# 11-Ideas

## Overview

- Intro
- Think
  - 100 Words
  - 10 Sentences
  - Rate & Select
- Sketch
  - Idea
  - Requirments
  - Stories
  - Diagrams
- Paint
  - Write
  - Build
  - Call
- Outro

## 🧠 Think

### 100 Words

- rich pictures

Perls, Chair, Haircut, Lipstick, Technican,
Mandala, Hancok, OnePiece, Sarah, Amsterdam,
Babej, Seoul, Blender, Beer, Bike,
Coffee, Braclet, Tooth, Thailand, Smile,

### 10 Sentences

| Theme | Word | Sentence | Rating |
|---|---|---|---|
| Roadtrip | Bike | AI for you roadtrip | 8 |
| Growth | Thailand | Growth thought diffrent | 4 |
| AfterSells | Haircut | Your Hair, your product | 4 |
| Experience | OnePiece | Manga on steroids | 3 |

### Rate & Select

The winner is `roady.xyz`

## 📝 Sketch

### Idea

Problem: Giving Instruction on live events is tricky,especially on roadtrips
Idea: Write an AI which answers it correctly
Solution: `roady.xyz`

### Requirments

#### Functional Requirments

- The App must do generate an AI to generate a Live Video

#### Non functional Requirments

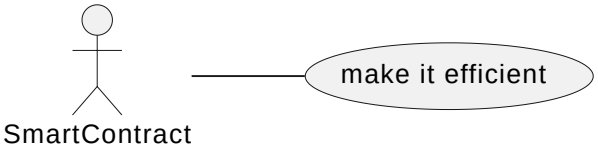
- The App must be profitable

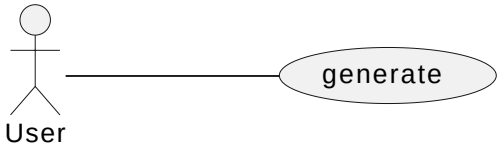
### User Stories

- As a User I want to generate an AI which give my instructions
- As a SmartContract I want remove the centralised entity, I handle it.

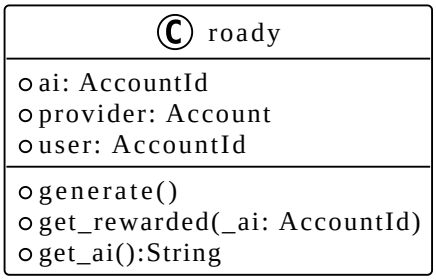
## Diagrams

### Use Case Diagram

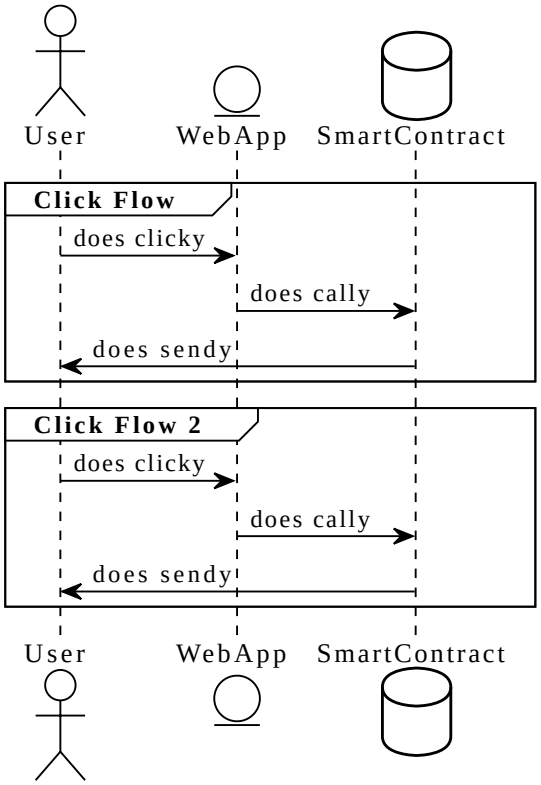



### Class Diagram




### Sequence Diagram



# 🎨 Paint

```
#![cfg_attr(not(feature = "std"), no_std, no_main)]

#[ink::contract]
mod roady {
    use ink::prelude::string::String;

    #[ink(storage)]
    pub struct Roady {
        value: bool,
        user: AccountId,
        ai: String,
    }

    impl Roady {
        #[ink(constructor)]
        pub fn new() -> Self {
            Self {
                value: true,
                user: AccountId::from([0xff; 32]),
                ai: String::from("no great link to wonderful ai"),
            }
        }

        #[ink(message)]
        pub fn get_ai(&self) -> String {
            self.ai.clone()
        }

        #[ink(message)]
        pub fn generate(&mut self) {
            self.ai = String::from("link to great ai");
        }

        #[ink(message)]
        pub fn who_is_the_owner(&self) -> AccountId {
            self.user
        }

        #[ink(message)]
        pub fn flip(&mut self) {
            self.value = !self.value;
        }

        #[ink(message)]
        pub fn get(&self) -> bool {
            self.value
        }
    }
```

```
}
```