REPORT 1

Pindery

Anna Tranquillini
Edoardo Debenedetti
Edoardo Ghignone

Software Engineering Course, 2018 Spring Semester

Table of Contents

PART 1

Customer Statement Of Requirements

As social interactions are becoming more and more connected to internet-based communications, we search for a platform where we can connect to other people nearby to organise social events.

We would like an app which simplifies at best the steps needed to set up a party and which shares the duties for the event equally between all the party guests.
This application would both give the ability to people to organise their own parties and to join already existing ones.

PROBLEM: Finding cheap and fun events nearby

As students or more in general youngsters it is difficult to search for unofficial, cheap and fun parties. We are constantly overwhelmed with offers from big events organisers, but what we are actually looking for are smaller parties organised by people like us, who know what we enjoy and what is needed to organise a nice event.
In order to solve this problem, we came up with a few ideas.

We would like to have a mobile application where we would be able to directly scroll from a list of parties sorted by our closeness to their geographical position: of course, the closest would be the first to show up and the ones way too far wouldn't even be given as an option.
As browsing parties, we would like to be given more information about them such as the party theme, the location and a comprehensive description of the event.
We would also like to see whether the party organiser is reliable or not.

As we find a party that we like, we want to be able to share the information with our friends on social platforms such as WhatsApp, WeChat or Facebook.

PROBLEM: Party duties and expenses sharing

Another big problem when organising a party is given by expenses.
Of course, as party organisers we don't want to pay for everything, but at the same time it becomes difficult to collect money from everyone invited and, more often than we want to admit, we end up paying in advance for guests that don't even show up.
Anyway, both in a party between friends or a party with strangers, the 'money related' business is always the root of some problems.

We would like to find a method which does not involve any money exchange and does not give the party organiser too much duties and responsibilities.
A main function we would like to be implemented is a system that gives the possibility to share the expenses in a fair way.

This is how it should work: everyone who wants to take part to the party will have to bring something with a specific predefined value. Of course, it should also be crucial to check that the items are the ones needed for the specific event and that not everyone is bringing the same stuff. For these reasons, the application should require a well-developed control system that checks what the party guests are going to bring and makes sure that the duties are fairly shared.

In this way there would be no direct exchange of money and the wrongful behavior of one of the guests would neither influence in a major way the party organisation nor damage anyone on a monetary point of view.

PROBLEM: Promote locals

A big problem for us, venue owners, is advertising our own events and locals in a profitable way. Of course, we are given the possibility to post on social media or do some traditional 'flyer advertising', but we would like to have a direct contact with youngsters that are specifically interested in parties and events.
We came up with the idea of adding some advertisements inside the app, that should show up on the Parties list view.

These insertions should promote specific events from our locals so that the users would get to know us and if interested could join our parties.
These party views should be different from the 'traditional' ones: they should contain general information about the event and a direct link to a more specific description in an external page.

PROBLEM: Product placement

As Brands we want to be able to sponsor our products inside the application so that users could be encouraged to bring them at the parties.
In our opinion the best way to make this work would be to directly advise the party guests to choose our selected items when deciding what to bring.

It shouldn't be compulsory for the users to choose our brand, but we would like the application to strongly advise our products.
These sponsorships wouldn't only be useful to directly sell our brand products, but also for people to get to know our brand-name.

Glossary Of Terms

Non-technical terms

1. **Party Master:** the user that organises the party.
2. **Party Goer:** the user that takes part to the party.
3. **Party Stuff Brand:** the brand that wants to advertise its own product through the application.
4. **Venue Owner:** the owner of a bar, club or pub who wants to sponsor his events through the application.
5. **Party catalogue:** the list of items (food, beverages and any other kind of supply) that can be brought to the party.
6. **Pinder-point:** the point used to give a precise value to food, beverages and other items needed for the party, proportional to the actual monetary value. The total number of Pinder-points required to join a Party is decided by the algorithm explained below, while the value of every item is decided a priori and stored on the Database.
7. **Sponsored event:** the private event displayed in the app whose insertion is paid by a Venue owner.
8. **Sponsored product:** the brand product suggested as a party item whose insertion in the party catalogue is paid by a party stuff brand.
9. **Parties list:** the list of parties offered in the nearby.
10. **My Parties list:** the list of parties the user has been to and is going to join the future.
11. **Secret party:** a kind of party which is not showed directly on the parties list, but needs a direct link to be joined.
12. **Closed party:** a kind of party in which user participation has to be controlled and accepted from the party master before they can officially join the event.
13. **Public party:** a kind of party which doesn't need the party master confirmation to be joined and is directly accessible from the parties list page.
14. **Ratings:** the score displayed through a star representation (from 1 to 5) that is assigned from every party-goer to the party-master from the day after the party date.
15. **Thumbs-up/Thumbs-down:** the evaluation given from the party-master to every party-goer as soon as he arrives at the party. This evaluation is meant to positively or negatively rate what was brought from the user.
16. **Party details view:** the informative view about each party: contains the party description, the date, the location and the theme.

Technical terms

17. **Database:** the file where app data are stored.
18. **Graphical User Interface (GUI):** the interface that allows easier user communication via pictures and texts.
19. **Deep link:** a link that directly takes the user to a specific view of the application.
20. **Client:** the graphical interface that the users interacts with.

User Stories

As a Party Goer

Code	User story
PG1	I can see parties nearby so that I can choose one according to my interests.
	I can open a specific party page so that I can get more detailed information.
PG2	I can select to join a party so that I can inform the organiser that I will take part to it.
	I can select the items I want to bring to the party from a list reaching the amount of Pinder-points per person required by the party , so that food and beverages amount will be well balanced.
PG3	I can select to cancel my participation to a party so that the organiser will be informed.
PG4	I can directly share the party with my friends so that they can also have easy access to the event.
PG5	I can rate the party-master so that I can inform other users about the person reliability and ability to organise parties.
PG6	I can receive a notification whether the Party Master approves my participation to a Closed party or cancels a Party I wanted to take part to.

As a Party Master

Code	User story
PM1	I can create a new party so that I can organise a social event.
	I can add a theme, photos, information about the party so that I can give useful insights to the Party Goers.
	I can set the party's privacy so that I can define the visibility of the event.
	I can select a list of items I want to be brought to the party so that party-goers will precisely know what is needed.
PM2	I can accept/refuse party-goers so that I can control the users that will join the event.

PM3	I can edit the party information so that I can keep the party details page up to date.
	I can cancel the party so that party-goers will be informed that the event is not going to take place.
PM4	I can rate the party-goers with a thumbs-up/thumb-down so that I can give a public feedback on their reliability.
PM5	I can get Pinder-points in reward of party organisation so that I am stimulated to organise new events.
PM6	I can receive a notification if a Party Goer cancels its participation to a Party I am organising.
PM7	I can share the party with my friends and invite other people to the party I am organising.

As a Venue Owner

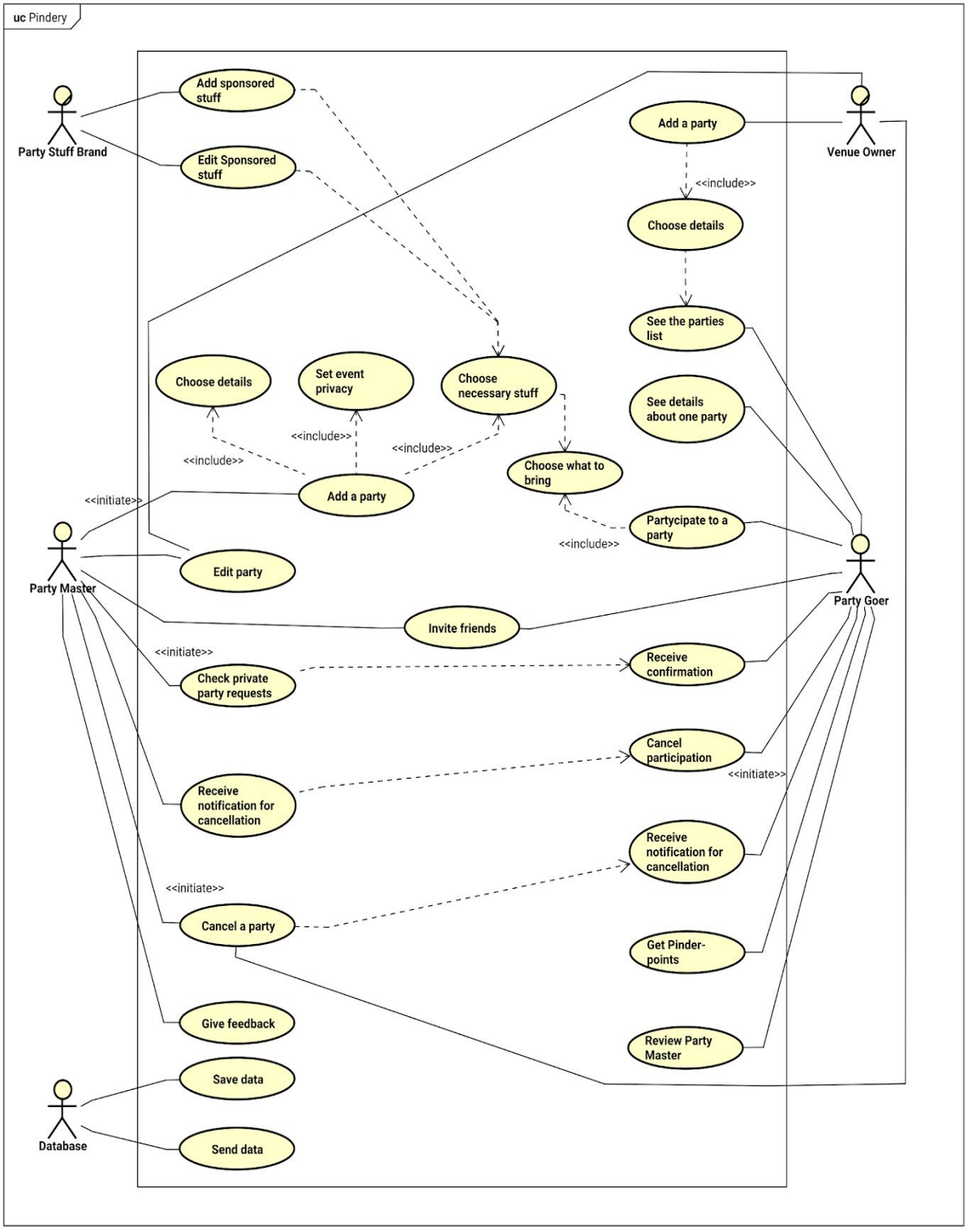
Code	User story
VO1	I can create a new party so that I can advertise an event in my local.
	I can add a theme, photos and information about the party so that I can give useful insights to the Party Goers.
VO2	I can edit the party information so that I can keep the party details page up to date.
	I can cancel the party so that party-goers will be informed that the event is not going to take place.

As a Party Stuff Brand

Code	User story
PB1	I can add items to the catalogue so that I can increase the sale of my own brand products.
	I can edit items of my own brand from the party catalogue so that I can keep the catalogue up to date.

PART 2

Use cases diagram



Use cases glossary

Code	Use Case name	Use case description	Participating actors and roles
UC1	See the parties list	See the list of the parties nearby on the Parties List of the application.	Party Goer
UC2	See details about one party	See the details about each party (i.e. place, date, time).	Party Goer
UC3	Participate to a party	Initiate the process which will take the Party Goer to participate to a party, clicking on the designed button on the party details page.	Party Goer
UC4	Choose what to bring	Choose from the Party catalogue what to bring at the party.	Party Goer
UC5	Cancel participation	Decide not to go to a party, clicking on the designed button in the My Parties List.	Party Goer
UC6	Receive notification for cancellation	Receive a notification whether a Party Goer decides not to go to a party or a Party Master cancels a party.	Party Goer Party Master
UC7	Review Party Master	Give feedback about a party and its organiser picking some characteristics from a predefined list and from 1 to 5 stars in the designed screen.	Party Goer
UC8	Receive Confirmation	Receive a confirmation notification if the Party Master approves his participation to a private party.	Party Goer
UC9	Invite Friends	Invite friends with an auto-generated deep link via a prompt with the main social applications.	Party Goer Party Master
UC10	Add a party	Initiate the process which will take to organise a new party, clicking on the designated button on the main screen.	Party Master Venue Owner
UC11	Choose details	Choose the details about a Party-to-be (i.e. place, date, time).	Party Master Venue Owner
UC12	Set event privacy	Set the Party-to-be privacy (i.e. secret, private, public).	Party Master

UC13	Choose necessary stuff	Choose the stuff to be inserted in the Party catalogue and their quantity.	Party Master
UC14	Edit party	Edit party details (i.e. place, date, time).	Party Master Venue Owner
UC15	Check closed party requests	Check and approve request to participate to closed parties.	Party Master
UC16	Cancel a party	Cancel a party.	Party Master Venue Owner
UC17	Give feedback	Give a feedback to the Party Goers with a thumbs-up/thumbs-down.	Party Master
UC18	Add sponsored stuff	Add sponsored products which are advertised in the list from which the Party Master picks the stuff to fill up the Party catalogue.	Party Stuff Brand
UC19	Get Pinder-points	Get Pinder-points as reward for the organisation of the party.	Party Master
UC20	Edit sponsored stuff	Edit the sponsored stuff of the Party Stuff Brand.	Party Stuff Brand
UC21	Save data	Save data from the client on the Database	Database
UC 22	Send data	Send data from the Database to the Client	Database

Use cases rank diagram

Code	Use Case name	Ranking Criteria, 1 to 5						Total Score	Priority	Build Cycle
		1	2	3	4	5	6			
UC1	See the parties list	5	3	5	5	2	2	22	High	1
UC2	See details about one party	4	3	3	5	3	2	20	High	1
UC3	Participate to a party	5	3	5	3	3	2	23	High	1
UC4	Choose what to bring	4	3	5	5	3	5	25	High	1
UC5	Cancel participation	4	3	4	5	1	1	18	Medium	2
UC6	Receive notification for cancellation	2	3	3	3	1	1	13	Low	3
UC7	Review Party Master	4	3	3	3	1	1	15	Low	3
UC8	Receive Confirmation	3	5	4	3	1	1	17	Medium	2
UC9	Invite Friends	2	5	5	4	1	4	21	High	1
UC10	Add a party	5	4	5	5	5	5	29	High	1
UC11	Choose details	4	3	4	5	1	4	21	High	1
UC12	Set event privacy	4	3	4	5	1	3	20	Medium	2
UC13	Choose necessary stuff	5	3	4	5	1	3	21	High	1
UC14	Edit party	4	3	4	5	1	2	19	Medium	2
UC15	Check closed party requests	3	3	2	5	1	3	17	Medium	2
UC16	Cancel a party	4	3	2	4	1	2	16	Low	3
UC17	Give feedback	4	3	2	2	1	3	16	Low	3
UC18	Add sponsored stuff	3	3	2	5	5	5	23[1]	Low	3
UC19	Get Pinder-points	3	2	4	3	1	2	16	Medium	2
UC20	Edit sponsored stuff	2	3	2	4	2	5	18	Low	3
UC21	Save data	3	5	5	4	2	5	24	High	1
UC22	Send data	3	5	5	4	1	5	23	High	1

Priority ranking criteria legend:

1. Significant impact on the architectural design.
2. Easy to implement but contains significant functionality.
3. Includes risky, time-critical, or complex functions.
4. Involves significant research or new or risky technology.
5. Includes primary business functions.
6. Will increase revenue or decrease costs.

[1] Even though this Use Case received a high total score, its priority is ranked as low, because monetization is not included in the first release

Traceability matrix

	UC	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Req.	PW																						
PG1	3	✓	✓																			✓	
PG2	4			✓	✓																	✓	✓
PG3	2					✓																✓	
PG4	1									✓													
PG5	3							✓														✓	✓
PG6	3						✓		✓													✓	
PM1	6										✓	✓	✓	✓								✓	✓
PM2	3															✓						✓	✓
PM3	4														✓		✓					✓	✓
PM4	3																	✓				✓	✓
PM5	2																			✓		✓	
PM6	2						✓															✓	
PM7	1									✓													
VO1	3										✓	✓											✓
VO2	4														✓		✓					✓	✓
PB1	4																		✓		✓	✓	✓
Max PW		3	3	4	4	2	3	3	3	1	6	6	6	6	4	3	4	3	4	2	4	6	6
Total PW		3	3	4	4	2	5	3	3	2	9	9	6	6	7	3	8	3	4	2	4	38	39

A legend for the codes can be found in User stories and Use cases tables.

Use cases descriptions

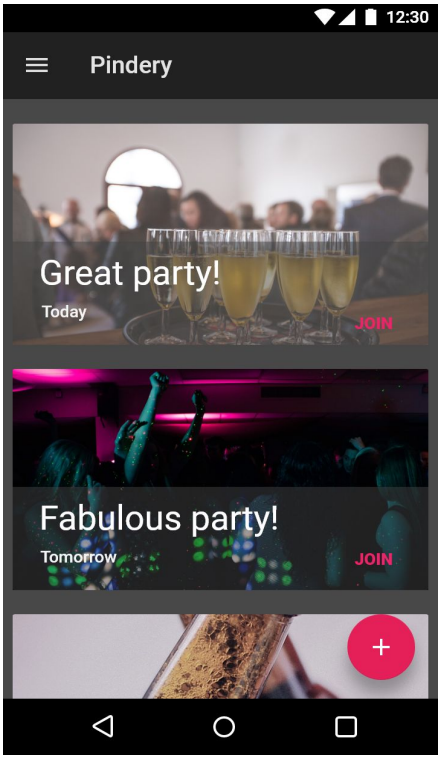
Use Case UC1 & UC2	See the parties list & See details about one party
Related requirements:	PG1
Initiating actor:	Party Goer
Actor's goal:	To see the list of nearby parties and see the details about one party.
Participating actors:	Database (DB)
Preconditions:	The **Party Goer** has already logged into the application.
Postconditions:	The **Party Goer** will have seen the list of nearby parties and the details about them. Potentially he will choose to take part to one party.

Flow of events for main success scenario:

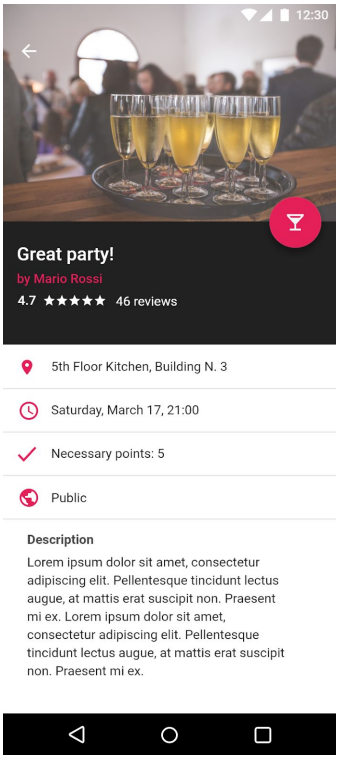
→	1.	The **Party Goer** scrolls the parties list, which is downloaded from the **DB**.
←	2.	The **Party Goer** eventually chooses one party whose details is interested in, and opens its specific page, whose data are downloaded from the **DB**.

Flow of events for extensions:

←	1a.	The **Party Goer** doesn't like any of the parties, closes the app.



UC1



UC2

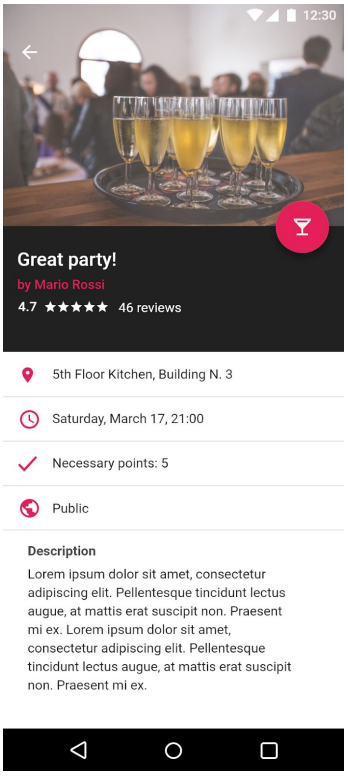
Use Case UC3 & UC4	Participate to a party & Choose what to bring
Related requirements:	PG2
Initiating actor:	Party Goer
Actor's goal:	To initiate the participation to a party.
Participating actors:	Database (DB)
Preconditions:	The **Party Goer** has already opened the Party details view and pushed the designated button.
Postconditions:	The **Party Goer** will have chosen what to bring to the party from the Party catalogue.

Flow of events for main success scenario:

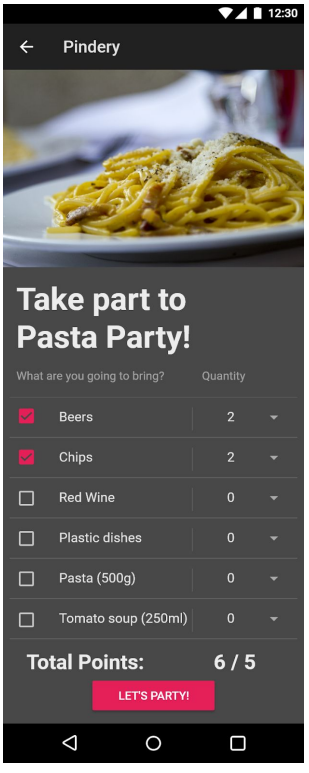
→	1.	The **Party Goer** chooses to participate to the party.
←	2.	The **Party Goer** chooses what to bring from the Party catalogue, downloaded from the **DB**, in order to gain the necessary Pinder-points, which are defined by the algorithm. Every piece of stuff is equivalent to a predetermined amount of points, decided a priori, and stored on the **Database**.

Flow of events for extensions:

←	1a.	The **Party Goer** doesn't like the requested stuff and goes back to the Parties list.



UC3

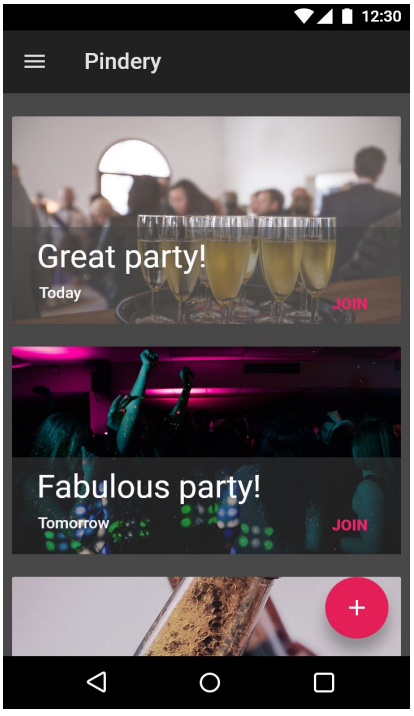


UC4

Use Case UC10, UC11 & UC 12	Add a party, Choose details & Set event privacy
Related requirements:	PM1
Initiating actor:	Party Master
Actor's goal:	To initiate the creation of a party, choose the details and set the event's privacy.
Participating actors:	None
Preconditions:	The **Party Master** has already logged in.
Postcontotions:	The **Party Master** will have chosen what Party Goers will bring to the party in the Party catalogue screen.

Flow of events for main success scenario:		
→	1.	The **Party Master** pushes the "+" button.
←	2.	The **Party Master** sets the details about the party (i.e. location, description, date, time, maximum number of people and necessary Pinder-points).
←	3.	The **Party Master** sets the event's privacy (i.e. Public, Closed, Secret).

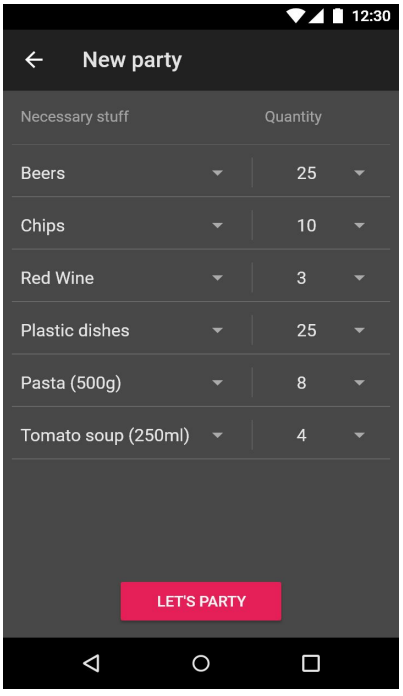
Flow of events for extensions:		
→	1a.	The **Party Master** wants to cancel the creation of the party: pushes the back arrow, goes back to the Parties list screen.
←	2a.	The **Party Master** wants to edit the details while on the Party catalogue screen: pushes the back arrow, goes back to the Party details screen.



UC10

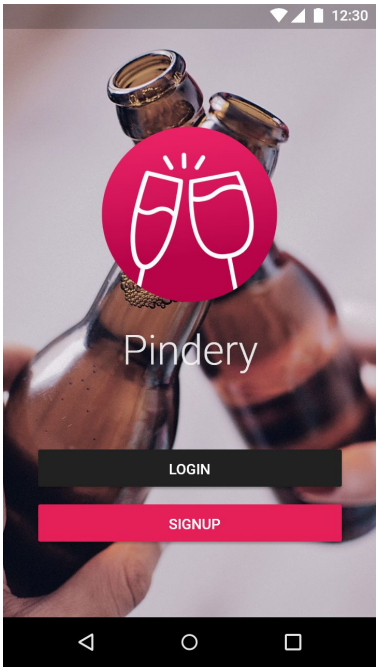
UC11 & UC12

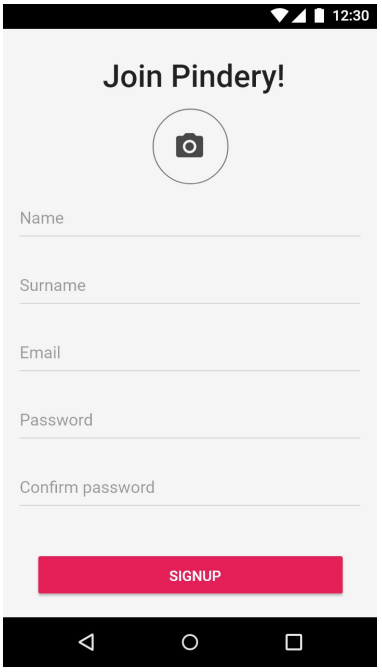
Use Case UC13		Choose necessary stuff
Related requirements:		PM1
Initiating actor:		Party Master
Actor's goal:		To initiate the list of necessary material for the party.
Participating actors:		Database (DB)
Preconditions:		The **Party Master** has already chosen the details about the party.
Postconditions:		The **Party Master** will have published the party.
Flow of events for main success scenario:		
→	1.	The **Party Master** chooses from the pre-defined list a product, downloaded from the **DB.**
→	2.	The **Party Master** sets how much of the product is necessary for the entire party.
←	3.	The Party Master publishes the party, pushing the "Let's party" button. The **DB** saves the data.
Flow of events for extensions:		
→	1a.	The **Party Master** wants to delete an element from the list: sets its correspondent quantity to 0.
←	2a.	The **Party Master** wants to edit the details while on the Party catalogue screen: pushes the back arrow, goes back to the Party details screen.

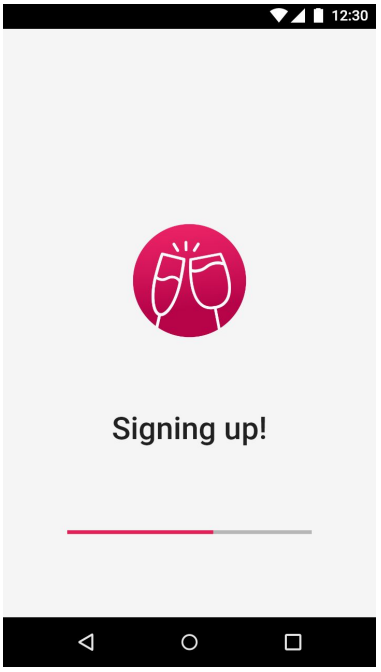


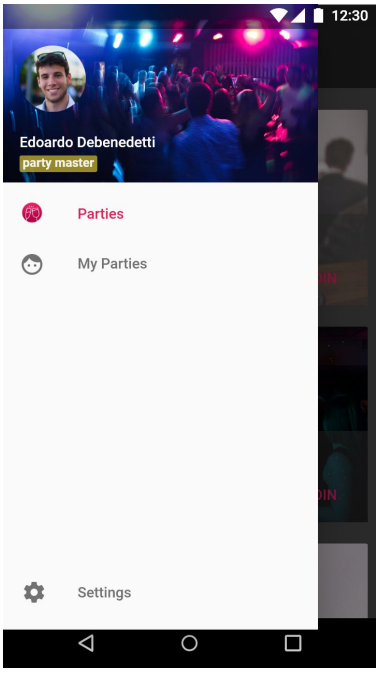
UC13

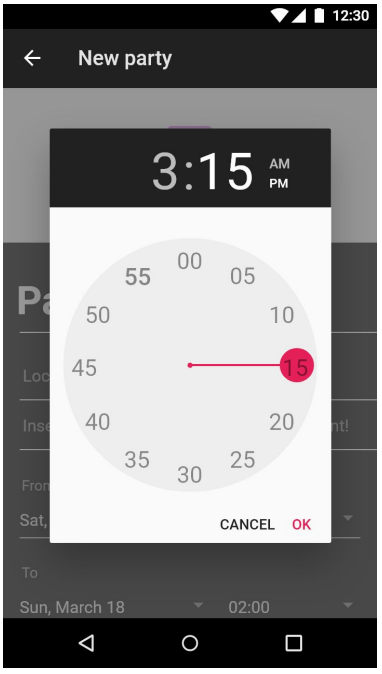
GUI mockups

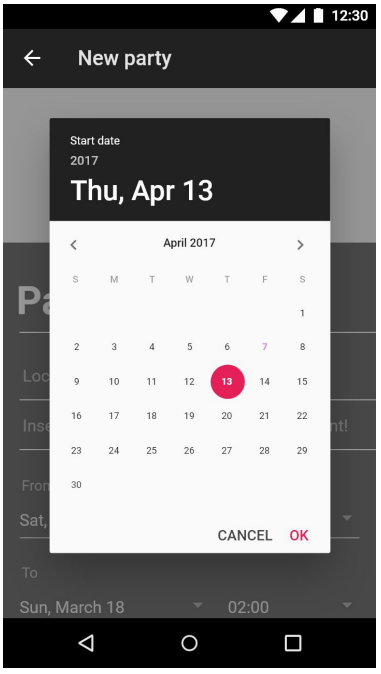

12:30
Pindery
LOGIN
SIGNUP

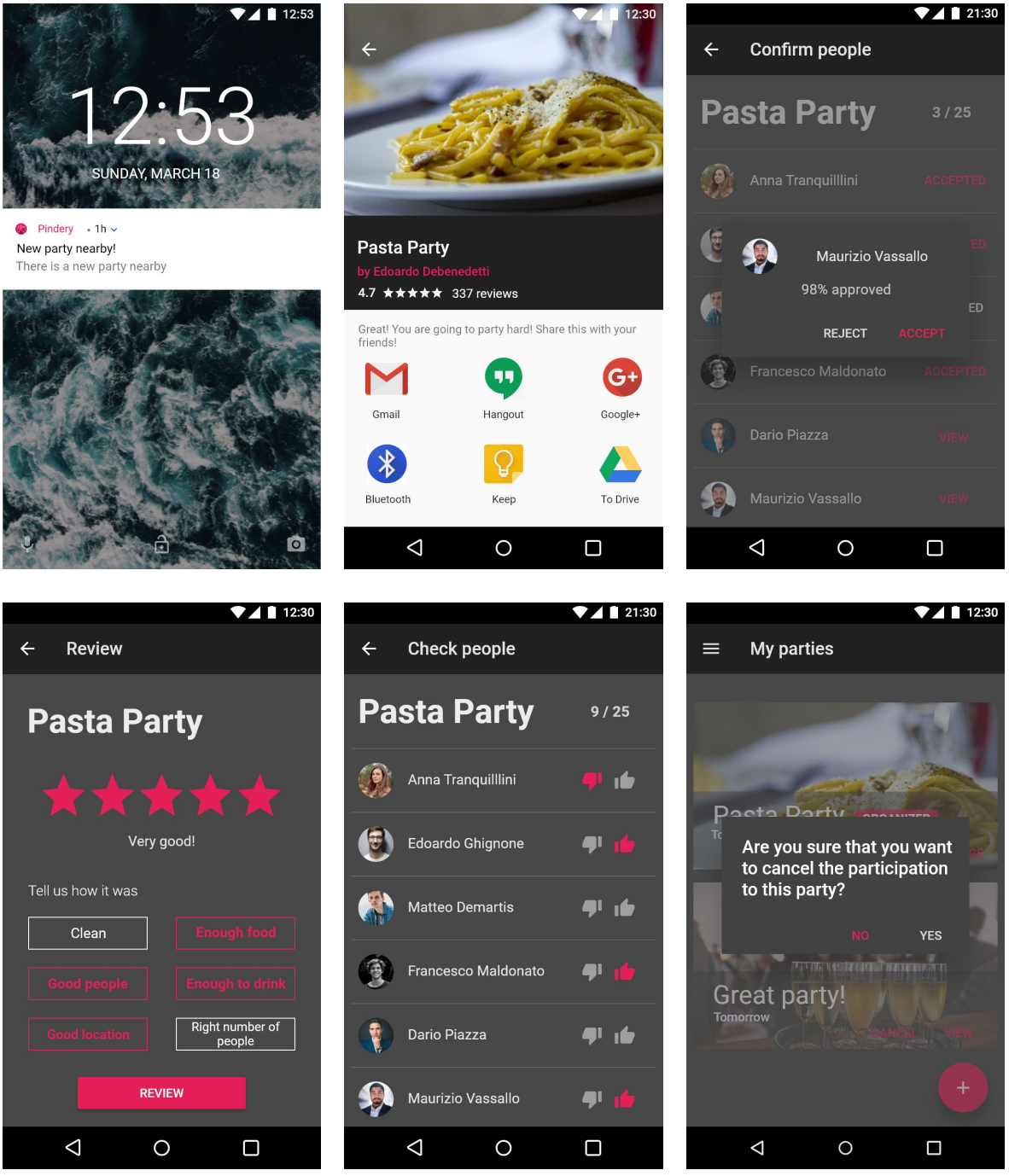

12:30
Join Pindery!
Name
Surname
Email
Password
Confirm password
SIGNUP


12:30
Signing up!


12:30
Edoardo Debenedetti
party master
Parties
My Parties
Settings


12:30
New party
3:15
AM
PM
CANCEL
OK
Sun, March 18
02:00


12:30
New party
Start date
2017
Thu, Apr 13
April 2017
CANCEL
OK
Sun, March 18
02:00



More mockups can be found here.

PART 3

Mathematical model

The essential feature that makes Pindery better than its competitors is the point system that substitutes any exchange of money. To make it work we have thought about an entire model.

Point Attribution

By analysing the market value of a set of basic items we developed this elementary table, that correlates the items in the catalogue with the number of points one needs to gain to bring that item to the party.

Party Catalogue	Points attributed
General beverage (Cola, Sprite, etc)	2 pt
Beer (330ml)	1 pt
Beer (660ml)	2 pt
Beer(3*330ml or 1 l)	3 pt
Wine (750 ml)	5 pt
Vodka	8 pt
Rum	10 pt
Snacks (200gr pack)	1 pt
Chips (200gr pack)	1 pt
Napkins (50 pieces pack)	1 pt
Cups (50 pieces)	2 pt

Clearly the party catalogue, is constantly expanding and this is only an elementary set.

Point Subdivision

The algorithm to determine the minimum number of points a Party Goer needs to gain is pretty straightforward.
First, the sum of all the points for the party is evaluated.

$$Sum = \sum_{i=0}^{n} p_i n_i$$

Where p_i is the value (in points) of the item i and n_i is the number of occurrences of the item i necessary for the entire party, as the party master determined.

For instance, if we have this kind of party:

PARTY 1	
Catalogue items	**Occurrences**
Beer (66ml)	25
Chips (150gr pack)	20
Napkins (50 pieces pack)	4

The sum will be then:

$$Sum = 25 \cdot 2\,pt + 20 \cdot 1\,pt + 4 \cdot 1\,pt = 74\,pt$$

Then we divide the sum by the total number of participants and approximate the result to the closest entire. Let's say this party has 20 people attending it, then:

$$ppp = 74\,pt / 20\,people = 4pt$$

Where ppp means "points per person".
Now we have the number of points that each person will need to bring to the party, and since this number is a little bit overestimated there is no need to worry for a lack of food, drinks or anything else.

Smart Grouping System for catalog items division among people

We have developed a system to manage the distribution of items from the catalogue among the Party Goers in order to maintain the balance between **freedom** to choose what to bring to a party and the **necessary balance** among all the items.
The core of this algorithm is the delta of items brought or Δ_{ib}.
This delta points out the maximum difference, in percentual points, that there can be between the most chosen item and the least chosen one.

Let us go back to the example to better understand what does it mean.
Let us assume that in the previous situation of Party 1 already some people took part to the party.

PARTY 1		
Catalogue item	**Occurrences**	**Brought**
Beer (66ml)	25	12
Chips (150gr pack)	20	6
Napkins (50 pieces pack)	4	3

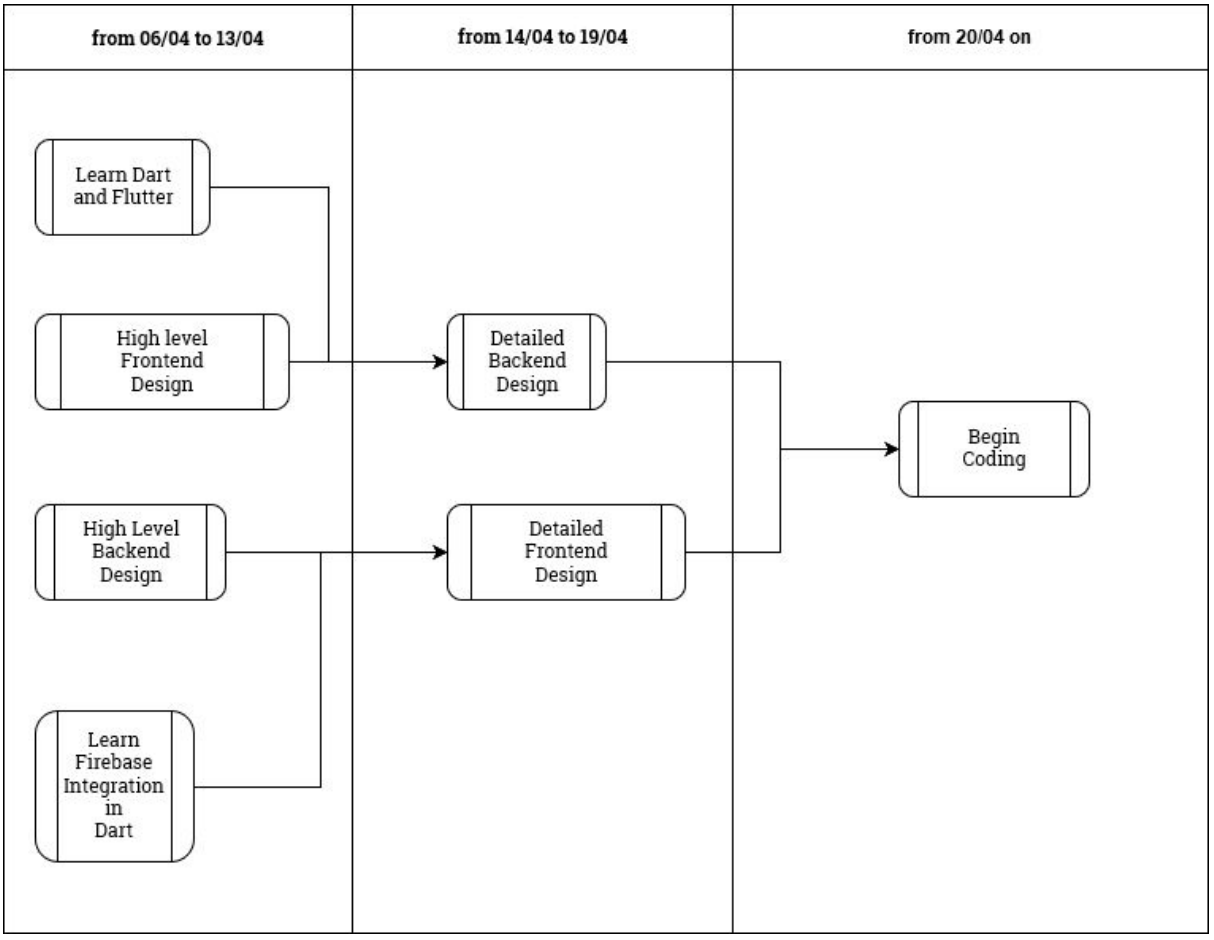
48% of the beers are already taken while only 30% of the chips are going to be at the party, so that means there is a 18% difference.
In this case, because the gap between the two percentages is higher than $\Delta_{ib} = 15,71\%$, the next person who will try to join the party, will not be able anymore to choose beers.
Assuming then that he will bring a pack of chips, the percentage of chips brought will rise to 35% and the difference between chips and beer will be lowered to 13% so that the user will be able again to select beer.

The Δ_{bi} algorithm follows:

$$\Delta_{ib} = \frac{\pi}{(people\ attending\ the\ party)}$$

Clearly, when the total number of points will reach Sum, no more limitations will be imposed on what the party goer should choose to bring.
On the other hand, when a single item will reach its limit, it will not be selectable until the total number of points reaches Sum.

Next steps



Since we follow the Agile Process Development through Scrum, the **Testing** and **Writing Documentation** steps will be carried on in parallel with the **Coding.**

Work Distribution

We will all contribute the same, but in different ways.

Anna Tranquillini: she is working on the front end, studying the language to suit the need of a flexible, disruptive and scalable client, that will be suited on everybody's need for partying.

Edoardo Debenedetti: he is working on the synergy between the parts of the application, working on both front end and back end, controlling everything and always looking one step forward.

Edoardo Ghignone: he is working on the backend, exploiting the power of Google Firebase, to satisfy all the needs of a connected distributed service such as Pindery.

References

- Flutter, Beautiful Native Apps in Record Time
- Dart Programming Language
- Google Firebase, Build Apps Fast, without Managing Infrastructure
- Draw.io, Flowchart Maker and Online Diagram Software
- UML Use Case diagrams tutorial
- Adobe XD. Design. Prototype. Experience.
- Random Cats on Google Images
- Agantty, free GANTT chart