Bilkent University

Department of Computer Engineering

CS353 Term Project Design Report

DatAnimal

Assigned Teacher Assistant

Duygu Durmuş

Group 34

Team Members

Asım Güneş Üstünalp

Radman Lotfiazar

Turan Mert Duran

Berdan Akyürek

Table of Content

Revised ER Diagram **4**

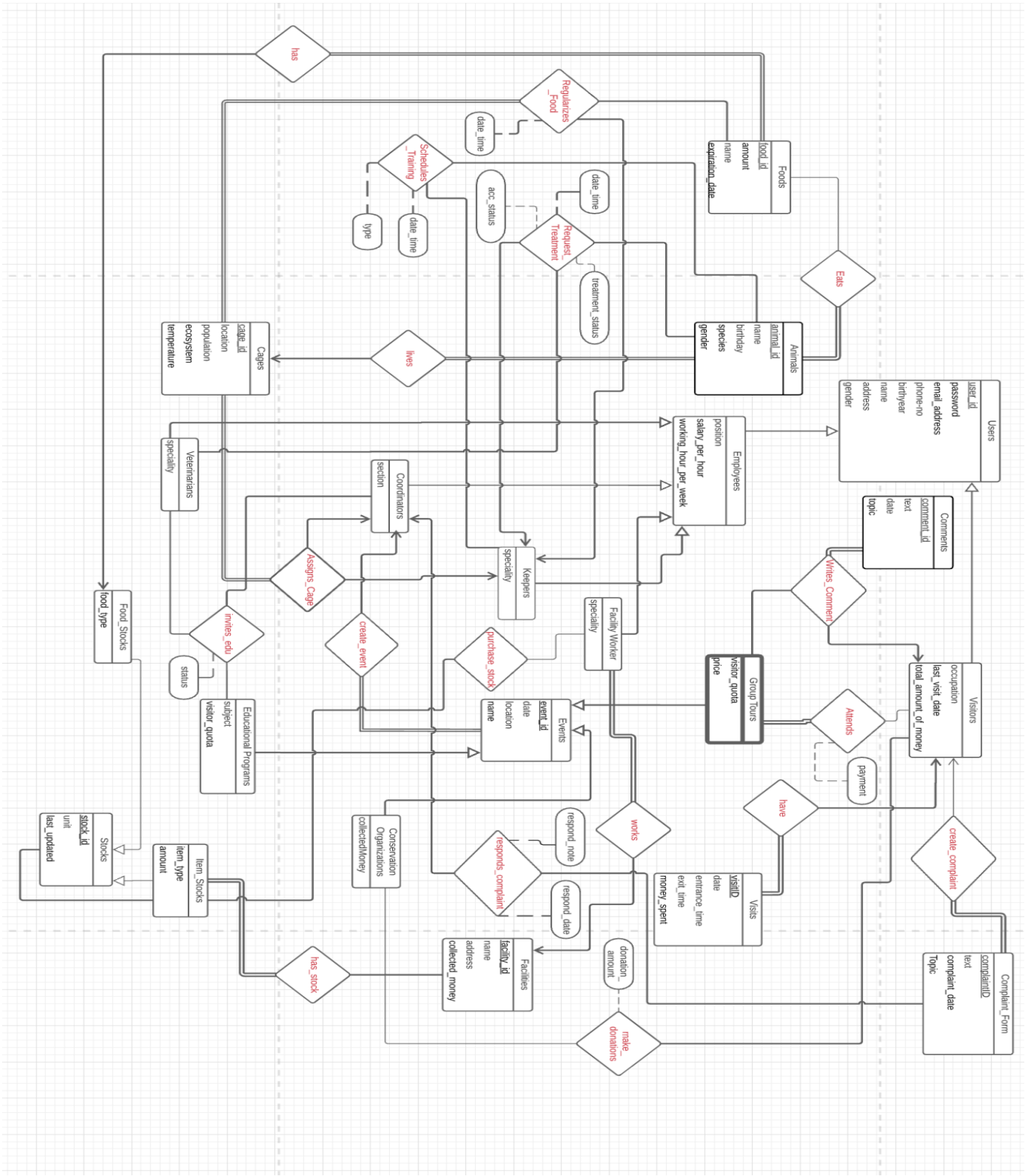
Table Schemas **5**

Users 5
Employees 5
Visitors 6
Facility_Worker 7
Keepers 7
Coordinators 8
Veterinarians 8
Animals 9
Cages 10
Foods 10
Stocks 11
Food_Stocks 12
Item_Stocks 12
Facilities 13
Events 13
Group_Tours 14
Educational_Program 15
Conservation_Organizations 15
Comments 16
Complaint_Forms 16
Respond_Complaint 17
Visit 18
Schedules_Training 18
Regularizes_Food 19
Request_Treatment 20
Writes_Comment 21
Attends 21
Assigns_Cage 22
Make_donation 23
Purchase_Stock 23

Invites_Edu 24
Has_Stock 24
Eats 25

User Interface Design and Database Statement 26
Home Page 27
Visitors Login Page 28
Employees Login Page 29
Visitors Sign Up Page 30
Employees Sign Up Page 31
Visitors Main Page 32
Visitors Donation 33
Visitors Past Visit 35
Visitors Attend Group Tours 36
Visitors Complaints 38
Employees Main Page 39
Employees Complaints 40
Employees Create Events 41
Assign Cages To Keepers 42
Send Email to Visitors 43
Create Conservation Organization 44
Keeper Main Page 45
Schedule New Training 46
Request New Treatment 47
Cage Food Stock 48
Facility Worker Main Page 49
Past Stock Order Changes 50
Current Stock 51
Veterinarian Main Page 52
Animals in Cage 53
select animal_id, name, birthday, species, gender 54
Visitors Comment on Attended Group Tours 55

1. Revised ER Diagram



2. Table Schemas

2.1. Users

Rational Model

Users(<u>user_id</u>, password, email_address, phone_no, birth_year, name, address, gender)

Candidate Keys

({user_id}, {email_address})

Table Definition

```
create table  Users(
user_id       int not null,
password      varchar(10) not null,
email         varchar(30) not null,
phone_no      int not null,
birth_year    date not null,
name          varchar(15) not null,
address       varchar(100) not null,
gender        varchar(5) not null,
primary key   (user_id)
);
```

2.2. Employees

Rational Model

Employees(<u>user_id</u>, position, salary_per_hour, working_hour_per_week)

Candidate Keys

({user_id})

Table Definition

create table Employees(
user_id **int** not null,
position **varchar(10)** not null,
salary_per_hour **in** not null,
working_hour_per_week **int** not null,
primary key (user_id),
foreign key (user_id) references Users
);

2.3. Visitors

Rational Model

Visitors(<u>user_id</u>, occupation, last_visit_date, total_amount_of_money)

Candidate Keys

({user_id})

Table Definition

create table Employees(
user_id **int** not null,
occupation **varchar(10)** not null,
last_visit_date **date**,
Total_amount_of_money **int**,
primary key (user_id),
foreign key (user_id) references Users

);

2.4. Facility_Worker

Rational Model

Facility_Worker(<u>user_id</u>, speciality, facility_id)

Candidate Keys

({user_id})

Table Definition

```
create table   Facility_Worker (
user_id        int not null,
speciality     varchar(10) not null,
facility_id    int not null,
primary key    (user_id),
foreign key    (user_id) references Users,
foreign key    (facility_id) references Facilities
);
```

2.5. Keepers

Rational Model

Keepers(<u>user_id</u>, speciality)

Candidate Keys

({user_id})

Table Definition

```
create table   Keepers(
user_id        int not null,
speciality     int not null,
primary key    (user_id),
foreign key    (user_id) references Users
);
```

2.6. Coordinators

Rational Model

Coordinators(<u>user_id</u>, section)

Candidate Keys

({user_id})

Table Definition

```
create table   Coordinators (
user_id        int not null,
section        varchar(10) not null,
primary key    (user_id),
foreign key    (user_id) references Users
);
```

2.7. Veterinarians

Rational Model

Veterinarians (<u>user_id</u>, speciality)

Candidate Keys

({user_id})

Table Definition

create table	Veterinarians (
user_id	**int** not null,
speciality	**varchar(10)** not null,
primary key	(user_id),
foreign key	(user_id) references Users
);	

2.8. Animals

Rational Model

Animals (animal_id, name, birthday, species, gender, cage_id)

Candidate Keys

({animal_id})

Table Definition

create table	Animals(
animal_id	**int** not null,
name	**varchar(20)** not null,
birthday	**date** not null,
name	**varchar(15)** not null,
cage_id	**int** not null,
species	**varchar(20)**,
gender	**varchar(5),**

primary key (animal_id),
foreign key (cage_id) references Cages
);

2.9. Cages

Rational Model

Cages (cage_id, location, population, ecosystem, temperature)

Candidate Keys
({cage_id})

Table Definition

create table Cages (
cage_id **int** not null,
location **varchar(100)** not null,
population **int** not null,
ecosystem **varchar(20)** ,
temperature **int**,
primary key (cage_id)
);

2.10. Foods

Rational Model

Foods(food_id, name, expiration_date, stock_id, amount)

Candidate Keys
({food_id})

Table Definition

create table	Foods (
food_id	**int** not null,
name	**varchar(20)** not null,
expiration_date	**date** not null,
Stock_id	**int** not null,
amount	**int** not null,
primary key	(food_id),
foreign key	(animal_id) references Animals,
foreign key	(stock_id) references Stocks
);	

2.11. Stocks

Rational Model

Stocks(<u>stock_id</u>, unit, last_updated)

Candidate Keys

({stock_id})

Table Definition

create table	Stocks (
srtock_id	**int** not null,
unit	**varchar(5)** not null,
last_update	**date** not null,
primary key	(stock_id)
);	

2.12. Food_Stocks

Rational Model

Food_Stocks(stock_id, food_type)

Candidate Keys

({stock_id})

Table Definition

```
create table Food_Stocks (
stock_id        int not null,
food_type       varchar(10) not null,
primary key     (stock_id),
foreign key     (stock_id) references Stocks
);
```

2.13. Item_Stocks

Rational Model

Item_Stocks(stock_id, item_type, amount)

Candidate Keys

({stock_id})

Table Definition

```
create table    Item_Stocks (
stock_id        int not null,
item_type       varchar(10) not null,
amount          int not null,
```

primary key (stock_id),
foreign key (stock_id) references Stocks
);

2.14. Facilities

Rational Model

Facilities(facility_id, name, address, collected_money)

Candidate Keys
({facility_id})

Table Definition

create table	Facilities (
facility_id	**int** not null,
name	**varchar(20)** not null,
address	**varchar(100)** not null,
collected_money	**int** not null,
primary key	(facility_id)
);	

2.15. Events

Rational Model

Events (event_id, date, location, user_id, name)

Candidate Keys
({event_id})

Table Definition

```
create table   Events (
event_id       int not null,
date           date not null,
location       varchar(100) not null,
user_id        int not null,
name varchar(100) not null,
primary key    (event_id),
foreign key    (user_id) references Users
);
```

2.16. Group_Tours

Rational Model

Group_Tours(<u>event_id,</u> visitor_qouta, price)

Candidate Keys

({event_id})

Table Definition

```
create table   Group_Tours (
event_id       int not null,
visitor_qouta  int not null,
price          int not null,
primary key    (event_id),
foreign key    (event_id) references Events
);
```

2.17. Educational_Program

Rational Model

Educational_Program(event_id, visitor_qouta, subject)

Candidate Keys

({event_id})

Table Definition

```
create table   Educational_Program (
event_id       int not null,
visitor_qouta  int not null,
subject        varchar(20) not null,
primary key    (event_id),
foreign key    (event_id) references Events
);
```

2.18. Conservation_Organizations

Rational Model

Conservation_Organizations (event_id, collectedMoney)

Candidate Keys

({event_id})

Table Definition

```
create table        Conservation_Organizations (
event_id            int not null,
collectedMoney      int not null,
```

primary key (event_id),
foreign key (event_id) references Events
);

2.19. Comments

Rational Model

Comments(comment_id, topic, text, date)

Foriegn Keys

None

Candidate Keys

({event_id})

Table Definition

create table Comments (
comment_id **int** not null,
topic **varchar(20)** not null,
text **varchar(200)** not null,
date **date** not null,
primary key (event_id)
);

2.20. Complaint_Forms

Rational Model

Complaint_Form (complaint_id, text, complaint_date, topic, user_id)

Candidate Keys

({complaint_id}, {complaint_date, user_id})

Table Definition

create table Complaint_Form (

comment_id **int** not null,

topic **varchar(20)** not null,

text **varchar(200)** not null,

complaint_date **date** not null,

user_id **int** not null,

primary key(complaint_id**),**

foreign key (user_id) references Users

);

2.21. Respond_Complaint

Rational Model

Respond_complaint (complaint_id, user_id, respond_note, respond_date)

Candidate Keys

({complaint_id})

Table Definition

create table Respond_complaint (

complaint_id **int** not null,

user_id **int** not null,

respond_note **varchar** not null,

respond_date **date** not null,

primary key (complaint_id),

foreign key (complaint_id) references Complaint_Form,
foreign key (user_id) references Users
);

2.22. Visit

Rational Model

Visit(visit_id, date, entrance_time, exit_time, money_spent, user_id)

Candidate Keys

({visit_id})

Table Definition

create table Visit (
visit_id **int** not null,
date **date** not null,
entrance_time **time** not null,
exit_time **time** not null,
money_spent **int** not null,
user_id **int** not null,
primary key (visit_id),
foreign key (user_id) references Users
);

2.23. Schedules_Training

Rational Model

Schedules_Training(animal_id, user_id, date_time, type)

Candidate Keys

({animal_id, user_id, date_time})

Table Definition

create table Schedules_Training (
animal_id **int** not null,
user_id **int** not null,
date_time **datetime** not null,
type **varchar(30)**,
primary key (animal_id, user_id, date_time)
foreign key (animal_id) references Animals,
foreign key (keeper_id) references Keepers
);

2.24. Regularizes_Food

Rational Model

Regularizes_Food(food_id, user_id, cage_id, date_time)

Candidate Keys

({food_id, user_id, cage_id, date_time**})**

Table Definition

create table Regularizes_Food (
animal_id **int** not null,
user_id **int** not null,
cage_id **int** not null,
date_time **datetime** not null,
primary key (food_id, user_id, cage_id, date_time).
foreign key (food_id) references Foods.

foreign key (user_id) references Users.

foreign key (cage_id) references Cages

);

2.25. Request_Treatment

Rational Model

Request_Treatment(<u>animal_id, keeper_user_id, vet_user_id, date_time</u>, treatment_status, acc_status)

Candidate Keys

({animal_id, keeper_user_id, vet_user_id, date_time**})**

Table Definition

create table	Regularizes_Food (
animal_id	**int** not null,
keeper_user_id	**int** not null,
vet_user_id	**int** not null,
date_time	**datetime** not null,
treatment_status	**varchar(30)**,
acc_status	**varchar(30)**,
primary key	(animal_id, keeper_user_id, vet_user_id, date_time),
foreign key	(food_id) references Foods,
foreign key	(keeper_user_id) references Users,
foreign key	(vet_user_id) references Users,
foreign key	(cage_id) references Cages,

);

2.26. Writes_Comment

Rational Model

Writes_Comment(comment_id, event_id, user_id)

Candidate Keys

({comment_id})

Table Definition

```
create table    Writes_Comment (
comment_id      int not null,
event_id        int not null,
user_id         int not null,
primary key     (comment_id),
foreign key     (comment_id) references Comments,
foreign key     (event_id) references Events,
foreign key     (user_id) references Users,
);
```

2.27. Attends

Rational Model

Attends(user_id, event_id, payment)

Candidate Keys

({user_id, event_id})

Table Definition

```
create table   Attends(
user_id        int not null,
event_id       int not null,
payment        float not null,
primary key    (),
foreign key    (event_id) references events,
foreign key    (user_id) references Users,
);
```

2.28. Assigns_Cage

Rational Model

Assigns_Cage(<u>coordinator_user_id, keeper_user_id, cage_id</u>)

Candidate Keys

({coordinator_user_id, keeper_user_id, cage_id}**)**

Table Definition

```
create table          Assigns_Cage(
coordinator_user_id   int not null,
keeper_user_id        int not null,
cage_id               int not null,
primary key           (coordinator_user_id, keeper_user_id, cage_id),
foreign key           (coordinator_user_id) references Users,
foreign key           (keeper_user_id) references Users,
foreign key           (cage_id) references Cages
);
```

2.29. Make_donation

Rational Model

Make_donation (event_id, user_id, donation_amount)

Candidate Keys

({event_id, user_id})

Table Definition

```
create table      Make_Donation (
event_id          int not null,
user_id           int not null,
donation_amount   float not null,
primary key       (event_id, user_id),
foreign key       (cevent_id) references Events,
foreign key       (user_id) references Users
);
```

2.30. Purchase_Stock

Rational Model

Purchase_Stock (user_id, stock_id)

Candidate Keys

{(user_id, stock_id)}

Table Definition

```
create table   Purchase_Stock (
user_id        int not null,
stock_id       int not null,
```

primary key (user_id, stock_id),

foreign key (user_id) references Users,

foreign key (stock_id) references Stocks

);

2.31. Invites_Edu

Rational Model

Invites_Edu (vet_user_id, coor_user_id, event_id, status)

Candidate Keys

({vet_user_id, coor_user_id, event_id}**)**

Table Definition

create table Invites_Edu**(**

vet_user_id **int** not null**,**

coor _user_id **int** not null**,**

event_id **int** not null**,**

Primary key (vet_user_id, coor_user_id, event_id),

foreign key (coor_user_id) references Users,

foreign key (vet_user_id) references Users,

foreign key (evet_id) references Events

);

2.32. Has_Stock

Rational Model

Has_Stock (stock_id, facility_id)

Foriegn Keys

stock_id: FK to Stocks

facility_id: FK to Facility

Candidate Keys

{(stock_id, facility_id)}

Table Definition

```
create table   Has_Stock(
stock_id       int not null,
facility_id    int not null,
primary key    (stock_id, facility_id),
foreign key    (stock_id) references Stocks,
foreign key    (facilityr_id) references facilities,
);
```

2.33. Eats

Rational Model

Eats(animal_id, food_id)

Candidate Keys

{(animal_id, food_id)}

Table Definition

```
create table Eats(
animal_id      int not null,
food_id        int not null,
primary key    (animal_id, food_id),
```

```
foreign key    (animal_id) references animals,
foreign key    (food_id) references foods,
);
```

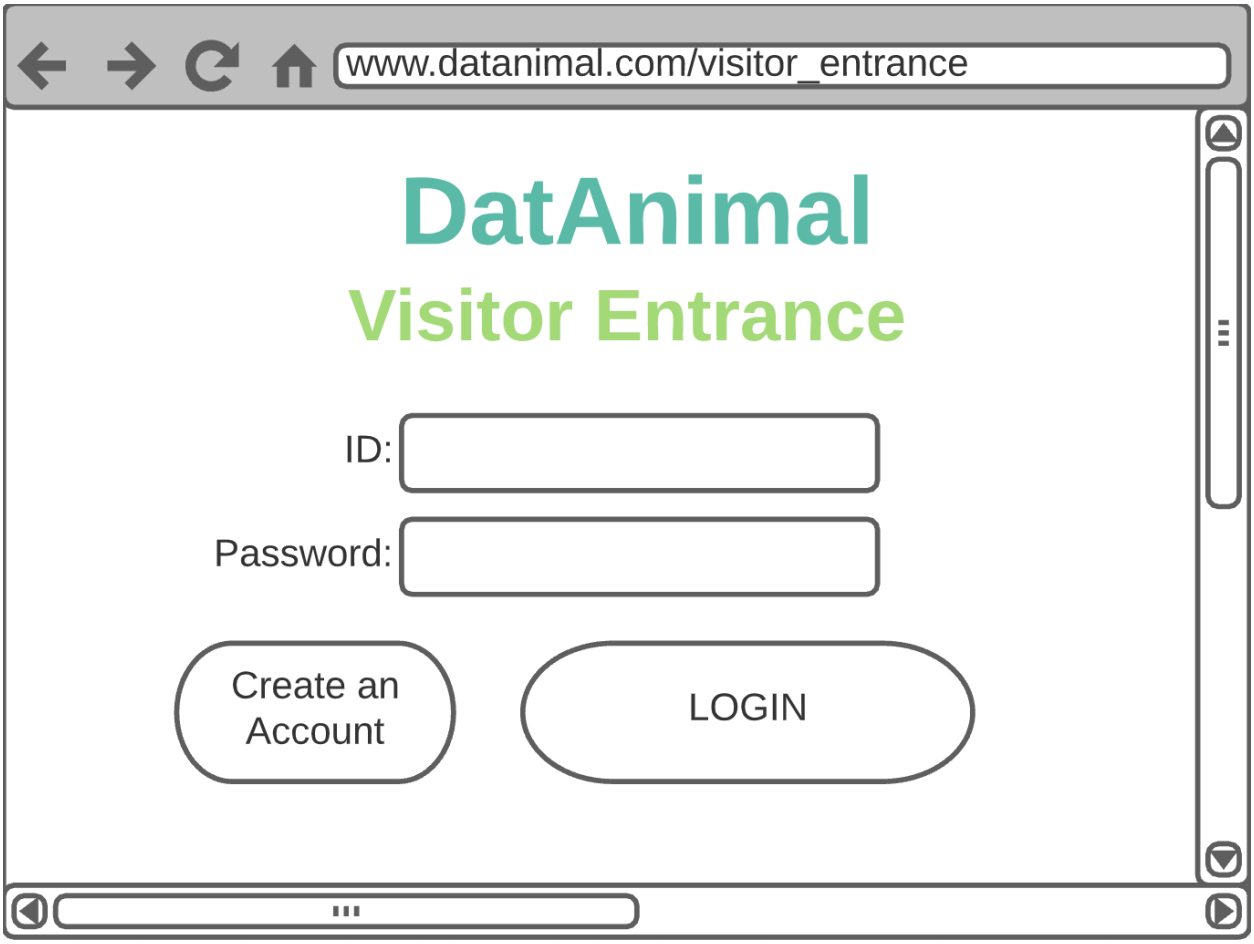
3. User Interface Design and Database Statement

In this part of the report we indicate the user interface of DatAnimal and corresponding SQL statements for each of them.

3.1. Home Page

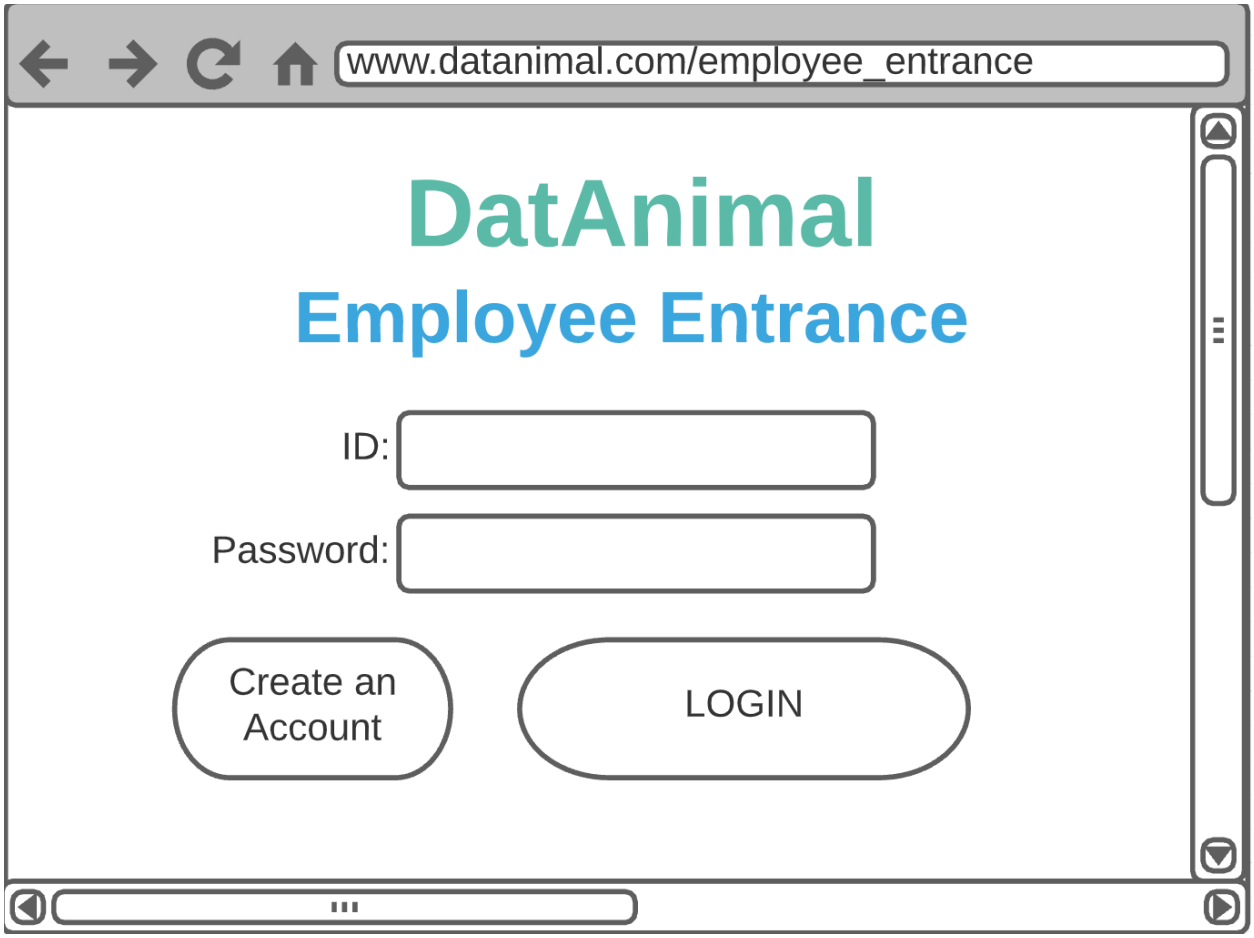

www.datanimal.com
Welcome to
DatAnimal
Are you a visitor or an employee?
Visitor
Employee

3.2. Visitors Login Page



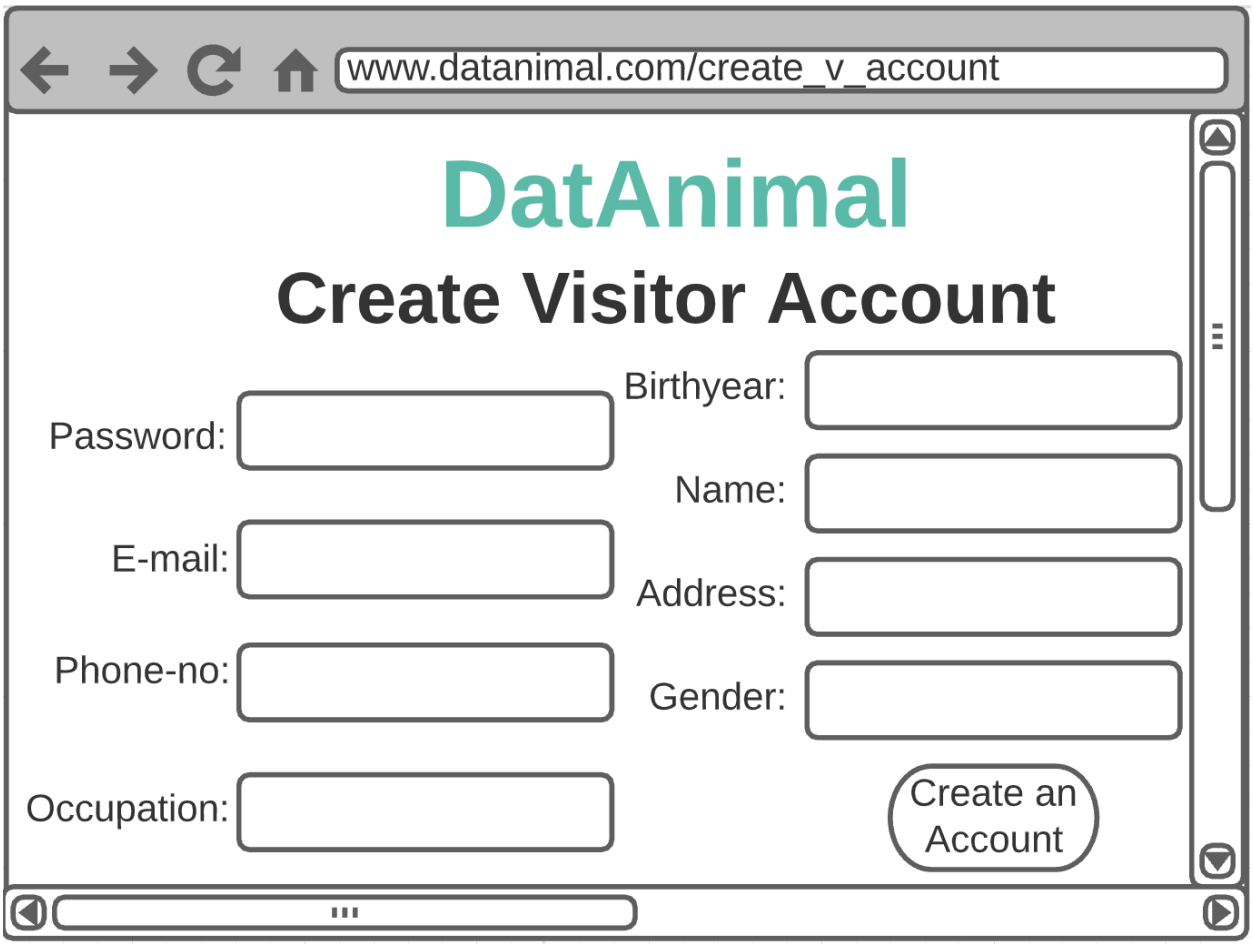
select user_id from Users where user_id = ID and user_password = PASSWORD;

3.3. Employees Login Page



select user_id from Users where user_id = ID and user_password = PASSWORD;

3.4. Visitors Sign Up Page



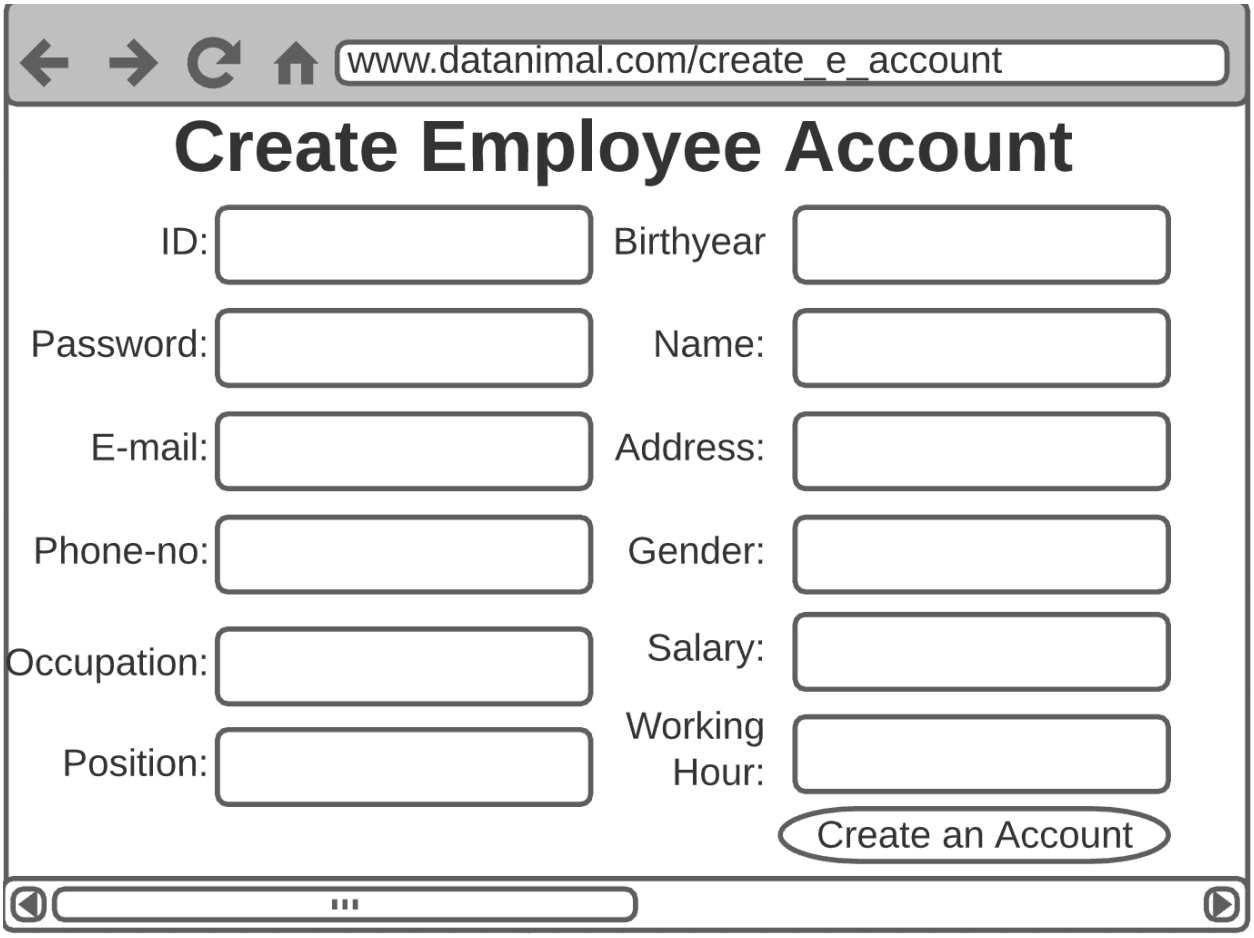
<u>Check existence of email:</u>

```
select id
from Users
where  email = EMAIL;
```

<u>Create account:</u>

```
insert into
Users( user_id, password, email_address, phone_no, birth_year, name, address, gender)
values(ID, PASSWORD, EMAIL, PHONE, BYEAR, NAME, ADDR, GENDER);

insert into
Visitors( user_id, occupation, last_visit_date, total_amount_of_money)
values(ID, OCCUPATION, null, 0);
```

3.5. Employees Sign Up Page



Check existence of email:

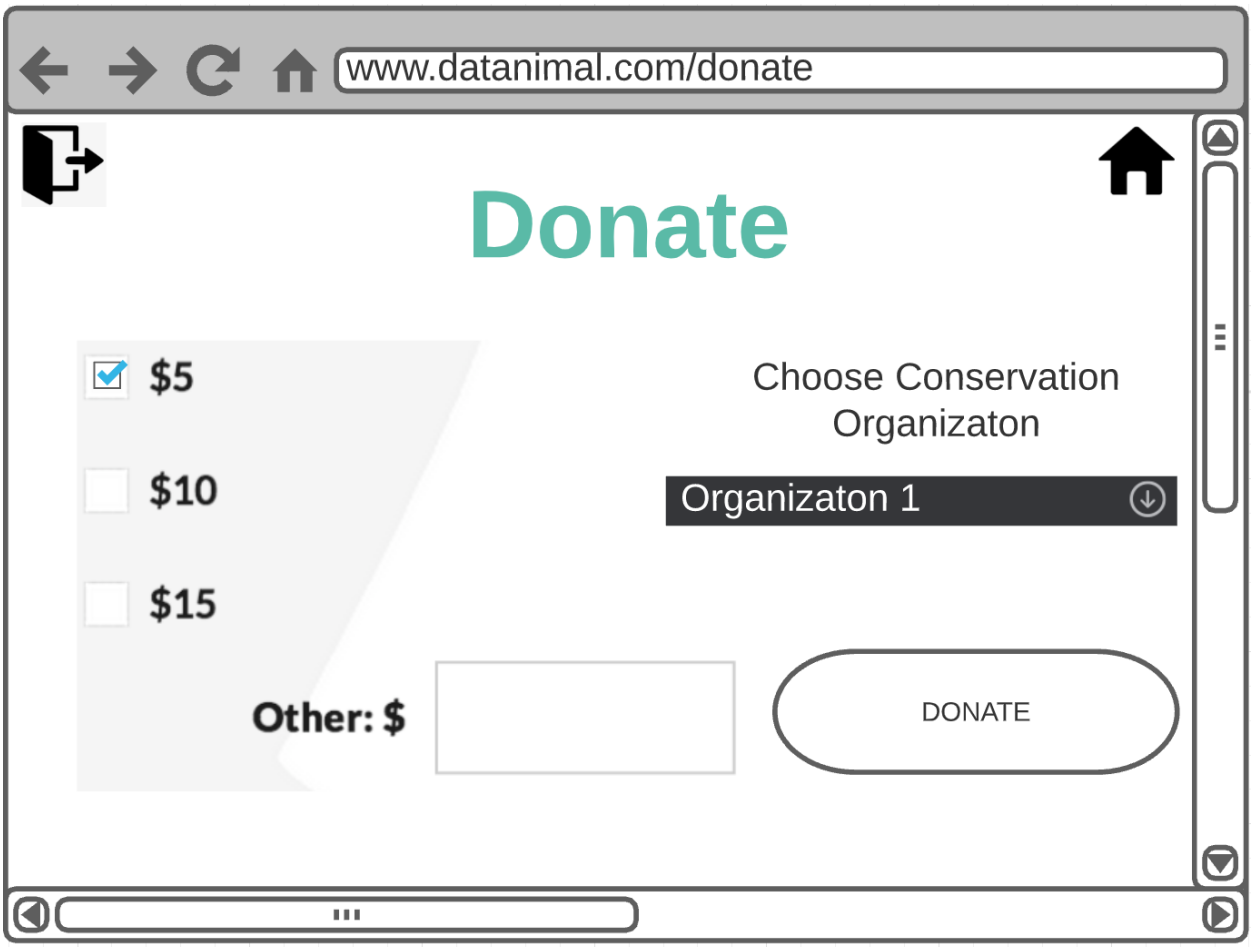
```
select id
from Users
where  email = EMAIL;
```

Create account:

```
insert into
Users( user_id, password, email_address, phone_no, birth_year, name, address, gender)
values(ID, PASSWORD, EMAIL, PHONE, BYEAR, NAME, ADDR, GENDER);

insert into
Employees( user_id, position, salary_per_hour, working_hour_per_week)
values(ID, POSITION, SALARY, WORKINGH);
```

3.6. Visitors Main Page

3.7. Visitors Donation



<u>Get current balance to check:</u>

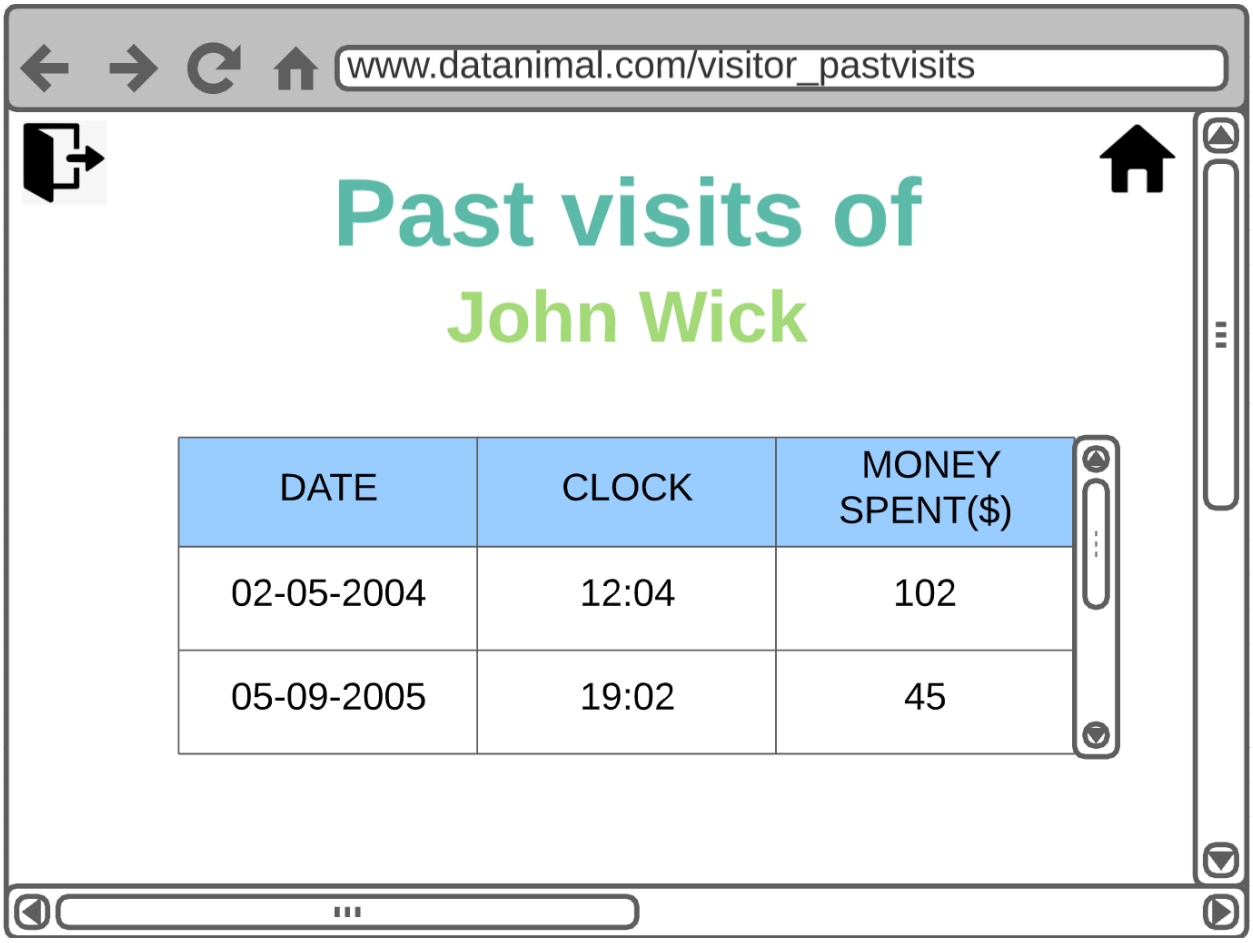
```
select total_amount_of_money
from Visitors
where user_id = ID;
```

<u>Update balances:</u>

```
update Visitors
set total_amount_of_money = total_amount_of_money - DONATION_AMOUNT
where user_id = ID;

update Conservation_Organizations
set collected_money = collected_money + DONATION_AMOUNT
```

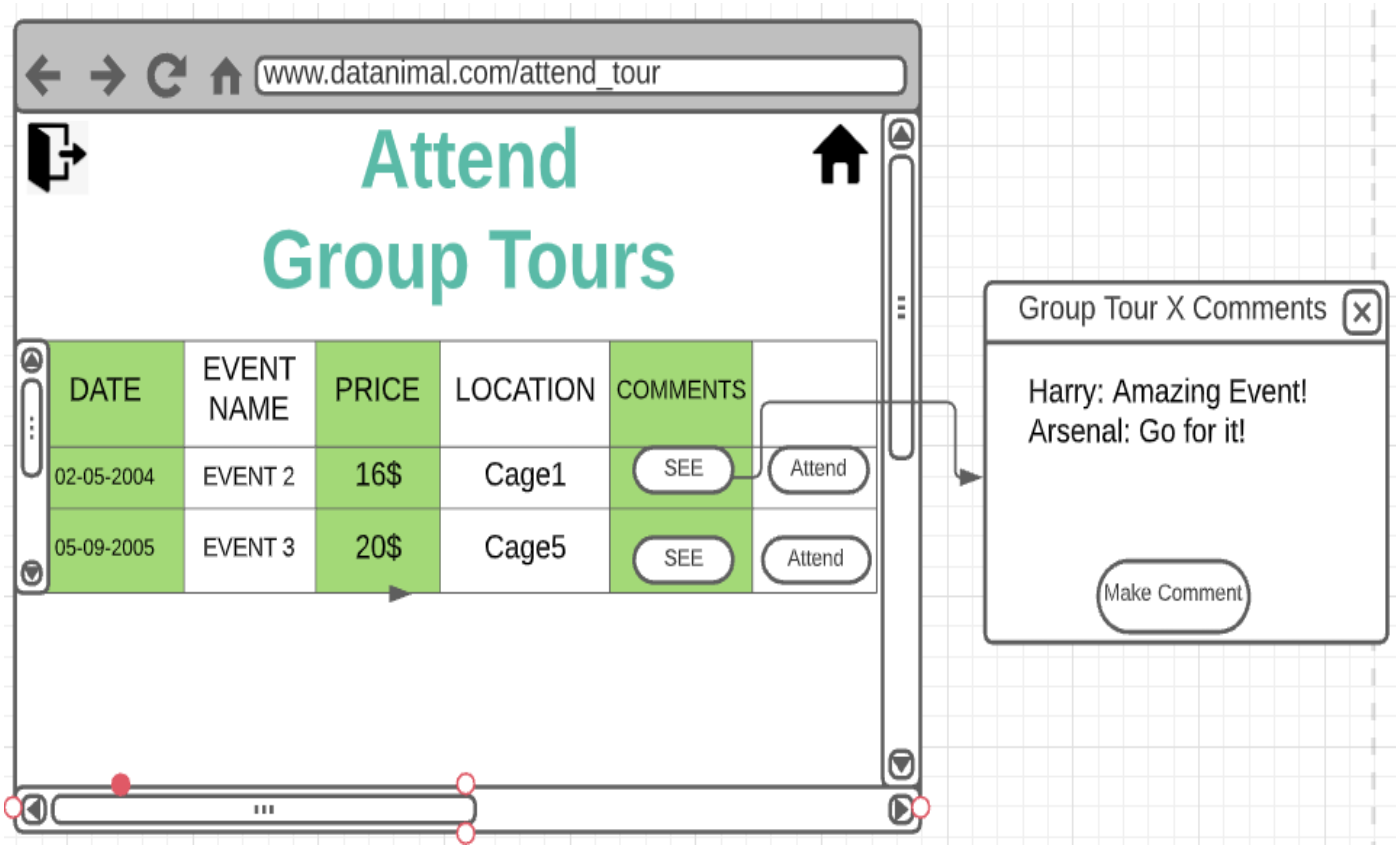
```
where event_id = ORG_ID;

insert into Make_donation (event_id, user_id, donation_amount)
values(ORG_ID, ID, DONATION_AMOUNT );
```

3.8. Visitors Past Visit




```
select date, entrance_time, exit_time, money_spent
from Visits
where user_id = ID;
```

3.9. Visitors Attend Group Tours



List events:

select date, name, price, location
from Events natural join Group_Tours
where date > CURRENT_DATE;

See comments:

select text, topic
from Writes_Comment natural join Comments
where event_id = TOURID;

Check available quota:

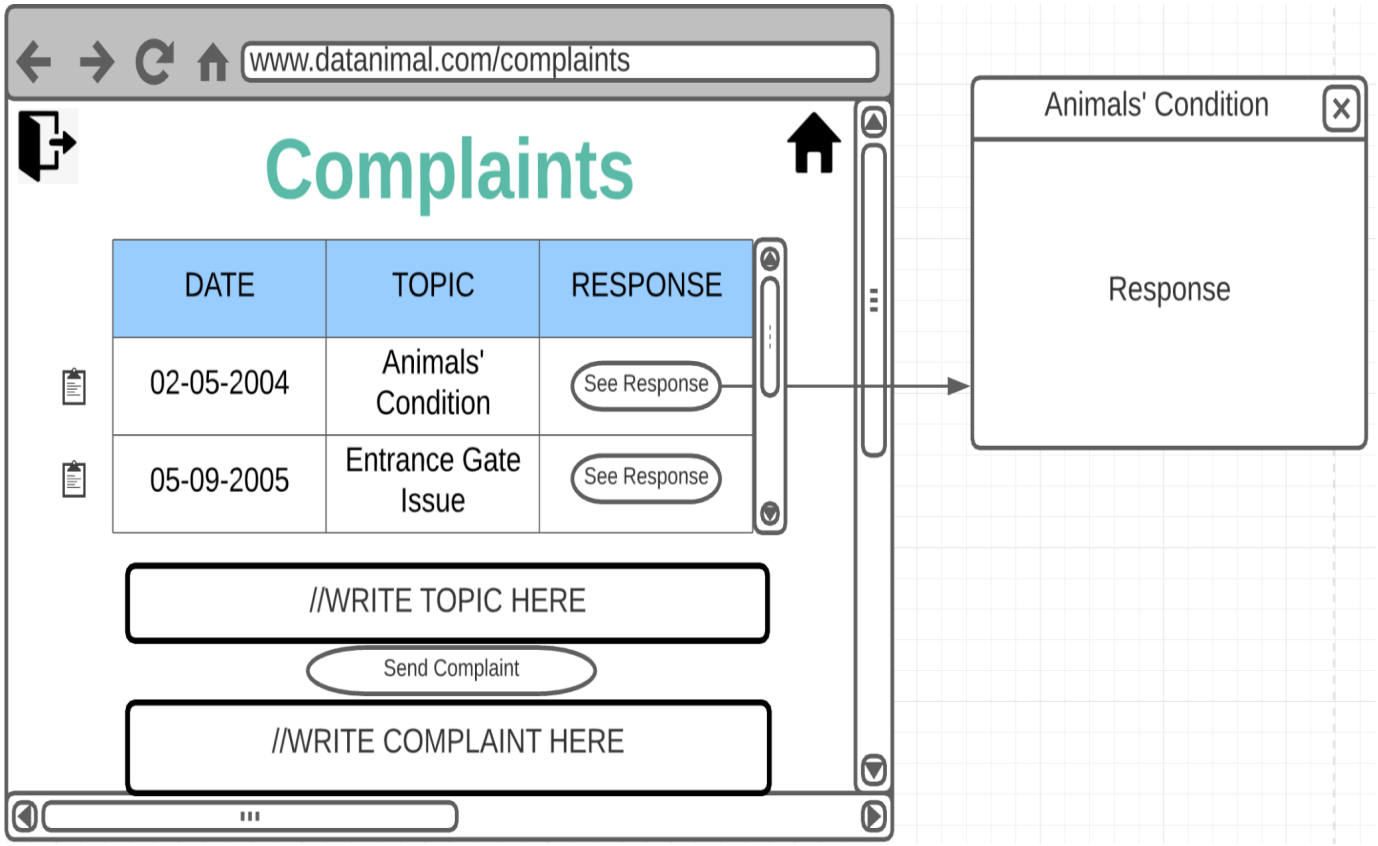
select event_id

```
from Group_Tours
where event_id = TOURID
and visitor_quota >
(select count(user_id)
from Attends
where event_id = TOURID
group by event_id);
```

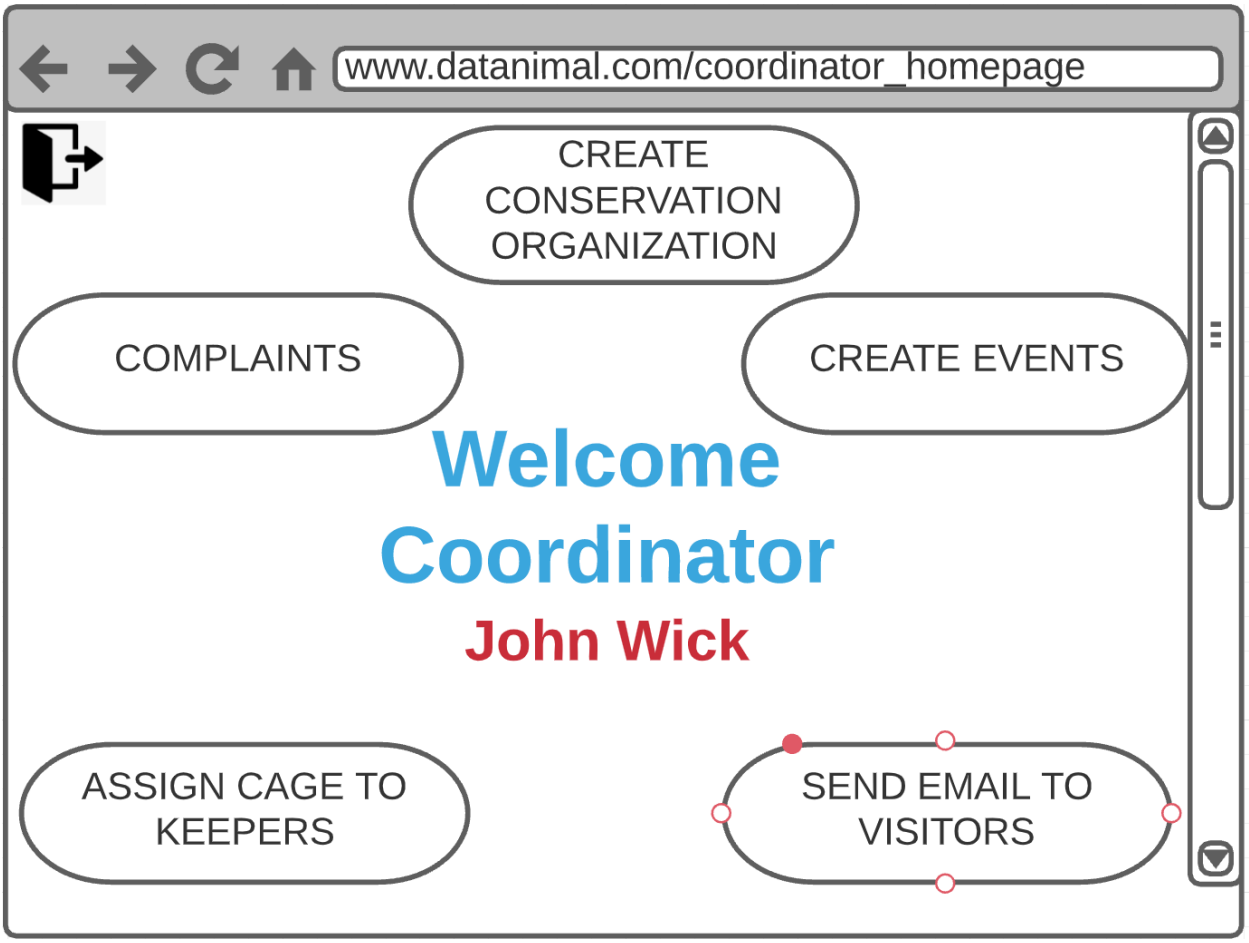
Attend if possible:

```
insert into Attends(user_id, event_id, payment)
values(ID, TOURID, PAYMENT);
```

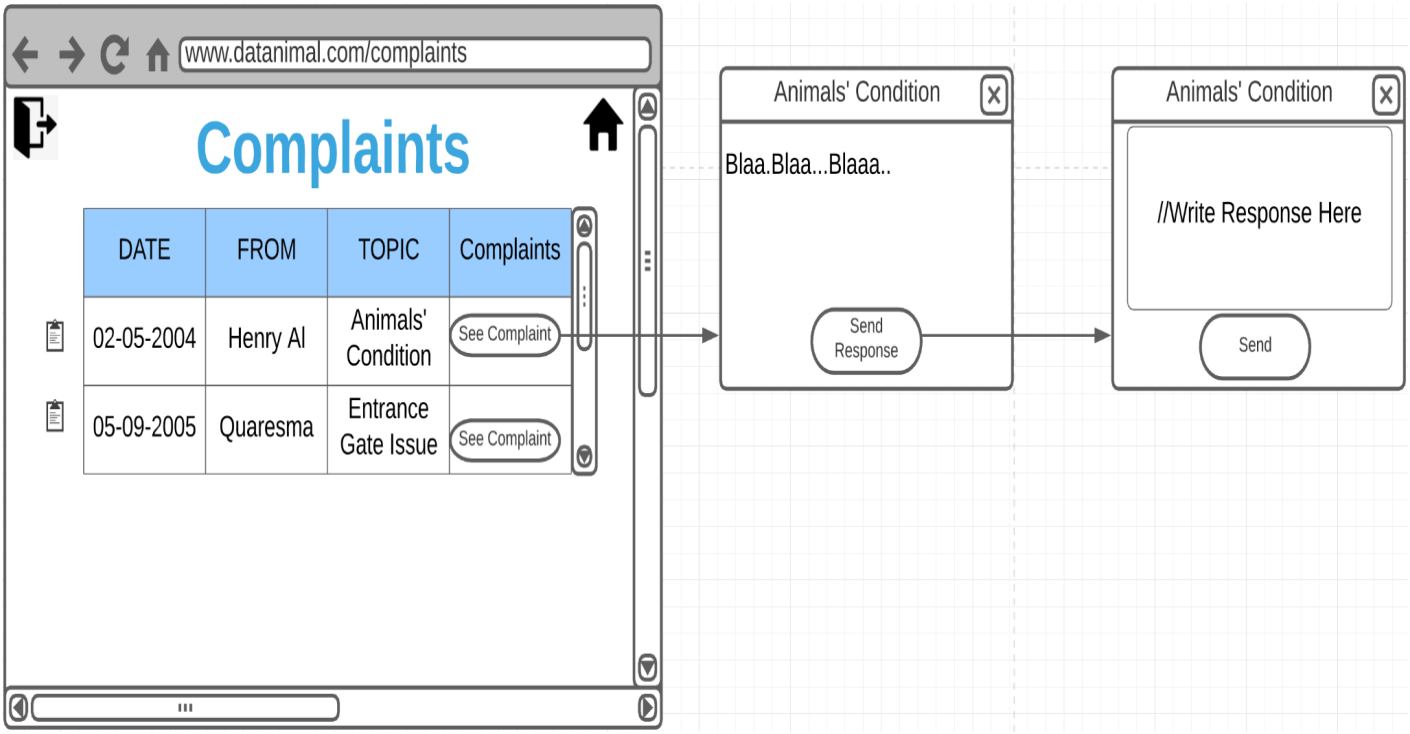
3.10. Visitors Complaints



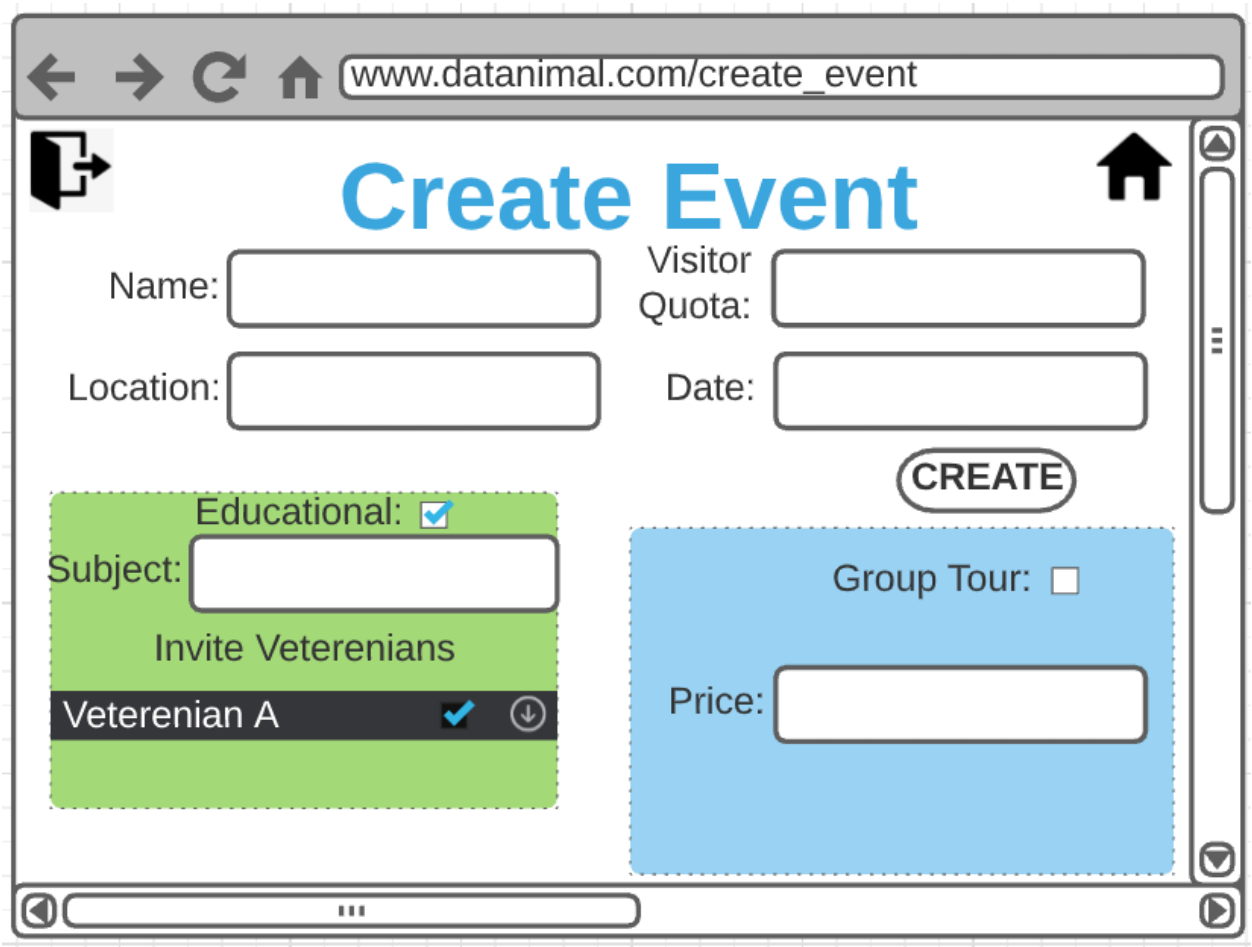
3.11. Employees Main Page



3.12. Employees Complaints



3.13. Employees Create Events



Any type of event:
insert into Events (event_id, date, location, user_id, name)
values(NEWEVENTID, DATE, LOCATION, ID, NAME);

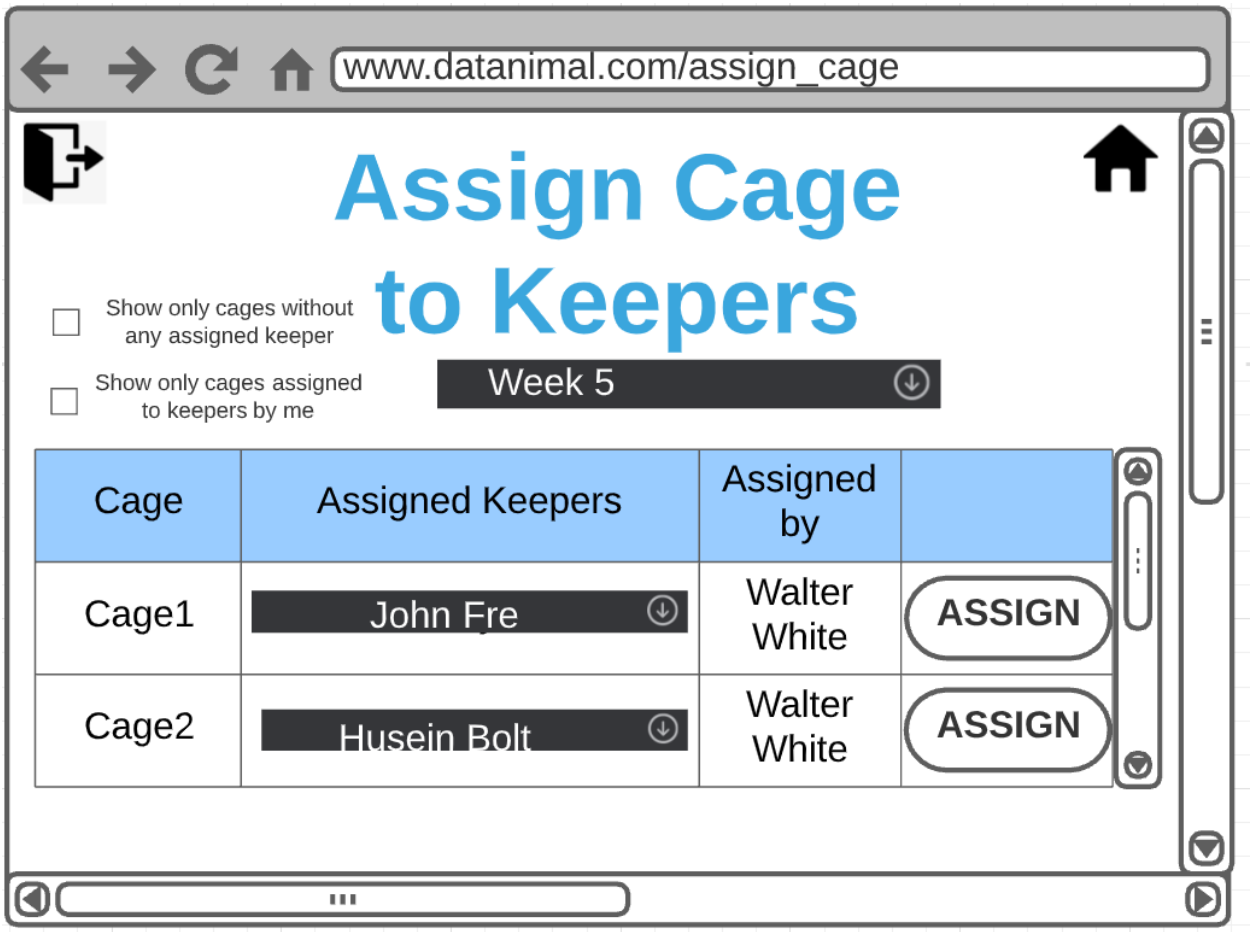
Educational:
insert into Educational_Program(event_id, visitor_qouta, subject)
values(NEWEVENTID, QUOTA, SUBJECT);

Group:
insert into Group_Tours(event_id, visitor_qouta, price)
values(NEWEVENTID, QUOTA, PRICE);

Conservation:

```
insert into Conservation_Organizations (event_id, collectedMoney)
values(NEWEVENTID, 0);
```

3.14. Assign Cages To Keepers



<u>List not assigned cages:</u>

```
select cage_id
from ((select cage_id
from Cages  )
except (
select cage_id
from Assigns_Cage)) natural join Cages natural join Assigns_Cage ;
```

<u>Assign cage to keeper:</u>

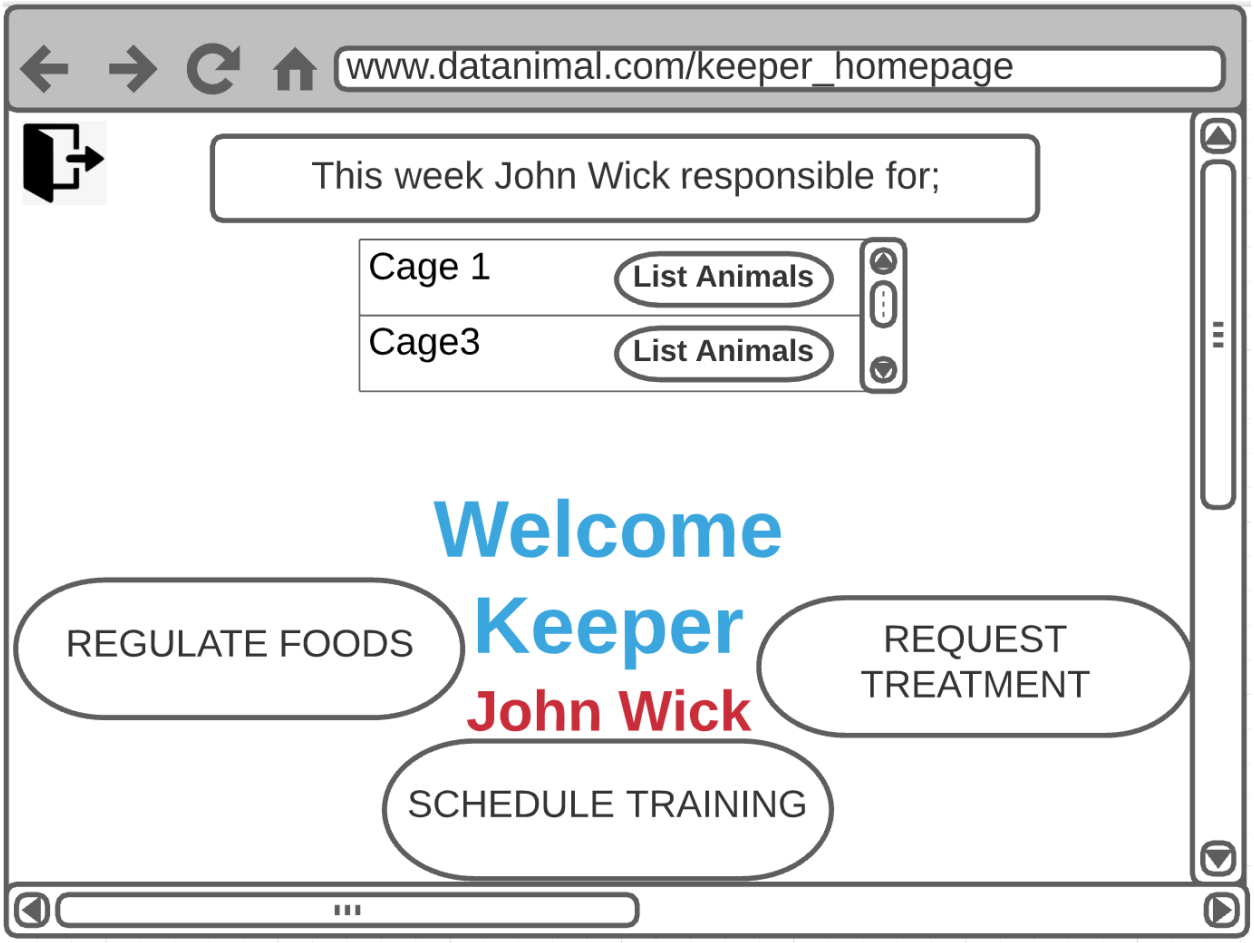
```
insert into Assigns_Cage(coordinator_user_id, keeper_user_id, cage_id)
values(ID, KEEPERID, CAGEID);
```

3.15. Send Email to Visitors

3.16. Create Conservation Organization

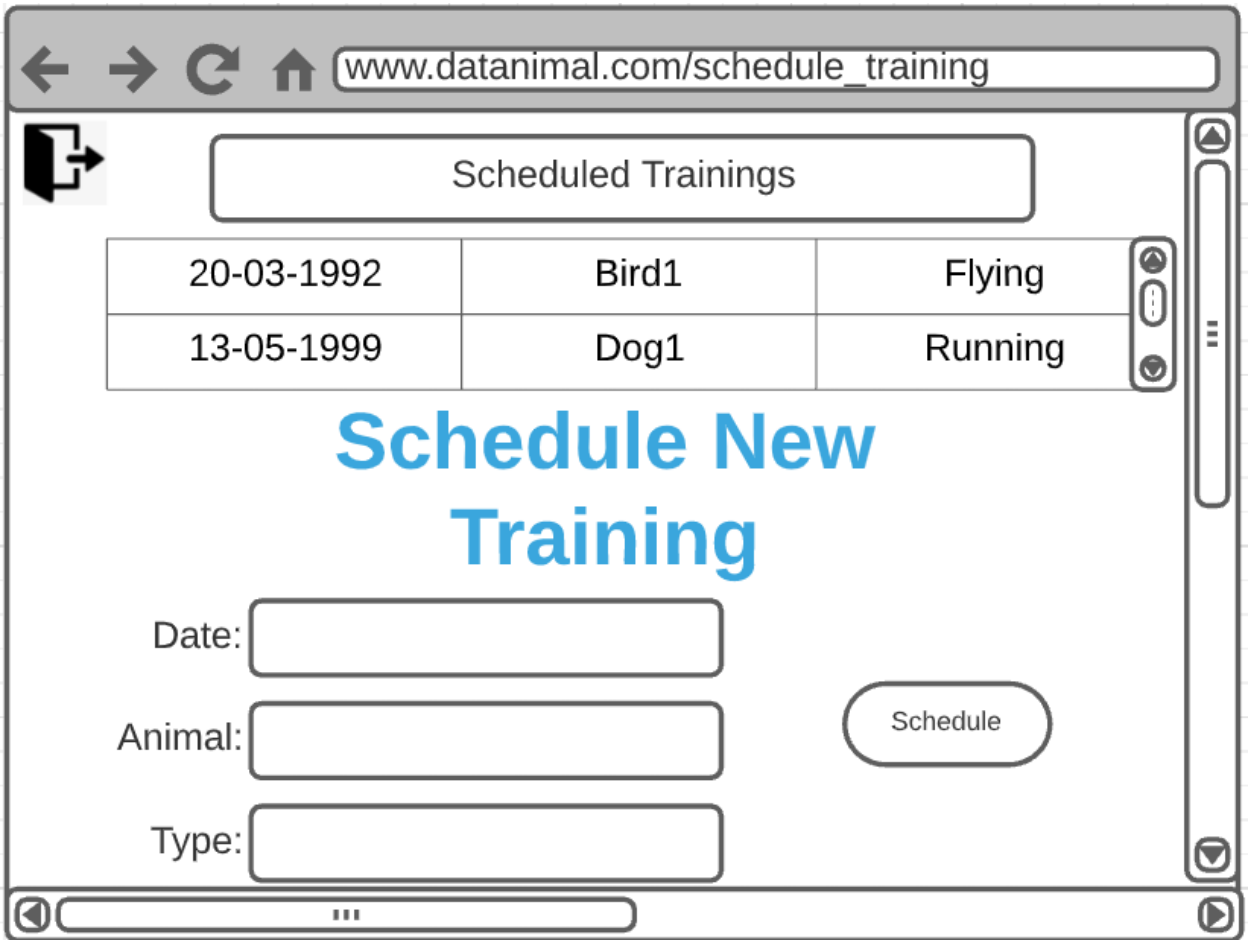

```
insert into Events (event_id, date, location, user_id, name)
values(NEWEVENTID, DATE, LOCATION, ID, NAME);


insert into Conservation_Organizations (event_id, collectedMoney)
values(NEWEVENTID, 0);
```

3.17. Keeper Main Page



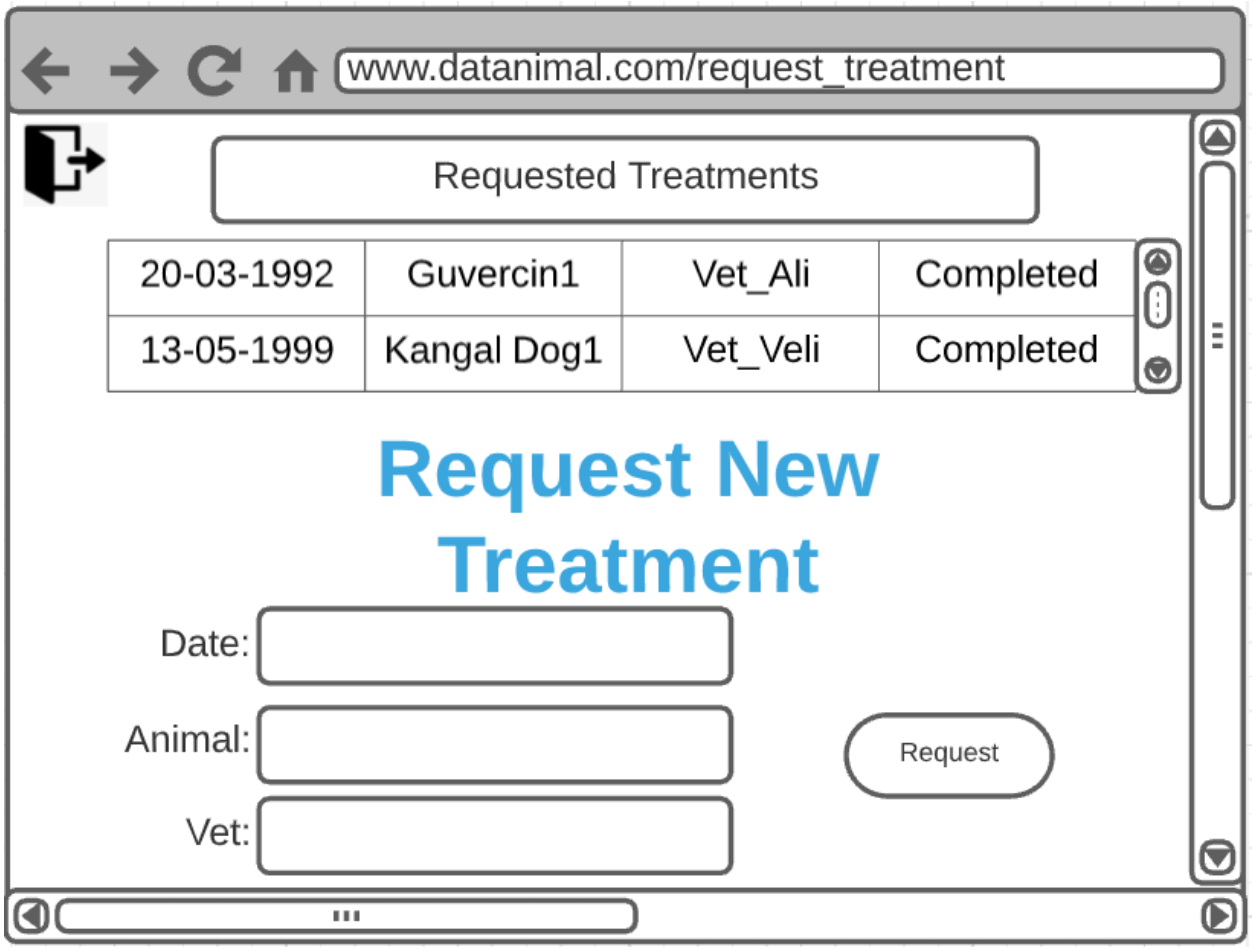

```
select cage_id
from Assigns_Cage
where keeper_user_id = ID;
```

3.18. Schedule New Training



insert into Schedules_Training(animal_id, user_id, date_time, type)
values(ANID, ID, DATETIME, TYPE);

3.19. Request New Treatment



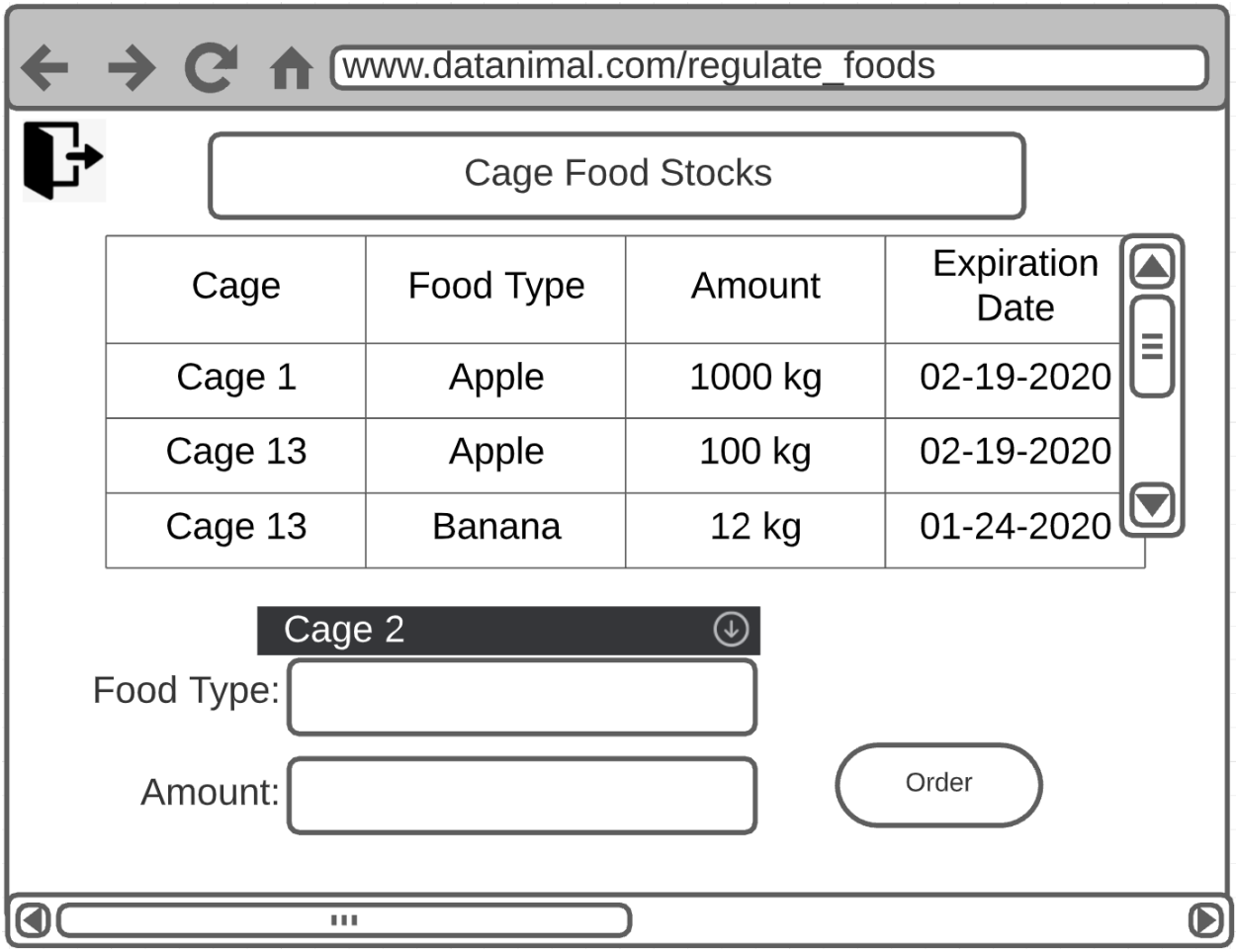
List:

select *
from Animals natural Join Request_Treatment
where user_id = ID;

New Request:

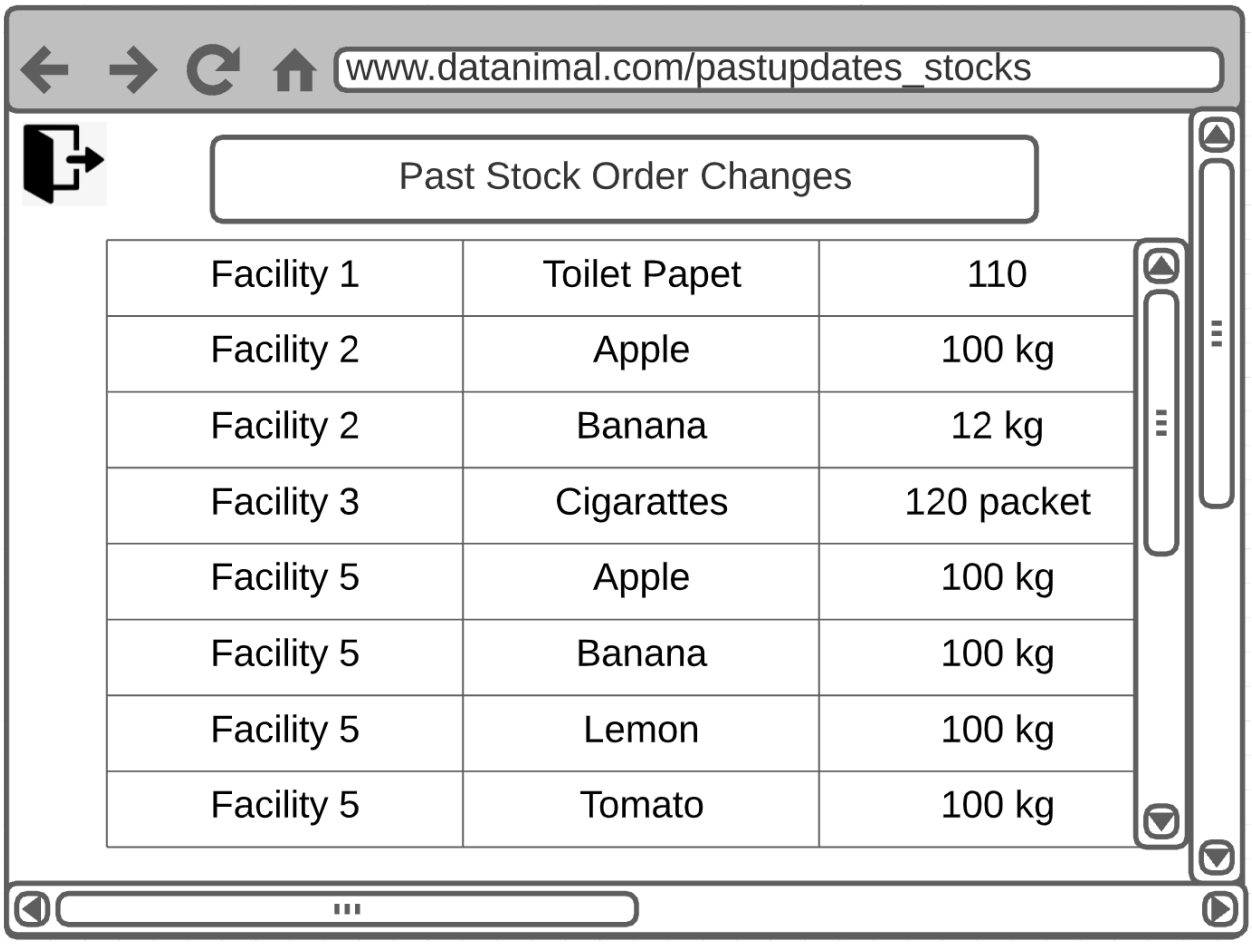
insert into Request_Treatment(animal_id, keeper_user_id, vet_user_id, date_time, treatment_status, acc_status)
values(ANID, ID, VETID, DATETIME, ‘waiting approval’, ‘not accepted’);

3.20. Cage Food Stock



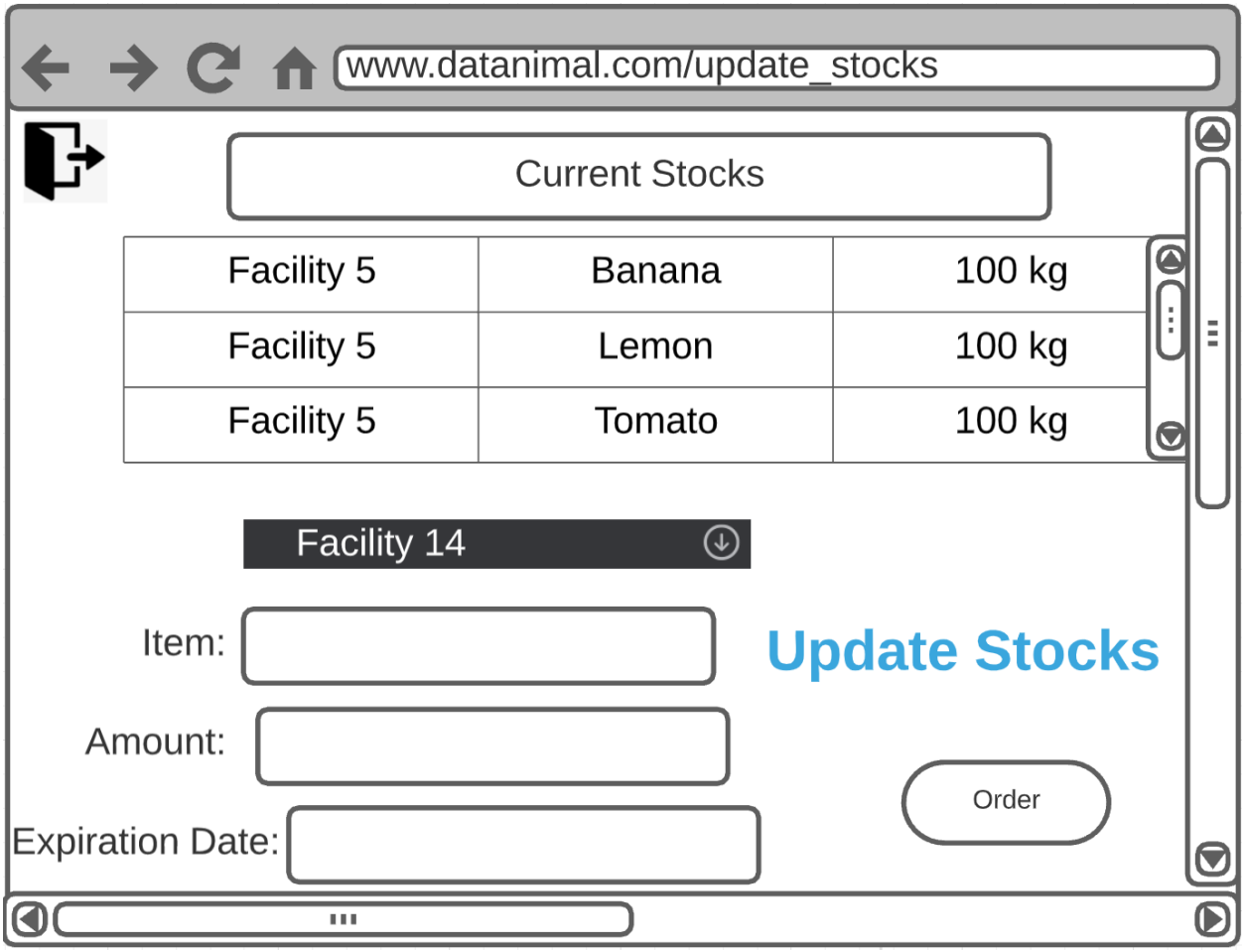
3.21. Facility Worker Main Page

3.22. Past Stock Order Changes



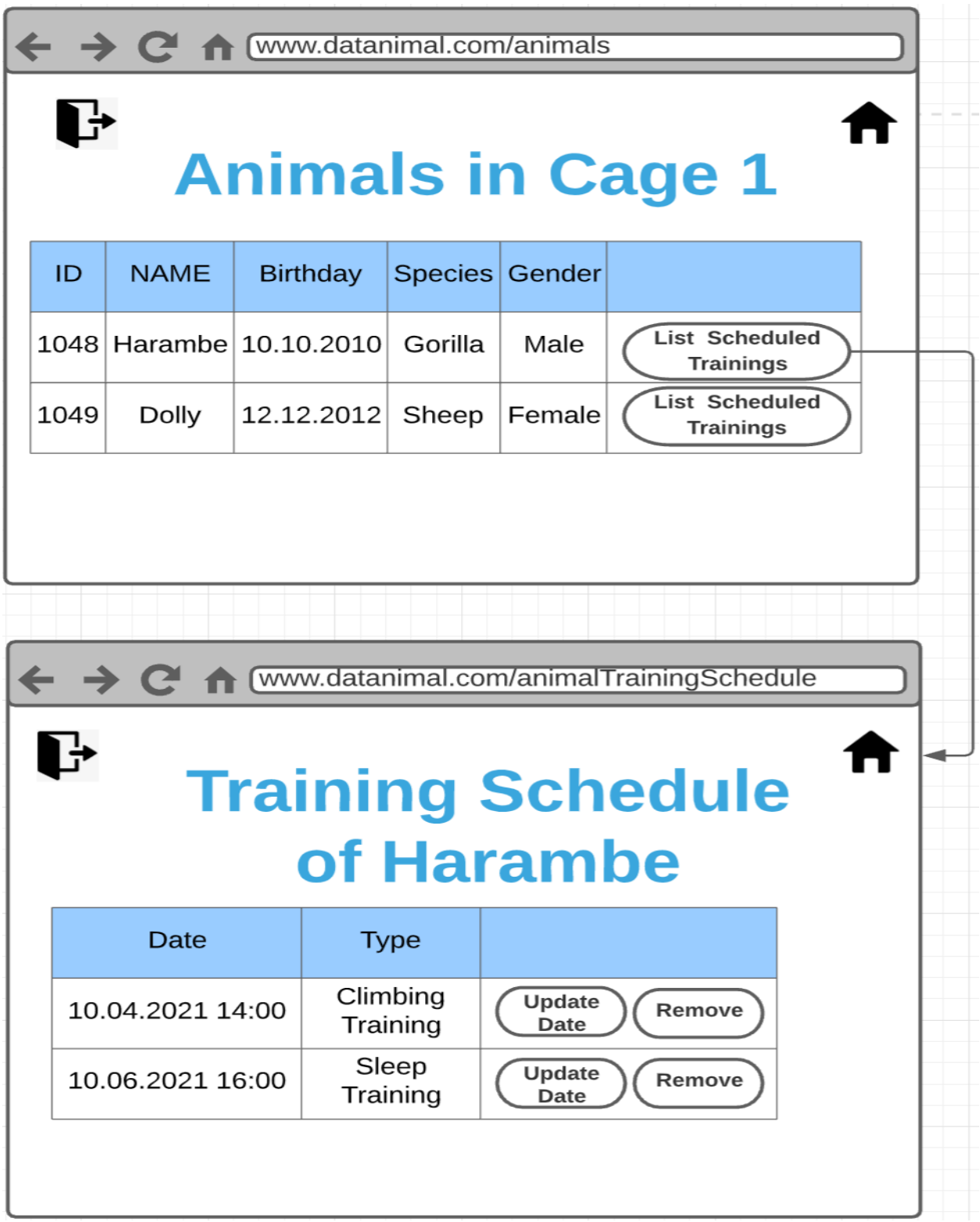
Facility 1	Toilet Papet	110
Facility 2	Apple	100 kg
Facility 2	Banana	12 kg
Facility 3	Cigarattes	120 packet
Facility 5	Apple	100 kg
Facility 5	Banana	100 kg
Facility 5	Lemon	100 kg
Facility 5	Tomato	100 kg

3.23. Current Stock



3.24. Veterinarian Main Page

3.25. Animals in Cage



```
select animal_id, name, birthday, species, gender
from Cages natural join Animals natural join Assigns_ Cage
where user_id = ID;
```

List trainings:

```
select date_time, type
from Schedules_Training
where animal_id = ANID;
```

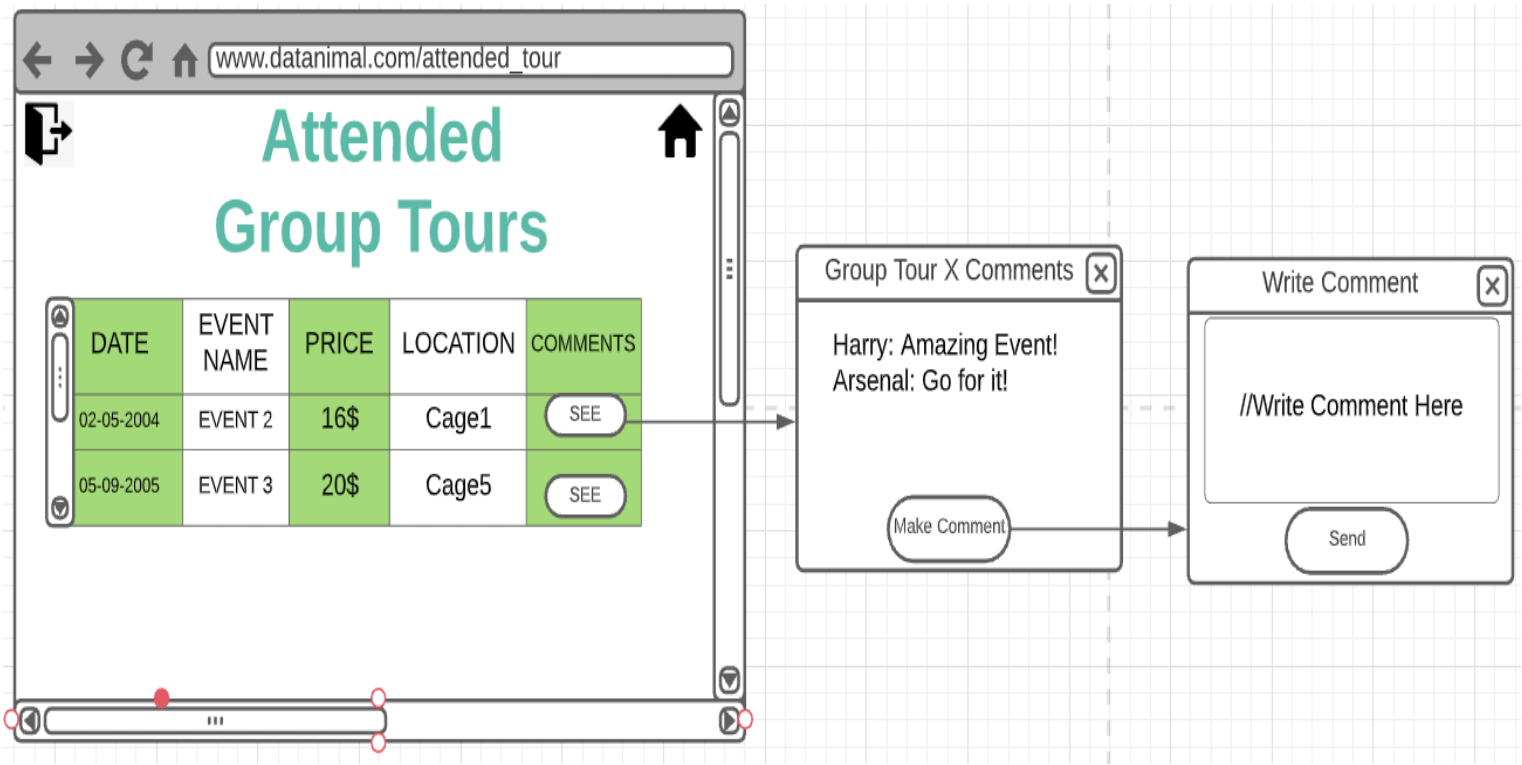
Reschedule:

```
update Schedules_Training
set date_time = NEWDATETIME
where animal_id = ANID
and user_id = ID
and date_time = CURRENTDATETIME;
```

Cancel training:

```
delete from Schedules_Training
where animal_id = ANID
and user_id = ID
and date_time = CURRENTDATETIME;
```

3.26. Visitors Comment on Attended Group Tours



<u>List attended tours:</u>

select date, name, price, location
from Group_Tours natural join Attends natural join Events
where user_id = ID;

<u>Make comment:</u>

insert into Comments(comment_id, topic, text, date)
values(COMMID, TOPIC, TEXT, DATE);

insert into Writes_Comment(comment_id, event_id, user_id)
values(COMMID, TOURID, ID);