03-Idea

- Github - PDF (https://github.com/FrankBevr/100-Ideas/blob/main/practice/03/03-Ideas.pdf)

Overview

- Intro
- Think
  - ☑ Words
  - ☑ Sentences
  - ☑ Rate
  - ☑ Select
- Sketch
  - ☑ Idea
  - ☑ Requirements
  - ☑ Stories
  - ☑ Diagrams
    - ☑ Use Case
    - ☑ Class
    - ☑ Sequence
- Paint
  - ☐ cargo contract new our_great_idea
  - ☐ cargo contract build
  - ☐ cargo contract call

🧠 Think

100 Words

Words				
Music	Hackathon	Germany	Village	Cow
Headphones	Female	Iot	Blockchain	Merging
Painting	Picures	Photoshop	3D	Animation
Manufacturing	Lecturing	University	Grapefruits	
Thailand	Bangkok	Friends	Jackets	Table
Bread	opencv	poster	coffee	hands
biology	brain	neuron	neural network	ai
chatgpt	copilot	smart contracts	frontend	react
vue	angular	html	javascript	figma
editing	sounds	sound effect	games	leather
metal	wood	cement	construction	carpenter
roof	statue	bin	digital bin	history
renesseaice	1600	machines	ships	cannons
games	laws	governance	union	health
cancer	health data	dolphin	ar	vr
xr	cards	copouns	hair	cutting
listenting	headphones	taste	food	feeling
avatars	metaverse	matrix	hacking	morhpheus
tutorial	rust	fireship	family	kids
uiux	video	caputring	laugh	smile
chair	greetings	ethics	chemics	language

and again

100 Words

Words				
Kid	Blocks	Kindergarten	Mandela	Pen
Writing	Letters	Football	Basketball	Stone
Dam	Water	Earth	Friends	Mon
Grandma	House	Bycylce	Slide	Stofftier
School	Math	Community	Clown	Horses
Computer	Mouse	Book	Reading	MiddleShool
ClassTeacher	Karate	JackyChan	Pokemon	Indianer
Tent	Choir	Piano	Walking	Straw
Dog	Cat	History	Grammar	SchoolTrip
River	Skiing	Traveling	America	LasVegas
Car	Tractor	Forrest	Playstation	Sand
Icecream	Certificate	Grades	Stress	Love
Procrastination	Homework	Exam	Kebab	Noodles
Party	Music	Rock	Metal	Festival
Dancing	Chachacha	ClassLeader	SMV	InlineSkating
Beer	Screaming	HighSchool	Newspaper	Science
biotechnology	Food	Chemistry	Ethics	Popularity
Disco	Motorcycle	Park	Foodstand	Rollercoaste
HTML	Money	Photoshop	Youtube	Myvideo
AnimeMusicVideo	Graduation	Swimming	Farm	Marathon
Energy	Trapolin	Salto	LanParty	Warcraft3
Guildwars	Dota	HungryHungryFellhound	Electricity	Volt
Presentation	CoffeeAutomat	Latex	Lyrics	PianoConce
Suits	Protocol	Paragliding	Sleepovers	Couches
Smart	McDonalds	Boxing	University	Apprentices
Movies	Cinema	Apprenctiship	Shoes	Insoles
InsuranceCompany	Quote	Hammer	Clue	Plastic
Leather	Processing	Renting	Privilege	Confrences
Group	Mentor	Hierarchry	Private	Work

Words				
Comuncation	Professors	Biomechanik	Walking	Styles
Diabetes	Doctors	Recepetion	Notes	Patience
Sorting	Finding	Cards	AfterSells	Documenta
University				

Sidethought

1. Kid
2. School
3. Middle School
4. High School
5. Apprenticeship
6. University
7. Industry
8. Games
9. Web
10. Blockchain

10 Sentences

Word	Theme	Phrase	Rating
Tractor	Decentralised Farming	John Dear on chain	7
Slide	Futuruma Post on Chain		6
Earth	Supply Tracking	Your Construction your Materials	5
Blocks	Blockexplorer	Its a block, a real block	4
Indianer	Heritage Finder	Find your past	4
Mouse	Immutable Input Devices	Your Input in your wallet	3
Grammar	Decentralised Computation	Check your Text	3
Writing	Decentralised Writing	Make your letter count count	2
Math	Zero Knowledge	Hide your secrets	2
Straw	Food Supply	Ethical Food to your door	2

Select: Tractor - Decentralised Farming - John Dear on Chain

📝 Sketch

Idea

Problem: Food Production is decapsulated from the consumers

Idea: `tractor.xyz`

Solution: Bring automatised Production on chain, make it available

Requirements

Functional Requirements

- The Smart Contract must pay for the Tractor
- The Smart Contract must be fundable
- The App must allow Users to fund
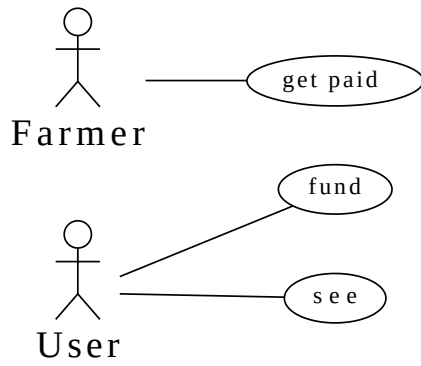- The App must display current Tractors

Non Functional Requirements

- It should be law compliant
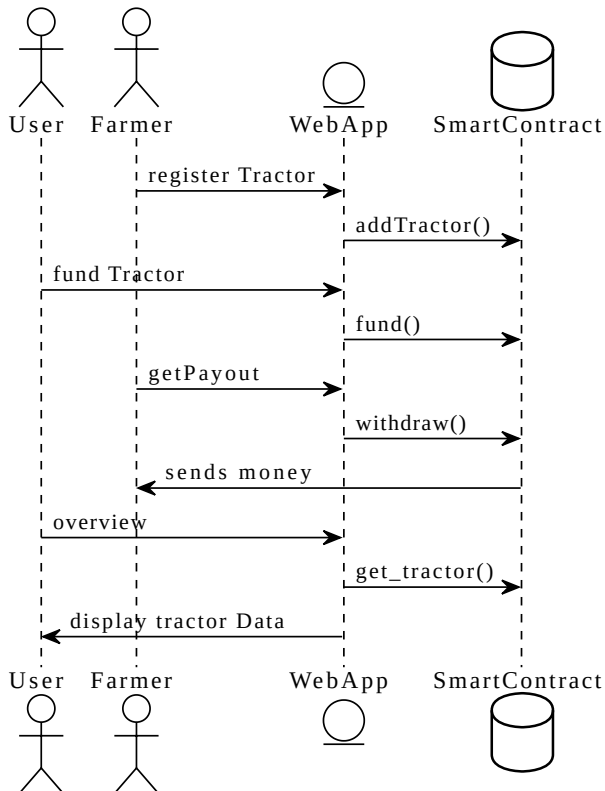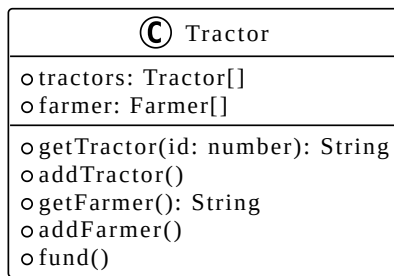
User Stories

- As a User I want to fund my food production
- As a User I want to see where is my funds are going
- As a Farmer I want that my production material is paid

Diagrams

Use Case

Farmer — get paid

User — fund, see

Class

```
┌─────────────────────────────────┐
│        Ⓒ  Tractor               │
├─────────────────────────────────┤
│ ○ tractors: Tractor[]           │
│ ○ farmer: Farmer[]              │
├─────────────────────────────────┤
│ ○ getTractor(id: number): String│
│ ○ addTractor()                  │
│ ○ getFarmer(): String           │
│ ○ addFarmer()                   │
│ ○ fund()                        │
└─────────────────────────────────┘
```

User   Farmer   WebApp   SmartContract

Farmer → WebApp: register Tractor

WebApp → SmartContract: addTractor()

User → WebApp: fund Tractor

WebApp → SmartContract: fund()

Farmer → WebApp: getPayout

WebApp → SmartContract: withdraw()

SmartContract → Farmer: sends money

User → WebApp: overview

WebApp → SmartContract: get_tractor()

WebApp → User: display tractor Data

User   Farmer   WebApp   SmartContract

The End.

I call it a day. Not the best run.