CS2102: Database Systems

2015/2016 Semester 2

Project Report - Crowd Funding

Group 20:

Bjorn Lim (A0116538A)

Brendan Yong (A0108541M)

Kester Tan (A0110594L)

Jeremy Ong (A0108310Y)

Nicholas Low (A0110574N)

Content Page

1. Introduction

CrowdFund! is a crowdfunding website that provides entrepreneurs with a platform where they are able to pitch and seek the funding required to realize their business ideas. Users of the website will be able to browse and support the ideas which appeal to them through the funding option.

1.1 Design Principles

Practicality and **Minimalistic** are the two main design principles that we followed in the development of this web application.

In terms of **Practicality**, the web application is able to fulfill the following basic requirements of both the users and the entrepreneurs:

Users

1. Ability to search for projects/ideas that appeal to them
2. Ability to provide funding for interesting projects/ideas
3. Ability to view the projects that he/she has provided funding for

Entrepreneurs

1. Ability to post up and advertise their detailed business ideas
2. Ability to view the amount of funding acquired for their posted ideas
3. Ability to modify the project that they have posted

As for the Administration team, **CrowdFund!** also provides ease of access to administrative tools (such as Modification, Removal and Creation of any entries in the core database)

In terms of the **Minimalistic** approach to design of the web application, our development team has opted for a very simple web page layout where users,

entrepreneurs and administrators alike will be able to navigate between the various functionalities intuitively.

1.2 Developmental Assets & Programming Languages

Two major components identified in the development of this web application are the **Database** and the **Web Page**.

The assets and programming languages used in each of the identified components are as follows:

Database

PostgreSQL (Open source relational database management system)
phpPgAdmin (Web Application for managing of PostgreSQL databases)

Web Page

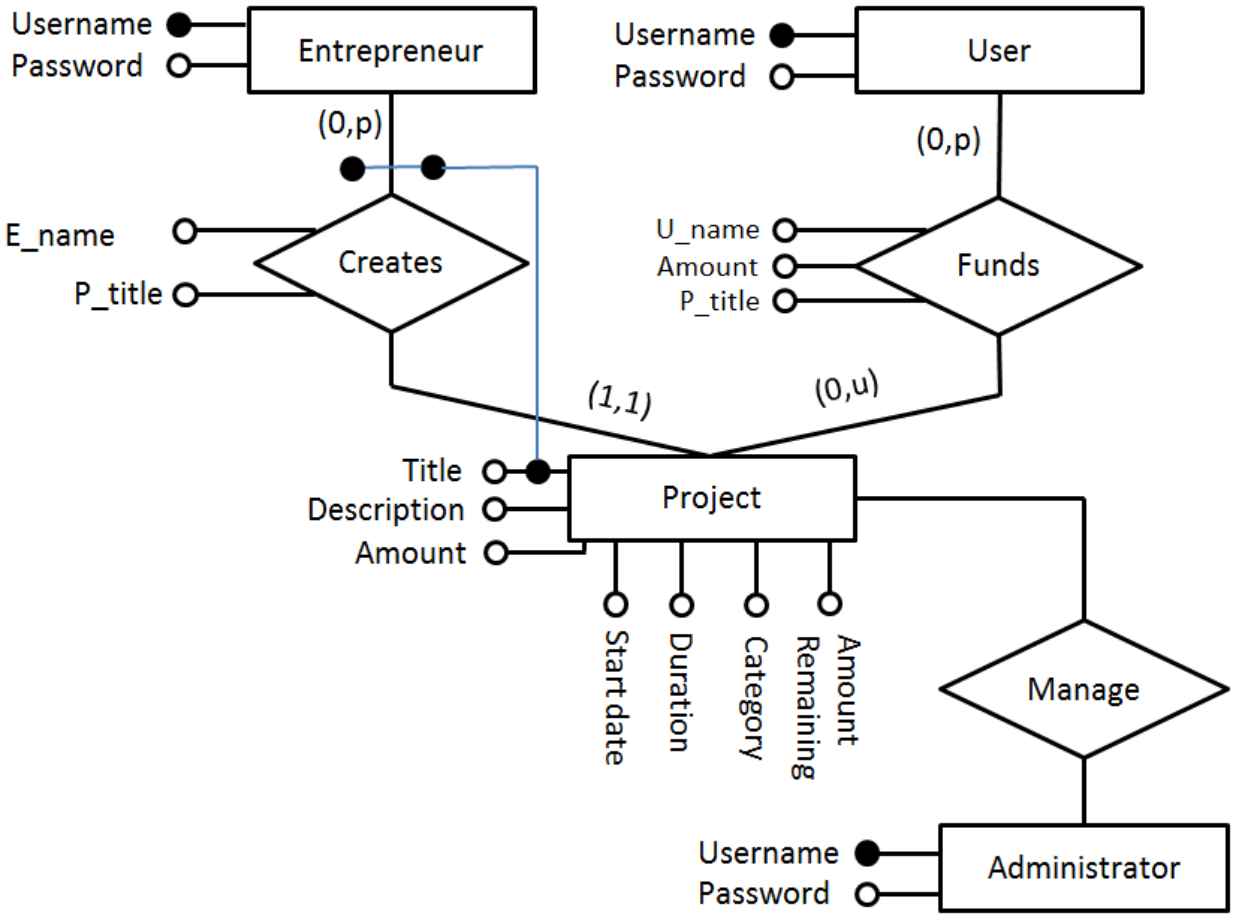
Hosting of web page:
Apache Web Server - Version 2.4 (Hosting of our web application page)

Design of web page:

1. HTML
2. CSS
3. PHP
4. Javascript

2. Database Design

2.1 Entity Relationship Diagram



2.2 Entities

The entities present within the Entity-Relationship Diagram for **CrowdFund!** are as follows:

Entrepreneur	
Attribute	**Domain**
username (primary key)	varchar(128)
password	varchar(64)

User	
Attribute	**Domain**
username (primary key)	varchar(128)
password	varchar(64)

Administrator	
Attribute	**Domain**
username (primary key)	varchar(128)
password	varchar(64)

Project	
Attribute	**Domain**
title	varchar(128)
description	varchar(2048)
amount	int
start_date	date
duration	int
category	varchar(128)
remaining_amount	int
e_name	varchar(128)
title, e_name **(primary key)**	-

2.3 Relationships

The relationships present within the Entity-Relationship Diagram for **CrowdFund!** are as follows:

Relationship between Entrepreneur and Project entities:

create_project	
Attribute	**Domain**
e_name	varchar(128)
p_title	varchar(128)

Relationship between User and Project entities:

funding	
Attribute	**Domain**
u_name	varchar(128)
amount	int
p_title	varchar(128)

Note:

e_name - corresponds to - username attribute (entrepreneur entity)

u_name - corresponds to - username attribute (user entity)

p_title - corresponds to - title attribute (project entity)

2.4 Database Schema (DDL)

```
CREATE TABLE user (
      username VARCHAR(128) PRIMARY KEY,
      password VARCHAR(64) NOT NULL
);
CREATE TABLE entrepreneur(
      username VARCHAR(128) PRIMARY KEY,
      password VARCHAR(64) NOT NULL
);

CREATE TABLE administrator(
      username VARCHAR(128) PRIMARY KEY,
      password VARCHAR(64) NOT NULL
);

CREATE TABLE project(
      title VARCHAR(128) UNIQUE,
      description VARCHAR(256),
      amount INT CHECK (amount > 0) NOT NULL,
      start_date DATE NOT NULL,
      duration INT CHECK (duration > 0) NOT NULL,
      category VARCHAR(128),
      remaining_amount INT CHECK (amount > 0) NOT NULL,
      e_name VARCHAR(128),
      (title, e_name) PRIMARY KEY,
      FOREIGN KEY e_name REFERENCES entrepreneur(username) ON DELETE
      CASCADE
```

```
);

CREATE TABLE create_project(
      e_name VARCHAR(128),
      p_title VARCHAR(128),
      (e_name, p_title) PRIMARY KEY,
      FOREIGN KEY e_name REFERENCES entrepreneur(username) ON DELETE
      CASCADE,
      FOREIGN KEY p_title REFERENCES project(title) ON DELETE CASCADE
);

CREATE TABLE funding(
      u_name VARCHAR(128)
      amount INT,
      p_title VARCHAR(128)
      (u_name, p_title) PRIMARY KEY,
      FOREIGN KEY u_name REFERENCES user(username) ON DELETE CASCADE,
      FOREIGN KEY p_title REFERENCES project(title) ON DELETE CASCADE
);
```

3. SQL Samples & Functions

3.1 Login

Visitors to the **CrowdFund!** web page are first presented with a login page. Three fields are to be filled in here, namely:

- Username
- Password
- Account Type (Entrepreneur, User or Administrator)

For example, we have a login form as follows:

Username : John

Password: correcthorsebatterystaple

Account Type: Entrepreneur

After submitting the above form, the resulting DML statement in the back-end will be as follows:

```
SELECT EXISTS(
      SELECT 1
      FROM Entrepreneur e
      WHERE e.name = “John” AND e.password =
      “correcthorsebatterystaple”)
);
```

If the above SELECT statement returns a Boolean TRUE value, login is successful. Else, login is unsuccessful.

3.2 Entrepreneur Functionalities

Adding Projects:

Entrepreneurs are able to post and advertise their projects on **CrowdFund!** through by filling up a simple form with all the details of the project and submitting said form.

Suppose we have a form that contains the following details filled out by an Entrepreneur account with the username **John**:

Title: Sample Project

Description: Just a test project

Amount: 322

Start Date: 7/11/2016

Duration: 3

Category: Movie

Remaining Amount: 322

The resulting DML statements in the back-end would be:

```
INSERT INTO project(title, description, amount, startdate, duration,
category, remaining_amount, entrepreneur)
VALUES (‘Sample Project’, ’Just a test Project’, 322, ‘7/11/2016’, 3,
‘Movie’, 322, ‘John’);

INSERT INTO create_project(entrepreneur_name, project_title) VALUES
('John', 'Sample Project')
```

Note: For adding of projects, two tables are affected within the database, namely **project** and **create_project**

3.3 User Functionalities

Searching:

Users of **CrowdFund!** are able to search through all available projects posted on the page through filling up a form that contains several specific fields which will then act as filters during the searching process.

Fields include:

- Project Title
- Amount
- Category
- Entrepreneur Name
- Remaining Amount

Users can choose to fill in any combination of fields above and the input for said fields will be fed into the DML statement in the back-end.

Suppose a user with the username **Jane** filled out the search form with the following field values:

Project Title: (empty)
Remaining Amount: 555
Category: Movies
Entrepreneur Name: John

The resulting DML statement in the back-end would be:

```
SELECT p.title
FROM project p
WHERE p.e_name = 'John' AND p.category = 'Movies' AND
p.remaining_amount <= 555
```

Funding:

Users are also able to providing funding for projects through a very simple process where they can simply click on the project which they are interested in funding and filling in the amount of funding that they would like to provide.

Suppose a user with username **Jane** has selected a project with the title "**Sample Project**" and filled in **322** as the amount to fund.

The resulting DML statement in the back-end will be as follows:

```
UPDATE project SET remaining_amount=remaining_amount - 322 WHERE
title='Sample Project'

INSERT INTO funding(u_name, amount, p_title) VALUES ('Jane', 322,
'Sample Project')
```

Note: For funding of projects, two tables will be affected within the database, namely: the **project** and **funding** tables.

3.4 Administrator Functionalities

Accounts with administrator level privilege are given specific tools with which they are able to carry out administrative work on **CrowdFund!** in order to maintain the integrity and order within the app.

An example of such a tool would be the ability to remove inappropriate projects by selecting and deleting the selected project.

Suppose an administrator with the username **Jacob** has identified an inappropriate project with the title of "**Inappropriate Project**" and has clicked the delete button.

The resulting DML statement in the back-end will be:

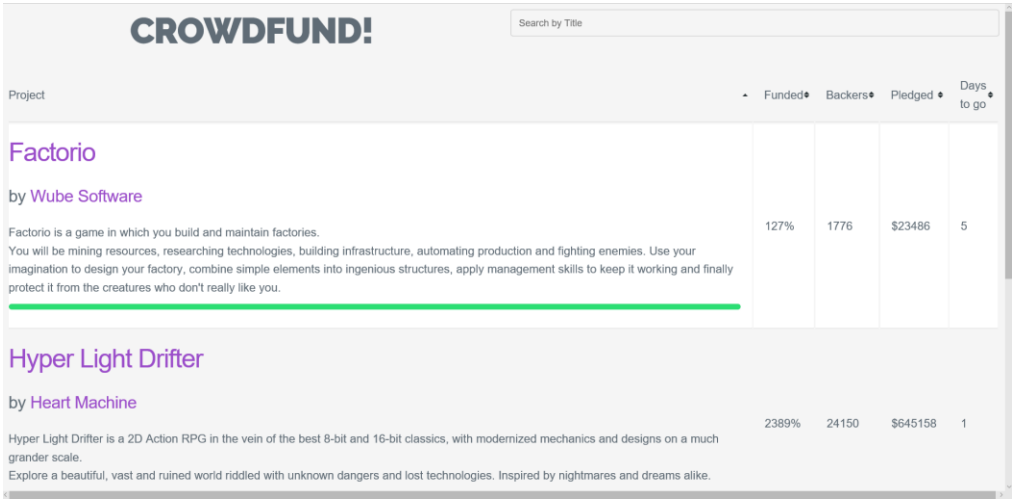
```
DELETE FROM project WHERE title = 'Inappropriate Project'
```

Note: The tables **create_project** and **funding** will also be affected in this scenario since there will be a cascading of deletion as a result of the removal of the project from the **project** table.

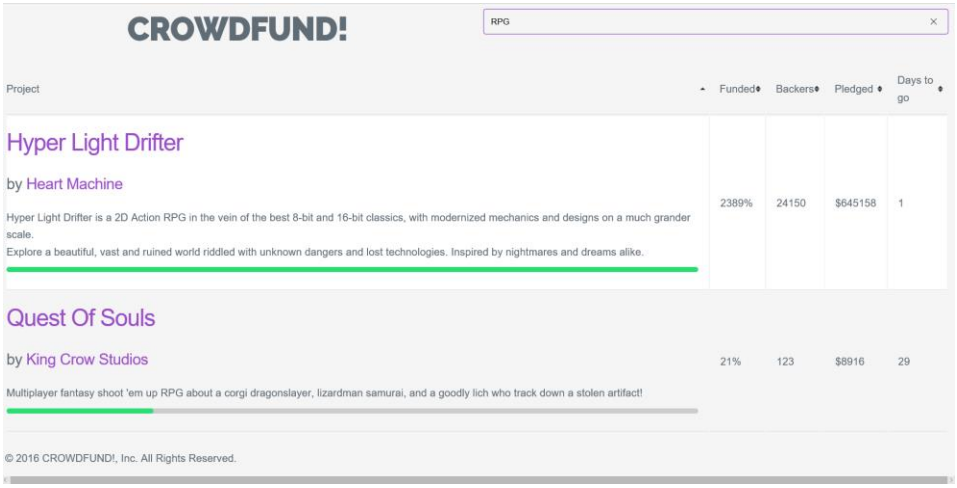
4. Web Page Design

As stated above, the design direction that CrowdFund! took with regards to the web page is one of *minimalistic* and *intuitivity* in order to provide Entrepreneurs, Users and Administrators with the best possible experience.

CrowdFund! screenshots (more to be revealed on the demo session)



<Main page that **users** will be seeing>



<Search function highlighted at the top right corner>