



**Bilkent University**

**Department of Computer Engineering**

---

# CS353 Term Project Design Report

## DatAnimal

**Assigned Teacher Assistant**

Duygu Durmuş

Group 34

**Team Members**

Asım Güneş Üstüinalp

Radman Lotfiazar

Turan Mert Duran

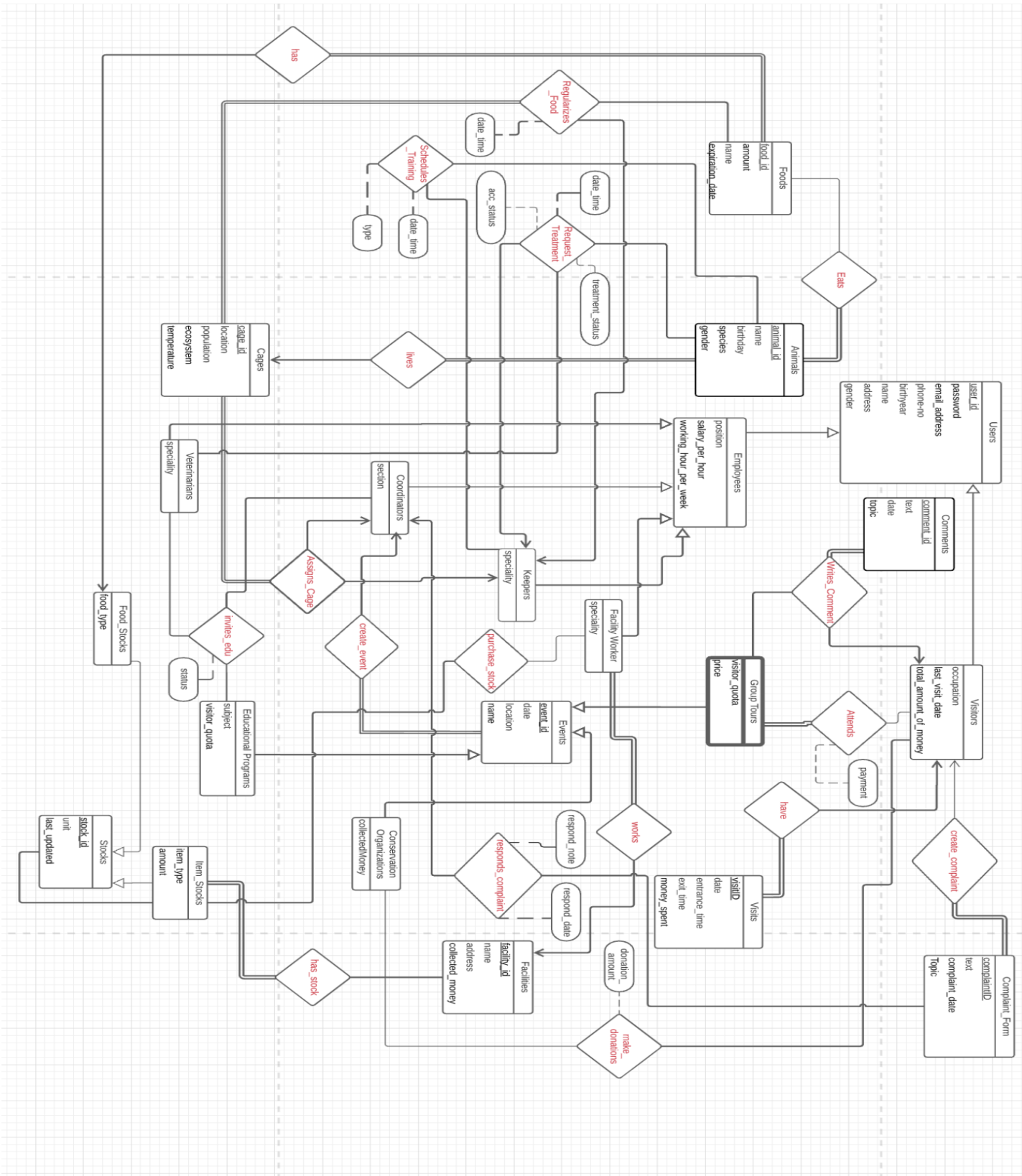
Berdan Akyürek

# Table of Content

<b>Revised ER Diagram</b>	<b>4</b>
<b>Table Schemas</b>	<b>5</b>
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
<b>User Interface Design and Database Statement</b>	<b>26</b>
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( user\_id, 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( user\_id, 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( user\_id, 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( user\_id, 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( user\_id, 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( user\_id, 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 ( user\_id, 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(stock\_id, 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(event\_id, 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)

### Foreign 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 (  
  complaint_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(animal\_id, keeper\_user\_id, vet\_user\_id, date\_time,  
treatment\_status, acc\_status )

### Candidate Keys

({animal\_id, keeper\_user\_id, vet\_user\_id, date\_time})

### Table Definition

<b>create table</b>	Regularizes_Food (
animal_id	<b>int</b> not null,
keeper_user_id	<b>int</b> not null,
vet_user_id	<b>int</b> not null,
date_time	<b>datetime</b> not null,
treatment_status	<b>varchar(30)</b> ,
acc_status	<b>varchar(30)</b> ,
<b>primary key</b>	(animal_id, keeper_user_id, vet_user_id, date_time),
<b>foreign key</b>	(food_id) references Foods,
<b>foreign key</b>	(keeper_user_id) references Users,
<b>foreign key</b>	(vet_user_id) references Users,
<b>foreign key</b>	(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(coordinator\_user\_id, keeper\_user\_id, cage\_id)

### 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      (event_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** (event\_id) references Events  
 );

## 2.32. Has\_Stock

### Rational Model

Has\_Stock (stock\_id, facility\_id)



### **Foreign 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 (facility_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,  
);
```

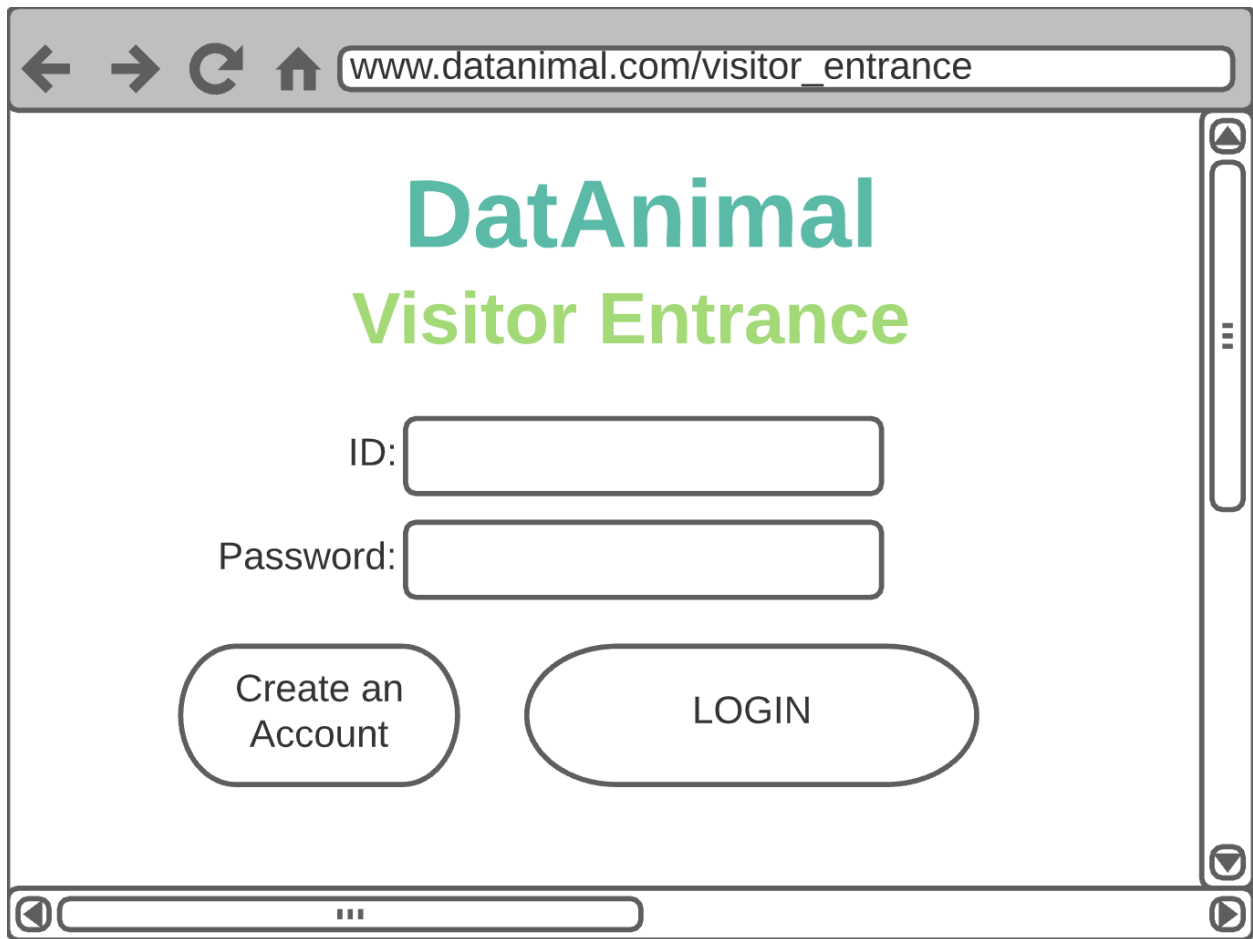
### 3. User Interface Design and Database Statement

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



### 3.2. Visitors Login Page



The image shows a web browser window with the address bar displaying `www.datanimal.com/visitor_entrance`. The page content includes the title "DatAnimal" in teal and "Visitor Entrance" in green. Below the title are two input fields: "ID:" and "Password:". At the bottom, there are two buttons: "Create an Account" and "LOGIN".

DatAnimal  
Visitor Entrance

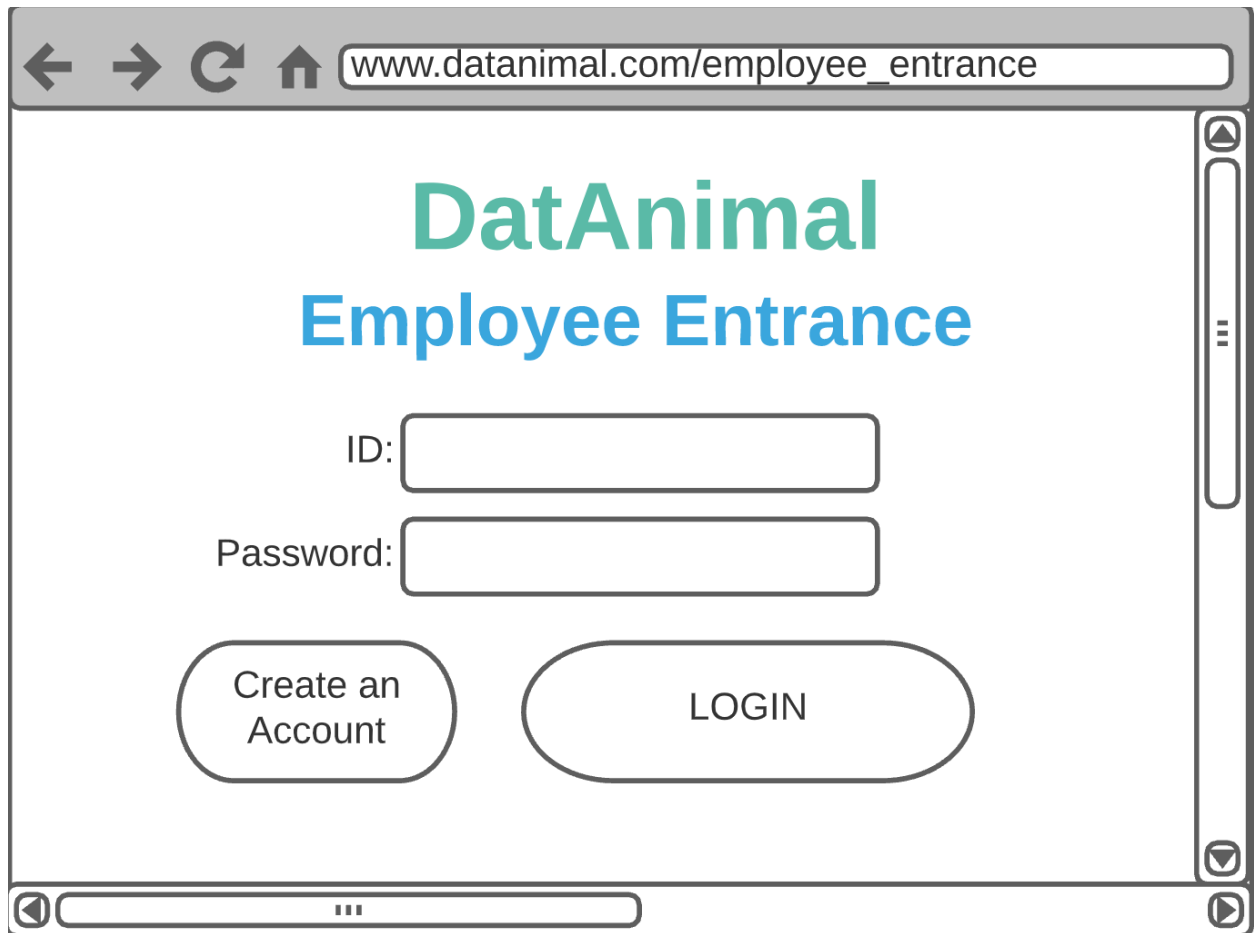
ID:

Password:

Create an Account LOGIN

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

### 3.3. Employees Login Page



The image shows a web browser window displaying the 'DatAnimal Employee Entrance' login page. The browser's address bar shows the URL 'www.datanimal.com/employee\_entrance'. The page features the 'DatAnimal' logo in green and 'Employee Entrance' in blue. Below the title, there are two input fields: 'ID:' and 'Password:'. At the bottom, there are two buttons: 'Create an Account' and 'LOGIN'. The browser window includes standard navigation icons (back, forward, refresh, home) and a scrollbar on the right side.

← → ↻ ⬆ www.datanimal.com/employee\_entrance

# DatAnimal

## Employee Entrance

ID:

Password:

Create an Account LOGIN

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

### 3.4. Visitors Sign Up Page

The screenshot shows a web browser window with the address bar displaying `www.datanimal.com/create_v_account`. The page content includes the **DatAnimal** logo and the heading **Create Visitor Account**. The form consists of the following fields and labels:

- Password:
- Birthyear:
- Name:
- E-mail:
- Address:
- Phone-no:
- Gender:
- Occupation:

A button labeled "Create an Account" is located at the bottom right of the form area.

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

### 3.5. Employees Sign Up Page

← → ↻ ⬆ www.datanimal.com/create\_e\_account

## Create Employee Account

ID:  Birthyear:

Password:  Name:

E-mail:  Address:

Phone-no:  Gender:

Occupation:  Salary:

Position:  Working Hour:

#### **Check existence of email:**

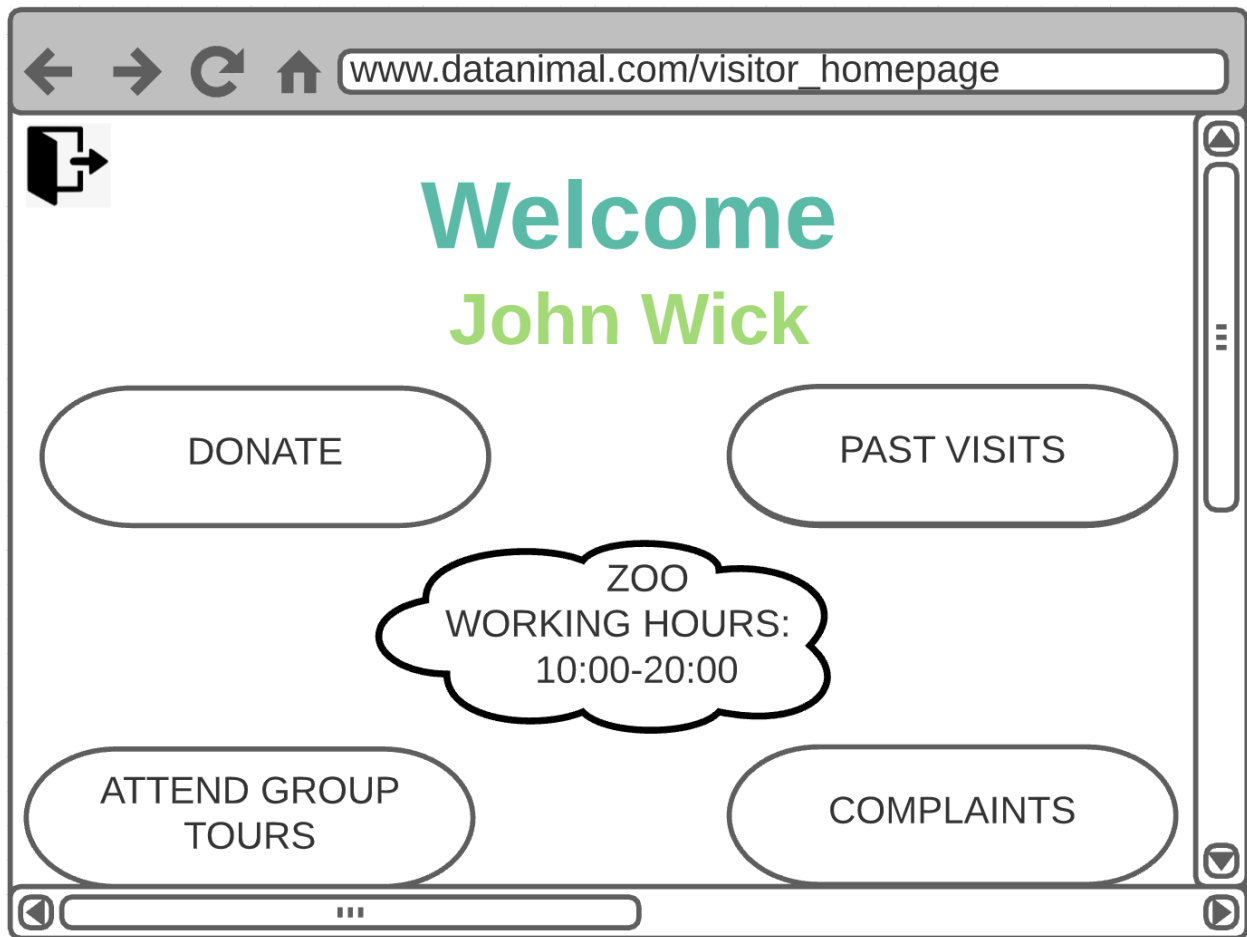
```
select id
from Users
where email = EMAIL;
```

#### **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

← → ↻ 🏠 [www.datanimal.com/donate](http://www.datanimal.com/donate)

🏠

# Donate

☒ \$5  
☐ \$10  
☐ \$15

Other: \$

Choose Conservation Organizaton

Organizaton 1 ⌵

DONATE

**Get current balance to check:**

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

**Update balances:**

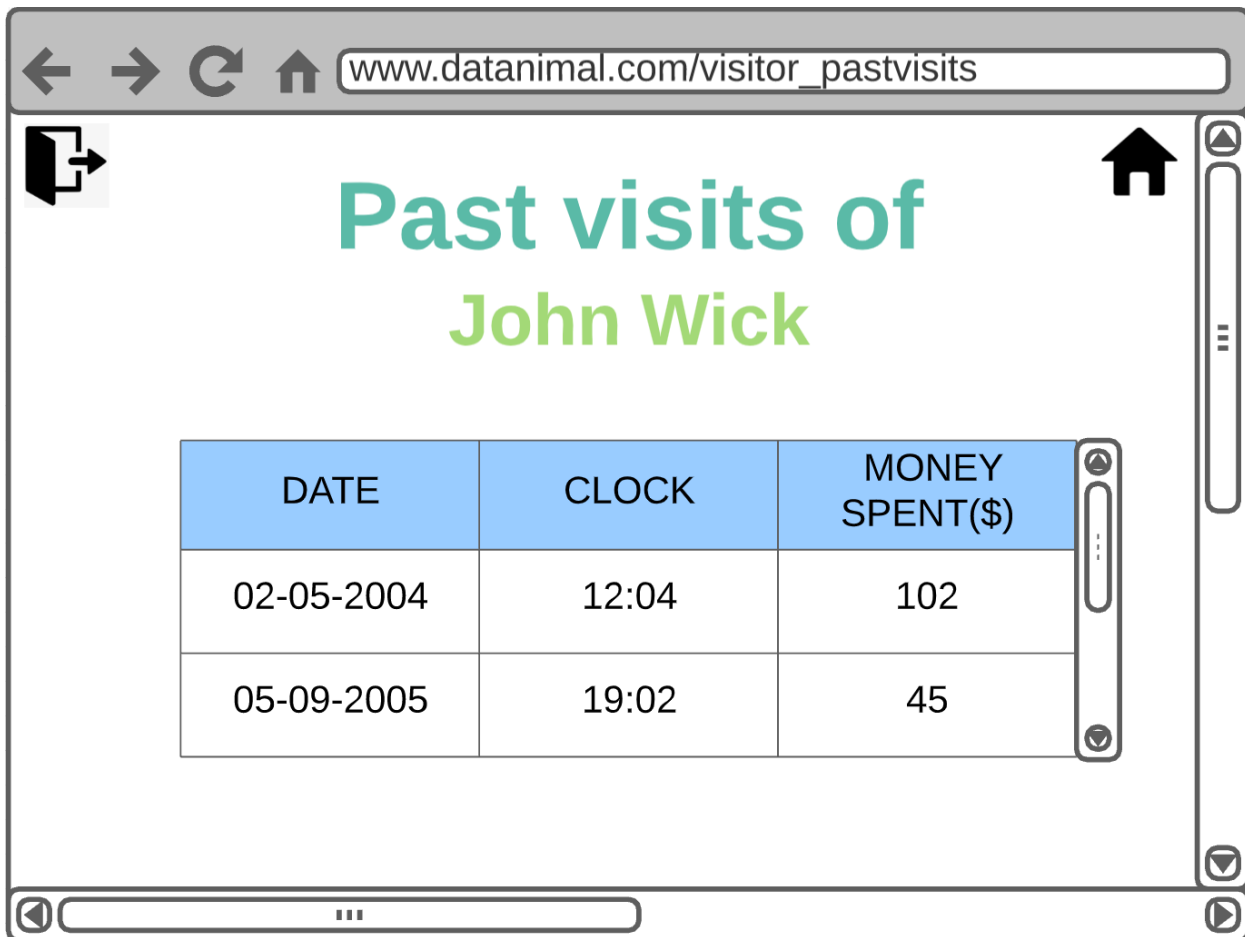
```
update Visitors  
set total_amount_of_money = total_amount_of_money - DONATION_AMOUNT  
where user_id = ID;
```

```
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

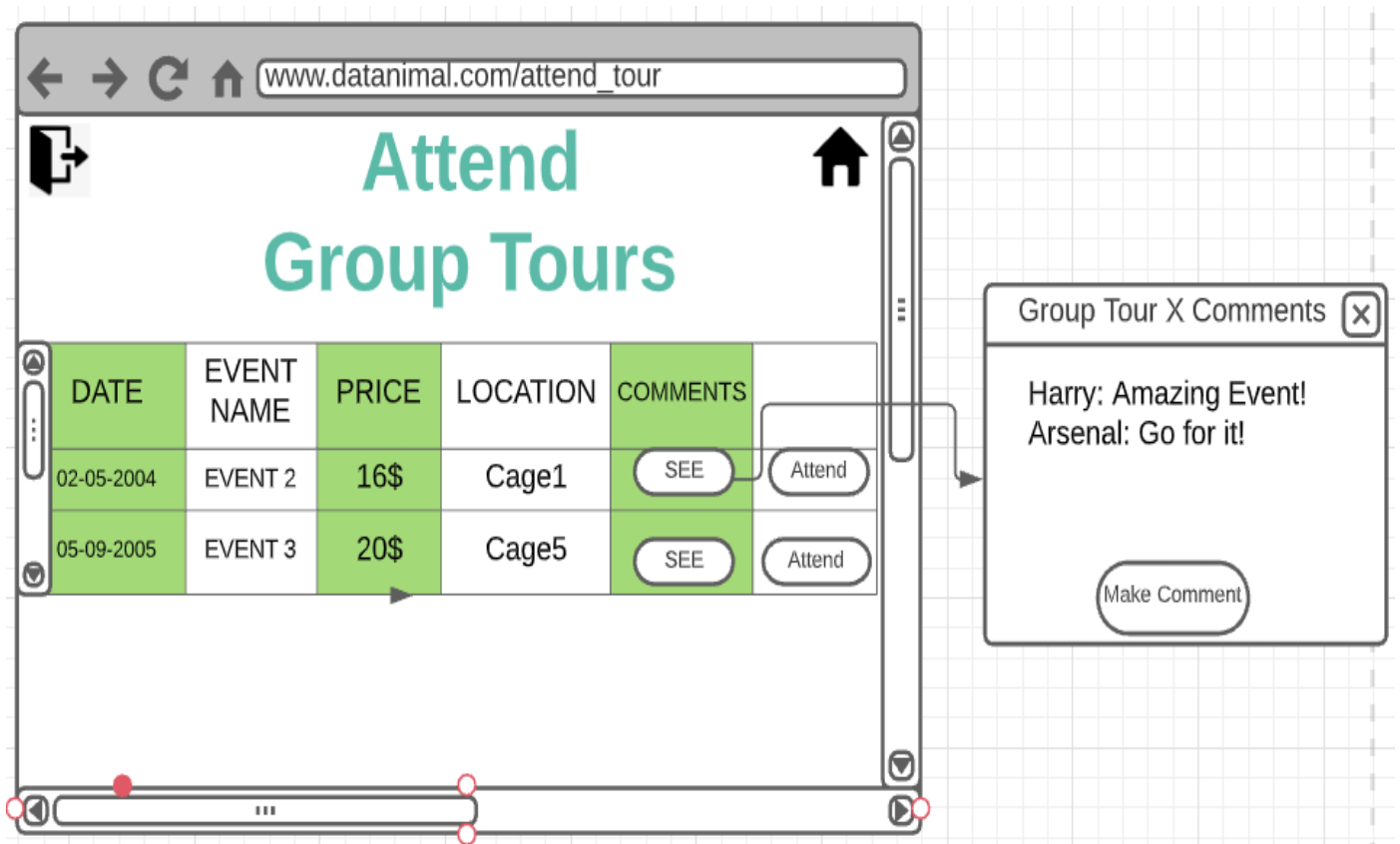


The screenshot shows a web browser window with the address bar displaying `www.datanimal.com/visitor_pastvisits`. The page title is "Past visits of John Wick". Below the title is a table with three columns: DATE, CLOCK, and MONEY SPENT(\$). The table contains two rows of data. The browser interface includes navigation buttons (back, forward, refresh, home), a search icon, and a home button. A scrollbar is visible on the right side of the page content area.

DATE	CLOCK	MONEY SPENT(\$)
02-05-2004	12:04	102
05-09-2005	19:02	45

```
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:**

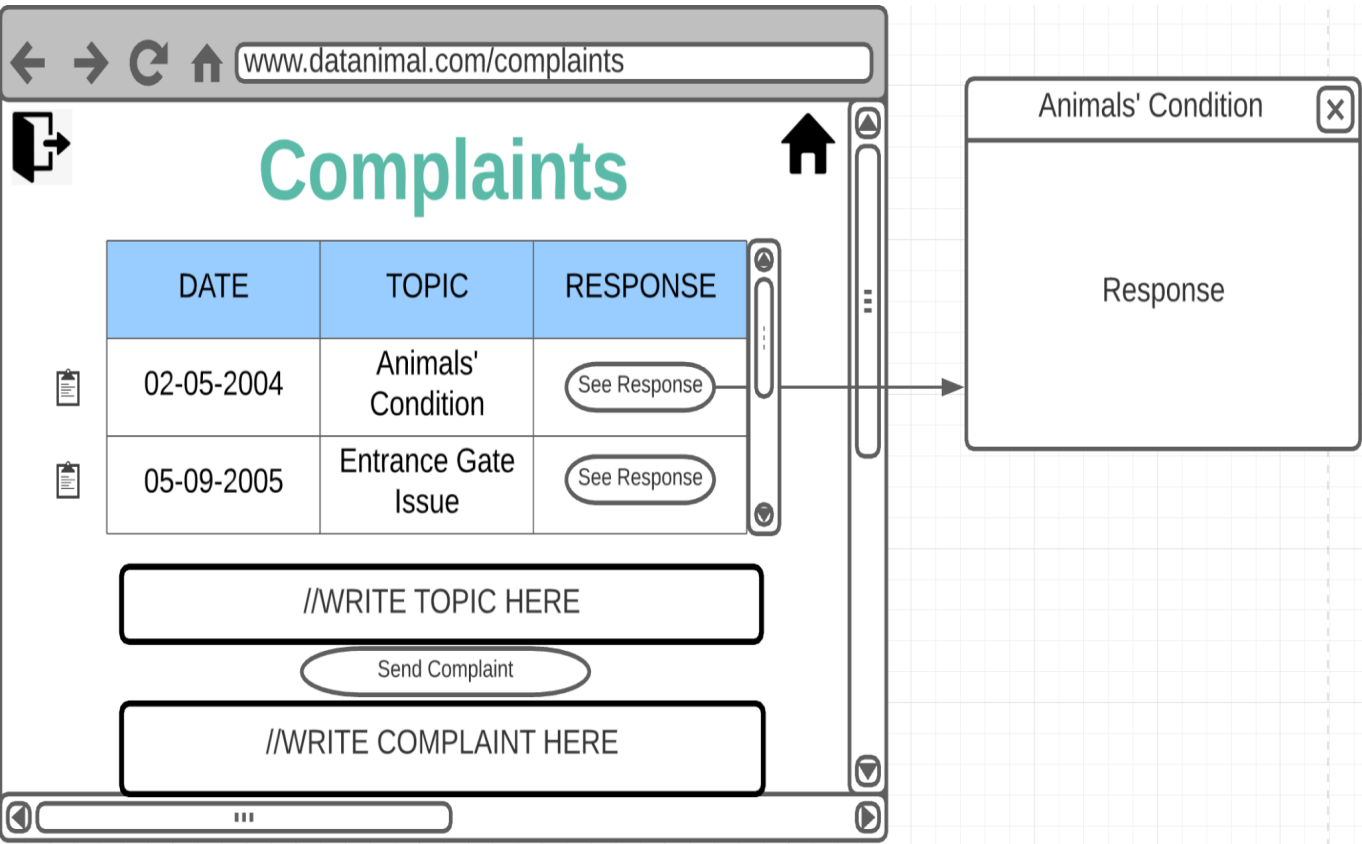
```
select event_id
```

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

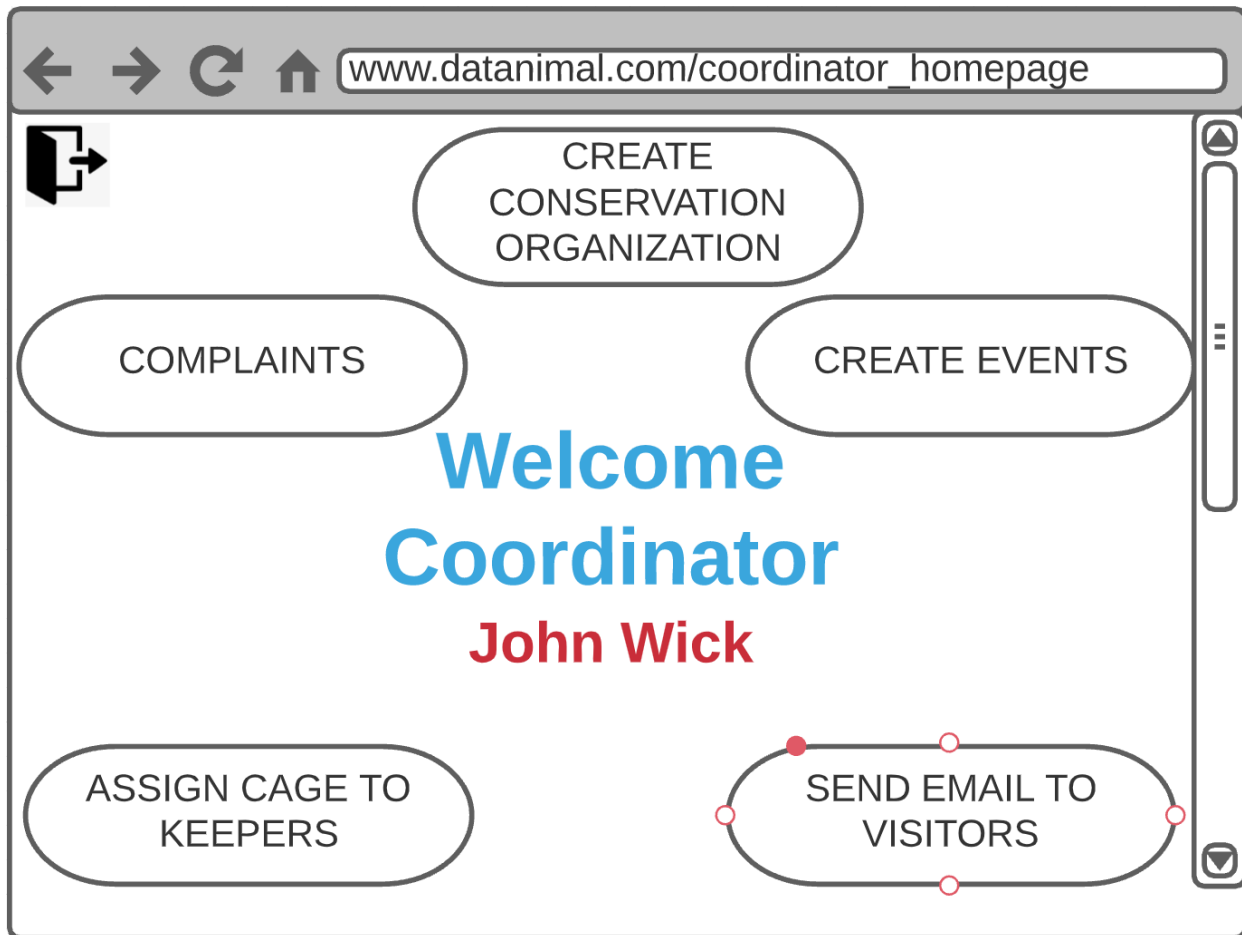
**Attend if possible:**

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

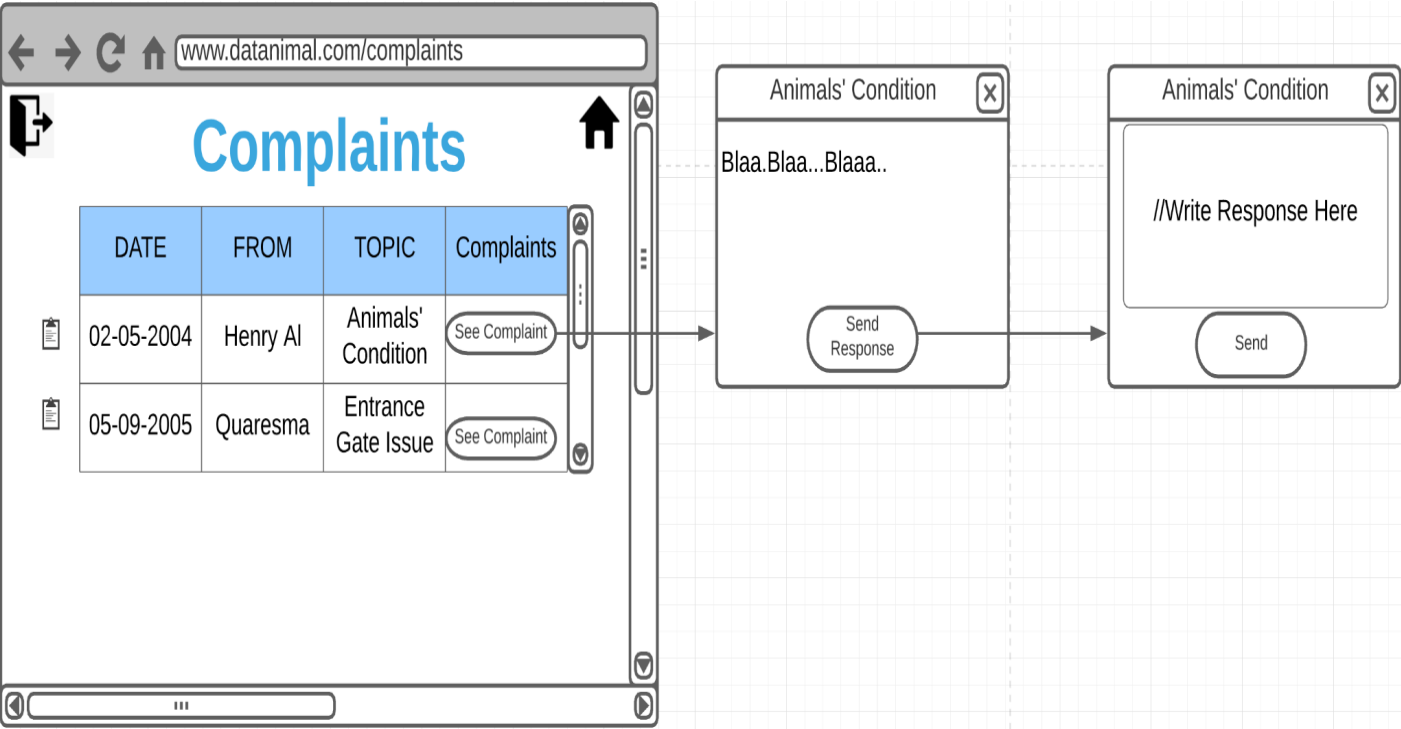
### 3.10. Visitors Complaints



### 3.11. Employees Main Page



### 3.12. Employees Complaints





### 3.13. Employees Create Events

The screenshot shows a web browser window with the address bar displaying `www.datanimal.com/create_event`. The page title is "Create Event". The form contains the following fields and controls:

- Name:** Text input field.
- Location:** Text input field.
- Visitor Quota:** Text input field.
- Date:** Text input field.
- CREATE:** A button to submit the form.
- Educational:** A checkbox that is checked.
- Subject:** Text input field.
- Invite Veterenians:** A section with a dropdown menu showing "Veterenian A" with a checkmark and a download icon.
- Group Tour:** A checkbox that is unchecked.
- Price:** Text input field.

**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

← → ↻ ↑ [www.datanimal.com/assign\\_cage](http://www.datanimal.com/assign_cage)

## Assign Cage to Keepers

☐ Show only cages without any assigned keeper

☐ Show only cages assigned to keepers by me

Week 5

Cage	Assigned Keepers	Assigned by	
Cage1	John Fre	Walter White	ASSIGN
Cage2	Husein Bolt	Walter White	ASSIGN

#### **List not assigned cages:**

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

#### **Assign cage to keeper:**

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

### 3.15. Send Email to Visitors



The image shows a web browser window with the address bar displaying `www.datanimal.com/send_email`. The page content includes a large blue heading "Send E-mail to Visitors", a dark button labeled "Event 1" with a download icon, a text input field labeled "Title", a larger text area labeled "//Inside" with a vertical scrollbar, and a cloud-shaped "Send" button. The browser interface includes standard navigation buttons and a status bar at the bottom.

← → ↻ ⬆ `www.datanimal.com/send_email`

 **Send E-mail to Visitors** 

Event 1 

Title

//Inside 

Send



### 3.16. Create Conservation Organization



The screenshot shows a web browser window with the address bar displaying `www.datanimal.com/create_c_organization`. The page title is "Create Conservation Organization" in large blue text. Below the title, there are two input fields: "Date:" and "Location:". Below these fields is a rounded button labeled "CREATE". The browser window includes standard navigation icons (back, forward, refresh, home) and a scrollbar on the right side.

```
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

← → ↻ ⬆ www.datanimal.com/schedule\_training

⏏

Scheduled Trainings		
20-03-1992	Bird1	Flying
13-05-1999	Dog1	Running

## Schedule New Training

Date:

Animal:

Type:

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

### 3.19. Request New Treatment

Requested Treatments			
20-03-1992	Guvercin1	Vet_Ali	Completed
13-05-1999	Kangal Dog1	Vet_Veli	Completed

## Request New Treatment

Date:

Animal:

Vet:

**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





Cage Food Stocks

Cage	Food Type	Amount	Expiration Date
Cage 1	Apple	1000 kg	02-19-2020
Cage 13	Apple	100 kg	02-19-2020
Cage 13	Banana	12 kg	01-24-2020

Cage 2

Food Type:

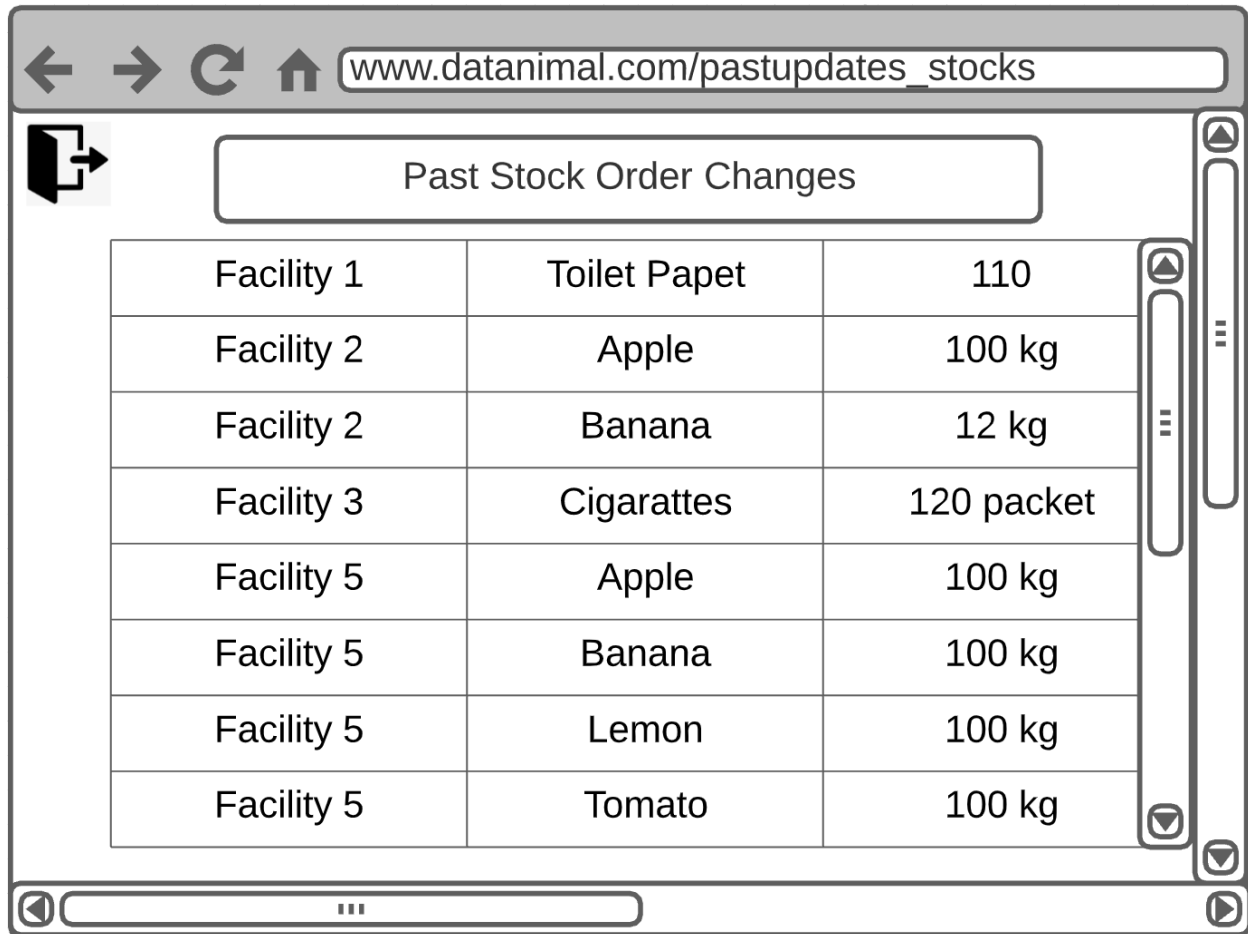
Amount:

Order

3.21. Facility Worker Main Page



### 3.22. Past Stock Order Changes



www.datanimal.com/pastupdates\_stocks

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

www.datanimal.com/update\_stocks

Current Stocks

Facility 5	Banana	100 kg
Facility 5	Lemon	100 kg
Facility 5	Tomato	100 kg

Facility 14

Item:

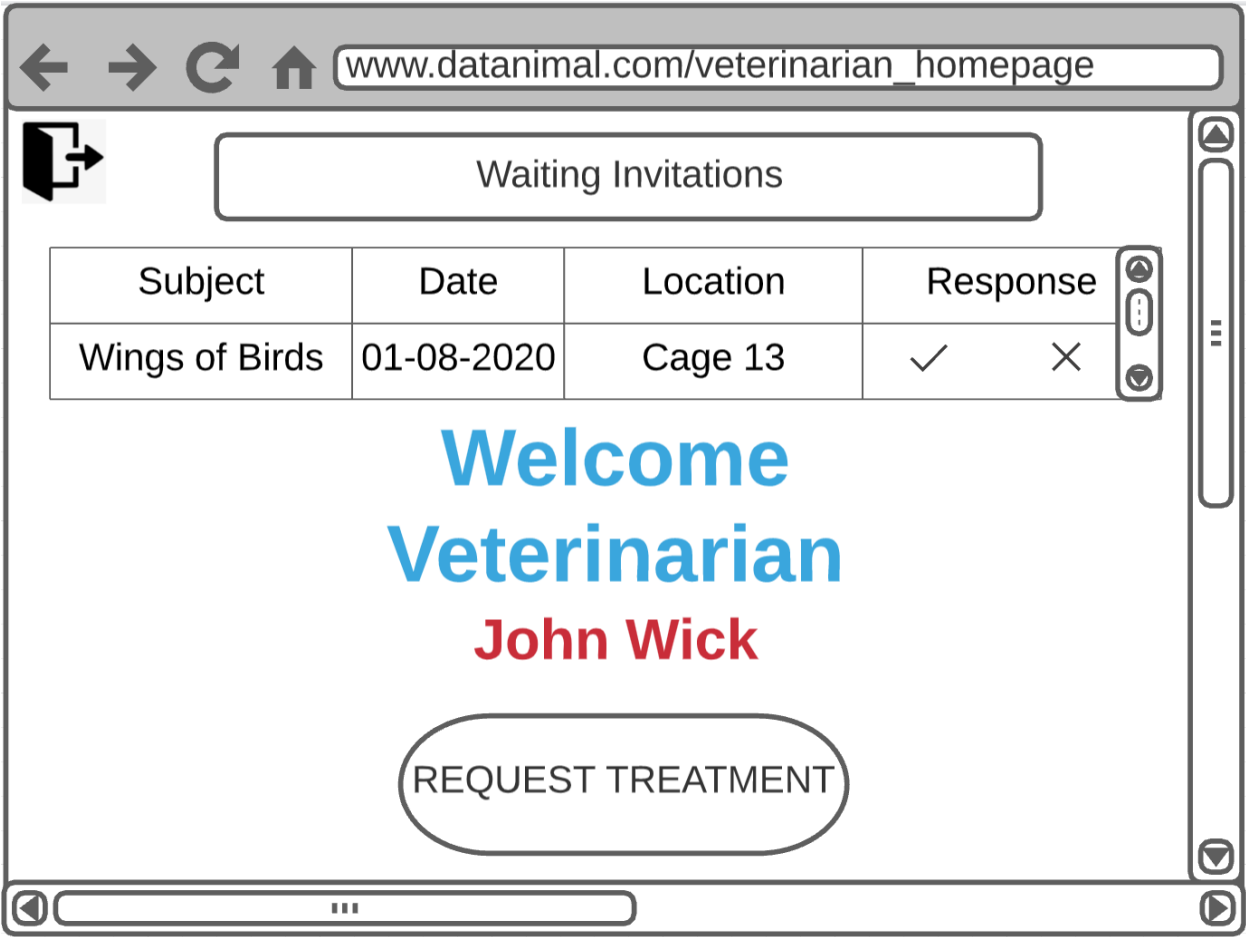
Amount:

Expiration Date:

Update Stocks

Order

3.24. Veterinarian Main Page



### 3.25. Animals in Cage

ID	NAME	Birthday	Species	Gender	
1048	Harambe	10.10.2010	Gorilla	Male	List Scheduled Trainings
1049	Dolly	12.12.2012	Sheep	Female	List Scheduled Trainings

Date	Type	
10.04.2021 14:00	Climbing Training	Update Date Remove
10.06.2021 16:00	Sleep Training	Update Date Remove

```
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;
```

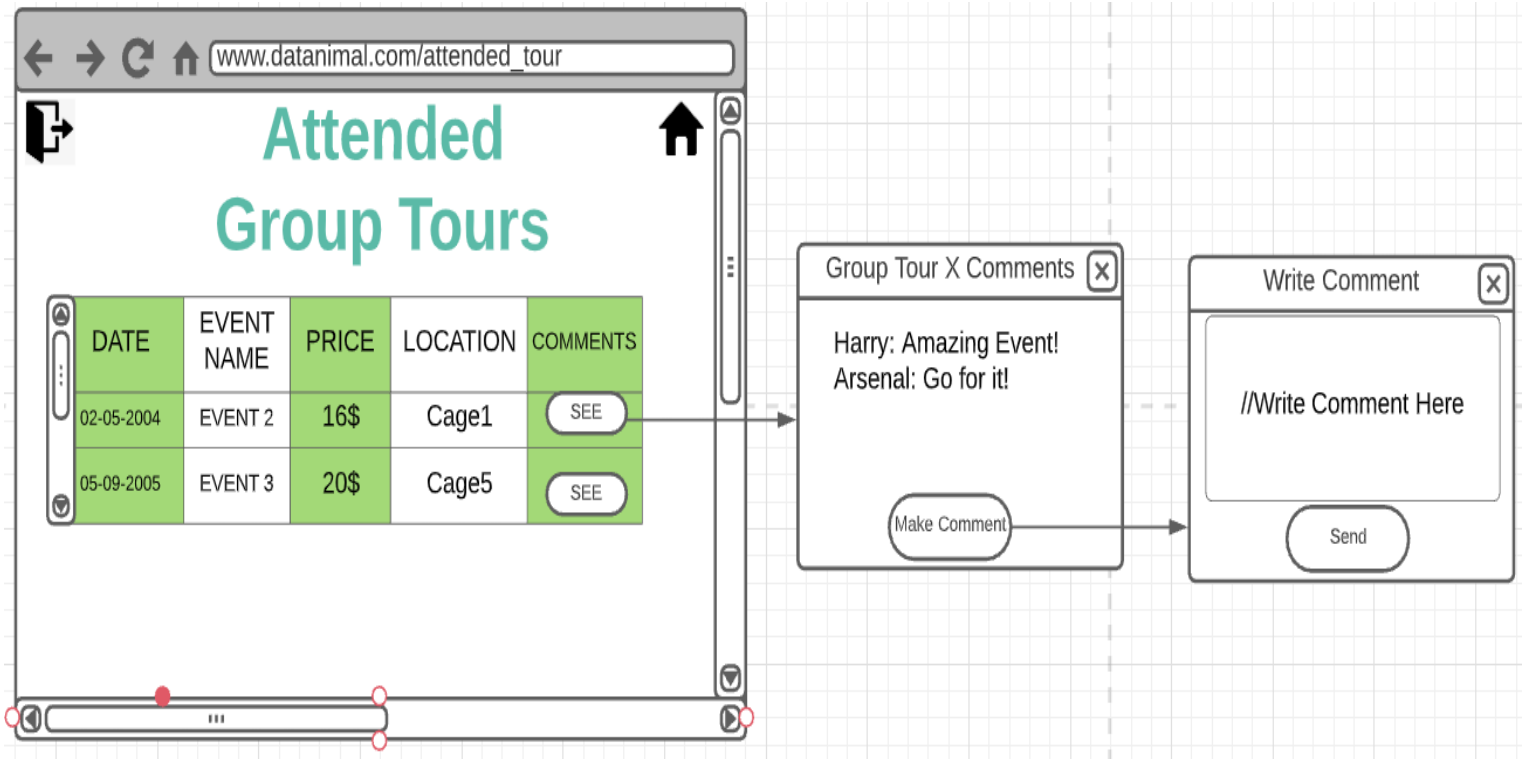
**Reschedule:**

```
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



#### **List attended tours:**

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

#### **Make comment:**

```
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);
```