NUS

National University
of Singapore

CS2102

Project 1

Team 109

Team Members:

Hoang Huu Chinh
Gupta Ananya Vikas
Tan Jing Xue Andre
Phua Anson

List of Constraints:

1) Projects can be almost anything
2) Users are either creator or backer or both
3) User may have up to 2 Credit Cards and each must have at least one
4) 2 projects can have same name and created by same creator
5) Only one creator per project
6) No projects that are not created by a creator
7) Many backers for each project
8) Backer may not fund any project or can back multiple projects at the same time
9) Each backer can fund the same project only once
10) Unique reward level within the same project
11) User can back the reward level with higher amount but not less than stated value
12) User can fund only through reward level and only fund one level in a particular project
13) Creator can announce one update at a single time (time + date)
14) Backers request for refund within 90 days of project deadline
15) Employee approves/rejects a refund request
16) If approved, record date of acceptance and if rejected, record date of rejection
17) User is not charged if the funding did not meet goal (no refund request can be made)
18) When refund is pending, there may not be an associated employee
19) Each user can request for refund only once
20) If rejected request, no other request can be made for that project
21) Employees verify users but not all need to be verified
22) User verified by at most one employee

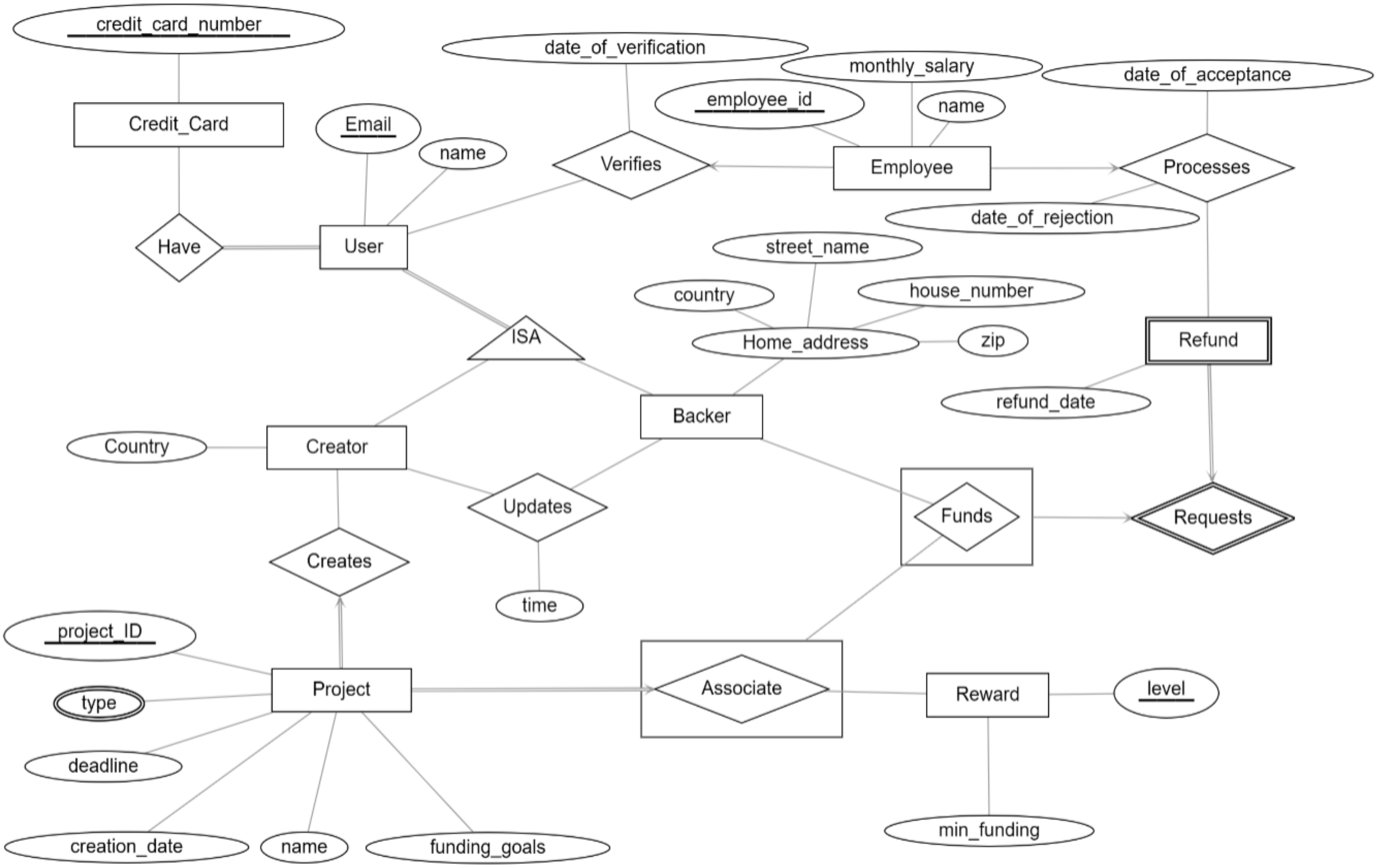
Assumptions and Justifications for design decisions

1. Projects can be almost anything, can have more than 1 category or have no categories, so we put “type” as a multivalued attribute under the [Project] entity.
2. Because each project is said to be associated with a reward level, each reward level is said to record minimum amount of money for funding and a user can only fund a project using the reward level, so we make the whole relation [Project]-<Associate>-[Reward] to be an aggregation, and then this [Project-Associate-Reward] have relation <Funds> to [Backer]. In the end, it is [[Project]-<Associate>-[Reward]]-<Funds>-[Backer]
3. For each funding that a backer made, they can request for a refund. Because a refund request requires data like backer’s id, the funding project’s id, we decided to make <Funds> to be an aggregation [<Funds>] so that we can make use of data from this aggregation later for the design of refund requests. Since a refund request can only exist

when the backer made a funding, we decided to make Refund to be a weak entity set of [Backer]

4. An user is said to be either creator or backer and must specify whether they want to be a creator or backer or both, so we decided to use ISA hierarchy with both covering and overlap constraints.
5. An user is said to have at least one and up to two credit cards. Taking into consideration that in reality, maybe a credit card can be used by more than one user (family members use/borrow another one card), we decided to make the relation <Has> between [User] and [CreditCard] to be a many-to-many relation with full participation constraint from [User]. The upper bound constraint "up to two credit cards" will be covered in relational by having 2 attributes "cc_number1" and "cc_number2" schema.

ER Model



Constraints not captured by the ER

1. For Constraint 3: A user may have up to 2 Credit Cards and each must have at least one. However, the ER diagram does not indicate the maximum limit of 2 and instead uses the total participation constraint of at least one shown between user and credit_card entity
2. For Constraint 9: Each backer can fund the same project only once which cannot be implemented above as it would add further restrictions to the relation between backer and project, disapproving the constraint 8
3. For Constraint 11: User can back the reward level with higher amount but not less than the stated value but this cannot be shown in the diagram comparing the amounts
4. For Constraint 14: Backers request for refund within 90 days of project deadline but since comparison cannot be indicated above, it cannot be implemented in ER Model
5. For Constraint 18: When refund is pending, there may not be an associated employee but since the status of pending cannot be reflected in ER, this statement cannot be justified completely

Relational Schema

```
DROP TABLE IF EXISTS Employee,CreditCard, User,Creator,BackerReward,
Project,Updates,Verifies,Processes,Associate,Funds,Refund CASCADE;

/* Relational schema*/

CREATE TABLE Employee (
  employee_id SERIAL PRIMARY KEY,
  name TEXT,
  monthly_salary NUMERIC
);

CREATE TABLE CreditCard (
  credit_card_number INTEGER,
  PRIMARY KEY(credit_card_number),
);

CREATE TABLE User (
  email VARCHAR(200) PRIMARY KEY,
  name TEXT,
  cc_number1 VARCHAR(20) NOT NULL,
  cc_number2 VARCHAR(20),
  FOREIGN KEY cc_number1 REFERENCES CreditCard(credit_card_number),
  FOREIGN KEY cc_number2 REFERENCES CreditCard(credit_card_number)
);

CREATE TABLE Creator (
  country TEXT,
  email VARCHAR(200),
  PRIMARY KEY (email),
  FOREIGN KEY email REFERENCES User(email) ON DELETE CASCADE
);

CREATE TABLE Backer (
  street_name TEXT,
  house_number VARCHAR(20),
  country TEXT,
  zip INTEGER,
  email VARCHAR(200),
  PRIMARY KEY (email),
  FOREIGN KEY email REFERENCES User(email) ON DELETE CASCADE
);

CREATE TABLE Reward (
  level VARCHAR(20) PRIMARY KEY,
  min_funding NUMERIC
);

CREATE TABLE Project (
  project_id SERIAL PRIMARY KEY,
  name TEXT,
  deadline VARCHAR(20),
  funding_goals NUMERIC,
  creation_date DATE,
  type TEXT,
  c_email VARCHAR(200),
  rlevel TEXT,
  PRIMARY KEY(project_id,level),
  FOREIGN KEY c_email NOT NULL REFERENCES Creator(email),
  FOREIGN KEY rlevel REFERENCES Reward(level)
);
```

```
CREATE TABLE Updates (
  time TIMESTAMP,
  u1_email VARCHAR(200),
  u2_email VARCHAR(200),
  PRIMARY KEY(u1_email,u2_email,time),
  FOREIGN KEY u1_email REFERENCES Creator(email),
  FOREIGN KEY u2_email REFERENCES Backer(email)
);

CREATE TABLE Verifies(
  date_of_verification DATE,
  e_id SERIAL,
  u_email VARCHAR(200),
  PRIMARY KEY (e_id),
  FOREIGN KEY e_id REFERENCES Employee(employee_id),
  FOREIGN KEY u_email REFERENCES User(u_email)
);

CREATE TABLE Processes (
  date_of_rejection DATE,
  date_of_acceptance DATE,
  e_id SERIAL,
  PRIMARY KEY (e_id),
  FOREIGN KEY e_id REFERENCES Employee(employee_id)
);

CREATE TABLE Associate (
  p_id SERIAL,
  rlevel TEXT,
  PRIMARY KEY (p_id,rlevel),
  FOREIGN KEY p_id REFERENCES Project(project_id),
  FOREIGN KEY rlevel REFERENCES Reward(level),
);

CREATE TABLE Funds (
  p_id SERIAL,
  rlevel TEXT,
  u_email VARCHAR(200),
  PRIMARY KEY (p_id,rlevel,u_email),
  FOREIGN KEY (p_id,rlevel) REFERENCES Associate(p_id,rlevel),
  FOREIGN KEY u_email REFERENCES Backer(email)
);

CREATE TABLE Refund (
  refund_date DATE,
  p_id SERIAL,
  rlevel TEXT,
  u_email VARCHAR(200),
  PRIMARY KEY(p_id,rlevel,u_email),
 FOREIGN KEY (p_id,rlevel,u_email) REFERENCES Funds(p_id,rlevel,u_email) ON DELETE
CASCADE
);
```

Constraints not captured by the Relational Schema

1. For Constraint 1: The composite attribute ‘type’ under Project entity couldn’t be represented in the schema to portray that project can be almost anything.
2. For Constraint 11: User can back the reward level with higher amount but not less than stated value couldn’t be implemented completely as minimum funding cannot be referenced across other tables above
3. For Constraint 17: If funding goal is not met, there can be no refund request is partially shown using the ON DELETE CASCADE condition, with regard to the weak entity set but is yet not directly shown for the funding goals specifically.
4. For Constraint 20: Once rejected request, no request allowed again is also not implemented the schema above since the link between rejection and request cannot be directly shown above
5. For Constraint 2: The relational schema shows that user can be backer or creator but does not show the case when user is both