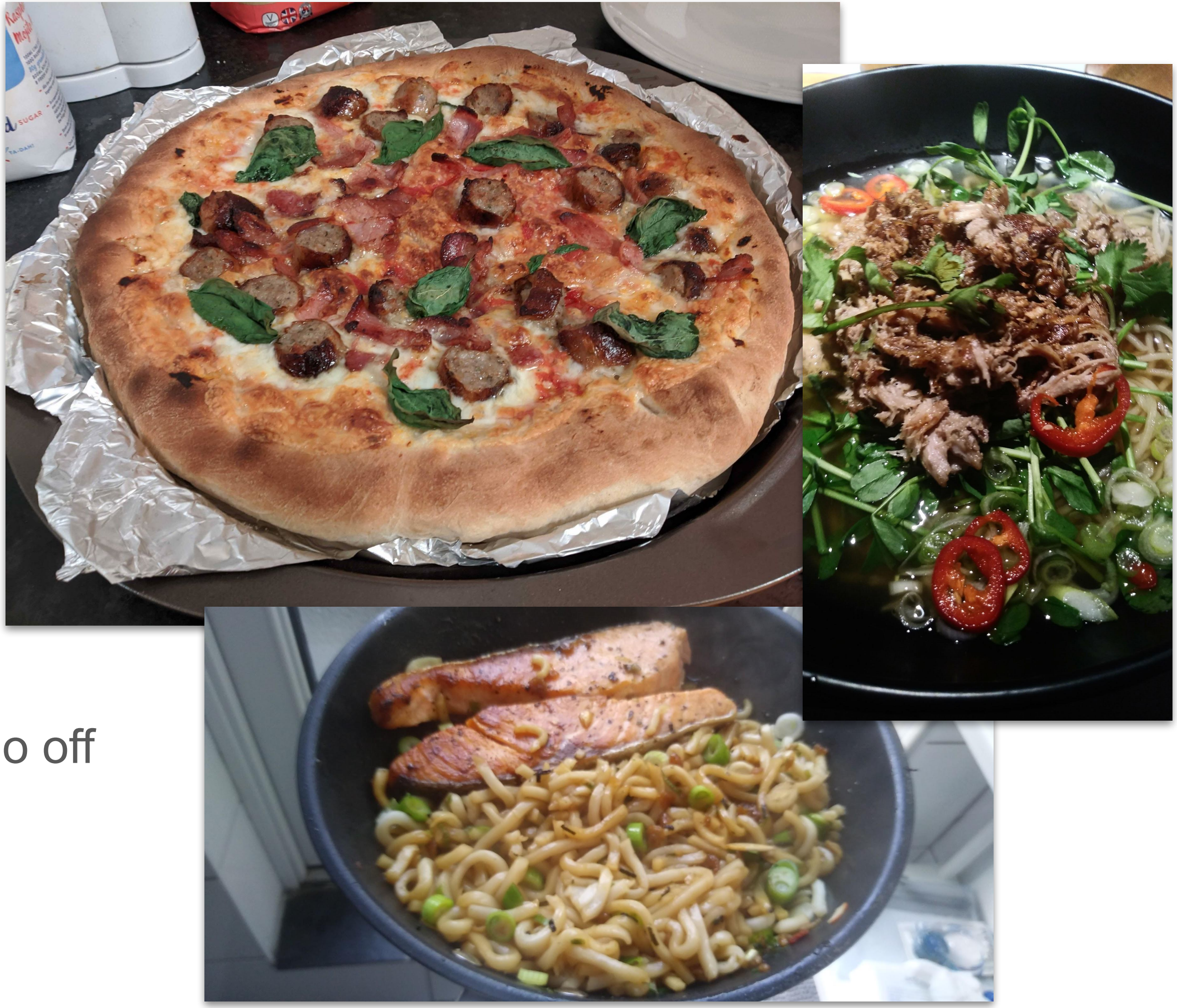
“Tell us about an achievement that you are particularly proud of.”

Cass

Cooking

Well, related…



One thing of annoyance...

Fresh herbs and spices!

Pre-packaged herbs tend to go off fairly quickly...

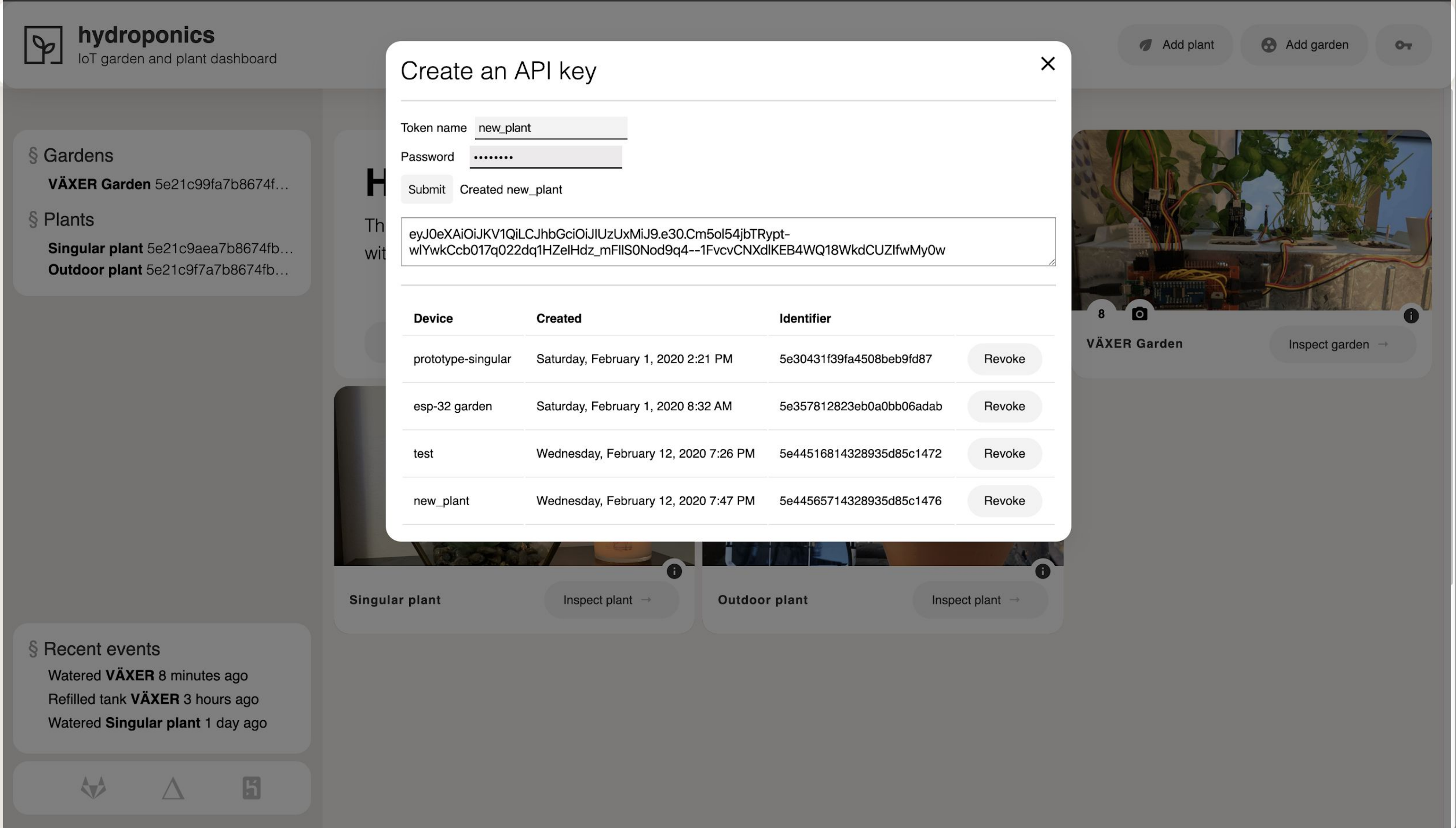
Issues with growing your own plants

- Outdoor pests
- Constrained by space of accomodation (no garden)
- Cooking frequently requires high yield farming
- Different herbs require different environments for optimal growth
  - Temperature, humidity, moisture, light levels, water pH etc...

Hydroponics!

“The process of growing plants in sand, gravel, or liquid, with added nutrients but without soil.”

….The simplest thing would’ve been to just put some plants in a pot and waited patiently



§ Gardens

VÄXER Garden 5e21c99fa7b8674f…

§ Plants

Singular plant 5e21c9aea7b8674fb…
Outdoor plant 5e21c9f7a7b8674fb…

§ Recent events

Watered **VÄXER** 8 minutes ago
Refilled tank **VÄXER** 3 hours ago
Watered **Singular plant** 1 day ago

Create an API key

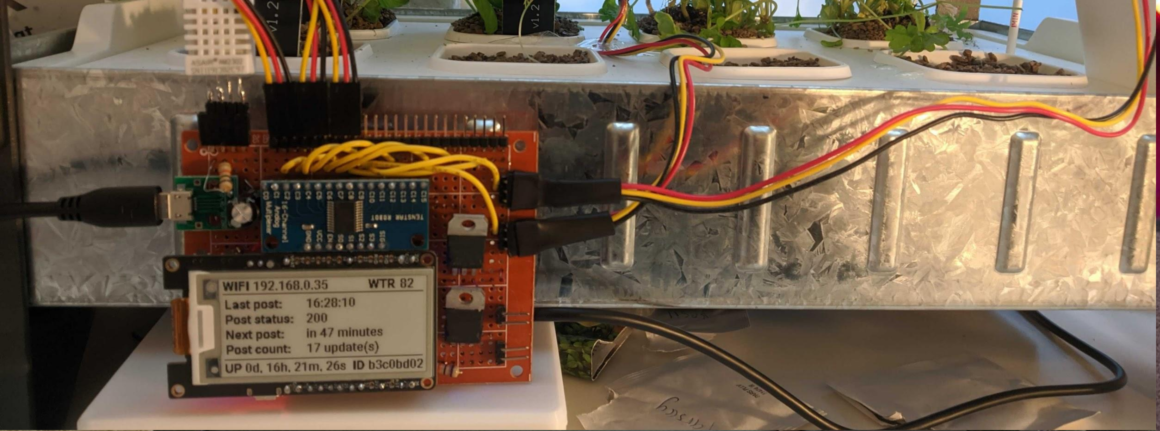
Token name new_plant

Password ••••••••

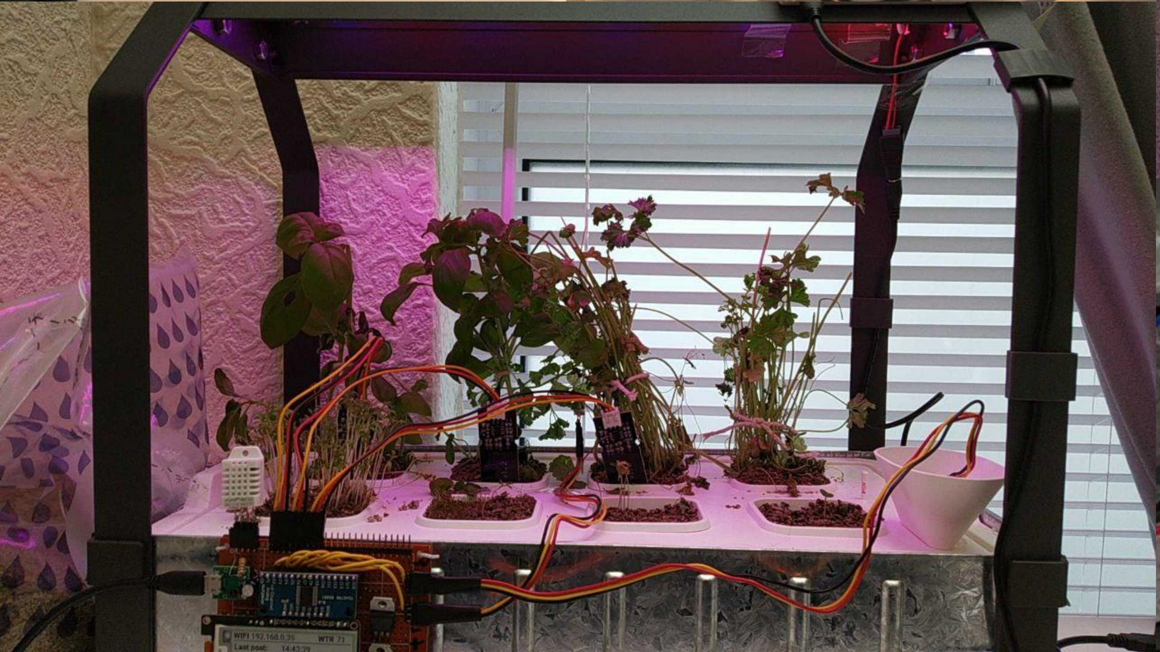
Submit Created new_plant

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.e30.Cm5ol54jbTRypt-
wlYwkCcb017q022dq1HZelHdz_mFIlS0Nod9q4--1FvcvCNXdlKEB4WQ18WkdCUZIfwMy0w
```

Device	Created	Identifier	
prototype-singular	Saturday, February 1, 2020 2:21 PM	5e30431f39fa4508beb9fd87	Revoke
esp-32 garden	Saturday, February 1, 2020 8:32 AM	5e357812823eb0a0bb06adab	Revoke
test	Wednesday, February 12, 2020 7:26 PM	5e44516814328935d85c1472	Revoke
new_plant	Wednesday, February 12, 2020 7:47 PM	5e44565714328935d85c1476	Revoke



```
WIFI 192.168.0.35            WTR 82
Last post:       16:28:10
Post status:     200
Next post:       in 47 minutes
Post count:      17 update(s)
UP 0d, 16h, 21m, 26s  ID b3c0bd02
```



```
WIFI 192.168.0.35            WTR 71
Last post:       14:42:39
```

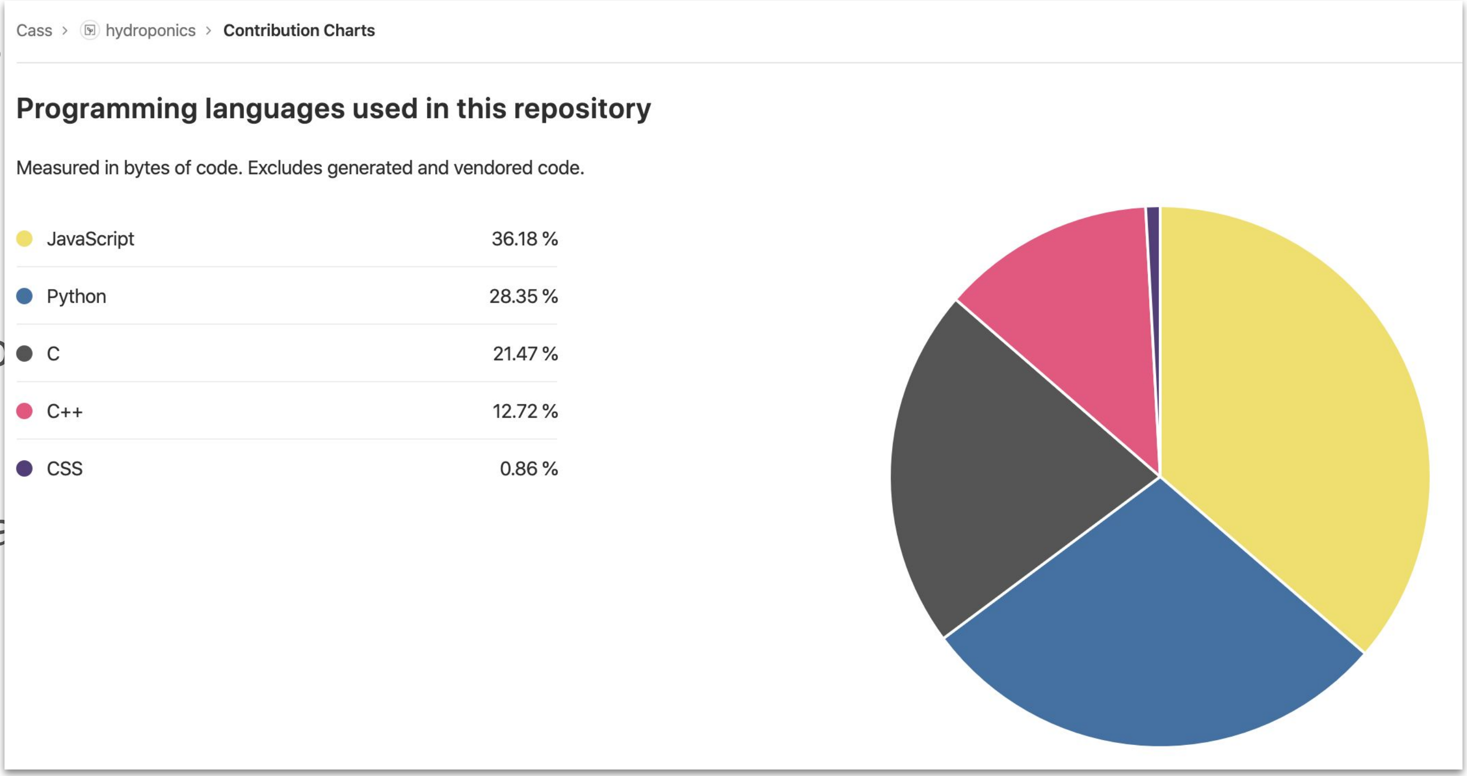
The plan

Internet of Things garden & plant management with dashboard.

- Gardens and Plants, manage entire gardens constituted of sub-plants
- Event based system notifies when an action needs completing / parameters out of range e.g. needs watering, moisture too low, humidity too high
- Remotely trigger events on micro-controllers, e.g. turn on light, open watering valve
- Monitor all metrics on front-end dashboard with graphs
- Code support for different hardware (ESP-32, Wemos D1 Mini & Raspberry Pi)

Architecture

- AP
- Fro
- Ha

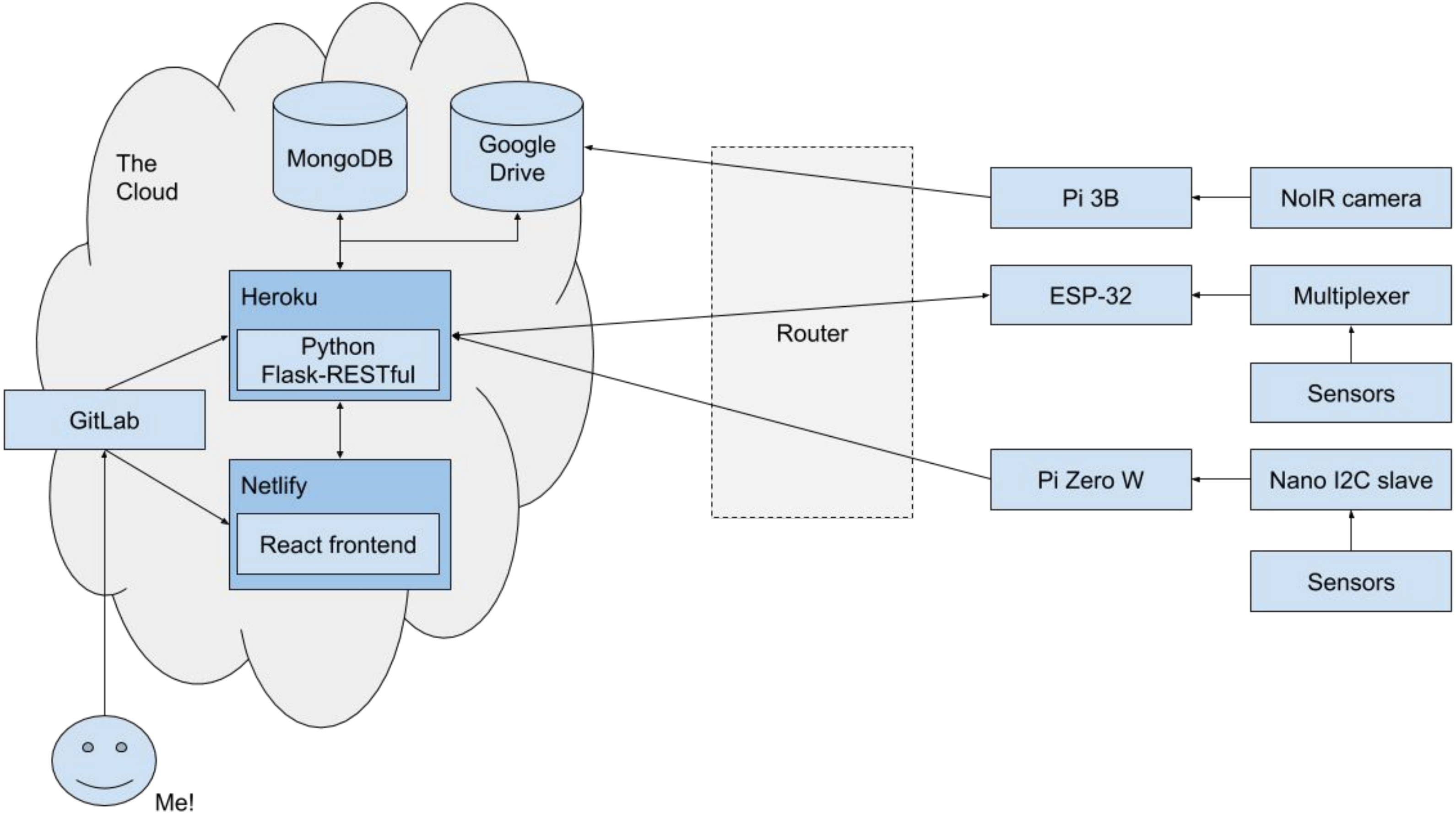


Cass › hydroponics › **Contribution Charts**

Programming languages used in this repository

Measured in bytes of code. Excludes generated and vendored code.

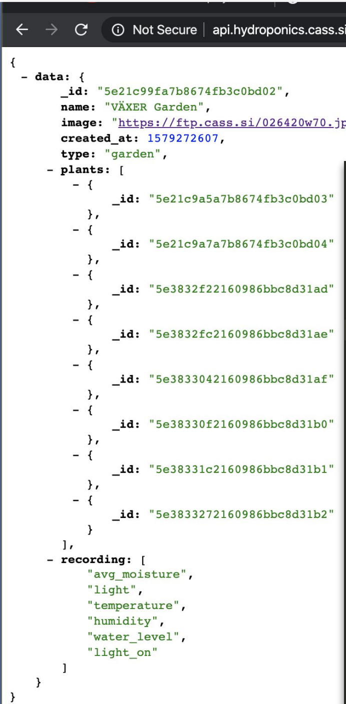
● JavaScript	36.18 %
● Python	28.35 %
● C	21.47 %
● C++	12.72 %
● CSS	0.86 %


The Cloud
MongoDB
Google Drive
Heroku
Python Flask-RESTful
GitLab
Netlify
React frontend
Me!
Router
Pi 3B
NoIR camera
ESP-32
Multiplexer
Sensors
Pi Zero W
Nano I2C slave
Sensors

API

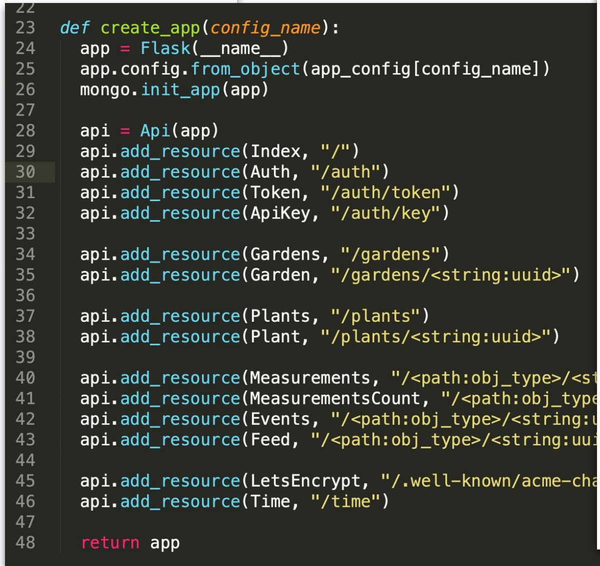
Python Flask-RESTful, extension to Flask
Application factory pattern

Fully documented and tested

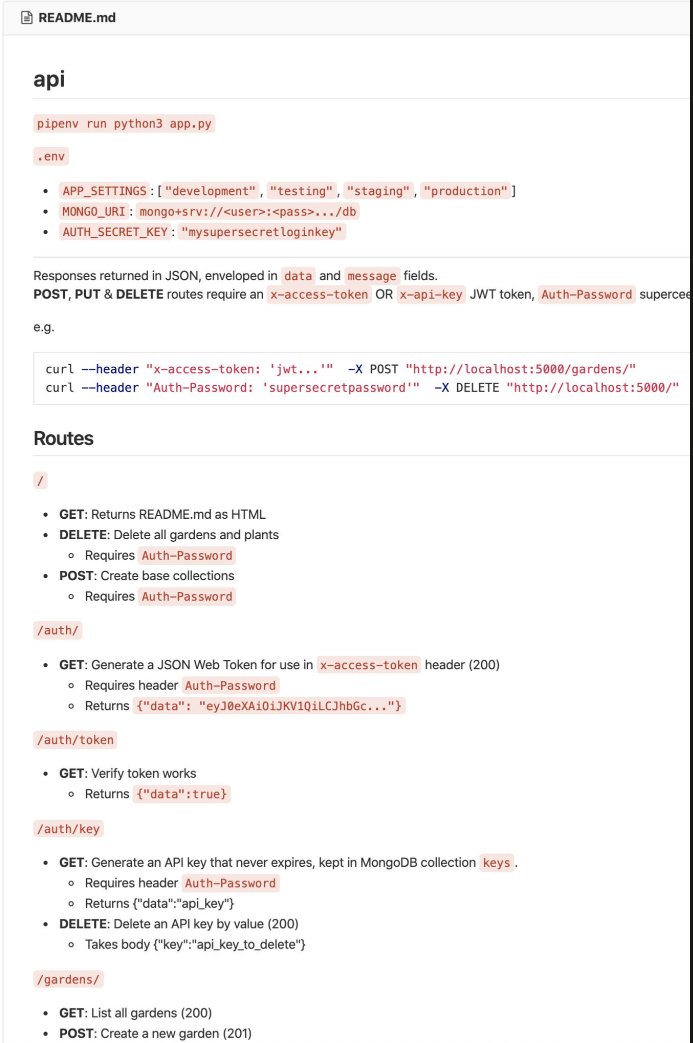


Not Secure | api.hydroponics.cass.si/gardens/5e21c99fa7b8674fb3c0bd02

```
{
  - data: {
        _id: "5e21c99fa7b8674fb3c0bd02",
        name: "VÄXER Garden",
        image: "https://ftp.cass.si/026420w70.jpeg",
        created_at: 1579272607,
        type: "garden",
      - plants: [
          - {
                _id: "5e21c9a5a7b8674fb3c0bd03"
            },
          - {
                _id: "5e21c9a7a7b8674fb3c0bd04"
            },
          - {
                _id: "5e3832f22160986bbc8d31ad"
            },
          - {
                _id: "5e3832fc2160986bbc8d31ae"
            },
          - {
                _id: "5e3833042160986bbc8d31af"
            },
          - {
                _id: "5e38330f2160986bbc8d31b0"
            },
          - {
                _id: "5e38331c2160986bbc8d31b1"
            },
          - {
                _id: "5e3833272160986bbc8d31b2"
            }
        ],
      - recording: [
            "avg_moisture",
            "light",
            "temperature",
            "humidity",
            "water_level",
            "light_on"
        ]
    }
}
```



```python
def create_app(config_name):
  app = Flask(__name__)
  app.config.from_object(app_config[config_name])
  mongo.init_app(app)

  api = Api(app)
  api.add_resource(Index, "/")
  api.add_resource(Auth, "/auth")
  api.add_resource(Token, "/auth/token")
  api.add_resource(ApiKey, "/auth/key")

  api.add_resource(Gardens, "/gardens")
  api.add_resource(Garden, "/gardens/<string:uuid>")

  api.add_resource(Plants, "/plants")
  api.add_resource(Plant, "/plants/<string:uuid>")

  api.add_resource(Measurements, "/<path:obj_type>/<st
  api.add_resource(MeasurementsCount, "/<path:obj_type
  api.add_resource(Events, "/<path:obj_type>/<string:u
  api.add_resource(Feed, "/<path:obj_type>/<string:uui

  api.add_resource(LetsEncrypt, "/.well-known/acme-cha
  api.add_resource(Time, "/time")

  return app
```



README.md

api

`pipenv run python3 app.py`

`.env`

- `APP_SETTINGS` : [`"development"`, `"testing"`, `"staging"`, `"production"`]
- `MONGO_URI` : `mongo+srv://<user>:<pass>.../db`
- `AUTH_SECRET_KEY` : `"mysupersecretloginkey"`

Responses returned in JSON, enveloped in `data` and `message` fields.
POST, **PUT** & **DELETE** routes require an `x-access-token` OR `x-api-key` JWT token, `Auth-Password` supercee

e.g.

```
curl --header "x-access-token: 'jwt...'"  -X POST "http://localhost:5000/gardens/"
curl --header "Auth-Password: 'supersecretpassword'"  -X DELETE "http://localhost:5000/"
```

Routes

`/`

- **GET**: Returns README.md as HTML
- **DELETE**: Delete all gardens and plants
  - Requires `Auth-Password`
- **POST**: Create base collections
  - Requires `Auth-Password`

`/auth/`

- **GET**: Generate a JSON Web Token for use in `x-access-token` header (200)
  - Requires header `Auth-Password`
  - Returns `{"data": "eyJ0eXAiOiJKV1QiLCJhbGc..."}`

`/auth/token`

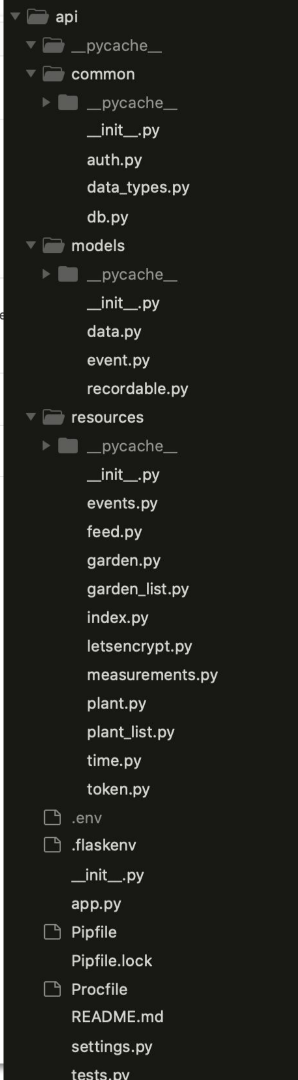
- **GET**: Verify token works
  - Returns `{"data":true}`

`/auth/key`

- **GET**: Generate an API key that never expires, kept in MongoDB collection `keys`.
  - Requires header `Auth-Password`
  - Returns {"data":"api_key"}
- **DELETE**: Delete an API key by value (200)
  - Takes body {"key":"api_key_to_delete"}

`/gardens/`

- **GET**: List all gardens (200)
- **POST**: Create a new garden (201)

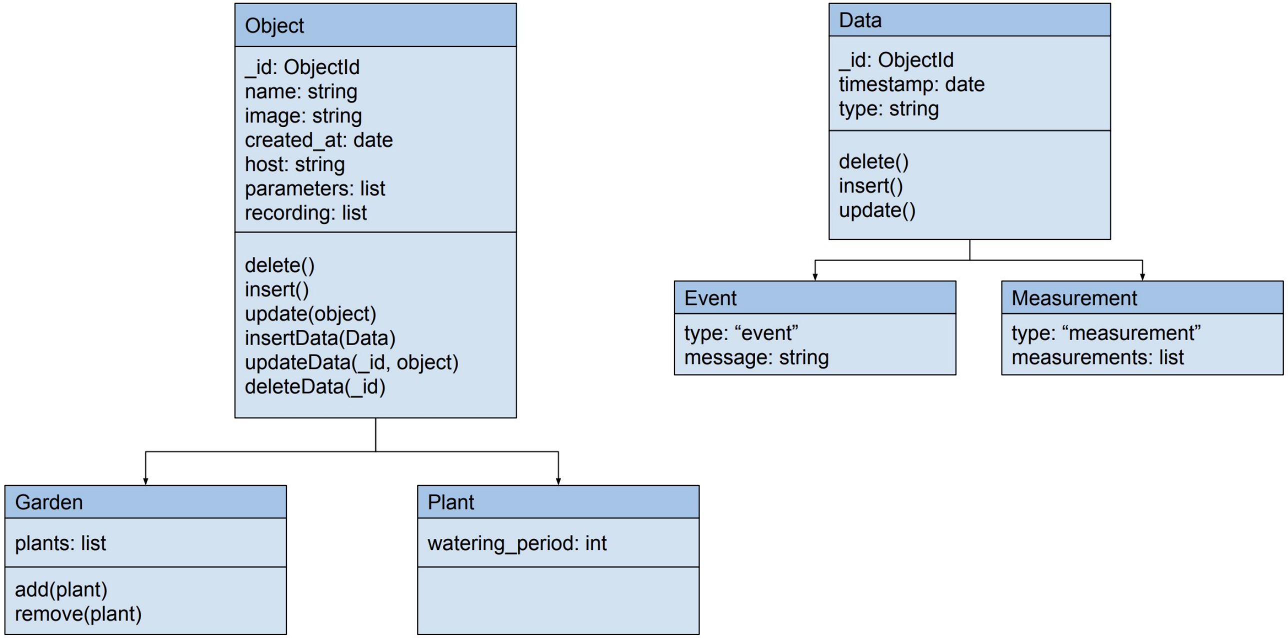


```
api
  __pycache__
  common
    __pycache__
    __init__.py
    auth.py
    data_types.py
    db.py
  models
    __pycache__
    __init__.py
    data.py
    event.py
    recordable.py
  resources
    __pycache__
    __init__.py
    events.py
    feed.py
    garden.py
    garden_list.py
    index.py
    letsencrypt.py
    measurements.py
    plant.py
    plant_list.py
    time.py
    token.py
  .env
  .flaskenv
  __init__.py
  app.py
  Pipfile
  Pipfile.lock
  Procfile
  README.md
  settings.py
  tests.py
```

Models

Plants and gardens are very similar in nature, both record data, have names, events etc., as such should have a parent class

Same for events and measurements



Object

_id: ObjectId
name: string
image: string
created_at: date
host: string
parameters: list
recording: list

delete()
insert()
update(object)
insertData(Data)
updateData(_id, object)
deleteData(_id)

Garden

plants: list

add(plant)
remove(plant)

Plant

watering_period: int

Data

_id: ObjectId
timestamp: date
type: string

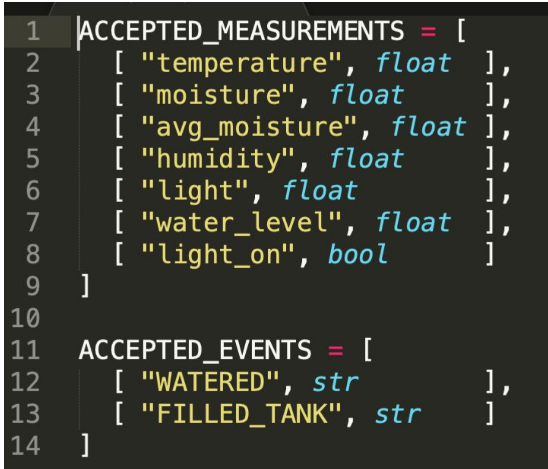
delete()
insert()
update()

Event

type: “event”
message: string

Measurement

type: “measurement”
measurements: list

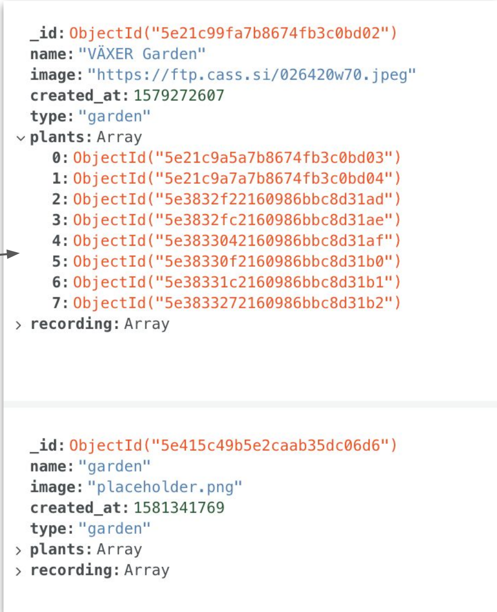


```
ACCEPTED_MEASUREMENTS = [
    [ "temperature", float  ],
    [ "moisture", float     ],
    [ "avg_moisture", float ],
    [ "humidity", float     ],
    [ "light", float        ],
    [ "water_level", float  ],
    [ "light_on", bool      ]
]

ACCEPTED_EVENTS = [
    [ "WATERED", str        ],
    [ "FILLED_TANK", str    ]
]
```

MongoDB

One-to-Many

Points to plant that exists in this garden



```
_id: ObjectId("5e21c99fa7b8674fb3c0bd02")
name: "VÄXER Garden"
image: "https://ftp.cass.si/026420w70.jpeg"
created_at: 1579272607
type: "garden"
⌄ plants: Array
    0: ObjectId("5e21c9a5a7b8674fb3c0bd03")
    1: ObjectId("5e21c9a7a7b8674fb3c0bd04")
    2: ObjectId("5e3832f22160986bbc8d31ad")
    3: ObjectId("5e3832fc2160986bbc8d31ae")
    4: ObjectId("5e3833042160986bbc8d31af")
    5: ObjectId("5e38330f2160986bbc8d31b0")
    6: ObjectId("5e38331c2160986bbc8d31b1")
    7: ObjectId("5e3833272160986bbc8d31b2")
› recording: Array

_id: ObjectId("5e415c49b5e2caab35dc06d6")
name: "garden"
image: "placeholder.png"
created_at: 1581341769
type: "garden"
› plants: Array
› recording: Array
```

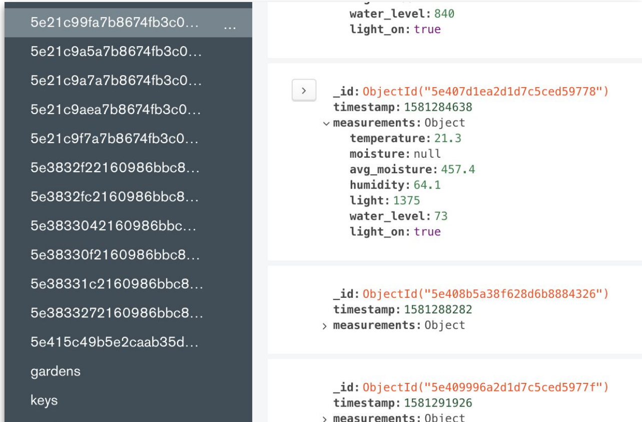
```
subplants = db[“plants”].find({
  _id: {$in : garden.plants}
}).toArray() ;
```

```
garden._id = “ABCDEF”
// return all measurements for garden
db[“ABCDEF”].find({type:”measurement”})

// return all events for garden
db[“ABCDEF”].find({type:”event”})
```

One-to-Squillions

1 update/hr = 180 updates/week = 8765 updates/year **for one plant**

Move “measurements” array into own collection because MongoDB access times on very large arrays in documents suffers (+ max 16MB doc size)

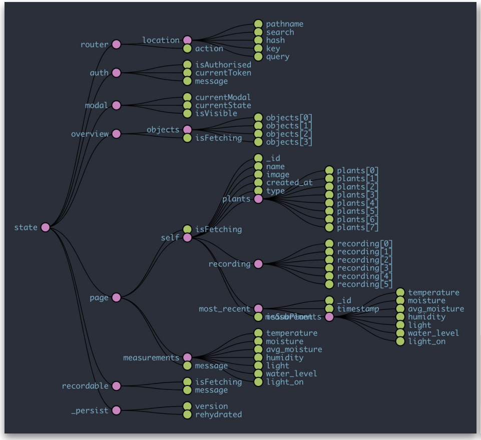


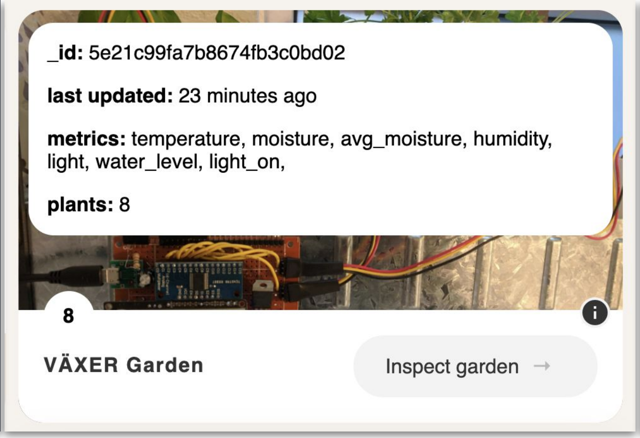
```
5e21c99fa7b8674fb3c0...  ...
5e21c9a5a7b8674fb3c0...
5e21c9a7a7b8674fb3c0...
5e21c9aea7b8674fb3c0...
5e21c9f7a7b8674fb3c0...
5e3832f22160986bbc8...
5e3832fc2160986bbc8...
5e3833042160986bbc...
5e38330f2160986bbc8...
5e38331c2160986bbc8...
5e3833272160986bbc8...
5e415c49b5e2caab35d...
gardens
keys
```

```
    water_level: 840
    light_on: true

 _id: ObjectId("5e407d1ea2d1d7c5ced59778")
 timestamp: 1581284638
⌄ measurements: Object
    temperature: 21.3
    moisture: null
    avg_moisture: 457.4
    humidity: 64.1
    light: 1375
    water_level: 73
    light_on: true

 _id: ObjectId("5e408b5a38f628d6b8884326")
 timestamp: 1581288282
› measurements: Object

 _id: ObjectId("5e409996a2d1d7c5ced5977f")
 timestamp: 1581291926
› measurements: Object
```

Frontend

React, Redux, redux-persist, styled-components, jwt, redux-thunk, connected-react-router, chart.js, toasted notifications ….



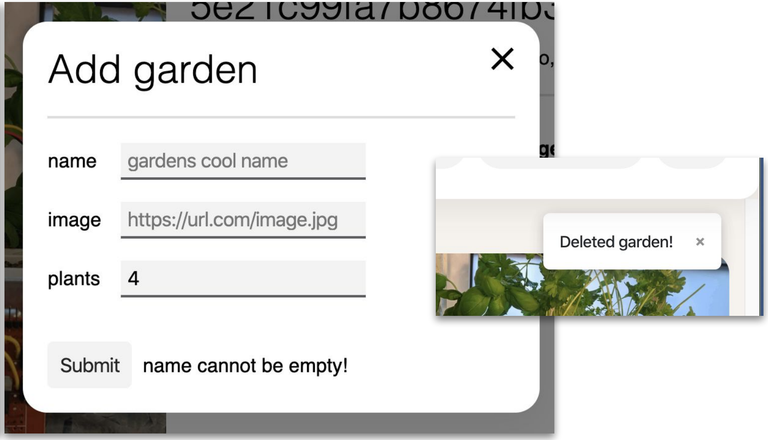


_id: 5e21c99fa7b8674fb3c0bd02

last updated: 23 minutes ago

metrics: temperature, moisture, avg_moisture, humidity, light, water_level, light_on,

plants: 8



Add garden ✕

name gardens cool name

image https://url.com/image.jpg

plants 4

Submit name cannot be empty!

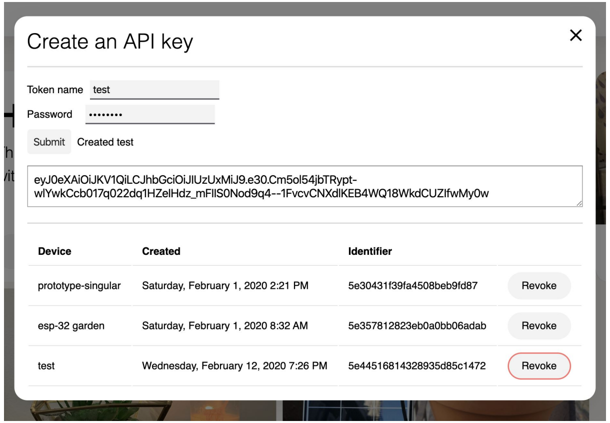
Deleted garden! ×

Modals, implemented own modal system with HOCs as opposed to baking in a premade library as a learning exercise - No one way to achieve it.

Netlify *rewrites* requests to `/api/*` in production to Heroku URL using a netlify.toml file

Frontend

- Manage API keys
- Manage plants & gardens
    - Target metrics
    - Events
- Trigger actions on microcontrollers
- View live stats



Create an API key

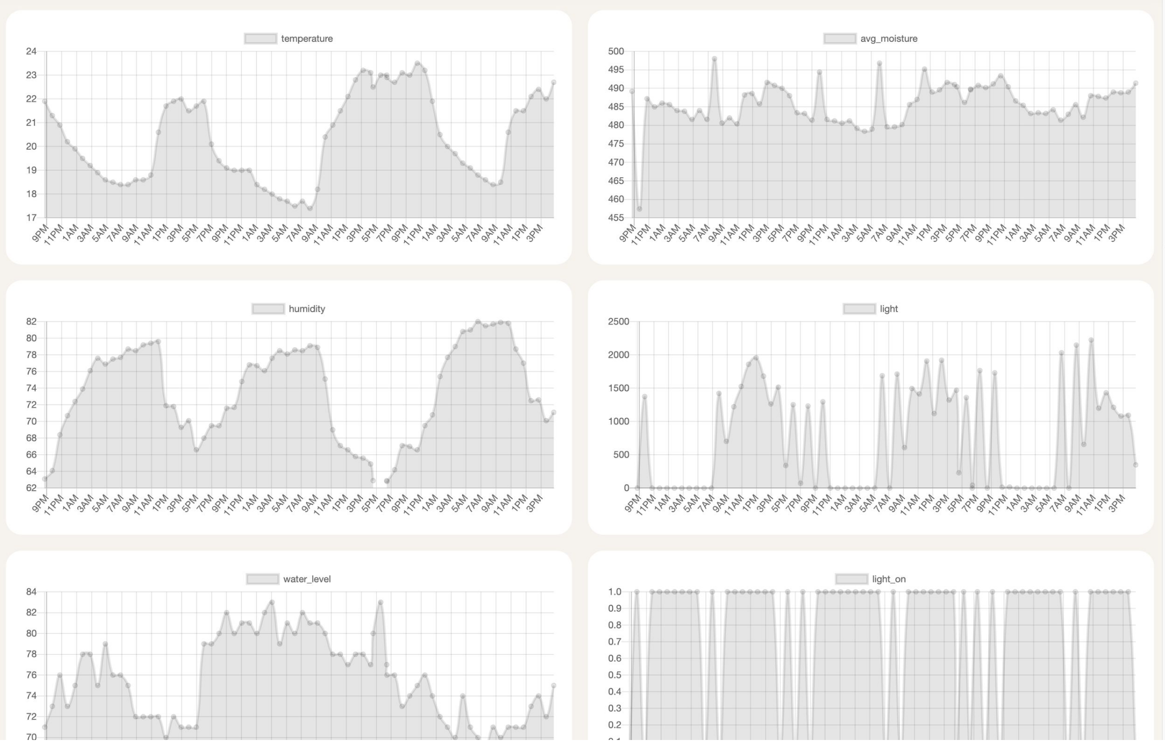
Token name test

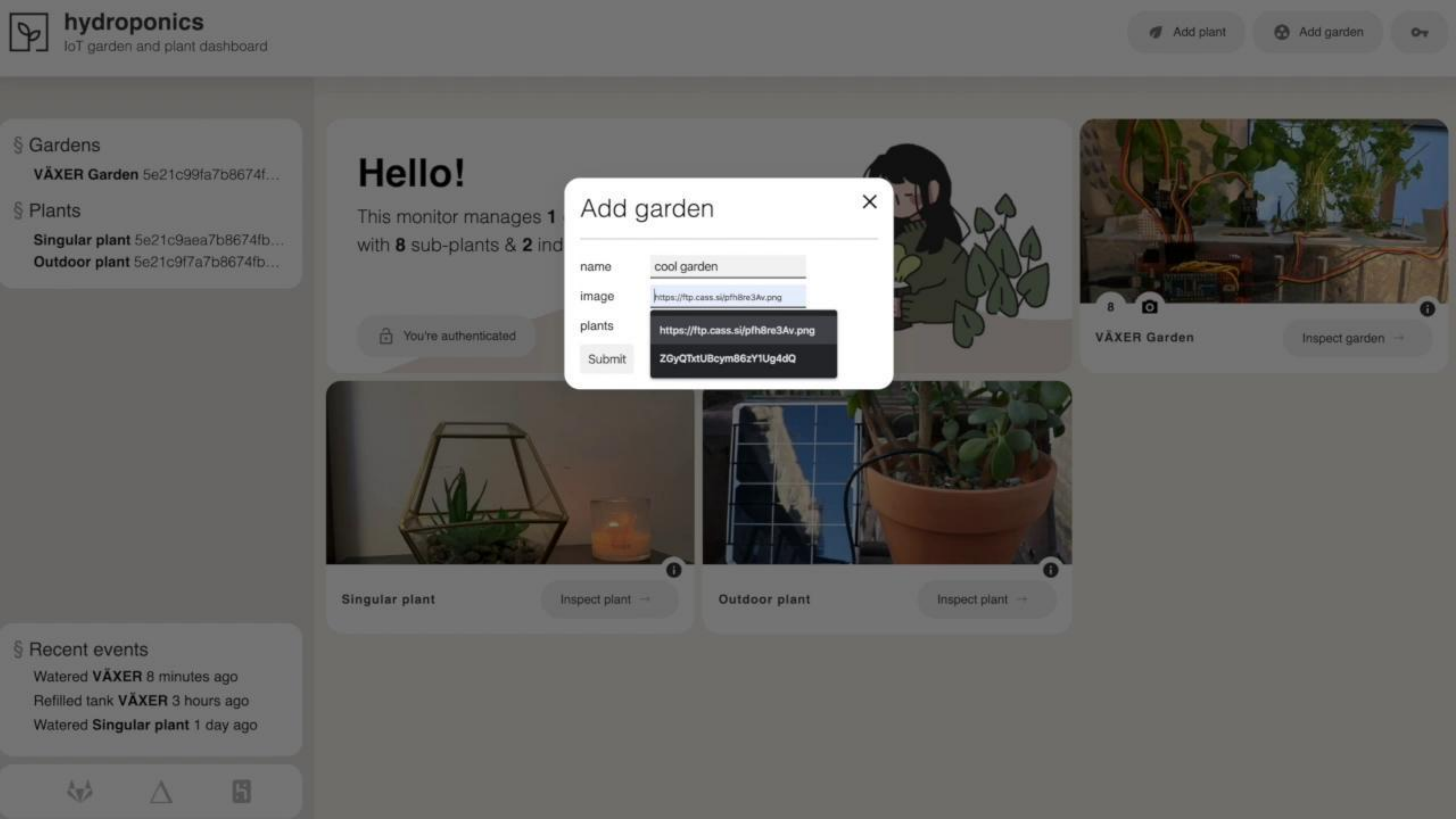
Password ••••••••

Created test

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.e30.Cm5ol54jbTRypt-
wlYwkCcb017q022dq1HZelHdz_mFIlS0Nod9q4--1FvcvCNXdlKEB4WQ18WkdCUZIfwMy0w
```

Device	Created	Identifier	
prototype-singular	Saturday, February 1, 2020 2:21 PM	5e30431f39fa4508beb9fd87	Revoke
esp-32 garden	Saturday, February 1, 2020 8:32 AM	5e357812823eb0a0bb06adab	Revoke
test	Wednesday, February 12, 2020 7:26 PM	5e44516814328935d85c1472	Revoke





hydroponics

IoT garden and plant dashboard

§ Gardens

VÄXER Garden 5e21c99fa7b8674f...

§ Plants

Singular plant 5e21c9aea7b8674fb...
Outdoor plant 5e21c9f7a7b8674fb...

Hello!

This monitor manages **1** ... with **8** sub-plants & **2** ind...

You're authenticated

Add garden

name: cool garden

image: https://ftp.cass.si/pfh8re3Av.png

plants

https://ftp.cass.si/pfh8re3Av.png
ZGyQTxtUBcym86zY1Ug4dQ

§ Recent events

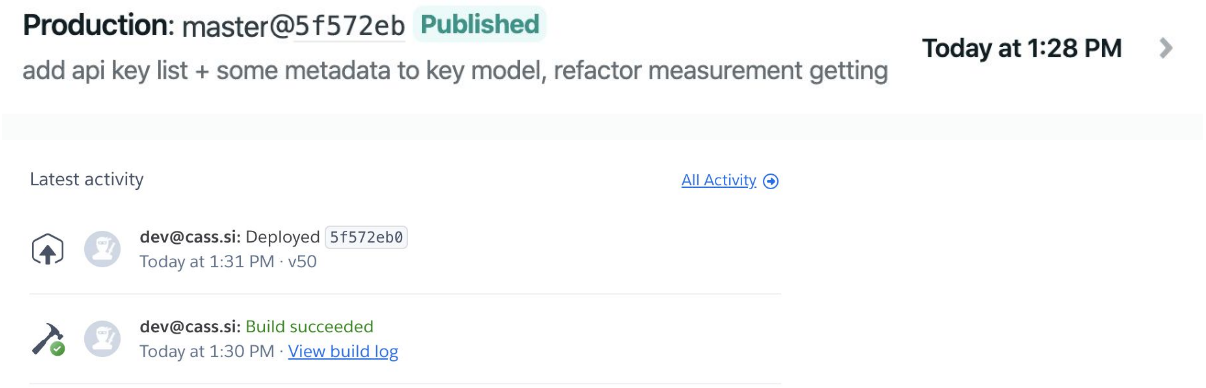
Watered **VÄXER** 8 minutes ago
Refilled tank **VÄXER** 3 hours ago
Watered **Singular plant** 1 day ago

CI/CD pipeline

Commit → GitLab Task Runner started using .gitlab-ci.yml

Testing stage → Run Newman tests on API

Production → Deploy API on Heroku and FE on Netlify



Production: master@5f572eb **Published**

add api key list + some metadata to key model, refactor measurement getting

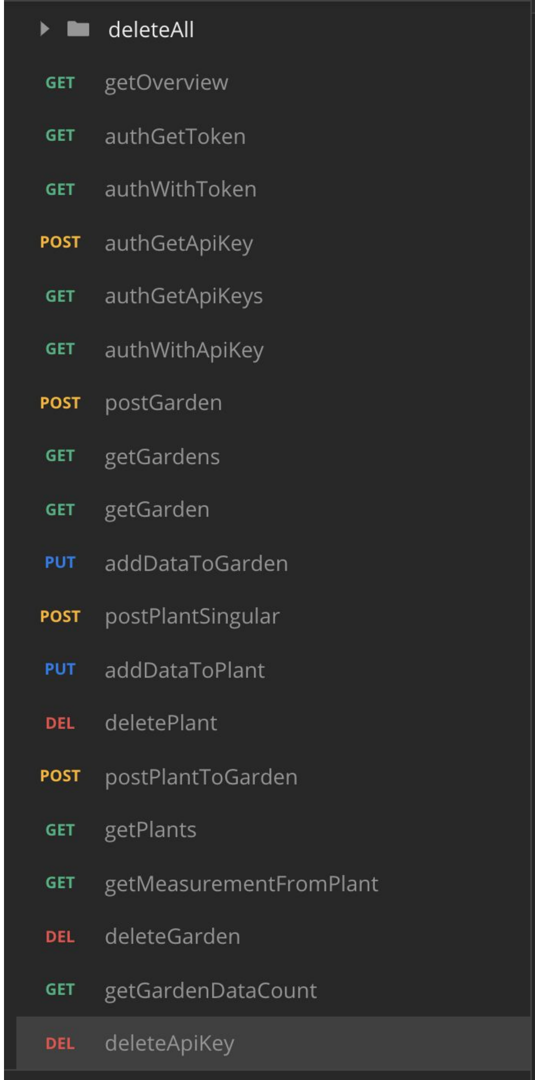
Today at 1:28 PM

Latest activity

All Activity

dev@cass.si: Deployed 5f572eb0
Today at 1:31 PM · v50

dev@cass.si: Build succeeded
Today at 1:30 PM · View build log

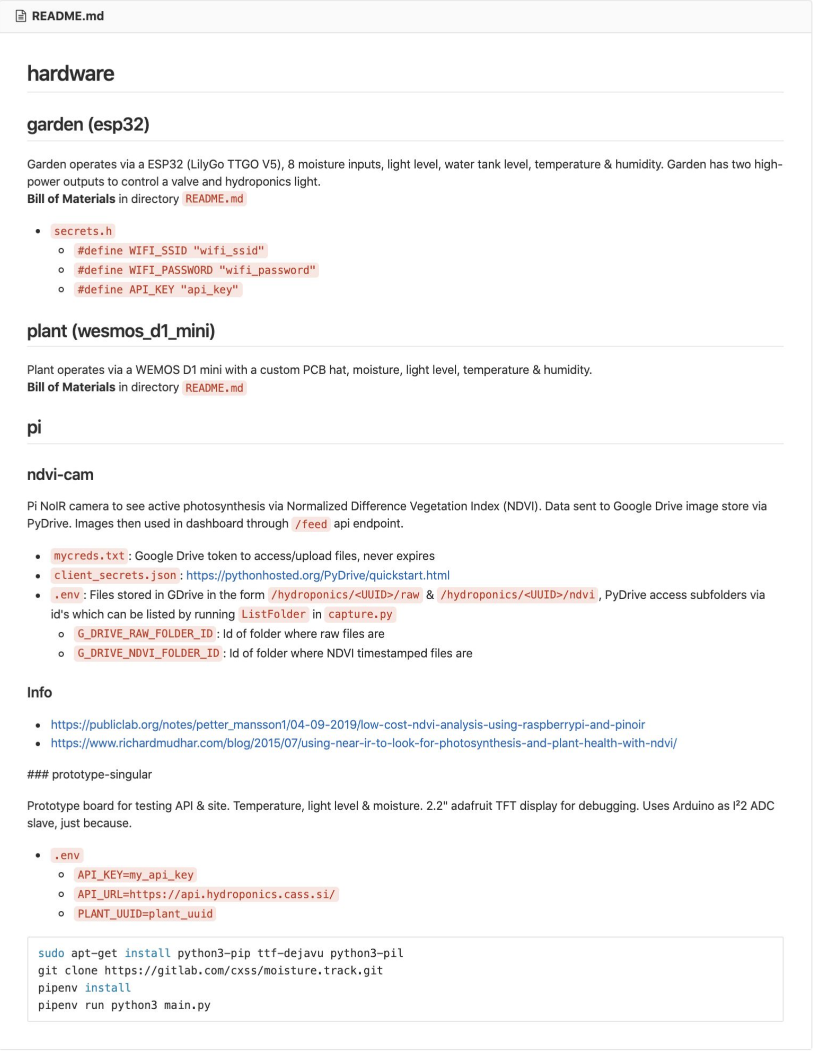


deleteAll

GET getOverview
GET authGetToken
GET authWithToken
POST authGetApiKey
GET authGetApiKeys
GET authWithApiKey
POST postGarden
GET getGardens
GET getGarden
PUT addDataToGarden
POST postPlantSingular
PUT addDataToPlant
DEL deletePlant
POST postPlantToGarden
GET getPlants
GET getMeasurementFromPlant
DEL deleteGarden
GET getGardenDataCount
DEL deleteApiKey

Hardware

- **ESP32** - Testing gardens composed of several plants
- **Raspberry Pi**
  - NDVI-Camera: Use NoIR camera to capture active photosynthesis
  - prototype-singular: First hardware prototype to interact with API
- **Wemos D1 Mini** - Designed and manufactured a circuit board to monitor individual plants (to eventually sell...)



README.md

hardware

garden (esp32)

Garden operates via a ESP32 (LilyGo TTGO V5), 8 moisture inputs, light level, water tank level, temperature & humidity. Garden has two high-power outputs to control a valve and hydroponics light.
Bill of Materials in directory `README.md`

- `secrets.h`
  - `#define WIFI_SSID "wifi_ssid"`
  - `#define WIFI_PASSWORD "wifi_password"`
  - `#define API_KEY "api_key"`

plant (wesmos_d1_mini)

Plant operates via a WEMOS D1 mini with a custom PCB hat, moisture, light level, temperature & humidity.
Bill of Materials in directory `README.md`

pi

ndvi-cam

Pi NoIR camera to see active photosynthesis via Normalized Difference Vegetation Index (NDVI). Data sent to Google Drive image store via PyDrive. Images then used in dashboard through `/feed` api endpoint.

- `mycreds.txt` : Google Drive token to access/upload files, never expires
- `client_secrets.json` : https://pythonhosted.org/PyDrive/quickstart.html
- `.env` : Files stored in GDrive in the form `/hydroponics/<UUID>/raw` & `/hydroponics/<UUID>/ndvi` , PyDrive access subfolders via id's which can be listed by running `ListFolder` in `capture.py`
  - `G_DRIVE_RAW_FOLDER_ID` : Id of folder where raw files are
  - `G_DRIVE_NDVI_FOLDER_ID` : Id of folder where NDVI timestamped files are

Info

- https://publiclab.org/notes/petter_mansson1/04-09-2019/low-cost-ndvi-analysis-using-raspberrypi-and-pinoir
- https://www.richardmudhar.com/blog/2015/07/using-near-ir-to-look-for-photosynthesis-and-plant-health-with-ndvi/

prototype-singular

Prototype board for testing API & site. Temperature, light level & moisture. 2.2" adafruit TFT display for debugging. Uses Arduino as I²2 ADC slave, just because.

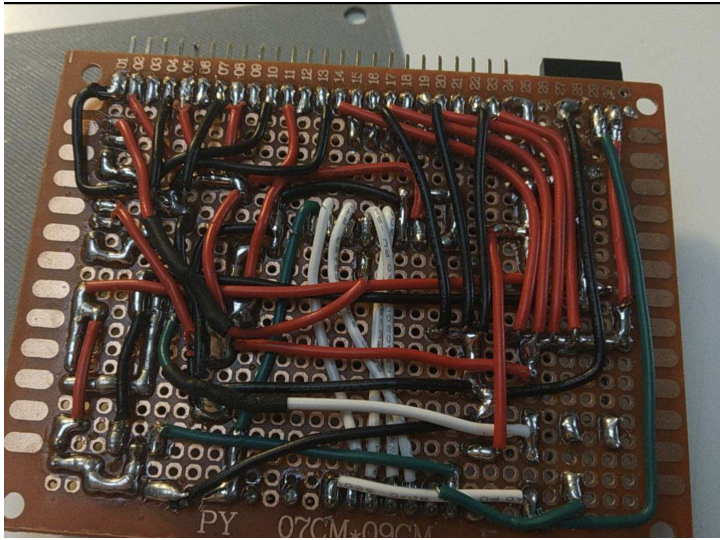
- `.env`
  - `API_KEY=my_api_key`
  - `API_URL=https://api.hydroponics.cass.si/`
  - `PLANT_UUID=plant_uuid`

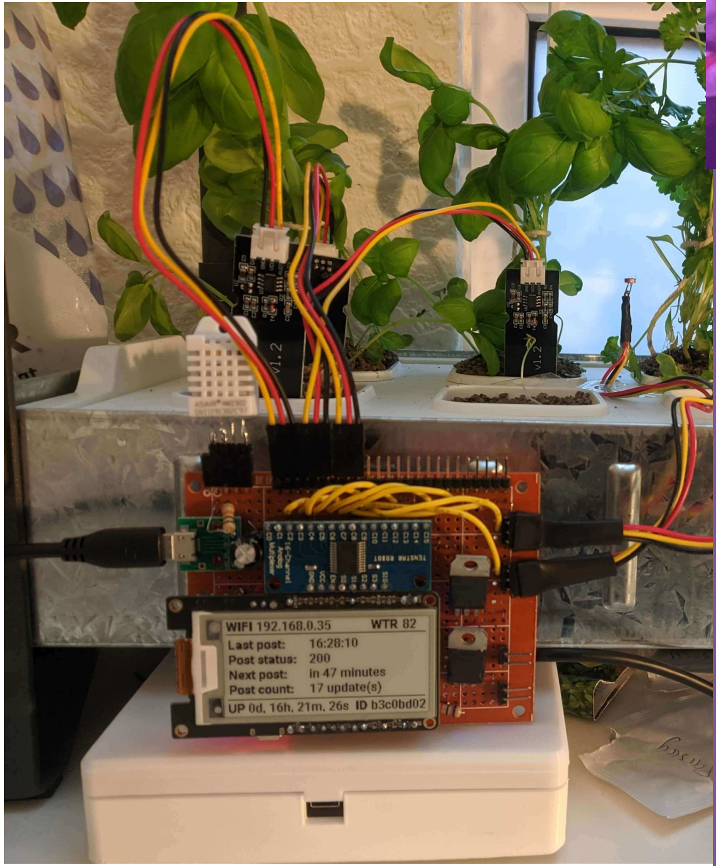
```
sudo apt-get install python3-pip ttf-dejavu python3-pil
git clone https://gitlab.com/cxss/moisture.track.git
pipenv install
pipenv run python3 main.py
```

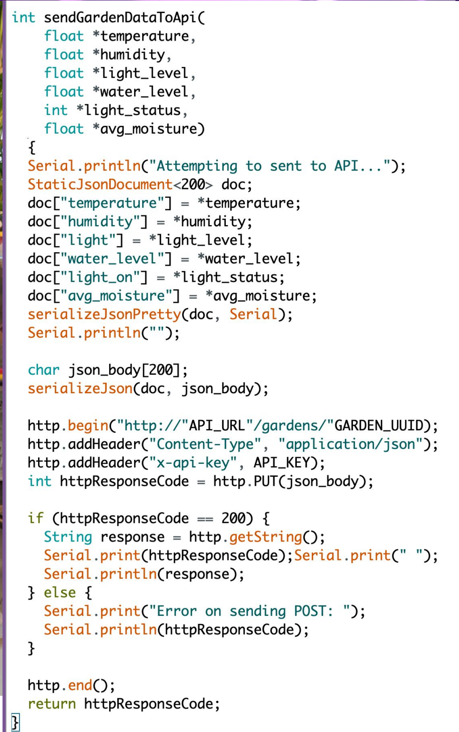
ESP32

Written in C++, queries Python API

16 channel multiplexer supports up to 8 plants alongside temperature, humidity light level, water tank level & grow light switch







```
int sendGardenDataToApi(
    float *temperature,
    float *humidity,
    float *light_level,
    float *water_level,
    int *light_status,
    float *avg_moisture)
  {
  Serial.println("Attempting to sent to API...");
  StaticJsonDocument<200> doc;
  doc["temperature"] = *temperature;
  doc["humidity"] = *humidity;
  doc["light"] = *light_level;
  doc["water_level"] = *water_level;
  doc["light_on"] = *light_status;
  doc["avg_moisture"] = *avg_moisture;
  serializeJsonPretty(doc, Serial);
  Serial.println("");

  char json_body[200];
  serializeJson(doc, json_body);

  http.begin("http://"API_URL"/gardens/"GARDEN_UUID);
  http.addHeader("Content-Type", "application/json");
  http.addHeader("x-api-key", API_KEY);
  int httpResponseCode = http.PUT(json_body);

  if (httpResponseCode == 200) {
    String response = http.getString();
    Serial.print(httpResponseCode);Serial.print(" ");
    Serial.println(response);
  } else {
    Serial.print("Error on sending POST: ");
    Serial.println(httpResponseCode);
  }

  http.end();
  return httpResponseCode;
}
```

ESP32

Expose micro-controller to API by port-forwarding through router,

Call routes on the microcontroller (IP not exposed):

GET https://api.hydroponics.cass.si/plants/<uuid>/events/WATER_PLANT

calls ↓

GET http://83.9.93.103:8080/events/WATER_PLANT

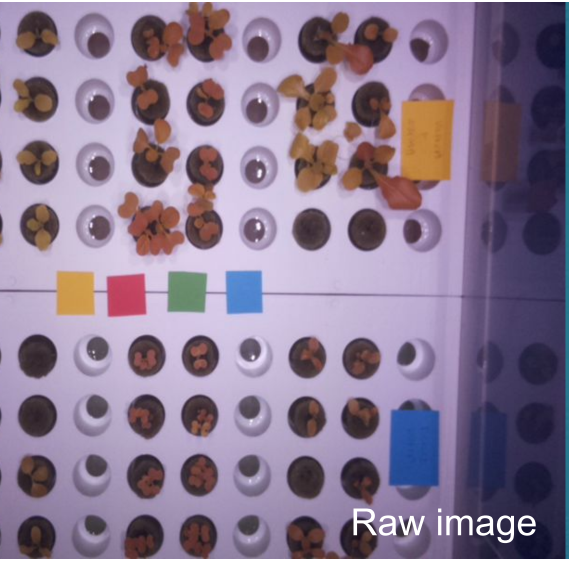
runs ↓

```
void waterPlant(int current_level) { … } //open water valve for n seconds
```

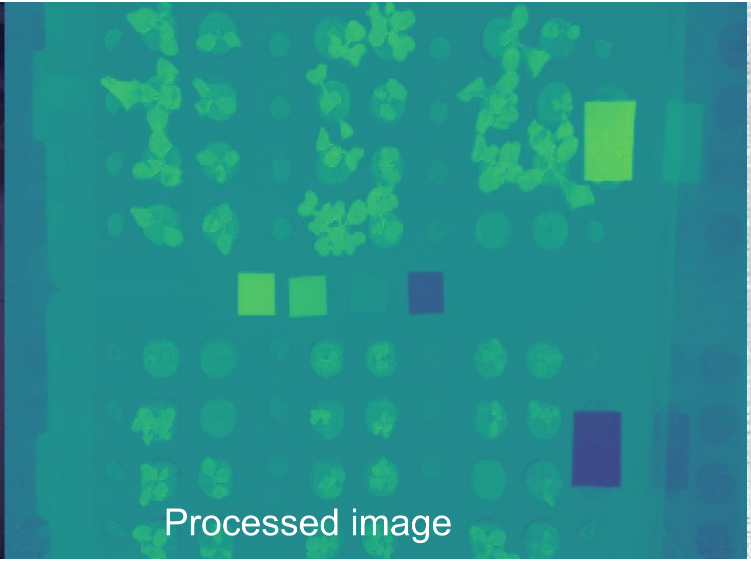
Normalized Difference Vegetation Index (NDVI) Cam

No Infrared Camera + Python + Matlab + PyDrive

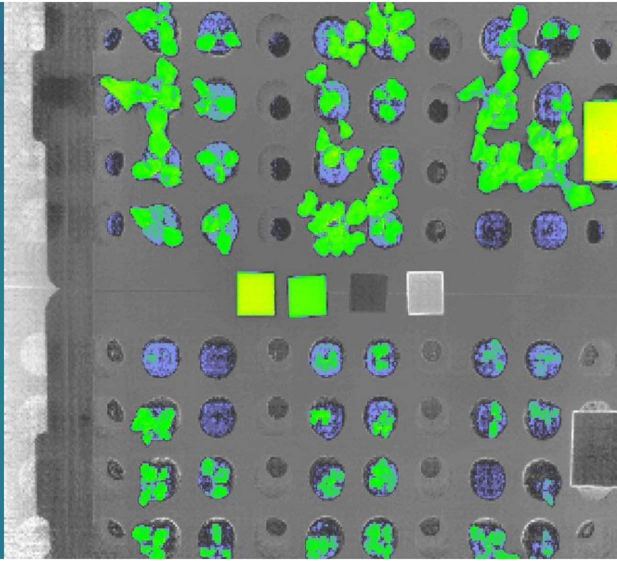
Uses Google Drive as a image store to be later served by API for a live feed



Raw image



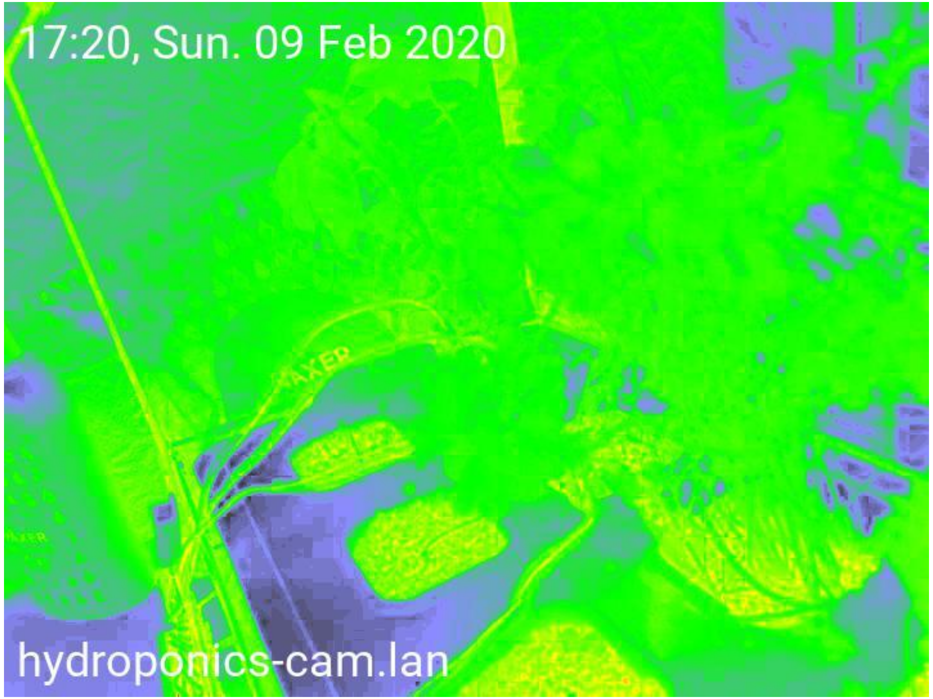
Processed image



Custom colour map

https://publiclab.org/notes/petter_mansson1/04-09-2019/low-cost-ndvi-analysis-using-raspberrypi-and-pinoir

NDVI



Needs a blue light filter...

Raspberry Pi

First prototype to send real data to the API, tracked:

- Light level
- Moisture
- Temperature

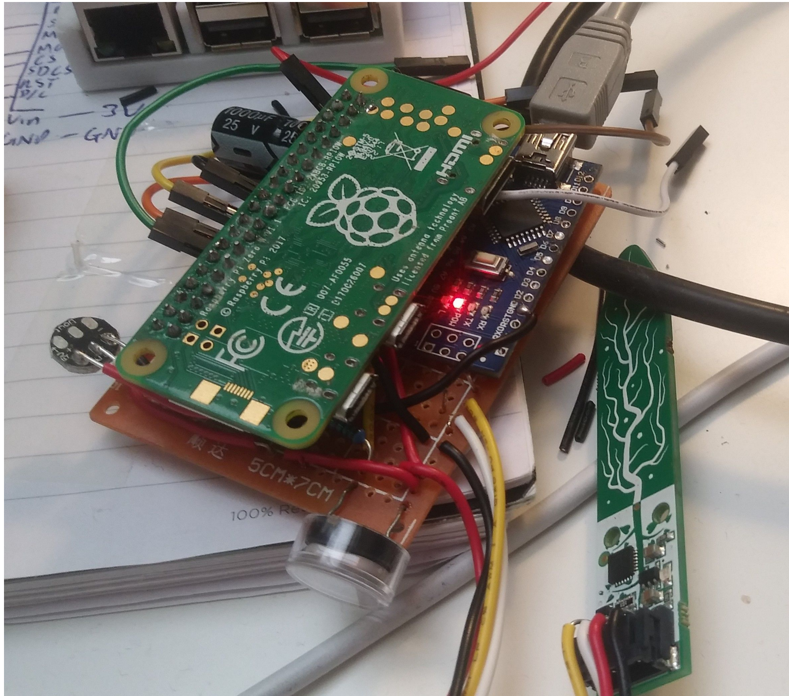
Realised pretty quickly I'd need non-expiring

API authentication keys… added that

Learned how to use an Arduino Nano as

an I2C slave to communicate data between

Pi and Nano

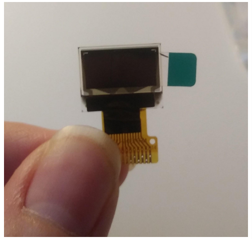


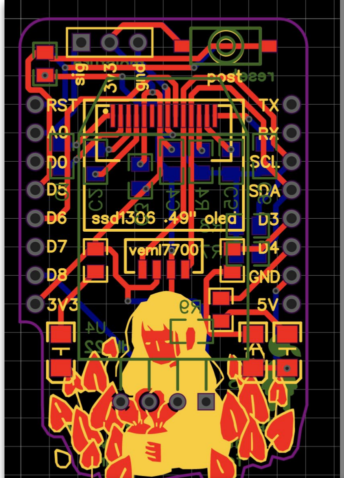
Wemos D1 Mini

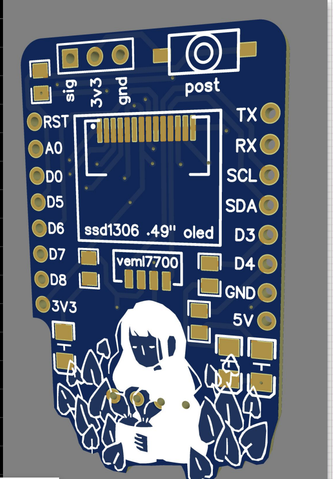
MicroPython, 0.49” display (tiny!!)

Designed circuit and PCB layout on EasyEDA

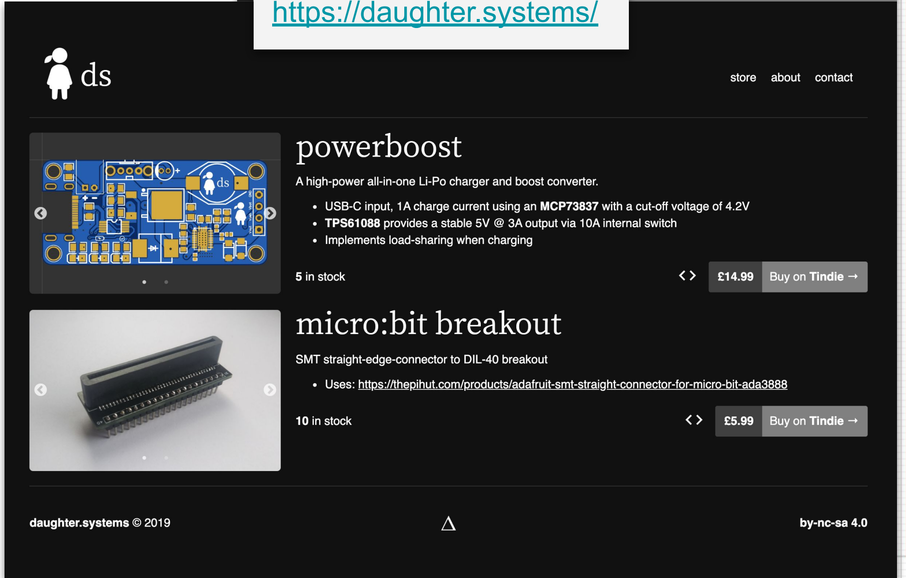
End goal is to sell on my website







https://daughter.systems/



ds

store about contact

powerboost

A high-power all-in-one Li-Po charger and boost converter.

- USB-C input, 1A charge current using an **MCP73837** with a cut-off voltage of 4.2V
- **TPS61088** provides a stable 5V @ 3A output via 10A internal switch
- Implements load-sharing when charging

5 in stock

< > £14.99 Buy on Tindie →

micro:bit breakout

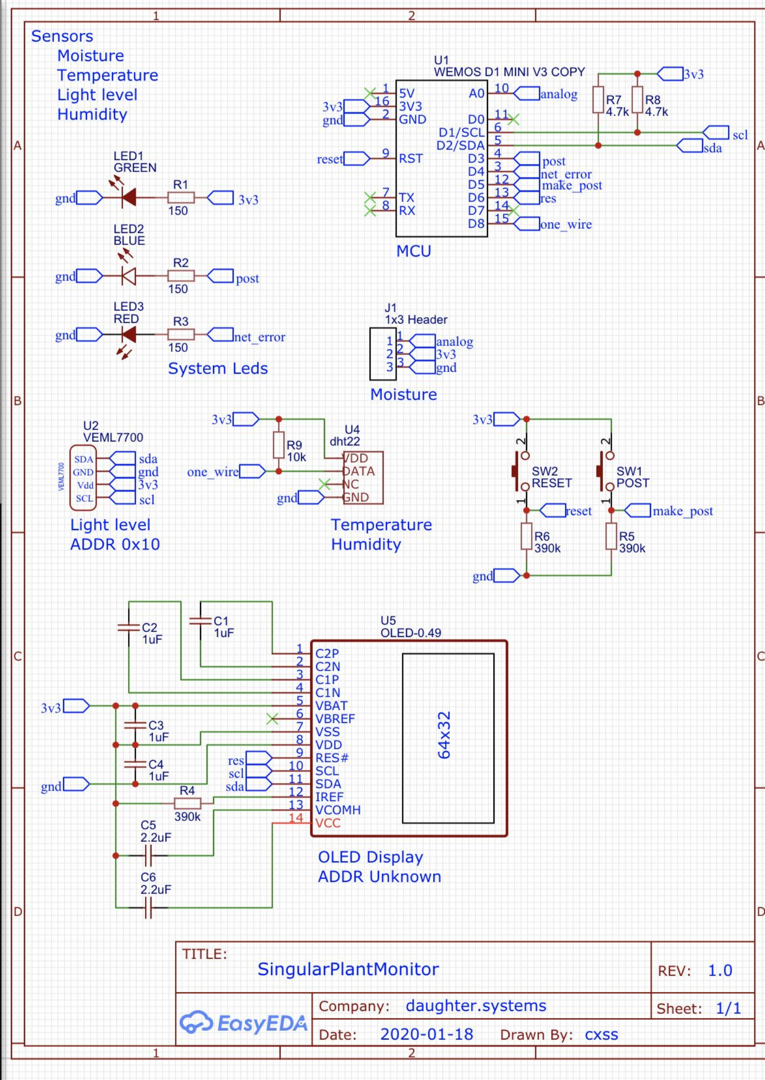
SMT straight-edge-connector to DIL-40 breakout

- Uses: https://thepihut.com/products/adafruit-smt-straight-connector-for-micro-bit-ada3888

10 in stock

< > £5.99 Buy on Tindie →

daughter.systems © 2019 △ by-nc-sa 4.0



The future

Extending usability of system through Google Home/Alexa.

“Hey Google, water the plants”

PCB to manage gardens of N plants

Identifying plants through Pl@ntNet API (https://my-api.plantnet.org/)

...Social network for tracking plants and gardens?

https://gitlab.com/cxss/hydroponics