BILKENT UNIVERSITY

CS353 DATABASE SYSTEMS

UptownFANK Project Tracking Software

Project Proposal

Authors:	*Student IDs:*
Fatbardh Feta	21600334
Alemdar Salmoor	21500430
Naisila Puka	21600336
Kunduz Efronova	21600469

March 4, 2019

Contents

1 Introduction **2**

2 Description **2**

2.1 Why use a database 3

3 Requirements **3**

3.1 Functional Requirements 3

3.2 Non-Functional Requirements 5

3.2.1 Usability 5

3.2.2 Security 5

3.2.3 Reliability 5

3.2.4 Performance 6

3.2.5 Capacity 6

4 Limitations **6**

5 Entity Relationship Diagram **7**

6 Online Access **9**

7 Conclusion **10**

1 Introduction

This is a proposal for a project tracking software called UptownFank. The software will be developed as a term project of CS353 Database Systems course at Bilkent University. UptownFank aims to be a simple useful tool for tracking different projects, usually shared between some team members. It will pave the way for maintaining project communication and keeping track of the accomplished and to-do tasks. The proposal starts with a high-level description of the software system, followed by functional and non-functional requirements. Then the proposed design will be described with a detailed Entity-Relationship diagram. Updates of the project can be followed through online information found in Section 6.

2 Description

UpTownFank will provide a complete and effective project management system for large teams. Each project has boards where lists can be added and seen by all the team members. Lists hold a group of cards that are related to each other by their content. Users can navigate to each of the lists of the project boards and mark as "Completed" any of the cards in that list. Cards can also have labels, attachments, deadlines or comments.

Our system provides support for two type of users, SuperUser and BasicUser. Both of these users will firstly register to the system by providing basic information like their name, age, address, phone number, e-mail address, username and password. When a costumer wants to register as a SuperUser they just have to sing in using the Create SuperUser button and provide information regarding their professional background like the company name/university etc. Then they can create teams, projects and boards, lists and cards as well as delete any of the entities mentioned above. BasicUsers need to have an invitation code, or team key, in order to be part of a team. When the BasicUser gets the invitation code from the SuperUser then they can be added to the team created by the SuperUser, navigate through the project board of their team, check out the lists and cards posted there, mark

the cards that they have completed or even add comments to Cards. BasicUsers do not have the privilege to edit any of the project boards or lists that they can access. When a BasicUser is removed from a team they can no longer access the team's boards.

2.1 Why use a database

As is understood from the above description, a lot of data with many relationships between data entities need to be maintained for this project tracking software. That is a huge amount of data to be organized. Also, we intend to make our project scale as much as possible: many teams are expected to track their projects through our system. Therefore we need a strong database system to provide strong tools to access, manipulate and save data in a proper way. In addition, establishing and maintaining different relations is much easier by using a Relational Database Management System [1]. Considering these advantages, we decided to use a Database System for our project.

3 Requirements

For this project software, we we will follow some functional and non-functional requirements as described below.

3.1 Functional Requirements

Following are the functional requirements of our project tracking software. You may find what a user will be able to do in UptownFANK in order to keep track of their project:

- Users should be able to register to the system with a unique valid email and password combinations.

- Users should be able to register to different teams with a unique team key provided by team supervisor (a superuser).
- Super users should be able to own multiple projects in the system.
- Super users should be able to create/remove/modify boards within the project in the system.
- Super users should be able to supervise multiple teams within the system.
- Super users should be able to add/remove other users to the existing teams within the system.
- Super users should be able to create/remove/modify lists for every board they create.
- Super users should be able to create/remove/modify cards to every list they create.
- Super users should be able to add/remove/modify due dates to every card they create.
- Super users should be able to add/remove members in the cards they create.
- All users should be able to add/remove/modify attachments to every card they create or get assigned to.
- All users should be able to add/remove/modify comments to every card they create or get assigned to.
- All users should be able to add/remove/modify labels to every card they create or get assigned to.
- All users should be able to complete checklists in their assigned cards.

3.2 Non-Functional Requirements

To provide a good user experience, we have planned the following non-functional requirements for our software:

3.2.1 Usability

One of the important design goals of Uptown FANK project is to provide user-friendly interface. This requirement consists in:

- First time users should be able to easily navigate themselves through our project without any help. This will be achieved by using minimum amount of buttons that will help to avoid complications and confusion among numerous buttons.
- Additionally, we will refer to the same layout and theme throughout the project in order to make our interface consistent.
- Finally, we will make sure user can access main page from any page.

3.2.2 Security

Protecting users' data is very important, especially, when it comes to things like saving projects, completed tasks. Due to the reason that we don't have any feature related with payments, the only thing related with security that concern us is user-names and passwords. Uptown FANK promises to make authentication procedure system as reliable as possible.

3.2.3 Reliability

The system will have all kinds of error handling so that it does not crash due to incorrect inputs. Among many others checks we will check incorrect inputs:

- Trying to access board user is not invited to or delete board user is not supervising.

- Trying to set username that is already taken.
- Being standard user and trying to access features of privileged user.

3.2.4 Performance

For Uptown FANK, along with user-friendly interface, short response time is essential to provide smooth experience to the user. Response time will not exceed 2 seconds time limit and concurrent high number of users will not be obstacle for response time.

3.2.5 Capacity

Being a project tracking software, Uptown FANK must be ready to handle large amount of projects and their subtasks. Database must provide enough storage to keep and organize large amounts of data.

4 Limitations

Following are some limitations of our project:

- Only "privileged" (super) users can create and delete boards.
- Only "privileged" (super) users can create teams and obtain team key to invite basic users. Only the supervisor can assign teams to boards.
- Only "privileged" users can assign users to cards and remove standard users from cards.
- Standard users can neither create lists nor invite others to team.

5 Entity Relationship Diagram

Here we present our entity-relationship diagram created in Draw.io [2], followed by main entities' and relationships' explanations.

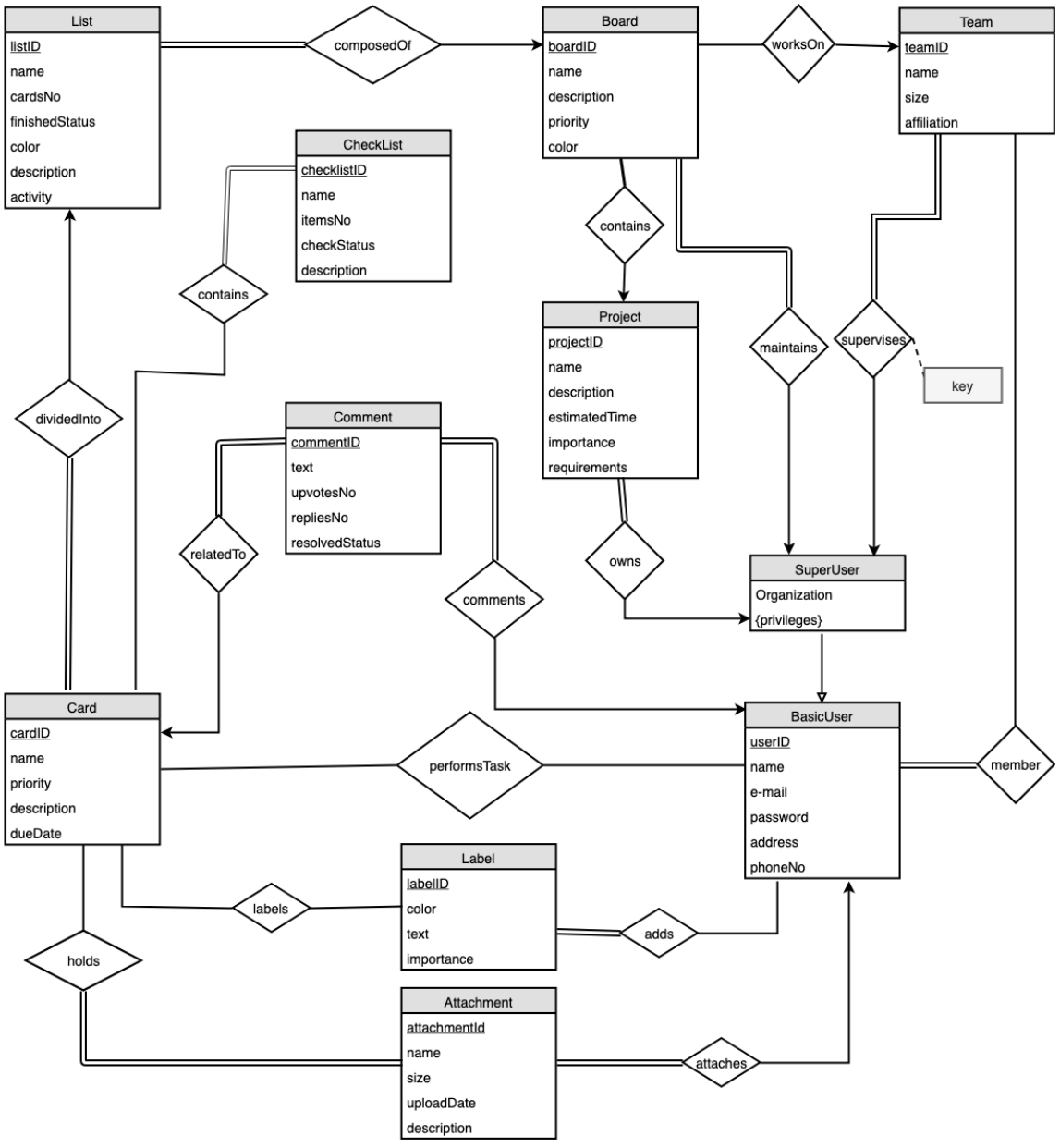

Figure 1: Entity-Relationship Diagram

Regarding the entities shown in Figure 1, we have put whatever stands out in a project trasking software. Information about users (standard and privileged ones), teams, projects, boards, lists, cards, along with comments, labels, checklists and attachments, will be saved in the database. Following are explanations of our main chosen relationships:

owns: This is a relationship between SuperUser and Project. It is a one-to-many relationship, with full participation of Project. This means each project has exactly one owner. Only superusers can own projects.

supervises: This is a relationship between SuperUser and Team. It is a one-to-many relationship, with full participation of Team. This means each team has exactly one supervisor. Only superusers can supervise teams. They associate a key with each team, so that standard users can be part of the team based on that key.

member: This is a relationship between BasicUser and Team. It is a many-to-many relationship, with full participation of BasicUser. This means a user can belong to one or more teams, and a team can have many members.

contains: This is a relationship between Project and Board. It is a one-to-many relationship. This means that a project can have 0 or more boards, and a board belongs to at most one project.

maintains: This is a relationship between SuperUser and Board. It is a one-to-many relationship, with full participation of Board. This means that a board is maintained by exactly one superuser, but a superuser can maintain multiple boards. By maintaining, we mean that only superuser can create/delete/modify boards.

worksOn: This is a relationship between Team and Board. It is a one-to-many relationship. This means that a team can be assigned to work on multiple boards, and at most one team works on a specific board.

composedOf and **dividedInto**: These are one-to-many relationships between Board and List, and List and Card respectively. It comprises full participation of the "many" side. In this way the project can be tackled with the method "Divide and Conquer". It will be easier for the superuser to divide the work in this way.

performsTask: This is a relationship between BasicUser and Card. It is a many to many relationship, which means that different users can be assigned to perform a task together, and also different cards can be assigned to a single user.

relatedTo and **comments**: These are one-to-many relationships between Card and Comment, and BasicUser and Comment respectively. It comprises full participation of the "many" side belonging to the comment. This means that each comment belongs to exactly one card, and is created by exactly one user. Cards can have 0 or more comments, and users can make 0 or more comments.

adds and **attaches**: These are one-to-many relationships between BasicUser and Label, and BasicUser and Attachment respectively. It requires full participation of the "many" side belonging to the Label and Attachment. This means that each label is added by one user and each attachment is attached by one user. Users can add multiple labels and attach multiple attachments to the cards.

labels: This is a many-to-many relationship between Card and Label. This means that a card can have multiple labels and a particular label may belong to multiple cards (For example label *Safety*).

contains and **holds**: These are one-to-many relationships between Card and Checklist, and Card and Attachment respectively. It requires full participation of the "many" side belonging to the Checklist and Attachment. This means that each card can have multiple checklists and may hold attachments, but a specific checklist belongs to exactly one card and a specific attachment is uploaded to exactly one card.

6 Online Access

Our project can be accessed online through our:

Webpage: `https://uptownfank.github.io`

GitHub: `https://github.com/NaisilaPuka/cs353project` (currently private)

7 Conclusion

With this project, we aim to get the best out a database management system in order to create a useful tool for tracking various projects divided to different teams. Making use of different entities and relationships, the database system will provide full referential integrity for us, which will make UptownFANK a reliable and safe system to use for maintaining project information.

References

[1] What is RDBMS? `https://searchdatamanagement.techtarget.com/definition/RDBMS-relational-database-management-system` Accessed March 4 2019

[2] Draw.io: free online diagram software for making flowcharts, process diagrams, org charts, UML, ER and network diagrams `https://www.draw.io/` Accessed March 4 2019