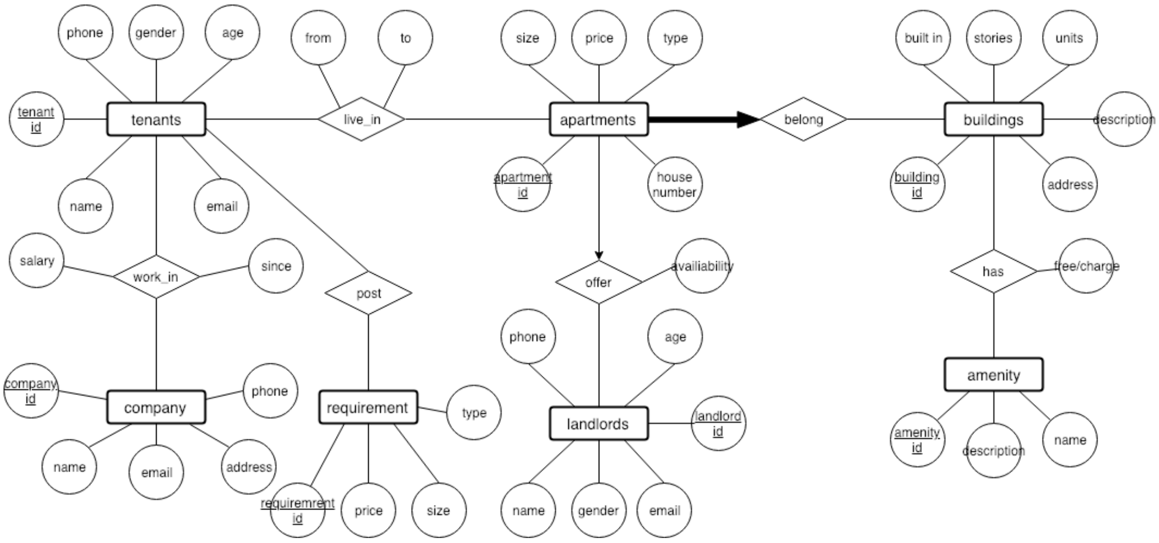
4111 Project 1. Proposal

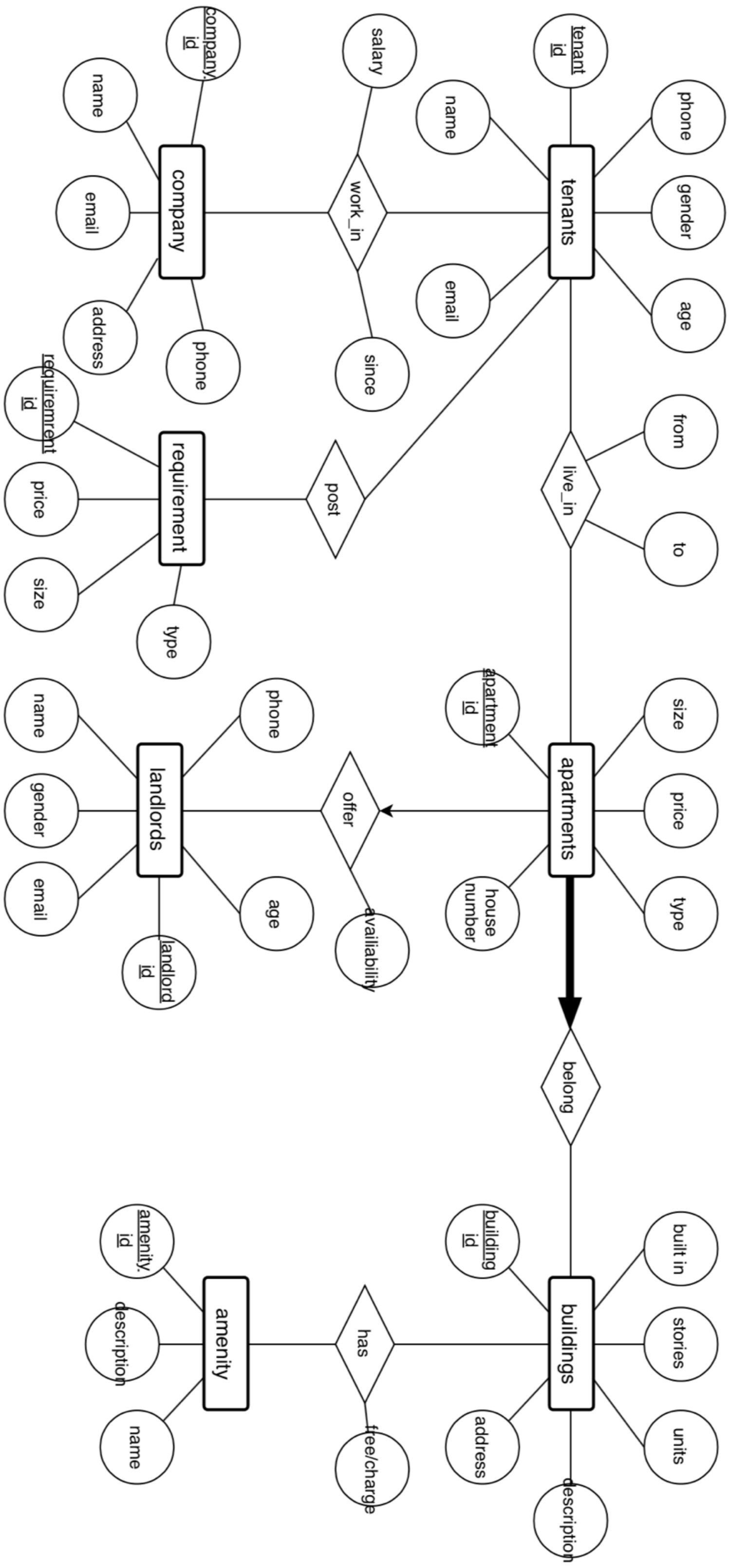
Yingling Wang (yw3152) and Anran Li (al3804)

Introduction

Our project is to establish a database system for a house rental website. Users of the website consist of house seekers and landlords. Our website helps tenants find appropriate houses they want conveniently and efficiently. For both house seeker or landlord, the users can compile their profile including name, gender, age, email and phone on the website. For house seekers, they can complete current living section, company section and requirement section on the website and get noticed of the appropriate apartment they may like. For landlords, they can post apartments they have on the website and get noticed of tenants who may want to rent the apartment. This website gives an open and free platform for renting houses. As for data, we can crawl them from real-world house rental website like streeteasy(https://streeteasy.com/)

Entity-Relationship Diagram




tenants
tenant id
name
email
phone
gender
age
work_in
salary
since
company
company id
name
email
address
phone
post
requirement
requiremrent id
price
size
type
live_in
from
to
apartments
apartment id
house number
size
price
type
offer
availiability
landlords
name
gender
email
landlord id
phone
age
belong
buildings
building id
address
description
built in
stories
units
has
free/charge
amenity
amenity id
description
name

Contingency Plan

If my teammate leaves, I will drop entities: **company** and **amenity**. And I'll drop relationships: **work_in** and **has.**

Relational Schema in SQL

Entity:

1. tenants

```
CREATE TABLE tenants (
tenant_id INTEGER,
name CHAR(20),
email CHAR(40),
phone INTERGER,
gender CHAR(10),
age INTERGER,
PRIMARY KEY(tenant_id)
);
```

2. apartments_belong

```
CREATE TABLE apartments_belong (
apartment_id INTEGER,
size CHAR(20),
type CHAR(10),
price INTERGER,
building_id INTERGER NOT NULL,
house_number CHAR(10),
PRIMARY KEY(apartment_id),
FOREIGN KEY(building_id) REFERENCES buildings, ON DELETE NO ACTION
);
```

3. landlords

```
CREATE TABLE landlords (
landlord_id INTEGER,
name CHAR(20),
email CHAR(40),
phone INTERGER,
gender CHAR(10),
age INTERGER,
PRIMARY KEY(landlord_id)
);
```

4. buildings

```
CREATE TABLE buildings (
building_id INTEGER,
built_in DATE,
stories CHAR(40),
units CHAR(10),
description CHAR(40),
address CHAR(40),
PRIMARY KEY(building_id)
);
```

5. amenity

```
CREATE TABLE amenity (
amenity_id INTEGER,
name CHAR(20),
description CHAR(40),
PRIMARY KEY(amenity_id)
);
```

6. requirement

```
CREATE TABLE requirement (
Requirement_id INTEGER,
size CHAR(20),
type CHAR(10),
price INTERGER,
PRIMARY KEY(requirement_id)
);
```

7. company

```
CREATE TABLE company (
company_id INTEGER,
name CHAR(20),
address CHAR(40),
phone INTERGER,
email CHAR(40),
PRIMARY KEY(company_id)
);
```

Relationship

1. work_in

work_in(tenant_id, company_id, since, salary, FK(company_id)-->company, FK(tenant_id)-->tenants)

```
CREATE TABLE work_in (
tenant _id INTEGER,
company_id INTEGER NOT NULL,
salary INTERGER,
since DATE,
PRIMARY KEY(tenant _id),
FOREIGN KEY(tenant _id) REFERENCES tenants,
FOREIGN KEY(company_id) REFERENCES company
);
```

2. live_in

live_in(apartment_id, tenant_id, from, to, FK(apartment_id)-->apartment, FK(tenant_id)-->tenants)

```
CREATE TABLE live_in (
tenant _id INTEGER,
apartment_id INTEGER,
from DATE,
to DATE,
PRIMARY KEY(apartment_id, tenant _id),
FOREIGN KEY(tenant _id) REFERENCES tenants,
FOREIGN KEY(apartment _id) REFERENCES apartments
);
```

3. offer

offer(apartment_id, landlord_id, availability, FK(apartment_id)-->apartments, FK(landlord_id)-->landlords)

```
CREATE TABLE offer (
apartment _id INTEGER,
landlord_id INTEGER NOT NULL,
availability BOOLEAN,
PRIMARY KEY(apartment _id),
FOREIGN KEY(apartment _id) REFERENCES apartments,
FOREIGN KEY(landlord_id) REFERENCES landlords
);
```

4. post

post(requirement_id, tenant_id, from, to, FK(requirement_id)-->requirement, FK(tenant_id)-->tenants)

```
CREATE TABLE post(
tenant _id INTEGER,
requirement_id INTEGER,
PRIMARY KEY(requirement_id, tenant _id),
FOREIGN KEY(tenant _id) REFERENCES tenants,
FOREIGN KEY(requirement _id) REFERENCES requirement
);
```

5. has

has(building_id, amenity_id, free_or_charge, FK(building_id)-->buildings, FK(amentity_id)-->amenity)

```
CREATE TABLE has (
building_id INTEGER,
amenity_id INTEGER,
free_or_charge BOOLEAN,
PRIMARY KEY(building_id, amentity_id),
FOREIGN KEY(building _id) REFERENCES buildings,
FOREIGN KEY(amenity_id) REFERENCES amenity
);
```

Approved By: Mengyu Han.
Suggestion: Datasource Link. Contingent Plan.

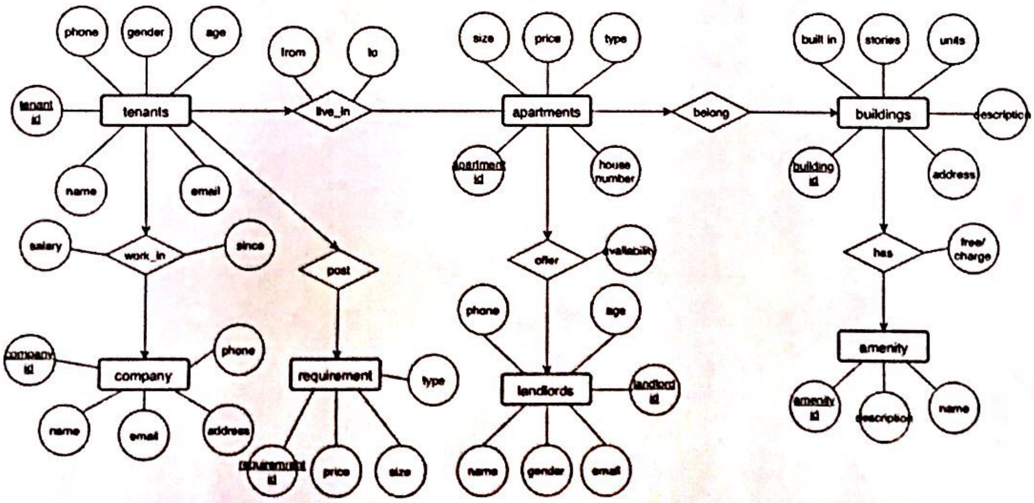
4111 Project 1. Proposal

Yingling Wang (yw3152) and Anran Li (al3804)

Introduction

Our project is to establish a database system for a house rental website. Users of the website consist of house seekers and landlords. Our website helps tenants find appropriate houses they want conveniently and efficiently. For both house seeker or landlord, the users can compile their profile including name, gender, age, email and phone on the website. For house seekers, they can complete current living section, company section and requirement section on the website and get noticed of the appropriate apartment they may like. For landlords, they can post apartments they have on the website and get noticed of tenants who may want to rent the apartment. This website gives an open and free platform for renting houses. As for data, we can crawl them from real-world house rental website like streeteasy.

Entity-Relationship Diagram


phone
gender
age
tenant id
tenants
name
email
from
to
live_in
salary
work_in
since
post
company id
company
phone
name
email
address
requirement
type
requirement id
price
size
size
price
type
apartments
apartment id
house number
offer
availability
phone
age
landlords
landlord id
name
gender
email
belong
built in
stories
units
buildings
description
building id
address
has
free/ charge
amenity
amenity id
description
name