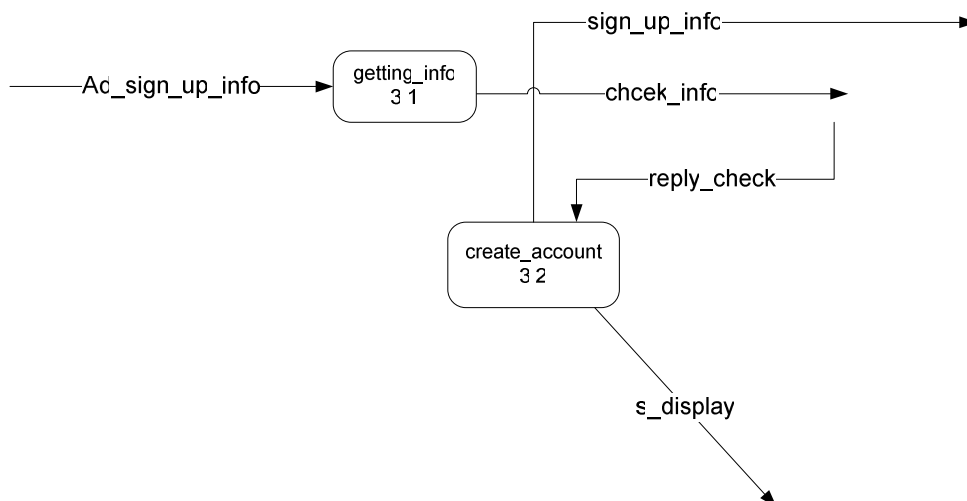
d. DFD : Level 3 Seeker Sign Up 2.1



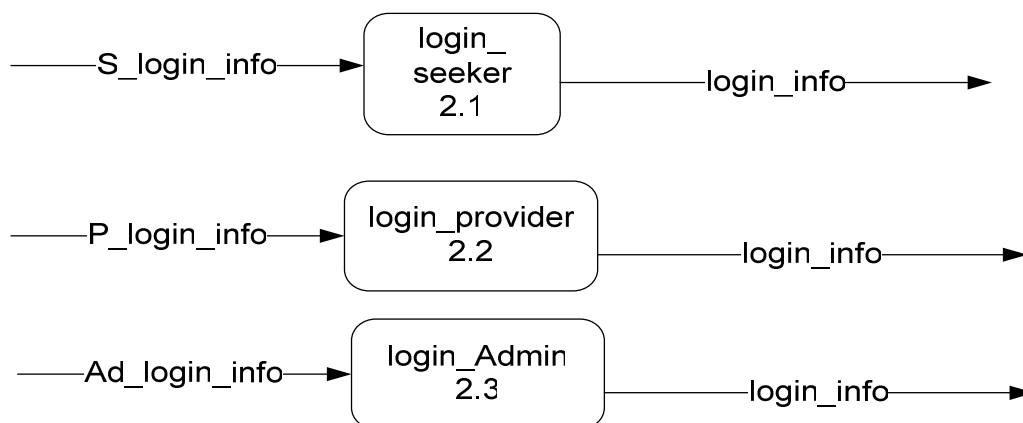
e. DFD : Level 3 Provider Sign Up 2.2



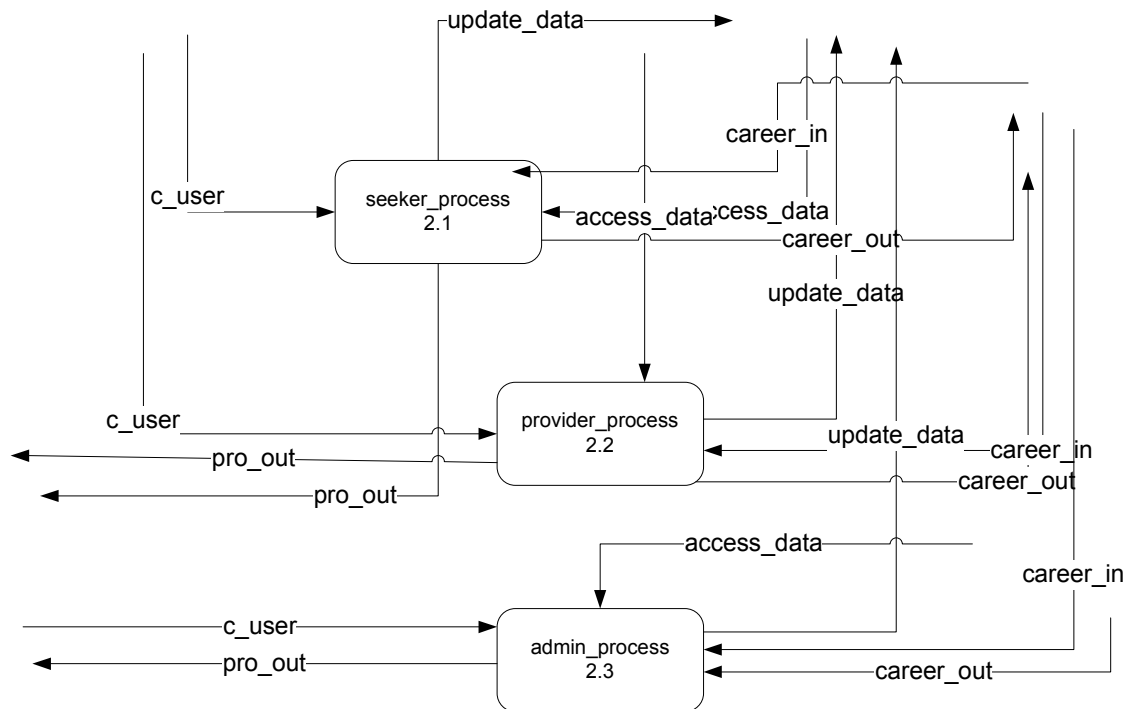
f. DFD : Level 3 Admin Sign Up 2.3



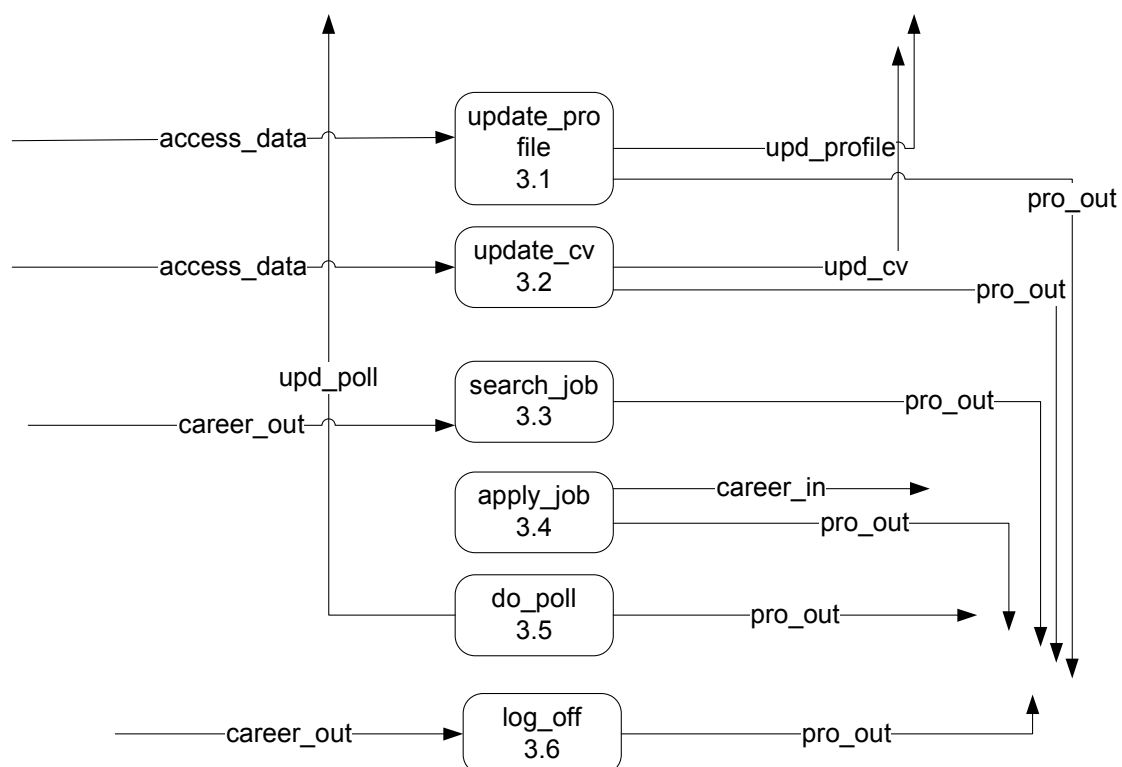
g. DFD : Level 2 Login 1.4



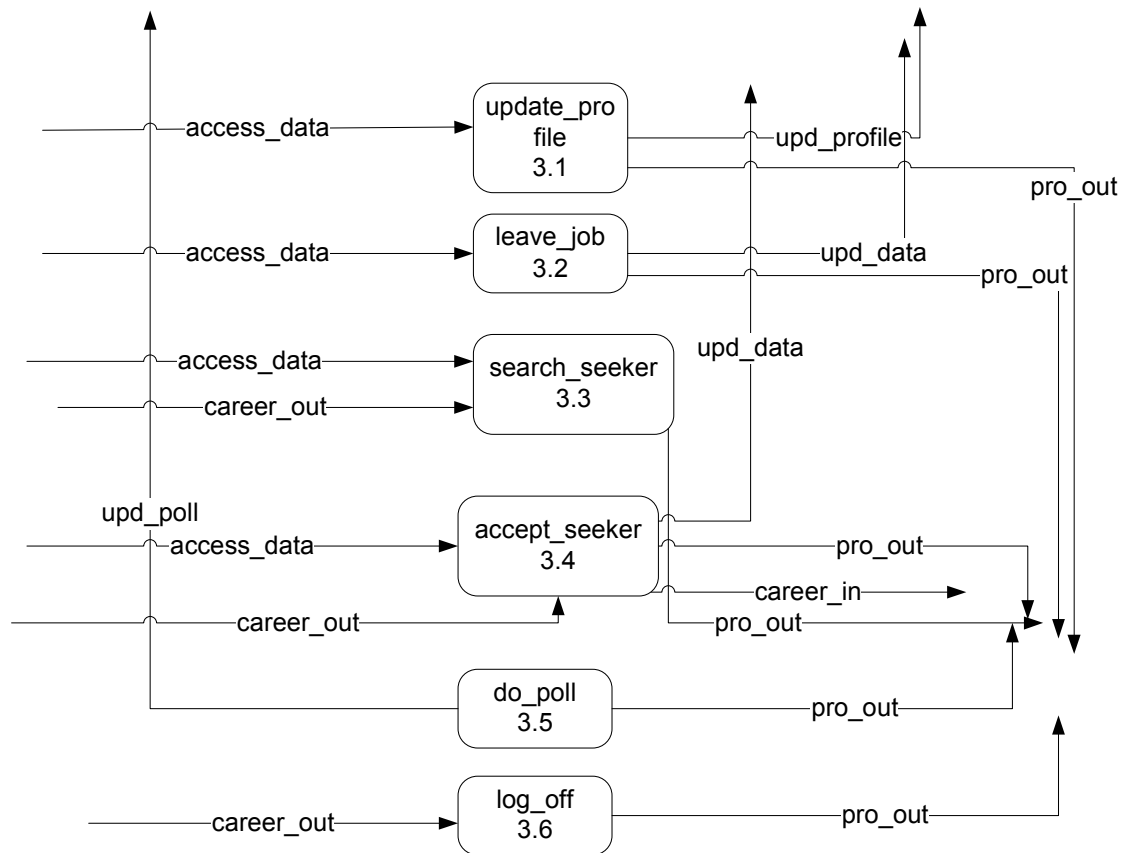
h. DFD : Level 2 Process for user 1.3



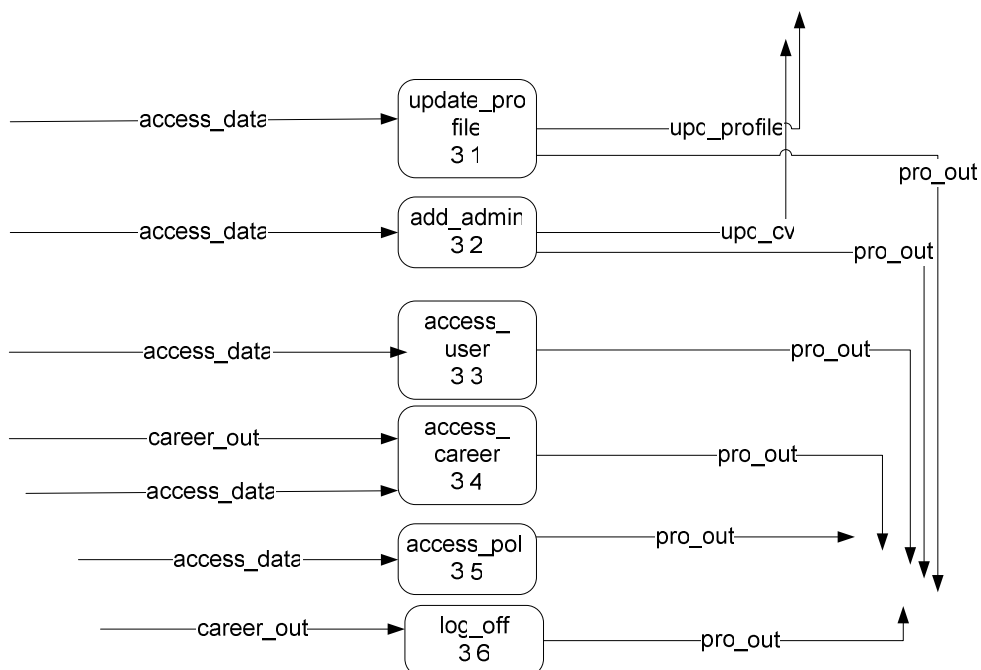
i. DFD : Level 3 seeker_process 2.1



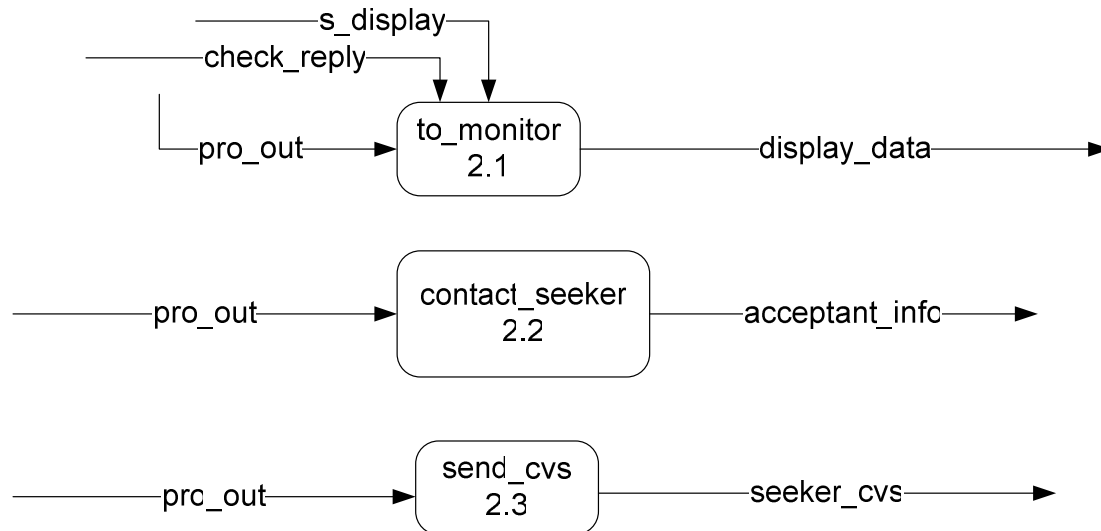
j. DFD : Level 3 provider_process 2.2



k. DFD : Level 3 admin_process 2.3



I. DFD : Level 2 Process Output 1.5



2.3.3 Data Dictionary

Data Dictionary for Data Flow Diagrams of our Project.

Name	S_info
Input	Our Project
Output	Job Seeker
Description	Information of job seeker for sign up and login

Name	P_info
Input	Our Project
Output	Job Provider
Description	Information of job provider for sign up and login

Name	Ad_info
Input	Our Project
Output	Admin
Description	Information of Administrator user for sign up and login

Name	seeker_cvs
Input	Job Provider
Output	Process Output
Description	CVs of job seekers

Name	Display_data
Input	Monitor
Output	Process Output
Description	Displayed data on the monitor

Name	acceptant_info
Input	Job Seeker
Output	Process Output
Description	Information or email telling job seeker that the company accepts or offers a position to the him/her.

Name	career_in
Input	Career
Output	Process for users
Description	Information sent to the career web pages.

Name	career_out
Input	Process for users
Output	Career
Description	Accessed information from the career web pages.

Name	Sign_up_info
Input	Data for user
Output	Sign Up
Description	Sign up information sent to database

Name	check_info
Input	Checking
Output	Sign Up
Description	Information to check that user is already existing in database

Name	reply_check
Input	Sign Up
Output	Checking
Description	Result of the checking process

Name	s_display
Input	Sign Up
Output	Process Output
Description	Result of sign up process to be displayed in the monitor

Name	Data_out
Input	Data for users
Output	Checking
Description	Data selected from the database to do checking

Name	access_data
Input	Process for user

Output	Data for users
Description	Accessed data from the database such as current user information, search result, etc

Name	update_data
Input	Data for users
Output	Process for user
Description	Data to be updated in the database

Name	c_user
Input	Process for user
Output	Checking
Description	Pointer of the user

Name	check_reply
Input	Process output
Output	Checking
Description	Result of the checking process to be displayed in the monitor

Name	pro_out
Input	Process for user
Output	Process output
Description	Data to be shown in the monitor and to be sent to the user

2.3.4 Database Design

Below are our database tables, data attributes and sql statements.

Career_Site

Data	Type and Size	Format
id	INTEGER	Number
site_name	VARCHAR - 60	Text

Career_Site.sql

```
CREATE TABLE Career_Site (
```

```
id          INTEGER NOT NULL,  
site_name   VARCHAR(60) NOT NULL,
```

```
PRIMARY KEY(id),  
FOREIGN KEY (id) REFERENCES Job_Provider (id),  
);
```

Job Seeker

Data	Type and Size	Format
id	INTEGER	Number
name	VARCHAR – 20	Text
last_name	VARCHAR -10	Text
username	VARCHAR – 20	Text
password	VARCHAR – 20	Text
e-mail_1	VARCHAR – 30	Text
e-mail_2	VARCHAR – 30	Text
father_name	VARCHAR – 30	Text
birthdate	DATETIME	Date/Time
secret_question	VARCHAR – 100	Text
secret_answer	VARCHAR – 100	Text
employer_id	INTEGER	Number

Job_Seeker.sql

```
CREATE TABLE Job_Seeker (
```

```
id          INTEGER NOT NULL,  
name        VARCHAR(20) NOT NULL,  
last_name   VARCHAR(10) NOT NULL,  
username    VARCHAR(20) NOT NULL,  
password    VARCHAR(20) NOT NULL,  
e-mail_1    VARCHAR(30) NOT NULL,
```

e-mail_2 VARCHAR(20),
 father_name VARCHAR(30) NOT NULL,
 birthdate TIMESTAMP NOT NULL,
 secret_question VARCHAR(100) NOT NULL,
 secret_answer VARCHAR(100) NOT NULL,
 employer_id INTEGER,

PRIMARY KEY(id),

);

Job Provider

Data	Type and Size	Format
id	INTEGER	Number
career_site_id	INTEGER	Number
company_name	VARCHAR – 30	Text
applicant_full_name	VARCHAR – 30	Text
applicant_position	VARCHAR – 30	Text
phone_no	VARCHAR – 20	Text
country	VARCHAR – 20	Text
province	VARCHAR – 20	Text
city	VARCHAR – 20	Text
district	VARCHAR – 20	Text
address	VARCHAR – 60	Text
sector	VARCHAR – 20	Text
employee_no	INTEGER	Number
username	VARCHAR – 20	Text
password	VARCHAR – 20	Text
secret_question	VARCHAR – 100	Text
secret_answer	VARCHAR – 100	Text

Job_Provider.sql

CREATE TABLE Job_Provider (

id INTEGER NOT NULL,
 career_site_id INTEGER,
 company_name VARCHAR(30) NOT NULL,
 applicant_full_name VARCHAR(30) NOT NULL,
 applicant_position VARCHAR(30) NOT NULL,
 phone_no VARCHAR(20) NOT NULL,
 country VARCHAR(20) NOT NULL,
 province VARCHAR(20),
 city VARCHAR(30) NOT NULL,
 district VARCHAR(30),
 address VARCHAR(60) NOT NULL,
 sector VARCHAR(20) NOT NULL,
 employee_no INTEGER,
 username VARCHAR(20) NOT NULL,

password VARCHAR(20) NOT NULL,
secret_question VARCHAR(100) NOT NULL,
secret_answer VARCHAR(100) NOT NULL,

PRIMARY KEY(id),

);

Personal Information

Data	Type and Size	Format
seeker_id	INTEGER	Number
gender	VARCHAR – 5	Text
phone_no	VARCHAR – 20	Text
mobile_no.	VARCHAR – 20	Text
tc_no	INTEGER	Number
military_status	VARCHAR – 10	Text
marital_status	VARCHAR – 10	Text
country	VARCHAR – 20	Text
province	VARCHAR – 20	Text
district	VARCHAR – 20	Text

Personal_Information.sql

CREATE TABLE Personal_Information (

seeker_id INTEGER NOT NULL,
gender VARCHAR(5) NOT NULL,
phone_no VARCHAR(20) NOT NULL,
mobile_no VARCHAR(20) NOT NULL,
tc_no INTEGER NOT NULL,
military_status VARCHAR(10) NOT NULL,
marital_status VARCHAR(10) NOT NULL,
country VARCHAR(20) NOT NULL,
province VARCHAR(20),
district VARCHAR(20) NOT NULL,

PRIMARY KEY(seeker_id),

FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Educational Information

Data	Type and Size	Format
seeker_id	INTEGER	Number
education_level	VARCHAR – 20	Text
status	VARCHAR – 20	Text

graduation_date	DATETIME	Date/Time
department	VARCHAR – 20	Text
school	VARCHAR – 30	Text

Educational_Information.sql

CREATE TABLE Educational_Information (

seeker_id INTEGER NOT NULL,
education_level VARCHAR(20) NOT NULL,
status VARCHAR(20) NOT NULL,
graduation_date TIMESTAMP NOT NULL,
department VARCHAR(20) NOT NULL,
school VARCHAR(30) NOT NULL,

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Work Experience Information

Data	Type and Size	Format
seeker_id	INTEGER	Number
company_name	VARCHAR – 30	Text
business_sector	VARCHAR – 20	Text
work_area	VARCHAR – 20	Text
city_country	VARCHAR – 40	Text
start_date	DATETIME	Date/Time
end_date	DATETIME	Date/Time

Work_Experience_Information.sql

CREATE TABLE Work_Experience_Information (

seeker_id INTEGER NOT NULL,
company_name VARCHAR(30),
business_sector VARCHAR(20),
work_area VARCHAR(20),
city_country VARCHAR(40),
start_date TIMESTAMP,
end_date TIMESTAMP,

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Language Information

Data	Type and Size	Format
seeker_id	INTEGER	Number
language_name	VARCHAR – 40	Text
reading_level	VARCHAR – 10	Text
writing_level	VARCHAR – 10	Text
speaking_level	VARCHAR – 10	Text
place_learned	VARCHAR – 20	Text

Language_Information.sql

CREATE TABLE Language_Information (

seeker_id INTEGER NOT NULL,
language_name VARCHAR(40) NOT NULL,
reading_level VARCHAR(10) NOT NULL,
writing_level VARCHAR(10) NOT NULL,
speaking_level VARCHAR(10) NOT NULL,
place_learned VARCHAR(20),

PRIMARY KEY(seeker_id),

FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Computer Programs

Data	Type and Size	Format
seeker_id	INTEGER	Number
program_name	VARCHAR – 20	Text
knowledge_level	VARCHAR – 10	Text

Computer_Programs.sql

CREATE TABLE Computer_Programs (

seeker_id INTEGER NOT NULL,
program_name VARCHAR(20) NOT NULL,
knowledge_level VARCHAR(10) NOT NULL,

PRIMARY KEY(seeker_id),

FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Programming Languages

Data	Type and Size	Format
seeker_id	INTEGER	Number
language_name	VARCHAR – 20	Text
knowledge_level	VARCHAR – 10	Text

Programming_Languages.sql

CREATE TABLE Programming_Languages (

seeker_id INTEGER NOT NULL,
language_name VARCHAR(20),
knowledge_level VARCHAR(10),

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Operating Network Systems

Data	Type and Size	Format
seeker_id	INTEGER	Number
system_name	VARCHAR – 20	Text
knowledge_level	VARCHAR – 10	Text

Operating_Network_Systems.sql

CREATE TABLE Operating_Network_Systems (

seeker_id INTEGER NOT NULL,
system_name VARCHAR(20),
knowledge_level VARCHAR(10),

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Databases

Data	Type and Size	Format
seeker_id	INTEGER	Number
DBMS_name	VARCHAR – 20	Text
knowledge_level	VARCHAR – 10	Text

Databases.sql

CREATE TABLE Databases (

seeker_id INTEGER NOT NULL,
DBMS_name VARCHAR(20),
knowledge_level VARCHAR(10),

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Institutional Software

Data	Type and Size	Format
seeker_id	INTEGER	Number
software_name	VARCHAR – 20	Text
knowledge_level	VARCHAR – 10	Text

Institutional_Software.sql

CREATE TABLE Institutional_Software (

seeker_id INTEGER NOT NULL,
software_name VARCHAR(20),
knowledge VARCHAR(10),

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

Office Utilities

Data	Type and Size	Format
seeker_id	INTEGER	Number
utility_name	VARCHAR – 20	Text

Office_Uilities.sql

CREATE TABLE Office_Uilities (

seeker_id INTEGER NOT NULL,
utility_name VARCHAR(20),

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),
);

Certificate Information

Data	Type and Size	Format
seeker_id	INTEGER	Number
certificate_name	VARCHAR – 30	Text
date	DATETIME	Date/Time
institution	VARCHAR – 20	Text

Certificate_Information.sql

CREATE TABLE Certificate_Information (

seeker_id INTEGER NOT NULL,
certificate_name VARCHAR(30),
date TIMESTAMP,
institution VARCHAR(20),

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),
);

Seminar Information

Data	Type and Size	Format
seeker_id	INTEGER	Number
certificate_name	VARCHAR – 30	Text
date	DATETIME	Date/Time
institution	VARCHAR – 20	Text

Seminar_Information.sql

CREATE TABLE Seminar_Information (

seeker_id INTEGER NOT NULL,
certificate_name VARCHAR(30),
date TIMESTAMP,
institution VARCHAR(20),

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),
);

Examinations

Data	Type and Size	Format
seker_id	INTEGER	Number
examination_name	VARCHAR – 20	Text
examination_grade	VARCHAR – 5	Text
institution	VARCHAR – 20	Text
date	DATETIME	Date/Time

Examinations.sql

```
CREATE TABLE Examinations (

seeker_id    INTEGER NOT NULL,
examination_name  VARCHAR(20),
examination_grade  VARCHAR(5),
institution    VARCHAR(20),
date          TIMESTAMP,

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);
```

References

Data	Type and Size	Format
seeker_id	INTEGER	Number
reference_name	VARCHAR – 20	Text
reference_last_name	VARCHAR – 10	Text
company	VARCHAR – 30	Text
position	VARCHAR – 30	Text
phone	VARCHAR – 20	Text
e-mail	VARCHAR – 30	Text
reference_type	VARCHAR – 20	Text

References.sql

```
CREATE TABLE References (
```

```

seeker_id          INTEGER NOT NULL,
reference_name     VARCHAR(20),
reference_last_name VARCHAR(10),
company           VARCHAR(30),
position          VARCHAR(30),
phone             VARCHAR(20),
e-mail            VARCHAR(30),
reference_type     VARCHAR(20),

PRIMARY KEY(seeker_id),
FOREIGN KEY (seeker_id) REFERENCES Job_Seeker (id),

);

```

XML Schema Definitions

seekers.xsd:

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:annotation>
        <xsd:documentation>
            This document is the XML Schema of a Job Seeker
        </xsd:documentation>
    </xsd:annotation>

    <xsd:element name="Seeker">
        <xsd:element name="id" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="name" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="last_name" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="username" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="password" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="email1" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="email2" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="fathername" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="birthdate" type="xsd:date" minOccurs="1"
maxOccurs="1"/>
    </xsd:element>
</xsd:schema>

```

```

        <xsd:element name="secret_question" type="xsd:string"
minOccurs="1" maxOccurs="1"/>
        <xsd:element name="secret_answer" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </xsd:element>
</xsd:schema>

```

providers.xsd:

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:annotation>
        <xsd:documentation>
            This document is the XML Schema of a Job Provider
        </xsd:documentation>
    </xsd:annotation>

    <xsd:element name="Seeker">
        <xsd:element name="id" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="company_name" type="xsd:string"
minOccurs="1" maxOccurs="1"/>
        <xsd:element name="applicant_name" type="xsd:string"
minOccurs="1" maxOccurs="1"/>
        <xsd:element name="applicant_position" type="xsd:string"
minOccurs="1" maxOccurs="1"/>
        <xsd:element name="phone" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="country" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="province" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="city" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="district" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="address" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="sector" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="employee#" type="xsd:integer" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="username" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="password" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="secret_question" type="xsd:string"
minOccurs="1" maxOccurs="1"/>
        <xsd:element name="secret_answer" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </xsd:element>
</xsd:schema>

```

cv.xsd:

```

<?xml version="1.0" encoding="UTF-8"?>

```



```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation>
      This document is the XML Schema of a CV
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="CV">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="personal" ref="personalType"/>
        <xsd:element name="school" ref="schoolType"
maxOccurs="unbounded"/>
        <xsd:element name="company" ref="companyType"
maxOccurs="unbounded" />
        <xsd:element name="language" ref="languageType"
maxOccurs="unbounded"/>
        <xsd:element name="program" ref="programType"
maxOccurs="unbounded"/>
        <xsd:element name="pl" ref="plType" maxOccurs="unbounded"/>
        <xsd:element name="dbms" ref="dbmsType"
maxOccurs="unbounded"/>
        <xsd:element name="software" ref="softwareType"
maxOccurs="unbounded"/>
        <xsd:element name="certificate" ref="certificateType"
maxOccurs="unbounded"/>
        <xsd:element name="utility" type="xsd:string"
maxOccurs="unbounded"/>
        <xsd:element name="os" ref="osType" maxOccurs="unbounded"/>
        <xsd:element name="seminar" ref="seminarType"
maxOccurs="unbounded"/>
        <xsd:element name="reference" ref="referenceType"
maxOccurs="unbounded"/>
        <xsd:element name="examination" ref="examinationType"
maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="personalType">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="gender" type="xsd:string" maxOccurs="1"/>
        <xsd:element name="phone" type="xsd:string" maxOccurs="1"/>
        <xsd:element name="mobile_phone" type="xsd:string"
maxOccurs="1"/>
        <xsd:element name="tc_identity#" type="xsd:string"
maxOccurs="1"/>
        <xsd:element name="marital_status" type="xsd:string"
maxOccurs="1"/>
        <xsd:element name="country" type="xsd:string" maxOccurs="1"/>
        <xsd:element name="province" type="xsd:string" maxOccurs="1"/>
        <xsd:element name="district" type="xsd:string" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="schoolType">
    <xsd:complexType>
      <xsd:sequence>

```

```

        <xsd:element name="school_level" type="xsd:string"
maxOccurs="1"/>
        <xsd:element name="status" type="xsd:string" maxOccurs="1"/>
        <xsd:element name="graduation_date" type="xsd:date"
maxOccurs="1"/>
        <xsd:element name="department" type="xsd:string" maxOccurs="1"/>
        <xsd:element name="school" type="xsd:string" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>
</xsd:element>

<xsd:element name="companyType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="company_name" type="xsd:string"
maxOccurs="1"/>
            <xsd:element name="business_sector" type="xsd:string"
maxOccurs="1"/>
            <xsd:element name="work_area" type="xsd:string" maxOccurs="1"/>
            <xsd:element name="city/country" type="xsd:string"
maxOccurs="1"/>
            <xsd:element name="start_date" type="xsd:date" maxOccurs="1"/>
            <xsd:element name="end_date" type="xsd:date" maxOccurs="1"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="languageType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="language_name" type="xsd:string"
maxOccurs="1"/>
            <xsd:element name="reading_level" type="xsd:integer"
maxOccurs="1"/>
            <xsd:element name="writing_level" type="xsd:integer"
maxOccurs="1"/>
            <xsd:element name="speaking_level" type="xsd:integer"
maxOccurs="1"/>
            <xsd:element name="place_learned" type="xsd:string"
maxOccurs="1"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="programType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="program_name" type="xsd:string"
maxOccurs="1"/>
            <xsd:element name="program_level" type="xsd:integer"
maxOccurs="1"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="plType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="pl_name" type="xsd:string" maxOccurs="1"/>
            <xsd:element name="pl_level" type="xsd:integer" maxOccurs="1"/>

```

```

        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>

    <xsd:element name="dbmsType">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="dbms_name" type="xsd:string"
maxOccurs="1"/>
          <xsd:element name="dbms_level" type="xsd:integer"
maxOccurs="1"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>

    <xsd:element name="softwareType">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="software_name" type="xsd:string"
maxOccurs="1"/>
          <xsd:element name="software_level" type="xsd:integer"
maxOccurs="1"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>

    <xsd:element name="osType">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="os_name" type="xsd:string" maxOccurs="1"/>
          <xsd:element name="os_level" type="xsd:integer" maxOccurs="1"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>

    <xsd:element name="certificateType">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="certificate_name" type="xsd:string"
maxOccurs="1"/>
          <xsd:element name="certificate_institution" type="xsd:string"
maxOccurs="1"/>
          <xsd:element name="certificate_date" type="xsd:date"
maxOccurs="1"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>

    <xsd:element name="seminarType">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="seminar_name" type="xsd:string"
maxOccurs="1"/>
          <xsd:element name="seminar_institution" type="xsd:string"
maxOccurs="1"/>
          <xsd:element name="seminar_duration" type="xsd:integer"
maxOccurs="1"/>
          <xsd:element name="seminar_date" type="xsd:date"
maxOccurs="1"/>
        </xsd:sequence>

```

```

        </xsd:complexType>
    </xsd:element>

    <xsd:element name="referenceType">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="reference_name" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="reference_lastname" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="reference_company" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="reference_position" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="reference_phone" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="reference_email" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="reference_type" type="xsd:string"
maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="examinationType">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="examination_name" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="examination_institution" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="examination_grade" type="xsd:string"
maxOccurs="1"/>
                <xsd:element name="examination_date" type="xsd:date"
maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:schema>

```

polls.xsd:

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:annotation>
        <xsd:documentation>
            This document is the XML Schema of a Poll
        </xsd:documentation>
    </xsd:annotation>

    <xsd:element name="Poll">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="vote" ref="Votes" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

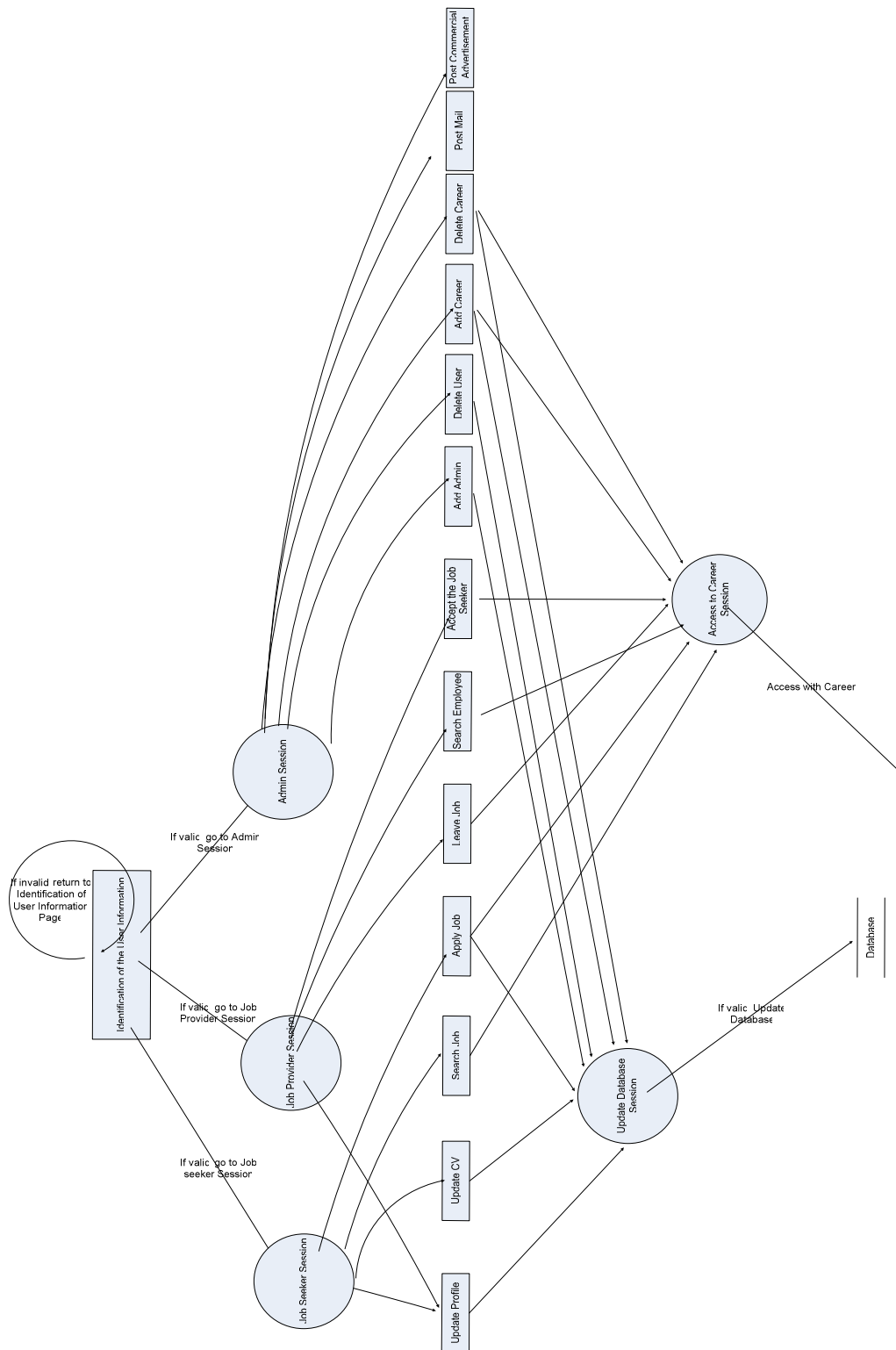
```

```
</xsd:element>

<xsd:complexType name="Votes">
  <xsd:sequence>
    <xsd:element name="id" type="xsd:string" maxOccurs="1"/>
    <xsd:element name="votestring" type="xsd:string" />
  </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

2.4 Behavioral Design

2.4.1 State Transition Diagram



As you can see in the above figure, there are two main session in the system, inactive and active sessions. First of all, the user have to create account and identify the user name and password to log in to the system. If the identification is invalid, the user can not pass to active system and the system is still in inactive status and waiting for the correct identification. If the system gets the valid identification, then it lets the user pass to the active status.

There are three main sessions in active system, “Job Seeker Session”, “Job Provider Session” and “Admin Session”. The system can specify the type of the user by the input identification value and allow the access to the appropriate functions according to the user type. As shown in the figure, job seeker can update profile, update cv, apply job and search job , while job provider can update profile, leave job and search employee and accept job seeker. The administrator can add new admin user to the system, delete user, add source career web pages and delete these pages, post mail to all kinds of users and post commercial advertisements on to portal. Since these functions are cooperating with database and career web pages, so next sessions are update database and access to career web page sessions. In the update database, if the system gets the valid input and valid command, it will access to the database of the portal and do the appropriate work. If the system gets the valid command and input to access the career web site, then it will connect to the career web page services and do the appropriate job.

3. SYSTEM DESIGN

3.1 Use Case Diagrams and Description

We have grouped use cases according to our user groups : Job Seekers and Job Providers and our System Administrators.

Job Seeker Use Cases

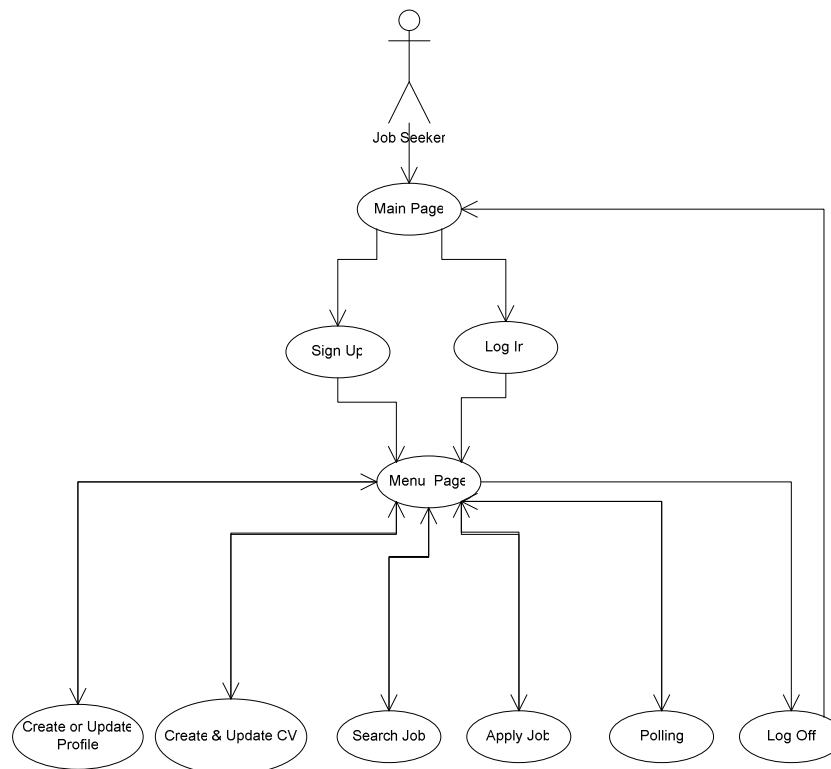
The set of activities or functions performed by a job seeker on our portal are listed below.

A job Seeker can ,

- Sign up and then log in to our portal.
- Log off from system.
- Create a profile and update it in time.

- Create or upload CV in any format (video , .doc , standard form)
- Search through job offers
- Apply to a position
- Take our evaluation polls

Use Case Diagram for Job Seeker on our Portal



Job Provider Use Cases

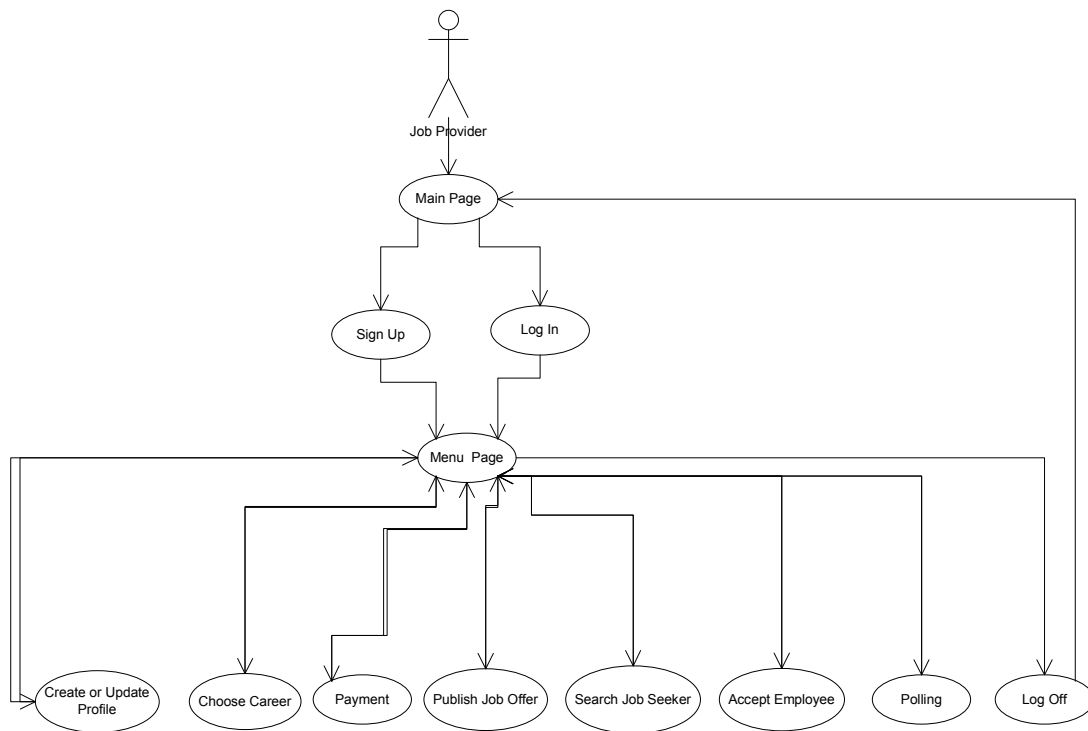
The activities of other user group of our portal , Job Providers can be listed as ;

A Job Provider can

- Sign up to our portal by paying the membership fee.
- Log off from system.
- Log in and then create a profile with necessary information.
- Update profile informations

- Choose career web site to become a member and publish job offers.
- Pay the fee of the career web site that is chosen.
- Search through job seekers to find appropriate employer.
- See job seekers applied to their published job offers.

Use Case Diagram for Job Provider on our Portal



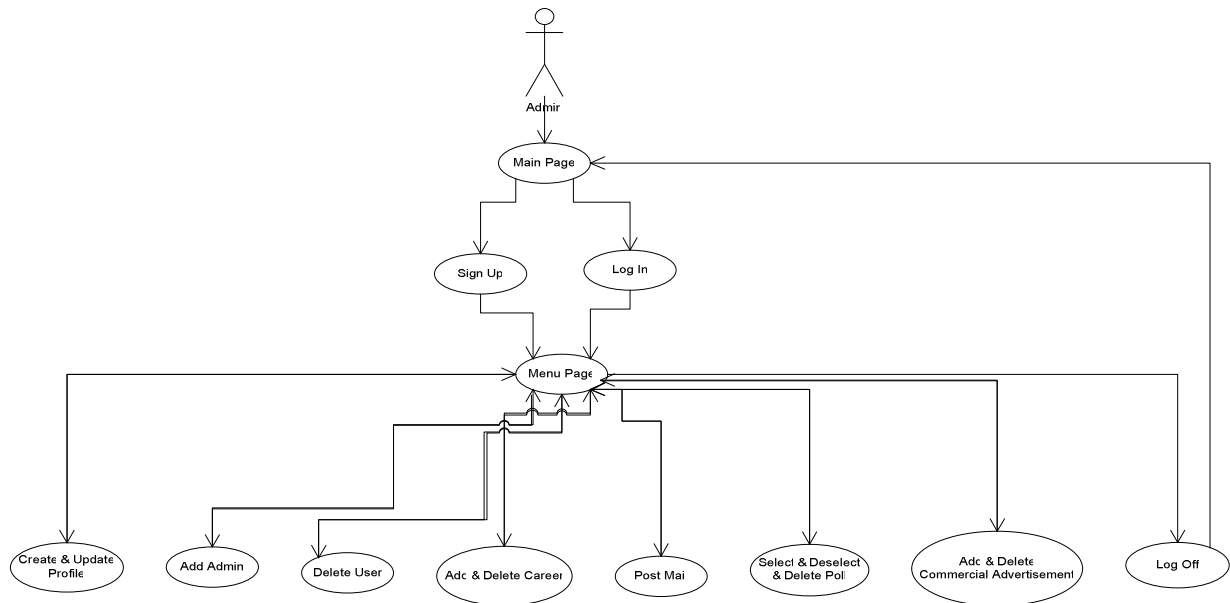
System Administrator Use Cases:

The activities of this user group of our portal can be listed as ;

A System Administrator can

- Sign up (this action is done by other administrator at first) and Log in to system.
- Log off from system.
- Create or update profile.
- Add (any system administrator) / Delete user.
- Add / Delete member career web pages.
- Post announcement, information or feedback mails to users.
- Manage polls.

Use Case Diagram for System Administrators on our Portal



Use case: Sign up and Log in

Primary Actors: Job Seekers, Job Providers, System Administrators

This use case applies to all user groups. Job seekers, job providers and administrators have to sign up to use our portal. We provide each group a different sign up module. Administrators' sign up process is done by one of the other system administrators. A job seeker needs to fill in his/her user id, password, personal identification number, e-mail address and an answer to secret question for security issues. However, job providers have to enter their tax id besides the company user id, password, e-mail address and a secret question. It is necessary to pay the membership fee to finalize the sign up process for companies.

For log in process all user groups has to enter correct user ids and passwords to the spaces provided if any of these values unmatched with the ones provided during sign up process, a warning will appear on the screen.

Use case: Log off from System

Primary Actors: Job Seekers ,Job Providers, System Administrators

For logging of from the sytem all user have to click on log out button provided on each of the pages that they are using at that moment. By clicking the log out button , the users will lead to the main page of our system.

Use Case: Create and/or Update Profile

Primary Actors: Job Seekers, Job Providers, System Administrators

After signing up to our portal, all user groups have had a profile which they can change or update whenever they need or want. On a job seeker's profile, we can see their actual names, surnames, mother's surnames (for security issues), if they have uploaded any CVs and if they have, we can see these documents. Job seekers can change their user ids or passwords through their profiles. We can also see their situation whether they applied to a job, they are still searching or their positioning has occurred. Job seekers can see all the job offers that they apply or want to follow through their profiles.

Job providers' profile holds their company informations, user ids, passwords, they can change these informations any time they need or want. And also they can see the list of career web pages that they have a membership on, their payment situations on both career web pages and our portal. The job offers that are being published on any career web site and their previous job offers.

System Administrators profile holds their personal information , name, surname , address, phone number and user ids and passwords. They can change their personal information whenever they need or want but for changing their user ids and passwords on system they need an aproval from one of other system administrators.

Use case: Create and/or Upload CVs

Primary Actors: Job Seekers

Job seeker users of our portal can either create their CVs by filling in standard form that we provide on our portal or they can upload their CVs as documents in .pdf or .doc format or as video streams to one of the video upload web pages form the list that we have provided.

Use case: Search Job

Primary Actors: Job Seeker

We provide a job search module for job seekers where they can see resulting job offers coming from all career web pages that our portal cooperates. They can also see on which career web site the subjected offer is being published. The categories which jobs are grouped are mainly the country, the city, company, publish date, job area, departmant, position. All the

choices in each category will be listed in a drop down menu to make the searching process easy for job seekers. Also more detailed search parameters can be provided for users in detailed search screen like title, position level, education level, job experience, job style.

Use case: Apply Job

Primary Actors: Job Seeker

We will implement an application module which works in the same way as the source career web page of listed offers. We will provide the description of the job offers as they are published on the career web page and our users can apply to any of these offers through this module.

Use case: Polls

Primary Actors: Job Seeker, System Administrators

We will give polls to our job seeker users after their positioning occurs to get feedback from them about both our portal and our cooperate career web sites. These feedbacks help us updating our statistical data on career web pages and etc.

The management of these polls given to job seekers are made by our system administrators. They can update , add or delete polls and send them to job seekers.

Use case: Choose a Career Web Page

Primary Actors: Job Provider

Job provider users need to choose and then sign up to a career web page to publish their job offers on at least one of them. To make this easy we provide a selection module for them. We have compared and scored our cooperate career web pages on different categories such as their membership fees, extra services and the statistics of their preference by job seekers, numbers of positioned job seekers to published offers on their web sites on this module.

Use case: Pay the Membership fees

Primary Actors: Job Provider

A simple payment module will be built for both payment of our portal's membership fee and the transfer of payment fees of chosen career web pages. Job providers just transfer the total needed amount to become a member of both our portal and their selected career web

pages to the provided bank account for our system with the necessary information such as company name, company address, the respondents' name, company tax number etc. then the transformation of the amount that has to be paid to career web pages is going to be made by our portal's system administrators. Job providers will be informed about the situation on each step of this action.

Use case: Publish Job Offers on Career Web Pages

Primary Actors: Job Provider

After becoming a member of the chosen career web sites , job providers can publish their job offers on these web sites through our portal. We will provide a job offer form which is like the standard form of the actual career page that publishes the offer. Job providers need to fill in this form for each of their offers and set the date of publishment. After they approve all these information, their offer will be sent to career web page and published.

Use case: Search Employee

Primary Actors: Job Provider

Job providers can search employers for their vacant positions by this module on our portal. The categorization of job seekers can be made according to their experience on the job area, educational information, age, sex. Job providers only allowed to see job seekers who are also members of the career web pages on which they have membership. Job seekers' CVs and contact informations will be available for job providers through this module.

Use case: See Who Applied to an Offer

Primary Actors: Job Provider

Job providers can see who applied through our cooperate career web pages to any of their published offers by this module. The same detiled information available about job seekers applied to these offers on Search Employee Module will also be reached through this module.

Use case: Add (any system administrator) / Delete user

Primary Actors: System Administrators

System administrators can add any new administrators to the system and delete job seeker users who are not using their account for a long period in time.

Use case: Add / Delete member career web pages

Primary Actors: System Administrators

The membership of any career web page is approved by system administrators by controlling their payment status and their given necessary information. If there is any unsatisfied condition, at first a warning message is sent to these companies and the action is taken according to their responses. If any career web page wants to call off or do not renew their membership in a given time, system administrator will delete these job provider users from the system.

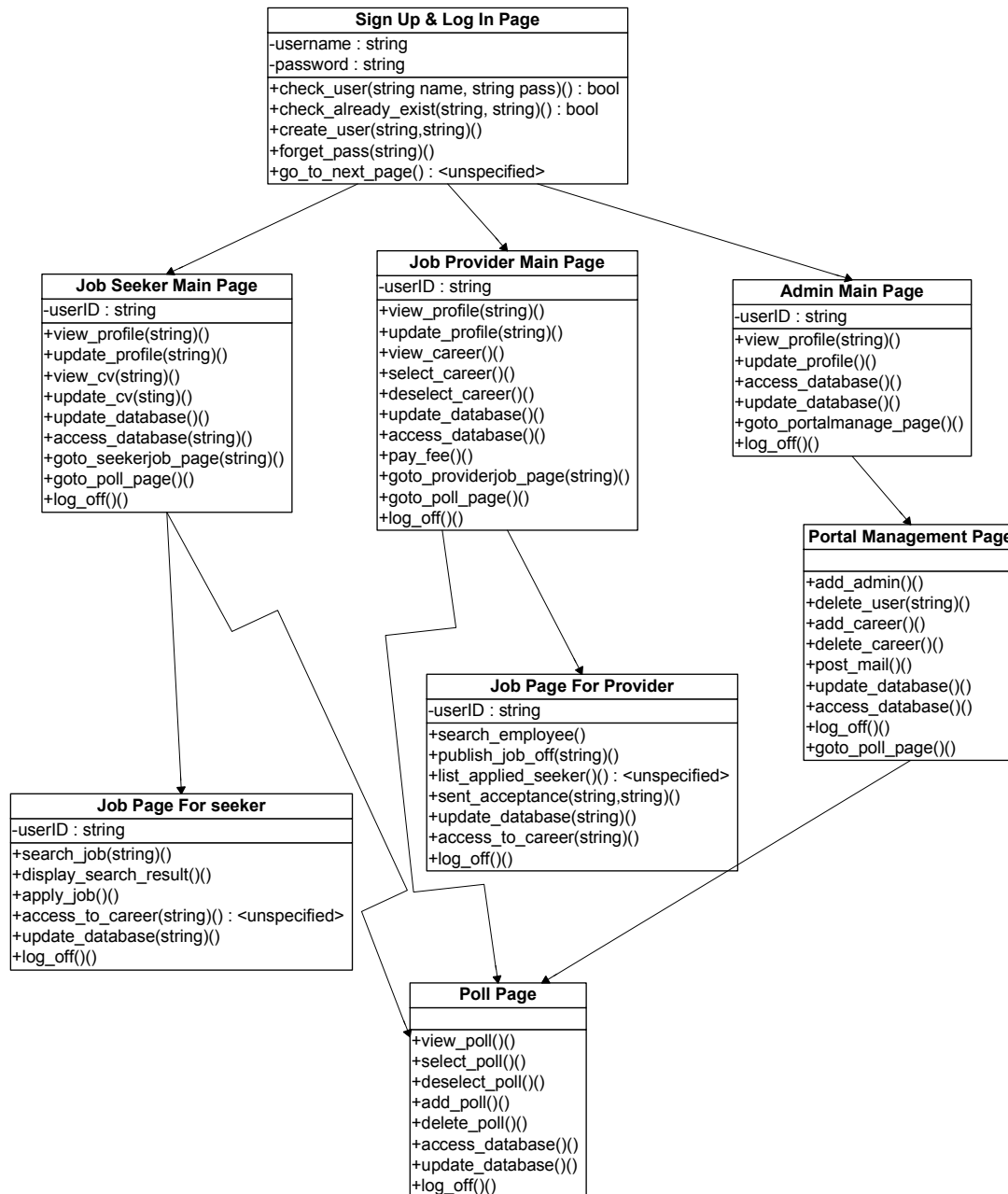
Use case: Post information, announcement, feedback mails to users**Primary Actors: System Administrators**

We have information and feedback modules for job seekers and job providers, these modules are invoked by our system administrators. They can send mails to all users to inform and get feedback from them. Any announcement about our portal or our source career web pages can be made through mails and seen on announcement section of our systems main page.

Use case: Post commercial adds on system**Primary Actors: System Administrators**

System administrators also manage the commercial adds on our portal, they can add, delete or update any commercial adds publishing on our system.

3.2 Class Diagram



This component of the system starts when the user identifies the correct username and password. As shown in the above figure, there are 8 classes working in the system and the details of the classes and attributes are as below:

Sign Up and Log In Page

In this page, the system will check the user's identification. If the user's information is not valid, then the system don't let the user to access the other components.

Attributes:

username :: string

password :: string

Methods:

check_user:

This method checks the identification of username and password. If the input identification is valid, then the system classifies the type of the user and then calls goto_next_page according to the user type. If the input information is not valid, then the user can not get in to the system.

create_user:

This method allows the user to create the new account on the system. It will take the detailed information of the user and check the user name whether it is conflicted with any of the already existing users by calling check_already_exist method. If the information is valid, then it creates a new account and calls goto_next_page to let the user continue on the system. If the information is invalid, then the system will ask for the appropriate username and password.

check_already_exist:

This method is called from create_user to check whether the user name is already existed in the system or not.

forget_pass:

When the user forgets his/her password, this method allows him/her to get his/her own password by checking the user's secret information according to the user's identification. If the user can give the correct information again, then the system tells his/her own password. If the user can not give information correctly, then the system doesn't allow him/her to get his/her password and get in to system.

goto_next_page:

This method allows the user to go to Job Seeker main page, Job Provider main page and Admin main page. By checking the type of the parameters and then allowing the user to continue to next page according to its type.

Job Seeker Main Page

Attributes:

userID :: string

Methods:

view_profile:

This method allows the user to check his/her profile, according to the userID. This method will call the access_database to get the correct information of the user which is stored in the system database.

update_profile:

This method allows the user to update his/her profile. This method will take appropriate identification of the user and update the current profile which is stored in the system database if the information is valid. This method will call update_database and update the current profile.

view_cv:

This method allows the user to check his/her CV if the CV is already stored in the database. It will call access_database to get the appropriate CV if he/she has already put the CV already.

update_cv:

This method allows the user to update the CV if he/she has already put the CV in the database. It takes appropriate information and update the CV by calling update_database method.

update_database:

This method updates the database according to the parameters. In Job Seeker main page, this method allows updating the profile and CV of the user.

access_database:

This method allows the system to access the database and get the appropriate results from the database. Here, it can access the profile and CV of the user.

goto_poll_page:

This method allows the user to connect to poll page.

goto_seekerjob_page:

This method allows the user to connect to seekerjob page.

log_off:

The system is logging off the user and send him/her to the main page of the system.

Job Provider Main Page**Attributes:**

userID :: string

Methods:**view_profile:**

This method allows the user to check his/her profile, according to the userID. This method will call the access_database to get the correct information of the user which is stored in the system database.

update_profile:

This method allows the user to update his/her profile. This method will take appropriate identification of the user and update the current profile which is stored in the system database if the information is valid. This method will call update_database and update the current profile.

select_career:

This method allows the job provider users to select the career web pages from the ones already in cooperation with our portal.

deselect_career:

This method allows the provider to deselect the career web page on which their job offers are already being published.

pay_fee:

This method checks the user's profiles whether the user already done the payment or not. If the payment is not done yet, the system will show the procedure of the payment process.

update_database:

This method updates the database according to the parameters. On Job Provider main page, this method allows updating of profile of the user.

access_database:

This method allows system to access the database and get the appropriate results from the database. Here, it can access the profile of the user.

goto_providerjob_page:

This method allows the user to connect to providerjob page.

goto_poll_page:

This method allows the user to connect to poll page.

log_off:

The system is logging off the user and send him/her to the main page of the system.

Admin Main Page**Attributes:**

userID :: string

Methods:**view_profile:**

This method allows the user to check his/her profile, according to the userID. This method will call the access_database to get the correct information of the user which is stored in the system database.

update_profile:

This method allows the user to update his/her profile. This method will take appropriate identification of the user and update the current profile which is stored in the

system database if the information is valid. This method will call `update_database` to update the current profile.

update_database:

This method updates the database according to the parameters. On this Admin main page, this method allows the updating of the profile of the admin user.

access_database:

This method allows the system to access the database and get the appropriate results from the database. Here, it can access the profile of the user.

goto_portalmanage_page:

This method allows the user to connect to portalmanage page.

log_off:

The system is logging off the user and send him/her to the main page of the system.

Job Page For Seeker

Attributes:

`userID :: string`

Methods:

search_job:

This method searches the job offers according to input keywords and other filters then calls the `display_search_result` function to display the available jobs' list.

display_search_result:

This method displays the list of available jobs according to the filtering options. User can choose how the listing occurs by this method.

apply_job:

This method allows the user to apply to the available job and the information is also updated in the system's database.

access_to_career:

This method is called in search_job and display_search_result, to get the appropriate information from the career web pages.

update_database:

This method will update the system's database.

log_off:

The system is logging off the the user and send him/her to the main page of the system.

Job Page For Provider**Attributes:**

userID :: string

Methods:**search_employee:**

This method searches the available job seeker through both our portal and the source career web pages that the employer company is a member of according to the filtering options.

publish_job_offer:

This method puts the job offer of the provider company on the career web page on which this provider company has a membership. This announcement will also be added to the system's database.

list_applied_seeker:

This method lists the seekers who applied to the published job offers of the provider company.

sent_acceptance:

This method is called when the provider accepts the applied job seeker and acceptant information is sent to the applied seeker.

update_database:

This method will update the system's database.

access_to_career:

This method is called in search_employee and publish_job_offer, to get the appropriate information from the career web pages.

log_off:

The system is logging off the user and send him/her to the main page of the system.

Poll Page**Methods:****view_poll:**

This method will show the selected poll by the job seeker and provider users which is stored in the system database.

add_poll:

This method is called by both job seeker and provider. The user can specify the appropriate result for the system, and can add their poll result.

select_poll:

This method is called by the administrator. The selected poll is shown in the system.

deselect_poll:

This method is called by the administrator. The deselected poll is not shown in the system.

delete_poll:

This method can only be called by the administrator user type. This method allows to delete the poll and its results.

update_database:

This method updates the database according to the parameters. This method is called from add_poll and delete_poll.

access_database:

This method allows the system to access the database and get the appropriate results from the database. Here, it is called from view_poll.

log_off:

The system is logging off the the user and send hime/her to the main page of the system.

Portal Management Page**Methods:****add_admin:**

The administrator can add the new administrator on the system and this method creates the new administrator user account.

delete_user:

The administrator can delete the any type of user on the system.

add_career:

This method adds new source career web page to the system.

delete_career:

This method deletes any source career web page from the system.

post_mail:

The method allows the administrator to send mail to all kinds of users on the system.

update_database:

This method updates the database according to the parameters. This method is called from add_admin, delete_user, add_career and delete_career.

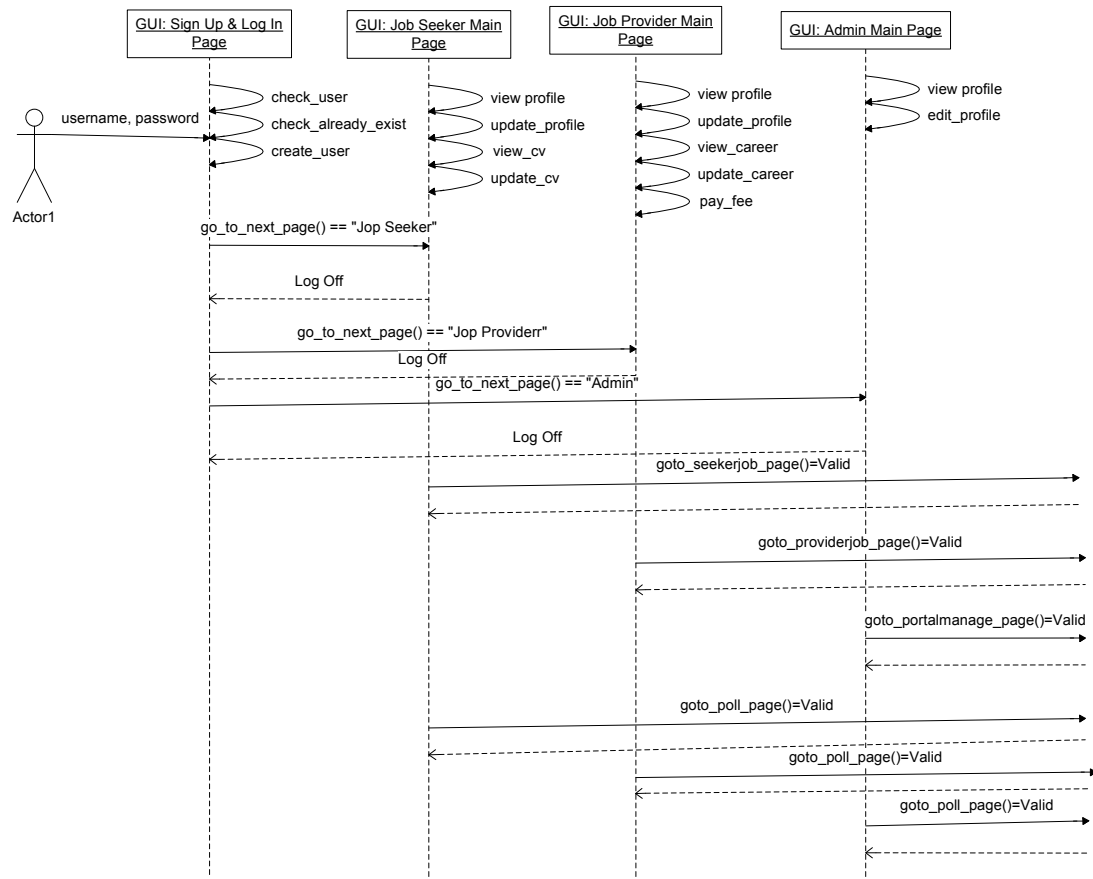
access_database:

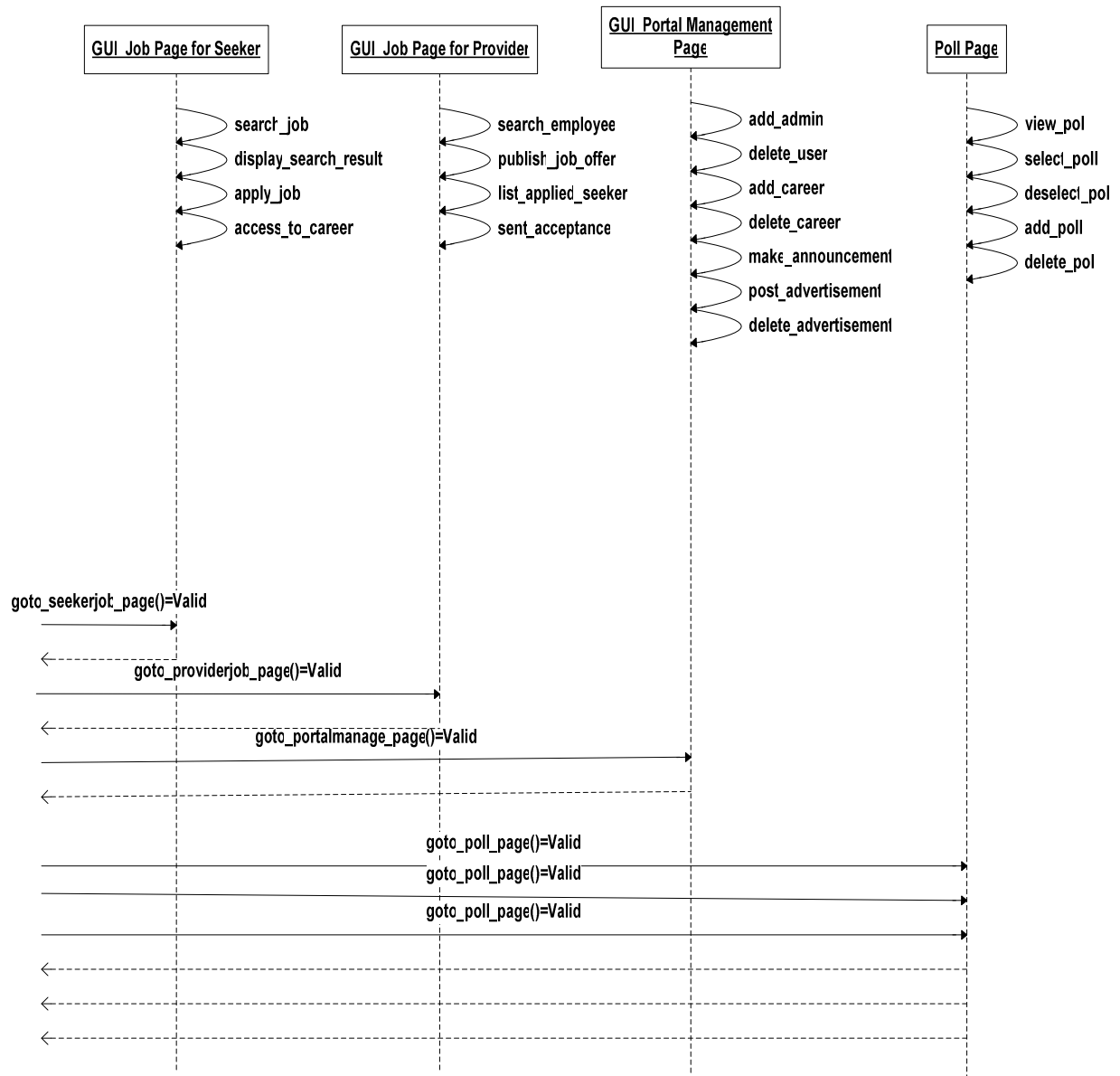
This method allows the system to access the database and get the appropriate results from the database.

log_off:

The system is logging off the the user and sent to the main page of the system.

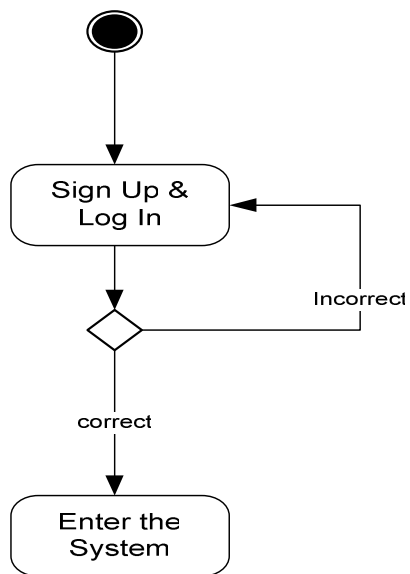
3.3 Sequence Diagrams



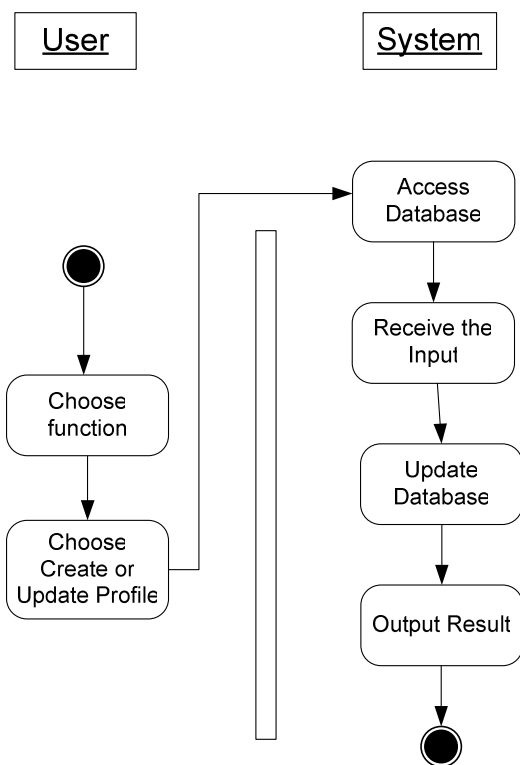


3.4 Activity Diagrams

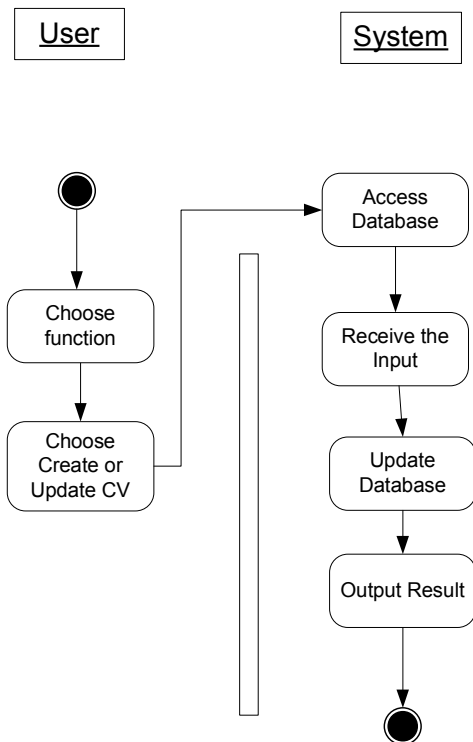
3.4.1 Sign Up and Log In Activity Diagram



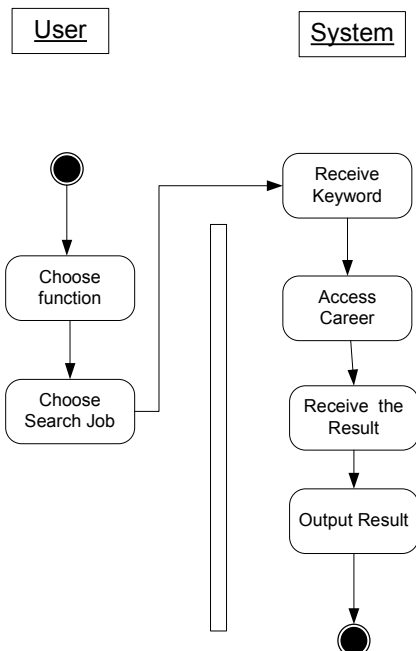
3.4.2 Create or Update Profile Activity Diagram



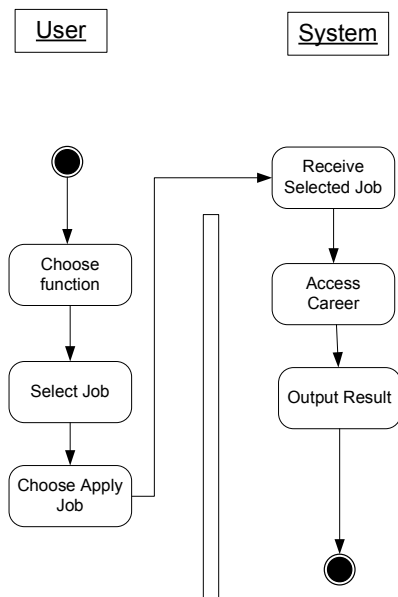
3.4.3 Create or Update CV Activity Diagram



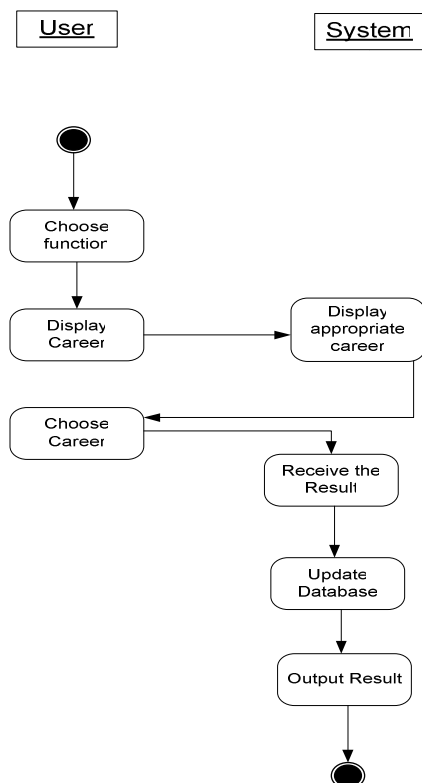
3.4.4 Search Job Activity Diagram



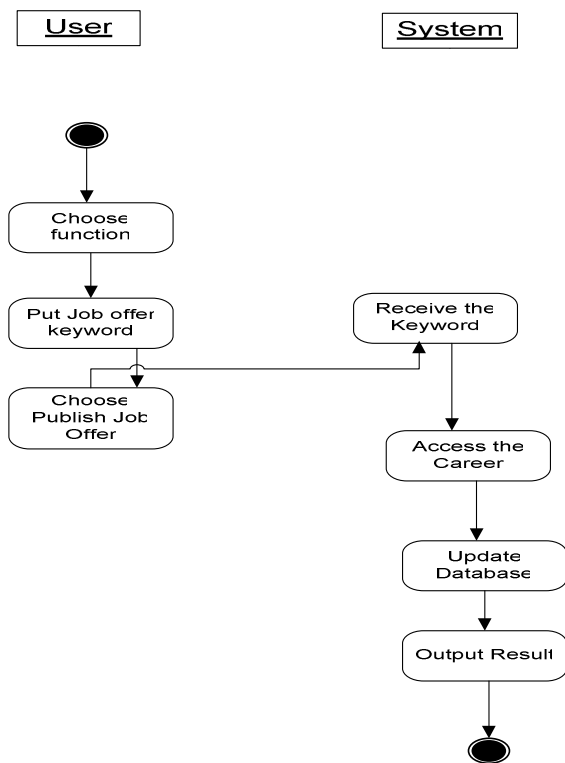
3.4.5 Apply Job Activity Diagram



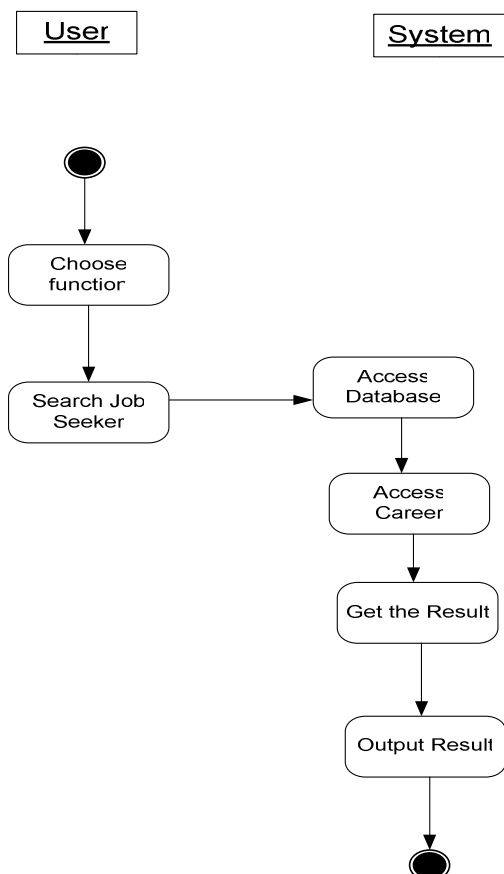
3.4.6 Choose Career Web Page Activity Diagram



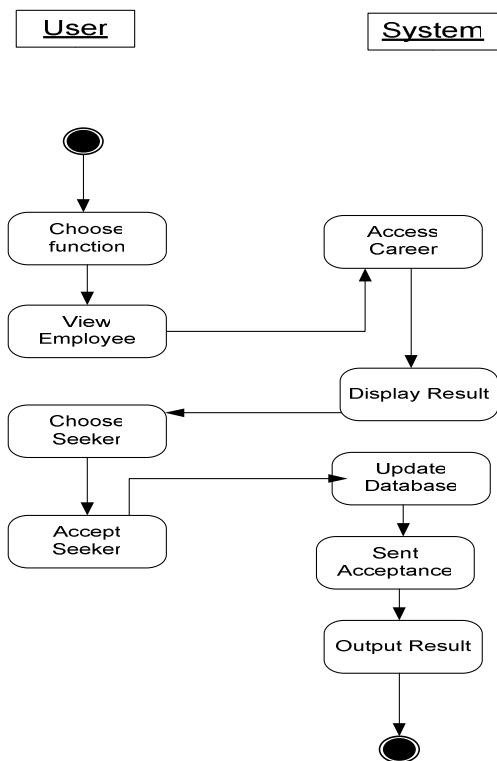
3.4.7 Publish Job Offer Activity Diagram



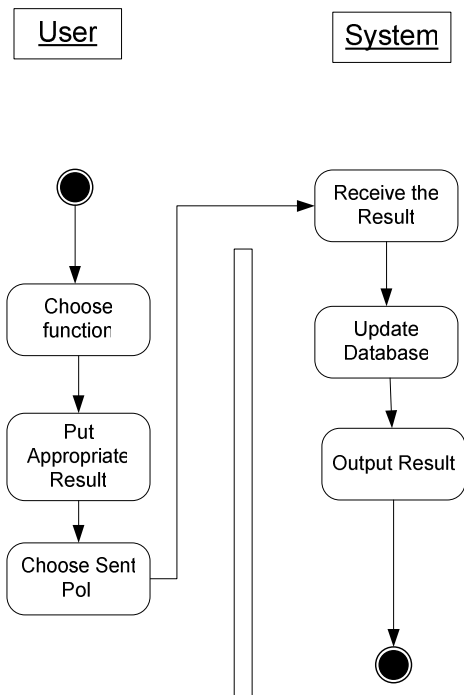
3.4.8 Search Job Seeker Activity Diagram



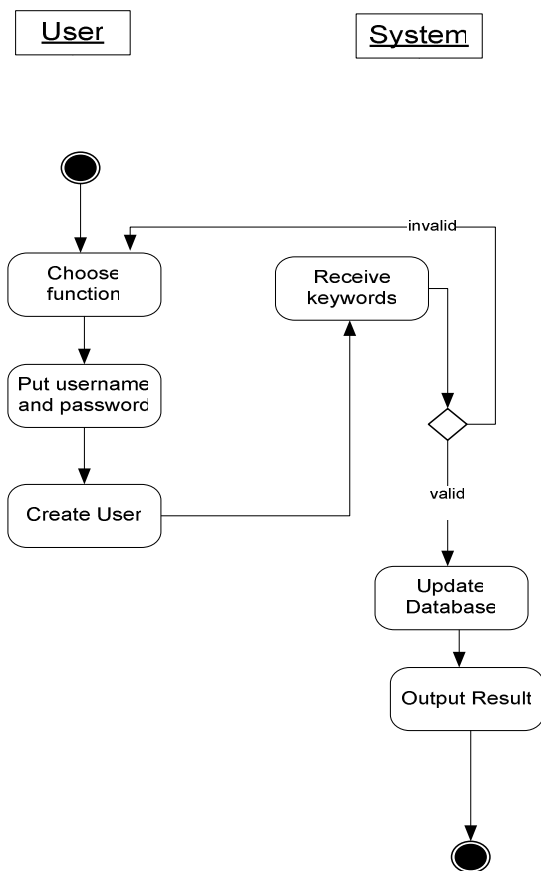
3.4.9 Accept Seeker Activity Diagram



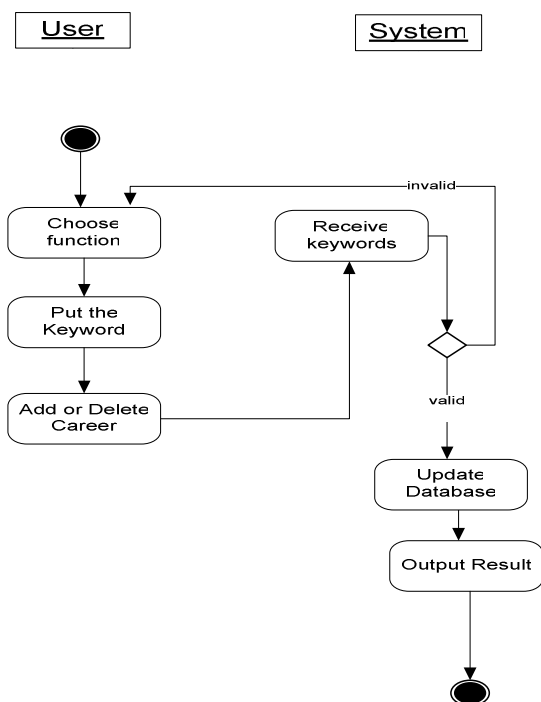
3.4.10 Sent Poll Activity Diagram



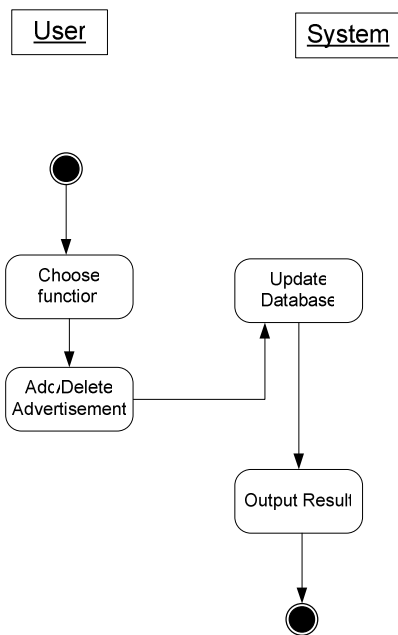
3.4.11 Create User Activity Diagram



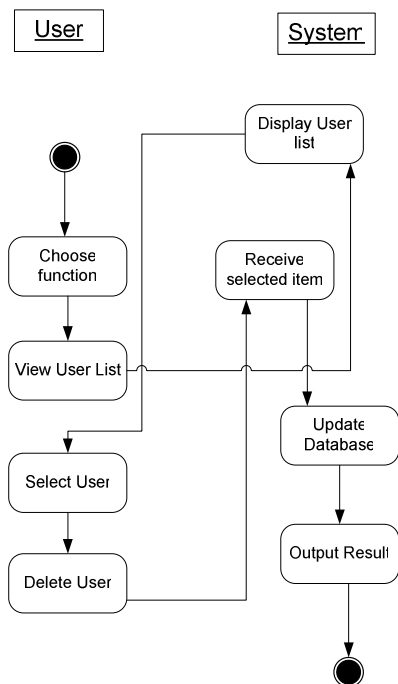
3.4.12 Add or Delete Career Activity Diagram



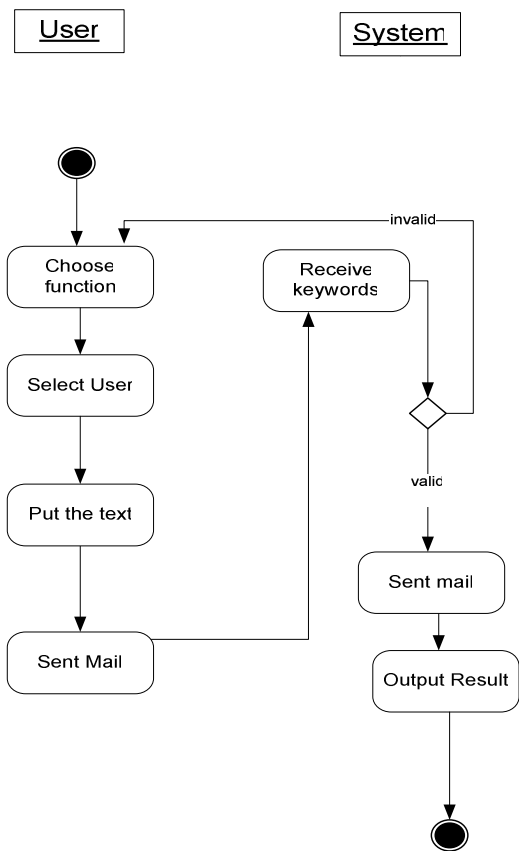
3.4.13 Add or Delete Advertisement Activity Diagram



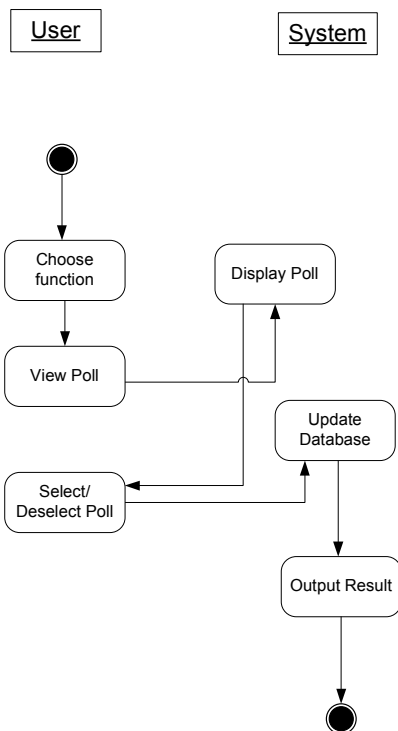
3.4.14 Delete User Activity Diagram



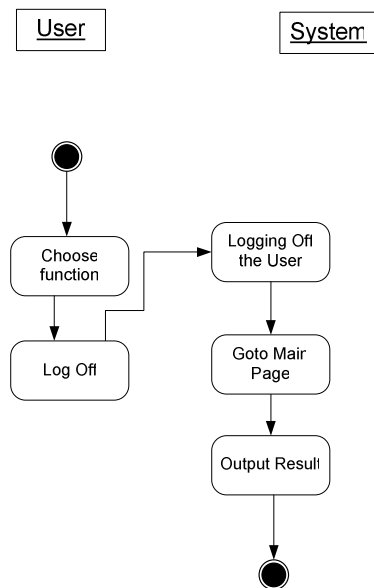
3.4.15 Sent Mail Activity Diagram



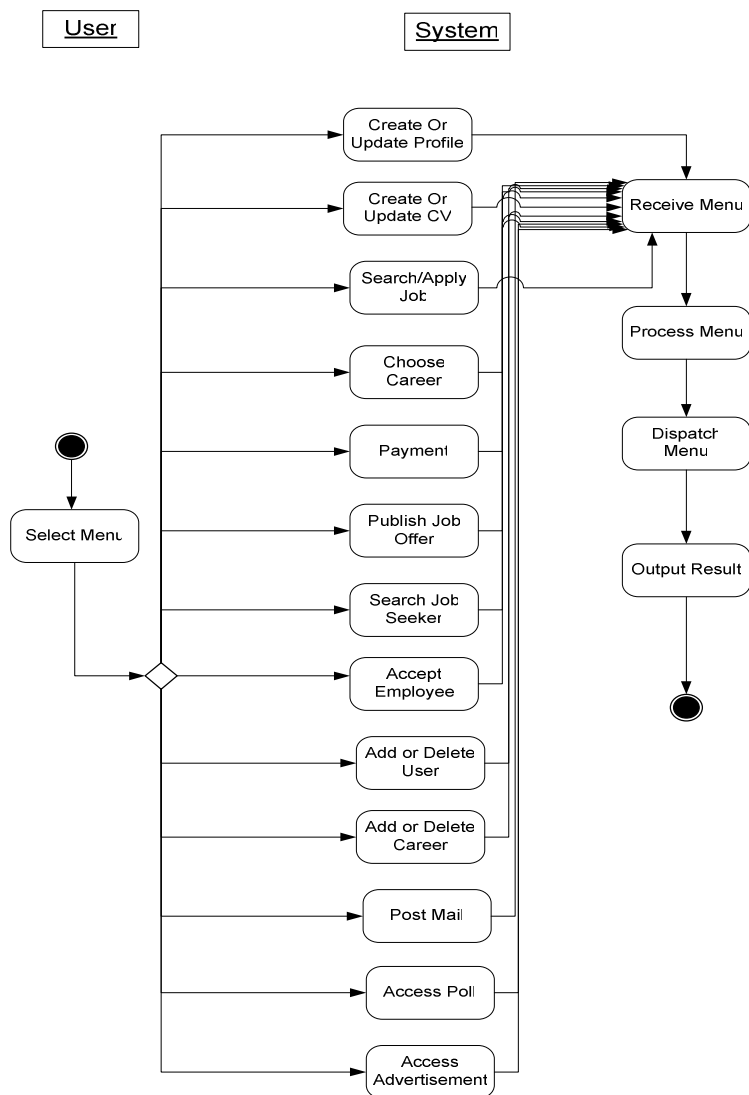
3.4.16 Select or Deselected Poll Activity Diagram



3.4.17 Log Off Activity Diagram



3.4.18 Menu Activity Diagram



4.USER INTERFACE DESIGN

4.1 Main Page

Below is the layout of the main page of our portal.

The screenshot shows the main page of the isteis.com portal. The header features the site logo 'isteis.com' and the tagline 'türkiye'nin kariyer durağı!'. Below the header is a navigation bar with links to 'www.kariyer.net', 'yenibiris.com', and 'secretcv.com'. The main content area is divided into two sections. The top section, titled 'haberler', contains a large green rectangular area. The bottom section, titled 'isteis-ler', contains a search form with fields for 'sektor', 'pozisyon', 'şehir', and 'anahtar sözcük', and a 'ara' button. To the left of the search form is a login section with a 'giris' dropdown, 'kullanıcı adı' and 'şifre' input fields, a 'giris' button, and a 'şifremi unuttum' link. To the right of the login section are two large red circular buttons with white text: 'üye ol' and 'üye firma ol'.

isteis.com türkiye'nin kariyer durağı!

www.kariyer.net "iş, insan, teknoloji"

yenibiris.com secretcv.com

haberler

giris

kullanıcı adı

şifre

giris

şifremi unuttum

üye ol

üye firma ol

isteis-ler

sektor

pozisyon

şehir

anahtar sözcük

ara

4.2 Sign Up Pages

Below are the sign up pages' layout for both job seekers and job providers.

Job Seeker Sign Up Page

isteis.com

türkiye'nin kariyer durağı!

[www.kariyer.net](#)
"iş, insan, teknoloji"

yenibiris:com

secretcv.com

üye ol

genel bilgiler

adi

soyadı

e-posta adresi

kullanıcı adı

şifre

şifre (tekrar)

güvenlik bilgileri

güvenlik sorusu

cevabı

kisisel bilgiler

tc kimlik no


doğum tarihi

cinsiyet

medeni durum

askerlik durumu

ev telefonu



halka açılıyor!

[www.taseronyazilim.com](#)
kapat

Job Provider Sign Up Page

isteis.com

türkiye'nin kariyer durağı!

[www.kariyer.net](#)
"iş, insan, teknoloji"

yenibiris:com

secretcv.com

uygulamalar

ilan olustur

üyelik güncelle

pazar analizi

ilan güncelle

ilan olustur

iste personel

sektor

deneyim süresi

pozisyon

ülke

şehir

çalışma şekli

anahtar sözcük

eğitim düzeyi

ara

anket

Sitemizin görünümünden memnun musunuz?

Evet

Hayır

4.3 User Profile Main Pages

The below layouts are showing the profile main pages of all portal users.

Job Seeker Main Page

isteis.com

türkiye'nin kariyer durağı!

www.kariyer.net

yenibiris:®

secretcv.com

uygulamalar

►cv olustur

►üyelik güncelle

►uygun ilanları

►cv güncelle

göster

is uyarisi

cv olustur

isteis-ler

sektor

deneyim süresi

pozisyon

ülke

şehir

çalışma şekli

anahtar sözcük

eğitim düzeyi

ara

anket

Sitemizin
görünümünden
memnun musunuz?

☐ Evet

☐ Hayır

Job Provider Main Page

isteis.com

türkiye'nin kariyer durağı!

www.kariyer.net

yenibiris:®

secretcv.com

uygulamalar

►ilan olustur

►üyelik güncelle

►pazar analizi

►ilan güncelle

ilan olustur

iste personel

sektor

deneyim süresi

pozisyon

ülke

şehir

çalışma şekli

anahtar sözcük

eğitim düzeyi

ara

anket

Sitemizin
görünümünden
memnun musunuz?

☐ Evet

☐ Hayır

Administrator Main Page



isteis.com türkiye'nin kariyer durağı!

www.kariyer.net
"iş, insan, teknoloji"

yenibiris.com

secretcv.com

uygulamalar

- profil güncelle
- admin ekle
- anketleri yönet
- duyuru yap
- kullanıcı sil
- reklamaları yönet

anketleri yönet

- anket ekle
- anketleri görüntüle

anket adı

alan sayısı

çoklu seçim ☐

ekle

4.3 CV Creation Page

Below is the standard cv form on our portal.



isteis.com türkiye'nin kariyer durağı!

www.kariyer.net
"iş, insan, teknoloji"

yenibiris.com

secretcv.com

cv olustur

kişisel bilgiler

cinsiyet

ev telefonu

cep telefonu

tc kimlik no

medeni hali

ülke

il

ilçe

eğitim bilgileri

seviye mezuniyet durumu mezuniyet tarihi bölüm okul

iş deneyimi

firma sektör çalışma alanı ülke/şehir başlangıç bitiş

yabancı dil

dil okuma yazma konuşma öğrenilen yer

4.4 Job Search Results Page

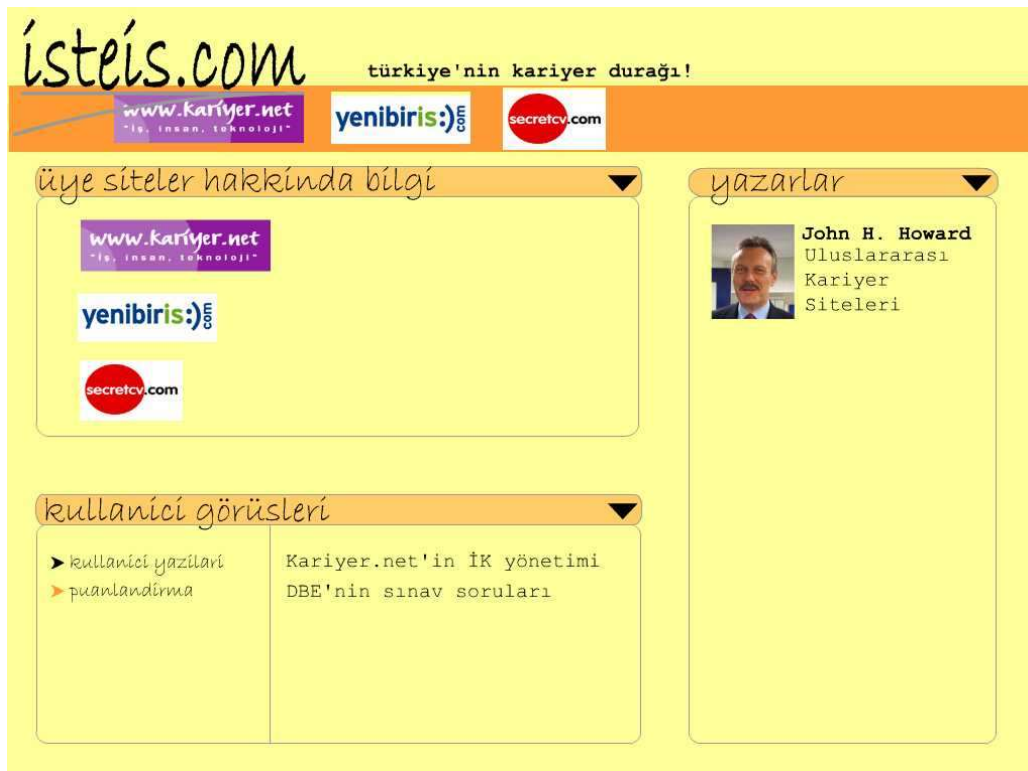
Job search results are shown like below with references of source career web pages.



firma	sektor	pozisyon	şehir	kariyer sitesi	başvuru
Taşeron Yazılım	Yazılım	Mühendis	Ankara	www.kariyer.net	✓

4.5 Market Analysis Page

This page contains general information for all users on our portal.



üye siteler hakkında bilgi

www.kariyer.net
yenibiris:
secretcv.com

yazarlar

John H. Howard
Uluslararası
Kariyer
Siteleri

kullanıcı görüşleri

► kullanıcı yazıları
► puanlandırma

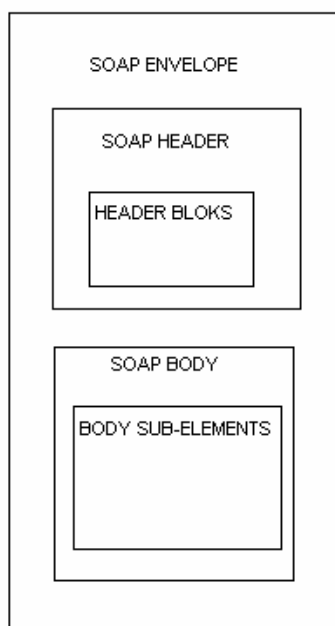
Kariyer.net'in İK yönetimi
DBE'nin sınav soruları

5.SYNTAX SPECIFICATION

As every software development has its own syntax, we decided to determine our own syntax specification for the Project. The specifications are listed below:

File format:

Since our project consists of web services we will work with XML database, SOAP messages and WSDL components, which are also in XML format. So XML format and syntax will be used in the files. The form of the SOAP messages are shown below:



Other than, we will have Java source codes. Since we are using webMethods to develop our Project, we will hold to the formatting of their tools.

Variable names:

If a variable name consists of more than one word, first letter of each word except the first one will be capitalized. Global variables will be named with the prefix “g_”, and static variables will be named with the prefix “s_”.

Function names:

If a function name consists of more than one word, ‘_’ will be put between words to connect and form a one word function name.

File names:

Files will be named in accordance with its content. Documentation related file names will be initiated with “Doc_” prefix.

Commenting:

Commenting of code is essential for later understanding and maintenance of the code. So, we decided to make commenting mandatory on implementation of our modules. However, it is not efficient to over-comment the code. Only the parts that can not be immediately understood by looking at the code should be commented. Comments will be short, simple and understandable.

6.TESTING ISSUES

6.1 Testing Plan and Strategy

To allow errors to be identified and removed, our code must be tested carefully. We also took into consideration that we don't have much time for testing. Therefore, we decided on some testing strategies and methods. The generated code should have some characteristics in order to be testable. These characteristics are listed below:

- Operability
- Observability
- Controllability
- Decomposability
- Simplicity
- Stability
- Understandability.

6.2 Testing Methods

Firstly we will implement unit testing using white box technique. We will test each module separately. The modules which will be tested are:

- Sign-up & login

- Profile & CV update
- Job search
- Job application
- Career job site selection & membership
- Payment
- Employee search
- Job posting
- User Management
- Poll Management
- Commercial Ads Management

Although we can find errors in modules by unit test, we must also make an integration test in order to find errors due to integration of the modules. In this case, bottom-up integration method will be used.

7.PROJECT SCHEDULE

7.1. Project Task Set and Workpackages

The tasks and the schedule of our project is determined in detail since we have a strict deadline. Consequently our work packages are predicated on the reports that have to be prepared and the scheduling is predicated on their deadlines.

Our first work package (WP0) is based on understanding the project. Determining the project scope and subtasks were included in this package. They can be derived from our proposal report.

The rest of the work packages are listed below:

WP1: Requirement Analysis

This workpackage includes mostly the researches and specifications about the project. The tasks are listed below:

- Literature Survey
- Existing System Analysis

- SOA&Web Services Research
- Decision of Services
- *DER: Project & Technology Overview Report
- Project Scheduling
- Data Modeling
- *DER: Metadata Report
- Requirement Specification
- Functional Requirements
- Non-functional Requirements
- System Requirements
- DFD Modeling
- Use Case Modeling
- Time & Effort Estimation
- *DER: Requirements Analysis Report

WP2: Administrator Side Design

WP2, WP3, and WP4 and WP5 are based on designing. WP2 consists of graphical user interface , database, and the process management for the functions which is provided for a System Administrator. Tasks are :

- User Management Module
- (Add/Delete User, announcements , information and feedback mails)
- Poll Management Module
- (Add /Delete/Update Polls to be sent to job seeker users)
- Commercial Adds Management Module
- (Add/Delete/Update commercial adds publishing on our portal)

WP3: Job Seeker Side Design

Work Package 3 includes designing the graphical user interface, the database, and the process management for the functions which will be provided to a job seeker. Tasks are indicated below:

- Sign up& Log in Modules

- Profile & CV Update
- Job Search Module
- Job Application Module
- *DER: Initial Design Report

WP4: Job Provider Side Design

Work Package 4 is identical to WP3, except it includes the design of the job provider side.

- Sign up& Log in Modules
- Career-Job Site Selection & Membership Module
- Payment Module
- Employee Search
- Job Posting Module
- *DER: Initial Design Report

WP5: Final Design

Final design starts with the review of the initial design and continues with doing the last modifications and adjustments on it. Tasks are indicated below:

- Initial Design Review
- Detailed GUI Design
- Detailed Database Design
- *DER: Final Design Report

Our implementation packages are consisting of the implementation of the designed modules in design packages.

WP6: Admin Side Implementation

The modules are again ;

- User Management Module
- (Add/Delete User, announcements, information and feedback mails)
- Poll Management Module
- (Add /Delete/Update Polls to be sent to job seeker users)

- Commercial Adds Management Module
- (Add/Delete/Update commercial adds publishing on our portal)

WP7: Job Seeker Side Implementation

The modules are again;

- Sign up& Log in Modules
- Profile & CV Update
- Job Search Module
- Job Application Module

WP8: Job Provider Side Implementation

The modules are again ;

- Sign up& Log in Modules
- Career-Job Site Selection & Membership Module
- Payment Module
- Employee Search
- Job Posting Module

Testing is made up of two modules , unit testing and integration testing at the end we also have to validate all system.

WP7: Unit Testing

We will be working on different modules of the system throughout the Project so at first we have to test every unit itself. Modules are the same with the implemented ones.

WP8: Integration Testing

After finishing unit testing , we are going to integrate all parts and test the whole system.

***("DER:" stands for deliverable from)**

Up to this point we have completed work packages WP0, WP1, WP2, WP3, WP4, WP5 and submitted milestone deliverables namely; proposal report, requirement analysis report, initial design report and this is the fourth deliverable final design report. These

workpackages simply involve understanding the Project and doing literature and market surveys, defining software, hardware and system requirements. However, our system's job categorization and job search filters are not clearly stated on previous reports. A job seeker will be able to enter a key word, select job location, and select job category while doing a job search. So we have determined a categorization for the jobs depending on their work areas. We have defined these categories as:

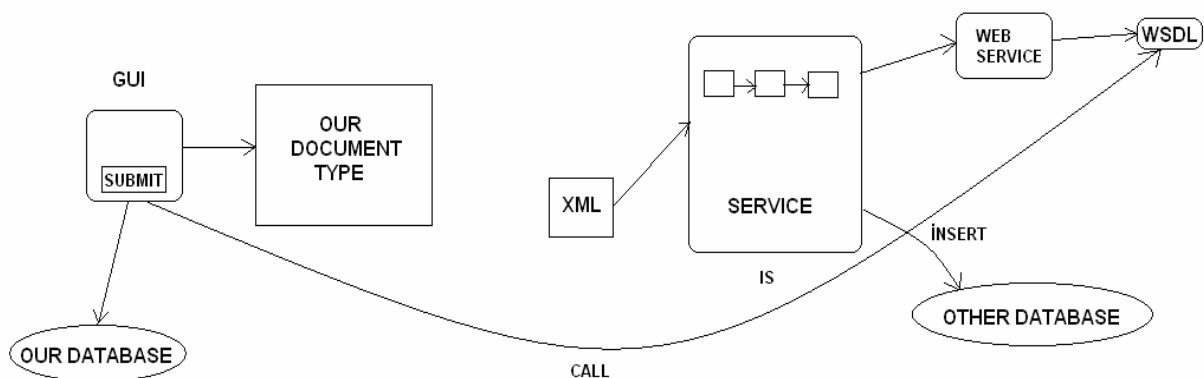
- Accounting/auditing
- Administrative and support services
- Advertising/marketing/public relations
- Aerospace/aviation/defense
- Agriculture, forestry, & fishing
- Airlines
- Architectural services
- Arts, entertainment, and media
- Automotive/motor vehicle/parts
- Banking
- Biotechnology and pharmaceutical
- Buildings and grounds maintenance
- Business opportunity/investment required
- Career fairs
- Computer services
- Computers, hardware
- Computers, software
- Construction, mining and trades
- Consulting services
- Consumer products
- Creative/design
- Customer service and call center
- Education, training and library
- Electronics
- Employment placement agencies
- Energy/utilities
- Engineering

- Environmental services
- Executive management
- Finance/economics
- Financial services
- Government and policy
- Healthcare
- Hospitality/tourism
- Human resources
- Information technology
- Installation, maintenance , and repair
- Insurance
- Internet/e-commerce
- Law enforcement/security services
- Legal
- Manufacturing and production
- Military
- Nonprofit
- Operations management
- Personal care and service
- Printing/editing/writing
- Product management/marketing
- Project/program management
- Purchasing
- Quality assurance/safety
- Real estate/mortgage
- Research & development
- Restaurant and food service
- Retail/wholesale
- Sales
- Science
- Sports and recreation/fitness
- Supply chain/logistic
- Telecommunications
- Textiles

- Transportation
- Veterinary services

With this final report , we became having completed the analysis and design levels of our Project. Moreover we have completed a prototype which is being able to show a basic feature of our project. Our prototype can receive an xml file of a cv document of any job seeker user on our portal with its all fields are filled and sends a new cv document to the corresponding career web page with the needed fields are filled. The values of these needed fields are inserted into the database of the corresponding source career web page.

It can be seen and understood more easily on the below figure :



When the user enters values to the all necessary fields and presses to the submit button, the information is sent to our database and also a web service is called which creates an xml document from the entered fields according to our document type and does the mapping of our document type and corresponding career web page`s document type. Later, within this service the information in the document type of the career web page is inserted into their database.

However; next semester, the focus will be on implementation and testing issues. The explained modules on workpackages will be implemented and tested. Our implementation plan will be according to our gantt chart where we grouped the work packages by different types of users on our portal. Every module for one of the user groups will be completed before starting to develop any other module of a different user group. We will rather be working on implementation of modules in groups of twos than as individuals. By this grouping, it will be more easy to take control of every step in development of the whole Project. To have a partner in the process will give us more energy and enthusiasm. And also

the integration of parts will become less time consuming, so we can tolerate any delay that would occur. The testing of each module will be done after the completion of that module. The integration and whole system testing will occur later. Following this implementation plan, we would have completed our Project and deployed by the June.

7.2. Gantt Chart

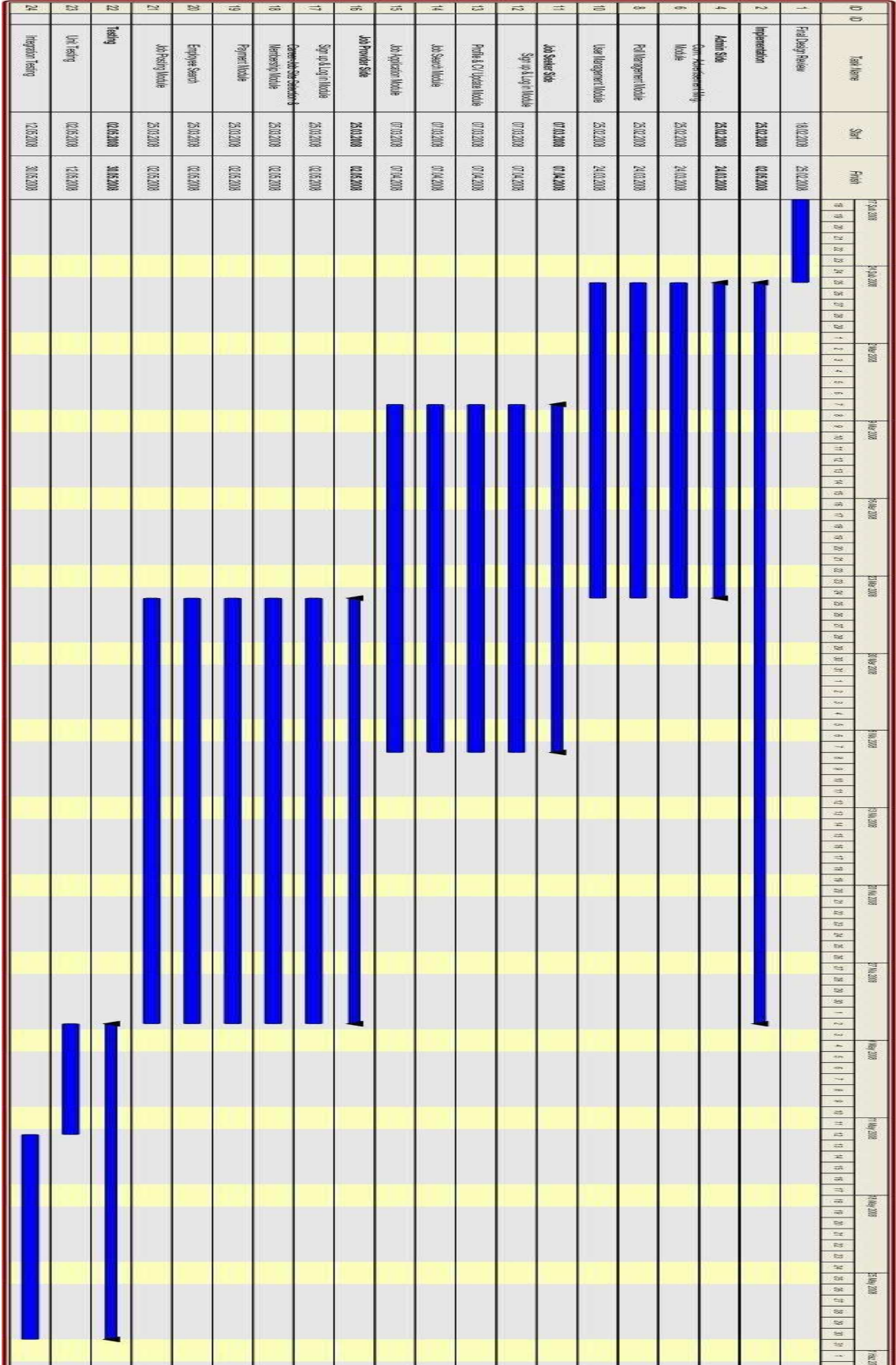
You can see our Schedule for first semester (fig.1) and second semester(fig.2) as Gantt Charts under Appendix.

8. APPENDIX

Fig. 1



Fig. 2



9.REFERENCES:

- [1] Wikipedia, http://en.wikipedia.org/wiki/Web_service
- [2] Sun Developer Network, <http://java.sun.com/developer/technicalArticles/WebServices/soa/>