TEAMAGOCHI - てぃーまごっち

REVISED ARCHITECTURE PRESENTATION

VIRTUAL RIOT PET BY THE RIOT PROJECT SOSE24 TEAM

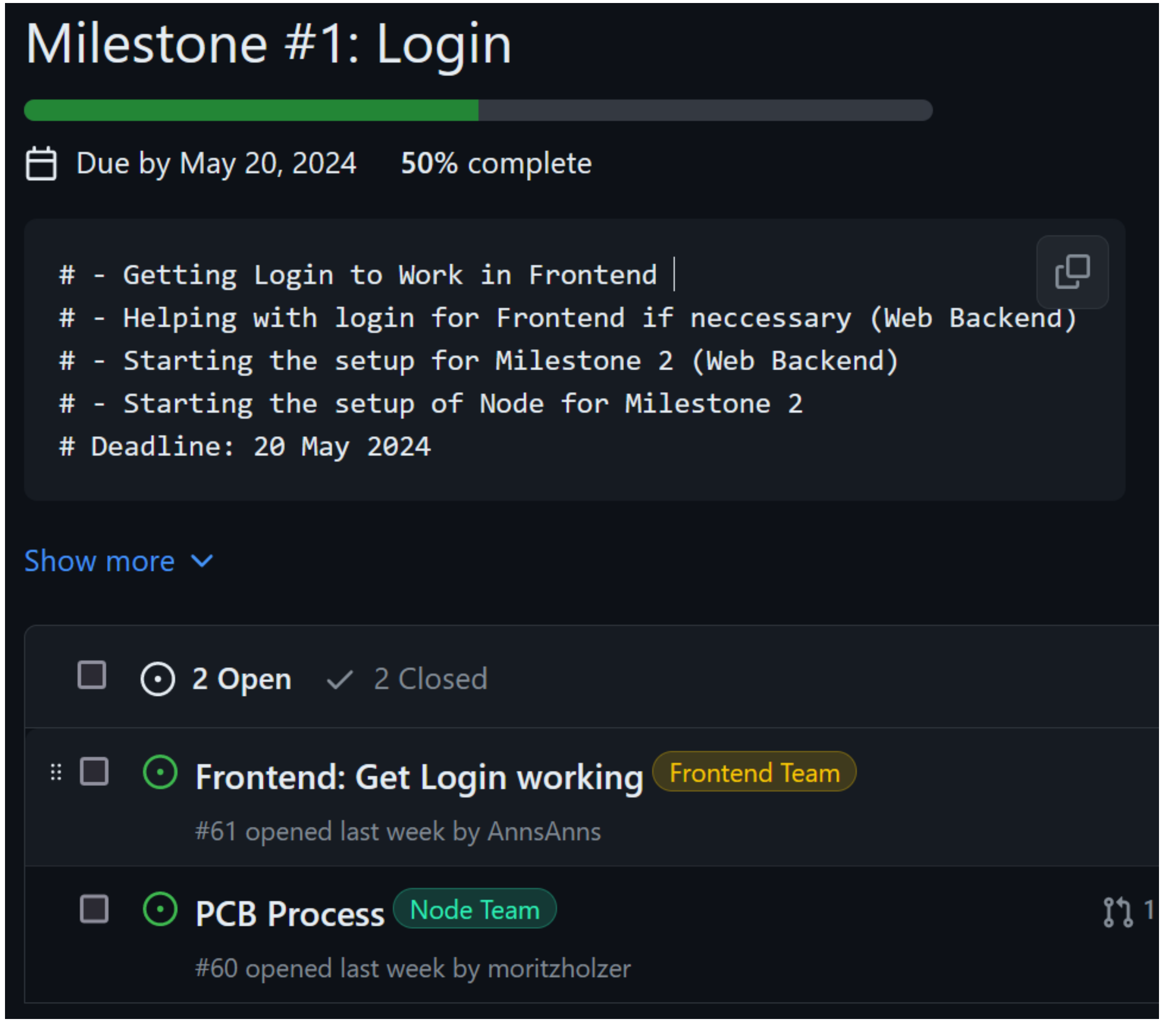
OVERVIEW (ORGANIZATION & CI & RECAP)

- Tom Hert

RECAP OF IDEA

A synchronized always-online pet simulator with multiplayer functionality

SHORT-TERM MILESTONES



Milestone #1: Login

Due by May 20, 2024 **50% complete**

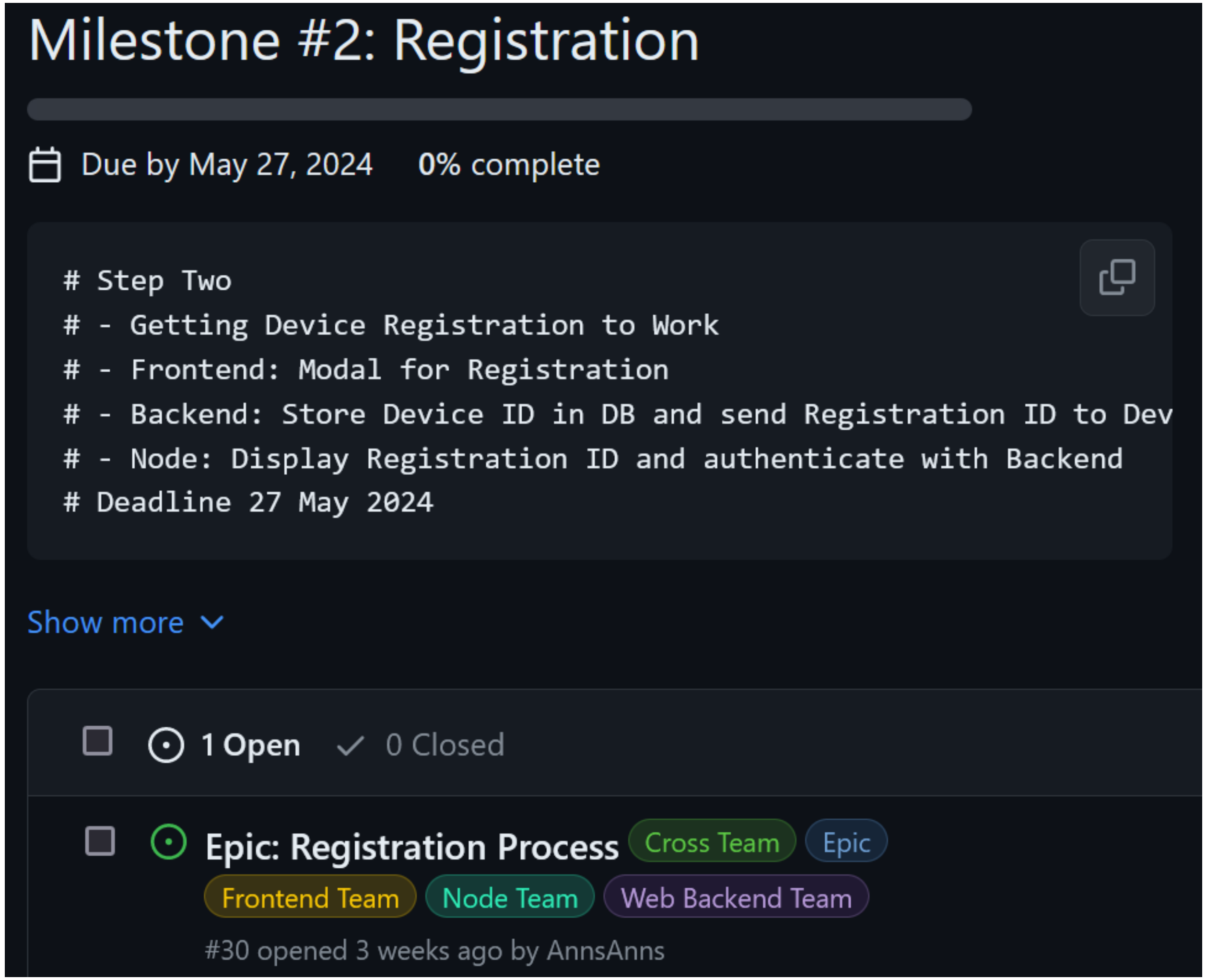
```
# - Getting Login to Work in Frontend
# - Helping with login for Frontend if neccessary (Web Backend)
# - Starting the setup for Milestone 2 (Web Backend)
# - Starting the setup of Node for Milestone 2
# Deadline: 20 May 2024
```

Show more

□ ○ **2 Open** ✓ 2 Closed

□ ○ **Frontend: Get Login working** Frontend Team
#61 opened last week by AnnsAnns

□ ○ **PCB Process** Node Team 1
#60 opened last week by moritzholzer



Milestone #2: Registration

Due by May 27, 2024 **0% complete**

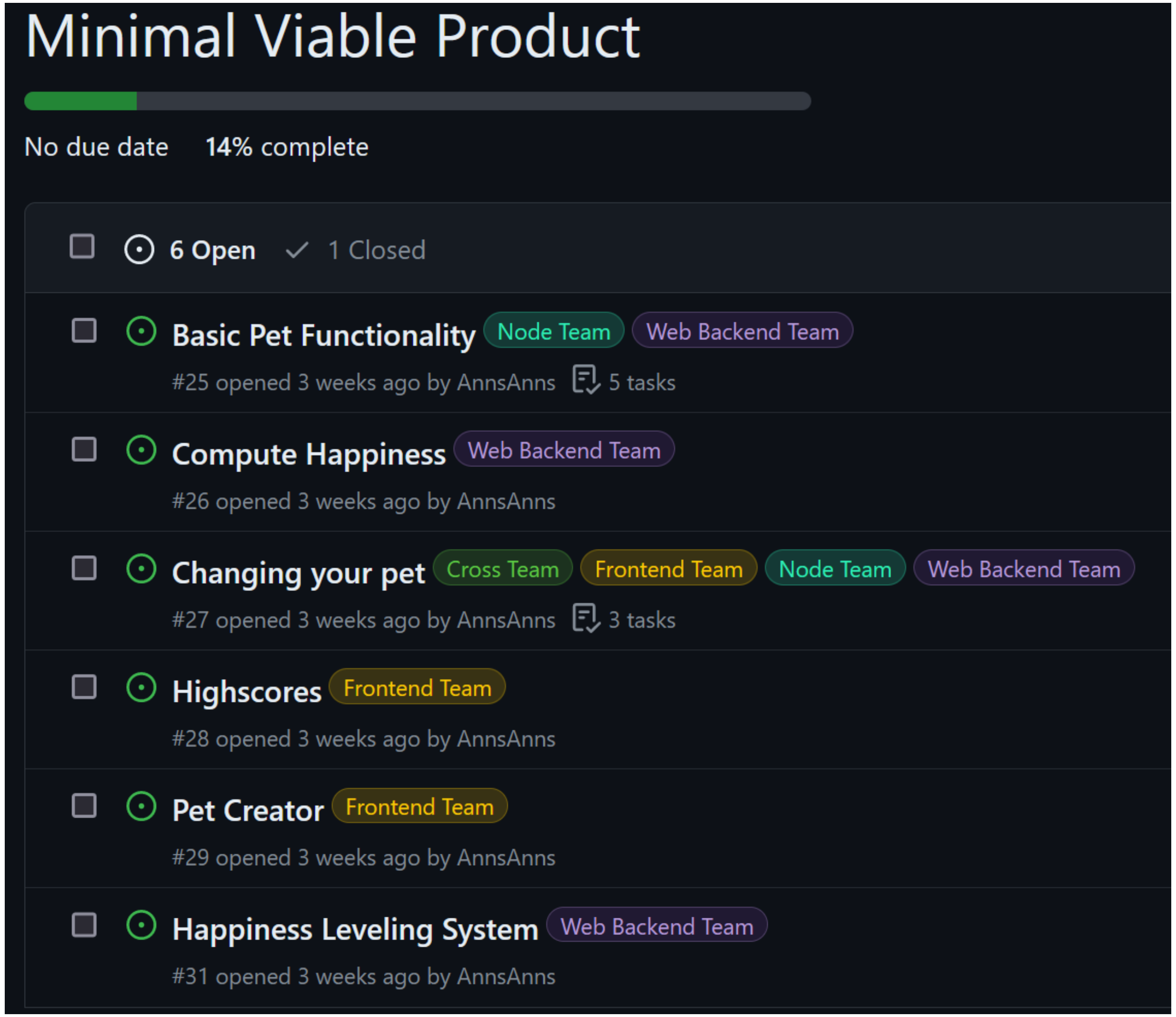
```
# Step Two
# - Getting Device Registration to Work
# - Frontend: Modal for Registration
# - Backend: Store Device ID in DB and send Registration ID to Dev
# - Node: Display Registration ID and authenticate with Backend
# Deadline 27 May 2024
```

Show more

□ ○ **1 Open** ✓ 0 Closed

□ ○ **Epic: Registration Process** Cross Team Epic Frontend Team Node Team Web Backend Team
#30 opened 3 weeks ago by AnnsAnns

LONG-TERM MILESTONE



Minimal Viable Product

No due date 14% complete

□ ⊙ 6 Open ✓ 1 Closed

□ ⊙ Basic Pet Functionality Node Team Web Backend Team
#25 opened 3 weeks ago by AnnsAnns 5 tasks

□ ⊙ Compute Happiness Web Backend Team
#26 opened 3 weeks ago by AnnsAnns

□ ⊙ Changing your pet Cross Team Frontend Team Node Team Web Backend Team
#27 opened 3 weeks ago by AnnsAnns 3 tasks

□ ⊙ Highscores Frontend Team
#28 opened 3 weeks ago by AnnsAnns

□ ⊙ Pet Creator Frontend Team
#29 opened 3 weeks ago by AnnsAnns

□ ⊙ Happiness Leveling System Web Backend Team
#31 opened 3 weeks ago by AnnsAnns

Our Vision for something cool enough for the Presentation

All these features can then be polished, enhanced and extended

„Feature Completeness“

CI WORKFLOW(S)

- Frontend (Currently Vite+TS+ReactJS) gets build automatically and deployed to https://smartuni.github.io/teamagochi
- Docs (Astro Starlight) gets build automatically and deployed to https://smartuni.github.io/teamagochi/docs/
- Node (Riot Project) gets build to verify that it at least compiles
- Currently In Dev: Node CI in a private repo starts tests via cronjob

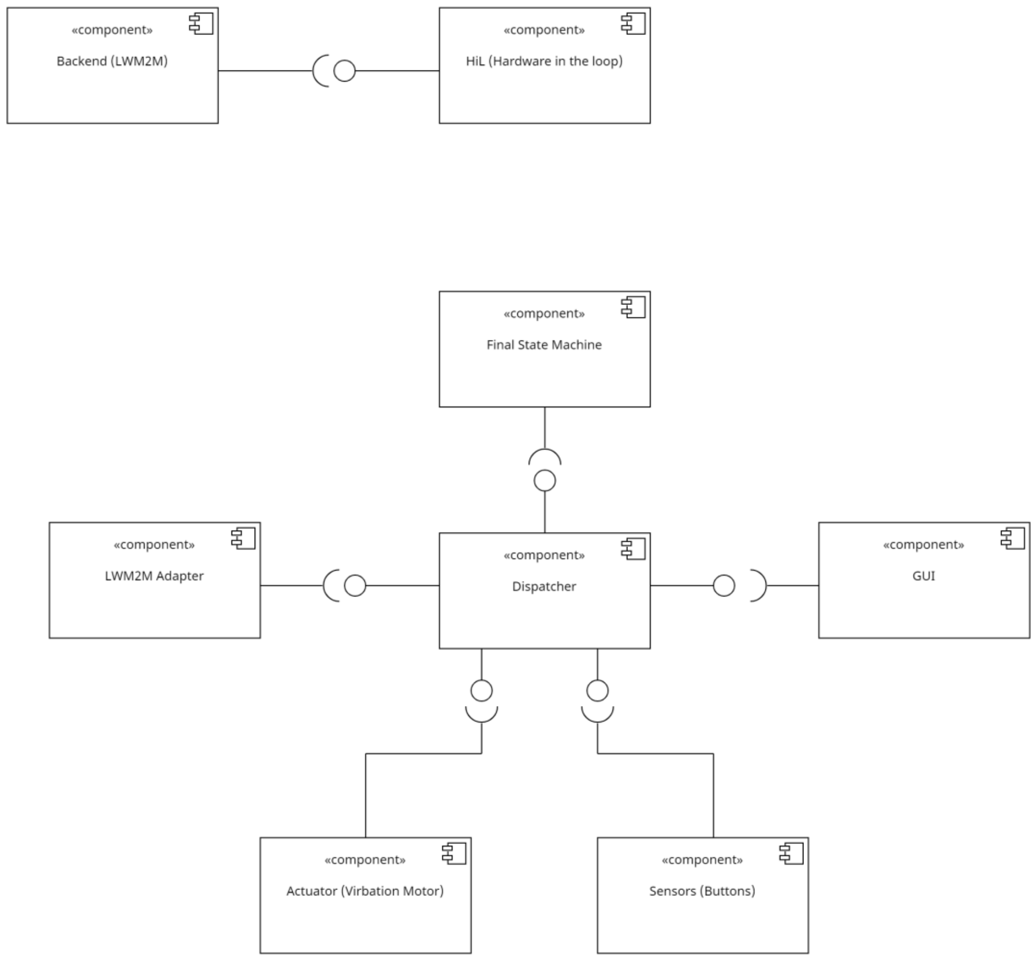
DEVELOPMENT WORKFLOW

1. Create a feature branch
2. Commit your feature with meaningful commit messages
3. Open a Pull Request
4. Get somebody to review your branch
5. (Hopefully) get your Pull Request approved
6. Merge into Main
7. Success

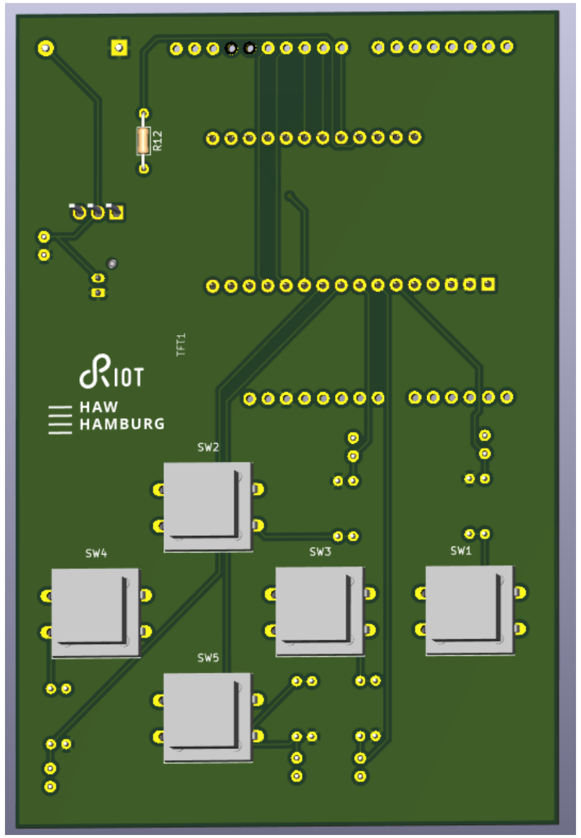
NODES TEAM

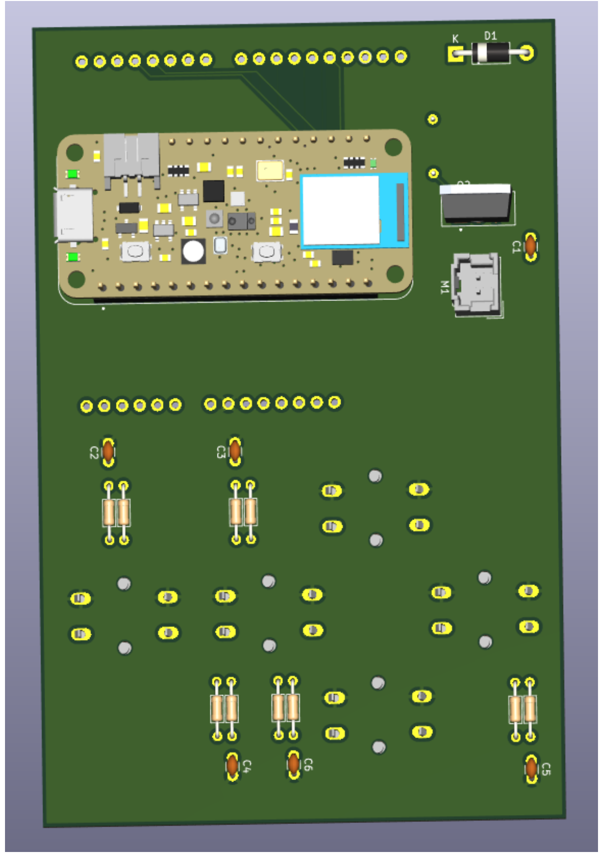
- Eduard Lomtadze
- Nils Voepel
- Moritz Holzer
- Dong Yuanzhe
- Lukas Sebrantke
- Tom Hert
- Justin Sanker

FIRST COMPONENT DIAGRAM



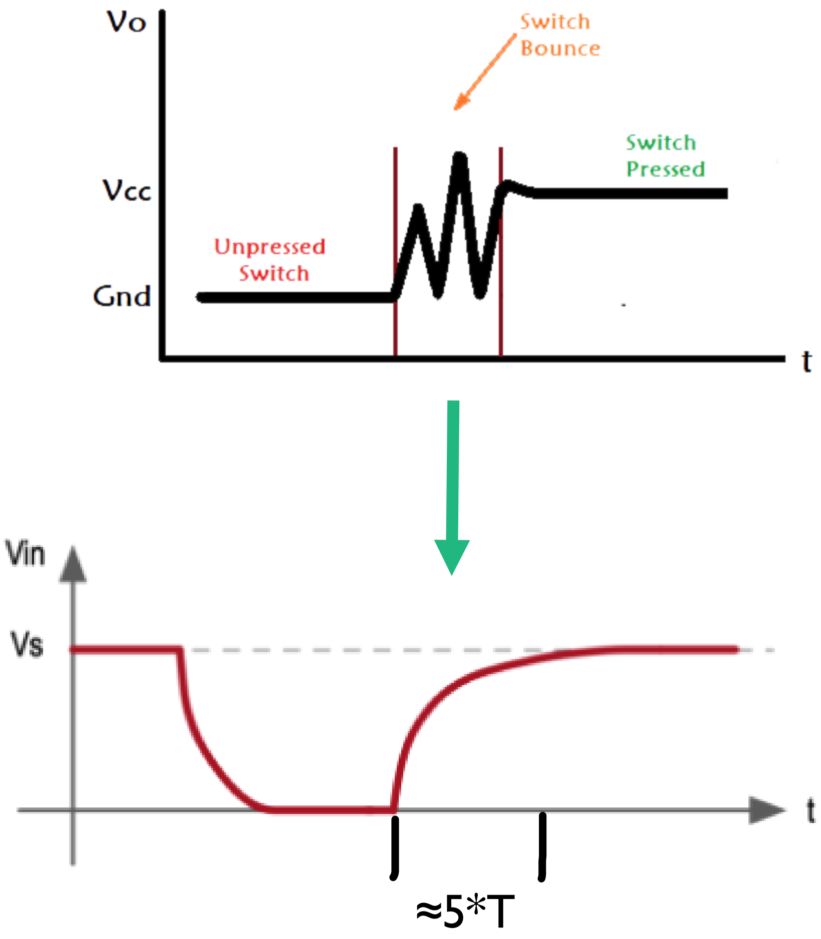
FINAL PCB


R12
TFT1
RIOT
HAW
HAMBURG
SW2
SW4
SW3
SW1
SW5

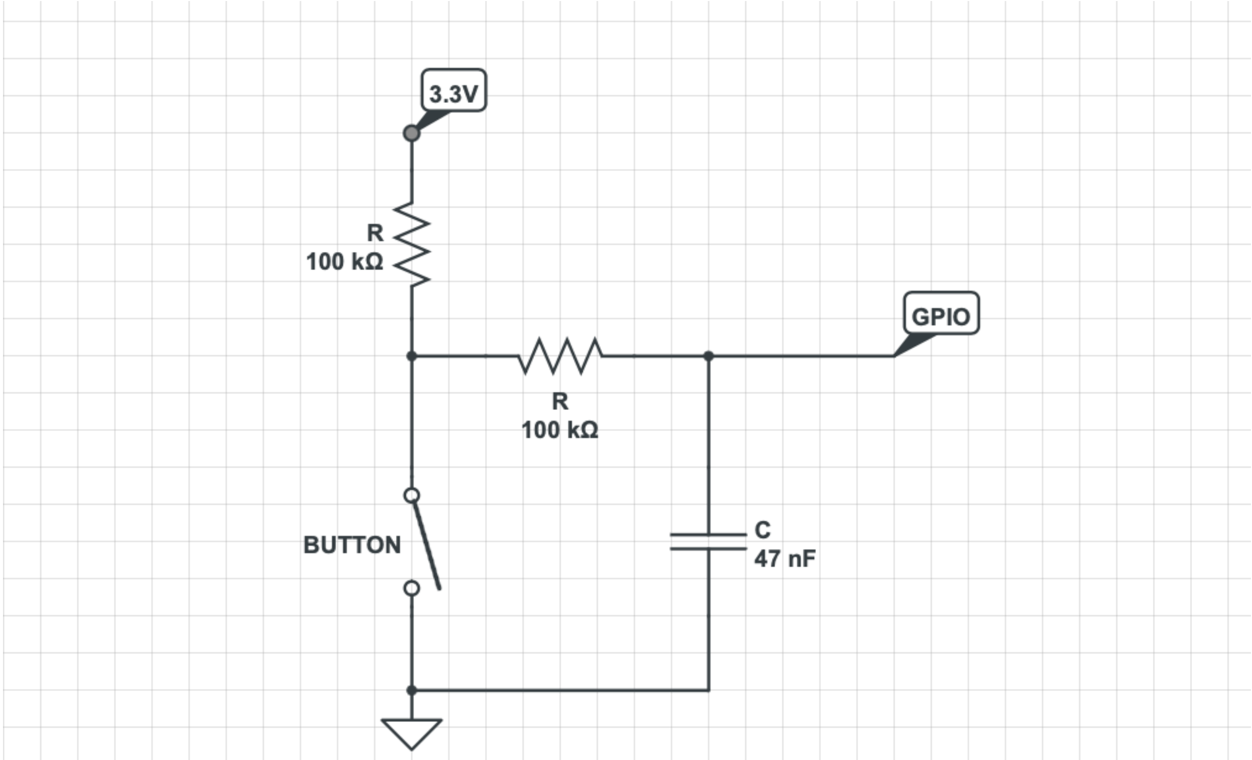

K
D1
C1
M1
C2
C3
C4
C6
C5

BUTTON IMPLEMENTATION

GOAL:



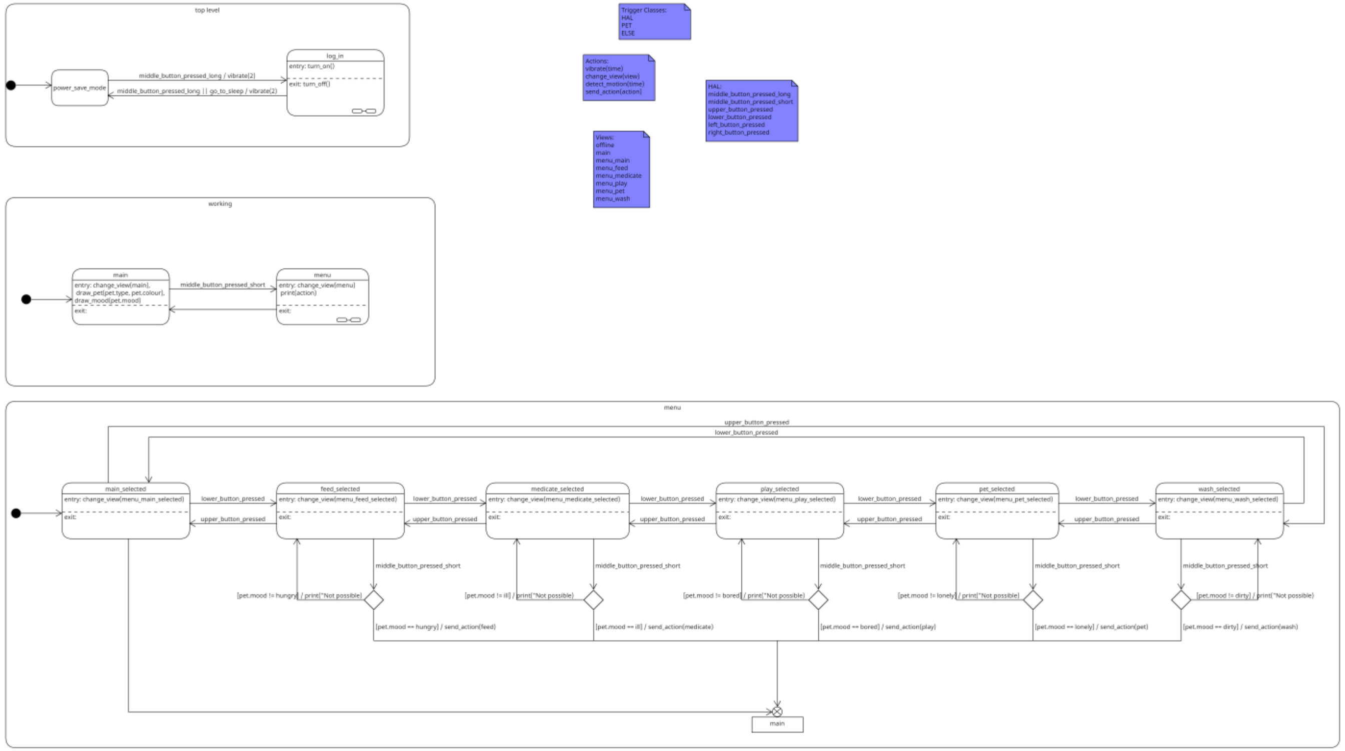
Low-pass filter for button debouncing:



T = R*C = 100kΩ*47nF = 4,7ms

FIRST FINAL STATE MACHINE

- Only for the MVP
- 3 layers
- First menu

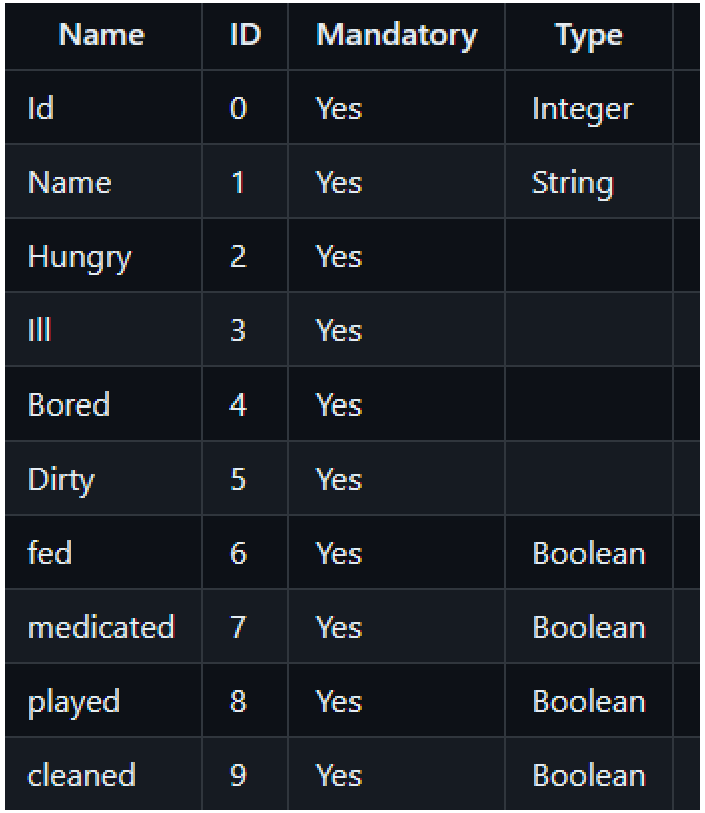

top level
power_save_mode
middle_button_pressed_long / vibrate(2)
middle_button_pressed_long || go_to_sleep / vibrate(2)
log_in
entry: turn_on()
exit: turn_off()
Trigger Classes:
HAL
PET
ELSE
Actions:
vibrate(time)
change_view(view)
detect_motion(time)
send_action(action)
HAL:
middle_button_pressed_long
middle_button_pressed_short
upper_button_pressed
lower_button_pressed
left_button_pressed
right_button_pressed
Views:
offline
main
menu_main
menu_feed
menu_medicate
menu_play
menu_pet
menu_wash
working
main
entry: change_view(main),
draw_pet(pet.type, pet.colour),
draw_mood(pet.mood)
exit:
middle_button_pressed_short
menu
entry: change_view(menu)
print(action)
exit:
menu
upper_button_pressed
lower_button_pressed
main_selected
entry: change_view(menu_main_selected)
exit:
lower_button_pressed
upper_button_pressed
feed_selected
entry: change_view(menu_feed_selected)
exit:
lower_button_pressed
upper_button_pressed
medicate_selected
entry: change_view(menu_medicate_selected)
exit:
lower_button_pressed
upper_button_pressed
play_selected
entry: change_view(menu_play_selected)
exit:
lower_button_pressed
upper_button_pressed
pet_selected
entry: change_view(menu_pet_selected)
exit:
lower_button_pressed
upper_button_pressed
wash_selected
entry: change_view(menu_wash_selected)
exit:
middle_button_pressed_short
[pet.mood != hungry] / print("Not possible)
[pet.mood == hungry] / send_action(feed)
middle_button_pressed_short
[pet.mood != ill] / print("Not possible)
[pet.mood == ill] / send_action(medicate)
middle_button_pressed_short
[pet.mood != bored] / print("Not possible)
[pet.mood == bored] / send_action(play)
middle_button_pressed_short
[pet.mood != lonely] / print("Not possible)
[pet.mood == lonely] / send_action(pet)
middle_button_pressed_short
[pet.mood != dirty] / print("Not possible)
[pet.mood == dirty] / send_action(wash)
main

LWM2M OBJECT

Pet Object

Hungry, Ill, Bored and Dirty are executed

The boolean variables get updated from backend and node

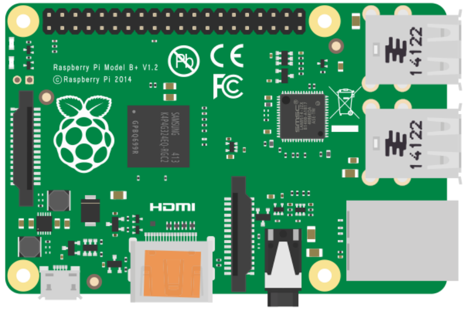


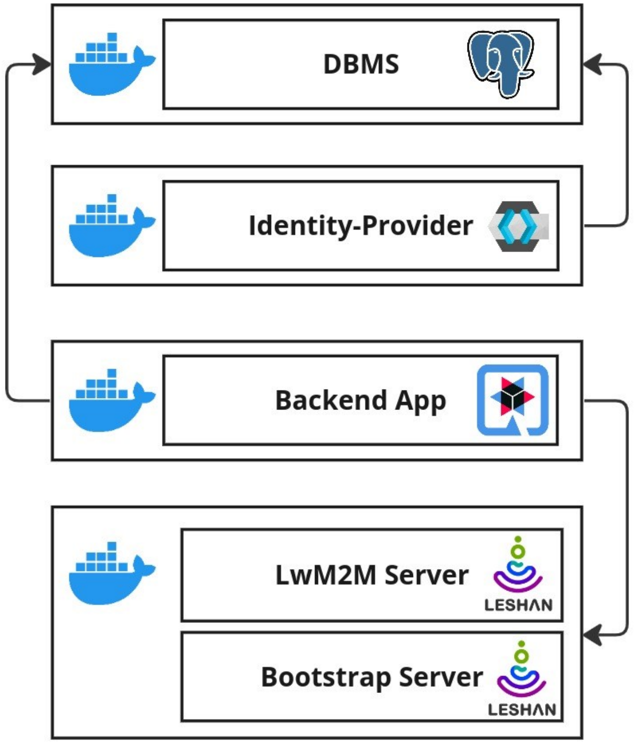
Name	ID	Mandatory	Type
Id	0	Yes	Integer
Name	1	Yes	String
Hungry	2	Yes	
Ill	3	Yes	
Bored	4	Yes	
Dirty	5	Yes	
fed	6	Yes	Boolean
medicated	7	Yes	Boolean
played	8	Yes	Boolean
cleaned	9	Yes	Boolean

WEB BACKEND TEAM

- Merlin Trefflich
- Leo Graf
- Jessica Broese
- Van Khoi Pham

VERTEILUNGSSICHT





Additional services:

- Reverse-Proxy
- Fake SMTP-Server
- Development Dashboard

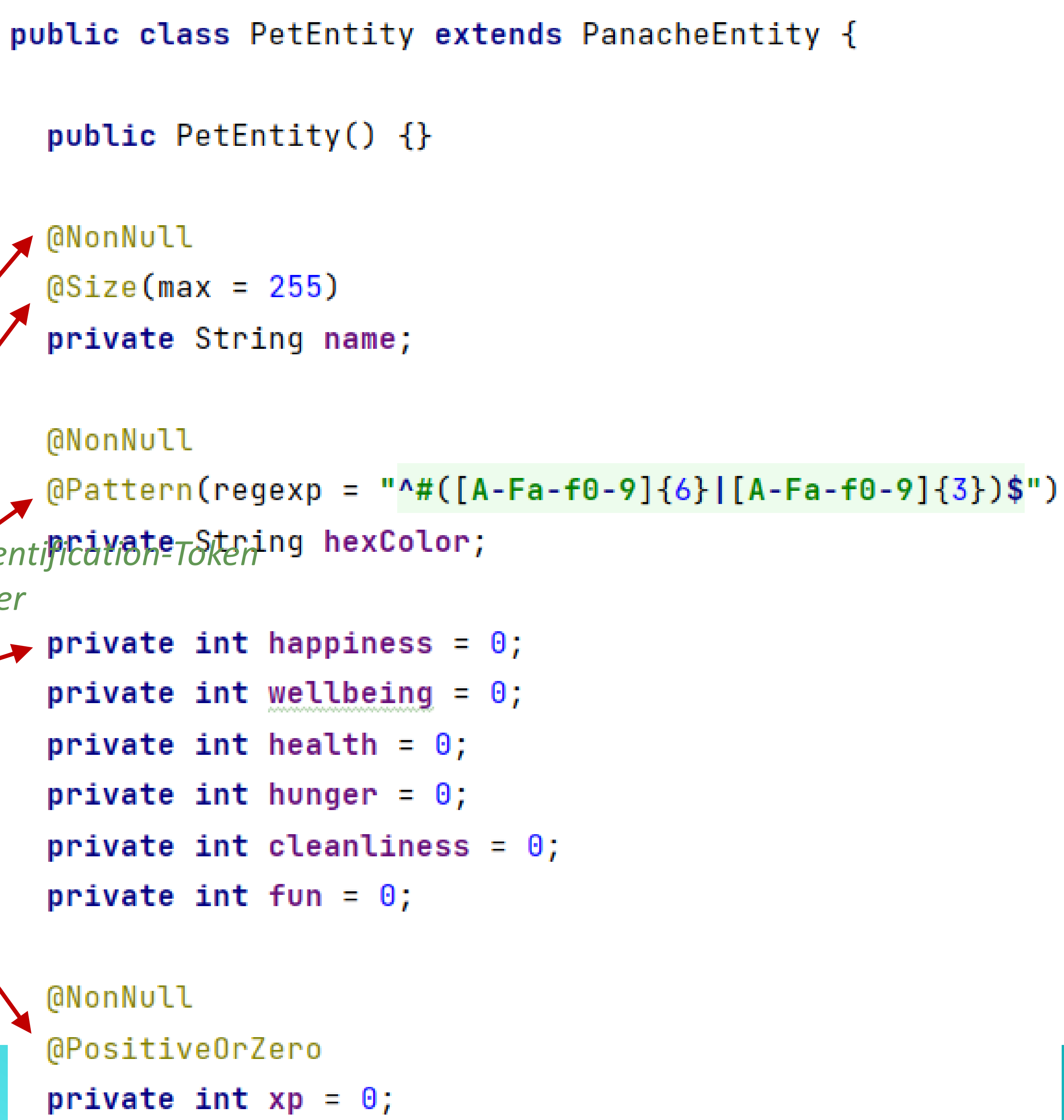
DATABASE ENTITIES

Attributes (of a pet e.g.):

- may be Null?
- may be how long/big?
- must follow a pattern?
- have a start value?
- must be in a value range?

Source classes:
PetEntity, PetTypeEntity, UserEntity, DeviceEntity

the Authentification-Token of the User

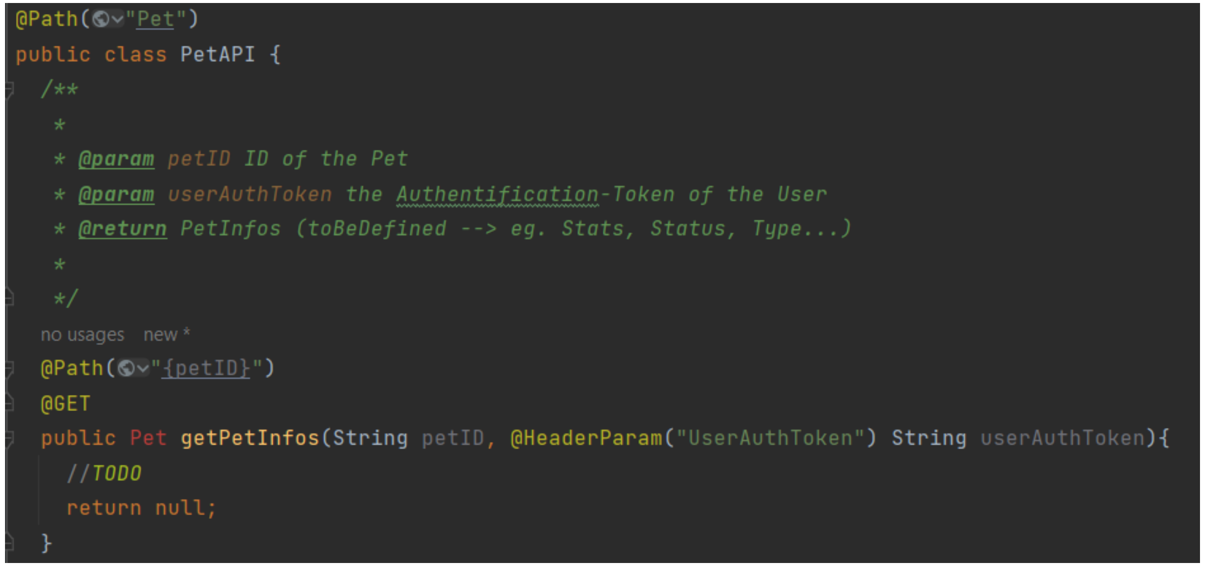


```
public class PetEntity extends PanacheEntity {

    public PetEntity() {}

    @NonNull
    @Size(max = 255)
    private String name;

    @NonNull
    @Pattern(regexp = "^#([A-Fa-f0-9]{6}|[A-Fa-f0-9]{3})$")
    private String hexColor;

    private int happiness = 0;
    private int wellbeing = 0;
    private int health = 0;
    private int hunger = 0;
    private int cleanliness = 0;
    private int fun = 0;

    @NonNull
    @PositiveOrZero
    private int xp = 0;
```

FRONTEND-API

API-Excerpt:

- Devided for future-proving: Device and Pet-API
- User-Verification required
- Return-Types need to be suitable for the Frontend-Team
- Implementation-Order in accordance to the milestones

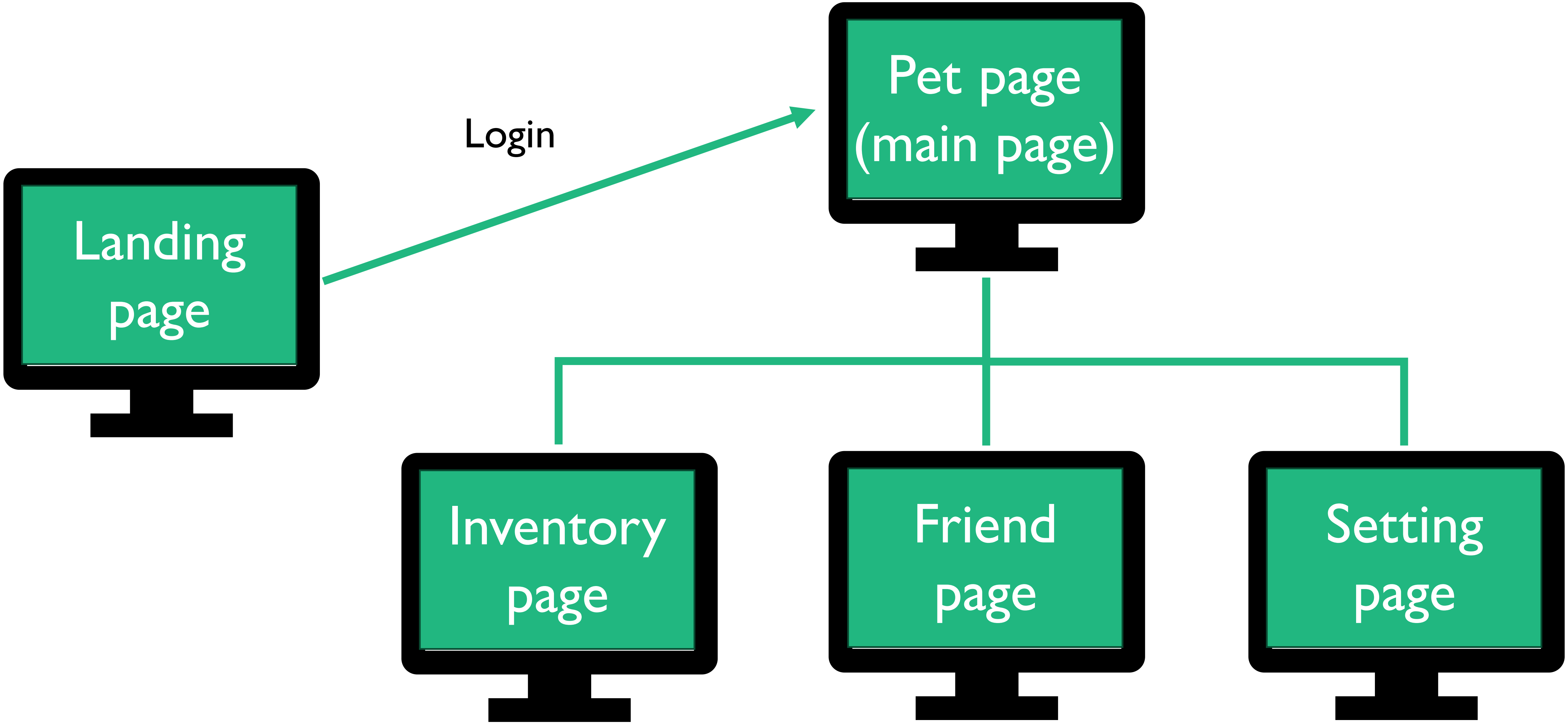


```java
@Path("Pet")
public class PetAPI {
  /**
   *
   * @param petID ID of the Pet
   * @param userAuthToken the Authentification-Token of the User
   * @return PetInfos (toBeDefined --> eg. Stats, Status, Type...)
   *
   */
  no usages   new *
  @Path("{petID}")
  @GET
  public Pet getPetInfos(String petID, @HeaderParam("UserAuthToken") String userAuthToken){
    //TODO
    return null;
  }
```

FRONTEND TEAM

- Samuel Costa
- Rares Stefea
- Yousef Taha
- Hamdy Elmorsy
- Yasuaki Kumazaki
- Omar Shaban

SCREEN TRANSITION DIAGRAM



Landing page

Login

Pet page (main page)

Inventory page

Friend page

Setting page

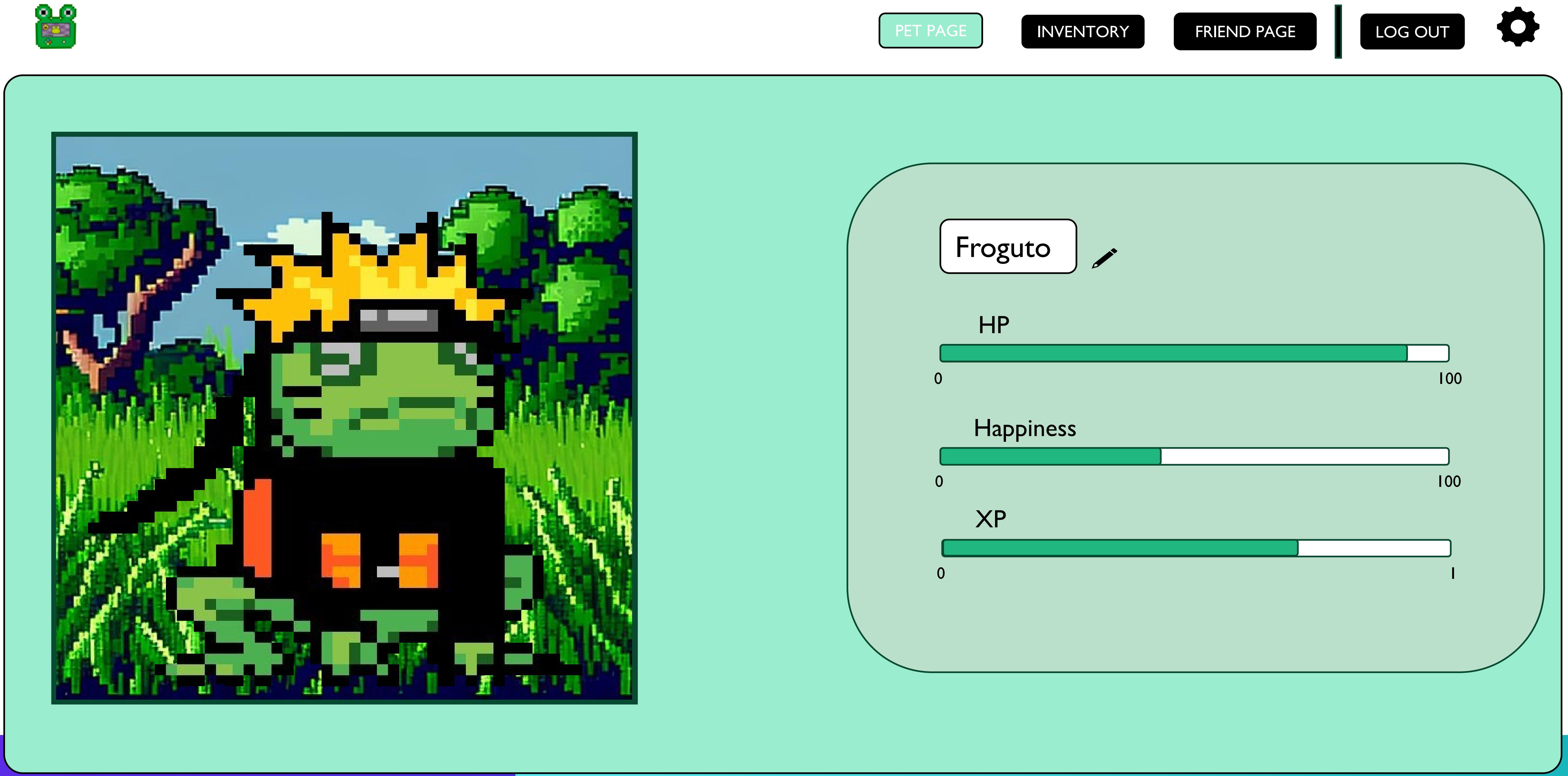
LANDING PAGE

SIGN IN

TEAMAGOCHI

(Introduction about the game)

PET PAGE



Froguto

HP

0 100

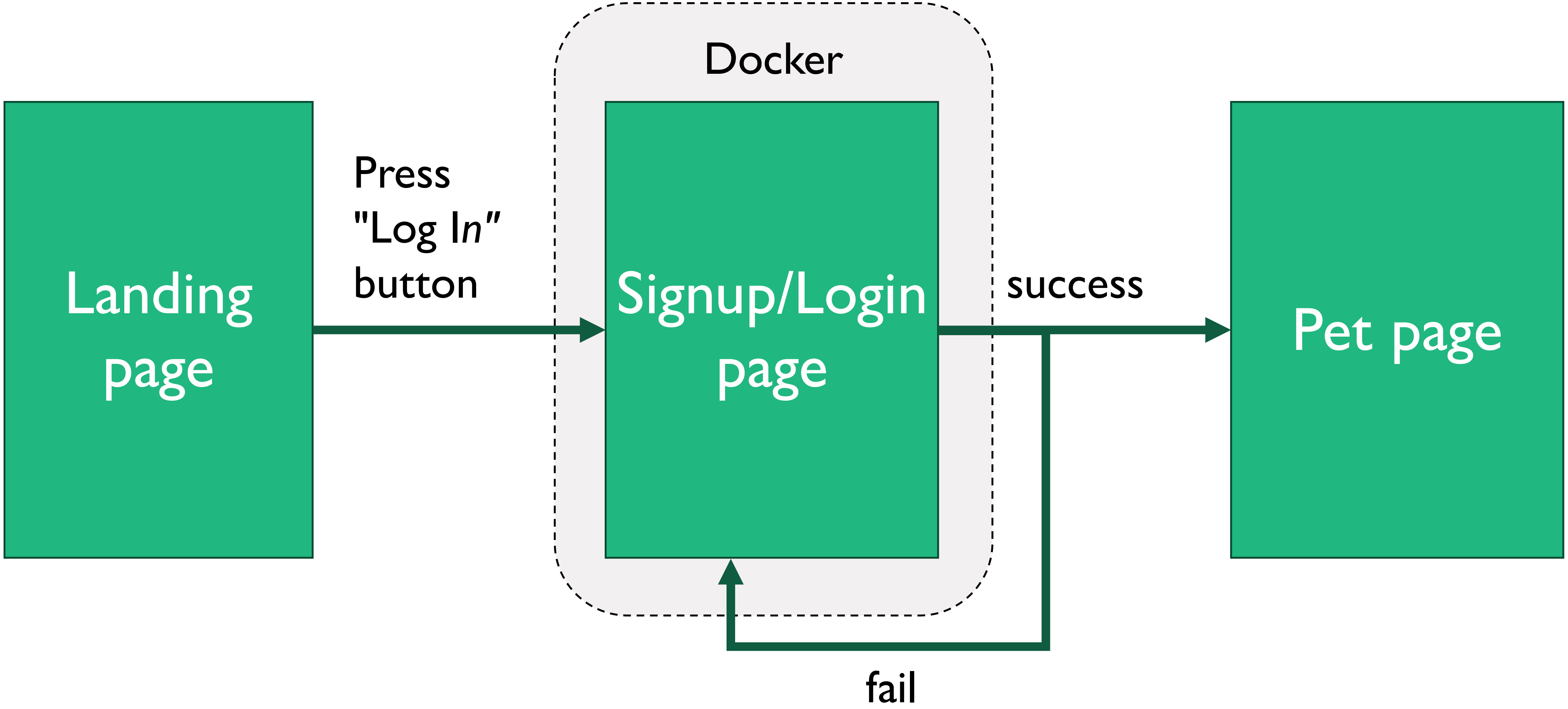
Happiness

0 100

XP

0 1

SIGNUP/LOGIN FUNCTION



THE END