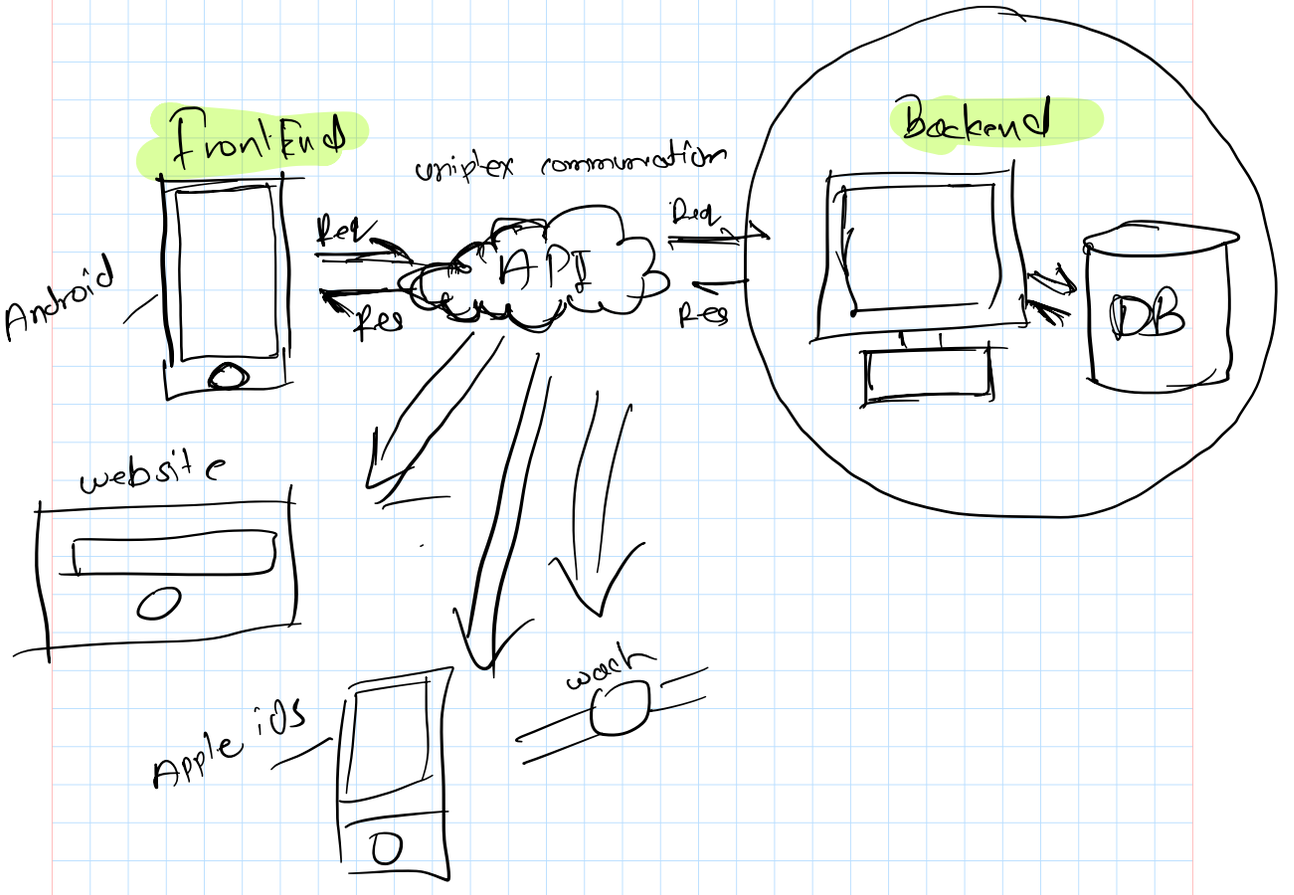
// Todo, post content on Git

Product Design



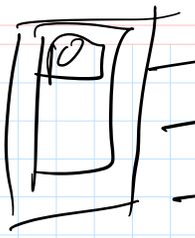
Phases of S/W Development Life Cycle.

1) Planning
2) Design
3) Development
4) Testing
4) Deploy.

Plan

Collaborators — Business, Development Designers Team.

Product Design Mindset



- Feature
- Requirement
- Approach – optimised Solution

Teams Involved

Business – , UI UX - Team , Backend

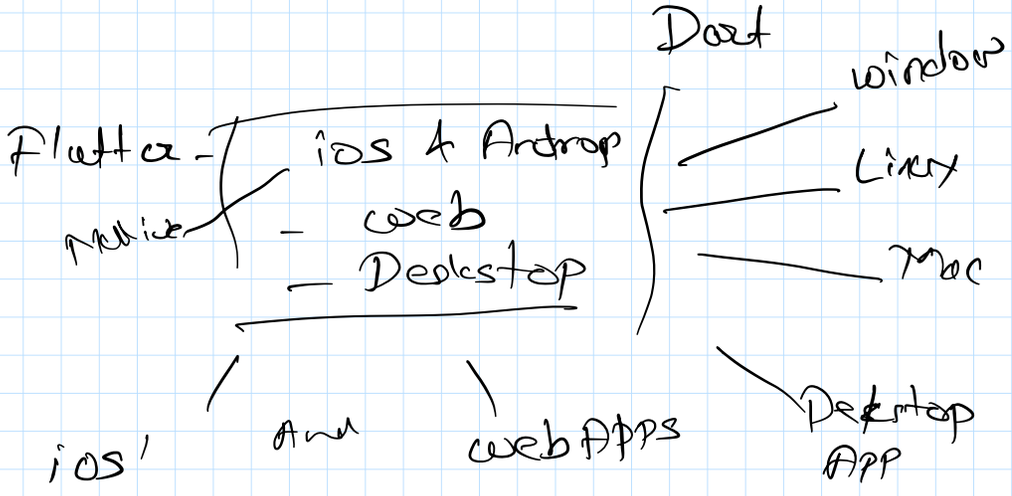
Front End Team: Flutter Developer

(Mobile Engineers)

Programming Lang

Native
- ios → swift
- Android → Java & kotlin

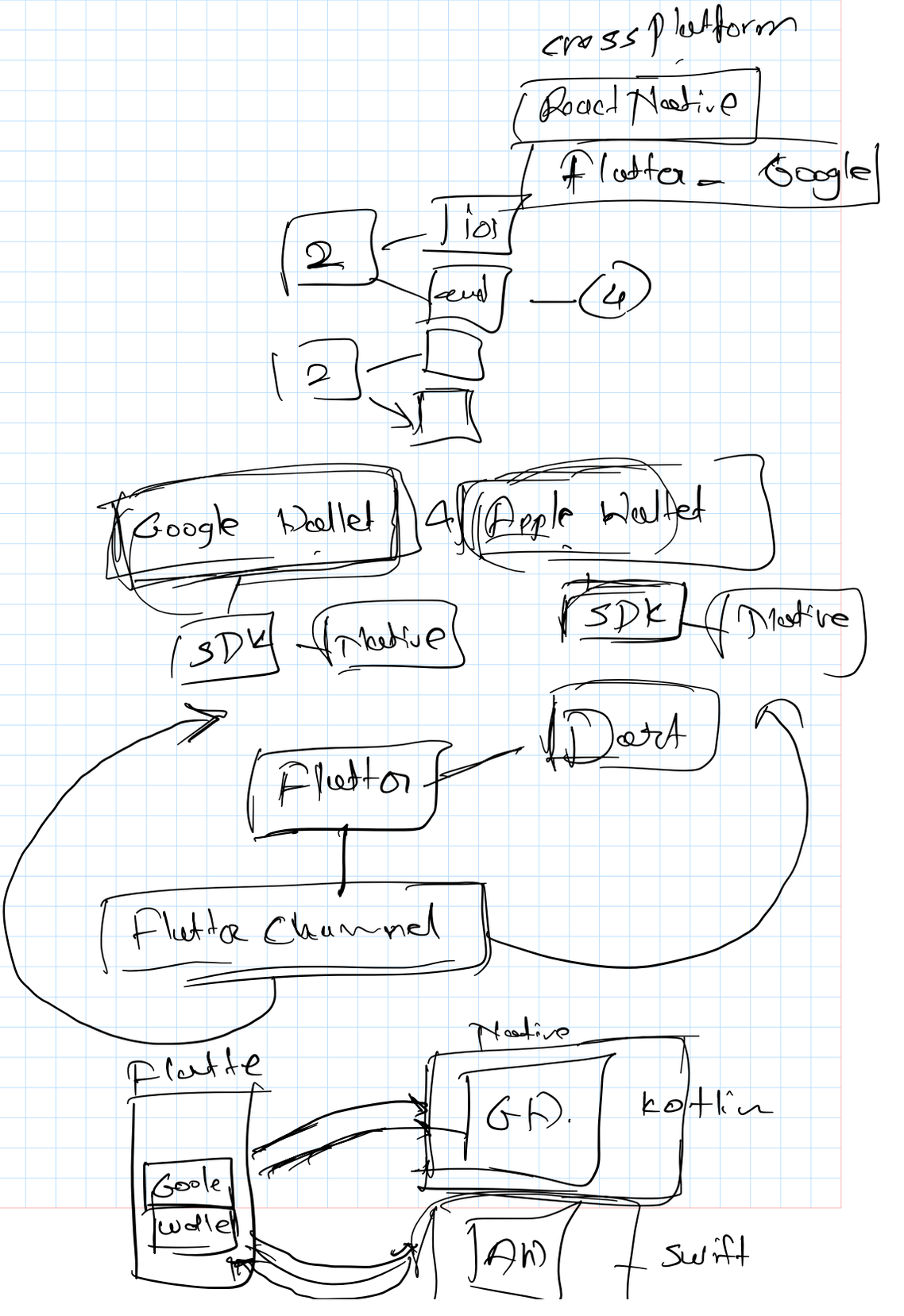
Cross Platform → React Native → JS, ReactJS.
→ Flutter → Dart.



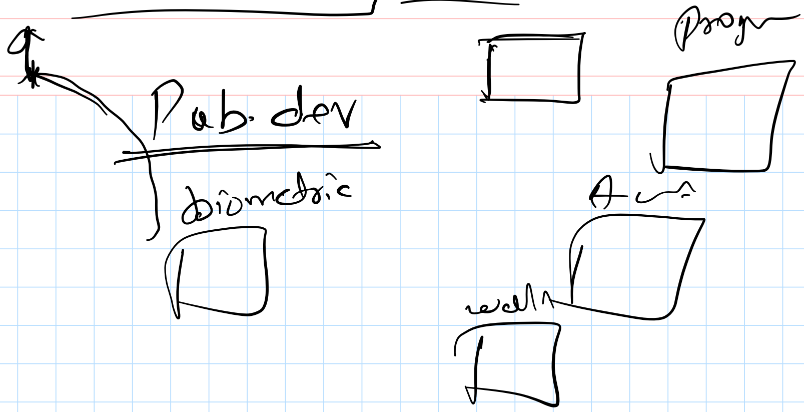
Dart Language – Flutter Development

Native 2 android - 2 QA → 8 member Team
2 ios - 2 QA

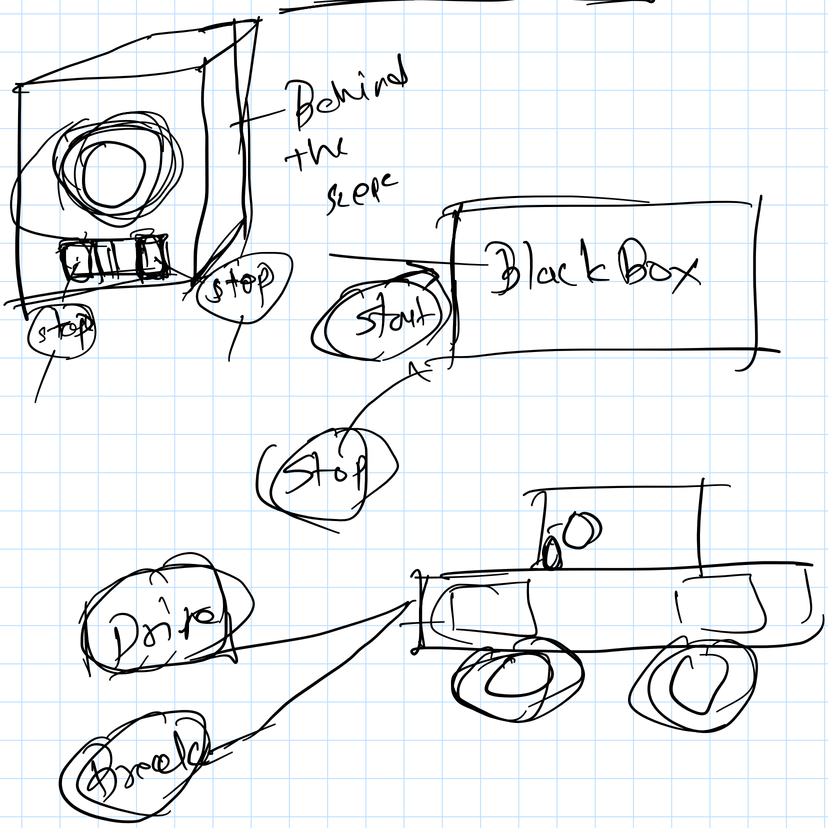
Flutter 2 Flutter - 2 QA → 4 member Teams.



Flutter channel | Native



Encapsulation & Abstraction.



car

Engine
Tyre
Petr

Car car = new Car
car.drive()
car.break();

car.en
car.ti

public void drive( )
{ Engine start

}
public void break( )
{ Engine stop.

}
{

}

Abstraction-
Abstract class & Interfaces

method
FlyingObjects

plane
bird
God created planets

Flesh
Phosphorus

Fly

Mammals
eat
sleep
fly

Elephant
Goat
Bird

Bird
eat
sleep
fly

Goat
eat
sleep

abstract class Mammal { — Goat Elephant Bird

eat( ) { }
sleep( ) { }

fly ()

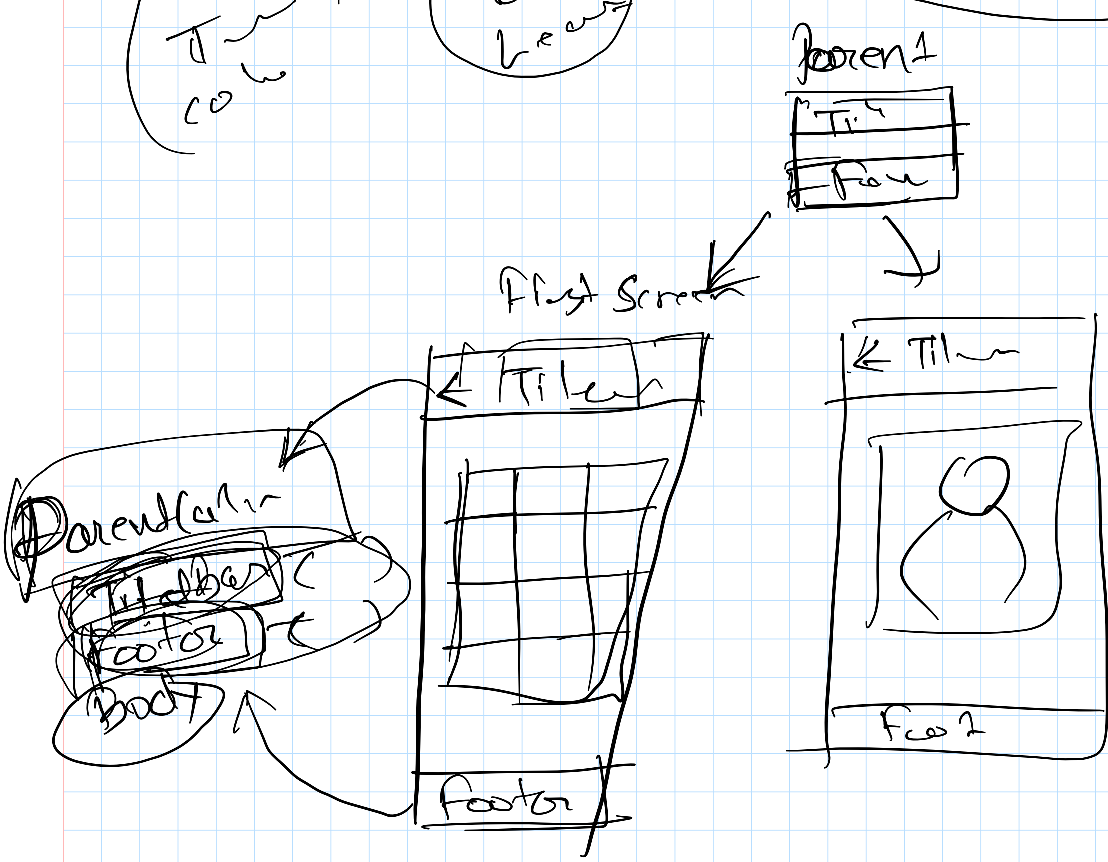
Bird extends Mammal
{
fly = true;

Good ext Ma
{
fly = false

Flier ex
fly = false

Inter Car.

Car (
Color
FeatureType
Engine Power



screenFirst extend Parent

{ Tu

Body { Table content

}

screen Second extends Parent

{ Tu

Body (profile screen)

}

130+

Abstraction.

H.W. = Abstract classes & Interfaces.

= Polymorphism — overridding

— overloadding

= Encapsulation -

= Inheritance — Base Classes

Custom Button extends

shape (Oval)

→ - Introction of Mobile Engineering
- How (design) & product Mindset Works
→ (Front) + (Backend)
→ Native vs Cross Platform (flutter)

→ Oops (oncepts. H/w

- Encapsulation
- Abstraction (Abstract class, Interface)
- Polymorphism (overloading, Overriding)
- Inheritance.

How in real project we leverage concept of