1)

LinkedIn is a networking web site for businesses and people. Aim of this web site is to allow people and businesses create networks with each other. LinkedIn lets people to prepare a profile page where they can share their education and work experiences with other people. In LinkedIn people can follow businesses to be informed about changes in businesses or job offers. This website offers great convenience for community to share and know others’ experiences.

Kariyer.net is a web site for employees to search and apply for job offers and for companies to search and hire employees. Kariyer.net is apart from the LinkedIn is only created with the purpose of seeking a job. Kariyer.net lets employees to prepare CV which can be viewed by companies. Kariyer.net lacks the networking feature compared to Linkedn, but it makes it more centre on jobs and companies.

Moodle is a web site and a server for universities. Unlike the other web sites in this project, Moodle only focuses on education and supplying it. In Moodle universities can have seperate web pages for each of their departments. Each department’s page is composed of the course pages given in that department. In these course pages, teachers can share course materials or initialize homework submit areas where users can upload their homework files to the system. Moodle serves as a cloud storage so users of Moodle can upload and keep their files and access them anywhere.

2)

a)LinkedIn aims connecting users in their each respectable line of work. Allowing them to build a network with people who are relevant to their career. It aims to be a professional networking site with social overtones.

Kariyer.net aims to be a bridge between many workplace who are in need of new employees and the people who can place this vacancies. It stores peoples Cvs and lets the business owner s see available candidates.

Moodle aims to be a learning platform which provides educators, administrators and learners with a secure, easy to access, integrated system to create learning environments.

b)Main entities of LinkedIn are; Member, Group, Company, Address,University.

Main entities of Kariyer.net are; Member, Skill, Address, Company, Job Offer and University

Main entities of Moodle are; Teacher, Student, Project, File, Faculty, Course and University.

c)

d)

e)

3)

4)

5) DESIGN-LOGICAL MODEL

İteration1

Step1

Skill(skill\_id, name, category)

Address(address\_id, country, city, street, zip)

Group(group\_id, name, description)

Job\_offer(offer\_id, job\_title, description)

Office(office\_id, name)

File(file\_id, name, privacy)

Course(course\_id, name, code, credit)

Project(project\_id ,title, description)

Graduate\_level(grad\_id, name)

Department(dept\_id, name)

Faculty(faculty\_id, name)

Step2

-

Step3

-

Step4

Job\_offer(offer\_id, job\_title, description, address\_id, office\_id)

Project(project\_id ,title, description, course\_id)

File(file\_id, name, privacy, person\_id)

Course(course\_id, name, code, credit, dept\_id)

Project(project\_id ,title, description, course\_id)

Graduate\_level(grad\_id, name, dept\_id)

Department(dept\_id, name, grad\_id, address\_id)

Step 5

-

Step 6

-

Step 7

-

Step 8

Person(person\_id, fname, lname, phone, mail, bday,type)

Teacher(teacher\_id, branch)

Student(student\_id, gpa, grade)

Member(member\_id, password)

Organization(org\_id, name, phone, mail, type)

Step 9

-

Iteration 2

Step1

-

Step2

-

Step 3

-

Step 4

Person(person\_id, fname, lname, phone, mail, bday,type, address\_id)

Teacher(teacher\_id, branch, dept\_id)

Group(group\_id, name, description, member\_id)

Office(office\_id, name, address\_id, org\_id)

Faculty(faculty\_id, name, org\_id)

Step 5

Educates\_in(start\_date, end\_date, gpa, grad\_id, person\_id)

Messages(from\_person\_id, to\_person\_id, title, context,date)

References(teacher\_id, referenced\_person\_id, date, context)

Knows(person\_id, skill\_id)

Teaches(teacher\_id, course\_id, semester)

Enrolls(student\_id, course\_id, semester, grade)

Works\_on(student\_id, project\_id, grade)

Recommends(member\_id, recommended\_person\_id, date, context)

Follows(member\_id, org\_id)

Works\_for(member\_id, start\_date, end\_date, job\_title, office\_id)

Applies(member\_id, status, date, address\_id, offer\_id)

Joins(member\_id, group\_id)

Connects(member\_id, connected\_member\_id)

Step 6

-

Step7

-

Step8

-

Step9

-

6)

CREATE TABLE tbl\_skill (

skill\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL UNIQUE,

category TEXT NOT NULL,

CHECK(category IN ('Software', 'FineArts', 'Science','Sports'))

);

Create TABLE tbl\_address(

address\_id SERIAL PRIMARY KEY,

country TEXT NOT NULL,

city TEXT NOT NULL,

street TEXT NOT NULL,

zip INTEGER NOT NULL

);

CREATE TABLE tbl\_organization(

org\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL UNIQUE,

phone VARCHAR(11) NOT NULL,

mail TEXT NOT NULL UNIQUE,

type VARCHAR(3) NOT NULL,

CHECK( type IN ('CMP,'UNI'))

);

CREATE FUNCTION get\_type(data\_id INT)

RETURNS VARCHAR(3)

AS $$

BEGIN

RETURN (SELECT type FROM tbl\_organization WHERE org\_id = data\_id);

END; $$

LANGUAGE PLPGSQL;

CREATE TABLE tbl\_office(

office\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

org\_id INTEGER NOT NULL REFERENCES tbl\_organization(org\_id),

address\_id INTEGER NOT NULL REFERENCES tbl\_address(address\_id),

CHECK( 'CMP' = get\_type(org\_id))

);

CREATE TABLE tbl\_job\_offer(

offer\_id SERIAL PRIMARY KEY,

job\_title TEXT NOT NULL,

description TEXT NOT NULL,

address\_id INTEGER NOT NULL REFERENCES tbl\_address(address\_id),

office\_id INT NOT NULL REFERENCES tbl\_office(office\_id)

);

CREATE TABLE tbl\_faculty(

faculty\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

org\_id INTEGER NOT NULL REFERENCES tbl\_organization(org\_id),

CHECK( 'UNI' = get\_type(org\_id))

);

CREATE TABLE tbl\_department(

dept\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

faculty\_id INT NOT NULL REFERENCES tbl\_faculty(faculty\_id),

address\_id INTEGER NOT NULL REFERENCES tbl\_address(address\_id)

);

CREATE TABLE tbl\_graduate\_level(

grad\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

dept\_id INT NOT NULL REFERENCES tbl\_department(dept\_id)

);

CREATE TABLE tbl\_person(

person\_id SERIAL PRIMARY KEY,

fname TEXT NOT NULL,

lname TEXT NOT NULL,

phone INTEGER,

mail TEXT NOT NULL UNIQUE,

bday DATE NOT NULL,

type VARCHAR(3) NOT NULL,

address\_id INTEGER NOT NULL REFERENCES tbl\_address(address\_id),

CHECK( type IN ('TEA','STU','MEM'))

);

CREATE TABLE tbl\_teacher(

teacher\_id INT UNIQUE REFERENCES tbl\_person(person\_id),

branch TEXT,

dept\_id INT NOT NULL REFERENCES tbl\_department(dept\_id),

CHECK('TEA' = get\_type(teacher\_id))

);

CREATE TABLE tbl\_student(

student\_id INT UNIQUE REFERENCES tbl\_person(person\_id),

gpa FLOAT,

grade varchar(1),

CHECK( grade IN ('1', '2', '3', '4')),

CHECK ('STU' = get\_type(student\_id))

);

CREATE TABLE tbl\_member(

member\_id INT UNIQUE REFERENCES tbl\_person(person\_id),

password TEXT NOT NULL,

CHECK( char\_length(password)>=8 ),

CHECK ('MEM' = get\_type(member\_id))

);

CREATE TABLE tbl\_file(

file\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

privacy BOOLEAN DEFAULT TRUE,

person\_id INTEGER NOT NULL REFERENCES tbl\_person(person\_id),

CHECK( 'TEA' = get\_type(person\_id) OR 'STU' = get\_type(person\_id))

);

CREATE TABLE tbl\_course(

course\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

code TEXT NOT NULL,

credit INTEGER NOT NULL,

dep\_id INT NOT NULL REFERENCES tbl\_department(dept\_id)

);

CREATE TABLE tbl\_project(

project\_id SERIAL PRIMARY KEY,

title TEXT NOT NULL,

description TEXT NOT NULL,

course\_id INTEGER NOT NULL REFERENCES tbl\_course(course\_id)

);

CREATE TABLE tbl\_group(

group\_id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

description TEXT NOT NULL,

member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id)

);

CREATE TABLE tbl\_educates\_in(

start\_date DATE NOT NULL,

end\_date DATE,

gpa FLOAT,

grad\_id INT NOT NULL REFERENCES tbl\_graduate\_level(grad\_id),

person\_id INT REFERENCES tbl\_person(person\_id)

);

CREATE TABLE tbl\_references(

teacher\_id INTEGER NOT NULL REFERENCES tbl\_teacher(teacher\_id),

referenced\_person\_id INTEGER NOT NULL REFERENCES tbl\_person(person\_id),

date DATE NOT NULL,

context TEXT NOT NULL,

CHECK(teacher\_id != referenced\_person\_id),

CHECK('TEA' = get\_type(teacher\_id))

);

CREATE TABLE tbl\_knows(

person\_id INT REFERENCES tbl\_person(person\_id),

skill\_id INT REFERENCES tbl\_skill(skill\_id)

);

CREATE TABLE tbl\_teaches(

teacher\_id INTEGER NOT NULL REFERENCES tbl\_teacher(teacher\_id),

course\_id INTEGER NOT NULL REFERENCES tbl\_course(course\_id),

semester TEXT NOT NULL,

CHECK('TEA' = get\_type(teacher\_id))

);

CREATE TABLE tbl\_works\_on(

student\_id INTEGER NOT NULL REFERENCES tbl\_student(student\_id),

course\_id INTEGER NOT NULL REFERENCES tbl\_course(course\_id),

semester TEXT NOT NULL,

CHECK('STU' = get\_type(student\_id))

);

CREATE TABLE tbl\_recommends(

member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id),

recommended\_member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id),

date DATE NOT NULL,

context TEXT NOT NULL,

CHECK(member\_id != recommended\_member\_id)

);

CREATE TABLE tbl\_follows(

member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id),

org\_id INTEGER NOT NULL REFERENCES tbl\_organization(org\_id),

CHECK( 'CMP' = get\_type(org\_id))

);

CREATE TABLE tbl\_works\_for(

member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id),

start\_date DATE NOT NULL,

end\_date DATE,

job\_title TEXT NOT NULL,

office\_id INT NOT NULL REFERENCES tbl\_office(office\_id)

);

CREATE TABLE tbl\_applies(

member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id),

date DATE NOT NULL,

status TEXT DEFAULT 'NOTEXAMINED',

offer\_id INT NOT NULL REFERENCES tbl\_job\_offer(offer\_id),

CHECK(status IN ('NOTEXAMINED', 'REJECTED', 'ACCEPTED'))

);

CREATE TABLE tbl\_joins(

member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id),

group\_id INTEGER NOT NULL REFERENCES tbl\_group(group\_id)

);

CREATE TABLE tbl\_connects(

member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id),

connected\_member\_id INTEGER NOT NULL REFERENCES tbl\_member(member\_id),

CHECK(member\_id != connected\_member\_id)

);

7)