SOCIALENDAR

By:
Yunus Güngör
Sinan Kartal
Mustafa Sağlam
M. Enes Kayılıoğlu
Recep Can Babaoğlu

PROJECT PRESANTATION

What Did We Plan?

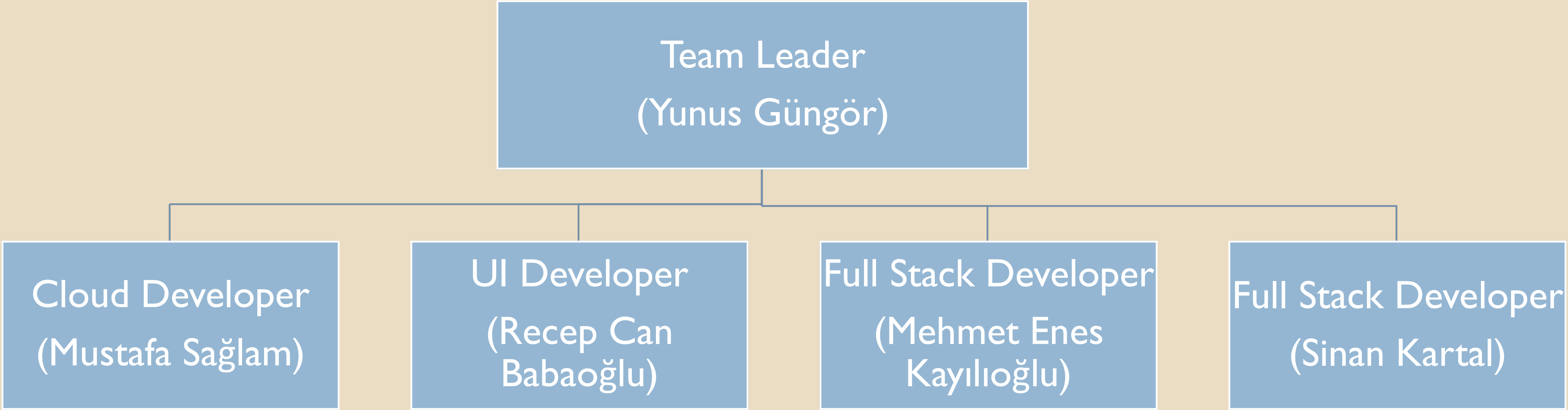
Scope:

- Socialander is an event based chat application for Android and iOS
- Users attending to the same event grouped in a chatroom
- Events taken out of device calendar so everything automatic and user does not show any effort to make things happen

Description:

- The Socialendar application can bring together people who will same events efficiently, easily and precisely. These people are using mobile phone calendar and this app brings together these people in a chatroom if they want. User only need to login to add a new event or to edit current events. Then application take users in a chat room and provide users a communication channel. Users can sign up and sign in with Facebook, Google and mail accounts and application will take their events from device calendar or Facebook. The Socialendar is a JavaScript based application that uses React-Native for mobile phones which have iOS and Android. The application will interface with and utilize third party resources to facilitate all the users' meeting scheduling needs (e.g. database services, email communication, etc.).

Our Team



Team Leader
(Yunus Güngör)

Cloud Developer
(Mustafa Sağlam)

UI Developer
(Recep Can Babaoğlu)

Full Stack Developer
(Mehmet Enes Kayılıoğlu)

Full Stack Developer
(Sinan Kartal)

What Did We Plan?

Work Breakdown Structure

Number of work package	Task	Assigned to	Epic	Dependence to work package	Size
1	Firebase, React native setup	Member 1	7		S
2	Adding necessary modules and libraries	Member 1	7	1	S
3	Authentication	Member 1	6	2	M
4	User log in and log out	Member 1	6	2	M
5	Sign in with Facebook	Member 1	6	3	M
6	Sign in with Google	Member 1	6	3	M
7	Sign in with Mail	Member 1	6	3	M
8	Adding new user on database	Member 1	6	2	M
9	Checking authentication status	Member 1	6	3	M
10	Getting event information from device calendar	Member 2	3	2	M
11	Getting access to Facebook events of user	Member 2	3	2	M
12	Getting data from Facebook events	Member 2	3	11	M
13	Parsing Facebook event data and synchronization	Member 2	2	12	M
14	Synchronization between calendar and online database	Member 2	2	10	M
15	Matching events that occur at the same time on cloud	Member 3	5	14	M
16	Calculating a point between events occur at the same time	Member 3	5	15	M
17	Matching events that has close points	Member 3	5	16	T
18	Creating chat structure	Member 4	4	3	L
19	Being able to send message	Member 4	4	18	M
20	Being able to receive message	Member 4	4	19	M
21	Being able to create chatrooms between users	Member 4	4	20	M
22	Being able to add more than one user to chatrooms	Member 4	4	21	M
23	Creating chatrooms for matched events	Member 3	5	22,17	S
24	Adding events to calendar	Member 2	3	2	M
25	Adding events to database	Member 2	3	2	M
26	Creating add event ui	Member 5	1	24	S
27	Organizing and polishing add event ui	Member 5	1	24	L
28	Creating ui to view events	Member 5	1	14	S
29	Organizing and polishing view event ui	Member 5	1	14	L
30	Creating home screen ui	Member 5	1		M
31	Creating a function to access recent messages	Member 4	4	22	M
32	Showing recent messages on home screen	Member 4	1	22	S
33	Showing events on home screen	Member 3	1	14	M
34	Creating a function to calculate next empty time of the user	Member 2	3	14	M
35	Showing next empty time on home screen	Member 3	1	24	S
36	Creating chat ui	Member 5	1	20	M
37	Organizing and polishing style of chat ui	Member 5	1	20	M
38	Being able to see weather forecast on event time	Member 5	1	29	M
39	Optimizing event points	Member 3	5	17	L
40	Coding event editing function on calendar	Member 3	3	10	M
41	Coding event editing function on database	Member 4	2	10	M
42	Creating event editing ui	Member 4	1	41,40	M

What Did We Plan?

Deliverables:

- –Mobile Application
- –Project Plan
- –Project Requirement
- –Project Design
- –Mobile Application Source Code

What We Did?

- Deliverables
  - Mobile Application
    - Only a part delivered
  - Project Plan
    - Delivered
  - Project Requirement
    - Delivered
  - Project Design
    - Delivered
  - Mobile Application Source Code
    - Only a part delivered

- Customer Meetings
  - 1 meeting with group
  - 2 meeting with group leader

- Work Packages
  - 9 out of 42 work packages completed
  - Most of the components did not implemented

- Live Demo
  - Users can login to the application

Overall: Project is Failed

How did we Implement?

- React Native
- Redux
- Firebase
- And other react packages:
  - React Native Calendar Events
  - React Native Device Info
- Facebook and Google API's for sign in
- Weather API

React Native

- React Native is a framework for building native applications for Android and iOS with JavaScript and React.

Why React Native ?

- Easy to code
- Using ES6 Syntax of JS, most of the can be done asynchronously which means butter smooth UI
- Implementing an UI design is pretty easy
- Supports object oriented approach but does not have any structure to hide data
    - No application wide structure for storing context – Solved by Redux
    - Every component acts as a one page application of its own therefore there is no navigation – Solved by React Native Router Flux

Redux

- Redux is a React component that creates an application wide state for use with any component

How to Redux?

- Redux has a structure that separates data and functions.This approach is very useful for application state
- Data called reducer in redux
- Functions that interacts with data (read or write) called actions
- Every time an action called a new state created this very useful when debugging application

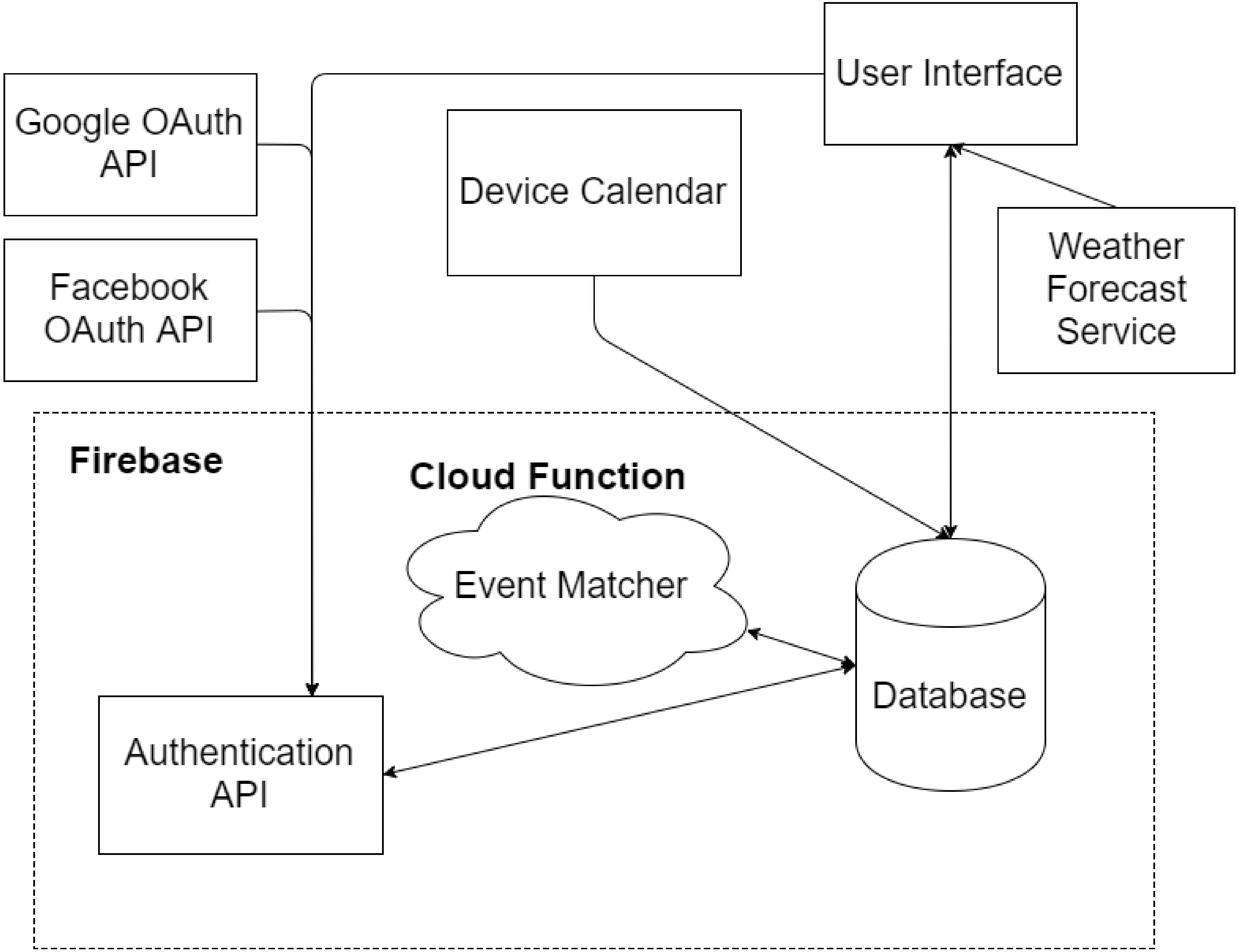
Firebase

- Firebase is a web service for application development backend. Firebase provides database, authentication server, storage and statistical tracking tools

Why Firebase?

- Easy to code
- Easy to provide security
- Easy to authenticate users
- Real time database – Very important for chatting
- Cloud functions written in JS

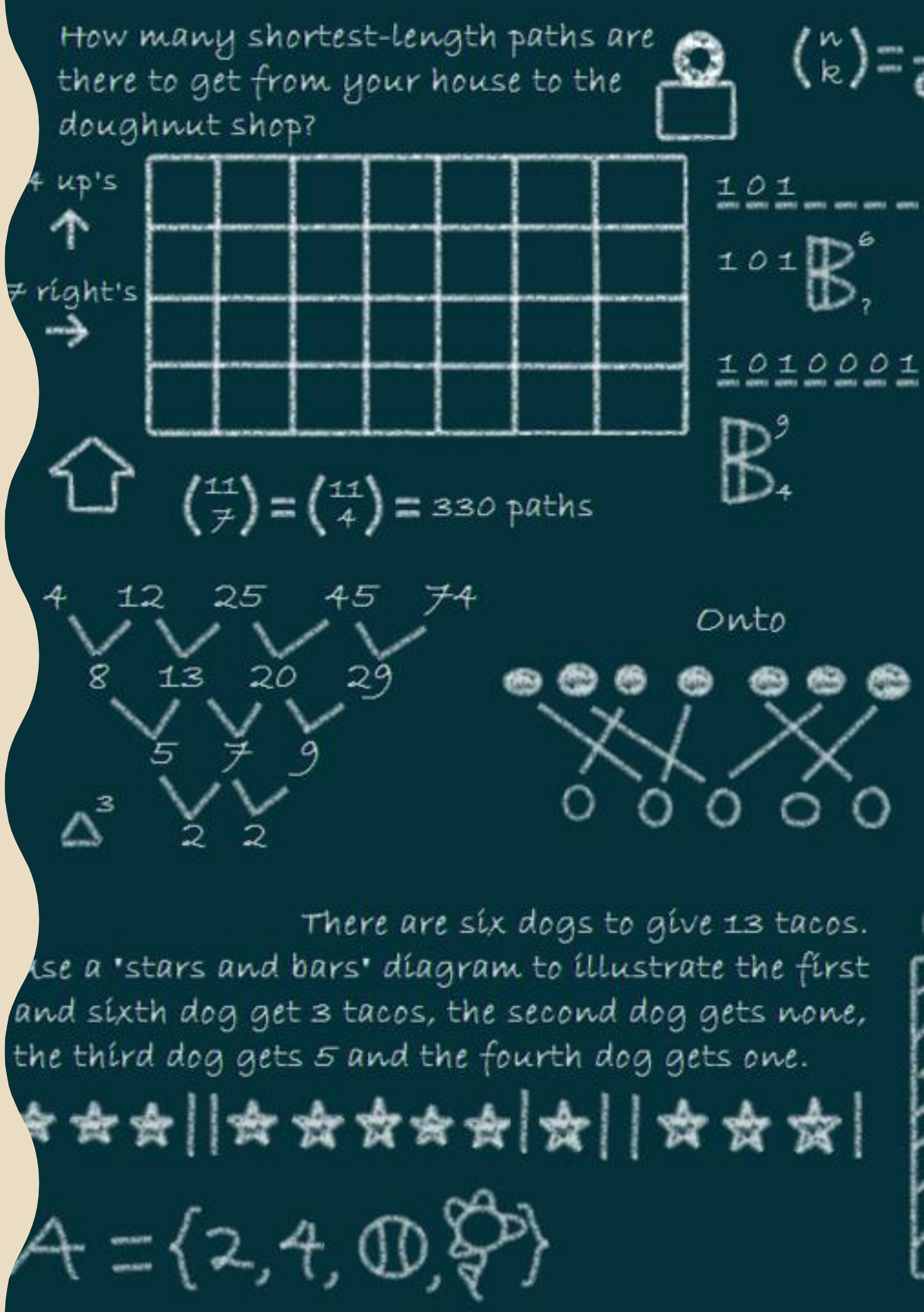
SYSTEM ARCHITECTURE



Heart of the application: Event Matching Algorithm

- Each event has a globalId value that matches with a chatroom
- Algorithm creates a matching point between events (likelihood)
- Matching point created according to the event’s name and similarities in name
- A threshold value used for eliminating non similar events
- Event migrates highest point’s globalId
- If there is no event left after thresholding, a new globalId and chatroom created

  – This algorithm has been planned to run on the cloud automatically


How many shortest-length paths are there to get from your house to the doughnut shop?
4 up's
7 right's
$\binom{11}{7} = \binom{11}{4} = 330$ paths
101
1010001
4 12 25 45 74
8 13 20 29
5 7 9
2 2
Onto
There are six dogs to give 13 tacos.
Use a 'stars and bars' diagram to illustrate the first and sixth dog get 3 tacos, the second dog gets none, the third dog gets 5 and the fourth dog gets one.
A = {2, 4,

Other aspects of the application:

- Besides Event Matching application mostly acts as a data input output process. Most of the components take data from a source and redirects it to another source.
- Those sources might be
  - User
  - Device Calendar
  - Database
  - Forecast Service

Testing

- Only implemented module was login module
  - This module tested manually due to high dependence to the other APIs and User Interface
- Test Cases
  - Login with a nonexistent user
  - Sign up as a new user
  - Login with a wrong password
  - Login with an existing user
  - Sign Up with Google
  - Sign Up with Facebook
  - Login with Google
  - Login with Facebook
  - Login with Google to an account signed up with Facebook
  - Login with Facebook to an account signed up with Google
  - Login automatically with previously authenticated device

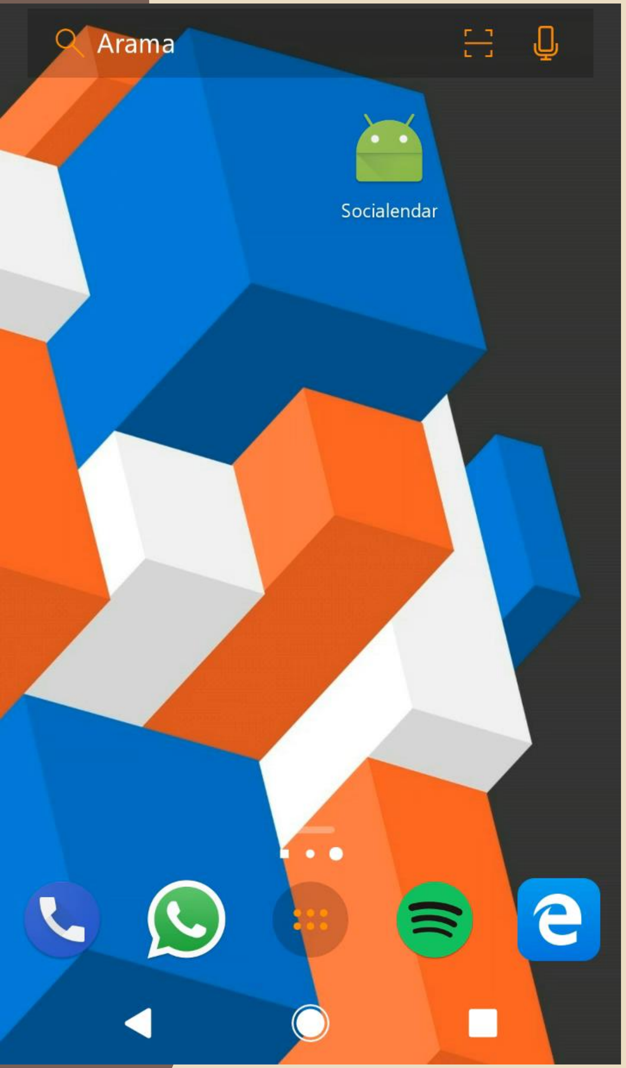
How we failed?

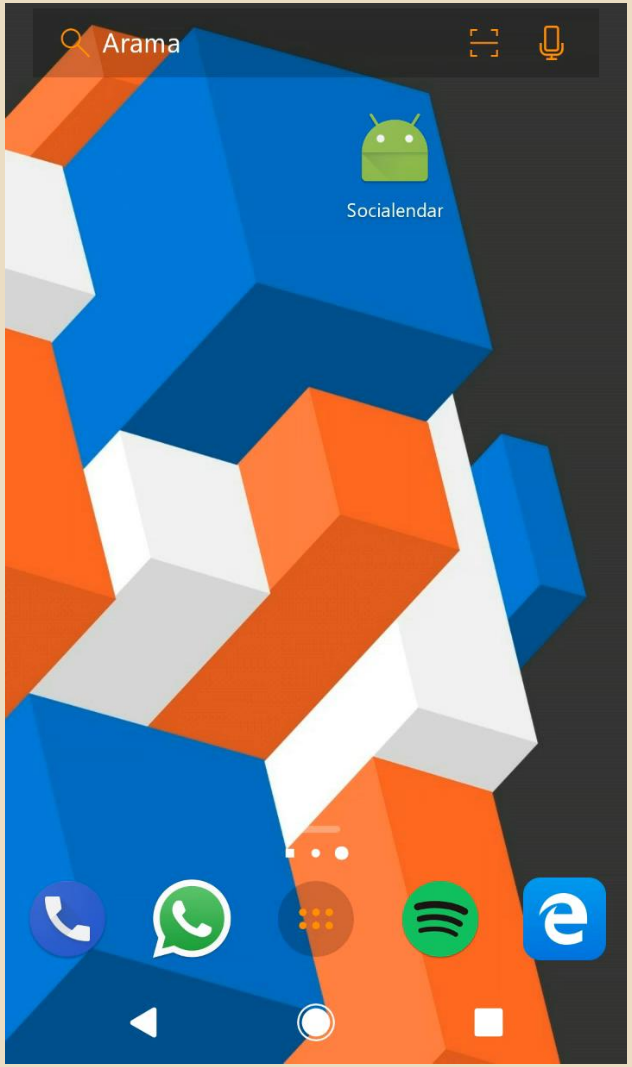
- Lack of team work. Our team working together for the first time.
- Lack of time. There is simply not enough time catch up with every project student has.
- Lack of ambition. Since this is a mandatory assignment without no benefit other than grade and knowledge participants was not ambitious.
- Lack of supervising. Since each member is an equal member of team, there was nobody to enforce rules and push team members. Consider that team leader has no power besides telling people `can you do this in time please`.
- Lack of self management. Since each member is an independent student, each member supposed to manage themselves to achieve deadlines on schedule. But this did not happen

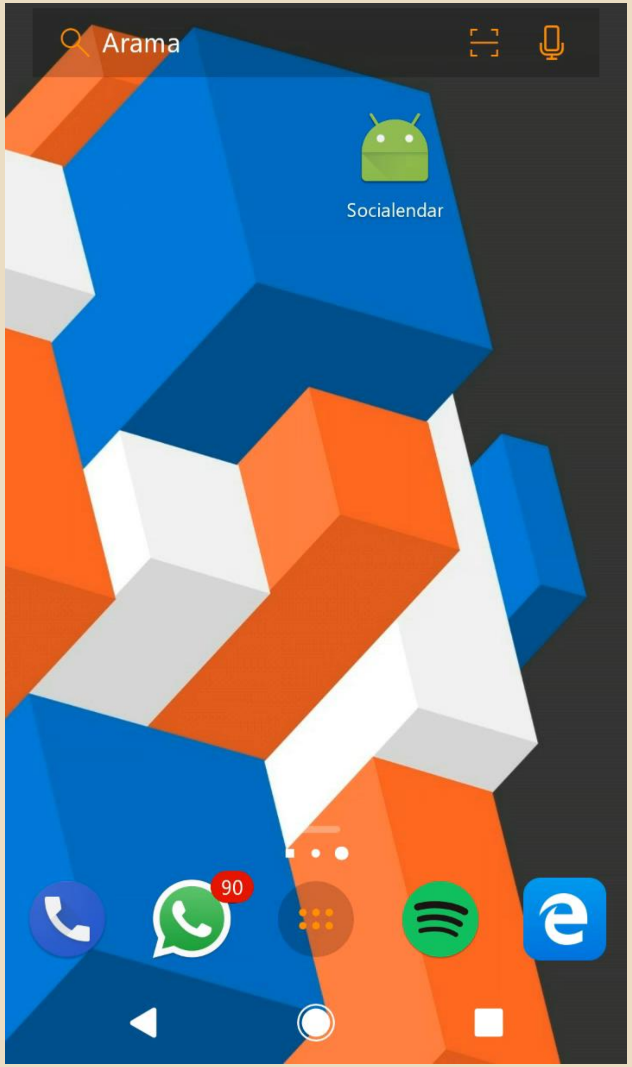
Risks that come true

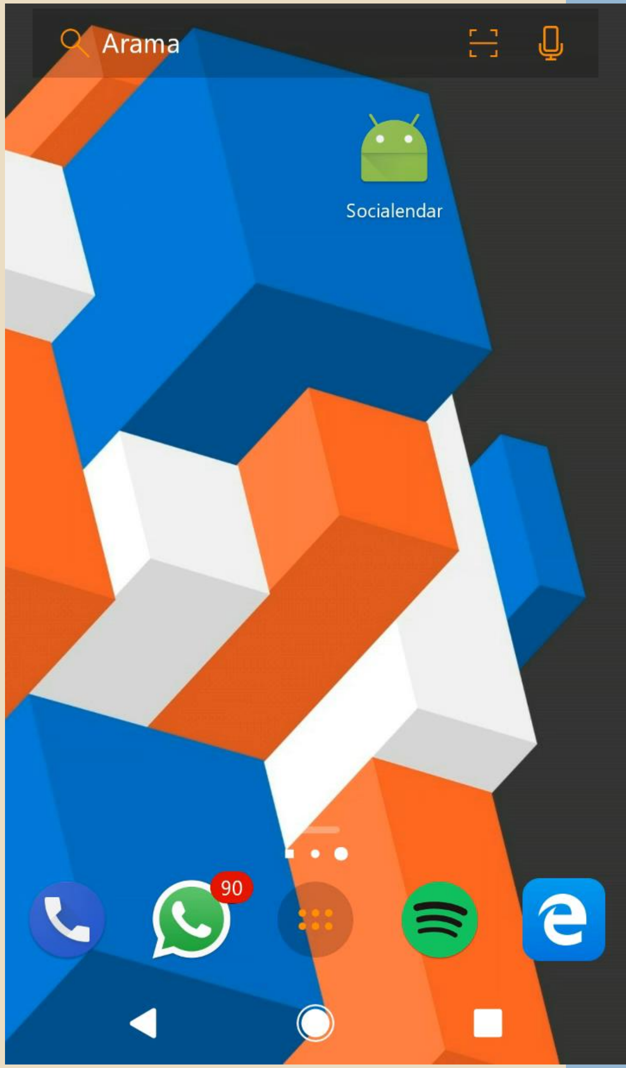
- Wrong time estimation (risk no 1 in project plan)
- Low communication in team (risk no 6 in project plan)
- Steps are not tracked properly (risk no 9 in project plan)

Demo


Arama
Socialendar


Arama
Socialendar


Arama
Socialendar
90


Arama
Socialendar
90